# （1）基于Yocto构建嵌入式u-boot，内核，文件系统

Yocto是什么？简单地概括，它是一个工具，可以用来构建u-boot，kernel，文件系统，交叉编译工具链，等等。Yocto提供了一套完整的全面的嵌入式Linux移植解决方案。使嵌入式Linux系统移植告别了以前通过源码一步步移植的刀耕火种年代。Yocto解决了嵌入式Linux行业内各自为政的乱象，它为芯片厂家，操作系统，和设备厂商之间的协作提供广泛一致的行业支持。

对于如何基于Yocto构建 i.MX6UL 的u-boot，kernel，rootfs，NXP官方提供了一份指导手册《i.MX\_Yocto\_Project\_User's\_Guide.pdf》，参考NXP官方提供的指导手册，我们开始基于Yocto构建i.MX6UL 的u-boot，kernel，和rootfs文件系统。

在构建系统之前，需要进行一系列的准备工作，构建前准备：

1. 主机电脑的Linux版本：ubuntu16.04.2 LTS

2. 主机电脑建议配置：Intel i7 四核以上处理器，8G内存，SSD硬盘更佳。

3. 主机用于搭建Yocto环境的磁盘空间需要保留大于150GB的空间。

4. 主机能正确访问互联网（最好网速比较快）。

5. 需要在普通用户的环境下进行构建。

**以下是整个Yocto构建过程，构建过程中，如遇到权限问题，使用sudo工具解决**

（1）在主机Linux上安装Yocto构建过程中需要的依赖包，执行以下命令。

$ sudo apt-get install gawk wget git-core

$ sudo apt-get install diffstat unzip texinfo gcc-multilib

$ sudo apt-get install build-essential chrpath socat libsdl1.2-dev

$ sudo apt-get install libsdl1.2-dev

$ sudo apt-get install xterm sed cvs

$ sudo apt-get install subversion coreutils texi2html

$ sudo apt-get install docbook-utils python-pysqlite2

$ sudo apt-get install help2man make gcc g++

$ sudo apt-get install desktop-file-utils

$ sudo apt-get install libgl1-mesa-dev libglu1-mesa-dev mercurial

$ sudo apt-get install autoconf automake groff

$ sudo apt-get install curl lzop asciidoc

$ sudo apt-get install u-boot-tools

安装软件依赖包这一步骤，只要网络环境正常，一般不会出现问题。如果某些依赖包安装失败，可以尝试更换软件源，笔者默认使用ubuntu16.04的官方源。

（2）Yocto工程是使用repo工具进行版本管理的，因此，需要对ubuntu主机的repo工具进行安装和配置。整个Yocto的构建 在目录 /opt/yocto\_project下进行。注意，以下操作需要在普通用户下进行，如果遇到需要权限的问题，就使用sudo命令进行解决。执行以下命令，安装repo工具：

$ mkdir  /opt/yocto\_project/bin -p

$ chmod -R  777  /opt/yocto\_project

$ PATH=/opt/yocto\_project/bin:$PATH

$ curl  https://mirrors.tuna.tsinghua.edu.cn/git/git-repo  >  /opt/yocto\_project/bin/repo

$ chmod  a+x  /opt/yocto\_project/bin/repo

（3）repo工具安装完成后，需要设置Git版本管理工具的信息，执行以下命令：

$ git config --global user.name "your\_name"

$ git config --global user.email "your\_email"

$ git config --list

（4）对于imx6ul平台，在使用Yocto构建BSP包的过程中，会从NXP官方的Git仓库下载很多文件，并在编译过程中产生很多文件信息，所以，我们把整个构建过程的下载文件和安装文件都放在/opt/yocto\_project/fsl-release-bsp目录，方便统一进行管理。执行以下命令：

$ mkdir /opt/yocto\_project/fsl-release-bsp

$ chmod 777 -R /opt/yocto\_project/fsl-release-bsp

$ cd /opt/yocto\_project/fsl-release-bsp

$ repo init -u git://git.freescale.com/imx/fsl-arm-yocto-bsp.git -b imx-4.1-krogoth

$ repo sync -f -j4

1. 因为国内某些不可描述的原因，在repo进行同步的过程中，可能会出现网络中断或网络连接不可访问的错误，如以下错误可能会出现：

解决方法是,先单独克隆repo

$ git clone https://gerrit-googlesource.lug.ustc.edu.cn/git-repo

然后将git-repo里面的repo可执行文件复制到/opt/yocto\_project/bin目录

$ cp  git-repo/repo  /opt/yocto\_project/bin  -a

$ chmod  a+x  /opt/yocto\_project/bin/repo

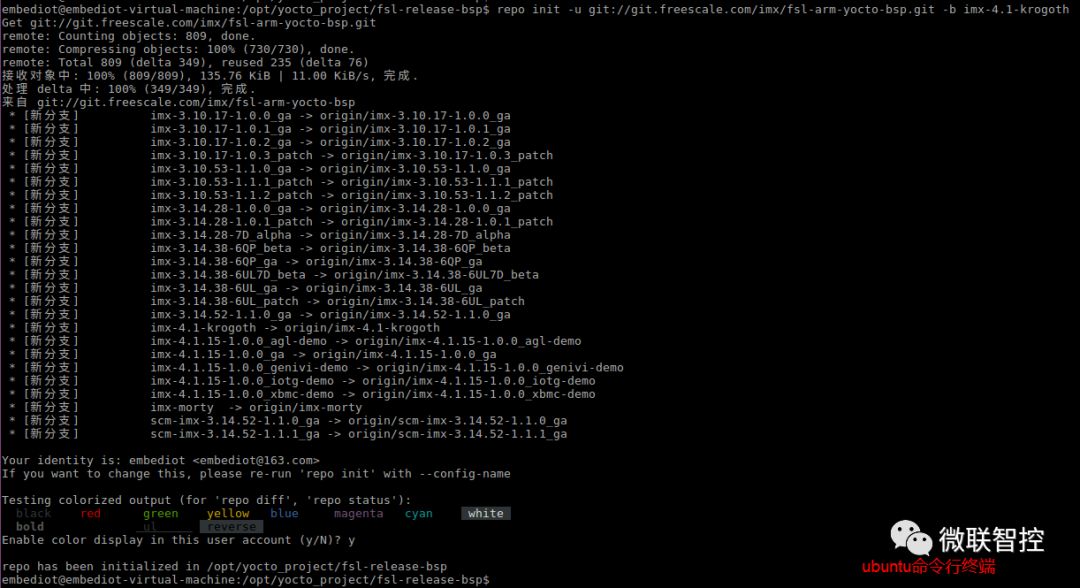
再在同步源码的工作目录fsl-release-bsp新建.repo文件夹（若已存在则不用新建），把git-repo文件夹重命名为repo文件夹，并复制到 .repo 目录下：

$ mv  git-repo  ./.repo/repo

重新执行以下命令，初始化repo仓库：

$ repo init -u git://git.freescale.com/imx/fsl-arm-yocto-bsp.git -b imx-4.1-krogoth

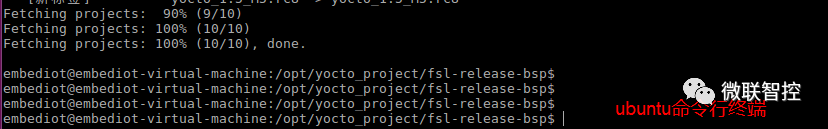
初始化成功后，如下图所示：



重新开始进行repo仓库同步：

$ repo sync -f -j8

仓库同步需要一段时间，同步完成后，如下图所示：



（6）repo同步成功后，就可以开始进行镜像构建了，在fsl-release-bsp目录下，使用以下命令格式配置镜像的构建参数：

DISTRO=<distro name> MACHINE=<machine name> source fsl-setup-release.sh -b <build dir>

**DISTRO**有以下参数可选：

        fsl-imx-x11

        fsl-imx-wayland

        fsl-imx-xwayland

        fsl-imx-fb

在这里，我们选择fsl-imx-x11这个选项，同时注意，NXP官方不再支持DirectFB。

**MACHINE**有以下参数可选：

        imx6qpsabreauto

        imx6qpsabresd

        imx6ulevk

        imx6ull14x14evk

        imx6ull9x9evk

        imx6dlsabreauto

        imx6dlsabresd

        imx6qsabreauto

        imx6qsabresd

        imx6slevk

        imx6solosabreauto

        imx6solosabresd

        imx6sxsabresd

        imx6sxsabreauto

        imx7dsabresd

在这里，我们选择imx6ulevk这个选项。

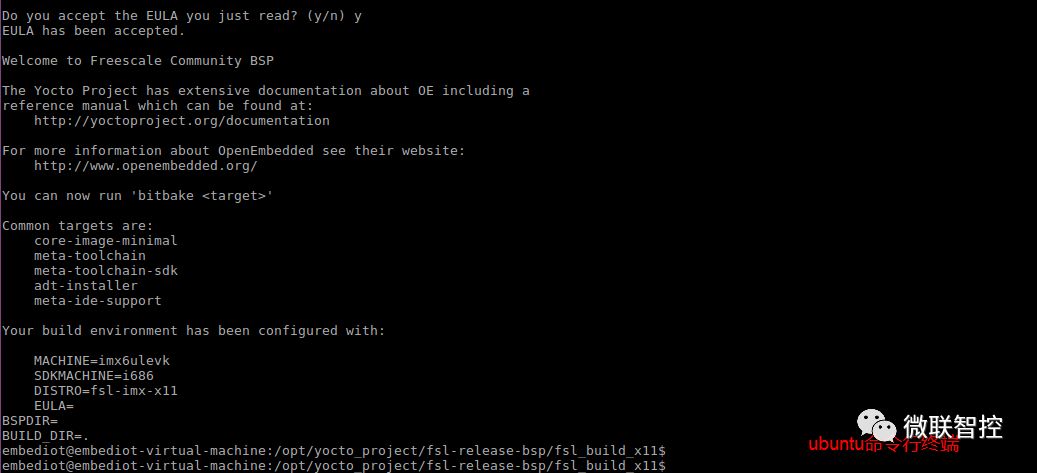
最终，配置镜像构建参数的完整命令，如下所示：

$ DISTRO=fsl-imx-x11 MACHINE=imx6ulevk source fsl-setup-release.sh -b ./fsl\_build\_x11

fsl-setup-release.sh是构建脚本，如脚本不可执行，需使用chmod命令添加可执行权限。

 -b <build dir>表示构建的在指定目录。

执行完以上的配置参数的命令后，会提示需要接受EULA协议，按空格键直到协议最后，然后按“y”确认接受，最后，完成后如下图所示：

 （7）在Yocto这个系统中，使用bitbake工具来进行各种系统或第三方库的构建和安装。bitbake是一个软件组建的自动化构建工具，它可以像make命令一样控制如何构建一个系统或解决系统构建过程中的依赖问题。但bitbake又不像make依赖makefile那样单一，bitbake可以搜集和管理大量没有依赖关系的配置描述文件（一般称为配方recipe），然后根据这些描述文件自动按照正确的顺序进行构建。注意：bitbake工具只能在普通用户下进行使用，在本次配置的Yocto环境中，bitbake支持以下镜像的构建：

        core-image-minimal

        meta-toolchain

        meta-toolchain-sdk

        adt-installer

        meta-ide-support

在这里，我们需要构建一个最小的嵌入式Linux系统，因此，执行以下命令构建一个最小的嵌入式Linux系统：

        $ bitbake core-image-minimal

这个最小的嵌入式Linux系统，构建的过程非常漫长，取决于你的网速带宽和电脑配置。因为bitbake是一边下载一边编译的，构建过程如下图所示：

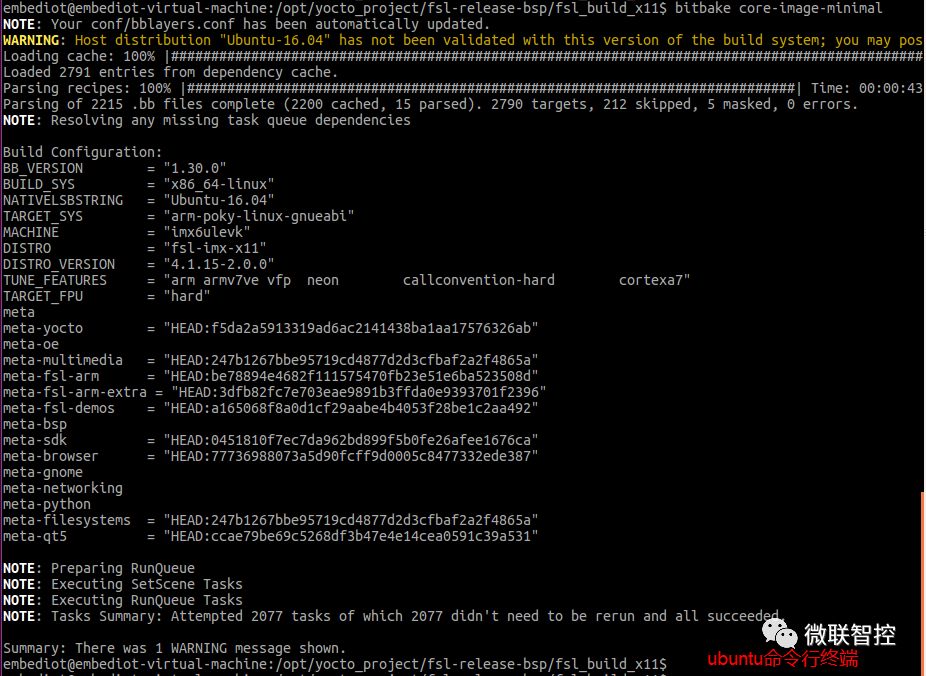


注意：如果构建过程中出现网络中断或关闭了构建的终端，只需要在fsl-release-bsp目录下执行以下命令：

$ DISTRO=fsl-imx-x11 MACHINE=imx6ulevk source fsl-setup-release.sh -b ./fsl\_build\_x11

重新配置构建的环境，然后再进入fsl\_build\_x11目录下，重新执行bitbake命令就可以继续构建了。

Yocto构建core-image-minimal成功后，如下图所示：



（8）由于各种不可描述的原因，国内的普通网络在访问外国的源码服务器时，可能会出现源码下载错误，源码下载中断等等情况，一般遇到这种情况的时候，可以先忽略其错误或警告，等整个Yocto构建完成后，再重新执行bitbake。网络环境这个因素不是每个开发者都能主导的，这个只能看运气。有条件的开发者可以不断寻找优质的网络环境。

（9）最小的嵌入式Linux系统构建完成后，在Yocto工程的构建目录里，已经包含了kernel和u-boot的源代码，以及根文件系统目录。其源码目录如下图所示：

**u-boot源码目录：**

/opt/yocto\_project/fsl-release-bsp/fsl\_build\_x11/tmp/work/imx6ulevk-poky-linux-gnueabi/u-boot-imx/2016.03-r0/git

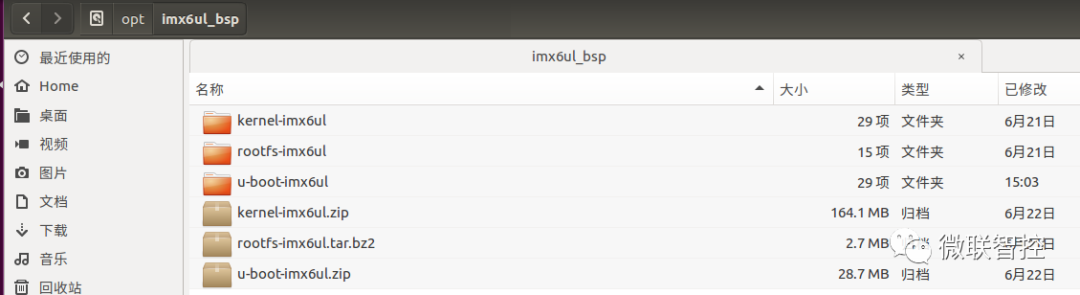
**Linux kernel源码目录：**

/opt/yocto\_project/fsl-release-bsp/fsl\_build\_x11/tmp/work-shared/imx6ulevk/kernel-source

**根文件目录：**

/opt/yocto\_project/fsl-release-bsp/fsl\_build\_x11/tmp/deploy/images/imx6ulevk/core-image-minimal-imx6ulevk-20190621012322.rootfs.tar.bz2

为了方便管理，我们把u-boot源码、Linux kernel源码和rootfs根文件目录统一存放在/opt/imx6ul\_bsp目录下，并对源码文件夹重新命名，完成后，如下图所示：



（10）使用Yocto除了能获取源码，还帮我们成功编译出各个源码的镜像文件，这些镜像可以直接烧录至官方的开发套件（imx6ul-evk）里面运行，镜像文件存放在以下目录：

/opt/yocto\_project/fsl-release-bsp/fsl\_build\_x11/tmp/deploy/images/imx6ulevk

（11）至此，imx6ul的最小系统，u-boot，kernel，rootfs已经构建完成，下一步可以基于使用Yocto构建的最小系统进行移植工作，以适配我们自己定制的开发板平台。在这个章节里面，只是简单地描述了如何使用Yocto构建i.MX6UL的最小系统，并且构建出来的i.MX6UL的各种镜像是以NXP官方开发套件（imx6ul-evk）为基础的，对于用户自定义的开发板，还需要进行一些移植工作，才能使u-boot，kernel，rootfs运行在自定义的开发板上。

（12）Yocto 是一个很强大的构建工具，其功能不仅仅是用来获取BSP源码和简单地编译源码，开发者还可以使用Yocto对其开发板添加各种第三方开发库，而不需要每次都使用原始的刀耕火种（从零开始编译源码，解决第三方依赖）方式进行开发，对开发效率的提高有很大的帮助。也可以基于Yocto的构建规则，把自己定义的开发板卡添加到Yocto的构建工程中，便于向第三方进行发布。但同时，Yocto也是一个很庞大的构建系统，里面有很多基本的构建规则需要开发者不断去熟悉才能灵活运用，遇到问题要善于使用搜索引擎去寻找答案，最好的参考教程还是Yocto Project的官方文档，开发者只有不断学习，不断提高，才能把Yocto这个工具用好，才能在开发中起到事半功倍的作用。