

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE ELETRÔNICA /INFORMÁTICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

EDMAR ANDERSON LANES JUNIOR

FELIPE MICHELS FONTOURA

LUCAS LONGEN GIOOPPO

MARLON SUBTIL MARÇAL

WILLIAM HITOSHI TSUNODA MEIRA

FUNICON - PRANCHA ELETRÔNICA

MONOGRAFIA

CURITIBA, PARANÁ

2010

EDMAR ANDERSON LANES JUNIOR
FELIPE MICHELS FONTOURA
LUCAS LONGEN GIOOPPO
MARLON SUBTIL MARÇAL
WILLIAM HITOSHI TSUNODA MEIRA

FUNICON - PRANCHA ELETRÔNICA

Monografia apresentada a Unidade Curricular de Oficina de Integração III do Curso de Engenharia da Computação da Universidade Tecnológica Federal do Paraná.

prof. João Alberto Fabro
prof. Heitor Silvério Lopes

CURITIBA, PARANÁ
2010

RESUMO

JUNIOR, Edmar; FONTOURA, Felipe; GIOOPPO, Lucas; MARÇAL, Marlon; MEIRA, William. FunIcon – Prancha Eletrônica. Monografia – Curso de Engenharia da Computação, Universidade Tecnológica Federal do Paraná. Curitiba, 2010.

Este projeto teve como objetivo a montagem de um protótipo de prancha eletrônica com quatro ícones feitos de matrizes de LEDs monocromáticas, cada qual associada a um botão que, ao ser pressionado, dispara a emissão de um som. Os ícones e sons correspondentes da dita prancha podem ser programados pelo usuário final através de um software de interface gráfica, que se comunica com a prancha através de uma interface USB.

Palavras-chave: prancha eletrônica, matriz de LEDs, memória flash, comunicação serial, USB.

ABSTRACT

JUNIOR, Edmar; FONTOURA, Felipe; GIOOPPO, Lucas; MARÇAL, Marlon; MEIRA, William. FunIcon – Electronic Board. Monograph – Computer Engineering, Universidade Tecnológica Federal do Paraná. Curitiba, 2010.

This project objective is the of assembling an electronic board prototype with four icons made of monochromatic LED matrices, each associated to a button which, once pressed, plays a sound. The icons and corresponding sounds of such board can be programmed by the end user in a graphical interface software which communicates with the board through an USB interface.

Keywords: electronic board, LED matrices, flash memory, serial communication, USB.

LISTA DE FIGURAS

Figura 1: Casos de uso do programa da estação base.....	43
Figura 2: Diagrama de classes do programa da estação base.....	45
Figura 3: Diagrama de objetos do programa da estação base (janela principal).....	46
Figura 4: Diagrama de objetos do programa da estação base (janela de edição de botão).....	47
Figura 5: Diagrama de sequência do caso de uso "editar ícone"	48
Figura 6: Diagrama de sequência do caso de uso "importar ícone"	49
Figura 7: Diagrama de sequência do caso de uso "gravar som"	50
Figura 8: Diagrama de sequência do caso de uso "importar som"	51
Figura 9: Diagrama de sequência do caso de uso "adicionar botão à prancha".....	52
Figura 10: Diagrama de sequência do caso de uso "remover botão da prancha".....	53
Figura 11: Diagrama de sequência do caso de uso "carregar arquivo de prancha" ..	54
Figura 12: Máquina de estados do firmware.....	55
Figura 13: Camadas do sistema de transmissão.....	56
Figura 14: Estrutura de uma mensagem.....	57
Figura 15: Estrutura de uma mensagem de estabelecimento de conexão.....	58
Figura 16: Estrutura de uma mensagem de abortar da estação base.....	58
Figura 17: Estrutura de uma mensagem de abortar da prancha.....	59
Figura 18: Estrutura de uma mensagem de negação de envio.....	59
Figura 19: Estrutura de uma mensagem de permissão de envio.....	59
Figura 20: Estrutura de uma mensagem de envio de dados.....	60
Figura 21: Estrutura de uma mensagem de confirmação de segmento.....	60
Figura 22: Estrutura de uma mensagem de término de gravação.....	61
Figura 23: Máquina de estados do computador.....	62
Figura 24: Diagrama de blocos do circuito.....	63
Figura 25: Matriz de LEDs HS-2088AS.....	64
Figura 26: Rede R2R para 4 bits.....	67
Figura 27: Circuito de conversão de dados em sinais.....	69
Figura 28: Circuito relativo à conversão digital analógica e amplificação do som.	70
Figura 29: Diagrama do circuito.....	73
Figura 30: Mensagem de inicialização do programa.....	78

Figura 31: Janela principal do programa da estação base “FunIconGUI”.....	79
Figura 32: Janela de edição de botão do programa da estação base.....	81
Figura 33: Janela de seleção da porta serial.....	82
Figura 34: Mensagem de erro - nenhuma porta serial disponível.....	82
Figura 35: Janela que indica a progressão da transmissão dos dados.....	83
Figura 36: Placa do sistema de som.....	87
Figura 37: Diagrama da placa do microcontrolador	88
Figura 38: Foto da placa do microcontrolador montada.....	88
Figura 39: Foto da Placa de Conexão.....	90
Figura 40: Diagrama da placa de memórias.....	91
Figura 41: Foto da placa das memórias.....	91
Figura 42: Diagrama da placa da matriz.....	93
Figura 43: Foto da placa da matriz.....	94
Figura 44: Sistema integrado - quatro matrizes.....	95

LISTA DE TABELAS

Tabela 1: Comparação das características respectivas às linguagens de programação no computador.....	21
Tabela 2: Comparação das características respectivas a forma de comunicação.....	23
Tabela 3: Taxa de transmissão de dados de cada porta.....	24
Tabela 4: Comparação das características respectivas às linguagens de programação da biblioteca de comunicação pela porta serial.....	25
Tabela 5: Comparação das características respectivas aos microcontroladores.	27
Tabela 6: Comparação das características respectivas às linguagens de programação embarcada.....	30
Tabela 7: Comparação das características respectivas às linguagens de programação embarcada.	31
Tabela 8: Pinagem da matriz de LEDs.....	65
Tabela 9: Resumo da pinagem: linhas da Matriz de LEDs.....	65
Tabela 10: Resumo da pinagem: colunas da Matriz de LEDs.....	65
Tabela 11: Seletor de dispositivo.....	66
Tabela 12: Estimativa de custos.....	98
Tabela 13: Tabela de gastos.....	98

SUMÁRIO

1 INTRODUÇÃO.....	10
1.1 DECLARAÇÃO DE ESCOPO.....	11
1.2 ESPECIFICAÇÃO DE OBJETIVOS E METAS.....	12
1.3 PREMISSAS E RESTRIÇÕES.....	13
1.4 DESIGNAÇÃO DO GERENTE E DA EQUIPE.....	14
2 REFERÊNCIAS A TRABALHOS SEMELHANTES.....	15
2.1 PRANCHA DE COMUNICAÇÃO COM PREDIÇÃO DE SÍMBOLOS.....	15
2.2 JOGO ELETRÔNICO SIMON.....	15
3 REQUISITOS.....	17
3.1 REQUISITOS DA ESTAÇÃO BASE (COMPUTADOR).....	17
3.2 REQUISITO DA COMUNICAÇÃO.....	17
3.3 REQUISITO DO BOTÃO E ÍCONE.....	18
3.4 REQUISITO DE SOM.....	18
3.5 REQUISITO DA PRANCHA ELETRÔNICA OU ARTEFATO.....	18
3.6 REQUISITO DE SOFTWARE.....	19
4 ESTUDO DE ALTERNATIVAS TECNOLÓGICAS.....	20
4.1 ESTAÇÃO BASE.....	20
4.1.1 Linguagem de programação no computador.....	20
4.2 COMUNICAÇÃO.....	22
4.2.1 Meio de comunicação dos dados.....	22
4.2.2 Linguagem da biblioteca de comunicação.....	24
4.3 SISTEMA EMBARCADO.....	25
4.3.1 Microcontrolador.....	26
4.3.2 Linguagem de programação embarcada.....	29
4.3.3 Memória de armazenamento.....	31
5 PLANEJAMENTO DE RISCOS.....	32
6 PLANEJAMENTO.....	42
6.1 SOFTWARE.....	42
6.1.1 Software de interface.....	42
6.1.2 Software do sistema embarcado.....	54
6.2 PROTOCOLO DE COMUNICAÇÃO.....	56
6.2.1 Mensagens de aplicação.....	57
6.2.2 Máquina de estados.....	61
6.3 HARDWARE.....	63
6.3.1 Sistema de imagem.....	63
6.3.2 Circuito seletor.....	66
6.3.3 Sistema de armazenamento.....	66
6.3.4 Sistema de som.....	67
6.3.5 Microcontrolador.....	70
7 CRONOGRAMA	76
8 DESENVOLVIMENTO PRÁTICO.....	77
8.1 ESTAÇÃO BASE.....	77
8.1.1 Software de interface	77
8.1.2 Software do sistema embarcado.....	83

8.2 PRANCHA.....	85
8.2.1 SISTEMA DE SOM.....	85
8.2.2 PLACA DO MICROCONTROLADOR.....	87
8.2.3 PLACA DE CONEXÃO.....	89
8.2.4 PLACA DAS MEMÓRIAS.....	90
8.2.5 PLACA DA MATRIZ DE LEDS E DO BOTÃO.....	92
8.3 SISTEMA INTEGRADO.....	94
8.4 RISCOS PLANEJADOS OCORRIDOS.....	96
8.5 CUSTO DO PROJETO.....	96
9 CONCLUSÃO.....	99
REFERÊNCIAS BIBLIOGRÁFICAS.....	100

1 INTRODUÇÃO

As crianças, no decorrer do crescimento, descobrem o mundo ao seu redor associando sons a imagens e aprendendo, gradativamente, como representar ou nomear tudo que as cerca. Os brinquedos são importantes aliados neste reconhecimento do ambiente pela criança. Cada idade apresenta novos desafios a serem transpostos: conhecer as formas, os objetos, o som e o nome de cada animal, o som dos instrumentos ou mesmo as notas musicais. Para cada uma destas descobertas, um ou mais brinquedos podem acompanhar a criança sob a supervisão de um adulto, facilitando o aprendizado.

Alguns brinquedos têm como o intuito o desenvolvimento de certas habilidades. Jogos que trabalham reflexos, memorização visual ou sonora e coordenação motora se prestam para este fim e em alguns casos podem ser utilizados para tratamento de crianças com dificuldades de movimentação condicionadas por alguma deficiência mental ou física.

Pode se inferir, então, que para cada idade ou situação existem brinquedos mais apropriados, o que gera uma necessidade de compra de acordo com o desenvolvimento da criança. É aí que se faz interessante um equipamento que auxilia no aprendizado infantil e que possa ser adaptado à fase atual da criança. Algo que para uma idade, possa ajudar no reconhecimento das formas e, posteriormente, nas letras do alfabeto e alguns fonemas da língua portuguesa.

A ideia deste projeto é desenvolver uma ferramenta que consiste em uma prancha com botões. Estes possuem um ícone que ao ser pressionado emite um som correspondente. Uma criança aprendendo a ler, descobrindo ainda os fonemas de cada letra poderá utilizar a prancha configurada para este exercício, outro exemplo é para aprender música com sons de notas musicais ou ainda aprender o nome de formas geométricas.

Existem algumas outras aplicações para este dispositivo diferente da área de ensino. Como auxiliar deficientes físicos ou mentais em suas atividades cotidianas como pedir água ou ir ao banheiro. O som possibilita chamar a atenção de outras pessoas por perto para atenderem ao seu pedido.

1.1 DECLARAÇÃO DE ESCOPO

O ideia do projeto consiste em um conjunto de botões que ao serem pressionados emitam sons correspondentes ao seu ícone e que podem ser programados em uma estação base, no entanto funcionando em separado dela.

Cada botão será formado por uma matriz 8x8 de LEDs (*light emitting diode*) monocromáticos programada pela estação de maneira a representar uma imagem ilustrativa. A conexão entre a estação base e o protótipo será realizada por um cabo que utiliza a porta USB (*universal serial bus*) do computador quando se deseja reconfigurar a prancha (sons e ícones).

A alimentação deverá ser por bateria evitando fios que comprometeriam a mobilidade e trariam perigos no momento em que a criança utiliza o equipamento. A estação base é um microcomputador pessoal que tenha basicamente compatibilidade com o sistema de conexão escolhido e um programa de fácil operação para que seja feita a configuração da prancha.

Tem que ser ressaltado que este projeto não aspira à elaboração de um produto, contudo, apenas um protótipo funcional para demonstrar o conceito que poderá ser empregado, posteriormente, na elaboração de algo comercializável.

Não é foco do projeto, mas será buscada uma extensibilidade de forma que ele, posteriormente, possa agregar outros usos. Um deles seria como uma plataforma de comunicação para indivíduos com dificuldade de comunicação, de modo que, com os botões devidamente configurados, elas possam associar a imagem ao que desejam. Outra possibilidade de extensão futura é trabalhar como forma de realizar pedidos remotamente em um bar ou restaurantes: uma espécie de cardápio interativo customizado pelo cliente cadastrado no sistema do estabelecimento. O cliente pediria o que deseja pela prancha agilizando o trabalho dos garçons. Para estes projetos poderia ser desenvolvida uma conexão *wireless* da estação-base com o dispositivo já que utilizarão esta ligação em tempo de utilização. Enfim, existe uma gama de possibilidades que não cabem no escopo deste projeto, mas que poderão ser consideradas em trabalhos futuros.

1.2 ESPECIFICAÇÃO DE OBJETIVOS E METAS

O principal objetivo do projeto é construir um protótipo funcional da prancha eletrônica e o seu software de configuração.

A prancha deverá ter botões com ícones representados através de matrizes de LEDs que interagem com o usuário quando pressionados emitindo sons (um para cada matriz), configurados de maneira simples através de um *software* facilmente utilizável por usuários que tenham um conhecimento básico de informática.

As imagens e o áudio poderão ser configurados por qualquer usuário através do *software*, onde ele poderá selecionar alguns botões (ícone mais som) de uma base de dados já existente no programa ou criar seus próprios botões desenhando ponto a ponto (correspondente a cada LED da matriz) e gravando um pequeno arquivo de som caso possua um dispositivo de gravação de áudio no seu computador.

Para que o projeto tenha um real sucesso algumas metas devem ser almejadas: espera-se que o custo de desenvolvimento do protótipo não seja elevado, não poderá ultrapassar muito o valor estimado. Outra importante meta a ser cumprida é referente ao prazo de desenvolvimento do mesmo que deverá durar cerca de 2 meses e meio, 10 semanas.

Quanto ao *hardware* nossas metas são: construir quatro botões, o som deverá ter um fator de amostragem mínima de oito mil amostras por segundo, a mensagem gravada deve ter uma duração de até 4 segundos, a prancha após receber o comando de atualização (mudança no áudio e nos ícones) não deverá demorar mais do que 1 minuto para que ela possa ser utilizada novamente, os ícones apresentados devem ter uma resolução mínima de 8x8, os botões devem ser suficientemente grandes para serem facilmente pressionados por uma criança, os botões não podem ser rígidos (quanto à força necessária para que eles se desloquem para baixo).

Quanto ao Software nossas metas são: seja possível reconfigurar os botões, consiga enviar as atualizações para a prancha, seja possível desenhar um novo ícone para a prancha ponto por ponto (no caso utilizando uma resolução mínima de 8 por 8), seja capaz de gravar um som de áudio de até 4 segundos, possua alguns

exemplos de ícones e sons.

O projeto também tem como uma de suas metas a utilização de tecnologias acessíveis, de custo não muito elevado e que podem ser encontradas sem grandes dificuldades no mercado, para que o mesmo possa ser futuramente expandido para outras aplicações.

1.3 PREMISSAS E RESTRIÇÕES

Premissas: referem-se às coisas que devemos ter preparado para começarmos o desenvolvimento do projeto.

- Botões que possam ser facilmente utilizados pela criança (usuário direto): 4 para as matrizes e 1 direcionado à funcionalidade de liga/desliga.
- 4 matrizes de LEDs monocromáticos (testadas e funcionando), com resolução mínima de 8x8 LEDs.
- Circuitos Integrados (testados e funcionando), junto com os respectivos datasheets.
- Material para o estudo e aprendizado específico de conceitos de áreas diretamente envolvidas no projeto (Circuitos Digitais, Microcontroladores, Comunicação de Dados, Linguagens de Programação).
- Escopo bem definido.
- Requisitos bem definidos.
- Especificações técnicas bem definidas.

Restrições: referem-se às funcionalidades que **não** estarão envolvidas no artefato final.

- A prancha não será direcionada a deficientes físicos (apesar de ser extensível para tal funcionalidade).
- A prancha não será direcionada para o uso em restaurantes e bares (apesar de ser extensível para tal funcionalidade).
- A prancha não irá se comunicar com a estação base durante o seu uso.
- A prancha de maneira alguma substitui a integridade de todo o processo de aprendizado de uma criança. Ela tem como escopo atuar apenas como auxiliador no

processo de aprendizagem, tendo, o artefato, um cunho de entretenimento educativo para a criança.

1.4 DESIGNAÇÃO DO GERENTE E DA EQUIPE

Gerente: William Hitoshi Tsunoda Meira

Desenvolvedores: Edmar Anderson Lanes Júnior, Felipe Michels Fontoura, Lucas Longen Gioppo, Marlon Subtil Marçal.

2 REFERÊNCIAS A TRABALHOS SEMELHANTES

Comparação do projeto Fun-Icon com outros projetos semelhantes encontrados.

2.1 PRANCHAS DE COMUNICAÇÃO COM PREDIÇÃO DE SÍMBOLOS

A visão do protótipo de brinquedo tratado neste projeto adveio de um equipamento utilizado por pessoas portadoras de deficiências que dificultem a comunicação, denominado prancha de comunicação alternativa. Em seu modelo tradicional, esta prancha é uma espécie de cartão com figuras, para as quais o usuário aponta, indicando demandar algo relativo à figura apontada. Mas a ideia que mais se aproxima do nosso projeto foi apresentada no artigo “Prancha de Comunicação com Predição de Símbolos” [1] de Laudelino Cordeiro Bastos e Denis Hattori Matias, que comenta um projeto em implementação de software do equipamento supracitado para ser utilizado em um computador pessoal com capacidade de emitir sons relativos a cliques nas figuras, dentre outras funções. Mas, evidentemente, existem divergências abruptas entre este projeto e o discutido neste documento, pois o que foi apresentado por Matias e Bastos não é embarcado, possui sons gerados por programas de computador ao invés de gravados pelo usuário e agrupa diversas funcionalidades não presentes na Fun-Icon, como um botão “Avançar” para trabalhar com mais figuras.

2.2 JOGO ELETRÔNICO SIMON

Se trata de um artefato eletrônico em que se disponibiliza determinados estímulos visuais e sonoros para o usuário, sendo que a ordem em que esses estímulos são disponibilizados dependem das ações que usuário toma com relação à interface do sistema microcontrolado além do estado atual do sistema. Apesar de não haver a possibilidade de, via uma estação base, personalizar os ícones visuais e os estímulos sonoros, os quais são pré-programados no projeto e na implementação do sistema, o artefato SIMON [2] se assemelha muito ao nosso projeto sob o ponto de vista de um usuário final. Pode-se dizer que os elementos finais de SIMON são

mais simples que os empregados em nosso projeto, pois nele: em termos visuais disponibiliza-se cor apenas pela ativação ou não ativação de um LED específico (não imagens em matrizes de LEDs); em termos sonoros disponibiliza-se sons de frequência constante (não sons gravados e armazenados). No nosso caso, para o elemento visual faremos uma varredura de frequência o suficientemente alta em um dos eixos da matriz de LEDs enquanto que no outro eixo iremos dispor as entradas específicas com a mesma frequência da varredura. Quanto ao elemento sonoro, iremos converter um código digital em um sinal analógico através de um conversor D/A (ao contrário da abordagem do projeto SIMON, onde o microprocessador já tem encapsulada uma saída analógica em um de seus pinos de saída). No projeto FunIcon iremos armazenar em uma memória tanto os dados relativos à imagem quanto os dados relativos ao som, além dos dados referentes ao estado atual da aplicação corrente.

3 REQUISITOS

Neste tópico serão descritos os requisitos e as respectivas especificações técnicas do projeto a ser desenvolvido.

Os requisitos são condições ou capacidades impostas ao produto e/ou ao seu desenvolvimento. E, especificações ou características técnicas são descrições rigorosas e detalhadas de um determinado equipamento ou material que o produto deverá ter.

Para o projeto os requisitos foram divididos em seis categorias: estação base, comunicação, botão e ícone, som, artefato (prancha) e software.

3.1 REQUISITOS DA ESTAÇÃO BASE (COMPUTADOR)

- Deverá ter instalado o sistema operacional Windows versão XP ou posterior.
- Deverá ter porta USB.
- Deverá ter um hardware instalado que grave som, como um microfone.

Especificação técnica: o computador deverá ter pelo menos uma porta USB para conectar o cabo de comunicação e deverá possuir um hardware como um microfone para que possa gravar sons caso necessite reconfigurar a prancha.

3.2 REQUISITO DA COMUNICAÇÃO

- Deverá ter uma velocidade de transmissão mínima de 2000 bits por segundo.
- Deverá ser bidirecional, ou seja, possibilite enviar mensagens do computador para prancha e da prancha para computador.
- A prancha ao ser reconfigurada deverá ser atualizada em um tempo limite de 1 minutos.
- A comunicação não terá tratamento de falhas, dado o baixo nível de ruído da interface serial.

Especificação técnica: para a transmissão de mensagens é necessário existir uma comunicação entre o receptor e o emissor e por este motivo deve ser bidirecional. A velocidade deverá suficientemente grande para que seja possível gravar todos os dados do computador para a prancha em menos de um minuto. Será utilizado o protocolo de comunicação serial, porém por ser uma porta que está em desuso optamos por utilizar um adaptador USB-Serial embutido e assim utilizar a porta USB.

3.3 REQUISITO DO BOTÃO E ÍCONE

- Deverá ter 4 botões e eles devem ser monocromáticos.
- Os botões deverão ter um tamanho superior a 6x6 centímetros e menores que 15x15 centímetros.
- Deverá ter um sistema de botões que não necessite que o usuário exerça muita força para pressioná-lo e que seja pressionado na vertical.

Especificação técnica: quatros matrizes de LED's 8x8 monocromáticos atualizados por varredura. Quatro sistemas botão-mola, tendo a mola um pequeno coeficiente de rigidez.

3.4 REQUISITO DE SOM

- O som deve ser emitido apenas quando o botão correspondente for pressionado.
 - O som emitido deverá ser superior a 30 dB de intensidade sonora.
 - O som deverá ter no máximo quatro segundos de duração.

Especificação técnica: na saída do som deverá ter um amplificador que consiga obter mais de 30 dB. Qualidade de som de 8 mil amostras por segundo. A duração do som deverá ser limitada pelo software e configurada, exatamente, para o botão que se deseja.

3.5 REQUISITO DA PRANCHA ELETRÔNICA OU ARTEFATO

- Não deverá ser alimentado pela rede elétrica.

- O som e o botão devem ser reconfiguráveis.
- Deverá guardar na memória não volátil os sons e os ícones dos botões mesmo após ter sido desligado.
- Deverá sinalizar quando a bateria estiver acabando.

Especificação técnica: A prancha utilizará uma bateria e um circuito de detecção de bateria descarregada. O software deverá, através da comunicação, conseguir configurar os sons e os ícones dos botões da prancha. O sistema deve armazenar o mapeamento dos ícones e os sons em uma memória de no mínimo 32 KB para cada botão. Os sons em um bloco de memória separada para cada um. Para controlar o sistema um microcontrolador com 20 portas de entrada/saída e que execute em média 0,1 MIPS (Milhões de Instruções por segundo). Na alimentação serão utilizados 5 pilhas do tipo AA de 1,2 V em série.

3.6 REQUISITO DE SOFTWARE

- Deverá ser compatível com o sistema Windows versão XP ou posteriores.
- Deverá conseguir se comunicar com a prancha quando conectado com a estação base.
- Deverá conseguir configurar a prancha.

Especificação técnica: Software desenvolvido em linguagem compatível com a plataforma. Interface intuitiva para o usuário inexperiente.

4 ESTUDO DE ALTERNATIVAS TECNOLÓGICAS

Analisar as opções tecnológicas permite a equipe identificar possíveis opções e descobertas de novas ferramentas mais apropriadas para resolver um determinado problema. Além de fornecer um plano alternativo ao grupo em caso de falha da primeira diminuindo os riscos do projeto.

A partir dos requisitos foi levantado um estudo sobre as alternativas tecnológicas de três grupos do projeto: estação base, sistema de comunicação e sistema embarcado. Sendo alguns divididos em subgrupos quando necessário.

4.1 ESTAÇÃO BASE

A estação base é o computador e nele deverá estar instalado o software que será utilizado para reconfigurar à prancha eletrônica. A estação e o software deverão atender aos requisitos previstos.

As tecnologias estudadas para esta etapa envolvem a linguagem utilizada para desenvolver o software que faz a interface entre o usuário e a prancha. Ela está dividida na linguagem de programação para a interface gráfica e linguagem de programação da biblioteca de comunicação.

4.1.1 Linguagem de programação no computador

O software de interface no computador deve permitir a edição de modelos de botão, ou seja, dos ícones e sons associados a cada botão em particular, o que inclui a gravação ou importação de sons, e a edição ou importação dos ícones. Ele deve também permitir a colocação desses modelos de botão em modelos de prancha, ou seja, conjuntos de modelos de botão a serem transferidos para a prancha. Finalmente, deve permitir a transferência de modelos de prancha para a prancha em si através da porta USB, respeitando o protocolo de comunicação elaborado. Há ainda a necessidade de armazenar e carregar esses modelos no computador.

Há algumas alternativas para sua confecção: programá-lo em linguagem C ou C++, em linguagem Java, ou mesmo criar um sistema cliente-servidor, de forma que o software seja utilizado no navegador. Outras alternativas, como linguagem Python ou C# foram descartadas pelo fato de nenhum membro da equipe conhecê-las em profundidade, e não ser possível estudá-las no intervalo de tempo proposto. Apresenta-se um resumo da comparação dos critérios na tabela 1.

Critério / Linguagem de Programação no Computador	Cliente-servidor	C/C++	Java
Independência do S.O.	Alta	Baixa	Alta
Número de programas	2	1	1
Eficiência	Baixa	Alta	Média
API gráfica	Sim (navegador)	Não	Sim
Documentação	Sim	Sim	Sim
Familiaridade	Média	Alta	Alta

Tabela 1: Comparação das características respectivas às linguagens de programação no computador

A alternativa cliente-servidor necessitaria da criação de dois programas (um servidor e uma interface), em linguagens diferentes. O servidor poderia ser criado em PHP ou em JSP, e a interface com o cliente em HTML. Alguns membros da equipe têm familiaridade com PHP, outros com JSP, mas todos têm certo grau de conhecimento a respeito de HTML.

A programação de interfaces em linguagens C ou C++ é potencialmente complexa, e muito dependente de bibliotecas externas. O processo de programação de uma interface utilizando essas linguagens envolverá, entre outras coisas, a aprendizagem do uso de bibliotecas gráficas. Atualmente, todos os membros da equipe têm familiaridade com linguagens C e C++, e apenas alguns membros da equipe têm conhecimentos superficiais a respeito do uso da biblioteca gráfica como Allegro e QTCreator.

A linguagem Java apresenta-se como uma alternativa para a confecção da interface gráfica. Java apresenta uma API simples e bem documentada para a criação de interfaces, tornando a programação mais simples. Java também tem a vantagem de ser facilmente portável, pouca dependência de SO, os arquivos não

são compilados para linguagem de máquina, e sim são interpretadas por uma máquina virtual, sendo possível usar um mesmo executável em Java em qualquer sistema para o qual haja uma máquina virtual compatível. Os membros da equipe têm grande familiaridade com programação em Java, e alguns têm prática na confecção de interfaces gráficas utilizando essa linguagem.

Comparativamente, a alternativa cliente-servidor foi descartada, pois se apresenta como a mais complexa, e a menos vantajosa; seriam necessários dois programas, levando à demanda de mais tempo. Também se descartou a possibilidade da programação da interface na linguagem C ou C++, devido à dificuldade do uso de bibliotecas gráficas, e pela comparação de suas vantagens com as de Java. Optou-se, portanto, pelo uso de Java na programação da interface com o usuário.

4.2 COMUNICAÇÃO

Nesta parte estão descritos as tecnologias que serão utilizados no sistema de comunicação.

4.2.1 Meio de comunicação dos dados

Segundo os requisitos, a prancha deverá ser configurada pela estação base através de um canal de comunicação bidirecional com taxa de 2000 bps, sendo o tempo de configuração menor ou igual a um minuto. De acordo com essas condições, não é estritamente necessário haver uma conexão constante entre a estação base e a prancha, visto que a comunicação é destinada apenas para configuração das imagens e dos sons correspondentes na prancha. Para isso, utilizaremos uma dentre as seguintes alternativas tecnológicas para transmissão de dados: porta serial, porta paralela, porta USB (via cabo), porta USB (com suporte a *wireless*). Para a análise, tomamos como base um microcontrolador hipotético que possa ter os três tipos de porta, sendo: o conector de saída da porta serial RS232 um conector de 9 pinos macho; o conector de saída da porta paralela um conector de 25 pinos macho ou de 36 pinos macho; o conector de saída da porta USB um conector do tipo macho B. Na tabela 2 está uma comparação entre estas tecnologias.

Critérios / Comunicação	Porta Serial RS232	Porta Paralela	Porta USB (via cabo)	Porta USB (XBee - com suporte a Wireless)
Custos Envolvidos*	R\$ 64,99 (1)	R\$ 40,00 (2)	R\$ 11,50 (3)	R\$ 221,88 (4)
Disponibilidade no Mercado Local	Sim	Sim	Sim	Não
Acessível Economicamente	Sim	Sim	Sim	Não
Suporte a nível de hardware no 8052	Sim	Parcialmente	Não	Sim

Tabela 2: Comparação das características respectivas a forma de comunicação.

* Os preços referidos na tabela tem como referência os valores de tais artefatos disponíveis em websites de comércio eletrônico, como: <http://www.buscape.com.br/>, <http://www.mercadolivre.com.br/> e <http://todaoferta.uol.com.br/>. Nos informamos do preço do kit Xbee no próprio site do fabricante <http://www.digi.com>.

(1) R\$ 64,99 (Adaptador USB Macho tipo A / DB9 Macho com conversor para DB9 Fêmea). (2) R\$ 40,00 (Adaptador USB Macho tipo A x DB25 Fêmea ou DB36 Fêmea). (3) R\$ 11,50 (Macho tipo A x Macho tipo B). (4) R\$ 221,88 (XBEE 802.15.4 Starter Development Kit w/ 2 XBee Modules).

Desconsideraremos o uso do módulo de transmissão *wireless* XBee, por ser inviável economicamente (haveria a necessidade de comprar mais de um kit, para diminuir o risco de queima ou de quebra de componente) e por não estar facilmente disponível. É necessário ressaltar que o tipo de interface que existirá nas duas extremidades dos cabos envolvidos na transmissão depende do tipo de porta de comunicação existente no microcontrolador adotado ou que possa ser facilmente adaptado a ele. Independentemente do tipo de porta existente nós iremos trabalhar com a porta USB na interface da estação base, usando um conversor de porta serial para porta USB ou de um conversor de porta paralela para porta USB. Decidimos empregar a porta USB, especificamente um adaptador USB-serial, devido à facilidade de haver suporte do hardware (8052) à comunicação serial, e porque nem todos os computadores atuais possuem suporte às portas antigas (porta paralela e porta serial), tornando o artefato mais conveniente ao cliente.

Existe a opção de implementar o módulo de rádio frequência (*wireless*), mas a consideramos inviável pois não temos familiaridade o suficiente com o assunto. Isso afetaria negativamente o cronograma, pois teríamos de ter um tempo de

aprendizado do assunto de rádio frequência considerável e aumentaria o orçamento caso utilizássemos o módulo pronto.

Na tabela 3 é comparada a taxa de transmissão de dados de cada porta.

Tipo de Comunicação	Taxa de Transmissão de Dados (bits por segundo)
Porta Serial	19200
Porta USB 2.0	480 M
Porta Paralela EPP	1500000

Tabela 3: Taxa de transmissão de dados de cada porta

Como podemos ver, as taxas de transmissão que cada tipo de porta possui sobrepõem a taxa requerida pela aplicação (2000 bps). Portanto não teremos problemas quanto à taxa de transmissão de dados pelo canal de comunicação.

4.2.2 Linguagem da biblioteca de comunicação

Com o uso de uma interface serial através da porta USB, faz-se necessário o uso de um software que seja capaz de se comunicar com a porta serial. Há poucas alternativas para a programação da biblioteca de comunicação entre o computador e o sistema embarcado, devido a poucas linguagens de programação que permitirem a comunicação com dispositivos externos na qual temos algum conhecimento.

Duas alternativas principais se apresentam para essa aplicação. A primeira refere-se à criação de uma aplicação, em C ou C++, que use as funções nativas dos sistemas operacionais, e receba mensagens de outras aplicações (através de sockets) para o comando da comunicação pela porta serial de forma indireta, lembrando que o adaptador USB-Serial simula uma porta serial no computador. A segunda refere-se ao uso da biblioteca RxTx, para linguagem Java, liberada sob licença LGPL e que trabalha com serial. Uma comparação resumida entre as tecnologias é apresentada na tabela 4.

Critério	RxTx (Java)	Sockets (C/C++)
Familiaridade	Baixa	Baixa
Dependência do S.O.	Nenhuma (Windows, Linux e MAC)	Alta
Facilidade de programação	Alta	Média

Tabela 4: Comparação das características respectivas às linguagens de programação da biblioteca de comunicação pela porta serial.

A alternativa de aplicação em C ou C++ que permita o comando da porta serial remotamente envolve o problema de ser muito dependente do sistema operacional. Também se apresenta a necessidade de ligar a aplicação, em segundo plano, toda a vez que a comunicação se fizer necessária. Os membros da equipe não têm familiaridade com a programação de sockets nem de comunicação serial nessas linguagens.

A alternativa que envolve a biblioteca RxTx, em Java, apresenta a facilidade de ser feita na mesma linguagem que se decidiu utilizar para o software no computador. Também tem a facilidade de ser independente do sistema operacional (desde que a biblioteca RxTx esteja disponível para o mesmo, o que é o caso para a maioria dos sistemas operacionais populares). Há ainda a vantagem da biblioteca ser liberada com a licença LGPL. Após ponderar as vantagens e desvantagens das duas opções apresentadas, decidiu-se usar a segunda alternativa (biblioteca RxTx em Java) na elaboração da biblioteca de comunicação.

4.3 SISTEMA EMBARCADO

O circuito da prancha deve permitir que o microcontrolador comunique-se de forma bilateral com as memórias flash e com os botões. Também é preciso que comunique-se, de forma unilateral, com um conversor analógico-digital, que será utilizado em conjunto com um amplificador para a sintetização do som. Além desses dispositivos, ainda é necessária a confecção de um circuito que verifique a carga da bateria.

A prancha eletrônica deverá atender vários requisitos e existem várias decisões de opções tecnológicas para cada parte do artefato. A análise está dividida

em: microcontrolador, linguagem de programação embarcada e memória para armazenamento.

4.3.1 Microcontrolador

O microcontrolador é um dos principais dispositivos utilizados no projeto, pois através dele é possível uma “fácil” interação entre a estação base (computador) e o sistema embarcado (a prancha). Ele também é o responsável pelo controle dos outros dispositivos empregados no projeto como: as matrizes de LEDs, o emissor de áudio, os dispositivos de armazenamento de dados, os dispositivos de comunicação, dentre outros, funcionando, de maneira análoga, como um “cérebro” para o sistema. Por exercer um papel tão importante no projeto a escolha do microcontrolador deve ser feita de maneira criteriosa para que isso não traga grande impacto tanto no planejamento de custos quanto no planejamento de riscos.

No levantamento de requisitos foram especificadas algumas características necessárias para que o sistema funcione de maneira adequada. Elas foram essenciais para a escolha do microcontrolador que deverá ser utilizado no projeto já que impactam diretamente em alguns traços dele. Depois de feita uma pesquisa de maneira geral, incluindo a facilidade de encontrar tais dispositivos no mercado foram selecionados 3 microcontroladores (Atmel 89S52, ATmega8 e PIC 16) que foram analisados e comparados para ver qual traria um melhor custo benefício para o projeto, contando não somente custo financeiro mas também o quanto a utilização de tal microcontrolador irá impactar no possível tempo de aprendizado do mesmo, a tabela 5, abaixo, apresenta diversos critérios comparando os 3 microcontroladores.

Critério/ Microcontrolador	Atmel 89S52	ATmega8	PIC16F
Clock	0-33 MHz	1-16 MHz (externo) 8 MHz (interno)	4 MHz (interno)
MIPS executados na média.	1,4 Milhões de Instruções por seg. (CK de 24 MHz)	7,85 Milhões de Instruções por seg. (CK de 8 MHz)	0,85 Milhões de Instruções por seg. (CK de 4 MHz).
Portas de I/O	32 pinos de I/O	23 pinos de I/O	16 pinos de I/O
Timers	3 timers de 16 Bits	2 timers de 8 Bits e 1 de 16 Bits	2 timers de 8 Bits e 1 de 16 Bits
Entradas de Interrupção	4 internas e 2 externas	16 internas e 2 externas	4 internas e 1 externa
Tensão de Operação	3 a 5,5 Volts	4,5 a 5,5 Volts	4,5 a 5,5 Volts
Consumo de Corrente médio do processador	25mA (Clock de 12 MHz)	15mA (Clock de 8 MHz)	10mA (Clock de 4 MHz)
Corrente Média de saída das portas de I/O	15 mA	30 mA	18 mA
Memória ISP	8 KB	8 KB	1 KB
Memória SRAM interna	256 Bytes	256 Bytes	128 Bytes
Capacidade de Endereçamento Externo	64 KB	64 KB	Não possui
Possibilita I/O mapeado em Memória	Sim	Sim	Não
Familiaridade do grupo com o Microcontrolador	Baixo/Médio	Mínimo	Nulo
Aprendizado paralelo ao projeto do microcont.	Sim	Não	Não
Custo do Kit de interfaceamento e gravação	Cerca de R\$ 80,00, mas integrantes do grupo já possuem.	Preço médio de R\$ 180,00 (já montado)	Preço médio de R\$ 80,00 (já montado)
Preço individual (médio) do Microcontrolador	R\$ 7,00	R\$ 8,00	R\$ 7,00

Tabela 5: Comparação das características respectivas aos microcontroladores.
Fonte: [3], [4] e [5]

Como o microcontrolador deverá controlar as quatro matrizes de LEDs, a emissão de um som com frequência de amostragem de 8 KHz e a leitura do nível de bateria, tudo de maneira simultânea, na verdade simultânea ao sentidos humanos, foi estimado (mantendo uma boa margem de segurança) que o microcontrolador deverá possuir no mínimo uma capacidade de processamento de 0,1 MIPS, ou seja, 100 mil instruções por segundo. Quanto a esse fator todos os 3 microcontroladores, apresentados na tabela 5, o cumprem com bastante folga.

Outra importante característica levantada se refere ao número de portas de entrada e saída de dados para que possam ser acessados, lidos e controlados os periféricos utilizados no sistema embarcado. Nesse quesito tanto o processador Atmel 89S52 quanto o ATmega8 conseguem cumpri-lo sem muitos problemas já que além de possuírem uma quantidade maior ou igual de PORTS utilizáveis que o PIC16 apresentam a possibilidade de realizar mapeamento de entrada e saída como memória facilitando o controle e transferência de dados quando são necessários diversos periféricos ao microcontrolador. Nesse critério o PIC16 torna-se quase inviável, pois caso seja realmente possível controlar todos os periféricos necessária com suas portas de entrada e saída disponíveis, a complexidade do circuito deverá aumentar bastante fazendo com que seja demandado muito tempo com o projeto e construção deste.

O consumo de energia, apesar de não ser tão crítico quanto o fator apontado anteriormente, pesa de alguma maneira na escolha do microcontrolador já que o mesmo não será alimentado pela rede elétrica, mas sim por algum conjunto de baterias ou pilhas. Nesse critério o PIC16 seria o mais bem cotado caso não apresentasse o problema do item anterior, o que faria com que o consumo de energia aumentasse ainda mais por conta do número de dispositivos que seriam utilizados para que fosse possível controlar todos os dispositivos. Com isso tanto o ATmega8 quanto o Atmel 89S52 são mais recomendados, tendo o primeiro um consumo menor por conta da frequência de clock analisada e como o processamento máximo oferecido por ambos os microcontroladores é mais do que suficiente para realizar todas as tarefas dirigidas aos mesmos, caso seja necessário, eles poderão ter seu clock diminuído para que o consumo de energia seja menor.

As outras “características técnicas” levantadas na tabela 5 acabam sendo equivalentes entre os microcontroladores e não são tão críticas para a escolha dos mesmos para o projeto, entretanto, dois fatores foram de grande importância e acabaram pesando bastante para a escolha: o custo do mesmo e a familiaridade com relação ao nível de conhecimento da equipe. Como pode ser visto na tabela 5 o Atmel 89S52 junto com o PIC16 são os dois microcontroladores que exigem um menor custo para que seja montada a placa de interfaceamento e gravação, mas,

um fator que faz com que o Atmel 89S52 seja mais viável é o fato de alguns integrantes do grupo já possuírem a placa citada anteriormente, devido a utilização dela na disciplina de microcontroladores, por esse motivo, ele também é, dentre os três analisados, o que a equipe tem um maior nível de conhecimento do seu funcionamento.

Analizados diversos aspectos a equipe chegou à conclusão que o microcontrolador que deverá ser utilizado para o projeto pertence à família 8051, mais especificamente o Atmel 89S52, pois além dele cumprir todas as características necessárias para que o mesmo seja utilizado no projeto também é o que a equipe tem mais familiaridade fazendo com que o tempo de estudo necessário com o mesmo seja inferior aos outros e com isso tanto o custo financeiro quanto o tempo de desenvolvimento serão reduzidos se comparados aos demais. Como alguns integrantes da equipe frequentam uma disciplina em que ele é utilizado eles terão um aprendizado provavelmente mais rápido, reduzindo ainda mais o tempo gasto com estudo do mesmo, podendo assim focar melhor no desenvolvimento do projeto, na montagem do circuito e no estudo dos outros componentes que serão utilizados.

O microcontrolador também será o responsável pelo controle dos dados de áudio que serão armazenados na memória de maneira digital, mas que por sua vez quando forem reproduzidos serão enviados para um conversor Digital Analógico, externo ao microcontrolador, para que ele converta o sinal digital em sinal analógico que possa ser amplificado e reproduzido em um alto falante.

4.3.2 Linguagem de programação embarcada

Para a programação em sistemas embarcados, há algumas alternativas de linguagens, como Assembly e C (bem como seus dialetos e variantes, como Objective-C e C++). Há outras alternativas, como a linguagem Ada, que não serão levadas em conta devido ao fato de serem incomuns ou pouco conhecidas. Todas as linguagens apresentadas permitem usar o hardware para as funcionalidades necessárias (comunicação serial, entrada e saída de dados, etc). A comparação dos critérios respectivos às linguagens apresenta-se resumida na tabela 6.

Critério	Assembly	C
Dependência da arquitetura	Alta	Baixa
Nível de abstração	Baixo	Médio
Eficiência	Maior	Menor
Familiaridade	Média (MIPS)	
	Média (MCS-51)	Alta
	Nenhuma (outras)	
Documentação	Sim	Sim
Semântica	Complexa	Simples

Tabela 6: Comparação das características respectivas às linguagens de programação embarcada.

A linguagem Assembly é apenas um conjunto de mnemônicos (conhecidos como "opcodes"), equivalentes a instruções em linguagem de máquina. Por se programar diretamente em instruções do processador, Assembly normalmente leva à criação de programas muito eficientes. Entretanto, a dificuldade de programação faz com que o trabalho de programação demande mais tempo, e que haja menos a confiabilidade do software resultante. Também é necessário considerar que a linguagem Assembly é diferente para cada família de microcontroladores, pois seus conjuntos de instruções são diferentes. Dessa forma, ao optar-se pelo uso de Assembly, dependendo da opção de microcontrolador adotada, pode ser possível que seja necessário aprender o Assembly daquela arquitetura. Atualmente, os membros da equipe têm familiaridade apenas com o Assembly da arquitetura MIPS e com o da arquitetura Intel MCS-51.

A linguagem C e seus dialetos apresentam a facilidade de terem uma sintaxe mais amigável, permitindo programação em mais alto nível que aquela realizada em Assembly. É possível fazer um mesmo programa que seja o mais independente possível da arquitetura, mas também é possível adicionar trechos de código em Assembly dentro de um programa nessa linguagem. Os membros da equipe têm grande familiaridade com a linguagem C. Após ponderar as vantagens e desvantagens das opções tecnológicas para a programação do software embarcado, optou-se pela programação em linguagem C.

4.3.3 Memória de armazenamento

Com o fim de selecionar a memória mais adequada ao projeto deve-se verificar alguns requisitos impostos ao projeto. Primeiramente, o terceiro requisito com relação à prancha eletrônica expressa que a mesma deve manter a configuração mesmo que o sistema seja desligado. Isto elimina as memórias voláteis da escolha. A especificação técnica referente ao som relata que se necessita armazenar em cada memória relativa a cada botão, um som de 4 segundos com 8000 amostras de 1 byte por segundo, ou seja $8000 \times 4 = 32\text{KB}$. Além disso, a memória deve armazenar uma matriz de 64 bits = 8 bytes. Algo em torno de 33KB de capacidade deve satisfazer com sobra para cada memória. Então, a tabela 7 compara as memórias com capacidades superiores mais próximas a esta:

Critérios/ Memória	EPROM (27c512)	EEPROM (24LC512A)	FLASH (29F010B-70PD)
Apagável Eletricamente	Não	Sim	Sim
Modo de escrita	Byte a byte	Byte a byte	Blocos
Capacidade	64 KB	64 KB	128KB (menor encontrada)
Preço	Cerca de R\$13,00	Cerca de R\$10,00	Cerca de R\$ 20,00

Tabela 7: Comparação das características respectivas às linguagens de programação embarcada.

Fonte [6], [7] e [8].

O segundo requisito da mesma seção aborda uma necessidade de reconfiguração da prancha. Para que isto seja feito sem necessidade de abrir o circuito, devem ser utilizadas memórias apagáveis eletricamente, o que elimina a EPROM. A memória FLASH tem um sistema de escrita por blocos que pode tornar mais simples as escritas e leituras na memória. Assim, optamos pela memória Flash como ideia inicial. Se fatores como atrasos na entrega do componente ou problemas de integração com o resto do sistema atrasarem o cronograma do projeto, teremos a EEPROM como plano B.

5 PLANEJAMENTO DE RISCOS

Neste tópico está descrito os principais possíveis riscos que podem ocorrer durante o desenvolvimento deste trabalho. Cada risco está contido em um formulário que foi sugerido por Gasnier, 2000 Editora IMAN e alterado por Wille.

Projeto: Fun-Icon – Prancha Eletrônica

1º Etapa: Identificação do Risco

Denominação do risco: Queima de algum componente do circuito	Nº Identificação: 1
---	---------------------

Descrição do risco: Algum componente eletrônico utilizado no projeto queimou por uso incorreto do mesmo ou algum problema de fabricação.

2ª Etapa: Avaliação do Risco

Impacto: 5(alto) 4(média/alto) 3(médio) 2(médio/baixo) 1(baixo)

Explicação: Dependendo do componente que foi queimado ele pode aumentar significativamente o custo do projeto e atrasar o seu desenvolvimento, entretanto o impacto no custo e no atraso varia de componente para componente. Consideramos o impacto de nível 4 como uma média dos impactos gerados pela queima de componente.

Probabilidade: 5(alta) 4(média/alta) 3(média) 2(média/baixa) 1 (baixa)

Explicação: Como a queima de componentes é um fator que está diretamente relacionado com a sua utilização existe uma chance razoável de um erro ser cometido durante seu manuseio.

3ª Etapa: Desenvolvimento da Resposta ao Risco

Ações, Responsáveis e Datas de Conclusão

Estratégias e Ações para eliminar ou reduzir este risco (minimizar impacto e/ou probabilidade):

Prevenir: Prestar bastante atenção na hora de manusear os componentes, sempre estar com seu respectivo Datasheet do lado e ao “ligar” componentes cruciais estar com mais de uma pessoa presente para verificar tudo.

Mitigar: Comprar alguns componentes extras para reduzir o impacto no atraso do tempo, caso algum venha a queimar.

Impacto reavaliado: 3

Probabilidade reavaliada: 2

Projeto: Fun-Icon – Prancha Eletrônica

1º Etapa: Identificação do Risco

Denominação do risco: Adoecimento de um membro da equipe	Nº Identificação: 2
---	---------------------

Descrição do risco: Um membro da equipe ter o seu desempenho no trabalho prejudicado por ter contraído uma enfermidade ou estar impossibilitado de exercer sua função por quarentena ou necessidade de repouso.
--

2ª Etapa: Avaliação do Risco

Impacto: 5(alto) <input type="checkbox"/> 4(média/alto) <input type="checkbox"/> 3(médio) <input type="checkbox"/> 2(médio/baixo) <input type="checkbox"/> 1(baixo)
--

Explicação: Isto pode atrasar o cronograma impossibilitando a execução de alguma determinada tarefa, o que pode desencadear um acúmulo de tarefas dependentes desta.
--

Probabilidade: <input type="checkbox"/> 5(alta) <input type="checkbox"/> 4(média/alta) <input type="checkbox"/> 3(média) <input type="checkbox"/> 2(média/baixa) <input type="checkbox"/> 1 (baixa)
--

Explicação: Apesar da normalidade que existe em contrair alguma virose, dificilmente será algo que efetivamente prive o integrante da possibilidade de trabalhar sozinho ou, até mesmo, em grupo.

3ª Etapa: Desenvolvimento da Resposta ao Risco

Ações, Responsáveis e Datas de Conclusão
--

Estratégias e Ações para eliminar ou reduzir este risco (minimizar impacto e/ou probabilidade):
--

Prevenir: Acrescentar dias livres no cronograma para recuperar o trabalho que será acumulado se o problema ocorrer.

Mitigar: Aconselhar os membros da equipe a terem melhores hábitos higiênicos, consultarem um médico regularmente e evitarem situações em que ocorra muita perda de calor e trabalho muito exaustivo. Evitar reuniões que terminem muito tarde, assegurando boas noites de sono. Os membros também devem procurar trabalhar em lugares adequados.
--

Aceitar: Apertar o cronograma sobrecarregando alguns membros, sem exageros.

Impacto reavaliado: 3	Probabilidade reavaliada: 2
------------------------------	------------------------------------

Projeto: Fun-Icon – Prancha Eletrônica

1º Etapa: Identificação do Risco

Denominação do risco: Desistência de um membro da equipe.	Nº Identificação: 3
---	---------------------

Descrição do risco: Algum membro da equipe desista do projeto por algum motivo.

2ª Etapa: Avaliação do Risco

Impacto:	<input type="checkbox"/> 5(alto) <input checked="" type="checkbox"/> 4(média/alto) <input type="checkbox"/> 3(médio) <input type="checkbox"/> 2(médio/baixo) <input type="checkbox"/> 1(baixo)
-----------------	--

Explicação: Aumentará a carga de trabalho de cada membro, podendo gerar atrasos. Áreas funcionais podem ser afetadas, devido à perda de conhecimento técnico no setor.

Probabilidade:	<input type="checkbox"/> 5(alta) <input type="checkbox"/> 4(média/alta) <input type="checkbox"/> 3(média) <input checked="" type="checkbox"/> 2(média/baixa) <input type="checkbox"/> 1 (baixa)
-----------------------	---

Explicação: Todo o grupo está envolvido e motivado na elaboração do projeto e a probabilidade de acontecer algo grave com algum membro (lembrando que o grupo é relativamente pequeno) em que ele tenha que vir a se ausentar e desistir do projeto é pequena. Outro motivo pode ser desentendimentos internos que poderá vir a ocorrer.

3ª Etapa: Desenvolvimento da Resposta ao Risco

Ações, Responsáveis e Datas de Conclusão
--

Estratégias e Ações para eliminar ou reduzir este risco (minimizar impacto e/ou probabilidade):
--

Prevenir: Manter o grupo conciso e empolgado com o projeto para que o grupo se sinta motivado a terminar-lo. Procurar manter uma união e valorizar a opinião e idéia de cada membro para evitar desentendimentos internos.

Mitigar: Manter mais de uma pessoa informada sobre um determinado setor, assim caso ocorra uma desistência o membro possa ser substituído com mais facilidade.

Aceitar: Se ocorrer tentar substituir a pessoa e sobrestrar alguns membros para que o projeto seja entregue no tempo determinado.

Impacto reavaliado: 3	Probabilidade reavaliada: 1
------------------------------	------------------------------------

Projeto: Fun-Icon – Prancha Eletrônica

1º Etapa: Identificação do Risco

Denominação do risco: Inexperiência da tecnologia utilizada	Nº Identificação: 4
Descrição do risco: Dificuldade de utilização da tecnologia por não entender o funcionamento e a utilização desta.	

2ª Etapa: Avaliação do Risco

Impacto: 5(alto) 4(média/alto) 3(médio) 2(médio/baixo) 1(baixo)

Explicação: Este risco pode provocar um atraso considerável, pois para algumas tecnologias é complexo encontrar referências esclarecedoras sobre pontos que só são entendidas com a prática com a utilização destas.

Probabilidade: 5(alta) 4(média/alta) 3(média) 2(média/baixa) 1 (baixa)

Explicação: Considerando que vários integrantes ainda não estão estudando disciplinas essenciais que tratam como, por exemplo, de micro-controladores, há uma possibilidade relativamente alta de se encontrar dificuldades relacionadas à falta de experiência.

3ª Etapa: Desenvolvimento da Resposta ao Risco

Ações, Responsáveis e Datas de Conclusão

Estratégias e Ações para eliminar ou reduzir este risco (minimizar impacto e/ou probabilidade):

Prevenir: Inserir no plano de projeto datas para estudo e ensaio das tecnologias utilizadas no projeto para se adquirir a experiência necessária.

Mitigar: Escolher preferencialmente tecnologias em que já se tem experiência ou que se saiba que há documentação clara e simples a respeito. Procurar alguém que tenha conhecimento sobre a tecnologia, como por exemplo os professores.

Impacto reavaliado: 3

Probabilidade reavaliada: 2

Projeto: Fun-Icon – Prancha Eletrônica

1º Etapa: Identificação do Risco

Denominação do risco: Atraso na entrega de componentes ou materiais.	Nº Identificação: 5
Descrição do risco: Todos os componentes ou materiais que necessitem de entrega estão sujeitos ao risco de atrasarem.	

2ª Etapa: Avaliação do Risco

Impacto: 5(alto) 4(média/alto) 3(médio) 2(médio/baixo) 1(baixo)

Explicação: O atraso de um item importante para continuação do projeto provocará problemas com o cronograma podendo comprometer seriamente o desenvolvimento dependendo do tempo perdido. Atrasos pequenos ou de itens não tão importantes terão um impacto menor.

Probabilidade: 5(alta) 4(média/alta) 3(média) 2(média/baixa) 1 (baixa)

Explicação: Atraso na entrega de peças devido a extravio ou problemas por parte do correio ou empresa responsável pode ocorrer com relativa probabilidade.

3ª Etapa: Desenvolvimento da Resposta ao Risco

Ações, Responsáveis e Datas de Conclusão

Estratégias e Ações para eliminar ou reduzir este risco (minimizar impacto e/ou probabilidade):

Prevenir: Utilizar peças ou componentes que se encontrem na região próxima sem a necessidade de depender de terceiros, principalmente peças essenciais para a continuação do projeto.

Mitigar: Realizar o pedido o mais cedo possível, já prevendo um possível atraso.

Impacto reavaliado: 2

Probabilidade reavaliada: 1

Projeto: Fun-Icon – Prancha Eletrônica

1º Etapa: Identificação do Risco

Denominação do risco: Erro no planejamento de tempo	Nº Identificação: 6
Descrição do risco: Incompatibilidade entre o tempo planejado para se realizar uma tarefa com o tempo real gasto para realizá-la.	

2ª Etapa: Avaliação do Risco

Impacto: 5(alto) 4(média/alto) 3(médio) 2(médio/baixo) 1(baixo)

Explicação: O atraso de uma determinada tarefa pode prejudicar e atrasar todo o cronograma planejado. No pior caso isso pode acarretar em um atraso na entrega do artefato final (situação inaceitável para todo o contexto do projeto).

Probabilidade: 5(alta) 4(média/alta) 3(média) 2(média/baixa) 1 (baixa)

Explicação: Por falta de experiência dos integrantes estimamos que existe uma probabilidade média de ocorrer essa incompatibilidade entre o tempo estimado e o tempo real destinado às tarefas. Vale lembrar que pode ser necessário dedicar um tempo considerável para o aprendizado de uma determinada tecnologia e isso acaba influenciando no risco de má alocação de tempo.

3ª Etapa: Desenvolvimento da Resposta ao Risco

Ações, Responsáveis e Datas de Conclusão

Estratégias e Ações para eliminar ou reduzir este risco (minimizar impacto e/ou probabilidade):

Prevenir: Considerar o nível de familiaridade que o integrante destinado a realizar uma tarefa possui com as tecnologias envolvidas e se o integrante possui pouca familiaridade então será destinado um período de aprendizado ou a inserção de um ou mais integrantes na tarefa (de modo a ajudar no progresso e na devida conclusão dela). Deixar um tempo de folga no cronograma para ajudar na prevenção.

Aceitar: Diminuir o período destinado a outras tarefas subsequentes se elas dependem da conclusão da tarefa atrasada. Lembrando que isso pode acarretar na diminuição da qualidade do rendimento dessas tarefas dependentes.

Impacto reavaliado: 4

Probabilidade
reavaliada: 2

Projeto: Fun-Icon – Prancha Eletrônica

1º Etapa: Identificação do Risco

Denominação do risco: Desentendimento entre os membros da equipe	Nº Identificação: 7
Descrição do risco: Conflito ou estado social que acabe prejudicando o relacionamento de cunho profissional entre os membros do projeto.	

2ª Etapa: Avaliação do Risco

Impacto: 5(alto) 4(média/alto) 3(médio) 2(médio/baixo) 1(baixo)

Explicação: O desentendimento da equipe seria altamente prejudicial para o progresso do projeto como um todo, pois a falta de comunicação traz impactos no andamento do cronograma e na integração das tarefas concedidas a cada membro.

Probabilidade: 5(alta) 4(média/alta) 3(média) 2(média/baixa) 1 (baixa)

Explicação: Todos os integrantes da equipe já possuem um bom período de convivência acadêmica e social, portanto avaliamos com segurança que a probabilidade de haver um conflito de nível prejudicial seja bastante baixa.

3ª Etapa: Desenvolvimento da Resposta ao Risco

Ações, Responsáveis e Datas de Conclusão

Estratégias e Ações para eliminar ou reduzir este risco (minimizar impacto e/ou probabilidade):

Prevenir: Manter uma comunicação clara e detalhada sobre o estado atual do projeto com todos os integrantes, bem como o planejamento para o desenvolvimento futuro (execução) e a manutenção dele (controle). Isso previne conflitos desnecessários.

Mitigar: O gerente (ou algum outro membro disposto da equipe) deve intervir e solucionar/amenizar as insatisfações de ambas as partes do conflito. Caso uma parte grupo venha a ter uma opinião sobre determinado assunto e a outra parte tenha uma opinião oposta o gerente terá autoridade para tratar e resolver as decisões.

Aceitar: Se não houver possibilidade de acordo, cada integrante deve-se lembrar de seu compromisso como engenheiro, tendo sempre uma atitude profissional frente aos problemas.

Impacto reavaliado: 2

Probabilidade reavaliada: 2

Projeto: Fun-Icon – Prancha Eletrônica

1º Etapa: Identificação do Risco

Denominação do risco: Perda de dados do projeto	Nº Identificação: 8
Descrição do risco: Parte dos dados do projeto foi perdida por algum motivo, tanto dados referentes ao desenvolvimento quanto a documentação.	

2ª Etapa: Avaliação do Risco

Impacto: 5(alto) 4(média/alto) 3(médio) 2(médio/baixo) 1(baixo)

Explicação: Dependendo de quais dados forem perdidos as consequências podem ser mínimas que implicam apenas em um pequeno atraso no tempo e um pequeno aumento no custo ou gravíssimas caso venha a comprometer grande parte do projeto. Se ocorrer nas etapas finais, fazem com que o aumento no tempo e no custo passe a ser intratável.

Probabilidade: 5(alta) 4(média/alta) 3(média) 2(média/baixa) 1 (baixa)

Explicação: Geralmente este tipo de perda de dados não ocorre com tanta frequência, entretanto, como estamos tratando com várias etapas do projeto e com um grupo de pessoas esse risco é um pouco maior.

3ª Etapa: Desenvolvimento da Resposta ao Risco

Ações, Responsáveis e Datas de Conclusão

Estratégias e Ações para eliminar ou reduzir este risco (minimizar impacto e/ou probabilidade):

Prevenir: Utilização de algum software/mecanismo de backup que ocorra em um período determinado. Utilização de algum software de controle de versões para gerenciar diferentes etapas e versões que os softwares venham a ter.

Mitigar: Utilizar um software que sempre distribua todos os dados entre todas as pessoas do projeto e guardar os dados em locais separados, descentralizar os dados.

Impacto reavaliado: 2	Probabilidade reavaliada: 1
------------------------------	------------------------------------

Projeto: Fun-Icon – Prancha Eletrônica

1º Etapa: Identificação do Risco

Denominação do risco: Documentação errônea das tecnologias Utilizadas.	Nº Identificação: 9
Descrição do risco: A documentação de qualquer das tecnologias utilizadas seja incompleta ou imprecisa.	

2ª Etapa: Avaliação do Risco

Impacto: <input type="checkbox"/> 5(alto) <input checked="" type="checkbox"/> 4(média/alto) <input type="checkbox"/> 3(médio) <input type="checkbox"/> 2(médio/baixo) <input type="checkbox"/> 1(baixo)
Explicação: Os componentes escolhidos podem não fazer o que deveriam, ou fazer de forma errada.
Probabilidade: <input type="checkbox"/> 5(alta) <input type="checkbox"/> 4(média/alta) <input type="checkbox"/> 3(média) <input type="checkbox"/> 2(média/baixa) <input checked="" type="checkbox"/> 1 (baixa)
Explicação: Grande parte das tecnologias que se pensou em utilizar é muito utilizada, então é pouco provável que sejam mal documentadas ou não façam o que prometem.

3ª Etapa: Desenvolvimento da Resposta ao Risco

Ações, Responsáveis e Datas de Conclusão	
Estratégias e Ações para eliminar ou reduzir este risco (minimizar impacto e/ou probabilidade):	
Prevenir: Busca-se o uso de tecnologias que já sejam bem estabelecidas, de forma que haja bastante informação sobre elas.	
Mitigar: Procurar saber se existe uma tecnologia alternativa que possa ser utilizado no lugar da tecnologia falha e estudá-la para caso o problema ocorra.	
Aceitar: Diminuir a funcionalidade do projeto devido a incapacidades técnicas ou trocar de tecnologia se possível.	
Impacto reavaliado: 3	Probabilidade reavaliada: 1

6 PLANEJAMENTO

6.1 SOFTWARE

6.1.1 Software de interface

Antes de começar a elaborar o projeto do software buscamos analisar quais seriam seus casos de uso. Desta análise obtemos quatro casos e alguns casos inclusos neles, o diagrama completo dos casos de uso está descrito na figura 1.

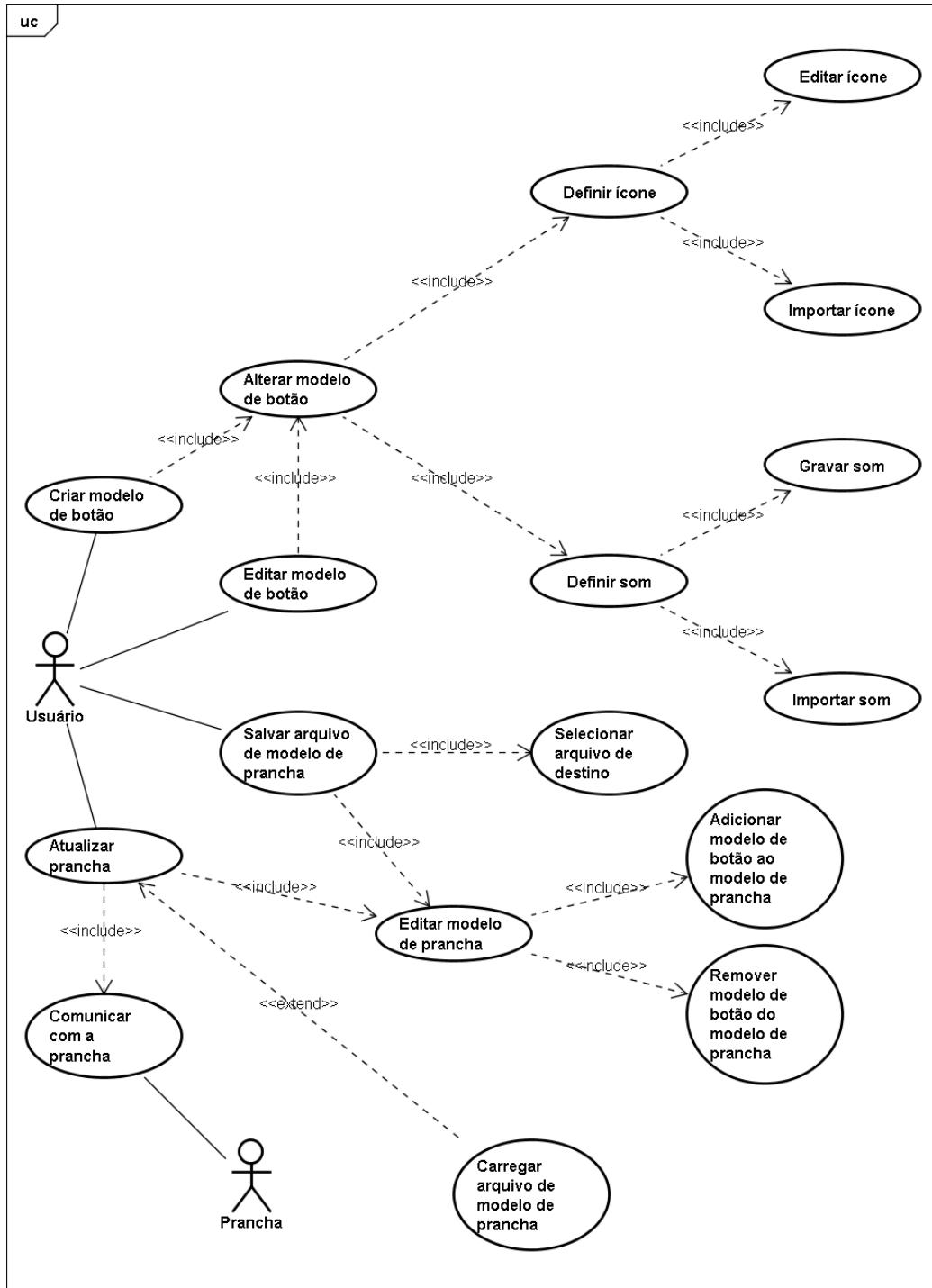


Figura 1: Casos de uso do programa da estação base

Desenvolvemos no planejamento do software alguns diagramas UML baseado nestes casos de uso: diagrama de classes, diagrama de objetos e

diagramas de sequência.

A seguir, nós dispomos a documentação UML de todo o programa. Nela incluímos: diagrama de classes, diagrama de objetos e diagramas de sequência (para alguns casos de uso).

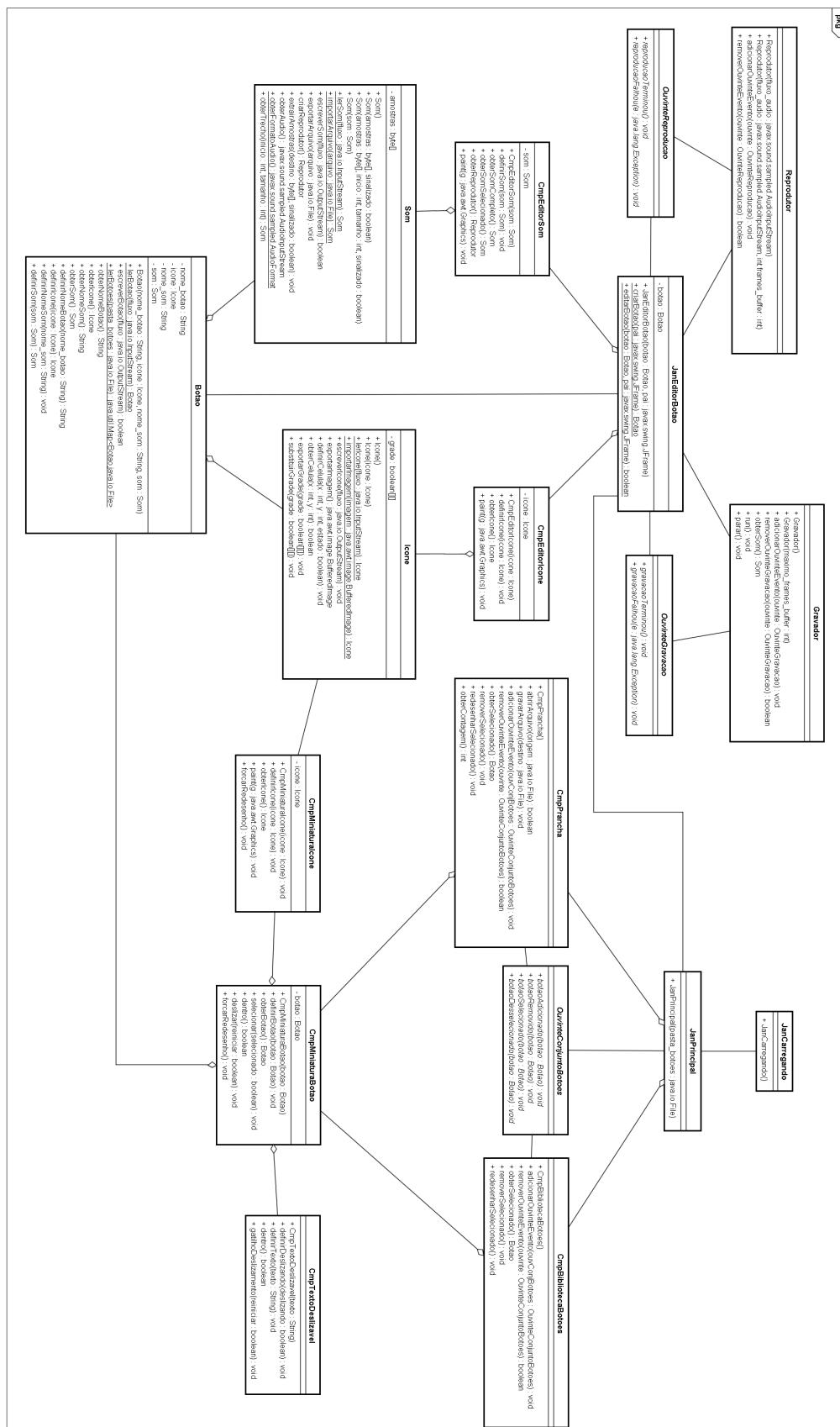


Figura 2: Diagrama de classes do programa da estação base

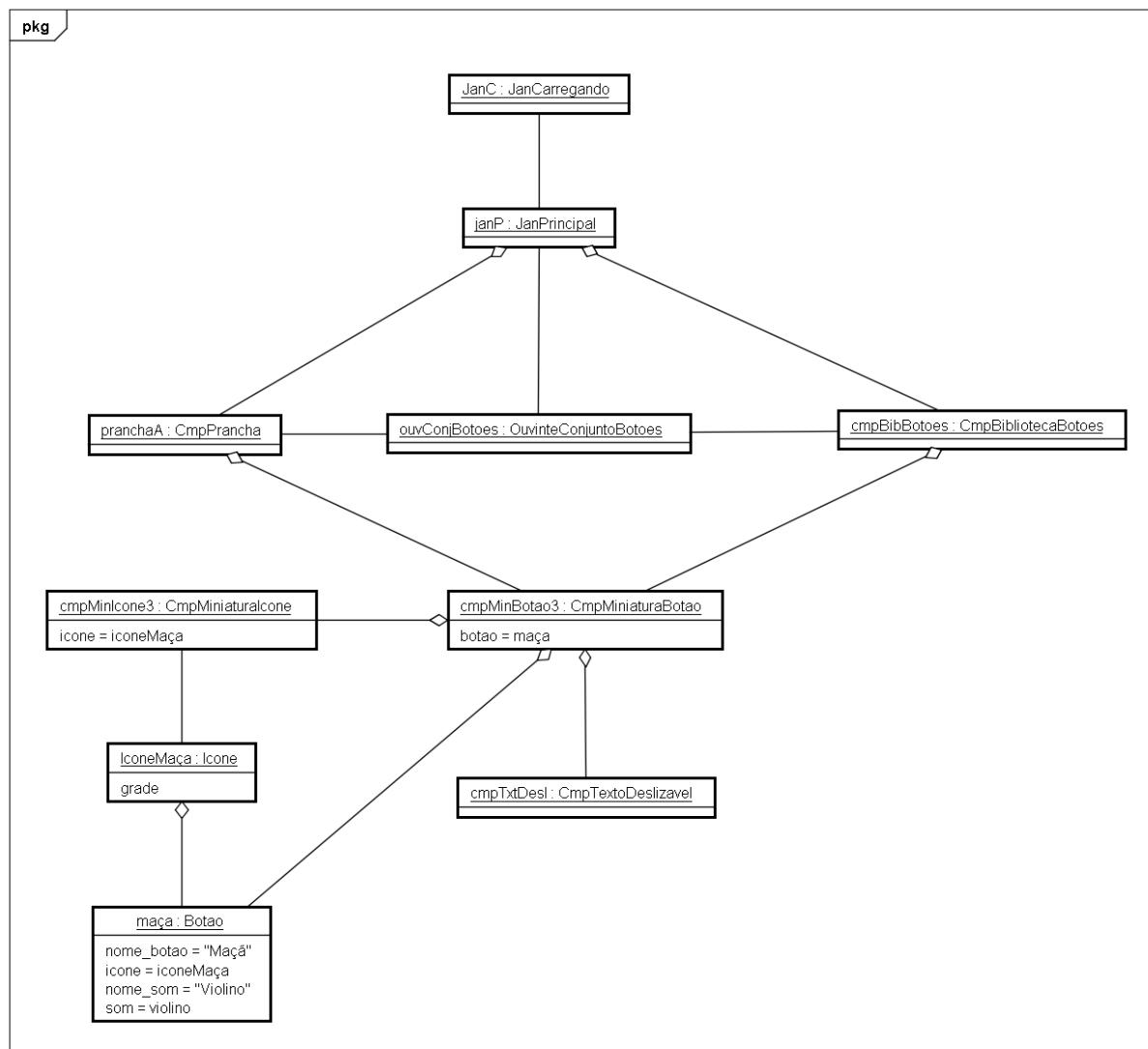


Figura 3: Diagrama de objetos do programa da estação base (janela principal)

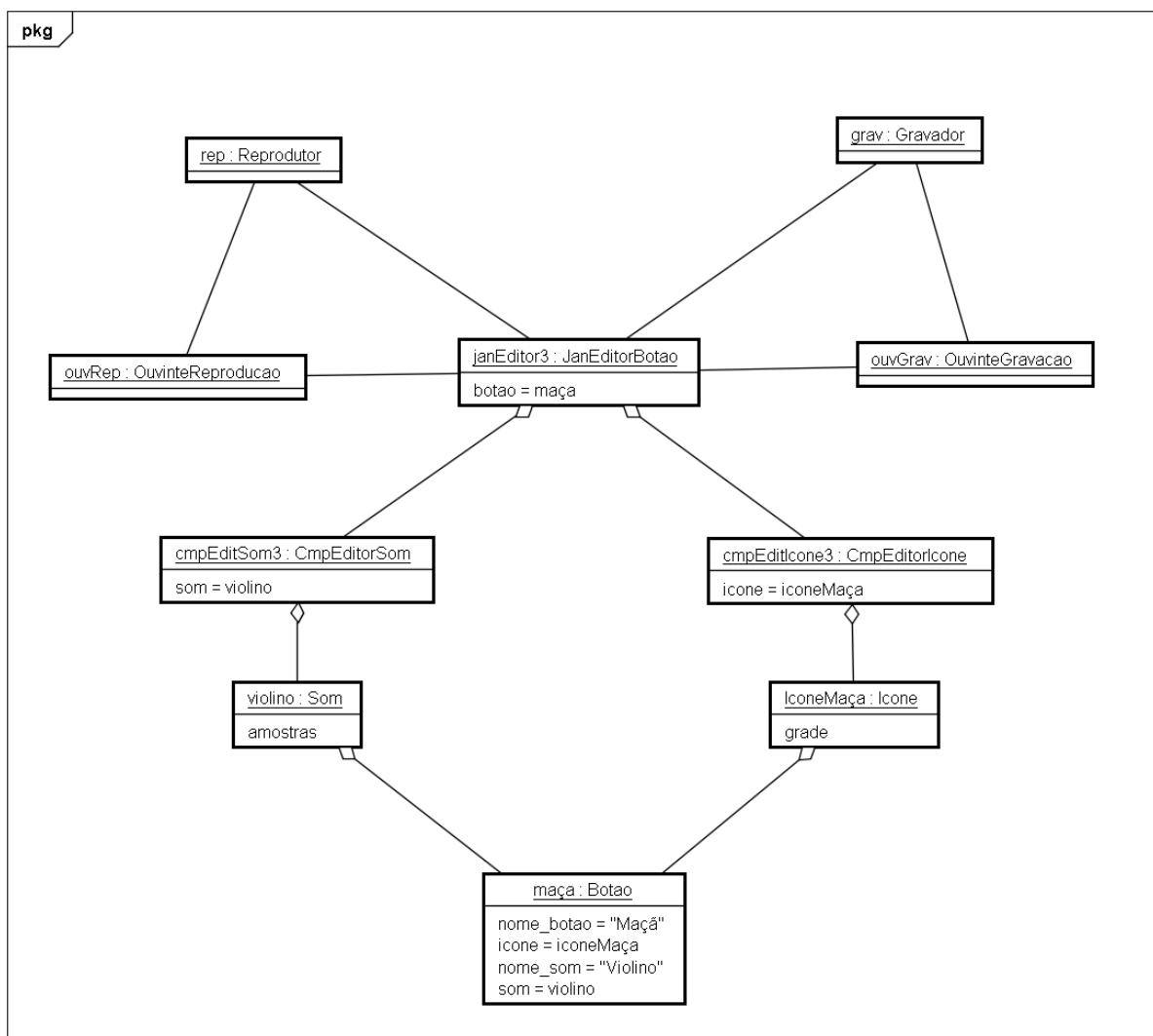


Figura 4: Diagrama de objetos do programa da estação base (janela de edição de botão)

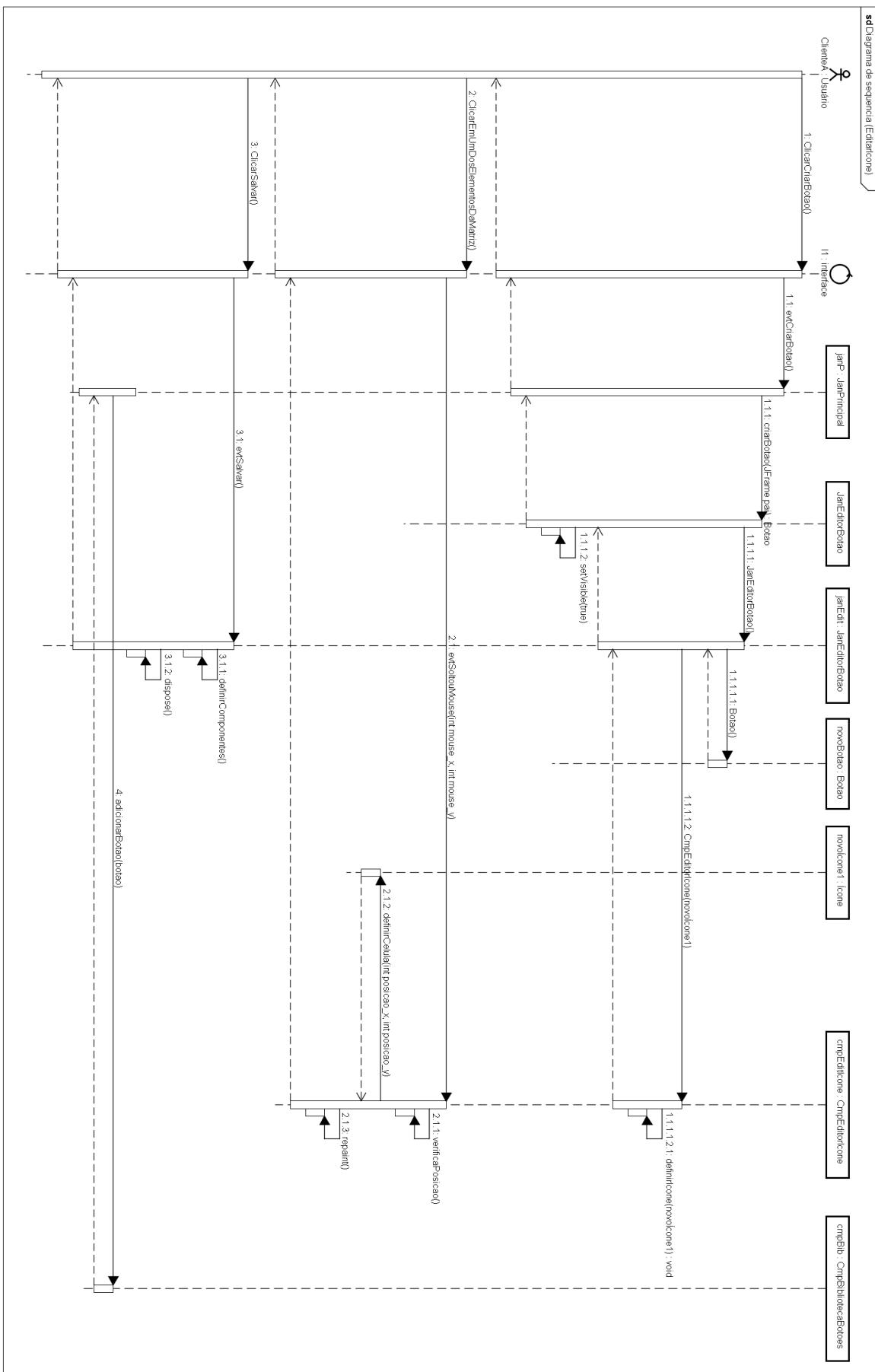


Figura 5: Diagrama de sequência do caso de uso "editar ícone"

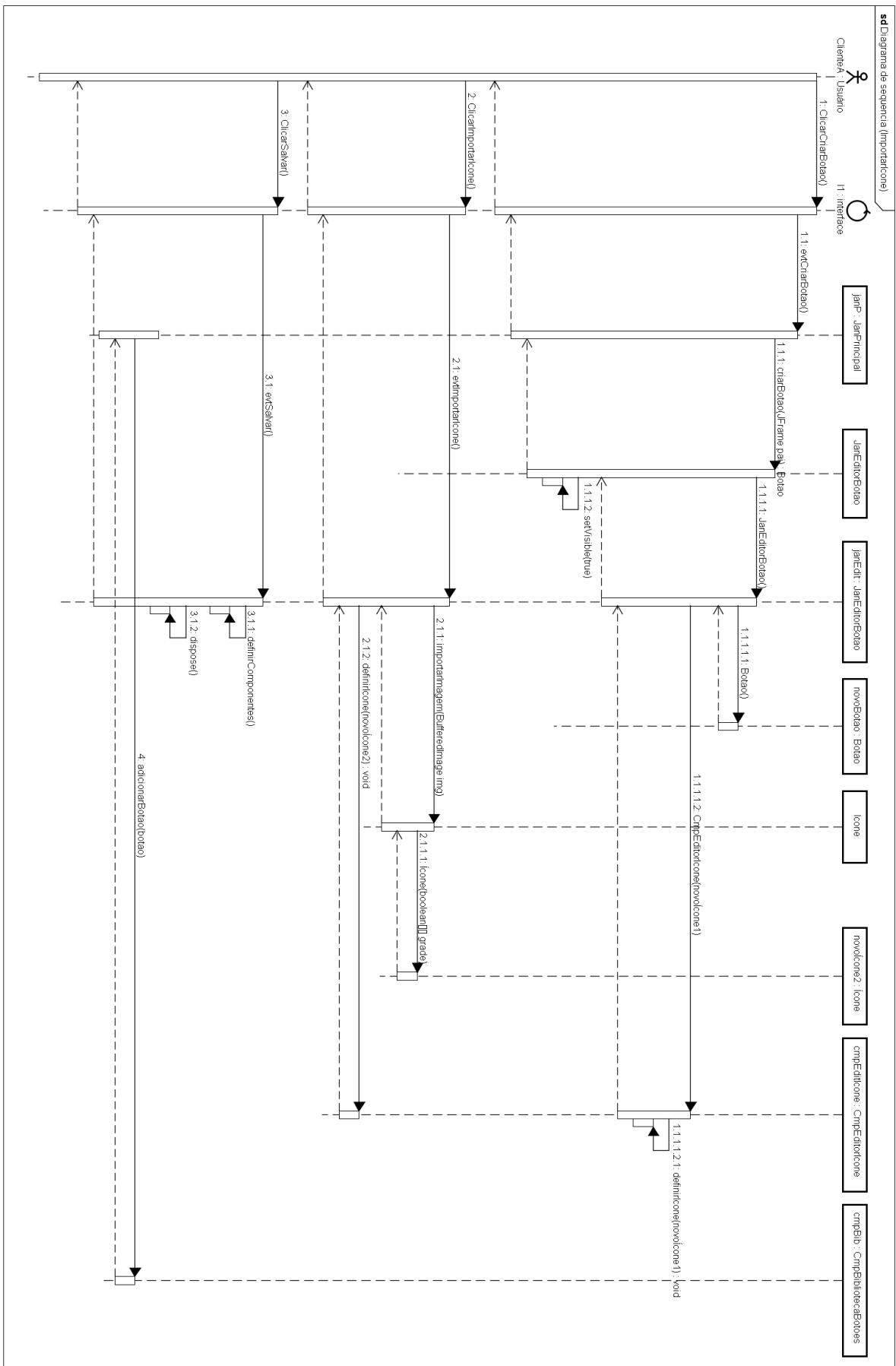


Figura 6: Diagrama de sequência do caso de uso "importar ícone"

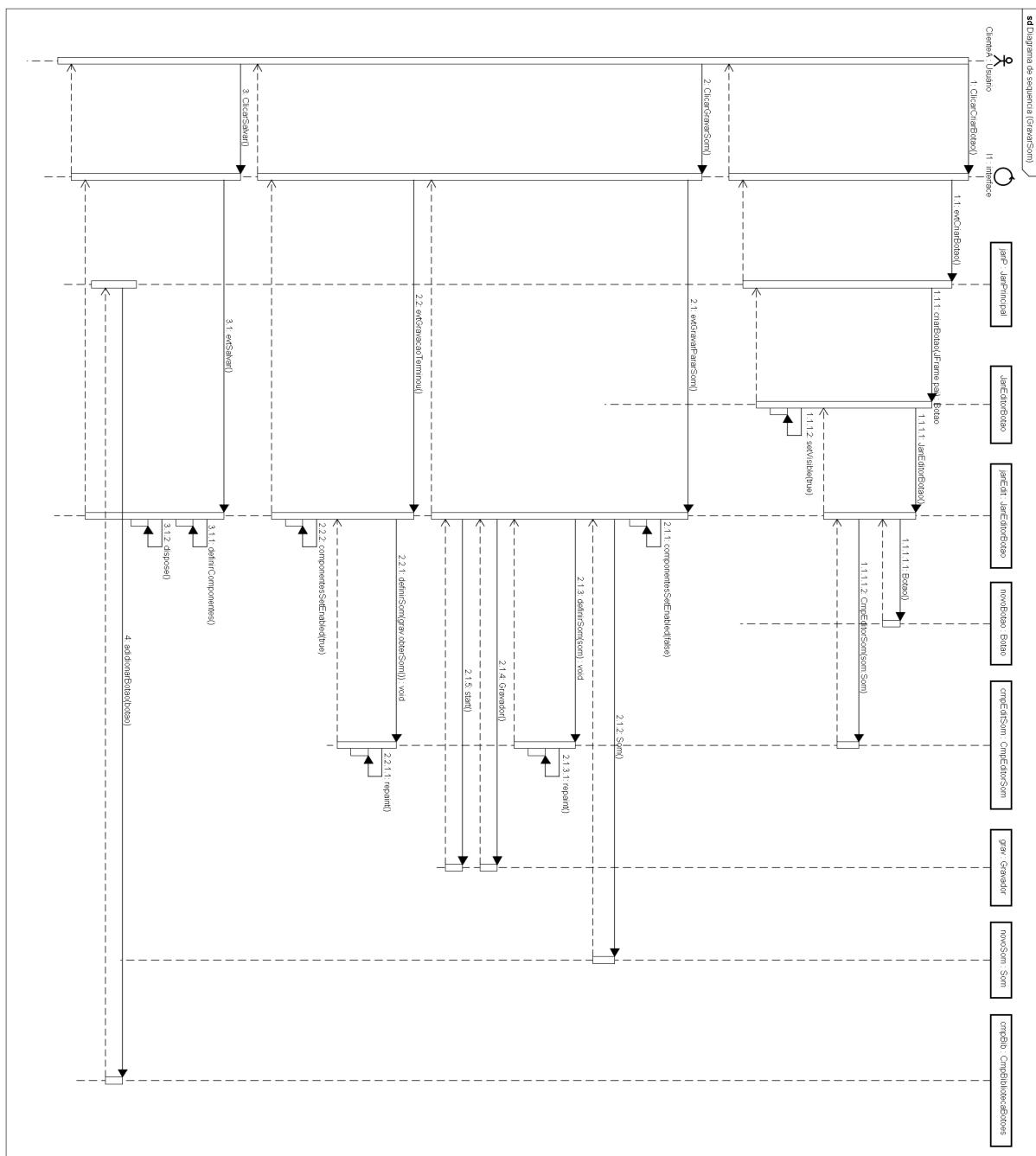


Figura 7: Diagrama de sequência do caso de uso "gravar som"

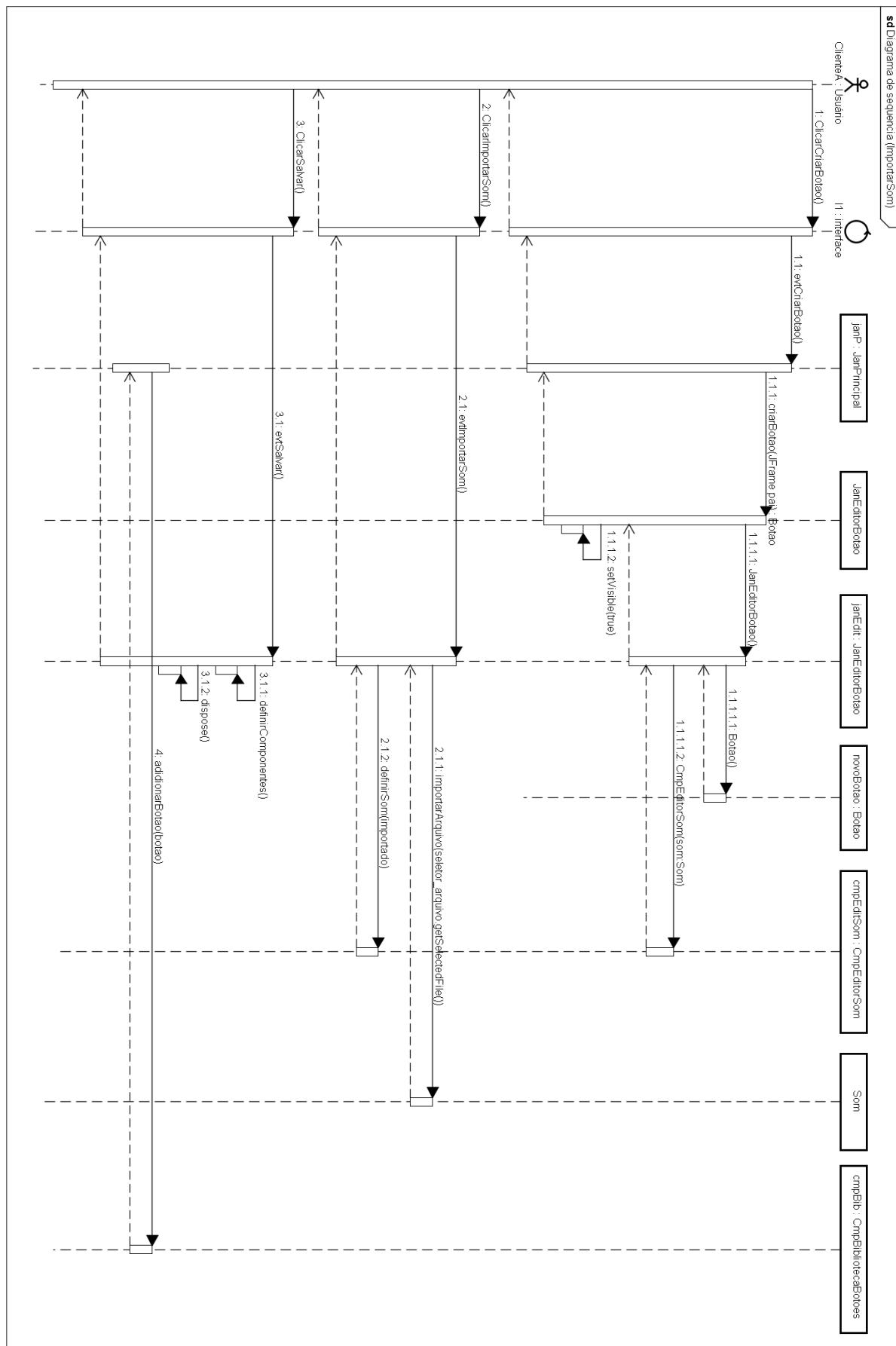


Figura 8: Diagrama de sequência do caso de uso "importar som"

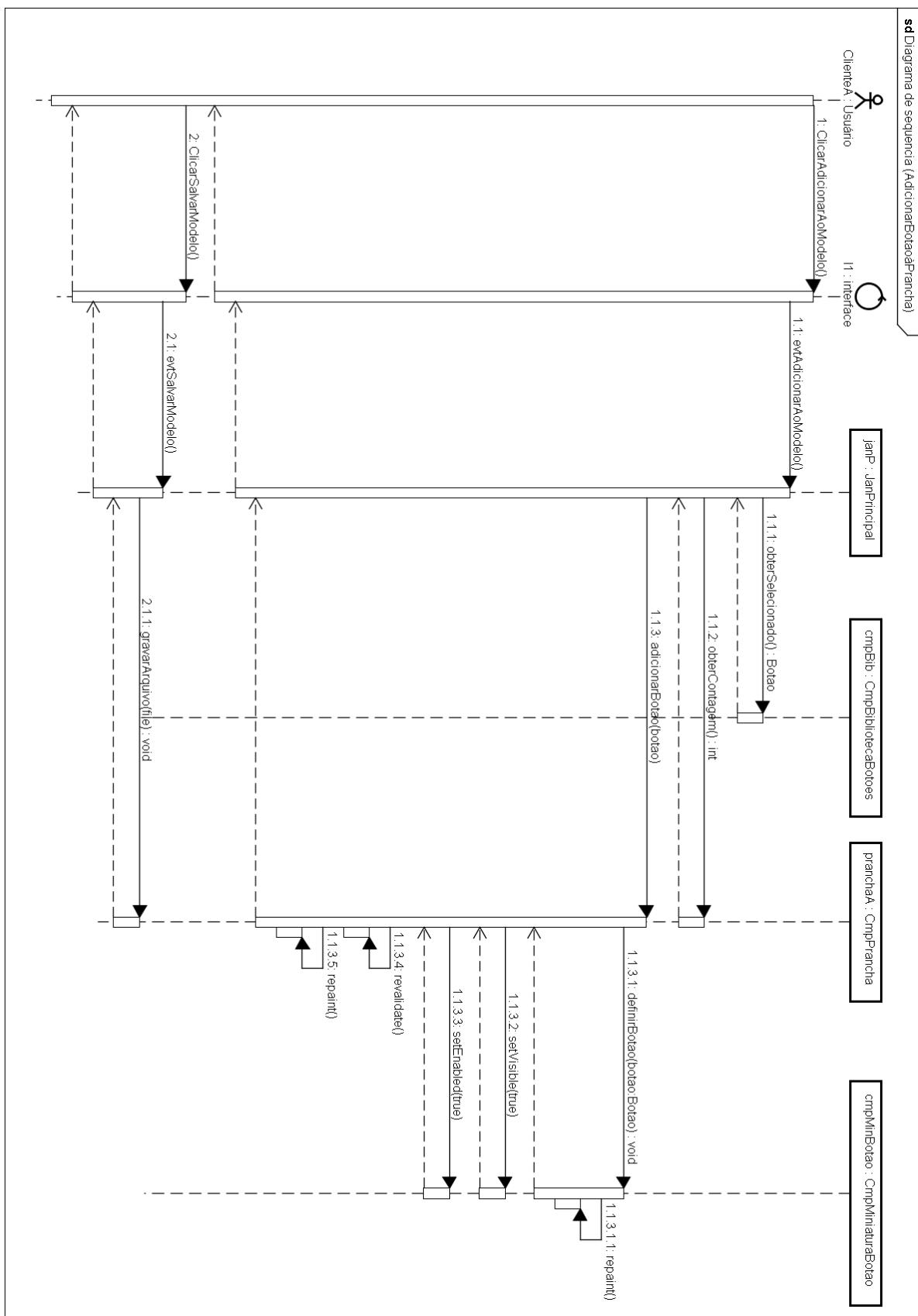


Figura 9: Diagrama de sequência do caso de uso "adicionar botão à prancha"

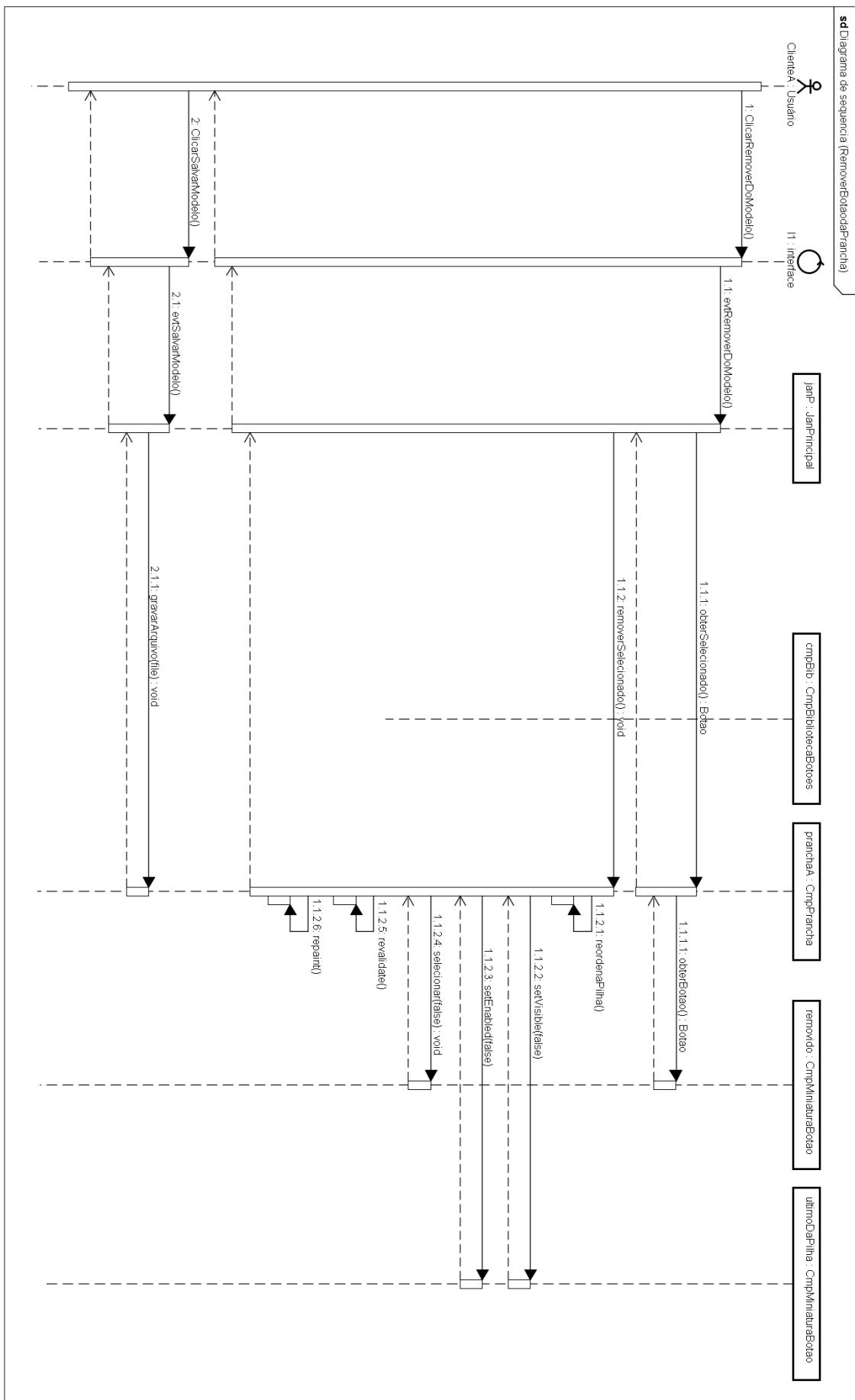


Figura 10: Diagrama de sequência do caso de uso "remover botão da prancha"

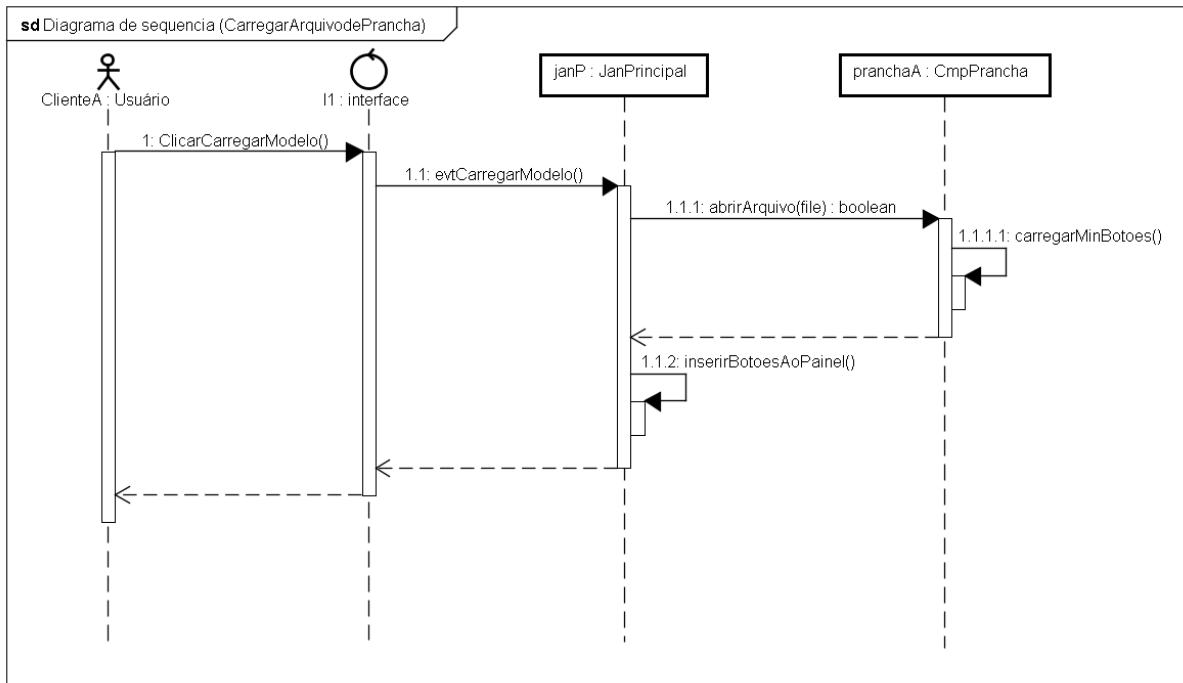


Figura 11: Diagrama de sequência do caso de uso "carregar arquivo de prancha"

6.1.2 Software do sistema embarcado

O software do sistema embarcado, também conhecido como firmware foi planejado com base nos estados de operação da prancha. Os estados da prancha estão detalhados na máquina de estados apresentado na figura 12

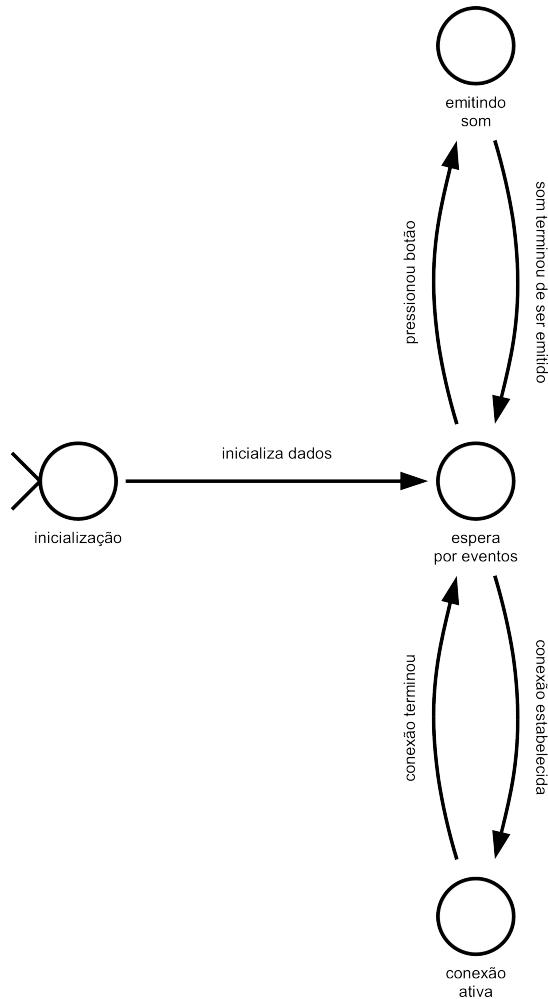


Figura 12: Máquina de estados do firmware.

O firmware quando é ligado possui uma rotina de inicialização dos dados e então entra em uma estado de espera por algum evento. Enquanto este evento não ocorre o programa fica realizando a varredura das matrizes.

Ao apertar o botão de alguma matriz o programa entra no estado “emitindo som”, não podendo realizar outras tarefas, quando o som acaba ele retorna ao estado de espera.

Quando conectamos o cabo de conexão serial a prancha entra no estado de conexão ativa, transferindo dados. Vale ressaltar que no estado de conexão ativa ele não poderá emitir sons e do contrário se emitindo som não poderá entrar no estado de conexão.

6.2 PROTOCOLO DE COMUNICAÇÃO

A comunicação entre a estação base (computador) e o dispositivo móvel (prancha) é representada em um sistema com duas camadas: a física e a de aplicação. A camada física refere-se especificamente ao meio físico de transporte dos dados, bem como o protocolo utilizado, nesse meio, para a transmissão dos dados. A camada de aplicação recebe dados e os processa. Em qualquer meio de transmissão é possível que alguns dados sejam corrompidos ou perdidos, caso ocorra algum problema a camada de aplicação o identifica e interrompe a operação. A estrutura em camadas é apresentada na figura 13.

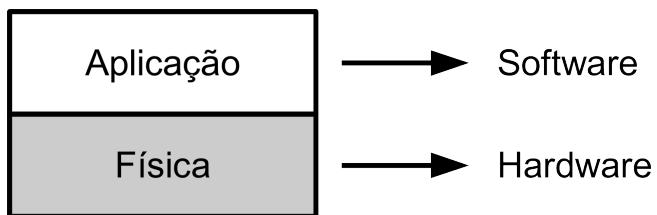


Figura 13: Camadas do sistema de transmissão.

O meio físico utilizado é o cabo serial, no padrão RS-232. A unidade mínima de dados nesse meio são *bytes*, transmitidos de sequencial, *bit a bit*. Nas camadas superiores, os *bytes* são agrupados, formando mensagens. A camada de aplicação valida a mensagem, se positivo continua a sua operação, caso contrário aborta a comunicação.

As mensagens seguem uma estrutura fixa. É formada de um *byte* para o “tipo” (identificador da natureza da mensagem), dois bytes para o “tamanho” (tamanho do corpo da mensagem), seguidos de diversos *bytes* de “conteúdo” (dados em si da mensagem), e finalmente dois *bytes* de “checksum” (para detecção de erros de transmissão). Os tipos são escolhidos de forma que a distância de *Hamming* seja maior que a unidade, ou seja, o número de bits diferentes entre eles seja maior que um. Faz-se isso para que a corrupção de um *byte* de tipo não cause a interpretação errônea da natureza de uma mensagem. O *checksum* é calculado através da soma de todos os *bytes* da mensagem, exceto os dois *bytes* que compõem o próprio *checksum*; para fins de simplicidade, aceita-se que a validação

do *checksum* seja feita apenas com o *byte* menos significativo. A estrutura de uma mensagem é apresentada na figura 14.

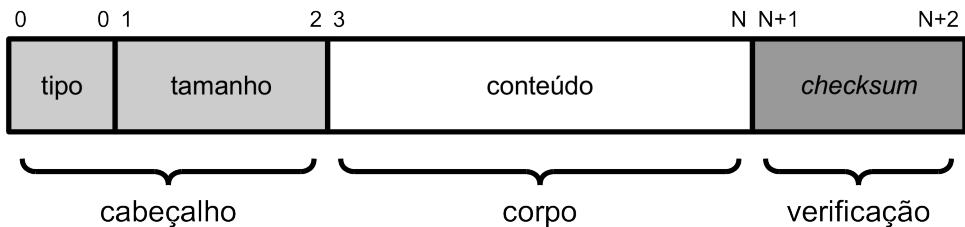


Figura 14: Estrutura de uma mensagem

6.2.1 Mensagens de aplicação

Quando se deseja iniciar a comunicação quem envia a primeira mensagem é a estação base com uma “mensagem de estabelecimento de conexão”. Esta mensagem poderia ser traduzida de uma maneira informal como “Oi prancha, você está aí? Podemos começar a transmissão de dados?” que além de ser um pedido de *handshake* é também uma mensagem que pergunta à prancha se no estado em que se encontra é possível iniciar a transmissão dos dados.

Esta mensagem é identificada pelo tipo 0x03 com um tamanho de dois bytes, sendo um para informar o número do botão que será alterado e o outro para identificar a versão (a versão aqui apresentada é zero) do protocolo que está sendo usado de acordo com o modelo da prancha e por fim dois bytes de *checksum*. Se houver uma versão mais nova com mais funcionalidades deve ser estabelecido compatibilidade reversa.

Para o devido estabelecimento da conexão, o computador envia essa mensagem e espera o recebimento de uma outra mensagem com a resposta. Se a mensagem falhar por *timeout*, assume-se que não há placa conectada ao computador, ou que outro dispositivo está conectado à porta serial. Com o pedido aceito, a prancha entra no estado de comunicação ativa. A estrutura da mensagem de estabelecimento de comunicação é apresentada na figura 15.

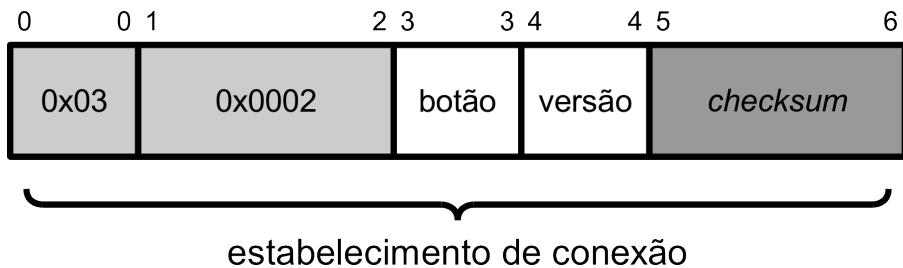


Figura 15: Estrutura de uma mensagem de estabelecimento de conexão

É possível que, devido a problemas no canal de comunicação, ou em uma das entidades comunicantes, haja corrupção dos dados ou perda de dados das mensagens; a esses fenômenos, dá-se o nome de “falhas de comunicação”. Há três tipos de falhas de comunicação previstas. O primeiro tipo de falha de comunicação previsto é que uma mensagem enviada seja recebida de forma incompleta (com bytes faltantes), ou que uma mensagem demore mais para ser recebida ocorrendo assim um erro denominado “timeout”. É possível também que a mensagem recebida tenha mais bytes que o desejado, num erro chamado “mensagem grande demais”. Ainda outro problema é caso algum bit corrompa na transmissão e então falhando na verificação de checksum gerando um “erro de checksum”.

São testados estes três casos de falha de transmissão e se alguma delas ocorrer em alguma mensagem o lado afetado para a comunicação enviando uma mensagem de abortar. Esta mensagem é composta pelo tipo 0x0F se o erro ocorrer na estação base e 0xBF se for no *firmware*. A sua estrutura está representado na figura 16 e 17.

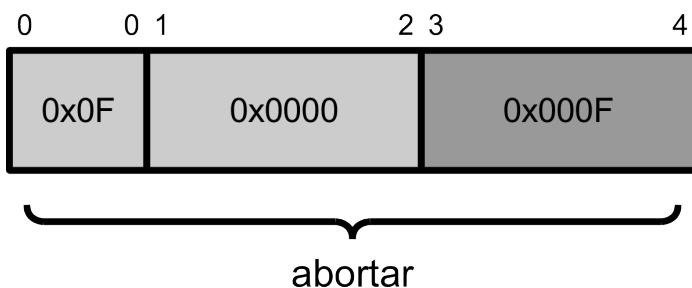


Figura 16: Estrutura de uma mensagem de abortar da estação base

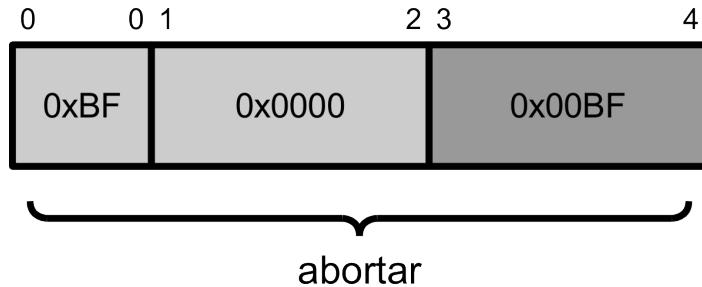


Figura 17: Estrutura de uma mensagem de abortar da prancha

A resposta para a primeira mensagem pode ser a “mensagem de permissão de envio” ou a “mensagem de negação de envio”.

Se for negado o pedido de transmissão a comunicação se encerra e o usuário receberá uma mensagem indicando que a prancha está ocupada no momento (emitindo som, por exemplo). Esta mensagem é composta pelo tipo 0xBC com tamanho 0 e checksum 0x00BC, ela está representada na figura 18.

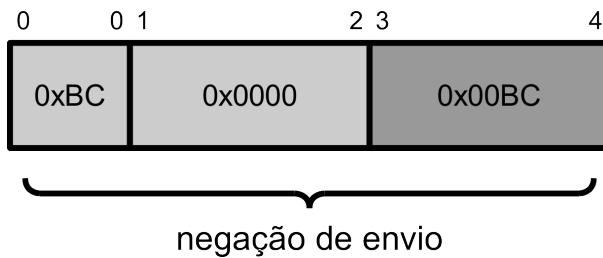


Figura 18: Estrutura de uma mensagem de negação de envio

A mensagem de permissão de envio é formada pelo tipo 0xB3 de tamanho 0 e checksum 0x00B3. Esta representada na figura 19.

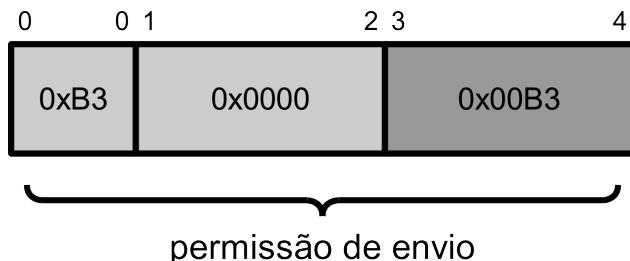


Figura 19: Estrutura de uma mensagem de permissão de envio

Com a permissão o computador inicia o envio dos dados através de uma “mensagem de envio de dados” que é composta pelo tipo 0x0C com um tamanho variável que depende do tamanho do segmento e por fim o checksum. Quanto maior a mensagem maior a possibilidade de alguma bit corromper ou ser perdido, por este motivo não mandamos completamente o som e ícone em uma só mensagem e sim a dividimos em segmentos de 256 bytes determinado. Este tamanho foi determinado apenas baseado em testes que fizemos e foi um valor que apresentou poucos erros na transmissão. A figura 20 representa tal mensagem.

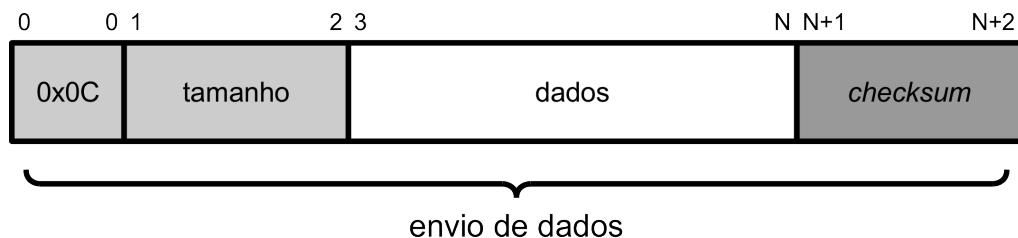


Figura 20: Estrutura de uma mensagem de envio de dados

Quando a prancha recebe esta mensagem, ela é primeiramente validada e se positivo o dispositivo envia para o computador uma mensagem de confirmação de recebimento do segmento e permitindo a estação base enviar outro segmento. Esta mensagem tem o tipo 0xB6 com tamanho nulo e checksum 0x00B6, esta representado na figura 21.

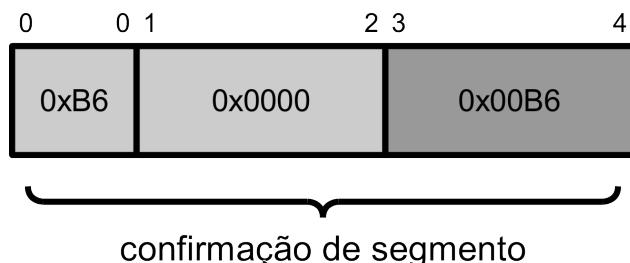


Figura 21: Estrutura de uma mensagem de confirmação de segmento

A transferência termina com uma mensagem de envio de dados com tamanho zero, assim a prancha reconhece o final do envio e começa a gravar as mensagens que estão armazenadas na memória temporária (RAM) para a memória

permanente (FLASH). Quando a transferência se encerra a prancha envia uma “mensagem de término de gravação” avisando a estação base o fim da gravação e assim podendo encerrar a comunicação com sucesso. Esta mensagem é composta pelo tipo 0xB0 com tamanho zero e o checksum igual ao tipo, a figura 22 representa esta mensagem.

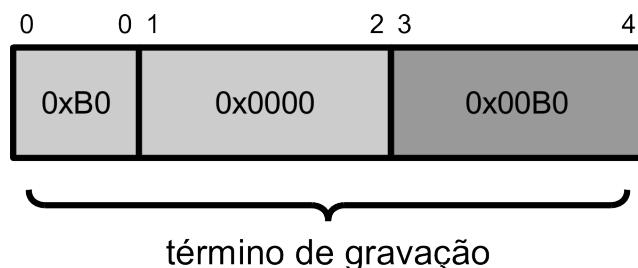


Figura 22: Estrutura de uma mensagem de término de gravação

Caso se queira enviar dados de outro botão a comunicação é reiniciada a partir da mensagem de estabelecimento de conexão.

6.2.2 Máquina de estados

A máquina de estados do protocolo de comunicação está descrito na figura 23.

Inicialmente quando solicitado a transmissão dos dados, a prancha envia a mensagem de “estabelecimento de conexão” se respondida positivamente vai para um estado se negativamente para outro. Pode ocorrer algum erro na comunicação entrando no estado final “falha de comunicação”.

Se respondido negativamente, o que pode ocorrer caso a prancha esteja emitindo algum som quando solicitado ele vai para o estado final “prancha ocupada”. No caso positivo, a estação base inicia a transferência dos dados.

Os dados quando recebidos com sucesso pelo embarcado é enviado uma mensagem “confirmação de recebimento” avisando a estação base que já pode enviar mais um segmento. Esse ciclo fica acontecendo até que depois do envio da última parte ou alguma falha ocorra.

Depois da última parte a base envia uma mensagem de “envio de segmento” com tamanho zero que é entendida pelo firmware como fim da transferência entrando assim no estado “prancha gravando”. No fim da gravação dos dados da RAM para a FLASH a prancha envia uma mensagem avisando e assim encerrando a comunicação.

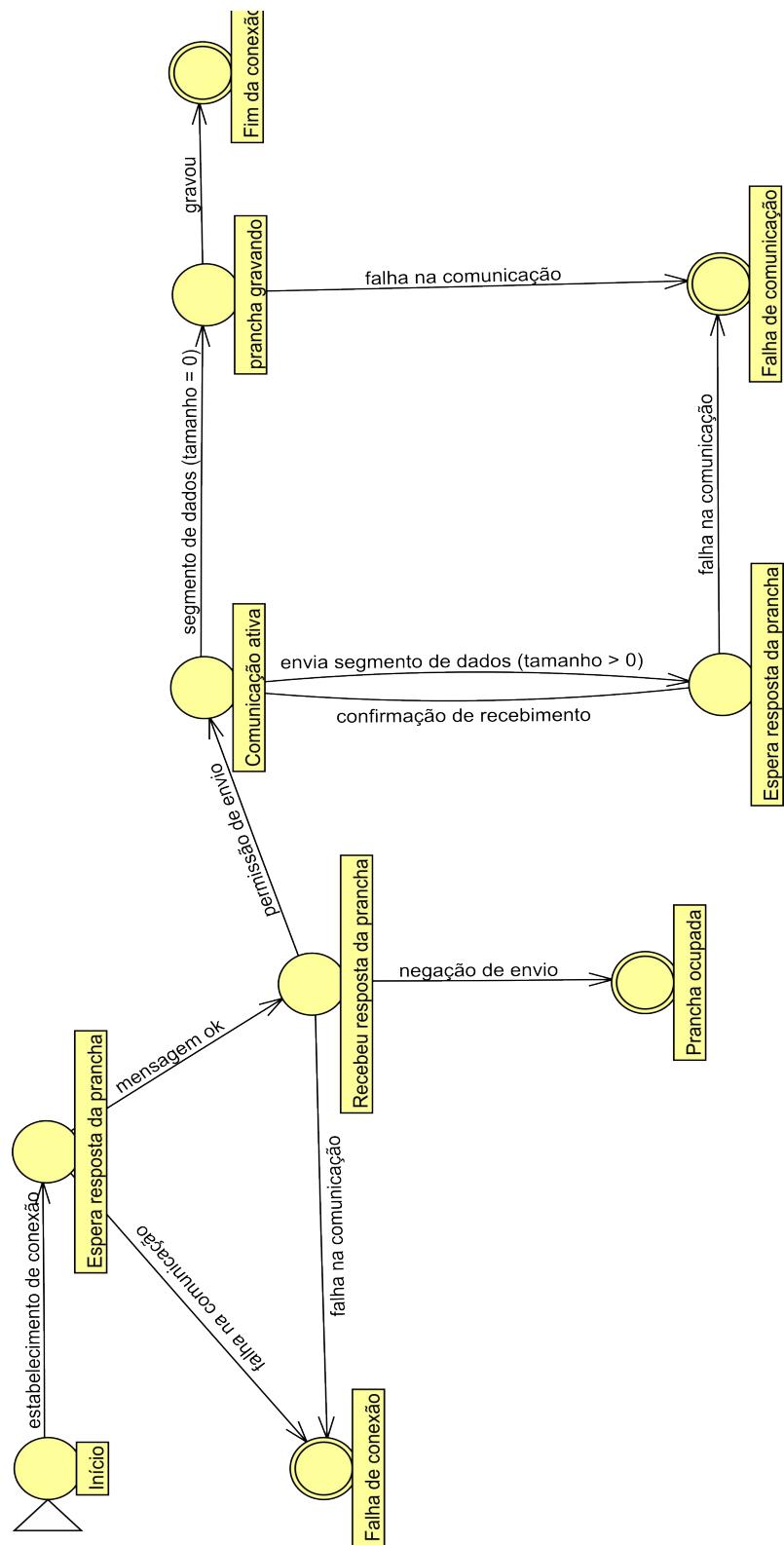


Figura 23: Máquina de estados do computador

6.3 HARDWARE

Na figura 24 está representado o diagrama de blocos do sistema, ele representa uma visão geral de como ficará a organização do circuito.

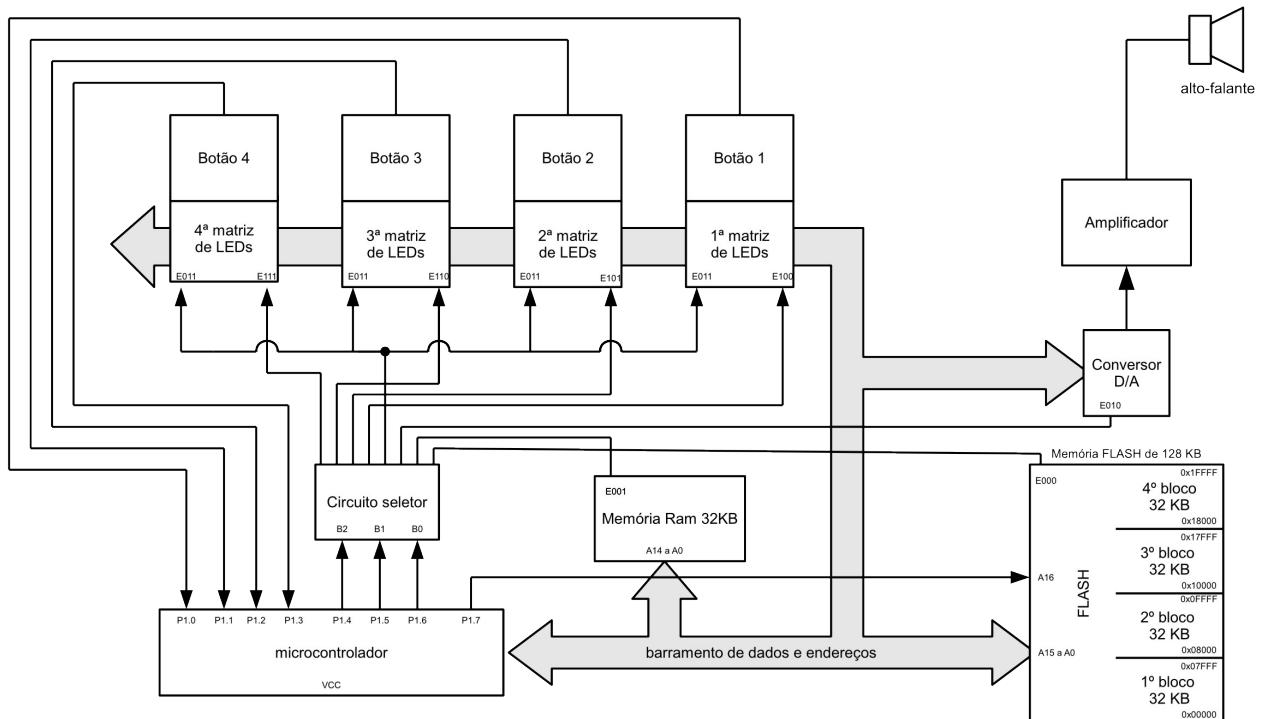


Figura 24: Diagrama de blocos do circuito

Nos tópicos seguintes será explicado com mais detalhes cada parte deste circuito e ficará mais claro o entendimento do diagrama de blocos.

A parte do hardware está dividido em: sistema de imagem, microcontrolador, circuito seletor, sistema de armazenamento e sistema de som.

6.3.1 Sistema de imagem

O sistema de imagem representa os ícones (imagens) que o sistema possui gravado na memória. Para isso utilizamos para cada ícone uma matriz de LEDs 8x8 monocromática HS-2088AS.

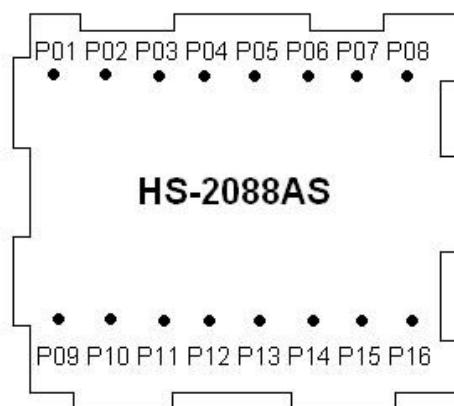


Figura 25: Matriz de LEDs HS-2088AS

Segue nas tabelas 8, 9 e 10 a configuração dos pinos da matriz de LEDs apresentada na figura 25. Como o *datasheet* fornecida pelo fabricante continha uma configuração dos pinos diferente da matriz adquirida pelo grupo, foi necessário realizar um trabalho de mapeamento dos pinos para verificar a função de cada um deles.

PINO	ÂNODO/CÁTODO	LINHA/COLUNA	POSICAO
P01	Ânodo	Linha	6
P02	Ânodo	Linha	3
P03	Cátodo	Coluna	4
P04	Ânodo	Linha	1
P05	Cátodo	Coluna	6
P06	Cátodo	Coluna	7
P07	Ânodo	Linha	2
P08	Ânodo	Linha	4
P09	Ânodo	Linha	8
P10	Cátodo	Coluna	5
P11	Cátodo	Coluna	3
P12	Ânodo	Linha	5
P13	Cátodo	Coluna	8
P14	Ânodo	Linha	7
P15	Cátodo	Coluna	2
P16	Cátodo	Coluna	1

Tabela 8: Pinagem da matriz de LEDs

Ânodos Linha	1	2	3	4	5	6	7	8
Pino	P4	P7	P2	P8	P12	P1	P14	P9

Tabela 9: Resumo da pinagem: linhas da Matriz de LEDs

Cátodos Coluna	1	2	3	4	5	6	7	8
Pino	P16	P15	P11	P3	P10	P5	P6	P13

Tabela 10: Resumo da pinagem: colunas da Matriz de LEDs

Esta matriz trabalha com o sistema de varredura, pois apenas uma linha pode estar acesa a cada instante. O sistema passa por todas as linhas muitas vezes por segundo, em um ciclo enganando nossos olhos dando a impressão que todas as linhas estão acesas a todo instante.

A este circuito estão ligados dois *latches* (74LS373N), um para a coluna e um que é comum as quatro matrizes referente a linha. Eles tem a finalidade de

armazenar a ultima informação que estava contida nele antes da matriz deixar de estar selecionada.

6.3.2 Circuito seletor

O seletor nada mais é que um CI do tipo 74LS138 que ao receber os 3 bits de entrada (B2, B1, B0) correspondente aos pinos P1.4, P1.5 e P1.6 do microcontrolador ativa o enable dos dispositivos do circuito seguindo a convenção apresentada na tabela 11:

B2	B1	B0	Dispositivo Habilitado
0	0	0	Memória Flash 128K
0	0	1	Memória RAM 32K
0	1	0	Latch do Áudio
0	1	1	Latch da Linha das Matrizes
1	0	0	Latch da Coluna da Matriz 1
1	0	1	Latch da Coluna da Matriz 2
1	1	0	Latch da Coluna da Matriz 3
1	1	1	Latch da Coluna da Matriz 4

Tabela 11: Seletor de dispositivo

6.3.3 Sistema de armazenamento

O sistema de armazenamento é composto pela memória RAM de 32 KB (HY62256) e pela memória FLASH (29F010) que é a memória permanente do sistema.

A memória RAM é utilizada para armazenamento de dados temporários e é utilizada na etapa de atualização da prancha. Enquanto a estação base não termina de enviar o botão completo (som mais ícone) a prancha armazena na memória RAM e após a transferência total os dados são mandados para a memória permanente.

O motivo para utilizar esta memória é para evitar que a prancha a cada recebimento da “mensagem de envio de dados” tenha que fazer a sequência de comandos para gravação e se ocorrer algum erro no meio da transferência o programa tem que apagar os dados já escritos para não ter um botão corrompido.

Com a memória RAM é necessário apenas um comando de escrita no final da transferência do botão e com a certeza que ele está completo sem nenhum dado faltante.

Não é utilizada a memória interna do microcontrolador por ser muito menor do que o tamanho do botão completo (32KB).

A memória permanente é uma memória FLASH de 128 KB dividida em quatro blocos de 32 KB, um para cada botão. Os primeiros 8 *bytes* de cada bloco correspondem ao ícone da matriz e o restante corresponde ao som.

6.3.4 Sistema de som

Como se trabalha com amostras de som armazenadas digitalmente, o sistema responsável pela emissão sonora deve ser capaz de converter os dados a ele enviados em sinais analógicos e amplificá-los, transformando sequências digitais em algo audível.

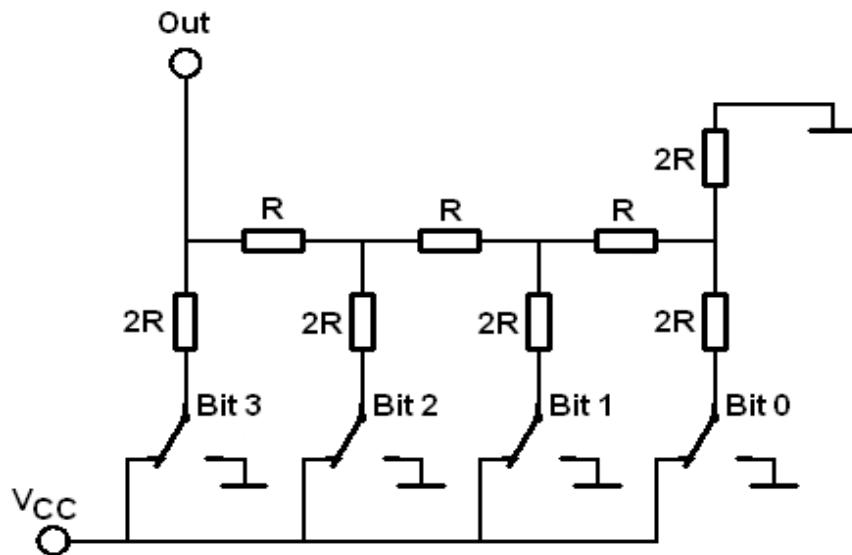


Figura 26: Rede R2R para 4 bits.

Para a etapa de conversão digital-analógica, primeiramente, analisou-se o DAC0808, contudo havia a necessidade de uma tensão negativa, o que geraria a necessidade de outra tensão de referencial, complicando o projeto. O DAC 0832, não necessitava desta tensão negativa, mas demandava uma realimentação negativa, o que exigiria um amplificador operacional atuando como inversor, o que

por sua vez exigiria a tensão negativa que tanto se desejava evitar. Por fim, entendendo melhor o processo da conversão, optou-se por uma solução mais simples, uma rede ou escada R2R. A figura 26 representa uma rede R2R para 4 bits, sendo que no nosso projeto utilizamos uma de 8 bits para maior precisão.

A rede R2R utiliza a relação de resistências supracitada e a lei da superposição para transformar chaves fechadas e abertas em valores de tensão na saída. Se uma chave se fechar somente no bit 3 na saída teremos metade da tensão total, se só o bit 2, um quarto, se só o bit um, um oitavo e se só o bit 0, um dezesseis avos. Pela lei da superposição esses valores podem ser somados podendo gerar qualquer valor de tensão entre 0 e 15/16 de Vcc e como são controlados por chaves podem responder a valores de circuitos digitais “chave fechada” e “chave aberta”, possibilitando a conversão. Seguindo recomendações, foram escolhidos os valores de 10k e 20k para resistores ambos com precisão de 1%, evitando erros significativos na conversão.

Contudo, para se utilizar deste sistema, deve-se lembrar que as portas digitais possuem um limite de corrente relativamente baixo para aplicações que envolvem som, por exemplo. Desta forma, se faz necessária a utilização de um buffer no sistema de conversão que nada mais é que um amplificador de ganho unitário à tensão, mas, capaz de fornecer mais corrente. Para tal tarefa foi avaliado o simples e barato LM741, mas este apresentou problemas, pois cortava o sinal quando operando sem tensão simétrica apenas com uma alimentação positiva e a massa. Já o CA3140, um amplificador com entrada em FET, mostrou-se perfeito à aplicação por ter uma alta resistência de entrada, não atrapalhando no circuito do R2R e operando com a faixa de sinal desejada mesmo com apenas uma tensão de alimentação e o GND. Houve ainda a recomendação da inserção de um filtro na entrada do CA3140 com um resistor de 20k e um capacitor de 47nF ligados a esta, estando a outra ponta do resistor ligada à saída do R2R e do capacitor ligado à massa. A alimentação do CA3140 ficou como a total fornecida pela pilha (6 pilhas de 1.2 V) e na saída deste foi usado um capacitor de acoplamento de $10\mu\text{F}$. Assim o circuito responsável pela conversão ficou como esta demonstrado na figura 27.

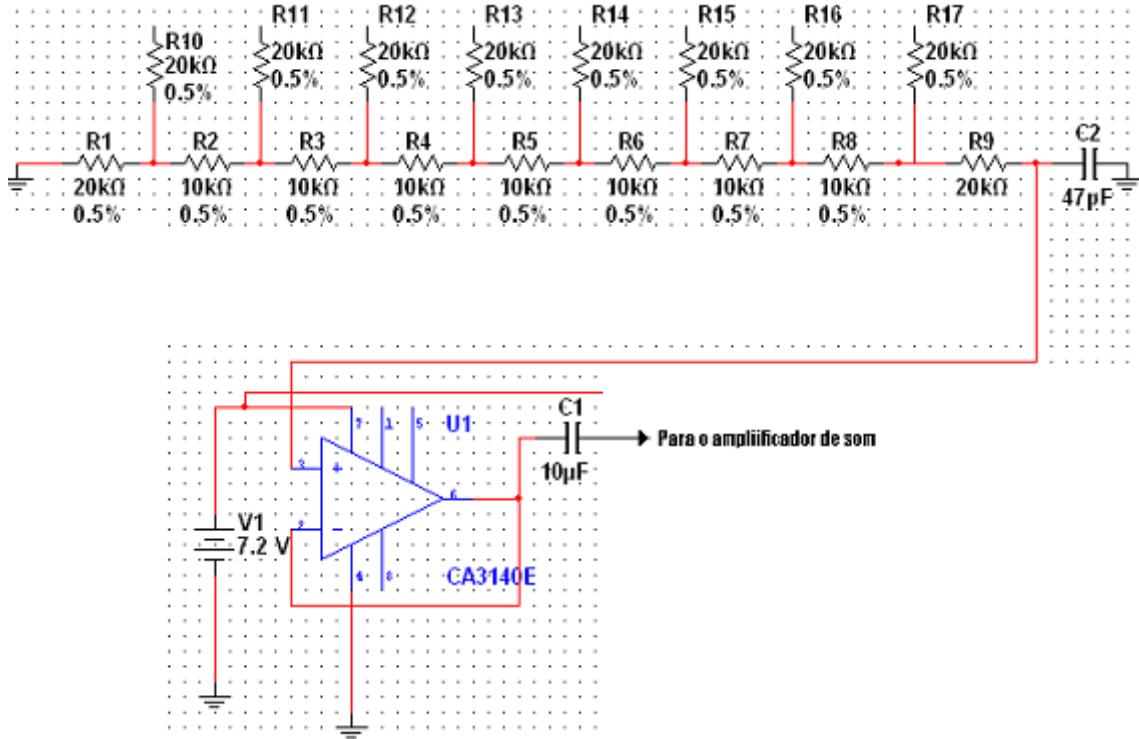


Figura 27: Circuito de conversão de dados em sinais.

Com relação à tensão, o circuito de conversão, por si só, já emitia sinais de aproximadamente 5 Vpp, com uma carga de alta impedância na saída, mas, como o auto falante escolhido possui apenas 8Ω , não seria transferida a ele um sinal de tensão com uma potência suficiente para que saísse dele som. Então foi utilizado um amplificador de áudio. Avaliando as opções disponíveis, o primeiro CI encontrado foi o LM386. Contudo a potência dele não passava de 0.25W, praticamente inaudível, em um ambiente com mínimo ruído. Para obter mais volume, encontrou-se uma alternativa interessante: o TDA7052, amplificador de 1W de relativo baixo custo e fácil operação. Assim o circuito para sua operação foi montado, seguindo as recomendações relativas a aplicações típicas do *datasheet* [11].

Seu único problema se resumia no fato do ganho máximo não ser regulável com realimentação, estando fixo em 30 dB. Como se procurava utilizar nele um controle de volume, o volume fixado como máximo, deveria corresponder à posição de maior resistência do potenciômetro instalado no controle DC de volume, logo ao ganho máximo de 30 dB ou de aproximadamente 32 vezes. Como se desejava uma saída de 7.2 Vpp a entrada deveria estar em no máximo 225 mVpp, portando uma

divisão em pouco menos de 10 vezes. Para tanto um resistor de 42k na saída do estágio de conversão associado em série com o recomendado de 5k para entrada do amplificador de potência, foi utilizado para contornar o problema. O circuito completo ficou como está representado pela figura 28.

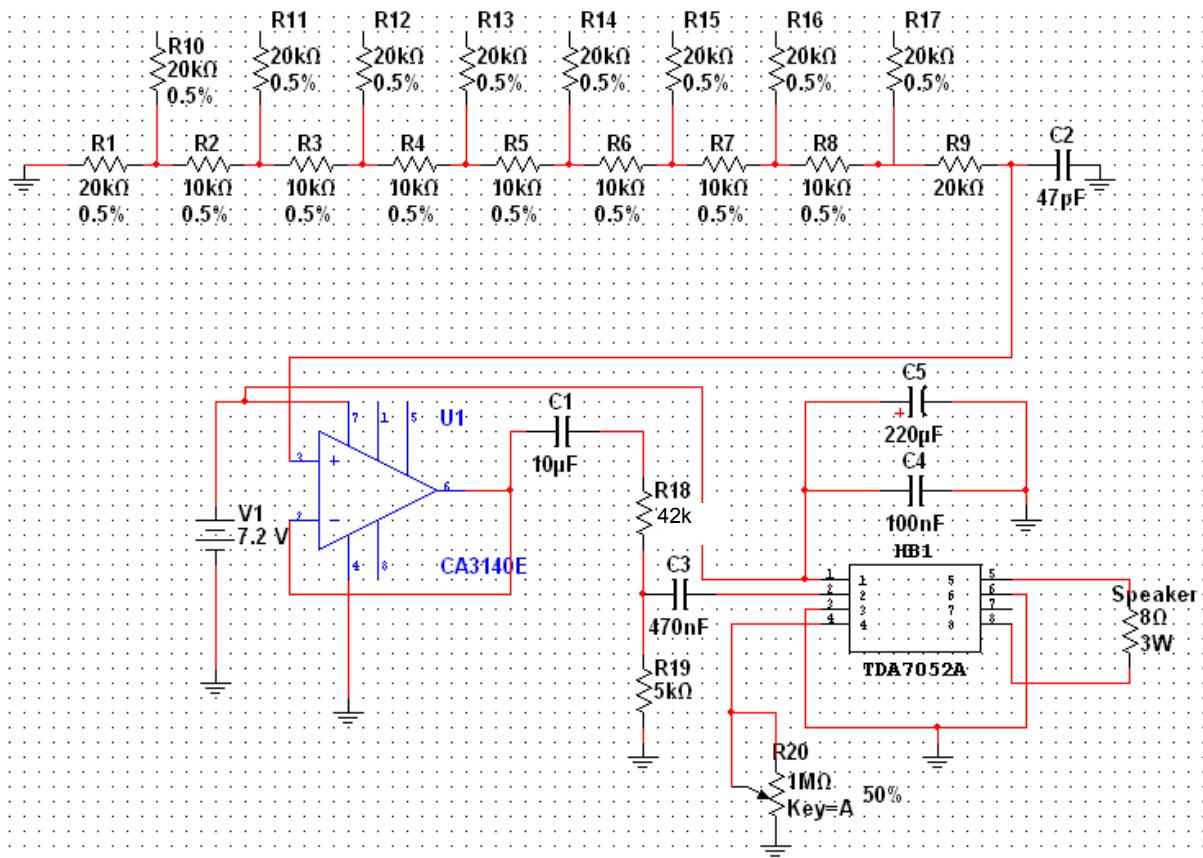


Figura 28: Circuito relativo à conversão digital analógica e amplificação do som.

6.3.5 Microcontrolador

O microcontrolador utilizado nesse projeto foi o Atmel 89S52 [5] pelos motivos explicados no estudo da alternativas tecnológicas.

A principal função do microcontrolador é coordenar todos os dispositivos que estão interligados pelo barramento principal e os que estão conectados por algum port. Nele está contido o programa que controla a entrada e a saída do barramento de dados e endereços, além de coordenar a comunicação com a estação base.

Do Port P0 sai o barramento de dados que conecta todos os 8 principais

periféricos do circuito (retirando a parte de comunicação serial e de alimentação):

- **Memória FLASH:** ela possui um barramento de dados (entrada e saída) conectados diretamente no PORT P0, sendo o barramento com os 8 primeiros bits (A0-A7) do endereço conectado a um CI 74LS373 (*latch*) que recebe o endereço do microcontrolador pelo PORT P0 quando o pino de ALE é ativado segurando o valor que estava sobre ele antes do evento [12]. O barramento com os bits de endereçamento de A8 a A15 que é conectados ao PORT P2 e um bit de endereço mais significativo (A16) conectado ao pino P1.7 do Port P1. Os pinos de Read e Write são conectados diretamente aos pinos de Read e Write do microcontrolador At89S52 P3.6 e P3.7, e o chip enable é conectado a saída Out0 do 74LS138 (Seletor) que é ativado quando uma instrução utiliza comandos de mapeamento da memória externa do microcontrolador.
- **Memória RAM:** seus bits de endereçamento são iguais aos da memória FLASH, mas sem possuir o ultimo bit do Port P2 (P2.7) e sem utilizar o pino P1.7 pois a mesma possui 32KB de memória. O chip enable dela é conectado a saída Out1 do seletor.
- **Latch de áudio:** ele armazena os dados que serão repassados para o conversor D/A (uma rede R2R no caso) para ser utilizado para emissão dos sinais sonoros. Seu *latch* Enable é conectado a saída Out2 do seletor.
- **Latch do seletor de linhas:** ele armazena a informação de qual linha deve ser setada para todas as matrizes e tem seu *latch* enable conectado na saída Out3 do seletor, lembrando que todos os *latches* (74LS373) por terem o *latch* enable ativo em nível lógico alto estão conectados a um CI de porta NOT logo após a saída do seletor e que sempre estão com o Output enable ativados.
- **Latch da coluna da matriz 1:** Ele armazena a informação da coluna

representada na matriz de LEDs 1 e seu *latch enable* é conectado na saída out4 do seletor.

- **Latch da coluna da matriz 2:** basicamente igual ao anterior, porém armazenando a informação das colunas da matriz 2 e seu *latch enable* é conectado na saída out5 do seletor.
- **Latch da coluna da matriz 3:** basicamente igual ao anterior, porém armazenando a informação das colunas da matriz 3 e seu *latch enable* é conectado na saída out6 do seletor.
- **Latch da coluna da matriz 4:** basicamente igual ao anterior, porém armazenando a informação das colunas da matriz 4 e seu *latch enable* é conectado na saída out7 do seletor.

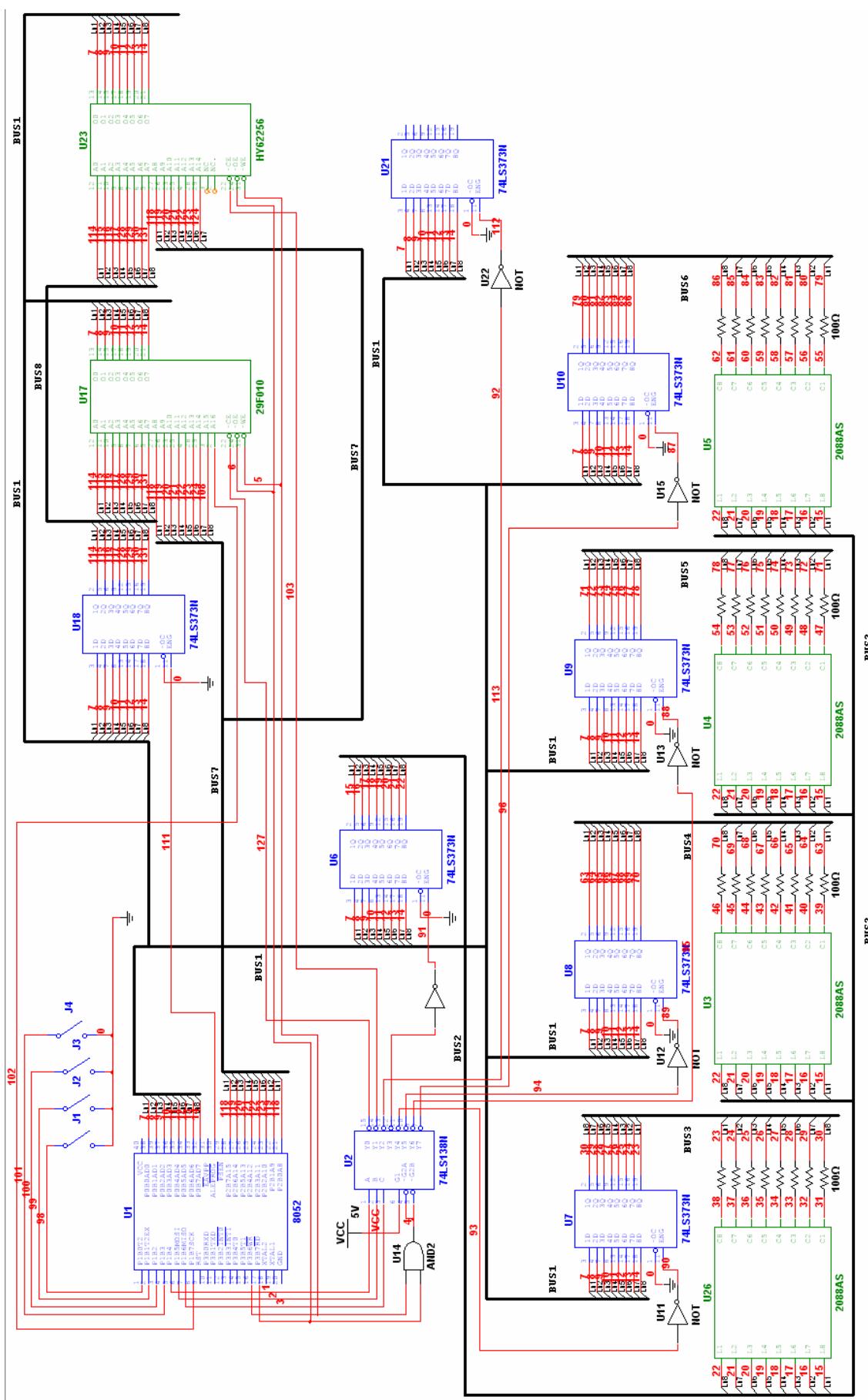


Figura 29: Diagrama do circuito.

Os botões de cada matriz estão ligados a um pino do Ports P1, mais precisamente nos 4 primeiros pinos (P1.0 a P1.3) e representam as matrizes (1 a 4), na ordem respectiva, e são ativados em nível lógico alto (5 Vcc neste caso). O diagrama geral do circuito, sem algumas conexões do Microcontrolador como regulador de tensão e o MAX232 para comunicação serial, encontra-se na figura 29. Estes outros dispositivos utilizados fazem parte do Kit P51 V2.0 que foi utilizado como placa para o microcontrolador e será melhor detalhado na sessão 8.

Abaixo se encontra um resumo do mapeamento dos Ports do microcontrolador e sua descrição:

P0.0 - D0 de todos os dispositivos / A0 da RAM e da Flash

P0.1 - D1 de todos os dispositivos / A1 da RAM e da Flash

P0.2 - D2 de todos os dispositivos / A2 da RAM e da Flash

P0.3 - D3 de todos os dispositivos / A3 da RAM e da Flash

P0.4 - D4 de todos os dispositivos / A4 da RAM e da Flash

P0.5 - D5 de todos os dispositivos / A5 da RAM e da Flash

P0.6 - D6 de todos os dispositivos / A6 da RAM e da Flash

P0.7 - D7 de todos os dispositivos / A7 da RAM e da Flash

P1.0 - Entrada do botão 1

P1.1 - Entrada do botão 2

P1.2 - Entrada do botão 3

P1.3 - Entrada do botão 4

P1.4 - B0* do seletor de periféricos

P1.5 - B1* do seletor de periféricos

P1.6 - B2* do seletor de periféricos

P1.7 - MSB da memória Flash (A16)

P2.0 - A8 da RAM e da Flash

P2.1 - A9 da RAM e da Flash

P2.2 - A10 da RAM e da Flash

P2.3 - A11 da RAM e da Flash

P2.4 - A12 da RAM e da Flash

P2.5 - A13 da RAM e da Flash

P2.6 - A14 da RAM e da Flash

P2.7 - A15 da Flash

P3.0 - RXD da comunicação serial

P3.1 - TXD da comunicação serial

P3.2 – Não utilizado

P3.3 – Não utilizado

P3.4 – Utilizado na comunicação serial

P3.5 – Utilizado na comunicação serial

P3.6 – Pino \neg WR utilizado para sinalizar escrita

P3.7 – Pino \neg RD utilizado para sinalizar leitura

7 CRONOGRAMA

Planejamos o nosso cronograma elaborando um diagrama de Gantt, utilizando o software OpenProj [14]. O planejamento inicial está localizado em anexo na pasta 'Cronograma' arquivo 'Fun_Icon (Planejado).pod'.

Comparando com o arquivo final 'Fun_Icon (Executado).pod' pode-se fazer uma comparação do tempo que levamos em cada etapa. Alguns itens foram excluídos do nosso projeto como a criação de uma estrutura para a prancha e o circuito de detecção de bateria fraca, devido a falta de tempo para terminar estas tarefas e por terem uma importância reduzida foram retiradas.

Ocorreram atrasos em algumas etapas, ocasionadas pelo mau planejamento do tempo. Principalmente pela pequena relevância dada a problemas com trabalhos de outras disciplinas e semana de provas que em conjunto atrapalharam bastante o acompanhamento do cronograma.

8 DESENVOLVIMENTO PRÁTICO

O planejamento descreveu o hardware o software agora vamos ver o que foi implementado e montado a partir dele e as principais dificuldades em cada etapa do desenvolvimento.

8.1 ESTAÇÃO BASE

O desenvolvimento prático da estação base foi dividida em dois: software de interface e software do sistema embarcado.

Existe uma comunicação entre estes dois softwares, comunicando a estação base à prancha. A comunicação segue o protocolo descrito no planejamento e é utilizado o protocolo serial como determinado no estudo das tecnologias.

8.1.1 Software de interface

O software de interface possui as funcionalidades necessárias para que os casos de uso da figura 1 serem satisfeitos. A linguagem de implementação utilizada é a linguagem Java, e o ambiente de desenvolvimento escolhido foi o Eclipse.

Quando o arquivo está carregando os dados aparecem em uma janela com uma mensagem de inicialização do programa que é mostrada na figura 30.

A primeira janela depois da mensagem que o programa abre é a “Janela Principal” (figura 31), nela existem dois painéis com uma coleção de funções associadas a cada um deles. O primeiro painel “Biblioteca de Botões” envolve miniaturas de ícones com título de botão acima do ícone e título de som abaixo dele. As miniaturas representam os botões que estão atualmente armazenados como arquivos “.fib” na memória do computador (mais especificamente na subpasta “botoes” presente na pasta em que foi executado “FunIconGUI.jar”). O segundo painel “Prancha” compreende miniaturas de botões que estão presentes na “Biblioteca de Botões”, sua funcionalidade é representar um projeto da configuração da prancha (4 botões, com ícones e sons associados). Com relação ao primeiro painel “Biblioteca de Botões”, pode-se acionar as funções: “Criar botão”, que imediatamente abre a Janela de Edição de Botão com todos os parâmetros nulos;

“Ouvir Som”, que reproduz o som associado à miniatura selecionada; “Editar Botão”, o qual abre a Janela de Edição de Botão com os parâmetros associados ao botão que fora selecionado; “Clonar Botão”, função que faz uma cópia do botão selecionado (adicionando ao final do título do botão a string “(clonado)”; “Apagar Botão”, função que deleta permanentemente a miniatura de botão e o arquivo “.fib” associado; “Adicionar à prancha”, função que transfere a miniatura de botão selecionada ao painel “Prancha”. Com relação ao segundo painel “Prancha” temos as seguintes funções associadas: “Remover da Prancha”, que remove a miniatura selecionada do projeto de prancha; “Ouvir som”, função que reproduz o som associado à miniatura selecionada; “Copiar botão para biblioteca”, função que faz uma cópia do botão selecionado no painel “Prancha” para o painel “Biblioteca de Botões” (adicionando ao final do título do botão a string “(importado)”; “Gravar na prancha real”, função que transfere os dados relativos ao projeto de prancha atualmente carregado para o microcontrolador; “Salvar prancha”, função que grava em um arquivo no computador os dados do projeto de prancha em uso (armazenados em conjunto como um arquivo “.fip” na subpasta “prancha”); “Carregar prancha”, que requisita uma especificação da localização do arquivo de prancha a ser carregado no painel de projeto de prancha.

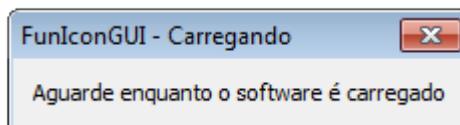


Figura 30: Mensagem de inicialização do programa

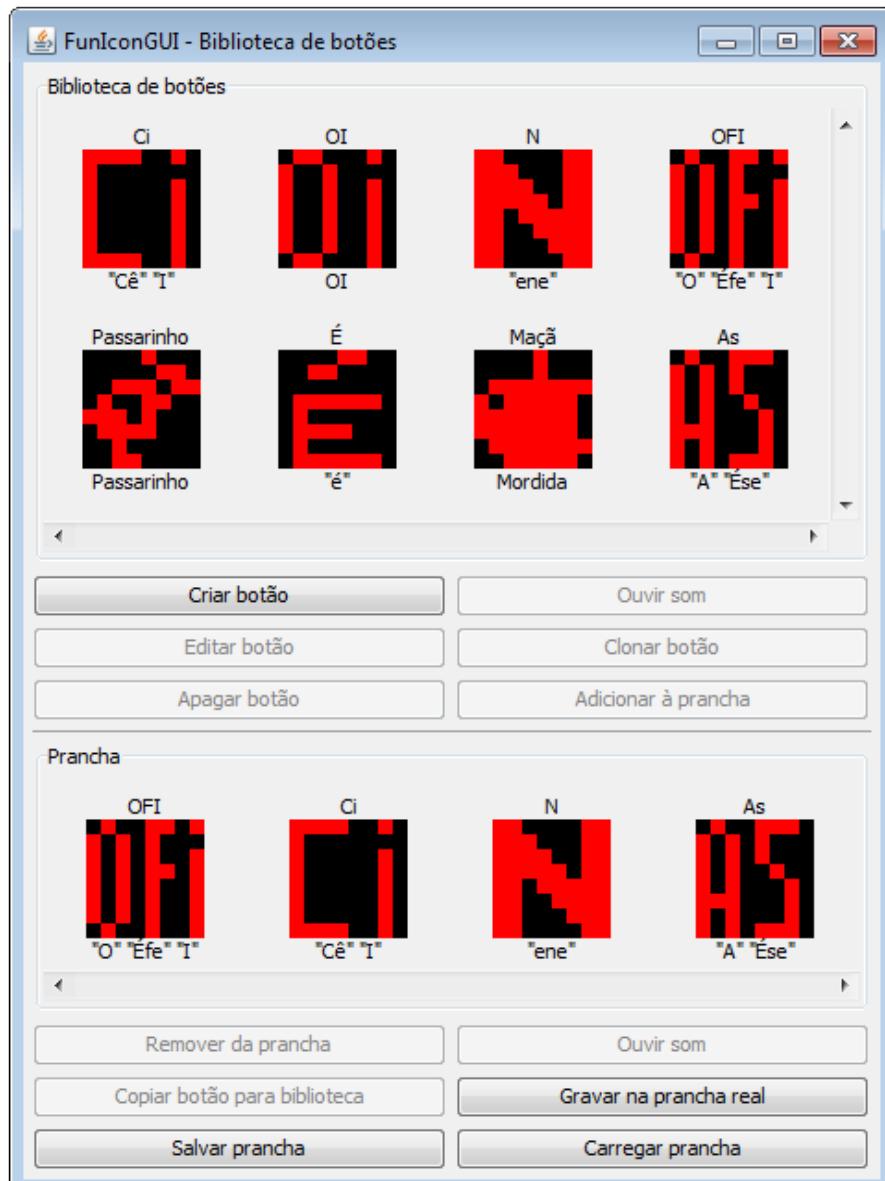


Figura 31: Janela principal do programa da estação base “FunIconGUI”

A segunda janela (figura 32) que abre quando usa-se o programa é a “Janela de Edição de Botão”. Nela existem quatro painéis: “Nome do botão”, no qual pode-se escrever um título desejado para o botão (funcionalidade que foi implementada para melhor organizar os botões no programa); “Ícone”, em que há uma imagem/matriz de oito linhas e oito colunas, oferecendo ao usuário a possibilidade de edição dela através de cliques do mouse; “Nome do som”, mesma funcionalidade do primeiro painel só que direcionada a nomear o som armazenado; “Som”, painel que dispõe

um gráfico de amplitude (eixo vertical) por tempo (eixo horizontal) das amostras do som. Associados aos painéis, existe uma coleção de funções: “Gravar”, funcionalidade que adquire sinais de som através um microfone periférico; “Ouvir”, função que reproduz o som atual; “Importar”, função que disponibiliza para a memória do programa um arquivo de áudio especificado pelo usuário; “Exportar”, função que grava o som presente na memória do programa em um arquivo de formato “.wav” com título e localização especificados pelo usuário; “Cancelar”, ação que elimina as configurações atuais do botão, retornando à “Janela Principal”; “Salvar”, ação que agrupa as configurações ajustadas pelo usuário e grava os dados binários em um arquivo “.fip”.

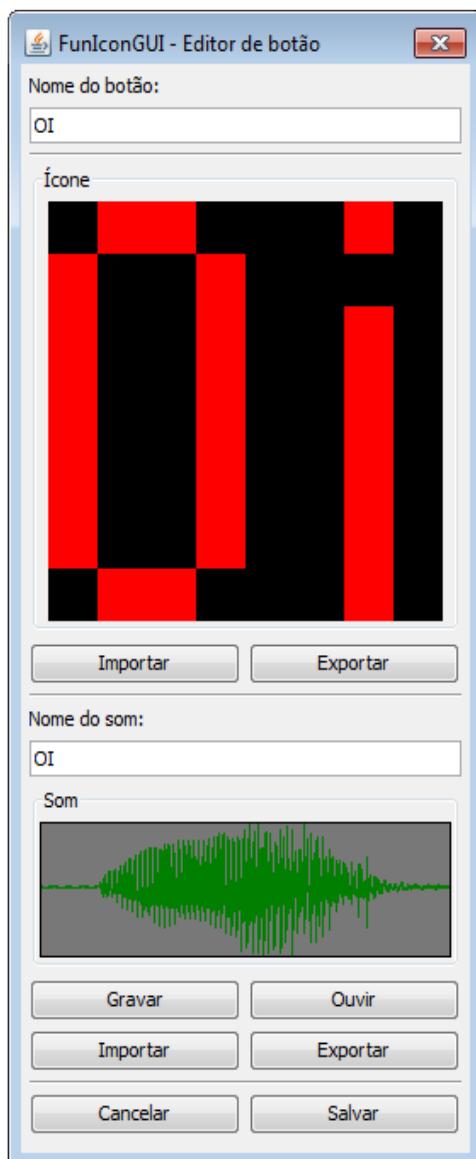


Figura 32: Janela de edição de botão do programa da estação base

Como foi dito, o software têm dois formatos de arquivo próprios, o formato “.fib” (FunIcon – botão) e “.fip” (FunIcon – prancha), que armazenam, respectivamente, informação de um botão, e de uma prancha (quatro botões). Os arquivos de botão têm a seguinte estrutura: uma *cstring* (cadeia de caracteres codificada em ASCII terminada em *byte* 0) com os caracteres “FIB1.0”, seguida de uma cadeia de caracteres codificada segundo os padrões da linguagem Java (dois bytes para o tamanho da cadeia, seguidos da cadeia em codificação UTF-8

modificada) com o nome do botão, uma cadeia de caracteres codificada segundo os mesmos padrões com o nome do som, em seguida 64 *bits* (8 *bytes*) com o ícone (cada *byte* é uma coluna, sendo o primeiro *byte* a primeira linha, e o *bit* menos significativo de cada *byte* o pixel mais à esquerda), e finalmente o som (codificado da seguinte forma: um short com o número de amostras, seguido das amostras, um byte cada). Os arquivos de prancha são formados da *cstring* “FIP1.0” seguida de um *byte* com a contagem de botões (maior que zero, menor que quatro), e os dados crus dos botões (idênticos aos conteúdos de seus arquivos “.fib”).

Quando mandamos o programa gravar na prancha real, estamos querendo iniciar uma comunicação do software da base para o embarcado. Lembrando que esta etapa satisfaz o protocolo de comunicação apresentado no planejamento.

Antes de iniciar a comunicação o usuário escolhe a porta serial que deseja utilizar. Através da janela apresentada na figura 33. Ou pode ocorrer um erro caso não tenham portas seriais disponíveis (figura 34).

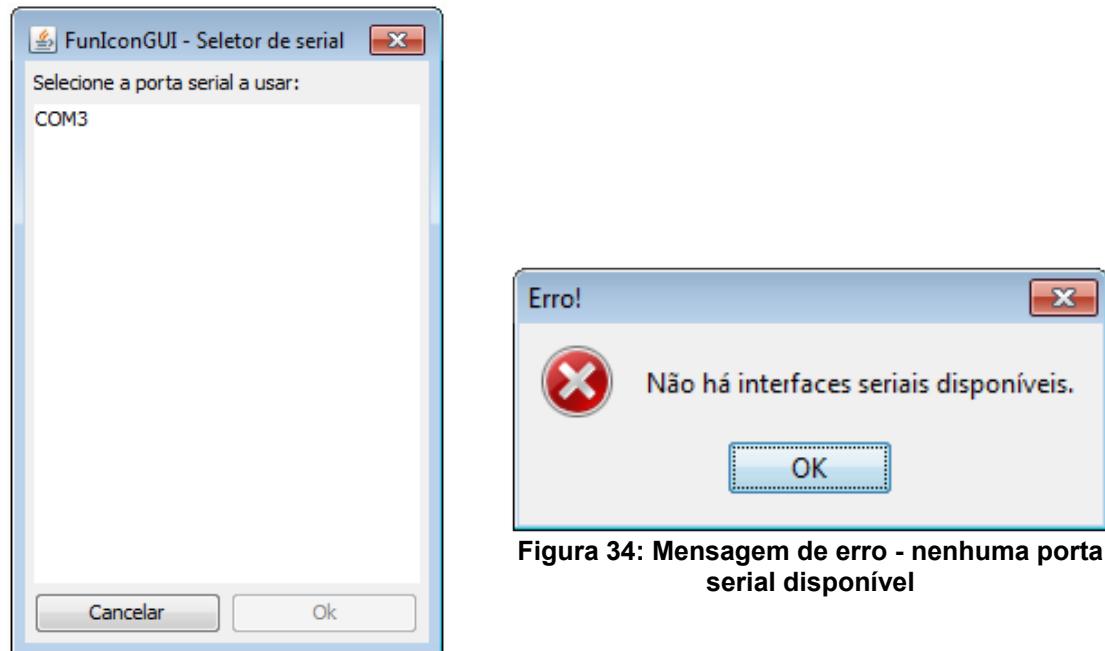


Figura 33: Janela de seleção da porta serial

Ao iniciar a transferência uma janela de progressão acompanha o andamento

da transmissão dos dados (figura 35).

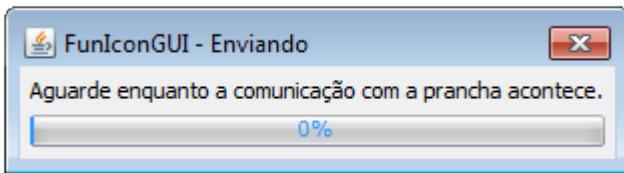


Figura 35: Janela que indica a progressão da transmissão dos dados

O código-fonte está documentado utilizando o recurso javadoc de documentação e se encontra na pasta Software em anexo a este documento. Nele se encontra uma descrição geral de cada função e das variáveis que o software possui.

8.1.2 Software do sistema embarcado

Nesse documento, o funcionamento do *software* embarcado não será esmiuçado em nível de código-fonte. Entretanto, com as explicações aqui fornecidas, os diagramas, e o código-fonte comentado (todas as funções e variáveis são descritas junto ao código-fonte), acredita-se que não haverá problemas em compreender seu funcionamento. O *firmware* do projeto é programado em linguagem C, e é dividido em módulos, cada qual associado a um arquivo de cabeçalho e outro de código-fonte correspondente. Os módulos não são completamente independentes, e são relativos às funcionalidades principais do software embarcado.

O módulo principal do *firmware* é o único que não é associado a um arquivo de cabeçalho, sendo associado apenas ao arquivo “main.c”. Ele apenas conta com a inclusão dos demais módulos e com o código principal do sistema, e basicamente contém inicialização das variáveis e os laços (*loops*) associados a cada estado do *firmware*. O tratamento de interrupções é realizado no código relativo aos arquivos “interrupts.h” e “interrupts.c”. Há um módulo de software, onde são declaradas as variáveis e macros de uso geral, que é incluído em todos os demais, associado aos arquivos “variaveis.c” e “variaveis.h”. Há ainda um módulo relativo às funções de

varredura das matrizes de LEDs, nos arquivos “varredura.h” e “varredura.c”. O módulo com as funções relativas à escrita na memória FLASH encontra-se nos arquivos “flash.h” e “flash.c”. Finalmente, o módulo com as funcionalidades de comunicação pela porta serial estão nos arquivos “comunicacao.h” e “comunicacao.c”.

A única interrupção utilizada no sistema é a do *timer 0* que, quando ativada, é disparada a cada 125 μ s, intervalo esse escolhido por ser o necessário para a emissão de um som a 8000 amostras por segundo. No tratamento da interrupção, não é executado o código de varredura ou emissão de som; alternativamente, ativa-se uma *flag* indicando o acontecimento de uma interrupção de *timer*, e trata-se sua ocorrência no laço principal. Esse método foi escolhido visto que o compilador de C geraria um grande overhead com salvamento e recarregamento dos registradores a cada interrupção, o que inviabilizaria a emissão das amostras de som e a varredura em tempo útil. O *timer 0* é usado no modo temporizador com recarga automática, e dispara a cada 115 ciclos de máquina, o que equivale a um tempo de 124,79 μ s com um cristal externo de 11,059 MHz, sendo esse o valor mais próximo de 125 μ s alcançável usando tal cristal, com um erro de aproximadamente 0,17 %. Dessa forma, a amostragem do som acaba sendo, na verdade, de aproximadamente 8014 amostras por segundo. O código de varredura das matrizes de LEDs e de emissão das amostras de som teve de ser bastante otimizado, visto que deveria ser executado completamente em menos de 115 ciclos de máquina. Testes em simulador sugerem que foi possível otimizar esses códigos a ponto que fossem realizados em menos de 100 μ s.

Após a prancha ser inicializada, entra-se no estado “espera”. No laço principal desse estado, verifica-se por *pooling* se os botões associados a cada matriz de LEDs foram pressionados, e verifica-se também a *flag* de recepção serial e a *flag* de interrupção de *timer*. Quando a *flag* de interrupção de *timer* está ativa, executa-se o código de continuidade da varredura. Quando um dos botões é pressionado, entra-se no estado “soando”, e quando acontece um recebimento de byte pela serial, entra-se no estado “comunicação”. No laço principal do estado “soando”, verifica-se continuamente a *flag* de interrupção de *timer* e a *flag* de término

de reprodução. Quando a *flag* de interrupção *timer* está ativa, continua-se a verredura da matriz selecionada e a emissão de seu som. No estado de comunicação, recebem-se e enviam-se *bytes* pela serial de acordo com os padrões estipulados no capítulo desse documento sobre o protocolo de comunicação, e, se a comunicação ocorre satisfatoriamente, grava-se o recebido na FLASH. Em conformidade com a máquina de estados apresentado na figura 12.

O procedimento adotado para a varredura das matrizes de LEDs é puxar para a RAM interna do microcontrolador todas as grades dos ícones, de forma que o acesso às mesmas seja bastante veloz; a cada intervalo do *timer*, exibe-se uma única linha da varredura. Durante a emissão de som, as amostras de áudio são lidas diretamente da memória externa e mandadas para o *latch* da rede R2R, e apenas a matriz relativa ao botão pressionado é exibida.

A recepção de dados pela serial é feita por *pooling*, verificando continuamente a *flag* RI (de recepção). Um procedimento semelhante é adotado na escrita pela serial, esperando a *flag* TI (de envio) ser ativada após comandar o envio de cada *byte*. O código de recepção, especificamente, envolve a espera pela recepção até um tempo limite, ativando uma *flag* de falha se esse tempo passar sem que haja recebimento. Todas as temporizações usadas na comunicação são feitas com o uso do *timer* 0, exceto a temporização do *baud rate* da comunicação serial, que é feita com uso do *timer* 1. A comunicação serial é adequadamente configurada para a comunicação acontecer com *baud rate* (taxa de transições) de 57600 *bauds* por segundo, sem bit de paridade (o que permite uma taxa de transferência máxima de aproximadamente 5760 *bytes* por segundo).

8.2 PRANCHA

8.2.1 SISTEMA DE SOM

O primeiro obstáculo a ser ultrapassado para a construção da placa foi a escolha do sistema de conversão seguindo as limitações de tensão exclusivamente positiva e funcionasse de forma paralela. Após a escolha da rede R2R foram feitos testes em protoboard. Na primeira experimentação houveram dificuldades com irregularidades no valor de tensão. Descobriu-se então que havia erros de mal

contato e curtos na placa de protótipo. Substituída a protoboard, foi feita uma segunda bateria de testes na qual foi possível verificar o funcionamento do sistema.

Elaborado o sistema de conversão, foi testado o LM741 como buffer, aplicando-se uma onda senoidal em sua entrada não-inversora e verificando a saída. Mas sem a alimentação simétrica, o sinal de saída apresentou cortes. Num segundo teste, agora com o CA3140, este problema não ocorreu.

Com o buffer escolhido, foi feito um teste integrando este ao amplificador LM386 e ao auto falante. A altura do som na saída, correspondeu à frequência de entrada, contudo o som era praticamente inaudível, mesmo em um ambiente relativamente silencioso. Após pesquisas, foi feito o teste com o TDA7052, e o resultado foi satisfatório. A integração à R2R foi quase que imediata. Utilizando um Arduíno foi colocada na entrada da rede um sinal senoidal de 1 kHz amostrado em 6 kAmostras/s e foi constatado o funcionamento.

O processo de passagem para a placa universal incluiu a integração de um 74LS373, CI com 8 *latches* muito usado no projeto, na entrada da rede R2R. A primeira placa, pela inexperiência do soldador, apresentou erros de solda, curtos e resistências indesejáveis. Mas um segundo protótipo, elaborado com o auxilio de um membro da equipe mais habituado com o processo de solda, apresentou sucesso logo nos primeiros testes, não tendo um volume muito alto, contudo, satisfatório para o projeto.

A placa final do sistema de som é apresentada na figura 36.

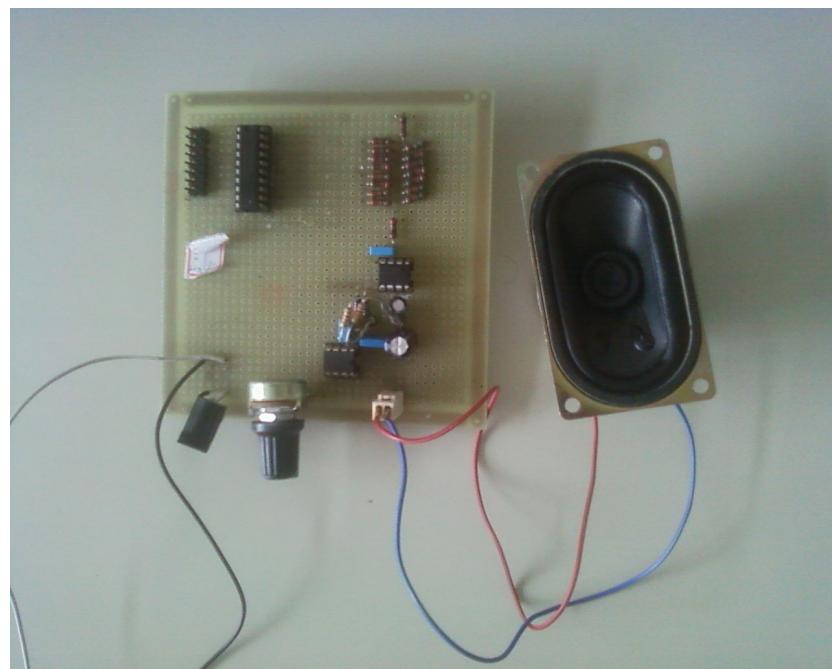


Figura 36: Placa do sistema de som

8.2.2 PLACA DO MICROCONTROLADOR

A placa utilizada pelo microcontrolador foi uma placa de montagem relativamente simples, pois o mesmo foi soldado no KIT P51 V2.0 projetado pelo Prof. Dr. Douglas Renaux e pelo Eng. Eduardo Bregante e cuja placa de circuito impresso confeccionada foi obtida através do Prof. Dr. Heitor Silvério Lopes. O Esquemático do KIT P51 V2.0 pode ser visto com mais detalhes no Manual Da Placa P51 [18].

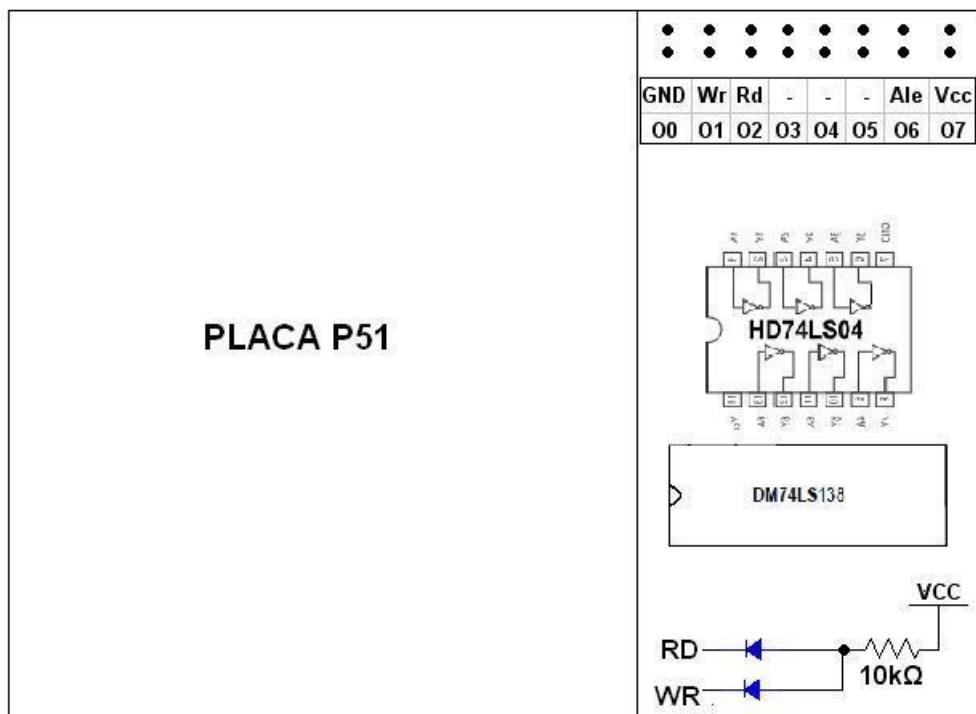


Figura 37: Diagrama da placa do microcontrolador



Figura 38: Foto da placa do microcontrolador montada

Apesar de usarmos o KIT P51 para facilitar na montagem do circuito algumas

coisas foram alteradas no nosso projeto. Primeiramente não foram utilizados os Cls 74HC373 (Latch) e o 62256 (memória RAM) e foram adicionados ao circuito da placa alguns componentes na área de prototipagem. Foram conectados os pinos de Read Enable e Write Enable (P3.6 e P3.7) através de uma porta lógica AND (feita com diodos) no “chip select” do seletor (74LS138) e a entrada do mesmo é feita através dos pinos P1.4, P1.5 e P1.5 (A, B e C do seletor respectivamente). As saídas de Out2 a Out7 do 74LS138 são ligados ao CI 74LS04 (CI com 6 Portas NOT). Além disso foram soldadas 2 barras de pinos macho para conectar esta parte dos circuito com as outras placas através de cabos FLAT. A figura 37 apresenta um esquemático geral da disposição dos dispositivos adicionados na área de prototipagem do KIT P51, as conexões entre os dispositivos segue o diagrama do circuito na figura 29. A figura 38 apresenta uma foto de como ficou a placa do microcontrolador pronta.

Na Figura 37 os pinos O0 a O7 representam as saídas Out0 a Out7 do 74LS138, lembrando que de O2 a O7 eles primeiramente vão para o 74LS04 (Porta NOT). Os pinos de Gnd e Vcc são a alimentação (0 e 5V), Wr e Rd representam o Write Enable e o Read Enable (P3.6 e P3.7) do AT89S52 e o ALE é o pino de Ale do microcontrolador.

As maiores dificuldades encontradas nesta placa foi por conta dos 2 Cls que não possuem uma pinagem que facilite muito na soldagem, na verdade esta foi basicamente a maior dificuldade de toda a parte de hardware do projeto no que diz respeito ao desenvolvimento prático. Como a soldagem foi feita com muita cautela e sempre medindo a continuidade do circuito ela não chegou a apresentar problemas posteriormente, e para evitar possíveis quebras de fios o grupo optou por utilizar cabos flexíveis durante a soldagem que apesar de serem mais trabalhosos e grandes na hora de soldar eles costumam ser mais resistentes e apresentar menos problemas desta natureza quando a placa é muito manuseada e transportada.

8.2.3 PLACA DE CONEXÃO

A placa de conexão é apenas uma placa que basicamente recebe os cabos FLAT da placa do microcontrolador e distribui para as outras placas. Uma foto da

placa de conexão pode ser vista na figura 39.



Figura 39: Foto da Placa de Conexão

8.2.4 PLACA DAS MEMÓRIAS

A placa das memórias, que na verdade também possui o *latch* (74LS373) que guarda a informação das linhas que vão para as placas de matrizes. Na placa encontram-se os Cis da memória RAM de 32KB (HY62256ALP) e da memória Flash de 128KB (AM29F010). Além disso nesta placa também encontra-se outro *latch* 74LS373 que serve para reter os 8 LSB dos endereços das 2 memórias, lembrando que ele é necessário por conta do PORT P0 ser utilizado tanto para endereçamento quanto para dados e por isso o Latch Enable (LE) dele está conectado com o pino ALE do microcontrolador. O Diagrama da disposição dos CIs na placa pode ser visto na figura 40 e a Foto da placa montada pode ser vista na figura 41.

No canto superior esquerdo da Figura 40 temos os pinos: Vcc (alimentação), Wr (Write Enable do AT89S52), Rd (Read Enable do AT89S52), Ale (do 89S52), O0 (Out0 do 74LS138), O1 (Out1 do 74LS138), O3 (\neg Out3 do 74LS138), GND (alimentação), P2.0 a P2.7 (PORT P2 do AT89S52). No canto inferior esquerdo temos os pinos P0.0 a P0.7 (PORT P0 do AT89S52). No canto superior direito temos L0 a L7 (saída do Latch 74LS373 que funciona como seletor de linha) e o P1.7

(saída do PORT P1 do AT89S52 que funciona como A16 para a memória Flash).

Essa provavelmente é a placa que deu mais trabalho na hora de soldar por conta da quantidade de pinos existentes nas memórias e pela disposição não amigável dos pinos destes Cls. Esta sem dúvida seria uma das placas que deveriam ter sido feitas em uma placa de circuito impresso, entretanto, por falta de tempo pois tínhamos um cronograma bem apertado não foi possível mandar fazer a PCI em um tempo hábil.

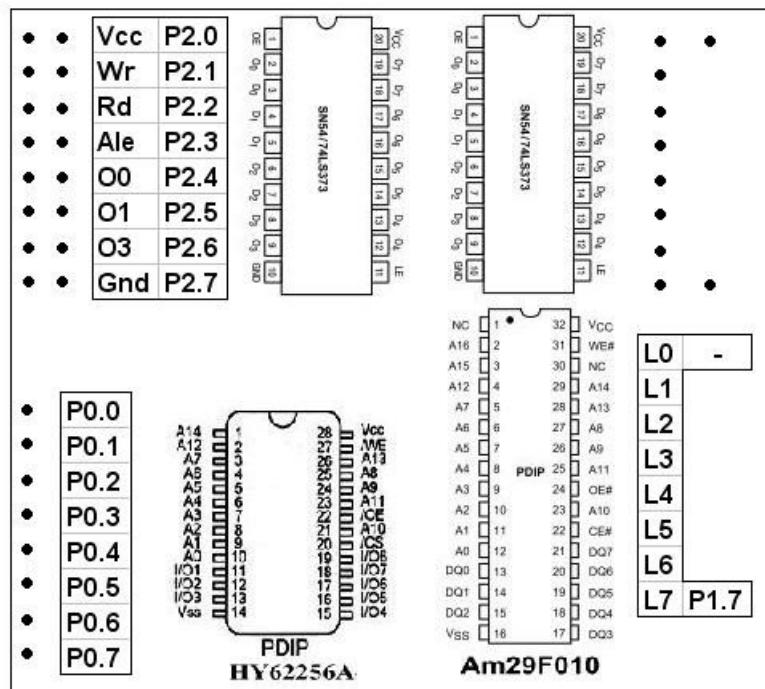


Figura 40: Diagrama da placa de memórias



Figura 41: Foto da placa das memórias

8.2.5 PLACA DA MATRIZ DE LEDS E DO BOTÃO

A placa da Matriz de LEDs e do botão, que na verdade foram 4 placas iguais, consiste basicamente da Matriz de LEDs (HS-2088AS) que é encaixada em 2 barras de pinos fêmea, um CI de Latch das Colunas (74LS373), 8 resistores de 100 Ohms ligados na saída do Latch para limitarem a corrente de entrada na matriz de LEDs já que ela não possui resistência interna, somente a resistência interna dos diodos dela, e um botão simples. A disposição dos CIs, conectores e dos botões na placa podem ser vistos na figura 42 e a foto do circuito soldado pode ser visto na figura 43.

Na Figura 42, canto superior direito os pinos de L0 a L7 representam a saída do Latch seletor de linha, Vcc e Gnd representam a alimentação, O[4-7] representa a saída do 74LS138 que seleciona o Latch da matriz (lembrando que O4 representa a matriz 1, O5 a matriz 2, O6 a matriz 3, O7 a matriz 4) e o pino P1.[0-3] representa a saída do botão da placa, ativa em nível lógico 0 (lembrando que P1.0 representa o botão da primeira matriz, P1.1 representa o botão da segunda matriz, P1.2 representa o botão da terceira matriz, P1.3 representa o botão da quarta matriz).

Essa placa era relativamente simples se comparada com a placa das memórias, os maiores problemas encontrados foram a disposição dos pinos da Matriz de LEDs que aparecia de maneira errônea no Datasheet, ou mais provavelmente nosso modelo era genérico, e com isso foi necessário testar pino a pino da matriz, realizar uma convenção dos pinos e de posição e descrever a posição de cada pino (como pode ser visto na Tabela 8). Como a disposição dos pinos de saída da matriz não são “amigáveis” o trabalho envolvido tinha de ser mais minucioso. Uma das coisas que ajudou muito durante as soldagens foi ter colocado etiquetas nas placas (iguais as etiquetas de preço encontradas em supermercados) para anotar algumas informações importantes como disposição dos pinos e dos conectores e com isso diminuir o risco de soldar pinos em posições erradas.

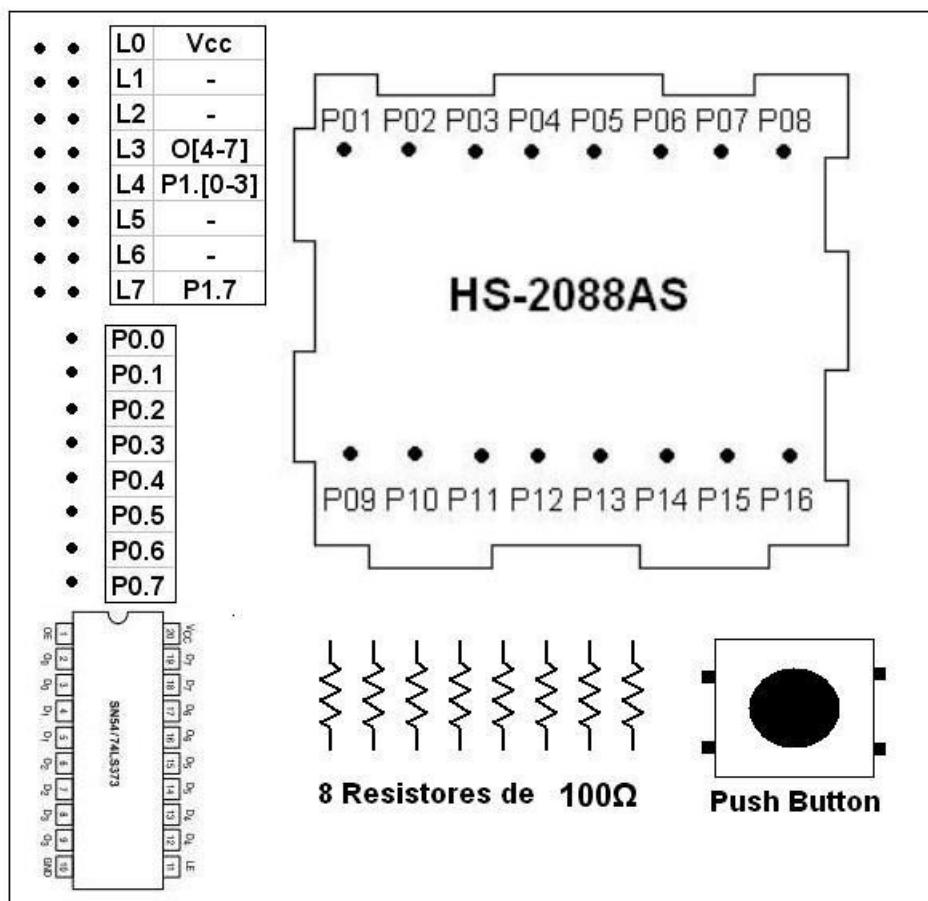


Figura 42: Diagrama da placa da matriz

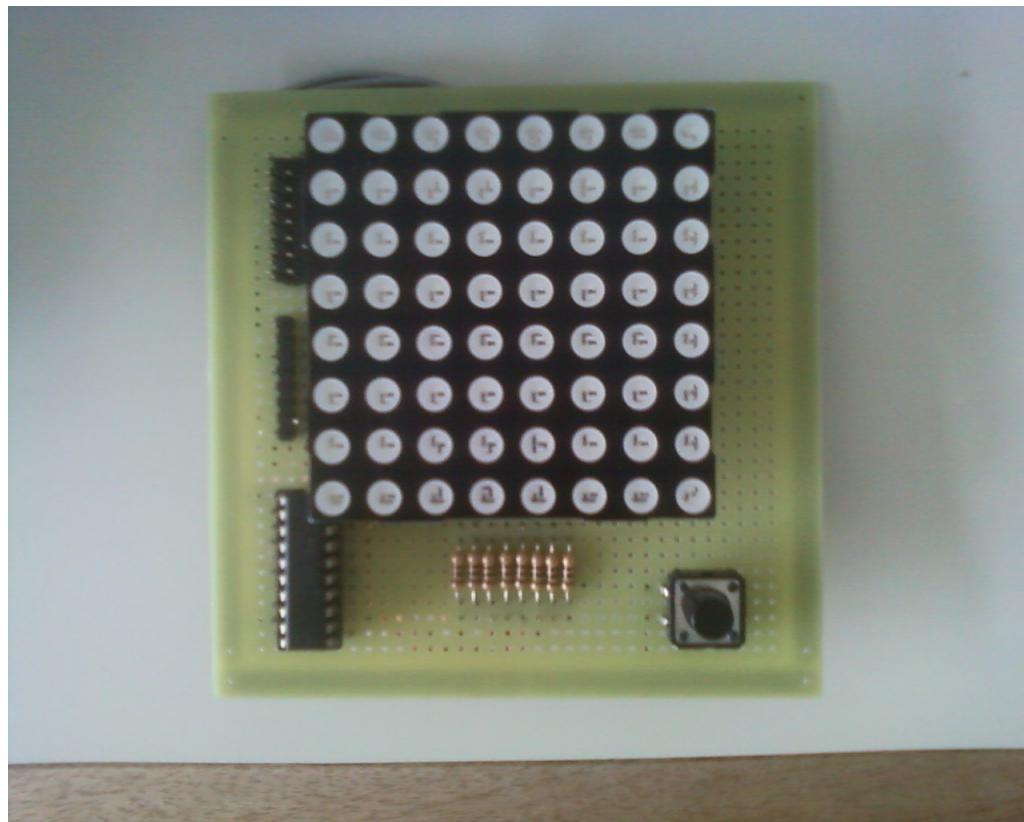


Figura 43: Foto da placa da matriz

8.3 SISTEMA INTEGRADO

Depois de cada parte do circuito estar montado, foi testado a integração dele. Começando apenas com um simples programa que fazia os LEDs da matriz acenderem e depois juntando o sistema de áudio e adicionando novas matrizes.

O maior problema da integração foi ligar a parte da estação base com o sistema embarcado. Foram necessários vários testes de comunicação e diversos problemas foram encontrados para que o software respondesse corretamente a comunicação.

Depois de superado este problema conseguimos atingir nossas o que foi proposto, com o sistema da estação base conseguimos alterar os ícones e sons da prancha.

A figura 44 apresenta uma imagem do sistema com as quatro matrizes acesas. Percebe-se que as quatro matrizes apresentam o mesmo brilho, sem que uma fica mais fraca que a outra, este era um dos problemas que poderiam ocorrer.

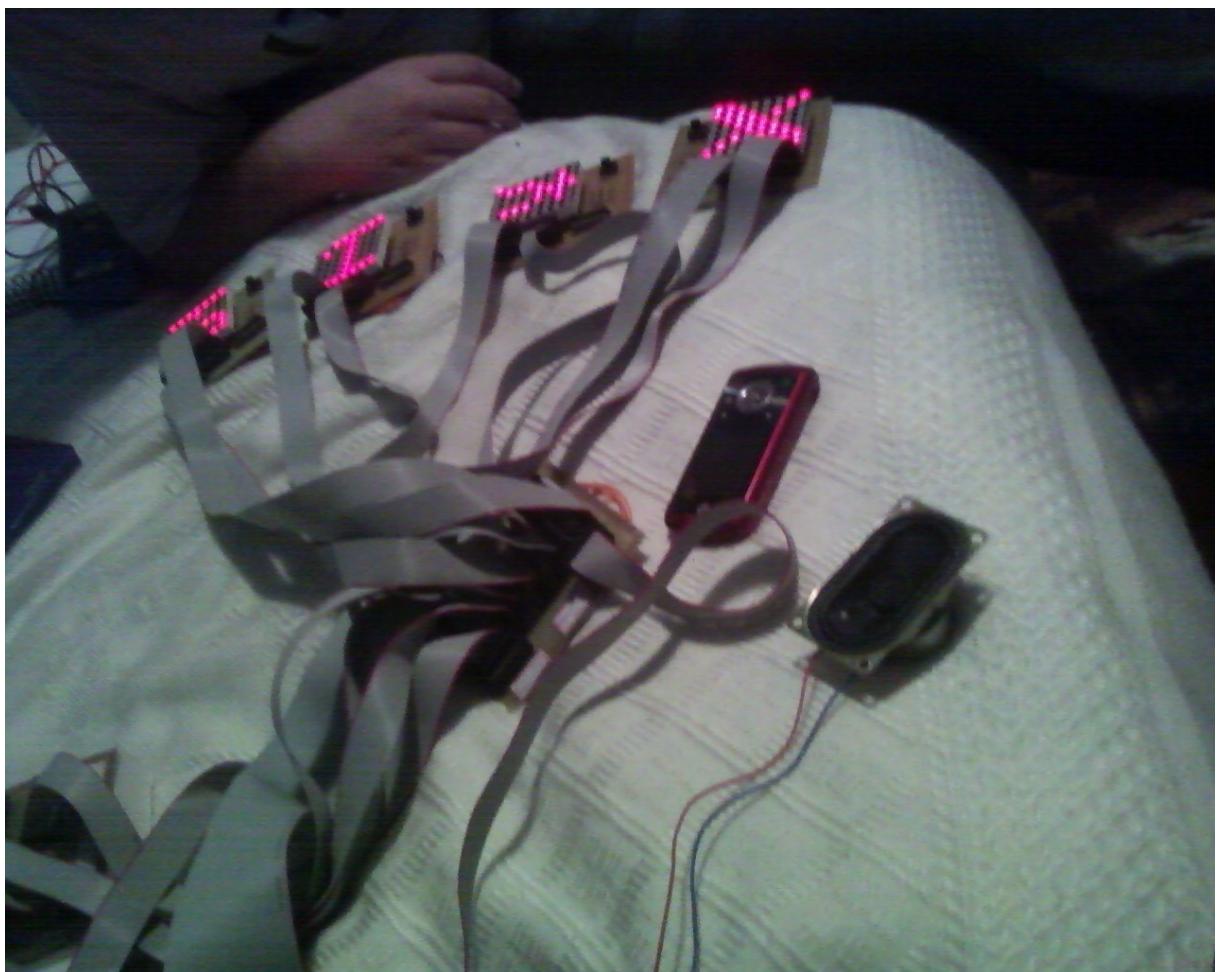


Figura 44: Sistema integrado - quatro matrizes

8.4 RISCOS PLANEJADOS OCORRIDOS

Dos principais riscos que colocamos em enfoque no planejamento alguns acabaram ocorrendo. Neste tópico destacamos alguns deles.

Devido a nossa experiência como projetos e criação de cronogramas, tivemos problemas no planejamento como não analisar direito o período de provas e prováveis trabalhos que ocupariam nosso tempo de trabalho em oficinas. Por grande parte do projeto estávamos sempre alguns dias atrasados e só na reta final que conseguimos nos aproximar do que propusemos.

No desenvolvimento do hardware, tivemos problemas de queima de alguns componentes devido a inversão de polaridade, mas para isto nos prevenimos comprando componentes extras conforme especificamos como forma de mitigação,

Os membros da equipe apresentaram inexperiência com algumas tecnologias que utilizamos, tendo que gastar um tempo com estudo do CI, como por exemplo a forma de apagar setores da memória flash do 29F010.

A falta de experiência em utilizar matrizes de LEDs associado a documentação errônea no datasheet foi um obstáculo que gastou um tempo para transformos. Tivemos que mapear todos os pinos de acordo com o que ele realmente é.

8.5 CUSTO DO PROJETO

Na tabela 12 está representada a estimativa de custos para o nosso trabalho. As estimativas de custo do projeto são baseadas nos preços aproximados de componentes encontrados no mercado. Considerou-se um preço de R\$: 35,00 por hora de trabalho, num total de 400 horas de trabalho (80 por membro da equipe), e adicionou-se um custo relativo ao trabalho de recuperação de cronograma atrasado, bem como para a reposição de componentes e peças que venham a estragar.

Tarefa	Preço/custo estimado
Criação do software de interface	R\$: 3220,00 (92 horas)
Planejamento	R\$: 1330,00 (38 horas)
Implementação	R\$: 1750,00 (50 horas)
Testes	R\$: 140,00 (4 horas)
Montagem do hardware	R\$: 4680,00 (124 horas)
Planejamento	R\$: 1330,00 (38 horas)
Placa do microcontrolador	R\$: 60,00
Memórias não-voláteis	R\$: 50,00
Montagem da placa	R\$: 280,00 (8 horas)
Mini alto-falante	R\$: 50,00
Componentes do amplificador	R\$: 10,00
Montagem do amplificador	R\$: 350,00 (10 horas)
Botões	R\$: 50,00
Montagem dos botões	R\$: 1050,00 (30 horas)
Montagem do sistema de controle dos botões	R\$: 490,00 (14 horas)
Material para a prancha	R\$: 40,00
Montagem da prancha	R\$: 350,00 (10 horas)
Reposição de peças e componentes	R\$: 80,00
Montagem do circuito de bateria fraca	R\$: 350,00 (10 horas)
Testes	R\$: 140,00 (4 horas)
Sistema de comunicação	R\$: 2790,00 (78 horas)
Criação do protocolo	R\$: 910,00 (26 horas)
Implementação no firmware	R\$: 350,00 (10 horas)
Implementação da biblioteca no computador	R\$: 630,00 (18 horas)
Cabo serial	R\$: 15,00
Adaptador USB-serial	R\$: 45,00
Integração com o software de interface	R\$: 700,00 (20 horas)
Testes	R\$: 140.00 (4 horas)
Criação do firmware	R\$: 1680,00 (48 horas)

Planejamento	R\$: 910,00 (26 horas)
Implementação	R\$: 630,00 (18 horas)
Testes	R\$: 140,00 (4 horas)
Testes gerais	R\$: 350,00 (10 horas)
Recuperação de cronograma atrasado	R\$: 1680,00 (48 horas)
TOTAL	R\$: 14400 (400 horas)

Tabela 12: Estimativa de custos

Na tabela 13 estão todos os gastos com hardware que tivemos no decorrer do projeto. Incluindo gastos com reposição de peças.

TABELA DE GASTOS				
Data	Qde.	Produto	Preço	
30/09/10	1	Memoria Flash 29F010	R\$ 19,90	
14/10/10	1	Cabo conversor USB/Serial	R\$ 49,50	
14/10/10	4	Matriz de LEDS 8x8	R\$ 47,00	
15/10/10	3	Par de pilhas AA para teste	R\$ 2,40	
15/10/10	1	Matriz de LEDS 8x8 reposição	R\$ 11,75	
28/10/10	1	Componentes microcontrolador	R\$ 13,23	
28/10/10	2	Microcontroladores AT89S52	R\$ 11,80	
04/11/10	1	Speaker, buffer e TDA	R\$ 12,00	
05/11/10	1	Placa Microcontrolador + comp.	R\$ 50,00	
08/11/10	1	Componentes: Cis e componentes placas	R\$ 67,00	
09/11/10	1	Componentes	R\$ 5,15	
10/11/10	1	Componentes placa de conexao	R\$ 4,72	
11/11/10	1	Componentes Audio	R\$ 26,00	
11/11/10	1	Componentes + matriz extra	R\$ 17,00	
11/11/10	1	Componentes (flat)	R\$ 12,32	
16/11/10	1	Microcontrolador 8052	R\$ 5,00	
16/11/10	1	Componentes Audio	R\$ 11,16	
19/11/10	4	Placas Universais (botao)	R\$ 20,00	
19/11/10	1	Componentes: réplica placa da matriz	R\$ 35,95	
25/11/10	1	Componentes	R\$ 3,30	
25/11/10	1	Componentes	R\$ 12,60	
25/11/10	1	Componentes (cabo flat e resistor)	R\$ 5,50	
25/11/10	1	Fonte	R\$ 22,00	
TOTAL				R\$ 465,28

Tabela 13: Tabela de gastos

Em comparação com a previsão que tínhamos gastamos R\$ 65,28, ou seja, 16,32%. a mais que o planejado para o hardware.

9 CONCLUSÃO

O projeto, apesar de ainda possuir algumas poucas falhas, cumpre essencialmente ao que se propôs. O produto dele, a prancha, recebe informações com imagens e sons de uma estação base com um software organizado e simples de usar, com velocidade e taxa de erros satisfatórias.

É fato que um elemento não implementado, mesmo estando nos planos dos idealizadores, foi o medidor de bateria. Contudo, é uma parte bastante pequena de um projeto que engloba diversos outros conceitos como conversão digital/analógica, microprocessamento e transmissão de dados.

Na realização do trabalho, mais do que conhecimento teórico, foi possível trabalhar diversos elementos presentes no cotidiano de um engenheiro como os prazos a seguir, os incansáveis e muito necessários testes, o estudo de tecnologias, a comunicação e o trabalho em equipe. Neste contexto foi necessário o desenvolvimento de aspectos necessários para uma equipe como a cobrança (contudo, aliada a paciência e a compreensão) e a visão de responsabilidade de cada um sobre o sucesso ou o fracasso do grupo inteiro.

Por conta da análise destes fatores, a visão que se tem na equipe é de que o trabalho foi um sucesso, pois conseguiu convergir conhecimentos, experiências, idéias em um protótipo que talvez um dia, já que não há limites aos sonhos, possa se tornar um grande produto no mercado.

REFERÊNCIAS BIBLIOGRÁFICAS

- [1] BASTOS, Laudelino Cordeiro e MATIAS, Denis Hattori, Prancha de Comunicação com Predição de Símbolos, Pontifícia Universidade Católica do Paraná, Curitiba. Disponível em: <<http://www.sbis.org.br/cbis9/arquivos/443.pdf>> Acessado em: 19/09/2010.
- [2] Patente do jogo eletrônico SIMON. Disponível em: <<http://v3.espacenet.com/publicationDetails/biblio?CC=US&NR=4207087&KC=&FT=E>>. Acessado em: 19/09/2010.
- [3] Datasheet de ATmega8. Disponível em: <www.atmel.com>. Acessado em: 29/10/2010.
- [4] Datasheet do PIC16. Disponível em: <www.labtools.com>. Acessado em: 29/10/2010.
- [5] NICOLOSI, Denys E. C. Microcontrolador 8051 Detalhado. 5ª Edição. Ano 2000 . Editora Érica.
- [6] Datasheet de 27c512. 512 Kbit (64Kb x 8) UV EPROM and OTP EPROM. Disponível em: <http://www.datasheetcatalog.org/datasheets/150/223686_DS.pdf>. Acessado em: 29/10/2010.
- [7] Datasheet do 24LC512 . 512K CMOS Serial EEPROM. Microship. Disponível em: <<http://www.datasheetcatalog.org/datasheet2/7/0t5ls8qi8zac4o9goy4fu7oykg3y.pdf>>. Acessado em: 29/10/2010.
- [8] Datasheet 29F010. Memória Flash 128 k x 8 -bit. AMD. Disponível em: <http://www.datasheetcatalog.org/datasheets/400/135239_DS.pdf>. Acessado em: 29/11/2010.
- [9] Datasheet 74LS138. Decoders/Demultiplexers. National Semiconductor. Disponível em: <http://www.datasheetcatalog.org/datasheets/166/375388_DS.pdf>. Acessado em: 29/11/2010.
- [10] Datasheet HY62256A-(I). DRAM e SRAM Memory. Hyundai. Disponível em: <<http://pdf1.alldatasheet.com/datasheet-pdf/view/65363/HYNIX/HY62256A.html>>. Acessado em: 29/11/2010.
- [11] Datasheet TDA7052A/AT. 1 W mono audio amplifier with DC volume control. Disponível em: <http://www.datasheetcatalog.org/datasheet/philips/TDA7052_CNV_2.pdf>.

Acessado em: 30/11/2010.

[12] Datasheet 74LS373N. Octal transparent latch with 3-state outputs; octal d-type flip-flop with 3-state output .Motorola. Disponível em:
<<http://www.datasheetcatalog.org/datasheet/motorola/SN74LS373N.pdf>>. Acessado em: 29/11/2010.

[13] Datasheet CA3140. 4.5 Mhz, BiMOS Operational Amplifier with MOSFET Input/Bipolar Output .Intersil. Disponível em:
<<http://www.datasheetcatalog.org/datasheet/intersil/fn957.pdf>>. Acessado em: 30/11/2010.

[14] OpenProj: open source desktop project management. Serena Software.
Disponível em: <<http://openproj.org/>>. Acessado em: 01/12/2010.

[15] Cx51 User's Guide. Keil, tool by ARM. Disponível em:
<<http://www.keil.com/c51/c51.asp>>. Acessado em: 29/11/2010.

[16] RxTx: The Prescription for Transmission. Disponível em:
<<http://users.frii.com/jarvi/rxtx/>>. Acessado em: 29/11/2010.

[17] Java Platform, Standard Edition 6 API Specification. Oracle Inc. Disponível em:
<<http://download.oracle.com/javase/6/docs/api/>>. Acessado em: 29/11/2010.

[18] LOPES, Heitor Silvério; FARIA, Rubens Alexandre de Faria. Manual da placa P51N V2.0. Disponível em:
<http://bioserver.cpgei.ct.utfpr.edu.br/disciplinas/sistmicro/docs/ManualP51NV2_agosto_2009.pdf>. Acessado em: 29/11/2010.