Linux yocto – зачем, почему и как

Разработал: Максимов А.Н. начальник отдела перспективных разработок ЦПР РТСофт
Версия 0.1 2020

RTSoft
SOFTWARE FOR INTELLIGENT THINGS

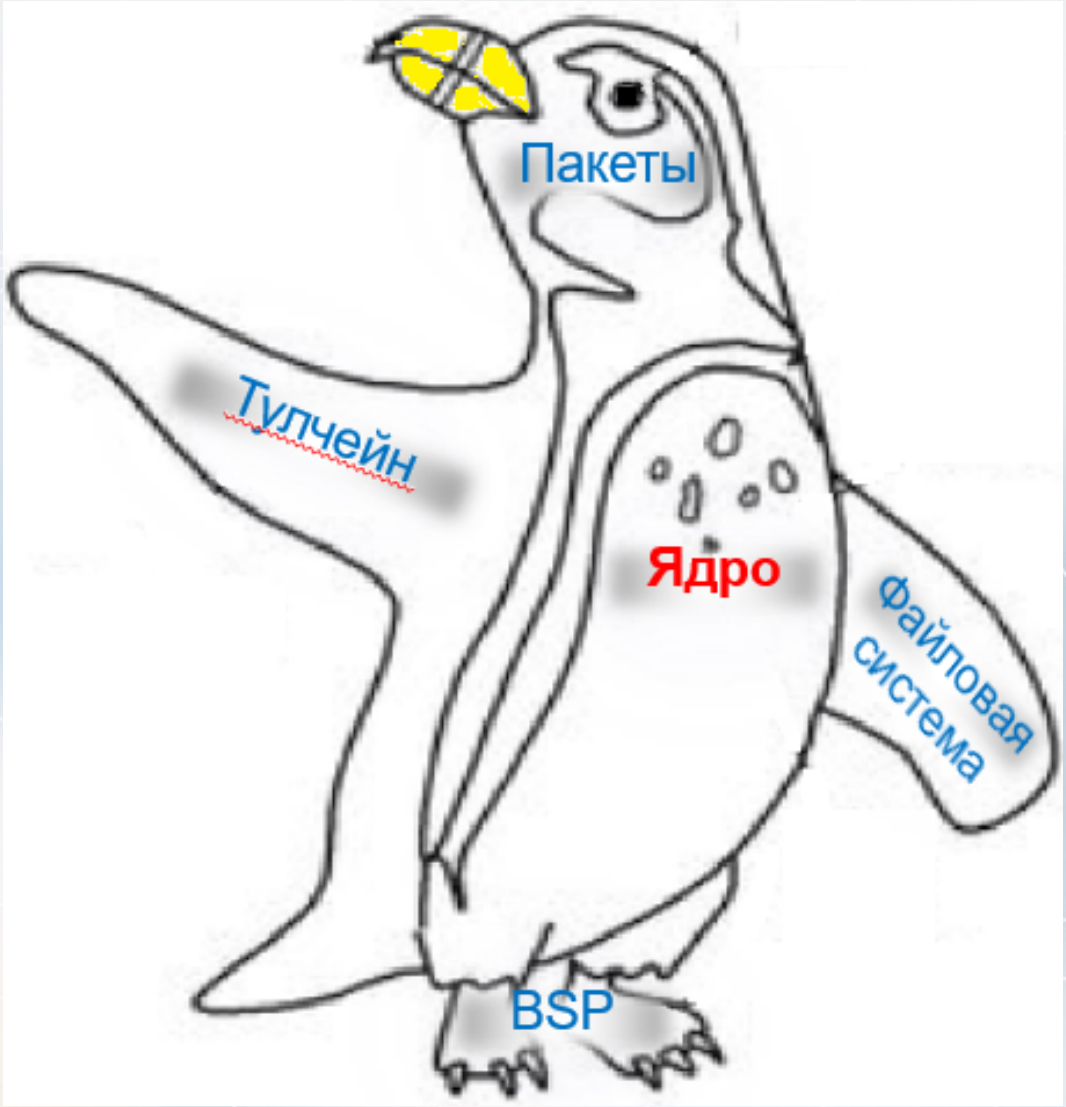
Содержание

- Обзор Yocto Project
- Система сборки
- Рецепты
- Использование эмуляции
- Слои

RTSoft

Что включает Linux

- Тулчейн ☺ (binutils, gcc и т.д
- Ядро linux
- Файловая система
- Набора пакетов



RTSoft

Что включает разработка встраиваемого решения

- Разработка BSP, портация загрузчика и настройка ядра Linux
- Выбор и настройка пакетов и системных компонент
- Разработка и настройка механизмов обновления и восстановления
- Разработка прикладного приложения

RTSoft

Какой Linux взять

- Бинарный дистрибутив (Debian, Ubuntu, fedora и др.)

- Сборочную систему (yocto, buildroot, ptxdist и др.)

- Собрать все самому (см. http://www.linuxfromscratch.org/)

RTSoft

Вот ребята разрабатывали умное зеркало

Ссылка на проект:

https://www.youtube.com/watch?v=WQR0fv9C5dU

Взяли Operating System (2020) Raspbian Buster

Почемум это не хорошо для продуктовой разработки?

RTSoft

Какой Linux взять

- Бинарный дистрибутив (Debian, Ubuntu, fedora и др.)

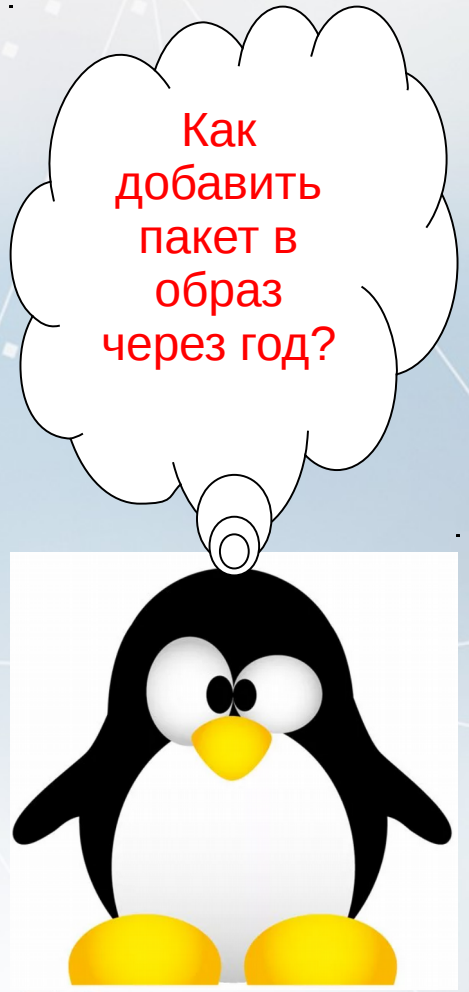
- Сборочную систему (yocto, buildroot, ptxdist и др.)

- Собрать все самому (см. http://www.linuxfromscratch.org/)

RTSoft

“Debian” в embedded, как это работает?

- Взять существующий образ файловой системы или установить на целевую платформу
- Исключить (деинсталлировать) ненужные пакеты из системы
- Добавить нужные пакеты
- Собрать прикладные программы нативно в целевой системе
- Сконфигурировать параметры целевой системы (порядок загрузки, пользователей и т.д.)
- Создать при помощи dd “Golden Master”образ системы ☺



“Debian” плюсы и минусы для embedded

Дополнительная информация тут:

https://www.youtube.com/watch?v=iDllXa8SzUg

Плюсы:

-Можно получить результаты быстро
-Много пакетов и их просто ставить
-Долгая поддержка для LTS версии

Минусы:

-Трудности кастомизации
-Трудно оптимизировать время загрузки и размер
-Трудно собрать все из исходников
-Большой образ и нет стандартного способа сборки ☺
-Компиляция на целевой платформе
-Много обязательных зависимостей
-Могут остаться следы разработчика (логи, пароли и т.д.)
-Не для всех архитектур

RTSoft

Yocto project

Yocto Project это опенсорс проект нацеленный на создание встраиваемых Linux систем.

Репозитории исходного кода:

http://git.yoctoproject.org

http://git.openembedded.net

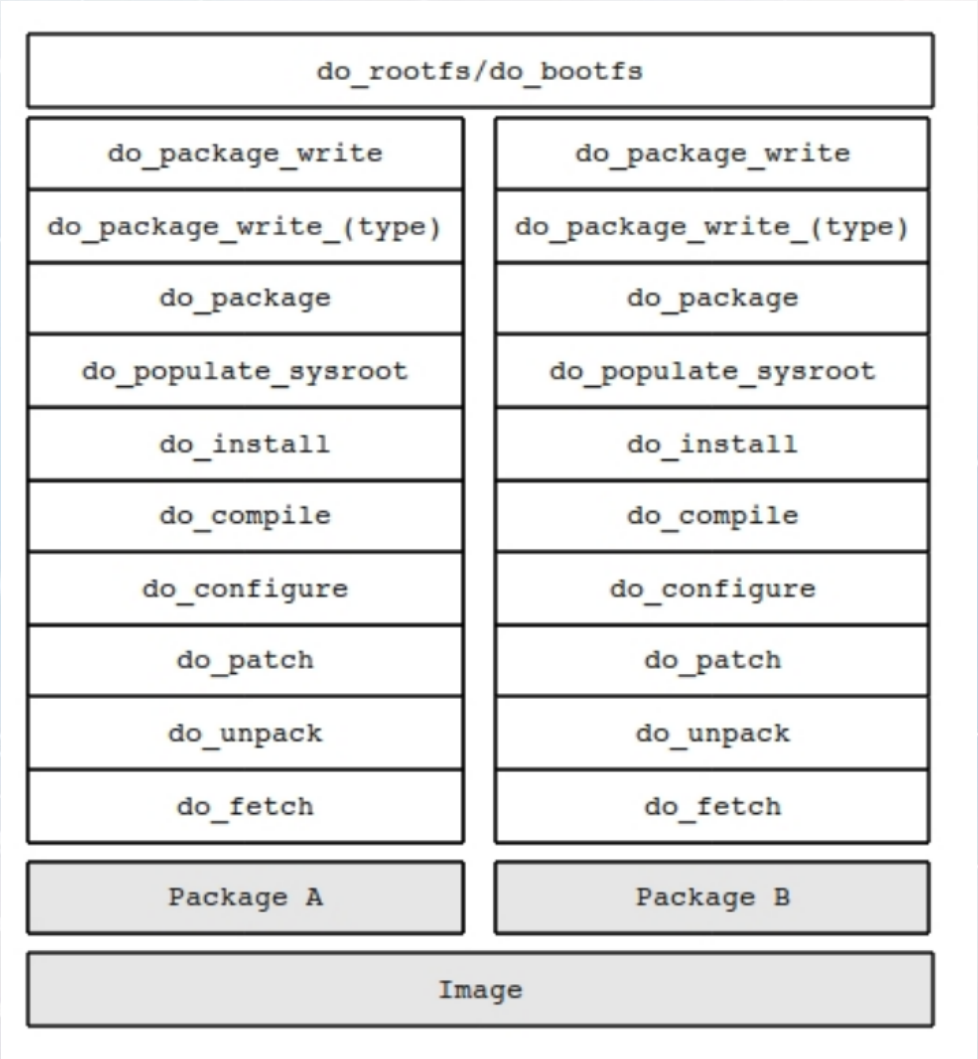
Документация:

http://www.yoctoproject.org/documentation

Yocto Project развивается под потранажем Linux Foundation

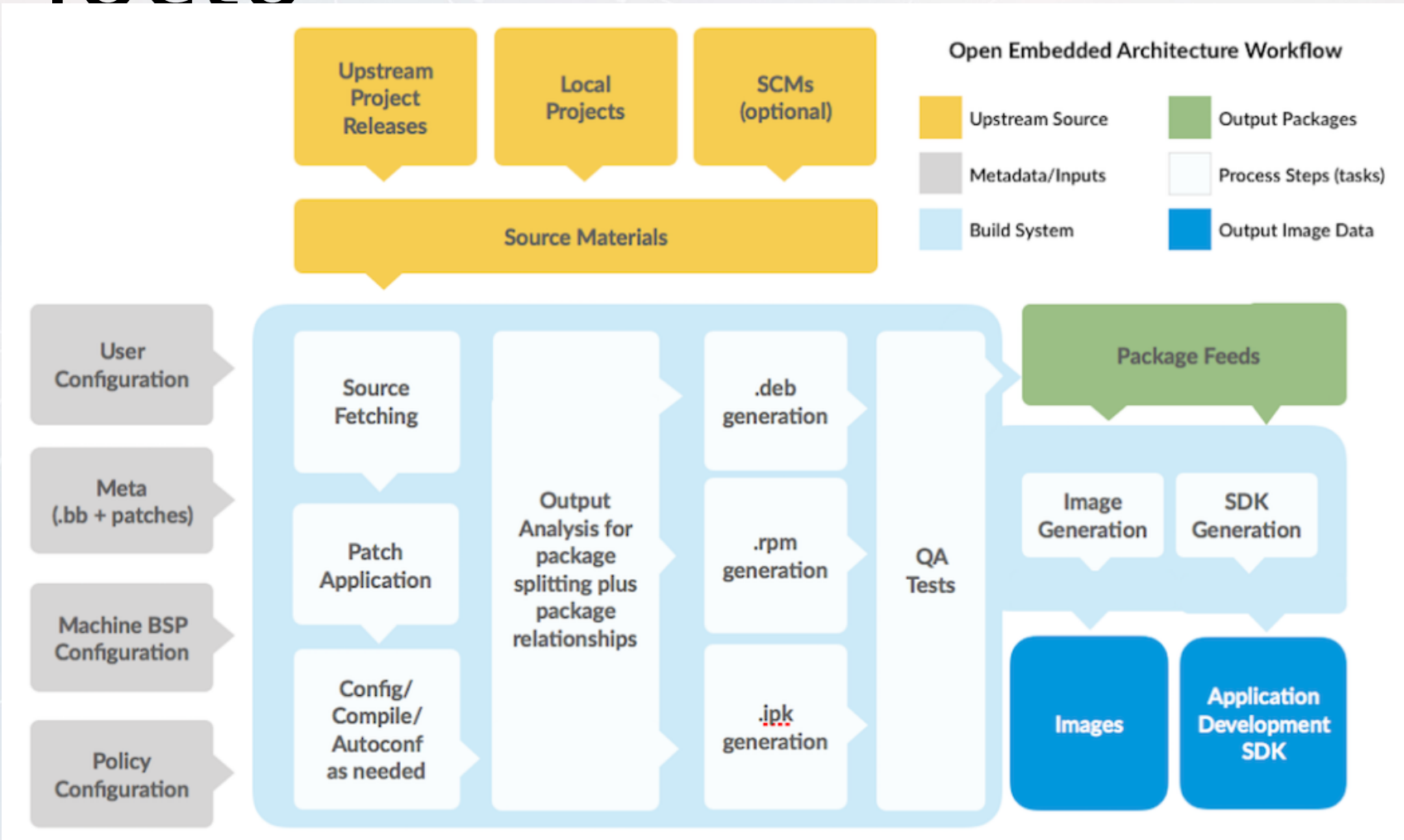
RTSoft

Выполнение задач в yocto



RTSoft

Общая схема работы Yocto



https://www.yoctoproject.org/software-overview/

RTSoft

Основные компоненты yocto

BitBake – система сборки. Она обрабатывает конфигурационные файлы и рецепты. В процессе обработки скачивается исходный код, собираются указанные в рецептах приложения и формируется образ файловой системы.

OpenEmbedded-Core – набор рецептов, классов и слои. совместно поддерживается проектами Yocto Project и OpenEmbedded. Слой, отделяющий проект Yocto Project от проекта OpenEmbedded, называется meta-yocto и содержит конфигурацию дистрибутива Poky, а также базовый набор эталонных BSP-пакетов.

Poky – эталонная система сборки в рамках проекта yocto. Содержит набор инструментов для создания нового дистрибутива на базе. Yocto Project.

https://www.yoctoproject.org/software-overview/

RTSoft

Что нажно для работы yocto

sudo apt-get install wget git-core unzip make gcc g++ build-essential subversion sed autoconf automake texi2html texinfo coreutils diffstat python-pysqlite2 docbook-utils libsdl1.2-dev libxml-parser-perl libgl1-mesa-dev libglu1-mesa-dev xsltproc desktop-file-utils chrpath groff libtool xterm gawk fop

А еще 50-100 Gb места на диске

Доступ к Internet

Как можно более быстрый CPU или много терпения :)

https://www.codeproject.com/Articles/1079666/How-to-run-Yocto-Linux-OS-in-a-virtual-machine

RTSoft

Как попробовать собрать образ в yocto

Предполагаем, что необходимые компоненты для сборки у нас есть :)

Загрузим систему сборки poky:

```
git clone git://git.yoctoproject.org/poky.git
git checkout yocto-2.6.2
```

```
source ./poky/oe-init-build-env
bitbake core-image-minimal
```

https://www.yoctoproject.org/software-overview/

RTSoft

Результаты сборки

ls ./tmp/deploy/images/qemux86/

bzImage--4.18.27+git0+9e348b6f9d_62f0a3acff-r0-qemux86-20200703160242.bin
core-image-minimal-qemux86-20200703160242.qemuboot.conf
core-image-minimal-qemux86-20200703160242.rootfs.ext4
core-image-minimal-qemux86-20200703160242.rootfs.manifest
core-image-minimal-qemux86-20200703160242.rootfs.tar.bz2
core-image-minimal-qemux86-20200703160242.testdata.json
modules--4.18.27+git0+9e348b6f9d_62f0a3acff-r0-qemux86-20200703160242.tgz

RTSoft

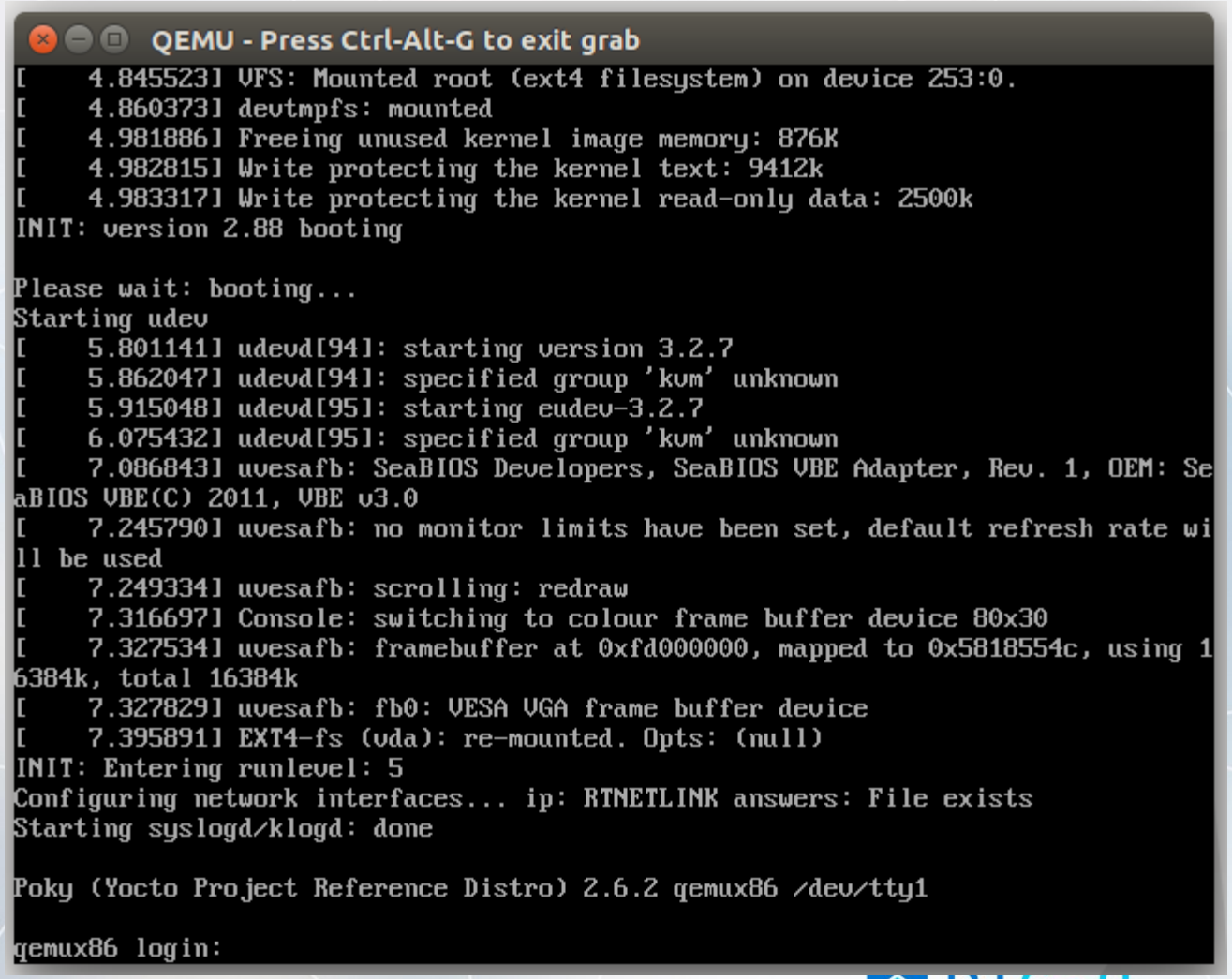
Запуск yocto в qemu

cd build

ls ./tmp/deploy/images/qemux86/

runqemu qemux86

Войдите, как root



RTSoft

Если хочется GUI для конфигурирования образа

Hob – устарел – используем toaster :)

sudo apt-get install python3-pip

$ pip3 install --user -r /opt/yocto/poky/bitbake/toaster-requirements.txt

cd /opt/yocto/poky$ source oe-init-build-env$ source toaster start

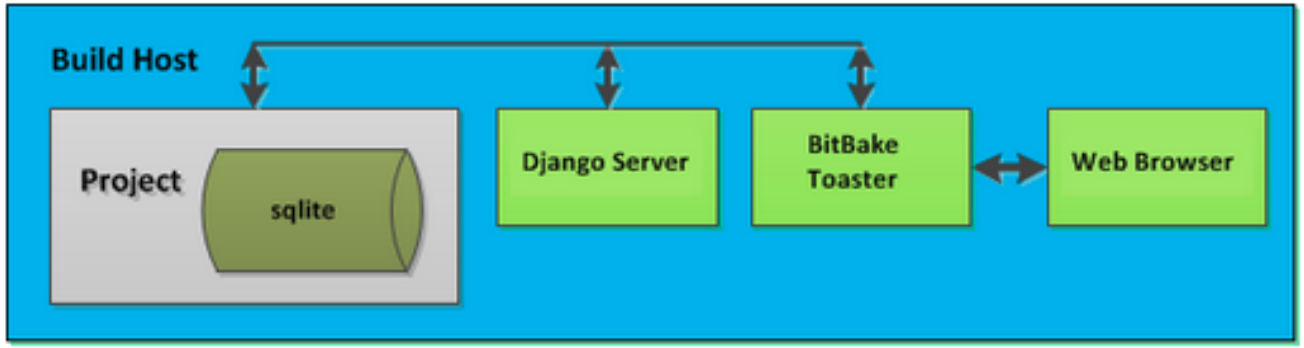
To access the web user interface, go to http://127.0.0.1:8000

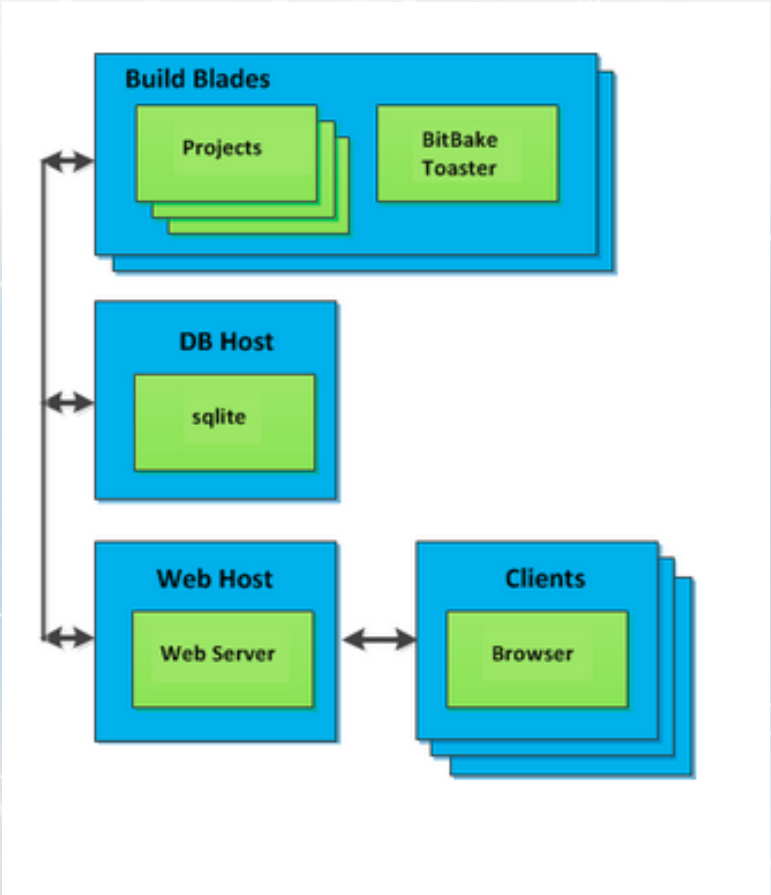
http://127.0.0.1:8000/admin.

https://www.yoctoproject.org/docs/latest/toaster-manual/toaster-manual.html

RTSoft

Архитектура Toaster





https://www.yoctoproject.org/docs/latest/toaster-manual/toaster-manual.html

RTSoft

Задание 1

1. Собрать образ core-image-minimal
2. Запустить образ в qemu

RTSoft

Добавление приложения

main.c

```
#include <stdio.h>
void main() {
    printf("Hello World!!\n");
}
```

Makefile

```
obj = main.o
target = hello-world
all: $(obj)
	${CC} $(obj) -o $(target)
%.o:%.c
	${CC} -c $^ -o $@
.PHONY: clean
clean:
	rm -rf $(obj) $(target)
```

RTSoft

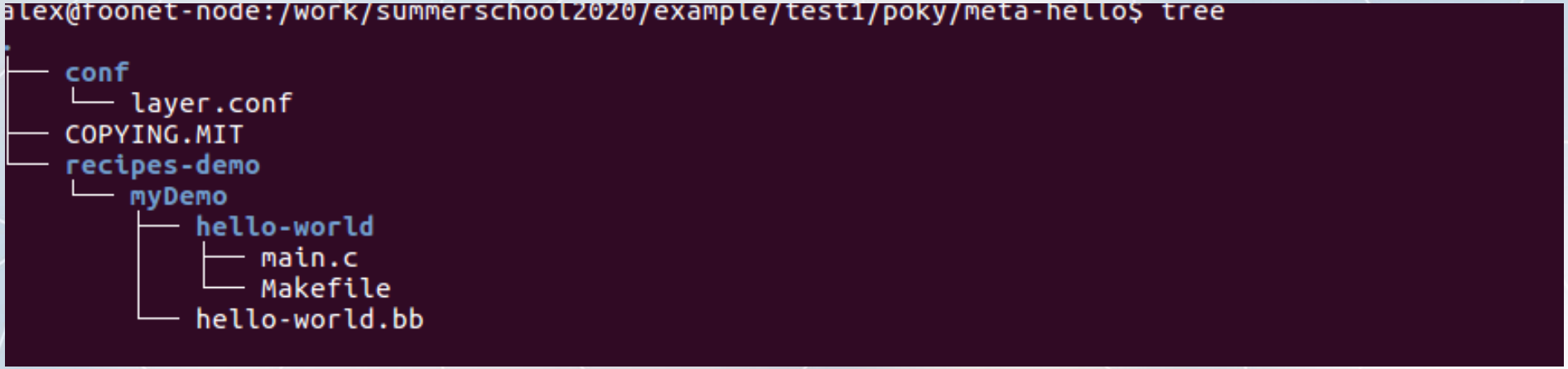
Добавление рецепта

```
$ cd ../meta-hello
$ mkdir -p recipes-demo/myDemo/hello-world
$ cd recipes-demo/myDemo
$ touch hello-world.bb
$ cd hello-world
$ touch main.c Makefile
```

RTSoft

Добавление рецепта вручную :)

```
$ cd ../meta-hello
$ mkdir -p recipes-demo/myDemo/hello-world
$ cd recipes-demo/myDemo
$ touch hello-world.bb
$ cd hello-world
$ touch main.c Makefile
```



RTSoft

Собственно рецепт hello-world.bb

```
SUMMARY = "Hello World Demo"
SECTION = "apps"
LICENSE = "CLOSED"
APP_NAME = "hello-world"
localdir = "/usr/local"
bindir = "${localdir}/bin"
TARGET_CC_ARCH += "${LDFLAGS}"
SRC_URI = "file://main.c \
           file://Makefile \
               "
S = "${WORKDIR}"
do_compile() {
    make -f Makefile
}
do_install () {
    install -m 0755 -d ${D}${localdir}
    install -m 0755 -d ${D}${bindir}
    cd ${S}
    install -m 0755 ${APP_NAME} ${D}${bindir}
}
FILES_${PN}-dev = ""
FILES_${PN} = "${bindir}/*"
```

RTSoft

Конфигурация layer.conf

```
# We have a conf and classes directory, add to BBPATH
BBPATH .= ":${LAYERDIR}"

# We have recipes-* directories, add to BBFILES
BBFILES += "${LAYERDIR}/recipes-*/*/*.bb \
      ${LAYERDIR}/recipes-*/*/*.bbappend"

BBFILE_COLLECTIONS += "example"
BBFILE_PATTERN_example = "^${LAYERDIR}/"
BBFILE_PRIORITY_example = "6"
```

RTSoft

Конфигурация COPYING.MIT

RTSoft

Собственно сборка рецепта

Добавить layer в конфигурацию build/conf/bblayers.conf

```
# POKY_BBLAYERS_CONF_VERSION is increased each time build/conf/bblayers.conf
# changes incompatibly
POKY_BBLAYERS_CONF_VERSION = "2"
BBPATH = "${TOPDIR}"
BBFILES ?= ""
BBLAYERS ?= " \
  /work/summerschool2020/example/test1/poky/meta \
  /work/summerschool2020/example/test1/poky/meta-poky \
  /work/summerschool2020/example/test1/poky/meta-yocto-bsp \
  /work/summerschool2020/example/test1/poky/meta-hello \
"
```

Выполнение рецепта:

```
. ./poky/oe-init-build-env
bitbake hello-world
```

RTSoft

Собственно сборка рецепта

Результат tmp/deploy/rpm/i586/hello-world-1.0-r0.i586.rpm

```
alex@foonet-node:/work/summerschool2020/example/test1$ bitbake hello-world
WARNING: Layer example should set LAYERSERIES_COMPAT_example in its conf/layer.conf file to list the core layer names it is compatible with.
WARNING: Layer example should set LAYERSERIES_COMPAT_example in its conf/layer.conf file to list the core layer names it is compatible with.
WARNING: Layer example should set LAYERSERIES_COMPAT_example in its conf/layer.conf file to list the core layer names it is compatible with.
WARNING: Layer example should set LAYERSERIES_COMPAT_example in its conf/layer.conf file to list the core layer names it is compatible with.
Loading cache: 100% |################################################################################################################################| Time: 0:00:00
Loaded 1268 entries from dependency cache.
NOTE: Resolving any missing task queue dependencies

Build Configuration:
BB_VERSION           = "1.40.0"
BUILD_SYS            = "x86_64-linux"
NATIVELSBSTRING      = "universal"
TARGET_SYS           = "i586-poky-linux"
MACHINE              = "qemux86"
DISTRO               = "poky"
DISTRO_VERSION       = "2.6.2"
TUNE_FEATURES        = "m32 i586"
TARGET_FPU           = ""
meta
meta-poky
meta-yocto-bsp       = "HEAD:e7f0177ef3b6e06b8bc1722fca0241fef08a1530"
meta-example         = "master:8148d045406cbb2b66a6848ea93371808d206323"
meta-hello           = "HEAD:e7f0177ef3b6e06b8bc1722fca0241fef08a1530"

Initialising tasks: 100% |###########################################################################################################################| Time: 0:00:00
Sstate summary: Wanted 0 Found 0 Missed 0 Current 74 (0% match, 100% complete)
NOTE: Executing SetScene Tasks
NOTE: Executing RunQueue Tasks
NOTE: Tasks Summary: Attempted 446 tasks of which 446 didn't need to be rerun and all succeeded.

Summary: There were 4 WARNING messages shown.
```

RTSoft

Добавить в образ :)

Добавить в файл build/conf/local.conf

IMAGE_INSTALL_append = "hello-world"

Собрать образ:

bitbake core-image-minimal

Запустить образ:

runqemu qemux86

RTSoft

Как добавить модуль ядра

```
$ tree
.
├── files
│   ├── COPYING
│   ├── hellokernel.c
│   └── Makefile
└── kernmodule.bb
```

cp meta-skeleton/recipes-kernel/hello-mod/files/COPYING meta-mod/kernmodule/files/

https://wiki.koansoftware.com/index.php/Howto_build_a_kernel_module_out_of_the_kernel_tree

RTSoft

Как добавить модуль ядра

```
#
# Yocto recipe to build a kernel module out of the kernel tree
# kernmodule.bb
# Marco Cavallini - KOAN sas - www.koansoftware.com
#

DESCRIPTION = "Hello kernel module out of the kernel tree"
SECTION = "examples"
LICENSE = "GPLv2"
LIC_FILES_CHKSUM = "file://COPYING;md5=12f884d2ae1ff87c09e5b7ccc2c4ca7e"
PR = "r0"

inherit module

SRC_URI =   "file://hellokernel.c \
                        file://Makefile \
                        file://COPYING \
                        "

S = "${WORKDIR}"
```

RTSoft

Литература

1. Getting Started with the Yocto Project - New Developer Screencast Tutorial https://www.youtube.com/watch?v=zNLYanJAQ3s
2. https://elinux.org/images/6/62/Building-Container-Images-with-OpenEmbedded-and-the-Yocto-Project-Scott-Murray-Konsulko-Group-1.pdf
3. Yocto Project Quick Start https://www.yoctoproject.org/docs/1.7.1/yocto-project-qs/yocto-project-qs.html
4. Создание специальных дистрибутивов Linux для встраиваемых систем с помощью Yocto Project
5. Yocto Project first steps https://ocw.cs.pub.ro/courses/iot/labs/09

RTSoft