Guide to BOOT linux(OS) on UETRV_PCORE:

Setting up the Buildroot, RISC-V GNU Toolchain, Linux Kernel, and and OpenSBI development environment

Downloading Buildroot:

- Visit the Buildroot download page at https://buildroot.org/downloads/.
- Download the Buildroot version named buildroot-2021.05.tar.gz with a file size of approximately 6.7MB.

Creating a Directory:

- Open your terminal.
- Create a directory named linux_boot using the mkdir command.
- Use the cd command to navigate to the linux_boot directory where you want to store the Buildroot tarball.
- Copy the downloaded buildroot-2021.05.tar.gz file into this directory.

Extracting Buildroot:

- In the terminal, execute the following command to extract the contents of the tarball:
  `tar -xvzf buildroot-2021.05.tar.gz`

Configuration

- To configure Buildroot interactively, use the following command inside the directory buildroot-2021.05 :
  `make ARCH=riscv CROSS_COMPILE=riscv32-unknown-linux-gnu- menuconfig menuconfig`
  The above command is used to make a default .config file for buildroot
- Copy the .config file provided as a golden reference and paste it (replace the default one) the buildroot-2021.05 folder.
- Go to the directory ~/linux_boot/buildroot-2021.05/output/target/etc/init.d and edit the rcS as;

```
#!/bin/sh echo
"*********************************************************************
************"
echo "UETRV_PCORE Rootfs"

echo
"*********************************************************************
************"
exec /bin/sh

# Start all init scripts in /etc/init.d
# executing them in numerical order.
#
for i in /etc/init.d/S??* ;do

     # Ignore dangling symlinks (if any).
     [ ! -f "$i" ] && continue

     case "$i" in
    *.sh)
```

```
        # Source shell script for speed.
        (
        trap - INT QUIT TSTP
        set start
        . $i
        )
        ;;
    *)

        # No sh extension, so fork subprocess.
        $i start
        ;;
    esac
done

exec /bin/sh
```

Configuring Busybox:

- Configure Busybox by running the following command:

  ```
  make ARCH=riscv CROSS_COMPILE=riscv32-unknown-linux-gnu- busybox-menuconfig
  ```
  The above command is used to make a default .config file for busybox
- Copy the .config file provided as a golden reference and paste it (replace the default one) ~/linux_boot/buildroot-2021.05/package/busybox.

- Execute the **make** command to initiate the build process for Buildroot and Busybox.

Installing the RISC-V GNU Toolchain:

Pre-Requisites

- Install the necessary pre-requisite packages by running:

  ```
  sudo apt-get install autoconf automake autotools-dev curl python3 python3-pip libmpc-dev libmpfr-dev libgmp-dev gawk build-essential bison flex texinfo gperf libtool patchutils bc zlib1g-dev libexpat-libexpat-dev ninja-build git cmake libglib2.0-dev
  ```

Cloning the Repository:

- Clone the RISC-V GNU Toolchain repository from GitHub using the following command

  ```
  git clone https://github.com/riscv/riscv-gnu-toolchain.git
  ```

Navigating to the Directory:

- Change your working directory to the riscv-gnu-toolchain directory:

  ```
  cd riscv-gnu-toolchain
  ```

Configuring the Toolchain:

- Configure the toolchain with the desired installation path (in this example, we use /opt/riscv) using the following command:

  ```
  ./configure --prefix=/opt/riscv --with-arch=rv32ima
  ```

Building the Toolchain:

- Build and install the RISC-V GNU Toolchain for Linux with administrative privileges using using sudo:

  ```
  sudo make linux
  ```

OR :

```
sudo make -j<n> linux
```

Add the following line to ~/.bashrc to add the toolchain to your PATH:

```
export PATH="/opt/riscv/bin:$PATH"
```

Downloading Kernel:

- Visit https://www.kernel.org/ to download the Linux kernel. Ensure you download version linux-6.1.tar.gz with a size of approximately 206MB.

Extracting Kernel:

- In the terminal, navigate to the linux_boot directory where you want to store the Linux kernel.
- Execute the following command to extract the contents of the linux-6.1.tar.gz file:

```
tar -xvzf linux-6.1.tar.gz
```

Configuring the Linux Kernel:

- Navigate to the directory where you have extracted the Linux kernel, in this case, it's assumed to be linux-6.1:

```
cd ~/linux_boot/linux-6.1
```

Running menuconfig:

- Use the following command to configure the Linux kernel interactively for the RISC-V architecture:

```
make ARCH=riscv CROSS_COMPILE=riscv32-unknown-linux-gnu- menuconfig menuconfig
```

- Copy the files named .config and run.sh, and paste them in the linux-6.1 folder.
- Copy uetrv_pcore_defconfig and paste it in ~/linux_boot/linux-6.1/arch/riscv/configs/.
- Copy the file rootfs.cpio from ~/linux_boot/bootroot-2021.05/output/images and paste it into the linux-6.1 folder.
- Navigate to the directory path ~/linux_boot/linux-6.1/drivers/tty/serial and proceed to make modifications to the file named sifive.c located within that directory. Specifically, comment out lines 956 and 957, which are responsible for displaying the error message "missing aliases entry" and returning the 'id' value. Following the commented lines, insert the code 'id = 0;' .

Building:

- Build the Linux kernel using the following command:
  ```
  make V=1 ARCH=riscv CROSS_COMPILE=riscv32-unknown-linux-gnu- vmlinux vmlinux
  ```

OpenSBI:

- Visit the OpenSBI GitHub repository using the following link: https://github.com/riscv-software-src/opensbi.git.
- Go to version 0.9 from the branch/tags tab and download the code zip.
- Go to the linux_boot directory and extract the files there.
  ```
  unzip opensbi-0.9.zip
  ```
- Copy and Paste vmlinux from ~/linux_boot/linux-6.1 folder to linux_boot.
- Copy and Paste run.sh and the util folder.

Remember not to paste vmlinux, run.sh and util inside linux-6.1 instead paste them in linux_boot directory

- Run **./run.sh** command on the terminal inside directory linux_boot.

Credits: This guide has been written by Masooma Zia, Abdullah Azhar, Wajid Ali, and Uneeb Kamal.