Guida porting RTEMS su Raspberry Pi

Clark Ezpeleta

November 7, 2020

Contents

1 Introduzione

Guida per l'installazione di RTEMS RSB e della tool-suite per l' utilizzo del kernel di RTEMS per l'architettura ARM con target le BSP di Raspberry Pi 1 e Raspberry Pi 2, quest'ultima è compatibile con la Raspberry pi 3.

Il RSB (RTEMS Source Builder) è un tool per buildare i pacchetti dai file sorgenti. Viene usato in RTEMS per buildare i suoi compilatori e OS.

La BSP (Board Support Package) è il codice di supporto per una specifica scheda, esso contiene librerie di RTEMS utili per l'utilizzo della scheda.

Poichè RTEMS è un progetto open source ancora in sviluppo, al momento stanno sviluppando la v6, ma non è ancora stabile, per questo motivo in questa guida si installerà la v5.1 che è la versione stabile più recente.

Durante questa guida viene usato il sistema operativo su cui si basa è Ubuntu LTS 20.04.

2 Configurazione iniziale del sistema

Prima di iniziare l'installazione di RTEMS RSB e tool-suite bisogna configurare l'ambiente di sviluppo.

Procediamo con l'installazione dei seguenti pacchetti:

- build-essential :

```
$ sudo apt-get install build-essential
```

- git:

```
$ sudo apt-get install git
```

- python-dev:

```
$ sudo apt-get install python-dev
```

A questo punto definiamo la struttura delle cartelle in cui verranno installati i componenti di RTEMS.

- $HOME/rtems-dev : base directory
- $HOME/rtems-dev : base directory dove vengono clonati da git la tool-suite e la RSB di RTEMS
- $HOME/rtems-dev/src/rsb : RTEMS source builder
- $HOME/rtems-dev/src/rtems : RTEMS tool-suite
- $HOME/rtems-dev/rtems/ : dove verrà installata la RTEMS tool-suite
- $HOME/rtems-dev/build : dove verranno installate le BSP di Rpi1 e Rpi2

3 Installazione RTEMS

Dopo aver preparato l'ambiente di sviluppo, possiamo procedere con l'installazione di RTEMS:

- Clonazione di RTEMS RSB e tool-suite :

```
$ mkdir - p $HOME/rtems-dev/src
$ cd $HOME/rtems-dev/src
$ git clone -b 5.1 git://git.rtems.org/rtems-source-builder.git rsb
$ git clone -b 5.1 git://git.rtems.org/rtems.git
```

- Installiamo la tool suite utilizzando la RSB :

```
$ cd $HOME/rtems-dev/src/rsb/rtems
$ ../source-builder/sb-set-builder --prefix= $HOME/quick-start/rtems/5 5/rtems-arm
```

- Dopo aver completato l'installazione possiamo controllare che il C cross compiler di RTEMS funzioni :

```
$ $HOME/rtems-dev/rtems/5/bin/arm-rtems5-gcc --version
```

- Inseriamo nel variabili di ambiente i comandi della toolchain e procediamo con il bootstrap :

```
$ export PATH=$HOME/rtems-dev/rtems/5/bin:"$PATH"
$ cd $HOME/rtems-dev/src/rtems
$ ./rtems-bootstrap
```

- Adesso possiamo configurare ed installare le BSPs che ci servono :

```
$ mkdir -p $HOME/rtems-dev/build
$ cd $HOME/rtems-dev/build
$ $HOME/rtems-dev/src/rtems/configure \
        --prefix=$HOME/quick-start/rtems/5 \
        --target=arm-rtems5 \
        --enable-rtemsbsp="raspberrypi raspberrypi2"\
        --enable-tests=samples --enable-networking --enable-posix
$ make
$ make install
```

Svolti tutti i passaggi precedenti abbiamo come risultato RTEMS installato sul computer host.

4 Prova sample test RTEMS

Installando le BSP, RTEMS ci fornisce dei sample test .exe da cui possiamo generare i file .img e utilizzarli per testare il funzionamento della raspberry pi

I sample test sono in :

- rpi 1:

  ```
  $ $HOME/rtems-dev/build/arm-rtems5/c/raspberrypi1/testsuites/samples
  ```

- rpi 2 e 3:

  ```
  $ $HOME/rtems-dev/build/arm-rtems5/c/raspberrypi2/testsuites/samples
  ```

In questa guida utilizziamo il sample 'ticker.exe', ma i passaggi che verranno illustrati valgono anche per gli altri sample presenti in cartella. Per poter utilizzare il sample test dobbiamo fare 2 passaggi:

- Creazione kernel file .img:

  assicurarsi di avere come variabile di ambiente i comandi della tool-suite:

  ```
  $ echo $PATH
  ```

  se non è presente '$HOME/rtems-dev/rtems/5/bin' allora bisogna inserirla.

  posizionarsi nella cartella dove verrà creato il file .img:

  ```
  $ cd $HOME/rtems-dev/rtems
  ```

  generare il file .img:

  Per Rpi1:

  ```
  $ arm-rtems5-objcopy -Obinary $HOME/rtems-dev/build/arm-rtems5/c/raspberrypi1/
  testsuites/samples/ticker.exe ticker.img
  ```

  Per Rpi2 e Rpi3:

  ```
  $ arm-rtems5-objcopy -Obinary $HOME/rtems-dev/build/arm-rtems5/c/raspberrypi2/
  testsuites/samples/ticker.exe ticker.img
  ```

- Configurare la SD card: copiamo nella scheda sd il firmware di Rpi (v4.19.11.3) compatibile con RTEMS. Il firmware puo' essere scaricato da questo link:
  https://github.com/raspberrypi/firmware/tree/5574077183389cd4c65077ba18b59144ed6ccd6d/boot
  Eliminiamo tutti i file kernel*.img, questi verranno sostituiti dal file kernel che abbiamo generato precedentemente. Copiamo nella sd il file ticker.img creato precendetemente. Creiamo nella sd il file config.txt che contiene il seguente testo:

  ```
  enable_uart=1
  kernel_address=0x200000
  kernel=ticker.img
  ```

  Nel campo kernel mettiamo il nome del kernel file che vogliamo che rpi esegua.

A questo punto siamo pronti con il test.

Un comando per vedere la log della UART per debug è :

```
$ sudo minicom -b 115200 -D /dev/serial/
by-id/<indirizzo periferica utilizzata per UART di solito un USB TTL>
```

Colleghiamo il UART del Rpi al computer e vediamo il risultato:

5 Configurazione Eclipse C/C++

Per creare i file sorgenti per programmi di RTEMS possiamo utilizzare l'IDE Eclipse C/C++ scaricabile da questo link : https://www.eclipse.org/downloads/packages/

Una volta installato Eclipse C/C++ bisogna installare il plugin di RTEMS:

- Andiamo su Help¿ Install New Software
- Aggiungiamo come Software site quello di RTEMS, quindi clicchiamo add e inseriamo il seguente url ftp://ftp.rtems.org/pub/rtems/eclipse/updates
- Selezioniamo il plugin RTEMS CDT Support e installiamo

A questo punto abbiamo installato il plugin di RTEMS ed è pronto ad essere utilizzato.

Quando creiamo un progetto RTEMS bisogna configurare la base e il BSP path di RTEMS in modo che si riesca ad utilizzare le librerie che ci servono:

- Window → Preferences → C/C++ → RTEMS
- Base path : base path dell'installazione , nel nostro caso home/¡username¿/rtems-dev/rtems/5
- BSP path : path dell'installazione della BSP, nel nostro caso home/¡username¿/rtems-dev/rtems/5/arm-rtems5/raspberrypi2 (o 1 se si utilizza Rpi1)

Abbiamo completato la configurazione del plugin di RTEMS su Eclipse C, adesso possiamo procedere alla creazione di un progetto RTEMS:

- New project→ C Project
- Selezionare il project type corretto: Others→ RTEMS Executable
- Selezionare la Toochain corretta : RTEMS Toolchain

Possiamo controllare che il progetto è stato creato correttamente controllando le sue properties → C/C++ Build → RTEMS i path siano quelli corretti.

6 Prova con Eclipse C/C++

Possiamo adesso fare un semplice programma di prova "hello world", creiamo 2 file sorgenti: - init.c : conterrà il task principale (Init) che è il task di "ingresso". - system.c : conterrà il codice di configurazione di RTEMS.

in init.c inseriamo il seguente codice:

```
/*
 * Hello world example
 */
#include <rtems.h>
#include <stdlib.h>
#include <stdio.h>
#include <bsp/mmu.h>

rtems_task Init(rtems_task_argument ignored) {
        printf("\n\n*** HELLO WORLD TEST ***\n");
        printf("Hello World\n");
        printf("*** END OF HELLO WORLD TEST ***\n");
        exit(0);
}
```

in system.c inseriamo il seguente codice :

```
/*
 * Simple RTEMS configuration
 */

#define CONFIGURE_APPLICATION_DOES_NOT_NEED_CLOCK_DRIVER
#define CONFIGURE_APPLICATION_NEEDS_SIMPLE_CONSOLE_DRIVER

#define CONFIGURE_UNLIMITED_OBJECTS
#define CONFIGURE_UNIFIED_WORK_AREAS

#define CONFIGURE_RTEMS_INIT_TASKS_TABLE

#define CONFIGURE_INIT

#include <rtems/confdefs.h>
```

per una migliore comprensione del codice si rimanda ai seguenti link:

- per le api: https://devel.rtems.org/browser/rtems/bsps/arm/raspberrypi
- per la configurazione di rtems su codice (system.c): https://ftp.rtems.org/pub/rtems/people/joel/docs-eng/c-user/configuring_a_system.html

A questo punto possiamo buildare il programma, e nella cartella del progetto troveremo la cartella 'RTEMS Executable Configuration' che contiene il file .exe che utilizzeremo per creare il file .img da mettere nella scheda SD. I passaggi per la creazione del file .img sono sopracitati.

7 Creazione file kernel senza IDE

Se si vuole creare il file .img partendo dai file sorgente allora dobbiamo usare i seguenti comandi nel terminale:

- ci posizioniamo nella cartella dove ci sono i file sorgenti.
- eseguiamo i comandi del cross compiler di RTEMS su tutti i file .c:

```
$ $HOME/rtems-dev/rtems/5/bin/arm-rtems5-gcc -B \
  $HOME/rtems-dev/rtems/5/arm-rtems5/raspberrypi2/lib/ \
  -specs bsp_specs -qrtems -ffunction-sections -fdata-sections \
  -march=armv7-a -mthumb -mfpu=neon -mfloat-abi=hard -mtune=cortex-a7 \
  -Os -g -Wall -c -fmessage-length=0 -pipe -MMD -MP -MF"init.d" \
  -MT"init.o" -o "init.o" "../init.c"
```

- utilizziamo il linker per la creazione del file .exe :

```
$ $HOME/rtems-dev/rtems/5/bin/arm-rtems5-gcc -B \
  $HOME/rtems-dev/rtems/5/arm-rtems5/raspberrypi2/lib/ \
  -specs bsp_specs -qrtems -ffunction-sections -fdata-sections \
  -march=armv7-a -mthumb -mfpu=neon -mfloat-abi=hard -mtune=cortex-a7 \
  -Wl,--gc-sections -o "hello-rpi.exe"  ./init.o ./system.o
```

a questo punto possiamo creare il file .img e copiarlo sulla scheda sd e fare i passaggi già esposti precedentemente.

8 Troubleshooting

9 Sitografia

- Documentazione RTMES :
  https://docs.rtems.org/branches/master/user/index.html
- Guida RTEMS di Giorgio :
  https://gist.github.com/giorgiobasile/1c1930a8a3ff8e36061cd7f4ef83da95
- Sito spiegazione porting RTEMS (Alan Tech):
  http://alanstechnotes.blogspot.com/2013/03/rtems-on-raspberry-pi.html
- Documentazione API RTEMS:
  https://docs.rtems.org/branches/master/c-user/index.html