



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL
Construindo uma Engenharia Decolonial para a Soberania Digital e Popular
29 a 31 de outubro de 2025
Campinas - SP, Brasil

Recaracterização de aparelhos de TV Box para uso como minicomputadores – Procedimentos para investigação e formatação

Hildo Guillard Junior, Universidade Estadual Paulista, h.guillard@unesp.br

RELATO DE EXPERIÊNCIA TÉCNICA
EIXO TEMÁTICO: TECNOLOGIA SOCIAL E INOVAÇÃO SOCIAL

RESUMO

Este estudo explora a transformação de *TV Boxes* em minicomputadores por meio da instalação da distribuição Linux Armbian, servindo como um tutorial para diferentes modelos de dispositivos, abordando de forma prática e técnica o processo. Tais dispositivos, comumente usados para acesso ilegal a conteúdos audiovisuais, podem ser aproveitados para finalidades sociais e de pesquisa em eletrônica embarcada, oferecendo alternativa a produtos comerciais. O processo de descaracterização se inicia com a inspeção do *hardware* e identificação de suas características, seguida da escolha de uma distribuição Linux compatível com o processador, no caso ARM 64 bits, e da extração da *Device Tree Blob* (DTB) com os parâmetros para configuração do sistema. Após a instalação e conexão dos periféricos, a *TV Box* se torna um computador funcional, promovendo a redução de lixo eletrônico e ampliando o acesso a tecnologias em ambientes educacionais. Ao final, são fornecidos *scripts* para automatização do projeto.

PALAVRAS-CHAVE: ARM; Linux; Receita Cidadã; Sistemas Eletrônicos Embarcados; *TV Box*.



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL
Construindo uma Engenharia Decolonial para a Soberania Digital e Popular
29 a 31 de outubro de 2025
Campinas - SP, Brasil

CONTEXTO

O volume de apreensões de aparelhos ilegais do tipo *TV Boxes* no Brasil (AGÊNCIA PARÁ, 2024; GLOBO, 2023; MELO, 2023; RECEITA FEDERAL, 2023) é preocupante, pois estes permitem acesso ilegal a conteúdos protegidos por direitos autorais, prejudicando a indústria e a arrecadação (O GLOBO, 2021). Muitos ainda possuem instalados *malwares* para roubo de dados e ataques cibernéticos (DIGITAL; SÉRVIO, 2024; MELO, 2024; TUDOCELULAR.COM, 2023), prejudicando diretamente o usuário.

Tais dispositivos possuem características de *hardware* compatíveis com produtos comerciais tipicamente empregados em produtos de computação, desenvolvimento e/ou pesquisa em eletrônica embarcada (Tabela 1).

Figura 1. Descrição do aparelho de TV Box modelo BTV E10.

Dispositivo (modelo)	Características da CPU			Características de memória RAM	
	Modelo e arquitetura	Número de núcleos	Frequência de clock (GHz)	Tecnologia usada	Capacidade (GB)
TV Box Youit TX2	ARM Cortex-A7MP (32 bits)	4	1,2	LPDDR3	2
TV Box BTV E10	ARM Cortex-A53 (64 bits)	4	1,9	LPDDR4	2
Raspberry Pi 3B+	ARM Cortex-A53 (64 bits)	4	1,4	LPDDR2	1
Raspberry Pi 4B+	ARM Cortex-A72 (64 bits)	4	1,5	LPDDR4	2

Fonte: LUZ, G. P. C. P.; SATO, G.M.; GONZALEZ, L. F. G.; BORIN, J. F (2025)

Transformar esses aparelhos em minicomputadores para órgãos públicos é uma forma de reaproveitamento do recurso e conscientização (O TEMPO, 2024; RECEITA FEDERAL, 2023; UNIVERSIDADE FEDERAL DO CEARÁ, 2023). Nesta visão, o projeto Receita Cidadã “utiliza o processo de destinação de mercadoria (anteriormente aprendida) para contribuir com a gestão ambiental, ecoeficiência e responsabilidade social com ações benéficas para toda a sociedade” (RECEITA CIDADÃ, [S.d.]). Seu principal objetivo é a criação de uma rede colaborativa com pesquisadores de instituições públicas para conversão de itens apreendidos em objetos úteis para a sociedade, no caso das *TV Boxes*, minicomputadores para escolas.



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL
Construindo uma Engenharia Decolonial para a Soberania Digital e Popular
29 a 31 de outubro de 2025
Campinas - SP, Brasil

DESCRIÇÃO DA EXPERIÊNCIA

Para estudo da descaracterização dos equipamentos do tipo TV Box foi utilizado o modelo BTV E10 (Figura 1) cedido pela Receita Federal do Brasil.

Figura 1. Descrição do aparelho de TV Box modelo BTV E10.

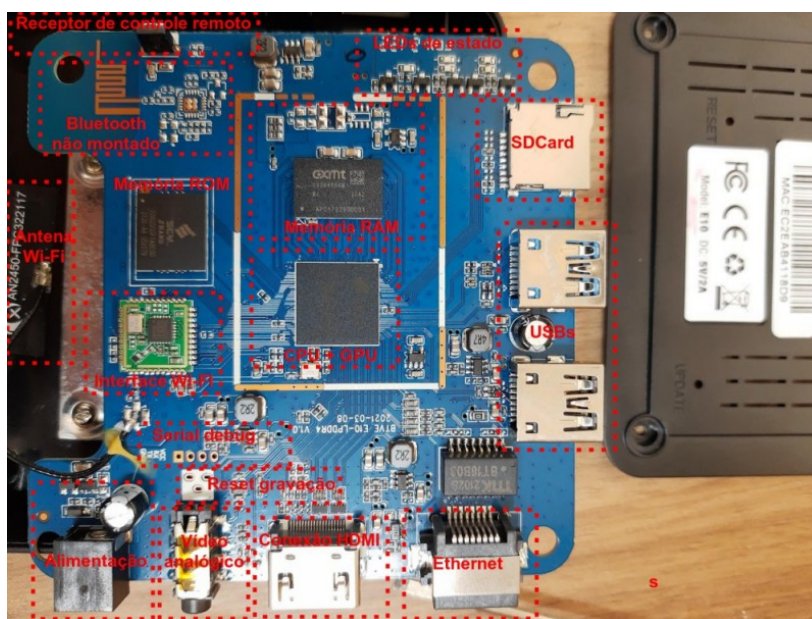


Características comerciais nominais

Tensão de alimentação: 5V através de fonte
Resolução: Full HD 1080p
Processador CPU: Amlogic S9005x2 ARM Cortex-A53 Quad Core 2.0GHz
Processador gráfico (GPU): Mali-G31 MP2
Codec de Vídeo: H.264, H.265 e VP9
Memória ROM (para programas): 8GB
Memória RAM: 2GB
Sistema Operacional: Baseado em Android
Conexão a Internet: Wireless e LAN (Porta Ethernet)
Interfaces: 2 USBs, HDMI, Ethernet, Wi-Fi, AV, Slot Micro SD
Fonte: autoria própria.

O hardware interno ao dispositivo (Figura 2) possui todas as capacidades de um computador em versão reduzida: LEDs indicativos de operação, interfaces de comunicação USB (permitindo teclado e mouse), Ethernet, Wi-Fi e Bluetooth (neste caso não montada), memória ROM (para instalação de *softwares*) e RAM, unidades de processamento geral e de vídeo (CPU e GPU) e, saídas de áudio e vídeo.

Figura 2. Visão do hardware interno da TV Box modelo BTV E10 e identificação de seus periféricos.



Fonte: autoria própria.



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL
Construindo uma Engenharia Decolonial para a Soberania Digital e Popular
29 a 31 de outubro de 2025
Campinas - SP, Brasil

Esta etapa de inspeção é fundamental para validar as informações comerciais dos equipamentos do tipo *TV Box*, visto que eles possuem variantes e versões modificadas por substituição de componentes ou falsificação entre os próprios modelos.

A informação mais relevante da inspeção do *hardware* é a identificação da interface serial, normalmente no padrão de protocolo RS232 de tensão TTL 3,3V, para *debug* (depuração). Esta conexão permite o envio de comandos ao *kernel* Linux do Android previamente instalado e a checagem do *log* de inicialização (Figura 3).

Figura 3. Informações obtidas através do *log* serial de inicialização do aparelho.



```
CPU clk: 1200 MHz
Set clk81 to 166.6M
eMMC boot @ 0
DDR driver_vesion: LPDDR4_PHY_V_0_1_11
LPDDR4 probe
ddr clk to 984MHz
DRAM: 2 GiB
register usb cfg[0][1] = 0000000077f3f6d0
NAND: get_sys_clk_rate_mtd() 270, clock
setting 200!
vpp: g12a/b osd1 matrix rgb2yuv
.....
[OSD]load fb addr from dts:/meson-fb
## Booting Android Image at 0x01080000 ...
Kernel command line:
androidboot.dtbo_idx=0 --cmdline
root=/dev/mmcblk0p18
buildvariant=userdebug
Apply dtbo 0
Starting kernel ...
uboot time: 5951772 us
```

Fonte: autoria própria.

A extração de parâmetros adicionais deste *log* permite a correta configuração do sistema que se pretende instalar, como velocidades de comunicação (*clocks*), tipo da memória de armazenamento (FLASH, NAND, ...) e seu tamanho, bem como a extração do arquivo *Devicetree Blob* (DTB) utilizado. O arquivo DTB possui, de forma compilada a partir de um arquivo texto *Devicetree Overlays* (LINUX KERNEL, [S.d.]), todas as informações de pinagem e interfaces conectadas ao *chip* principal: mapeamento de endereços, barramentos e velocidades de comunicação para cada um dos periféricos. Sua função é a de prover as configurações, em um processo análogo de inicialização e configuração de pinos/memória realizado ao se utilizar microcontroladores e, sem suas



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL
Construindo uma Engenharia Decolonial para a Soberania Digital e Popular
29 a 31 de outubro de 2025
Campinas - SP, Brasil

corretas definições, o dispositivo formatado pode não “bootar” ou inicializar sem o funcionamento de uma interface (vídeo, Wi-Fi, Ethernet, USBs, ...).

A extração da DTB pode ser realizada por diferentes maneiras:

1. Realizando um *dump* do arquivo carregado para a memória, por exemplo, pelo comando `cat /sys/firmware/fdt > dtb_loaded.dtb;`
2. Copiando o setor de *boot* do Android, ou outro sistema, para uma imagem em *flash-drive* através do comando `dd if=/dev/block/boot of=/sdcard/boot.img`, estando atento que “if” aponte para o nome do *device* de inicialização do dispositivo (partição de boot) e “of” para local que se deseja salvar a imagem. A extração do arquivo `.dtb`, a partir da imagem salva, deve ser realizada procurando-se o trecho correspondente ao arquivo através da interpretação de seu cabeçalho (DEVICETREE SPECIFICATION, 2017). Porém, esta interpretação varia com a versão do arquivo `.dtb`, atualmente na versão 17, mas corriqueiramente encontrado em versões anteriores nos aparelhos de TV Box.

A manipulação do arquivo `.dts` (*Devicetree Source*), descompilado do arquivo `.dtb` através do comando Linux `dtc -I dtb -O dts -f file.dtb file.dts`, e sua posterior recompilação pelo comando reverso (`dtc -I dts -O dtb -f file.dts file.dtb`), pode habilitar ou desabilitar funcionalidades do *hardware* utilizado e expô-las (deixá-las acessíveis) dentro da *tree* (árvore de diretórios) do sistema operacional Linux instalado. Tais comandos são disponibilizados em um Linux através de `da` instalação: `sudo apt install device-tree-compiler`.

Descaracterização do equipamento TV Box

Para formatação é necessário seguir os seguintes passos:



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL
Construindo uma Engenharia Decolonial para a Soberania Digital e Popular
29 a 31 de outubro de 2025
Campinas - SP, Brasil

1. Identifica-se uma distribuição Linux compatível com o *hardware* da TV Box. Sendo um dispositivo ARM (*Advanced RISC Machine*, em que RISC significa *Reduced Instruction Set Computer*) de 64 bits, optou-se pela distribuição Armbian que possui um suporte da comunidade a versões compatíveis com tais dispositivos em <https://github.com/armbian/community>;
2. Realiza-se o *download* da versão mais atual. Podendo optar-se pela versão mínima (sem interface gráfica, apenas modo texto ou *prompt* de comandos) para se testar a capacidade da TV Box e compatibilidades, realizando a instalação da interface usuário e aplicativos *a posteriori*. Tal escolha permite a criação de um dispositivo otimizado para uso em processamento científico, ao não se gastar processamento com a interface gráfica. A imagem escolhida foi a [Armbian community 24.11.0-trunk.273 Aml-s9xx-box noble current 6.6.56 minimal.img.xz](#) que, no dia da escrita deste texto, correspondia a versão mais nova da imagem Armbian sem interface gráfica (apenas modo texto) mantida pela comunidade GNU para dispositivos baseados na CPU Amlogic modelos S9xx;
3. Cria-se de um *flash-drive* “bootável” através do *software* Balena Etcher ou, equivalentemente, se executa o comando Linux `dd if=Armbian_community_24.11.0-trunk.202_Aml-s9xx-box_noble_current_6.6.53_minimal.img of=/dev/sdb`, sendo `/dev/sdb` o caminho Linux para o *device* correspondente ao pendrive;
4. Modifica-se os parâmetros de iniciação da imagem no *flash-drive*:
 - a) Trocam-se os arquivos [aml_autoscript](#) (arquivo geral de *boot*), [s905_autoscript](#) (especificação para *boot* por *flash-drive*) e [emmc_autoscript](#) (especificação para *boot* na memória interna da TV Box) pelos disponíveis em <https://github.com/devmfc/amlogic-bootscripts-Armbian>. Este procedimento se faz necessário devido características específicas e endereçamento das memórias utilizadas no modelo de TV Box;



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL
Construindo uma Engenharia Decolonial para a Soberania Digital e Popular
29 a 31 de outubro de 2025
Campinas - SP, Brasil

- b) Adiciona-se o arquivo DTB extraído anteriormente. Em caso de dificuldade com o processo, a comunidade GNU disponibiliza configurações DTB compatíveis com diferentes versões de hardware, neste caso BTV E10, na internet ou com a imagem a ser instalada (na pasta `/dtbs/meson-g12a-sei510.dtb` da partição de *boot*). Lembrando que um arquivo DTB errado impossibilita a inicialização do sistema ou uso de algum recurso;

Figura 4. Configuração necessária do arquivo de *boot* `armbianEnv.txt`.

```
extraargs=earlycon rootflags=data=writeback rw
no_console_suspend consoleblank=0 fsck.fix=yes fsck.repair=yes
net.ifnames=0
bootlogo=false
verbosity=7
usbstoragequirks=0x2537:0x1066:u,0x2537:0x1068:u
console=both

# DTB file for this TV box
fdtfile=amlogic/meson-g12a-sei510.dtb
# set this to the UUID of the root partition (value can be found
# in /extlinux/extlinux.conf after APPEND root= or with blkid)
rootdev=UUID=92139c84-3871-41d7-a3f2-e8a943cbfa87

# Enable ONLY for gxbb (S905) / gx1 (S905X/L/W) to create fake
u-boot header
soc_fixup=gx1-

APPEND root=LABEL=ROOT_EMMC rootflags=data=writeback rw
console=ttyAML0,115200n8
console=tty0 no_console_suspend consoleblank=0 fsck.fix=yes
fsck.repair=yes net.ifnames=0
```

Fonte: autoria própria.

- c) Modificam-se as partes destacadas do arquivo `armbianEnv.txt` (Figura 4) que são utilizadas para carregamento das informações iniciais do sistema. Caso tal arquivo não exista, deve-se criá-lo na raiz da partição *boot*. Especial atenção deve ser dado aos valores de:
- `fdtfile=`, que deve apontar para o arquivo DTB com a correta configuração do dispositivo;



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL
Construindo uma Engenharia Decolonial para a Soberania Digital e Popular
29 a 31 de outubro de 2025
Campinas - SP, Brasil

II. `rootdev=UUID=`, que deve apontar para a partição do *flash-drive* em que o sistema Linux (partição `root`) se encontra. Para obter este valor, pode ser utilizado o *software* Disks presente na maioria das distribuições Linux ou através do utilitário de partições do Windows, em acessórios;

5. Formatação da TV Box:

- a) Inicializa-se a TV Box pela imagem no *flash-drive* segurando-se o botão *update* da TV Box, muitas vezes presente ao interior do conector AV (áudio-vídeo);
- b) Executa-se o comando `./install-aml.sh` para realizar a instalação (cópia dos arquivos do Linux no *flash-drive*) para a memória interna da TV Box;
- c) Crie-se o usuário e senha de administrador (superusuário) do sistema, conforme pedido durante a instalação;
- d) Refaz-se o procedimento 4.c.ii para que o arquivo `armbianEnv.txt` internamente gravado na TV Box. O novo valor de `rootdev` de apontar para a memória interna onde está instalado o sistema. Para isso:
 - I. Utiliza-se o comando `blkid` para encontrar a UUID correspondente a partição de `root` do sistema, agora instalado;
 - II. Executa-se `nano /mnt/armbiEnv.txt` para editar o arquivo e finalize com `<Ctrl>+x`.
- e) Reinicializa-se o minicomputador e remove-se o *flash-drive*;
- f) Instalam-se os pacotes gráficos necessários para a aplicação pelo comando `sudo apt xfce4 xfce4-goodies lightdm`. Sendo que `LightDM` corresponde a interface de gerenciamento de *displays*/monitores e `Xfce` ao pacote do ambiente *desktop* mais leve disponível para o Armbian.



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL
Construindo uma Engenharia Decolonial para a Soberania Digital e Popular
29 a 31 de outubro de 2025
Campinas - SP, Brasil

Alternativamente, pode-se escolher uma imagem com o ambiente gráfico Xfce instalado, ao invés da imagem mínima (apenas com *prompt* de comandos) selecionada na etapa 2;

- g) Instalam-se os demais *softwares* necessários, por exemplo navegador *web* e pacote *office*, através do comando `sudo add-apt-repository ppa:mozillateam/ppa && sudo apt libreoffice firefox-esr`. Neste momento outros *software* específicos, como pacotes GNU educacionais, traduções e dicionários, podem ser instalados;
- h) Realiza-se a reinicialização final do sistema para uso.

Scripts e automatização dos procedimentos de instalação

Para simplificação e reprodução do processo de criação da imagem e instalação do Armbian, foram criados *scripts* em linguagem Bash (para serem executados em Linux). Tais códigos, usados na sequência descrita abaixo, estão disponíveis no Apêndice A, enquanto versões atualizadas poderão ser encontradas em <https://gitlab.com/hildogir/tvbox>.

1. `bash bootdisk-download.sh` para *download* da imagem mais recente compatível para o *hardware* da TV Box. Durante sua execução foram usadas as opções de *device*/processador `Aml-s9xx-box`, imagem base `bookworm` (codinome do [Debian mais recente](#)) e ambiente/*desktop* `current*_xfce_desktop`. Este *script* substitui a etapa 2 do procedimento manual;
2. `bash bootdisk-create.sh` cria uma imagem de instalação a partir das baixadas. Este *script* substitui a etapa 3 do procedimento manual e, adicionalmente, executa `bootdisk-config.sh`, para copiar arquivos adicionais e modificar alguns valores como na etapa 4 do procedimento manual, além de configurar o teclado para ABNT2 (*layout* usado no Brasil);



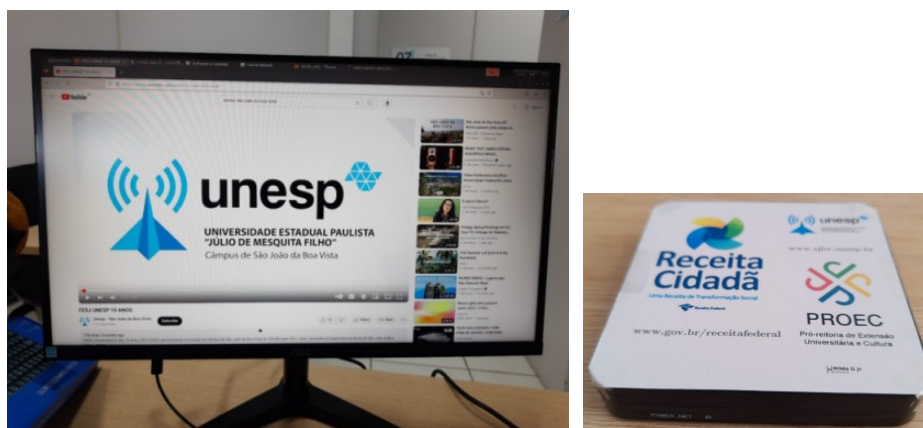
XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL
Construindo uma Engenharia Decolonial para a Soberania Digital e Popular
29 a 31 de outubro de 2025
Campinas - SP, Brasil

3. `sudo bash install-armbian.sh` instala o Armbian na memória interna da TV Box, realizando toda a configuração necessária para seu correto processo de *boot*, além se preocupar em manter o *MAC address* original da TV Box durante a instalação, para o proposito de rastreabilidade. Este é o arquivo mais importante e necessita ser executada como super usuário pois, a imagem portátil (criada em 2) pode já possuir seus pacotes instalados (procedimento 5) e, este *script* realiza o trabalho de cópia do Sistema Operacional para dentro de cada TV Box, automatizando toda a etapa 5 do procedimento manual;
4. `bash install-packages.sh` instala a lista de pacotes de programas pré-selecionados nos arquivos `*.list`, automatizando a etapa 5-g do procedimento manual. Para descaracterização de muitos aparelhos, esta etapa pode ser realizada antes do passo 3, criando uma imagem com os *softwares* instalados no *flash-drive*, para que seja copiada para a memória interna da TV Box.

RESULTADOS

Com a formatação e conexão de periféricos (teclado, *mouse* e monitor de vídeo), um aparelho de TV Box se torna um microcomputador (Figura 6).

Figura 6. TV Box funcionando como minicomputador para navegação na internet (esquerda) e aspecto final com descaracterização visual da “caixinha” (direita).



Fonte: autoria própria.



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL
Construindo uma Engenharia Decolonial para a Soberania Digital e Popular
29 a 31 de outubro de 2025
Campinas - SP, Brasil

A conexão de periféricos é opcional, e o aparelho mostra potencial de uso para sistemas embarcados para coleta de dados, armazenamento e processamento matemático em aplicações científicas, diminuindo o custo de pesquisas para aquisição de equipamentos comerciais de *hardware* equivalente.

A Receita Federal do Brasil, através do programa Receita Cidadã, tem mobilizado instituições e pesquisadores (Figura 7) para modificação de tais equipamentos e destinação a escolas que, inicialmente não possuem laboratórios de informática e/ou não são atualizados há tempo (O TEMPO, 2024; RECEITA FEDERAL, 2024; REDE FEDERAL, 2022; UNIVERSIDADE FEDERAL DO CEARÁ, 2023).

Figura 7. Evento do projeto “Bem-te-vi” de doação de TV Boxes recaracterizadas ocorrido na sede da Receita Federal de Sorocaba-SP no dia 04/12/2023, com a presença do autor.



Fonte: Receita Federal do Brasil (2023).

Tal iniciativa tem contribuído em: redução de lixo eletrônico, redução de custos com incineração/destruição de tais equipamentos, propagação da cultura de *software* livre dentro do ambiente educacional (BONILLA, 2014), podendo se transformar em resultados governamentais de médio e longo prazo (NASCIMENTO, MARTÍNEZ, 2025).



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL
Construindo uma Engenharia Decolonial para a Soberania Digital e Popular
29 a 31 de outubro de 2025
Campinas - SP, Brasil

CONCLUSÕES

O *hardware* das *TV Box*, devidamente descaracterizado, se torna uma plataforma interessante para uso como minicomputadores e aplicações científicas de baixo custo, podendo ser utilizada no lugar de dispositivos conhecidos como Raspberry Pi e outros.

Dada a capacidade de execução de um sistema Linux em arquitetura ARM, seu potencial de uso apenas se limita pela sua capacidade de processamento e bibliotecas/*softwares*, sendo disponíveis todos os programas tipicamente utilizados em um computador: interfaces de usuários, gerenciadores de arquivos, navegadores *web*, pacote *office* e outros. *Softwares* adicionais da iniciativa GNU podem ser compilados para sua arquitetura a partir do código fonte.

O reuso de tais equipamentos, além de criar uma economia circular com diminuição de lixo eletrônico, colabora para a conscientização da população à respeito do uso de tais equipamentos e, indiretamente, sobre questões sobre o uso de *software* livre/proprietário por ambientes governamentais/educacionais.

AGRADECIMENTOS

O autor agradece a parceria com a Receita Federal e a oportunidade de colaborar com o projeto Bem-te-vi no programa Receita Cidadã, propagando sua experiência técnica em Linux e *hardware* embarcado em prol de uma causa social.

REFERÊNCIAS

AGÊNCIA PARÁ. Sefa apreende 1.560 aparelhos de tv box em Dom Eliseu no sudeste paraense. 31 mar. 2024. Disponível em: <<https://www.agenciapara.com.br/noticia/52864/sefa-apreende-1.560-aparelhos-de-tv-box-em-dom-eliseu-no-sudeste-paraense>>. Acesso em: 3 out. 2024.

BONILLA, M.H.S., 2014. **Software Livre e Educação: uma relação em construção.** *Perspectiva*, 32(1), pp. 205-234. DOI: <https://doi.org/10.5007/2175-795X.2014v32n1p205>.



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL
Construindo uma Engenharia Decolonial para a Soberania Digital e Popular
29 a 31 de outubro de 2025
Campinas - SP, Brasil

DIGITAL, O.; SÉRVIO, G. **Malware transforma TV Box em 'zumbi'; veja como se proteger.** Olhar Digital, 26 jan. 2024. Disponível em: <<https://olhardigital.com.br/2024/01/26/seguranca/malware-transforma-tv-box-em-zumbi-veja-como-se-proteger/>>. Acesso em: 3 out. 2024.

Flattened Devicetree (DTB) Format. Disponível em: <<https://devicetree-specification.readthedocs.io/en/stable/flattened-format.html>>. Acesso em: 17 out. 2024.

GLOBO. **Em ação contra pirataria, Anatel retira de circulação mais de 1,4 milhão de aparelhos de "gatonet".** 9 fev. 2023. Disponível em: <<https://g1.globo.com/economia/noticia/2023/02/09/em-acao-contr-pirataria-anatel-retira-de-circulacao-mais-de-14-milhao-de-tv-box-ilegais.ghtml>>. Acesso em: 3 out. 2024.

LUZ, G. P. C. P.; SATO, G.M.; GONZALEZ, L. F. G.; BORIN, J. F. **Repurposing of TV boxes for a circular economy in smart cities applications.** Scientific Reports. Volume 15. 2025. <https://doi.org/10.1038/s41598-025-97379-4>.

MELO, C. **Malware atinge 1,3 milhão de TV Boxes Android; Brasil é o mais afetado.** Disponível em: <<https://www.mundoconectado.com.br/seguranca/malware-atinge-13-milhao-de-tv-boxes-android-brasil-e-o-mais-afetado/>>. Acesso em: 3 out. 2024.

NASCIMENTO, E. F., MARTÍNEZ, M. P., 2025. **Levantamento da produção científica sobre o uso do software livre na governança de tecnologia da informação no Governo Federal.** OBSERVATÓRIO DE LA ECONOMÍA LATINOAMERICANA, 23(3), pp.e9291-e9291. DOI: <https://doi.org/10.55905/oelv23n3-079>.

_____. **Operações da Anatel resultam na apreensão de 1,4 milhão de TV Box pirata.** Disponível em: <<https://www.mundoconectado.com.br/corporativo/operacoes-da-anatel-resultam-na-apreensao-de-14-milhao-de-tv-box-pirata/>>. Acesso em: 3 out. 2024.

O GLOBO. **Governo deixa de arrecadar R\$ 2 bi por ano com transmissões de TV pirata, estima Associação Brasileira de Televisão.** Disponível em: <<https://oglobo.globo.com/economia/governo-deixa-de-arrecadar-2-bi-por-ano-com-transmissoes-de-tv-pirata-estima-associacao-brasileira-de-televisao-25031158>>. Acesso em: 3 out. 2024.



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL
Construindo uma Engenharia Decolonial para a Soberania Digital e Popular
29 a 31 de outubro de 2025
Campinas - SP, Brasil

O TEMPO. **Universidade de MG transforma TV box apreendidas em microcomputadores para escolas públicas.** 26 jun. 2024. Disponível em: <<https://www.otempo.com.br/brasil/2024/6/26/universidade-de-mg-transforma-tv-box-apreendidas-em-microcomputa>>. Acesso em: 3 out. 2024.

Open Firmware and Devicetree — The Linux Kernel documentation. Disponível em: <<https://docs.kernel.org/devicetree/index.html#devicetree-overlays>>. Acesso em: 17 out. 2024.

Projeto Receita Cidadã. Disponível em: <<https://www.gov.br/receitafederal/pt-br/aceso-a-informacao/acoes-e-programas/cidadania-fiscal/novos-destinos/novos-destinos>>. Acesso em: 3 out. 2024.

RECEITA FEDERAL. **Aparelhos de Tv Box apreendidos pela Receita Federal se transformam em 350 minicomputadores.** 24 ago. 2023. Disponível em: <<https://www.gov.br/receitafederal/pt-br/assuntos/noticias/2023/agosto/receita-federal-ufla-e-cefet-de-varginha-destinam-tv-box-transformadas-em-minicomputadores-para-prefeituras-da-regiao>>. Acesso em: 3 out. 2024.

_____. **Receita Federal apreende 158 TV Box em operação no Centro de Distribuição dos Correios em Belo Horizonte.** 4 set. 2023. Disponível em: <<https://www.gov.br/receitafederal/pt-br/assuntos/noticias/2023/setembro/receita-federal-apreende-158-tv-box-em-operacao-no-centro-de-distribuicao-dos-correios-em-belo-horizonte>>. Acesso em: 3 out. 2024.

_____. **Receita Federal e Unifal destinam TV Box transformadas em Minicomputadores às prefeituras da região de Alfenas.** 21 set. 2023. Disponível em: <<https://www.gov.br/receitafederal/pt-br/assuntos/noticias/2023/setembro/receita-federal-e-unifal-destinam-tv-box-transformadas-em-minicomputadores-as-prefeituras-da-regiao-de-alfenas>>. Acesso em: 3 out. 2024.

_____. **Receita Federal em Sorocaba doa 1.500 minicomputadores a 30 municípios.** 13 dez. 2023. Disponível em: <<https://www.gov.br/receitafederal/pt-br/assuntos/noticias/2023/dezembro/receita-federal-em-sorocaba-doa-1-500-minicomputadores-a-30-municipios>>. Acesso em: 4 set. 2025.

REDE FEDERAL. **Projeto de Extensão do IFPA transforma TV Box em minicomputadores para escolas.** 15 dez. 2022. Disponível em:



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL
Construindo uma Engenharia Decolonial para a Soberania Digital e Popular
29 a 31 de outubro de 2025
Campinas - SP, Brasil

<<https://portal.redefederal.org.br/noticias/noticias/geral/projeto-de-extensao-do-ifpa-transforma-tv-box-em-minicomputadores-para-escolas>>. Acesso em: 3 out. 2024.

TELE.SÍNTESE. **Apps maliciosos povoam TV Boxes, alerta empresa de segurança.** 26 já. 2024. Disponível em: <<https://telesintese.com.br/apps-maliciosos-povoam-tv-boxes-alerta-empresa-de-seguranca/>>. Acesso em: 3 out. 2024.

TUDOCELULAR.COM. **TV Box com Android passa por análise e malware de espionagem é encontrado.** 13 jan. 2023. Disponível em: <<https://www.tudocelular.com/seguranca/noticias/n201005/tv-box-com-android-possui-malware-pre-instalado.html>>. Acesso em: 3 out. 2024.

UNIVERSIDADE FEDERAL DO CEARÁ. **Projeto transforma receptores de TV Box piratas em minicomputadores.** 15 jun. 2023. Disponível em: <<https://www.ufc.br/noticias/noticias-ufctv/17881-projeto-transforma-receptores-de-tv-box-piratas-em-minicomputadores>>. Acesso em: 3 out. 2024.

APÊNDICE A – ARQUIVOS DE SCRIPTS

Scripts para automatização do processo de instalação e configuração do Armbian para TV Boxes usados durante o desenvolvimento deste trabalho.

Script bootdisk-download.sh

```
#!/bin/bash
# Download available Armbian community released image.
# Written by Hildo Guillard Junior.
# Written on 25/July/2025.
set -e

API_URL="https://api.github.com/repos/armbian/community/releases/latest"
echo "Fetching latest Armbian Community release info..."
RELEASE_JSON=$(curl -s "$API_URL")

# Extract file names.
mapfile -t FILE_NAMES <<(echo "$RELEASE_JSON" | grep '"name":' | sed -E 's/.*"name": "(.*)",/\1/')
mapfile -t FILE_URLS <<(echo "$RELEASE_JSON" | grep '"browser_download_url":' | sed -E 's/.*"browser_download_url": "(.*)"/\1/')
mapfile -t FILE_SIZES <<(echo "$RELEASE_JSON" | grep '"size":' | sed -E 's/.*"size": ([0-9]+),?/\1/')

# Extract release tag.
RELEASE_TAG=$(echo "$RELEASE_JSON" | grep '"tag_name":' | head -1 | sed -E 's/.*"tag_name": "(.*)",/\1/')
```




XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL

Construindo uma Engenharia Decolonial para a Soberania Digital e Popular

29 a 31 de outubro de 2025

Campinas - SP, Brasil

```
# Parse .img and .img.xz files, ignore .asc/.sha/etc.
declare -a VALID_FILES VALID_URLS VALID_SIZES
declare -A VERSIONS DEVICES DEVELOPMENTS KERNELS ENVIRONMENTS FILE_INDEXES
declare -A ENV_TO_KERNEL ENV_TO_ENVIRONMENT

for i in "${!FILE_NAMES[@]}; do
    NAME="${FILE_NAMES[$i]}"
    URL="${FILE_URLS[$i]}"
    SIZE="${FILE_SIZES[$i]}"
    if [[ "$NAME" =~ \.img$ || "$NAME" =~ \.img\.xz$ ]]; then
        if [[ ! "$NAME" =~ \.(asc|sig|sha|sha256|sha512|sha256sum)$ ]]; then
            VALID_FILES+=("$NAME")
            VALID_URLS+=("$URL")
            VALID_SIZES+=("$SIZE")

            REAL_INDEX=$(( ${#VALID_FILES[@]} - 1 ))
            BASENAME=$(echo "$NAME" | sed -E 's/\.img(\.xz)?$//')
            PARTS=${BASENAME#Armbian_community_}
            IFS='_' read -r -a arr <<< "$PARTS"

            VERSION="${arr[0]}"
            DEVICE="${arr[1]}"
            DEVELOPMENT="${arr[2]}"
            KERNEL="${arr[3]}"

            if (( ${#arr[@]} > 4 )); then
                ENVIRONMENT="${arr[4]}"
                for ((j=5; j<${#arr[@]}; j++)); do
                    ENVIRONMENT+="_${arr[j]}"
                done
            else
                ENVIRONMENT=""
            fi

            ENV_KEY="${KERNEL}_${ENVIRONMENT}"
            ENV_TO_KERNEL["$ENV_KEY"]="$KERNEL"
            ENV_TO_ENVIRONMENT["$ENV_KEY"]="$ENVIRONMENT"

            KEY="${VERSION}|${DEVICE}|${DEVELOPMENT}|${KERNEL}|${ENVIRONMENT}"
            FILE_INDEXES["$KEY"]=$REAL_INDEX

            VERSIONS["$VERSION"]=1
            DEVICES["$DEVICE"]=1
            DEVELOPMENTS["$DEVELOPMENT"]=1
            ENVIRONMENTS["$ENV_KEY"]=1
        fi
    fi
done

prompt_choice() {
    local -n options=$1
    local label="$2"
    local allow_all="$3"
    local values=("${!options[@]}")
    IFS=$'\n' sorted=($(sort <<< "${values[*]}"))
    unset IFS

    if (( ${#sorted[@]} == 1 )); then
        echo "Only one $label option: ${sorted[0]}"
        REPLY="${sorted[0]}"
    else
        echo
```



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL

Construindo uma Engenharia Decolonial para a Soberania Digital e Popular

29 a 31 de outubro de 2025

Campinas - SP, Brasil

```
echo "Available $label options:"
if [[ "$allow_all" == "true" ]]; then
    echo " 0) ALL"
fi

NUM_ITEMS=${#sorted[@]}
if [ "$label" = "environment (kernel + environment)" ]; then
    for i in "${!sorted[@]"; do
        ENV_KEY="${sorted[$i]}"
        for key in "${!FILE_INDEXES[@]"; do
            IFS='|' read -r ver dev devp ker env <<< "$key"
            if [[ "${ker}_${env}" == "$ENV_KEY" ]]; then
                index="${FILE_INDEXES[$key]}"
                size_bytes=${VALID_SIZES[$index]}
                size_mb=$(( (size_bytes + 512000) / 1048576 ))
                printf "%2d) %5s MB    %s\n" $((i+1)) "$size_mb"
                "$ENV_KEY"
                break
            fi
        done
    done
else
    if (( NUM_ITEMS < 15 )); then
        for i in "${!sorted[@]"; do
            printf "%2d) %s\n" $((i + 1)) "${sorted[$i]}"
        done
    else
        COLS=3
        ROWS=$(( (NUM_ITEMS + COLS - 1) / COLS ))
        for ((r=0; r<ROWS; r++)); do
            line=""
            for ((c=0; c<COLS; c++)); do
                idx=$(( c*ROWS + r ))
                if (( idx < NUM_ITEMS )); then
                    entry=$(printf "%2d) %-25s" $((idx+1)) "${sorted[$idx]}")
                    line+="$entry"
                fi
            done
            echo "$line"
        done
    fi
fi

read -p "Choose a $label: " selection
if [[ "$allow_all" == "true" && "$selection" == "0" ]]; then
    REPLY="ALL"
    return
fi
if ! [[ "$selection" =~ ^[0-9]+$ ]] || (( selection < 1 || selection >
NUM_ITEMS )); then
    echo "Invalid selection. Exiting."
    exit 1
fi
REPLY="${sorted[$((selection - 1))]}"
fi
}

build_filtered_options() {
    local filter_version="$1"
    local filter_device="$2"
    local filter_development="$3"
    local -n out_versions=$4
```



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL

Construindo uma Engenharia Decolonial para a Soberania Digital e Popular

29 a 31 de outubro de 2025

Campinas - SP, Brasil

```
local -n out_devices=$5
local -n out_developments=$6
local -n out_envs=$7

out_versions=()
out_devices=()
out_developments=()
out_envs=()

for key in "${!FILE_INDEXES[@]}; do
    IFS='|' read -r ver dev devp ker env <<< "$key"

    [[ -n "$filter_version" && "$ver" != "$filter_version" ]] && continue
    [[ -n "$filter_device" && "$dev" != "$filter_device" && "$dev" !=
"$filter_device" ]] && continue
    [[ -n "$filter_development" && "$devp" != "$filter_development" ]] &&
continue

    out_versions["$ver"]=1
    out_devices["$dev"]=1
    out_developments["$devp"]=1
    out_envs["${ker}_${env}"]=1
done
}

prompt_choice VERSIONS "version"
SELECTED_VERSION="$REPLY"

declare -A FILTERED_DEVICES
build_filtered_options "$SELECTED_VERSION" "" "" VERSIONS FILTERED_DEVICES
DEVELOPMENTS ENVIRONMENTS
prompt_choice FILTERED_DEVICES "device" true; SELECTED_DEVICE="$REPLY"

declare -A FILTERED_DEVELOPMENTS
build_filtered_options "$SELECTED_VERSION" "$SELECTED_DEVICE" "" VERSIONS
FILTERED_DEVICES FILTERED_DEVELOPMENTS ENVIRONMENTS
prompt_choice FILTERED_DEVELOPMENTS "development";
SELECTED_DEVELOPMENT="$REPLY"

declare -A FILTERED_ENVIRONMENTS
build_filtered_options "$SELECTED_VERSION" "$SELECTED_DEVICE"
"$SELECTED_DEVELOPMENT" VERSIONS FILTERED_DEVICES FILTERED_DEVELOPMENTS
FILTERED_ENVIRONMENTS
prompt_choice FILTERED_ENVIRONMENTS "environment (kernel + environment)";
SELECTED_ENV_KEY="$REPLY"

SELECTED_KERNEL="${ENV_TO_KERNEL[$SELECTED_ENV_KEY]}"
SELECTED_ENVIRONMENT="${ENV_TO_ENVIRONMENT[$SELECTED_ENV_KEY]}"

FINAL_KEY="${SELECTED_VERSION}|${SELECTED_DEVICE}|${SELECTED_DEVELOPMENT}|$
{SELECTED_KERNEL}|${SELECTED_ENVIRONMENT}"
FINAL_INDEX="${FILE_INDEXES[$FINAL_KEY]}"

if [[ -z "$FINAL_INDEX" ]]; then
    echo
    echo "No matching file found for your selection."
    exit 1
fi

FILENAME="${VALID_FILES[$FINAL_INDEX]}"
#FILEURL="${VALID_URLS[$FINAL_INDEX]}"
FILEURL="https://github.com/armbian/community/releases/download/$RELEASE_TAG/
$FILENAME"
```



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL

Construindo uma Engenharia Decolonial para a Soberania Digital e Popular

29 a 31 de outubro de 2025

Campinas - SP, Brasil

```
echo
echo "Selected image: $FILENAME from $FILEURL"
read -p "Download this file? [y/N] " confirm
if [[ "$confirm" =~ ^[Yy]$ ]]; then
    if command -v curl >/dev/null 2>&1; then
        curl -L -o "$HOME/Downloads/$FILENAME" "$FILEURL"
    elif command -v wget >/dev/null 2>&1; then
        wget -O "$HOME/Downloads/$FILENAME" "$FILEURL"
    else
        echo "Error: Neither curl nor wget is installed."
        exit 1
    fi
    echo "Download complete."
else
    echo "Download cancelled."
fi
```

Script bootdisk-create.sh

```
#!/bin/bash
# Create a boot disk for Armbian from images available into the Downloads
# folder,
# also configure the boot for a TVBox.
#
# Written by Hildo Guillard Junior.
# Written on 13-march-2025 due several images to test in a BTV-E10 model.

#####
# Define the source folders.
SCRIPT_DIR="$(cd "$(dirname "${BASH_SOURCE[0]}")" && pwd)"
PATH_IMGS="$HOME/Downloads"

#####
# Write the image.

# Find image files (.img and .img.xz) and ask to recovery it to a flash drive.
IMAGES=$(find "$PATH_IMGS" -maxdepth 1 -type f \( -name "*.img" -o -name
"*.img.xz" \) | sort)
if [ ${#IMAGES[@]} -eq 0 ]; then
    echo "❌ No .img or .img.xz files found in $PATH_IMGS"
    exit 1
fi
# Display the list and ask the user to choose.
if [ ${#IMAGES[@]} -eq 1 ]; then
    # Auto-select the only available image
    CHOICE=0
    SELECTED_IMAGE="${IMAGES[0]}"
else
    # Print the list if there are multiple files.
    echo -e "\n📁 Available image files:"
    printf "%-5s %-10s %s\n" "Index" "Size" "Filename"
    echo "-----"
    for i in "${!IMAGES[@]}; do
        FILE_SIZE=$(du -h "${IMAGES[$i]}" | awk '{print $1}')
        RELATIVE_PATH=$(realpath --relative-to="$PATH_IMGS" "${IMAGES[$i]}")
        printf "[%2d] %-5s %s\n" "$i" "$FILE_SIZE" "$RELATIVE_PATH"
    done
```



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL
Construindo uma Engenharia Decolonial para a Soberania Digital e Popular
29 a 31 de outubro de 2025
Campinas - SP, Brasil

```
echo "-----"
# Ask for user input.
read -p "🔍 Enter the number of the image you want to recover: " CHOICE
if ! [[ "$CHOICE" =~ ^[0-9]+$ ]] || [ "$CHOICE" -lt 0 ] || [ "$CHOICE" -ge
${#IMAGES[@]} ]; then
    echo "❌ Invalid choice."
    exit 1
fi
SELECTED_IMAGE="${IMAGES[$CHOICE]}"
fi
echo "📌 Selected image: $SELECTED_IMAGE"

# List available flash drives.
echo "📁 Available flash drives:"
lsblk -d -n -o NAME,SIZE,MODEL | grep -E 'sd[a-z]\mmcblk[1-9]'
read -p "🔍 ⚠️ ⚠️ ⚠️ Enter the target device (e.g., /dev/sdc): "
TARGET_DEVICE
if [[ ! -b "$TARGET_DEVICE" ]]; then
    echo "❌ Invalid device."
    exit 1
fi

# Confirm before proceeding.
read -p "🔥 WARNING: This will erase all data on $TARGET_DEVICE. Continue?
(y/N): " CONFIRM
if [[ "$CONFIRM" != "y" && "$CONFIRM" != "Y" ]]; then
    echo "❌ Operation canceled."
    exit 1
fi
echo "🔄 Writing image..."
if [[ "$SELECTED_IMAGE" == *.xz ]]; then
    xzcat "$SELECTED_IMAGE" | sudo dd of="$TARGET_DEVICE" bs=4M
status=progress
else
    sudo dd if="$SELECTED_IMAGE" of="$TARGET_DEVICE" bs=4M status=progress
fi
sync
echo "✅ Image writing completed."
sleep 4
#sudo partprobe

#bash "$SCRIPT_DIR/bootdisk-config.sh" "$TARGET_DEVICE"
```

Script bootdisk-config.sh

```
#!/bin/bash
# Config the boot parameters of a Armbian image.
#
# Written by Hildo Guillard Junior.
# Written on 13-march-2025 due several images to test in a BTV-E10 model.

#####
# Define the source folders.
SCRIPT_DIR="$(cd "$(dirname "${BASH_SOURCE[0]}")" && pwd)"
PATH_BOOT_FILES="$(dirname "$SCRIPT_DIR")/Boot files"

#####
# Configure the boot.
```



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL
Construindo uma Engenharia Decolonial para a Soberania Digital e Popular
29 a 31 de outubro de 2025
Campinas - SP, Brasil

```
echo "🔧 Configuring boot..."

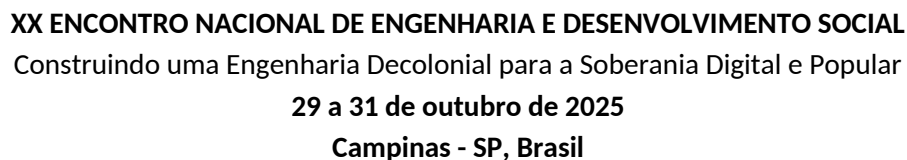
# Get all devices with partitions labeled 'armbi_boot' or 'armbi_root'
if [[ $# -eq 0 ]]; then
    DEVICES=$(lsblk -rno NAME,LABEL | awk '$2=="armbi_boot" || $2=="armbi_root" {print $1}' | sed 's/[0-9]*$//' | sort -u)
    if [[ ${#DEVICES[@]} -eq 0 ]]; then
        echo "❌ No devices found with 'armbi_boot' or 'armbi_root' partitions."
        exit 1
    fi
    if [[ ${#DEVICES[@]} -eq 1 ]]; then
        TARGET_DEVICE="/dev/${DEVICES[0]}"
    else
        # Multiple devices found, show selection menu
        echo "📁 Multiple devices found. Please choose:"
        for i in "${!DEVICES[@]}"; do
            echo "[${i}] /dev/${DEVICES[$i]}"
        done
        read -p "🔍 Enter the number of the device: " CHOICE
        if ! [[ "$CHOICE" =~ ^[0-9]+$ ]] || [ "$CHOICE" -lt 0 ] || [ "$CHOICE" -ge ${#DEVICES[@]} ]; then
            echo "❌ Invalid choice."
            exit 1
        fi
        TARGET_DEVICE="/dev/${DEVICES[$CHOICE]}"
    fi
else
    TARGET_DEVICE="$1"
fi
echo "📌 Selected device: $TARGET_DEVICE"

#####
# Configure parameters for boot the image.

# Get the device where "armbi_boot" is mounted.
DEVICE=${TARGET_DEVICE#/dev/}
BOOT_MOUNT=$(lsblk -rno MOUNTPOINT,LABEL,PKNAME,NAME | awk -v dev="$DEVICE" '
$2 == "armbi_boot" && $3 == dev {print $1}')
ROOT_MOUNT=$(lsblk -rno MOUNTPOINT,LABEL,PKNAME,NAME | awk -v dev="$DEVICE" '
$2 == "armbi_root" && $3 == dev {print $1}')
if [ -z "$BOOT_MOUNT" ]; then
    echo "❌ Error: 'armbi_boot' partition not found."
    exit 1
fi
if [ -z "$ROOT_MOUNT" ]; then
    echo "❌ Error: 'armbi_root' partition not found."
    exit 1
fi

# Copy files to "armbi_boot".
echo "🔧 Copying boot files needed..."
cp -r "$PATH_BOOT_FILES"/* "$BOOT_MOUNT"/
rsync -av --exclude='*.command' "$PATH_BOOT_FILES"/ "$BOOT_MOUNT"/
echo "🔧 Creating a u-boot files backup for `apt upgrade` restore..."
# cp "$BOOT_MOUNT/*_autoscript" "$BOOT_MOUNT/*_autocript.bak"
find "/media/$USER/armbi_boot" -maxdepth 1 -type f -name '*_autoscript' -exec cp {} {}.bak \;

# Check if armbianEnv.txt exists.
ENV_FILE="$BOOT_MOUNT/armbianEnv.txt"
if [ -f "$ENV_FILE" ]; then
```



Script install-armbian.sh

```
#!/bin/bash
# Make the installation and necessary configurations to boot Armbian on a TV
Box.
# Write by Hildo Guillard Junior.
# Written on 12/November/2024.

echo "Installation script bof Armbian by"
echo "
echo "( )      ( )      ( )      ( )              ( )      ( )          ,
echo "( )__ ( ) _ ( )      __ ( )              ( )      ( ) ( ) _ ,
echo "( )      ( )      ( )      ( )      ( )      ( )      ( )      ( )      ,
echo "( )      ( )      ( )      ( )      ( )__ ( )      ( )__ ( )      ,
echo "( )      ( )      ( )      ( )      ( )      ( )      ( )      ( )      ,
echo ""

# General definitions used into the script.
boot_hw_dev=/dev/mmcblkpl1
root_hw_dev=/dev/mmcblkpl2
boot_hw=/mnt/boot_tv
root_hw=/mnt/root_tv

# All installation procedure and file access must be made as super user.
#sudo su
echo Starting mounting drive services...
systemctl daemon-reload
```




XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL
Construindo uma Engenharia Decolonial para a Soberania Digital e Popular
29 a 31 de outubro de 2025
Campinas - SP, Brasil

```
echo Present partitions...
lsblk

echo Making the Armbian installation...
#armbian-install # Not use this.
# sudo su # Necessary to return to installation super user.
/root/install-aml.sh

#dd if=/dev/sdb1 of=./installed_boot.iso bs=1M
#dd if=/dev/sdb1 of=./installed_root.iso bs=1M

echo Mounting installed system partitions...
mkdir -p $boot_hw $root_hw
mount $boot_hw_dev $boot_hw
mount $root_hw_dev $root_hw

echo Modifying the memory recognition and initialization...
cp /boot/s905_autoscript $boot_hw
cp $boot_hw/s905_autoscript $boot_hw/emmc_autoscript
chmod -w $boot_hw/emmc_autoscript

echo Making boot point to root partition...
boot_hw_uuid=$(blkid | grep $root_hw_dev | sed -n 's/.*
UUID="\([^"]*\)".*/\1/p')
sed -ri "s|^rootdev=UUID=[^ ]*|rootdev=UUID=$boot_hw_uuid|"
$boot_hw/armbianEnv.txt

echo Deleting current file from installed OS...
rm -f "$root_hw/$0"
file_base_name="$(basename "$0" | cut -d'.' -f1)"
rm -f "${root_hw}${HOME}/.config/autostart/${file_base_name}.desktop"
# This is used to delete the file for ordinary user,
# because I have created a image if this file into.

echo Unmounting partitions...
umount $boot_hw $root_hw
rm -r $boot_hw $root_hw

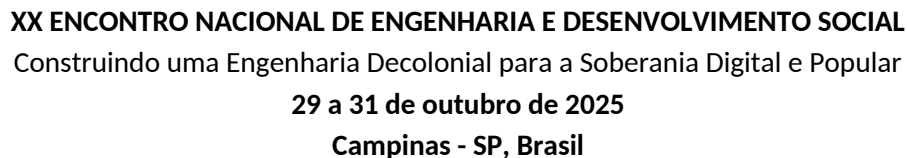
# echo Adding script for first run...
# cp -r * $root_hw

echo Restarting...
#shutdown -r now
# shutdown -r 5
echo Remove the pendrive before reboot!!!
echo 5; sleep 1
echo 4; sleep 1
echo 3; sleep 1
echo 2; sleep 1
echo 1; sleep 1
poweroff
```

Script install-packages.sh

```
#!/bin/bash
# Install the softwares and packages that I like to use.
# Write by Hildo Guillard Junior.
# Written on 12/November/2024.

ACTUAL_DIR="$( cd "$( dirname "${BASH_SOURCE[0]}" )" && pwd )"
```



```
_USER=$(basename ~)

echo 'Configuration script by'
echo '
echo '( ) ( ) ( ) ( )          ( )      ( )      '
echo '( )__ ( )    ( )     ( )      ( )      ( )      '
echo '( )      ( )   ( )    ( )      ( )      ( )      '
echo '( )      ( )   ( )    ( )      ( )      ( )      '
echo '( )      ( )   ( )    ( )      ( )      ( )      '
echo ''

# test -f /usr/bin/python || ln -s /usr/bin/python3 /usr/bin/python
# rm /usr/bin/pip -f && ln -s /usr/bin/pip3 /usr/bin/pip
#sudo dpkg-reconfigure tzdata

# Internet connection.
if [[ -f eduroam-linux-unesp.py ]]; then
    read -p 'Configure Unesp Eduroam Wi-Fi connection? (Y/n) [DEFAULT = n;'
TIMEOUT=5s] ' -n 1 -r -t 5 REPLY
    # echo      # (optional) move to a new line
    if [[ $REPLY =~ ^[Yy]$ ]]; then
        python3 eduroam-linux-unesp.py
    else
        echo 'Ignored.'
    fi
fi

# Personal configurations.
if [[ -d "$ACTUAL_DIR/TEMPLATE" ]]; then
    cp TEMPLATE "$HOME/TEMPLATE"
fi

# Function to recursively replace the "list:FILE.packages.list" lines
read_file_packages() {
    local file="$1"
    # Ensure the file exists
    if [[ ! -f "$file" ]]; then
        echo "Error: File '$file' not found."
        return 1
    fi
    # Avoid processing the same file multiple times to prevent infinite loops
    for f in "${processed_files[@]}"; do
        if [[ "$f" == "$file" ]]; then
            return
        fi
    done
    processed_files+=("$file")
    # Read the content of the file line by line without a pipe (avoid
subshell)
    while IFS= read -r line || [[ -n "$line" ]]; do
        # Check for lines starting with "include:"
        if [[ "$line" =~ ^include:(.*) ]]; then
            local included_file="${BASH_REMATCH[1]}"
            included_file="$ACTUAL_DIR/$included_file"
            # Including $included_file
            if [[ -f "$included_file" ]]; then
                read_file_packages "$included_file"
            else
                echo "Warning: Included file '$included_file' does not exist."
            fi
        else
            echo "$line"
        fi
    done
}
```



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL

Construindo uma Engenharia Decolonial para a Soberania Digital e Popular

29 a 31 de outubro de 2025

Campinas - SP, Brasil

```
        fi
    done < "$file"
}
processed_files=()

# Install software packages.

IFS=$'\n' packages_files_list=$(find "$ACTUAL_DIR" -type f -name
'*.packages.list' | sort)
unset IFS
echo 'Available installation level intended:'
select REPLY in "${packages_files_list[@]}"; do
    echo "Selected '$REPLY'";
    break
done
packages=$(read_file_packages "$REPLY")
packages=$(echo "$packages" | sed 's/\s*#.*$//' | grep -vE '^\\s*$') # Remove
comments.

# Remove comand by lines started with "-".
to_remove=""
while IFS= read -r line; do
    [[ "$line" =~ ^-(.) ]] && to_remove+=" ${BASH_REMATCH[1]}"
done <<< "$packages"
filtered=""
while IFS= read -r line; do
    [[ "$line" =~ ^- ]] && continue
    for rem in $to_remove; do
        [[ "$line" == "$rem" ]] && continue 2
    done
    filtered+="$line"$'\n'
done <<< "$packages"
packages="$filtered"

# Packages, softwares and repositories to install.
repositories_debian=$(echo "$packages" | grep -E "^ppa:.*")
weblinks=$(echo "$packages" | grep -E "^(https*://|www\\.).*")
packages_snap=$(echo "$packages" | grep -E "^snap.*")
packages_debian="$packages"
packages_debian=$(echo -e "${packages_debian}\n${repositories_debian}" | sort
| uniq -u)
packages_debian=$(echo -e "${packages_debian}\n${weblinks}" | sort | uniq -u)

#echo '->web:' "$weblinks"
#echo '->snap:' "$packages_snap"
#echo '->repositories:' "$repositories_debian"
#echo '->packages:' "$packages_debian"

packages_debian=$(echo "$packages_debian" | tr '\n' ' ')
weblinks=$(echo "$weblinks" | tr '\n' ' ')
packages_snap=$(echo "$packages_snap" | tr '\n' ' ')

#echo '->packages:' "$packages_debian"
#exit 0

echo 'Type the admin password for initial configuration...'
#sudo su
sudo apt update && sudo apt install --yes software-properties-common # To add
the `apt-add-repository` command, necessary to next steps.

if [[ -n "$repositories_debian" ]]; then
    for repository in $repositories_debian; do
```



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL
Construindo uma Engenharia Decolonial para a Soberania Digital e Popular
29 a 31 de outubro de 2025
Campinas - SP, Brasil

```
        sudo apt-add-repository --yes --no-update $repository
    done
    sudo apt update # Update once after add all repositories.
fi
if [[ -n "$packages_debian" ]]; then
    sudo apt install --yes $packages_debian
fi
if [[ -n "$weblinks" ]]; then
    #sudo -i -u $USER \
    xdg-open $weblinks & # Open with default browser.
fi
if command -v snap >/dev/null 2>&1 && [[ -n "$packages_snap" ]]; then
    sudo snap install $packages_snap
fi

# Deal with some package installation error.
#sudo rm /var/lib/dpkg/info/libc-bin.*
#sudo apt clean
#sudo apt update
#sudo apt install libc-bin

read -t 5 -p "Type the packages with installation errors (space-separated): "
packages
echo

if [ -n "$packages" ]; then
    for pkg in $packages; do
        echo "Fixing '$pkg' package installation..."
        rm -f /var/lib/dpkg/info/${pkg}.*
        apt clean
        apt update -y
        apt install -y $pkg
    done
fi

read -t 5 -p "Do you want to add install-armbian.sh script to autostartup?
[Y/n] " answer
answer=${answer:-Y}
if [[ "$answer" =~ ^[Yy]$ ]]; then
    file_name="install-armbian.sh"
    if [ -n "$DISPLAY" ] || [ -n "$WAYLAND_DISPLAY" ]; then
        file_base_name="$(echo "$file_name" | cut -d'.' -f1)"
        autostart_file="$HOME/.config/autostart/$file_base_name.desktop"
        cp $file_name "$HOME/Downloads"
        mkdir -p "$HOME/.config/autostart"
        cat > "$autostart_file" << EOF
[Desktop Entry]
Type=Application
Exec=xfce4-terminal -e "bash -c 'echo Type the SuperUser password to proceed
installation or \<Ctrl\>+c to cancel.; sudo bash \"$HOME/Downloads/install-
armbian.sh\"'"
Hidden=false
X-GNOME-Autostart-enabled=true
Name=Install Armbian Script
Comment=Run Armbian installation script at XFCE startup.
EOF
        chmod +x "$HOME/.config/autostart/install-armbian.desktop"
    else
        echo "⚠ Not in a graphical environment, skipping autostart creation."
    fi
else
    echo "👉 Skipped adding to autostart."
```



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL
Construindo uma Engenharia Decolonial para a Soberania Digital e Popular
29 a 31 de outubro de 2025
Campinas - SP, Brasil

fi

Lista de pacotes de software 01_Only_prompt.packages.list

```
# Basic applications needed just for prompt use of the computer.
software-properties-common # To add the `add-apt-repository` command.
python3-pip
python-is-python3
```

Lista de pacotes de software 02_Office_apps.packages.list

```
# Applications for basic and office use.
include:01_Only_prompt.packages.list

# Base graphical interface.
# Prefer to use the already installed graphical desktop into the distribution
due the personalization made.
#xorg
#dbus-x11
#lightdm # Display manager.
#xfce4 xfce4-goodies # Utilities: network configuration, ...
##lxde-core

# Browser.
ppa:mozillateam/ppa # It is not needed on XFCE pre-installeted image.
#firefox-esr
chromium chromium-l10n
#libavcodec-extra

# Office applications.
libreoffice libreoffice-help-pt-br libreoffice-l10n-pt-br
hunspell-en-us hunspell-pt-br # Realtime typing checking.
hyphen-en-us hyphen-pt-br
aspell-en aspell-pt-br

# Other utils.
#vlc
#totem
#gimp
#inkscape
#p7zip-full
```

Lista de pacotes de software 03_Education_apps.packages.list

```
# Applications for education proposes.
include:02_Office_apps.packages.list

# Prioritize Chromium as web browser instead Firefox due some issues on
restart.
-firefox-esr
chromium chromium-l10n

# Educational applications.
geogebra
```



XX ENCONTRO NACIONAL DE ENGENHARIA E DESENVOLVIMENTO SOCIAL

Construindo uma Engenharia Decolonial para a Soberania Digital e Popular

29 a 31 de outubro de 2025

Campinas - SP, Brasil

#geogebra-gnome

kalgebra

#kalzium

kbruch

#kmpplot

kturtle scratch # Programming.