Chameleon Mixer

Mesa controladora de efeitos de vídeo em tempo real

Por Jorge Augusto Moreira Ribeiro

Júri:

Orientador: Professor Paulo Peixoto

Presidente: Professor Jorge Baptista

Vogal: Professor Paulo Menezes

Mestrado em Tecnologia de Informação Visual

Departamento de Engenharia Electrotécnica e de Computadores

Faculdade de Ciências e Tecnologia

Universidade de Coimbra

Setembro de 2010

Agradecimentos

Gostaria de agradecer ao professor Paulo Peixoto que me apoiou sempre, mesmo nas minhas escolhas mais ambiciosas.

À minha família, por todo o apoio, motivação e formação que me deu, permitindo que os meus sonhos fossem realizados.

Aos meus amigos, que me apoiaram em todas as etapas académicas e estiveram sempre presentes nos momentos menos bons, assim como nos momentos mais marcantes, pelo que deixarão muitas saudades deste percurso.

A todos os que me ajudaram e motivaram ao longo do trabalho na tese, bem como àqueles que me ajudaram a corrigir este documento.

Um agradecimento especial para o Luís Maricato, pelo apoio em momentos de problemas com electrónica, ao Pedro Ângelo, pela motivação nas escolhas mais arrojadas e complicadas, e à Irina, pela oportunidade de aprender mais e praticar *VJing*.

À tvAAC, por me ter dado a conhecer o *VJing* e por todo o apoio de material técnico, à Audiência Zero e seus laboratórios, pelas oportunidades que me deu de discutir, criar e mostrar o meu trabalho.

Aos meus colegas do ISR, pelos conselhos, pela união e pela amizade em todo o tempo que passámos juntos.

A todos os que seguiram o meu percurso, mesmo aos que não acreditaram em mim, pois foram uma grande força de motivação.

Resumo

O projecto Chameleon Mixer consiste na criação do *hardware* e do *software* para uma mesa controladora de efeitos digitais em tempo real.
Foi inspirada no camaleão e pretende dar uma visão diferente daquilo que assumimos como real. Recebe informação dos seus sensores e aplica efeitos que vão repercutir-se na imagem final.
A mesa de efeitos é controlada pelo utilizador através de potenciómetros ou de sensores digitais, de forma a poder controlar mais fácil e autonomamente os efeitos.
O *hardware* é baseado numa placa de desenvolvimento conhecida como Arduino [1] e que contém o microcontrolador ATmega1280. É utilizado o Processing [2] para a comunicação entre o microcontrolador e o *software* da Chameleon Mixer. O *software* é desenvolvido com o OpenFrameworks [3] (C++), onde se faz o processamento de imagem em tempo real e se aplicam os efeitos.

Palavras-chave:
Mesa de Mistura, controlador de efeitos, processamento de imagem, efeitos em tempo real, *VJing*.

Abstract

Chameleon Mixer projects consists on software and hardware creation for a real time digital video mixer.
It is inspired on chameleon and it aims to give us a different sight of the things we assume as real. The Chameleon Mixer receives information through its sensors and adds effects to the input image.
The mixer is controlled by the user through sliders or digital sensors. In that way, it can control easily and by itself the effects.
The hardware platform is an Arduino that contains the microcontroller ATmega1280. It is used Processing to do the communication between the microcontroller and Chameleon Mixer software. The software is developed with OpenFrameworks using C++ language. OpenFrameworks is also the framework where real time image processing is done and where effects are applied.

Keywords:
Mixer, effects controller, image processing, effects in real time, VJing.

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Pseudo-Código

Lista de Acrónimos

Termos Essenciais	Definição
Arduino	O *Arduíno* é uma plataforma de *hardware* livre, com um microcontrolador de placa única. Corre em *Windows*, *Mac* e *Linux*.
DJ	O "Disc Jockey" é um artista profissional que selecciona e toca diferentes músicas, previamente gravadas.
Flat cable	O cabo plano é um cabo formado por vários fios dentro de uma única fita plástica.
GUI	"Graphic User Interface", em português, interface gráfica do utilizador, é um tipo de interface no qual os utilizadores interagem com dispositivos digitais através de elementos gráficos.
LCD	O *display* de cristais líquidos é um painel fino usado para exibir informações electronicamente.
LDR	O "Light Depedent Resistor" é um tipo de sensor cuja resistência varia conforme a intensidade de radiação electromagnética do espectro visível que incide sobre ela.
LED	O diodo emissor de luz, "Light Emitting Diode" é um diodo semicondutor. Quando lhe é fornecido uma fonte de energia, emite luz visível.
MIDI	Interface Digital para Instrumentos Musicais ("Musical Instrument Digital Interface") é uma tecnologia *standard* para comunicação entre instrumentos musicais e equipamentos electrónicos.
Mixer	*Mixer* ou misturador é um aparelho electrónico que combina dois ou mais sinais electrónicos num ou dois sinais de *output* combinados.

OpenFrameworks	É um *toolkit* multiplataforma de código aberto desenhado para o desenvolvimento de código criativo. O OpenFrameworks é escrito em C++.
Pino (Arduino)	São portas de comunicação físicas, de *input* ou *output*, digitais ou analógicas, de comunicação com outros equipamentos electrónicos.
Processing	Linguagem de programação multiplataforma de código aberto que contém um ambiente integrado de desenvolvimento.
stand-alone	São programas completamente auto-suficientes. Não necessitam de outro *software* auxiliar para o seu funcionamento.
Vídeo Arte	A vídeo arte é uma forma de expressão artística que utiliza a tecnologia do vídeo em artes visuais.
VJ	VJ (pronuncia-se em inglês víi-djêi) ou *video jockey* é a denominação geralmente dada aos profissionais responsáveis pela manipulação em tempo real de vídeos em eventos ao vivo ou programas de televisão, tendo estes vídeos como função, na maioria das ocasiões, a ilustração de músicas e a transmissão de sensações diversas para o público.
VJing	O *vjing* (pronuncia-se víi-djáin), nome dado à profissão/actividade profissional dos manipuladores de vídeo em tempo real. Usa elementos de vídeo puro e simples, videoarte, tv, videojogos, cinema e uma infinidade de influências do meio audiovisual.

Tabela 1.3.1 – Glossário

1 Introdução

Surgem, cada vez mais, ambientes de informação visual, assim como a necessidade de controlar esta informação em tempo real. É necessário orientar e processar os meios visuais em meios artísticos, informativos ou mesmo em meios industriais. Na vertente artística, a performance de vídeo em tempo real impõe a existência de sistemas inovadores e cada vez mais desenvolvidos para controlo de informação visual tanto em instalações artísticas como em *VJing* [4].

O trabalho desenvolvido e descrito neste relatório envolve a criação de um sistema controlador de efeitos visuais em tempo real.
Para este propósito foi desenhado e montado o *hardware,* que consiste numa mesa controladora de efeitos. Foi também desenvolvido um *software*, que recebe os *inputs* do *hardware* e processa-os em tempo real, de forma a obter uma imagem final manipulada pelo utilizador.
Este sistema é direccionado especialmente para a utilização em contextos de vídeo arte em tempo real.
Este capítulo faz uma introdução ao projecto, estado da arte, objectivos, planeamento e estrutura do projecto.

1.1 Motivação

"Uma jovem suíça consegue ver cores ao ouvir uma música e é capaz de experimentar gostos que variam do azedo ou amargo a sabores mais específicos como o de relva recém-aparada (...) Um intervalo musical de segunda menor fá-la sentir acidez, enquanto o de segunda maior deixa um gosto amargo na sua boca. (...) ela apresenta o caso mais extremo de sinestesia, no qual a música estimula na jovem uma resposta em outros órgãos sensoriais."[5]

A motivação da criação do Chameleon Mixer é baseada na capacidade cerebral de construir imagens, criadas através de sensações obtidas por outros sentidos além da visão.

Foi inspirada no camaleão e pretende construir uma imagem sintetizada tendo como ponto de partida a percepção sensorial.

Com a utilização de diversos sistemas de *hardware* e *software* desde 2006, ao praticar *VJing*, rapidamente se constatou a sua limitação na possibilidade de manipular vídeo em tempo real.

Tendo conhecimento de sistemas de microcontroladores e de criação de *software* de imagem, estática e em movimento, surgiu naturalmente a ideia de um novo conceito de manipulação de informação visual em tempo real, de modo prático, intuitivo e instantâneo. Surge a mudança do paradigma actual de mistura de vídeos para um paradigma de mistura de efeitos, tendo como base a mesma fonte de vídeo.

Neste paradigma, a exploração de efeitos é maximizada e optimizada para uma *performance* baseada numa fonte de vídeo. A aplicação de efeitos é realizada de forma manual ou automática através dos sensores presentes na mesa.

1.2 Estado da Arte

1.2.1 Vídeo Arte

A arte aplicada ao vídeo é um processo que tem início no cinema de vanguarda europeu dos anos 20 e no cinema *underground* americano. Por analogia, transmite-se ao formato vídeo, pela manipulação e produção de imagens através de meios electrónicos e digitais.

Nos anos 60, a vídeo arte surge como movimento de expressão não comercial, muito crítico à cultura televisiva. Pretende-se, desta forma, explorar novas relações entre a imagem e o espectador, abordar novos conceitos espaciais de tendência abstracta.[6]

A vídeo arte revelou-se como um dos movimentos de vanguarda mais importante da última metade do século XX. Pode-se entender a vídeo arte enquanto forma expressiva no domínio das artes visuais e das novas tecnologias multimédia. Actualmente, o campo da vídeo arte é muito abrangente, podendo definir-se vários ramos: vídeo arte, instalação vídeo, vídeo *performance*, *video jamming*, entre outros.

A instalação de vídeo assume-se como um método de arte contemporânea que combina tecnologia de vídeo com instalação de arte. É uma arte que utiliza o ambiente envolvente como veículo de comunicação com a audiência.

Hoje em dia, as instalações de vídeo estão presentes numa variedade de ambientes que vão desde as galerias e museus, a trabalhos específicos em paisagens urbanas ou industriais. Os formatos mais populares incluem projecção e performance. Os requisitos passam pela electricidade e por um ambiente relativamente escuro.

Desde há algum tempo, os espectáculos musicais contam com elementos visuais, como a iluminação. Actualmente, com o avanço tecnológico, torna-se comum a presença do VJ. De uma forma resumida, trata-se de um artista que geralmente utiliza *software,* num computador ligado a um projector, para exibir imagens criadas ou transformadas pelo artista no momento.

O *VJing* é a designação para uma performance visual. As características do *VJing* prendem-se com a criação ou a manipulação de imagens em tempo real e em sincronia com a música, para uma determinada audiência, recorrendo à utilização de tecnologia [4].

A criação, produção e manipulação de imagens projectadas para exibição durante concertos musicais é considerada uma nova linguagem artística e torna-se cada vez mais comum.

No contexto da música electrónica, a imagem tem o papel da criação de ambientes imersivos multissensoriais, completando a experiência musical com um conjunto de sinestesias [7].

Equivalente ao DJ no campo imagético, o VJ é o profissional responsável pela edição em tempo real das imagens projectadas. Desta forma, a construção de vídeos ao vivo revela-se uma desconstrução do sistema tradicional de produção visual, pois não existe um produto acabado, mas sim diferentes criações vinculadas a cada performance.

1.2.2 Conteúdos

"As fontes dos nossos conteúdos vêm de toda a parte, também filmamos o nosso próprio material. Criamos mundos e objectos em 3D, compomos elementos em After Effects, Combustion e no Shake. Ideias vêm de uma grande variedade de fontes - cadernos, caixas de fotos, recortes e embalagem, (...) Os acidentes e os erros são incluídos como possibilidades."[8]

Habitualmente, os VJ's criam os seus próprios conteúdos, filmando, reeditando ou desenvolvendo novos vídeos, através de *software* de criação de vídeo. Constroem, assim, imagens, vídeos ou animações, que serão depois misturados e mostrados por si numa performance. Também é possível comprar ou descarregar gratuitamente da internet este material, embora os VJ's com mais experiência se dediquem a construir os seus próprios conteúdos.

"A produção de música electrónica é baseada em amostras, e nas suas repetições e variações. Da mesma forma, clipes de vídeo (ou programas algorítmicos) são os elementos

básicos da performance visual em tempo real. (...) Isto significa que, mesmo que um clipe dure 10 segundos, ele pode ser apresentado em loop por um minuto ou mais.
Muitas das vezes estes vídeos são preparados para que o primeiro e último sejam iguais, para que, quando esse vídeo estiver a ser tocado repetidamente, não seja possível detectar o seu início e o seu fim, dando a ilusão de continuidade. A estes vídeos dá-se o nome de loop." [9]

É frequente um VJ ter entre 10 a 100 *loops* sobre o mesmo tema, podendo na sua performance criar um momento, uma narrativa, através da sequência cinematográfica dos *loops*. A este conjunto de *loops* sobre o mesmo tema dá-se o nome de *deck*.

Numa performance, um VJ toca vários *decks* de acordo com a música, o ambiente e o público, criando desta forma um *Set*.

1.2.3 Performance

"A arte performativa é uma arte em que as acções de um indivíduo ou de um grupo, num determinado lugar e num tempo particular, constituem a obra. Pode acontecer em qualquer lugar, a qualquer momento, ou por qualquer período de tempo. Performance pode ser qualquer situação que envolve quatro elementos básicos: tempo, espaço, o corpo do performer e a relação entre artista e audiência." [10]

Cada performance de *VJing* localiza-se temporalmente num espaço, num determinado contexto, e todos estes factores fazem com que a performance seja única. Outros agentes que influenciam a performance são o estado de espírito do VJ, a música, o público, o ambiente, o *hardware* e o *software* com que se está a trabalhar. Todas estas condicionantes podem influenciar radicalmente uma performance.

1.2.4 Hardware

O *hardware* utilizado normalmente em VJ pode dividir-se nas seguintes categorias:

- *Hardware* para *input*, que gera imagem de vídeo e que pode ser manipulado pelo VJ, p. ex., câmaras ou sintetizadores de vídeo.
- *Hardware* para reproduzir vídeo a partir de um disco ou um suporte multimédia, como DVD ou VHS.

- *Hardware* para mistura, que permite a combinação de múltiplos canais de vídeo, como p. ex., mesa de mistura de vídeo, ou um computador utilizando *software* para *VJing*.
- *Hardware* de *output* para exibir o sinal de vídeo, como p. ex., um projector de vídeo, ecrãs de LED ou plasma.

O *setup* de *hardware* mais comum em VJ é um computador portátil, cuja placa gráfica tem capacidade de processar vídeo em tempo real, um projector e uma tela de projecção de imagens. A este *setup* adiciona-se, muitas vezes, uma mesa de mistura de vídeo, que permite comutar a entrada de vídeo, p. ex., entre o computador e um leitor de DVD's. A noção do funcionamento e a disponibilidade de múltiplos cabos, conectores e adaptadores são essenciais para ligar todo o equipamento e resolver possíveis problemas técnicos.

Um *setup* mais completo expande-se normalmente com uma câmara de filmar ou mesmo com mais VJ's ligados simultaneamente à mesa de vídeo.

Uma peça comum é um controlador *MIDI* ligado ao computador, com o objectivo de obter um controlo do *software* de forma física e com *feedback* táctil.

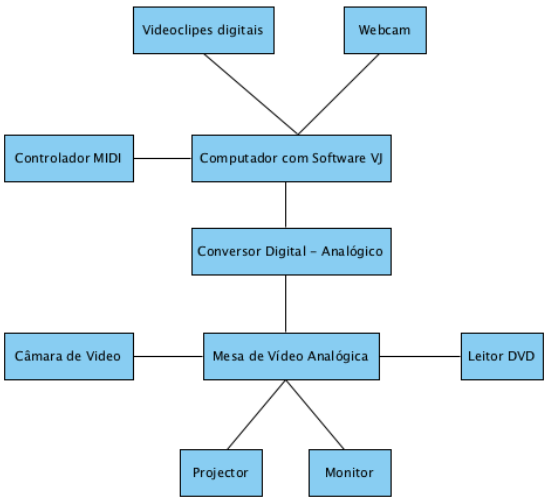


Figura 1.1 - Diagrama de Componentes de um Sistema de VJ frequente com Computador

Outros VJ´s dispensam o computador e operam apenas com leitores de DVD's e uma mesa de mistura, aproximando-se mais do paradigma de um DJ.

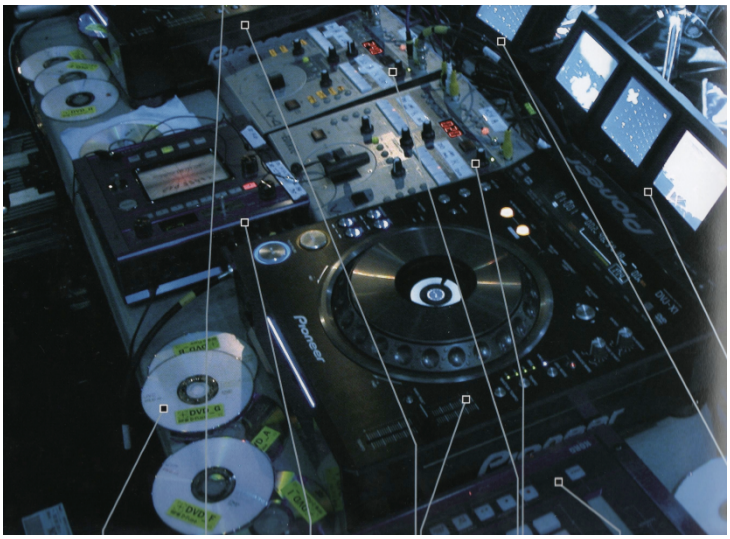


Figura 1.2 - *Setup* dos VJ D-FUSE[8]

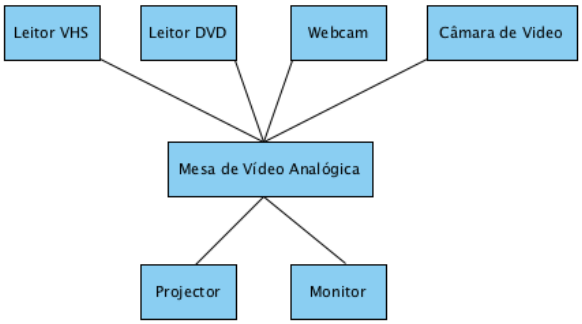


Figura 1.3 -Diagrama de Componentes de um Sistema de VJ sem Computador

Setups mais radicais incorporam microscópios, comandos de consola, mesas de desenho, tintas, equipamento musical, ou mesmo *hardware* criado pelo próprio VJ.

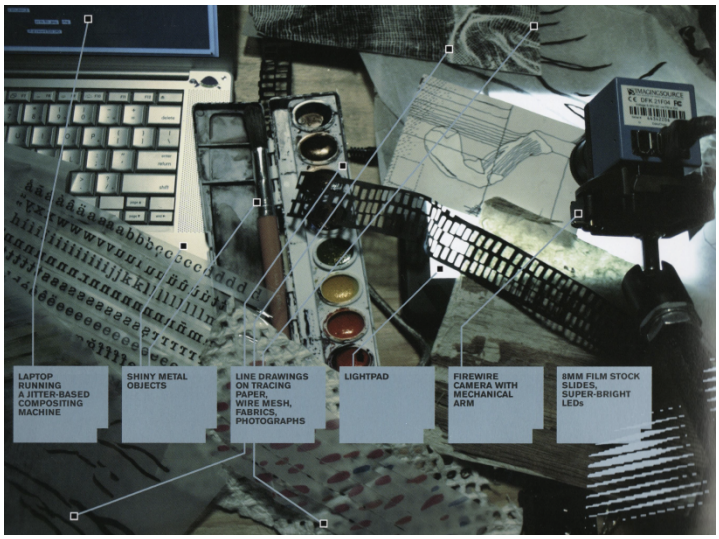


Figura 1.4 - *Setup* dos VJ SUE.C[8]

A mesa de mistura de vídeo mais comum no meio *VJing* é a Edirol V4, pois revela-se uma mesa de baixo custo com características direccionadas para esta actividade. Conta com quatro entradas de vídeo, incluindo entrada para ligar um computador, um manípulo de mistura, que pode ser facilmente rodado para ficar numa posição horizontal, vários tipos de transições, *input* Midi e possui alguns efeitos pré-definidos que podem aplicar-se ao vídeo.

Figura 1.5 – Mesa de edição linear de vídeo Edirol V4[11]

Em anexo, no apêndice A encontra-se mais informação sobre vários *setups* e tipo de ligações.

1.2.5 Software

Os *software*s mais comuns para *VJing* têm como objectivo principal a mistura de *loops*, por uma ordem dada pelo VJ. É também possível aplicar alguns efeitos pré-definidos pelo *software*.

Actualmente, os *software*s mais utilizados para a prática do *VJing* são o Modul8 [12] Grand VJ/Arkaos [13], Resolume [14], VJamm [15], FLxER [16], VDMX [17] e o Livid CellDNA [18]. Todos eles possuem uma interface que permite utilizar várias *layers* de *loops*, em tempo real. Alguns destes *software*s, como o Modul8 e VDMX, funcionam apenas com o sistema operativo MAC OS e outros, como o VJamm, apenas para Windows.

"O novo hardware e as inovações de software impulsionam a aparição do VJ superstar. Devido ao influxo de novas ferramentas (tanto comerciais como freeware) no mercado, há mais oportunidades para os VJs (...) No entanto, os VJs também programam o código das

suas próprias ferramentas, tornando-as disponíveis para os outros. O VJ torna-se um criador de software, para além de mostrar imagens ao vivo. Artistas como Netochka Nezvanova que escreveu Nato.0 55 ou Puckette Miller, autor de pd, tornam-se influentes, devido à popularidade de seus softwares." [19]

Muitos VJ´s desenvolvem as suas próprias ferramentas, orientadas ao seu estilo de performance. Alguns ambientes gráficos de programação são o MAX/MSP/Jitter, VVVV, o Isadora e o Pure Data, criados para um rápido desenvolvimento, sem existir a necessidade de profundos conhecimentos de programação.

Foi desenvolvida uma arquitecta de *plugins* para efeitos de vídeo chamada FreeFrame, de forma a poder partilhar efeitos de vídeo em tempo real entre os diferentes *softwares* de VJ.

No entanto, todos estes efeitos não permitem uma interacção eficaz com o público ou uma grande variação de parâmetros, de modo a estabelecer uma boa identificação e ligação com a assistência. Devido a este factor, muitos VJ´s utilizam poucos efeitos nos seus vídeos, pois não se cria nenhuma relação de intimidade com eles.

1.3 Objectivos da Tese

Os objectivos deste projecto baseiam-se na criação de um sistema de mistura de efeitos no vídeo, que possam ser utilizados em contextos de *VJing* ou em instalações artísticas.

Ao contrário dos sistemas actuais de *VJing,* não se pretende criar uma mesa que misture *loops*, mas sim um sistema de sobreposição ou criação e mistura de efeitos, com algoritmos programados para um contexto específico.

O sistema proposto é um conjunto formado por uma mesa controladora e um *software* composto pela comunicação, aplicação de efeitos e interface gráfica para a visualização dos vídeos.

Este sistema é controlado manualmente através de potenciómetros e possui sensores para controlar a mesa de uma forma automática.

O *hardware* é construído de raíz, reutilizando *hardware* obsoleto e comprando apenas o material impossível de reciclar. Terá como microcontrolador um processador ATmega1280 numa plataforma de desenvolvimento Arduino Mega, para controlar e receber os *inputs* dos vários componentes.

Genericamente, a mesa pode ser dividida em quatro partes distintas: na selecção de efeitos, no controlo do canal A, no controlo do canal B e nos sensores.

Devido à inexistência no mercado de peças de *hardware* com estas características ou de fácil adaptação, foi necessário criar o *hardware* de forma a responder fielmente ao conceito proposto.

O *software* é repartido entre o *firmware* do Arduino, o Processing para a comunicação com o Arduino, e o OpenFrameworks (C++) para desenvolvimento da interface gráfica e aplicação de efeitos no vídeo em tempo real.

Pretende-se que o *software* receba um *input* de vídeo e que o transforme, aplicando os efeitos que o utilizador seleccionar na mesa.

Cada efeito terá cinco parâmetros que poderão ser controlados directamente pelos potenciómetros da mesa, ou então utilizar a informação proveniente de um sensor.

1.4 Planeamento do trabalho

No início deste projecto foram definidos os objectivos anteriormente referidos e ainda os requisitos específicos da aplicação. Tratam-se dos requisitos de Interface, Funcionais, Restrições de Design, e outros requisitos. Fez-se também a categorização dos requisitos para a Criação da Estrutura, para a Montagem de *Hardware*, *Firmware*, e *Software*. Estes requisitos podem ser consultados detalhadamente no Anexo Apêndice B. É importante referir que muitos deles tiveram que ser adicionados ao longo do projecto conforme as necessidades práticas. Aqui mostram-se igualmente as várias etapas do trabalho desenvolvido e a carga de esforço aplicada a cada uma delas. Em cada dia de trabalho criou-se um registo da data, do tempo de trabalho, da fase do projecto e colocou-se uma descrição com os desenvolvimentos efectuados, as dificuldades e as notas sobre os progressos seguintes, como mostra o exemplo em anexo Apêndice B. Este registo foi bastante importante para o desenvolvimento do presente relatório.

Com base nesta informação foi criado o Diagrama de Gantt seguinte:

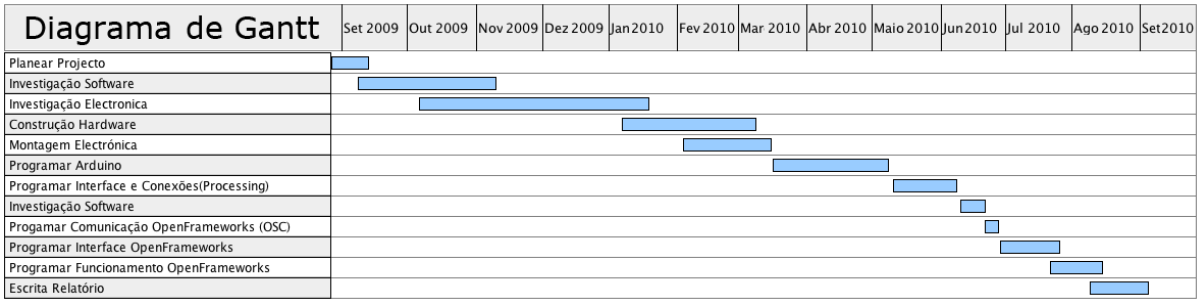


Figure 1.6 - Diagrama de Gantt demonstrativo do tempo dedicado a cada etapa

Etapa	Data Início	Data Final	Percentagem de Tempo
Planear Projecto	31-Ago-09	13-Set-09	2.06%
Investigação *Software*	09-Ago-09	21-Jan-10	8.66%
Investigação Electrónica	10-Jun-09	24-Jan-10	15.26%
Construção *Hardware*	01-Jul-10	03-Set-10	12.85%
Montagem Electrónica	02-Abr-10	16-Mar-10	11.64%
Programar Arduino	17-Mar-10	05-Nov-10	11.57%
Programar Interface e Conexões(Processing)	05-Nov-10	06-Set-10	0.92%
Programar Comunicação OpenFrameworks (OSC)	22-Jun-10	28-Jun-10	1.77%
Programar Interface OpenFrameworks	29-Jun-10	26-Jul-10	8.45%
Programar Funcionamento e Efeitos no OpenFrameworks	22-Jul-10	15-Ago-10	11.78%
Escrita Relatório	08-Ago-10	03-Set-10	15.05%
Total	31-Ago-09	03-Set-10	100.00%

Tabela 1.1 - Tempo despendido em cada etapa

Com esta tabela é possível verificar que as etapas de investigação ocuparam bastante tempo, cerca de 24% do tempo total; a etapa de construção da mesa, bem como da parte electrónica, cerca de 24% e a parte de programação do *firmware* e do *software* cerca de 34%.

2 Sistema

Este capítulo apresenta o conceito proposto para a Chameleon Mixer, tanto ao nível da sua utilização como da arquitectura de todo o sistema.

2.1 Conceito

O sistema proposto afasta-se do sistema de *VJing* comum, que é direccionado para a mistura de imagens diferentes. Neste sistema, o paradigma assenta na mistura de efeitos visuais, tendo como fonte o mesmo canal de vídeo.

Em relação ao actual *VJing*, este sistema propõe uma mudança, que visa mudar o foco dos *loops* de vídeo para os efeitos. Aqui, os efeitos terão o papel principal, pois devem criar-se ou adaptar-se em função da situação, do público, do espaço, do tempo e da presença de música, entre outros factores.

A mesa terá como cor o verde e o preto. Estas cores estarão presentes nos LEDs, no LCD e nas descrições encontradas no *hardware*, bem como nos elementos gráficos no *software*. O verde é a cor representativa do camaleão e funciona muito bem em ambientes nocturnos e de pouca luz. Tanto no *hardware* como no *software,* a cor dominante será o preto, de modo a ser visível apenas o verde em ambientes escuros. Deste modo, o *hardware* mistura-se com o restante material e o *software* não emite luz desnecessária incomodando o trabalho do VJ, óptimo em ambientes escuros.

Estão previstos vários cenários de utilização da Chameleon Mixer.

2.1.1 Modelo “Interacção”

O primeiro cenário visa criar um efeito de realidade aumentada que interaja de forma semi-automática ou automática com esse espaço ou público. São aplicados efeitos sobre a entrada de vídeo, tendo um processamento sobre este. É possível ter uma câmara virada para o público e com um algoritmo de detecção de faces, criar um efeito que acrescente, p. ex., algo a essa realidade. Neste paradigma, o público passa a fazer parte do próprio ambiente, interagindo com ele.

Outra forma de explorar este sistema é ter uma imagem digitalizada para o *software* da fachada do edifício onde se vai projectar o vídeo final. O sistema poderá interpretar essa fachada e criar, p. ex., falsas sombras no edifício que interajam com a música de uma forma automática (Ver anexo apêndice B).

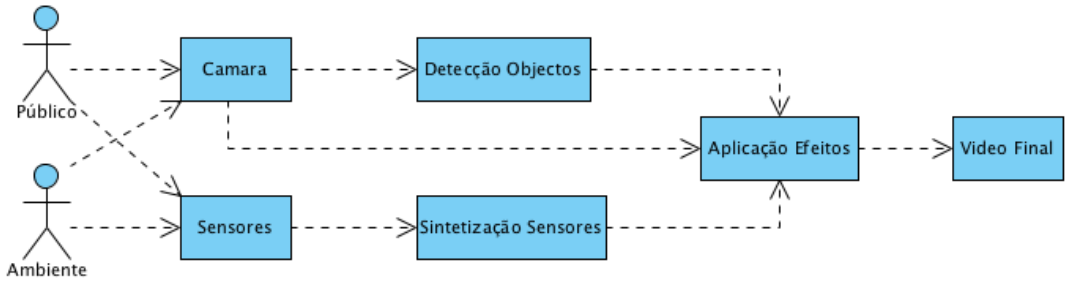


Figura 2.1 - Diagrama de como trabalhar com o sistema Chameleon Mixer – Modelo "Interacção"

2.1.2 Modelo "Série"

O sistema poderá ainda funcionar com o trabalho de um outro VJ e receber o seu *output* de vídeo para o processar com os efeitos pretendidos.

Ao contrário dos sistemas "tradicionais" com um modelo paralelo de mistura de vídeo e aplicação de efeitos, propõe-se aqui um sistema em série, onde o primeiro VJ mistura apenas os vídeos e o segundo, com o sistema Chameleon Mixer, mistura os efeitos.

Neste sistema existe uma vantagem clara ao modelo paralelo em relação à congruência visual. Enquanto num sistema paralelo os dois VJ's tentam conjugar o estilo de vídeos, o ritmo, a cor e os efeitos, neste sistema o VJ dos efeitos tem apenas de conjugar o estilo de efeitos para aquele vídeo.

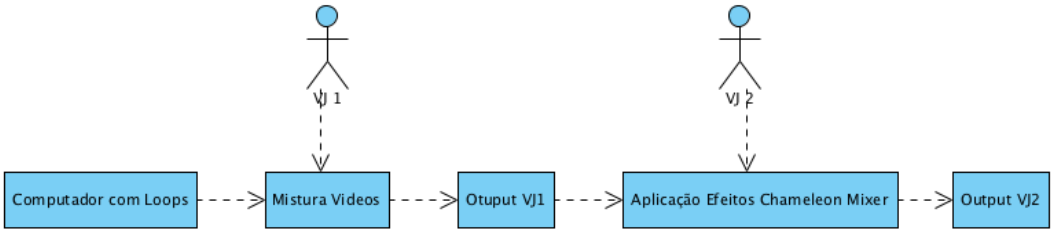


Figura 2.2 - Diagrama de como trabalhar com o sistema Chameleon Mixer – Modelo "Série"

2.1.3 Modelo "Gráfico"

Por último, uma terceira forma prevista para trabalhar com este sistema é utilizá-lo na criação de gráficos, tendo como base as primitivas gráficas dos respectivos efeitos. Aqui, o que se propõe é misturar efeitos numa abordagem mais próxima da mistura de vídeo. Neste caso, a mistura de efeitos gráficos tira partido em relação aos vídeos, visto que podem ter *inputs* pensados para modificar parâmetros de acordo com a música, tempo da *performance*, ou outros factores. A mistura entre efeitos também poderá representar um factor a explorar, dado que os métodos de mistura com este tipo de primitivas não se limitam à opacidade, luminosidade ou cor, como nos sistemas actuais de *VJing*.

Um exemplo para este tipo de paradigma, será a criação gráfica para uma peça de teatro, onde são desenvolvidos vários grafismos que complementam o *décor*. Neste caso, poderemos ter um efeito para cada cena que possua vários parâmetros variáveis conforme o dramatismo e o evoluir da cena. Neste exemplo, a transição entre os efeitos poderá ser uma sobreposição de cor apenas em *pixéis* que tenham uma variação de cor muito grande. Desta forma, é criada uma transição suave e de acordo com o movimento, ao contrário da mistura de vídeo, que pode parecer uma mistura forçada.

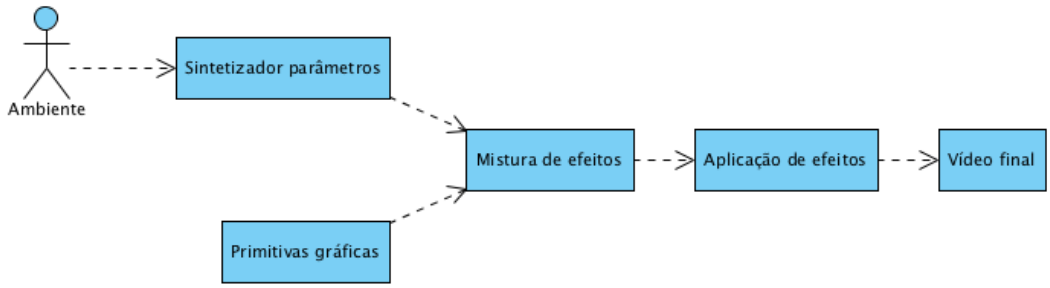


Figura 2.3 - Diagrama de como trabalhar com o sistema Chameleon Mixer – Modelo "Gráfico"

2.2 Arquitectura do Sistema

Ao nível da arquitectura, a mesa de efeitos visuais tem um conjunto de potenciómetros para controlar o valor de um efeito pré-definido ou então controlar a quantidade de um efeito, utilizando os valores do sensor.

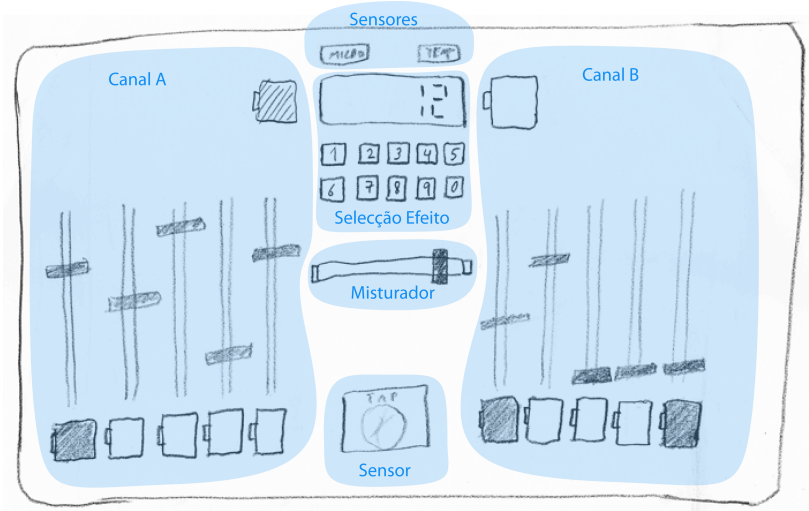


Figura 2.4 – Rascunho da mesa controladora de efeitos visuais e as suas diferentes componentes

No meio da Mesa existe um misturador entre o canal A e B. O valor que aparece no visor é o efeito seleccionado no canal A, se o botão do lado esquerdo for premido, ou então do canal B,

se o botão do lado direito for premido. Na restante mesa encontram-se os outros sensores, como se pode visualizar na figura 2.4.

Quanto ao *software*, será composto pela pré-visualização no lado esquerdo e, no lado direito, o vídeo final. Na pré-visualização podemos trocar de fonte do canal A ou B e podemos visualizar o *feedback* dos potenciómetros da mesa.
Por baixo de cada elemento gráfico do sensor é possível trocar os parâmetros dos efeitos, entre parâmetros controlados pelo potenciómetro ou valores vindos dos sensores. No caso de ser seleccionado um efeito, o potenciómetro apenas variará de forma automática conforme o valor recebido pelo sensor.

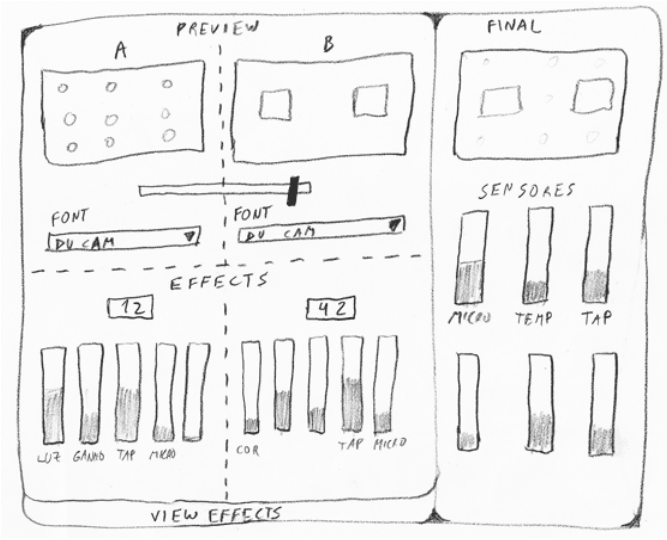


Figura 2.5 - *Layout* do *software* no modo de mistura de efeitos

Quando se clica na aba em baixo (“VIEW EFFECTS”) é possível ter uma rápida visualização e descrição dos efeitos existentes. Na secção do lado direito, por baixo do vídeo final, é visível o *output* dos sensores.

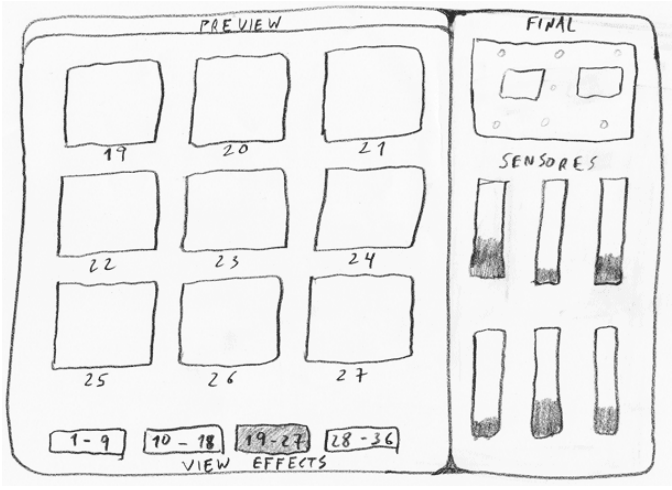


Figura 2.6 - *Layout* do *software* no modo de visualização de efeitos

Na actualidade, com a expansão de plataformas e com sistemas operativos como o Linux e Mac OS a ganhar cada vez mais utilizadores, no desenvolvimento deste sistema optou-se apenas por *software opensource* e multiplataforma.

Tomando como analogia o mundo animal e o Camaleão especificamente, a nível conceptual foi tomada a opção de reciclar a maior quantidade possível de material e componentes.

Trata-se de um sistema que prima pela versatilidade e pela abertura de canais de comunicação a outras fontes de sensores, bem como pela facilidade de interligação de efeitos.

Esta mesa tem como entradas os potenciómetros e os sensores. Através do Arduino far-se-á a comunicação desses valores para o *software*.

O *Software* Processing terá como *input* os valores enviados pelo Arduino e uma fonte de vídeo. Este *software* terá uma interface para o utilizador no ecrã principal e o *output* final no segundo ecrã, sendo assim possível ligar um projector ou outro dispositivo de imagem ao *output*.

A mesa terá entradas analógicas e digitais. No esquema seguinte pode ver-se a amarelo os componentes da mesa com entradas analógicas e a azul os componentes com entradas digitais.

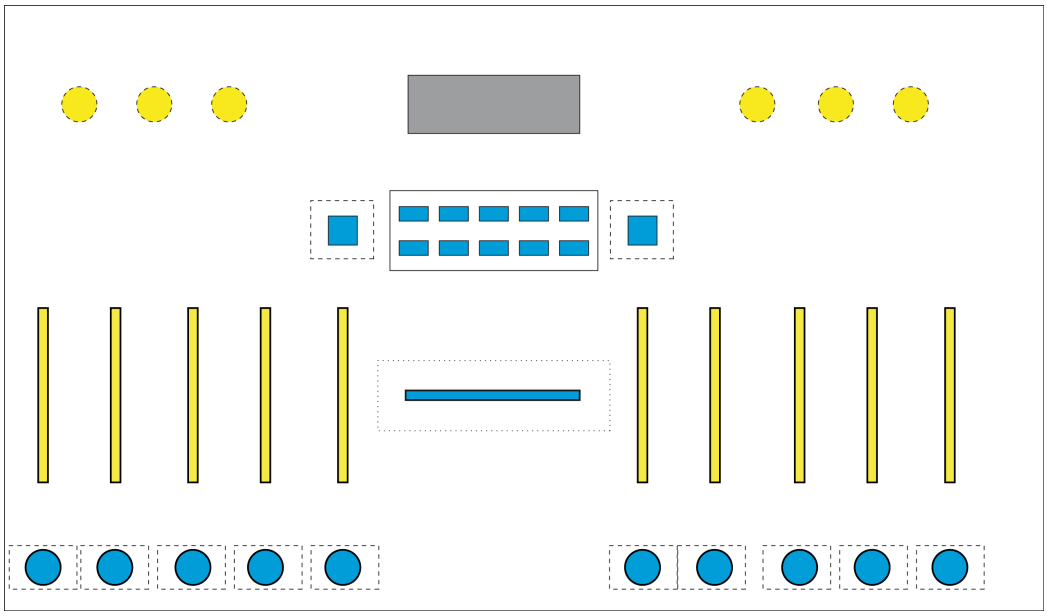


Figura 2.7 - Tipo de *inputs* na mesa

O Arduino recebe os valores dos componentes, processa-os e depois comunica para o *software* através do protocolo Firmata.

O *software* Processing receberá os *inputs* do Firmata, processa-os e utiliza esses valores para fazer o processamento de vídeo. Também mostrará os valores actuais da mesa através de uma interface gráfica.

As entidades que interagem com o sistema são o VJ, o ambiente (música, luz, temperatura, etc.), os *inputs* externos e a fonte de vídeo. Como *outputs* do sistema temos: a saída com os *previews* e os controlos para o monitor principal; o segundo monitor com a saída de vídeo final, que pode ser utilizada na projecção.

Os efeitos são *plugins* colocados numa pasta específica. O *software* vê constantemente se existem novos *plugins* e carrega-os automaticamente para o sistema, mesmo encontrando-se em execução.

No diagrama seguinte é possível ver a arquitectura do sistema:

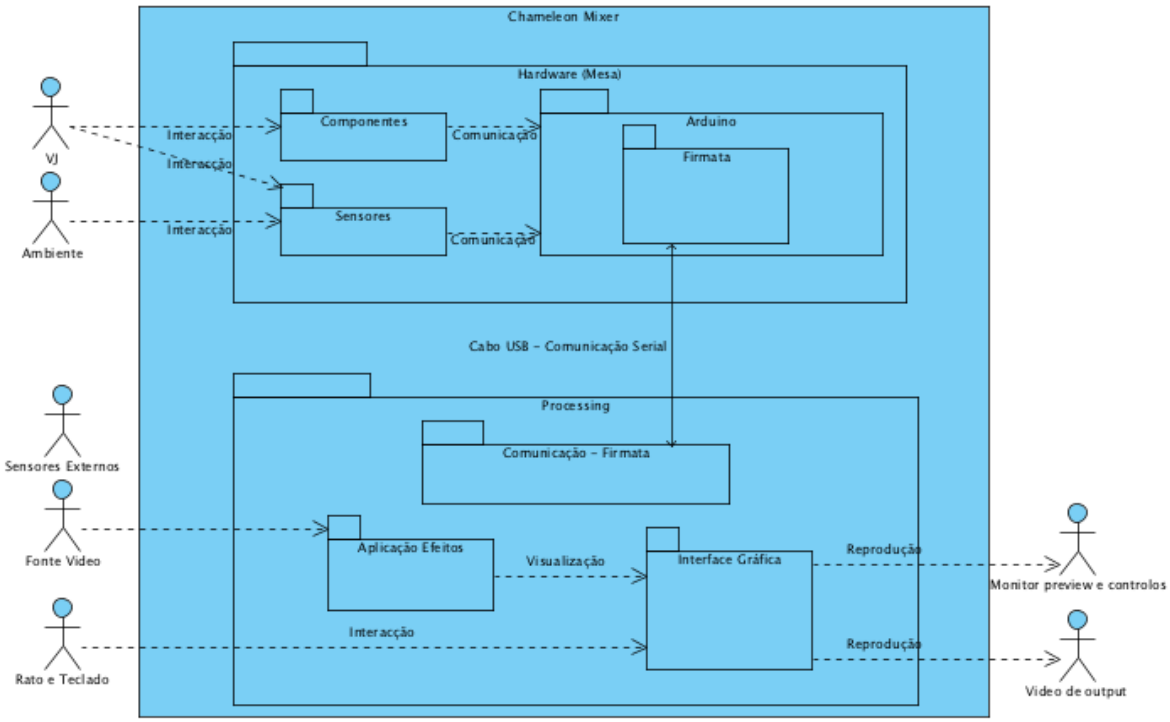


Figura 2.8 - Diagrama Conceptual Funcional do Chameleon Mixer

3 Hardware

Para desenvolver o projecto Chameleon Mixer foi necessário investir bastante tempo no desenvolvimento de conhecimentos de electrotecnia e na investigação dos componentes electrónicos disponíveis no mercado.

Em todo o processo adaptou-se constantemente o conceito original, de forma a conseguir integrar componentes equivalentes aos pretendidos, mas com tamanhos ou características diferentes.

Muitos componentes também tiveram de ser adaptados para conseguir o efeito pretendido.

Neste capítulo é descrito todo o processo de construção de estrutura da mesa, bem como a arquitectura e desenvolvimento do circuito electrónico.

3.1 Investigação de Electrónica

3.1.1 Arduino

O Arduino é uma plataforma de desenvolvimento *open-source* para protótipos electrónicos baseado numa plataforma flexível, com *hardware* e *software* de fácil utilização. O Arduino pode "sentir" o ambiente ao receber *inputs* de uma grande variedade de sensores e afectar o seu meio com luzes, motores ou outro tipo de actuadores. O microcontrolador da placa está programado na linguagem de programação própria, "Arduino" que possui um ambiente de desenvolvimento baseado no Processing. Os projectos em Arduino podem ser *stand-alone* ou comunicar com o *software* a correr no computador, como o Flash ou Processing.

Existe uma grande quantidade de literatura sobre o Arduino, tanto em livros [20], como na internet. No *site* do Arduino [1] é possível encontrar uma boa referência sobre as funções existentes, assim como exemplos de código para esta plataforma.

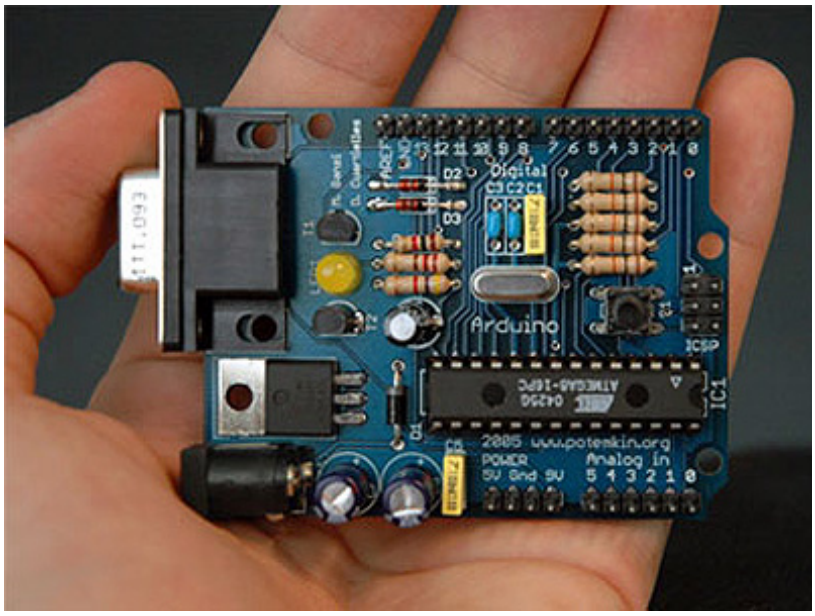


Figura 3.1 - Arduino Duemilanove [1]

3.1.2 Componentes

O desenvolvimento do projecto começou pela morosa tarefa de investigação electrónica e dos componentes mais indicados a utilizar nesta mesa.

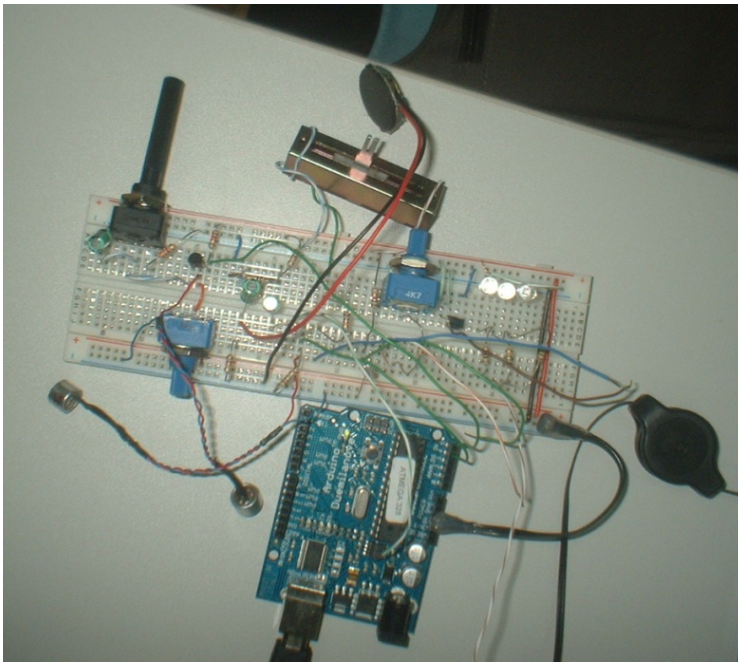


Figura 3.2 - Componentes a serem testados com um Arduino

Os primeiros componentes a serem testados foram os LEDs. Diferentes tipos de LEDs, de várias cores e com tensões nominais diferentes [21].

Nos LEDs é necessário ter atenção ao colocar resistências, de forma a diminuir a tensão de entrada.

O valor da resistência, R é dado por:

$$R = \frac{(V_s - V_l)}{I}$$

V_s = Tensão de entrada (em volts)

V_l = Tensão nominal do LED (em volts)

I = Corrente nominal do LED (em miliampere)

Em anexo Apêndice D é possível ver mais detalhes sobre os LEDs e as resistências utilizadas.
Foram testados potenciómetros rotativos e deslizantes, com variação logarítmica e linear. Foram escolhidos os potenciómetros lineares, visto que o *software* só interpreta valores com variação linear, e deslizantes por ser uma característica típica de uma mesa de mistura. Esta característica deve-se à possibilidade de utilizar vários potenciómetros ao mesmo tempo, de uma forma rápida e fácil [22].

Testaram-se vários LDR's, que são sensores compostos por células fotossensíveis. Este componente varia a sua resistência conforme a quantidade de luz recebida. O sensor que mostrou os melhores resultados, e com variações mais precisas foi o LDR SILONEX, que apresenta uma maior área fotossensível (mais características técnicas em Anexo apêndice D).

Foi testado um alcoolímetro, no entanto não foi visto nenhum enquadramento possível no conceito actual do Chameleon Mixer.

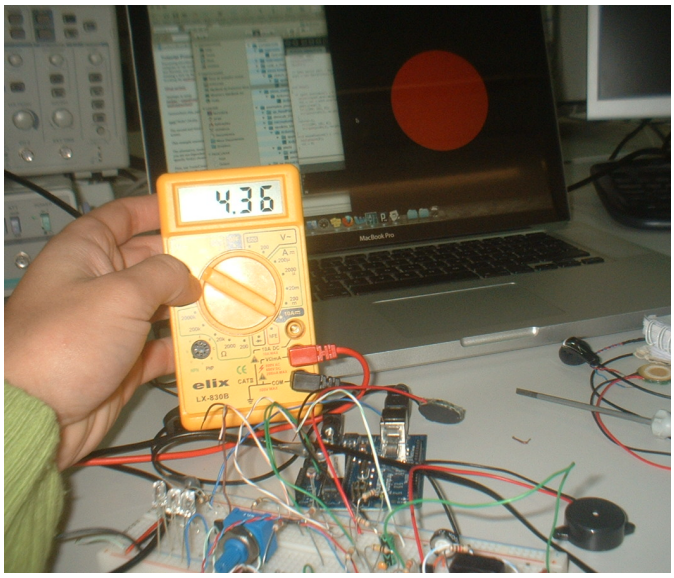


Figura 3.3 - Testes de sensores com multímetro e visualização no computador

Testaram-se sensores piezoeléctricos como microfones. Este componente tem algumas propriedades interessantes, visto que pode utilizar-se como um microfone [23] ou como um sensor de pressão [24].

Construíram-se alguns circuitos amplificadores para o piezoeléctrico, mas nenhum revelou resultados interessantes do ponto de vista prático [25].

Devido à reduzida disponibilidade de microfones, a sua escolha foi muito limitada. A maioria dos microfones não tem qualquer tipo de amplificação. Nos testes efectuados com alguns circuitos de amplificação [26], principalmente para um microfone dinâmico de condensador *phanton power,* não se obtiveram resultados fiáveis. Optou-se, finalmente, por um microfone auto-alimentado e amplificado. Em anexo apêndice E é possível encontrar os esquemas electrónicos testados.

3.2 Estrutura da Mesa

3.2.1 Projecto

Ao longo da construção, alguns pormenores tiverem de ser alterados ou corrigidos.

Depois da fase de investigação, os sensores seleccionados para este projecto foram um piezoeléctrico (sensor de pressão), um microfone, um sensor de temperatura e um LDR, pois são os que se revelaram mais adequados ao tipo de interacção pretendida.

Os métodos de interacção com a mesa foram revistos, de forma a reduzir ao mínimo a interacção com o teclado do computador. Apesar do rato poder colocar-se ao lado da mesa e ter uma interacção rápida com o sistema, o seu uso foi também minimizado. As interacções dedicadas ao rato foram: iniciar e fechar o sistema e mudar a fonte dos parâmetros do efeito, prevendo-se pouca frequência nestas interacções.

Para uma interacção mais rápida e simples com o *software,* foi adicionado um botão de dois estados. Este botão (cor-de-rosa na figura 3.17) serve para comutar entre o modo de visualização e o modo de descrição rápida dos efeitos.

A disposição do *hardware* foi pensada com o objectivo de se obter uma utilização mais eficiente dos sensores. Estes foram colocados na parte superior, ao lado do ecrã LCD, de forma a não variarem a leitura quando não se pretende. Deste modo, qualquer gesto para manipular a Chameleon Mixer não vai interferir de um modo intencional com outros

sensores. Teve-se o cuidado de não misturar sensores que pudessem interferir com o funcionamento do sistema; p. ex., o sensor de pressão não foi colocado junto do LDR, evitando assim que qualquer gesto para manipular o sensor de pressão interfira com o LDR, e vice-versa. Os sensores com a maior utilização manual foram colocados exactamente em extremos da mesa: o sensor de pressão do lado esquerdo e o LDR do lado direito. O microfone e o sensor de temperatura ficaram no interior, pois não necessitam de ser manipulados directamente, podendo ficar mais perto dos outros componentes.

Apesar de todas estas mudanças, a lógica de funcionamento e o conceito original continuam a ser respeitados, como se pode observar na figura 2.8.

3.2.2 Materiais

Os materiais utilizados para a construção da mesa Chameleon Mixer foram sobretudo materiais reciclados. Apesar de conceptualmente optar-se por materiais reciclados, outras soluções de melhor qualidade seriam preferidas se não houvessem limitações orçamentais.

A maior parte dos componentes retiraram-se de uma mesa de mistura de som, a ssm-2200, foram igualmente obtidas algumas peças electrónicas de uma fotocopiadora antiga e criaram-se botões a partir de teclas de um teclado antigo.

Algum material teve de ser adquirido, como acrílico, componentes electrónicos, fio eléctrico e parafusos. Adquiriram-se também algumas ferramentas, como ferro de soldar, pistola de cola quente, *dremel*, entre outras.

O Arduino é o sistema mais adequado a este projecto, pois é versátil, fácil de utilizar e com uma grande comunidade de utilizadores.

O Arduino é um computador físico baseado numa plataforma de *hardware* livre, projectado com um microcontrolador e suportes de *inputs*/*outputs* embutidos na mesma placa, e uma linguagem de programação padrão, a qual tem origem em *Wiring*, e é essencialmente escrito em C/C++. O objectivo do Arduino é criar ferramentas acessíveis, com baixo custo, flexíveis e fáceis de usar [1]. Pode usar-se para o desenvolvimento de objectivos interactivos independentes, ou ainda para ser conectado a um computador hospedeiro. Uma placa típica de Arduino é composta por um controlador, algumas linhas de E/S digital e analógicos, uma interface série e outra por USB, de forma a ligar-se ao hospedeiro. As interfaces permitem programar o Arduíno oferecendo a possibilidade de interacção em tempo real. Por defeito não possui qualquer recurso de rede, porém é possível expandir este e outros recursos através de

shield's. A interface do hospedeiro é simples, podendo escrever-se em várias linguagens. A mais popular é o Processing, mas existem outras que podem comunicar por porta série: Max/MSP, Pure Data, SuperCollider, ActionsScript e Java.

Dentro da gama de Arduinos, os Duemilinove e o Mega são os melhores por serem os microcontroladores mais genéricos.

O Arduino Duemilinove utilizado na fase de testes teve que ser substituído pelo Arduino Mega, pois era muito limitado quanto ao número de portas digitais e analógicas que o sistema Chameleon Mixer iria necessitar. Em anexo Apêndice D, podem ver-se as principais características e diferenças entre estes dois Arduinos. As restantes versões de Arduino foram descartadas, pois têm características mais limitadas, como menor capacidade, alem de serem utilizados principalmente para situações de limitações de espaço, necessidades de comunicação sem fios ou outras situações particulares [1].

No Apêndice E, em anexo, encontra-se o esquemático de prototipagem e os esquemas electrónicos usados na construção da Chameleon Mixer.

O microfone, o piezoeléctrico e o LDR são sensores para uma interacção instantânea. O sensor de temperatura e o LDR podem estar inseridos numa interacção mais prolongada num contexto de uma instalação artística.

O cristal piezoeléctrico, quando submetido a uma pressão, gera um campo eléctrico num eixo transversal à pressão aplicada e pode ser medido como uma diferença de potencial.

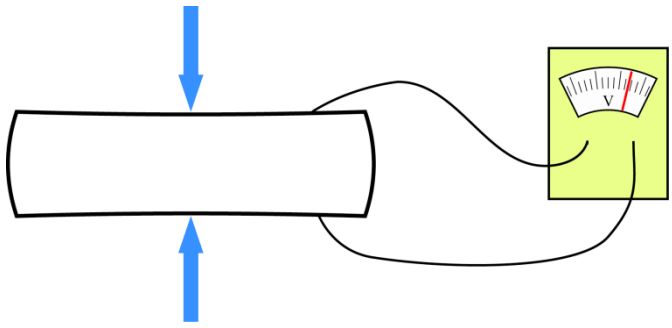


Figura 3.4 - Piezoeléctrico a ser pressionado

Ao fornecer energia eléctrica a uma cerâmica piezoeléctrica esta deforma-se. Se esta energia for sob a forma de sinal eléctrico, vai-se obter um som acústico. O efeito contrário também pode ser obtido, aplicando uma pressão na cerâmica piezoeléctrica, obtém-se uma tensão de saída como se pode ver na figura seguinte [27].

Este é o princípio utilizado para obter um sensor de pressão para a mesa Chameleon Mixer.

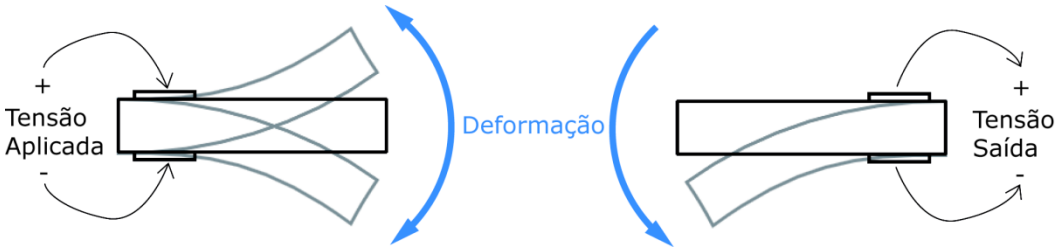


Figura 3.5 - Esquema de funcionamento de um piezoeléctrico

O microfone utilizado é um microfone dinâmico de condensador, de modelo cardióide, alimentado a 1.5V [28].

O LDR usado foi um NORPS de 320V DC com uma potência de 250 mW.

O sensor de temperatura utilizado foi um transístor de medição de temperatura de precisão LM35.

Os detalhes técnicos destes sensores podem ser encontrados no anexo, apêndice D.

3.2.3 Construção

A construção da mesa Chameleon Mixer começou pela desmontagem de uma mesa de mistura de som, a ssm-2200, que serviu de base à construção da nova mesa. Aproveitaram-se alguns componentes electrónicos como potenciómetros deslizantes, interruptores, a carcaça, entre outros.

Figura 3.6 - Mesa Realistic SSM-2200 antes de ser desmontada

A estrutura exterior desta mesa servirá de estrutura de suporte para a Chameleon Mixer, sendo efectuadas algumas alterações. Ao sistema será adicionado uma placa com o circuito principal da Chameleon Mixer. A placa metálica terá que ser completamente adaptada e a parte

superior terá que ser construída completamente de raiz, de acordo com as necessidades desta mesa.

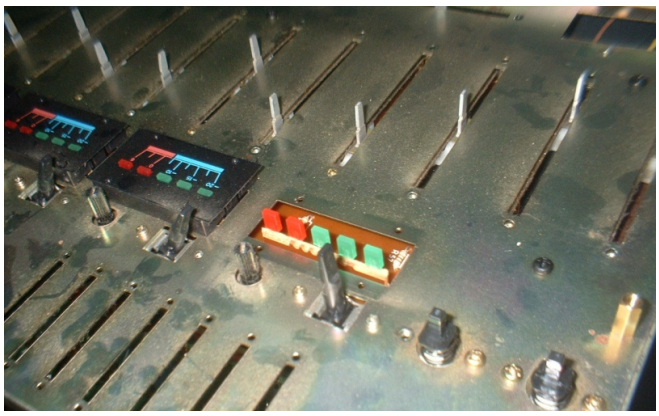


Figura 3.7 - Suporte metálico dos componentes

A mesa de som *Realistic SSM-2200* contém 12 potenciómetros deslizantes, 14 potenciómetros de estados, 4 potenciómetros rotativos, 4 botões, 1 comutador de 2 estados, 4 comutadores de 3 estados, vários *inputs* e *outputs* de áudio com saídas XLR ou RCA. Foram utilizados para a Chameleon Mixer 11 potenciómetros e 2 botões e 1 comutador.

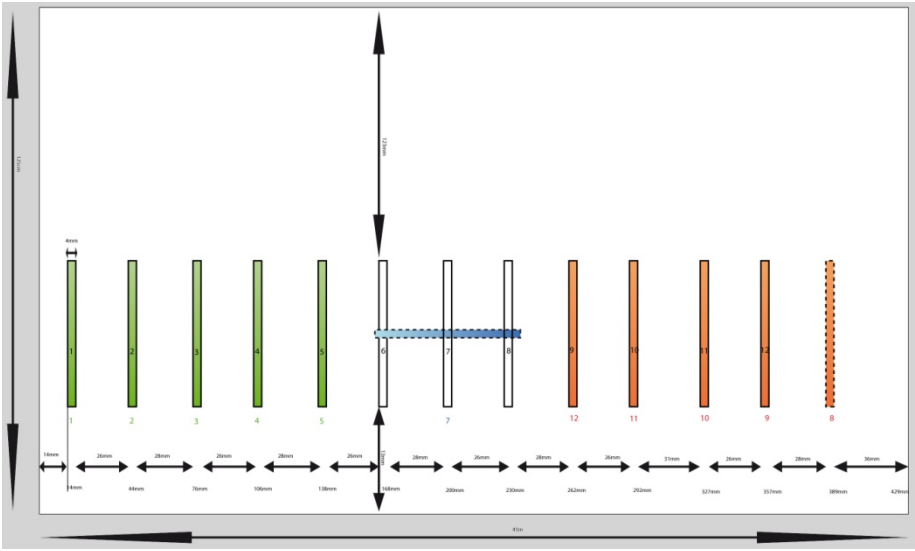


Figura 3.8 - Desenho para a recolocação dos potenciómetros na nova placa

A placa metálica de suporte aos componentes foi também reaproveitada, mas foi necessária a sua adaptação. Entre as várias alterações teve de criar-se mais uma abertura para o potenciómetro mais à direita, de forma a ganhar espaço para o potenciómetro central que fará a mistura do efeito.

No desenho anterior pode ver-se que os potenciómetros de 1 ao 5 continuam na mesma posição. No entanto, os potenciómetros do 6 ao 12 tiveram que ser alterados. Criou-se a abertura para o 13º potenciómetro, e é possível visualizar que os potenciómetros do 8 ao 12

sofreram uma rotação de 180º encaixando nas últimas aberturas, ficando com a ordem 12, 11, 10, 9 e 8.

Figura 3.9 - Furação da placa de metal com uma furadora de coluna

O teclado numérico de uma fotocopiadora foi aproveitado para servir de teclado para a Chameleon Mixer.

Figura 3.10 - Painel frontal de uma fotocopiadora

As teclas têm o dimensionamento adequado, bastando testar todos os botões e ver como está montado o circuito electrónico.

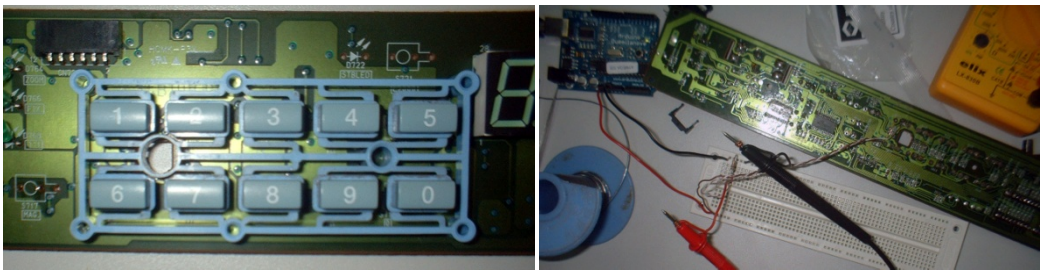


Figura 3.11 - Testar o circuito do teclado numérico

Reaproveitaram-se os botões da mesa de mistura SSM-220. Estes botões são indicados para a troca do canal A e B, no entanto, têm à partida uma desvantagem: são botões de 2 estados.

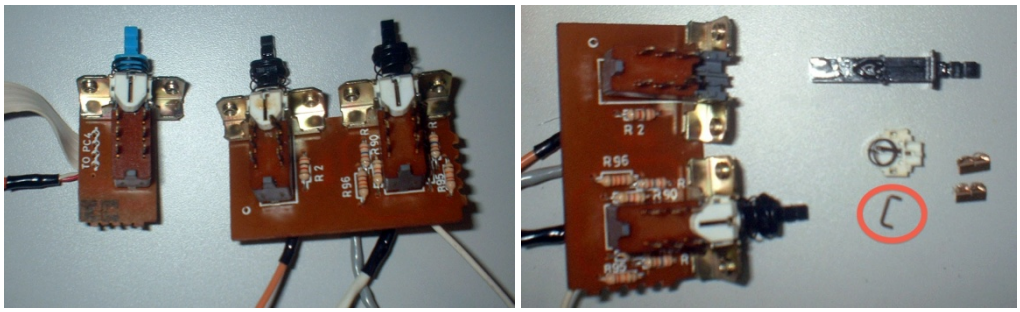


Figura 3.12 - Botões de 2 estados e desmontagem para adaptação

Estes botões foram abertos, e depois de perceber como funcionavam, retirou-se uma pequena peça que os faz comutar para os 2 estados, (assinalado na imagem anterior a vermelho) tornando-se assim botões de 1 estado, como pretendido.

Foi necessária uma placa de acrílico para a nova parte superior e para a placa do circuito da Chameleon Mixer.

Figura 3.13 - Marcação e corte das placas de acrílico

Como não estava disponível nenhuma máquina adequada para cortar acrílico, o acrílico foi cortado com um simples tico-tico. Com o corte, este derreteu, e voltou a solidificar nas arestas do corte. Os restos tiveram de ser retirados com um alicate e depois lixado.

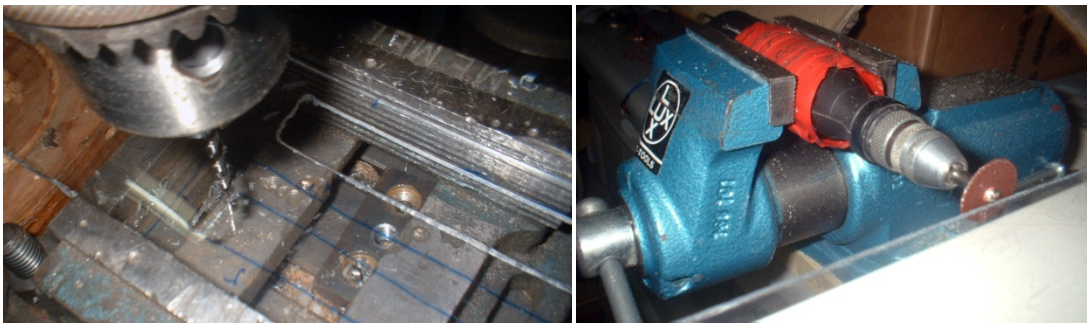


Figura 3.14 - Furar e lixar as placas de acrílico (À esq. furadora de coluna, à dir. uma dremel)

Para partes mais duras utilizou-se uma furadora de coluna para fazer o corte. Esta também foi utilizada para fazer alguns furos no acrílico.

Uma dremel foi a ferramenta preferida para furos mais pequenos ou para fazer alguns acabamentos com uma pequena lixa rotativa.

Desmontaram-se vários teclados até encontrar teclas que servissem para os botões localizados abaixo dos potenciómetros de efeitos.

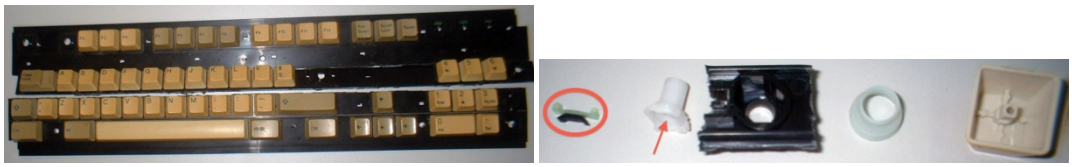


Figura 3.15 - Desmontagem do teclado final

Depois de dezenas de experiências encontrou-se uma solução: desmontar a tecla, retirar uma parte de borracha e substituir por um parafuso. Assim, tornou-se numa cápsula robusta e precisa quando colocada em cima do botão. Após fazer esta operação para todas as teclas, foi cortado acrílico, para que a tecla e o botão electrónico ficassem à distância ideal. Em cima do suporte da tecla encontra-se o acrílico da parte superior da mesa.

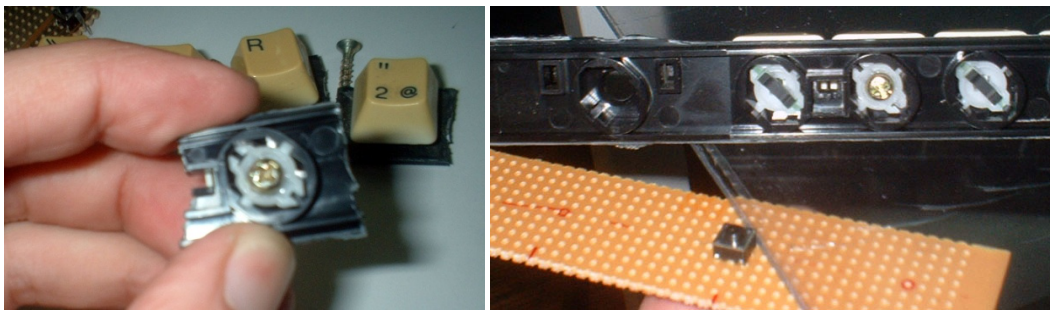


Figura 3.16 - Botões a serem remontados

Foram tiradas as medidas da estrutura metálica para se criar o esquema seguinte, que mostra todos os sensores e a sua localização na mesa. Depois da montagem dos potenciómetros deslizantes, pode-se verificar que as teclas numéricas e o ecrã LCD vão ficar no lugar pretendido. Colocou-se um botão de 2 estados ao lado do LCD para facilitar a mudança do *software* entre o modo "VIEW EFFECTS" e "PREVIEW".

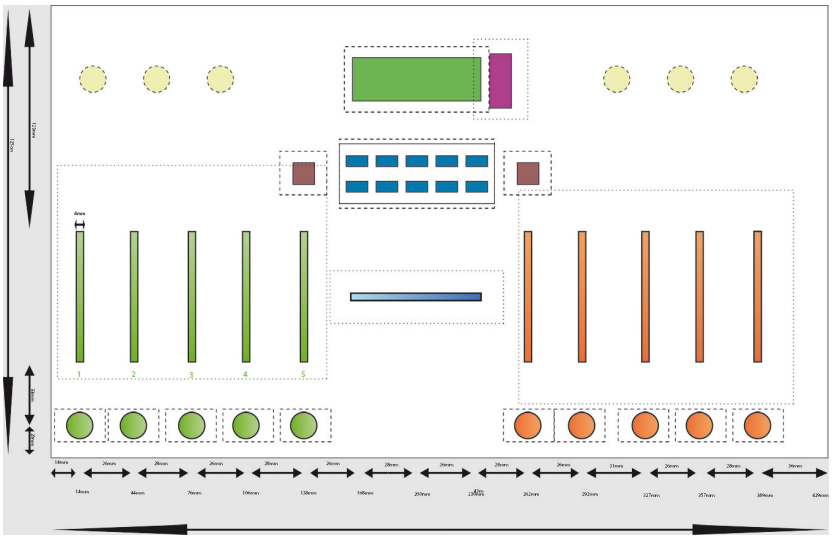


Figura 3.17 - Desenho vectorial com as medidas e disposição dos componentes na placa metálica

No corte da placa de acrílico para a colocação de componentes, acidentalmente a placa partiu-se. Devido a este contratempo, o projecto atrasou-se em alguns dias com a criação da nova placa de acrílico.

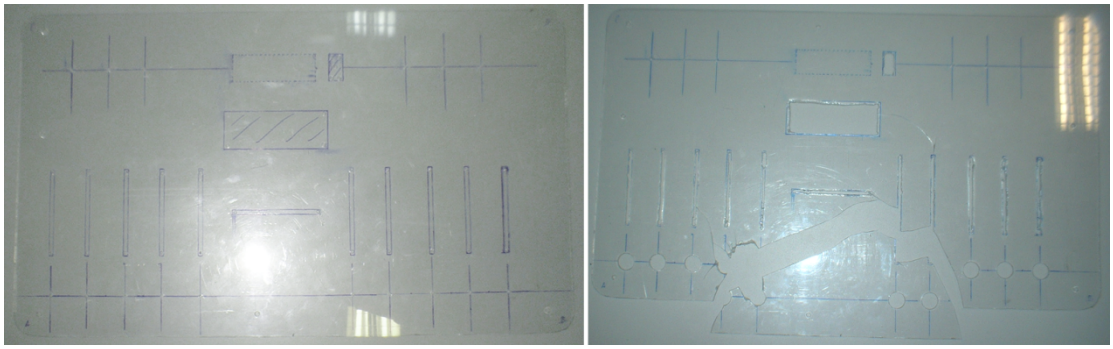


Figura 3.18 - À esquerda a nova placa superior e à direita a placa que se partiu

Esta placa teve uma técnica de corte mais aperfeiçoada, para não correr o risco de partir. Desta vez foi utilizada essencialmente a dremel para todos os cortes, furos e polimentos da placa.

Para suportar o Arduino na placa perfurada de forma reversível, criaram-se suportes em acrílico para se encaixarem e ficarem perfeitamente unidos um com o outro. Através de parafusos e porcas, o Arduino fica preso à placa perfurada que, por sua vez, fica presa ao acrílico. Num destes buracos teve-se o cuidado de colocar uma borracha para isolar a porca do parafuso dos contactos metálicos do Arduíno, prevenindo a possibilidade de existência de curto-circuito.

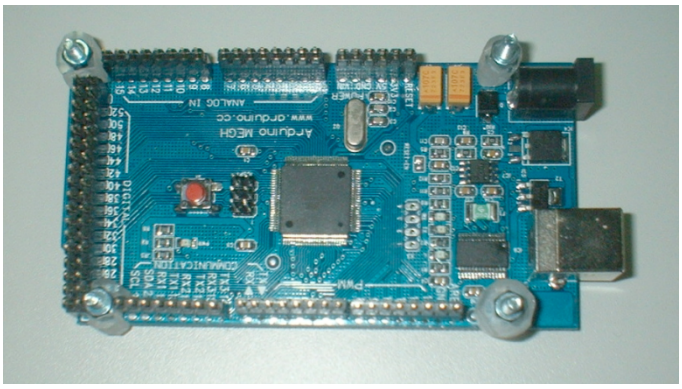


Figura 3.19 - Suportes para o Arduino encaixar na placa perfurada

De forma a facilitar a montagem dos componentes electrónicos na placa, o Arduino fica colocado com a parte da frente virada para a parte de trás da placa perfurada. Desta forma, os componentes serão colocados na parte de trás da placa e as ligações entre o Arduino e os componentes pela parte da frente, onde se encontram os contactos de soldadura.

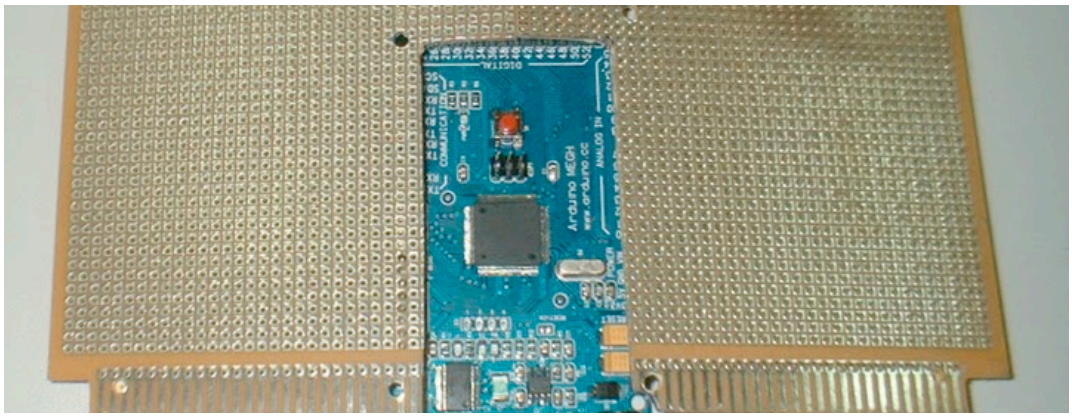


Figura 3.20 - Arduino a ser montado na placa perfurada

As ligações do Arduino para a placa fazem-se facilmente com barras de conectores, excepto um conjunto de 8 pinos do Arduino onde estão colocados os pinos PWM (do 8 ao 13), que se encontra desalinhado dos outros pinos (como consta no desenho original do Arduino mega). Este facto obrigou a um alargamento dos furos da placa perfurada para que a barra de pinos entrassem nessa zona específica, como se pode ver na imagem.

3.3 Montagem Electrónica

A montagem da parte electrónica começou com a criação do esquemático no *software* fritzing [29]. Este *software* contém praticamente todos os componentes necessários para este projecto. Um dos componentes que não está disponível é a placa perfurada que foi substituída por

breadboards. Todas as ligações foram projectadas tendo em conta o conhecimento adquirido na fase de investigação electrónica.

Posteriormente foi feito o esquemático de todas a ligações dos componentes e Arduino. Pode-se ver em anexo Apêndice E o esquema de ligação do protótipo em fritzing e todos os esquemas electrónicos.

O esquema das ligações foi feito para projectar todas as ligações e conseguir organizá-las melhor na placa perfurada, bem como entre os vários componentes. Este esquema teve que sofrer algumas alterações, até ter a decisão final de quais os sensores que iriam permanecer na mesa, que foi tomada depois do *hardware* estar praticamente concluído.
Os cabos representados a azul significam que é uma ligação digital, os amarelos, analógica e os verdes, digital, que ligam ao PWM do Arduino. O vermelho significa os 5V e o preto, o GND.

A alimentação de todos os componentes da mesa é feita pelo Arduino. Por sua vez, a alimentação do Arduino será feita através do USB, mas para tornar esta plataforma independente do computador, o Arduino poderá também ser alimentado externamente por uma fonte de alimentação de 9V.

Desta forma, é possível desligar o USB e a mesa continuar em funcionamento ou desligar o transformador e a mesa ser alimentada pelo computador, o que se torna bastante útil no caso de ser necessário utilizar a Chameleon Mixer num local sem fonte de energia.

O sistema também pode ser alimentado pelo USB e pela alimentação externa, que fornece no máximo 900mA, 500mA do USB e 400mA da alimentação externa.

Depois de projectado todo o esquemático, passou-se à fase de soldadura dos componentes.

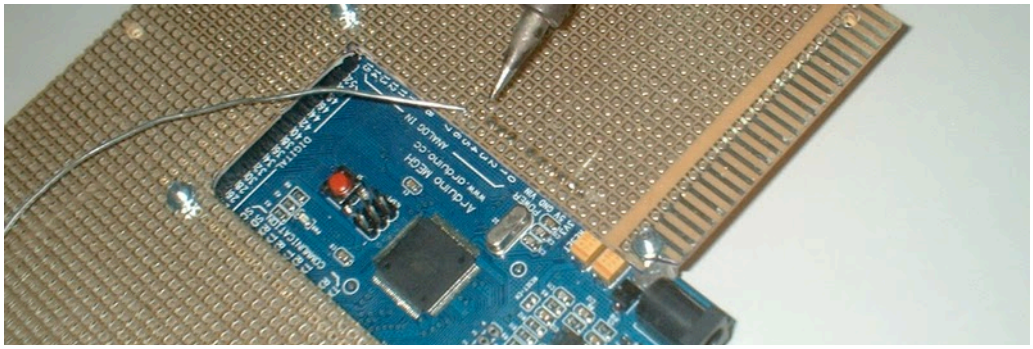


Figura 3.21 - Soldadura do Arduino à placa perfurada

Inicialmente marcou-se com uma caneta de acetato os pinos onde se iriam soldar-se os cabos. Esta marcação ajudou a assinalar a localização dos componentes e a minimizar os erros de soldadura.

Nas ligações digitais, o Arduino é composto por 2 filas de pinos. Este facto dificultou a soldadura dos fios, pois foi necessário criar pontes para a 2ª fila. Estes fios foram soldados com bastante cuidado e precisão, pois qualquer problema criado nesta fase seria um grande problema para fases posteriores.

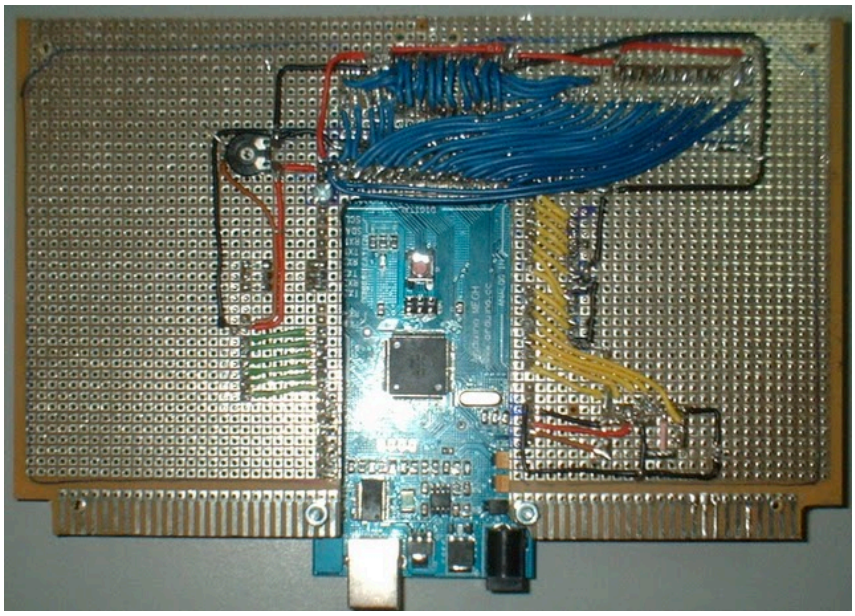


Figura 3.22 - Placa com as soldagens concluídas

A cor dos fios foi escolhida de acordo com o projecto criado no fritzing, tornando-se mais fácil ver as ligações físicas e comparar com as projectadas no *software*.

Foi necessário escolher os conectores de pinos que minimizassem a quantidade e percurso do *flat cable* entre os componentes e a placa perfurada.

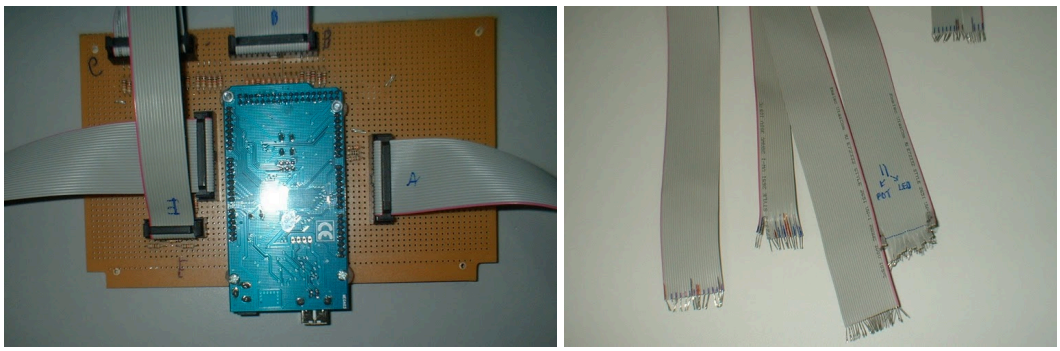


Figura 3.23 - *Flat Cable* na placa perfurada

O acrílico onde vai colocar-se a placa perfurada foi projectado para que os cabos possam desligar-se com facilidade a qualquer momento e sem erros de polaridade. Esta característica permite que a placa de acrílico seja retirada da mesa para eventuais *upgrades* a nível de *hardware*.

Os fios foram agrupados em cinco grupos:

A - Ligações ao LCD, e LEDs RGB

B - Ligações Botões ON/OFF dos *mixers* e aos respectivos LEDs

C - Ligações a Botões do Teclado Numérico, Botões de selecção Canal A/B, Botão de selecção Visualização/Efeitos

D - Ligações aos potenciómetros de todos os canais e o *fader* entre canal A e B

E - Ligação a todos os sensores

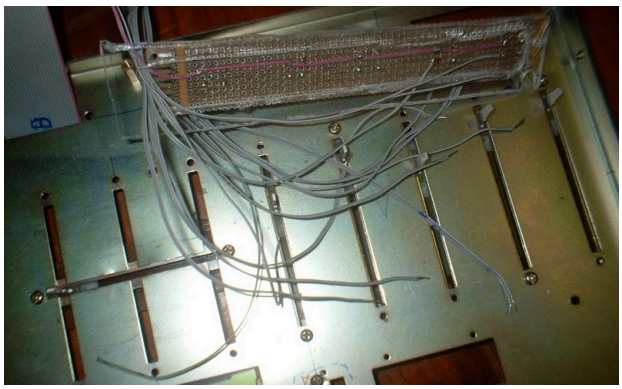


Figura 3.24 - Ligações dos fios que ligam os botões dos *mixers* aos LEDs

Os botões que ligam e desligam os *mixers* foram montados sobre pequenas placas perfuradas, de forma a serem colocados com precisão e a oferecerem uma plataforma de contacto de soldadura entre os fios e os botões.

Também se colocaram neste suporte os fios de ligação aos LEDs dos *mixers*.

Figura 3.25 - Superfície de espuma preta para não reflectir a luz dos outros LEDs

Para a luz que ilumina o canal de cada *mixer* não ser reflectida pela superfície metálica ou para os canais vizinhos, introduziu-se uma espuma com cerca de 2mm, preta e opaca, que cobre toda a superfície dos potenciómetros, como é visível na figura anterior.

Posteriormente foi criada uma estrutura mais forte, em acrílico, que oferece a rigidez necessária para a colocação desta espuma entre os canais dos LEDs.

Figura 3.26 - Estrutura e soldadura dos fios aos LEDs dos *mixers*

Todos os LEDs foram montados em pequenos furos feitos na placa de acrílico e ligados de um lado ao outro através de fios que atravessam sempre a parte exterior aos canais.

Desta forma, aplica-se apenas um fio *ground* oriundo da placa principal e um fio de dados digitais do Arduino, ficando todos estes contactos somente num lado deste suporte.

No final, esta estrutura foi encaixada na placa metálica, tornando-se removível, caso seja necessário. Por fim, soldaram-se os fios entre esta estrutura e as placas perfurados dos botões dos canais.

A montagem do circuito dos LEDs foi feita em ligação paralela. Os cálculos encontram-se em anexo Apêndice D.

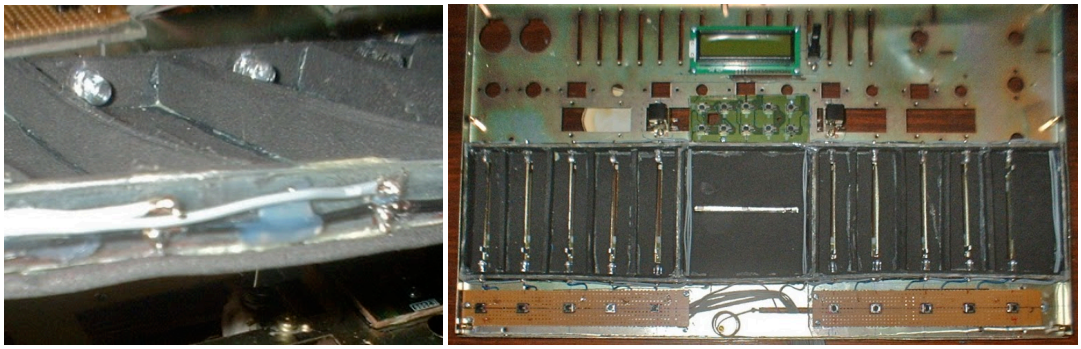


Figura 3.27 - Instalação da placa dos LEDs na placa metálica

Todos os restantes cabos foram directamente soldados aos respectivos componentes, colocando-se no final cola quente para fixar os condutores em pontos estratégicos. Os

próprios cabos foram também presos à placa metálica para que qualquer força exercida sobre eles não danifique as ligações aos componentes.

Figura 3.28 - Instalação dos sensores na placa acrílica

Foram criadas estruturas em acrílico fundamentalmente com a ferramenta *dremel*. Estas estruturas são cubos esculpidos de forma a que os sensores ficassem com uma fixação estável, tal como as suas soldaduras.

Do lado direito, montaram-se os sensores de temperatura e luminosidade (LDR). Do lado esquerdo, o microfone e sensor de pressão (piezoeléctrico).

Todo este processo de soldadura foi iterativo e com bastantes contratempos. Foram vários os componentes que tiveram que voltar a ser soldados devido às limitações de entradas do Arduino e, por uma questão de optimização, tiveram que ser dessoldados e voltar a ser soldados noutra entrada, nomeadamente os fios do teclado numérico e os fios dos LEDs RGB.

O ecrã LCD utilizado é um ecrã de 2 linhas com 16 caracteres em cada linha. Também teve que sofrer uma mudança que obrigou a voltar ao processo de soldadura de todos os fios, devido a limitações de portas analógicas. Optou-se por uma forma de conectar o LCD apenas com portas digitais (PWM) [30] [31], apesar de serem necessárias mais ligações em relação a outros esquemáticos [32]. O esquema utilizado pode ser consultado em anexo Apêndice E.

4 Firmware

A comunicação entre o microcontrolador Arduino e o *software* Processing é feita através do chip FTDI que se encontra no Arduíno [33] e os *drivers* responsáveis por esta comunicação no computador.

De forma resumida, o código necessário para o funcionamento do Arduino é composto por dois métodos, o setup() e o loop(), e a inicialização de variáveis que deve fazer-se antes de se chamarem estes métodos. O setup() é o primeiro método a ser chamado na aplicação e onde devem colocar-se todas as inicializações e o loop() é o método que se chama repetidamente e onde deve colocar-se o código para verificar se existe um novo valor de controlo, enviar informação para o computador, enviar um sinal a algum pino ou enviar informação de *debug* [20].

Veremos neste capítulo como é feita a comunicação e o *software* criado para este efeito.

4.1 Comunicação Arduino

4.1.1 Série

Série significa que é um após o outro. A transferência de dados em série é uma transferência de um bit de cada vez, um após o outro. Assim, passam-se as informações entre o Arduino e o *software* utilizado, neste caso o Processing.

A transmissão de dados entre o Arduino Mega e o Processing é feita através de um *chip* adaptador USB – Série, o FTDI FT232.

Quando se efectua uma comunicação série, o Arduino indica essa comunicação com dois pequenos LEDs, o TX e o RX. O TX significa que está a efectuar uma transmissão de dados e o RX significa que está a efectuar uma recepção de dados.

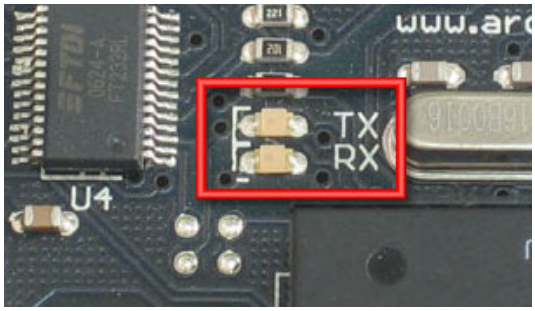


Figura 4.1 - LEDs de comunicação de dados do Arduino

A taxa de transmissão de dados pode programar-se facilmente no Arduino. A seguir, podemos ver o código para uma taxa de transmissão de 9600 bits por segundo (bps) que deve colocar-se dentro da função setup().

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

O método "begin" é chamado da biblioteca "Serial".

4.1.2 Firmata

Firmata é um protocolo genérico de comunicação entre o *hardware* e o *software* do computador. Basicamente, este *firmware* estabelece um protocolo para comunicação entre o Arduino e o *software* hospedeiro [34]. O objectivo é trabalhar com o Arduino a partir de um computador remoto, utilizando qualquer pacote de *software*.

Desta forma, os utilizadores do Chameleon Mixer podem controlar, em tempo real, o Arduino através do Processing.

Inicialmente na fase de testes, instalou-se o protocolo Firmata no Arduino Duemilinove onde todas as funções operam correctamente, sendo possível o seu controlo em tempo real através do Processing.

No caso do Arduino Mega, o protocolo Firmata falhou, pois a arquitectura é diferente, sendo necessário procurar outras alternativas. Foram encontrados outros protocolos para sistemas, como o Maxduino[35]. No entanto, as únicas opções viáveis encontradas para o Arduino Mega passavam por criar um novo protocolo com base na comunicação série, ou então adaptar o protocolo do Arduino Duemilinove para o Arduino Mega. Decidiu-se, então, criar um novo protocolo tendo como base o Firmata. Este processo revelou-se bastante moroso, pois esta adaptação não se revelou trivial; foi necessário estudar a arquitectura de comunicação do Arduino Mega em profundidade. A inexistência de documentação deste protocolo foi o principal problema, sendo necessário recorrer a protocolos mais antigos para perceber o seu funcionamento.

Devido ao uso de potenciómetros logarítmicos, foi necessário aplicar uma função exponencial para tornar a sua variação linear. Foi estimada a melhor função exponencial a ser aplicada, que é:

$$y = a * b^x$$
$$a = 4.527$$
$$b = 1.0053$$
$$y = 4.527 * 1.0053^x - 5.5$$

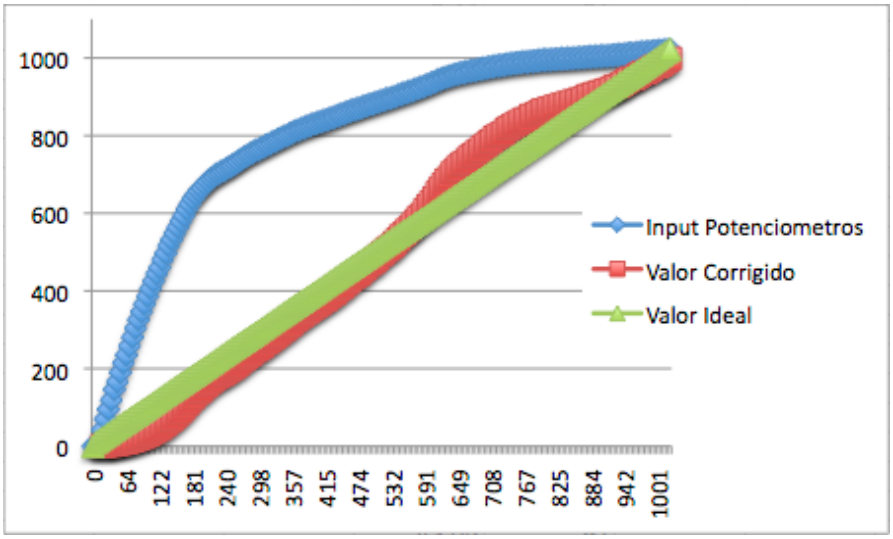


Figura 4.2 - Gráfico de Correcção dos Inputs dos potenciómetros

Na figura 3.30, a cor azul indica os valores de *input* dos potenciómetros, a verde o valor ideal de correcção e a vermelha a correcção efectuada pela função exponencial, com valores mais próximos dos ideais.

Para simplificar todos os cálculos, a função exponencial foi aperfeiçoada para converter a escala de 0-1024 para 0-1000, tal como se observa na figura anterior.

Foram aplicados, adicionalmente, métodos de interpolação linear e um método de correspondência de ponto a ponto, mas o processamento adicional para estes métodos não justificou a sua aplicação.

O Arduino Duemilinove tem apenas 16KB de memória disponíveis para o *firmware*. Deste modo, foi necessário ter em conta o tamanho do *firmware* a desenvolver, ficando a ocupar apenas 13 Kbytes. Na passagem para o Arduino Mega não surgiu este problema, pois tem 128Kbytes de memória disponíveis.

É necessário ter em atenção os tipos de variáveis que se usam dado que o *debugging* da API do Arduino não prevê problemas como o *overflows* de variáveis [36].

A velocidade de processamento é limitada pela velocidade do relógio do *Arduino*, o que se reflectiu nas funções em *loop*. Deste modo, as funções que correm concomitantemente foram cuidadosamente construídas de forma a evitar atrasos indesejados.

Neste trabalho tirou-se partido de todo o processamento possível do Arduino, tornando este sistema independente da plataforma de *software* usada, podendo ser facilmente aplicado noutros contextos.

Devido ao facto da mesa possuir alimentação externa, pode desconectar-se a porta USB do computador e voltar a conectar-se, que continuará a aplicar o mesmo efeito, devido à modularidade do sistema.

A impressão para o LCD do efeito seleccionado também se faz dentro do Arduino, através da biblioteca LiquidCrystal que faz parte do *core* do Arduino.

O *firmware* também é responsável por activar ou desactivar cada canal do efeito. A activação realiza-se quando se carrega no botão por baixo do canal, acendendo o LED. Quando se volta a carregar, o estado do botão fica desligado e o LED também é desligado.

Ao nível dos *inputs* analógicos, foi relativamente fácil colocar a funcionar a comunicação.

O pseudo-código do funcionamento do Arduino para a comunicação analógica e *inputs* digitais encontra-se em Anexo Apêndice F.

Na comunicação digital, a etapa de envio de dados de saída dos pinos digitais é mais complexa. Como o Arduino Mega tem mais portas digitais que o Arduino Duemilinove, o funcionamento dos *outputs* digitais é diferente. Foi necessário compreender o esquemático do Arduino Mega, que se encontra em Anexo Apêndice E, e a tabela dos registos dos portos do Arduino Mega, em Anexo Apêndice D.

Cada *port register* é responsável por enviar um sinal de um byte (8 bits), e portanto é responsável pelo estado de ON/OFF de 8 portas digitais do Arduino.

Enquanto o Arduino Duemilinove tem apenas 3 *port register* (A, B e C) o Mega tem 12 (do A ao L), mas nem todas as portas estão disponíveis e algumas têm funções específicas, como a comunicação de sinal em PWM.

De forma exemplificativa, o *input* dos pinos 35 e 37 (correspondentes a dois LEDs) e os pinos 36 e 34 (correspondente a duas teclas do teclado numérico) foram activados no *port register* C com a descrição de *input* "IN": PINC, utilizando o bit 0. A numeração é feita da direita para a esquerda.

outputPort(5, PINC &~ B11110000); //(2/0) pinos 35 37 | (3/1) pinos 36 34

Depois de muitas tentativas frustradas, percebeu-se que as portas da função outputPort() têm que estar por ordem decrescente, senão os *outputs* "encravam" na primeira iteração.

Em anexo Apêndice C encontra-se a descrição de todos os pinos existentes no Arduíno, assim como a sua funcionalidade na Chameleon Mixer.

É possível ver mais informação sobre os *port registers* relativamente ao Arduino Duemilinove no site do Arduino [37].

O pseudo-código do funcionamento do Arduino relativamente ao temporizador para os botões de selecção, encontra-se em Anexo Apêndice F.
Como o Arduino tem apenas 16 portas analógicas e 11 estão a ser ocupadas com os potenciómetros, restam 5 portas para os sensores e para a passagem da informação dos efeitos do canal A e B.

De forma a optimizar recursos, 4 portas analógicas foram dedicadas a sensores e a restante porta a passar a informação do efeito seleccionado dos canais A e B. Tendo o conhecimento de que cada porta analógica pode passar 1024 valores diferentes, optou-se por utilizar uma gama de 1000 efeitos (entre 0 a 999). Nesta optimização apenas um dos canais pode estar seleccionado, tendo uma porta digital complementar para dar a informação de que canal (A ou B) está seleccionado no momento.

O pseudo-código do funcionamento do Arduino relativamente à criação do efeito com o teclado numérico encontra-se em Anexo Apêndice F.
Para se inserir um efeito num canal é necessário carregar na tecla do efeito A ou B, de forma a seleccionar o canal pretendido, e de seguida seleccionar um efeito até 3 dígitos (0 - 999). Caso seja inserido um 4º algarismo, o número passa a ser composto por apenas um algarismo e cada vez que for pressionada uma tecla numérica adiciona-se um caractere até um máximo de 3. Desta forma, é possível, em tempo real, seleccionar um efeito diferente de uma forma simples e rápida. Este paradigma é especialmente útil, no caso de engano.

Quando passa um determinado tempo a partir da última tecla premida, o efeito é bloqueado e é comunicado o seu estado de “activo” ao sistema.

Para colocar um novo efeito nesse canal basta carregar na tecla de efeito respectiva.

5 Software

Neste capítulo são descritas as várias etapas de desenvolvimento de *software* e as dificuldades encontradas na integração de algumas tecnologias, bem como a sua resolução.

5.1 Investigação de Linguagens

Inicialmente foram investigadas várias linguagens livres e multiplataforma para a programação da aplicação.

A linguagem que se pensou, à partida, foi o C++, por ser uma linguagem bastante familiar e completa. No entanto, por ser uma linguagem com um desenvolvimento moroso, foram procuradas soluções alternativas.

Para este projecto procuraram-se linguagens direccionadas para a visualização e interacção, que tivessem suporte de comunicação com o Arduino. Encontraram-se duas possibilidades viáveis a este desenvolvimento, o OpenFrameworks e o Processing.

Analisou-se uma Framework, o OpenFrameworks [3] que é uma plataforma para computação criativa escrita em C++ que corre em Windows, Mac OS X e Linux. É actualmente usada por programadores *designers* e artistas, para criar conteúdo interactivo e visual. Compreende uma série de ferramentas e utilitários que ajudam no processo de desenvolvimento de projectos e ideias.

O Processing [2] é uma nova ferramenta desenvolvida para que programadores da área de *design* e artes possam desenvolver aplicações num contexto visual. Trata-se de uma linguagem simplificada, construída sobre Java, que a torna limitada em alguns aspectos, especialmente em relação à velocidade de processamento. [38][39] Por ser uma linguagem bastante acessível foi a preferida para criar a aplicação.

5.2 Desenvolvimento da Aplicação

5.2.1 Processing

No desenvolvimento da aplicação teve que se criar a comunicação com o Arduino através do protocolo Firmata. Como se pode ver na figura seguinte, o Processing recebe todas as comunicações enviadas pelo Arduino. Os pinos digitais estão representados como quadrados que ficam preenchidos quando o valor daquele pino está a 1, e vazio quando esta a 0. As

comunicações analógicas estão representadas como círculos, que ficam com um raio maior quando o valor recebido é maior, e com um raio 0 quando o valor é 0.

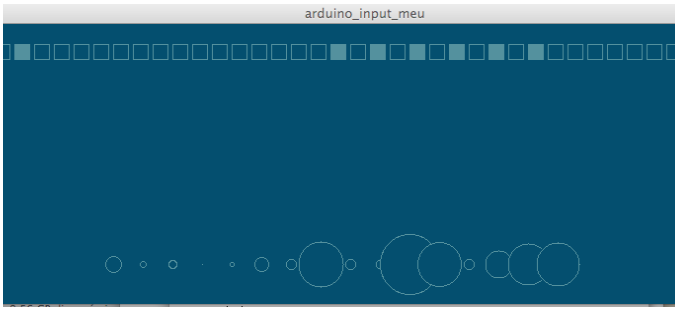


Figura 5.1 - Inputs Digitais e Analógicos visualizados no Processing

Após ter a comunicação de todos os componentes a funcionar, passou-se ao teste das câmaras.

Foram testadas 2 câmaras em simultâneo e a aplicar um efeito de cor simples. O Processing processa as imagens das duas câmaras e aplica o efeito de cor visível na imagem.

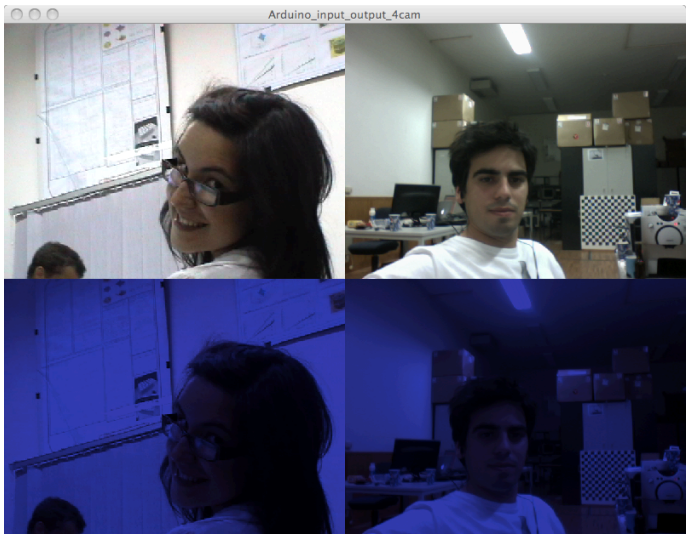


Figura 5.2 - Processing com duas câmaras em simultâneo e aplicado um efeito

Foram procuradas formas de criar interfaces gráficas. Existem algumas bibliotecas para Processing de elementos gráficos, a Interfascia [40], Quarck [41] e controlP5 [42]. Esta última é a que responde melhor aos requisitos da interface da Chameleon Mixer. Trata-se da única com o elemento gráfico *slider*, e com um aspecto gráfico melhor que as outras bibliotecas.

Nesta interface utilizaram-se os elementos controlP5.addSlider Horizontal e Vertical, controlP5.addNumberbox, controlP5.addListBox. Os rectângulos e texto foram feitos com as primitivas gráficas e texto do Processing.

Os elementos gráficos foram criados com valores relativos, bastando mudar o tamanho geral da janela para todos os elementos serem recalculados e reposicionados na janela.

A interface gráfica construída pode na seguinte figura:

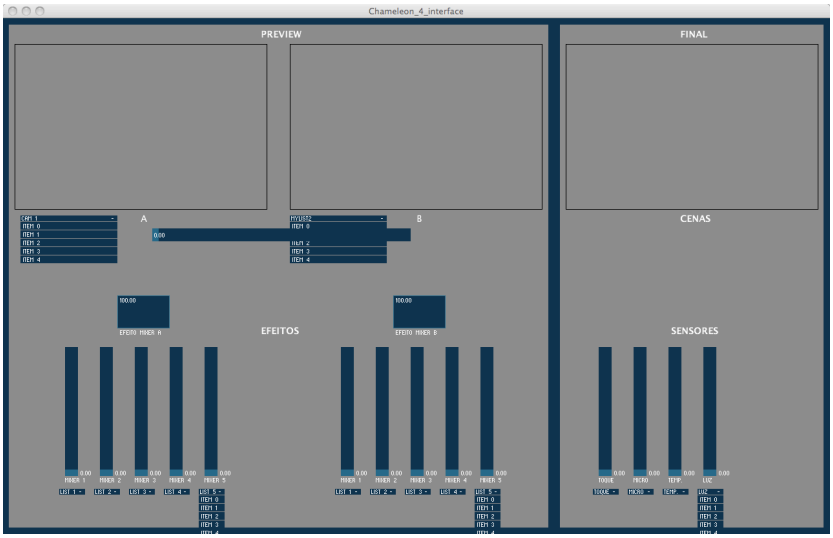


Figura 5.3 - Interface gráfica em controlP5 no Processing

A partir do momento em que foram colocados vídeos a serem processados com algum efeito a aplicação começou a ficar muito lenta, chegando aos 2 frames por segundo.

Nesta altura, teve que ser pensada uma alternativa e a solução mais viável foi a mudança para o OpenFrameworks.

5.2.2 Openframeworks (C++)

Devido ao protocolo Firmata já estar a funcionar bem com o Arduino através do Processing e, por uma questão de optimização e de divisão de recursos, decidiu-se continuar a utilizar a comunicação com o Processing. O Processing reenvia através do protocolo de comunicação OpenSoundControl (OSC) [43] todos os valores para o OpenFrameworks.

O OSC é um protocolo de comunicação entre computadores, sintetizadores de som e outros dispositivos multimédia que é optimizado para a tecnologia de rede *tcp/ip*. Revela-se um protocolo simples, mas poderoso que fornece tudo o que é necessário para o controlo em tempo real de controladores, sendo flexível e fácil de implementar. A biblioteca utilizada no Processing para o OSC foi a oscP5 [44].

Desta forma, é possível estabelecer comunicação com um IP ou fazer *broadcast* e enviar para toda a rede os dados da Chameleon Mixer. Isto permite que a mesa Chameleon Mixer possa controlar não só a aplicação da Chameleon Mixer como outros sistemas. O contrário também é verdadeiro, podendo a aplicação receber comunicações de outros sistemas. Isto será útil caso se pretenda adicionar sensores para além dos 4 sensores existentes na mesa. Pensando

num caso específico, numa instalação artística, isto pode ser bastante útil. Podem estar vários sensores (p. ex. de proximidade) ao longo da instalação e enviar essa informação, através dos protocolos de rede *tcp/ip*, para o computador que efectua o processamento de imagem. Com esses dados, pode produzir-se informação visual que diga, p. ex., onde se encontram as pessoas naquele instante e as mostre visualmente.

É igualmente possível controlar toda a aplicação remotamente. Encontra-se desenvolvida com sucesso esta comunicação entre sistemas através do protocolo OSC. Esta funcionalidade abre claramente portas a novas aplicações da Chameleon Mixer.

São muitos os programas de computador que estão a adoptar este protocolo, inclusivamente os de *VJing* como o Resolume [14]. Ver [43] para mais implementações de OSC.

Com estas modificações apresenta-se um novo diagrama conceptual funcional da Chameleon Mixer que complementará o Diagrama da Figura 2.8. Este diagrama poderá ser encontrado com uma escala maior em anexo, no apêndice D.

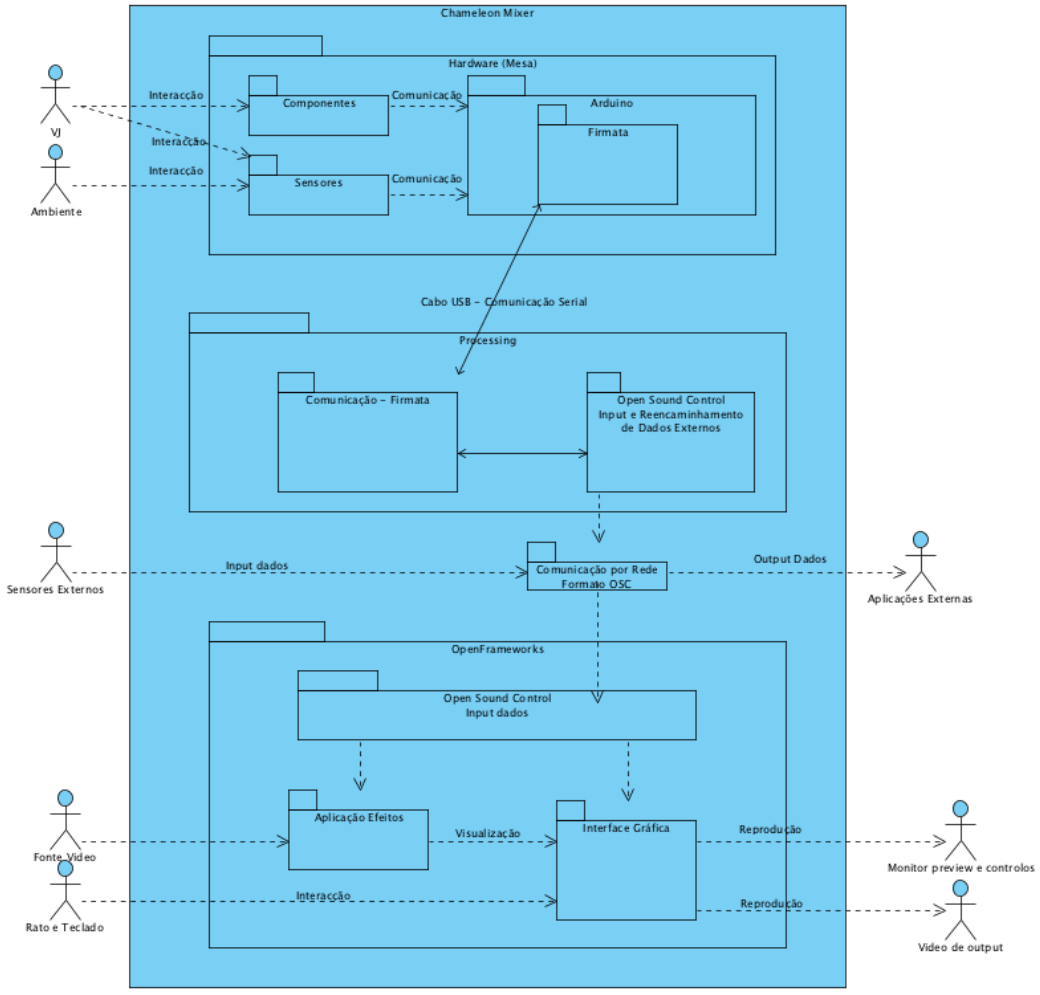


Figura 5.4 - Actualização do Diagrama Conceptual Funcional do Chameleon Mixer

Como o OpenFrameworks é baseado em C++, torna-se possível utilizar qualquer biblioteca para a interface gráfica que funcione em C++.

Inicialmente, tentou-se usar a aplicação Argos [45], que é um construtor *drag-and-drop* (como se pode ver na imagem a seguir) de GUI com os elementos necessários. Apesar de ser uma aplicação promissora, por ainda não ter documentação praticamente nenhuma e neste momento apenas ter a integração em ambiente Windows, foi necessário procurar outra solução.

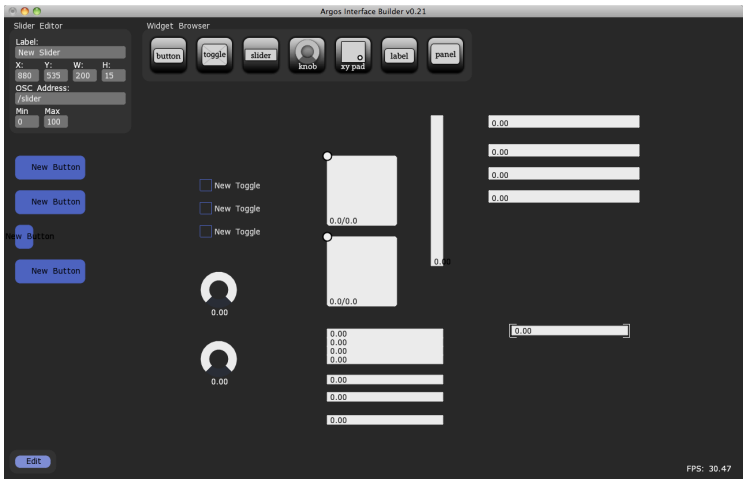


Figura 5.5 - Criação de Interface no Argos

Surgiu a ideia de colocar o Processing a funcionar com a interface gráfica e o OpenFrameworks a processar o vídeo, mas tornou-se inexequível a junção das duas plataformas. Outras soluções que surgiram basearam-se na utilização Qt[1], mas é uma Framework que juntamente com o *OpenFrameworks* revela muitos erros, devido a conflitos entre as duas *frameworks*.

Foram procuradas outras bibliotecas gráficas de baixo nível, que permitam alterar, de forma simples, os elementos gráficos que se adaptem às necessidades da Chameleon Mixer. Encontraram-se o *Clutter* [46], o *GLV*[47], o *GTK+* [48], e o *ofxGUI* [49].

O *Clutter* e *GTK+* não se adequam graficamente ao projecto e são bibliotecas que não são fáceis de usar e modificar tendo em conta a limitação temporal. O GLV e o ofxGUI adequam-se graficamente ao que se pretende e permitem, de uma forma exequível, fazer alterações ao código.

Inicialmente usou-se o GLV, mas por fim passou-se para o ofxGUI por ser mais intuitivo. Como o GLV é bastante recente, ainda não tem muita documentação e exemplos, enquanto o

[1] Qt é um Framework multiplataforma para desenvolvimento de interface gráfica

ofxGUI é uma biblioteca já com alguns exemplos disponíveis como o CCV [50] (*software* para multi-toque).

Para colocar uma segunda janela a funcionar e comunicar com a primeira, utilizou-se a biblioteca ofxFenster [51]. O ofxFenster é um *addon* para aplicações em OpenFrameworks que cria e gere uma segunda janela. Devido aos problemas que surgiram a desenhar objectos do ofxGUI depois do ofxFenster ser chamado, foi decidido colocar os controlos e o *output* na mesma janela, desenhando o *output* com as coordenadas XY (largura do monitor actual, 0). Quando colocada a janela em *fullscreen* o resultado é o desejável, sendo uma melhor opção, ao evitar perder-se *performance* com a comunicação entre as duas janelas.

Surgiram vários problemas na compilação do GUI devido à inconsistência inicial entre a informação contida no código XML e no código C++. O ficheiro XML é criado automaticamente, caso não exista, quando se compila a aplicação com os valores definidos como *standard*. Depois de compilada a aplicação, é possível alterar muitas variáveis, como a cor ou o tamanho de cada objecto através da alteração do ficheiro XML. Por defeito, esta biblioteca vem com os objectos em várias cores, e no ficheiro XML foram colocados todos com a cor verde de forma a ficarem congruentes com a estética da mesa.

Ao nível da própria biblioteca do *ofxGUI,* fizeram-se alguns melhoramentos. Foi colocada uma variável que pode definir-se no código XML como "left" que indica a distância a que o objecto vai ser desenhado à esquerda do painel onde está inserido. Outra característica que foi implementada reside na possibilidade de desenhar *sliders* verticais. Por defeito, esta biblioteca vem apenas com a possibilidade de criar *sliders* horizontais. Esta alteração faz-se trocando os parâmetros do *slider* (largura, altura) no caso de verificar-se que altura > largura.

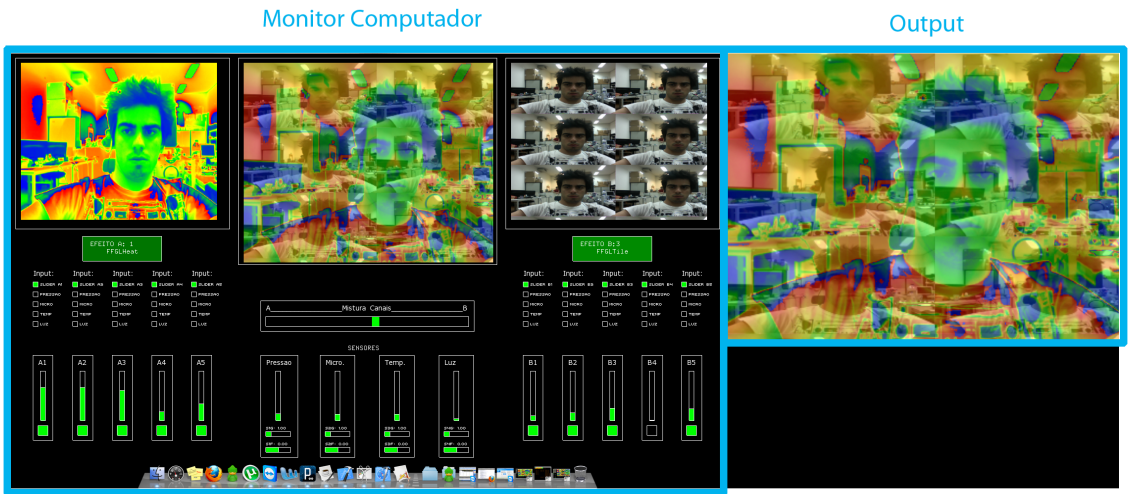


Figura 5.6 - Aspecto da aplicação Chameleon Mixer

Como se pode ver na figura anterior, foram colocados os seguintes elementos do *ofxGUI*: texto em cada painel, os *sliders* verticais para representar os potenciómetros e sensores, *sliders* horizontais para os ajustes dos sensores, uma barra de mistura entre o canal A e B e seleccionador para a fonte do parâmetro.

Quando a aplicação recebe a informação de que foi seleccionado o Canal A ou B, a área de selecção do efeito fica a piscar entre um verde com mais intensidade e o preto, enquanto não passar o tempo de selecção. Quando passa o tempo de selecção, a cor do *mixer* seleccionado fixa e o efeito seleccionado é activado.

Para o utilizador ter o *feedback* do efeito seleccionado, o nome do efeito correspondente ao número que está a ser marcado, aparece no ecrã. Deste modo, se o utilizador vir que o efeito que acabou de marcar não é o pretendido, pode continuar a marcar algarismos até ter o efeito que pretende. Caso não exista nenhum efeito correspondente ao número seleccionado não aparece nenhum nome.

O nome do efeito é automaticamente retirado do nome do ficheiro do efeito. É utilizada a biblioteca *ofxDirList*, umas das bibliotecas do core do *OpenFrameworks,* [52] para obter o nome dos ficheiros. Como os ficheiros têm que obedecer ao nome "FFGL" + Nome do Efeito + ".bundle" facilmente se retira o nome do efeito.

O efeito só é carregado no sistema a partir do momento em que ele é chamado. De modo explicativo, caso seja pressionado o botão 1, e de seguida o 2, até fixar o efeito 12, vai ser carregado o efeito com o número 1 e o efeito com o número 12. Assim, consegue-se que os efeitos sejam carregados antes de poderem ficar activos. Garante-se ainda que nenhum efeito é carregado quando já está activo (no 'ar').

Existem várias formas de protecção de dados. No caso do nome dos efeitos, caso este tenha um nome maior que 16 dígitos, apenas são mostrados no *software* os primeiros 16 algarismos. No caso dos sensores, caso exista algum valor maior que 1 ou menor que 0, esses valores são descartados.

A aplicação de 1 em 1 segundo verifica se existem novos efeitos na pasta respectiva ("/data") e caso existam, esses efeitos são registados no sistema, tal como os seus nomes.

Inicialmente foi colocada uma textura como fundo da aplicação, mas constatou-se que a aplicação perdia *performance*, portanto foi retirada.

Na imagem seguinte pode-se ver as bibliotecas de C++ do *core* do OpenFrameworks, bem como as bibliotecas externas integradas no sistema.

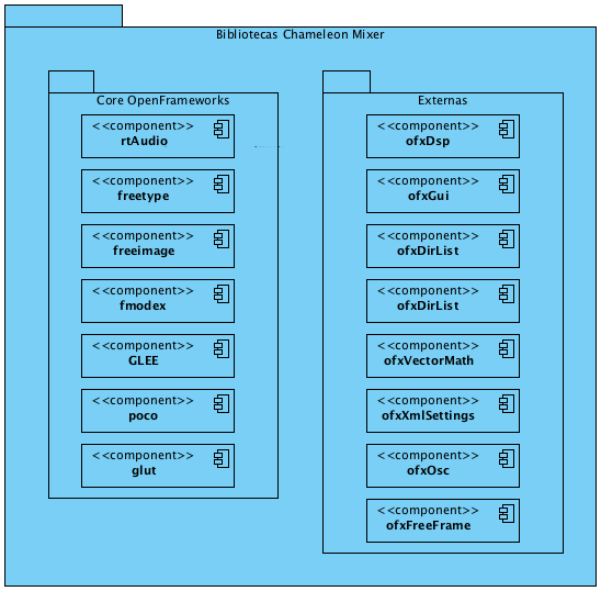


Figura 5.7 - Diagrama de bibliotecas utilizadas pelo sistema

As bibliotecas do OpenFrameworks utilizadas foram: GLUT (The OpenGL Utility Toolkit) e a GLEE (GL Easy Extension library) para o desenvolvimento gráfico, a rtAudio para o *input* e *output* de áudio, a freeImage para suportar imagens de vários formatos, a freeType utilizada para "rasterizar" caracteres em *bitmaps*, a fmodex para criação e reprodução de áudio, e a poco para a comunicação entre sistemas.

Para além destas bibliotecas, foram adicionadas a ofxDsp e ofxGui para a construção de interface gráfica, ofxDirList para a inclusão de *strings* de nomes de ficheiros, ofxVectorMath para trabalhar com vectores, ofxXmlSettings para trabalhar com ficheiros XML, ofxOsc (OpenSoundControl) para a comunicação de dados por rede, e ofxFreeFrame para trabalhar com a biblioteca FreeFrame em efeitos de vídeo.

5.3 Efeitos

Todos os efeitos são incorporados neste sistema como *plugins*. São construídos e importados com a codificação de efeitos FreeFrame, sendo utilizadas primitivas de OpenGL. Encontram-se muitos *plugins* nesta plataforma a serem comercializados ou com o código aberto, mas para o sistema operativo Windows.

Assim, todos os efeitos tiveram que ser implementados ou adaptados para interagirem correctamente com a Chameleon Mixer.

6 Conclusões

O trabalho desenvolvido passou primeiramente pela projecção e criação de um *hardware*, o qual se revelou bastante moroso. No entanto, esta etapa foi fundamental para validar todo o conceito inerente a este projecto. A inexistência de um sistema controlador com estas características foi determinante para o desenvolvimento deste *hardware,* e deste modo provar o conceito inovador de uma mesa de mistura de efeitos em tempo real.

Em relação à construção do *hardware,* surgiram diversos contratempos que obrigaram a várias reconstruções. Tal deveu-se à falta de prática e conhecimento nesta área, ao recurso a material reciclado, assim como à utilização de ferramentas nem sempre adequadas às necessidades. Contudo, é reconfortante ter um equipamento que se encontra a funcionar de uma forma estável, que cumpre os requisitos iniciais, e que pode ser utilizado tanto no contexto de *VJing*, como no controlo de outros programas de computador devido à sua construção modular. Com esta construção, este sistema permite também controlar várias aplicações em simultâneo, recorrendo à comunicação de rede implementada.

Na parte do *firmware* não era esperada a criação de um novo *firmware* para o Arduino Mega. No entanto, foi uma boa forma de aprendizagem intrínseca do seu funcionamento.

A aplicação de *software* teve um desenvolvimento bastante complicado até se optar por uma linguagem mais adequada a este sistema e encontrarem-se as ferramentas que permitiram todo o seu desenvolvimento. Surgiram muitos problemas na criação da interface gráfica, no entanto foram criados diversos elementos de raiz, e o resultando final correspondeu às expectativas iniciais. Percebeu-se o colossal potencial da ferramenta C++ ao utilizar e interligar o OpenFrameworks com várias bibliotecas. Outra linguagem, nomeadamente o Processing, não permitiria criar uma aplicação tão robusta como esta.

Tanto no *software* como no *hardware*, a utilização de ferramentas de código aberto e multiplataforma impõe várias restrições que limita o alcance de soluções. Deste modo, tornou-se necessária a procura e construção de soluções alternativas.

Relativamente aos efeitos, foi importante a utilização de métodos académicos e a sua adaptação ao paradigma da Chameleon Mixer.

De uma forma global, os objectivos foram claramente cumpridos com sucesso, tendo até ultrapassado as expectativas iniciais. Numa tentativa de obter um resultado que

correspondesse ao conceito inicial, foi dedicado muito tempo à investigação de novas tecnologias, ferramentas e soluções. Caso o objectivo fosse apenas construir um *software* de efeitos digitais em tempo real, independente das restrições de *hardware*, de interface e de todo o conceito, o tempo despendido para o seu desenvolvimento seria apenas uma fracção do que foi necessário neste projecto.

As constantes discussões e sessões de *brainstorming* informais com pessoas de diversas áreas, influenciaram positivamente o desenvolvimento deste projecto. Também foi importante a apresentação da Chameleon Mixer em estado funcional, na exposição artística multidisciplinar da Audiência Zero, presente no Espaço do Tempo (Montemor-o-Novo) em Julho de 2010, testando a validade deste sistema.

Fica assim demonstrado, com sucesso, um novo conceito de criação e manipulação de vídeo em tempo real.

A restrição do relatório a 50 páginas foi uma clara limitação para a descrição de todas as etapas deste projecto. Foi necessário dedicar bastante tempo à sua escrita de forma a sintetizá-las neste espaço.

Todo o trabalho realizado contribuiu para o desenvolvimento pessoal e profissional, passando não só pela aplicação e consolidação dos vastos conhecimentos adquiridos durante o percurso académico, mas também pela aprendizagem de novos conceitos e técnicas.

6.1 Trabalho Futuro

Em relação ao trabalho futuro podem ser adicionadas novas funcionalidades, bem como optimizar as já existentes.

A área onde poderão ser efectuados mais progressos será na criação dos efeitos. Serão construídos efeitos dedicados a uma instalação ou a um evento específico.

Uma das tarefas futuras será anexar mais sensores, efectuando a comunicação com o sistema através do protocolo de rede acima mencionado. A interface gráfica terá que conter mais elementos gráficos para a *input* dos sensores externos, utilizando a mesma analogia dos *inputs* actuais.

Outra funcionalidade a ser implementada está relacionada com a forma como o sistema faz a mistura de vídeo. Neste momento, a transição é feita através da mistura da opacidade dos dois

canais. Outras possibilidades passam por misturar, adicionar ou subtrair componentes como a luminosidade, o canal *alpha* e a cor.

Será adicionada a funcionalidade de gravar o *output* da aplicação para a posterior visualização e reutilização.

Será implementado um *shield* para tornar possível a comunicação tcp/ip no próprio Arduino. Deste modo, a mesa liga-se directamente à rede, sem necessidade de um computador para fazer de *bridge*, podendo colocar-se o computador que processa os efeitos num local remoto.

7 Bibliografia

[1] Arduino. Arduino. [Online]. http://www.arduino.cc/en/Guide/Introduction

[2] Processing. (2009, Aug.) Processing. [Online]. http://processing.org/

[3] OpenFrameworks. (2009, Aug.) OF. [Online]. http://www.openframeworks.cc/

[4] VJ Theory, *VJam Theory: Collective Writings on Realtime Visual Performance*. Falmouth, Reino Unido: realtime Books, Aug. 2008. [Online].

[5] Grupo Folha. (2005, Mar.) Folha Online. [Online]. http://www1.folha.uol.com.br/folha/ciencia/ult306u13024.shtml

[6] Patricia Silveirinha. A arte vídeo. [Online]. http://bocc.ubi.pt/pag/_texto.php3?html2=silveirinha-patricia-Arte-Video.html

[7] Nicolau Centola, "Laptop music e a nova estética da performance na música electrónica," in *II Simpósio ABCiber*, Osasco, 2008, p. 15.

[8] D-Fuse, *VJ Audio-Visual Art + VJ Culture*. Inglaterra: Laurence King Publishers, 2006.

[9] Mia Mekela, *LIVE CINEMA: Language and Elements*. Helsinquia, Finlandia, 2006.

[10] ART WE LOVE. ART WE LOVE. [Online]. http://artwelove.com/explore/Techniques-and-Media/Performance-Art

[11] Roland. Roland. [Online]. http://www.musicap.rs

[12] http://www.modul8.ch/. (2010, Aug.) Modul 8. [Online]. http://www.modul8.ch/

[13] arkaos. arkaos interactive visual technologies. [Online]. http://www.arkaos.net/product/index.php?catid=1&pid=1001&iid=42

[14] Resolume. (2010, Aug.) Resolume. [Online]. http://www.resolume.com/

[15] vjamm. VJAMM A/V software. [Online]. http://www.vjamm.com/

[16] FLxER. FLxER.net. [Online]. http://www.flxer.net/

[17] VIDVOX. VIDVOX. [Online]. http://www.vidvox.net/

[18] LIVID. LIVID. [Online]. http://www.lividinstruments.com/software_celldna.php

[19] Timithy Jaeger, *live cinema unraveled*.: UCSD Russel Grant, 2005.

[20] Joshua Noble, *Programming Interactivity: A designer's Guide to Processing, Arduiono, and OpenFrameworks*, First Edition ed., Steve Weiaa, Ed. Beijing, Cambridge, Farnham, Koln, Sebastopol, Taipei, Tokyo, United States of America: O'Reilly, 2009.

[21] John Hewes. (2010) The Electronics Club. [Online]. http://www.kpsec.freeuk.com/components/led.htm

[22] Jacob Holst, and Melvin Ochsmann Anders Gran. (2005) ARDUINO meets PROCESSING. [Online]. http://webzone.k3.mah.se/projects/arduino-workshop/projects/arduino_meets_processing/instructions/poti.html

[23] NerdKits, L.L.C. NerdKits. [Online]. http://www.nerdkits.com/videos/sound_meter/

[24] JOSE LOZANO. The Blow(n) Sensor !. [Online]. http://www.dtic.upf.edu/~jlozano/interfaces/blow_sensor.html

[25] Susan Riedel James W. Nilsson, *Electric Circuits*, 8th ed., Maria J.Horton, Ed. Upper Saddle River, New Jersey: Pearson Prentice Hall, 2008.

[26] Hobbyprojects.com. (2009, Sep.) Hobby Projects. [Online]. http://www.hobbyprojects.com/A/audio_amplifiers.html

[27] Kenneth C. Smith Adel S. Sedra, *Microelectronic Circuits*, 6th ed., Oxford, Ed. New york Oxford: Oxford University Press, Inc., 2010.

[28] Howard W. Sams, *Handbook for Sound Engeneers*, 2nd ed., Glen Ballou, Ed. Carmel, Indiana: Macmillan Computer Publishing, 1991.

[29] Fritzing. FRITZING ALPHA. [Online]. http://fritzing.org/

[30] Arduino. Arduino. [Online]. http://www.arduino.cc/en/Tutorial/LiquidCrystal

[31] Ladyada. Character LCDs. [Online]. http://www.ladyada.net/learn/lcd/charlcd.html

[32] embedds. Embedded projects from around the web. [Online]. http://www.embedds.com/arduino-controls-lcd-via-3-wires/

[33] FTDI Chip. (2010) Future Technology Devices International Ltd. [Online]. http://www.ftdichip.com/

[34] Firmata. (2010, Mar.) Firmata. [Online]. http://firmata.org/wiki/Main_Page

[35] Maxuino. Maxuino. [Online]. http://www.maxuino.org/

[36] Ladyada. Arduino Tutorial. [Online]. http://www.ladyada.net/learn/arduino/lesson4.html

[37] Arduíno. Arduíno. [Online]. http://www.arduino.cc/en/Reference/PortManipulation

[38] Ben Fry Casey Reas, *Processing A programming Handbook for Visual Designers and Artists*, 1st ed. Massachusetts, USA: MIT Press, 2007.

[39] Ira Greenberg, *Processing: Creative Coding and Computational Art*, 1st ed., Chris Mills, Ed. USA: FriendSofed, 2007.

[40] Brendan Berg. (2010, May) INTERFASCIA 003 ALPHA. [Online]. http://www.superstable.net/interfascia/

[41] Peter Lager. Quark's place. [Online]. http://www.lagers.org.uk/g4p/index.html

[42] Andreas Schlegel. (2010, May) controlP5. [Online]. http://www.sojamo.de/libraries/controlP5/

[43] opensoundcontrol.org. opensoundcontrol.org. [Online]. http://opensoundcontrol.org/introduction-osc

[44] Andreas Schlegel. (2010) OscP5. [Online]. http://www.sojamo.de/libraries/oscP5/#examples

[45] NUI Group. Argos An OpenFrameworks. [Online]. http://argos.dimitridiakopoulos.com/source/

[46] Clutter. Clutter. [Online]. http://www.clutter-project.org/

[47] GLV. (2008) GLV. [Online]. http://mat.ucsb.edu/glv/

[48] GTK+. The GTK+ Project. [Online]. http://www.gtk.org

[49] alphakanal. Alphakanal. [Online]. http://blog.alphakanal.de/2008/08/22/ofxgui

[50] Nuicode. Community Core Vision. [Online]. http://nuicode.com/projects/tbeta

[51] Philip Whitfield. [Online]. http://underdoeg.com/

[52] OpenFrameworks. (2010, Sep.) [Online]. http://www.openframeworks.cc/addons

[53] (VJ Setup Image), Velvet Systems. Velvet Systems. [Online]. http://www.velvetsystems.com/menu.43.Video.Equipment.Hire.cfm

[54] UrbanScreen. UrbanScreen. [Online]. http://www.urbanscreen.com/

[55] (2009, Sep.) reconn's World. [Online]. http://www.reconnsworld.com/forum/read.php?9,10

[56] Lazano. The Blow(n) Sensor !. [Online]. http://www.dtic.upf.edu/~jlozano/interfaces/blow_sensor.html

[57] Guilherme Martins. (2009, May) Lab Guilherme Martins. [Online]. http://lab.guilhermemartins.net/2009/05/03/sound-sensor/

8 Anexos

8.1 Apêndice A – *VJing*

8.1.1 Esquemas de Setup de VJ

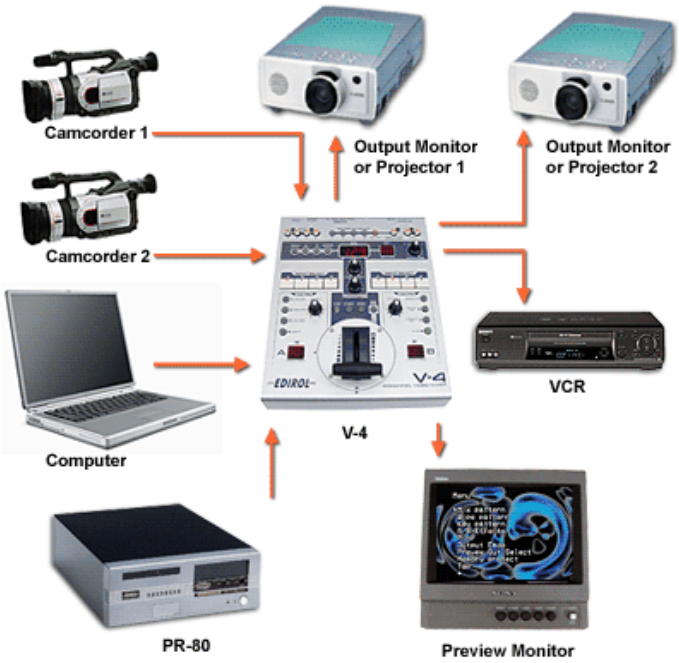


Figura 8.1 - Esquema de um *Setup* de VJ [53]

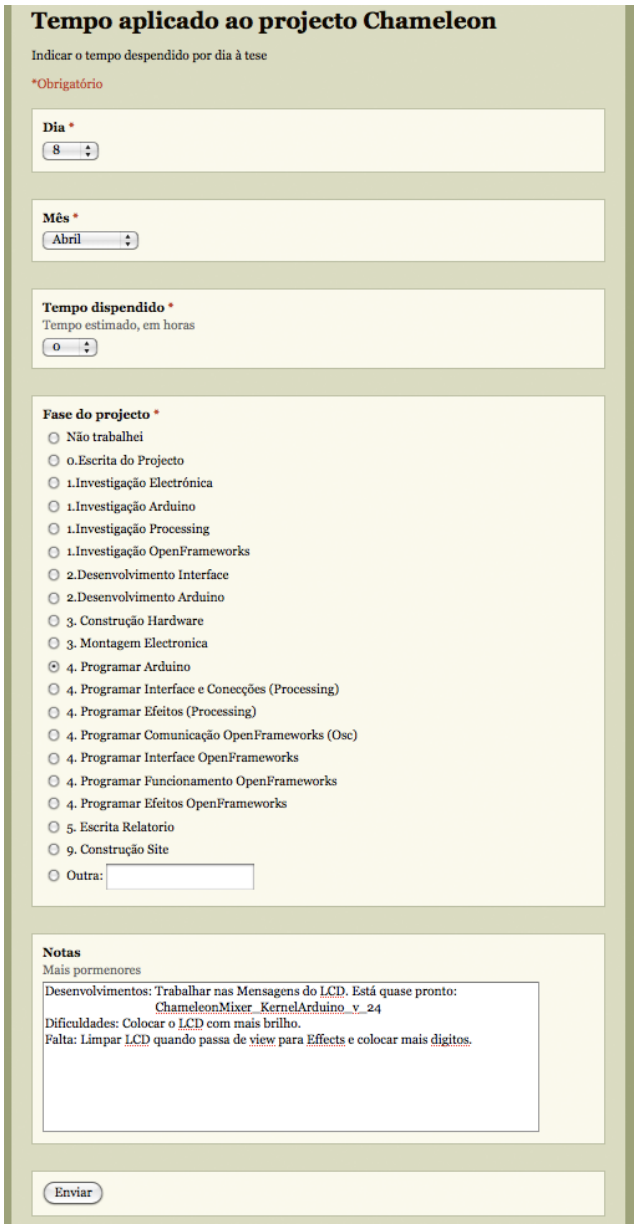
8.1.2 Resultados de *project mapping*

Figura 8.2 - *Project mapping* por urbanscreen [54]

8.2 Apêndice B - Planeamento

8.2.1 Registo

O registo é feito segundo o formulário que se encontra em baixo. Este formulário permitiu o registo em todas as etapas do projecto da seguinte informação: data, tempo de trabalho no dia em questão, fase do projecto em que se encontra, notas sobre o desenvolvimento, dificuldades e próximos passos a ser executados.



Tempo aplicado ao projecto Chameleon

Indicar o tempo despendido por dia à tese

*Obrigatório

Dia *
8

Mês *
Abril

Tempo dispendido *
Tempo estimado, em horas
0

Fase do projecto *
- ○ Não trabalhei
- ○ 0.Escrita do Projecto
- ○ 1.Investigação Electrónica
- ○ 1.Investigação Arduino
- ○ 1.Investigação Processing
- ○ 1.Investigação OpenFrameworks
- ○ 2.Desenvolvimento Interface
- ○ 2.Desenvolvimento Arduino
- ○ 3. Construção Hardware
- ○ 3. Montagem Electronica
- ◉ 4. Programar Arduino
- ○ 4. Programar Interface e Conecções (Processing)
- ○ 4. Programar Efeitos (Processing)
- ○ 4. Programar Comunicação OpenFrameworks (Osc)
- ○ 4. Programar Interface OpenFrameworks
- ○ 4. Programar Funcionamento OpenFrameworks
- ○ 4. Programar Efeitos OpenFrameworks
- ○ 5. Escrita Relatorio
- ○ 9. Construção Site
- ○ Outra:

Notas
Mais pormenores

Desenvolvimentos: Trabalhar nas Mensagens do LCD. Está quase pronto:
ChameleonMixer_KernelArduino_v_24
Dificuldades: Colocar o LCD com mais brilho.
Falta: Limpar LCD quando passa de view para Effects e colocar mais digitos.

Enviar

Figura 8.3 - *ScreenShot* do registo diário de trabalho

8.2.2 Requisitos da *Interface*

Nesta parte do documento é feita uma descrição das interfaces que envolvem a aplicação de *software*, de forma a tornar mais sintética a forma como o utilizador poderá interagir com a aplicação e usufruir das suas potencialidades.

8.2.2.1 Interface para o utilizador

Quanto à interface gráfica, é necessário uma interface simples e intuitiva. Sendo esta uma aplicação que poderá ser controlada em tempo real, serão minimizados os cliques do rato e será dado o maior controlo a partir do *hardware*. A interface do *software* será maioritariamente para dar o *feedback* do *hardware*. É composta por 3 grandes grupos: pré-visualização de efeitos, mistura de efeitos e o vídeo final. Na pré-visualização de efeitos será possível ver uma imagem (ou animação) ilustrativa do efeito pretendido. Na parte de Mistura de efeitos será dado o *feedback* dos níveis de cada parâmetro dos efeitos (níveis dos potenciómetros e botão *on/off*), e na secção do vídeo final mostram-se os níveis dos sensores e a imagem misturada. A interacção na aplicação é feita pelo rato ou teclado, apenas quando não é possível pela mesa, e em casos em que o tempo de selecção não é um factor crítico, como no caso de mudança de canal, ou substituição de um potenciómetro por um sensor.

8.2.2.2 Output do sistema

A aplicação é composta por uma segunda janela que servirá de *output* para o projector ou outro equipamento de visualização como um ecrã de LEDs. Este *output* tem a mesma imagem visualizada no vídeo final na interface do utilizador, mas esta é uma imagem processada em *fullscreen*.

8.2.2.3 Dispositivos de *hardware* para a interface

A interface de *hardware* é composta por um rato, teclado, ecrã e a mesa construída para o efeito.

A mesa cobre a maior parte de *inputs* para toda a aplicação. Tem *inputs* de selecção de modo da aplicação entre pré-visualização do efeito e mistura de efeito, selecção de canal A ou B e respectiva selecção de efeito. Conta com um potenciómetro para a mistura entre o canal A e B e um potenciómetro para a selecção de cada parâmetro do efeito, sendo possível desligá-lo ou ligá-lo. Ainda contempla um LCD com o *output* do efeito seleccionado.

8.2.2.4 Interface de *Software*

A aplicação será desenvolvida para correr no sistema operativo Windows, Mac OS ou Linux, dado que a interface é desenvolvida na linguagem Processing que é compatível com estes sistemas operativos. Esta aplicação também comunicará com o Arduino, também compatível com estes sistemas operativos.

8.2.2.5 Interface de comunicação

A aplicação utilizará o protocolo de comunicação *Firmata*. Este protocolo utiliza uma comunicação série.

8.2.3 Requisitos Funcionais

É necessário que a aplicação funcione com o máximo de interacções possível, dado que será para se utilizar em tempo real. Para isso, conta com uma disposição uniforme e simétrica para a maximização da utilização das duas mãos em simultâneo. A distância entre os potenciómetros foi pensada para ser confortável na sua utilização, visto que serão estes os componentes a ter uma maior utilização.

8.2.4 Requisitos de performance

A aplicação terá que ser estável, não devendo *crashar* no tempo que estiver em actividade. Deverá ser rápida e fluida na transição entre os vários efeitos e mudança de modo.

Todo o processamento inerente aos efeitos deverá ser em tempo real, sempre que possível aproveitando o processamento da placa gráfica com as funções de OpenGL.

8.2.5 Restrições de design

A aplicação deverá ter uma interface simples, intuitiva, com botões exemplificativos do menu ou função provocada quando se clica nele. O esquema de cores deve ser adequado ao público-alvo a que se destina e visível, dado que poderá ser utilizado em ambientes escuros. O número de cliques do rato será minimizado, maximizando a utilização da mesa. A quantidade e o tamanho de botões dos menus terão em conta três factores: visibilidade, organização e design. O design da aplicação será coerente entre o *software* e o *hardware*.

8.2.6 Outros Requisitos

Contemplaremos os requisitos do sistema de *hardware* que passamos a descrever:

Computador e processador: processador de 2 giga hertz (GHz) e 2 Gb de RAM ou superior;

Disco rígido: espaço livre de 500MB;

Resolução do monitor: 1024x768 ou superior;

Para correr a aplicação será necessário uma porta USB para a comunicação entre a mesa e o computador;

Apesar de a mesa ser alimentada pelo computador, é aconselhado ligar uma tomada para fornecer a corrente adicional.

8.2.7 Considerações para a implementação

Face aos requisitos a ser implementados, surge a necessidade de os categorizar de forma a impor uma ordem e desenvolver as respectivas funcionalidades.

Os critérios utilizados para categorizar os requisitos a ser implementados basearam-se sobretudo na facilidade de implementação, grau na hierarquia de requisitos (componentes mais abstractos dependem de componentes mais tangíveis), familiarização com as ferramentas de programação e linha de aprendizagem. Para a categorização, a prioridade 1 será máxima, enquanto a prioridade mínima será 5.

8.2.7.1 Categorização de requisitos investigação

Requisito	Prioridade	Cumprido
Testar vários tipos de LEDs	1	Sim
Testar vários tipos de potenciómetros	1	Sim
Testar vários tipos de LDR´s	2	Sim
Testar piezoeléctrico	2	Sim
Amplificar piezo	2	Optou-se por um microfone
Colocar piezo como sensor sopro	5	Sim
Colocar piezo como sensor de toque	3	Sim
Colocar Alcoolímetro a funcionar	5	Sim
Colocar sensor de temperatura a funcionar	3	Sim
Colocar microfone a funcionar	3	Sim
Investigação Processing	1	Sim
Investigação Arduino	1	Sim
Investigação OpenFrameworks	1	Sim
Comunicar entre Arduino Duemilinove - Processing	1	Sim
Comunicar entre Arduino Duemilinove - OpenFrameworks	1	Sim

Tabela 8.2 - Categorização dos requisitos da fase de investigação

8.2.7.2 Categorização de requisitos para criação de estrutura

Requisito	Prioridade	Cumprido
Desmontar Mesa Mistura Som	1	Sim
Desmontar impressora	1	Sim
Desmontar teclado	1	Sim
Criar desenho para a placa metálica	1	Sim
Criar desenho para a placa superior	1	Sim
Criar desenho para placa interior	1	Sim
Adaptar estrutura	1	Sim
Adaptar potenciómetros	2	Sim
Adaptar botões do teclado	2	Sim
Adaptar teclado Numérico	2	Sim
Adaptar botão 2 estados	3	Sim
Criar placa superior	2	Sim
Criar rodas	5	Criou-se mas não foram utilizadas
Criar suportes Arduino	1	Sim
Criação de estrutura para os LEDs	2	Sim

Tabela 8.3 - Categorização dos requisitos para a criação da estrutura

8.2.7.3 Categorização de requisitos para montagem electrónica

Requisito	Prioridade	Cumprido
Criar esquema prototipagem Electrónico no Fritzing	1	Sim
Criar esquemático	4	Sim
Marcação dos componentes nas placas	1	Sim
Soldar Arduino à Placa Perfurada	1	Sim
Soldar as ligações na placa perfurada	1	Sim
Montagem dos *flat cables*	1	Sim
Soldar fios dos falt cables aos componentes	1	Sim
Montagens dos LEDs	2	Sim
Montar LEDs RGB	5	Sim
Montar Microfone	3	Sim
Montar Sensor Pressão	3	Sim
Montar Sensor Temperatura	3	Sim
Montar LDR	3	Sim
Montar LCD	1	Sim

Tabela 8.4 - Categorização dos requisitos para a montagem electrónica

8.2.7.4 Categorização de requisitos para o firmware

Requisito	Prioridade	Cumprido
Estabelecer comunicação com Arduino	1	Sim
Adaptar protocolo	1	Sim
Comunicação de Pins digitais *Output*	1	Sim
Comunicação de Pins *Input*	1	Sim
Comunicação de Pins digitais *input*(PWM)	1	Sim
Mapeamento de Portas e Pinos	1	Sim
Linearização dos Potenciómetros	3	Sim
Criação de temporizador e variável de estados para botões canais dos efeitos	5	Sim
Criação de número do efeito com teclado numérico	3	Sim
Criar *output* do LCD	1	Sim
Comunicar com a aplicação	1	Sim

Tabela 8.5 - Categorização dos requisitos para o firmware

8.2.7.5 Categorização de requisitos para o *software*

Requisito	Prioridade	Cumprido
Criação do *layout*	1	Sim
Criar pré-visualização de efeitos	5	Trabalho futuro
Receber valores do Firmware	1	Sim
Reencaminhar valores para outras aplicações	4	Sim
Receber *inputs* de outras aplicações	3	Sim
Output 2º ecrã	1	Sim
Efeito de selecção do *mixer*	5	Sim
Input vídeo câmara	1	Sim
Input vídeo do computador	2	Sim
Carregar dinamicamente efeitos (depois de iniciar)	4	Sim
Interligação de Efeitos FF	2	Sim
Criar efeitos	1	Sim

Tabela 8.6 - Categorização dos requisitos para o *software*

8.3 Apêndice C – Tabelas Chameleon Mixer

Arduino Mega (Chameleon Mixer)					
Porta	Digital Pin	Pino Analógico	Uso Interno	Uso Chameleon Mixer	
				Output(Sai Arduino)	Input (Entra Arduino)
PE 0	0		USART0 RX, pin Int 8		
PE 1	1		USART0 TX		
PE 4	2		PWM T3B, INT4	LED Azul	
PE 5	3		PWM T3C, INT5	LED Vermelho	
PG 5	4		PWM T0B	LED Verde	
PE 3	5		PWM T3A	controlar LCD	
PH 3	6		PWM T4A	controlar LCD	
PH 4	7		PWM T4B	controlar LCD	
PH 5	8		PWM T4C	controlar LCD	
PH 6	9		PWM T2B	controlar LCD	
PB 4	10		PWM T2A, Pin Int 4	controlar LCD	
PB 5	11		PWM T1A, Pin Int 5	Ver Efeitos Existentes seleccionado	
PB 6	12		PWM T1B, Pin Int 6		
PB 7	13		PWM T0A, Pin Int 7		
PJ 1	14		USART3 TX, Pin Int 10		
PJ 0	15		USART3 RX, Pin Int 9		
PH 1	16		USART2 TX		
PH 0	17		USART2 RX		
PD 3	18		USART1 TX, Ext Int 3		
PD 2	19		USART1 RX, Ext Int 2		
PD 1	20		I2C SDA, Ext Int 1		
PD 0	21		I2C SCL, Ext Int 0		
PA 0	22		Ext Memory addr bit 0	Canal A pressionado	
PA 1	23		Ext Memory addr bit 1	Canal B pressionado	
PA 2	24		Ext Memory addr bit 2		Botão Led
PA 3	25		Ext Memory addr bit 3		Botão Led
PA 4	26		Ext Memory addr bit 4		Botão Led
PA 5	27		Ext Memory addr bit 5		Botão Led
PA 6	28		Ext Memory addr bit 6		Botão Led
PA 7	29		Ext Memory addr bit 7		Botão Led
PC 7	30		Ext Memory addr bit 15		Botão Led
PC 6	31		Ext Memory addr bit 14		Botão Led
PC 5	32		Ext Memory addr bit 13		Botão Led
PC 4	33		Ext Memory addr bit 12		Botão Led
PC 3	34		Ext Memory addr bit 11	Teclado Numérico	
PC 2	35		Ext Memory addr bit 10	Leds	
PC 1	36		Ext Memory addr bit 9	Teclado Numérico	
PC 0	37		Ext Memory addr bit 8	Leds	
PD 7	38			Teclado Numérico	
PG 2	39		ALE Ext Mem	Leds	
PG 1	40		RD Ext Mem	Teclado Numérico	
PG 0	41		Wr Ext Mem	Leds	
PL 7	42			Teclado Numérico	
PL 6	43			Leds	
PL 5	44		*PWM 5C	Teclado Numérico	
PL 4	45		PWM 5B	Leds	
PL 3	46		PWM 5A	Teclado Numérico	
PL 2	47		T5 external counter	Leds	
PL 1	48		ICP T5	Teclado Numérico	
PL 0	49		ICP T4	Leds	
PB 3	50		SPI MISO	Teclado Numérico	
PB 2	51		SPI MOSI	Leds	
PB 1	52		SPI SCK	Teclado Numérico	
PB 0	53		SPI SS	Leds	
PF 0		0		Sensor	
PF 1		1		Sensor	
PF 2		2		Sensor	
PF 3		3			Passar Efeito do canal actual
PF 4		4		Sensor	
PF 5		5		Potenciometro	
PF 6		6		Potenciometro	
PF 7		7		Potenciometro	
PK 0		8	Pin Int 16	Potenciometro	
PK 1		9	Pin int 17	Potenciometro	
PK 2		10	Pin Int 18	Potenciometro	
PK 3		11	Pin Int 19	Potenciometro	
PK 4		12	Pin Int 20	Potenciometro	
PK 5		13	Pin Int 21	Potenciometro	
PK 6		14	Pin Int 22	Potenciometro	
PK 7		15	Pin Int 23	Potenciometro	

Tabela 8.7 – Portas e a sua função na Chameleon Mixer

8.4 Apêndice D – Detalhes Técnicos

8.4.1 Diagrama final do funcionamento do sistema Chameleon Mixer

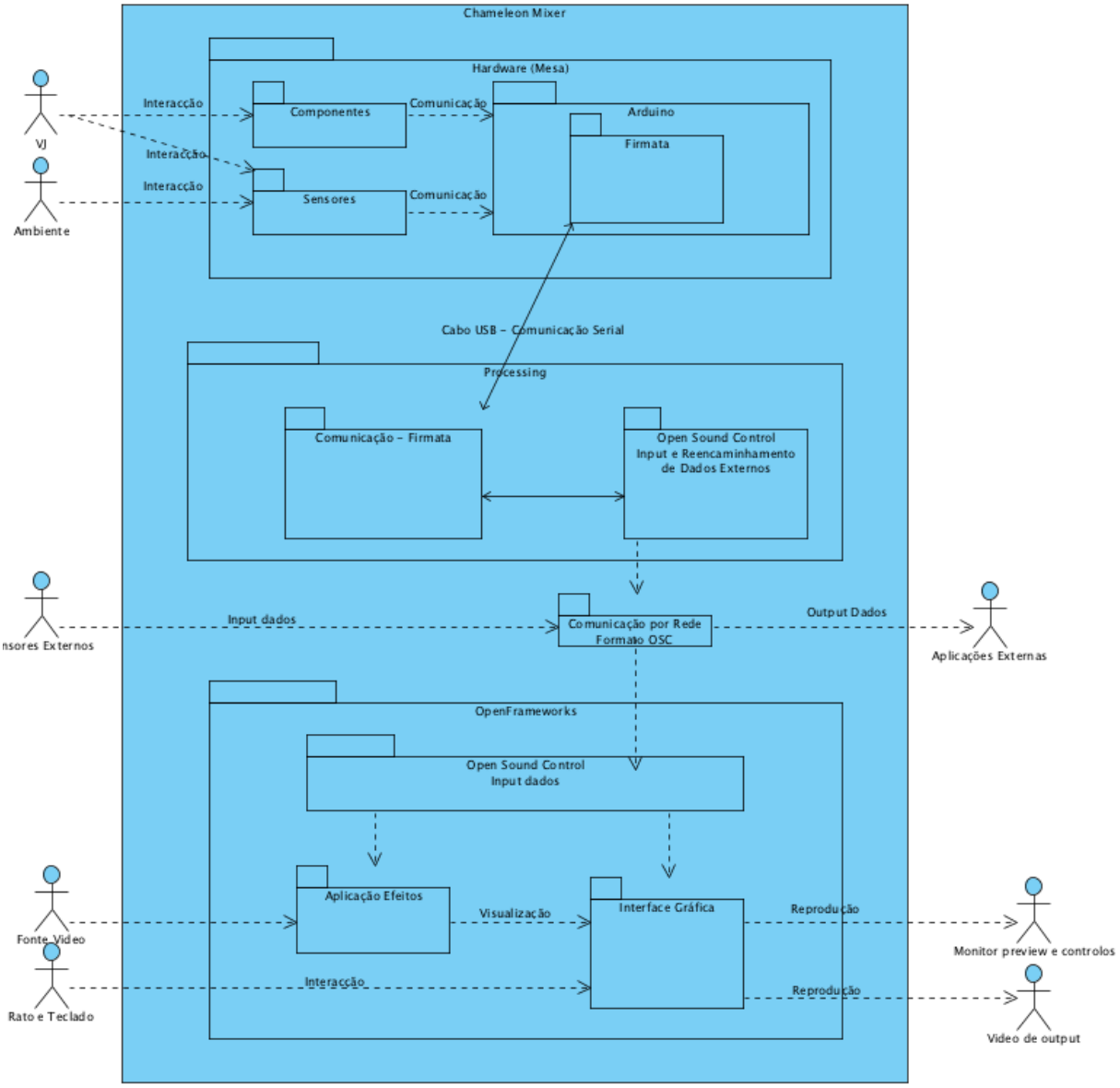


Figura 8.4 - Diagrama Conceptual Final Funcional do Chameleon Mixer

8.4.2 Arduino

Arduino Duemilinove		***Arduino Mega***	
Microcontroller	*ATmega168*	*Microcontroller*	*ATmega1280*
Operating Voltage	*5V*	*Operating Voltage*	*5V*
Input Voltage (recommended)	*7-12V*	*Input Voltage (recommended)*	*7-12V*
Input Voltage (limits)	*6-20V*	*Input Voltage (limits)*	*6-20V*
Digital I/O Pins	*14 (of which 6 provide PWM output)*	*Digital I/O Pins*	*54 (of which 14 provide PWM output)*
Analog Input Pins	*6*	*Analog Input Pins*	*16*
DC Current per I/O Pin	*40 mA*	*DC Current per I/O Pin*	*40 mA*
DC Current for 3.3V Pin	*50 mA*	*DC Current for 3.3V Pin*	*50 mA*
Flash Memory	*16 KB (ATmega168) or 32 KB (ATmega328) of which 2 KB used by bootloader*	*Flash Memory*	*128 KB of which 4 KB used by bootloader*
SRAM	*1 KB (ATmega168) or 2 KB (ATmega328)*	*SRAM*	*8 KB*
EEPROM	*512 bytes (ATmega168) or 1 KB (ATmega328)*	*EEPROM*	*4 KB*
Clock Speed	*16 MHz*	*Clock Speed*	*16 MHz*

Tabela 8.8 - Comparação das características do Arduino Duemilinove com as do Arduino Mega

8.4.2.1 Detalhes Técnicos dos port registers do Arduino Duemilinove / Mega:

Arduino Pins Functions						
	Arduino 168/328			Arduino Mega		
Digital Pin	Port	Analog pin	Usage	Port	Analog Pin	Usage
0	PD 0		USART RX	PE 0		USART0 RX, pin Int 8
1	PD 1		USART TX	PE 1		USART0 TX
2	PD 2		Ext Int 0	PE 4		PWM T3B, INT4
3	PD 3		PWM T2B, Ext Int 1	PE 5		PWM T3C, INT5
4	PD 4			PG 5		PWM T0B
5	PD 5		PWM T0B	PE 3		PWM T3A
6	PD 6		PWM T0A	PH 3		PWM T4A
7	PD 7			PH 4		PWM T4B
8	PB 0		Input Capture	PH 5		PWM T4C
9	PB 1		PWM T1A	PH 6		PWM T2B
10	PB 2		PWM T1B	PB 4		PWM T2A, Pin Int 4
11	PB 3		PWM T2A, MOSI	PB 5		PWM T1A, Pin Int 5
12	PB 4		MISO	PB 6		PWM T1B, Pin Int 6
13	PB 5		SCK	PB 7		PWM T0A, Pin Int 7
14	PC 0	0		PJ 1		USART3 TX, Pin Int 10
15	PC 1	1		PJ 0		USART3 RX, Pin Int 9
16	PC 2	2		PH 1		USART2 TX
17	PC 3	3		PH 0		USART2 RX
18	PC 4	4	I2C SDA	PD 3		USART1 TX, Ext Int 3
19	PC 5	5	I2C SCL	PD 2		USART1 RX, Ext Int 2
20				PD 1		I2C SDA, Ext Int 1
21				PD 0		I2C SCL, Ext Int 0
22				PA 0		Ext Memory addr bit 0
23				PA 1		Ext Memory addr bit 1
24				PA 2		Ext Memory addr bit 2
25				PA 3		Ext Memory addr bit 3
26				PA 4		Ext Memory addr bit 4
27				PA 5		Ext Memory addr bit 5
28				PA 6		Ext Memory addr bit 6
29				PA 7		Ext Memory addr bit 7
30				PC 7		Ext Memory addr bit 15
31				PC 6		Ext Memory addr bit 14
32				PC 5		Ext Memory addr bit 13
33				PC 4		Ext Memory addr bit 12
34				PC 3		Ext Memory addr bit 11
35				PC 2		Ext Memory addr bit 10
36				PC 1		Ext Memory addr bit 9
37				PC 0		Ext Memory addr bit 8
38				PD 7		
39				PG 2		ALE Ext Mem
40				PG 1		RD Ext Mem
41				PG 0		Wr Ext Mem
42				PL 7		
43				PL 6		
44				PL 5		*PWM 5C
45				PL 4		PWM 5B
46				PL 3		PWM 5A
47				PL 2		T5 external counter
48				PL 1		ICP T5
49				PL 0		ICP T4
50				PB 3		SPI MISO
51				PB 2		SPI MOSI
52				PB 1		SPI SCK
53				PB 0		SPI SS
54				PF 0	0	
55				PF 1	1	
56				PF 2	2	
57				PF 3	3	
58				PF 4	4	
59				PF 5	5	
60				PF 6	6	
61				PF 7	7	
62				PK 0	8	Pin Int 16
63				PK 1	9	Pin int 17
64				PK 2	10	Pin Int 18
65				PK 3	11	Pin Int 19
66				PK 4	12	Pin Int 20
67				PK 5	13	Pin Int 21
68				PK 6	14	Pin Int 22
69				PK 7	15	Pin Int 23
				Functions not available on Mega		
				PD 4 ICP1		
				PD 6 T1 (external counter)		
				PE 6 T3, Int 6		
				PE 7 Int 7, ICP3		
				PH 7 T4 (external counter)		
				PJ 2 Pin Int 11		
			PWM pins	Timer0	4,13	
				Timer1	11,12,(13)	
				Timer2	9,10	
				Timer3	2,3,5	
				Timer4	6,7,8	
				Timer5	45,46	(44 not implimented)
				PM mem on Arduino forums with any additions		

Tabela 8.9 - Comparação dos *Port Registers* entre o Arduino Duemilinove e o Arduino Mega

8.4.3 Cálculo das resistências dos LEDs

LEDs para os Potenciómetros
LEDs verdes: 2.2v 25mA

Resistência para 2 LEDs verdes em paralelo $= \frac{5V - 2.2V}{0.025A + 0.025A} = 56ohm$ (utilizadas de 33 ohm)

LEDs RGB
LED verde: 1.7V 25mA

Resistência para o LED verde $= \frac{5V - 1.7V}{0.025A} = 132ohm$ (utilizado uma de 150 ohm)

LED vermelho: 2.2V 25mA

Resistência para o LED vermelho $= \frac{5V - 2.2V}{0.025A} = 112ohm$ (utilizado uma de 100 ohm)

LED azul: 4.5V 25mA

Resistência para o LED azul $= \frac{5V - 4.5V}{0.025A} = 20ohm$ (utilizado uma de 33 ohm)

8.4.4 Microfone

Detalhes técnicos do microfone usado:

- *Element: Condenser*
- *Polar Pattern: Omnidirectional*
- *Frequency Response: 50 – 18,000 Hz*
- *Open Circuit Sensitivity: -54 dB*
- *Impedance: 1,000 ohms*
- *Weight: 6 g (0.2 oz)*
- *Cable: 6 m (20 foot) terminated with 3.5 mm (1/8-inch) mini-plug*
- *Accessories Furnished: Tie clip; battery; foam windscreen; 6.3 mm (1/4-inch) adapter*

8.4.5 LDR's

Detalhes Técnicos do LDR

Fabricante: SILONEX

Descrição

- *LIGHT DEPENDENT RESISTOR*
- *Power Rating:250mW*
- *Dark Resistance:1Mohm*
- *Resistance @ Lux:5.4kohm to 12.6kohm*
- *Series:NORPS*
- *Operating Temperature Range:-60°C to +75°C*
- *Resistor Element Type:Light Dependent*
- *Voltage Rating:320V dc*
- *External Length / Height:7mm*
- *Body Diameter:14.35mm*
- *Case Style:TO-8*
- *Insulator Material:Plastic or epoxy coat*
- *Lead Length:28.6mm*
- *Peak Spectral Response Wa*

8.4.6 Piezzo

Piezoeléctrico:

• *Rated Voltage: Max 30 Vp-p*

• *Current Consumption: 12mA @ 10Vp-p Square Wave 4.1kHz*

• *Sound Pressure Level(10cm): 90dB @ 10Vp-p Square Wave 4.1kHz*

• *Flying Leads Fitted*

• *High and Clear Sound, audible for many metres - KPE-110*

• *Dimensions: 24mm Diameter, 5mm High, 29mm between mounting holes*

8.4.7 Sensor Temperatura

Transístor LM35 Sensor de temperatura de precisão em graus centígrados

- *Calibrated directly in ° Celsius (Centigrade)*
- *Linear + 10.0 mV/°C scale factor*
- *0.5°C accuracy guaranteeable (at +25°C)*
- *Rated for full -55° to +150°C range*
- *Suitable for remote applications*
- *Low cost due to wafer-level trimming*
- *Operates from 4 to 30 volts*
- *Less than 60 µA current drain*
- *Low self-heating, 0.08°C in still air*
- *Nonlinearity only ±¼°C typical*
- *Low impedance output, 0.1 Ohm for 1 mA load*

8.5 Apêndice E – Esquemas electrónicos

8.5.1 Chameleon Mixer

8.5.1.1 Esquemáticos do Chameleon Mixer

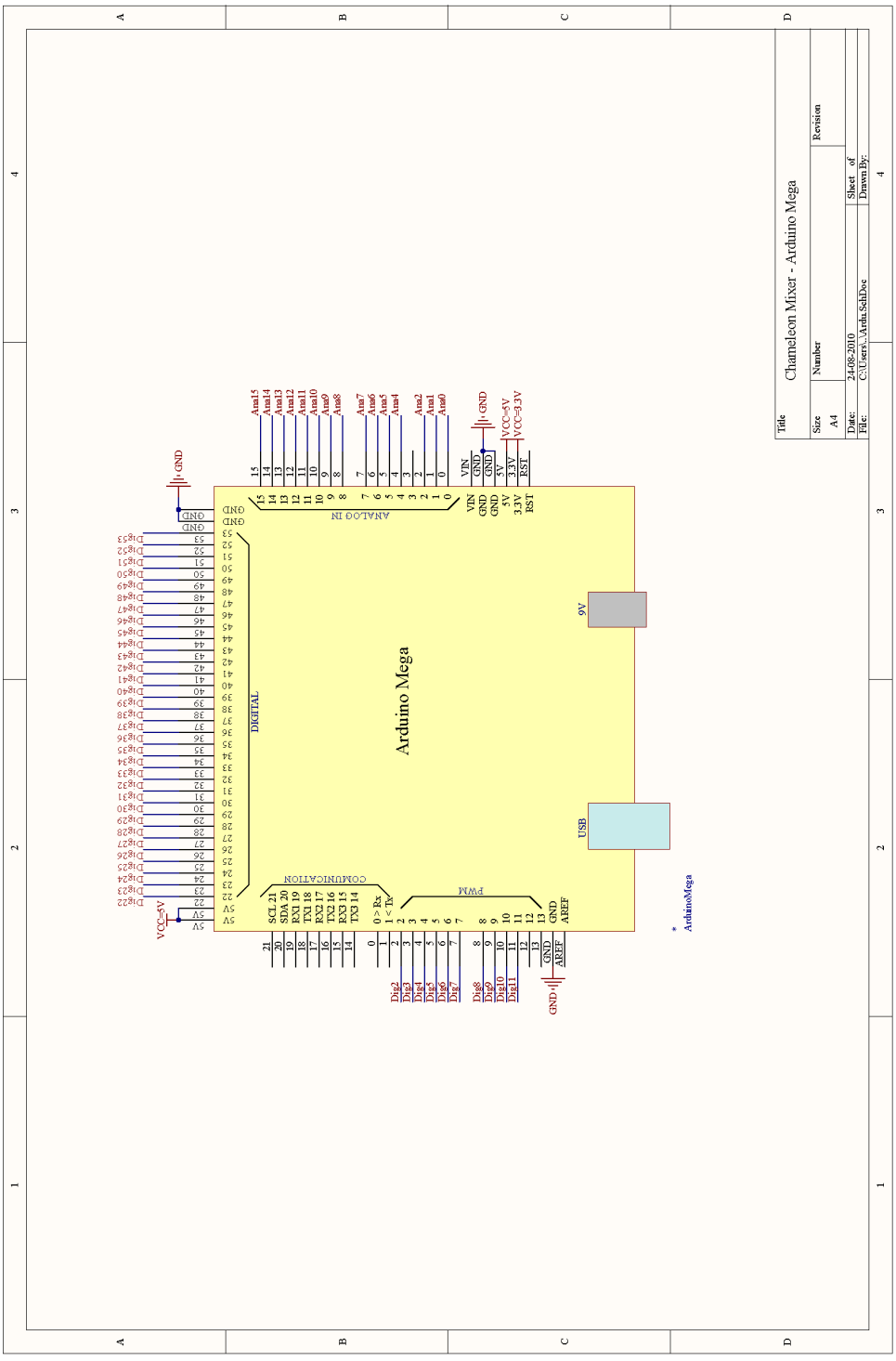


Figura 8.5 – Esquemático do Arduino no Chameleon Mixer

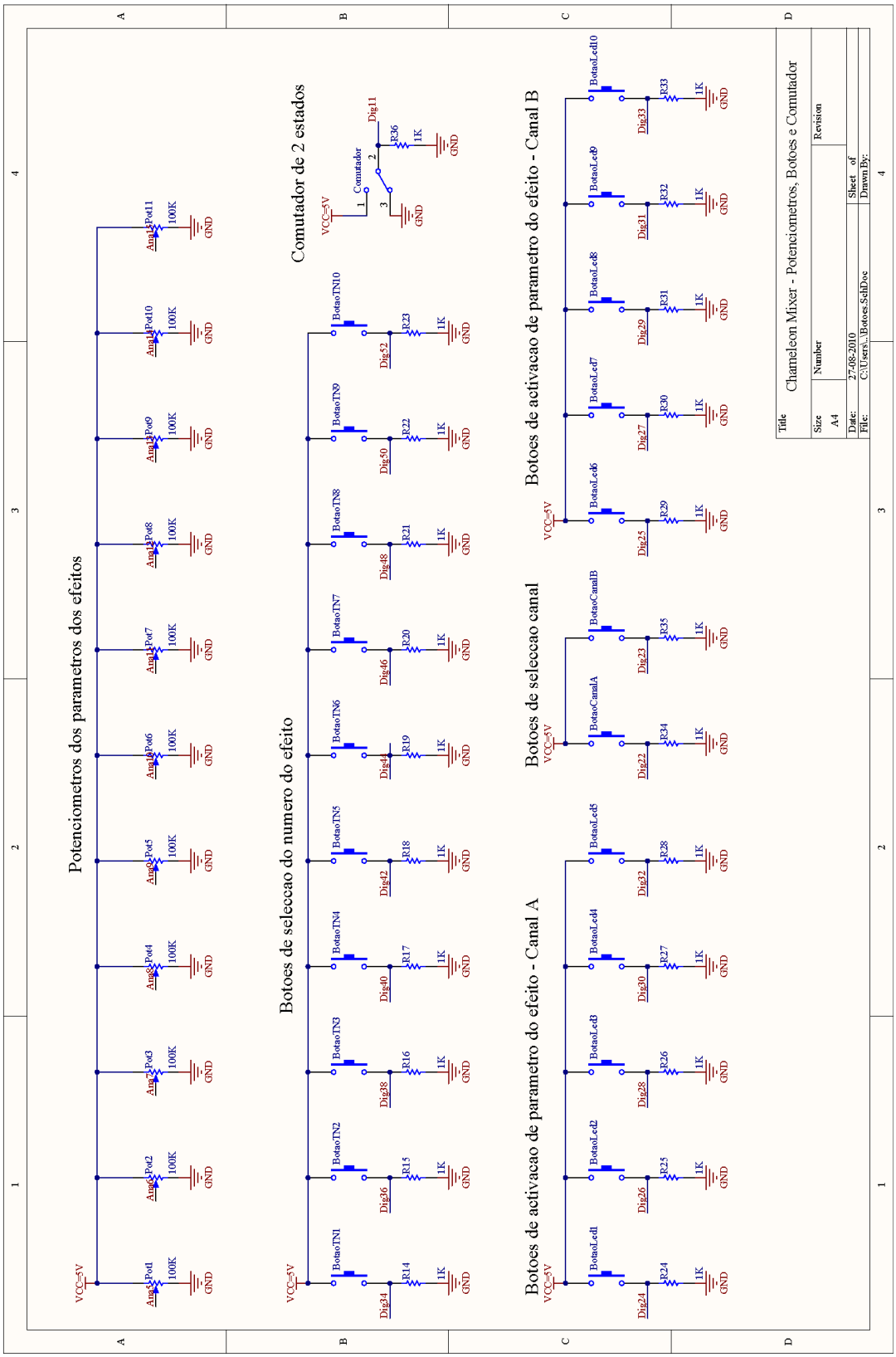


Figura 8.6 – Esquemático de Botões e Potenciómetros no Chameleon Mixer

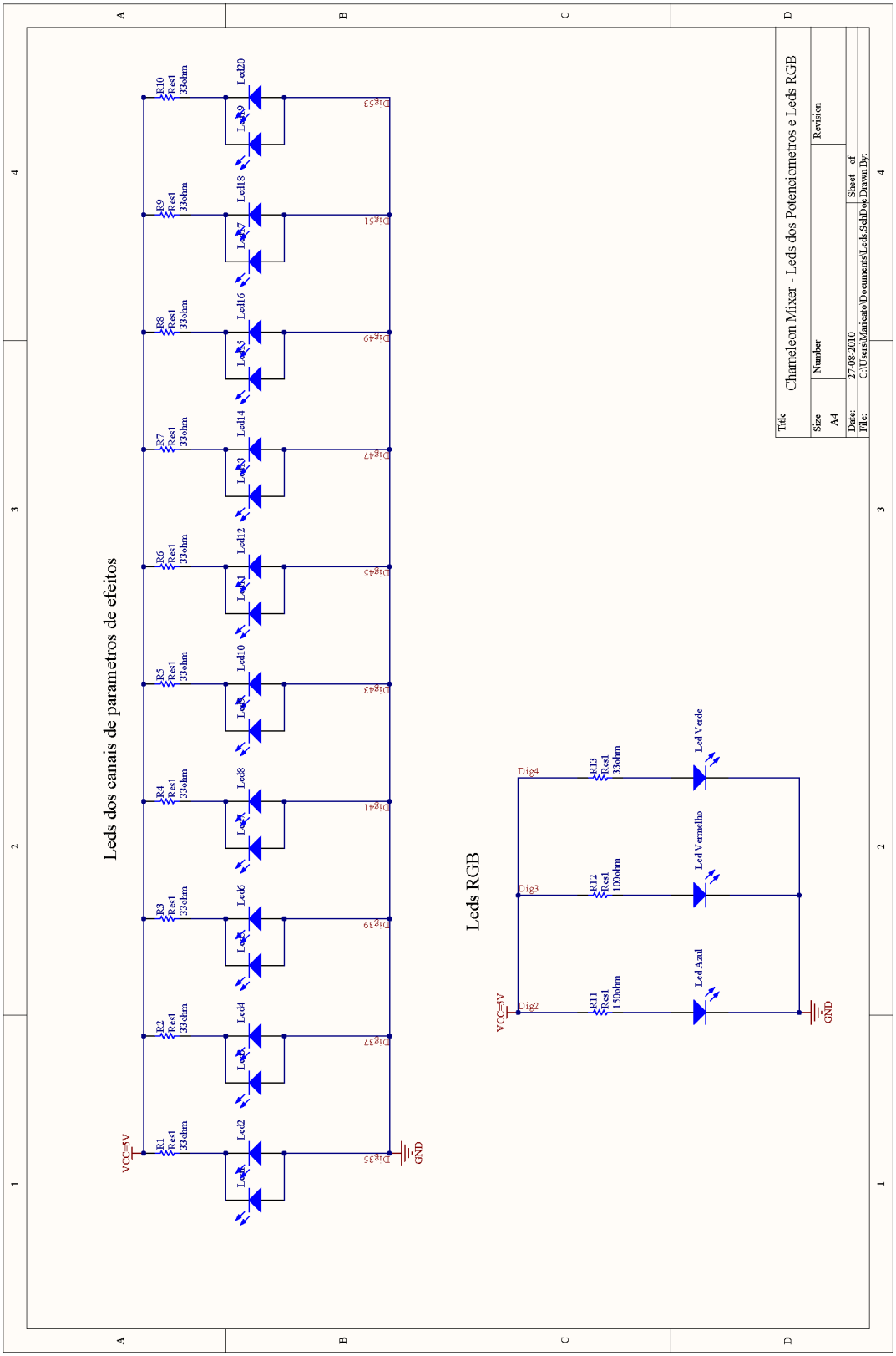


Figura 8.7 – Esquemático dos LEDs no Chameleon Mixer

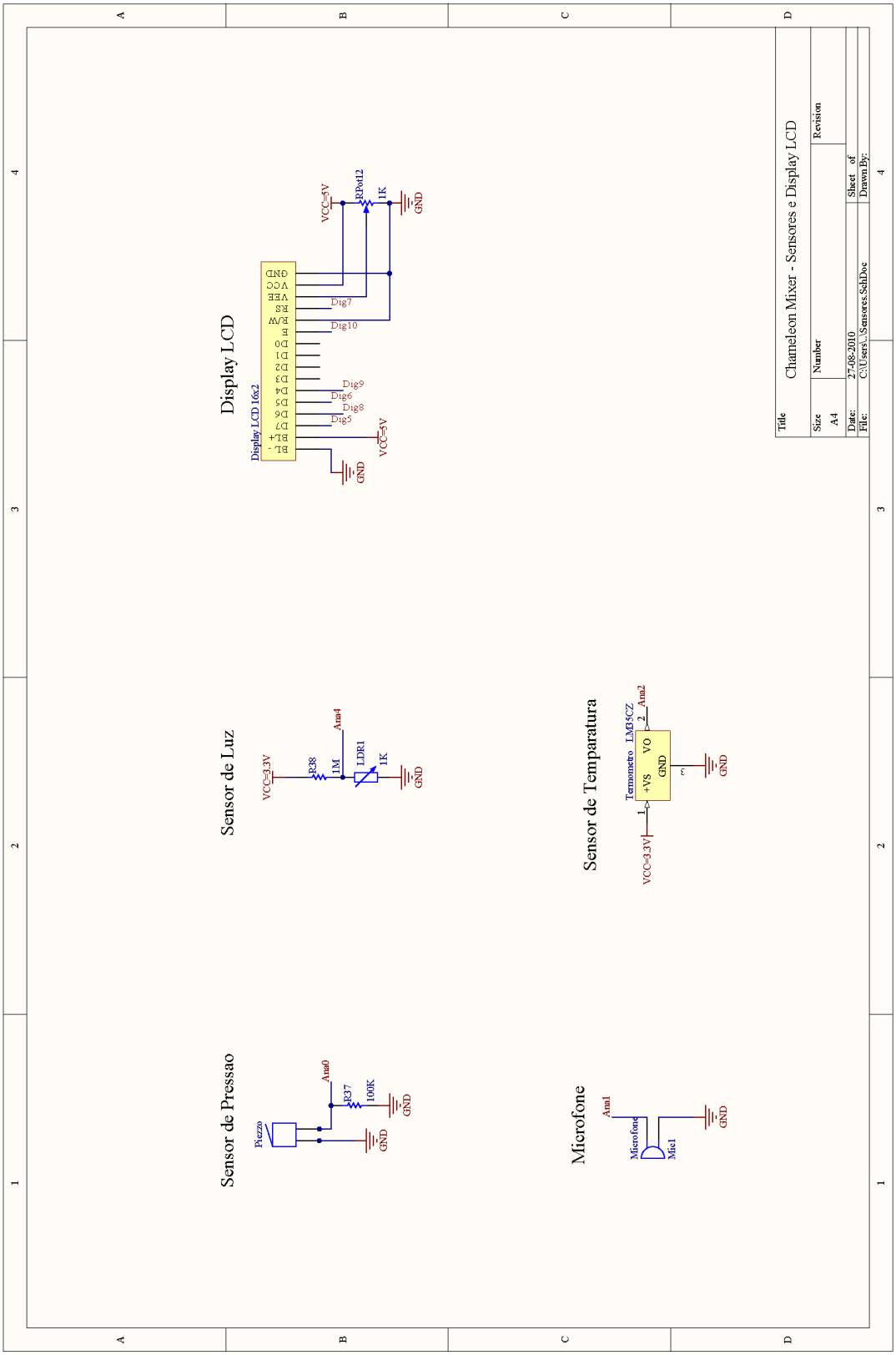


Figura 8.8 – Esquemático dos sensores no Chameleon Mixer

8.5.1.2 Esquema de prototipagem do Chameleon Mixer

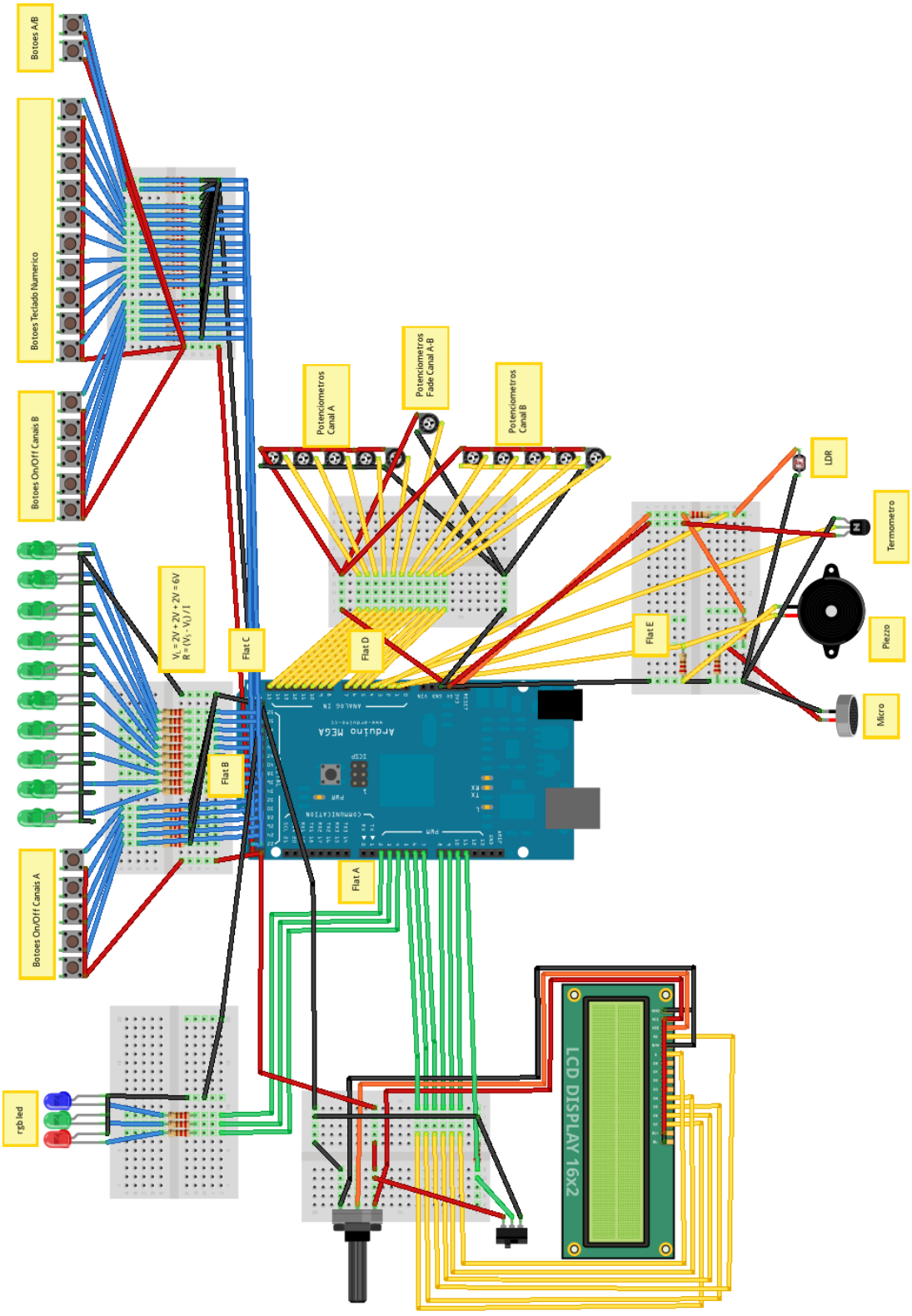


Figura 8.9 - Esquema das ligações electrónicas utilizando o *software* fritzing

8.5.2 Arduino

8.5.2.1 Esquemático do Arduino Mega

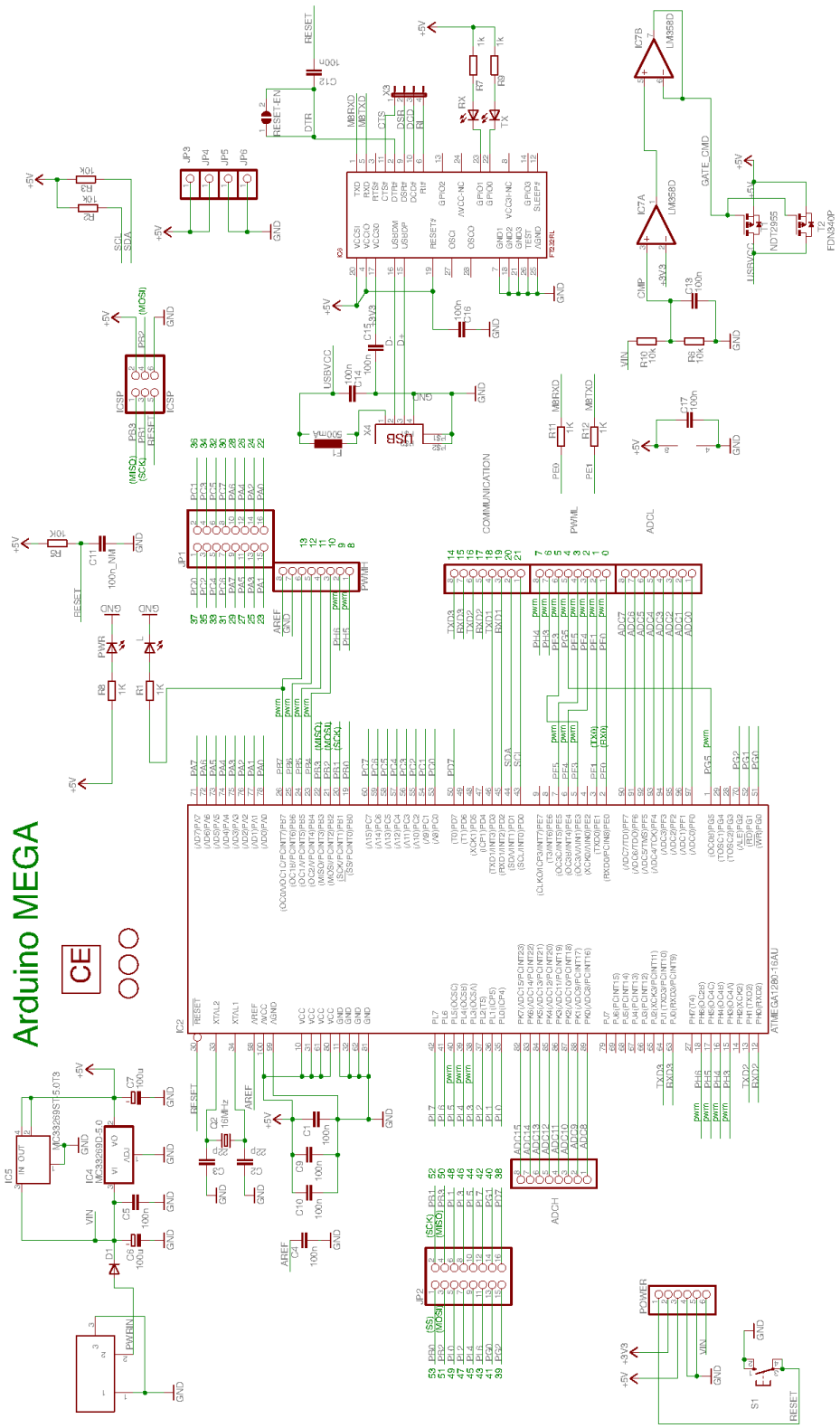


Figura 8.10 – Esquemático do Arduino Mega

Esquemático do Arduino Duemilinove

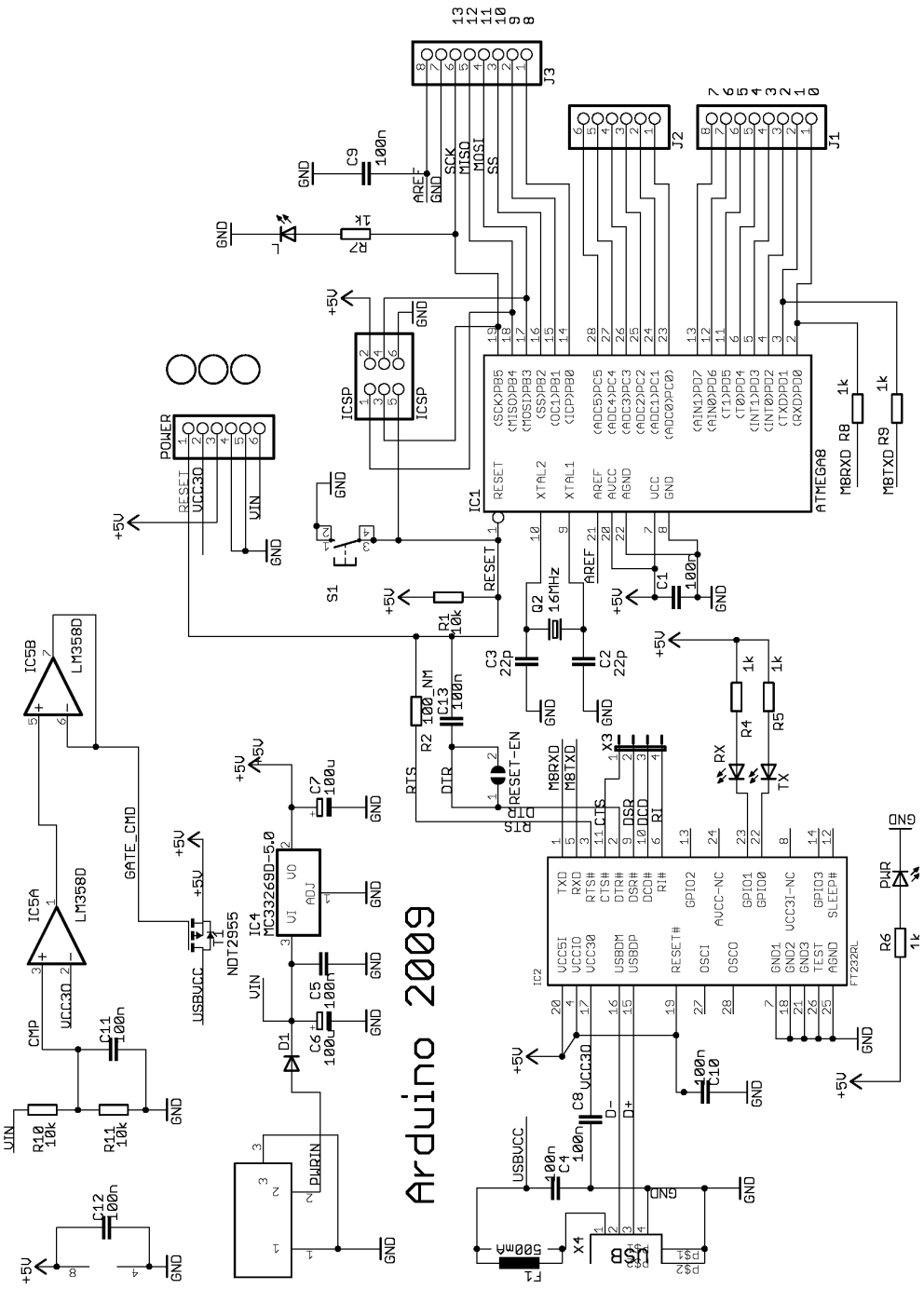


Figura 8.11 – Esquemático do Arduino Duemilinove

Esquemáticos de amplificação

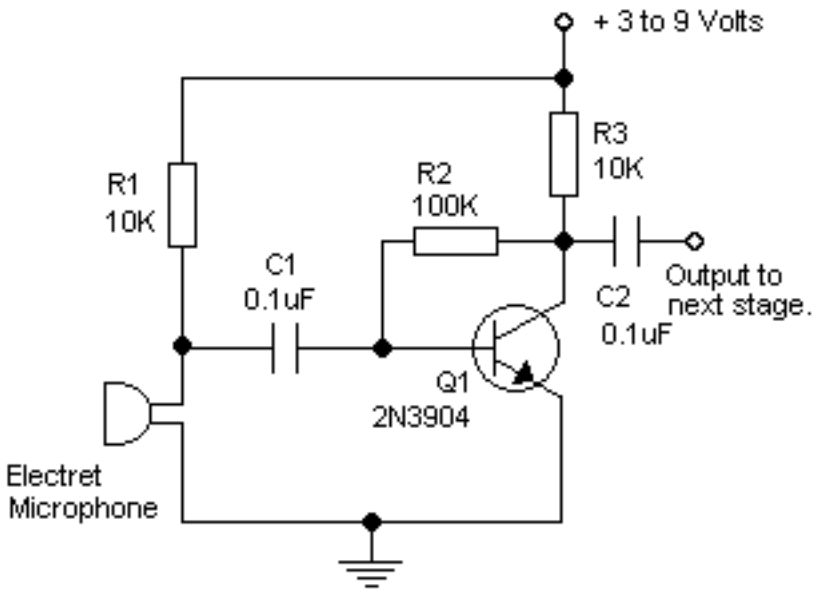


Figura 8.12 – Esquemático de amplificação do piezo [55]

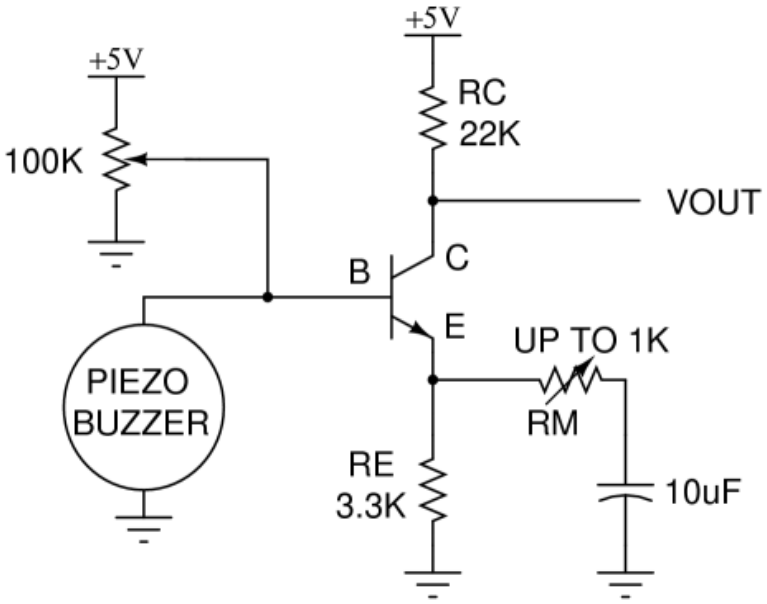


Figura 8.13 - Esquemático de amplificação do piezo [56]

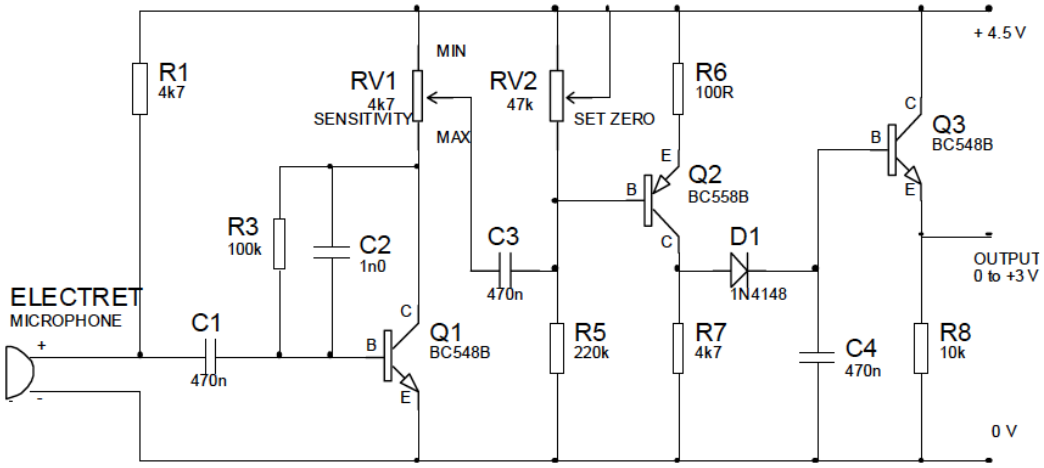


Figura 8.14 - Esquemático de amplificação do piezo [57]

Esquemático para o LCD

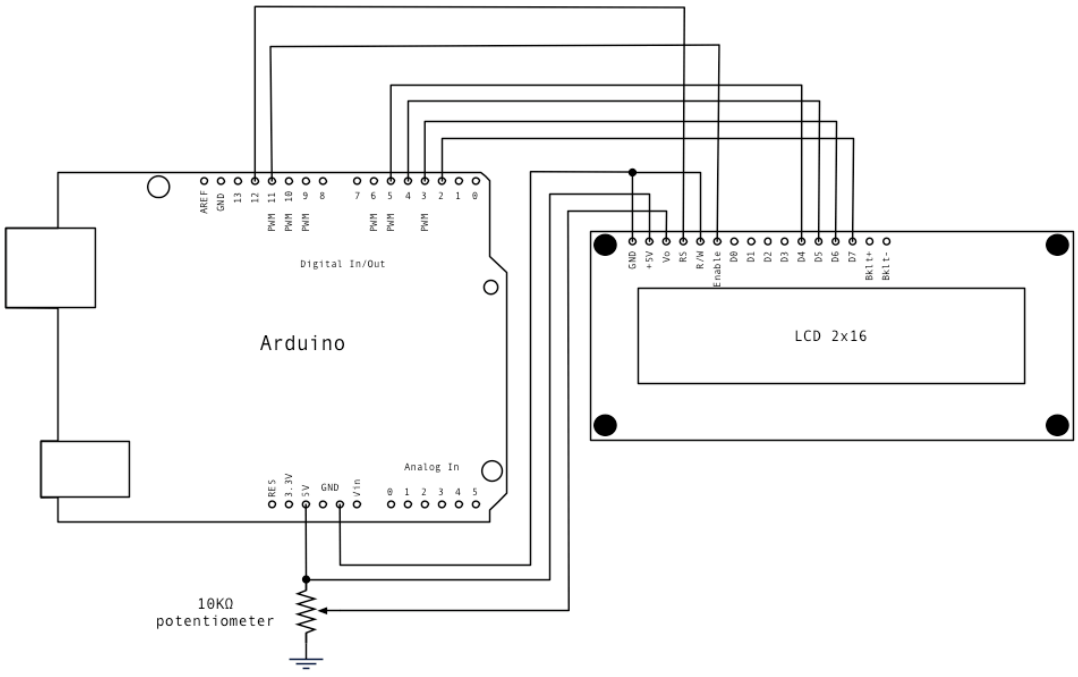


Figura 8.15 - Esquemático do LCD [30]

8.6 Apêndice F – Código do *Arduino*

8.6.1 Pseudo-código construído para o *Firmware*

```
Inicializar os callbacks analógicos
Se Pino analógico é diferente de 3
        Aplica função exponencial valor do potenciómetro
        Envia dados analógicos para o Firmata
Se não
        Se o canal seleccionado é o A
                Envia dados analógicos do efeito do canal A para o Firmata
        Se não
                Envia dados analógicos do efeito do canal B para o Firmata
Volta ao início
```

Pseudo-Código 8.1 – Comunicação Analógica

```
Inicializar os callbacks digitais
Inicializar os Modos de callbacks para diferentes pinos
Inicializar os pinos digitais e ligar os LEDs
Selecção Canal A ou B
Se Estiver modo selecção Efeito
        Mostra o efeito actual LCD e envia informação do modo seleccionar efeito
Se não
        Mostra "Select Mixer" no LCD e envia informação do modo visualização
Se passou tempo de selecção
        Mostra efeito actual do canal Seleccionado
Se ainda está dentro do tempo de selecção no canal ou B
        Junta caracteres do efeito a seleccionar
Envia dados de saída dos pinos digitais
Actualiza o estado on/off dos LEDs
Volta ao início
```

Pseudo-Código 8.2 - Comunicação Digital

Recebe um Pino de entrada

Lê o pino e coloca o valor numa variável temporária

Se o botão não estiver a ser pressionado

    Coloca estado do botão como "solto"

    Coloca soma do tempo "pressionado" a 0

Ou Se o botão estiver preso, mas ainda não passou tempo suficiente

    Incrementa soma de tempo "pressionado"

Ou Se o botão estiver preso e o temporizador chega ao final do tempo

    Coloca estado do botão como "preso"

    Coloca tempo de "pressionado" a 0

    Muda o estado do *mixer* de *input* que se está a actualizar

    Envia dados actuais para o *output*

Se não

Coloca o tempo de pressionado a 0

Pseudo-Código 8.3 - Temporizador para pressionar os botões

```
Recebe um botão do teclado numérico
Se o botão não está pressionado
        Coloca estado do botão como "solto"
        Coloca soma de tempo "pressionado" a 0
Se botão está pressionado, com o botão "preso", mas o tempo ainda não é suficiente
        Incrementa tempo de "pressionado"
Se botão está pressionado, com o botão "preso", e o tempo chega ao final
        Inicia temporizador para marcação de efeito
        Enquanto o temporizador não chegar ao fim e uma tecla for carregada
                Coloca o estado do botão como "preso"
                Coloca o tempo de "pressionado" a 0
                Vê se o canal seleccionado é o A ou o B
                Coloca o estado como "solto"
                Se o efeito é seleccionado a 1ª vez ou é composto por 3 algarismos
                        Iguala o efeito actual à próxima tecla pressionada
                        Reinicia temporizador
                Ou Se o efeito é menor que 10
                        Multiplica o número actual por 10 e soma o novo algarismo
                        Reinicia temporizador
                Ou Se o efeito é menor que 100
                        Multiplica o número actual por 10 e soma o novo algarismo
                        Reinicia temporizador
        Se não
                "Decrementa" o temporizador
```

Pseudo-Código 8.4 - Criação de efeito com o teclado numérico