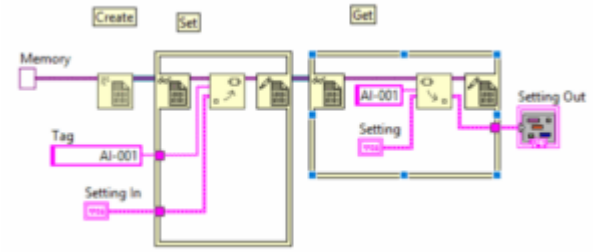
LabVIEW DVR

Category: LabVIEW Talk
scritto da admin | Maggio 28, 2023



Cosa sono i DVR (Data Value Reference)?

Nel modello di programmazione dataflow di LabVIEW, i dati vengono trasmessi in modo by value, dovendo comunicare tra loop paralleli, si deve ricorrere a variabili o altri meccanismi, generando duplicazioni del dato in memoria.

Quando si devono condividere grandi quantità di dati o aggiornare lo stesso dato in **esecuzione parallela**, la duplicazione può essere un serio problema di allocazione.

LabVIEW offre diversi metodi per “passare” dati tra loop, tra cui:

Variabili Locali:

consentono di leggere/scrivere dati all’interno dello stesso VI, ma non sono adatte per la comunicazione tra loop paralleli.

Variabili Globali:

offrono la condivisione dei dati tra VI differenti, pur avendo il limite di creare delle copie durante ogni accesso.

Meccanismi di Sincronizzazione by Reference:

come code (queues), notifier e altri, che permettono di comunicare tra processi con meccanismi di sincronizzazione.

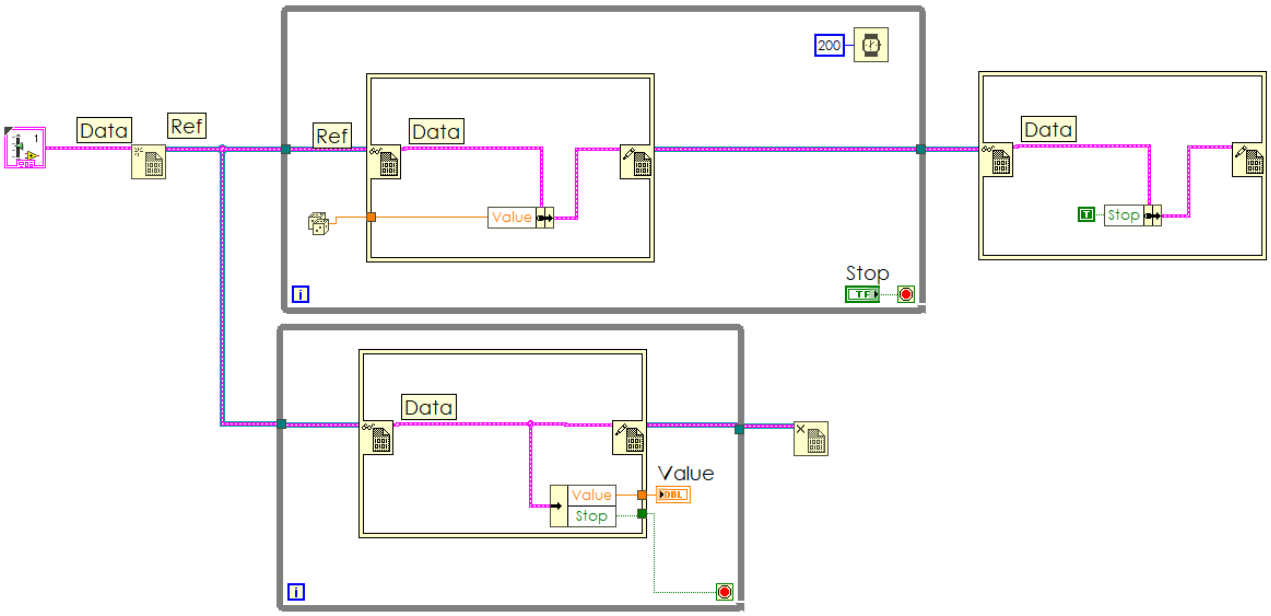
Allora, perché usare i DVR?

I Data Value Reference (DVR) offrono un approccio ibrido:

permettono di gestire internamente i dati come se fossero trattati “by value” (facilitando la costruzione, l’organizzazione e il debug del codice) e, contemporaneamente, di “passare” un semplice riferimento (by reference) a loop o thread paralleli per consentire l’accesso diretto ai dati.

Grazie all’uso della struttura In Place Element Structure (IPE), è possibile operare sul dato originale in memoria (senza duplicarlo) garantendo al contempo la sincronizzazione necessaria per evitare conflitti in ambienti

asincroni.



Come funziona?

Vediamo come si può utilizzare, creando una DVR come reference di accesso alle funzioni del cronometro che saranno tutte by-data eseguite in una In Place Element.

Creiamo quindi una DVR cronometro, con la sua API per Creazione, Reset, Read e Distruzione.

Qualcuno si ricorderà che questo è sempre stato fatto con FGV o meglio Action Engine.

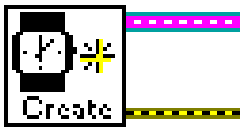
ActionEngine è un pattern SingleTone l'accesso concorrente è protetto dalla sua natura di vi non rientrante e dal fatto (AE) che le funzioni sono eseguite internamente, FANTASTICO, ma allora perché farlo con DVR?

Una risposta divertente potrebbe essere 42, ma torniamo seri e pensiamo a quanta memoria su disco occupiamo ogni copia inserita nel codice e pensiamo che la sua natura singletone ovvero non essendo rientrante una volta istanziato è sempre riferito a se stesso, se non facendo una copia fisica del crono non è possibile usarne due istanze.

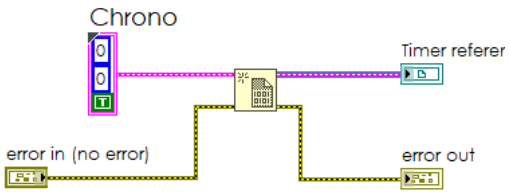
I veterani potranno dire si può usare una memoria integrata tipo lookup table, ma in ogni caso mi sto portando per ogni funzione una copia intera di tutte le funzioni interne.

Con la DVR si può creare invece una DVR reference,e le azioni solo dove e quelle necessarie. Ottimizzazione di memoria DISCO e RAM.

CreateChrono



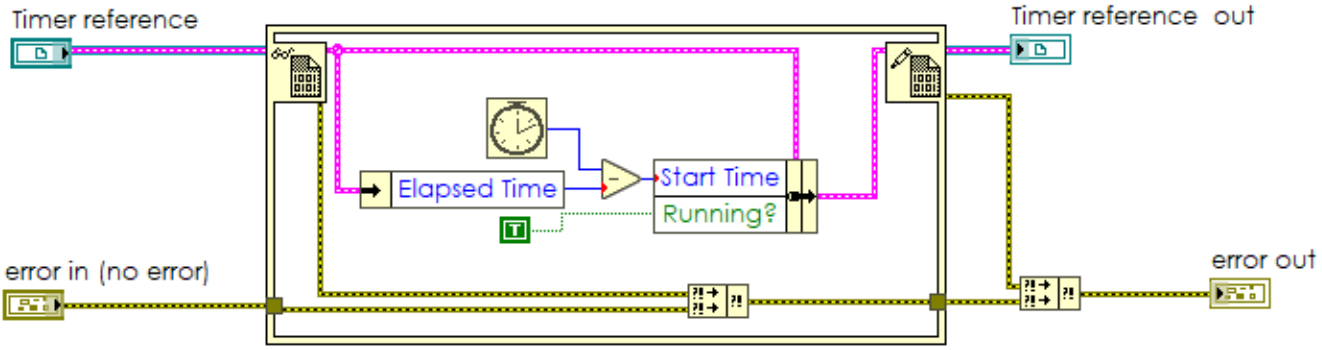
New DVR crea la reference.



Funzione di start

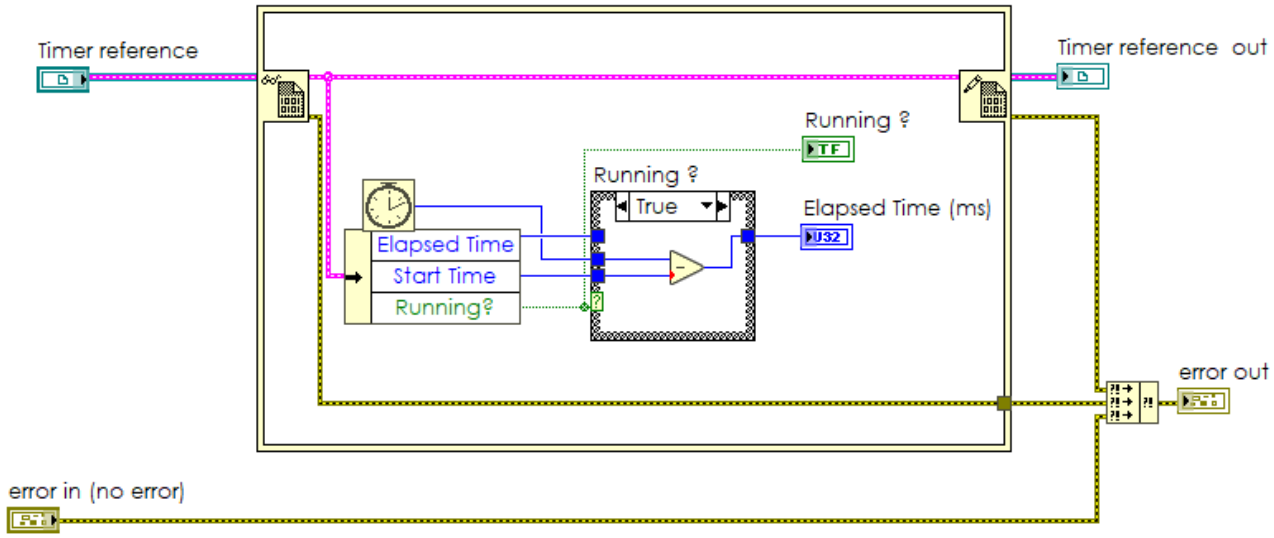


La struttura in place elemet con **D**ata**V**alue**R**eference Read (a destra) and Write (a sinistra), legge il dato lo utilizza nel codice e lo riscrive sulla memoria a cui punta la reference.



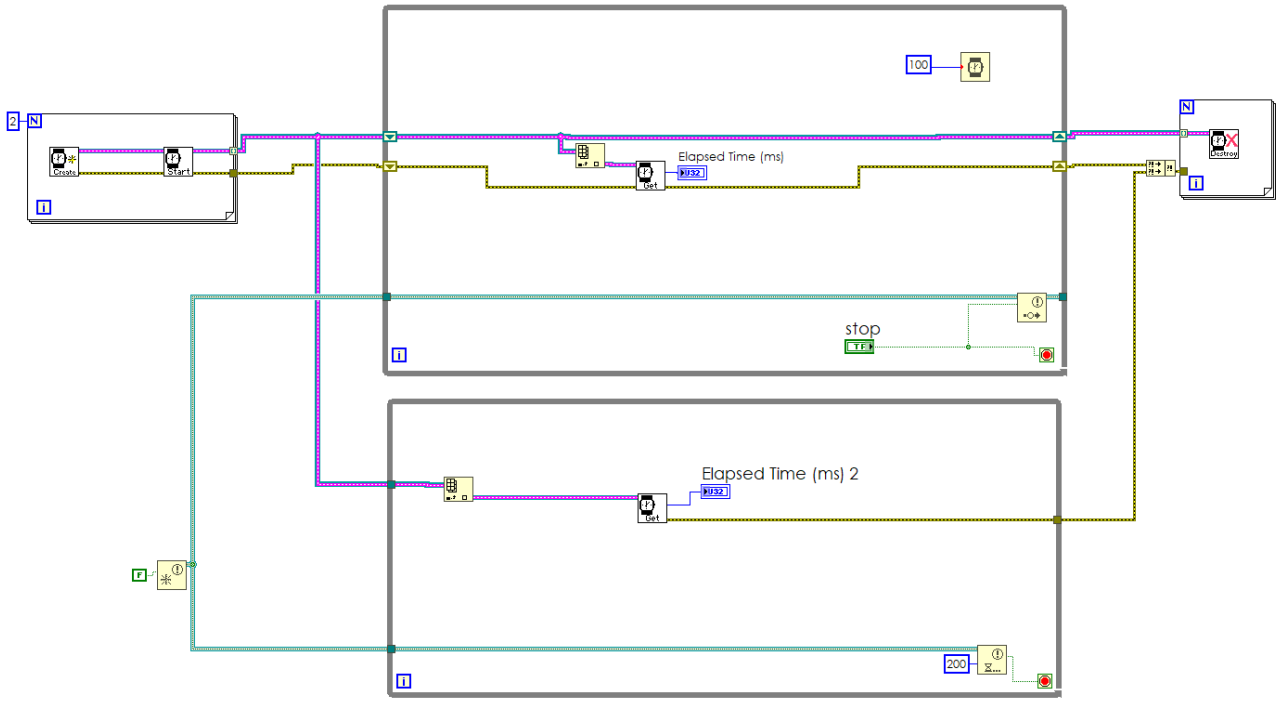
La funzione GetTime

Anche in questo caso riutilizzamo la struttura in place element per riprendere il dato dalla memoria e calcolare il tempo trascorso per differenza dallo start time.



Ecco il test dell'API creata.

Sono create **due sessioni** di cronometro indipendenti.



Rispetto alle FGV/AE hanno diversi vantaggi:

Migliore incapsulamento e protezione dei dati

API con funzioni istanziate singolarmente migliore scalabilità

Funzioni di create e destroy.

Allocazione ByRef

Memoria e OOP by ref

Con l’introduzione delle classi e della programmazione ad oggetti, che in LabVIEW rimane byData, ritorna maggiormente l’esigenza di avere uno switch byValue, byRef e contrario.

La DVR aiuta a condividere dati e funzionalità tra i dati.

Quando si usa con le classi si può poi evitare o abilitare la condivisione con le classi discendenti, configurabile nelle proprietà di classe.

