1. Introdução

O *Yocto Project* e o *Buildroot* podem ser definidos como **sistemas de build** para criação e distribuição de sistemas Linux embarcado. Durante o desenvolvimento de um sistema pode-se optar por utilizar distribuições Linux (Bootloader, Linux Kernel, Bibliotecas, Serviços e Aplicações) prontas ou customizadas, ambas contêm seus prós e contras que devem ser cuidadosamente avaliadas.

Desta forma, a Figura 1 abaixo ilustra os elementos de uma distribuição Linux genérica:

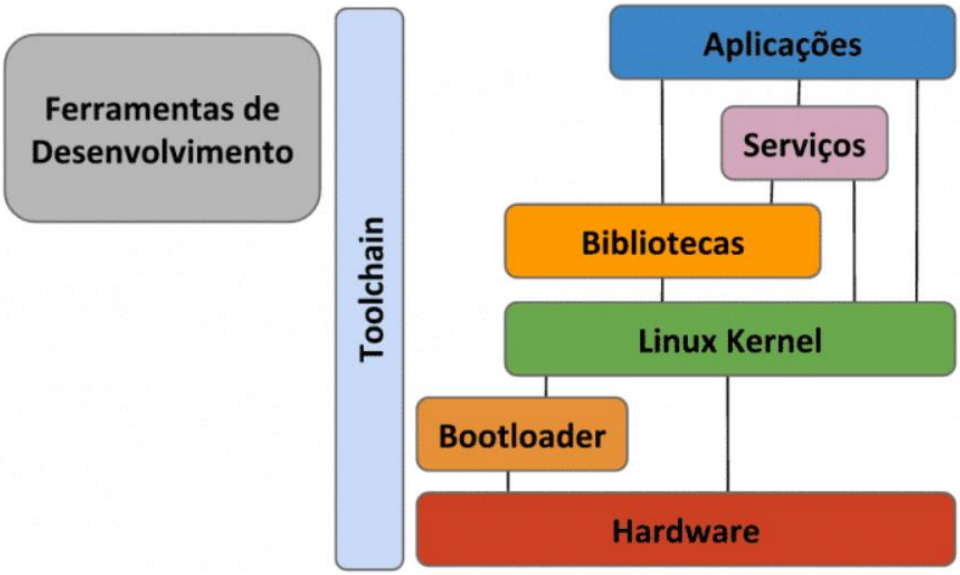

Figura 1 – Elementos de uma distribuição Linux Embarcado.

2. *Yocto Project*

É atualmente o sistema de Build mais completo disponível, liderado pela The Linux Foundation e recebe suporte de vários fabricantes de semicondutores. As principais características deste sistema são:

- Rápida validação e desenvolvimento;
- Suporte para as arquiteturas x86, ARM, MIPS e PowerPC;
- Baseado no Openembedded;
- Estruturado em camadas;
- Releases semestrais;
- SDK integrado com Eclipse e Qt Creator;
- Histórico no git;
- Filtro de licenças;
- Suporte às últimas versões;
- Suporta os pacotes rpm, deb e ipk;
- Host isolado do ambiente de Build;
- Interface gráfica para controle de Build;
- Extensa documentação.

2.1 Configuração e compilação

Para realizar a configuração do Yocto siga os passos abaixo:

- Clone o poky com a linha de comando “git clone –b jethro git://git.yoctoproject.org/poky”;
- Clone o Yocto seguindo a mesma lógica com a linha de comando “git clone git://git.yoctoproject.org/meta-raspberrypi” no diretório poky;
- Inicie o ambiente de compilação “source oe-init-build-env”;
- Adicione o arquivo de configuração bblayers.conf “vim conf/bblayers.conf” e configure de acordo com a arquitetura que será utilizada;
- Compile a imagem utilizando o bitbake, no exemplo dado utilize o comando “bitbake rpi-basic-image”;
- Copie a imagem gerada no diretório temporário “tmp/deploy/images” para a memória do sistema de destino.

Após estes passos o sistema já será iniciado, porém sem interface gráfica.

3. Buildroot

Ferramenta utilizada para auxiliar e automatizar a criação de distribuições Linux Embarcado. Para isso ele é capaz de realizar a cross-compilação para a arquitetura de destino. Como o resultado do build é apenas a imagem do sistema, o Buildroot é chamado de gerador de firmware, desta forma sempre que desejar atualizar o sistema é necessário criar toda a imagem novamente, isso evita que falhas danifiquem o sistema por completo.

3.1. Configuração, compilação e correção do console

Primeiramente deve ser feito o download do Buildroot pelo site oficial do projeto e do kernel para a arquitetura de destino. Deve-se criar uma pasta chamada “dl” no diretório principal do Buildroot e mover a imagem do kernel para a mesma.

Para realizar a configuração do Buildroot execute o comando “make menuconfig” e configure de acordo com a Figura 2 abaixo:

```
Target Architecture
    [*] arm

Target Architecture Variant
    [*] arm920t

Target ABI:
    [*] EABI

Toolchain
    Kernel Headers
        (X) Local Linux snapshot

Target Filesystem options
    [*] jffs2 root filesystem
    Flash type
        (X) NAND flash with 2kB page and 128kB erase
Kernel
    [*] Linux Kernel
        Kernel version
            (X) Same as toolchain kernel headers
    Defconfig name
        mini2440
```

Figura 2 - Configuração do Buildroot para gerar o toolchain e as imagens do kernel e do rootfs.

Como exemplo foi feita a configuração para uma placa Mini2440 com arquitetura ARM.

Para realizar a compilação basta usar o comando “make”. A compilação será feita no diretório “output” e as pastas build, host, images, staging, stamps, target e toolchain serão criadas.

A correção do console é realizada comentando a linha “ttyS0” e adicionando outra linha “ttySAC0” no arquivo “output/target/etc/inittab” conforme a Figura 3 seguinte.

```
#ttyS0::respawn:/sbin/getty -L ttyS0 115200 vt100 # GENERIC_SERIAL
ttySAC0::respawn:/sbin/getty -L ttySAC0 115200 vt100 # GENERIC_SERIAL
```

Figura 3 - Correção do console.

4. Principais diferenças entre *Yocto Project* e *Buildroot*

Primeiramente podemos citar a diferença das ferramentas de compilação: o Buildroot é baseado em duas ferramentas chamadas Kconfig, que diz para o Buildroot o que nós queremos fazer e o Make para fazer o build do sistema. Por outro lado, o Yocto utiliza apenas uma ferramenta chamada Bitbake que contém uma linguagem própria e sua configuração é feita através de arquivos originais.

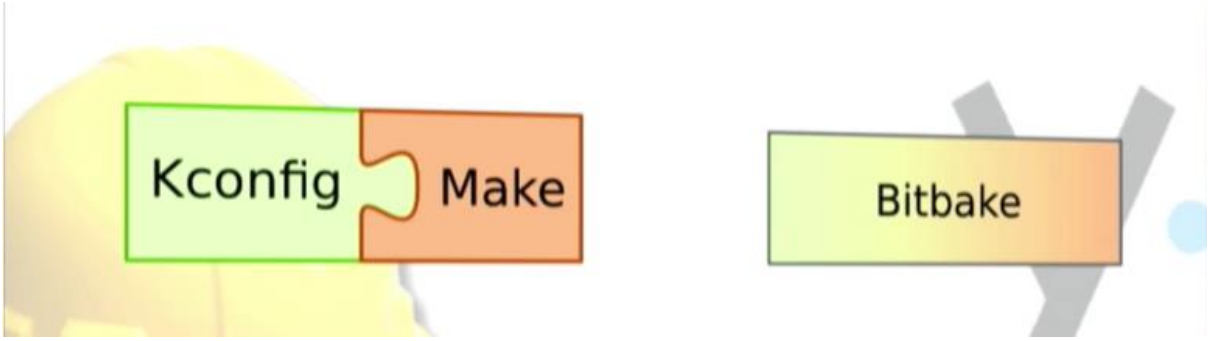

Figura 4 – Ferramentas de compilação do Buildroot e Yocto Project.

Além disso, o Buildroot é focado para sistemas embarcados e utiliza um sistema raiz completo de arquivos (root FS), já o Yocto trabalha com um sistema de pacotes (ipk, dpkg, rpm), ou seja, podemos utilizar o comando *apt install* para realizar a instalação dos pacotes que desejamos utilizar.

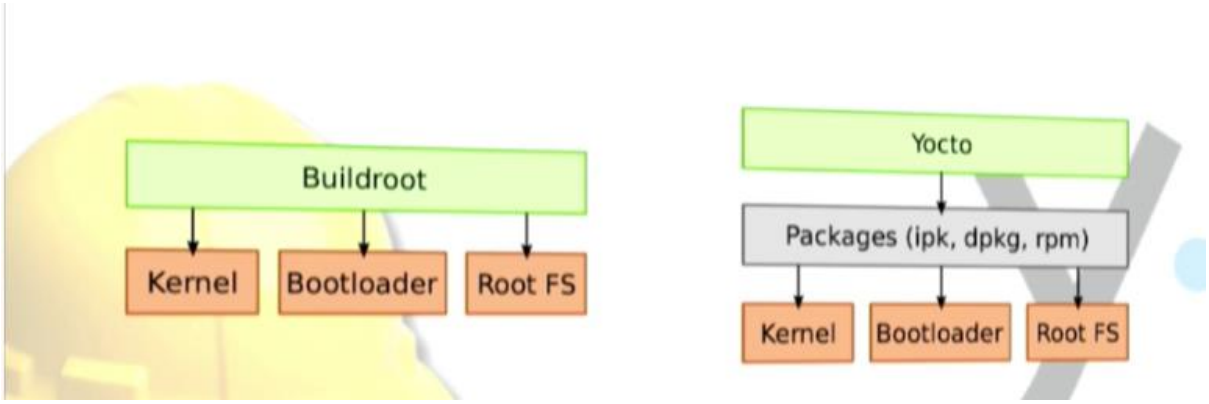

Figura 5 – Root filesystem vs Distribuição.

A complexidade no desenvolvimento de ambos também se difere, por um lado o Buildroot possui toda sua lógica (Makefiles) escrita em aproximadamente 1000 linhas de código, já o Yocto utiliza o Bitbake que é escrito em um programa Python de 60000 linhas e as “receitas” são combinações de linguagem Python, scripts de Shell e a linguagem específica do Bitbake.

5. Qual escolher?

A escolha de qual sistema escolher deverá ser feita com base no objetivo do projeto que será desenvolvido. O Buildroot nos entrega um sistema simples, basicamente apenas a imagem do sistema que deve ser posteriormente complementada. Desta forma temos um sistema de compilação rápida que não é desenvolvido para nenhum propósito específico. Por outro lado, o Yocto se mostra um sistema versátil que pode ser utilizado em uma grande gama de projetos, contém aproximadamente 8400 pacotes disponíveis no total.

Portanto, a escolha será baseada nos seguintes aspectos:

- Tamanho do sistema raiz de arquivos, caso necessite ser pequeno deve-se optar pelo uso do Buildroot;
- Necessidade de pacotes de gerenciamento, o Yocto será bastante útil neste caso;
- Projetos de baixo orçamento e recursos limitados o Buildroot é o suficiente;
- Grandes projetos e sistemas complexos o Yocto acaba sendo uma boa opção.

Vale destacar que não basta analisar apenas um destes aspectos, deve-se ter bom senso na hora da escolha, mesmo que o seu projeto não se encaixe totalmente em um ou outro sistema de build.