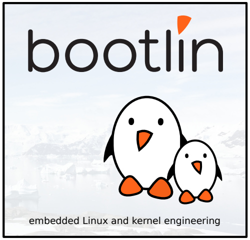
Using Yocto Project for module manufacturers

Alexandre Belloni
Bootlin
alexandre.belloni@bootlin.com

Put your business card in the box to participate in the raffle!

Alexandre Belloni

- Embedded Linux engineer at **Bootlin**
  - Embedded Linux **expertise**
  - **Development**, consulting and training
  - Strong open-source focus
- Open-source contributor
  - Contributing the **kernel support for Atmel ARM processors**
  - Contributing the **kernel support for Marvell ARM (Berlin) processors**
  - Maintainer of the Crystalfontz boards in the **meta-fsl-arm** layer



Crystalfontz

- SoM manufacturer
  - cfa-10036 is an i.MX28 based SoM
  - 128 or 256 MB RAM
  - optional OLED screen
  - SO-DIMM 200 connector
- Carrier boards
  - Prototyping: cfa-10037, screens available
  - Internet gadget: cfa-10057/cfa-10058 (commercial names: CFA920-TS and CFA921-TS)
- Linux and Barebox mainline support

`http://www.crystalfontz.com/CFA10036-Linux-SOM.php`

What is the Yocto Project ?

- Umbrella project, including:
  - pseudo
  - cross-prelink
  - matchbox
  - opkg
  - psplash
  - ...
- Provides `Poky`, a build system, based on `OpenEmbedded-Core`, uses:
  - `BitBake`, a build tool, task executor and scheduler
  - Metadata, organized in layers:

Configuration (.conf)	global definitions of variables
Classes (.bbclass)	encapsulation and inheritance of build logic, packaging, etc
Recipes (.bb)	set of instruction to build packages
Recipes extensions (.bbappend)	extension to an existing recipe

Why ?

Adding Yocto Project support allows to:

- use the initial support given by your silicon vendor
- leverage work from the community:
  - Thousands of packages available
  - Benefit from bug fixes
- generate a demo image to ship with the board
- generate an SDK so that customers are ready to start developing once they receive the board.
- easily switch between modules or evaluation boards for better benchmarking
- create a distribution (.ipk, .deb, .rpm)

Our goals

- get `core-image-*` working without any specific configuration
- inclusion in the official Freescale BSP, `meta-fsl-arm{-extra}`
    - good visibility of the platform
    - ease of use for the customers, only having to deal with one layer
    - (almost) free maintenance
- create a demo image to implement all the available features

What has been done

Upstreamed:

- machine configuration for each board
- bootloader (`Barebox`) support improvements
- new image type
- kernel recipes for the Crystalfontz boards
- formfactor configuration for the board having a touchscreen
- multiple package fixes

In `meta-crystalfontz`:

- machine configuration for all the boards
- multiple package tweaks and two demo images

First step

Working Buildroot SD card images existed, they were using:

- `imxbootlets` to boot...
- `Barebox` to select and load the correct device tree and boot...
- the Linux kernel to start...
- the Buildroot root filesystem

So the first step was to simply build `core-image-minimal` using the `imx28evk` machine configuration.

- generate an `ext3` root filesystem image and a `tar.bz2` archive along with the final SD card image
- use either of those to replace the root filesystem on an existing SD card.

This allows to quickly test the built root filesystem.

Building an image with Yocto

- Initialize the build environment

```
$ source poky/oe-init-build-env build
```

- Configure your `local.conf`

```
BB_NUMBER_THREADS = "16"
PARALLEL_MAKE = "-j 16"
MACHINE ?= "imx28evk"
```

- build the image

```
$ bitbake core-image-minimal
```

Layer creation

To make modifications, it is necessary to create a new layer:

- create a `meta-<machine>` directory
- inside that directory, create a `conf/layer.conf` file

Alternatively, you could use:

- `yocto-layer create` and select a high priority
- or `yocto-bsp create`

conf/layer.conf

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

BBFILES += "${LAYERDIR}/recipes-*/*/*.bb \
    ${LAYERDIR}/recipes-*/*/*.bbappend"

BBFILE_COLLECTIONS += "crystalfontz"
BBFILE_PATTERN_crystalfontz := "^${LAYERDIR}/"
BBFILE_PRIORITY_crystalfontz = "10"

LAYERDEPENDS_crystalfontz = "fsl-arm fsl-arm-extra"
```

conf/layer.conf

The Yocto Project documentation states:

> `At a minimum, the README file must contain a list of dependencies, such as the names of any other layers on which the BSP depends and the name of the BSP maintainer with his or her contact information.`

But it is actually quite better to also specify those dependencies in `conf/layer.conf` by using `LAYERDEPENDS`. Still, you can document how to get those dependencies in the `README`.

Adding the layer to the build

- The main drawback of having a layer separate from your silicon vendor is that your customers will have to add it to their configuration to use it.
- That configuration is done in `<builddir>/conf/bblayers.conf`. Add your layer to the `BBLAYERS` variable:

```
BBLAYERS += "${BSPDIR}/sources/meta-crystalfontz "
```

Machine configuration

Create a `<machine>.conf` file in `conf/machine/`. As we want to support multiple similar boards (all based on `cfa10036`), an include was created in `conf/machine/include/`.

conf/machine/include/cfa10036.inc

```
# Common definitions for cfa-10036 boards
include conf/machine/include/mxs-base.inc

SOC_FAMILY = "mxs:mx28:cfa10036"

PREFERRED_PROVIDER_virtual/kernel ?= "linux-cfa"
IMAGE_BOOTLOADER = "barebox"
BAREBOX_BINARY = "barebox"
IMXBOOTLETS_MACHINE = "cfa10036"
KERNEL_IMAGETYPE = "zImage"
KERNEL_DEVICETREE = "imx28-cfa10036.dtb"

# we need the kernel to be installed in the final image
IMAGE_INSTALL_append = " kernel-image kernel-devicetree"

SDCARD_ROOTFS ?= "${DEPLOY_DIR_IMAGE}/${IMAGE_NAME}.rootfs.ext3"
IMAGE_FSTYPES ?= "tar.bz2 ext3 barebox.mxsboot-sdcard sdcard"

SERIAL_CONSOLE = "115200 ttyAMA0"
MACHINE_FEATURES = "usbgadget usbhost vfat"
```

conf/machine/cfa10036.conf

The machine configuration for the module is simple:

```
#@TYPE: Machine
#@NAME: Crystalfontz CFA-10036
#@SOC: i.MX28
#@DESCRIPTION: Machine configuration for CFA-10036
#@MAINTAINER: Alexandre Belloni <alexandre.belloni@bootlin

include conf/machine/include/cfa10036.inc
```

It is always a good idea to put a contact as maintainer.

conf/machine/cfa10057.conf

For a carrier board, add the corresponding device tree and the supported features.

```
#@TYPE: Machine
#@NAME: Crystalfontz CFA-10057
#@SOC: i.MX28
#@DESCRIPTION: Machine configuration for CFA-10057, also ca
#@MAINTAINER: Alexandre Belloni <alexandre.belloni@bootlin.

include conf/machine/include/cfa10036.inc

KERNEL_DEVICETREE += "imx28-cfa10057.dtb"

MACHINE_FEATURES += "touchscreen"
```

Kernel support

For the kernel, you have multiple choices:

- patches over silicon vendor kernel tree
  - available as an include
  - using a `.bbappend`
- custom git tree
- mainline git

You also probably have to provide a configuration file.

Patches, include

The compilation logic is provided by your silicon vendor as an include file:

- create a `recipes-kernel/linux/`
- write a new recipe `linux-<vendor>_<version>.bb`
- copy your patches to `recipes-kernel/linux/linux-<vendor>-<version>`
- Example: for `linux-congatec`:

```
$ ls recipes-kernel/linux/linux-congatec*
recipes-kernel/linux/linux-congatec_3.0.35.bb

recipes-kernel/linux/linux-congatec-3.0.35:
0001-Add-linux-support-for-congatec-evaluation-board-qmx6q.patch
0001-perf-tools-Fix-getrusage-related-build-failure-on-gl.patch
0002-ARM-7668-1-fix-memset-related-crashes-caused-by-rece.patch
0003-ARM-7670-1-fix-the-memset-fix.patch
[...]
defconfig
```

Patches, include: recipe

```
SUMMARY = "Linux Kernel based on Freescale Linux kernel to add support for Cong
include recipes-kernel/linux/linux-imx.inc

SRCREV = "bdde708ebfde4a8c1d3829578d3f6481a343533a"
LOCALVERSION = "-4.1.0+yocto"
SRCBRANCH = "imx_3.0.35_4.1.0"

SRC_URI += "file://drm-vivante-Add-00-sufix-in-returned-bus-Id.patch \
            file://epdc-Rename-mxcfb_epdc_kernel.h-to-mxc_epdc.h.patch \
            file://0001-perf-tools-Fix-getrusage-related-build-failure-on-gl.pa
            file://0002-ARM-7668-1-fix-memset-related-crashes-caused-by-rece.pa
            file://0003-ARM-7670-1-fix-the-memset-fix.patch \
            file://0004-ENGR00271136-Fix-build-break-when-CONFIG_CLK_DEBUG-i.pa
            file://0005-ENGR00271359-Add-Multi-touch-support.patch \
            file://0006-Add-support-for-DVI-monitors.patch \
            file://0001-Add-linux-support-for-congatec-evaluation-board-qmx6q.p
            file://ENGR00278350-gpu-viante-4.6.9p13-kernel-part-integra.patch \
"

COMPATIBLE_MACHINE = "(cgtqmx6)"
```

Patches, include: recipe

SRCREV	The revision of the source code used to build the package.
SRCBRANCH	New in `daisy`, when using git it is required to specify in which branch the commit resides. `SRCBRANCH` is used in `SRC_URI`, in `linux-imx.inc`
SRC_URI	The list of source files. Here patches are added in the original `SRC_URI`
COMPATIBLE_MACHINE	A regular expression used to match against the `MACHINEOVERRIDES` variable which in turn includes `MACHINE`. Used to ensure the recipe won't build for other machines.

SRC_URI: file://

- When using `file://` in `SRC-URI`, OpenEmbedded will search files relative to the subdirectories listed in `FILESPATH`
- By default, this is:
  - `${BPN}`, the base recipe name
  - `${BP}`, which is `${BPN}-${PV}`, `${PV}` being the package version
  - `files`
- also looks in a subdirectory named `${MACHINE}` inside those directories
- if set, also looks for subdirectories named from `${MACHINEOVERRIDES}` and `${DISTROOVERRIDES}`
- Don't modify `FILESPATH` directly, use `FILESEXTRAPATHS`

custom git tree

When using a custom git tree, you'll have to write your own recipe. But this doesn't have to be difficult:

- inherit the kernel class, it already takes care of downloading, unpacking, configuring and compiling your kernel.
- if using device trees, include `recipes-kernel/linux/linux-dtb.inc`
- define `SRC_URI`
- define `S`
- define `COMPATIBLE_MACHINE`

recipes-kernel/linux/linux-cfa_3.10.bb

```
DESCRIPTION = "Linux kernel for Crystalfontz boards"
SECTION = "kernel"
LICENSE = "GPLv2"

LIC_FILES_CHKSUM = "file://COPYING;md5=d7810fab7487fb0aad327b76f1be7cd7"

inherit kernel
require recipes-kernel/linux/linux-dtb.inc

SRCBRANCH = "cfa-3.10.25"
SRC_URI = "git://github.com/crystalfontz/cfa_10036_kernel;branch=${SRCBRANCH} \
           file://defconfig"

SRCREV = "61dbe8ef338ce4cc1c10d5a6cdd418c047fb136d"

S = "${WORKDIR}/git"

COMPATIBLE_MACHINE = "cfa10036"
```

Bootloader support

- using `imxbootlets` to start `Barebox`
- the recipes are going in the `recipes-bsp` folder
- those recipes are extended using `.bbappend`

imx-bootlets/imx-bootlets_10.12.01.bbappend

- extends `recipes-bsp/imx-bootlets/imx-bootlets_10.12.01.bb` from `meta-fsl-arm`

```
FILESEXTRAPATHS_prepend := "${THISDIR}/${PN}:"

SRC_URI_append_cfa10036 = " file://cfa10036-support.patch"
```

- use immediate expansion `:=`
- conditionally adds the cfa10036 support patch when `MACHINEOVERRIDES` matches
- don't forget the space at the beginning of the string when using `_append`

barebox/barebox_2013.08.0.bbappend

- extends recipes-bsp/barebox/barebox_2013.08.0.bb from meta-fsl-arm

```
FILESEXTRAPATHS_prepend := "${THISDIR}/${PN}-${PV}:"

COMPATIBLE_MACHINE_cfa10036 = "cfa10036"
```

- simply adds the subdirectory, it contains the configuration

```
$ tree recipes-bsp/barebox/barebox-2013.08.0/
recipes-bsp/barebox/barebox-2013.08.0/
`-- cfa10036
    `-- defconfig
```

Image types

- an image type defines how to create the final image
- usually, it is not necessary to create a new one, simply select the one you want by using `IMAGE_FSTYPES` in your machine configuration.
- using `imxbootlets` with `barebox` was not supported so it has been added.
- adding image types is done by using classes, for boards based on Freescale SoCs, they are defined in `classes/image_types_fsl.bbclass` in `meta-fsl-arm`

Image recipes

- an image recipe is used to define the content of the final image
- it is the entry point of the build and defines all the necessary packages through dependencies
- image recipes are usually in `recipes-*/images/`

images/demo-image-cfa-mini.bb

```
DESCRIPTION = "Image for Crystalfontz boards"
LICENSE = "MIT"

IMAGE_INSTALL = "packagegroup-core-boot \
    ${ROOTFS_PKGMANAGE_BOOTSTRAP} \
    ${CORE_IMAGE_EXTRA_INSTALL}"

IMAGE_INSTALL += "init-ifupdown busybox-udhcpd iw"

IMAGE_INSTALL += "evtest tslib tslib-conf tslib-tests \
    tslib-calibrate"

IMAGE_LINGUAS = " "

inherit core-image
```

Image recipes

- use `IMAGE_INSTALL` to specify which packages you need on your target
- you can use packagegroups, they are useful when needing features with complex dependencies
- inherit the base image class `core-image`
- you can also include already existing image recipes

images/demo-image-cfa.bb (1)

```
include recipes-sato/images/core-image-sato.bb

IMAGE_FEATURES += "debug-tweaks"
WEB = "web-webkit"

IMAGE_INSTALL += " linux-firmware init-ifupdown busybox-udhcpd"

# we don't need the full tools-testapps
IMAGE_INSTALL += " evtest tslib tslib-conf tslib-tests tslib-calibrate xev"
IMAGE_INSTALL += " iw connman-client"

EXTRA_IMAGE_FEATURES += " \
    nfs-server \
    qt4-pkgs \
"

# more debugging and profiling
EXTRA_IMAGE_FEATURES += " \
    tools-debug \
    tools-profile \
"
```

images/demo-image-cfa.bb (2)

```
IMAGE_INSTALL += " \
    cpufrequtils \
    nano \
    packagegroup-qt-in-use-demos \
    qt4-demos \
    qt4-examples \
    cfa-config-extra \
    "

export IMAGE_BASENAME = "demo-image-cfa"
```

Image recipes

IMAGE_FEATURES The primary list of features to include in an image.

EXTRA_IMAGE_FEATURES List of additional features to include in an image, typically to be put in your `local.conf` file.

Available features: dbg-pkgs, dev-pkgs, doc-pkgs, nfs-server, read-only-rootfs, splash, ssh-server-dropbear, ssh-server-openssh, staticdev-pkgs, tools-debug, tools-profile, tools-sdk, tools-testapps, x11, x11-base, x11-sato

Image tweaks

There is a mechanism to describe what functionalities are available on the target, the formfactor configuration file.

- extend it with a .bbappend:
  `recipes-bsp/formfactor/formfactor_0.0.bbappend`

```
FILESEXTRAPATHS_prepend := "${THISDIR}/${PN}:"
```

- it install a file named `machconfig`

```
$ tree recipes-bsp/formfactor/
recipes-bsp/formfactor/
|-- formfactor
|   |-- cfa10057
|   |   `-- machconfig
|   `-- cfa10058
|       `-- machconfig
`-- formfactor_0.0.bbappend
```

cfa10058/machconfig

```
# Display options
HAVE_TOUCHSCREEN=1
```

Other available variables: `HAVE_KEYBOARD`, `HAVE_KEYBOARD_PORTRAIT`, `HAVE_KEYBOARD_LANDSCAPE`, `DISPLAY_CAN_ROTATE`, `DISPLAY_ORIENTATION`, `DISPLAY_WIDTH_PIXELS`, `DISPLAY_HEIGHT_PIXELS`, `DISPLAY_BPP`, `DISPLAY_WIDTH_MM`, `DISPLAY_HEIGHT_MM`, `DISPLAY_SUBPIXEL_ORDER`.

Adding extra configuration

You can create a recipe to simply install a few configuration files in your final filesystem. This is what `cfa-config-extra` is doing:
`recipes/cfa-config-extra/cfa-config-extra.bb`

```
DESCRIPTION = "Extra files for demo-image-cfa"
LICENSE = "GPLv2"
PR = "r1"
S="${WORKDIR}"
LIC_FILES_CHKSUM = "file://LICENSE;md5=c746876a5e2eaefef09efb9d7c1c463d"

SRC_URI += "file://qtbrowser.desktop \
            file://webkit.png \
            file://qtmediaplayer.desktop \
            file://qtmediaplayer.png \
            file://qtdemo.desktop \
            file://qtdemo.png \
            file://LICENSE"

inherit allarch

do_install () {
    install -d ${D}/${datadir}/pixmaps
    install -d ${D}/${datadir}/applications
    install -m 0644 ${WORKDIR}/webkit.png ${D}/${datadir}/pixmaps
    install -m 0644 ${WORKDIR}/qtbrowser.desktop ${D}/${datadir}/applications
    install -m 0644 ${WORKDIR}/qtmediaplayer.png ${D}/${datadir}/pixmaps
    install -m 0644 ${WORKDIR}/qtmediaplayer.desktop ${D}/${datadir}/applications
    install -m 0644 ${WORKDIR}/qtdemo.png ${D}/${datadir}/pixmaps
    install -m 0644 ${WORKDIR}/qtdemo.desktop ${D}/${datadir}/applications
}
```

Useful package tweaks

`recipes-connectivity/connman`, to be extended to install and `connman.defaults` file, especially to prevent `connman` from configuring some interfaces.

`recipes-connectivity/connman/connman_1.17.bbappend`

```
FILESEXTRAPATHS_prepend := "${THISDIR}/${PN}:"
SRC_URI += " file://connman.defaults"

do_install_append() {
    if ${@base_contains('DISTRO_FEATURES','sysvinit','true','false',d)}; then
        install -d ${D}${sysconfdir}/default
        install -m 0755 ${WORKDIR}/connman.defaults \
            ${D}${sysconfdir}/default/connman
    fi
}
```

`recipes-connectivity/connman/connman/connman.defaults`

```
EXCLUDED_INTERFACES="usb0"
```

Useful package tweaks

`recipes-core/busybox`, to be extended to install various configuration files for the `busybox applets` `recipes-core/busybox/busybox_1.21.1.bbappend`:

```
FILESEXTRAPATHS_prepend := "${THISDIR}/${PN}:"

SRC_URI_append_cfa10036 = " \
   file://udhcpd.conf \
"

do_install_append_cfa10036 () {
    install -m 0755 ${WORKDIR}/udhcpd.conf ${D}${sysconfdir}/
}
```

This recipe is better than the previous one as it restrict changes to a particular machine.

Useful package tweaks

recipes-core/psplash, can be extended to change the splash screen, needs more to change the color of the progress bar: recipes-core/psplash/psplash_git.bbappend:

```
FILESEXTRAPATHS_prepend := "${THISDIR}/files:"
DEPENDS += "gdk-pixbuf-native"
PRINC = "8"
SRC_URI += "file://psplash-colors.h \
         file://psplash-bar-img.png"

# NB: this is only for the main logo image; if you add multiple images here,
# poky will build multiple psplash packages with 'outsuffix' in name for
# each of these ...
SPLASH_IMAGES = "file://psplash-poky-img.png;outsuffix=default"

# The core psplash recipe is only designed to deal with modifications to the
# 'logo' image; we need to change the bar image too, since we are changing
# colors
do_configure_append () {
        cd ${S}
        cp ../psplash-colors.h ./
        # strip the -img suffix from the bar png -- we could just store the
        # file under that suffix-less name, but that would make it confusing
        # for anyone updating the assets
        cp ../psplash-bar-img.png ./psplash-bar.png
        ./make-image-header.sh ./psplash-bar.png BAR
}
```

Lessons learned

- due to the large amount of modifications in `oe-core` and `Poky`, your build will get broken!
- Don't use the following construct in `pkg_postinst`:

```
pkg_postinst_wpa-supplicant () {
        # If we're offline, we don't need to do this.
        if [ "x$D" != "x" ]; then
                exit 0
        fi

        killall -q -HUP dbus-daemon || true
}
```

It can't be extended using .bbappend

Lessons learned

- it may be better to create two layers from the start, one with what will be upstreamed, the other one with customization.
- submit early
- you won't get notified when your patches hit upstream
- you may have to fix things in `Poky`, especially regarding initialization and X11
- using `repo` to manage the layers is really helpful, example: `https://github.com/Freescale/fsl-community-bsp-platform`

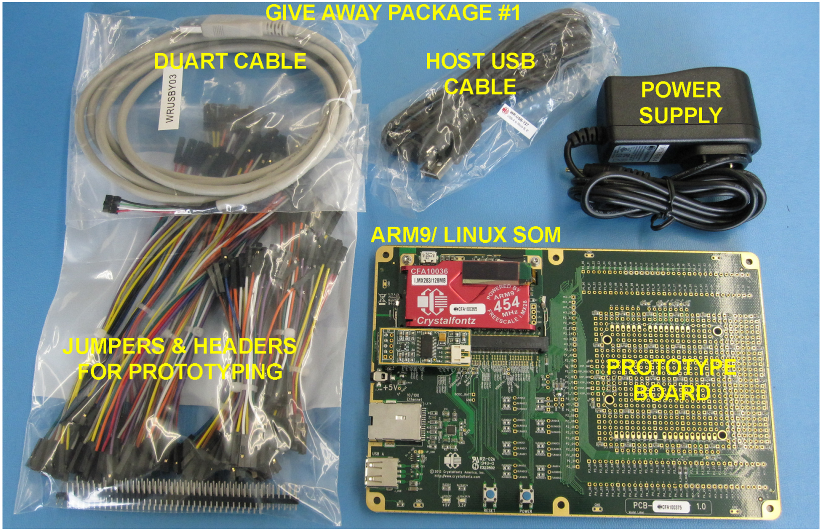
Documentation

- https://www.yoctoproject.org/documentation
- in particular the variable glossary: http://www.yoctoproject.org/docs/current/ref-manual/ref-manual.html#ref-variables-glossary
- and the BSP developer's guide: http://www.yoctoproject.org/docs/current/bsp-guide/bsp-guide.html
- Freescale BSP: http://freescale.github.io

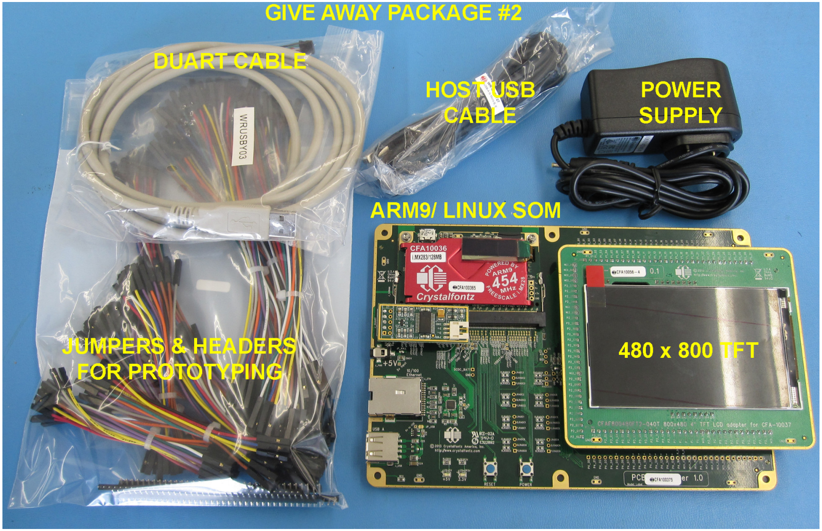
Questions?

Alexandre Belloni
alexandre.belloni@bootlin.com

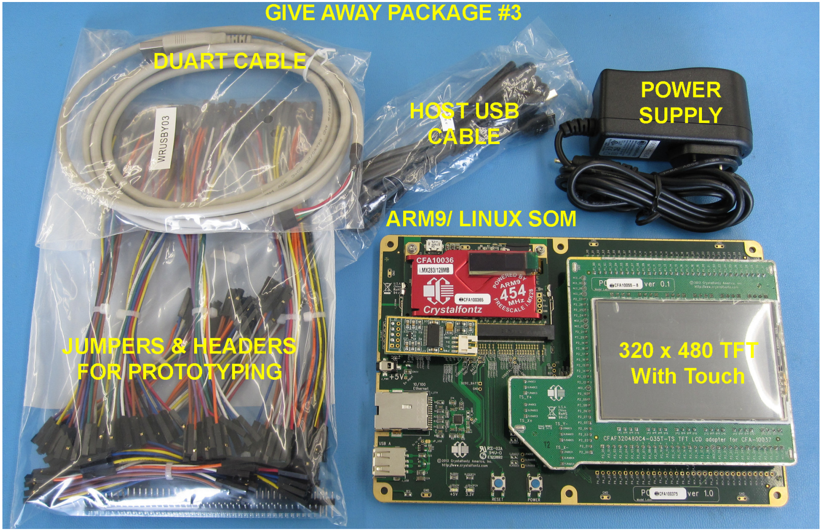
Give away


GIVE AWAY PACKAGE #1
DUART CABLE
HOST USB CABLE
POWER SUPPLY
ARM9/ LINUX SOM
JUMPERS & HEADERS FOR PROTOTYPING
PROTOTYPE BOARD
CFA10036
i.MX283/128MB
Crystalfontz
POWERED BY ARM9 454 MHz FREESCALE i.MX28
WRUSBY03

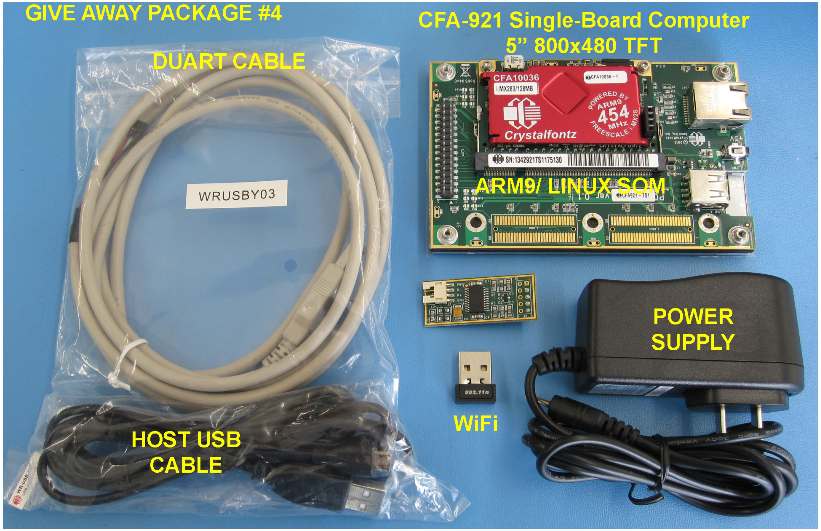
Give away


GIVE AWAY PACKAGE #2
DUART CABLE
HOST USB CABLE
POWER SUPPLY
ARM9/ LINUX SOM
JUMPERS & HEADERS FOR PROTOTYPING
480 x 800 TFT
CFA10036
Crystalfontz

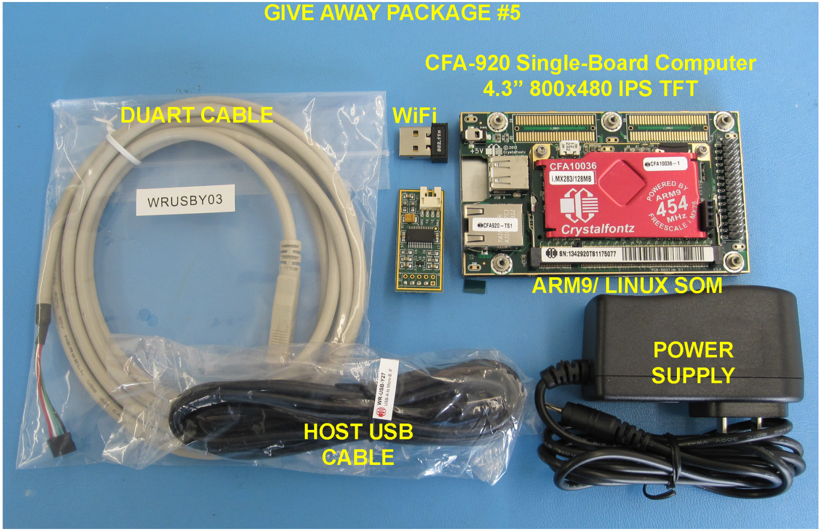
Give away


GIVE AWAY PACKAGE #3
DUART CABLE
HOST USB CABLE
POWER SUPPLY
ARM9/ LINUX SOM
JUMPERS & HEADERS FOR PROTOTYPING
320 x 480 TFT With Touch

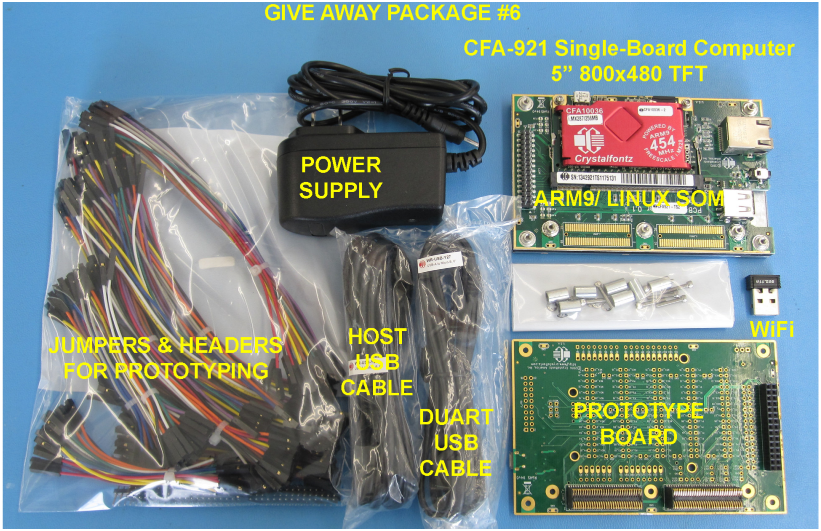
Give away


GIVE AWAY PACKAGE #4
CFA-921 Single-Board Computer
5" 800x480 TFT
DUART CABLE
WRUSBY03
ARM9/ LINUX SOM
POWER SUPPLY
WiFi
HOST USB CABLE

Give away


GIVE AWAY PACKAGE #5
CFA-920 Single-Board Computer
4.3" 800x480 IPS TFT
DUART CABLE
WiFi
WRUSBY03
ARM9/ LINUX SOM
POWER SUPPLY
HOST USB CABLE

Give away


GIVE AWAY PACKAGE #6
CFA-921 Single-Board Computer
5” 800x480 TFT
POWER SUPPLY
ARM9/ LINUX SOM
JUMPERS & HEADERS FOR PROTOTYPING
HOST USB CABLE
DUART USB CABLE
WiFi
PROTOTYPE BOARD
Crystalfontz