Introduction: Yocto Project 101

Bryan Thomas

Field Applications Engineer

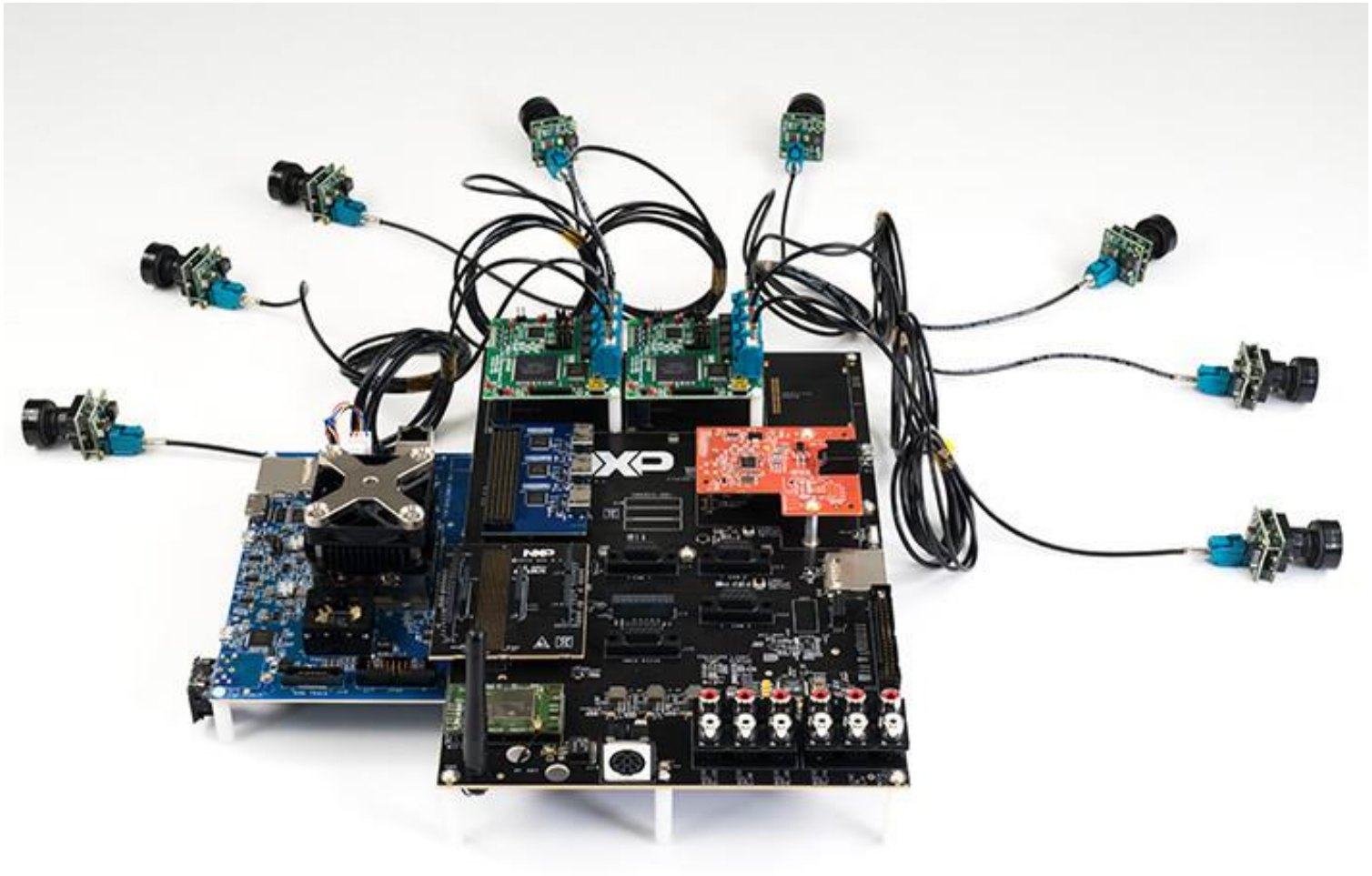
October 2019 | Session #AMF-AUT-T3891

SECURE CONNECTIONS FOR A SMARTER WORLD

Agenda

- i.MX8 Reference Boards
- Linux Introduction
- Yocto Introduction
- NXP Yocto BSP Release
- Host Machine Setup
- Yocto Setup
- Running a Yocto Build
- Using the Results
- Finding Help

i.MX8 Reference Boards



- i.MX 8QuadMax MEK
- i.MX 8QuadXPlus MEK

- i.MX 8M Evaluation Kit (consumer/industrial)

Linux Introduction

What is Linux?

Obtaining and Installing a Linux Distribution

The Terminal

Yocto Project: What is Linux?

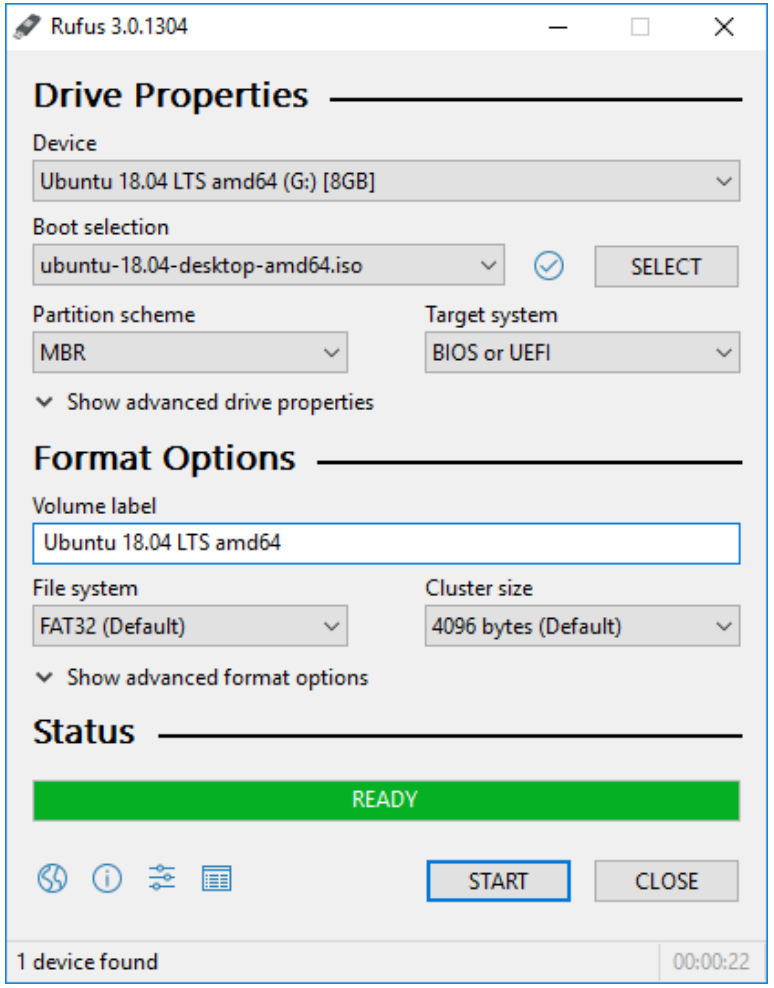
- Linux is an operating system
- GNU/Linux is a collection of programs and the Linux kernel that forms the Operating System
- Many of the components of are licensed under an open source license such as the General Public License (GPL)
- The open source nature of Linux allows you to view the source code of many of the components of the system

Yocto Project: Obtaining and Installing a Linux Distribution

- Select a distribution that others within your company are using.
- If you are completely new to Linux then pick a distribution like Debian, Fedora, or Ubuntu
- Distributions are distributed as iso files that can be written to a CD/DVD or USB flash Drive

Yocto Project: Obtaining and Installing a Linux Distribution

- Use https://rufus.ie/ to flash an iso to a USB flash drive



Yocto Project: Obtaining and Installing a Linux Distribution

- Set the computer to boot from USB and insert the flash drive into one of the USB ports
- Some distributions have LIVE preview modes that run out of RAM and allow you to try out the distribution before installing
- Follow the installation guides available on the distribution's homepage to install
- 500 GB is a good size for an install hard disk

Yocto Project: The Terminal

- The terminal will be your new home!
- Cheat sheet (http://overapi.com/linux/)

Yocto Introduction

What is the Yocto Project?

Yocto Project Development Environment

Yocto Project Components

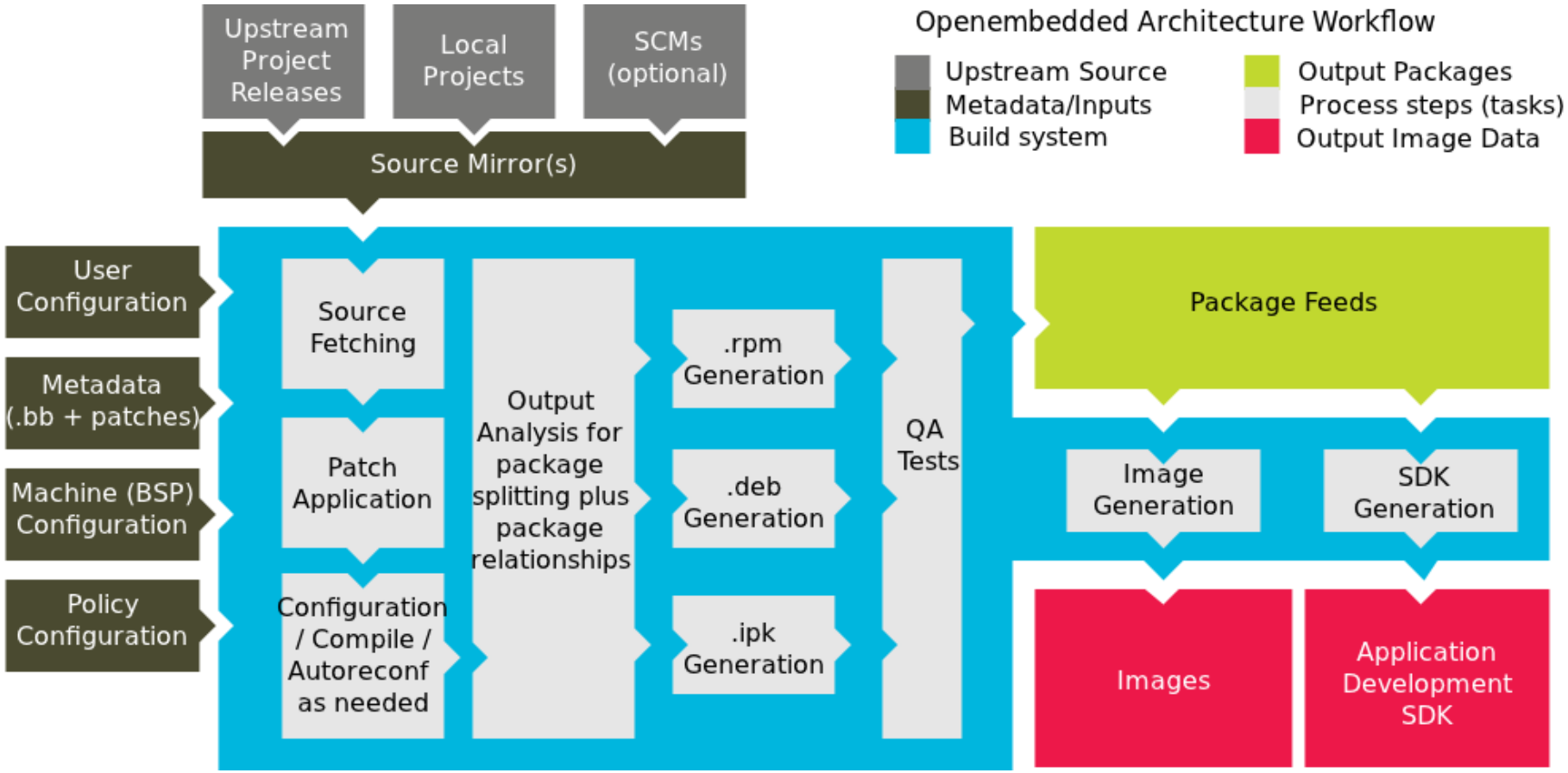
Yocto Project Documentation

Yocto Project: What is the Yocto Project?

- Open-source collaborative project focused on embedded Linux development
- Currently provides a build system that is referred to as OpenEmbedded build system in the Yocto Project Documentation
- Helps developers create custom Linux-based systems for embedded products

- Source: http://www.yoctoproject.org/docs/current/yocto-project-qs/yocto-project-qs.html

Yocto Project: Yocto Project Development Environment



Yocto Project: Yocto Project Components

Poky

- Poky is a *reference system* of the Yocto Project - a collection of Yocto Project tools and metadata that serves as a set of working examples. To use the Yocto Project tools, you can download Poky and use it to bootstrap your own distribution.
- Poky is the platform-independent, cross-compiling integration layer that utilizes OpenEmbedded Core. It provides the mechanism to build and combine thousands of distributed open source projects together to form a fully customizable, complete, coherent Linux software stack.
- Poky's objective is to provide all the features and functionalities an embedded developer needs from one solution.

Source: https://www.yoctoproject.org/tools-resources/projects/poky

Yocto Project: Yocto Project Components

- BitBake is a build engine that follows recipes in a specific format in order to perform sets of tasks. BitBake is a core component of the Yocto Project.

Yocto Project: Yocto Project Documentation

The Yocto Project documentation is available on the Yocto Project website: (https://www.yoctoproject.org/documentation)

Recommended Reading:

1. Yocto Project Quick Start (Link)
2. BitBake User Manual (Link)
3. Yocto Project Reference Manual (Link)

NXP Yocto Release

Introduction to the NXP Yocto BSP Release

Obtaining the NXP Yocto Release

Yocto Project: NXP Yocto BSP Release

The NXP Yocto BSP Release is an extension of OpenEmbedded and Poky that supports NXP reference boards

- NXP i.MX6Q SABRE Smart Device
- NXP i.MX6Q SABRE Auto
- NXP i.MX6DL SABRE Smart Device
- NXP i.MX6DL SABRE Auto
- NXP i.MX6SOLO SABRE Smart Device
- NXP i.MX6SOLO SABRE Auto
- NXP i.MX6 Solo Lite EVK
- NXP i.MX8 Family

Yocto Project: Obtaining the NXP Yocto BSP Release

The NXP Yocto BSP Release is available from our website under Software & Tools Section (Link)

- Contains pre-built images for NXP reference boards
- Contains a README file that explains the procedure to download the BSP using REPO
- Currently based on Yocto version 2.5 (Sumo)
- Next release based on Yocto version 2.6 (Thud)

i.MX Linux Roadmap

Yocto Project 2.2 – “Morty” | **Yocto Project 2.4 – “Rocko”** | **Yocto Project 2.5 – “Sumo”**

GA – Support for 1 year , Pro-Support available past one year
Alpha & Beta – Availability until next release

www.NXP.com/prosupport

L4.14.62_1.0.0, L4.14.78_1.x.0 — Beta, GA, GA

L4.9.88_2.x.x — GA, Alpha, Beta2

L4.9.123_2.x.x — GA

L4.9.51_1.x.x — GA

J	F	M	A	M	J	J	A	S	O	N	D	J	F	M	A	M	J	J	A	S	O	N	D	J	F	M
1Q			2Q			3Q			4Q			1Q			2Q			3Q			4Q			1Q		
2018												2019												2020		

Release Schedule: GA — Planning — Execution — Production — Alpha

Linux Kernel: 4.9.x, 4.14.x

Shiping (currently shipping i.MX 6 and i.MX 7 devices)

Host Machine Setup

Suitable Host Machines

Supported Linux Distributions

Required Host Packages

Repo

Yocto Project: A Suitable Host Machine

- Depending on the number of processors and cores, the amount of RAM, the speed of your Internet connection and other factors, the build process could take several hours the first time you run it. Subsequent builds run much faster since parts of the build are cached.
- Multiple build directories can consume large amounts of hard drive space
- Virtual Machines are not recommended for daily development

Yocto Project: Supported Linux Distributions

Each Yocto release adds additional supported distributions. Other distributions can also work but are not specifically tested by the Yocto Project.

These are supported by Sumo:

- Ubuntu 14.10
- Ubuntu 15.04
- Ubuntu 15.10
- Ubuntu 16.04 (LTS)
- Fedora release 22
- Fedora release 23
- CentOS release 7.x
- Debian GNU/Linux 8.x (Jessie)
- Debian GNU/Linux 9.x (Stretch)
- openSUSE 13.2
- openSUSE 42.1

Yocto Project: Required Host Packages

The list of packages you need on the host development system can be large when covering all build scenarios using the Yocto Project

Debian based distributions:

Essential packages

```
$ sudo apt-get install gawk wget git-core diffstat unzip texinfo gcc-multilib \
    build-essential chrpath socat cpio python python3 python3-pip python3-pexpect \
    xz-utils debianutils iputils-ping
```

Graphical extras (host)

```
$ sudo apt-get install libsdl1.2-dev xterm
```

Documentation packages

```
$ sudo apt-get install make xsltproc docbook-utils fop dblatex xmlto
```

OpenEmbedded Self-Test:

```
$ sudo apt-get install python-git
```

Yocto Project: Required Host Packages

Fedora based distributions:

Essential packages

```
$ sudo dnf install gawk make wget tar bzip2 gzip python3 unzip perl patch \
     diffutils diffstat git cpp gcc gcc-c++ glibc-devel texinfo chrpath \
     ccache perl-Data-Dumper perl-Text-ParseWords perl-Thread-Queue perl-bignum socat \
     python3-pexpect findutils which file cpio python python3-pip xz
```

Graphical extras (host)

```
$ sudo dnf install SDL-devel xterm
```

Documentation packages

```
$ sudo dnf install make docbook-style-dsssl docbook-style-xsl docbook-dtds docbook-utils fop
libxslt dblatex xmlto
```

OpenEmbedded Self-Test

```
$ sudo dnf install python3-GitPython
```

Yocto Project: Repo

What is Repo?

- Repo is a tool that Google built on top of Git
- Repo helps manage many Git repositories
- Repo is not meant to replace Git
- The repo command is an executable Python script that you can put anywhere in your path

- Repo repository and cheat sheet: Link

Yocto Project: Repo

Installing Repo

1. Add a bin directory to your home directory:

```
$ mkdir ~/bin
```

2. Download the script file to the bin directory:

```
$ curl https://storage.googleapis.com/git-repo-downloads/repo > ~/bin/repo
```

3. Add execute permissions:

```
$ chmod a+x ~/bin/repo
```

Source (Link)

Yocto Project: Repo

Adding ~/bin directory to your path

- Debian with BASH, other distributions will vary slightly

Temporary:

```
$ export PATH=/home/YOURUSERNAME/bin:$PATH
$ source .bashrc
```

Permanent:

```
      $ vim .profile
+     if [ -d "$HOME/bin" ] ; then
+     PATH="$HOME/bin:$PATH"
+     fi
* NOTE: most distros include ~/bin in your default path, the
directory just doesn’t exist!
```

Yocto Setup

Repo Initialization

Repo Sync

Sourcing a Build

Yocto Project: Repo Initialization

- We will use repo to pull our Yocto sources from multiple Git trees based on the manifest for the particular branch we are using. In this case we will use the current GA release based on the previous Yocto Project release
- Git.freescale.com (Link)

```
$ repo init -u https://source.codeaurora.org/external/imx/imx-manifest -b imx-linux-sumo -m
imx-4.14.98-2.0.0_ga.xml

-u is the manifest url of our new code repo at codeaurora
-m is the manifest of the yocto build that you will be using
-b is branch or revision in this case Sumo 2.5
```

- If we look in the directory we only see that a hidden directory has been created called .repo. This directory contains information that tells repo what Git repositories and commits to pull into our directory. Repo doesn't pull the source until we run the sync command.

Yocto Project: Repo Initialization

```
nxa12171@ontario:~/projects/nxp/imx/linux-bsp/imx-yocto-bsp$ repo init -u https://source.codeaurora.o
rg/external/imx/imx-manifest -b imx-linux-sumo -m imx-4.14.98-2.0.0_ga.xml
gpg: keybox '/home/nxa12171/.repoconfig/gnupg/pubring.kbx' created
gpg: /home/nxa12171/.repoconfig/gnupg/trustdb.gpg: trustdb created
gpg: key 16530D5E920F5C65: public key "Repo Maintainer <repo@android.kernel.org>" imported
gpg: key 67B7E448692B382C: public key "Conley Owens <cco3@android.com>" imported
gpg: Total number processed: 2
gpg:               imported: 2

Get https://gerrit.googlesource.com/git-repo/clone.bundle
Get https://gerrit.googlesource.com/git-repo
remote: Finding sources: 100% (8/8)
remote: Total 8 (delta 0), reused 8 (delta 0)
Unpacking objects: 100% (8/8), done.
From https://gerrit.googlesource.com/git-repo
   fb527e3..e37aa5f  master     -> origin/master
Get https://source.codeaurora.org/external/imx/imx-manifest
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   161  100   161    0     0    259      0 --:--:-- --:--:-- --:--:--   259
100  161k  100  161k    0     0   138k      0  0:00:01  0:00:01 --:--:--  138k
Receiving objects: 100% (614/614), 161.06 KiB | 17.90 MiB/s, done.
Resolving deltas: 100% (356/356), done.
From /home/nxa12171/projects/nxp/imx/linux-bsp/imx-yocto-bsp/.repo/manifests.git/clone.bundle
 * [new branch]      master                     -> origin/master
 * [new branch]      imx-linux-thud             -> origin/imx-linux-thud
 * [new branch]      imx-linux-sumo             -> origin/imx-linux-sumo
 * [new branch]      imx-linux-rocko            -> origin/imx-linux-rocko
 * [new branch]      imx-linux-morty            -> origin/imx-linux-morty
 * [new branch]      imx-android-pie            -> origin/imx-android-pie
 * [new branch]      imx-android-oreo           -> origin/imx-android-oreo
 * [new tag]         rel_imx_4.9.51_8qm_beta1 -> rel_imx_4.9.51_8qm_beta1
```

Yocto Project: Repo Initialization

```
<?xml version="1.0" encoding="UTF-8"?>
<manifest>

  <default sync-j="2"/>

  <remote fetch="git://git.yoctoproject.org" name="yocto"/>
  <remote fetch="git://github.com/Freescale" name="community"/>
  <remote fetch="git://github.com/openembedded" name="oe"/>
  <remote fetch="git://github.com/OSSystems" name="OSSystems"/>
  <remote fetch="git://github.com/meta-qt5"  name="QT5"/>
  <remote fetch="https://source.codeaurora.org/external/imx" name="CAF"/>


  <project remote="yocto" revision="c9bd4984f8f471ca2c43052714f4413ba99cf171" name="poky" path="sources/poky"/>
  <project remote="yocto" revision="27ca94f8a4336790ba117b4298566f6820e7e74c" name="meta-freescale" path="sources/meta-freescale"/>

  <project remote="oe" revision="8760facba1bceb299b3613b8955621ddaa3d4c3f" name="meta-openembedded" path="sources/meta-openembedded"/>

  <project remote="community" revision="70535e13dd2aabbad53243518f4cc5064d284592" name="fsl-community-bsp-base" path="sources/base">
    <linkfile dest="README" src="README"/>
    <linkfile dest="setup-environment" src="setup-environment"/>
  </project>

  <project remote="community" revision="82037216280a39957fb4272581637abec734ad50" name="meta-freescale-3rdparty" path="sources/meta-freescale-3rdparty"/>
  <project remote="community" revision="f7e2216e93aff14ac32728a13637a48df436b7f4" name="meta-freescale-distro" path="sources/meta-freescale-distro"/>

  <project remote="OSSystems" revision="75640e14e325479c076b6272b646be7a239c18aa" name="meta-browser" path="sources/meta-browser" />
  <project remote="QT5" revision="d4e7f73d04e8448d326b6f89908701e304e37d65" name="meta-qt5" path="sources/meta-qt5" />

  <project remote="CAF" revision="8eeb420fad668b733ab95b460895e1c337c66b25" name="meta-fsl-bsp-release" path="sources/meta-fsl-bsp-release" >
     <linkfile src="imx/tools/fsl-setup-release.sh" dest="fsl-setup-release.sh"/>
     <linkfile src="imx/README" dest="README-IMXBSP"/>
  </project>


</manifest>
~
~
~
~
~
```

Yocto Project: Repo Sync

```
 * [new branch]            master                -> yocto/master
 * [new branch]            master-next           -> yocto/master-next
 * [new branch]            master-next2          -> yocto/master-next2
 * [new branch]            morty                 -> yocto/morty
 * [new branch]            pinky                 -> yocto/pinky
 * [new branch]            purple                -> yocto/purple
 * [new branch]            pyro                  -> yocto/pyro
 * [new branch]            rocko                 -> yocto/rocko
 * [new branch]            sumo                  -> yocto/sumo
 * [new branch]            thud                  -> yocto/thud
 * [new branch]            thud-next             -> yocto/thud-next
 * [new branch]            warrior               -> yocto/warrior
 * [new branch]            warrior-next          -> yocto/warrior-next
Fetching projects: 100% (9/9)
Fetching projects: 100% (9/9), done.
Checking out project fsl-community-bsp-base
Checking out project meta-browser
Checking out project meta-freescale
Checking out project meta-freescale-3rdparty
Checking out project meta-freescale-distro
Checking out project meta-fsl-bsp-release
Checking out project meta-openembedded
Checking out project meta-qt5
Checking out project poky
Syncing work tree: 100% (9/9), done.
nxa12171@ontario:~/projects/nxp/imx/linux-bsp/imx-yocto-bsp$
```

Some output omitted for clarity

Yocto Project: Repo Sync

- After the sync command completes the source directory will contain a setup script and a sources directory.

```
├── fsl-setup-release.sh
├── README
├── README-IMXBSP
├── setup-environment
└── sources

1 directory, 4 files
```

Yocto Project: Sourcing a Build

- Now you need to pick a machine type that you want to set up to build. These machines are available in the specific <layer>/conf/machine/*.conf files
- You can find available machines with the following command in your yocto top directory

$ ls sources/meta-fsl-*/imx/meta-bsp/conf/machine/*.conf

sources/meta-fsl-arm/conf/machine/imx23evk.conf
sources/meta-fsl-arm/conf/machine/imx28evk.conf
sources/meta-fsl-arm/conf/machine/imx31pdk.conf
sources/meta-fsl-arm/conf/machine/imx35pdk.conf
sources/meta-fsl-arm/conf/machine/imx51evk.conf
sources/meta-fsl-arm/conf/machine/imx53ard.conf
sources/meta-fsl-arm/conf/machine/imx53qsb.conf
sources/meta-fsl-arm/conf/machine/imx6dlsabreauto.conf
sources/meta-fsl-arm/conf/machine/imx6dlsabresd.conf
sources/meta-fsl-arm/conf/machine/imx6qsabreauto.conf
sources/meta-fsl-arm/conf/machine/imx6qsabresd.conf
sources/meta-fsl-arm/conf/machine/imx6slevk.conf
sources/meta-fsl-arm/conf/machine/imx6solosabreauto.conf
sources/meta-fsl-arm/conf/machine/imx6solosabresd.conf
sources/meta-fsl-arm/conf/machine/twr-vf65gs10.conf

New Machines!!
imx8qxpmek
imx8qmmek

Yocto Project: Sourcing a Build

- The machine name tells setup script which board you will be using for development

```
$ DISTRO=<distro-name> MACHINE=<machine-name> source
fsl-setup-release.sh -b <build-directory>  -e
<backend fb, dfb, wayland, x11>
```

- Example using NXP i.MX8QXP with xwayland

```
$ DISTRO=fsl-imx-xwayland MACHINE=imx8qxpmek source \
fsl-setup-release.sh -b build-xwayland
```

Yocto Project: Sourcing a Build

```
nxa12171@ontario:~/projects/nxp/imx/linux-bsp/imx-yocto-bsp$ DISTRO=fsl-imx-xwayland MACHINE=imx8qxpmek source
 fsl-setup-release.sh -b build-xwayland

 Build directory is  build-xwayland

Welcome to Freescale Community BSP

The Yocto Project has extensive documentation about OE including a
reference manual which can be found at:
    http://yoctoproject.org/documentation

For more information about OpenEmbedded see their website:
    http://www.openembedded.org/

You can now run 'bitbake <target>'

Common targets are:
    core-image-minimal
    meta-toolchain
    meta-toolchain-sdk
    adt-installer
    meta-ide-support

Your configuration files at build-xwayland have not been touched.
BSPDIR=
BUILD_DIR=.
meta-freescale directory found
nxa12171@ontario:~/projects/nxp/imx/linux-bsp/imx-yocto-bsp/build-xwayland$
```

Yocto Project: Sourcing a Build

- The setup script will create the conf directory and copy some files that are needed to create the build. When the script is completed you will be placed into the build directory that you specified. We will use bitbake to spawn off the build from inside this directory

```
user@machine:~/projects/nxp/imx/linux-bsp/imx-yocto-bsp/build-xwayland$ tree
.
└── conf
    ├── bblayers.conf
    ├── bblayers.conf.org
    ├── local.conf
    ├── local.conf.org
    ├── local.conf.sample
    └── templateconf.cfg
```

Yocto Project: Optimizing Host Machine

- <build>/conf/local.conf has a few options that you can change in order to speed up building on your host machine

```
MACHINE ??= ‘imx8qxpmek'
DISTRO ?= ‘fsl-imx-xwayland'
PACKAGE_CLASSES ?= "package_rpm"
EXTRA_IMAGE_FEATURES = "debug-tweaks"
USER_CLASSES ?= "buildstats image-mklibs image-prelink"
PATCHRESOLVE = "noop"
BB_DISKMON_DIRS = "\
    STOPTASKS,${TMPDIR},1G,100K \
    STOPTASKS,${DL_DIR},1G,100K \
    STOPTASKS,${SSTATE_DIR},1G,100K \
    ABORT,${TMPDIR},100M,1K \
    ABORT,${DL_DIR},100M,1K \
    ABORT,${SSTATE_DIR},100M,1K“
PACKAGECONFIG_append_pn-qemu-native = “ sdl”
PACKAGECONFIG_append_pn-nativesdk-qemu = “ sdl”
CONF_VERSION = "1"

BB_NUMBER_THREADS = ‘8’                    <- increase to reflect the number of cores
PARALLEL_MAKE = '-j 8’   <- increase to reflect the number of cores

DL_DIR ?= "${BSPDIR}/downloads/”  <- make a common directory if you have several builds
SSTATE_DIR ?= "/share/sstate-cache“ <- make a common directory to speed builds

ACCEPT_FSL_EULA = ””
```

Running a Yocto Build

Using BitBake

First Build Time

Yocto Project: Using BitBake

- Now we use bitbake to spawn off the build

```
build-xwayland$ bitbake <image-name>
```

- <image-name> are images that are provided from the release. We can find them with find . –name fsl-image* from the yocto top directory

```
b35938@b35938-13:~/projects/fsl/yocto$ find . -name fsl-image*
./sources/meta-fsl-arm/recipes-fsl/images/fsl-image-mfgtool-initramfs.bb
./sources/meta-fsl-demos/recipes-fsl/images/fsl-image-multimedia.bb
./sources/meta-fsl-demos/recipes-fsl/images/fsl-image-machine-test.bb
./sources/meta-fsl-demos/recipes-fsl/images/fsl-image-multimedia-full.bb
./sources/meta-fsl-bsp-release/imx/meta-fsl-demos/recipes-fsl/images/fsl-image-qt5.bb
./sources/meta-fsl-bsp-release/imx/meta-fsl-demos/recipes-fsl/images/fsl-image-gui.bb
```

Yocto Project: Using BitBake

```
nxa12171@ontario:~/projects/nxp/imx/linux-bsp/imx-yocto-bsp/build-xwayland$ bitbake fsl-image-validation-imx
NOTE: Your conf/bblayers.conf has been automatically updated.
WARNING: Host distribution "ubuntu-18.04" has not been validated with this version of the build system; you may possibly
 experience unexpected failures. It is recommended that you use a tested distribution.
Parsing recipes: 100% |#################################################################################| Time: 0:00:58
Parsing of 2562 .bb files complete (0 cached, 2562 parsed). 3483 targets, 221 skipped, 8 masked, 0 errors.
NOTE: Resolving any missing task queue dependencies
NOTE: Multiple providers are available for runtime ptpd (ptpd, ptpd-qoriq)
Consider defining a PREFERRED_RPROVIDER entry to match ptpd
NOTE: Multiple providers are available for runtime ptpd-dev (ptpd, ptpd-qoriq)
Consider defining a PREFERRED_RPROVIDER entry to match ptpd-dev

Build Configuration:
BB_VERSION           = "1.38.0"
BUILD_SYS            = "x86_64-linux"
NATIVELSBSTRING      = "ubuntu-18.04"
TARGET_SYS           = "aarch64-poky-linux"
MACHINE              = "imx8qxpmek"
DISTRO               = "fsl-imx-xwayland"
DISTRO_VERSION       = "4.14-sumo"
TUNE_FEATURES        = "aarch64"
TARGET_FPU           = ""
meta
meta-poky            = "HEAD:c9bd4984f8f471ca2c43052714f4413ba99cf171"
meta-oe
meta-multimedia      = "HEAD:8760facba1bceb299b3613b8955621ddaa3d4c3f"
meta-freescale       = "HEAD:27ca94f8a4336790ba117b4298566f6820e7e74c"
meta-freescale-3rdparty = "HEAD:82037216280a39957fb4272581637abec734ad50"
meta-freescale-distro = "HEAD:f7e2216e93aff14ac32728a13637a48df436b7f4"
meta-bsp
meta-sdk             = "HEAD:8eeb420fad668b733ab95b460895e1c337c66b25"
meta-browser         = "HEAD:75640e14e325479c076b6272b646be7a239c18aa"
meta-gnome
meta-networking
meta-python
meta-filesystems     = "HEAD:8760facba1bceb299b3613b8955621ddaa3d4c3f"
meta-qt5             = "HEAD:d4e7f73d04e8448d326b6f89908701e304e37d65"

Initialising tasks: 100% |##############################################################################| Time: 0:00:05
NOTE: Executing SetScene Tasks
NOTE: Executing RunQueue Tasks
Currently  8 running tasks (105 of 7952)   1% |                                                              |
0: binutils-cross-aarch64-2.30-r0 do_fetch (pid 25611)  16% |########                                    | 27.7M/s
1: m4-native-1.4.18-r0 do_configure - 6s (pid 27129)
2: linux-libc-headers-4.14-r0 do_fetch (pid 31668)   0% |                                                    |
3: glibc-initial-2.27-r0 do_fetch (pid 31800)   0% |                                                     | 251K/s
4: glibc-2.27-r0 do_fetch - 1s (pid 360)
5: gettext-native-0.19.8.1-r0 do_fetch (pid 453)  72% |#####################################               | 19.5M/s
6: opkg-utils-0.3.6-r0 do_fetch (pid 2002)   0% |                                                           |
7: flex-2.6.0-r0 do_unpack - 0s (pid 2026)
```

Yocto Project: First Build Time

- The first build can take up to several hours depending on the speed of your machine
- Subsequent builds will take much less time because the built packages are cached to improved build times
- Only modified or added recipes will be built and deployed in the subsequent builds

Using the Results

Location of the Build Output

Flashing SD Card Image

Yocto Project: Location of the Build Output

- The build output is inside the ../tmp/deploy/images directory in the build directory you created:

```
user@machine:~/projects/nxp/imx/linux-bsp/imx-yocto-
bsp/yocto/build-xwayland/tmp/deploy/images/imx8qxpmek$ ls

*** More file are in the directory but here is a sample ***
fsl-image-gui-imx8qxpmek-20190921000528.rootfs.tar.bz2
fsl-image-gui-imx8qxpmek-20190921000528.rootfs.sdcard
uImage
u-boot.imx
fsl-image-gui-imx8qxpmek.sdcard
```

Yocto Project: Flashing SD Card Image

- Inside the images directory there is a .sdcar image file that is a complete SD card image that can be flashed and used on the reference board.
- Use dmesg or lsblk to find the name of your disk (sdX)

```
$ sudo dd if=fsl-image-imx8qxpmek.sdcard of=/dev/sdX bs=1M
516+0 records in
516+0 records out
541065216 bytes (541 MB) copied, 45.932 s, 11.8 MB/s
```

- Card can now be used to boot the reference platform

Yocto Project: Finding Help

- Community.NXP.com
- Meta-NXP mailing list
- Yocto Project Documentation

SECURE CONNECTIONS
FOR A SMARTER WORLD