Domande da fare prima di scegliere un RTOS

Come le differenze tra RTOS possono influenzare un progetto

Tom Barrett
Presidente
Quadros Systems, Inc.

Molti sviluppatori hanno difficoltà nel valutare la scelta di un RTOS. In apparenza gli RTOS possono sembrare molto simili ma ci sono differenze fondamentali che si dovrebbe conoscere prima di prendere una decisione. Se non si è mai usato prima un RTOS, il processo di indagine può essere difficile. Non si sa quali domande porre e, nella maggior parte dei casi, non si vuole sembrare inesperti. Sembra che ovunque si guardi ci sia un Sistema Operativo Real Time. In realtà abbiamo anche visto società di microcontrollori incoraggiare i loro clienti a usare RTOS sviluppati da studenti in poche settimane.

Come fornitori di RTOS siamo un po' di parte, ma crediamo che i migliori RTOS sono quelli che hanno anni, o anche decenni, di sviluppo e clienti che ne fanno un uso continuo.

Alla fine, si deve vivere con la propria decisione. Nessuno vuole dire al proprio capo che ha scelto l'RTOS sbagliato. Ci si deve adeguare e convivere con i problemi. Quindi fate un favore a voi stessi e prendete inizialmente del tempo per guardare a quei fattori critici che possono avere un impatto significativo sulla timeline del vostro progetto e la quantità di lavoro extra che verrebbe fatto per portare il progetto a compimento, per non parlare di manutenzione, una volta che il vostro prodotto è stato commercializzato.

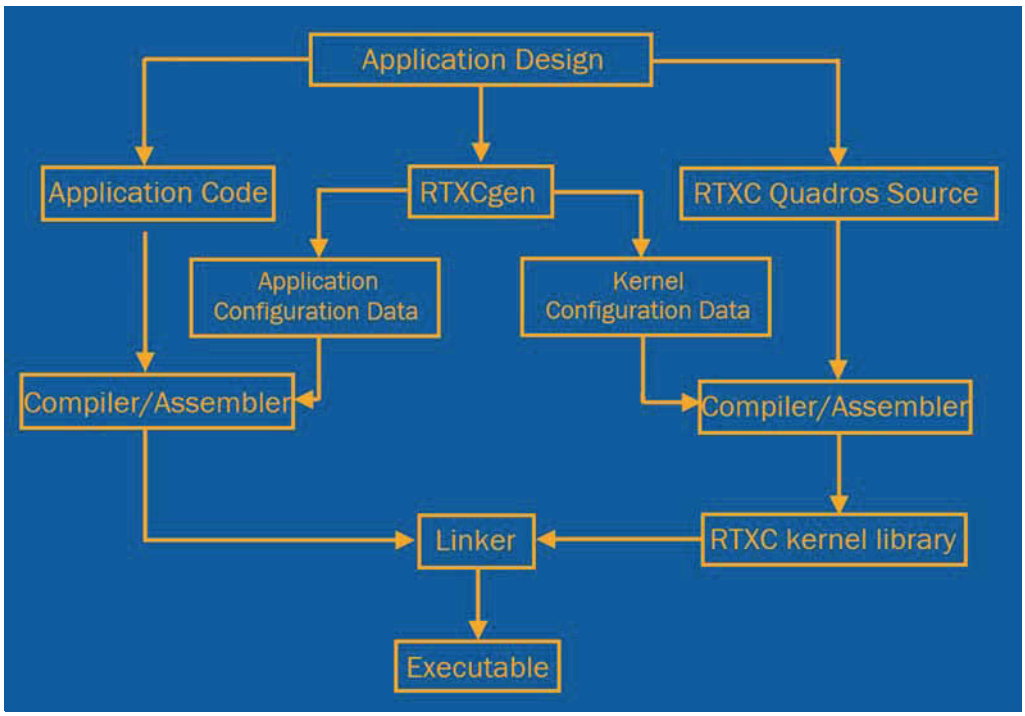

Fig. 1 – Configurazione del Kernel e dell'Applicazione con RTXCgen

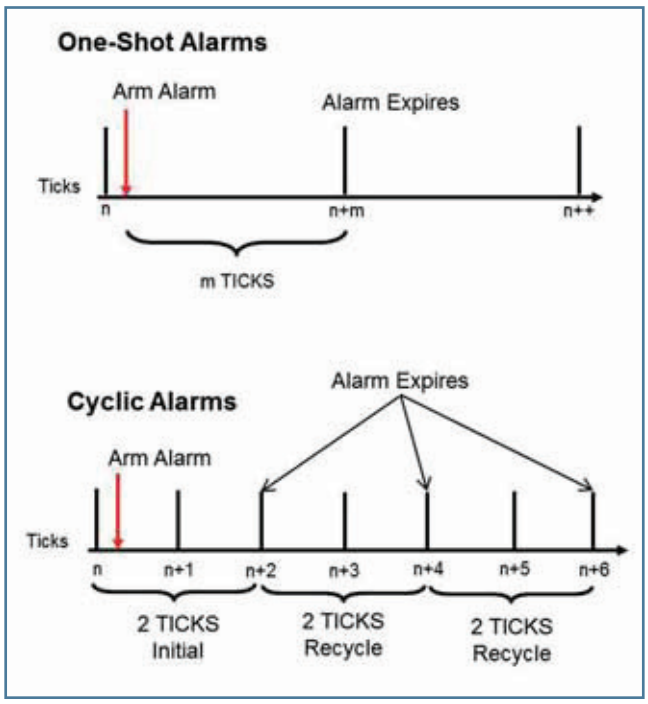

Fig. 2 – RTOS Alarms

Cose da tener presente guardando alle diverse soluzioni di RTOS:

1. Gli RTOS appaiono simili. La maggior parte sono stati progettati per realizzare multitasking. Ma come sono progettati, quali caratteristiche essi offrono e come queste siano attuate, può fare una grande differenza per il successo del vostro progetto.
2. Ogni situazione di progetto e di sviluppo è diverso. (NOTA: alcuni fornitori di RTOS offrono solo un modello one-size-fits-all e spetta quindi a voi adattare la vostra applicazione ai vincoli dell'RTOS).
3. Un elenco di funzioni non dà la risposta giusta. Non è sufficiente considerare, e alzare in graduatoria, i fornitori di RTOS in funzione del numero di caratteristiche e funzionalità. Si sta prendendo una decisione per una piattaforma che interesserà non solo il progetto in essere, ma anche gli altri a venire. Cercate di conoscere il fornitore di RTOS. Qual è la loro cultura aziendale? Come sarà il loro supporto dopo la vendita? Hanno a cuore il successo del vostro progetto o li state semplicemente aiutando a incrementare la loro quota di vendite mensile?
4. Prestare attenzione al modello di business. È importante conoscere la stabilità economica del fornitore di RTOS al fine di rimanere in attività e continuare a supportarvi. Come influisce su di voi il loro modello di business? Alcuni hanno costi iniziali bassi e poi richiedono costi elevati per ogni add-on e modifiche. Altri forniscono disponibilità per ogni processore conosciuto ma con un supporto tecnico molto limitato. Altri regalano la versione base ma fanno pagare cifre importanti per i componenti aggiuntivi.

Di seguito seguenti troverete informazioni dettagliate da chiedere quando parlate con i vari fornitori di RTOS. Anche se crediamo che la famiglia di Sistemi Operativi Real Time "RTXC" sia qualitativamente alta, l'obiettivo finale è farvi prendere una decisione che sia giusta per la vostra situazione. Ci auguriamo che queste informazioni facciano meglio comprendere alcuni fattori importanti.

Le pagine che seguono intendono aiutare a rimuovere gli strati superficiali e iniziare a capire alcune differenze tra le implementazioni di RTOS.

RTOS: architettura & caratteristiche

L'RTOS fornisce una gamma flessibile di modalità di scheduling? Un sistema embedded ben progettato può avere bisogno di una combinazione di due o più modalità di scheduling: Preemptive, Round robin, Cooperative, Time-sliced, Slicing con una ulteriore variabile oltre al tempo (ad esempio rotazione angolare, flusso, tick aperiodico o periodico).

Tutti gli oggetti di sistema (entità di codice, trasferimento dati, sincronizzazione e così via) devono essere stabiliti prima che l'applicazione possa avere successo. La maggior parte degli RTOS possono creare oggetti statici a runtime. Può l' RTOS creare oggetti dinamici a runtime? Solo i task o tutti gli oggetti?

L'RTOS utilizza l'indirizzo di un oggetto dinamico come

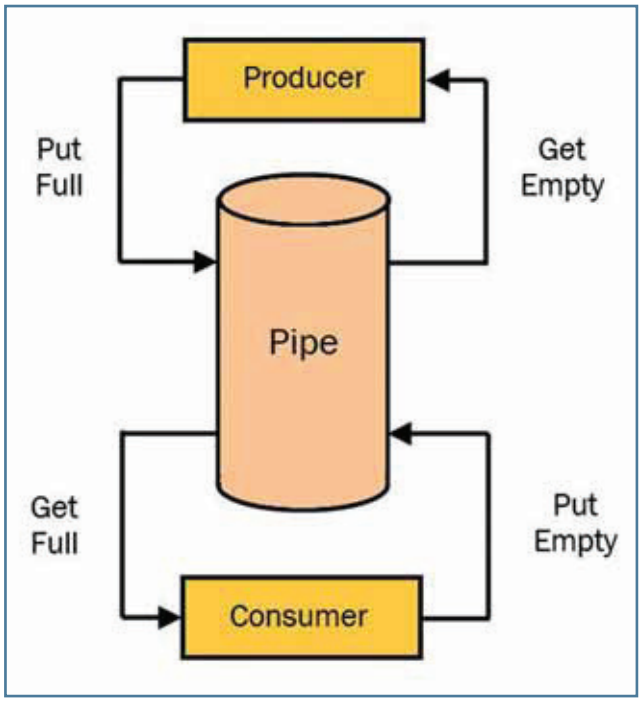

Fig. 3 – "Data passing" con Pipe

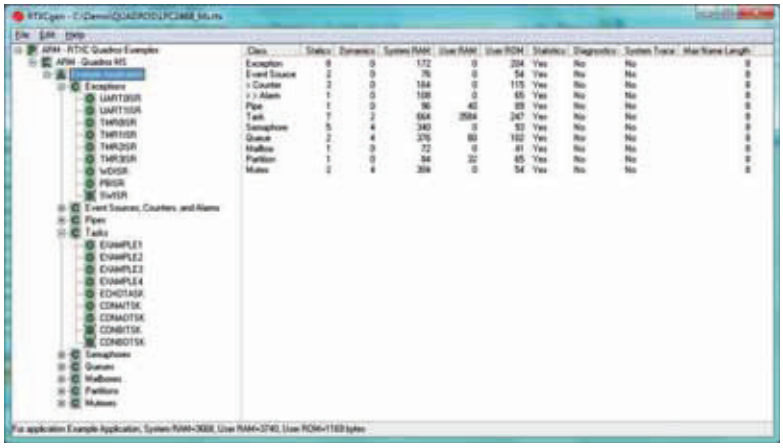

Fig. 4 – Tool di configurazione RTXCgen

suo identificatore? Ciò può portare al danneggiamento delle strutture interne dell'RTOS e ad un possibile errore.
Gli interrupt sono gestiti con una macro / funzione o ci si deve scrivere il proprio prologo (entry) ed epilogo (exit)? L'approccio a macro potrebbe far risparmiare un sacco di tempo di sviluppo - codifica e testing.
L'RTOS si sincronizza con gli eventi? Gli oggetti di sincronizzazione eventi si cancellano automaticamente su rilascio di un task oppure gli eventi possono essere "dimenticati"? Possono più tasks rimanere in attesa (wait) di un singolo evento? Può un singolo task rimanere in attesa di più eventi?
Molti RTOS limitano il conteggio per tick da un interrupt periodico per stabilire il system time. Qualsiasi altra forma di conteggio è lasciata all'utente. L'RTOS consente di contare entrambi i tick periodici o aperiodici?
Oltre alla variabile tempo, può l'RTOS contare anche i tick? Il tempo può non essere la sola variabile indipendente in un'applicazione.
Il conteggio della posizione angolare o dello spostamento è utile per i sistemi che dispongono di elementi fisici che ruotano a velocità variabile come il controllo motore, sistemi ABS, contatori a turbina, ecc. I Ticks potrebbero misurare la portata attraverso un contatore volumetrico. Non forzate il sistema in una struttura basata sul tempo se qualche altro meccanismo di conteggio è più adeguato.
Alcuni RTOS usano Timers; altri usano Allarmi. Lo scopo è quello di iniziare un'azione futura. Sono Timer / Allarmi disponibili a livello globale in modo che possano essere utilizzati da più task o sono legati ad una singola attività? L' RTOS consente di definire le azioni future da intraprendere quando un contatore raggiunge un valore predefinito?

La migliore gestione della RAM da parte di un RTOS è di solito fatta con partizioni in cui ogni blocco è della stessa dimensione. Questo impedisce la frammentazione e consente allocazioni deterministiche.
L' RTOS gestisce la RAM con un heap che possa creare una risposta non-deterministica e la frammentazione? Ciò può aggiungere ulteriore sovraccarico.
Possono i blocchi di una partizione di memoria essere usati per creare un'altra partizione dinamicamente?
Le applicazioni si passano dati da una entità di codice a un'altra. Queste operazioni di 'data passing' vengono identificate con diversi nomi: Queues, Mailboxes/Messages, Pipes. La maggior parte dello scambio dati è o con blocchi a dimensione fissa o a dimensione variabile.

- L'RTOS offre diverse modalità di scambio dati (data passing)? Maggiori sono le possibilità di scelta, migliore è la flessibilità per soddisfare i requisiti dell'applicazione.
- L'RTOS permette che i dati vengano scambiati tra i task e le ISR (o solo tra tasks)? Ciò aumenta l'efficienza del sistema.
- **In che modo l'RTOS** fornisce accesso esclusivo alle risorse? Utilizza binary semaphores o mutexes? Mutexes sono generalmente migliori perchè conoscono l'identità del task proprietario.

L'RTOS ha un meccanismo per evitare l'inversione di priorità (priority inversion) – quando un task a bassa priorità ha il controllo di una risorsa che è richiesta da un task con priorità superiore?

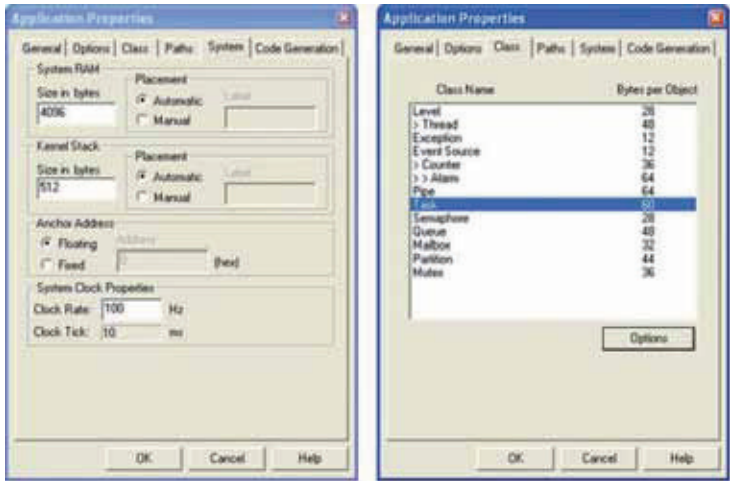

Fig. 5 – Proprietà dell'applicazione specificate col tool di configurazione RTXCgen

Solo perché un RTOS è real-time non significa che sia veloce. Chiedete al fornitore di RTOS come è stato sviluppato. È progettato per il funzionamento deterministico? Garantisce un basso overhead di sistema? Bassa latenza? Servizi di risposta?
Chiedete di vedere le API dell'RTOS. Alcuni RTOS forniscono solo poche primitive di servizio mentre altri ne incorporano diverse centinaia. Più l'RTOS è ricco di API, più è probabile che possa essere facilmente configurato per supportare la vostra applicazione.

Stack di comunicazione & middleware

La maggior parte degli RTOS commerciali offrono protocolli di comunicazione e middleware. Questo vi dà il vantaggio di un unico punto di riferimento di integrazione e supporto per questi prodotti. Ma ci sono domande importanti da porre:

- Alcuni fornitori RTOS sono rivenditori di software di terze parti. Cosa succede se c'è un problema? Sono in grado di aiutarvi in modo diretto o fanno riferimento a qualcun altro?
- Il software è bene integrato? Il fornitore di RTOS fornisce applicazioni di esempio che integrano le varie parti?
- Il fornitore RTOS può avere gli stack di comunicazione e middleware che avete bisogno oggi, ma per quanto riguarda il prossimo progetto? Saranno in grado di supportare tali requisiti? Fate attenzione a non arenarvi su una piccola isola.

Getting started & configuration tool

Che cosa fa il fornitore di RTOS per rendere più facile l'inizio dello sviluppo? Che tipo di assistenza di start-up è disponibile? Fornisce progetti di esempio legati all'ambiente di sviluppo? Codice di esempio? Esempi di ISR e driver? Chiamate one-to-one come guida introduttiva ("Getting Started")?
L'occupazione di memoria è una questione importante. Alcuni RTOS sono molto piccoli perché sono molto semplici. Altri forniscono un ricco contesto che può essere ridotto per adattarlo alle esigenze dell'applicazione. Può l'RTOS essere facilmente ridotto in dimensioni? Ci sono strumenti che consentono di automatizzare questo processo?
Alcuni RTOS hanno dimensione fissa, mentre altri sono scalabili. Una maggiore scalabilità permette di effettuare piccoli aggiustamenti a qualsiasi componente con lo scopo di creare una configurazione che si adatti alle esigenze dell'applicazione.
Sempre più RTOS necessitano di alcune forme di configurazione prima di poter essere utilizzati. Quanto è facile adattare e configurare l'RTOS? Ci sono tools che automatizzano questo processo? Ci sono molti approcci per la configurazione. Qual'è il migliore per voi? Quale vi consente di salvare tempo e fatica?
Dovete modificare i file sorgenti manualmente? Può anche essere facile da farsi, ma è soggetto a errori.
L'RTOS utilizza codice runtime per la creazione di oggetti? Questo può coinvolgere un sacco di codifica noiosa ed è soggetta a errori e tempo di debug supplementare.
L'RTOS include anche un tool (GUI) di configurazione? Chiedete di valutare il tool. Che possibilità vi fornisce? Vi consente di configurare sia l'RTOS che gli oggetti dell'applicazione?
E la documentazione utente? Quanto è completa? Fornisce descrizioni chiare di come l'RTOS e i suoi vari oggetti funzionano? Ci sono pagine di riferimento per ogni servizio di kernel, compresa la sua descrizione, parametri di chiamata e di ritorno, eventuali condizioni di errore, e un esempio di ciascun servizio?
L'RTOS offre un application design tool integrato? Un simile tool può far risparmiare settimane di sviluppo. La sua struttura organizzata può aiutare a meglio gestire, comprendere e comunicare il vostro progetto ad altri.

Altri fattori

Un RTOS può essere paragonato agli strumenti di un falegname professionista. Un utente non informato potrebbe decidere di utilizzare uno scalpello per stringere una vite senza rendersi conto che per tale scopo esiste l'avvitatore elettrico con la punta specifica. Sono disponibili training dal produttore di RTOS?
Chi gestisce le chiamate di supporto? Gli attuali sviluppatori dell'RTOS sono indubbiamente il miglior team di supporto (loro conoscono il codice e sono i più qualificati a rintracciare un problema).
Avete la garanzia che l'RTOS attuale sia in grado di soddisfare le esigenze future? Un RTOS commerciale ben supportato e manutenuto può far sì che la vostra applicazione sia a prova di futuro.
Selezionare il giusto RTOS per il vostro progetto può fare una grande differenza per il succcesso del vostro prodotto. Piccole differenze tra i sistemi operativi possono aggiungere giorni, o addirittura settimane di lavoro extra per il vostro team di sviluppo. Siate sicuri di fare le domande giuste prima di prendere una decisione.

Nota

I Sistemi Operativi Real Time RTXC di Quadros Systems sono distribuiti in Italia da Fenway Embedded Systems, Via Don Giovanni Minzoni 31, 20010 Arluno (MI) – Italy, Tel. +39 02 97310120, sales@fenwayembedded.com, www.fenwayembedded.com