



Prática 7: Desenvolvimento e customização de Sistemas com Linux Embarcado

Resumo

Introdução a customização de sistemas com Linux embarcado utilizando a plataforma Raspberry Pi. Explorando ferramentas build para criação de novas distribuições Linux para sistemas embarcados específicos. Geração de uma imagem Linux customizada para um Set-Top-Box utilizando um build (Armbian). Boot na placa e configuração da distribuição Linux a partir da imagem gerada.

Conceitos importantes:

Boot process, Kernel, device drivers, device tree, build system, toolchain,, cross-compiling, Buildroot, Yocto, Open-embedded, Armbian, SoC.

Objetivo

Sistemas embarcados de produtos comerciais são específicos e possuem funcionalidades restritas. Por vezes exigem tempo de Boot menor, controle de pacotes para licença comercial, requisitos de segurança, conjunto específico de aplicações, Boot via rede e sem uso de cartão micro SD. Neste âmbito, torna-se vantajoso uma distribuição Linux customizada para a aplicação embarcada. Por outro lado, as distribuições Linux “prontas”, não possuem tal grau de flexibilidade. Portanto, essa prática visa explorar as ferramentas build e recursos atualmente disponíveis para criação e customização de distribuições Linux para projeto de sistemas embarcados específicos que exigem alto grau de flexibilidade do sistema operacional.

I - *Build systems* para geração de imagem Linux customizada para sistemas embarcados

A estrutura e construção de um OS Linux embarcado é ilustrada na Fig. 1 e Fig. 2. Dentre os elementos já comentados anteriormente (bootloader, kernel, rootfs), é importante destacar que o hardware é nosso alvo (**target**). O ambiente de desenvolvimento, onde a distro é construída, é geralmente um computador com OS Linux (**host**). Ao final da compilação, os arquivos gerados são transferidos ao target.

Outro destaque na Fig. 3 é a **Toolchain**: conjunto de ferramentas de compilação para gerar os binários que estruturam o sistema operacional no target: bootloader, kernel, rootfs. Por exemplo, no processo de compilação de um código em C, é necessário pré-processamento (código C intermediário), compilação (conversão para assembly), arquivo objeto, e Linker (binário final, firmware, aplicação). Esse é o papel da toolchain, que integra todas essas funcionalidades, incluindo uma biblioteca C padrão (predominante em sistemas Linux - GNU C library - glibc).

Cross-compiling toolchain (compilação cruzada): quando a host possui arquitetura diferente da target (ex.: X86-64 para ARM). A diferença entre Toolchain nativa e *cross-compiling* é ilustrada na Fig. 3. Um exemplo de Toolchain bastante usada em sistemas embarcados é o projeto [Linaro](#).

O processo de desenvolvimento de um sistema com Linux embarcado depende de requisitos funcionais da aplicação: o que a placa deve controlar? Quais as condições ambientais? Necessidade de tempo real? Quais interfaces externas?

Ademais, a seleção da plataforma de hardware (target) adequada, também depende diretamente das condições acima e da relação custo vs. funcionalidades vs. suporte disponível.

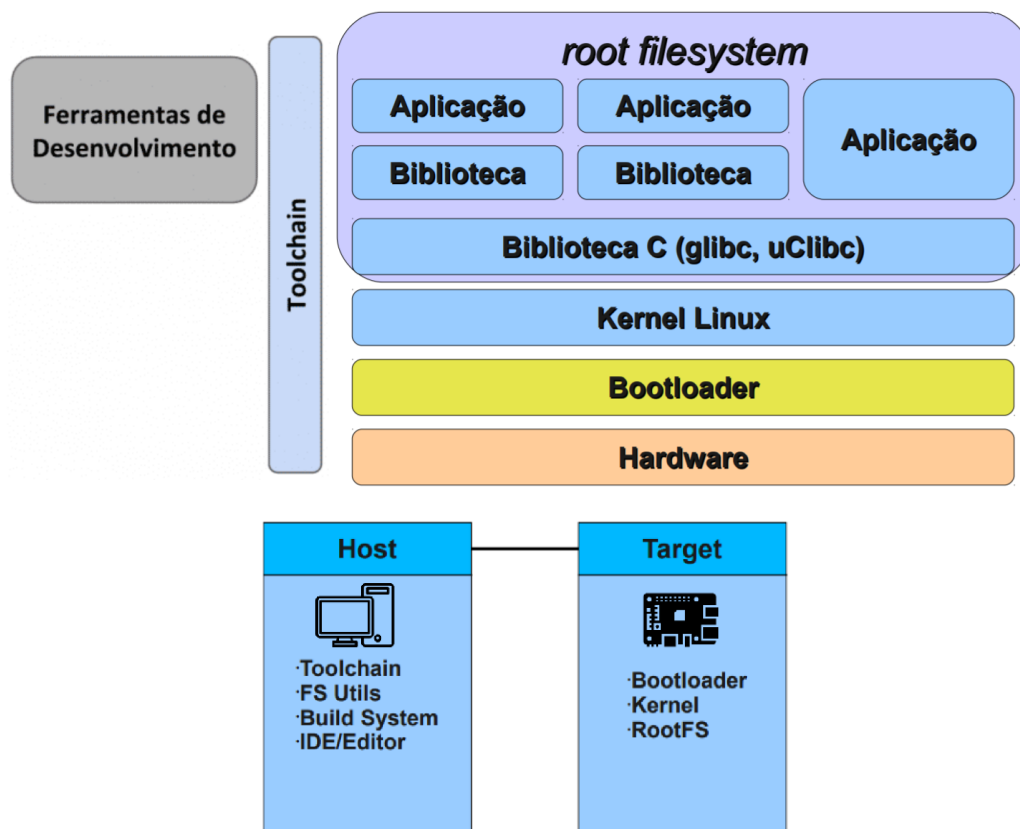


Fig. 1 - Estrutura de um sistema Linux e ambiente de desenvolvimento para criação de uma distro embarcada.

Fonte (imagem): [Embarcados/BoardWare](#) (adaptado)

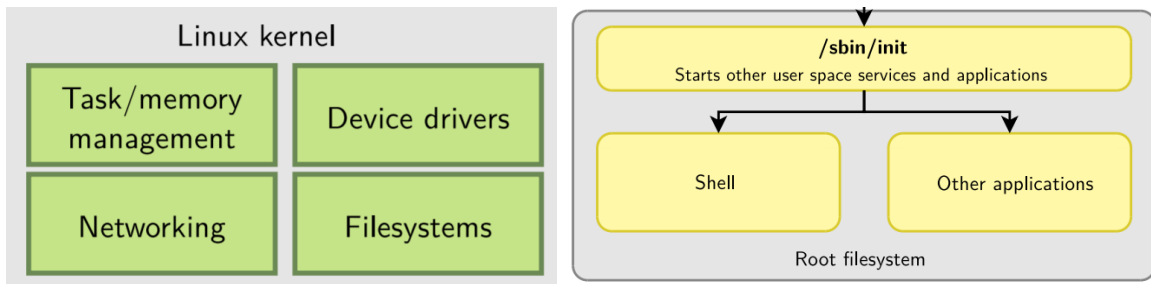


Fig. 2 - Outros detalhes do kernel e Root filesystem
Fonte (imagem): [Buildroot](#)

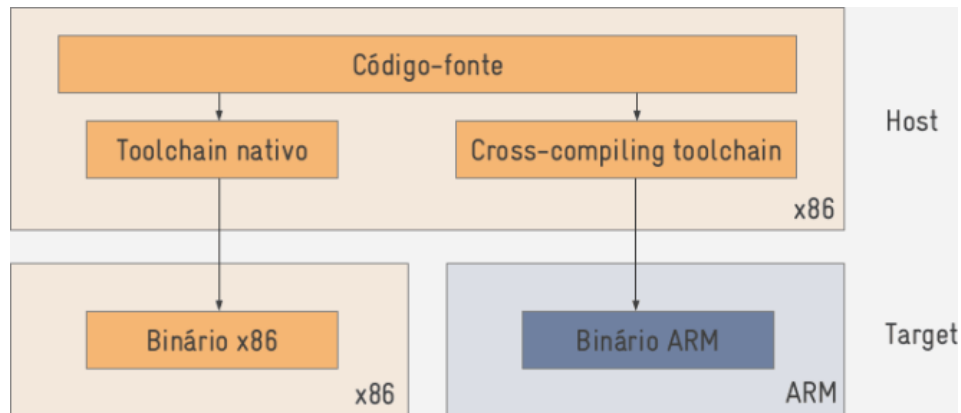


Fig. 3 -Estrutura de Toolchains.
Fonte (imagem): [Embedded Labworks](#)

Usar uma distro pronta (ex. Raspberry Pi OS, Ubuntu, soluções comerciais de empresas etc.), apesar da facilidade, funcionalidades já implementadas e framework pronto, possui baixo grau de flexibilidade para um target específico, não permite alterações, podem não possuir requisitos de licença ou segurança específicos etc. Supondo que uma aplicação prática não requer todas as funcionalidades da distro pronta, bem como deseja-se ter controle sobre pacotes open-source usados (considerada aplicação comercial), tempo de boot menor etc. ?

Uma opção seria criar a distro do zero (processo manual) - vantagens: controle total sobre as ferramentas utilizadas e grau de flexibilidade mais alto: distro Linux customizada para o target. Desvantagens: atividade suscetível a erros, mais demorada, trabalho manual. Ferramenta: [Linux from Scratch](#)

Build systems: Buildroot e Yocto Project

Uma outra opção é usar um *build system*, que permite gerar uma distro Linux customizada para um target com vantagem de usar ferramentas, por vezes open-source, parcialmente já configuradas, para construir o sistema operacional, facilitando o trabalho. Nesta opção, destacam-se duas ferramentas:

- **Buildroot** - um conjunto de Makefiles e patches que simplifica e automatiza o processo de construção de um ambiente Linux - sistema mais simples e com maior facilidade. [Documentação](#).
- **Yocto Project** - utiliza o build [Open Embedded](#) - é um projeto open-source colaborativo para criação de distribuições Linux. É mais completo, porém, não tão simples de se

trabalhar como Buildroot. Trabalha com um sistema de referência denominado “Poky”, que contém uma coleção de várias receitas para criação e customização de novas distros.

A Fig. 4 ilustra uma comparação entre as duas ferramentas. No caso do Yocto, na parte inferior, ele opera com uma ferramenta para execução de tarefas e gerenciamento de metadados escrita em python chamada BitBake; e com **metadados** (coleção estruturada de “receitas” e arquivos que “dizem” ao Bitbake o que e como construir o sistema operacional).

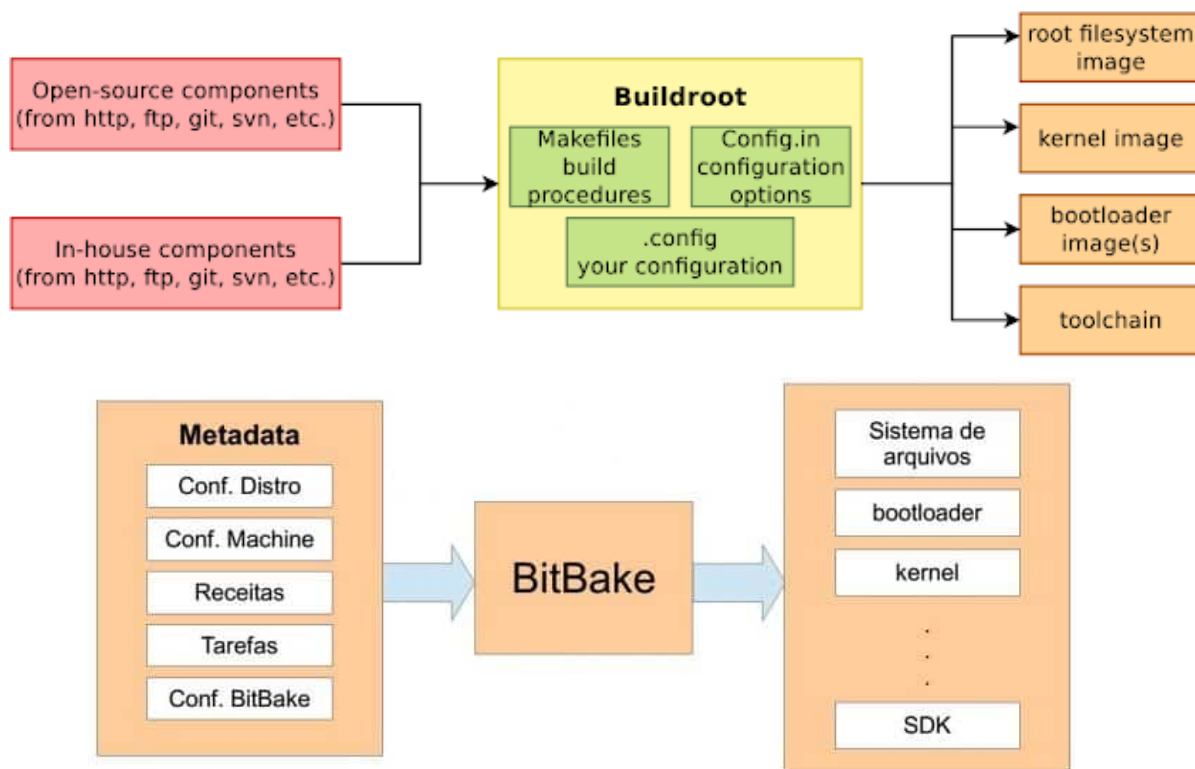


Fig. 4 - Buildroot e Yocto Project.
Fonte (imagens): [Buildroot](#) / [Embarcados](#)

Build systems: Armbian

O [Armbian](#) também é uma plataforma para geração de imagens de distribuições Linux personalizadas, especialmente focada no desenvolvimento de sistema operacional para computadores de placa única (SBCs), com base no framework do Debian e Ubuntu. Oferece suporte para várias placas SBCs de comunidades ([Raspberry Pi](#), Orange Pi, Banana Pi etc), que por vezes não são suportadas pelo Debian ou Ubuntu oficial, fazendo com que esse suporte aconteça por meio da otimização da imagem Linux. O projeto é focado na otimização da distribuição para uso em servidores de arquivo, servidores de mídia e desktops. O Armbian é uma plataforma popular quanto a geração de imagem Linux para uso em Set-Top-Box, ou visando sua descaracterização e transformação em minicomputadores (Projeto de Extensão “Além do Horizonte” da Receita Federal que descaracteriza aparelhos de TV Box ilegais por meio da instalação de uma imagem Linux gerada com Armbian).



Fig. 5 - Armbian

II - Gerando uma imagem personalizada para TV Box com Armbian

A documentação do processo encontra-se na [página](#) e no [GitHub](#) do Armbian. O objetivo desta atividade é conhecer a ferramenta que permite a geração da imagem customizada. Vamos usar como exemplo um hardware específico, Set-Top-Box modelo H7, com SoC Amlogic S905X3, e implementar roteiro e tutoriais indicados a seguir.

Roteiro

- Em um PC com Linux (Debian ou Ubuntu), que será a máquina de desenvolvimento (host), configurar o ambiente, instalando os pacotes e dependências necessárias.
- Para ser possível compilar a imagem Linux personalizada usando o Armbian, é necessário a instalação e configuração do Docker. O [Docker](#) é uma aplicação que simplifica o processo de gerenciamento de processos de aplicativos em contêineres no Linux. Os contêineres permitem a execução de aplicativos em processos isolados em termos de recursos (são semelhantes a máquinas virtuais, porém, mais portáteis, consomem menos recursos e dependem mais do sistema operacional físico).
- Portanto, execute os passos a seguir para instalação e configuração do Docker:
 - Atualizar pacotes, instalar pré-requisitos, adicionar uma chave GPG e repositório do Docker:

```
sudo apt update
sudo apt install apt-transport-https ca-certificates curl gnupg2
software-properties-common
curl -fsSL https://download.docker.com/linux/debian/gpg | sudo apt-key add -
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/debian $(lsb_release -cs)
stable"
```

- Verificar e instalar o Docker:

```
apt-cache policy docker-ce
sudo apt install docker-ce
sudo systemctl status docker
```

- Para executar o comando Docker sem uso do “sudo”:

```
sudo usermod -aG docker ${USER}
su - ${USER}
```

- Confirmando se o usuário foi adicionado ao grupo Docker:

```
id -nG
```

- Para adicionar diretamente um usuário ao grupo Docker (usuário sel):

```
sudo usermod -aG docker sel
```

- Para uso do docker (*docker [option] [command] [arguments]*) visualize os subcomandos:

```
docker
```

- Após ter instalado e configurado as dependências, seguir o tutorial abaixo para clonar o repositório “build” do Armbian e iniciar o processo de compilação da imagem Linux customizada:

```
sudo apt update
sudo apt -y install git
git clone --depth=1 --branch=main https://github.com/armbian/build
cd build
./compile.sh
```

- Após executar `./compile.sh`, abrirá uma janela que permitirá a customização da imagem Linux. Seguir o tutorial em anexo ao final deste documento para escolher as configurações mais pertinentes, de acordo com o target. Entretanto, como o processo de compilação pode levar várias horas (5, 6 ou 7 horas), vamos apenas explorar parcialmente esse recurso durante essa atividade prática. Entretanto, a ferramenta poderá servir como base caso, no futuro, se desejar compilar uma distro personalizada para uma aplicação embarcada específica. Portanto, após personalizar a imagem e realizar o último passo do tutorial a seguir (janela: “*choose desktop software to add*”). A imagem começará a ser compilada. Interromper o processo e continuar seguindo este roteiro.

- Caso aguardasse a imagem ser compilada, após fim do processo acima a mesma poderia ser encontrada em `/home/sel/build/outputs/image`. Ao invés disso, vamos fazer o download de uma imagem já compilada para o target (TV Box H7) que escolhemos e seguir os passos abaixo para dar boot e configurar a distribuição para uso como Desktop.
 - No PC (windows) do laboratório, faça o download da imagem do Armbian disponível [aqui](#).
 - Instale o Balena Etcher no PC (windows) do laboratório (ou verifique a possibilidade de usar o Raspberry Pi Imager e escolha a última opção: carregar uma imagem ISO a partir do PC).
 - Transfira a imagem para o seu cartão microSD a partir de um adaptador (USB-MicroSD) ou para um pendrive. Após a transferência da imagem para o cartão/pendrive, remova-o com segurança no Windows.
 - Antes de iniciar, recomenda-se que a TV Box esteja conectada à rede sem fio (portanto, acesse o modelo H7 que será disponibilizado pelo professor no laboratório, conecte o aparelho ao Wi-Fi LabMicros ou ao Wi-Fi do seu smartphone - para tanto pode ser necessário ter que conectar a placa a um monitor, via HDMI, liga-la (Android) e navegar com controle remoto para configurar e conectar ao Wi-Fi). Para a descaracterização do aparelho e sua transformação em mini computador, no entanto, por restrições da ANATEL, é necessário desativar o Wi-Fi posteriormente por orientação da Receita Federal (outra opção é usar a rede cabeada, ethernet, porém, será necessário descobrir o endereço Mac da placa para solicitar ao setor de informática cadastrá-la na rede da USP - neste caso, fazer as configurações a seguir sem uso da internet e quando tiver acesso ao terminal, descobrir essa informação via linha de comandos passar o endereço Mac ao professor para que o cadastro seja solicitado).
 - Insira o cartão/pendrive na sua TV Box (pode ser que não funcione inserindo diretamente o cartão micro SD, sendo necessário usar a entrada USB conectando-o com o adaptador).
 - Ligue a TV Box, a qual deverá iniciar pelo cartão/pendrive. Caso isso não aconteça, será necessário acionar o botão RESET concomitantemente ao procedimento de ligar o equipamento. Para tanto, com um clipe, pressione o botão que se encontra dentro do orifício acima da porta USB identificada como RECOVER USB. Pressione o botão, ligue o cabo de energia mantendo o RESET pressionado por 3 segundos (conte de 1 a 5) e solte.
 - O Armbian deverá iniciar pelo cartão/pendrive. Caso não funcione, repita o procedimento.
 - Após o carregamento do Armbian, deve-se definir:
 - Qual shell utilizar (selecionar bash).
 - Senha do root.
 - Nome de usuário comum (colocar sel).
 - Nome por extenso do usuário comum.
 - Senha do usuário comum (usp)
 - Fuso horário.

- Se deseja usar Wi-Fi ou não: a conexão será realizada pela ferramenta `armbian-config`.
- Se deseja fazer a codificação em função do seu local (`pt_br.utf-8`).
- Ao término, o prompt estará disponível, no usuário `root`, sendo disponível para configuração e/ou transferência para a TV Box.
- Reinicie a TV Box com o comando `sudo reboot`.
- Após o reinício, o equipamento aguardará no prompt do login do shell. Você pode logar com o nome e senha de usuário comum que definiu anteriormente.
- Lembre-se: a imagem utilizada neste protocolo diz respeito a uma imagem server. Todo o ajuste de ferramentas adicionais deverá ser feito posteriormente, incluindo a parte gráfica. Essa escolha se deu pelas características da TV Box, permitindo que se possa otimizar da forma que preferir.
- Atualizando a lista de pacotes: `sudo apt update` (será necessário fornecer a sua senha de usuário comum e estar conectado a internet, portanto, verifique o endereço Mac da placa por meio do comando `ifconfig` ou `ip addr`).
- Atualizando os pacotes (se for necessário): `sudo apt upgrade` (pode ser necessário fornecer a sua senha de usuário comum).
- Em tese, o Armbian está instalado e pronto para o restante da configuração. Antes de escolher a sua interface gráfica, é necessário instalar o pacote `xorg`. Para tanto, utilize o comando `sudo apt install xorg` (pode ser necessário fornecer a sua senha de usuário comum). Foi observado que a configuração padrão do `xorg.conf` apresenta conflito com o hardware da H6+. Caso também apresente conflito com o modelo H7, basta executar a sequência de comandos abaixo para resolver o problema:


```
cd /etc/X11
sudo rm xorg.conf
sudo wget http://www.lnrd.dev/xorg.conf
```
- A sequência apaga o arquivo original e o substitui por outro disponível neste site já testado pela comunidade de descaracterização de TV Box.
- Para configurar o teclado, digite o comando `sudo armbian-config`
- Escolha Personal, Keyboard. Em seguida, escolha as configurações mais adequadas ao equipamento que será utilizado.
- Para instalar o XFCE digite:

```
sudo apt install task-xfce-desktop xfce4-statusnotifier-plugin
network-manager-gnome
```

- Para instalar um leitor de PDF, captura de tela, compactador de arquivos, leitor automático de pendrives, etc... execute

```
sudo apt install thunar-volman gvfs policykit-1 xfce4-screenshooter file-roller
qpdfview ristretto ttf-mscorefonts-installer catfish thunar-archive-plugin
```


- Como são arquivos que ocupam bastante espaço, sempre executar `sudo apt-get clean` após cada instalação de grande quantidade de aplicativos.
- Configurando o LightDM para iniciar automaticamente no usuário comum: caso o usuário comum definido na instalação seja “sel”. A cada reinício, faz-se necessário fornecer o nome do usuário, sel, e a sua senha. Contudo, com essa senha torna-se possível realizar alterações no sistema de acordo com as configurações atuais. A solução é optar por iniciar automaticamente no usuário padrão sem a necessidade de acesso à senha para os discentes das escolas atendidas com esse aparelho descaracterizado.
- No prompt do Terminal, execute `sudo nano /etc/lightdm/lightdm.conf`
- Busque as linhas a seguir e altere-as de forma a ficar conforme descrito abaixo:

```
[SeatDefaults]
autologin-user=receitafederalunifal
autologin-user-timeout=0
```

- Salve o arquivo pressionando simultaneamente Ctrl+O.
- Execute `sudo reboot`
- O equipamento deverá reiniciar e logar automaticamente com o usuário comum.
- A partir daqui, o aparelho já estará configurado para ser usado como Desktop. O próximo passo seria executar a transferência da imagem do cartão microSD/ pen drive para a flash da TV Box, permitindo a sua descaracterização permanente. Como nesta prática o objetivo é testar apenas um equipamento, não vamos realizar este procedimento.
- Caso deseje replicar esse procedimento em vários aparelhos, execute todos os passos sugeridos aqui inicialmente no cartão SD/pen drive e, somente após ter tudo adequadamente configurado, efetuar a transferência por meio do procedimento B deste [tutorial](#) (não realizar nesta prática).

Entrega

Arquivo .txt com histórico dos comandos usados no host (PC) e target (TV Box), com comentários/explicação resumida dos conceitos da prática relativos à geração de uma imagem customizada para um hardware específico por meio de um build.

Documentação: demonstrar o funcionamento ao professor ou gravar um vídeo mostrando a tela do aparelho com a nova imagem instalada e configurada. Caso não queira gravar um vídeo, enviar um arquivo PDF, ou README.md do GitHub com fotografias da tela do aparelho com a nova imagem instalada e configurada. Ter mostrado o funcionamento ao professor substitui a necessidade de envio de um vídeo ou de um documento (PDF ou README.md).

Bibliografia

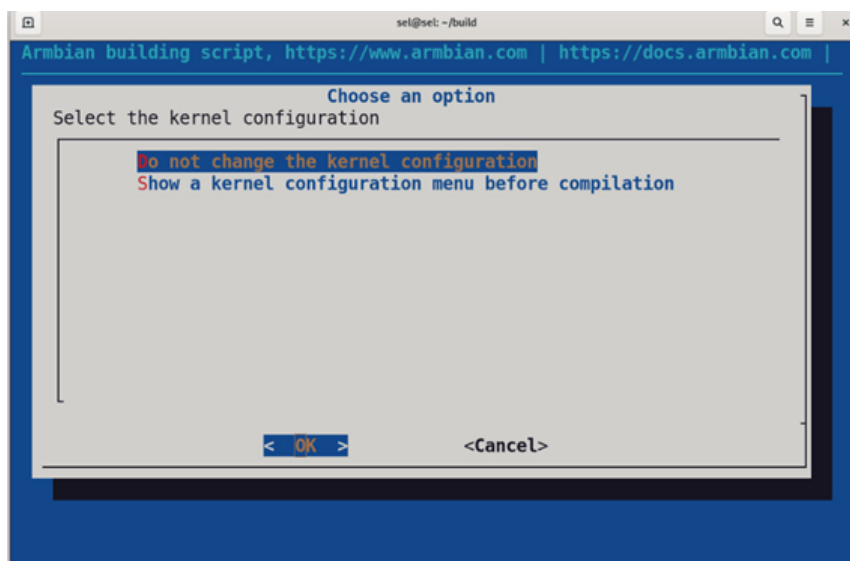
- Cap. 7 - Desenv. e Customização de Linux Embarcado
- [Ambian - documentação](#)
- [Armbian - tutorial GitHub](#)
- [Tutorial para descaracterização do modelo H6+ \(também válido para H7\).](#)
- [TV-Box já descaracterizadas e projetos desenvolvidos pela comunidade e instituições](#)

Tutorial para geração da imagem para o modelo H7 Amlogic S905X3

Seguir os passos do roteiro acima para instalação das dependências necessárias, como Docker, e seguir este tutorial somente após ter executado o comando: `./compile.sh` no PC host.

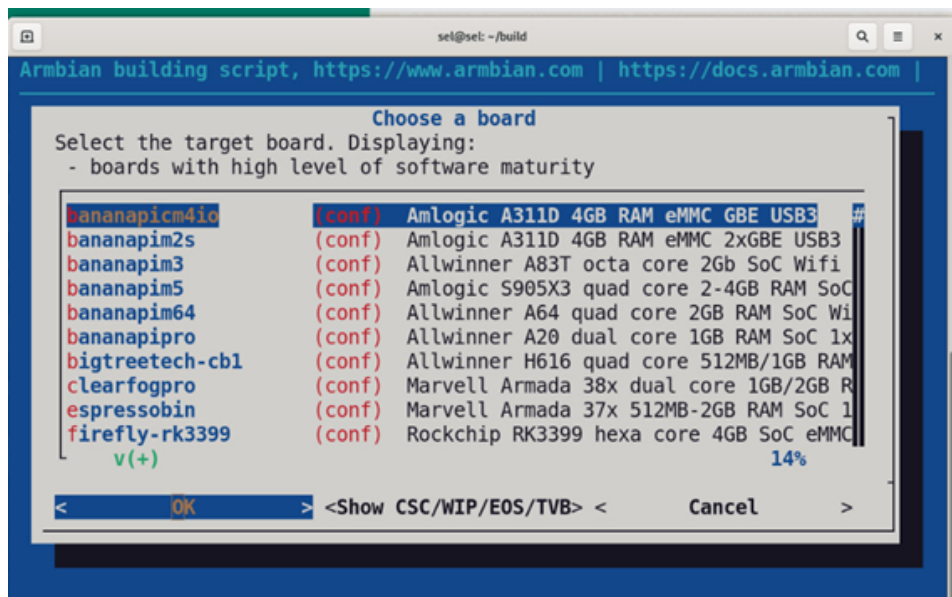
Primeiramente, pode ser necessário ligar o aparelho (Android), conectando-o em um monitor via HDMI e, navegando com controle remoto para conseguir ter acesso à loja de aplicativos, instalar o aplicativo AIDA64. Este software irá mostrar quais são as especificações técnicas do hardware da TV Box, incluindo a versão do processador, para especificá-lo corretamente na escolha da imagem a seguir.

1 - Após executar: `./compile.h`, o primeiro passo, na tela a seguir, é escolher a configuração do kernel. Caso deseje verificar/modificar, escolha a segunda opção. Navegar nesta interface usando o teclado do computador (setas e enter).



2 - Na próxima janela, poderá escolher uma imagem a ser compilada para um target da lista, segundo o modelo de SBCs e do SoC descrito ao lado. Entretanto, para a TV Box, é necessário

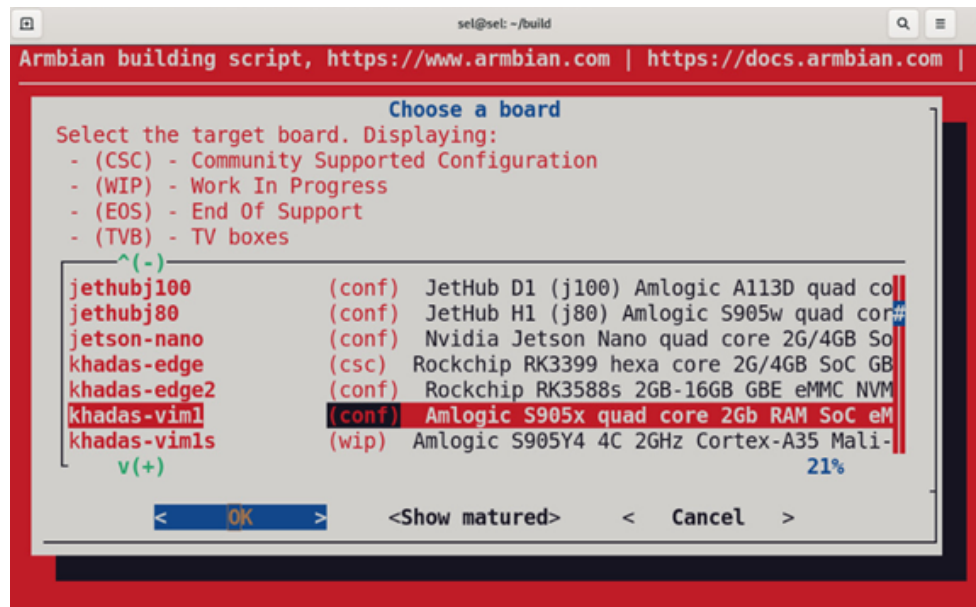
acessar a opção <Show CSC/WIP/EOS/TVB> ao lado de OK. Observe que é possível gerar imagens personalizada para diferentes modelos de SBCs.



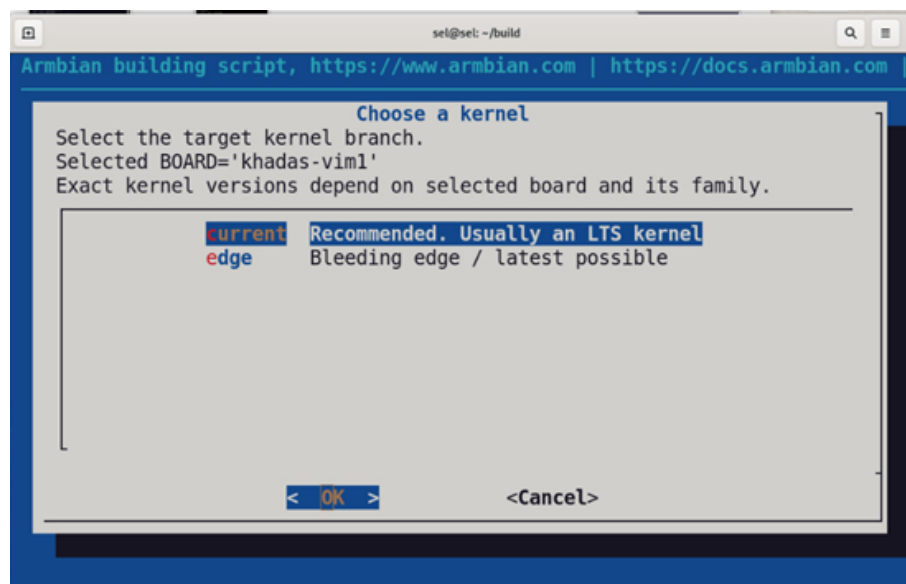
3- Na próxima janela, confirme que irá compilar uma imagem experimental por sua conta e risco, na opção < I understand and agree>.



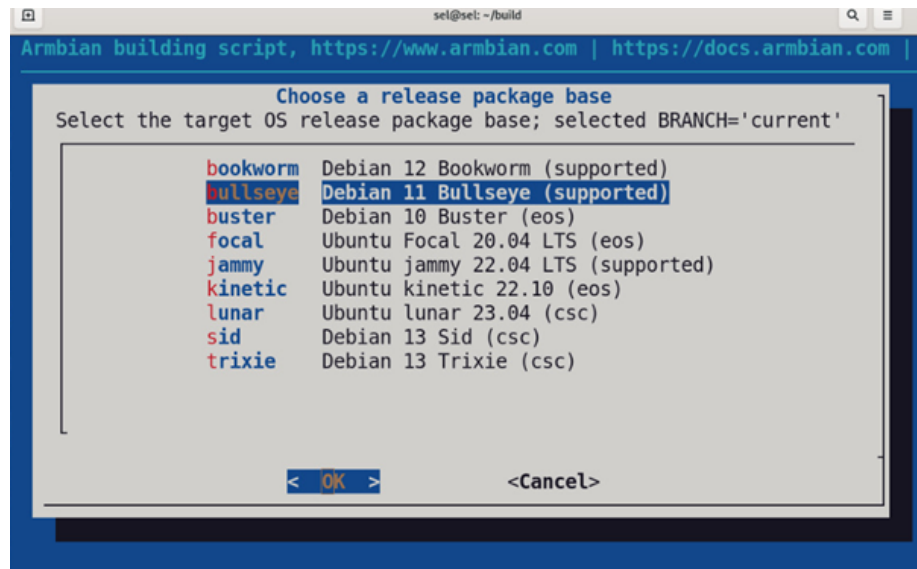
4 - Na próxima janela, procure pelo modelo de SoC (processador) presente na TV Box para qual deseja gerar a imagem (consulte a especificação na placa) e depois “OK”. Para o modelo H7, a versão é Amlogic S905X3.



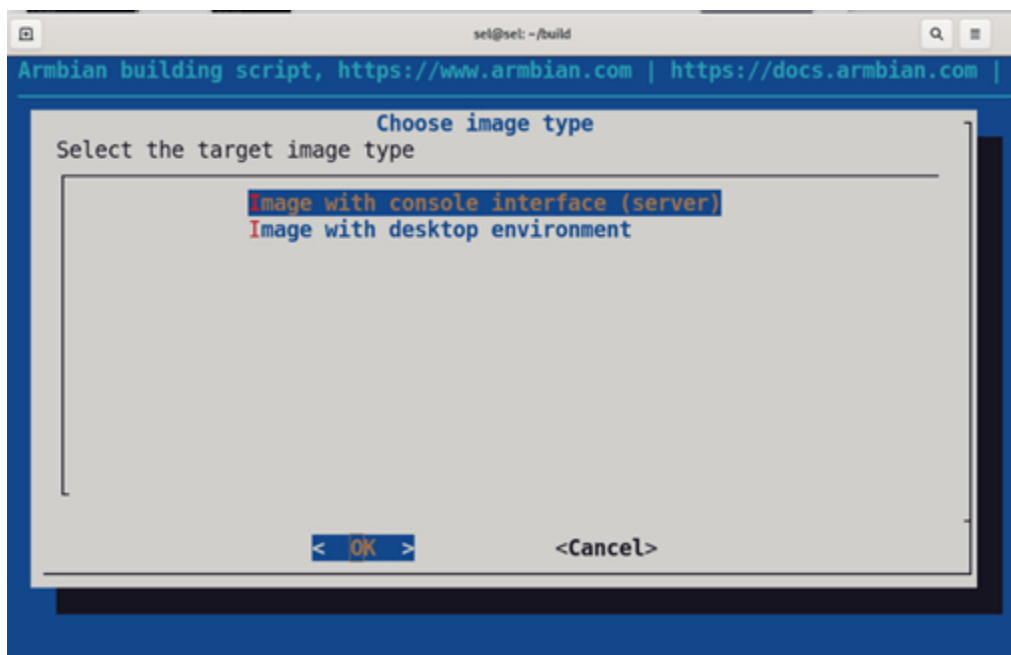
5- Na próxima janela, selecione a versão do kernel (pode ser a “recomendada”, pois é mais consolidada) e depois OK.



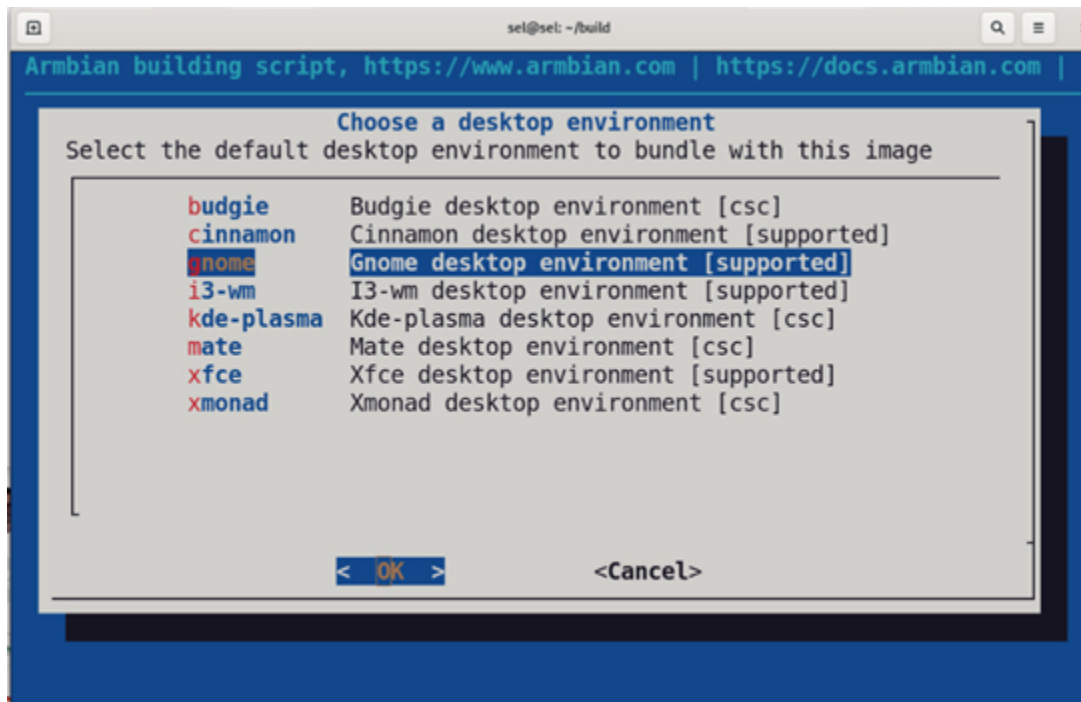
6- Na próxima janela, selecione o pacote base para instalação da distribuição que será a base da imagem a ser gerada. A “bullseye” está sendo a mais popular para Debian.



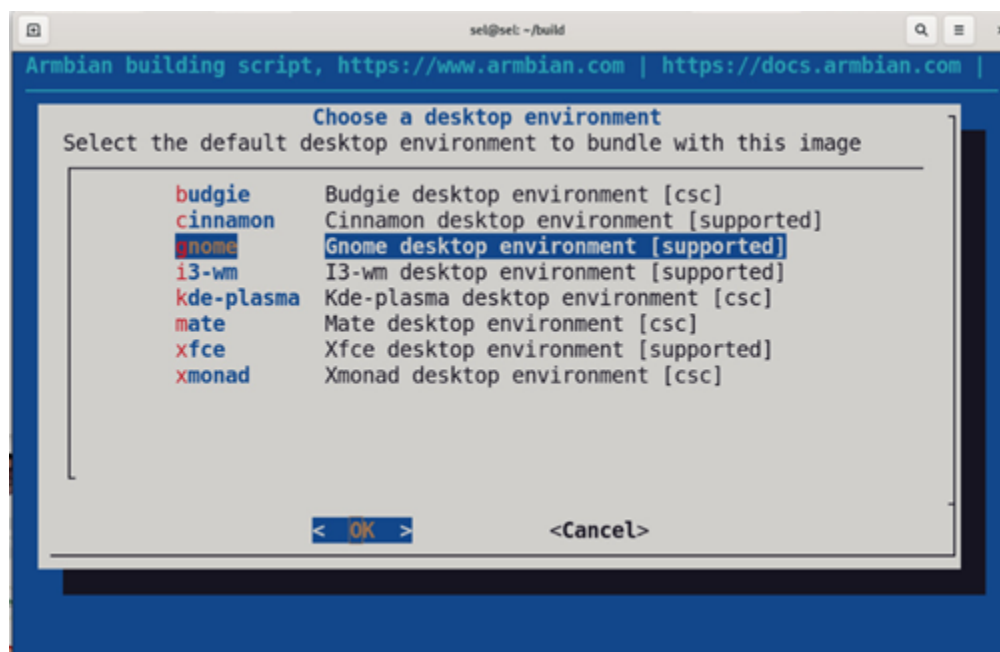
7 - Na próxima janela, escolha o tipo de imagem: com interface gráfica (se desejar transformar a TV Box em um mini computador de propósito geral) ou a versão “server”, sem interface gráfica, caso for usá-la para projetos específicos, como servidor.



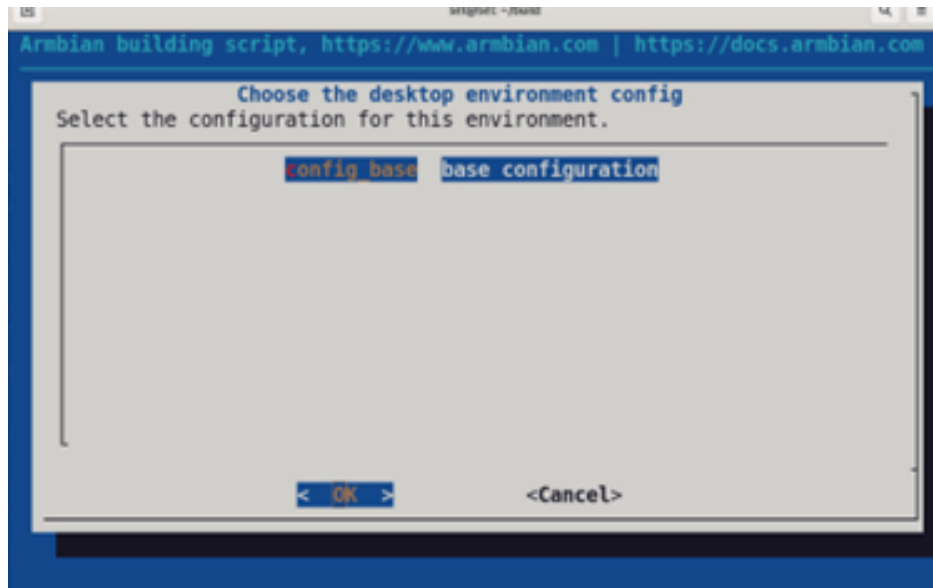
8 - Na próxima janela, escolha qual interface gráfica deseja usar, como Gnome, KDE etc.



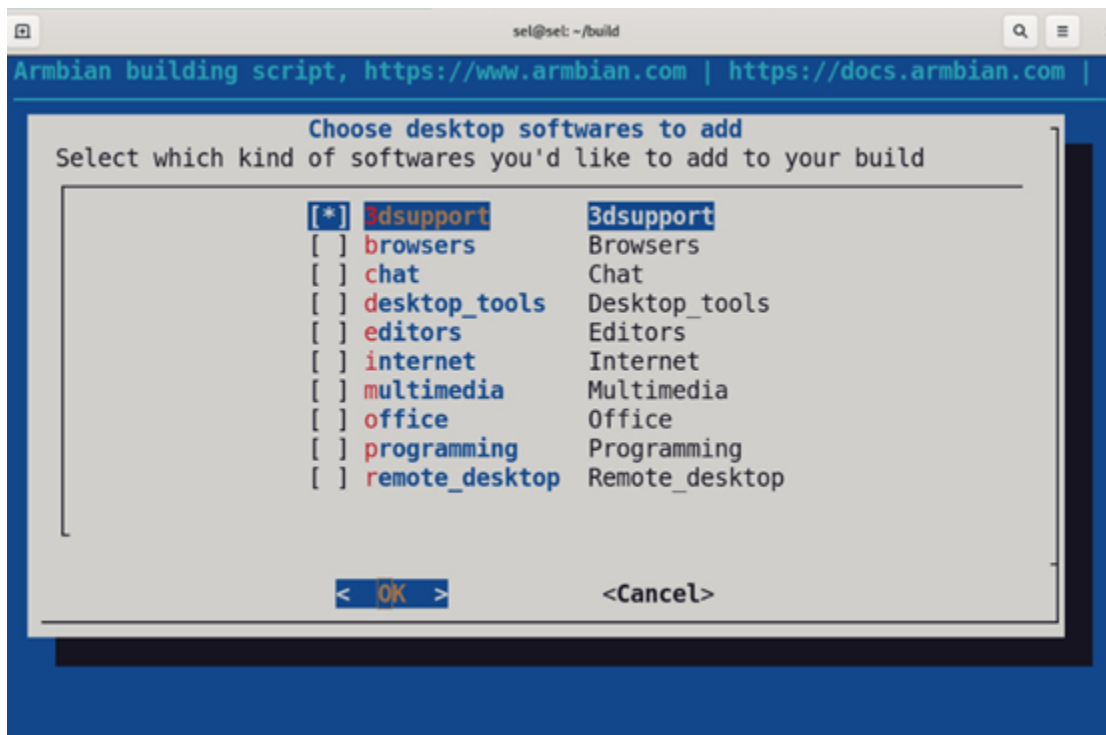
9 - Na próxima janela, escolha qual interface gráfica deseja usar, como Gnome, KDE etc.



10 - Em seguida, selecione a configuração base para essa interface

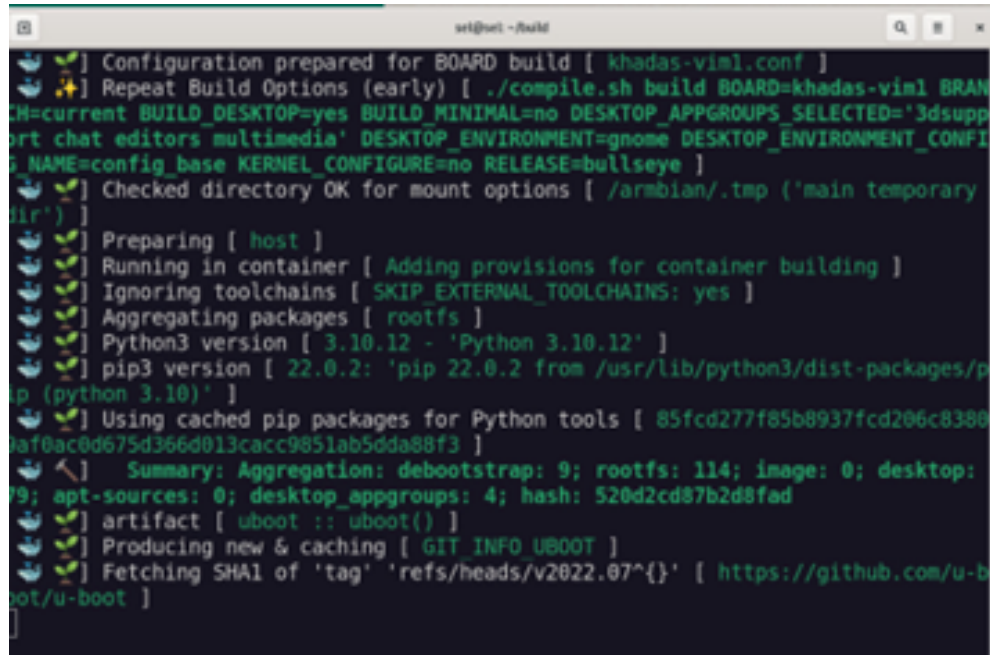


11 - Em seguida, selecione quais camadas de software deseja adicionar na imagem, a depender da aplicação/destino da TV Box (navegar com o teclado, e clicar com mouse para selecionar).



12 - Após o passo 11, a imagem será compilada com base nas configurações escolhidas. Esta etapa, no entanto, leva várias horas para completar (geralmente 4, 5, 6 horas). Ao final da compilação, buscar a imagem em: /home/sel/build/outputs/image.

OBS. Não é necessário aguardar no caso desta atividade prática, pois será usado uma imagem já criada anteriormente para ser viável a conclusão desta atividade. Nesse caso, pode-se cancelar essa operação e continuar seguindo o roteiro acima. No entanto, é importante ter executado estes passos para conhecimento da ferramenta.



```
sel@host: ~/build
[+] Configuration prepared for BOARD build [ khadas-viml.conf ]
[+] Repeat Build Options (early) [ ./compile.sh build BOARD=khadas-viml BRANCH=current BUILD_DESKTOP=yes BUILD_MINIMAL=no DESKTOP_APPGROUPS_SELECTED='3dsupport chat editors multimedia' DESKTOP_ENVIRONMENT=gnome DESKTOP_ENVIRONMENT_CONFIG_NAME=config_base KERNEL_CONFIGURE=no RELEASE=bullseye ]
[+] Checked directory OK for mount options [ /armbian/.tmp ('main temporary dir') ]
[+] Preparing [ host ]
[+] Running in container [ Adding provisions for container building ]
[+] Ignoring toolchains [ SKIP_EXTERNAL_TOOLCHAINS: yes ]
[+] Aggregating packages [ rootfs ]
[+] Python3 version [ 3.10.12 - 'Python 3.10.12' ]
[+] pip3 version [ 22.0.2: 'pip 22.0.2 from /usr/lib/python3/dist-packages/pip (python 3.10)' ]
[+] Using cached pip packages for Python tools [ 85fcd277f85b8937fcd206c8380aaf0ac0d675d366d013cacc9851ab5dda88f3 ]
[+] Summary: Aggregation: debootstrap: 9; rootfs: 114; image: 0; desktop: 79; apt-sources: 0; desktop_appgroups: 4; hash: 520d2cd87b2d8fed
[+] artifact [ uboot :: uboot() ]
[+] Producing new & caching [ GIT INFO UBOOT ]
[+] Fetching SHA1 of 'tag' 'refs/heads/v2022.07^{ }' [ https://github.com/u-boot/u-boot ]
```