

shipout/backgroundshipout/foreground



# Indice

<b>1</b>	<b>Tecnologie</b>	<b>4</b>
1.1	RTEMS . . . . .	4
1.2	Raspberrypi . . . . .	4
1.3	Eclipse . . . . .	4
<b>2</b>	<b>Porting di RTEMS su RPi</b>	<b>5</b>
<b>3</b>	<b>Integrazione hardware e software</b>	<b>6</b>
<b>4</b>	<b>Attività sperimentale</b>	<b>7</b>
4.1	Obiettivi . . . . .	7
4.2	Test set-up . . . . .	7
4.3	Test application software . . . . .	7
4.4	Risultati finali . . . . .	7
<b>5</b>	<b>Conclusioni</b>	<b>8</b>

# Introduzione

Il lavoro che ho svolto ha due obiettivi principali: il porting di RTEMS su Raspberry Pi e la creazione di applicativi RTEMS per testare corretto funzionamento delle interfacce GPIO, UART, I2C, SPI e l'utilizzo degli interrupts.

Per poter svolgere il porting ho dato una lettura al RTEMS User Manual[**rtemsUM**], per comprendere meglio i concetti di RSB(RTEMS Source Builder) e BSP (board support packages), ed ho utilizzato la guida fornita da Giorgio Basile [**giorgio**], membro della BIS-Italia, che raccoglie tutti i passaggi esposti sul blog di Alan Tech per il porting della versione 4.11 Purtroppo i passaggi illustrati sul blog non sono totalmente corretti, poichè sono per il porting della versione di RTEMS precedente a quella che ho utilizzato, cioè la 5.1 che è la più recente e stabile. Dopo aver effettuato correttamente il porting ho creato una guida in italiano che raggruppa tutti i passaggi effettuati integrando le correzioni necessarie, ed è stata corretta la guida di Giorgio Basile [**giorgio5**]. A questo punto si ha l'ambiente di lavoro settato e pronto per procedere alla creazione degli applicativi RTEMS da eseguire sulla Raspberry Pi.

Per poter creare gli applicativi RTEMS ho dovuto familiarizzare con il linguaggio C, leggere l'RTEMS Classi API Guide [**rtemsCAG**] ed analizzare i codici sorgente di esempio trovati nel git repository di asuol[**asuol**] per poter comprendere l'utilizzo delle API. Tutto il lavoro è stato eseguito in modalità "smart-working", per questo motivo mi è stata spedita dalla BIS-Italia la Raspberry Pi 3B+ per poter effettuare il porting, e dalla Microchip dei componenti aggiuntivi utili per gli applicativi RTEMS di test dell'interfaccia I2C e SPI.

Oltre all'attività software ho dovuto eseguire una piccola attività hardware, cioè creare dei circuiti saldando i vari componenti e utilizzando la breadboard in modo da poterli collegare alla Raspberry Pi, per fare ciò ho seguito gli schemi elettrici che mi ha inviato Fabrizio Bernardini e ho letto i datasheet dei componenti dell'interfaccia I2C [**microchipMCP3425**] [**microchipADC**] e SPI [**microchipMCP4822**] [**microchipMSOP10-8**] in modo da comprendere il loro funzionamento e poter creare i driver.

Premesso tutto ciò ho suddiviso la mia relazione nei seguenti capitoli:

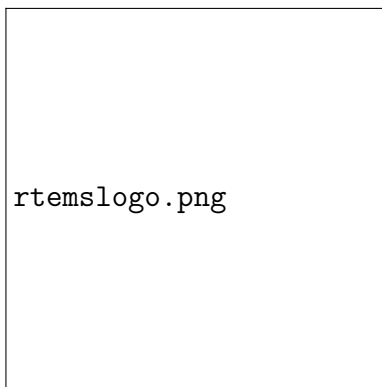
- **Capitolo 1** - in questo capitolo vengono descritte le principali tecnologie utilizzate durante il mio lavoro. Innanzitutto viene descritto RTEMS che è

il sistema operativo su cui si basa tutta l'attività, dopodichè viene descritta la Raspberry Pi su cui verranno eseguiti gli applicativi RTEMS, ed infine viene descritto Eclipse che è l'IDE utilizzato per creare gli applicativi.

- **Capitolo 2** - in questo capitolo viene descritta tutta l'attività di porting e la definizione della toolchain per poter realizzare applicativi RTEMS. Vengono esposte tutte le problematiche rilevate e le loro soluzioni.
- **Capitolo 3** - in questo capitolo vengono descritti i drivers che RTEMS ha a disposizione, la loro struttura e in che modo li ho testati e per quali motivi si è scelto di testarli.
- **Capitolo 4** - in questo capitolo viene descritta l'attività software che si basa sulla creazione degli applicativi RTEMS per testare il corretto funzionamento delle API di RTEMS per l'utilizzo delle interfacce GPIO, UART, SPI, I2C. Viene esposto anche una descrizione dei componenti aggiuntivi offerti da Microchip di cui ho creato i driver per poterli utilizzare con RTEMS.
- **Capitolo 5** - in questo capitolo viene descritto ciò che si è raggiunto e la possibile estensione del mio lavoro.

# 1 Tecnologie

## 1.1 RTEMS



RTEMS sta per Real-Time Executive MultiProcessor System ed è un sistema operativo real-time (RTOS) general purpose open source (licenza GPL 2.0 modificata) gestito da OAR Corporation. Il suo sviluppo iniziò nella fine degli anni 80 utilizzando il linguaggio Ada e C e venne usato inizialmente per scopi militari, invece le prime versioni utilizzabili sono state rese disponibili su server ftp nel 1994.

Attualmente invece viene utilizzato in molti settori tra cui quello aereo spaziale, infatti è stato utilizzato in alcune missioni spaziali ad esempio sul MRO (Mars Reconnaissance Orbiter) è in esecuzione un applicativo RTEMS che controlla il Electra UHT Transceiver (EUT) che serve per la comunicazione tra Marte e la Terra .

E' da notare che RTEMS è stato validato dall'ESA, European Space Agency, ciò vuol dire che sono stati scritti dei programmi che riproducono gli scenari critici (ad esempio la gestione di molti device oppure la gestione e l'esecuzione concorrente dei task) e sono stati eseguiti sul sistema operativo per poter verificare il suo corretto funzionamento in quei casi. La validazione viene tutt'ora fatta poiché è un sistema operativo che è in continua evoluzione e avrà più funzionalità nel tempo.

RTEMS può essere visto come un insieme di direttive raggruppate in una serie di manager, che si occupano di varie funzionalità tra cui il controllo e la sincronizzazione dei task e processori, la gestione della memoria e la mutua esclusione. Invece la gestione dello scheduling, dispatching e object management sono forniti dal executive core.

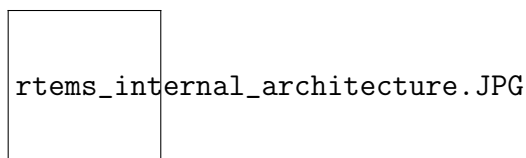


Figura 1.1: Architettura RTEMS

Utilizzando i managers di RTEMS lo sviluppatore può concentrarsi al solo sviluppo dell'applicativo e ciò riduce notevolmente il tempo di sviluppo.

Design di RTEMS :

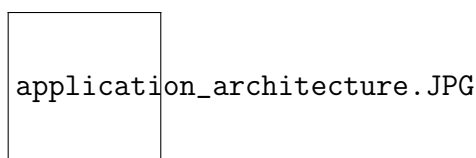
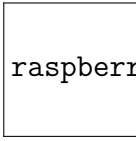


Figura 1.2: Architettura applicativo RTEMS

Da come si può notare RTEMS Executive è un intermediario tra il codice dell'applicativo e il target hardware, invece le dipendenze hardware con altri device sono localizzati nel livello "device drivers". Il RTEMS I/O manager incorpora queste dipendenze hardware nel sistema mentre allo stesso momento fornisce all'applicativo code l'accesso ad esse. Queste dipendenze hardware sono isolate in specifiche BSP, Board Support Packages, per questo motivo il porting di un applicativo RTEMS su altri processori è semplice poiché basterebbe selezionare la BSP del microprocessore su cui si vuol eseguire l'applicativo e compilare con le sue librerie.

In questo modo durante lo sviluppo di un applicativo real-time si ha la totale indipendenza dall'architettura dei microprocessori.

E' disponibile il porting di RTEMS su molte architetture CPU tra cui ARM, MIPS, LEON, ERC32 e i PowerPC; durante il mio lavoro ho trattato il porting su architettura ARM utilizzando una Raspberry Pi 3B+.

raspberrypiLogo.png

## 1.2 Raspberry Pi

Raspberry Pi è una serie di computer a scheda singola sviluppata da Raspberry Pi Foundation in collaborazione con la Broadcom, in Inghilterra.

Originariamente è stato usato per insegnare le basi dell'informatica nelle scuole e nei paesi in via di sviluppo, attualmente grazie al suo prezzo basso viene usato anche nel settore della robotica, domotica e in molti altri. Al momento sono state rilasciate quattro generazioni di Raspberry Pi, e tutti i modelli hanno un Broadcom SoC (System on Chip) con processore ARM integrato e on-chip GPU (Graphics Processing Unit).

Durante il mio lavoro viene usata la **Raspberry Pi 3B+** che utilizza il Broadcom BCM2837B0 SoC [**bcm2837**] con processore Cortex-A53 (ARMv8) 64-bit.



## 1.3 Eclipse

Eclipse è un IDE open source rilasciato con licenza EPL (Eclipse Public License) creato da IBM usato principalmente per la programmazione in Java ma grazie a vari plugin può essere usato per altri linguaggi di programmazione come C, C++, COBOL, Python e molti altri.

L'ambiente di sviluppo Eclipse include l'Eclipse Java development tools per Java, e l'Eclipse CDT per C/C++.

Per utilizzare RTEMS su Eclipse C, bisogna installare il plugin di RTEMS e settare le variabili di ambiente.

## 2 Porting di RTEMS su RPi

TODO

—

RTEMS in sé è complesso per questo il team di RTEMS ci fornisce l'ecosistema di RTEMS che è una collezione di strumenti, packages, code e documentazione, utile per definire come sviluppare, mantenere e usare RTEMS.

### **3 Integrazione hardware e software**

TODO

## **4 Attività sperimentale**

### **4.1 Obiettivi**

TODO

### **4.2 Test set-up**

TODO

### **4.3 Test application software**

TODO

### **4.4 Risultati finali**

TODO

## 5 Conclusioni

TODO

---

Il lavoro svolto sarà utile alla BIS-Italia, fazione italiana della British Interplanetary Society società storica britannica di cui sono membro, che mi ha seguito durante lo stage ed integrerà ciò al progetto di creazione di una replica in scala 1:3 di ExoMars Rover che verrà utilizzato per divulgazione. Tutto il lavoro è stato svolto con l'aiuto dei membri della BIS-Italia e la collaborazione con Microchip.

## **Ringraziamenti**