



UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
ENG04029 – PROJETO DE DIPLOMAÇÃO



Ferramenta para criação de *Transport Stream* com múltiplos programas para o padrão brasileiro de televisão digital

Autor: André Luís Bacarin Gobo

Orientador: Tiago Roberto Balen

Sumário

Agradecimentos	4
Resumo	5
Abstract	6
Lista de Figuras	7
Lista de Tabelas	9
Lista de Abreviaturas e Siglas	10
1. Introdução	12
2. Revisão Bibliográfica e Fundamentos Teóricos	13
2.1. Norma Internacional ISO/IEC 13818-1	13
2.2. Sistema Brasileiro de televisão Digital (SBTVD)	15
2.3. <i>Software</i> de multimídia FFmpeg	18
2.3.1. Introdução ao FFmpeg	18
2.3.2. Compatibilização do FFmpeg com o SBTVD	20
3. Implementação do Projeto	22
3.1. TS de múltiplos programas perfil 2	22
3.2. GUI para geração de TS	23
4. Testes e Resultados	32
4.1. Testes e resultados do FFmpeg	32
4.1.1. Análise feita com o FFmpeg	32
4.1.2. Análise do <i>software</i> SBTVD_parser	34
4.1.3. Análise <i>software</i> VLC Media Player	36
4.1.4. Análise de transmissão do TS	38
4.2. Testes e resultados do GUI_FFmpeg	39
5. Conclusões e Sugestões para Trabalhos Futuros	41
6. Referências	42
Apêndice	44
Help da GUI_FFmpeg	44
Anexo	53
Países que adotaram o ISDB-TB	53

Agradecimentos

Aos professores da Universidade Federal do Rio Grande do Sul – UFRGS, que de alguma forma contribuíram para a minha formação.

Em especial ao professor Dr. Tiago Roberto Balen, que esteve disponível para orientar-me e dar dicas sobre profissões da área, e ao professor Altamiro Amadeu Susin, por auxiliar-me e orientar-me durante o tempo que passei no LaPSI.

Ao Eng. Lucas Endres, pelas dicas e materiais passados, e à equipe do LaPSI, pelo seu compromisso com o desenvolvimento da televisão e por apoiar-me nos testes realizados.

Aos professores membros da banca, Eng. Luiz Sperotto Teixeira e Dr. André Borin Soares, por disponibilizarem seu tempo.

À minha família, por dar-me sustento em toda a minha vida, apoiando-me em cada dificuldade que passei. Em especial a minha irmã Ana Cristina Bacarin Gobo, por ser um exemplo para mim e incentivar-me a ser sempre melhor, aos meus pais Antônio João Gobo e Anagilda Bacarin Gobo, pelos princípios de vida e educação que me transmitiram, e ao meu avô João Atílio Rosa Gobo, por estimular a minha criatividade e ensinar-me o amor pelo trabalho.

Aos meus colegas e amigos, pela colaboração e compreensão no período de desenvolvimento deste projeto e por tornarem o tempo que passei em Porto Alegre uma experiência incrível.

Resumo

A televisão é um dos meios de comunicação de grande importância atualmente no Brasil. Está em constante evolução, passando a ter melhorias significativas a cada década. No momento, está passando por um processo de digitalização que, no Brasil, segue o padrão técnico denominado de Sistema Brasileiro de Televisão Digital (SBTVD). Este sistema é baseado no sistema japonês de televisão digital e é adotado por outros países além do Brasil. Uma das diferenças mais relevantes entre os dois sistemas é a utilização de múltiplos programas, obrigatórios para o SBTVD. Esta característica possibilita à emissora de televisão transmitir mais de um programa utilizando o mesmo *Transport Stream* (TS) e somente uma faixa de frequência. Com isso, necessita-se gerar TS's com esta característica para testar o *Set-top box* compatível com o SBTVD desenvolvido. Atualmente existe uma versão do *software Fast Forward MPEG* (FFmpeg) que suporta a geração de TS's compatíveis com o SBTVD utilizando múltiplos programas. Porém, esta versão só possibilita dois programas no mesmo TS, sendo, portanto, um dos objetivos deste trabalho a implementação da possibilidade de gerar um TS com mais de dois programas. Além disso, por ser um *software* com instrução dada pela linha de comando, foi implementado uma Interface Gráfica do Usuário (GUI) para o fácil e rápido manuseio do *software*. Para confirmar a implementação de múltiplos programas os TS's criados foram testados em diferentes ferramentas: *software* SBTVD_parser, *software* VLC Media Player, o próprio *software* Ffmpeg e transmissão/recepção do TS pelo ar. Os primeiros demonstraram a efetivação dos múltiplos programas e o último apresentou problemas deixados para futuros projetos. Além destes, também foram feitos testes com a GUI, os quais atestaram a sua necessidade. Deixando assim, os testes com TS's com múltiplos programas mais robustos e rápidos.

Abstract

Television is one media of great importance currently in Brazil. It's constantly evolving, going to have significant improvements every decade. At the time, it is going through a digitalization process that, in Brazil, follows the technical standard known as Sistema Brasileiro de Televisão Digital (SBTVD). This system is based on the Japanese digital television system and is adopted by other countries besides Brazil. One of the most relevant differences between the two systems is the use of multiple programs, mandatory for SBTVD. This feature allows the TV station to transmit more than one program using the same Transport Stream (TS) and only one frequency band. Thus, to test the Set-top box compatible with the SBTVD developed, it is necessary to generate TS 's with this feature. Currently there is a version of Fast Forward MPEG (FFmpeg) software which supports the generation of a TS compatible with SBTVD using multiple programs. However, this version only allows two programs in the same TS, and therefore one of the objectives of this project is the implementation of the possibility of generating TS with more than two programs. Moreover, It was implemented a Graphical User Interface (GUI) for easy and fast handling of the software. To confirm the implementation of multiple programs, the TS's created were tested on different tools: SBTVD_parser software, VLC Media Player software, the FFmpeg software and the transmission and reception of the TS through the air. The first demonstrated the effectiveness of multiple programs and the latter presented problems, left for future projects. In addition, tests were also made with the GUI, which attested to its need. With this, the tests with TS's with multiple programs become more robust and fast.

Lista de Figuras

Figura 1 – Visão geral simplificada da norma ISO/IEC 13818-1 (Fonte: ISO/IEC 13818-1). 13	13
Figura 2 – Diagrama de sintaxe do <i>Transport Stream</i> (valores em <i>bits</i>) (Fonte: ISO/IEC 13818-1)..... 14	14
Figura 3 – Sistemas de Televisão Digital na América Latina (Fonte: Fórum SBTVD)..... 16	16
Figura 4 – Funcionamento do FFmpeg com saída para Icecast (Fonte: Desenvolvimento e adaptação de tecnologias livres para solução de <i>streaming</i> HLS na UFRGS)..... 20	20
Figura 5 – Design da GUI_FFmpeg..... 24	24
Figura 6 – Botão Executar..... 25	25
Figura 7 – Botão Ajuda. 25	25
Figura 8 – Caixa de diálogo “Comando”..... 25	25
Figura 9 – Abas separando comandos..... 25	25
Figura 10 – Aba de entradas..... 26	26
Figura 11 – Caixa de diálogo para selecionar uma entrada. 26	26
Figura 12 – Escolha dos <i>Streams</i> 27	27
Figura 13 – Aba de saída..... 27	27
Figura 14 – Caixa de diálogo para selecionar a pasta de destino da saída. 28	28
Figura 15 – Aba Múltiplos Programas. 29	29
Figura 16 – Aba Múltiplos Programas com seleções (As 5 entradas estão definidas)..... 29	29
Figura 17 – Erro quando não há dois <i>streams</i> selecionados na entrada. 29	29
Figura 18 – Determinar duração do TS..... 29	29
Figura 19 – Selecionar utilização de <i>codecs</i> 30	30
Figura 20 – Configuração do <i>Muxrate</i> 30	30
Figura 21 – Aba avançado..... 31	31
Figura 22 – Comando executado para criar o TS de exemplo..... 32	32
Figura 23 – Análise feita pelo FFmpeg do TS criado..... 33	33
Figura 24 – Comando utilizado para fazer a análise do TS criado pelo FFmpeg. 33	33
Figura 25 – Análise do Parser_SBTVD (v0.34) para o TS criado. 34	34
Figura 26 - Análise do Parser_SBTVD (v0.34) para o TS criado, tabela PAT..... 35	35
Figura 27 – Análise do Parser_SBTVD (v0.34) para o TS criado, tabela PMT. a) programa 1; b) programa 2 e; c) programa 3..... 35	35
Figura 28 – Teste do TS criado com o VLC <i>Media Player</i> , programa 1..... 37	37
Figura 29 – Teste do TS criado com o VLC <i>Media Player</i> , programa 2..... 37	37
Figura 30 – Teste do TS criado com o VLC <i>Media Player</i> , programa 3..... 37	37
Figura 31 – Processo utilizado para testar os TS’s utilizando transmissão pelo ar. 38	38

Figura 32 – Design da GUI_FFmpeg.....	45
Figura 33 – Botão Rodar.	46
Figura 34 – Botão Ajuda.	46
Figura 35 – Caixa de diálogo “Comando”.....	46
Figura 36 – Abas separando comandos.....	46
Figura 37 – Aba de entradas.....	47
Figura 38 – Caixa de diálogo para selecionar uma entrada.	47
Figura 39 – Escolha dos <i>Streams</i>	48
Figura 40 – Aba de saída.....	48
Figura 41 – Caixa de diálogo para selecionar a pasta de destino da saída.	49
Figura 42 – Aba Múltiplos Programas.	50
Figura 43 – Aba Múltiplos Programas com seleções (As 5 entradas estão definidas).....	50
Figura 44 – Erro quando não há dois <i>streams</i> selecionados na entrada.	50
Figura 45 – Determinar duração.....	50
Figura 46 – Selecionar utilização de codecs.	51
Figura 47 – Configuração do <i>Muxrate</i>	51
Figura 48 – Aba avançado.....	52

Lista de Tabelas

Tabela 1 – Tabelas e níveis de transmissão (Fonte: ABNT NBR 15603-2).....	16
Tabela 2 – Informações do TS criado obtidas com o FFmpeg.....	33
Tabela 3 – Informações do TS criado obtidas com o SBTVD_parser.....	36
Tabela 4 – Testes de tempos GUI_FFmpeg.....	39

Lista de Abreviaturas e Siglas

AAC	Advanced Audio Coding
API	Application Programming Interface
CAT	Conditional Access Table
EIT	Event Information Table
FFmpeg	Fast Forward MPEG
FPGA	Field Programmable Gate Array
GNU	Gnu's Not Unix
GUI	Interface Gráfica do Usuário
IDE	Integrated Development Environment
IEC	International Electrotechnical Commission
IBGE	Instituto Brasileiro de Geografia e Estatística
ISDB-T	Integrated Services Digital Broadcasting Terrestrial
ISDB-TB	Integrated Services Digital Broadcasting Terrestrial Brazil
ISO	International Organization for Standardization
LATM	Low Overhead Audio Transport Multiplex
MPEG	Moving Picture Experts Group
NIT	Network Information Table
PAT	Program Association Table
PES	Packetized Elementary Stream
PID	Packet ID
PMT	Program Map Table
PS	Program Stream
PSI	Program Specific Information
SBTVD	Sistema Brasileiro de Televisão Digital
SDT	Service Description Table

TOT	Time Offset Table
TS	Transport Stream
VLC Media Player	VideoLAN Client Media Player

1. Introdução

A tecnologia utilizada nos aparelhos televisivos vem passando por um processo de digitalização no Brasil. Segundo pesquisas realizadas pelo Instituto Brasileiro de Geografia e Estatística (IBGE) (IBGE, 2013), em 2013, 97,2% dos domicílios pesquisados possuíam televisão. Provavelmente, muitos destes domicílios estão adaptando-se para receber o sinal digital.

Uma das possibilidades de adaptar os aparelhos televisivos a receber um sinal digital é utilizando um *set-top box* compatível com o Sistema Brasileiro de Televisão Digital (SBTVD). Esta e outras tecnologias de arquiteturas de *hardware* para processamento de imagens vêm sendo desenvolvidas no Laboratório de Processamentos de Sinais e Imagens (LaPSI).

Para testar este *set-top box* desenvolvido no LaPSI, é necessário gerar arquivos do tipo *Transport Stream's* (TS's) de testes compatíveis. Uma das ferramentas que o SBTVD contempla é a utilização de múltiplos programas, por isso, é necessário gerar TS's de testes com esta característica. O *software* Fast Forward MPEG (FFmpeg) (FFmpeg - <https://ffmpeg.org/about.html>, acesso em 31/05/2016) foi escolhido para servir como base desta geração, porém a versão utilizada gera no máximo dois programas por TS. Sendo assim, será necessário aprimorar o *software* para gerar mais programas, pois, segundo as normas brasileiras (ABNT NBR 15603-1, 2007; ABNT NBR 15603-2, 2007; ABNT NBR 15603-3, 2007), o decodificador deve estar apto a receber um maior número de programas.

O FFmpeg é um *software* livre com código aberto que tem compatibilidade com os mais variados tipos de multimídias. É executado por linha de comando, tornando-se assim fácil de ser utilizado em conjunto com outros tipos de programas. Porém, esta interface de linha de comando tem um tempo de configuração grande se utilizada sem o auxílio de programas.

Identificadas as necessidades descritas acima, foi definido como objetivo deste projeto: aprimorar as possibilidades do FFmpeg, possibilitando a geração TS's com mais programas e; elaborar uma forma mais fácil e rápida de gerar TS's com o FFmpeg, através de uma Guia de Interface para o Usuário (GUI).

2. Revisão Bibliográfica e Fundamentos Teóricos

Neste capítulo serão revisados os principais conceitos utilizados para implementar o projeto. Os temas abordados serão “Norma Internacional ISO/IEC 13818-1”, “Sistema Brasileiro de televisão Digital (SBTVD)” e “Software de multimídia FFmpeg”.

2.1. Norma Internacional ISO/IEC 13818-1

A norma internacional *International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) 13818-1* especifica, segundo sua descrição, a combinação de um ou mais *stream* elementares de vídeo e áudio, como também outros tipos de dados, em um ou múltiplos *streams*. Estes *streams* resultantes servem para transmitir ou armazenar a multimídia e são nomeados de *Transport Stream (TS)* ou *Program Stream (PS)*, dependendo de como foram montados. De modo geral, no procedimento de transformação em TS ou PS, os *streams* elementares passam por um processo de codificação (*encoder*), depois são empacotados para produzir *Packetized Elementary Stream (PES)*. Por fim, os PES são multiplexados e tornam-se TS ou PS. A Figura 1 ilustra o processo descrito acima. Toda a descrição desta seção é referenciada na norma ISO/IEC 13818-1 (ISO/IEC 13818-1, 2000).

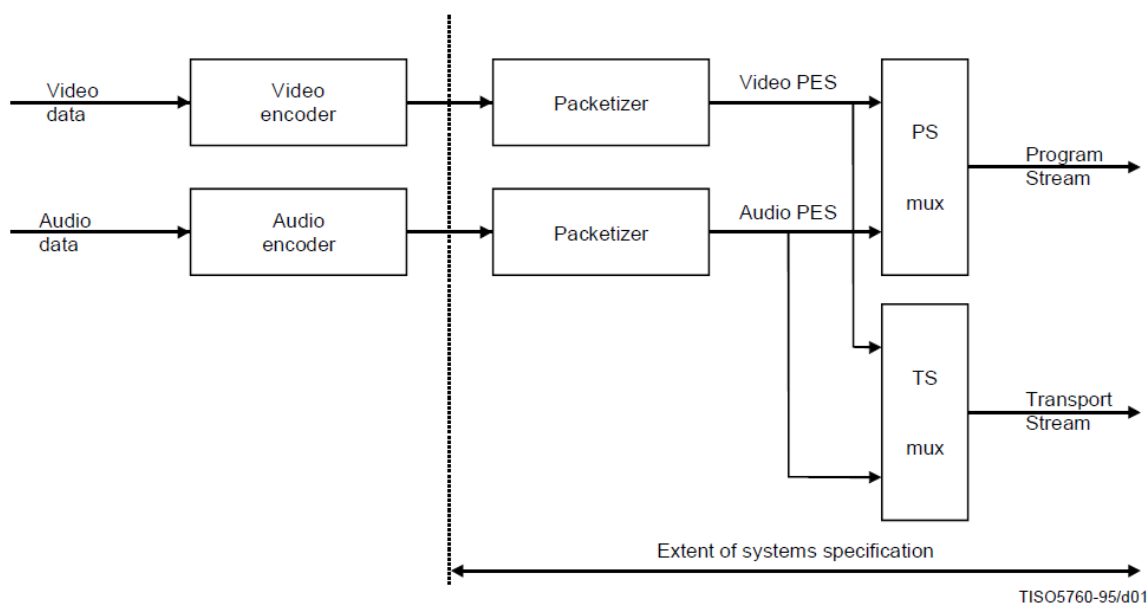


Figura 1 – Visão geral simplificada da norma ISO/IEC 13818-1 (Fonte: ISO/IEC 13818-1).

O PS é utilizado para comunicação ou armazenamento de um programa de dados codificados em um ambiente onde erros ocorrem raramente, por esta razão, é dividido em um ou mais PES. Por outro lado, o TS é utilizado para comunicação ou armazenamento de um ou mais programas de dados em um ambiente onde erros significantivos podem ocorrer. Para corrigir erros, os PES do TS são divididos em diferentes pacotes de tamanhos fixos em 188 *bytes* de dados. Portanto, como a transmissão de arquivos multimídia pelo ar é um ambiente onde erros significantivos podem ocorrer, o TS é mais indicado nas transmissões de sinais de televisão. Por esta

razão, serão apresentadas somente explicações referentes ao conteúdo de um TS. Semelhanças e igualdades com o PS não foram verificadas.

Os pacotes de dados de um TS contêm sempre 188 bytes. Estes bytes são compostos por um cabeçalho e um *payload*, como ilustra a Figura 2.

O *payload* pode tanto conter PES de uma e somente um *stream*, quanto dados de *Program Specific Information* (PSI). Este último é elucidado no decorrer desta seção. A informação de qual dado está sendo transmitido no *payload* é especificado pelo *payload_unit_start_indicator*, o qual faz parte do cabeçalho, cujo local pode ser observar na Figura 2.

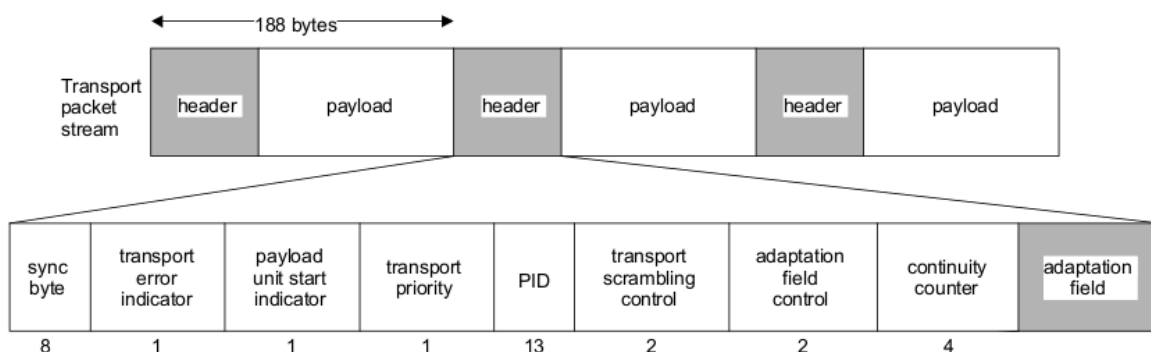


Figura 2 – Diagrama de sintaxe do *Transport Stream* (valores em bits) (Fonte: ISO/IEC 13818-1).

Cada pacote de TS contém o *Packet Identifier* (PID), que por sua vez contém as tabelas de *Program Specific Information* (PSI), cuja função é identificar o conteúdo dos dados contidos no pacote de TS do qual faz parte. Há quatro diferentes tabelas PSI e cada uma tem seu nome e função esclarecidos abaixo:

- *Program Association Table* (PAT): faz a correspondência entre um *program_number* e o valor do PID do pacote TS. O *program_number* é o rótulo numérico associado a um programa. Esta tabela é obrigatória no TS compatível com a norma em questão;
- *Program Map Table* (PMT): fornece o mapeamento entre os números do programa e os elementos do programa que os abrangem. Esta tabela é obrigatória no TS compatível com a norma em questão;
- *Conditional Access Table* (CAT): faz a associação entre um ou mais sistemas de condição de acesso, seus *streams* com níveis de autorização e qualquer outro parâmetro especial. Esta tabela é obrigatória no TS compatível com a norma em questão;
- *Network Information Table* (NIT): esta tabela é privada e não é descrita pela ISO/IEC 13818-1. Ou seja, esta tabela é opcional para TS compatível com a norma em questão.

As informações do PES e as tabelas PAT, PMT e CAT juntos contêm as informações necessárias e suficientes para demultiplexar e exibir um programa, sendo esse transmitido por um TS compatível com a norma em questão.

2.2. Sistema Brasileiro de televisão Digital (SBTVD)

O decreto presidencial número 4901 (Decreto Nº 4.901, 2013), de 2003, institui o Sistema Brasileiro de Televisão Digital (SBTVD). Segundo Art. 1º do mesmo decreto o SBTVD tem a finalidade de, entre outras, estimular a pesquisa e o desenvolvimento, além de propiciar a expansão de tecnologias brasileiras e da indústria nacional relacionada à tecnologia de informação e comunicação. Além disto, instaurou um Comitê de Desenvolvimento vinculado à Presidência da República.

Dispostos os objetivos do SBTVD e criado o Comitê de Desenvolvimento, criou-se, em 2006, por um outro decreto presidencial (número 5820) (DECRETO Nº 5.820, 2006), um Fórum do SBTVD (FÓRUM DO SISTEMA BRASILEIRO DE TV DIGITAL TERRESTRE - <http://forumsbtvd.org.br/>, acesso em 12/06/16) para assessorar o Comitê de Desenvolvimento acerca de políticas e assuntos técnicos referentes à aprovação de inovações tecnológicas, especificações, desenvolvimento e implantação do SBTVD. Este fórum, atualmente, já desenvolveu, em parceria com a Associação Brasileira de Normas Técnicas, 14 normas técnicas que especificam o SBTVD, totalizando cerca de 3.000 páginas. Além de criar o Fórum SBTVD, este decreto definiu que o SBTVD adotou, como base, o padrão de sinais do ISDB-T (*Integrated Services Digital Broadcasting Terrestrial*), conhecido como padrão japonês.

Sendo assim o padrão japonês de televisão digital foi aprimorado, sendo este aprimoramento conhecido internacionalmente de ISDB-TB. Algumas aprimoramentos são bastante relevantes, como, adotar como algoritmo de compressão o MPEG-4 (*Moving Picture Experts Group 4*) por ter qualidade de imagem e som melhores e um preço similar e adotar outro *software* para fazer a interação entre o sistema operacional e as aplicações nos *set-top boxes* e nos celulares. Estas mudanças tornaram o padrão japonês incompatível e, com isso, é necessário desenvolver aparelhos compatíveis para comercializá-los no mercado interno de, aproximadamente, 57 milhões de aparelhos.

Influenciou mercados vizinhos a também adotarem o padrão (chamado internacionalmente de ISDB-TB, com o acréscimo do “B” de Brasil), como ilustra a Figura 3. Além de países da América Latina, alguns países africanos também adotaram o padrão. A lista de países que adotaram o padrão encontra-se no Anexo. Por estar influenciando e desenvolvendo o ISDB-T o Brasil é chamado de líder da sua expansão (Andgulo *et al.*, 2012).



Figura 3 – Sistemas de Televisão Digital na América Latina (Fonte: Fórum SBTVD).

Dentre as 14 normas técnicas criadas para reger o ISDB-TB, a norma ABNT NBR 15603 (ABNT NBR 15603-1, 2007; ABNT NBR 15603-2, 2007; ABNT NBR 15603-3, 2007), dividida em três partes, refere-se às tabelas de serviço de informação, conhecidas por tabelas PSI (elucidadas na seção anterior). Nesta norma consta a informação de quais tabelas PSI são obrigatórias para um TS ser compatível com o SBTVD, contendo também seus objetivos e como implementá-las. Na Tabela 1 é possível observar as tabelas obrigatórias ou opcionais.

Tabela 1 – Tabelas e níveis de transmissão (Fonte: ABNT NBR 15603-2).

Tabelas	Nível de obrigatoriedade
PAT	Obrigatória
CAT	Obrigatória
PMT	Obrigatória
NIT (rede atual)	Obrigatória
NIT (outra rede)	Opcional
SDT (feixe atual)	Obrigatória
SDT (outro feixe)	Opcional
BAT	Opcional

EIT (programa presente e futuro do feixe atual)	Obrigatória
EIT (programa presente e futuro de outro feixe)	Opcional
EIT (programa em até 8 dias do feixe atual)	Opcional
EIT (programa após 8 dias do feixe atual)	Opcional
EIT (programa em até 8 dias de outro feixe)	Opcional
EIT (programa após 8 dias de outro feixe)	Opcional
TDT	Opcional
RST	Opcional
ST	Opcional
TOT	Obrigatória
PCAT	Opcional
BIT	Opcional
NBIT (corpo de informação de grupo)	Opcional
NBIT (informação de referência para obtenção da informação de grupo)	Opcional
LDT	Opcional

As tabelas PAT, CAT e PMT são idênticas às tabelas descritas na seção 2.1, ou seja, para um TS ser compatível com o ISDB-TB suas tabelas PAT, CAT e PMT devem obrigatoriamente atender à ISO/IEC 13818-1. As outras tabelas obrigatórias são elucidadas abaixo.

- *Network Information Table* (NIT): informa a organização física do agrupamento de TS existentes em uma mesma rede e as suas características;
- *Service Description Table* (SDT): deve descrever os serviços contidos no TS específico. Abaixo alguns parâmetros desta tabela que estão diretamente relacionados aos objetivos deste projeto:
 - `service_id`: este parâmetro deve obrigatoriamente ser único por estação geradora e conter a identificação do tipo e do número de programas transmitidos. É um campo de 16 bits, formado da seguinte maneira: os 11 bits mais significativos são iguais aos 11 bits menos significativos do campo de identificação da emissora (diferente para cada uma das emissoras); 2 bits para identificar o tipo de programa, se é identificado por “Televisão”, “Dados” ou “1-seg”; 3 bits representando o número do programa transmitido pela emissora;
 - `provider_name`: contém os caracteres do nome do provedor de programa;
 - `service_name`: contém os caracteres do nome do programa.
- *Event Information Table* (EIT): provê informações relativas aos eventos contidos dentro de cada programa e suas ordens cronológicas;
- *Time Offset Table* (TOT): deve conter a informação de horário UTC-3 (Tempo Universal Coordenado-3, horário de Brasília), informação de data e diferença de fuso horário.

2.3. Software de multimídia FFmpeg

2.3.1. Introdução ao FFmpeg

O *Fast Forward MPEG* (FFmpeg) (FFmpeg - <https://ffmpeg.org/about.html>, acesso em 31/05/2016) é um *framework*, comandado por linha de comando, que suporta os mais variados tipos de multimídias existentes atualmente. É utilizado para codificação, decodificação, conversão, análise e exibição destes formatos, além de suportar aquisição e codificação destas multimídias em tempo real.

Já foi utilizado amplamente pela comunidade acadêmica para desenvolver diferentes tipos de soluções, como: *codec* de áudio e vídeo para a plataforma Android (QURESHI *et al.*, 2013), *streaming* HLS (HTTP Live *Streaming*) (Bitzki e Kleemann, 2016) e módulo para inspeção automática de linhas de transmissão por termografia (SCHEEREN, 2011). Ou seja, é um *software* amplamente testado, tornando-se, de certa forma, confiável.

Segundo as considerações legais e de licenciamento do FFmpeg (FFmpeg License and Legal Considerations - <https://ffmpeg.org/legal.html>, acesso em 31/05/2016), o *software* se trata de um *framework* de multimídia licenciado sob a *GNU Lesser General Public License* (LGPL), versão 2.1 (Free Software Foundation, 1999) ou mais recente, e tem várias otimizações opcionais cobertas pela *GNU General Public License* (GPL), versão 2 (Free Software Foundation, 1991) ou mais recente. Ou seja, é licenciado para ser utilizado em programas próprios, porém, as otimizações opcionais só permitem a utilização em programas livres.

Segundo sua descrição (FFmpeg - <https://ffmpeg.org/about.html>, acesso em 31/05/2016), o FFmpeg é capaz de trabalhar com a maioria das multimídias criadas até o momento. Ferramentas disponíveis:

- *Encode*: Transforma multimídias de formatos primitivos em multimídias de formatos comprimidos. Estes formatos comprimidos são aqueles onde as redundâncias da multimídia primitiva são modificadas para transmissão ou armazenamento;
- *Decode*: Processo inverso ao *encode*, porém a multimídia resultante contém as modificações efetuadas pelo *encode*, diferenciando o resultado final em comparação ao formato primitivo da multimídia;
- *Transcode*: Transforma multimídias de formatos comprimidos em outro formato comprimido;
- *Mux*: De modo simples refere-se à aglutinação de *streams* em um único arquivo, como, por exemplo, a união de um *stream* de vídeo e um *stream* de áudio;
- *Demux*: Processo inverso do *Mux*, de modo simples, separa os *streams* contidos em um arquivo;

- *Streaming*: Fluxo de dados, que no caso do FFmpeg refere-se à fluxo de dados multimídia;
- *Filter*: Responsável por avaliar e modificar as redundâncias da multimídia;
- *Play*: O FFmpeg tem um media player simples, o qual é utilizado na maioria das vezes para testar as várias conversões do FFmpeg.

Por possuir inúmeras possibilidades e configurações o FFmpeg é organizado em sete diferentes bibliotecas, cada uma com um objetivo definido (FFmpeg - <https://ffmpeg.org/about.html>, acesso em 31/05/2016). Há a possibilidade de serem utilizadas fora do FFmpeg via um Application Programming Interface (API) público. As bibliotecas são:

- *Libavutil*: Contém funções para simplificar a programação, incluindo gerador de números aleatórios, estrutura de dados, rotinas matemáticas, multimídias centrais e muitas outras opções relacionadas;
- *Libavcodecs*: Contém os *encoder* e *decoders* para os programas que codificam e decodificam (*codecs*) áudio e vídeo;
- *Libaformat*: Contém os demuxers e muxers, inclui funcionalidades compatíveis com o ISO/IEC 13818-1;
- *Libavdevice*: Contém dispositivos de entrada e saída para renderizar (obter o produto final de um processamento digital) muitos *frameworks* comuns de entrada e saída, incluindo Video4Linux, Video4Linux2, Vfw e ALSA.
- *Libavfilter*: Contém os filtros de multimídia;
- *Libswscale*: Responsável por otimizar o dimensionamento de imagens e operações de conversão de formatos de cores para *espaço/pixel*;
- *Libwresample*: Responsável por otimizar o áudio, reamostrando-o, remasterizando-o e convertendo em outros formatos.

Esta divisão é feita para facilitar o desenvolvimento e aperfeiçoamento do *software* por não desenvolvedores, ou seja, a organização facilita a modificação e adição de novas funcionalidades e ferramentas.

Todavia, a sua utilização é complexa por ser uma aplicação controlada por linha de comando com muitos parâmetros de entrada. É necessário, para usos mais avançados, um estudo preliminar antes de ser possível utilizá-lo da maneira correta. Um exemplo de comando é apresentado na Figura 4. Todos estes parâmetros apresentados devem ser digitados manualmente ou colados no terminal para iniciar o funcionamento do FFmpeg.

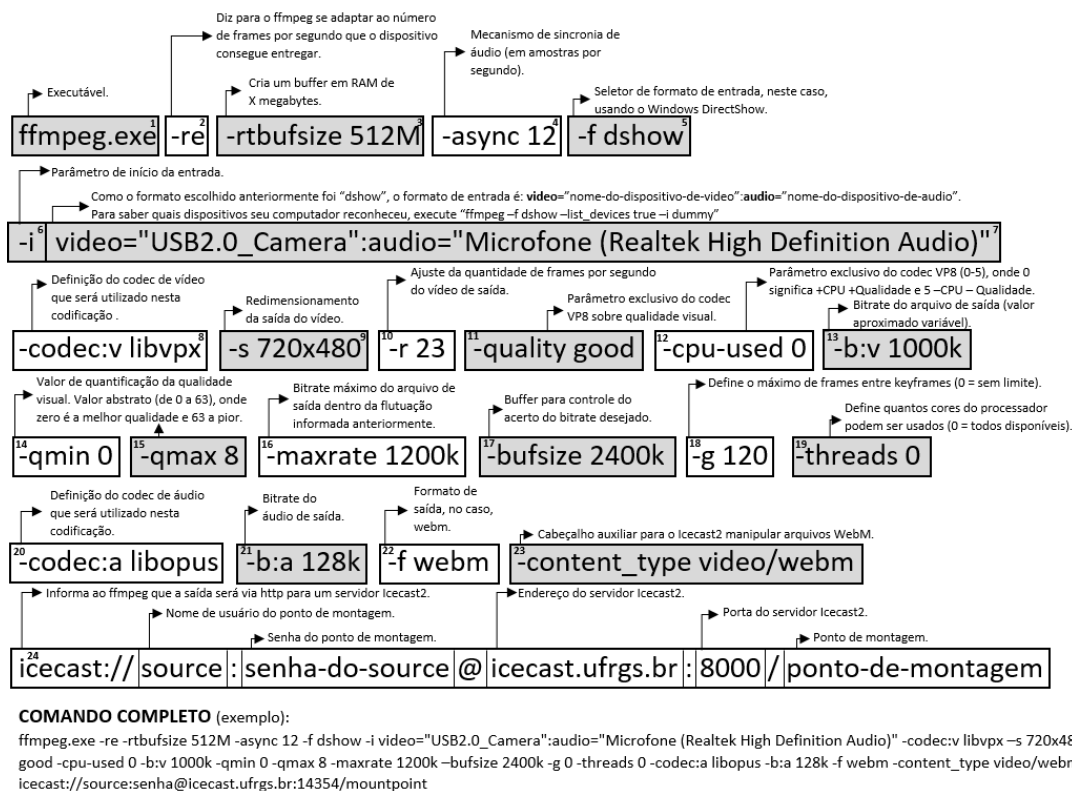


Figura 4 – Funcionamento do FFmpeg com saída para Icecast (Fonte: Desenvolvimento e adaptação de tecnologias livres para solução de *streaming* HLS na UFRGS).

2.3.2. Compatibilização do FFmpeg com o SBTVD

O FFmpeg foi utilizado por Endres em seu trabalho de conclusão de curso para o desenvolvimento do multiplexador MPEG2 compatível com o padrão de televisão digital SBTVD (Endres, 2014). Desenvolveu o suporte para somente dois programas e adicionou as tabelas que são opcionais para a norma ISO/IEC (utilizada pelo FFmpeg), mas são obrigatórias para o SBTVD. Este trabalho foi desenvolvido com intuito de testar o *set-top box* desenvolvido no Laboratório de Processamento de Sinais e Imagens (LaPSI) utilizando a transmissão do TS pelo ar ou decodificando o arquivo de TS armazenado.

Para gerar os arquivos TS's de teste foi escolhido o FFmpeg pelos seguintes motivos: ser um *freeware*, ter o código fonte sobre as licenças de código livre, a organização do código fonte facilitar a modificação e adição de novas funcionalidades, estar atualmente em desenvolvimento e conter compatibilidade com os *streams* elementares definidas no SBTVD, como o vídeo H.264 e o áudio *Advanced Audio Coding (AAC)/ Low Overhead Audio Transport Multiplex (LATM)*.

Após a escolha do FFmpeg utilizou-se da estrutura organizacional para adicionar as compatibilidades com o SBTVD. Fez modificações na biblioteca "*libaformat*", descrita acima, pois nela acontece a criação do TS. Adicionou parâmetros obrigatórios descritos na seção 2.2, e utilizou nomes diretamente relacionados com as suas funções. No total, foram sete opções adicionais. Estas estão detalhadas na bibliografia (Endres, 2014; ABNT NBR 15603-1, 2007; ABNT NBR 15603-2, 2007; ABNT NBR 15603-3, 2007) e no Apêndice. Modificou funções diretamente ligadas à compatibilização e outras funções que trabalhassem na maneira organizacional do FFmpeg.

A modificação de função abaixo é ligada diretamente aos objetivos deste projeto:

- Controle dos *streams* de entrada com o mapeamento: quando o usuário chama o FFmpeg e define as entradas. Cada entrada tem um determinado *stream* multimídia. Para controlar quais das multimídias de cada entrada serão utilizadas o comando “-map” é utilizado. Antes das alterações feitas na compatibilização, havia apenas um programa no TS de saída. Porém, agora é necessário definir qual *stream* corresponde a qual programa. Para isso foi implementada uma ordem de sequência para *streams*/programa. A ordem deve ser: *Streams* de vídeo (ordenadas do primeiro programa ao último) e *streams* de áudio (ordenadas do primeiro programa ao último).

Além destes parâmetros e modificações, foram adicionados dois *inputs* de controles de variáveis, os quais são elucidados abaixo:

- *mpegts_final_nb_services*: parâmetro onde o usuário informa o número total de programas que utilizará. Esta opção não é utilizada na versão criada por Endres, foi feita para ser utilizada em projetos futuros.
- *mpegts_transmission_profile*: responsável por informar ao FFmpeg se serão utilizados múltiplos programas e qual o perfil de múltiplos serviços. O chamado perfil 1 informa que serão utilizados dois programas, um *Full-seg*, com um *stream* de vídeo de alta definição e um *stream* de áudio, e outro *1-Seg*, com um *stream* de vídeo de baixa resolução e outro *stream* de áudio. Por fim, o perfil 2 informa que serão utilizados de três a cinco programas, dois a quatro programas de definição *standard*, com um *stream* de vídeo e um *stream* de áudio, e um programa *1-Seg*, com um *stream* de vídeo de baixa resolução e outro *stream* de áudio. Todavia, o perfil 2 não foi implementado na época.

3. Implementação do Projeto

O desenvolvimento do projeto e características relevantes da sua implementação são descritos nas seções seguintes.

3.1. TS de múltiplos programas perfil 2

O perfil 2 corresponde à criação de um TS com um número variado de programas, como mostrado na seção 2.3.2. Para implantação, o FFmpeg foi modificado em seu próprio código. A biblioteca modificada foi a *libaformat*, responsável pela multiplexação do TS, cuja versão utilizada é a 55.36.100.

No processo de testes de múltiplos programas do *set-top box* foram utilizados *streams* de vídeos e áudios de TS's com o correto funcionamento garantido. Isto, no intuito de testar somente a múltipla programação. Por esta razão os parâmetros “*-vcodec copy*” e “*-acodec copy*”, cuja função é, respectivamente, copiar o vídeo de entrada na saída e copiar o áudio de entrada na saída, devem ser utilizados. Devido a isso, não é necessário o FFmpeg ter os seus *codecs* compatíveis com o SBTVD, pois não será necessário utilizar codificação ou decodificação.

Portanto, no processo de geração de um TS o passo mais importante na sua construção é a multiplexação, passo pelo qual a biblioteca *libaformat* é responsável. Dentro desta biblioteca existem estruturas responsáveis por cada passo da construção do TS. As mais relevantes e suas respectivas funções são: *mpegTSSection* para as tabelas; *mpegTSService* para os programas; *mpegTSWrite* para o TS; e *mpegTSWriteStream* para os *streams*. Estas estruturas representam as definições das normas do SBTVD e são utilizadas somente quando o multiplexador reconhece a extensão “.ts” no arquivo de saída.

Além destas estruturas há três funções com grande relevância. A *av_write_header()* escreve o cabeçalho de cada pacote de TS, ou seja, é responsável por reproduzir as tabelas das normas do SBTVD. Também há a função *av_write_package*, esta escreve o pacote de TS e é responsável por deixá-lo sempre com 188-Bytes, condizente com a norma explicada na seção 2.1. Por fim, e não menos importante, a *av_write_trailer()* responsável por definir os PES de cada *stream*.

Na estrutura lógica da montagem do TS, foram adicionadas as funções referentes à criação de novos programas, esta chamada de *mpegts_add_service*. Cada *service_id* de cada programa foi definido para corresponder a valores compatíveis com a norma, ou seja, o tipo de programa e o número de programa foram definidos individualmente. O nome do provedor foi definido como “FFmpeg” para identificar esta alteração nos testes.

Além disso, o nome dado aos programas foi modificado (parâmetro *service_name*). Antes da atualização todos os programas eram nomeados de “Service01”. Agora, este nome varia e informa o tipo do programa. Por exemplo: se for gerado um TS com dois programas um terá o nome “Service 1 – TV” e o outro “Service 2 – 1Seg”; se for gerado um TS com três programas um terá o nome “Service 1 – TV”, outro “Service 2 – TV” e o último “Service 3 – 1Seg”. Vale notar que esta modificação também foi feita no perfil

múltiplos programas 1. Sendo assim, o programa do tipo *1-Seg* será sempre o último a ser adicionado ao TS. Portanto, ao gerar o arquivo de TS os parâmetros de *input* “-map” devem ser organizados de tal forma, que, o último “map” contendo vídeo e o último “map” contendo áudio sejam aqueles referentes ao programa *1-Seg*.

Foi modificado o parâmetro de entrada “mpegs_final_nb_services”, descrito na seção 2.3.2. Nesta versão, este *input* está relacionado diretamente com quantos programas o TS criará. Portanto, ele deve ser sempre definido quando a criação de múltiplos programas perfil 2 for utilizada. Sendo assim, o *range* dele foi alterado para no máximo 5. Para melhor entendimento, quando o *input* “-mpegs_transmission_profile 2” for utilizado, um range de 3 a 5 deve ser definido ao *input* “-mpegs_final_nb_services” e assim um número igual de programas serão criados.

Por se tratar de um *software* de código livre e de livre modificação, foi disponibilizado o arquivo com estas modificações no *link*: https://github.com/andregobo/TCC_FFmpeg.

3.2. GUI para geração de TS

O funcionamento do FFmpeg se dá por comandos no terminal. Por ser o objetivo deste projeto tornar mais fácil e ágil a criação de TS's foi desenvolvida uma Interface Gráfica do Usuário (GUI) com os principais comandos de *input* para a criação de TS's. Esta solução, de simplificar a utilização do FFmpeg criando uma GUI, também foi utilizado no desenvolvimento e adaptação de tecnologias livres para solução de *streaming* HLS na UFRGS (Bitzki e Kleemann, 2016).

A primeira etapa foi escolher a plataforma para o desenvolvimento da GUI. Para a decisão de qual programa utilizar, foi levada em conta a gratuidade do *Integrated Development Environment* (IDE), possibilidade de utilizá-la no Linux e as possíveis melhorias para o futuro. As soluções mais viáveis seriam utilizar:

- Code::Blocks com o plug-in RDA (*Rapid Application Design*) wxSmith (WxSmith tutorials - http://wiki.codeblocks.org/index.php/WxSmith_tutorials, 01/06/2016): IDE utilizada com mais frequência no laboratório, porém nem sempre utilizando o plug-in; grande parte dos programas criados a utilizam; possibilidade de utilizá-lo no Linux, Windows e Mac OS e; é um *software* livre

- Qt Creator (QtCreator - <https://www.qt.io/>, acesso em 01/06/2016): IDE utilizada com menos frequência no laboratório, interesse da equipe por conhecê-la; poucos programas criados no laboratório a utilizam, em especial o ControleTuner; possibilidade de utilizá-lo no Windows, Linux e Mac OS e plataformas *mobile* e; é um *software* livre.

O ControleTuner é um programa criado com o Qt Creator para capturar o sinal de televisão digital do ar, utilizando um PenTV (pendrive que capta sinal de televisão digital), e transmitir para a placa *Field Programmable Gate Array* (FPGA), na qual o decodificador da televisão digital é descrito. Além disso possibilita o envio de um arquivo armazenado no computador para o FPGA utilizando um cabo Ethernet.

A união dos programas de teste do *hardware* decodificador de televisão digital e do programa criador de TS torna-se atrativa, pois ambos têm o mesmo objetivo principal: produzir testes para a placa FPGA. Além disso, uma versão simplificada da GUI já vinha

sendo desenvolvida no LaPSI, utilizando o Qt Creator. Por estes motivos, além de ser gratuito e ter a possibilidade de utilização no Linux, o Qt Creator foi escolhido para esta solução. A versão utilizada foi a 3.5.0 (baseada em Qt 5.4.2), versão mais recente na data de início do desenvolvimento. Vale ressaltar que a versão em desenvolvimento no LaPSI não foi utilizada para a implementação deste projeto.

Após a escolha da IDE foi necessário apontar quais *inputs* do FFmpeg seriam utilizados. Se todos os *inputs* fossem levados em consideração, necessitaria mais tempo de desenvolvimento e tornaria a interface muito complexa, podendo até dificultar a geração de TS. Eleitos os *inputs*, passou-se à fase de implementação, o *design* final da GUI_FFmpeg é ilustrada na Figura 5. Pode-se dividir a GUI pelos seguintes comandos: executar o FFmpeg, abrir tutorial, mostrar comando a ser executado, organização das janelas, escolher entradas, definir saída, escolha dos *streams* a serem utilizados, manipular os *streams* para cada programa, duração do TS, copiar vídeo e copiar áudio, *muxrate* e funções do FFmpeg relacionados diretamente com o SBTVD.

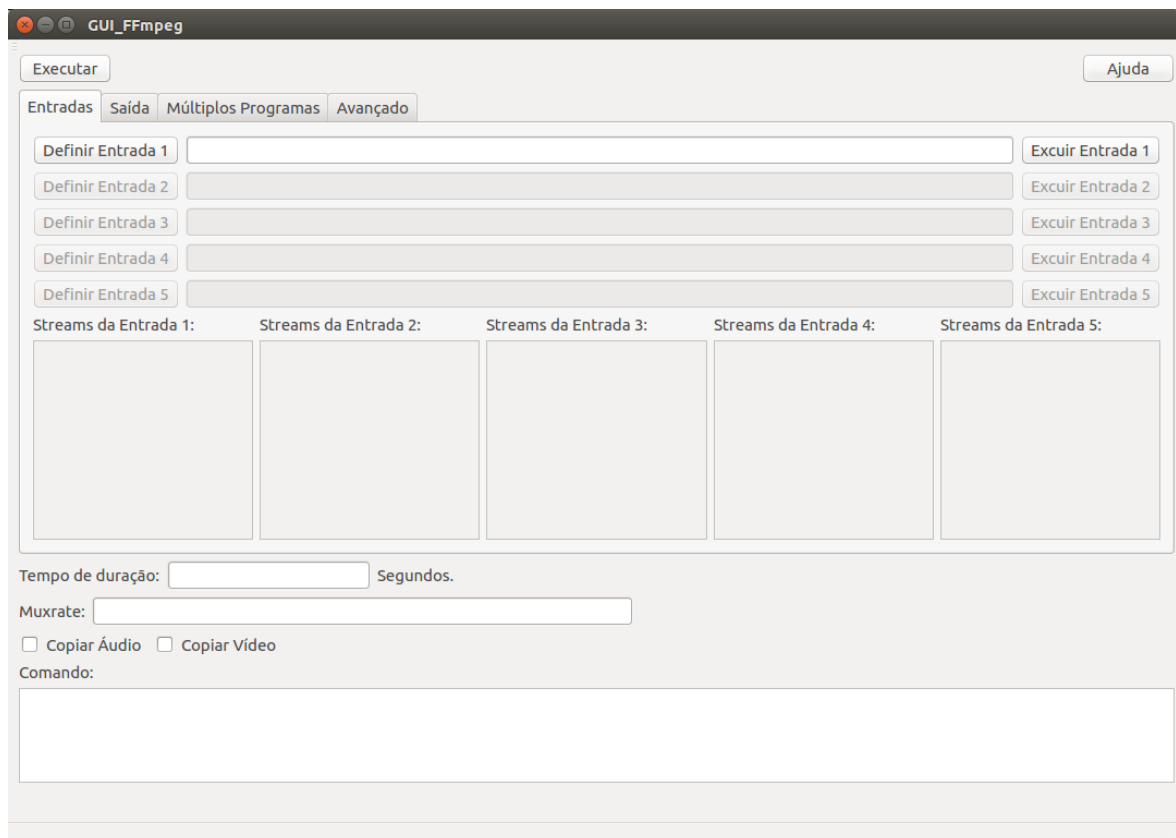


Figura 5 – Design da GUI_FFmpeg.

Cada um dos comandos criados são definidos e especificados abaixo. Há também um tutorial, no Apêndice, que contém mais detalhes.

- Executar o FFmpeg: este comando está por trás do botão "Executar", ilustrado na Figura 6. Deve ser clicado quando todas as configurações já estão prontas. Ou seja, quando selecionado, abre um terminal em outra janela e executa o comando montado pelo usuário. O terminal permanece aberto após a conclusão da execução. O comando executado é o mostrado na caixa de diálogo "Comando" explicado mais a frente no texto.

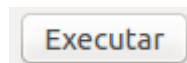


Figura 6 – Botão Executar.

- Abrir tutorial: este comando está por trás do botão "Ajuda", ilustrado na Figura 7. Este botão abre o arquivo contendo o tutorial do programa. Este tutorial também se encontra no Apêndice.

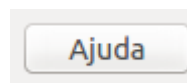


Figura 7 – Botão Ajuda.

- Mostrar comando a ser executado: optou-se por utilizar esta função para que os usuários conseguissem adicionar mais parâmetros de *input* além dos quais a GUI contempla. Portanto, esta caixa de diálogo, ilustrada na Figura 8, contém o comando executado quando o botão "Executar" é clicado. Como o programa faz diretamente um *link* com esta caixa de diálogo e o que é executado, todas as alterações feitas pelo usuário são levadas em conta. Esta função foi muito útil para testes da GUI_FFmpeg, pois enquanto o usuário seleciona as opções de *input* a GUI monta simultaneamente o comando a ser executado.



Figura 8 – Caixa de diálogo "Comando".

- Organização das janelas: foi utilizado uma divisão de abas, para alguns grupos de funções, no intuito de simplificar e organizar a GUI. Estes grupos são esclarecidos mais adiante no texto. Logo, foram criados quatro grupos de funções com nomes bastante explicativos, são eles: "Entradas", "Saída", "Múltiplos Programas" e "Avançado". Estas abas são ilustradas na Figura 9.

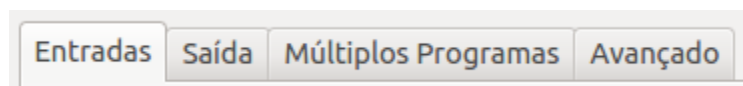


Figura 9 – Abas separando comandos.

- Escolher entradas: esta função, observada na Figura 10, tem como objetivo facilitar a inserção de entradas. Tem o seguinte funcionamento: quando o usuário clicar em "Definir Entrada 1" abrirá uma janela de diálogo para selecionar um arquivo de entrada, ilustrada na Figura 11. O mesmo acontece para as entradas 2, 3 e 4, no intuito de eliminar erros, cada entrada só pode ser ativada se o usuário já definiu a(s) entrada(s) antecedente(s). Ou seja, a entrada 2 só pode ser definida se a entrada 1 já foi definida, e assim sucessivamente. Além disso, há uma linha de edição das entradas, cuja finalidade é informar ao usuário qual caminho e qual arquivo foi selecionado e possibilitar a inserção manual do caminho/arquivo. Também há botões de exclusão das entradas, os quais servem para excluir a entrada referente e as outras entradas necessárias sem precisar apagar todos os textos das caixas de diálogo, por exemplo, após clicar em "Excluir Entrada 1" o programa excluirá a entrada 1 e todas as entradas com maior número.

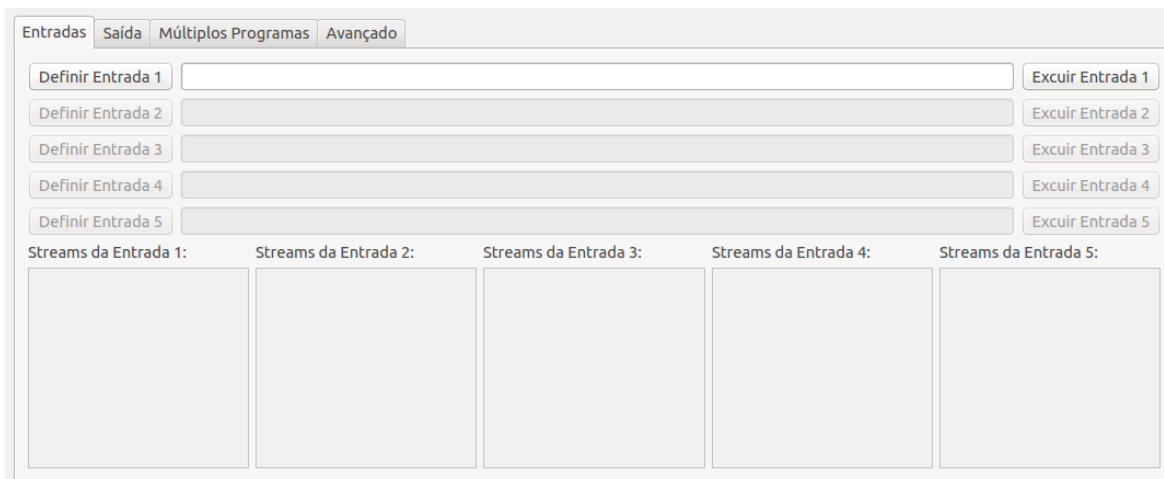


Figura 10 – Aba de entradas.

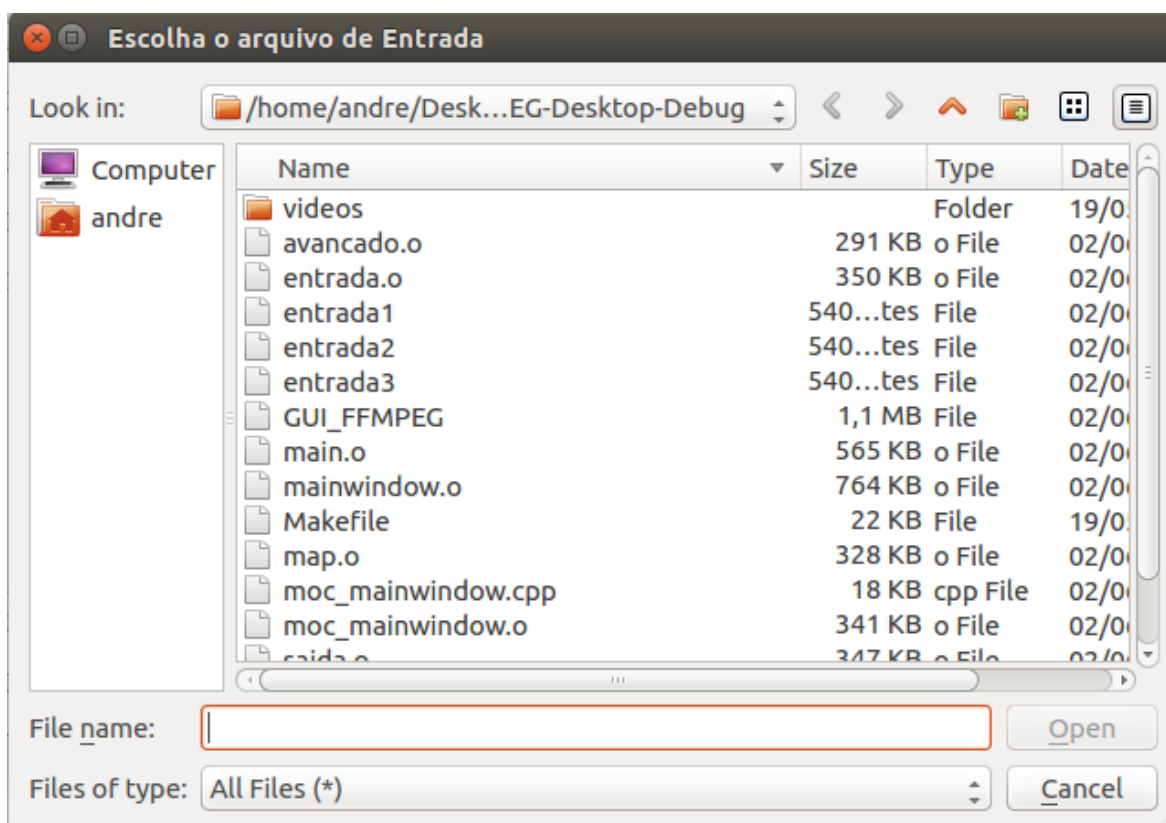


Figura 11 – Caixa de diálogo para selecionar uma entrada.

- Escolha dos *streams* a serem utilizados: funciona em conjunto com as funções de escolha de entradas. A área gráfica da função pode ser observada embaixo da área de inserção de entradas na Figura 12. Ao selecionar um arquivo de entrada, a GUI identifica os *streams* que fazem parte do arquivo, identifica seu número, seu tipo e sua compressão. Ao serem identificadas estas informações são escritas ao lado de *checkboxes*, cuja localização é dentro do retângulo referente à entrada, como é ilustrado na Figura 12. Ao marcar estes *checkboxes*, o usuário define quais *streams* serão utilizados na execução do comando.

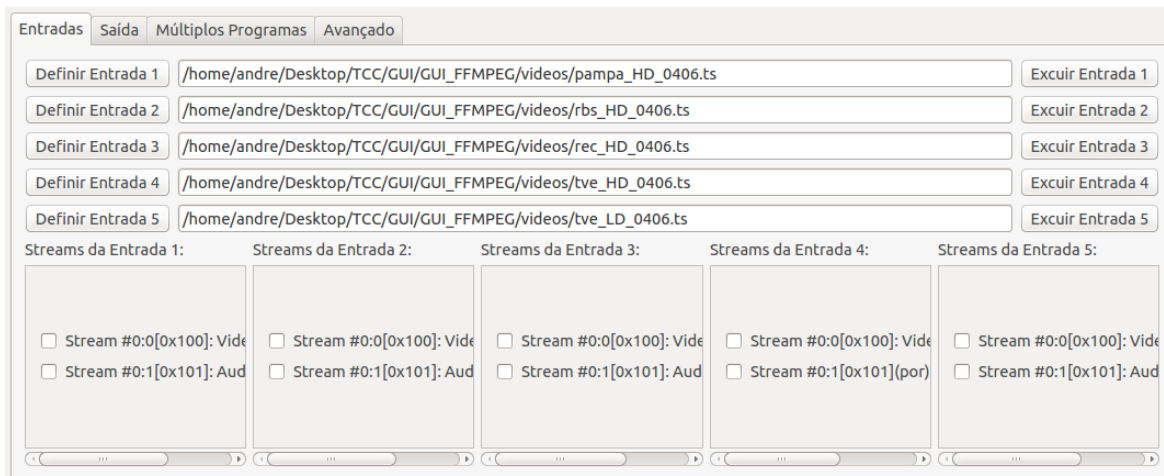


Figura 12 – Escolha dos *Streams*.

- Definir saída: nesta aba o usuário define qual a pasta e nome do arquivo gerado pelo FFmpeg, ilustrada pela Figura 13. Para facilitar esta definição, ao clicar em “Definir pasta de saída” uma caixa de diálogo se abrirá para que a pasta de destino seja selecionada, esta janela é ilustrada na Figura 14. Após selecionar a pasta de destino, o caminho selecionado será informado na linha de edição, à direita do botão “Definir pasta de saída”. Esta linha de edição também pode ser editada manualmente. Além disso, o usuário define o nome da saída e sua extensão na caixa de diálogo ao lado de “Nome do arquivo de saída”. A extensão não foi fixada em “.ts” para não limitar totalmente o programa, porém, se o usuário não escolher extensão alguma ela será automaticamente “.ts”. Também há o *checkbox* com a descrição “Sobrescrever saída”. Se marcado o FFmpeg irá sobrescrever, se existir, o arquivo com nome igual ao contido no campo “Nome do arquivo de saída” automaticamente.

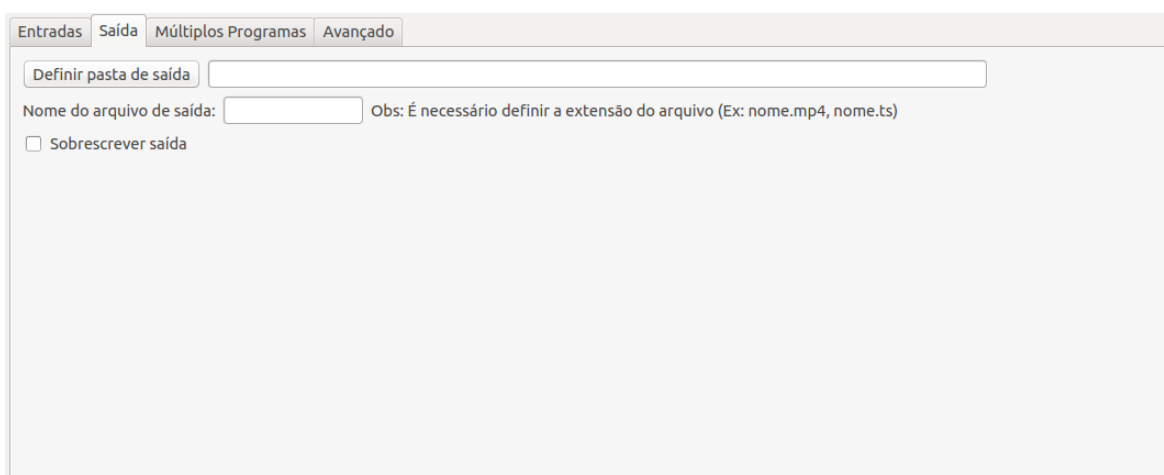


Figura 13 – Aba de saída.

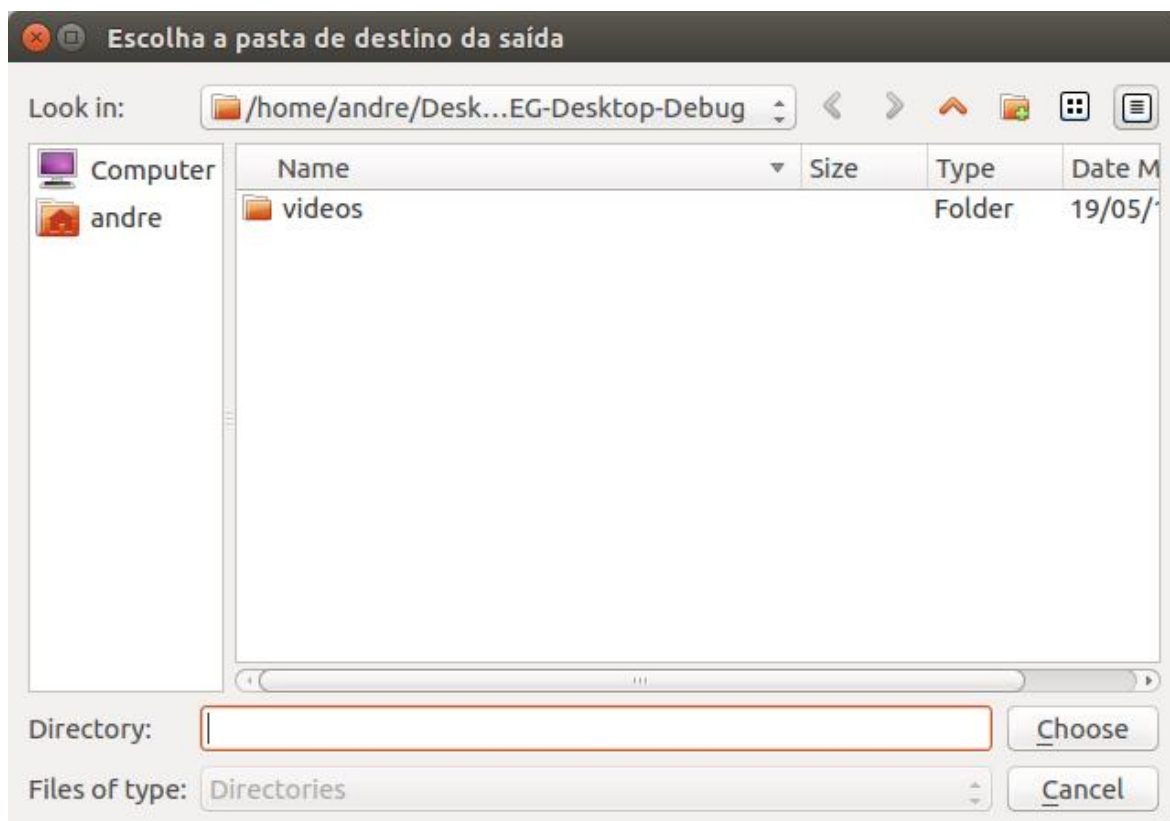
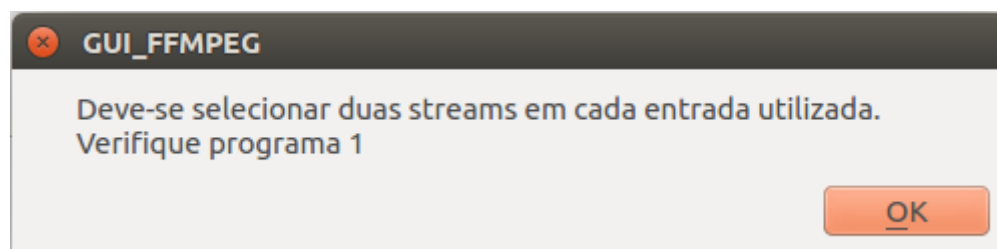


Figura 14 – Caixa de diálogo para selecionar a pasta de destino da saída.

- Manipular os *streams* para cada programa: esta aba tem a função de ligar os *streams* da entrada com o programa desejado, levando em conta que o usuário quer utilizar a opção de múltiplos programas. Por estar diretamente relacionado com as entradas, as suas opções também são dependentes delas. À medida que o usuário adiciona entradas, mais opções de escolhas são ativadas. Além disso, para não causar erros e confusão, os programas de maior número só são ativados se os programas de menor ordem foram definidos. Na Figura 15 e na Figura 16 observam-se os dois casos extremos. No primeiro caso, nenhuma entrada foi selecionada. No segundo caso, todas as entradas foram selecionadas e, a partir delas, será criado um TS com cinco programas, cada um utilizando uma entrada diferente. Além disso, há o botão “Utilizar múltiplos programas”, o qual, se marcado como vazio, faz um *reset* na aba “Múltiplos Programas”, voltando todos os programas para sua forma inicial (“Desativado”). Para lembrar o usuário que cada programa suporta somente dois *streams*, um aviso é mostrado quando há um desacordo. O aviso pode ser visto na Figura 17.

Figura 15 – Aba Múltiplos Programas.

Figura 16 – Aba Múltiplos Programas com seleções (As 5 entradas estão definidas).

Figura 17 – Erro quando não há dois *streams* selecionados na entrada.

- Duração do TS: o Ffmpeg possibilita ao usuário diminuir o tempo da multimídia. Como o objetivo é testar o *hardware* desenvolvido com diferentes tipos de configurações um tempo mais curto para o TS torna-se atrativo. A Figura 18 ilustra a área responsável pela configuração.

Figura 18 – Determinar duração do TS.

- Copiar vídeo e copiar áudio: quando selecionadas, informam ao FFmpeg que o vídeo e o áudio devem ser copiados diretamente, sem passar por um *codec* (codificador ou decodificador). Esta opção é ilustrada na Figura 19.

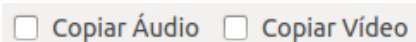


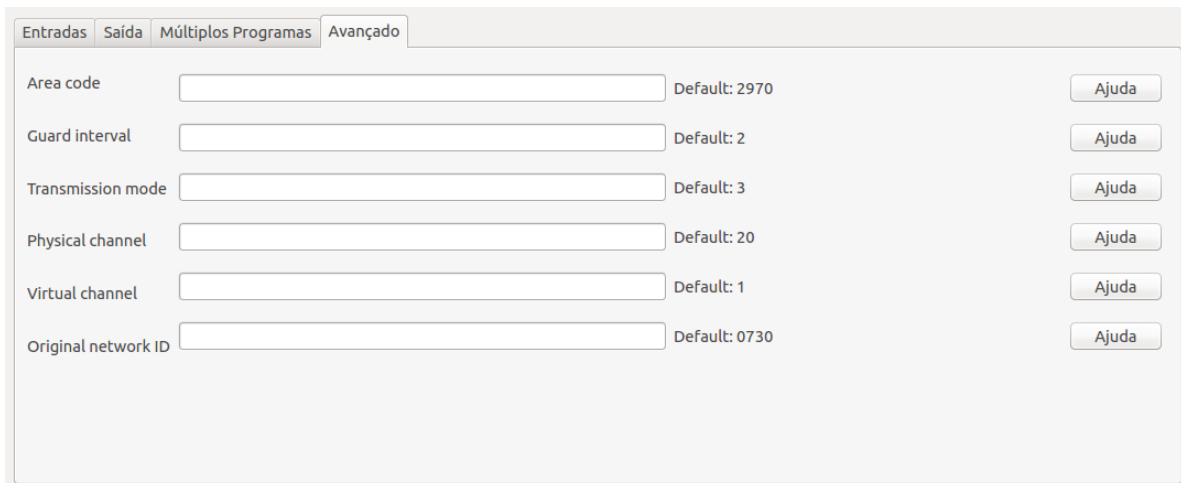
Figura 19 – Selecionar utilização de *codecs*.

- *Muxrate*: parâmetro de grande importância para o correto funcionamento do TS, pois fixa a sua taxa de *bits* (*bitrate*). Sem este parâmetro o TS de saída será formado com um *bitrate* variável, cuja compatibilidade com o MPEG2 não é satisfeita. Portanto este parâmetro deve ser sempre especificado. O valor deste parâmetro pode ser calculado analisando os *streams* de *input* pelo próprio FFmpeg. Após obter a informação de cada *stream* utilizado, o valor do *muxrate* deve ser maior que a soma de todos os *bitrates* observados. Se o valor for menor, o vídeo do TS de saída apresentará erros no *framerate*, o vídeo e áudio serão apresentados como se estivessem em câmera lenta. Ao contrário, se o *muxrate* for definido com um valor muito maior que a soma dos *bitrates*, o arquivo gerado apresentará muitos pacotes nulos e terá um tamanho de arquivo maior que o necessário. Portanto, para não ocorrer erros de sub ou sobre dimensionamento, esse parâmetro deve ser definido com o valor da soma dos *bitrates* dos *streams* utilizados, mais 11%. Este acréscimo de 11% se deve a uma margem de erro, a qual foi observada nos *streams* captados no ar por Endres (Endres, 2014). A parte de inserção do valor é ilustrada na Figura 20.



Figura 20 – Configuração do *Muxrate*.

- Funções do FFmpeg relacionados diretamente com o SBTVD: estas funções foram as referidas na seção 2.3.2, cuja adição foi feita por Endres (Endres, 2014). A aba contendo-as é mostrada na Figura 21. Os valores de default observados ao lado de cada caixa de texto são os mesmos utilizados por Endres em seus testes. Ao clicar nos botões “Ajuda” ao lado de cada parâmetro, uma janela com as mesmas informações do tutorial, presente no Apêndice, irá aparecer informando sua utilidade.



The image shows a software interface with a tabbed menu at the top containing 'Entradas', 'Saída', 'Múltiplos Programas', and 'Avançado'. The 'Avançado' tab is active. Below the tabs, there are six rows of configuration fields. Each row consists of a text input field, a 'Default' value, and an 'Ajuda' button. The fields and their default values are: 'Area code' (Default: 2970), 'Guard interval' (Default: 2), 'Transmission mode' (Default: 3), 'Physical channel' (Default: 20), 'Virtual channel' (Default: 1), and 'Original network ID' (Default: 0730).

Field Name	Default Value
Area code	2970
Guard interval	2
Transmission mode	3
Physical channel	20
Virtual channel	1
Original network ID	0730

Figura 21 – Aba avançado.

Para organizar e facilitar o acesso de outros desenvolvedores, os códigos da GUI foram divididos em cinco partes, são elas: “avancado”; “entrada”; “main”; “mainwindow”; “map” e; “saida”. Seus nomes estão diretamente relacionados com a suas funções.

Sendo o FFmpeg um *software* livre de código aberto, licenciado pela GNU *Lesser General Public License*, foi disponibilizado para amplo acesso todos os códigos desenvolvidos na criação da GUI_FFmpeg, além de disponibilizar também o tutorial encontrado no Apêndice, em https://github.com/andregobo/GUI_FFmpeg.

4. Testes e Resultados

Abaixo são mostrados os testes e resultados dos projetos desenvolvidos. Para organização, este capítulo é separado em duas seções: “Testes e resultados do FFmpeg” e “Testes e resultados do GUI_FFmpeg”.

4.1. Testes e resultados do FFmpeg

Para testar o resultado da geração de TS utilizando a opção de múltiplos programas perfil 2, foram utilizados: o próprio *software* FFmpeg, o *software* SBTVD_parser, o *software* VLC e o processo de retransmissão disponível no LaPSI. Por serem procedimentos repetitivos com grande número de dados, optou-se por tomar como exemplo o caso de múltiplos canais com dois programas *standard* e um programa *1-Seg*. Os outros casos com mais programas *standard's* foram analisados igualmente, apresentando resultados semelhantes ao exemplo que será utilizado. Os *streams* utilizados no teste são retirados de TS's exemplos. Os *streams* do programa *standard* foram retirados de: <http://www.pjdaniel.org.uk/mpeg/downloads/mux1-cp.zip>. Os *streams* do programa *1-Seg* foram capturados por Endres (Endres, 2014) e utilizados em seus testes. Contando com estas fontes de *streams*, um novo TS foi criado utilizando a solução descrita na seção 3.1. Este TS foi criado ao executar o código mostrado na Figura 22.

```
ffmpeg -i /home/andre/Desktop/testes/TS_exemplo_1.ts -i
/home/andre/Desktop/testes/TS_exemplo_1.ts -i
/home/andre/Desktop/testes/TS_exemplo_2.ts -map 0:0 -map 1:2 -map 2:0
-map 0:1 -map 1:3 -map 2:1 -vcodec copy -acodec copy -
mpegtts_original_network_id 0730 -mpegtts_area_code 2970 -
mpegtts_guard_interval 2 -mpegtts_transmission_mode 3 -
mpegtts_physical_channel 20 -mpegtts_virtual_channel 1 -muxrate
10000000 -mpegtts_final_nb_services 3 -mpegtts_transmission_profile 2
-mpegtts_flags latm -t 9 -y /home/andre/Desktop/testes/TS_criado.ts
```

Figura 22 – Comando executado para criar o TS de exemplo.

4.1.1. Análise feita com o FFmpeg

Ao inserir uma entrada na linha de comando do FFmpeg e executá-la, o *software* analisa este arquivo e mostra ao usuário diversos parâmetros relevantes. Dentre eles estão: nome, número e provedor dos programas, *streams* componentes dentro de cada programa, tipo de codificação utilizado em cada *stream*, duração total do TS, *bitrate* do TS e dos *streams* individualmente, entre outros. A Figura 23 ilustra a análise do TS criado. Para gerar esta análise foi utilizado os parâmetros de *input* do FFmpeg ilustrados na Figura 24 .


```

Input #0, mpegts, from '/home/andre/Desktop/testes/teste3_golf_oucho_policia.ts':
Duration: 00:00:09.07, start: 1.400000, bitrate: 9684 kb/s
Program 23360
Metadata:
  service_name      : Service 1 - TV
  service_provider: FFmpeg
  Stream #0:0[0x100]: Video: mpeg2video (Main) ([2][0][0][0] / 0x0002), yuv420p(tv), 72
0x576 [SAR 64:45 DAR 16:9], max. 6500 kb/s, 25 fps, 25 tbr, 90k tbn, 50 tbc
  Stream #0:1[0x103](eng): Audio: mp2 ([3][0][0][0] / 0x0003), 48000 Hz, stereo, s16p,
251 kb/s
Program 23361
Metadata:
  service_name      : Service 2 - TV
  service_provider: FFmpeg
  Stream #0:2[0x101]: Video: mpeg2video (Main) ([2][0][0][0] / 0x0002), yuv420p(tv), 72
0x576 [SAR 64:45 DAR 16:9], max. 4350 kb/s, 25 fps, 25 tbr, 90k tbn, 50 tbc
  Stream #0:3[0x104](eng): Audio: mp2 ([3][0][0][0] / 0x0003), 48000 Hz, stereo, s16p,
251 kb/s
Program 23386
Metadata:
  service_name      : Service 3 - 1Seg
  service_provider: FFmpeg
  Stream #0:4[0x102]: Video: h264 (Constrained Baseline) ([27][0][0][0] / 0x001B), yuv4
20p, 320x180 [SAR 1:1 DAR 16:9], 14.99 fps, 29.97 tbr, 90k tbn, 29.97 tbc
  Stream #0:5[0x105]: Audio: aac_latm ([17][0][0][0] / 0x0011), 48000 Hz, stereo, fltp
At least one output file must be specified

```

Figura 23 – Análise feita pelo FFmpeg do TS criado.

```
ffmpeg -i /home/andre/Desktop/testes/TS_criado.ts
```

Figura 24 – Comando utilizado para fazer a análise do TS criado pelo FFmpeg.

Na Figura 23 é possível observar que: o TS criado tem três programas diferentes, cada um com um nome (*service_name*) e número (*service_id*) diferentes dos demais e o mesmo provedor de programa (*provider_name*); cada programa tem dois *streams* diferentes, um de áudio e outro de vídeo; *bitrate* e forma de compressão de cada *stream*; *bitrate* do TS criado; tempo de duração do TS. Todas estas informações estão apresentadas na Tabela 2. Sendo observada a existência de três programas diferentes entre si e com *streams* diferentes.

Tabela 2 – Informações do TS criado obtidas com o FFmpeg.

Parâmetro informado	Valores para o programa 1	Valores para o programa 2	Valores para o programa 3
Nome do programa (<i>service_name</i>)	Service 1 -TV	Service 2 - TV	Service 3 - 1Seg
Número do programa (<i>service_id</i>)	23360	23361	23386
Provedor do programa (<i>provider_name</i>)	FFmpeg	FFmpeg	FFmpeg
<i>Streams</i> componentes	Vídeo/Áudio	Vídeo/Áudio	Vídeo/Áudio
Tipo de codificação	mpeg2video(Main)/mp2	mpeg2video(Main)/mp2	h264 (Constrained Baseline)/aac_latm
<i>Bitrate</i> TS	9684 kb/s	9684 kb/s	9684 kb/s
<i>Bitrate</i> dos <i>streams</i>	6500/251 kb/s	4350/251 kb/s	Não informado
Tempo de duração	9,07 segundo	9,07 segundo	9,07 segundo

4.1.2. Análise do software SBTVD_parser

O software SBTVD_parser (MARQUES, 2015) foi desenvolvido para analisar TS's compatíveis com o SBTVD. Informa ao usuário parâmetros do TS, tais como: tabelas PSI, EIT, SDT e TOT, característica de cada programa e *streams* componentes. A versão do SBTVD_parser utilizada nos testes descritos é v0.34.

Sabendo disso, foi analisado o TS criado, cujo resultado está na Figura 25. Podemos observar a presença das tabelas componentes do cabeçalho do TS. Não foram recriadas as tabelas EIT. As tabelas CAT não aparecem explicitamente, porém o CRC (*Cyclic Redundancy Check*) foi gerado, como mostra a análise da tabela PAT na Figura 26. Esta tabela também não apareceu explicitamente na análise dos TS exemplos.

```
Parsing [/home/andre/Desktop/testes/TS_criado.ts]
▶ SDT (actual) (pid 0x11 / onid 0x2da (ZYA730))
▶ NIT (actual) (pid 0x10)
▶ TOT (pid 0x14)
▶ PAT (pid 0x00)
▼ PMT (service_id: 0x5b40)
▶ Section info
  PID: 0x1fc8
▼ service_id: 0x5b40
  type: TV
  number: 1
  PCR_PID: 0x0100
▶ program info descriptors: (length 6)
▶ elementary streams:
▼ PMT (service_id: 0x5b41)
▶ Section info
  PID: 0x1fc9
▼ service_id: 0x5b41
  type: TV
  number: 2
  PCR_PID: 0x0101
▶ program info descriptors: (length 6)
▶ elementary streams:
▼ PMT (service_id: 0x5b5a)
▶ Section info
  PID: 0x1fca
▼ service_id: 0x5b5a
  type: 1-seg
  number: 3
  PCR_PID: 0x0102
▶ program info descriptors: (length 6)
▶ elementary streams:
Packet length set to 188 bytes.
```

Figura 25 – Análise do Parser_SBTVD (v0.34) para o TS criado.

```

Parsing [/home/andre/Desktop/testes/TS_criado.ts]
▶ SDT (actual) (pid 0x11 / onid 0x2da (ZYA730))
▶ NIT (actual) (pid 0x10)
▶ TOT (pid 0x14)
▼ PAT (pid 0x00)
  ▼ Section info
    Id: 0x00
    section CRC: 0xeabd3859
    section_length: 21
    section_syntax_indicator: 1
    Id extension: 0x02da
    version number: 0x00
    current/next: current
    section number: 0x00
    last section number: 0x00
    tx_network: 0x2da (ZYA730)
  ▶ program_number: 0x5b40
  ▶ program_number: 0x5b41
  ▶ program_number: 0x5b5a
  service_count: 3
  ▶ PMT (service_id: 0x5b40)
  ▶ PMT (service_id: 0x5b41)
  ▶ PMT (service_id: 0x5b5a)
  Packet lenght set to 188 bytes.
    
```

Figura 26 - Análise do Parser_SBTVD (v0.34) para o TS criado, tabela PAT.

<pre> ▼ PMT (service_id: 0x5b40) ▶ Section info PID: 0x1fc8 ▼ service_id: 0x5b40 type: TV number: 1 PCR_PID: 0x0100 ▼ program info descriptors: (length 6) descriptor_tag: 0x55 descriptor_length: 0x04 ▼ Parental Rating Descriptor country_code: BRA rating: Free - ▼ elementary streams: ▶ ES_PID: 0x0100 type: 0x02- H.262 Video ▶ ES_PID: 0x0103 type: 0x03- ISO/IEC 11172-3 Audio </pre> <p>a)</p>	<pre> ▼ PMT (service_id: 0x5b41) ▶ Section info PID: 0x1fc9 ▼ service_id: 0x5b41 type: TV number: 2 PCR_PID: 0x0101 ▼ program info descriptors: (length 6) descriptor_tag: 0x55 descriptor_length: 0x04 ▼ Parental Rating Descriptor country_code: BRA rating: Free - ▼ elementary streams: ▶ ES_PID: 0x0101 type: 0x02- H.262 Video ▶ ES_PID: 0x0104 type: 0x03- ISO/IEC 11172-3 Audio </pre> <p>b)</p>
<pre> ▼ PMT (service_id: 0x5b5a) ▶ Section info PID: 0x1fca ▼ service_id: 0x5b5a type: 1-seg number: 3 PCR_PID: 0x0102 ▼ program info descriptors: (length 6) descriptor_tag: 0x55 descriptor_length: 0x04 ▼ Parental Rating Descriptor country_code: BRA rating: Free - ▼ elementary streams: ▶ ES_PID: 0x0102 type: 0x1b- H.264 - ISO/IEC 14496-10 Video ▶ ES_PID: 0x0105 type: 0x11- ISO/IEC 14496-3 Audio </pre> <p>c)</p>	

Figura 27 – Análise do Parser_SBTVD (v0.34) para o TS criado, tabela PMT. a) programa 1; b) programa 2 e; c) programa 3.

Analisando as tabelas PMT com mais detalhes, ilustrado na Figura 27, é possível notar que os números dos programas (*service_id*) são diferentes. Comprova-se assim que foram criados três programas com números diferentes, pois o número dos programas dado pelo SBTVD_parser e pelo FFmpeg são os mesmos (no SBTVD_parser o número está em hexadecimal). Além disso, os tipos de programa variam (TV e *1-Seg*) e cada um deles tem dois *streams*, sendo um de vídeo e outro de áudio. Estas informações são resumidas na Tabela 3.

Tabela 3 – Informações do TS criado obtidas com o SBTVD_parser.

Parâmetro informado	Valores para o programa 1	Valores para o programa 2	Valores para o programa 3
Tipo do programa	TV	TV	<i>1-Seg</i>
Número do programa (<i>service_number</i>)	0x5b40	0x5b41	0x5b5a
<i>Streams</i> componentes	Vídeo/Áudio	Vídeo/Áudio	Vídeo/Áudio
Tipo de codificação	H.264/ISO/IES 11172-3	H.264/ISO/IES 11172-3	H.264-ISO/IEC 14496-10/ISO/IEC 14496-3
Código do país	BRA	BRA	BRA
Idade mínima telespectador	Livre	Livre	Livre

Nota-se na Tabela 3 mais características além das observadas com a análise do FFmpeg. Observa-se que o código do país está correto e foi alterado, pois este código de um dos *streams* exemplo era “GRA”. Também, a categoria de idade foi recriada como desejada, pois foi programado para o FFmpeg gerar o TS utilizando este parâmetro como “Livre”.

4.1.3. Análise software VLC Media Player

O VideoLAN *Client Media Player* (VLC *Media Player*) (VLC media player - <http://www.videolan.org/vlc/>, acesso em 15/06/16), criado pela organização VideoLAN, é um *software* livre de código aberto que reproduz a maioria das multimídias existentes atualmente. Consegue reproduzir arquivos TS (extensão “.ts”) e reconhece a múltipla programação. Devido a isso foi utilizado para testar o TS criado. As figuras (Figura 28, Figura 29 e Figura 30) ilustram o momento de reprodução dos testes. O resultado foi satisfatório, não ocorreram problemas com a sincronia entre os áudios e vídeos. A versão utilizada foi a 2.2.1.



Figura 28 – Teste do TS criado com o VLC *Media Player*, programa 1.



Figura 29 – Teste do TS criado com o VLC *Media Player*, programa 2.

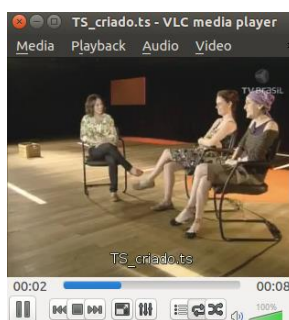


Figura 30 – Teste do TS criado com o VLC *Media Player*, programa 3.

Observe que a proporção de tamanho dos vídeos foi satisfatória, pois o programa 3 tem uma proporção menor. Esta se deve ao fato que o programa 3 é um programa *1-Seg*, com vídeo e áudio de menor qualidade, para serem reproduzidos em dispositivos móveis.

Além disso, a duração de todos os programas é a mesma, devido ao parâmetro “-t” utilizado para gerar o TS de teste.

4.1.4. Análise de transmissão do TS

Utilizando os equipamentos disponibilizados pelo LaPSI, placa DekTec DTA-115, conversor digital terrestre Visiontec VT7200 e televisão Philips para exibição, foram testados os TS’s gerados pela solução desenvolvida, transmitindo-os pelo ar. Os testes se resumiram em: transmitir o TS pelo canal 20 (509 MHz); testar o nível de sinal pelo decodificador; por fim, analisar a sincronia do áudio e vídeo transmitidos. A Figura 31 ilustra o procedimento utilizado.

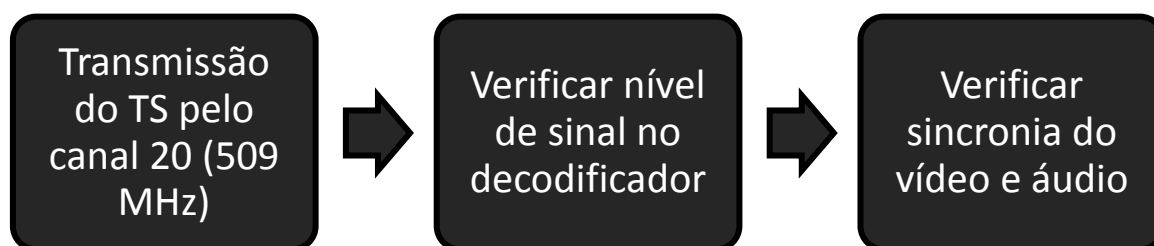


Figura 31 – Processo utilizado para testar os TS’s utilizando transmissão pelo ar.

Primeiro os TS’s de exemplos foram testados para confirmar a decodificação do vídeo e áudio e o funcionamento correto da transmissão. Para transmissão foi utilizado o *software StreamXpress* e uma placa de transmissão DekTec DTA-115 (*StreamXpress* - <http://www.dektec.com/products/applications/StreamXpress/>, acesso em 13/06/16). Os parâmetros utilizados foram os mesmos utilizados por Endres nos testes de TS’s com múltiplos programas perfil 1 (Endres, 2014). Após a configuração do programa e ativação da transmissão notou-se que o nível de sinal no conversor digital terrestre Visiontec VT7200 era extremamente relacionado à posição das antenas. Isso devido à baixa potência das antenas utilizadas, principalmente da antena de transmissão. Após ajuste do local ideal para as antenas o decodificador reconheceu o áudio e vídeo do TS exemplo, conseguindo reproduzi-las. Estes estavam em plena sincronização, não sendo notada nenhuma discrepância.

Depois de confirmada a transmissão dos TS’s exemplos, os TS’s criados foram testados. Todavia, o decodificador não reconheceu o sinal transmitido, sendo que as posições das antenas foram mantidas. Sabia-se que a TOT apresentava erros e não deviam ser transmitida, como Endres apontou em seus testes (Endres, 2014). Mesmo após eliminar a tabela TOT do TS transmitido, o decodificador não reconheceu o sinal. Devido à falha, foram testados outros decodificadores com o TS exemplo, apresentando sempre sintonia e sincronia entre áudio e vídeo, e com o TS criado, apresentando sempre falta de sinal.

Dada a negativa deste teste, passou-se à fase de examinar os possíveis problemas na transmissão. Com isso, foram testados TS’s múltiplos programas gerados pelo FFmpeg utilizando o perfil 1. Porém, novamente, todos os decodificadores apresentaram má sintonização. Considerando não ser do escopo deste projeto, não foram solucionados os problemas apresentados na transmissão, deixando como sugestões de trabalhos futuros.

4.2. Testes e resultados do GUI_FFmpeg

Todos os parâmetros e comandos foram testados e avaliados por membros da equipe do LaPSI. Vale ressaltar que, ao testar a GUI_FFmpeg, foram apontadas melhorias e sugestões que contribuiriam para um resultado compatível com o esperado. As mais relevantes são mostradas abaixo:

- Erros ao tentar ler ou gravar um arquivo com o caractere “espaço” (“ ”): internamente no GUI_FFmpeg, quando uma entrada ou saída são definidas o caminho referente a eles é salvo em uma QString. É salvo da forma usual, sendo o caractere “ ” realmente um espaço, porém, a linha de comando não funciona desta forma, pois para escrever o caractere “espaço”, ao definir um caminho ou arquivo, deve-se utilizar “barra invertida” + “espaço” (“\ ”). Por causa desta diferença o FFmpeg não reconhecia as entradas e saída corretamente, apontando este erro. Por esta razão foi criada a função “verif_esp”, a qual resolve este problema, substituindo o caractere espaço da QString por seu correspondente na linha de comando.
- Adicionar a aba “Múltiplos Programas”: em versões anteriores o ajuste dos “maps” eram realizados automaticamente. A ordem dos programas estava relacionada diretamente com as entradas (entrada 1 ligava ao programa 1 e assim consecutivamente), deixando para o usuário modificar a caixa de diálogo “Comando” se optasse por outras possibilidades.
- Modificar a localização do parâmetro “Muxrate”: inicialmente este parâmetro estava na aba “Avançados”, devido sua importância decidiu-se por inseri-lo em um lugar de maior destaque.

Além disso, foram feitos testes de tempo para gerar um TS. Comparou-se o tempo de montagem e modificação do comando ao utilizar o terminal e um documento de texto com comandos pré-prontos ou a GUI_FFmpeg. Os resultados são mostrados na Tabela 4. As tarefas para os testes são: gerar um novo comando, modificar uma das entradas, adicionar uma entrada e um novo programa utilizando os *streams* desta entrada.

Tabela 4 – Testes de tempos GUI_FFmpeg.

Tipo de teste	Tentativa 1	Tentativa 2	Tentativa 3	Média
Gerar um novo comando (GUI_FFmpeg)	01:06	01:09	00:56	01:03
Gerar um novo comando (manual)	06:55	09:38*	7:37	08:03
Modificar entrada (GUI_FFmpeg)	00:08	00:05	00:04	00:06
Modificar entrada (manual)	00:18	00:12	00:12	00:14
Gerar uma nova entrada e um novo programa (GUI_FFmpeg)	00:11	00:07	00:07	00:08

Gerar uma nova entrada e um novo programa (manual)	01:34	00:45	00:56	01:05
*Houve erro de digitação				

Se a GUI_FFmpeg for utilizada o tempo reduzirá consideravelmente, além de eliminar o número de erros de digitação nos comandos executados. Apesar disso, é recomendável salvar alguns comandos importantes, pois salva trabalhos e etapas relevantes dos testes.

5. Conclusões e Sugestões para Trabalhos Futuros

Por ser um trabalho destinado à equipe do LaPSI, esta que tem conhecimento em programação, atentou-se muito em demonstrar como foram desenvolvidas as soluções e quais os objetivos destas. Certamente estas soluções serão utilizadas pela equipe, pois estão diretamente ligadas com as rotinas presentes no laboratório.

A implementação do perfil 2 no FFmpeg modificou a maneira de geração de TS's com múltiplos programas. O parâmetro responsável por informar o número de programas tornou-se obrigatório e os nomes de cada programa (*service_name*) são diferentes entre si. Observaram-se, pelos *softwares* utilizados nos testes, que realmente existem mais de dois programas com *streams* diferentes inseridos no TS gerado, possibilitando até a visualização destes. Portanto, o cabeçalho do TS foi alterado para contemplar o perfil 2 de múltiplos programas.

Por possibilitar uma maior agilidade e facilidade, a GUI_FFmpeg, tornou-se uma ferramenta quase obrigatória ao gerar TS's pelo FFmpeg. Possibilitou uma redução de tempo considerável e uma eliminação de erros de digitação no comando enviado ao FFmpeg. Por ser amplamente testada e discutida, a solução passou por muitas fases de aprimoramento atingindo uma qualidade além do esperado inicialmente.

Sugere-se a solução dos problemas apresentados na transmissão do TS de múltiplos programas gerado pelo FFmpeg. Além disso, a adição de novos *inputs* no FFmpeg para criação das tabelas EIT tornariam a compatibilidade com o SBTVD mais próxima de acontecer.

Como a GUI_FFmpeg é a primeira versão de uma interface que tenta otimizar ao máximo o tempo de teste do *set-top box*, cada detalhe que facilite a utilização é considerado. Porém estas melhorias só são identificadas à medida que o programa é testado e utilizado por completo. Sendo assim, possíveis melhorias foram notadas: união da GUI_FFmpeg com o programa ControleTuner, facilitaria e uniria no mesmo programa a recepção, transformação e transmissão do TS; possibilitar ao usuário modificar os *streams* individualmente utilizando o GUI_FFmpeg; mudar a lógica da parte de exclusão de entradas para excluir somente a entrada desejada, em qualquer caso; adicionar a possibilidade do usuário identificar com mais facilidade o *bitrate* dos *streams* componentes da entrada; adicionar a possibilidade de o usuário analisar cada entrada com o FFmpeg; alterar a lógica de verificação ao conectar uma entrada em um programa na aba "múltiplos programas", no momento é verificado se existem dois *streams* não importando-se qual o tipo dele; adicionar mais possibilidades de *maps*; criar uma caixa de seleção para impossibilitar a modificação da caixa de diálogo "Comando" ao alterar uma configuração; por fim, implementar a lógica inversa, ao invés do programa escrever o comando, possibilitar que o comando configure a GUI_FFmpeg.

6. Referências

ANDGULO, J.; CALAZADA, J.; ESTRUCH, A. SELECTION OF STANDARDS FOR DIGITAL TELEVISION: THE BATTLE FOR LATIN AMERICA. **PROCEDIA COMPUTER SCIENCE**, v. 14, p. 301-310, 2012. ISSN 1877-0509. Disponível em: < <http://www.sciencedirect.com/science/article/pii/S187705091200796X>> . Acesso em: 13 Jun. 2016.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **ABNT NBR 15603-1: ABNT NBR 15603-1 -Televisão digital terrestre — Multiplexação e serviços de informação (SI) - Serviços de informação do sistema de radiodifusão**. Rio de Janeiro, 2007.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **ABNT NBR 15603-2: ABNT NBR 15603-2 - Televisão digital terrestre – Multiplexação e serviços de informação(SI) sintaxes e definições da informação básica de SI**. Rio de Janeiro, 2007.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **ABNT NBR 15603-2: ABNT NBR 15603-3 - Televisão digital terrestre — Multiplexação e serviços de informação (SI) — Parte 3: Sintaxe e definição de informação estendida do SI**. Rio de Janeiro, 2007.

ASSOCIAÇÃO DE RADIODIFUSORES, FABRICANTES, DESENVOLVEDORES E ENTIDADES DE ENSINO E PESQUISA EM FAVOR DA TV DIGITAL ABERTA. **FÓRUM DO SISTEMA BRASILEIRO DE TV DIGITAL TERRESTRE**. Disponível em: <http://forumsbtvd.org.br/>. Acesso em: 12 Jun. 2016.

Bitzki, L. J.; Kleemann, R. A. **Desenvolvimento e adaptação de tecnologias livres para solução de streaming HLS na UFRGS**. 2016. Disponível em: <<http://hdl.handle.net/10183/142236>>. Acesso em: 11 Jun. 2016

BRASIL. **DECRETO Nº 5.820, DE 29 DE JUNHO DE 2006**. Disponível em: <http://www.planalto.gov.br/ccivil_03/_Ato2004-2006/2006/Decreto/D5824.htm>. Acesso em: 06 Jun. 2016.

BRASIL. **DECRETO Nº 4.901, DE 26 DE NOVEMBRO DE 2003**. Disponível em: <<http://forumsbtvd.org.br/theoffice/wp-content/uploads/2011/07/decreto-tv-digital.pdf>>. Acesso em: 05 Jun. 2016.

Code::Blocks. **Code::Blocks - The IDE with all the features you need, having a consistent look, feel and operation across platforms**. Disponível em: <http://www.codeblocks.org/>. Acessado em: 01 Jun. 2016

Code::Blocks. **WxSmith tutorials**. Disponível em: http://wiki.codeblocks.org/index.php/WxSmith_tutorials. Acessado em: 01 Jun. 2016.

DekTec. **StreamXpress**. Disponível em: <http://www.dektec.com/products/applications/StreamXpress/>. Acesso em: 13 Jun. 2016.

ENDRES, L. P. **Development of an MPEG2 multiplexer compliant with SBTVD digital TV standard**. 2014. Disponível em: <<http://hdl.handle.net/10183/105054>>. Acessado em: 31 Mai. 2016.

FFMPEG. **ABOUT FFMPEG**. Disponível em: < <https://ffmpeg.org/about.html>>. Acesso em: 31 Mai. 16

Ffmpeg. **Ffmpeg License and Legal Considerations**. Disponível em: <https://Ffmpeg.org/legal.html>. Acesso em: 31 Mai. 2016.

FREE SOFTWARE FOUNDATION, **GNU General Public License, versão 2**. 1991. Disponível em: <<http://www.gnu.org/licenses/old-licenses/gpl-2.0.html>>. Acesso em: 31 Mai. 2016.

FREE SOFTWARE FOUNDATION, **GNU Lesser General Public License, versão 2.1**. 1999. Disponível em: <<http://www.gnu.org/licenses/old-licenses/lgpl-2.1.html>>. Acesso em: 31 Mai. 2016.

IBGE. **ACESSO À INTERNET E À TELEVISÃO E POSSE DE TELEFONE MÓVEL CELULAR PARA USO PESSOAL 2013**. Disponível em: <<http://ibge.gov.br/home/estatistica/populacao/acessoainternet2013/default.shtm>>. Acesso em: 16 Jun. 16

INTERNATIONAL STANDARDS ORGANIZATION / INTERNATIONAL ELECTROTECHNICAL COMMISSION. **ISO/IEC 13818-1 INTERNATIONAL STANDARD: INFORMATION TECHNOLOGY — GENERIC CODING OF MOVING PICTURES AND ASSOCIATED AUDIO INFORMATION: SYSTEMS**. 2000.

Marques, G. A. G. **The SBTVD parser (SBTVD_parser)**. Disponível em: <http://sbtvdparser.sourceforge.net/>. Acessado em: 18 Mai. 2016.

Qt Company. **QtCreator**. Disponível em: <https://www.qt.io/>. Acesso em: 01 Jun. 2016

QURESHI, M. A.; GOPAL, M.M.; SADIQ, M. **DESIGN AND IMPLEMENTATION OF AUDIO/VIDEO CODEC BASED ON ANDROID PLATFORM**, 2013. Disponível em: <http://interscience.in/IJCTT_Vol4Iss2/10-15.pdf>. Acesso em: 10 Jun. 2016

SCHEEREN, V. **MÓDULO PARA INSPEÇÃO AUTOMÁTICA DE LINHAS DE TRANSMISSÃO POR TERMOGRAFIA**. 2011. Disponível em: <http://hdl.handle.net/10183/33089>. Acesso em: 10 Jun. 2016.

VideoLan organization. **VLC Media Player**. Disponível em: <http://www.videolan.org/vlc/>. Acesso em: 15 Jun. 2016.

Apêndice

Help da GUI_FFmpeg

Abaixo segue documentação referente ao *tutorial* da GUI_FFmpeg. Este tutorial também se encontra na pasta em que o programa foi disponibilizado para amplo acesso. O link é o que segue: https://github.com/andregobo/GUI_FFmpeg.

Help da GUI_FFmpeg

Versão: 1.0

Autor: André Luís Bacarin Gobo
LaPSI - DELET - UFRGS
Dezembro de 2012
Porto Alegre, 07 de Junho de 2016

1. Introdução

Este guia tem como objetivo descrever o funcionamento da GUI_FFmpeg. Esta GUI foi criada para facilitar e agilizar a criação de *Transport Stream's* (TS's) com múltiplos canais. Este guia é baseado na plataforma Ubuntu. Assim outras plataformas podem apresentar códigos e funções diferentes dos utilizados. A GUI_FFmpeg foi um dos objetivos do Projeto de Diplomação de André Luís Bacarin Gobo.

2. Instalação do FFmpeg

Para instalar o FFmpeg acesse: https://github.com/andregobo/TCC_ffmpeg. Dentro da pasta criada há o arquivo "INSTALL", abra-o e siga as suas orientações. Pode ser necessário instalar as bibliotecas yasm. Digite no terminal o comando abaixo para instalar esta biblioteca.

```
sudo apt-get install yasm
```

3. Instalação do QtCreator

Para instalar o QtCreator , IDE utilizada pela GUI_FFmpeg, digite no terminal os seguintes comandos:

```
sudo apt-get install build-essential  
sudo apt-get install g++  
sudo apt-get install qtcreator
```

4. Guia de utilização do GUI_FFmpeg

Para baixar a GUI_FFmpeg acesse: https://github.com/andregobo/GUI_FFmpeg. Utilize-a no QtCreator. Abaixo constam explicações relacionadas a cada comando ou função da GUI.

O funcionamento do FFmpeg se dá por comandos no terminal. Por esta razão foi criado uma GUI (*Graphical User Interface*) para tornar mais fácil e ágil a sua utilização.

O *design* final da GUI_FFmpeg é ilustrada na Figura 32. Pode-se dividir a GUI pelos seguintes comandos: executar o FFmpeg, abrir tutorial, mostrar comando a ser executado, organização das janelas, escolher entradas, definir saída, escolha dos *streams* a serem utilizados, manipular os *streams* para cada programa, duração do TS, copiar vídeo e copiar áudio, *muxrate* e funções do FFmpeg relacionados diretamente com o SBTVD.

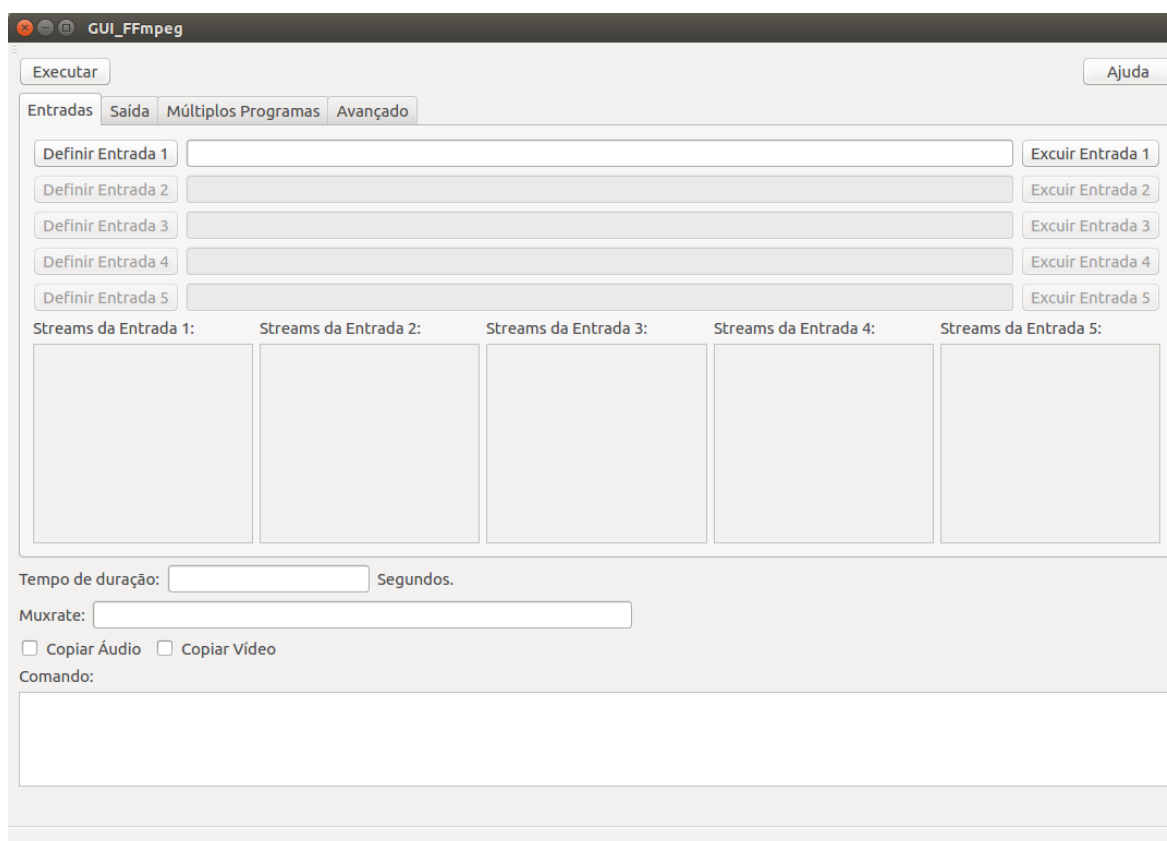


Figura 32 – Design da GUI_FFmpeg.

Cada um dos comandos criados são definidos e especificados abaixo.

- Executar o FFmpeg: este comando está por trás do botão "Executar", ilustrado na Figura 33. Deve ser clicado quando todas as configurações já estão prontas. Ou seja, quando selecionado abre um terminal em outra janela e executa o comando montado pelo usuário. O terminal permanece aberto após a conclusão da execução. Se isso não ocorrer modifique os parâmetros do termina em "Edit/Profile Preferences" selecione a barra "Command" e modifique a opção

“When command exits:” para “Hold the terminal open”. O comando executado é o mostrado na caixa de diálogo “Comando” elucidado mais a frente no texto.

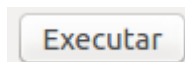


Figura 33 – Botão Rodar.

- Abrir tutorial: este comando está por trás do botão "Ajuda", ilustrado na Figura 34. Este botão abre este pdf.

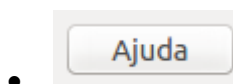


Figura 34 – Botão Ajuda.

- Mostrar comando a ser executado: optou-se por utilizar esta função para que os usuários mais avançados conseguissem adicionar mais parâmetros de *input* além dos quais a GUI contempla. Portanto, esta caixa de diálogo, ilustrada na Figura 35, contém o comando executado quando o botão "Executar" é clicado. Como o programa faz diretamente um *link* com esta caixa de diálogo e o que é executado, todas as alterações feitas pelo usuário são levadas em conta.



Figura 35 – Caixa de diálogo “Comando”.

- Organização das janelas: para simplificar e organizar a GUI foi utilizado uma divisão de abas, para alguns grupos de funções. Estes grupos são esclarecidos mais adiante no texto. Logo, foram criados quatro grupos de funções com nomes bastante explicativos, são eles: “Entradas”, “Saída”, “Múltiplos Programas” e “Avançado”. Estas abas são ilustradas na Figura 36.

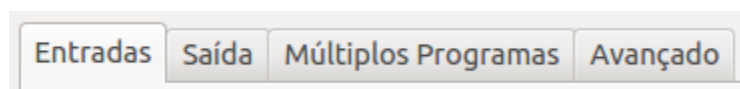


Figura 36 – Abas separando comandos.

- Escolher entradas: esta função tem como objetivo facilitar a inserção de entradas. Este comando pode ser observado na Figura 37. Esta parte da GUI tem o seguinte funcionamento: quando o usuário clicar em “Definir Entrada 1” abrirá uma janela de diálogo para selecionar um arquivo de entrada, esta janela é ilustrada na Figura 38. O mesmo acontece para as entradas 2, 3 e 4, no intuito de evitar, cada entrada só pode ser ativada se o usuário já definiu a(s) entrada(s) antecedente(s). Ou seja, a entrada 2 só pode ser definida se a entrada 1 já foi definida, e assim consecutivamente. Além disso, há uma linha de edição das entradas, cuja finalidade é informar o usuário qual caminho e qual arquivo foi selecionado e possibilitar a inserção manual do caminho/arquivo. Também há botões de exclusão das entradas. Estes servem para excluir a entrada referente sem precisar apagar todo o texto da caixa de diálogo e modificar as outras entradas, por

exemplo, após clicar em “Excluir Entrada 1” o programa excluirá a entrada 1 e todas as entradas com maior número.

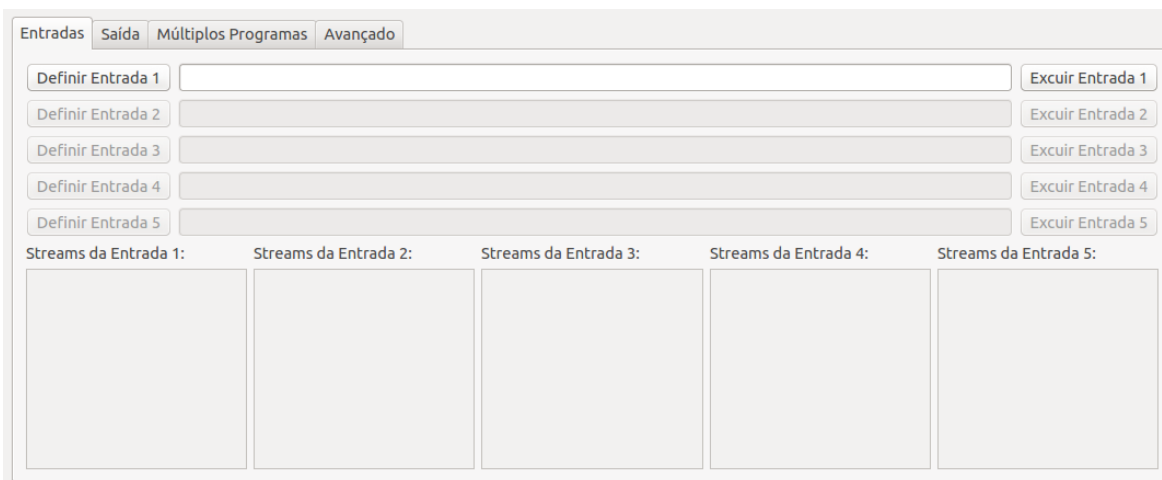


Figura 37 – Aba de entradas.

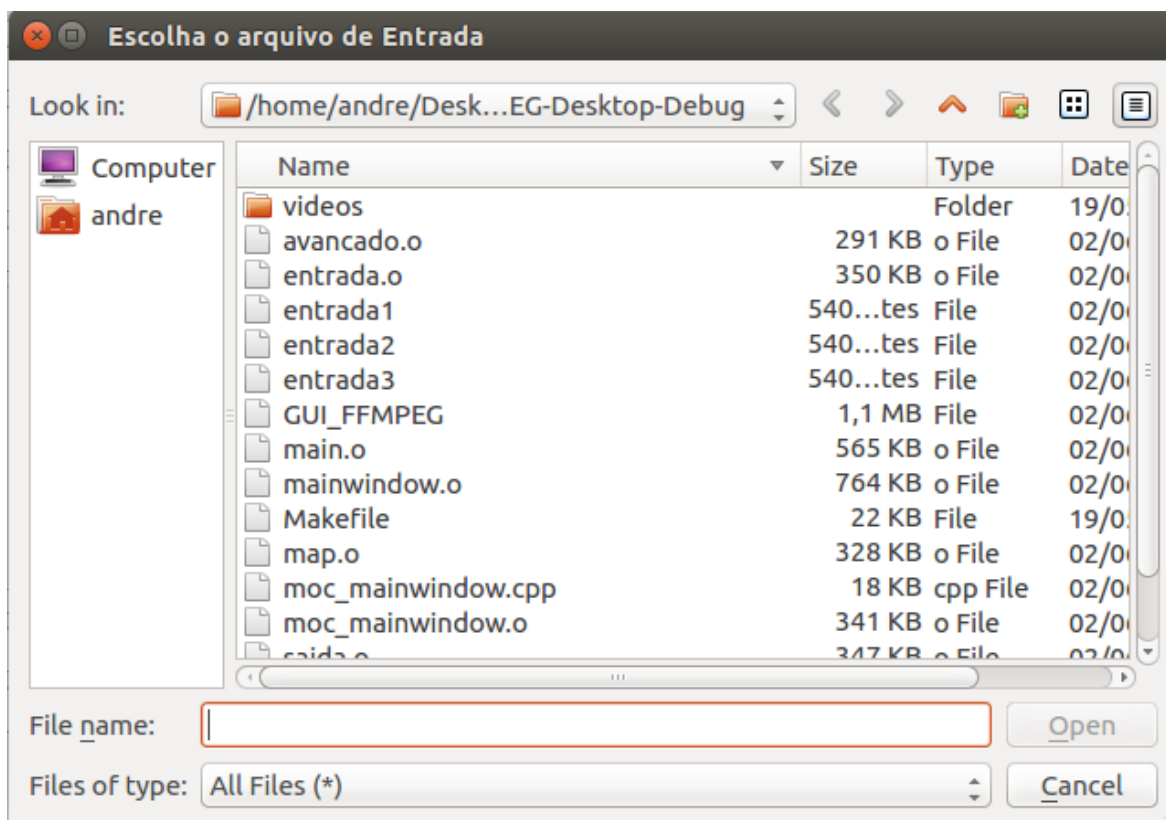


Figura 38 – Caixa de diálogo para selecionar uma entrada.

- Escolha dos *streams* a serem utilizados: funciona em conjunto com as funções de escolha de entradas. A área gráfica da função pode ser observada embaixo da área de inserção de entradas na Figura 39. Ao selecionar um arquivo de entrada, a GUI identifica os *streams* que fazem parte do arquivo, identifica seu número, seu tipo e sua compressão. Ao serem identificadas estas informações são escritas ao lado de checkboxes, cuja localização é dentro do retângulo referente à entrada, como se pode observar na Figura 39. Ao marcar estes

checkboxes, o usuário define quais *streams* serão utilizados na execução do comando.

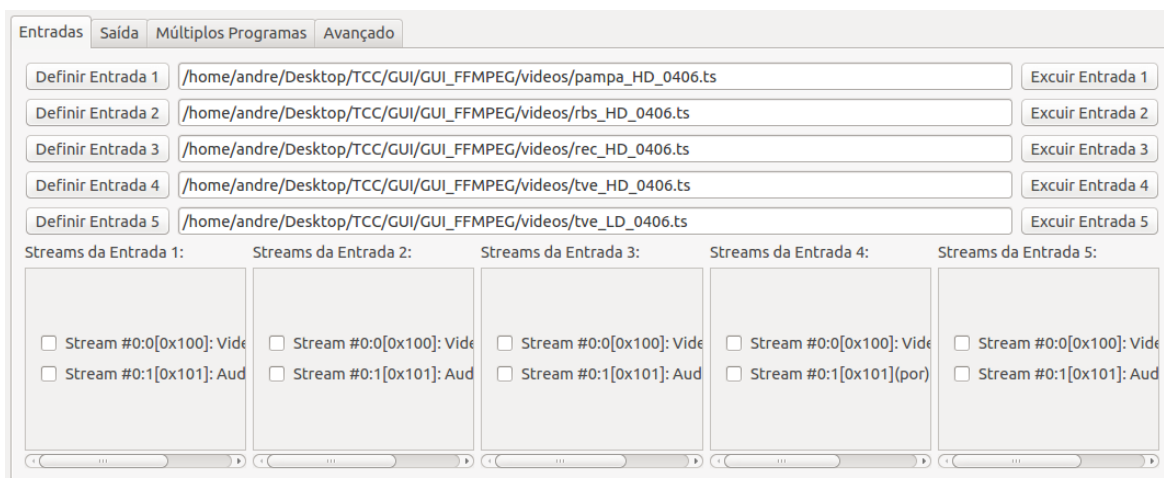


Figura 39 – Escolha dos *Streams*.

- Definir saída: nesta aba o usuário define qual a pasta e nome do arquivo gerado pelo FFmpeg, é ilustrada pela Figura 40. Para facilitar esta definição, ao clicar em “Definir pasta de saída” uma caixa de diálogo se abrirá para que a pasta de destino seja selecionada, esta janela é ilustrada na Figura 41. Após selecionar a pasta de destino o caminho selecionado será informado na linha de edição à direita do botão “Definir pasta de saída”. Esta linha de edição também pode ser editada manualmente. Além disso, o usuário define o nome da saída e sua extensão na caixa de diálogo ao lado de “Nome do arquivo de saída”. A extensão não foi fixada em “.ts” para não limitar totalmente o programa, porém se o usuário não escolher extensão alguma ela será automaticamente “.ts”.

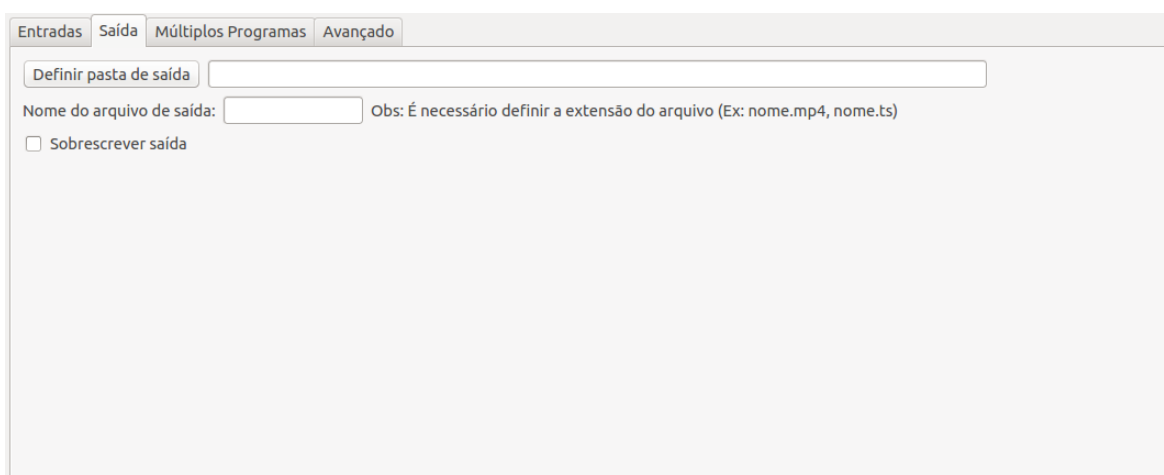


Figura 40 – Aba de saída.

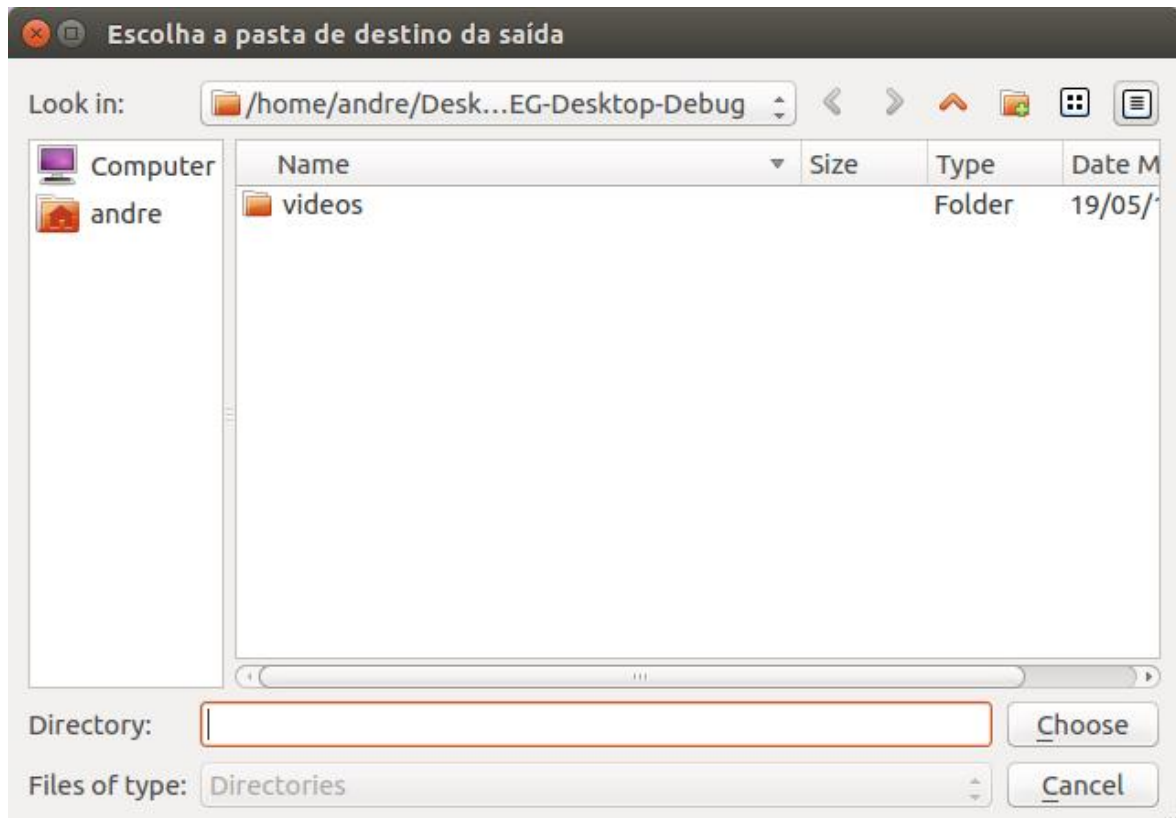


Figura 41 – Caixa de diálogo para selecionar a pasta de destino da saída.

- Manipular os *streams* para cada programa: esta aba tem a função de ligar os *streams* da entrada com o programa desejado, levando em conta que o usuário quer utilizar a opção de múltiplos programas. Por estar diretamente relacionado com as entradas as suas opções também são dependentes delas. À medida que o usuário adiciona entradas mais opções de escolhas são ativas. Além disso, para não causar erros e confusão, os programas de maior número só são ativados se os programas de menor ordem foram definidos. Na Figura 42 e na Figura 43 observam-se os dois casos extremos. No primeiro caso nenhuma entrada foi selecionada. No segundo caso todas as entradas foram selecionadas e a partir delas será criado um TS com cinco programas, cada um utilizando uma entrada diferente. Além disso, há o botão “Utilizar múltiplos programas”, o qual, se marcado como vazio, faz um *reset* na aba “Múltiplos Programas”, voltando todos os programas para sua forma inicial (“Desativado”). Para lembrar o usuário que cada programa suporta somente dois *streams*, um aviso é mostrado quando há um desacordo. O aviso pode ser visto na Figura 44.

Entradas Saída Múltiplos Programas Avançado

Utilizar múltiplos programas

Programa 1 utilizar streams de: Desativado Entrada 1 Entrada 2 Entrada 3 Entrada 4 Entrada 5

Programa 2 utilizar streams de: Desativado Entrada 1 Entrada 2 Entrada 3 Entrada 4 Entrada 5

Programa 3 utilizar streams de: Desativado Entrada 1 Entrada 2 Entrada 3 Entrada 4 Entrada 5

Programa 4 utilizar streams de: Desativado Entrada 1 Entrada 2 Entrada 3 Entrada 4 Entrada 5

Programa 5 utilizar streams de: Desativado Entrada 1 Entrada 2 Entrada 3 Entrada 4 Entrada 5

Figura 42 – Aba Múltiplos Programas.

Entradas Saída Múltiplos Programas Avançado

Utilizar múltiplos programas

Programa 1 utilizar streams de: Desativado Entrada 1 Entrada 2 Entrada 3 Entrada 4 Entrada 5

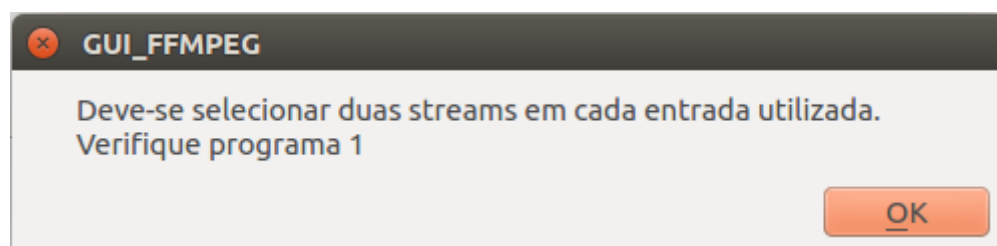
Programa 2 utilizar streams de: Desativado Entrada 1 Entrada 2 Entrada 3 Entrada 4 Entrada 5

Programa 3 utilizar streams de: Desativado Entrada 1 Entrada 2 Entrada 3 Entrada 4 Entrada 5

Programa 4 utilizar streams de: Desativado Entrada 1 Entrada 2 Entrada 3 Entrada 4 Entrada 5

Programa 5 utilizar streams de: Desativado Entrada 1 Entrada 2 Entrada 3 Entrada 4 Entrada 5

Figura 43 – Aba Múltiplos Programas com seleções (As 5 entradas estão definidas).

Figura 44 – Erro quando não há dois *streams* selecionados na entrada.

- Duração do TS: o FFmpeg possibilita ao usuário diminuir o tempo da multimídia. Como o objetivo é testar o *hardware* desenvolvido com diferentes tipos de configurações um tempo mais curto para o TS torna-se atrativo. A Figura 45 ilustra a área responsável pela configuração.

Tempo de duração: Segundos.

Figura 45 – Determinar duração.

- Copiar vídeo e copiar áudio: quando selecionadas, informam ao FFmpeg que o vídeo e o áudio devem ser copiados diretamente, sem passar por um codec (codificador ou decodificador). Esta opção é ilustrada na Figura 46.

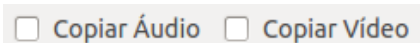


Figura 46 – Selecionar utilização de codecs.

- *Muxrate*: parâmetro de grande importância para o correto funcionamento, pois fixa a taxa de *bytes (bitrate)* do TS. Sem este parâmetro o TS de saída será formado com um *bitrate* variável, cuja compatibilidade com o MPEG2 não é satisfeita. Portanto este parâmetro deve ser sempre especificado. O valor deste parâmetro pode ser calculado analisando os *streams* de *input* pelo próprio FFmpeg. Após obter a informação de cada *stream* utilizado, o valor do *muxrate* deve ser maior que a soma de todos os *bitrates* observados. Se o valor for menor que o devido o vídeo do TS de saída apresentará erros no *framerate*, o vídeo e áudio serão apresentados como se estivessem em câmera lenta. Ao contrário, se o *muxrate* for definido com um valor muito maior que a soma dos *bitrates*, o arquivo gerado apresentará muitos pacotes nulos e terá um tamanho de arquivo maior que o necessário. Portanto, para não ocorrer erros de sub ou sobre dimensionamento esse parâmetro deve ser definido com o valor da soma dos *bitrates* dos *streams* utilizados mais 11%. Este acréscimo de 11% se deve a uma margem de erro. A parte de inserção do valor é ilustrada na Figura 47.



Figura 47 – Configuração do *Muxrate*.

- Funções do FFmpeg relacionados diretamente com o SBTVD: estas funções são relacionadas diretamente com a compatibilização do FFmpeg com o SBTVD. Portanto cada uma delas é explicada a seguir: “Area code” – campo com 12 bits que corresponde ao código de área especificado pelos órgãos responsáveis; “Guard interval” – campo com 2 bits que deve indicar o intervalo de guarda de guarda utilizado, “00” para intervalo de guarda igual a 1/32, “01” para intervalo de guarda igual a 1/16, “10” para intervalo de guarda igual a 1/8, “11” para intervalo de guarda igual a 1/4; “Transmission mode” – campo com 2 bits que indica o modo de informação, “00” para modo 1, “01” para modo 2, “10” para modo 3, “11” não definido; “Physical channel” – número do canal físico ao qual o TS está associado; “Virtual channel” – número do canal virtual ao qual o TS está associado; por fim, “Original_network_id” – campo de 16 bits que serve como etiqueta para especificar o identificador do sistema de distribuição original. Os valores de default observados ao lado de cada caixa de texto são exemplos de números. Ao clicar nos botões “Ajuda” ao lado de cada parâmetro, uma janela com as mesmas informações deste tutorial irá aparecer informando sua utilidade. Esta função é ilustrada na Figura 48.

The screenshot displays the 'Avançado' (Advanced) tab of a software interface. It contains six rows of configuration options, each with a text input field, a default value, and a 'Ajuda' (Help) button:

Parameter	Default Value
Area code	2970
Guard interval	2
Transmission mode	3
Physical channel	20
Virtual channel	1
Original network ID	0730

Figura 48 – Aba avançado.

Anexo

Países que adotaram o ISDB-TB

Segundo o secretário-executivo do Ministério das Comunicações, Francisco Ibiapina, a tecnologia ISDB-T chega hoje a 630 milhões de telespectadores. Os seguintes países adotaram o padrão ISDB-TB de TV digital: Argentina, Brasil, Bolívia, Botswana, Chile, Costa Rica, Equador, Guatemala, Honduras, Paraguai, Peru, Uruguai, Venezuela, Suriname e Filipinas Maldivas e Sri Lanka. Todos eles passam pelo processo de transição digital e todos podem adquirir equipamentos das empresas brasileiras (FÓRUM DO SISTEMA BRASILEIRO DE TV DIGITAL TERRESTRE - <http://forumsbtvd.org.br/>, acesso em 12/06/16).