FREE eBook

LEARNING embedded-linux

Free unaffiliated eBook created from
Stack Overflow contributors.

#embedded-linux

Table of Contents

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: embedded-linux

It is an unofficial and free embedded-linux ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official embedded-linux.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with embedded-linux

Remarks

This section provides an overview of what embedded-linux is, and why a developer might want to use it.

It should also mention any large subjects within embedded-linux, and link out to the related topics. Since the Documentation for embedded-linux is new, you may need to create initial versions of those related topics.

Examples

Installation or Setup

Detailed instructions on getting embedded-linux set up or installed.

ARM Versatile Express Emulation On Qemu

Environment Introduction:

Host ubuntu :- 12.04

Linux kernel version: linux-4.4

busybox version: 1.24.0

Cross compiler tool chain: arm-2014.05-29-arm-none-linux-gnueabi-i686-pc-linux gnu.tar.bz2

qemu version: qemu-2.5

Download and Installation of QEMU:

```
$ mkdir Source_Code
$ cd Source_Code
$ git clone git://git.qemu-project.org/qemu.git
$ cd qemu
$ git checkout remotes/origin/stable-2.5 -b stable-2.5
$ cd ../../
$ mkdir -p Binary_images/Qemu_Bin
$ cd Qemu_src/qemu
$ cd Source_Code/qemu
$ ./configure --target-list=arm-softmmu --prefix=/Path/to/your/Binary_images/Qemu_Bin
$ make
$ make install
```

ARM Cross_Compiler Tool chain installation:

Download Source code: http://sourcery.mentor.com/public/gnu_toolchain/arm-none-linux-gnueabi/

Download –> arm-2014.05-29-arm-none-linux-gnueabi-i686-pc-linux-gnu.tar.bz2

```
$ mkdir -p Binary_images/ARM_Cross_Tools
$ cd Binary_images/ARM_Cross_Tools
$ tar xvf arm-2014.05-29-arm-none-linux-gnueabi-i686-pc-linux-gnu.tar.bz2
```

Download Linux kernel source code :

```
$ cd Source_Code
$ git clone https://github.com/torvalds/linux
$ cd linux
# Switch to version v4.4
$ git checkout v4.4
```

Prepare for compilation: Load default config for target board i.e vexpress_defconfig.

```
$ make ARCH=arm CROSS_COMPILE=/path/to/your/Binary_images/ARM_Cross_Tools/arm-2014.05/bin/arm-none-linux-gnueabi- vexpress_defconfig
```

Adjust or enable some settings as of now not required in future it might required.

```
$ make ARCH=arm CROSS_COMPILE=path to your/Binary_images/ARM_Cross_Tools/arm-2014.05/bin/arm-none-linux-gnueabi- menuconfig
```

Compile the kernel

```
$ make ARCH=arm CROSS_COMPILE=/path/to/your/Binary_images/ARM_Cross_Tools/arm-2014.05/bin/arm-none-linux-gnueabi- all
```

Verify qemu and kernel can run successfully:

```
~/Binary_images/Qemu_Bin/qemu-system-arm -M vexpress-a9 -m 512M -dtb
./arch/arm/boot/dts/vexpress-v2p-ca9.dtb -kernel ./arch/arm/boot/zImage -append
"console=ttyAMA0" -serial stdio
```

Compiling Busybox for ARM on QEMU :

Download Busybox from https://busybox.net/downloads/

```
$ cd busybox
$ make ARCH=arm CROSS_COMPILE=/path/to/your/Binary_images/ARM_Cross_Tools/arm-2014.05/bin/arm-none-linux-gnueabi- defconfig
```

Enable or disable some settings as mentioned below

```
$make ARCH=arm CROSS_COMPILE=/path/to/your/Binary_images/ARM_Cross_Tools/arm-2014.05/bin/arm-none-linux-gnueabi- menuconfig
```

Busybox Settings —> Build Options —>

[*] Build BusyBox as a static binary (no shared libs)

```
$ make ARCH=arm CROSS_COMPILE=/path/to/your/Binary_images/ARM_Cross_Tools/arm-2014.05/bin/arm-none-linux-gnueabi- install
```

The above command builds Busybox and creates a directory called _install containing the root filesystem tree. Next, you need to create folder for mounting Virtual file systems like proc, sys and init scripts.

```
$ mkdir -p _install/proc/
$ mkdir -p _install/sys/
$ mkdir -p _install/tmp/
$ mkdir -p _install/root/
$ mkdir -p _install/var/
$ mkdir -p _install/mnt/
$ mkdir -p _install/etc/init.d/
```

Create a file name rcS inside folder _install/etc/init.d/ and edit rcS file with below content

```
#!/bin/sh
PATH=/sbin:/bin:/usr/sbin:/usr/bin
runlevel=S
prevlevel=N
umask 022
export PATH runlevel prevlevel
mount -a
echo /sbin/mdev /proc/sys/kernel/hotplug
mdev -s
```

Create a file name inittab inside _install/etc/ and edit it with below content.

```
# /etc/inittab
::sysinit:/etc/init.d/rcS
console::askfirst:-/bin/sh
::ctrlaltdel:/sbin/reboot
::shutdown:/bin/umount -a -r
::restart:/sbin/init
```

Download and copy fstab file to /etc/. folder in rootfs

```
$ wget clone
https://github.com/mahadevvinay/Embedded_Stuff/tree/master/Embedded_Linux_Virtual_Setup/fstab
```

Create ext3 image file and Copy all the files in our _install folder to image:

```
$ dd if=/dev/zero of=RootFS.ext3 bs=1M count=$((32))
$ sudo mkfs.ext3 RootFS.ext3
$ mkdir tmpfs
$ sudo mount -t ext3 RootFS.ext3 tmpfs/ -o loop
$ sudo cp -r _install/* tmpfs/.
$ sudo umount tmpfs
```

The complete command is to emulate:

```
~/Qemu_/Binary_images/Qemu_Bin/bin/qemu-system-arm -M vexpress-a9 -dtb path to your linux
folder/arch/arm/boot/dts/vexpress-v2p-ca9.dtb -kernel path to your linux
folder/arch/arm/boot/zImage -append root=/dev/mmcblk0 console=ttyAMA0 -sd path to your
busybox-1.24.0/RootFS.ext3 -serial stdio
```

The above setup is for Qemu, same procedure can be used to setup any Embedded Target.

Read Getting started with embedded-linux online: https://riptutorial.com/embedded-linux/topic/5479/getting-started-with-embedded-linux

Credits

S. No	Chapters	Contributors
1	Getting started with embedded-linux	Community, Thiru Shetty, vinay hunachyal