(Open)Embedded Linux Development
Introduzione && setup dell'ambiente di compilazione

Introduzione

Negli ultimi anni, la diffusione dei sistemi *Embedded* è diventata talmente capillare da attirare le attenzioni di una miriade di sviluppatori, professionisti e non: *smartphone*, sistemi di navigazione satellitare *handheld*, "*carputer*", tablet pc e sistemi domotici hanno avuto infatti un successo commerciale senza precedenti, fornendo grandi opportunità di guadagno o personalizzazione per gli usi più disparati. In questo campo il sistema operativo GNU/Linux la fa da padrone, grazie alla sua estrema scalabilità ed affidabilità. Prima di iniziare il tutorial vero e proprio, però, è necessario elencare in breve le particolarità principali di un sistema *embedded*:

- architettura del processore, solitamente ARM, profondamente diversa da quella dei normali pc per prestazioni e consumi;
- sistemi di input/output non convenzionali, come ad esempio schermi touchscreen di piccole dimensioni e/o pochi led e pulsanti;
- capacità di memorizzazione piuttosto limitata: in genere varia tra 32 MB e 1 GB di ROM e solo in alcuni casi il dispositivo è dotato di un lettore di memorie esterne.

In questo tutorial useremo come piattaforma di riferimento la scheda **mini2440** prodotta da **FriendlyARM**, basata su un processore Samsung ARM9 a 400 MHz e dotata di 64 MB di RAM e 256 MB di ROM [1]. Supponiamo inoltre che sul nostro pc desktop sia installata la distribuzione Ubuntu aggiornata alla versione 10.04.

Il modo migliore per iniziare a sviluppare su questa tipologia di sistemi è quello di creare una *toolchain* per la cross-compilazione dei propri software, cioè creare sul proprio pc desktop un ambiente che consenta di compilare i pacchetti desiderati (il proprio software + le eventuali dipendenze) per la differente architettura del processore della nostra piattaforma di sviluppo. La via più semplice ed immediata per raggiungere il nostro scopo si chiama **OpenEmbedded**, un ambiente basato su **BitBake** in grado di *buildare* non solo singoli pacchetti, ma persino intere distribuzioni! Prepariamoci quindi ad installare tutto il necessario per poi configurare correttamente OpenEmbedded!

Installazione delle dipendenze

Dopo esserci sincerati che il repository Universe sia abilitato nella nostra configurazione di apt, installiamo le dipendenze di base con il comando

 # apt-get install sed wget cvs subversion git-core coreutils unzip texi2html texinfo gawk make gcc chrpath libsdl1.2-dev docbook-utils diffstat python-pysqlite2 help2man build-essential g++ desktop-file-utils python-psyco

Talvolta, specie se si intende *buildare* la documentazione di alcuni pacchetti, potrebbe essere necessario installare anche le seguenti dipendenze opzionali:

 # apt-get install libxml2-utils xmlto docbook

Per maggiori informazioni ed una lista aggiornata dei pacchetti per tutte le principali distribuzioni, fare riferimento alla wiki del progetto alla pagina "**How OpenEmbedded fits in with your distribution**" [2].

Setup della toolchain

Scegliamo ora una cartella in cui posizionare tutto il necessario per far funzionare la nostra *toolchain*!

Creiamo, per esempio, le cartelle necessarie direttamente nella home del nostro utente e spostiamoci nella cartella "oe":

```
$ cd $HOME
$ mkdir -p oe/build/conf
$ cd oe/
```

Scarichiamo quindi il pacchetto compresso contentente BitBake ed estraiamone in contenuto in una sottocartella chiamata "bitbake":

```
$ wget http://download.berlios.de/bitbake/bitbake-1.8.18.tar.gz
$ tar xvfz bitbake-1.8.18.tar.gz && mv bitbake-1.8.18/ bitbake/
```

Facciamo il ***checkout*** del **repository git** di OpenEmbedded, contenente tutti i file (chiamati ***recipe***) necessari a BitBake, come indicato nell'apposita pagina della wiki [3] e procediamo alla configurazione della nostra *toolchain*!

Questo è un passaggio molto importante, poichè serve a specificare al compilatore l'hardware specifico e la distribuzione per i quali saranno buildati i nostri pacchetti!

[1] http://www.friendlyarm.net/products/mini2440
[2] http://wiki.openembedded.net/index.php/OEandYourDistro
[3] http://goo.gl/HvkP

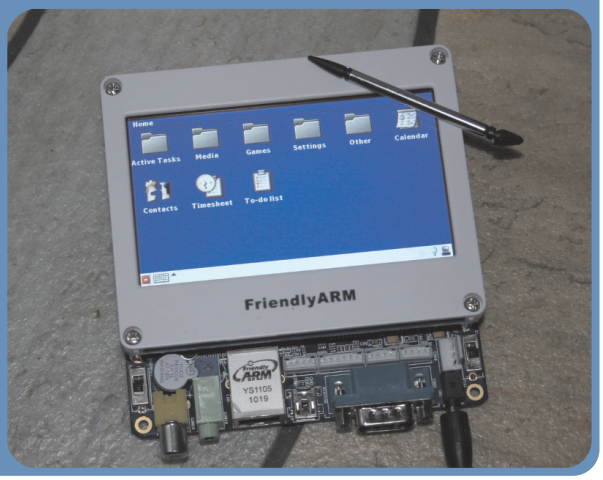
Spostiamoci nella cartella principale del nostro ambiente di lavoro e copiamo il file di configurazione d'esempio nella cartella apposita da noi precedentemente creata. Quindi adattiamolo alle nostre esigenze con il nostro editor testuale preferito (gedit, in questo caso):

$ cd $HOME/oe/
$ cp openembedded/conf/local.conf.sample build/conf/local.conf
$ gedit build/conf/local.conf

Di seguito trovate il mio local.conf, adatto a compilare software sulla scheda mini2440 per la distribuzione **Angstrom** (/home/ugoraffaele/ è la mia home utente, ndr).

```
# Where to store sources
DL_DIR = "/home/ugoraffaele/oe/downloads"
INHERIT += " rm_work "
# Make sure you have these installed
ASSUME_PROVIDED += "gdk-pixbuf-csource-native imagemagick-native librsvg-native"
# Which files do we want to parse:
BBFILES := "/home/ugoraffaele/oe/openembedded/recipes/*/*.bb"
BBMASK = ""
CACHE = "/home/ugoraffaele/oe/${DISTRO}-dev/cache"
# What kind of images do we want?
IMAGE_FSTYPES += "jffs2 ext3 tar.bz2"
# Set TMPDIR instead of defaulting it to $pwd/tmp
TMPDIR = "/home/ugoraffaele/oe/${DISTRO}-dev"
# Make use of SMP and fast disks
PARALLEL_MAKE = "-j4"
BB_NUMBER_THREADS = "4"

DISTRO = "angstrom-2008.1"
MACHINE = "mini2440"
```



La configurazione a questo punto è quasi ultimata, restano soltanto da esportare alcune variabili dell'ambiente appena scaricato nel $PATH. Questa operazione deve essere fatta ad ogni login dell'utente che intende effettuare una build, quindi per automatizzare il processo andremo a modificare con un editor di testo il file ***.profile*** presente nella nostra home!

$ gedit $HOME/.profile

Aggiungete, alla fine del file, le righe seguenti:

export BBPATH=$HOME/oe/build:$HOME/oe/openembedded
export PATH=$HOME/bitbake/bin:$PATH

La configurazione della vostra *toolchain* è terminata! Nel caso abbiate molta fretta di provarla, aprite il terminale e provate a buildare il *recipe* di prova "helloworld" con il comando

$ bitbake helloworld

BitBake scaricherà il necessario alla compilazione del pacchetto scelto ed una volta ultimata restituirà il vostro pacchetto .ipk

Un'ottima guida alla crezione di *recipe* .bb per BitBake è reperibile alla nota [4].

In conclusione, il pregio di OpenEmbedded è sicuramente quello di supportare un gran numero di dispositivi, quasi 300 al momento della stesura di questo tutorial, e ben 6 distribuzioni diverse!
La mole di materiale sullo sviluppo embedded è notevole e citare tutto in un articolo è a dir poco impossibile... Tuttavia, quest'anno sarà possibile assistere ad una dimostrazione pratica durante il GNU/Linux Day 2010 organizzato dal nostro GLUG per apprendere ancora di più riguardo questo campo affascinante! Non mancate!

[4] http://bitbake.berlios.de/manual/