Wind River Systems - TEC: Linux Workshop

Wind River Systems

Tecnológico de Costa Rica - Wind River Workshop

Ing. Jairo Ortega Calderón

jairo.ortegacalderon@windriver.com

Create and add a personal layer.

Create and a layer to our Wind River Linux build environment.

Create a layer

```
bitbake-layers create-layer ../layers/meta-tec-wr
```

Add the layer to your platform project configuration.

Add layer

```
bitbake-layers add-layer ../layers/meta-tec-wr
```

Check if the layer was added.

Verify if new layer was added

```
cat conf/bblayers.conf
```

Create a kernel patch.

Before creating the patch, you will need to create the directories structure from the kernel recipe.

Create kernel's directory

```
mkdir -p recipes-kernel/linux/files
cd recipes-kernel/linux/
```

Create an append to the kernel recipe

Create the bbappend

```
vim linux-yocto_%.bbappend
```

Add the following content inside the bbappend

bbappend file

```
FILESEXTRAPATHS_prepend := "${THISDIR}/files:"
SRC_URI += "\
file://kernel.cfg \
"
```

Create a configuration file to define

A kernel parameter could be defined as:

- **CONFIG_xyz=y:** indicates that it is enabled at the build and built along with the kernel build
- **CONFIG_xyz=m:** indicates that it is enabled to be loaded during run time
- ***# CONFIG_xyz*** *is not set:* indicates that this parameter is not enabled.

Create configuration file

```
vim files/kernel.cfg
```

configuration file

```
CONFIG_LOGO=y
CONFIG_LOGO_LINUX_VGA16=y
```

Recompile kernel

```
bitbake -c clean linux-yocto
bitbake -c build linux-yocto
```

Create a personal recipe to install a file in the filesystem

First, we need to create a new directory to locate all the content for our recipe.

Create the recipe structure

```
cd ../layers/meta-tec-wr
mkdir -p recipes-apps/config-files/files
cd recipes-apps/config-files
```

Inside the files folder, we have to create a config.txt file and put the following content inside it:

File to install

```
vim files/config.txt
```

File to install

```
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaa                                                                                           aaaa
aaaaa                                      .a8H((H8a                                            aaaa
aaaaa                                     (aH8x.((88x/                                          aaaa
aaaaa                                  ./8HHxHa(.HHH((                                          aaaa
aaaaa                                  .,.,,,,,HxH88xa,                                         aaaa
aaaaa                                   a88xxxHxHH(8x8                                          aaaa
aaaaa                                    /*aaax .H(x88x                                         aaaa
aaaaa                                   a     *//(HHx88x,                                       aaaa
aaaaa                                 aa       H((((((HHx8(                                     aaaa
aaaaa                              .aa(     ,*,H(((H((((((((Hx8                                 aaaa
aaaaa                            xaaa(,HxH.((  ,(Hxx8aaaaaaaax(8a                               aaaa
aaaaa                           aaaa Hx  (x     HHx8aaaaaaaaa8aaa(                              aaaa
aaaaa                           xaa.             Hx88aaaaaaaaaaaa                               aaaa
aaaaa                             /,.             x88aaaaaaaaaaa                                aaaa
aaaaa                             **,...          .88aaaaaaaaaa8H                               aaaa
aaaaa                              /*,,,........ ..aaaaaaaaaaa88(                               aaaa
aaaaa                              ,(/***,,,,,,..,(aaaaaaaa888xx8/                              aaaa
aaaaa                                x(//*****,*,,aaaaaaaa8H(HHxxH                              aaaa
aaaaa                                  *H((///*,aaaaaaaaa8H(/(Hxxx/                             aaaa
aaaaa                                    Haaaaaaaaaaaaa8xH(///(Hxx8                             aaaa
aaaaa                                     aaaa8888aaaa8H(((///(Hxxxx                            aaaa
aaaaa                                      //(   .xaa8(((((///(Hxxxx                            aaaa
aaaaa                                      ./,     ,//            .(x                           aaaa
aaaaa                                       **      /.                                          aaaa
aaaaa                           *,,,****,,,,,,,,*.*/,/*,                                        aaaa
aaaaa                              */*******, .*****,*,*,*                                      aaaa
aaaaa                                  ..  ,,,,,,,,,,,,,,*,**,                                  aaaa
aaaaa                                                                                           aaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
```

Then, create a recipe config-files_1.0.bb and put the following content inside it:

Recipe to install file

```
DESCRIPTION = "This package contains a sample to add a file."
LICENSE = "MIT"
LIC_FILES_CHKSUM = "file://${COMMON_LICENSE_DIR}/MIT;md5=0835ade698e0bcf8506ecda2f7b4f302"

SECTION = "sample"

SRC_URI = "file://config.txt"

S = "${WORKDIR}"


do_install() {
install -d ${D}${base_bindir}/
install -m 0755 ${S}/config.txt ${D}${base_bindir}/
}
```

After that, we need to secure the integrity of the recipe with the checksum of the license:

Verify the file's checksum

```
md5sum ../../../oe-core/meta/files/common-licenses/MIT
```

Review is the same value as the following line in the recipe:

Checksum validation in the recipe

```
LIC_FILES_CHKSUM = "file://${COMMON_LICENSE_DIR}/MIT;md5=0835ade698e0bcf8506ecda2f7b4f302"
```

Finally, to add the recipe into the image, we need to:

1. Since this new recipe isn't a supported one from Wind River, we have to "authorize" it.
2. Install the recipe

Open local.conf

```
cd {projectDir}/build
vim conf/local.conf
```

Add the following lines at the end of the file:

Add the personal recipe

```
#Whitelist personal recipe, for this usecase is'config-files'
PNWHITELIST_meta-tec-wr += 'config-files'
#Install the recipe name, for this usecase is'config-files'
IMAGE_INSTALL_append = " config-files"
```

Install supported recipes

For this workshop we need to install the git and gcc packages.

Add supported recipes

```
#Install supported libraries
IMAGE_INSTALL_append = " git gcc"
```

Recompile the image and SDK

Building the image and the SDK

```
bitbake wrlinux-image-std && bitbake wrlinux-image-std -c populate_sdk
```

SDK

First, create a C++ code with the following content. The idea is to set as input of the program the path of the file previously installed with the recipe.

C++ code

```
#include <iostream>
#include <fstream>
#include <string>

int main(int argc, char *argv[]){
    std::ifstream file_image;
    std::string linesFile;

    file_image.open(argv[1]);
    if ( file_image.is_open() ) {
        while (file_image) {
            std::getline (file_image, linesFile);
            std::cout << linesFile << '\n';
        }
                file_image.close();
    }
    else {
        std::cout << "Couldn't open the file. Make sure you specify the path well.\n";
    }
    return 0;
}
```

Install the SDK

The SDK must be installed with the sh script to use it:

SDK installation script

```
./wrlinux-10.19.45.19-glibc-x86_64-qemuarm64-wrlinux-image-std-sdk.sh
```

SDK install process

```
Wind River Linux LTS SDK installer version 19.45

================================================

Enter target directory for SDK (default: /opt/windriver/wrlinux/19.45): <path>

You are about to install the SDK to "/home/jortegac/TEC-workshop/sdk/<path>". Proceed [Y/n]? Y
```

Source the SDK environment

This step sets the required environment variables for the target architecture to allow you to immediately begin application development and cross-compiling. Specifically, the environment variables **CC**, **CXX**, and **CFLAGS** are set to cross-compile C and C++ programs using the corresponding cross-compilers.

Source SDK environment

```
. ./environment-setup-arch-wrs-linux
```

Compile the application

Compile the application

```
$CXX reader.cpp -o reader_<image> -O
```

Install the image

Once you have the image file (.wic), insert the SD card, identify the path of it, and execute this command.

Flash SD

```
$ bmaptool copy --nobmap <wrlinux-image.wic> /dev/<sd path>
```

For example:

Flash SD

```
$ bmaptool copy --nobmap wrlinux-image-std-bcm-2xxx-rpi4.wic /dev/mmcblk0
```

Update the default output values

By default, the boot partition of the wic image, the cmdline.txt file, contains "ip=dhcp" parameter. This causes a slow boot when the Raspberry Pi is not connected to a network. Also, the kernel messages are only printed to the serial console (console=serial0,115200). So, let's update this parameters.

Update default parameters

```
#Mount the boot partition
vim cmdline.txt
dwc_otg.lpm_enable=0 console=serial0,115200 root=/dev/mmcblk0p2 rootfstype=ext4 rootwait ip=none console=tty0
```

Run the qemu image

In case you are not able to use the Raspberry Pi 4, and you are following the workshop with a qemu configuration project, run the following command to test the image.

Run qemu

```
$ runqemu <qemu_arch> nographic slirp
```

For example:

Run qemu

```
$ runqemu qemuarm64 nographic slirp
```

Application test in the target

Insert the SD card in the target and power on it. In case you don't have the Ethernet cable connected it will take more time to boot up. Once you login, clone the application from the repository and run it.

Run the application

```
git clone https://github.com/jortegac-wr/wrlx-workshop.git
cd wrlx-workshop/
#<reader_image> "path"
./reader_<image> "/bin/config.txt"
```