APPLICAZIONE

DI RIFERIMENTO PER UN DATALOGGER A BORDO VEICOLO BASATO SU

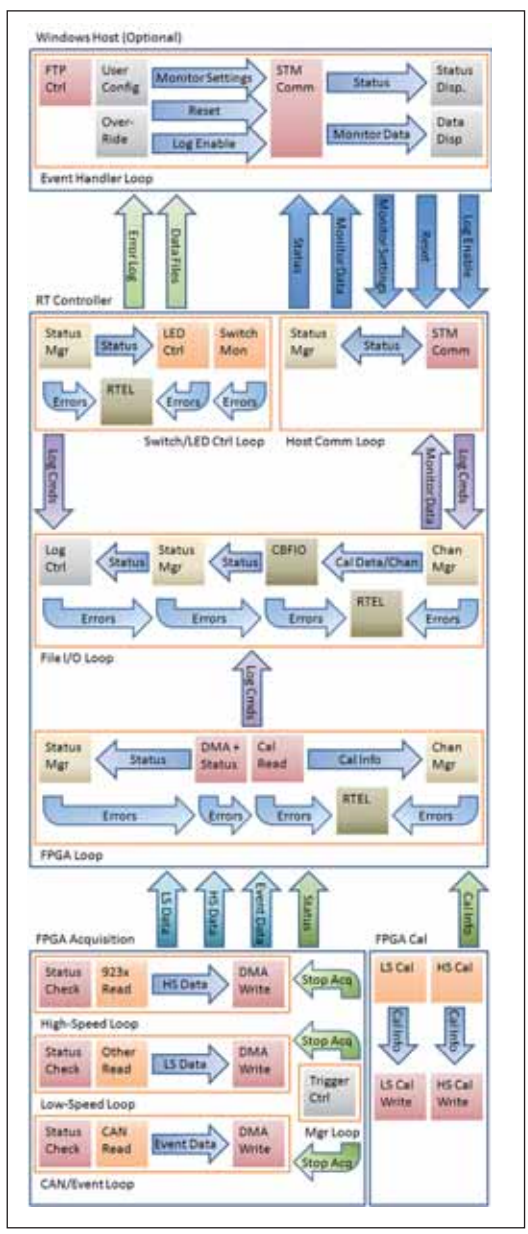
COMPACTRIO

Ryan King

Questa applicazione presenta una soluzione software per un datalogger embedded stand-alone basato su hardware CompactRIO. Le caratteristiche di questa applicazione sono adatte per applicazioni di logging a bordo veicolo, benché adattabili anche per altre tipologie di logging. Si tenga presente che l'applicazione qui mostrata rappresenta un esempio configurabile, non un'applicazione pronta all'uso; in molti casi occorrerà quindi apportare qualche modifica al codice in funzione dell'hardware specifico impiegato.

Le caratteristiche dell'applicazione di riferimento descritta nell'articolo includono:

- Supporto per velocità di scansione multiple
- Supporto per il logging di canali CAN con un appropriato database CAN
- Supporto per file TDMS
- Indicatori di logging sul controllore e loro override attraverso LED e switch
- Modalità monitor PC host e override del logging



Architettura

Lo schema a sinistra fornisce una visione dettagliata delle operazioni eseguite dai VI del datalogger e del flusso di dati attraverso il datalogger.

Meccanismi di comunicazione

I paragrafi seguenti descrivono i meccanismi di comunicazione utilizzati dal datalogger.

Flusso di dati

Il flusso di dati è il normale trasferimento di dati fra oggetti dello schema a blocchi mediante fili. Il flusso di dati è generalmente utilizzato per trasferire dati fra le funzioni all'interno di un loop.

Coda

Le code trasferiscono dati bufferizzati fra loop.

Notate che viene spesso incoraggiato l'uso delle FIFO Real-Time rispetto alle code per codice Real-Time. Ciò perché le code allocano memoria in fase di run-time, un'operazione non deterministica. Tuttavia, per mantenere il determinismo, le FIFO Real-Time impongono una serie di requisiti che limitano i tipi e le dimensioni degli elementi. Per rimanere flessibile, il datalogger in-vehicle deve essere in grado

di accettare pacchetti di dati da registrare su disco da una varietà di sorgenti e a diverse velocità. Ciò richiede un meccanismo flessibile di trasferimento fra i loop. La parte Fpga esegue tutte le operazioni deterministiche e non ha alcuna dipendenza temporale dal codice Real-Time. Pertanto, il datalogger in-vehicle implementa intenzionalmente la comunicazione fra i loop utilizzando le code anziché le Fifo.

DMA

Per trasferire i dati bufferizzati fra il sistema operativo Real-Time e l'Fpga su un CompactRIO viene utilizzato il Direct Memory Access (DMA). I trasferimenti dei dati in DMA permettono al sistema operativo Real-Time di leggere tutti i dati e, nello stesso tempo, all'Fpga di effettuare un'esecuzione deterministica senza alcuna dipendenza temporale dal codice Real-Time.
Il datalogger in-vehicle utilizza tre canali DMA da Fpga a RT: uno per trasferire dati ad alta velocità, uno per trasferire dati a bassa velocità ed uno per trasferire messaggi CAN ed altri dati basati su eventi.

I/O di controllo Fpga

Controlli e indicatori su un VI Fpga possono essere letti e scritti sia dall'interno del codice Fpga, utilizzando variabili locali, sia da un VI Real-Time, usando la funzione Read/Write Control.
Il datalogger in-vehicle utilizza gli I/O di controllo dell'Fpga per la scansione e l'impostazione dei dati che non richiedono buffer, come le informazioni di stato e configurazione.

STM

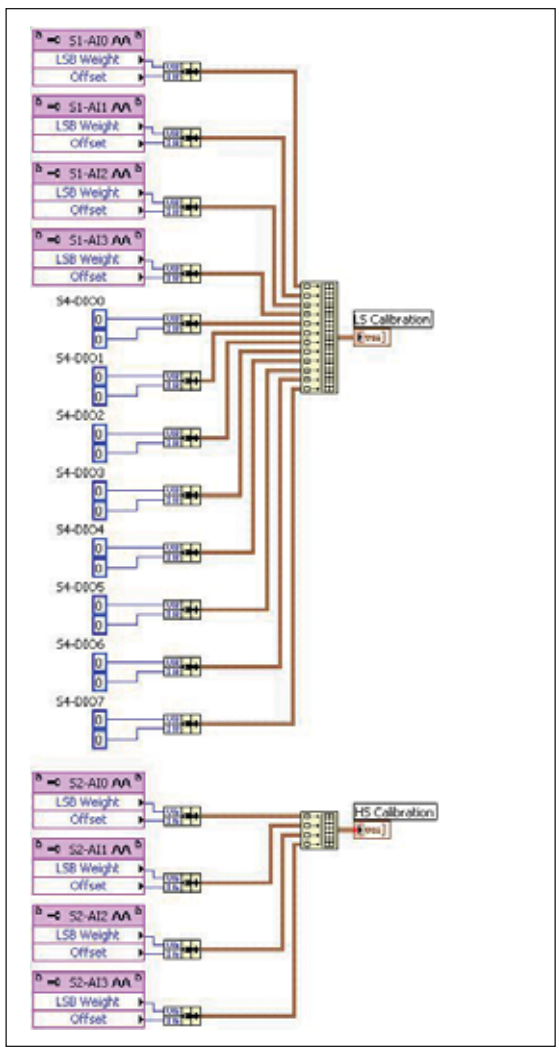
Il componente Simple TCP Messaging (STM) semplifica la comunicazione in rete stabilendo un protocollo basato su messaggi per il trasferimento dei dati.
Il datalogger in-vehicle utilizza messaggi STM per tutte le comunicazioni dirette fra il CompactRIO e l'host Windows.

FTP

Il File Transfer Protocol (FTP) offre un protocollo sicuro per il trasferimento dei file in rete.
L'host Windows utilizzato per il datalogger in-vehicle fa uso di VI FTP del LabVIEW Internet Toolkit per recuperare programmaticamente file di dati ed errori dal target. Se non avete l'Internet Toolkit, potete rimpiazzare questi VI con un'implementazione FTP alternativa, oppure disabilitare questa funzione dell'host e trasferire manualmente i file utilizzando un client FTP.

Dettagli del VI Fpga Calibration

Il VI Fpga Calibration recupera le costanti di calibrazione memorizzate su moduli hardware Serie C. Il VI è separato dal codice principale di acquisizione Fpga perché i valori di calibrazione devono essere recuperati una sola volta per acquisizione. Isolando il codice di calibrazione in un VI separato si utilizzano meno gate per il VI principale.



Il VI legge le costanti di calibrazione da ciascun modulo di acquisizione. Dovete modificare queste parti di codice per leggere le costanti di calibrazione dai moduli che configurate nel vostro sistema.

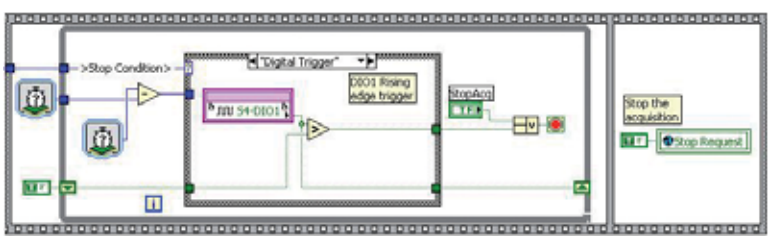
I dati di calibrazione sono riuniti nei cluster di weight/offset e quindi incorporati in array per il trasferimento. Gli array devono avere un punto d'ingresso per ogni canale. I canali che non richiedono calibrazione, come i canali digitali, sono rappresentati da un cluster di due valori zero. Gli indicatori del pannello frontale trasferiscono i dati utilizzando gli I/O di controllo dell'Fpga. Notate che non per tutti i dati basati su eventi viene fornito il codice di calibrazione. Se i vostri dati basati su eventi richiedono costanti di calibrazione, dovete implementare un indicatore separato per trasferire i dati di calibrazione.

Dettagli del VI Fpga Acquisition

Il VI Fpga Acquisition attende una condizione di trigger, acquisisce dati dai moduli Serie C finché non viene rilevato un trigger o una condizione di stop e trasferisce i dati al controllore Real-Time.

System Manager Loop

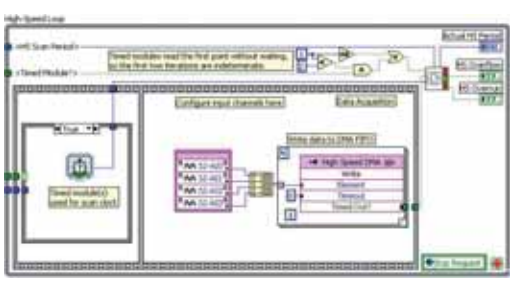
Il System Manager Loop verifica la presenza di trigger di stop o di un comando di stop dal controllore Real-Time e trasferisce i messaggi di stop agli altri loop nel sistema.

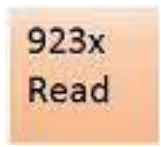


Il datalogger vi permette di configurare i trigger di start e stop. Dovete rimpiazzare i nodi di I/O dell'Fpga che rilevano i trigger esterni con nodi collegati ai vostri specifici canali di trigger. Se il vostro sistema non richiede trigger esterni, potete sostituire i nodi con valori costanti ed evitare l'uso di trigger di tipo digitale o analogico.

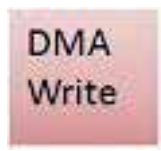
High-Speed Loop

Lo High-Speed Loop gestisce l'acquisizione da moduli di acquisizione temporizzati, come la serie 923x. In alternativa, questo loop può scansionare altri moduli ad una velocità di campionamento diversa da quella del Low-Speed Loop.





Il loop legge un singolo campione di dati da ogni canale ad alta velocità. Sostituite il nodo di I/O con uno o più nodi di I/O configurati per i vostri canali ad alta velocità. Potete leggere dati da moduli temporizzati, moduli normali o entrambi i tipi di moduli dall'interno di questo loop. Se sono presenti moduli temporizzati, la frequenza di acquisizione è controllata dal controllo HS Scan Rate e il controllo HS Scan Period è ignorato. Nel caso contrario, HS Scan Period determina la frequenza e HS Scan Rate è ignorato.



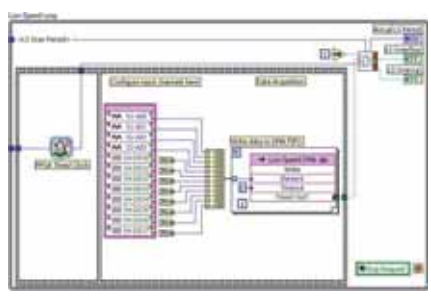
Tutti i dati vengono trasferiti in formato grezzo attraverso un buffer DMA. Potreste avere la necessità di modificare le dimensioni del buffer di memoria nel progetto per diminuire la possibilità di overflow o preservare risorse sull'FPGA. Per default, per il DMA ad alta velocità viene allocato un buffer molto più grande di quello del DMA a bassa velocità, per consentire velocità di scansione più elevate nel loop ad alta velocità.

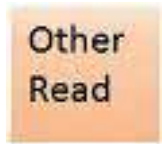
Ogni loop verifica se il loop stesso è stato eseguito nel periodo di tempo configurato (overrun) e se vi era spazio sufficiente nel buffer del DMA per i dati (overflow). Il loop memorizza gli eventuali errori, in modo che le iterazioni successive non cancellino l'indicazione di errore prima che possa essere letta.

Low-Speed Loop

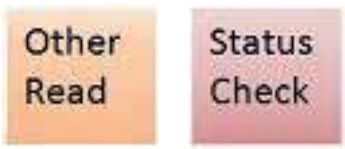
Il Low-Speed Loop gestisce l'acquisizione da altri moduli, ad una velocità di campionamento diversa da quella del loop ad alta velocità.

Il loop legge un singolo campione di dati da ogni canale a





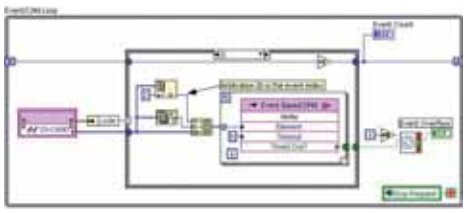
bassa velocità. Sostituite il nodo di I/O con uno o più nodi di I/O configurati per i vostri canali a bassa velocità. Potete leggere dati da qualsiasi modulo che non utilizzi una temporizzazione interna. LS Scan Period determina la frequenza di acquisizione.

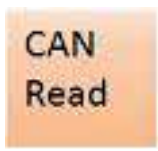


Questo loop utilizza anche la funzionalità DMA Write e Status Check.

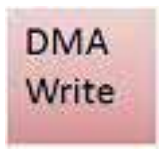
Event Loop
L'Event Loop acquisisce messaggi dai canali CAN e può essere opzionalmente utilizzato per acquisire altri dati basati su eventi.

Il loop legge messaggi da un canale CAN. Collegate il nodo



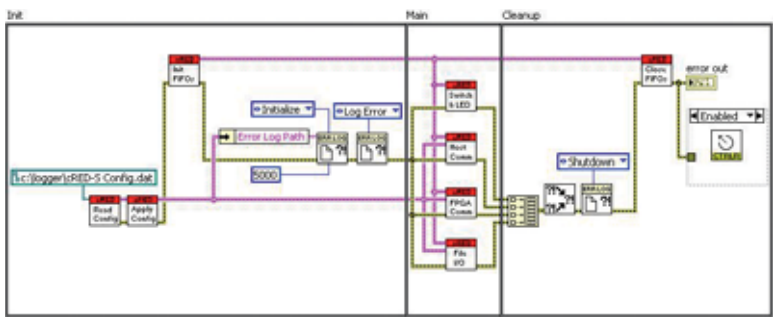


di I/O al canale CAN che state leggendo. Notate che il codice corrente è impostato per leggere solo un singolo canale CAN. Potete utilizzare il caso timeout in questo loop per gestire altri canali basati su eventi. Configurate il timeout del CAN attraverso le proprietà avanzate del modulo nel project explorer. Se sono richiesti più canali CAN, potrebbe essere necessario modificare il codice per gestire singolarmente il timeout da ciascun canale CAN.

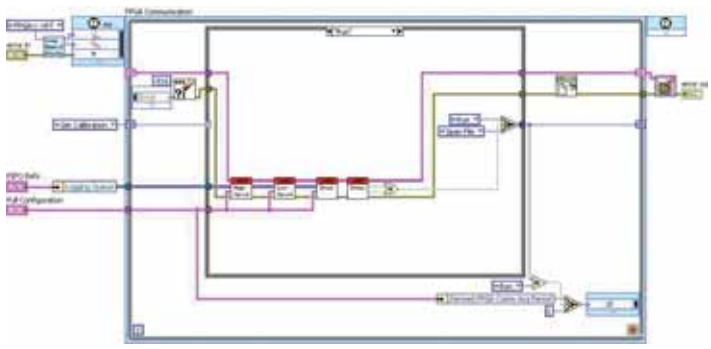


L'Event Loop trasferisce i dati su DMA costruendo pacchetti. Ogni pacchetto contiene un indice eventi, che corrisponde ad un canale eventi configurato, nonché le dimensioni dei dati di evento e i dati stessi. Potrebbe essere necessario modificare le dimensioni del buffer di memoria nel progetto per diminuire la probabilità di overflow o preservare risorse sull'Fpga.
L'Event Loop verifica se nel buffer DMA vi era spazio sufficiente per l'evento (overflow). L'Event Loop non verifica l'overrun perché non c'è un periodo di loop previsto. Il loop memorizza gli eventuali errori, in modo che le iterazioni successive non cancellino l'indicazione di errore prima che possa essere letta.

Dettagli del VI RT Controller
Il Real-Time Controller legge la configurazione del datalogger da un file, acquisisce i dati dall'Fpga, li memorizza su disco, comunica con l'host Windows ed interagisce con l'utente attraverso switch e LED.

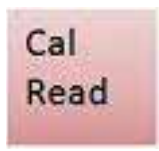


Loop Fpga
Il loop Fpga Comm si interfaccia con l'FPGA e trasmette dati da/verso il resto del sistema.



Nel suo stato Run, il loop Fpga Comm legge i tre canali DMA che trasferiscono dati di misura ed eventi dall'Fpga. Il loop raccoglie i dati letti dai canali DMA in pacchetti e li trasferisce al loop File I/O usando una coda. Il loop Fpga Comm legge inoltre le informazioni di stato dai controlli

Front Panel dell'Fpga. Le informazioni di stato sono trasferite agli altri loop usando lo Status Manager.
Nel suo stato Get Calibration, il loop Fpga Comm legge i dati di calibrazione dal VI Fpga Calibration usando i controlli Front Panel dell'Fpga. Il loop memorizza i dati di calibrazione nel Channel Manager per l'uso da parte del loop File I/O.

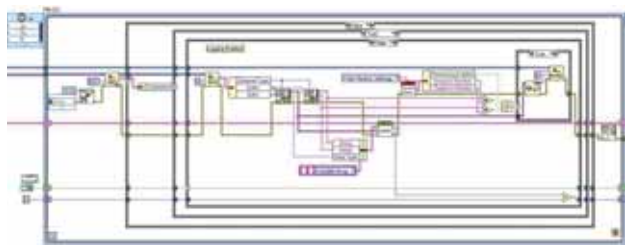


Una variabile globale funzionale Status Manager memorizza le informazioni di stato del sistema. Più loop possono accedere a questa locazione di memoria per leggere o modificare lo stato del sistema.

Una variabile globale funzionale del Channel Manager memorizza un elenco di canali e informazioni su ciascun canale. L'elenco dei canali è inizializzato con dati dal file di configurazione del Real-Time Controller. Le informazioni di calibrazione sono memorizzate dal loop Fpga Comm. Il loop File I/O utilizza le informazioni memorizzate per accedere alle informazioni dei canali e converte i dati grezzi in unità ingegneristiche.

Il componente Real-Time Error Log (Rtel) offre un semplice metodo per la registrazione degli errori che tiene conto di considerazioni sulle prestazioni Real-Time, come l'allocazione di memoria. Benché sia ottimizzato, non è sicuro utilizzare il componente da un loop realmente deterministico, perché esso accede ad un file su disco. L'Rtel memorizza gli errori in un semplice formato Ascii con una dimensione fissa per linea e dimensioni massime del file definibili. Quando le dimensioni massime del file sono state raggiunte, l'Rtel sovrascrive i nuovi errori sugli errori più vecchi. Notate che, per questo motivo, gli errori in un file di log Rtel non sono necessariamente in ordine cronologico. Il VI host include una utility per convertire i log Rtel in ordine cronologico.

Loop File I/O
Il loop File I/O riceve dati grezzi dal loop FpgaComm, categorizza i dati per canale e gruppo, converte i dati in unità ingegneristiche e registra i dati su disco.



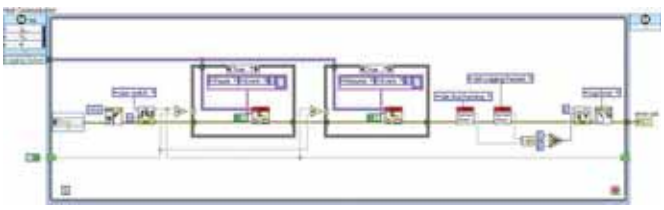
Il loop File I/O può brevemente mettere in pausa il logging per consentire la rimozione sicura dei dispositivi di memorizzazione. Il logging può essere messo in pausa da un host Windows o utilizzando uno switch sul CompactRIO. Notate che quando il logging è in pausa, i dati vengono ancora acquisiti e bufferizzati se il sistema riceve un trigger. Lasciare il logging in pausa per un periodo di tempo esteso porta ad un overflow del buffer. La quantità di tempo per cui si può mettere in pausa con sicurezza un sistema dipende dalle frequenze di acquisizione utilizzate.

Il component Channel Based File I/O (Cbfio) è un'API ad alto livello per scrivere dati basati su canale in un file. Attualmente, l'API memorizza i file in formato Tdms, usando un'implementazione Tdms basata su VI. Il Tdms offre un formato file flessibile che può essere scritto da qualsiasi sistema operativo ed è direttamente accessibile a programmi di analisi, ma non fornisce necessariamente le velocità di streaming più elevate disponibili. Il componente Cbfio è stato progettato tenendo presente la capacità di scrivere più tipi di file e in futuro potrebbero essere supportati altri formati di file.

Questo loop utilizza anche la funzionalità Status Manager, Channel Manager e Rtel.

Loop Switch/LED Control
Il loop Switch/LED Control aggiorna i LED con valori dallo

Status Manager ed invia comandi al loop File I/O tramite una coda quando gli switch sono modificati.



LED Ctrl

Il loop Switch/LED Control aggiorna il LED User1 sul controllore CompactRIO. Il valore del LED User1 visualizza l'abilitazione del logging. Potete usare questo LED per determinare quando è sicuro rimuovere un dispositivo di memorizzazione dopo avere disabilitato il logging. Potete anche alterare il comportamento di questo VI per visualizzare altre informazioni sul LED.

Notate che il sistema aggiorna anche il LED Fpga disponibile su alcuni sistemi CompactRIO, ma gli aggiornamenti sono eseguiti dal codice dell'Fpga. Il LED Fpgavisualizza se è in corso l'acquisizione di dati.

Switch Mon

Il loop Switch/LED Control monitorizza lo stato dello switch User1 sul controllore CompactRIO. Quando lo switch è nella posizione ON, il loop invia un comando al loop File I/O tramite la coda di logging per mandare in pausa il logging. Quando lo switch è nella posizione OFF, il logging è abiulitato. Potete modificare il comportamento di questo VI per cambiare la funzione dello switch User1. Per esempio, potreste fare in modo che lo switch User1 dia il trigger per avviare e fermare l'acquisizione, anziché mettere in pausa e riprendere il logging.

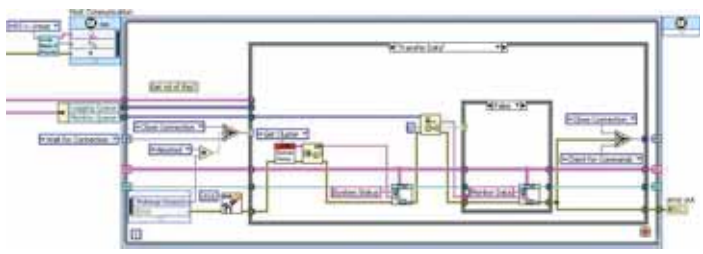
Status Mgr

RTEL

Questo loop utilizza anche la funzionalità Status Manager e Rtel.

Loop Host Communication

Il loop Host Communication accetta connessioni da una macchina host Windows opzionale. Quando esiste una connessione, il loop aggiorna la macchina host con informazioni di stato e permette alla macchina host di escludere il comportamento standard del logger.

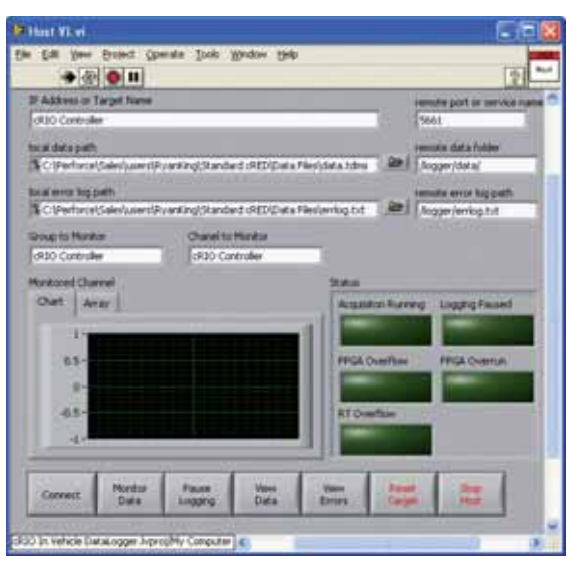


STM Comm

La comunicazione diretta con l'host utilizza il protocollo Simple TCP Messaging (STM). L'host può inviare messaggi per resettare il sistema, mettere in pausa il logging o cambiare le impostazioni del monitor. Il target invia messaggi per trasmettere informazioni di stato e dati monitorizzati all'host.

Status Mgr

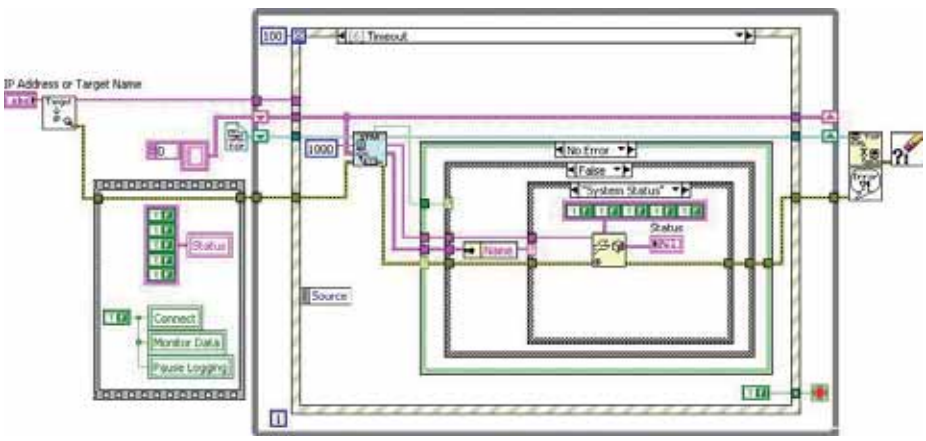
Il loop usa anche la funzionalità Status Manager. Esso non utilizza invece l'Rtel perché sono previsti errori nella comunicazione con l'host, dato che le connessioni possono cadere ed essere ripristinate. Inoltre, l'host è un elemento ausiliario del sistema, pertanto lo spazio di registrazione degli errori è riservato per un comportamento più critico del sistema stesso. Se non utilizzate il datalogger in un sistema dove l'host è un tool essenziale, potete aggiungere l'Rtel al loop Host Communication ed ignorare gli errori specifici associati con le connessioni cadute.



Dettagli del VI Windows Host

Il VI Windows Host offre un'interfaccia utente per monitorare e gestire il datalogger, nonché per visualizzare i dati acquisiti.

Loop Event Handler
Questo loop risponde ad eventi dall'interfaccia utente e legge i comandi STM in un evento di timeout.



FTP Ctrl

L'host usa l'FTP per trasferire file di dati e log di errori dal target.

STM Comm

L'host usa l'STM per inviare messaggi di pausa, reset e impostazioni di monitoraggio. L'host riceve dati di monitoraggio e messaggi di stato del sistema.

L'host permette all'utente di configurare la locazione del target sulla rete, la locazione dei file sul target, le locazioni locali per memorizzare i file trasferiti e, opzionalmente, un canale di acquisizione da monitorizzare.

User Config

Over-Ride

L'host permette di escludere il normale funzionamento del target mettendo in pausa il data logging o resettando il target.

L'host visualizza lo stato del sistema, ricevuto da messaggi STM. Esso include la segnalazione di un'eventuale acquisizione in corso e l'eventuale messa in pausa del logging. Indica inoltre errori comuni che possono verificarsi sul target.

Status Disp.

Un errore Fpga Overflow indica che il buffer DMA non è abbastanza grande per memorizzare i dati trasferiti, o che il loop Fpga Comm non è abbastanza veloce da svuotare il buffer.
Un errore Fpga Overrun indica che il loop di acquisizione non può essere eseguito alla velocità di campionamento desiderata. Notate che ogni nodo di I/O dell'Fpga richiede una minima quantità di cicli di clock in base ai canali che legge.
Un errore RT Overflow indica che non è stato possibile scrivere i dati su disco abbastanza velocemente da svuotare il buffer di memoria fra il loop Fpga Comm ed il loop File I/O. Ciò indica che il throughput totale del sistema è troppo elevato per l'architettura corrente o il formato di file impiegato.

Data Disp

L'host può monitorare dal vivo gli aggiornamenti dei dati su un singolo canale. Ciò è utile soprattutto per scopi di debugging o monitoraggio e potrebbe non essere disponibile se non vi è una sufficiente ampiezza di banda del processore sul target. L'host crea finestre di pop-up per visualizzare il contenuto di file dati e log di errori una volta trasferiti con successo.

Componenti richiesti
L'applicazione richiede la seguente configurazione hardware e software:

Application Software: LabVIEW Professional Development System 8.5.1
Toolkit e Add-On: LabVIEW Real-Time Module 8.5.1, LabVIEW Internet Toolkit 6.0, LabVIEW FPGA Module 8.5.1
Hardware: CompactRIO
Driver: NI-RIO 2.4.1

Inoltre, prima di aprire l'applicazione stessa, occorre installare alcuni componenti che troverete insieme al sorgente dell'applicazione al seguente link: ni.com/italian Infocode: it6ktd

Note sull'autore
Ryan King è un Systems Engineer in National Instruments Corporate. Si dedica alle applicazioni di datalogging imbarcabili e/o embedded e delle relative piattaforme hardware. Ha occupato in precedenza la posizione di Course Development Engineer nel dipartimento formazione di NI.