LABVIEW REAL-TIME: BILANCIARE IL LAVORO FRA LE CPU

Gerardo Garcia

Vediamo come specificare il set di CPU disponibili per il bilanciamento automatico del carico di lavoro in LabVIEW Real-Time

Il modulo Real-Time di LabVIEW 8.5 include le estensioni NI RT per il multiprocessing simmetrico (SMP), che potete installare per trarre vantaggio da sistemi multicore dotati fino a un massimo di 32 CPU. L'installazione di tali estensioni aggiunge funzionalità di bilanciamento automatico del carico di lavoro, in grado di distribuire i thread delle applicazioni fra le diverse CPU. Questo articolo discute i concetti di bilanciamento automatico in LabVIEW Real-Time e introduce l'API SMP, che potete utilizzare per specificare il set di CPU disponibili e riservarne alcune per l'uso esclusivo da parte di specifiche *Timed Structures*.
Consultate l'help di LabVIEW 8.5 Real-Time Module per un'introduzione alla programmazione multi-core in LabVIEW Real-Time e ni.com/multicore per avere maggiori informazioni sui concetti relativi alle tecnologie multi-core. Utilizzate il LabVIEW Real-Time Software Wizard in Measurement & Automation Explorer per installare le estensioni NI RT per l'SMP. Infine, consultate l'help di Measurement & Automation Explorer per informazioni sull'uso di Real-Time Software Wizard.

POOL DI CPU

Con le Estensioni NI RT per l'SMP installate, il sistema operativo real-time (Rtos) ETS mantiene due pool di CPU disponibili per il bilanciamento automatico del carico: un pool per il sistema operativ (OS) ed un pool per le Timed Structure (TS). Le CPU nel pool TS sono disponibili per il bilanciamento automatico del carico delle Timed Structures (Timed Loops e Timed Sequences) configurate per l'Assegnazione **Automatica** della CPU. Le CPU nel pool OS sono utilizzate per tutti gli altri processi, inclusi i task a basso livello del sistema operativo.

BILANCIAMENTO AUTOMATICO DEL CARICO

Il bilanciamento automatico del carico è il processo di assegnazione di thread alle CPU in modo da bilanciare il carico elaborativo fra le CPU. All'avvio, LabVIEW 8.5 Real-Time Module assegna tutte le CPU disponibili al pool OS ed al pool TS, quindi esegue il bilanciamento automatico del carico fra tutte le CPU.
Tuttavia, potete utilizzare l'API SMP introdotta in questo articolo per specificare pool OS e TS arbitrari.
LabVIEW Real-Time esegue il bilanciamento automatico del carico su tutti i thread, con l'eccezione dei thread Timed Structure assegnati manualmente. LabVIEW mappa l'esecuzione del codice da ciascuna Timed Structure al proprio thread dedicato, ma esegue il bilanciamento automatico del carico solo sui thread Timed Structure configurati per l'assegnazione automatica della CPU.
Potete assegnare manualmente una Timed Structure ad una particolare CPU collegando l'indice della CPU all'ingresso **Assigned CPU** della Timed Structure o utilizzando la parte **Processor Assignment** della corrispondente finestra di dialogo Configure Timed Structure. Il valore di **Assigned CPU** di default, pari a –2, indica che la Timed Structure è configurata per l'assegnazione automatica del processore attraverso il bilanciamento automatico del carico.
Generalmente, il processo di bilanciamento automatico del carico evita la necessità di commutare un dato thread fra le CPU, per minimizzare l'overhead di trasferimento dati fra le CPU. Tuttavia, lo schedulatore SMP potrebbe commutare ripetutamente un thread da una CPU a un'altra se il carico dovuto ad altri thread di priorità pari o superiore eseguiti su ciascuna CPU fluttua in modo significativo. Potete utilizzare il Real-Time Execution Trace Kit per determinare se il thread di una Timed Structure si sposta da CPU a CPU.
Per fare in modo che lo schedulatore SMP non assegni inaspettatamente Timed Structures ad una particolare CPU, potete rimuovere tale CPU dal pool TS usando l'API SMP descritta sotto.
Tuttavia, per fare in modo che una Timed Structure abbia l'accesso esclusivo ad una CPU assegnata manualmente, dovete anche rimuovere quella CPU dal pool OS. Così facendo, potete massimizzare le prestazioni di un Timed Loop deterministico, come descritto nel paragrafo **Isolare una Timed Structure** di questo articolo.

CONFIGURAZIONI DEI POOL

Ci sono quattro possibili stati per una CPU in un sistema SMP LabVIEW Real-Time. Una CPU può appartenere:

1. Al pool TS, nel qual caso lo schedulatore SMP può utilizzare la CPU per eseguire Timed Structures configurate per l'Assegnazione Automatica della CPU.
2. Al pool OS, nel qual caso lo schedulatore SMP può utilizzare la CPU per eseguire thread che non corrispondono a Timed Structures.
3. Ad entrambi i pool, nel qual caso lo schedulatore SMP può utilizzare la CPU in uno dei due modi precedenti.
4. A nessun pool (riservata), nel qual caso la CPU è dedicata solamente all'esecuzione di Timed Structures che vengano assegnate manualmente a quella CPU.

Quando un target RT esegue il boot up, tutte le CPU sono piazzate in entrambi i pool per default.
Tuttavia, potete utilizzare l'API SMP per assegnare ciascuna CPU ad uno qualsiasi dei quattro stati elencati sopra.

Nota: *Ogni pool deve contenere almeno una CPU, anche se entrambi i pool possono condividere una singola CPU.*

ISOLARE UNA TIMED STRUCTURE DETERMINISTICA

Per massimizzare le prestazioni e minimizzare il jitter in un Timed Loop deterministico, potete isolare quella Timed Structure su una particolare CPU. Per isolare una Timed Structure, dovere assegnare la Timed Structure ad una CPU che non è contenuta né nel pool OS, né nel pool TS.
Quando una CPU non è assegnata ad alcun pool, essa è riservata per eseguire solo la/le Timed Structure(s) che assegnate manualmente a quella CPU. Isolare una Timed Structure su una singola CPU vi permette di monopolizzare la capacità elaborativa di quella CPU e raggiungere un'elevata frequenza o un elevato throughput. Per esempio, potete sfruttare l'isolamento per raggiungere un'elevata frequenza di polling in un Timed Loop che esegue l'acquisizione dati.
Per minimizzare la latenza di un Timed Loop deterministico ad alte prestazioni, prendete in considerazione di isolare quel Timed Loop su una CPU ad indice elevato.
Per esempio, se il sistema contiene quattro core CPU con indici 0-3, considerare l'isolamento del Timed Loop deterministico sulla CPU 3. Quando le Timed Structures vengono schedulate, il sistema operativo real-time inizia dalla CPU con indice più alto e prosegue verso il basso.
Pertanto, se più Timed Structures di pari priorità sono schedulate per risvegliarsi nello stesso istante, la latenza di risveglio delle Timed Structures eseguite su CPU con indice minore può essere più alta di quella delle Timed Structures eseguite su core con indice maggiore, anche di diversi microsecondi.
È vero il contrario per i thread non Timed Loop, perché per tali thread lo schedulatore inizia dalla CPU 0 e prosegue verso l'alto. Quindi, per minimizzare la latenza, National Instruments raccomanda di assegnare le CPU di numero inferiore al pool OS e di riservare le CPU di numero superiore ai Timed Loops deterministici.

PREVENIRE IL RALLENTAMENTO DEI THREAD DELL'OS

Per prevenire il rallentamento dei thread dell'OS, considerate la possibilità di riservare almeno una CPU per i soli thread dell'OS. Per esempio, se assegnate la CPU 0 al pool OS ma non al pool TS, ed evitate di targetizzare Timed Structures sulla CPU 0, la CPU 0 è sempre disponibile per l'esecuzione dei thread dell'OS.

MASSIMIZZARE L'UTILIZZO DELLE CPU

Per massimizzare l'utilizzo dei processori, è utile determinare il numero di CPU in ciascun pool in base alla proporzione di carico che corrisponde ai thread del TS rispetto ai thread non TS. Potete usare il VI Get Core Loads allegato sotto per stimare la proporzione del tempo di elaborazione totale dedicata ai thread TS e non TS ed assegnare le CPU ai pool TS e OS di conseguenza. Se il target RT è collegato ad un monitor, potete utilizzare anche l'utilità di Misura del Carico della CPU per stimare la distribuzione del carico.

EVITARE LA SOVRAPPOSIZIONE PARZIALE DEI POOL

Se definite i pool OS e TS in modo che i due pool si sovrappongano parzialmente, il processo di bilanciamento automatico del carico potrebbe non riuscire a prendere decisioni ottimali di utilizzo dei processori, come illustrato nei seguenti esempi.

Esempio 1

Supponiamo che abbiate un'applicazione con tre Timed Loops. I periodi dei Timed Loops sono configurati in modo che, per la maggior parte del tempo, sia eseguito solo uno dei tre Timed Loops. Tuttavia, occasionalmente i periodi dei Timed Loops si allineano e volete fare in modo che quando ciò si verifica i tre Timed Loops possano essere eseguiti tutti in parallelo. Potreste essere tentati di assegnare tre CPU (per es. le CPU 0-2) di un sistema quad-core al pool OS e tre CPU (per es. le CPU 1-3) al pool TS, assumendo che la CPU 3 eseguirà sempre uno dei tre Timed Loops, mentre le CPU 1 e 2 eseguiranno generalmente thread dell'OS ed eseguiranno Timed Loops solo quando i periodi si allineano. Tuttavia, poiché lo schedulatore SMP cerca di eseguire ciascun thread sulla stessa CPU da iterazione ad iterazione, i tre Timed Loops potrebbero essere eseguiti tutti sulla CPU 1 o sulla CPU 2, riducendo quindi la capacità di elaborazione disponibile per i thread dell'OS e lasciando la CPU 3 inattiva. In questo caso, una soluzione migliore sarebbe quella di assegnare manualmente ciascun Timed Loop ad una CPU differente ed assegnare tutti i quattro core al pool OS.

Esempio 2

Supponiamo che abbiate un'applicazione con due Timed Loops che eseguono un polling continuo e due Timed Loops che sono eseguiti in modo intermittente. Si potrebbe pensare di assegnare tutte e quattro le CPU di un sistema quad-

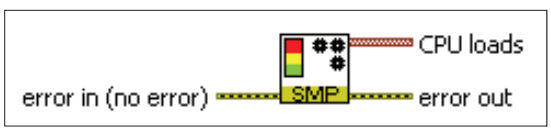
core al pool TS e due delle CPU, per es. le CPU 2 e 3, al pool OS. Potreste assumere che i due Timed Loops di polling saranno eseguiti sulle CPU 0 e 1 e che i Timed Loops intermittenti condivideranno il tempo di elaborazione sulle CPU 2 e 3. Tuttavia, lo schedulatore SMP potrebbe assegnare i Timed Loops di polling alle CPU 2 e 3, appesantendo gli altri thread dell'OS e lasciando sottoutilizzate le CPU 0 e 1. In questo caso, la soluzione migliore sarebbe quella di isolare i Timed Loops di polling sulle due CPU ed assegnare le due CPU rimanenti ad entrambi i pool.

API DI POOL UTILITIES PER SMP CPU

Usate l'API di Pool Utilities per SMP CPU, descritta sotto, per definire i pool OS e TS durante la fase di configurazione iniziale di un'applicazione. Potete ottenere i VI SMP CPU Pool Utilities scaricando il file SMP_CPU_Utilities.zip file.

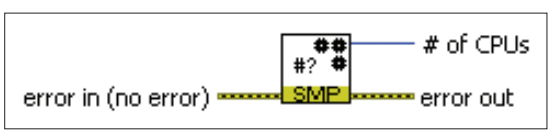
Per installare l'API, estraete il contenuto del file SMP_CPU_Utilities.zip nel folder \labview 8.5\user.lib sul vostro computer. Dopo avere estratto i file nel folder user.lib, la prossima volta che lancerete LabVIEW apparirà la palette SMP CPU Utilities come sottopalette della palette User Libraries.

Nota: *L'API SMP CPU Pool Utilities è supportata solo su target RT multi-CPU che eseguono il sistema operativo ETS Phar Lap con le NI RT Extensions per SMP installate.*



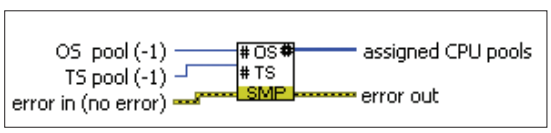
Get CPU Loads.vi

Monitorizza la distribuzione del carico fra le CPU nel sistema. Per ogni CPU nel sistema, questo VI restituisce il carico totale come percentuale della capacità, nonché la percentuale della capacità totale dedicata alle routine di servizio degli interrupt (ISR), alle Timed Structures e a tutti gli altri thread. L'n-simo elemento di ciascun array corrisponde all'n-sima CPU nel sistema.



Get Number of CPUs.vi

Restituisce il numero di CPU nel sistema.



Set CPU Pool Sizes.vi

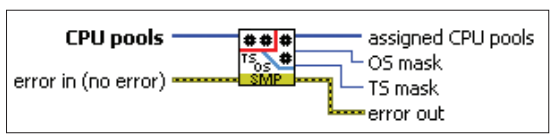
Imposta il numero di CPU in ciascun pool. Potete utilizzare questo VI per definire i pool OS e TS specificando il numero di CPU che volete sia contenuto in ciascun pool. Questo VI crea i pool OS e TS come pool adiacenti di CPU contigue.

Il pool OS inizia dalla CPU 0 ed il pool TS inizia dove finisce il pool OS.

Per esempio, su un sistema con otto CPU, se collegate un valore 3 ai controlli **OS Pool** e **TS Pool**, questo VI assegna le CPU 0-2 al pool OS, le CPU 3-5 al pool TS e lascia le CPU 6-7 riservate per l'uso da parte delle Timed Structures configurate per l'assegnazione manuale della CPU.

Nota: *Non potete utilizzare questo VI per creare pool vuoti o parzialmente sovrapposti. Questo VI restituisce un errore se specificate una dimensione 0 del pool OS o del pool TS o se i valori del pool OS e del pool TS che specificate danno una somma maggiore del numero di CPU disponibili nel sistema.*

Specificando un valore -1 per uno dei pool si indica che non si hanno preferenze di dimensioni per quel pool. Se si specifica un valore -1 per entrambi i pool, il VI crea la configurazione di default del pool, in cui entrambi i pool contengono ogni CPU nel sistema. Se si specifica una dimensione del pool pari a -1 per un pool, il VI assegna tutte le CPU rimanenti a quel pool. Per esempio, in un sistema a quattro CPU, se si specifica una dimensione del pool di -1 per il pool OS ed una dimensione del pool di 2 per il pool TS, il VI assegna le CPU 0 e 1 al pool OS e le CPU 2 e 3 al pool TS.

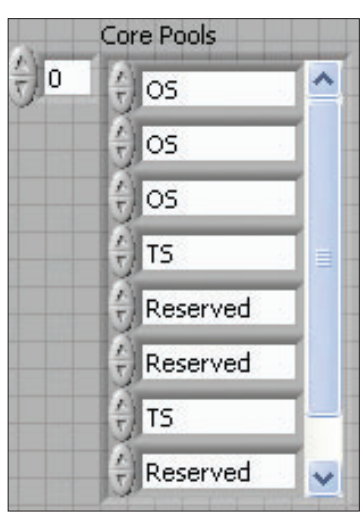


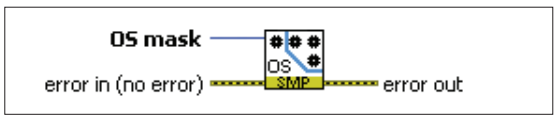
Set CPU Pool Assignment

Assegna le CPU ad uno di quattro possibili stati: solo pool OS, solo pool TS, entrambi i pool o nessun pool (riservata). Questo VI genera le maschere di bit che specificano le CPU assegnate a ciascun pool. In un sistema con n CPU, i bit della maschera di bit corrispondono alle CPU da 0 a n -1.

Il bit più a destra di ciascuna maschera di bit corrisponde alla CPU 0 ed il bit più a sinistra corrisponde alla CPU 31 (se tale CPU esiste nel sistema).

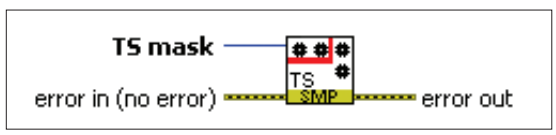
L'ingresso a questo VI è un array di enum. L'enum contiene i quattro possibili stati di una CPU e ciascun elemento dell'array rappresenta una CPU. Per esempio, nell'ipotesi di un sistema con otto CPU, l'array di enum nella figura qui sotto assegna le CPU 0, 1 e 2 al pool OS, le CPU 3 e 6 al pool TS e riserva le CPU rimanenti.





Set OS Pool.vi
Specifica il set di CPU contenute nel pool OS. L'ingresso **OS mask** è una maschera di bit nella quale il bit più a destra corrisponde alla CPU 0 ed il bit più a sinistra corrisponde alla CPU 31. Per assegnare un core CPU al pool OS, impostate il valore del bit corrispondente a 1. Per specificare che il pool OS non deve contenere una particolare CPU, impostate il valore del bit corrispondente a 0.

Nota: *Il pool OS deve contenere almeno una CPU valida.*

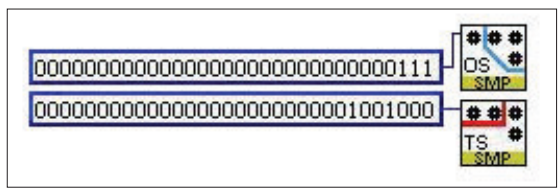


Set TS Pool.vi
Specifica il set di CPU contenute nel pool TS. L'ingresso **TS mask** è una maschera di bit nella quale il bit più a destra corrisponde alla CPU 0 ed il bit più a sinistra corrisponde alla CPU 31. Per assegnare un core CPU al pool TS, impostate il valore del bit corrispondente a 1. Per specificare che il pool TS non deve contenere una particolare CPU, impostate il valore del bit corrispondente a 0.

Nota: *Il pool TS deve contenere almeno una CPU valida.*

PANORAMICA SULL'API DI POOL UTILITIES PER SMP CPU

L'API di Utilities per SMP CPU offre un set di VI che potete utilizzare per monitorare il carico delle CPU e per assegnare le CPU ai pool TS e OS per il bilanciamento automatico del carico. L'API fornisce tre livelli di astrazione, pertanto potete scegliere il livello di astrazione che corrisponde meglio al vostro stile di programmazione ed alle esigenze della vostra applicazione.
Il VI di livello più elevato, Set CPU Pool Sizes.vi, è costruito dal VI di livello intermedio Set CPU Pool Assignments.vi, che a sua volta è costruito dai VI avanzati di basso livello Set OS Pool.vi e Set TS Pool.vi.
Il VI di alto livello, Set CPU Pool Sizes.vi, è una buona scelta per la maggior parte dei casi d'uso. Potete utilizzare il VI Set CPU Pool Sizes per creare la configurazione di default dei pool, in cui entrambi i pool contengono tutte le CPU, o per creare pool contigui e adiacenti.
Per configurationi custom dei pool di CPU, il VI di livello intermedio Set CPU Pool Assignments.vi è una buona scelta. Potete usare il VI CPU Pool Assignments per creare qualsiasi configurazione dei pool, purché ciascun pool contenga almeno una CPU. Se preferite lavorare con le maschere di bit, potete utilizzare i VI di basso livello sulla sottopalette Advanced. Per esempio, la figura seguente utilizza i VI di



basso livello set OS Pool.vi e Set TS Pool.vi per ottenere la stessa configurazione dei pool dell'array di enum descritto nell'esempio precedente.

CONCLUSIONE

Il Modulo Real-Time di LabVIEW 8.5 utilizza il bilanciamento automatico del carico per distribuire i thread fra due pool di CPU. Le CPU nel pool OS eseguono tutti i thread ad eccezione di quelli che corrispondono a Timed Structures. Le CPU nel pool Timed Structure (TS) esegue i thread che corrispondono a Timed Structures configurate per l'assegnazione automatica delle CPU. Per default, LabVIEW Real-Time assegna tutte le CPU ad entrambi i pool. Tuttavia, potete utilizzare l'API di Pool Utilities per SMP CPU per definire i pool OS e TS in modo da rispondere le vostre specifiche esigenze applicative.
Per installare l'API, estraete il contenuto del file SMP_CPU_Utilities.zip nel folder \labview 8.5\user.lib folder sul vostro computer. Dopo l'estrazione dei file nel folder user.lib folder, la prossima volta che eseguite LabVIEW appare la palette SMP CPU Utilities come sottopalette della palette User Libraries.

Potete scaricare il file dell'API descritta in questo articolo al seguente link: ni.com/italian (info code: it3nch)

Note sull'autore:
Gerardo Garcia è il product manager di LabVIEW Real-Time. È laureato in Ingegneria Elettrica alla Texas A&M University