Project JackIO

Technische Documentatie

Lars Werkman & Niels Hendrikx
1/20/2016

Inhoud

Inleiding

Dit document is een technische samenvatting van project JackIO.
De technische componenten worden toegelicht.
Het document is opgedeeld in vier stukken;

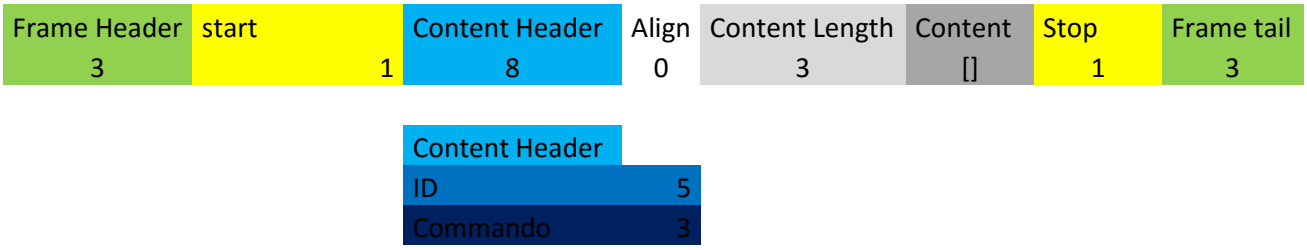
- Het communicatie protocol.
- Software, bestaande uit de Android software, en software dat op de IC draait. We lichten in het document toe hoe het globaal werkt. Voor de specifieke code verwijzen we u door naar de source code.
- Hardware, welke hardware zit er in JackIO, en hoe werkt het.

JackIO is een klein PCB board met een microprocessor. Er zitten analoge & digitale pinnen op.
De pinnen kunnen worden uitgelezen/aangestuurd op een Android telefoon met de JackIO App & API.
De communicatie vindt plaats via de audio jack.
De microprocessor wordt voorzien van stroom door een batterij of telefoon, dit is willekeurig aangeven d.m.v. een hardware switch.

Project JackIO is gemaakt als BU-Project door Lars Werkman & Niels Hendrikx, studenten technische informatica aan de hogeschool van Amsterdam.

Communicatie Protocol

Het JackIO pakket:



Frame Header: We hebben voor een frame header gekozen om het pakket makkelijk te detecteren. Deze wordt gevuld met 3 bits: 111.

Start Bit: De Start bit : 1.

Content Header: Bestaat uit 8 bits.
De Content Header hebben we later eigenlijk opgedeelt in 2 delen, het ID en het commando.
Het bestaat uit een ID waar het Pin nummer in zit die je wilt aanspreken op de microprocessor.
Het Commando bestaat uit: “read of write”.

Align: Deze is niet toebedeelt en staat altijd op 0.

Content Length: 3 bits, deze geeft aan hoeveel bytes er in de content zitten. Bij de waarde van 0 zal dit 1 zijn. De maximale waarde = 111=7. Bij deze waarde zullen er 8 bytes in de content zitten.

Content: de content lengte is variabel. Deze wordt gedefinieerd in de “content lenght”.
In de content zit de content van het commando, bijvoorbeeld bij write pin 5: “low”of “high”.
Bij een schrijven van een waarde kan dit natuurlijk ook bijvoorbeeld 200 = 11001000.

Stop: Vervolgens een stop bit om het einde van de content aan te duiden. Dit is een 1.

Frametail: De “staart van het pakket” bestaat uit 3 bits: 111. Dit is het einde van het pakket.

Analoge pakket:

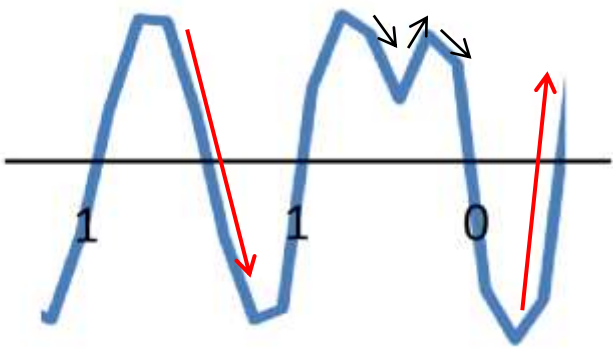
Hieronder een visuele weergave van het pakket als het uit de encoder komt.
Elke bit wordt ge-encodedeert naar een transitie van 0 naar 1(bit is 1), of 1 naar 0(bit is 0).
Deze transitie wordt vertaald naar een geluidsignaal zoals hieronder.

Dit signaal wordt verstuurd over de audio lijn, en kan later weer ge-decodeert worden naar een digitaal binair pakket.
Wij hebben ons eigen algoritme gemaakt om dit analoge pakket te decoderen.

Globaal Decoderen:

Het probleem is dat je zonder klok signaal niet weet wanneer je moet lezen of er een 1 of 0 op de lijn zit.
De oplossing is manchester Encoding, hier hebben wij onze eigen draai aan gegeven.
Een transitie van laag naar hoog stelt een 1 voor, een transitie van hoog naar laag stelt een 0 voor.
Bij het binnen komen van het signaal meten wij de transitie (dus een 0 of 1).
Deze is alleen geldig als het geen “ghost” is.
Een ghost is een transitie na een valide transitie die niet als bit mag worden geïnterpreteerd.(weergegeven met een rode pijl).
Dus na een valide transitie is *ghost = true*, zolang deze *true* is worden er geen bits geregistreerd.

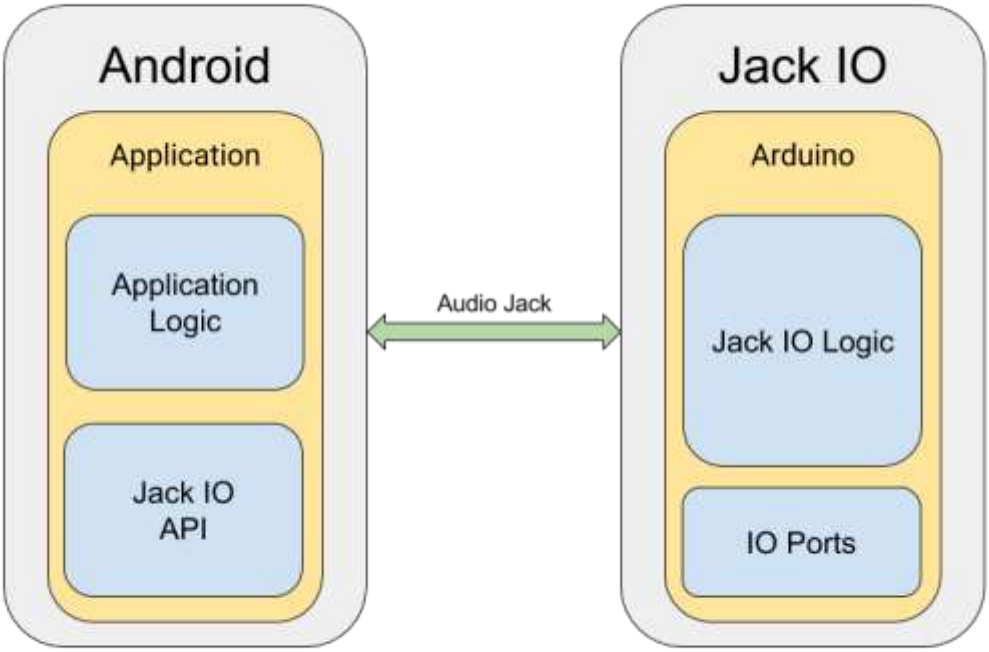
De ghost wordt weer gereset als er een (niet valide) transitie plaatsvindt (middenlijn wordt gepasseerd), of als er 3x een vector change plaats vindt zonder dat de middenlijn wordt gepasseerd.
Een vector change is een richting verplaatsing van het hellingsgetal (aangegeven met de kleine zwarte pijlen).
Na het resetten wordt volgende transitie weer interpreteert als bit, gaat dit door tot het einde van het pakket.



Software

Globaal overzicht

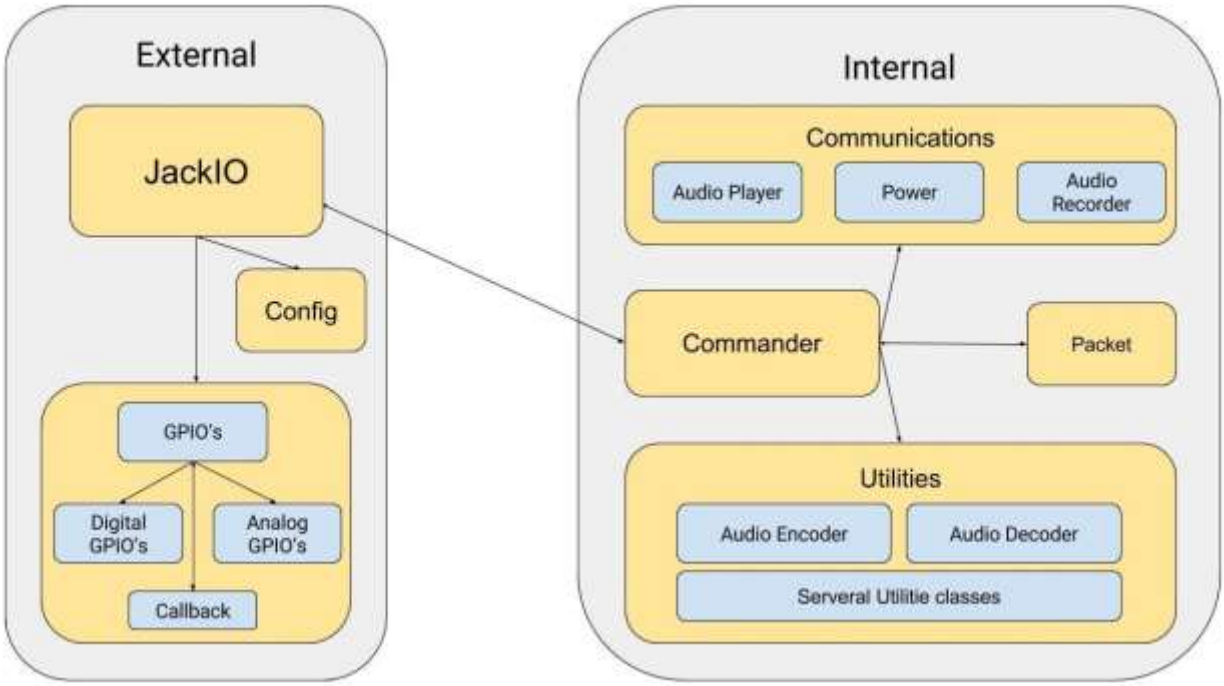
Figuur 1 geeft een globaal overzicht van het concept van JackIO.
Minimalistisch zijn het 2 CPU's die met elkaar communiceren via een audio jack.
Ze encoden/decoden data packets in beide richtingen voordat ze de data kunnen lezen of versturen.



Software telefoon

JackIO Api

De API is een Android specifieke library die alle connectie tussen het Android apparaat en het JackIO board. Figuur 2 geeft een globale overzicht van de JackIO Android API.



External

Het externe deel wordt alleen gebruikt om te communiceren met de applicatie, JackIO is een singleton instance die op ieder moment gekoppeld can worden aan de Context van een applicatie.

De JackIO instantie kan worden geconfigureerd voor verschillende board architecturen. De ATMEGA-8_L is huidig de enigste architectuur die wordt ondersteund. Deze configuraties vertelt de API hoe hoog de sample rate ligt en hoeveel analoge en digitale pinnen er gebruikt kunnen worden.

Instanties van GPIO's poorten kunnen worden aangevraagd van de JackIO instantie. Op deze poorten kunnen operaties worden gedaan, afhankelijk van Analoge of Digitale pin.

Resultaten worden asynchrone teruggevoerd aan de applicatie via een interface die als Callback werkt.

Een minimalistisch voorbeeld van het gebruik van de API in en Android Activity:

```
public class MainActivity extends Activity {

    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Button send = (Button) findViewById(R.id.send);

        //Verkrijg de JackIO instantie
        final JackIO jack = JackIO.get(this);

        send.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                //Zet de Digitale pin 1 HIGH
                jack.getDigitalGPIO(1).setHigh();
            }
        });
    }

    @Override
    protected void onResume() {
        super.onResume();
        JackIO.start(this);
    }

    @Override
    protected void onPause() {
        super.onPause();
        JackIO.stop();
    }
}
```

Internal

Zodra de JackIO instantie aan een Context wordt gekoppeld wordt er internal de Commander opgestart. Hierbij worden door de Commander ook gelijk de Power opgestart en de AudioRecorder.

De Power stuurt een constante sinus op een hoge frequentie uit. Deze frequentie bedraagt 5.5125 kilohertz. Deze sinus wordt gebruikt om stroom te voorzien aan de JackIO board.

De AudioRecorder leest constant over de audio jack de microfoon lijn uit. en probeert een start van een packet te vinden. om deze vervolgens verder te decoderen met behulp van de AudioDecoder.

Intern worden alle commandos naar de Commander gestuurd, deze commandos worden daar vertaald naar een Packet klassen. deze klassen werkt volgens ons gedefinieerde pakket protocol.

En vervolgens word er met dit pakket door de AudioEncoder klassen een audio sample gegenereerd volgens onze manchester encoding.

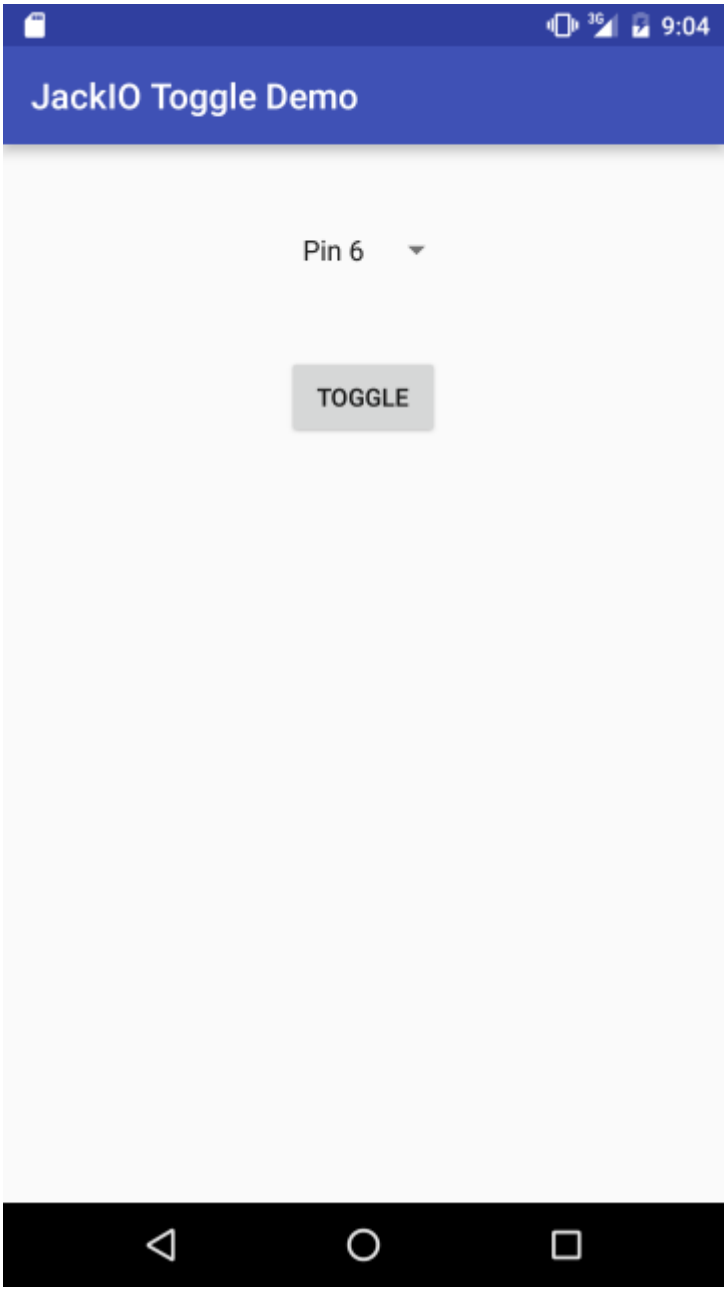
Dit gegeneerde audio samples wordt vervolgens door de Commander door gestuurd naar de Audio Player die stopt deze in een queue. En vervolgens dat wanneer de AudioPlayer er klaar voor is speelt hij deze audio sample over de linker stereo van de audio jack.

De AudioPlayer, AudioJack en Power werken allemaal op hun eigen Thread, op Interrupt basis om zo min mogelijk actief te zijn en nooit memory leaks mogen veroorzaken.

JackIO Applicatie

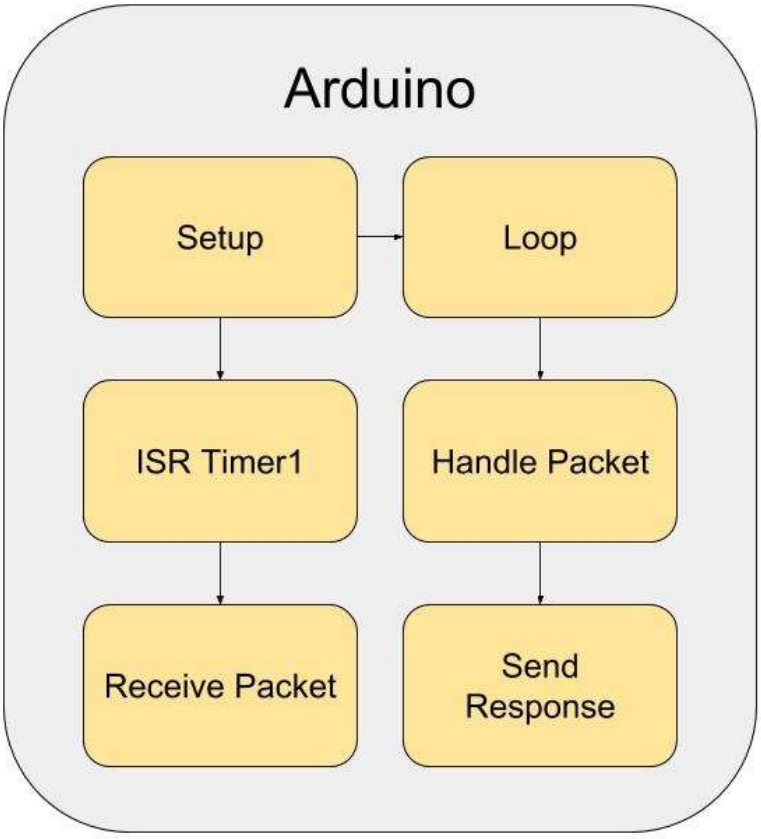
De applicatie is een Demo applicatie waarin het omhoog of omlaag zetten van GPIO pinnen kan worden getest. Het maakt hierbij gebruik van de JackIO API waardoor het gemakkelijk te implementeren viel.

Test versie Android App, pin naar keuze kan worden getoggled.



JackIO Board

Het JackIO board is gebaseerd op de Arduino Firmware. En wordt ook dusdanig geprogrammeerd hiermee. Figuur 3 is een globaal overzicht van de Arduino lifecycle.



De setup initialiseert het programma, en zorgt ervoor dat er 2 analog poorten gereserveerd worden als receiver en transmitter poort. Ook wordt de Analoge referentie pin op de EXTERNAL modus gezet om de analoge input waardes altijd te schalen op de actuele maximale voltage.
En wordt de ISR Interrupt Timer1 aangezet en overgenomen voor het constant uitlezen van de data lijn.

Om multithreading te simuleren gebruiken wij de constante clock van de Interrupt Timer1 om twee loops gelijktijdig te laten draaien. De hoofd Loop die altijd na de setup wordt uitgevoerd door Arduino en de ISR Timer1 functie.

De ISR Interrupt Timer wordt gebruikt van het uitlezen van de audio samples die over de data lijn naar binnen komen. Arduino leest standaard een 10bits getal uit (0 - 1024) en deze mapping is gemaakt van 0V tot aan de externe maximale voltage. Doordat in de hardware wij het signaal altijd met een spanningsverdeler omhoog trekken met de helft van het maximale voltage is de null lijn van het lezen altijd 512, en beschouwen wij samples die niet groter zijn dan 512 + 50 of 512 - 50 altijd als ruis om geen foutieve lezingen te begaan.

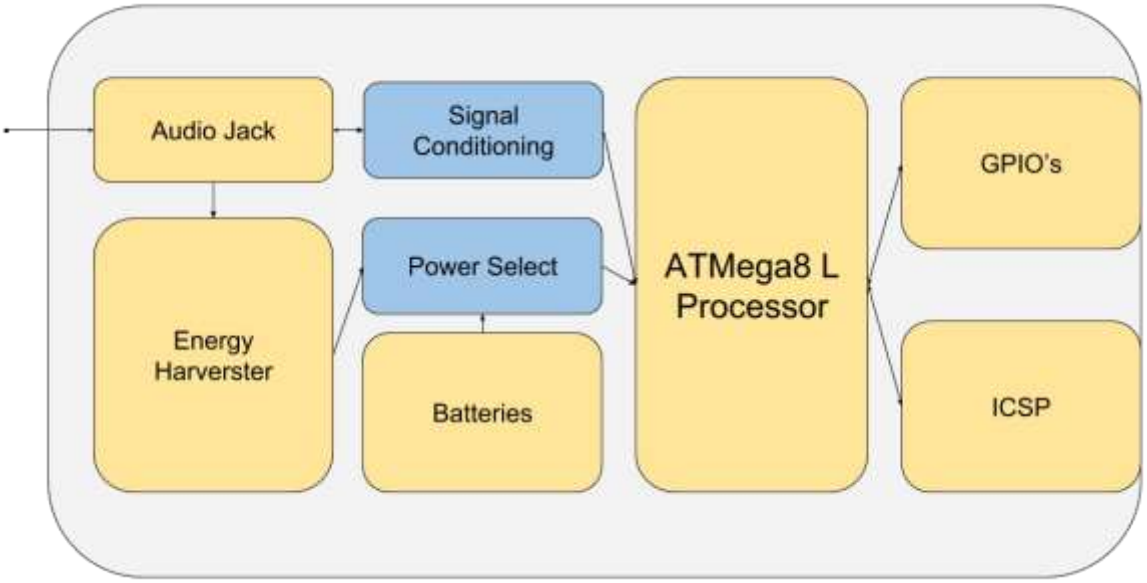
Vervolgens als er een start wordt gevonden in het bericht wordt hij verder gedecodeerd als manchester op dezelfde methode als boven beschreven. Uiteindelijk worden alle gevonden bytes omgezet een een Packet object en wordt er een flag omgezet dat er een paketje is gevonden.

De hoofd loop checkt of deze flag omhoog staat en handelt dit pakketje af. Hierbij checkt hij of er een pin moet worden uitgelezen of dat er een waarde naartoe moet worden geschreven. Als er een pin moeten worden uitgelezen, leest hij deze uit en bouwt daarvan een response pakketje. En stuurt dit berichtje terug over de transmit poort terug.

Hardware

De hardware van het JackIO board bestaat uit meerde componenten.

- Power Harvester, de stroomvoorziening vanuit de right audio uit de telefoon.
- Left audio, het data signaal wat uit de telefoon naar de microprocessor gaat.
- MIC, het signaal wat uit de microprocessor naar de telefoon gaat.

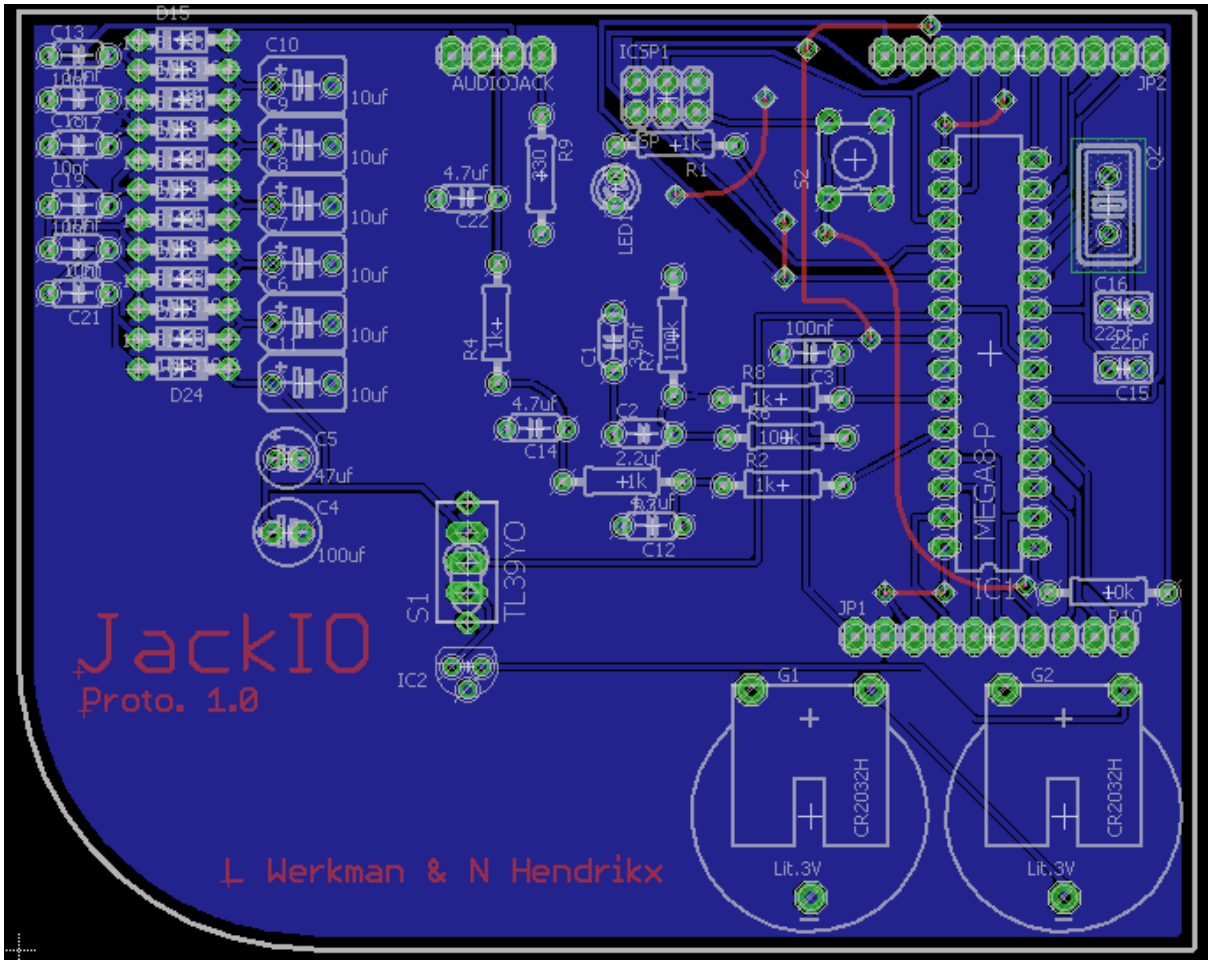


Specificaties

Het JackIO board kan worden aangesloten op een audiojack van een telefoon.
Er zit een Atmega8L microprocessor op, deze hebben we gekozen vanwege zijn lage energie verbruik en lage ontwikkel kosten.

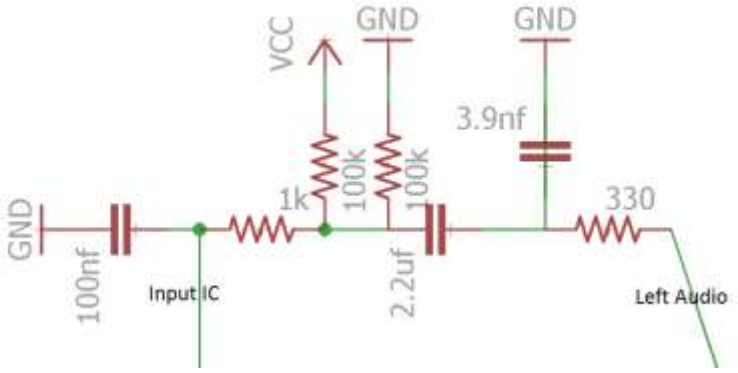
Tevens is deze microprocessor nog programmeerbaar vanaf het JackIO board, er zit namelijk een ICSP pinheader op.
Voor de stroomvoorziening kan er gekozen worden tussen een telefoon en de batterij.
Er zitten 4 analoge pinnen op, en 13 digitale pinnen.

Hieronder de weergave van de PCB, het is een 2 laags PCB.



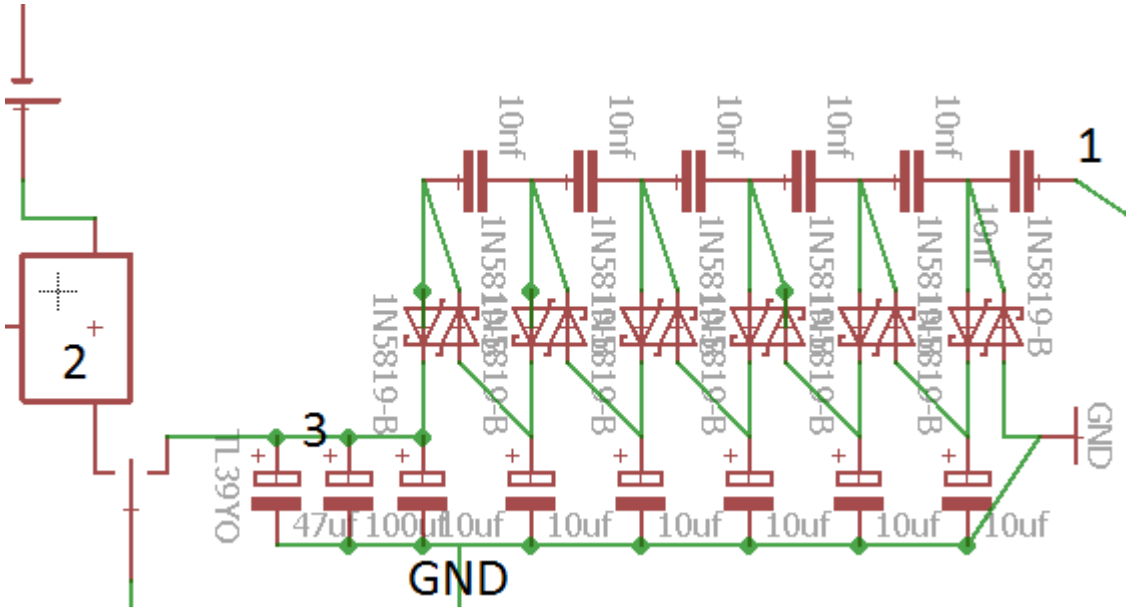
Left Audio

Hieronder een schematische weergave van de left-audio data ingang.
De weerstanden filteren het signaal optimaal voor de IC.
De 2x 100k weestanden zijn er voor de het maken van de spanning verdeler, dit verhoogt het signaal met de helft van het VCC signaal. Dit resulteert in dat de spanning nooit negatief is, en kan worden uitgelezen door de ADC.
De condensatoren zijn er voor de galvanische scheiding en signaal conditionering.



Power Harvester

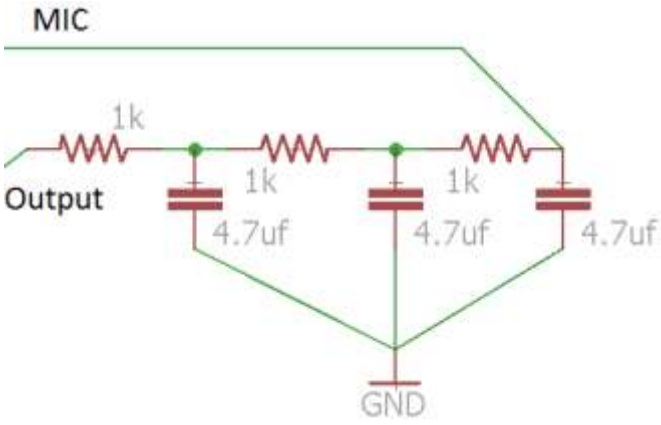
Hieronder een schematische weergave van de power harvester dit is een cascade.
Bij 1 komt de rauwe sinuswave uit het rechter audio kanaal van de telefoon.
Daarna wordt het voltage opgebouwd. Bij iedere trap wordt het voltage verdubbelt.
Elke trap in de cascade heeft een drempelwaarde van 300 millivolt.
Bij 3 zijn nog twee extra elco's aangesloten met een waarde van totaal 150uf. Deze zijn er om een buffer op te bouwen. Dit gaat bij 2 een switch in waar geschakeld kan worden tussen de stroomvoorziening van de telefoon of batterij.



MIC

Hieronder een schematische weergave van de MIC.

Deze bestaat uit een drie traps lowpass filter, dit zorgt voor een blokwave getransformeerd wordt in een sinuswave.



Afsluiting

Dit is project JackIO, voor vragen verwijzen we u door naar onze websites http://www.nielshendrikx.nl of http://www.larswerkman.com.
Natuurlijk kunt u ook persoonlijk contact met ons opnemen.