Manual para la instalación de herramientas

Martinez Hernandez. David Ricardo[1], Tobasura Madero. David Leonardo[2], and Urbano Vallejo. Oscar Andres[3]

[1]drmartinezhe@unal.edu.co
[2]dltobasuram@unal.edu.co
[3]oaurbanov@unal.edu.co

1. Comunicación con el exterior

La tarjeta posee múltiples pines de comunicación. Los que se han de usar para la programación y la interfaz con el usuario son:

- USB DP $(+)$ y DM $(-)$: para realizar carga de programas pequeños a la memoria interna del procesador y probar el funcionamiento de la tarjeta.
- Serial Port R_x y T_x: Para la interacción entre el usuario y la tarjeta usando Minicom.

Para la comunicación usando los terminales USB, se usa un cable USB a miniUSB, al cual se le recorta el extremo del terminal miniUSB; a los 4 cables ($+5\ V$, GND, DP, DM), se les adapta una regleta hembra tal que se puedan conectar de a pares a la regleta macho de la tarjeta ($+5V$ y GND juntos, y DP y DM juntos). Los terminales $+5\ V$ y GND energizan toda la tarjeta.
Para la comunicación serial se requiere un IC que sirva de interfaz entre el protocolo USB y serial, pero usando niveles lógicos de $0\ V$ y $3{,}3\ V$ que son los que usa el procesador. Esta característica en el mercado se conoce como ÜSB to serial - TTL compatible". En el mercado se consigue un pequeño PCB que tiene montado el conector USB, el $IC\ PL2303hx$, y 5 pines de salida, que son R_x, T_x, GND, $+5\ V$, $+3{,}3\ V$. Para comunicar la tarjeta con esta interfaz se usa un cable ribbon de 5 hilos, y en ambos extremos se le adapta regletas hembra para poderse conectar a los pines T_x R_x del $PL2303hx$, y a los pines R_x y T_x de la tarjeta.

2. Instalación Herramientas

2.1. Instalación de algunas librerías

Ejecutar en la consola lo siguiente
sudo apt-get install libftdipp1 libftdi1 urjtag sed wget cvs subversion git-core coreutils unzip te-

xi2html texinfo libsdl1.2-dev docbook-utils gawk python-pysqlite2 diffstat help2man make gcc build-essential g++ desktop-file-utils chrpath flex libncurses5 libncurses5-dev libxml-simple-perl zlib1g-dev pkg-config gettext libxml-simple-perl guile-1.8 cmake gmpc-dev build-essential libmpc-dev zlib1g-dev libncurses5-dev m4 uboot-envtools uboot-mkimage tree minicom gedit-dev gedit-common gedit-plugins gedit-latex-plugin gedit unrar libftdipp1 libftdi1 qucs kicad textlive-full kile okular

2.2. Instalación de compiladores

Para poder instalar las herramientas del sistema embebido es necesario descomprimir el archivo *buildroot.tar.bz2* el cual contiene las herramientas necesarias para desarrollar adecuadamente el proyecto. Para descomprimir dicho archivo se debe ejecutar lo siguiente en la terminal:

```
tar -xvjf buildroot.tar.bz2
```

Se accede a dicho directorio con el comando `cd buildroot` Para instalar las herramientas que se encuentran en ese archivo se utiliza el siguiente comando:

`make` o `make -j6`

El comando -jn es para crear **n** flujos de compilación dependiendo de la cantidad de procesadores que contenga su computador.
Este comando permite descargar algunas herramientas del toolchain necesarias para la compilación y correcta ejecución de el sistema embebido

2.2.1. Compilando la imagen del Kernel de Linux

- linux-2.6.35.3.tar.bz2: Contiene todo el código fuente del núcleo Linux. Además, también contiene un programa con una interfaz gráfica basada en la librería curses (la misma usada para el programa buildroot) donde se encuentran múltiples opciones de configuración del kernel.
- Imx-bootlets-src-10.05.02.tar.bz2: Son herrmientas para copiar la imagen de linux (creada en la carpeta anterior) a la SD, junto con un conjunto de configuraciones de registros internos del procesador i.MX, necesarios para que este pueda inicializar reguladores de tensión, memorias externas y buses de comunicación.

Estos archivos se deben descomprimir en el mismo directorio.

2.2.2. Modificación de la variable PATH

Un primer paso en modificar la variable de entorno PATH, para agregarle la ruta donde se encuentra el Cross_Compilador que fue creado usando buildroot. Esto se realiza modificando el archivo .bashrc, se debe ejecutar lo siguiente en consola:

```
cd
gedit .bashrc
```

Al final del archivo se añade la linea de comando

```
export PATH=$PATH:/home/usuario/directorio/en/donde/se encuentra/buildroot/output/host/usr/bin/
```

Esto modifica la variable PATH adicionando una nueva ruta de búsqueda de ejecutables. La ruta especificada es donde se encuentra el Cross-Compilador.

2.3. Prueba de funcionamiento

Es necesario descargar algunos paquetes adicionales los cuales se encuentran en bitbucket.org/cicamargoba/mini-open/downloads, en este enlace se encuentran los siguientes archivos

Cuadro 1: .

chibios_mini.tar.bz2	Contiene las librerías necesarias para la compilación del procesador en tiempo real
ejemplo_plataforma.mp4	ejemplo de la plataforma en formato MP4
linux-3.7.1.tar.bz2	kernel de linux versión 3.7.1
toolchain_sam3.tgz	Toolchain para el procesador SAM3
buildroot.tar.bz2	Toolchain general para la compilación
Imx-bootlets-src-10.05.02.tar.bz2	Genera el ejecutable que va en la SD
linux-2.6.35.3.tar.bz2	kernel de linux versión 3.6.35.3
material_curso.tgz	Material del curso que se puede utilizar

2.3.1. Descarga de Lua

En una terminal ejecute

```
wget http://www.lua.org/ftp/lua-5.2.2.tar.gz
tar zxvf lua-5.2.2.tar.gz
cd lua-5.2.2
make linux
```

Si genera algún problema por una libreria es necesario instalarla, de la siguiente forma

```
apt-cache search readline | grep readline
sudo apt-get install libreadline-dev
```

luego se vuelve a ejecutar el comando

```
make linux
sudo make install
sudo cp /usr/local/lib/liblua.a /usr/lib/
```

Lua es un lenguaje compacto, interpretado, ligero y sencillo, y es uno de los que soporta el i.MX.

2.4. Crear las particiones necesarias en la SD, formatear la memoria SD y grabar el bootloader y el sistema de archivos

Se necesitan dos particiones en la SD, la primera almacenará el boot-loader y la imágen del kernel; la segunda almacenará el sistema de archivos. Para los siguientes comandos se asume que la tarjeta es detectada como /dev/sdb (para conocer el dispositivo asignado a la tarjeta por Linux se debe insertar la tarjeta y ejecutar el comando `dmesg`)

2.4.1. Desmontar la partición y ejecutar `fdisk`

Ejecutar en la consola lo siguiente

```
sudo umount /dev/sdb*
sudo fdisk /dev/sdb
```

Borrar las particiones existentes: Dentro de fdisk se debe escribir “d” (delete a partition)

```
Command (m for help): d
Partition number (1-4):
```

En el campo “Partition number” se debe escribir el número de la partición a ser eliminada, el proceso se debe repetir hasta borrar todas las particiones existentes.

2.4.2. Crear una partición de 200MB

Dentro de fdisk se debe escribir “n” (add a new partition) . Seleccionar partición primaria: Escribir “p”

```
Command (m for help): n
Partition type:
p primary (0 primary, 0 extended, 4 free)
e extended
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-15122431, default 2048):
Last sector, +sectors or +sizeK,M,G (2048-15122431, default 15122431): +200M
```

2.4.3. Crear la segunda partición en el espacio sobrante en la SD

Dentro de `fdisk` se escribe

```
Command (m for help): n
Partition type:
p primary (0 primary, 0 extended, 4 free)
e extended
Select (default p): p
Partition number (1-4, default 1): 2
First sector (411648-15122431, default 15122431):
Last sector, +sectors or +sizeK,M,G (411648-7626751, default 15122431):
```

2.4.4. Cambiar el tipo de la partición 1 a OnTrack DM6 (53)

Dentro de `fdisk` se escribe

```
Command (m for help): t
Partition number (1-4): 1
Hex code (type L to list codes): 53
Changed system type of partition 1 to 53 (OnTrack DM6 Aux3)
```

"53" es el formato de partición que Freescale diseñó para el i.MX.

2.4.5. Cambiar el tipo de la partición 2 a Linux (83)

Dentro de `fdisk` se escribe

```
Command (m for help): t
Partition number (1-4): 2
Hex code (type L to list codes): 83
```

Al finalizar se debe tener la siguiente tabla de particiones, la cual se obtiene al ejecutar el comando "p" (print the partition table):

```
Device     Boot    Start      End      Blocks   Id  System
/dev/sdb1  2048    411647     204800   53       OnTrack DM6 Aux3
/dev/sdb2  411648  15122431   7355392  83       Linux
```

Una vez finalizada la creación de las particiones se deben guardar los cambios en la tabla de particiones de la SD, para esto debe ejecutar el comando "w"

```
Command (m for help): w
The partition table has been altered!
Calling ioctl() to re-read partition table.
Syncing disks.
```

2.4.6. Formatear la segunda partición como ext3

Dentro de la consola se escribe lo siguiente.

```
sudo mkfs.ext3 /dev/sdb2
```

2.4.7. Almacenar el boot-loader y la imágen del kernel a la SD

El bootloader y la imágen se encuentran en la carpeta Factory_Test/basic/files.

```
sudo dd if=sd_mmc_bootstream.raw of=/dev/sdb1
sudo umount /dev/sdb*
```

2.4.8. Grabar el sistema de archivos a la segunda partición

El .tar.bz2 se encuentra en la carpeta files.

```
sudo tar -jxvf buildroot_fs.tar.bz2 -C /media/disk/
```

3. Creación de la imagen de Linux

Dentro de la carpeta linux-2.6.35.3 hay un Makefile que será usado para compilar todo el núcleo Linux, usando el cross-compilador generado en la carpeta buildroot.
Para editar las opciones de configuración del kernel, se abre la interfaz gráfica parecida a la existente en buildroot, la cual permite listar y seleccionar las múltiples configuraciones. Para esto se ejecuta en el directorio raíz de linux-2.6.35.3:

```
make ARCH=arm CROSS_COMPILE=arm-linux- menuconfig
```

donde `ARCH=arm` especifica que se va a compilar para la arquitectura ARM, y `CROSS_COMPILE = arm-linux-` especifica el cross-compilador que se va a usar. Finalmente, se especifica la directriz a ejecutar:

```
menuconfig
```

Al ejecutar este comando, se presenta un error de un enlace roto a un archivo. En la carpeta buildroot/output/host/usr/bin/ existen múltiples archivos que son enlaces a otros archivos ubicados en esa misma carpeta. El enlace roto se presenta en este archivo: `arm-linux-ld.real`, que apunta a `arm-buildroot-linux-uclibcgnueabi-ld.real`. Este problema se soluciona creando manualmente el enlace, así:

```
ln -s arm-buildroot-linux-uclibcgnueabi-ld.real arm-linux-ld.real
```

donde la opción "-s" crea un enlace simbólico entre ambos archivos.
Al volver a ejecutar el comando `make ARCH=arm CROSS_COMPILE=arm-linux- menuconfig` se abre la ventana de la interfaz gráfica que muestra las opciones de configuración del kernel.
Al ejecutar el siguiente comando en consola, se genera la imagen del kernel de linux

```
make ARCH=arm CROSS_COMPILE=arm-linux-
```

La imagen final del Kernel Linux se crea en la carpeta `linux-2.6.35.3/arch/arm/boot`, y tiene como nombre `zImage`. Éste es el Sistema Operativo propiamente dicho, el cual se debe grabar en la primera partición de la SD, usando las herramientas ubicadas en la carpeta `Imx-bootlets-src-10.05.02`.

3.1. Pasar la imagen del Kernel a la SD

La carpeta Imx-bootlets-src-10.05.02 con la configuración necesaria para el I.MX es proporcionada por Freescale, y contiene la configuración de boot del procesador, donde inicializa los buses de comunicación con las memorias, los reguladores internos de voltaje y registros de configuración.
En el directorio raíz de la carpeta hay un Makefile, el cual se debe correr para pasar la imagen del Kernel a la tarjeta SD. Se debe asegurar que la tarjeta micro SD sea leída como "sdb", puesto que así está definida la ruta en el Makefile. Si se usa un adaptador microSD a USB, el sistema leerá la SD como un dispositivo de nombre "p", por lo que se debe modificar esta ruta a /dev/p1.
Para correr el Makefile, se ejecuta el comando siguiente:

```
make ARCH=mx23 CROSS_COMPILE=arm-linux-
```

Este comando arroja error en la compilación pues no encuentra el comando “elftosb2”, el cual es el programa encargado de convertir el formato de archivos elf (Executable & Linkable File) al formato sb, que es el que reconoce el procesador.
Este programa nos lo envió el Ing. Camargo. Es un archivo binario que se debe copiar en la carpeta /usr/bin, y se le deben cambiar los permisos de ejecución:

```
sudo chmod +x /usr/bin/elftosb2
```

Así, al volver a correr el Makefile, este realiza todos los procesos sin errores, y la imágen del kernel ya queda pasada sin problemas a la SD.

Referencias

[1] Referenicas Librerias MIDI http://www.faqs.org/docs/Linux-HOWTO/MIDI-HOWTO.html

[2] Referenicas Librerias MIDI https://ccrma.stanford.edu/~craig/articles/linuxmidi/

[3] Referenicas Librerias MIDI http://linux-sound.org/midi.html