

**ALMA MATER STUDIORUM – UNIVERSITA' DI BOLOGNA**

**SECONDA FACOLTA' DI INGEGNERIA  
CON SEDE A CESENA**

**CORSO DI LAUREA  
IN INGEGNERIA MECCANICA  
CLASSE LM-33**

**Sede di Forlì**

**TESI DI LAUREA**

**In Controllo dei motori a combustione interna LM**

**REALIZZAZIONE DI UN SISTEMA DI ANALISI COMBUSTIONE  
IN AMBIENTE LABVIEW**

**CANDIDATO**

Michele Rossetti

**RELATORE**

Dott. Ing. Enrico Corti

**Anno Accademico 2011/2012**

**Sessione III**



# INTRODUZIONE

Al giorno d'oggi la ricerca delle massime prestazioni di un motore a combustione interna, nel rispetto di limiti sempre più restringenti sulle emissioni inquinanti, richiede un'ottimizzazione continua del processo di combustione.

Diventa allora necessaria una analisi della reazione chimica che, a partire dal potere calorifico della carica introdotta nel cilindro nella fase di aspirazione, realizza la spinta sul pistone. Utilizzando un trasduttore piezoelettrico, direttamente affacciato alla parete interna della camera di combustione, è possibile studiare l'andamento del segnale di pressione nel cilindro durante un ciclo del motore. Questo segnale possiede un elevato contenuto informativo e da tale studio è possibile ricavare i due parametri che meglio identificano la validità del processo di combustione, ovvero:

- la pressione media indicata, che fornisce un riferimento quantitativo immediato sul grado di sfruttamento termico del motore, cioè del carico cui è sottoposto, indipendentemente dalle sue dimensioni;
- la curva del rilascio di calore, che esprime la modalità con cui viene sviluppata l'energia chimica della combustione.

L'azienda Alma Automotive ha quindi sviluppato un software in grado di svolgere il post-trattamento dei dati provenienti dall'analisi di combustione: tale strumento prende il nome di HeatITOff e

consente il calcolo preciso delle grandezze indicate sulla base dei dati grezzi acquisiti e organizzati dal sistema real-time.

Data la necessità di svolgere una notevole mole di calcoli e non essendoci scadenze temporali stringenti, il software è stato realizzato sul livello Host in modo da poter disporre di un set di operazioni più complesse e di una risoluzione più spinta. Può quindi essere affidato ad un normale personal computer dotato di processori di ultima generazione e sistema operativo Microsoft Windows.

Il linguaggio di programmazione scelto per la scrittura del software è Labview (Laboratory Virtual Instrument Engineering Workbench), un ambiente di sviluppo per applicazioni principalmente orientate all'acquisizione di dati, alla gestione di strumentazione elettronica e all'analisi ed elaborazione dei segnali. Labview è definito un linguaggio di programmazione grafico, dato che non è in generale necessario scrivere linee di codice. Gli algoritmi infatti vengono implementati come diagrammi costituiti da blocchi interconnessi: ogni blocco rappresenta una funzione, mentre le connessioni rappresentano il flusso delle informazioni.

# CAPITOLO 1

## MANUALE UTENTE

In questo capitolo vengono spiegate le funzioni che HeatITOff mette a disposizione, soffermandosi sulla descrizione dell'interfaccia principale e di tutti i menu disponibili nella barra delle applicazioni.

### 1.1–Interfaccia principale

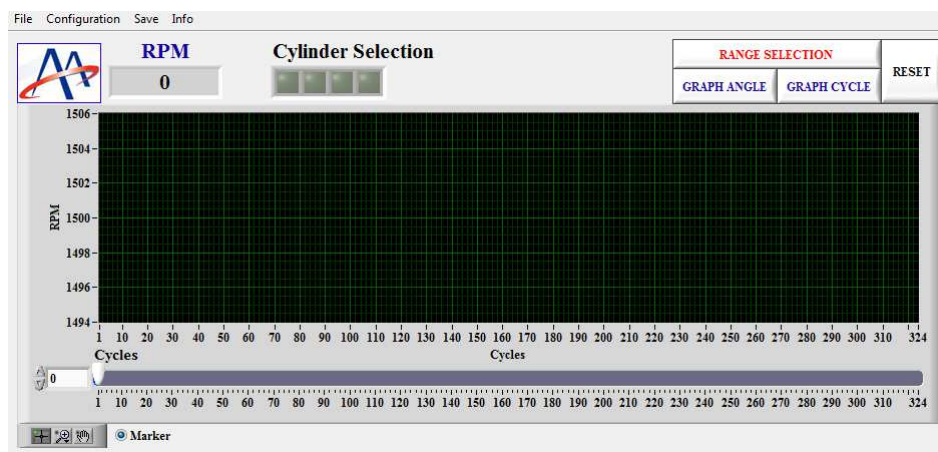
HeatITOff è un software che consente di accedere ai dati grezzi salvati durante il funzionamento del motore al banco prova, “spacchettando” i file e analizzandoli in maniera più approfondita rispetto a quanto viene fatto in real-time. Inoltre, il software permette di osservare i dati su base:

- angolo (pressione all'interno del cilindro, temperatura stimata,...);
- ciclo (PMI, MFB 50,...).

con l'obiettivo, nel prossimo step, di aprire l'applicazione anche a dati esterni (dati ECU), in modo tale da poter osservare nello stesso grafico le grandezze che provengono dalla centralina di controllo del motore e le grandezze relative ai dati indicating. Infine, l'applicazione consente di interpolare i dati campionati dal sistema su base tempo, trasformandoli in dati su base angolo, con la possibilità di selezionare la risoluzione angolare desiderata: le grandezze

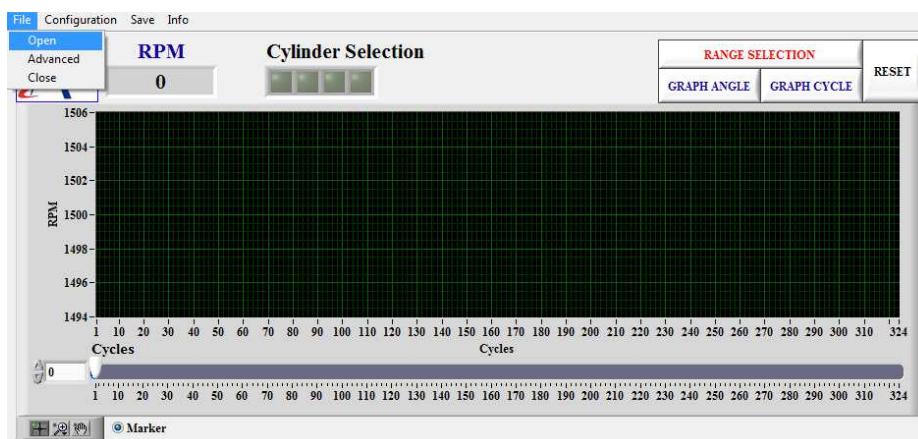
indicating vengono ricalcolate e tutti i parametri necessari al calcolo (posizione PMS, costante di guadagno dell'amplificatore di carica, ritardo del filtro, ritardo del sensore,...) possono essere reimpostati rispetto alla scelta iniziale fatta dall'operatore in real-time, in modo da poter attuare correzioni in caso di errore.

La prima schermata che compare all'attivazione del software presenta un diagramma con i cicli motore in ascissa e gli RPM in ordinata (Figura 1.1.1):



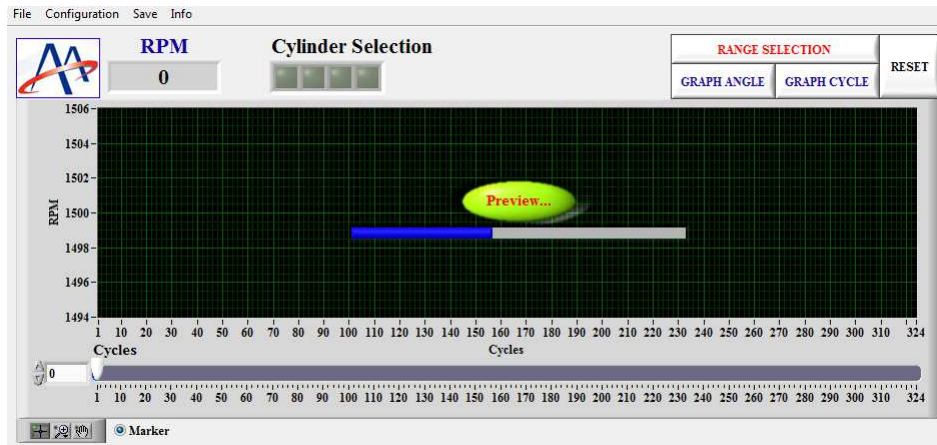
**Figura 1.1.1:** *Interfaccia di apertura HeatITOff*

Nella Figura 1.1.2 viene mostrato il percorso da seguire per caricare un file dati del tipo DataRAW.bin:



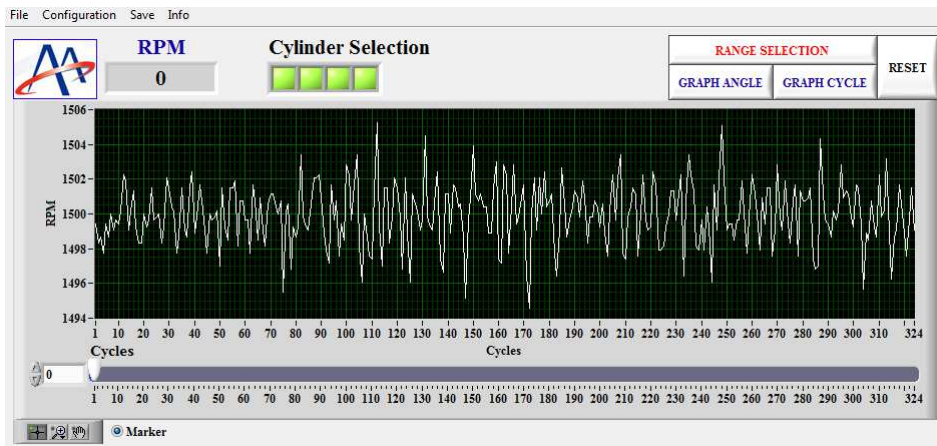
**Figura 1.1.2:** *Percorso File → Open*

Una volta selezionato il file da analizzare, sul grafico compaiono un led verde con la scritta *Preview...* ed una barra di progresso (Figura 1.1.3):



**Figura 1.1.3:** *Anteprima RPM in corso*

Dopo una breve attesa, viene visualizzata la traccia degli RPM (Figura 1.1.4):



**Figura 1.1.4:** *Visualizzazione RPM su base ciclo*

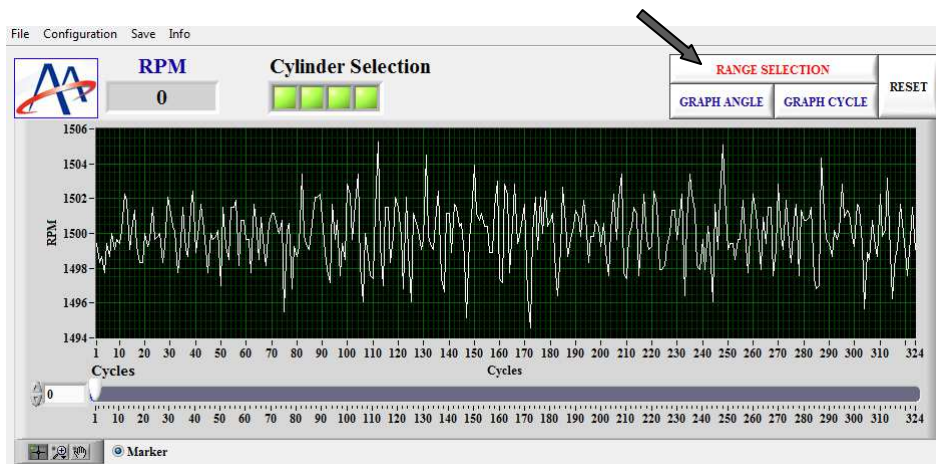
Il file viene aperto senza effettuare tutti i calcoli indicati, in modo tale da velocizzare l'operazione. L'andamento degli RPM può infatti essere determinato semplicemente dall'informazione sulla distanza temporale tra i vari marker di inizio ciclo.

A questo punto, viene data all'utente la possibilità di analizzare le grandezze indicate:

- ciclo per ciclo;
- a intervalli di cicli;
- durante l'intero arco della prova.

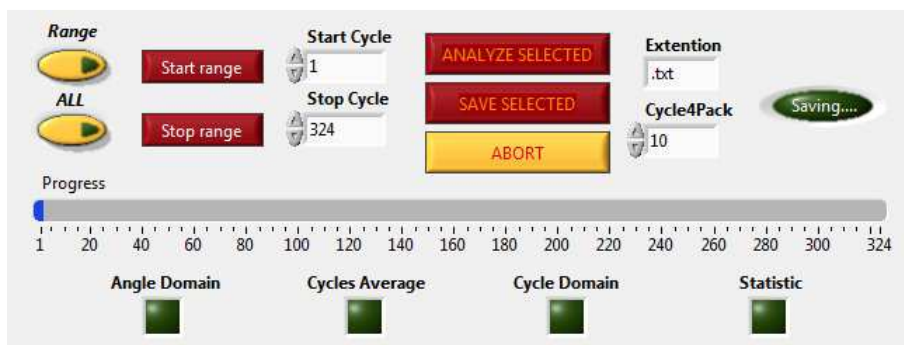
Si consideri che l'idea di visualizzare la sola traccia degli RPM nasce dalle problematiche legate ad acquisizioni di durata notevole (anche qualche minuto). In questo caso, l'utente potrebbe voler evidenziare quello che succede all'interno del cilindro durante una sola parte dell'acquisizione. Per esempio, se l'acquisizione è stata fatta durante una simulazione di un giro di pista (o durante un ciclo di omologazione), l'utente potrebbe desiderare di verificare cosa succede durante alcuni tratti significativi della prova (uscite di curva, funzionamento a freddo,...).

Per poter selezionare un intervallo di cicli da analizzare, si può impiegare lo strumento *Range Selection* (Figure 1.1.5 e 1.1.6):



**Figura 1.1.5:** Pulsante per l'attivazione dello strumento *Range Selection*



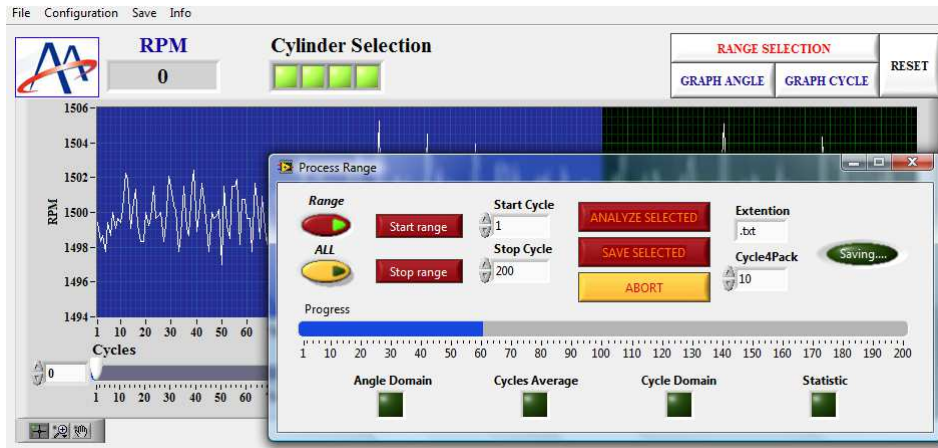


**Figura 1.1.6:** Finestra per la selezione dei cicli da analizzare

La finestra che compare consente di scegliere i cicli in tre modi differenti (i cicli selezionati verranno evidenziati in blu nel diagramma principale):

- si attiva l'opzione *Range* e si clicca con il tasto sinistro del mouse sul primo ciclo di interesse nel grafico degli RPM, trascinando il cursore fino all'ultimo ciclo di interesse;
- si attiva l'opzione *Range* e si clicca sul pulsante *Start range*, digitando il primo ciclo di interesse nel controllo *Start Cycle*; si prosegue, cliccando sul pulsante *Stop range* e digitando l'ultimo ciclo di interesse nel controllo *Stop Cycle*;
- l'opzione *ALL* permette di analizzare l'intero file.

Per iniziare i calcoli indicating basta premere sul pulsante *Analyze Selected*, mentre il pulsante *ABORT* interrompe l'operazione. Nella Figura 1.1.7 viene riportato un esempio di analisi:



**Figura 1.1.7:** Analisi di 200 cicli in corso

La selezione può essere semplicemente analizzata, in modo da visualizzare tutte le informazioni disponibili su base ciclo (IMEP, 50%MFB,...), oppure può essere salvata, premendo il pulsante *Save Selected*. I file possono essere salvati in due formati tramite il controllo *Extention*:

- testo (.txt);
- Matlab (.mat).

Visto l'elevato numero di informazioni, specie se la risoluzione angolare è elevata ( $0.1^\circ$ ), potrebbe nascere l'esigenza di suddividere tali informazioni in diversi file. Utilizzando l'opzione *Cycle4Pack*, è possibile scegliere il numero di cicli da inserire in ogni file generato durante il salvataggio.

I file generati contengono tutte le informazioni selezionate nel menu di configurazione del salvataggio (vedi paragrafo 1.2).

Una volta effettuata l'analisi, nella parte inferiore del diagramma che riporta gli RPM su base ciclo compare un menu a tendina, che consente di scegliere altre grandezze visualizzabili su base ciclo (Figura 1.1.8):

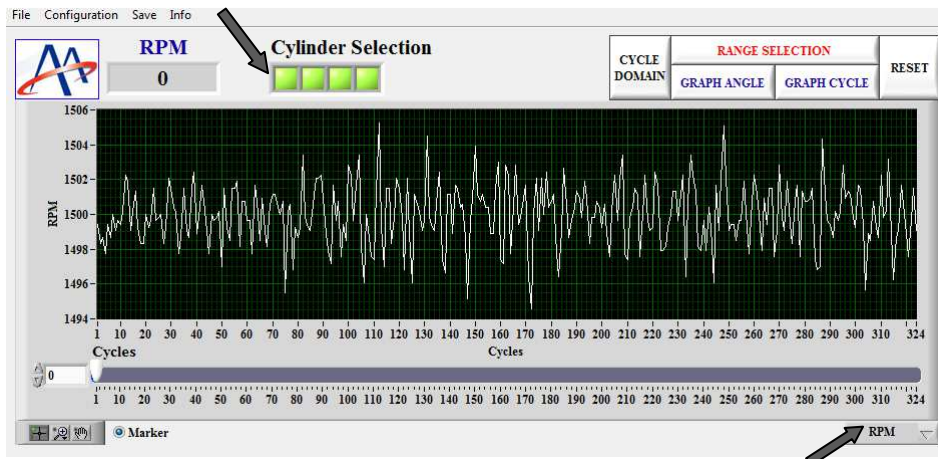


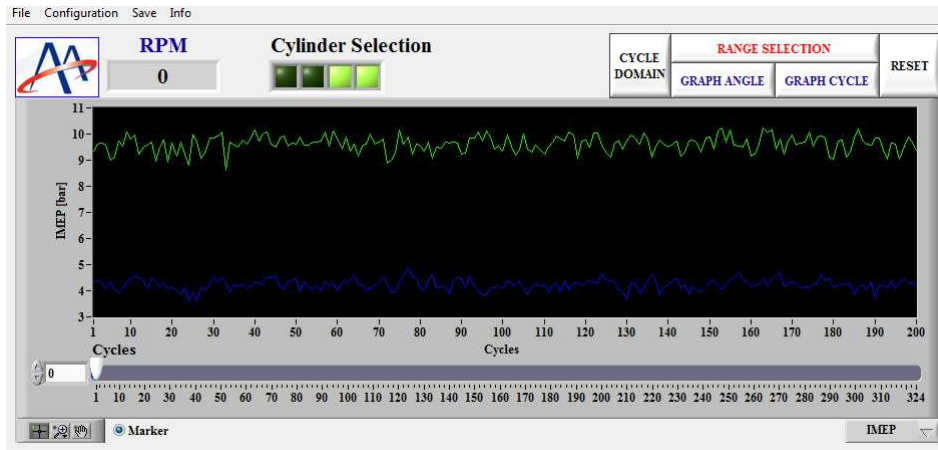
Figura 1.1.8: Comando *Cylinder Selection* e menu a tendina

Le altre grandezze che è possibile rappresentare sono:

- la pressione media indicata (*IMEP*);
- il massimo calore rilasciato (*Q<sub>tot</sub>*);
- l'indice di detonazione Knock Integral (*KINT*);
- l'indice di detonazione Maximum Amplitude of Pressure Oscillation (*MAPO*);
- la frazione di massa bruciata (*MFB*).

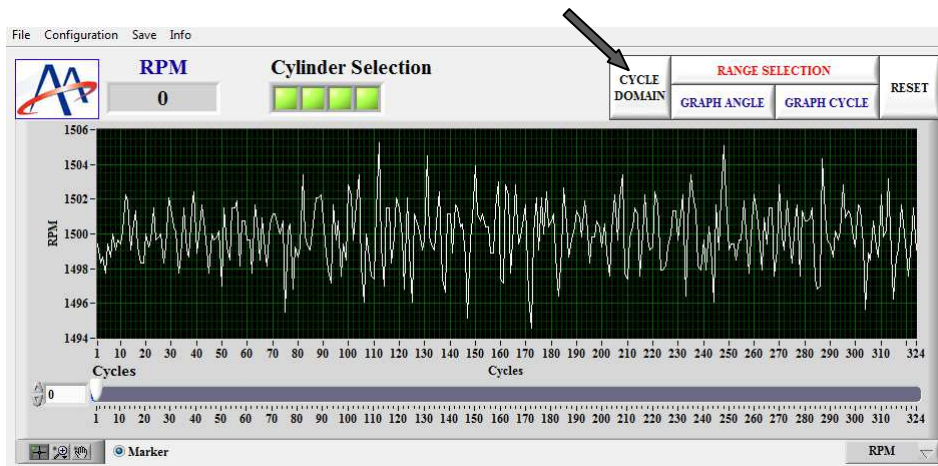
L'ultimo parametro, una volta selezionato, permette di scegliere fra i valori percentuali stabiliti nel menu di configurazione dei parametri (vedi paragrafo 1.4). Di default, i valori impostati sono *MFB 5*, *MFB 50* e *MFB 90*.

Il comando *Cylinder Selection* consente di attivare/disattivare l'analisi o la semplice rappresentazione di tutte le grandezze relative ai vari cilindri (Figura 1.1.9):



**Figura 1.1.9:** Andamento della IMEP per due soli cilindri

Viene poi reso disponibile anche lo strumento *Cycle Domain* (assente prima di avere lanciato l'analisi), che permette di rappresentare in forma di tabella tutte le grandezze calcolate su base ciclo (Figura 1.1.10):



**Figura 1.1.10:** Pulsante per l'attivazione dello strumento *Cycle Domain*

Agendo sugli appositi check, è possibile decidere quali grandezze rappresentare. Il comando *Cycle Selection* consente di evidenziare, con una cornice blu, il ciclo corrispondente alla posizione del marker (una linea verticale di colore giallo) sul diagramma principale in modo tale da avere, per il dato ciclo, direttamente tutte le

informazioni relative alle grandezze calcolate sulla base dei segnali di pressione nel cilindro. I tre valori riportati sotto il comando *Cycle Selection* rappresentano invece l'intervallo in cui i dati sono stati analizzati (primo e terzo valore) e il ciclo attuale, ovvero quello in corrispondenza del quale è stato posizionato il marker sul diagramma principale. Nella Figura 1.1.11 vengono riportati i comandi disponibili nello strumento *Cycle Domain*:

RPM	IMEP1	IMEP2	IMEP3	IMEP4	Qtot1	Qtot2	Qtot3	Qtot4	KINT1	KINT2
RPM	Bar	Bar	Bar	Bar	J	J	J	J	KINT	KINT
1499.438	8.096	0.000	9.345	4.300	1011.753	211.914	1118.269	473.046	0.007	0.000
1498.314	8.394	0.000	9.612	4.382	1016.242	211.914	1163.725	476.514	0.007	0.000
1498.689	7.991	0.000	9.663	4.326	1010.052	211.914	1133.530	476.425	0.007	0.000
1497.753	7.950	0.000	9.586	4.081	1009.958	211.914	1162.438	468.257	0.007	0.000
1499.438	7.346	0.000	9.003	4.329	1000.476	211.914	1152.017	473.496	0.007	0.000
1498.689	8.446	0.000	9.130	4.047	1020.406	211.914	1106.441	464.790	0.007	0.000
1500.000	7.988	0.000	9.742	3.920	1013.491	211.914	1136.062	460.907	0.007	0.000
1499.063	8.200	0.000	9.525	4.084	1009.345	211.914	1131.931	472.468	0.007	0.000
1499.625	8.207	0.000	10.092	4.338	1009.475	211.914	1152.461	475.664	0.007	0.000
1499.438	8.414	0.000	9.807	4.420	1006.933	211.914	1155.529	476.155	0.008	0.000
1500.188	8.214	0.000	9.974	4.545	1007.776	211.914	1162.236	479.702	0.007	0.000
1502.253	7.524	0.000	9.220	4.487	1005.673	211.914	1134.377	485.305	0.007	0.000
1501.877	8.193	0.000	9.504	4.387	998.602	211.914	1145.746	475.447	0.007	0.000
1499.063	8.565	0.000	9.574	4.093	1018.651	211.914	1140.694	462.556	0.007	0.000
1500.188	7.274	0.000	9.701	4.496	1005.727	211.914	1121.974	477.937	0.007	0.000
1501.314	7.735	0.000	8.948	4.336	1014.789	211.914	1097.265	482.355	0.007	0.000
1499.250	8.113	0.000	9.495	4.160	1017.535	211.914	1144.860	471.436	0.007	0.000
1498.314	7.290	0.000	9.776	4.281	1004.963	211.914	1140.431	473.141	0.008	0.000
1498.314	8.230	0.000	8.904	4.071	1025.011	211.914	1116.108	471.846	0.007	0.000
1500.000	7.660	0.000	9.655	4.143	1006.627	211.914	1115.600	463.895	0.007	0.000

Cycle Selection

1 | 10 | 200

ALL

NONE

Statistic

- RPM  Pmax
- IMEP  dPmax
- Qtot  IMEPL
- KINT  IMEPH
- MAPO  TComb
- MFB

**Figura 1.1.11:** *Strumento Cycle Domain*

I pulsanti *ALL* e *NONE* consentono, rispettivamente, di attivare e disattivare tutti i check, mentre il comando *Statistic* aggiunge altre cinque righe nella parte finale della tabella dove vengono riportati i valori massimo, minimo, medio, la deviazione standard e la covarianza di ogni grandezza (Figura 1.1.12):



RPM	IMEP1	IMEP2	IMEP3	IMEP4	Qtot1	Qtot2	Qtot3	Qtot4	KINT1	KINT2
RPM	Bar	Bar	Bar	Bar	J	J	J	J	KINT	KINT
1499.438	8.096	0.000	9.345	4.300	1011.753	211.914	1118.269	473.046	0.007	0.000
1498.314	8.394	0.000	9.612	4.382	1016.242	211.914	1163.725	476.514	0.007	0.000
1498.689	7.991	0.000	9.663	4.326	1010.052	211.914	1133.530	476.425	0.007	0.000
1497.753	7.950	0.000	9.586	4.081	1009.958	211.914	1162.438	468.257	0.007	0.000
1499.438	7.346	0.000	9.003	4.329	1000.476	211.914	1152.017	473.496	0.007	0.000
1498.689	8.446	0.000	9.130	4.047	1020.406	211.914	1106.441	464.790	0.007	0.000
1500.000	7.988	0.000	9.742	3.920	1013.491	211.914	1136.062	460.907	0.007	0.000
1499.063	8.200	0.000	9.525	4.084	1009.345	211.914	1131.931	472.468	0.007	0.000
1499.625	8.207	0.000	10.092	4.338	1009.475	211.914	1152.461	475.664	0.007	0.000
1499.438	8.414	0.000	9.807	4.420	1006.933	211.914	1155.529	476.155	0.008	0.000
1500.188	8.214	0.000	9.974	4.545	1007.776	211.914	1162.236	479.702	0.007	0.000
1502.253	7.524	0.000	9.220	4.487	1005.673	211.914	1134.377	485.305	0.007	0.000
1501.877	8.193	0.000	9.504	4.387	998.602	211.914	1145.746	475.447	0.007	0.000
1499.063	8.565	0.000	9.574	4.093	1018.651	211.914	1140.694	462.556	0.007	0.000
1505.268	9.001	0.000	10.227	4.871	1025.011	211.914	1187.302	490.283	0.008	0.000
1494.582	7.212	0.000	8.618	3.601	987.222	211.914	1095.577	457.145	0.007	0.000
1499.955	8.248	0.000	9.620	4.231	1010.452	211.914	1136.458	472.517	0.007	0.000
1.757	0.370	0.000	0.313	0.211	6.277	0.000	17.083	6.828	0.000	0.000
0.117	4.485	0.000	3.256	4.985	0.621	0.000	1.503	1.445	5.040	NaN

**Cycle Selection**

1 10 200

**ALL**

**NONE**

**Statistic**

RPM  Pmax

IMEP  dPmax

Qtot  IMEPL

KINT  IMEPH

MAPO  TComb

MFB

Max

Min

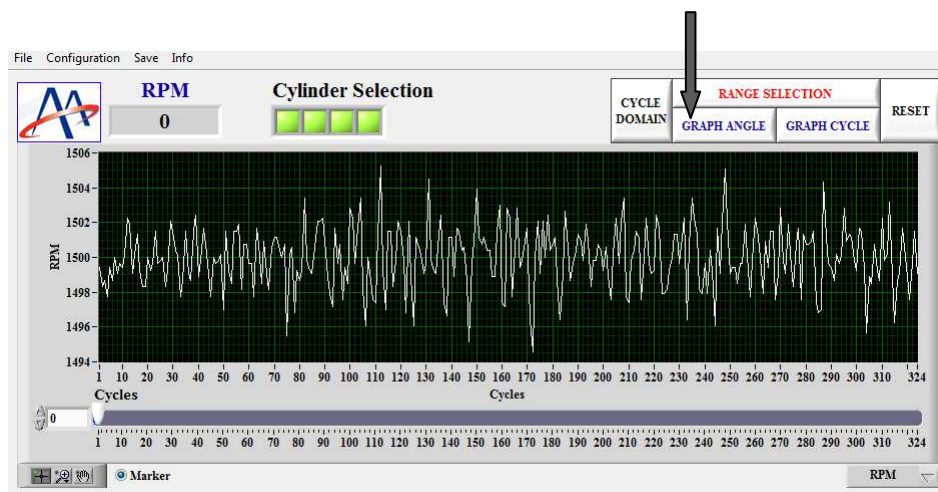
Mean

DevStd

Cov%

**Figura 1.1.12:** Strumento Cycle Domain a cui è stata aggiunta l'opzione Statistic

Se l'obiettivo è quello di vedere il comportamento del motore in corrispondenza di un dato ciclo, basta utilizzare lo strumento *Graph Angle* (Figura 1.1.13):



**Figura 1.1.13:** Pulsante per l'attivazione dello strumento Graph Angle

Questo strumento può essere utilizzato anche senza fare l'analisi ed apre una finestra grafica in cui l'utente può decidere che cosa riportare sull'asse delle ordinate, scegliendo tra diverse grandezze rappresentabili sul dominio angolare:

- pressione all'interno del cilindro (*Pressure*);
- lavoro indicato (*Indicated Work*);
- temperatura stimata a partire dalla pressione nel cilindro (*Temperature*);
- rilascio di calore istantaneo (*ROHR*);
- rilascio di calore cumulato (*Qrel*);
- coppia indicata (*Indicated Torque*);
- oscillazione di velocità (*RPM*).

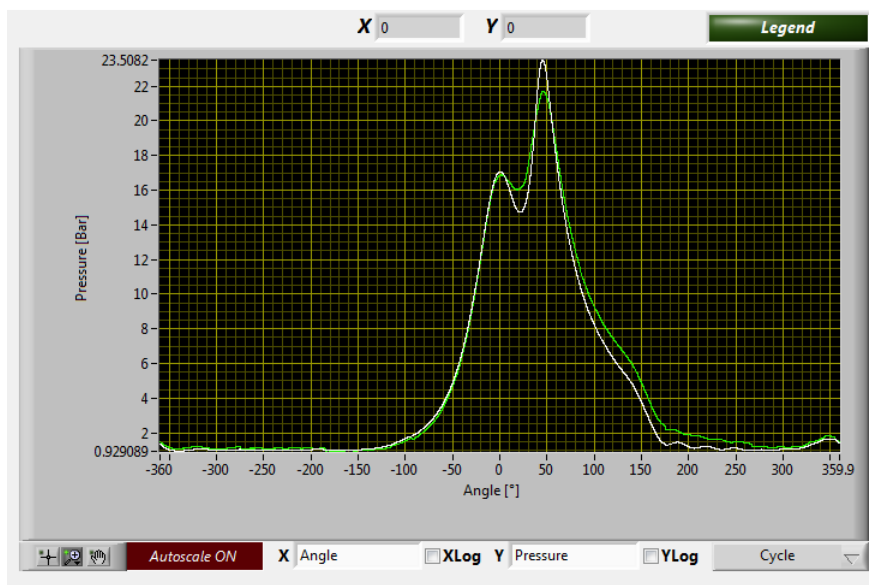
Viene data anche la possibilità di scegliere quale grandezza riportare sull'asse delle ascisse, scegliendo tra:

- angolo (*Angle*);
- volume (*Volume*);
- denti della ruota fonica (*Teeth*).

L'ultima opzione deve essere utilizzata solo se si vuole vedere l'oscillazione sul ciclo della velocità di rotazione.

Per i due assi è poi possibile impostare la scala logaritmica (ad esempio, se si vuole raffigurare il diagramma di indicatore nella sua rappresentazione più classica, ovvero volume-pressione).

Per visualizzare la grandezza desiderata sulla finestra *Graph Angle*, basta posizionare sul diagramma principale il marker relativo al ciclo di interesse. In Figura 1.1.14 è possibile vedere l'andamento che si ottiene per la pressione all'interno di due cilindri, se si clicca con il tasto sinistro del mouse sul ciclo in questione:



**Figura 1.1.14:** *Andamento della pressione nel ciclo*

Spostando il marker con le frecce presenti sulla tastiera del personal computer o posizionandolo in un'altra zona del grafico con un click del mouse, cambierà automaticamente il ciclo rappresentato sulla finestra grafica.

Il comando *Legend* può essere attivato o disattivato e riporta i cilindri per i quali si vogliono rappresentare gli andamenti delle varie grandezze. Anche il comando *Autoscale* può essere attivato (*ON*) o disattivato (*OFF*) a seconda delle esigenze di visualizzazione.

La *Graph Palette* che si trova in basso a sinistra consente di fare zoom a piacimento (pulsante centrale), mentre attivando il cursore (primo pulsante da sinistra) è possibile avere l'informazione sulla posizione *X* e *Y* del cursore stesso all'interno del grafico (indicatori nella parte alta del grafico).

Il menu a tendina in basso a destra compare solo dopo aver fatto l'analisi e permette di scegliere se rappresentare:

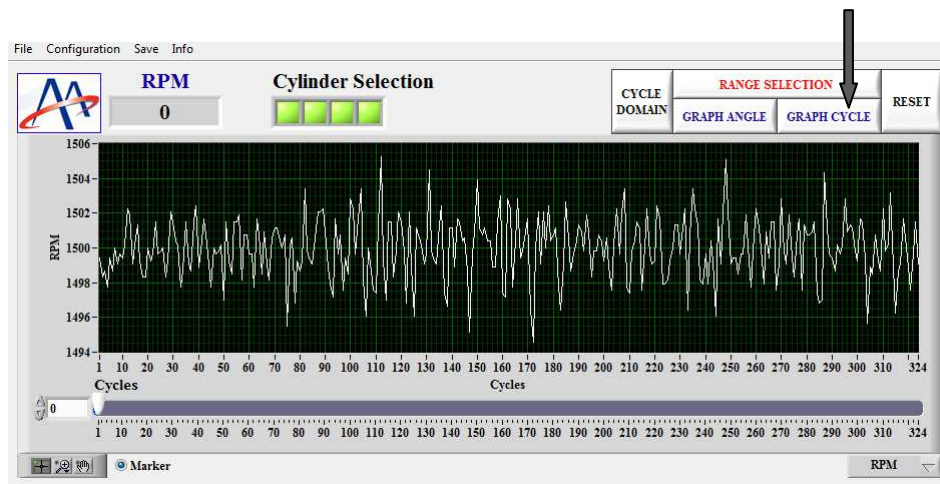
- gli andamenti relativi al ciclo selezionato attraverso il marker sul diagramma principale;



- gli andamenti medi (calcolati sull'intervallo selezionato attraverso lo strumento *Range Selection*);
- sia gli andamenti relativi al ciclo selezionato, sia quelli medi.

Gli andamenti della temperatura stimata, del rilascio di calore istantaneo e del rilascio di calore cumulato, sono rappresentati su un arco angolare limitato, ovvero quello in cui viene effettuato il calcolo del rilascio di calore (vedi paragrafo 1.4).

Per rappresentare le grandezze definite su base ciclo è possibile utilizzare lo strumento *Graph Cycle* (Figura 1.1.15):



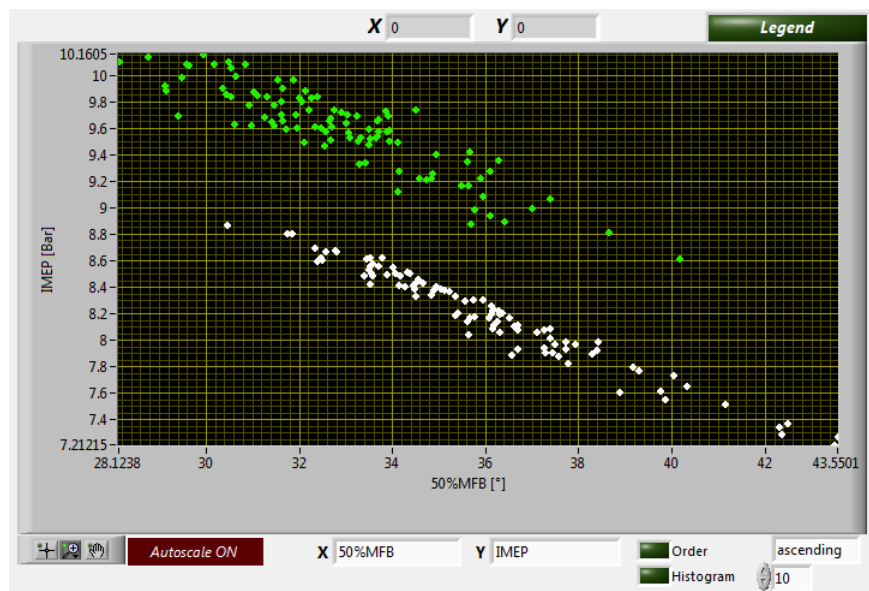
**Figura 1.1.15:** Pulsante per l'attivazione dello strumento *Graph Cycle*

In questo caso, la lista delle grandezze da rappresentare comprenderà tutti quei parametri che sono stati calcolati sulla base del segnale di pressione nel cilindro, ovvero:

- *IMEP*;
- *Qtot*;
- *KINT*;
- *MAPO*;
- pressione massima (*Pmax*);

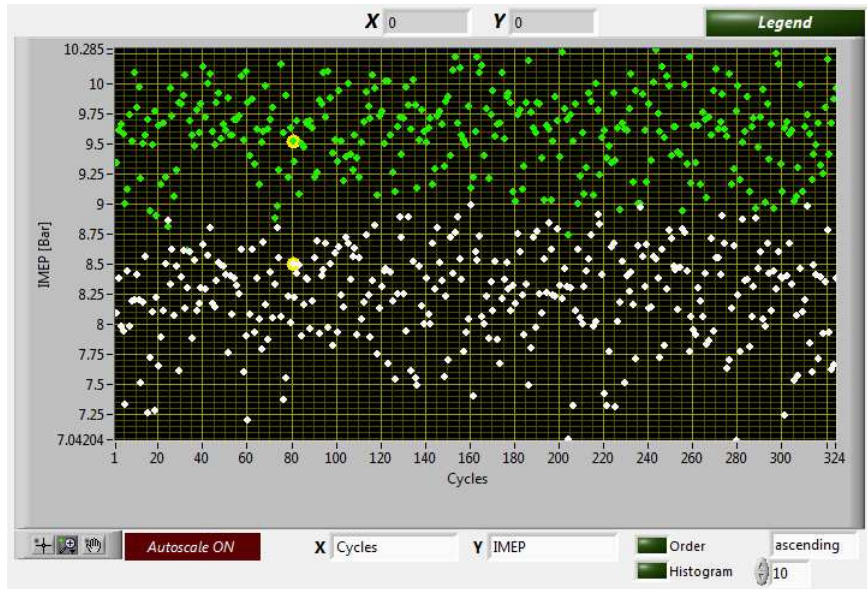
- angolo in corrispondenza del quale si ha la pressione massima ( $AP_{max}$ );
- massimo gradiente di pressione ( $dP_{max}$ );
- angolo in corrispondenza del quale si ha il massimo gradiente di pressione ( $AdP_{max}$ );
- pressione media indicata del ciclo a bassa pressione ( $IMEPL$ );
- pressione media indicata del ciclo ad alta pressione ( $IMEPH$ );
- durata angolare della combustione ( $TComb$ );
- 5% di massa bruciata ( $5\%MFB$ );
- 50% di massa bruciata ( $50\%MFB$ );
- 90% di massa bruciata ( $90\%MFB$ ).

La stessa lista di grandezze è disponibile sia sull'asse delle ascisse sia sull'asse delle ordinate, quindi è possibile costruire diversi diagrammi (x,y) di sintesi della combustione. Nella Figura 1.1.16 viene riportato un esempio di diagramma che è possibile realizzare:



**Figura 1.1.16:** Diagramma 50%MFB-IMEP

Nel caso in cui venga scelta la base *Cycles* per la grandezza che si vuole rappresentare, vengono utilizzati dei marker rotondi di colore giallo per evidenziare il ciclo che si sta analizzando (Figura 1.1.17):



**Figura 1.1.17:** Marker rotondi di colore giallo in corrispondenza del ciclo 80

## 1.2–Menu Save

Nella Figura 1.2.1 viene mostrato il percorso da seguire per aprire un'interfaccia in grado di organizzare il salvataggio delle informazioni, mentre nella Figura 1.2.2 viene riportata la finestra che permette di scegliere le grandezze da salvare:

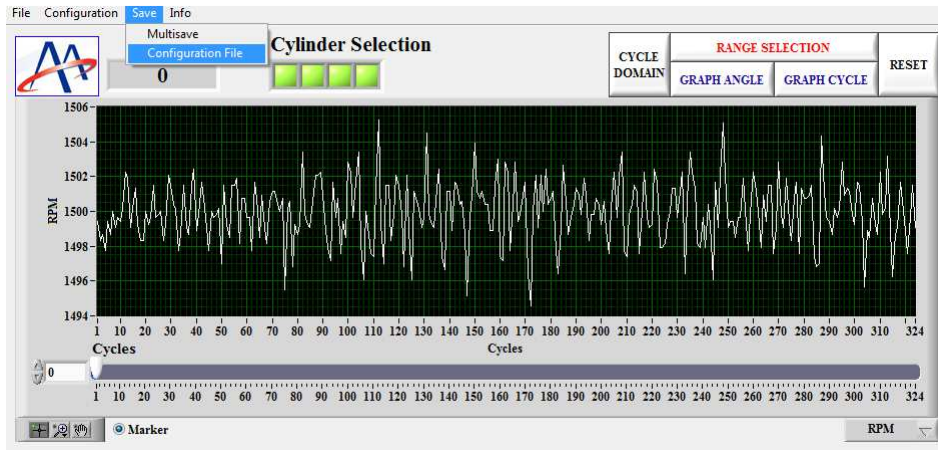


Figura 1.2.1: Percorso Save → Configuration File

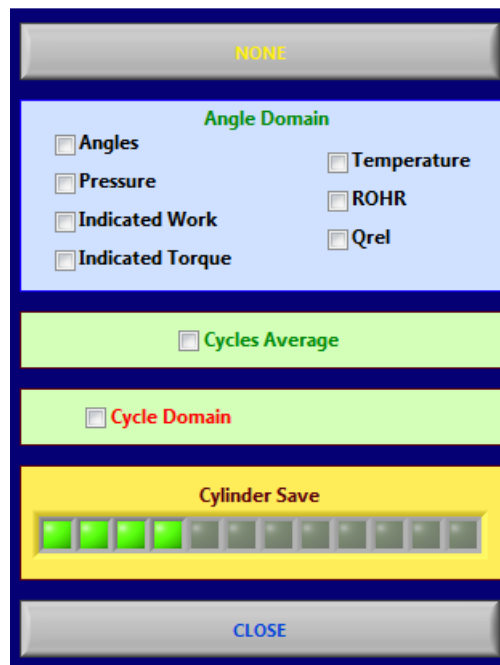


Figura 1.2.2: Interfaccia per la scelta delle grandezze da salvare

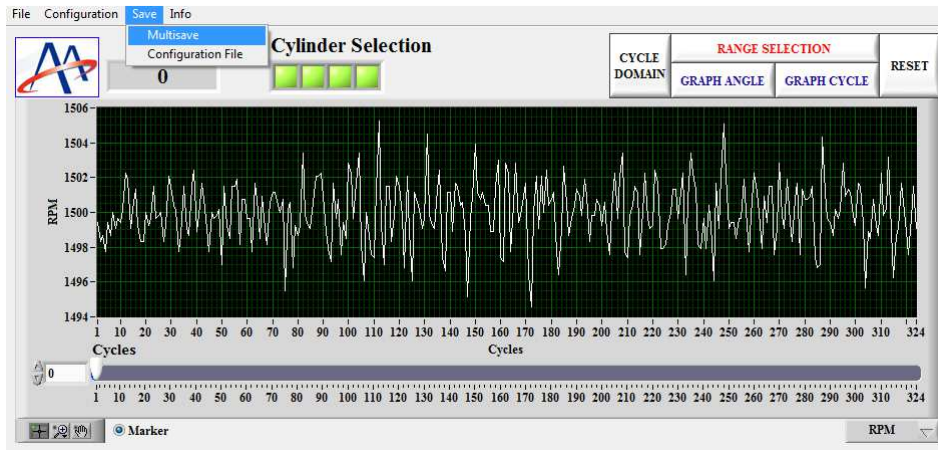
Le impostazioni stabilite in questa finestra sono applicate al salvataggio dei dati contenuti nella porzione di cicli selezionati tramite lo strumento *Range Selection*:

- pannello *Angle Domain*: contiene le grandezze che verranno inserite nel file contenente le informazioni su base angolo;

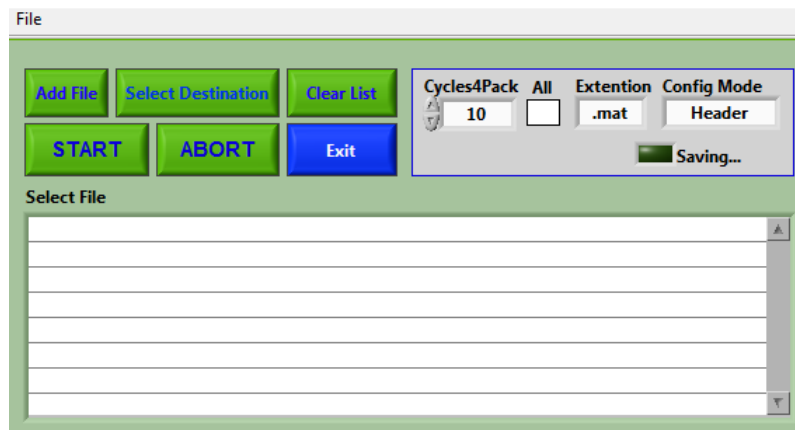
- comando *Cycles Average*: attivando questa opzione, viene data la possibilità di salvare la media su base angolo dell'intera selezione per cui viene svolto il calcolo;
- comando *Cycle Domain*: la sua attivazione consente il salvataggio, ciclo per ciclo, di tutte le grandezze disponibili per ogni cilindro attivo;
- comando *Statistic* (compare solo se prima viene posto il check su *Cycle Domain*): alla fine del file contenente le informazioni su base ciclo vengono aggiunte cinque righe che includono le informazioni statistiche, ovvero il valore massimo, minimo, medio, la deviazione standard e la covarianza di ogni grandezza;
- comando *Cylinder Save*: permette di scegliere il cilindro (o i cilindri) per il quale salvare le grandezze selezionate.

Una volta terminata la selezione delle grandezze da salvare, per uscire dall'applicazione basta cliccare sul pulsante *CLOSE*. Come promemoria, nella parte inferiore del pannello frontale dello strumento *Range Selection* si accenderanno i led relativi a tutte le informazioni selezionate nel menu di configurazione del salvataggio (vedi paragrafo 1.1, Figura 1.1.6).

Nella Figura 1.2.3 viene mostrato il percorso da seguire per aprire uno strumento che consente l'analisi ed il salvataggio automatico di più file in successione. Nella Figura 1.2.4 è invece possibile vedere il pannello frontale di tale strumento:



**Figura 1.2.3:** Percorso *Save* → *Multisave*



**Figura 1.2.4:** *Strumento Multisave*

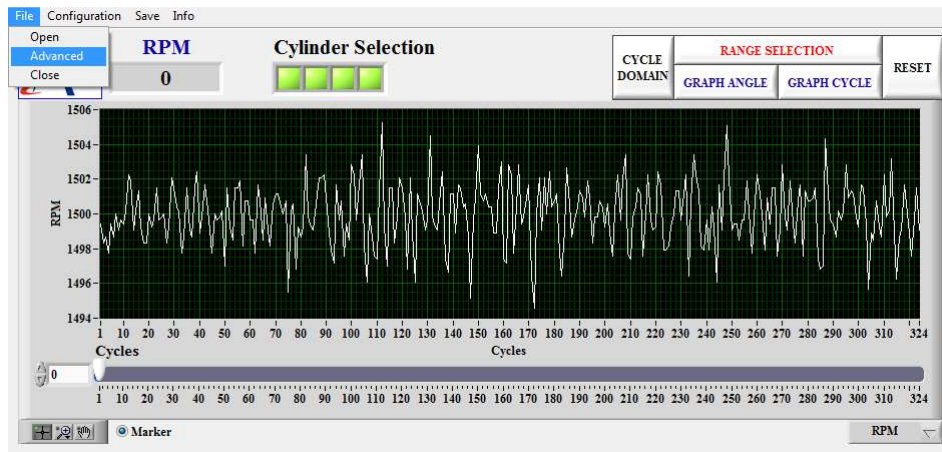
I comandi presenti nella Figura 1.2.4 consentono di:

- selezionare le grandezze da salvare tramite il percorso *File* → *Configuration File*;
- inserire una serie di file del tipo DataRAW.bin (comando *Add File*);
- scegliere la destinazione dei file (comando *Select Destination*);
- impostare l'estensione dei file (.mat o .txt);
- stabilire quanti cicli introdurre in ciascuno dei file (opzione *Cycles4Pack*).

Il comando *ALL*, se attivato, produce un unico file con tutti i cicli relativi ad una data prova.

### 1.3–Menu File

Nella Figura 1.3.1 viene mostrato il percorso da seguire per aprire un'applicazione in grado di monitorare alcune informazioni che interessano l'esecuzione del software:



**Figura 1.3.1:** *Percorso File →Advanced*



**Figura 1.3.2:** *Applicazione Advanced*

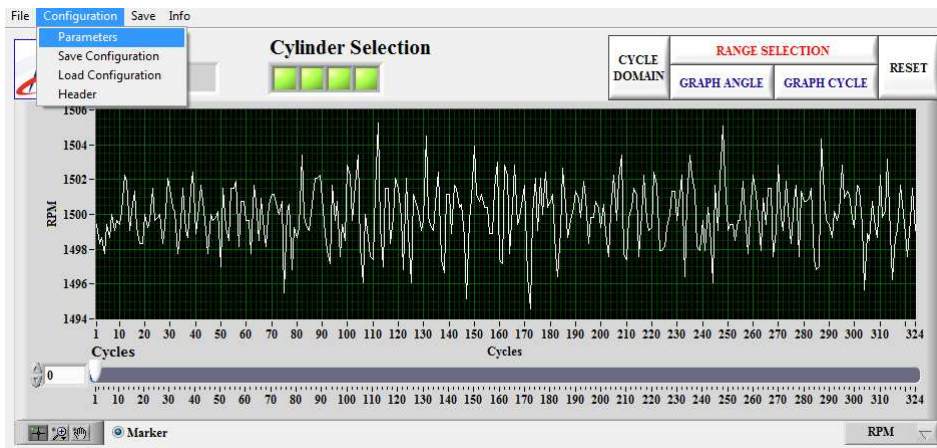
In particolare, i valori riportati nella Figura 1.3.2 sono relativi:

- alla RAM disponibile (*AP\_Memory*);
- al tempo di calcolo complessivo (*Time\_LC*);
- ad eventuali errori che bloccano l'esecuzione delle operazioni (*ERROR*).



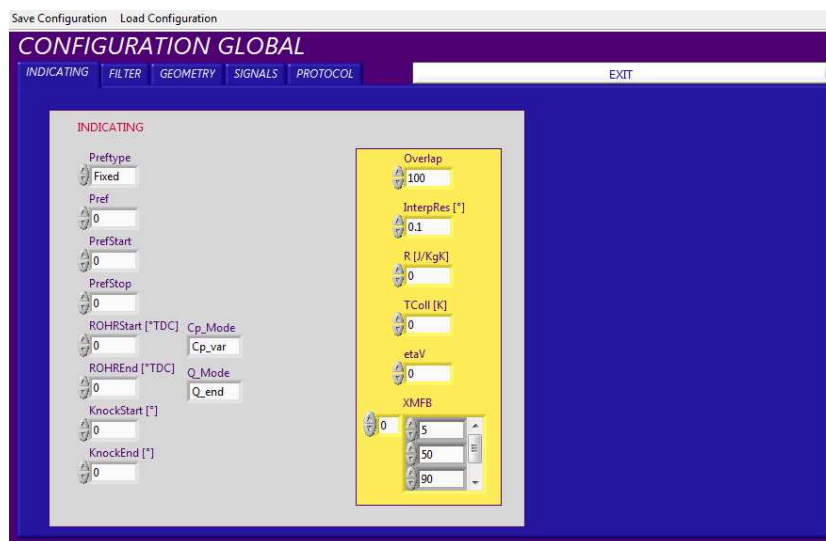
## 1.4–Menu Configuration

Nella Figura 1.4.1 viene mostrato il percorso da seguire per accedere a tutte le impostazioni utilizzate nell'esecuzione dei calcoli e per caricarne o salvarne di diverse rispetto a quelle memorizzate nell'header del file originale:



**Figura 1.4.1:** Percorso Configuration → Parameters

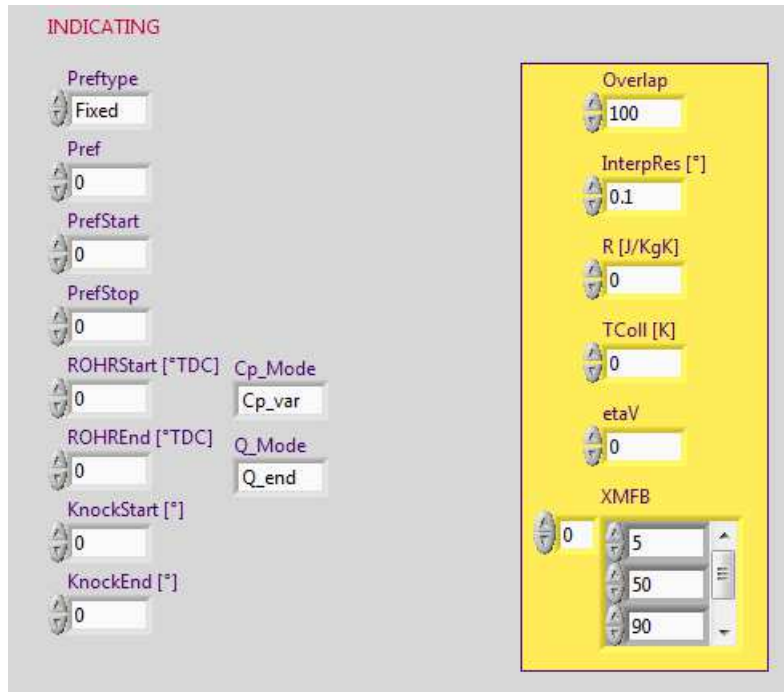
All'apertura del menu compare un Tab con cinque pagine (Figura 1.4.2):



**Figura 1.4.2:** Finestra Configuration Global



La prima pagina che viene presentata è la pagina *Indicating*, riportata in Figura 1.4.3:



**Figura 1.4.3:** Pagina *Indicating*

Da qui è possibile impostare:

- 1) la modalità di recupero della componente media del segnale di pressione (*Preftype*), scegliendo tra:
  - *Fixed*: si fissa il valore che deve assumere la pressione all'interno del cilindro, in corrispondenza di una determinata posizione angolare (ad esempio, 1 bar tra  $-190^\circ$  e  $-170^\circ$ );
  - *MAP*: si utilizza un segnale di riferimento (vedi pagina *Signals*, controllo *Pref\_Ch*), imponendo l'identità del valore medio di pressione nel cilindro e nel collettore di

aspirazione tra le posizioni angolari  $PrefStart$  e  $PrefEnd$  (solitamente  $-190^\circ$  e  $-170^\circ$ );

- $Poly$ : si deve scegliere il coefficiente della trasformazione politropica ( $Polycoeff$ ) in grado di rappresentare al meglio la trasformazione di compressione nella sua fase centrale (cioè sufficientemente lontano dalla fase di chiusura delle valvole di aspirazione e dalla fase di inizio combustione). Tipicamente, i valori da impostare sono:
  - i.  $Polycoeff=1.31 \div 1.32$ ,  $PolyStart=-100^\circ$  e  $PolyEnd=-55^\circ$  per i motori ad accensione comandata;
  - ii.  $Polycoeff=1.27 \div 1.28$ ,  $PolyStart=-100^\circ$  e  $PolyEnd=-55^\circ$  per i motori ad accensione per compressione;

2) gli estremi per il calcolo del rilascio di calore (devono comprendere l'inizio e la fine della combustione, escludendo i disturbi di accensione, di chiusura delle valvole di aspirazione e di apertura delle valvole di scarico), solitamente impostati a:

- $ROHRStart=-40^\circ$ ;
- $ROHREnd=140^\circ$ ;

3) gli estremi per il calcolo degli indici di knock: definiscono la finestra di osservazione del fenomeno della detonazione. Se l'indice utilizzato è il  $MAPO$ , la finestra scelta può essere abbastanza larga (l'indice è dato dal valore massimo dell'ampiezza di oscillazione del segnale di pressione, filtrato

passa-alto o passa-banda); se l'indice utilizzato è il *KINT*, si può scegliere una finestra più centrata sul fenomeno, per migliorare il rapporto tra segnale e rumore. Valori plausibili sono:

- *KnockStart*=0°;
- *KnockEnd*=50°;

- 4) l'*Overlap*, ovvero il numero di campioni in eccesso rispetto all'inizio e alla fine del ciclo, utilizzato per limitare l'effetto di bordo del filtro. L'impostazione a 100 campioni consente buoni risultati con ampie variazioni di impostazione del filtro;
- 5) l'*InterpRes*, ovvero la risoluzione angolare con cui si desidera analizzare il file: i dati grezzi, campionati su base tempo, sono interpolati linearmente su base angolo con risoluzione angolare selezionabile;
- 6) il valore della costante dei gas *R*, che viene usata per stimare la temperatura all'interno del cilindro secondo la legge:

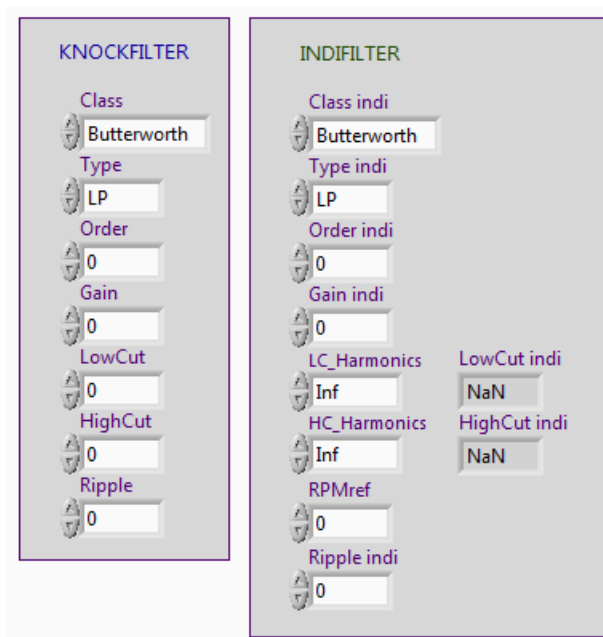
$$P \cdot V = m \cdot R \cdot T$$

- 7) il valore della temperatura nel collettore di aspirazione *TColl*, che viene utilizzata per calcolare la massa d'aria intrappolata all'interno del cilindro secondo la relazione:

$$m = \eta_v \cdot V_C \cdot \frac{P_{coll}}{R \cdot T_{coll}}$$

- dove il valore della pressione all'interno del collettore di aspirazione dipende dal metodo utilizzato per il recupero della componente media del segnale di pressione;
- 8) il rendimento volumetrico  $\eta_{aV}$ , utilizzato nell'espressione precedente;
  - 9) la lista di valori  $XMFB$  riguardanti la frazione di massa bruciata, valutati attraverso il calcolo del rilascio di calore netto. Le percentuali impostate di default sono modificabili e inoltre possono esserne aggiunte anche delle altre (ad esempio: 5-20-50-70-95).

La seconda pagina prende il nome di *Filter* (Figura 1.4.4):



**Figura 1.4.4:** Pagina *Filter*

Nella pagina *Filter* è possibile impostare i filtri digitali da applicare alla curva di pressione prima di effettuare le analisi:

- passa-basso (*IndiFilter*), per la valutazione delle grandezze legate al normale andamento della combustione. Questo filtro è applicato dopo il passaggio dal dominio temporale al dominio angolare, perciò è definito sulla base del numero di armoniche di ciclo che si vogliono preservare (filtraggio nel dominio angolare). In questo caso, l'armonica di taglio corrisponderà ad una certa frequenza per una data velocità di rotazione. È quindi possibile indicare una velocità di rotazione di riferimento, in corrispondenza della quale viene specificata la frequenza corrispondente all'armonica di taglio selezionata;
- passa-alto (*KnockFilter*), per l'analisi dei fenomeni di detonazione o per la valutazione della rumorosità. Questo tipo di filtro è applicato direttamente al segnale nel dominio temporale ed è quindi definito attraverso la frequenza di taglio (in Hz).

I parametri comuni ai due filtri, che possono essere variati, sono:

1) *Class*:

- *Butterworth*;
- *Bessel*;
- *Chebyshev*;
- *Elliptic*.

Si consideri che solo il *Butterworth* garantisce un andamento dell'ampiezza della funzione di risposta in frequenza monotono, senza *ripple*;

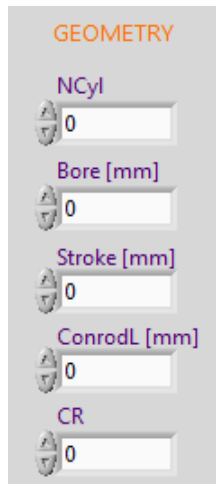
2) *Type*:

- *LP* (passa-basso);
- *HP* (passa-alto);
- *BP* (passa-banda);
- *BS* (elimina-banda).

- 3) *Order*: è l'ordine del filtro, legato al numero di campioni della serie di elementi non filtrati e già filtrati. Viene utilizzato per determinare l'ultimo campione della serie di elementi filtrati. È bene limitare questo valore per evitare problemi numerici tanto più, quanto più è stretta la banda da filtrare (più alto è l'ordine del filtro, più grande è il rapporto tra i coefficienti delle due serie);
- 4) *Gain*: è il guadagno del filtro;
- 5) *LowCut*: frequenza di taglio per i filtri *LP*, *BP* e *BS*. Se non è necessario (filtro *HP*), il parametro viene ignorato;
- 6) *HighCut*: frequenza di taglio per i filtri *HP*, *BP* e *BS*. Se non è necessario (filtro *LP*), il parametro viene ignorato;
- 7) *Ripple*: valore dell'oscillazione di ampiezza tollerata in banda passante (espressa in dB), se il filtro non è di tipo *Butterworth*.

Come accennato in precedenza, per il filtro *IndiFilter* le frequenze di taglio sono definite in termini di armoniche di ciclo. In questo caso allora, la casella *LowCut indi* riporta il valore della frequenza di taglio in Hz, corrispondente all'armonica selezionata per una data velocità di riferimento, impostabile attraverso il controllo *RPMref*.

La terza pagina che viene presentata è la pagina *Geometry* (Figura 1.4.5):

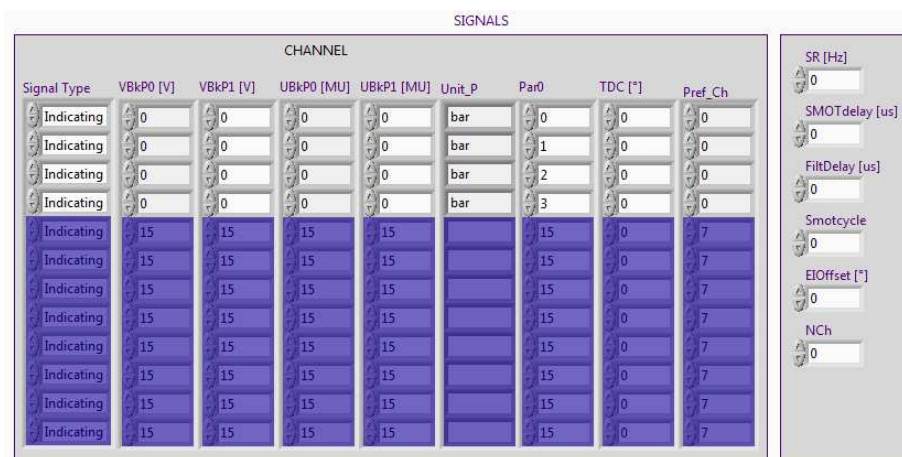


**Figura 1.4.5:** *Pagina Geometry*

Da qui è possibile impostare i principali dati geometrici del motore:

- 1) *NCyl*: numero di cilindri;
- 2) *Bore*: alesaggio;
- 3) *Stroke*: corsa;
- 4) *ConrodL*: lunghezza della biella;
- 5) *CR*: rapporto di compressione.

La penultima pagina prende il nome di *Signals* ed è riportata nella Figura 1.4.6:



**Figura 1.4.6:** *Pagina Signals*

Nella pagina *Signals* è possibile modificare le impostazioni riguardanti i segnali acquisiti:

1) *Signal Type*:

- *Indicating* (pressione all'interno del cilindro);
- *MAP* (pressione nel collettore di aspirazione, utilizzabile per il recupero della componente media);
- *Other* (altri segnali, come ad esempio la pressione nel runner di aspirazione, la corrente di ionizzazione,...).

2) *VBkP0* e *VBkP1*: valori di tensione utilizzati per determinare la caratteristica del sensore;

3) *UBkP0* e *UBkP1*: valori delle unità meccaniche corrispondenti alle tensioni *VBkP0* e *VBkP1*;

4) *Unit\_P*: unità di misura per la pressione all'interno del cilindro;

5) *Par0*: numerazione dei cilindri in ordine crescente, partendo da zero;

6) *TDC*: posizione dei PMS dei vari cilindri;

7) *Pref\_Ch*: canale sul quale è stato acquisito il segnale di pressione collettore, per effettuare il recupero della componente media col metodo del pressure referencing;

8) *SR*: frequenza di campionamento dei segnali di pressione;

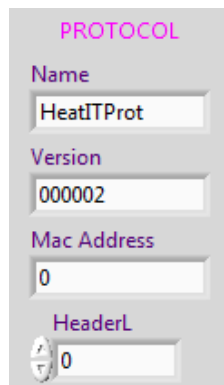
9) *SMOTdelay*: ritardo tra l'istante di raggiungimento di una data posizione angolare e l'istante di transizione del segnale di riferimento angolare (ritardo del sensore di posizione);

10) *FiltDelay*: ritardo introdotto nel segnale dal filtro passa-basso impostato sull'amplificatore di carica;



- 11) *Smotcycle*: numero di riferimento angolari per ciclo (ad esempio, 120 per una ruota fonica 60-2);
- 12) *EIOffset*: distanza angolare tra lo zero elettrico e lo zero meccanico del ciclo motore;
- 13) *NCh*: numero di canali attivi.

La pagina *Protocol* (Figura 1.4.7) è la quinta ed ultima del Tab che compare una volta aperto il menu per la configurazione dei parametri:



The image shows a software interface for configuring protocol parameters. The title 'PROTOCOL' is at the top in pink. Below it are four labeled input fields: 'Name' with the value 'HeatITProt', 'Version' with '000002', 'Mac Address' with '0', and 'HeaderL' with '0'. Each field has a small icon to its left.

**Figura 1.4.7:** Pagina *Protocol*

Nella pagina *Protocol* è possibile visualizzare e, eventualmente, modificare le caratteristiche del protocollo di salvataggio dei dati grezzi. Le informazioni vengono prelevate dall'header del file grezzo acquisito e sono:

- 1) *Name*: nome del protocollo;
- 2) *Version*: versione del protocollo;
- 3) *Mac Address*: codice identificativo della scheda di rete che ha trasmesso i dati;
- 4) *HeaderL*: lunghezza dell'header espressa in righe.

## 1.5–Menu Info

Cliccando su *Info*, compare una finestra (Figura 1.5.1) in cui sono presenti i recapiti dell'azienda produttrice del software:



**Figura 1.5.1:** Schermata menu *Info*

Per chiudere la schermata e tornare all'interfaccia principale, basta posizionare il puntatore del mouse sul logo Alma Automotive e cliccare con il tasto sinistro.

# **CAPITOLO 2**

## **SCHEMA GENERALE DEL CODICE SORGENTE**

In questo capitolo viene riportata una descrizione dell'ambiente Labview e mostrata l'architettura generale del codice che sta alla base di tutte le funzionalità di HeatITOff.

### **2.1–Labview**

Labview è un ambiente di sviluppo integrato per il linguaggio di programmazione visuale della National Instruments. Tale linguaggio è stato denominato G-Language (Graphic Language) perché è diverso dai codici tradizionali: la definizione di strutture ed algoritmi avviene con icone e altri oggetti grafici, ognuno dei quali include funzioni diverse, uniti da linee di collegamento (wire) in modo da formare una sorta di diagramma di flusso. La sequenza di esecuzione delle operazioni è definita e rappresentata dal flusso dei dati stessi (dataflow) attraverso i fili monodirezionali che collegano i diversi blocchi funzionali.

Labview viene utilizzato principalmente per acquisizione e analisi dati, controllo di processi, generazione di rapporti, o più in generale per tutto ciò che riguarda l'automazione industriale.

I programmi realizzati in Labview prendono il nome di Virtual Instruments (VI), cioè strumenti virtuali, dove il termine “strumento” è dovuto al fatto che, durante l'esecuzione, i programmi sviluppati presentano agli utenti un'interfaccia analoga a quella di uno

strumento di misura, mentre il termine “virtuale” si riferisce al fatto che l’interazione avviene con un programma in esecuzione e non con un dispositivo fisico dedicato. L’utente può modificare le diverse grandezze agendo su opportune manopole o interruttori visualizzati dal programma, e può osservare il risultato delle operazioni condotte internamente al VI su display grafici molto simili a quelli che si trovano nella strumentazione numerica tradizionale.

Un VI è composto da due parti fondamentali:

- il pannello frontale (Front Panel);
- il diagramma a blocchi (Block Diagram).

Il pannello frontale è l'interfaccia utente del VI. Si realizza con controlli e indicatori, che costituiscono rispettivamente i terminali interattivi di ingresso e uscita. I controlli (matrici, manopole, potenziometri, pulsanti, quadranti,...) simulano i dispositivi di ingresso degli strumenti e forniscono i dati allo schema a blocchi del VI; gli indicatori (grafici, tabelle, LED, termometri,...) simulano i dispositivi di uscita degli strumenti e visualizzano i dati che lo schema a blocchi acquisisce o genera.

Lo schema a blocchi è il diagramma di flusso che rappresenta il codice sorgente in formato grafico. Gli oggetti del pannello frontale appaiono come terminali di ingresso o uscita e comprendono:

- terminali;
- funzioni;
- costanti;
- strutture;

- chiamate ad altri VI (subVI, ovvero VI che fanno parte di altri programmi più estesi e che consentono di sintetizzare e replicare facilmente un certo numero di operazioni);
- fili di collegamento.

I fili di collegamento possono trasportare teoricamente dati di qualunque tipo e mole, anche aggregati (cluster) definiti dal programmatore. Il colore e lo spessore del filo cambiano per permettere una facile identificazione del dato trasportato. Ad esempio, i numeri interi scorrono su fili di colore blu e le stringhe su fili di colore rosa.

Lo schema a blocchi può essere reso visibile anche durante l'esecuzione del VI, caratteristica molto utile soprattutto in fase di debug in quanto è possibile visualizzare, con un'animazione al rallentatore, il flusso dei dati lungo i fili e il loro valore numerico.

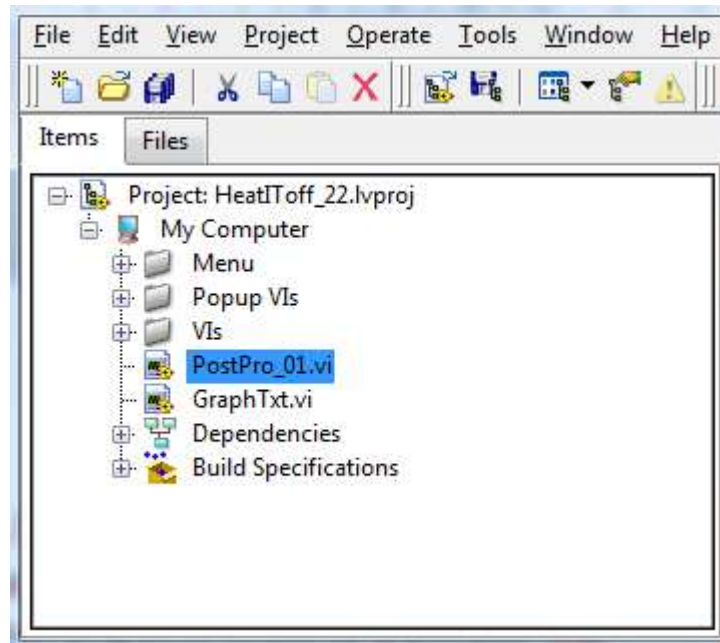
A partire dai VI si possono poi creare eseguibili a sé stanti e librerie condivise (DLL), perché Labview è un vero e proprio compilatore a 32 bit. Per usare tali eseguibili e DLL non occorre un'installazione di Labview sul computer di destinazione, ma è necessario comunque che sia presente almeno il run-time engine.

A livello software, per realizzare applicazioni complesse occorre creare un *Project*, ovvero un progetto che raccoglie in un unico spazio tutti i VI che lo compongono.

## **2.2–Organizzazione del progetto HeatITOff**

Nella cartella che racchiude tutti i VI che consentono il funzionamento di HeatITOff, il file da aprire per accedere al progetto

si chiama *HeatIToff\_22*. Una volta cliccato su tale file, si apre una schermata di questo tipo (Figura 2.2.1):

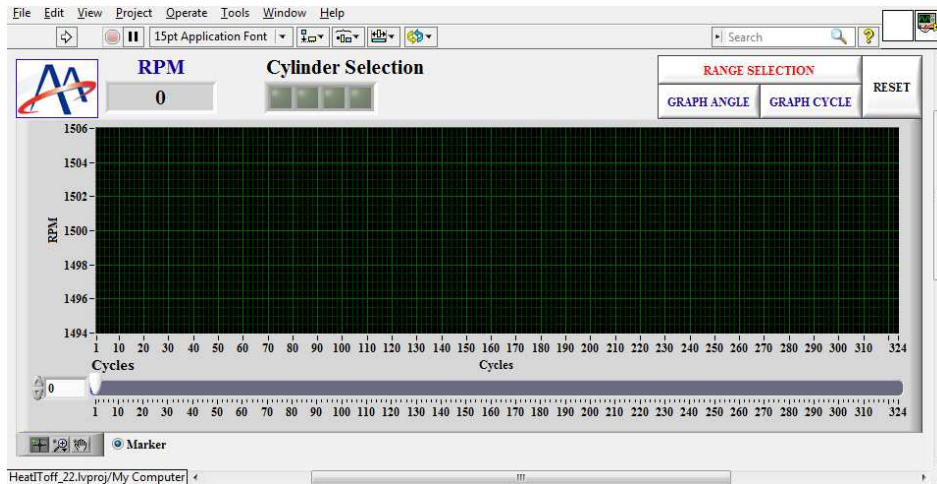


**Figura 2.2.1:** Finestra *Project Explorer*

al cui interno è possibile trovare:

- le funzioni *Run-Time menu* (cartella *Menu*), che permettono di allestire la barra dei comandi che compare nella parte superiore del pannello frontale;
- i VI pop up (cartella *Popup VIs*), che permettono di aprire le finestre relative agli strumenti *Range Selection*, *Graph Angle*, *Graph Cycle*,...;
- tutti i programmi richiesti dal VI di base (cartella *VIs*);

Per aprire il VI relativo all'interfaccia principale descritta nel manuale utente, basta cliccare su *PostPro\_01* (Figura 2.2.2):



**Figura 2.2.2:** *Pannello frontale del VI PostPro\_01*

La combinazione di tasti Ctrl+E consente poi di accedere al codice che permette il funzionamento di HeatITOff. Data l'elevata estensione del software, in Figura 2.2.3 viene riportata un'immagine semplificativa dei principali elementi che lo compongono e che è possibile visualizzare con lo strumento *Navigation Window* (combinazione di tasti Ctrl+Shift+N):

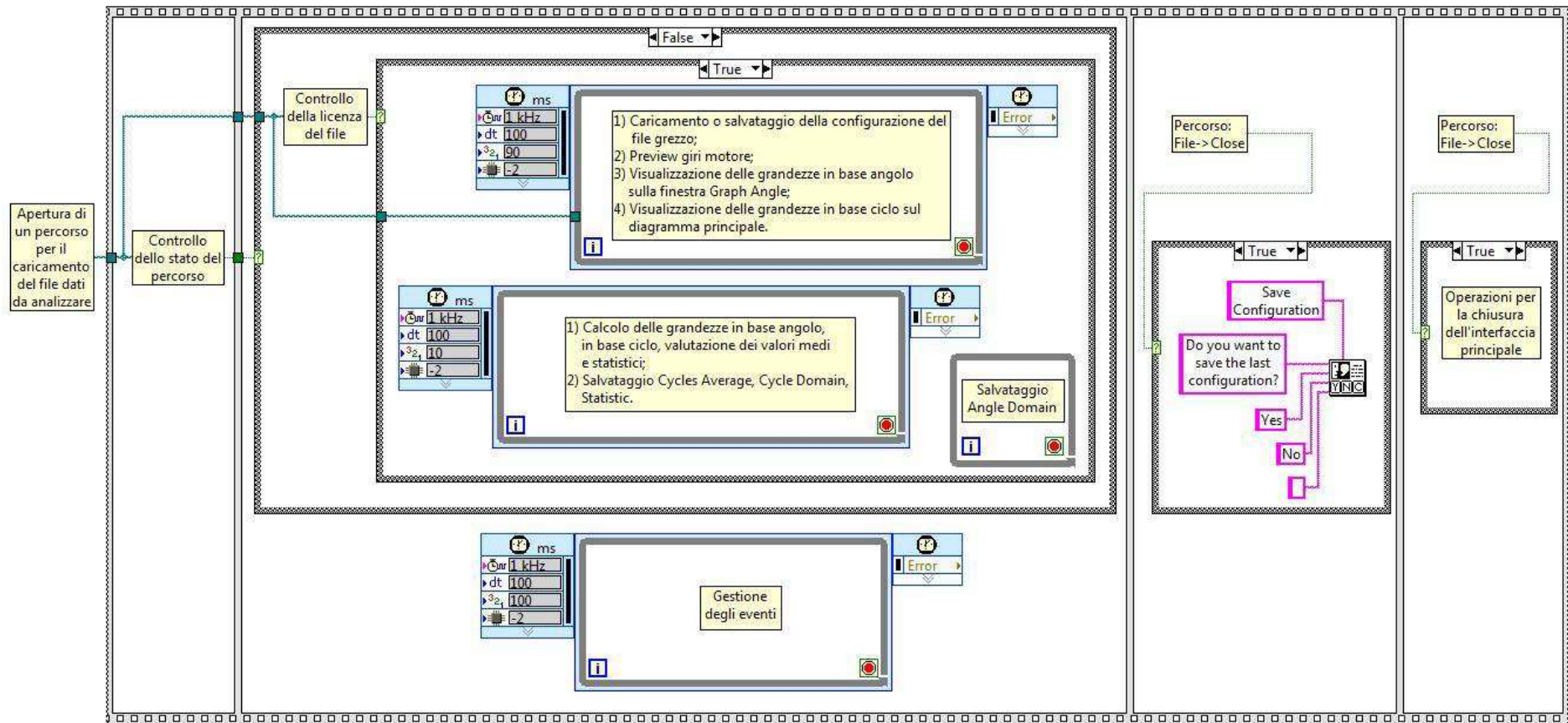




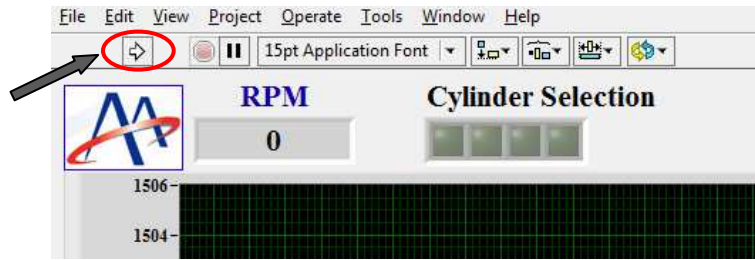
Figura 2.2.3: Architettura generale alla base di HeatIToff



Come è possibile vedere in Figura 2.2.3, gli elementi alla base del codice sono:

- 1 struttura *Flat Sequence*: è costituita da più frame che vengono eseguiti in sequenza da sinistra verso destra. Le operazioni all'interno di un frame vengono compiute solo quando tutti i valori ad esso collegati sono disponibili. Allo stesso modo, i dati lasciano il frame solo quando viene terminata l'esecuzione di tutte le operazioni al suo interno. Questo significa che l'input di un frame può dipendere dall'output di un altro frame;
- 4 strutture *Case*: la struttura *Case* contiene uno o più diagrammi secondari, la cui esecuzione dipende dal valore collegato al terminale di selezione (  ), che può essere un booleano (Vero/Falso), un intero (0,1,2,...), una stringa,... Per scorrere i diagrammi secondari disponibili, basta cliccare sulle frecce di incremento e decremento presenti nel selettore superiore della struttura;
- 1 struttura *While Loop*: ripete la porzione di codice al suo interno fino a quando il terminale condizionale (  ) riceve un particolare valore booleano;
- 3 strutture *Timed Loop*: esegue la porzione di codice al suo interno, temporizzando ogni iterazione al periodo specificato.

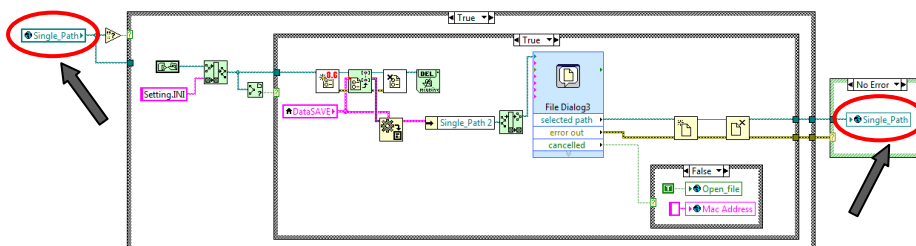
All'avvio di HeatITOff, che può essere eseguito cliccando sul pulsante *Run* (Figura 2.2.4) oppure usando la combinazione di tasti Ctrl+R:



**Figura 2.2.4:** Pulsante Run

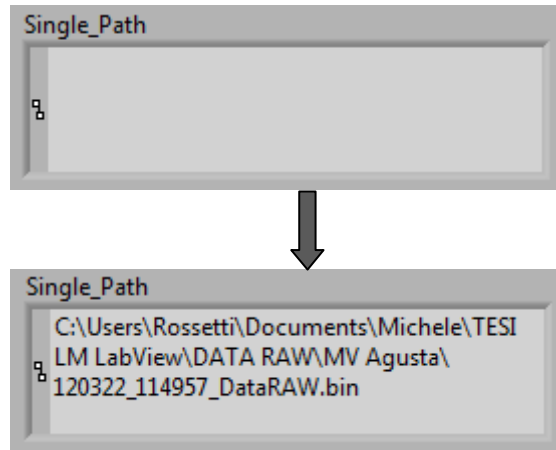
all'inizio della struttura *Flat Sequence* compare una variabile globale. Tale variabile è denominata *Single\_Path*, perché permette di aprire un percorso per il caricamento del file grezzo da analizzare. Non essendo ancora stato selezionato nessun file dati, tale variabile risulta vuota, di conseguenza Labview, nel percorrere la struttura *Flat Sequence* da sinistra verso destra, quando arriva al frame in cui sono presenti i diversi Loop, avvia solamente quello dedicato alla gestione degli eventi. Questo succede perché i rimanenti Loop si trovano all'interno della struttura *Case* comandata dal controllo dello stato del percorso: se l'uscita di tale controllo è vera (percorso vuoto), il sottodiagramma della struttura *Case* che viene eseguito dal programma è quello relativo al caso *True*, ovvero non viene fatta nessuna operazione.

Il percorso *File* → *Open* descritto nel manuale utente, a livello software richiama un pop up denominato *Open\_file*. Lo schema a blocchi di questo VI ha il compito di aggiornare la variabile globale *Single\_Path* (Figura 2.2.5):



**Figura 2.2.5:** Aggiornamento variabile *Single\_Path*

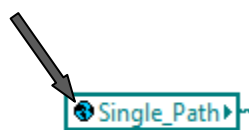
Selezionato il file desiderato, la variabile viene aggiornata in questo modo (Figura 2.2.6):



**Figura 2.2.6:** Variabile *Single\_Path* aggiornata col nome del file che si vuole analizzare

Ora che la variabile è stata aggiornata, l'uscita del controllo dello stato del percorso risulta falsa e anche i rimanenti Loop vengono avviati. Il file grezzo può quindi essere processato dal primo *Timed Loop*, una volta controllata la licenza.

*Single\_Path* è definita variabile "globale" (Figura 2.2.7):



**Figura 2.2.7:** Simbolo che caratterizza una variabile globale

perché viene scritta in un VI che è di fondamentale importanza quando si vuole realizzare un Tool complesso. Questo VI è denominato *gl\_PostPro* e la sua peculiarità sta nel non avere uno schema a blocchi (presenta solamente il pannello frontale). Nel pannello frontale vengono scritte e aggiornate tutte le variabili

globali richiamate all'interno dell'intero progetto, quindi sia nel VI di base sia nei subVI.

Per vedere in quali zone del software la variabile viene letta oppure scritta, bisogna cliccare con il tasto destro del mouse sulla variabile stessa e seguire il percorso *Find* → *Global References*.

Nei quattro capitoli successivi si entrerà più nel dettaglio delle operazioni che ogni singolo Loop è in grado di eseguire.

# CAPITOLO 3

## TIMED LOOP DI VISUALIZZAZIONE

In questo capitolo viene riportata una descrizione della serie di operazioni che vengono eseguite nel primo *Timed Loop* (schematizzato nelle Figure 3.1-3.3). Si tratta di una struttura ad alta priorità, perché il parametro *Priority* è impostato a 90 e più è alto il valore scelto, maggiore è la priorità con cui devono essere eseguite le operazioni all'interno del Loop.

Partendo a leggere lo schema a blocchi da sinistra, compare subito una struttura *Flat Sequence*. Questa struttura è composta da 3 frame numerati da delle etichette di colore blu:

**Frame 1**

**Frame 2**

**Frame 3**

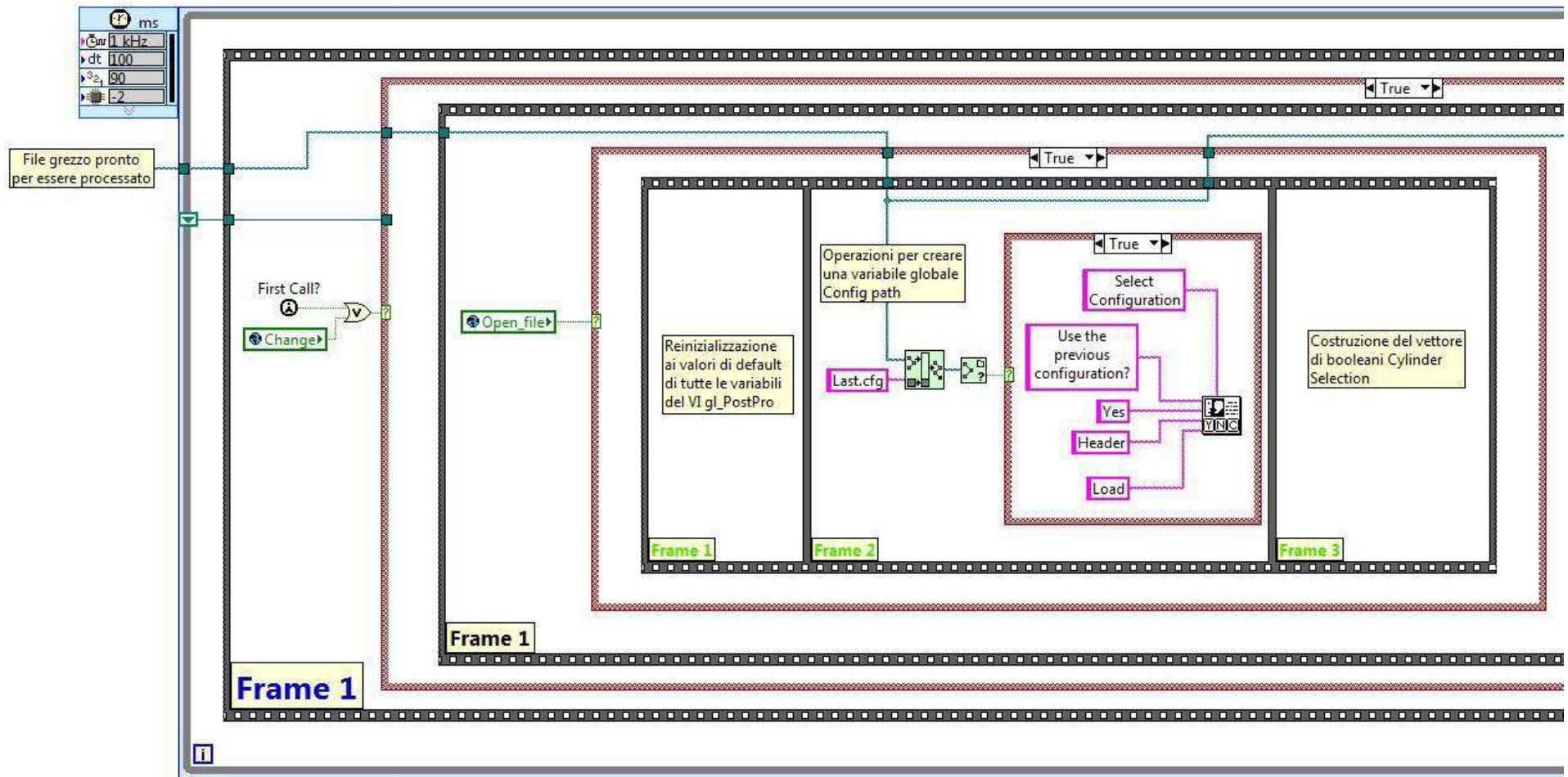
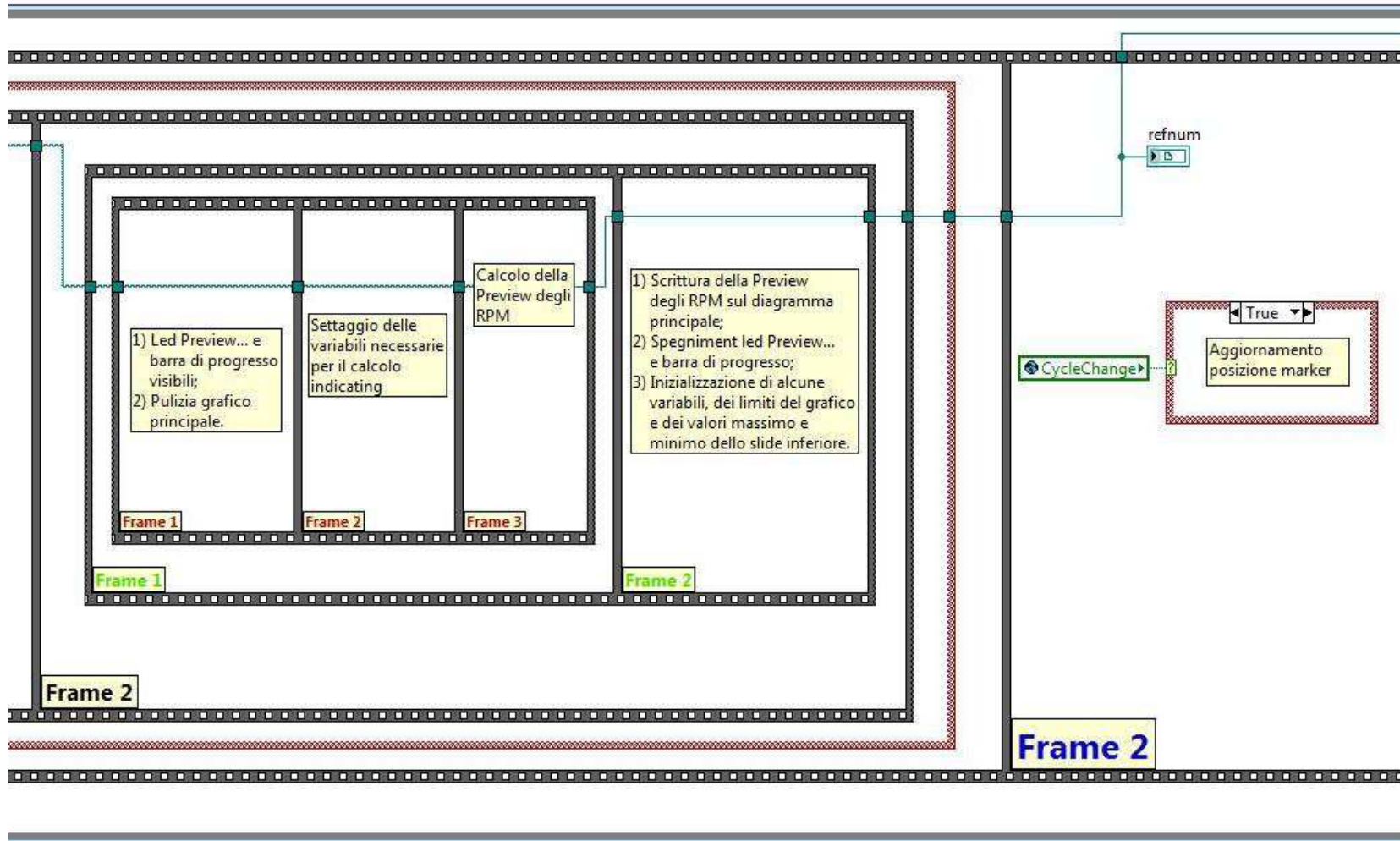
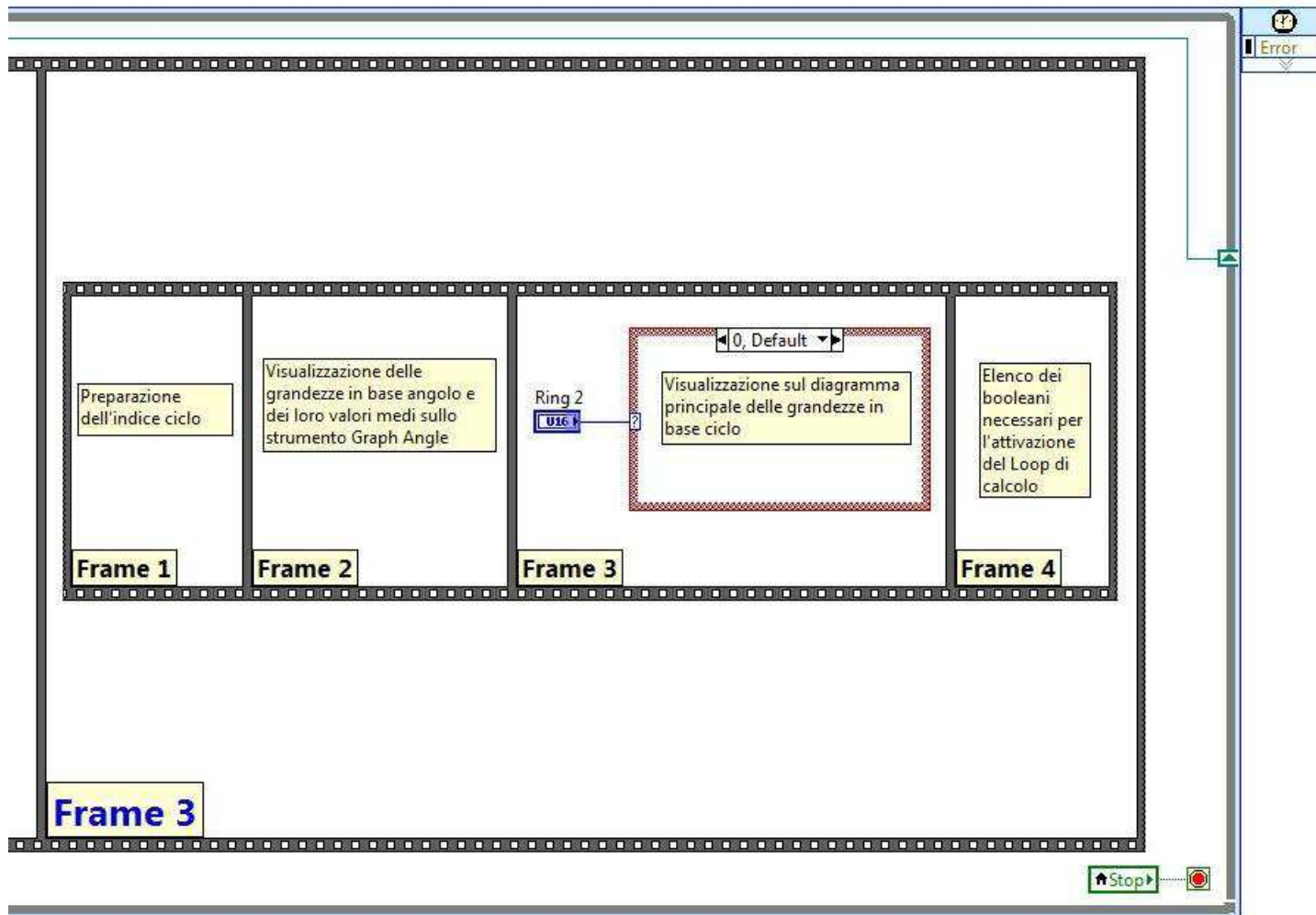


Figura 3.1: Parte iniziale Timed Loop di visualizzazione



**Figura 3.2:** Parte centrale Timed Loop di visualizzazione



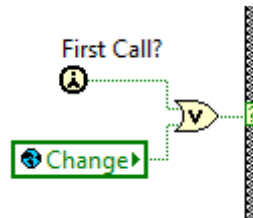
**Figura 3.3:** Parte finale Timed Loop di visualizzazione



Nel **Frame 1** è stata collocata una struttura *Case* che contiene:

- la parte di caricamento della vecchia configurazione del file o di salvataggio della configurazione attuale;
- la parte di calcolo dell'anteprima degli RPM in base ciclo.

Questa struttura *Case* si attiva (Figura 3.4) o quando il VI viene avviato per la prima volta (*First Call?*), oppure quando la variabile globale *Change* passa a *True* (come nel caso in cui venga apportata una modifica ai parametri di configurazione):



**Figura 3.4:** *OR logico*

Nel **Frame 2** invece è presente una struttura *Case* che, se attivata, permette di aggiornare la posizione del marker ogni volta che viene cambiato il ciclo selezionato.

Infine, il **Frame 3** contiene tutte le operazioni necessarie per la visualizzazione sulla finestra *Graph Angle* delle grandezze in base angolo e per la visualizzazione sul diagramma principale delle grandezze in base ciclo.

### 3.1–Caricamento o salvataggio della configurazione del file grezzo

Partendo dal **Frame 1**, nel caso in cui l'uscita dell'*OR* logico rappresentato in Figura 3.4 sia vera, Labview esegue ciò che si trova all'interno della struttura *Case* corrispondente. Entrando più nel dettaglio, è possibile vedere una struttura *Flat Sequence* suddivisa in due frame:

**Frame 1**

**Frame 2**

Nel **Frame 1** vengono eseguite le operazioni necessarie per il caricamento o il salvataggio della configurazione del file; nel **Frame 2** viene invece calcolata l'anteprima dei giri motore in base ciclo e verrà trattata nel paragrafo successivo.

Concentrandosi allora sul **Frame 1**, è possibile individuare un'altra struttura *Case* comandata dalla variabile globale *Open\_file*. Questo significa che le operazioni che si trovano all'interno vengono eseguite nel momento in cui viene richiamato il pop up *Open\_file*.

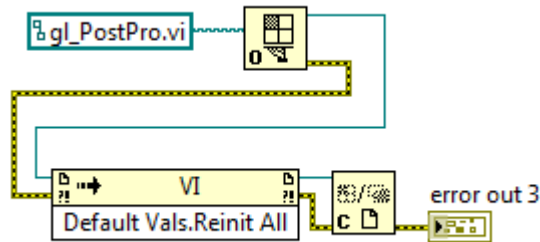
All'interno di questa struttura *Case* è presente una ulteriore struttura *Flat Sequence* divisa in 3 frame:

**Frame 1**

**Frame 2**

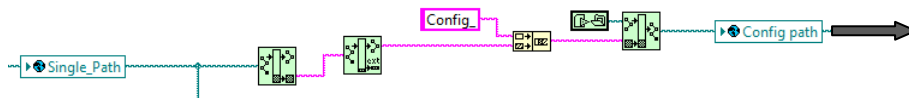
**Frame 3**

Nel **Frame 1** vengono inizializzate ai valori di default tutte le variabili globali presenti nel VI *gl\_PostPro* tramite un *Invoke Node*, ovvero un blocco di Labview che richiama una proprietà o un'azione sul VI che ha in input (Figura 3.1.1):



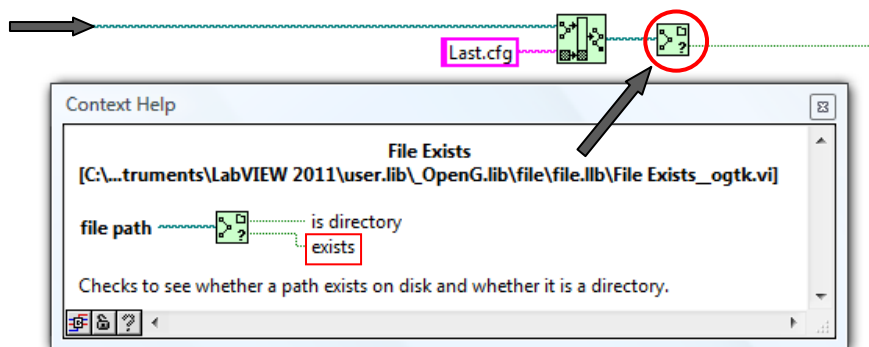
**Figura 3.1.1:** *Invoke Node che agisce sul VI gl\_PostPro*

Nel **Frame 2** viene creata una variabile globale *Config path* in cui il software scrive il percorso della cartella dove vengono salvate le diverse configurazioni (Figura 3.1.2):



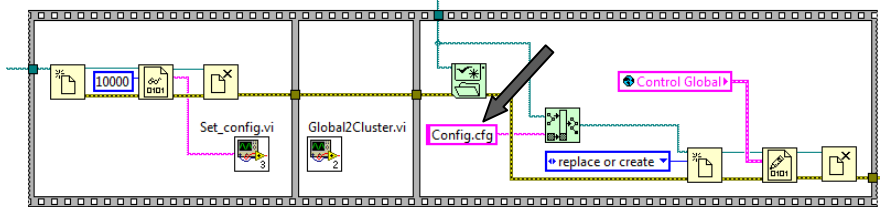
**Figura 3.1.2:** *Costruzione della variabile Config path*

Se in questa cartella non esiste un file *Last.cfg* (Figura 3.1.3):



**Figura 3.1.3:** *Context Help del blocco evidenziato*

allora viene eseguito il caso *False* della struttura e viene creato un file *Config.cfg* contenente la configurazione attuale (Figura 3.1.4):

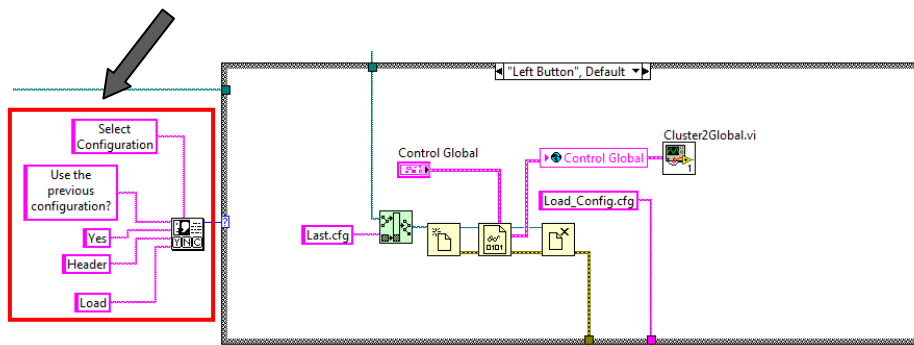


**Figura 3.1.4:** Operazioni per la costruzione del file *Config.cfg*

Altrimenti, se il file esiste già, viene data all'utente la possibilità di:

- caricare la configurazione precedente;
- caricare la configurazione dell'header originale;
- caricare un altro tipo di configurazione.

come è possibile vedere in Figura 3.1.5:



**Figura 3.1.5:** Cliccando su *Yes* nella finestra di dialogo, le operazioni svolte sono quelle raffigurate

Infine, nel **Frame 3** viene costruito il vettore di booleani *Cylinder Selection*, con un numero di elementi pari al numero di segnali *Indicating* (Figura 3.1.6):

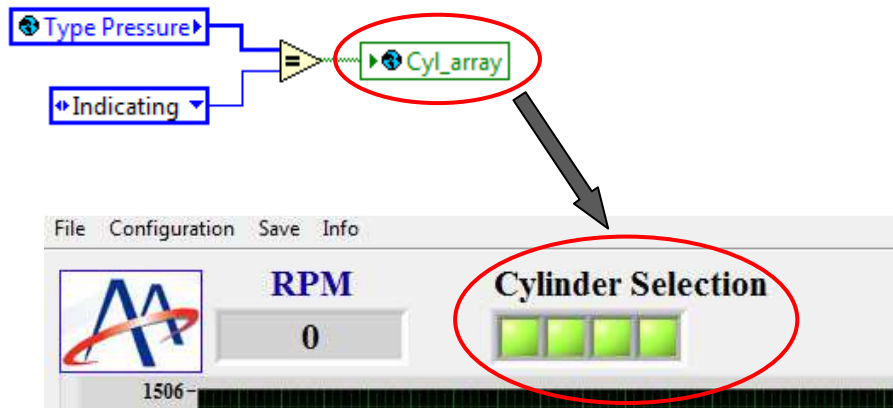


Figura 3.1.6: Confronto fra ogni elemento del vettore *Type Pressure* e il termine *Indicating*

### 3.2– Visualizzazione dell’anteprima giri motore in base ciclo

Ritornando alla *Flat Sequence* descritta all’inizio del paragrafo 3.1 e considerando questa volta il **Frame 2**, è possibile notare una struttura di questo tipo (Figura 3.2.1):

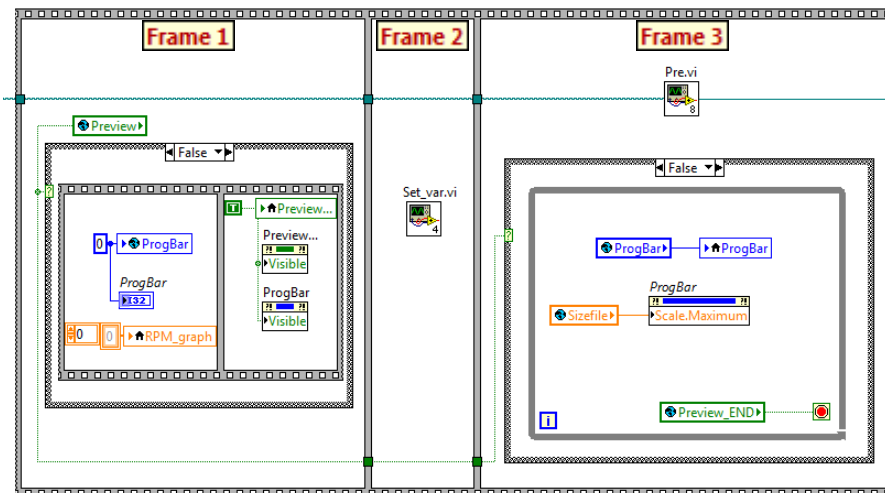


Figura 3.2.1: *Frame 1*

In sequenza, le operazioni svolte sono:

- 1) **Frame 1** → vengono resi visibili il led *Preview...* e la barra di progresso. La barra di progresso viene inizializzata a zero;
- 2) **Frame 2** → vengono impostate le variabili necessarie al calcolo indicating (nel VI *Set\_var* vengono calcolate e scritte le variabili globali relative alla cilindrata, alla variazione infinitesima di volume, ai coefficienti utilizzati nei filtri,...);
- 3) **Frame 3** → viene calcolata l'anteprima dei giri motore in base ciclo (Figura 3.2.2):

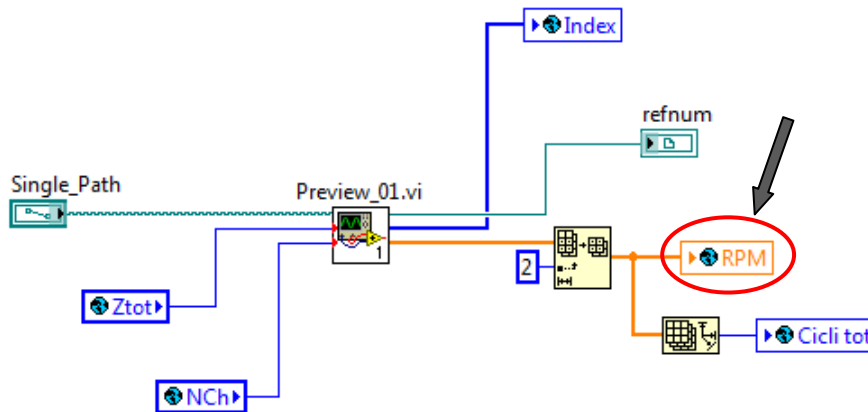


Figura 3.2.2: Input e output del VI Pre

Successivamente (**Frame 2**), il software provvede a scrivere l'anteprima degli RPM sul diagramma principale (Figura 3.2.3):



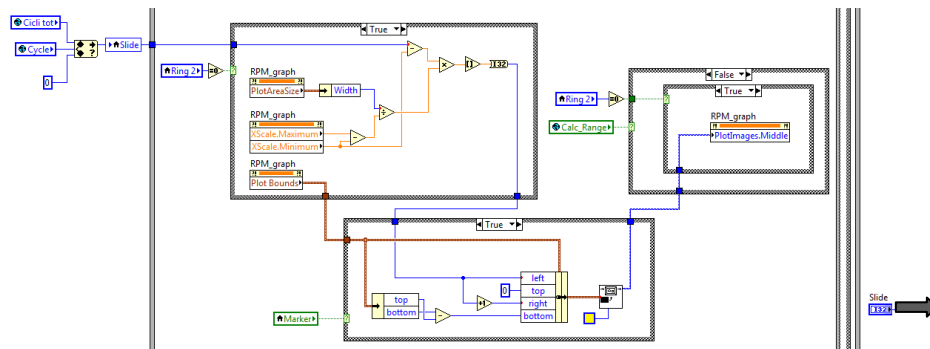
Figura 3.2.3: Scrittura preview giri motore sul diagramma principale

a rendere non più visibili il led *Preview...* e la barra di progresso, ad impostare i limiti del grafico e dello slide inferiore, ad inizializzare

alcune delle variabili che vengono poi utilizzate nella parte finale di visualizzazione.

### 3.3–Visualizzazione delle grandezze in base angolo sullo strumento Graph Angle

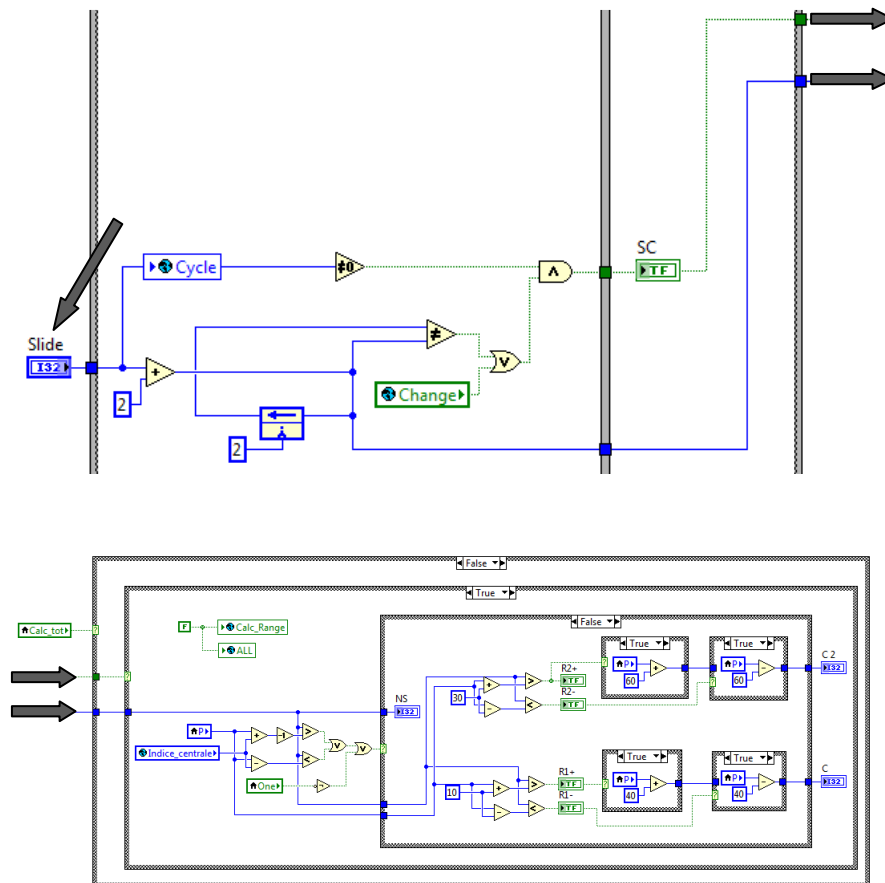
In Figura 3.3.1 viene riportato quello che accade nel **Frame 2**:



**Figura 3.3.1:** Aggiornamento posizione marker

ovvero, tutte le volte che la variabile globale *CycleChange* passa a *True*, vuol dire che con le frecce disponibili sulla tastiera del PC viene cambiato il ciclo di interesse. La struttura *Case* comandata da questa variabile globale permette allora di aggiornare la posizione del marker, noto il valore numerico dello slide *Cycles* (che dipende dal ciclo selezionato).

Nel passaggio successivo (**Frame 3**→**Frame 1**) viene preparato l'indice (il ciclo) in corrispondenza del quale andare a visualizzare le grandezze su base angolo (Figura 3.3.2):



**Figura 3.3.2:** Preparazione dell'indice in corrispondenza del quale andare a visualizzare le grandezze su base angolo

Il **Frame 2** si occupa solamente della visualizzazione delle grandezze in base angolo. Il calcolo di tali grandezze avverrà in un Loop dedicato, che verrà trattato nel capitolo successivo. In particolare, in questo frame viene letta una variabile locale denominata *output array* (Figura 3.3.3):



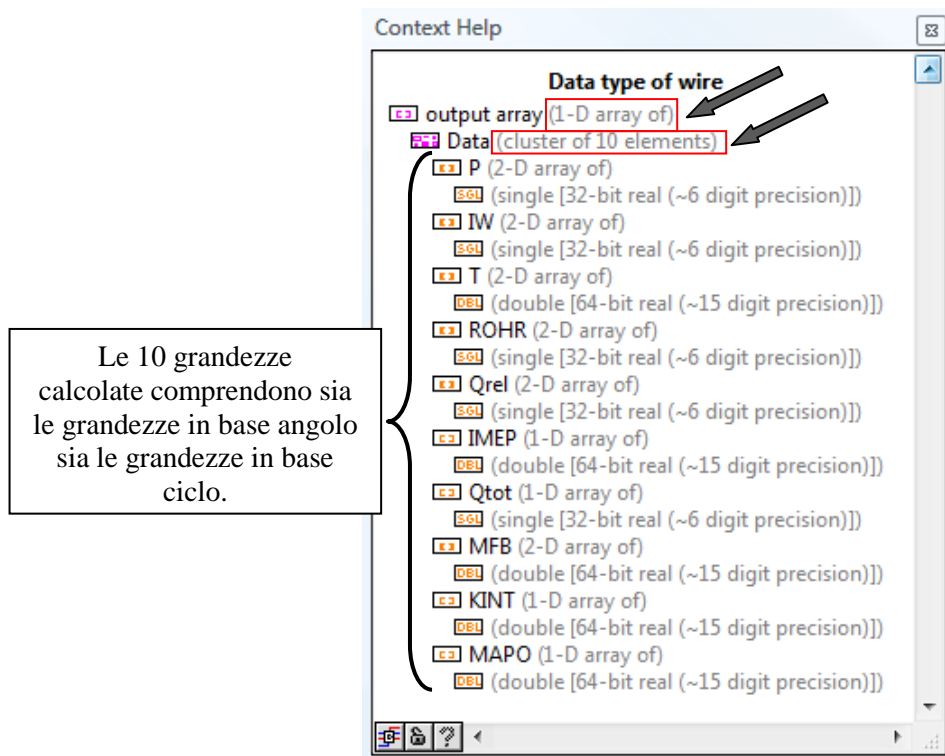
**Figura 3.3.3:** Simbolo che caratterizza una variabile locale

La variabile “locale”, a differenza della variabile globale, viene usata per leggere o scrivere da un controllo oppure un indicatore di un ben preciso VI, quindi non da tutti i programmi dell'intero progetto. Per



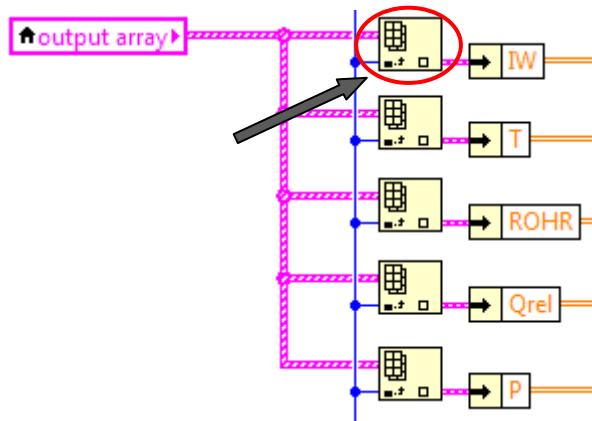
vedere in quali zone del software la variabile viene letta oppure scritta, bisogna cliccare con il tasto destro del mouse sulla variabile stessa e seguire il percorso *Find* → *Local Variables* oppure *Find* → *Terminal*.

In questo caso, *output array* non è altro che un vettore di 60 elementi (vedi capitolo 4, paragrafo 4.2), dove ogni elemento è un cluster (un “contenitore” che raggruppa dati di tipo diverso come stringhe, vettori, matrici,...) contenente tutte le grandezze calcolate per un solo ciclo motore (Figura 3.3.4):



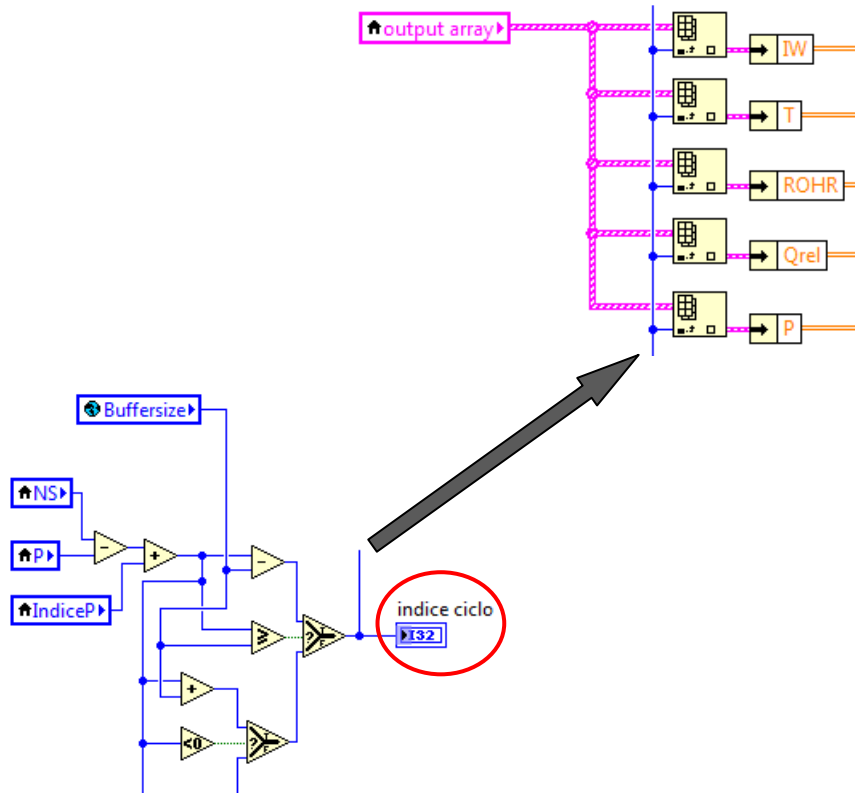
**Figura 3.3.4:** *Context Help della variabile output array*

Per visualizzare solo gli andamenti delle grandezze in base angolo relative al ciclo selezionato col marker sul diagramma principale, questo vettore viene collegato ad un blocco di Labview denominato *Index Array* (Figura 3.3.5):



**Figura 3.3.5:** *Blocco Index Array*

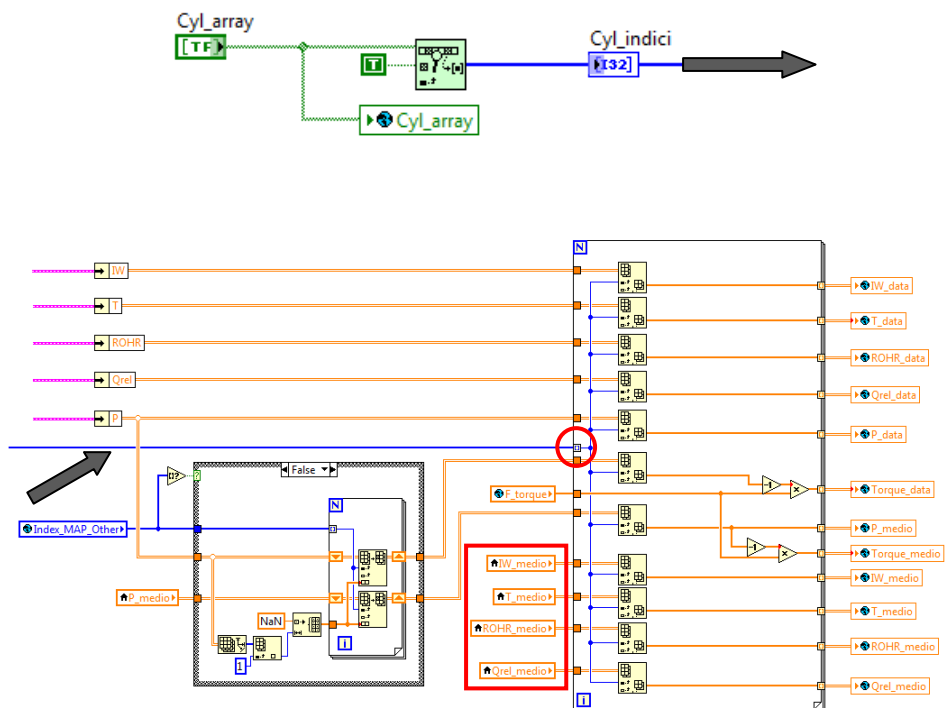
Questo blocco restituisce l'elemento del vettore corrispondente all'indice connesso. Successivamente, con la funzione *Unbundle By Name* si va a vedere nel cluster la grandezza che interessa e la si seleziona. In questo caso l'indice collegato al blocco *Index Array* sarà *indice ciclo* (Figura 3.3.6):



**Figura 3.3.6:** *Indice collegato ai diversi blocchi Index Array*

dove *indice ciclo* viene calcolato con i passaggi visti in precedenza (vedi Figura 3.3.2).

Una volta che si hanno tutte le grandezze calcolate per il ciclo di interesse, queste vengono inviate in un ciclo *For* che viene eseguito un numero di volte pari al numero di cilindri attivi nel comando *Cylinder Selection* (Figura 3.3.7):



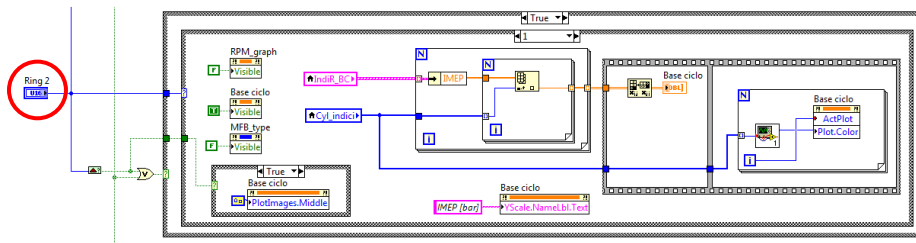
**Figura 3.3.7:** Valori medi e ciclo *For* indicizzato dal vettore *Cyl\_indici*

I valori medi entrano nella parte inferiore del ciclo *For* rappresentato in Figura 3.3.7: vengono valutati nel Loop di calcolo, poi scritti in variabili locali rilette in questa zona dello schema a blocchi.

Le variabili globali in uscita dal ciclo *For* verranno poi lette all'interno del pop up *GraphDisplay\_Angle*, che ne consente la visualizzazione sullo strumento *Graph Angle*.

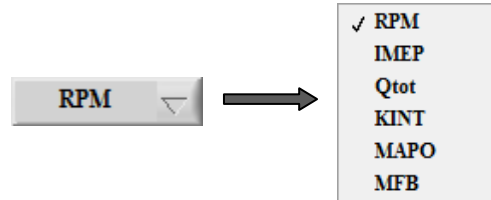
### 3.4–Visualizzazione delle grandezze in base ciclo sul diagramma principale

In Figura 3.4.1 viene riportata la sequenza di operazioni svolta nel **Frame 3**:



**Figura 3.4.1:** Controllo Ring 2 collegato alla struttura Case

Il controllo *Ring 2* non è altro che il menu a tendina descritto nel manuale utente (Figura 3.4.2):



**Figura 3.4.2:** Menu a tendina descritto nel manuale utente

La struttura *Case* avrà quindi 6 diagrammi secondari, numerati da 0 a 5, dove il primo corrisponde alla selezione di default, ovvero gli RPM. Analogamente a quanto detto per la visualizzazione delle grandezze in base angolo, nel Loop di calcolo viene costruito un vettore *IndiR\_BC*, richiamato in questa zona dello schema a blocchi con una variabile locale (Figura 3.4.3):



**Figura 3.4.3:** Variabile locale per le grandezze in base ciclo

Questo vettore contiene un numero di elementi pari al numero di cicli calcolati con lo strumento *Range Selection*. Anche in questo caso, ogni elemento del vettore è un cluster che racchiude però tutte le grandezze in base ciclo (Figura 3.4.4):

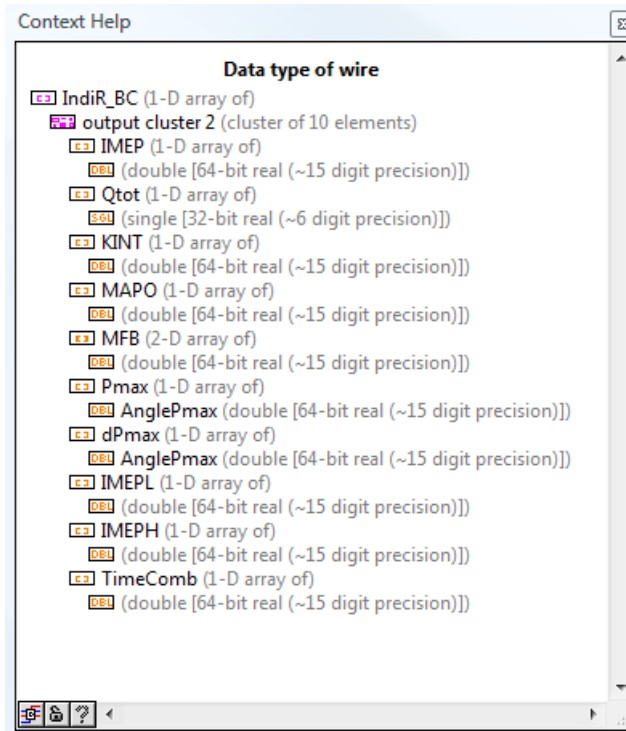


Figura 3.4.4: Context Help della variabile IndiR\_BC

Se ad esempio si vuole visualizzare la *IMEP*, il procedimento è analogo a quello spiegato per la visualizzazione delle grandezze in base angolo sulla finestra *Graph Angle* (Figura 3.4.5):

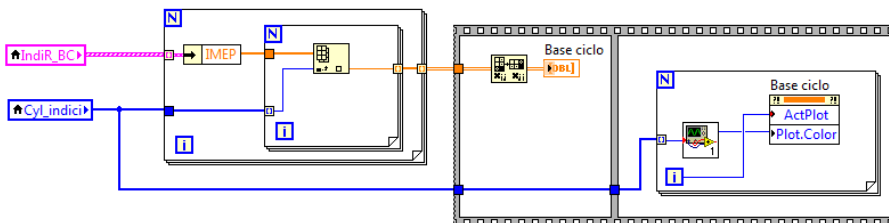


Figura 3.4.5: Operazioni per la visualizzazione della IMEP sul diagramma principale

### 3.5–Attivazione Loop di calcolo, disattivazione Loop di visualizzazione

Al termine di quest'ultima struttura *Flat Sequence* (**Frame 3**→**Frame 4**) viene riportato un elenco di variabili, collegate tutte quante ad un *OR* logico. Se l'uscita di questo *OR* è *True*, allora viene attivata la struttura *Case* che si trova all'inizio del Loop di calcolo (Figura 3.5.1):

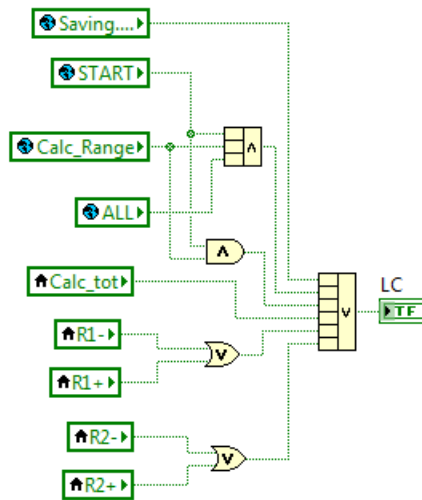


Figura 3.5.1: OR logico il cui output è collegato all'indicatore LC

In conclusione, la variabile locale *Stop* termina l'esecuzione del *Timed Loop* di visualizzazione una volta selezionato il percorso *File* → *Close* (Figura 3.5.2):

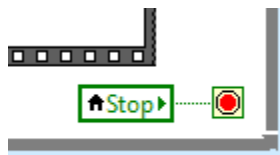


Figura 3.5.2: Terminale condizionale collegato alla variabile Stop

# CAPITOLO 4

## TIMED LOOP DI CALCOLO

In questo capitolo viene riportata una descrizione della serie di operazioni che vengono eseguite nel secondo *Timed Loop*. Si tratta di una struttura a bassa priorità, essendo il parametro *Priority* impostato a 10.

Entrando all'interno del Loop (schematizzato nelle Figure 4.1-4.3) e partendo sempre a leggere da sinistra, compare una struttura *Case*:

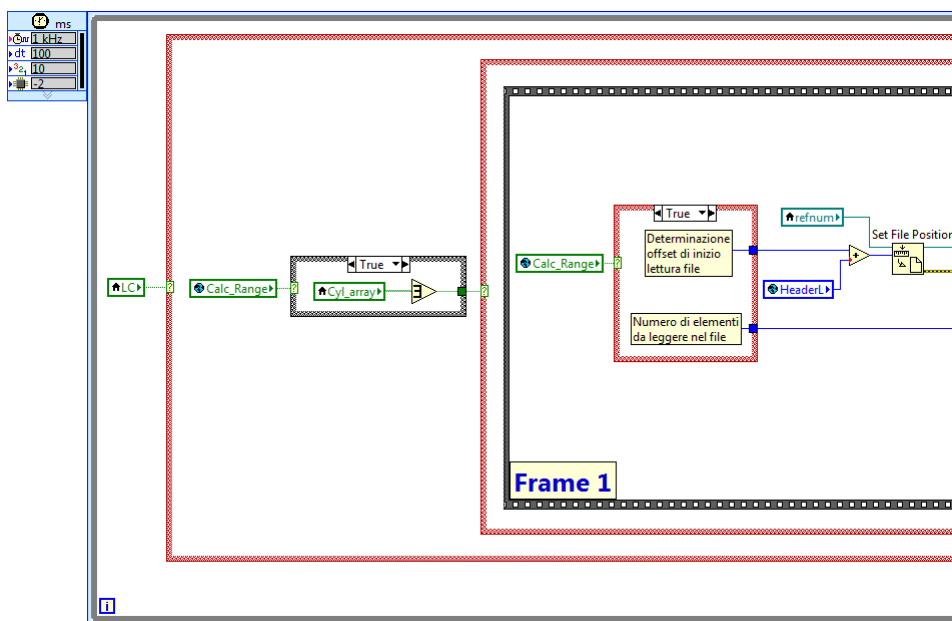


Figura 4.1: Parte iniziale *Timed Loop* di calcolo

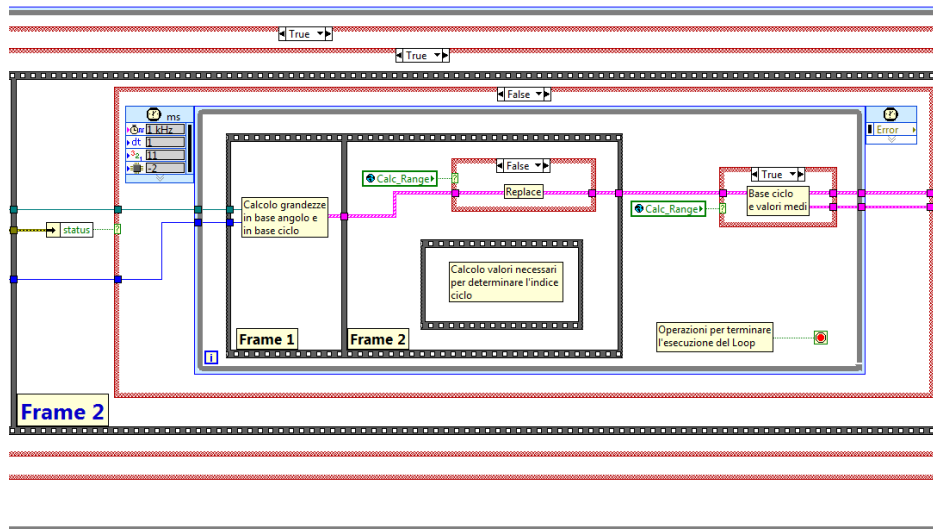


Figura 4.2: Parte centrale Timed Loop di calcolo

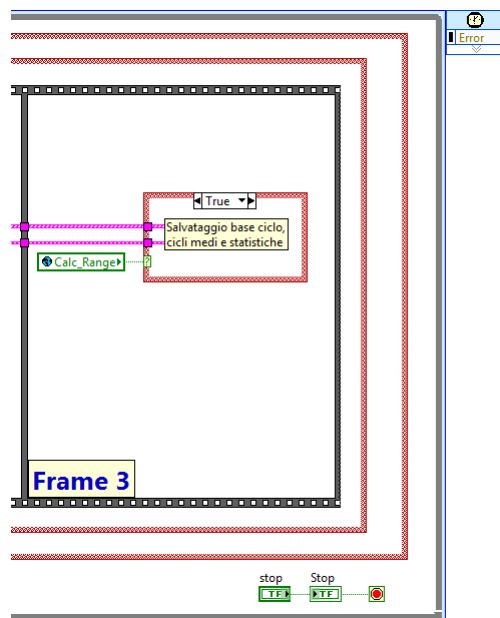
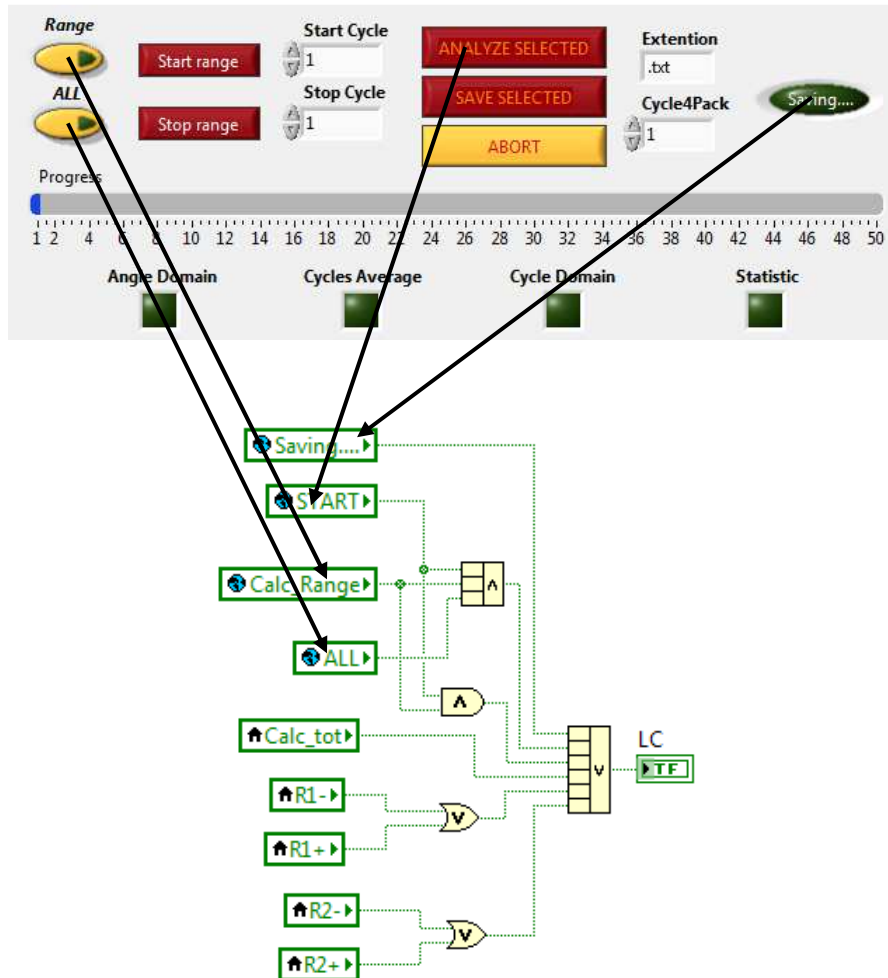


Figura 4.3: Parte finale Timed Loop di calcolo



Questa struttura è comandata dalla variabile locale *LC* e viene scritta dall'indicatore citato all'inizio del paragrafo 3.5. In particolare, alcune delle variabili che possono attivare il Loop di calcolo si trovano nel pop up *ProxRange*, che non è altro che l'interfaccia dello strumento *Range Selection* (Figura 4.4):



**Figura 4.4:** Controlli che attivano l'indicatore *LC*

La variabile globale *Calc\_Range* è molto importante perché distingue i due modi di funzionare del Loop di calcolo:

- *Calc\_Range=True* → indica che bisogna eseguire il calcolo per un numero di cicli compreso tra *Start Cycle* e *Stop Cycle*;

- *Calc\_Range=False* → indica che bisogna eseguire il calcolo per un vettore di elementi centrato esattamente sul ciclo selezionato con il marker.

## 4.1–Calcolo per un numero di cicli compreso tra Start Cycle e Stop Cycle

Attivato il Loop di calcolo, compare un'altra struttura *Case* comandata però da una logica di questo tipo (Figura 4.1.1):

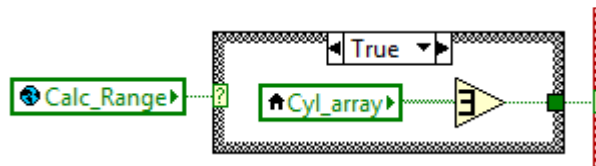
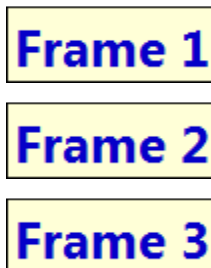


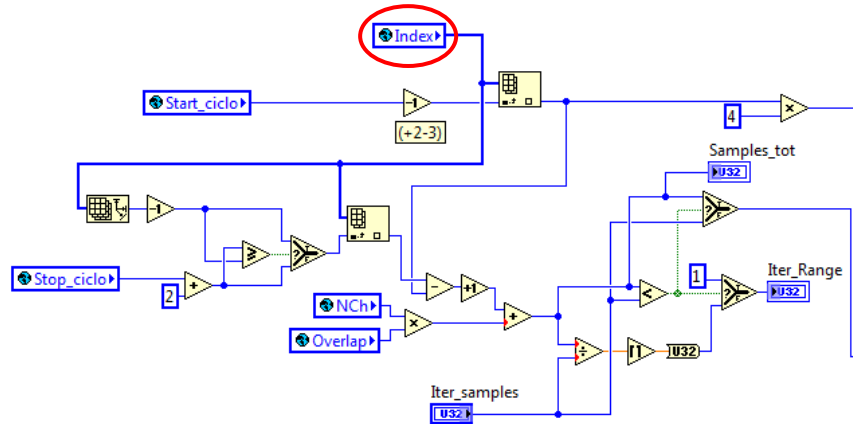
Figura 4.1.1: OR logico su un vettore

Se *Calc\_Range=True*, viene eseguito un *OR* logico su tutti gli elementi del vettore *Cylinder Selection*: basta che almeno uno degli elementi del vettore sia impostato a *True* per poter passare alle operazioni che vengono eseguite all'interno della struttura *Case*.

La prima struttura che compare all'interno è una *Flat Sequence* composta da 3 frame:



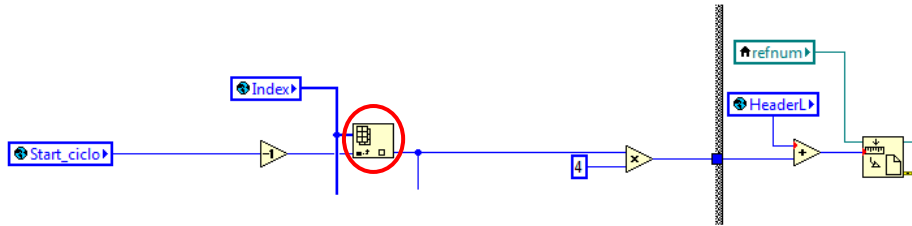
Nel **Frame 1** compare una ulteriore struttura *Case* comandata dalla variabile *Calc\_Range*. Se tale variabile è impostata a *True*, viene eseguita una logica di questo tipo (Figura 4.1.2):



**Figura 4.1.2:** Logica per la determinazione dell'offset di partenza e del numero di elementi da leggere nel file

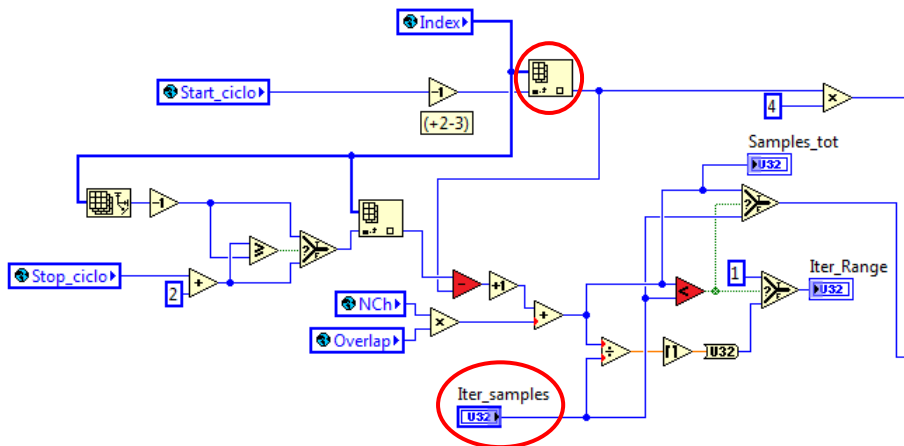
La variabile globale *Index* viene scritta nel VI *Pre* (vedi Figura 3.2.2, capitolo 3) e non è altro che un vettore dove ogni elemento rappresenta la posizione che i diversi marker di inizio ciclo hanno all'interno del file grezzo iniziale. Questo vettore viene collegato a due *Index Array*:

- nel primo (Figura 4.1.3) si va a cercare l'elemento corrispondente allo *Start\_ciclo* selezionato con lo strumento *Range Selection* e il risultato di questa operazione, una volta sommato alla lunghezza dell'header, sarà l'offset da cui iniziare a leggere il file grezzo per poter eseguire i calcoli:



**Figura 4.1.3:** Determinazione offset da cui iniziare a leggere il file grezzo

- nel secondo (Figura 4.1.4) si va invece a cercare l'elemento corrispondente allo *Stop\_ciclo* selezionato con lo strumento *Range Selection*. La differenza tra questo valore e quello corrispondente allo *Start\_ciclo* viene poi confrontata con il valore numerico *Iter\_samples* per determinare il numero di bytes da leggere nel file grezzo:



**Figura 4.1.4:** Determinazione numero di elementi da leggere nel file grezzo

dal momento che *Iter\_samples* è impostato di default al valore di 62000, per limitare il numero di cicli analizzati ad ogni iterazione del Loop di calcolo, e che la differenza tra gli elementi del vettore *Index* corrispondenti a *Start\_ciclo* e *Stop\_ciclo* assume sempre un valore superiore, il booleano in uscita dalla disuguaglianza è sempre *False*. Di conseguenza, il valore numerico in uscita dal *Select* è sempre pari a 62000 e

saranno quindi i bytes da leggere nel file grezzo di partenza (Figura 4.1.5):

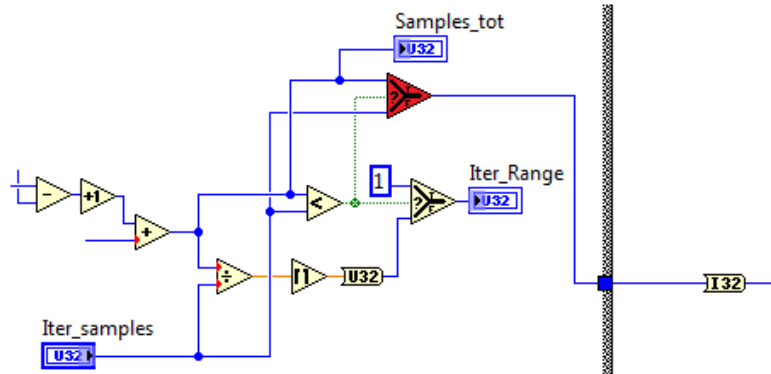


Figura 4.1.5: Blocco Select

Passando al **Frame 2**, è possibile vedere che la struttura *Case* rappresentata è comandata dalla variabile *status* (Figura 4.1.6). Questa variabile fa parte del cluster di 3 elementi in uscita dal blocco di Labview *Set File Position*:

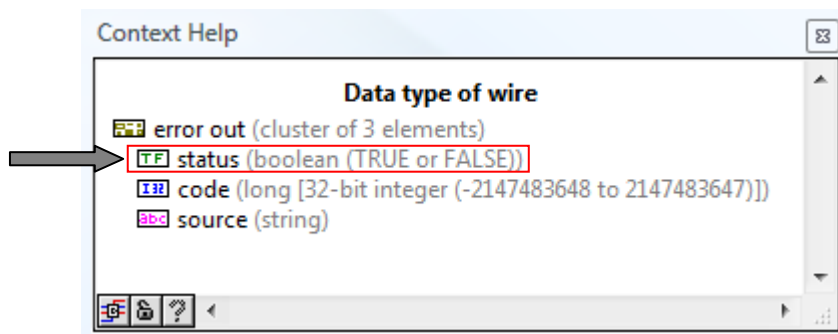


Figura 4.1.6: Context Help del cluster error out

se non sono sorti dei problemi, l'uscita della variabile *status* è sempre uguale a *False*. Infatti, la parte di codice riguardante il calcolo delle diverse grandezze è stata scritta proprio nel sottodiagramma *False* della struttura *Case*.



- pressione media indicata (*IMEP*);
- massimo calore rilasciato (*Q<sub>tot</sub>*);
- frazione di massa bruciata (*MFB*);
- Knock Integral (*KINT*);
- Maximum Amplitude of Pressure Oscillation (*MAPO*).

Il VI di calcolo fornisce in uscita un vettore, denominato *Indi*, dove ogni elemento è un cluster in cui sono state calcolate le grandezze relative ad un ciclo motore. Il cluster viene costruito in questo modo (Figura 4.1.8):

IW					Matrice
$\frac{A}{T}$	0	0	0	0	
$\frac{A}{T}$	0	0	0	0	Matrice
$\frac{A}{T}$	0	0	0	0	
T					Matrice
$\frac{A}{T}$	0	0	0	0	
$\frac{A}{T}$	0	0	0	0	Matrice
$\frac{A}{T}$	0	0	0	0	
ROHR					Matrice
$\frac{A}{T}$	0	0	0	0	
$\frac{A}{T}$	0	0	0	0	Matrice
$\frac{A}{T}$	0	0	0	0	
Qrel					Matrice
$\frac{A}{T}$	0	0	0	0	
$\frac{A}{T}$	0	0	0	0	Vettore
$\frac{A}{T}$	0	0	0	0	
IMEP					Vettore
$\frac{A}{T}$	0	0	0	0	
Q <sub>tot</sub>					Vettore
$\frac{A}{T}$	0	0	0	0	
MFB					Matrice
$\frac{A}{T}$	0	0	0	0	
$\frac{A}{T}$	0	0	0	0	Matrice
$\frac{A}{T}$	0	0	0	0	
KINT					Vettore
$\frac{A}{T}$	0	0	0	0	
MAPO					Vettore
$\frac{A}{T}$	0	0	0	0	
P					Matrice
$\frac{A}{T}$	0	0	0	0	
$\frac{A}{T}$	0	0	0	0	Matrice
$\frac{A}{T}$	0	0	0	0	

**Figura 4.1.8:** Cluster di 10 elementi

Le dimensioni di ogni singolo vettore o matrice del cluster sono:

<b>Grandezza</b>	<b>n° righe</b>	<b>n° colonne</b>
<i>P</i>	cilindri selezionati	$\frac{720^\circ}{\text{risoluzione angolare scelta}}$
<i>IW</i>	cilindri selezionati	$\frac{720^\circ}{\text{risoluzione angolare scelta}}$
<i>T</i>	cilindri selezionati	$\frac{\text{ROHREnd} - \text{ROHRStart}}{\text{risoluzione angolare scelta}}$
<i>ROHR</i>	cilindri selezionati	$\frac{\text{ROHREnd} - \text{ROHRStart}}{\text{risoluzione angolare scelta}}$
<i>Qrel</i>	cilindri selezionati	$\frac{\text{ROHREnd} - \text{ROHRStart}}{\text{risoluzione angolare scelta}}$
<i>IMEP</i>	1	cilindri selezionati
<i>Qtot</i>	1	cilindri selezionati
<i>MFB</i>	valori percentuali stabiliti in Configuration Global	cilindri selezionati
<i>KINT</i>	1	cilindri selezionati
<i>MAPO</i>	1	cilindri selezionati

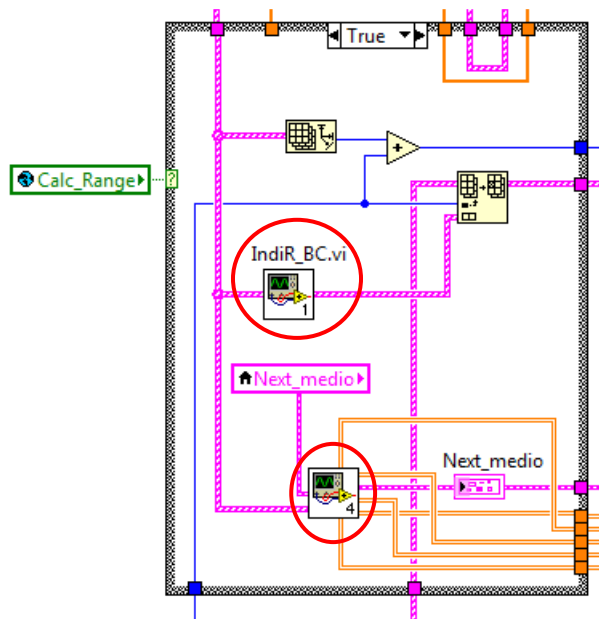
Se la velocità di rotazione del motore non è mantenuta costante, perché ad esempio il file grezzo è il risultato dell'acquisizione di un



giro di pista, il VI di calcolo fornisce in uscita un vettore con un numero di elementi variabile da iterazione a iterazione. Questo è dovuto al fatto che i segnali vengono campionati in base tempo, a frequenza costante. Di conseguenza, al variare della velocità di rotazione cambia il numero di cicli per i quali vengono calcolate le diverse grandezze (maggiore è la velocità di rotazione, maggiore sarà il numero di cicli contenuti nel vettore e viceversa).

Successivamente, il vettore *Indi* passa nel **Frame 2** dove, se *Calc\_Range=True*, non viene fatta nessuna operazione per quanto riguarda la struttura *Case*. Invece, la *Flat Sequence* ad 1 solo frame che si trova immediatamente sotto la struttura *Case* appena considerata verrà trattata nel paragrafo 4.2.

Finiti i frame, compare una ulteriore struttura *Case* comandata sempre dalla variabile globale *Calc\_Range* (Figura 4.1.9):



**Figura 4.1.9:** Determinazione grandezze in base ciclo e grandezze medie

Se *Calc\_Range=True*, il vettore *Indi* diventa l'input di due VI il cui compito è di preparare per il salvataggio rispettivamente:

- tutte le grandezze in base ciclo;
- le grandezze medie.

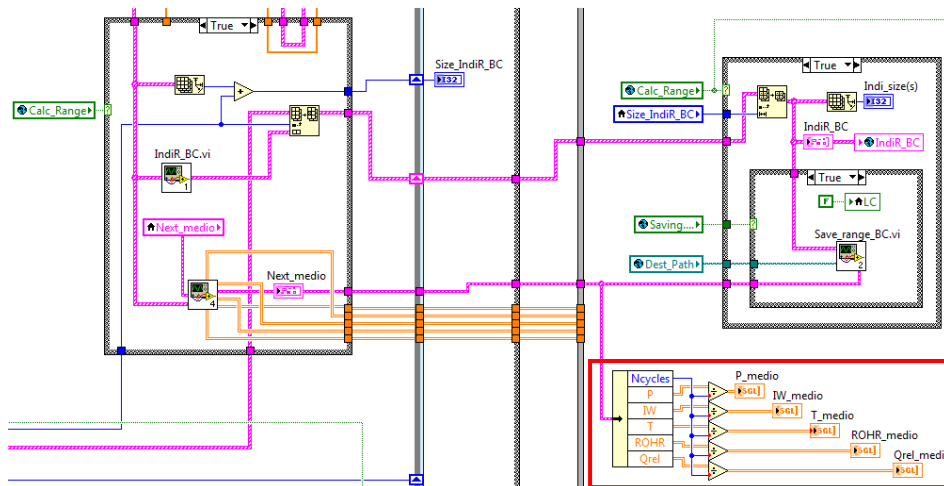
In particolare, nel VI *IndiR\_BC* alle grandezze in base ciclo già calcolate:

- *IMEP*;
- *Qtot*;
- *MFB*;
- *KINT*;
- *MAPO*.

vengono aggiunti i seguenti parametri:

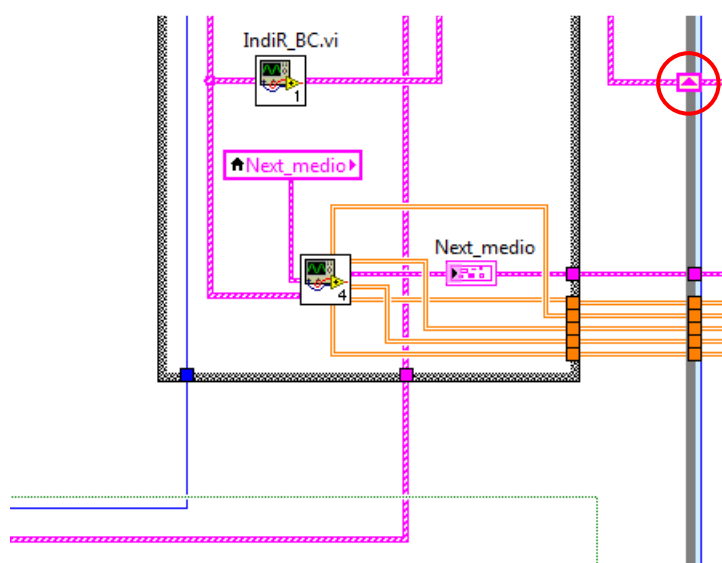
- *Pmax*;
- *dPmax*;
- *IMEPL*;
- *IMEPH*;
- *TimeComb*.

mentre le grandezze medie valutate sono poi spaccettate nel **Frame 3** (Figura 4.1.10) e richiamate con delle variabili locali nel Loop di visualizzazione (vedi Figura 3.3.7, capitolo 3):



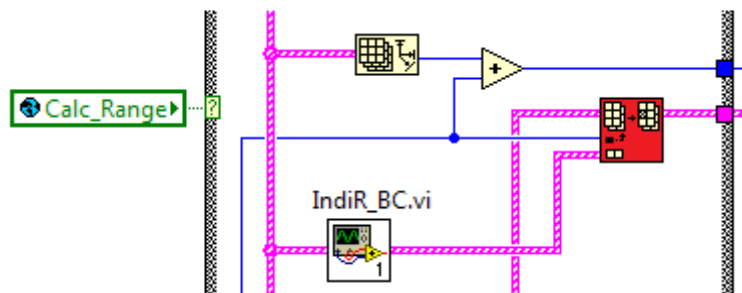
**Figura 4.1.10:** Valutazione grandezze medie per la visualizzazione sullo strumento Graph Angle

Le grandezze in base ciclo sin qui trattate vengono calcolate ad ogni iterazione per un numero di cicli che dipende dalla velocità di rotazione del motore. Per comporre uno ad uno i vettori costruiti ad ogni iterazione e costruire così un unico vettore che contenga un numero di cicli pari al range selezionato con i due controlli *Start Cycle* e *Stop Cycle*, viene utilizzato un registro a scorrimento sul *Timed Loop* (Figura 4.1.11):



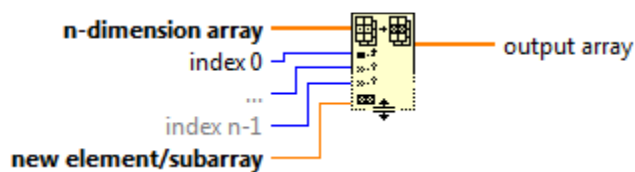
**Figura 4.1.11:** Shift Register sul Timed Loop

Il registro a scorrimento compare come una coppia di terminali con direzione opposta l'uno rispetto all'altro sui lati verticali della cornice del Loop e serve per trasferire i valori da un'iterazione del ciclo alla successiva. La logica utilizzata in questo caso per preparare un unico vettore contenente i diversi valori delle grandezze in base ciclo si serve sia del registro a scorrimento, sia di un blocco di Labview denominato *Replace Array Subset* (Figura 4.1.12):



**Figura 4.1.12:** *Blocco Replace Array Subset*

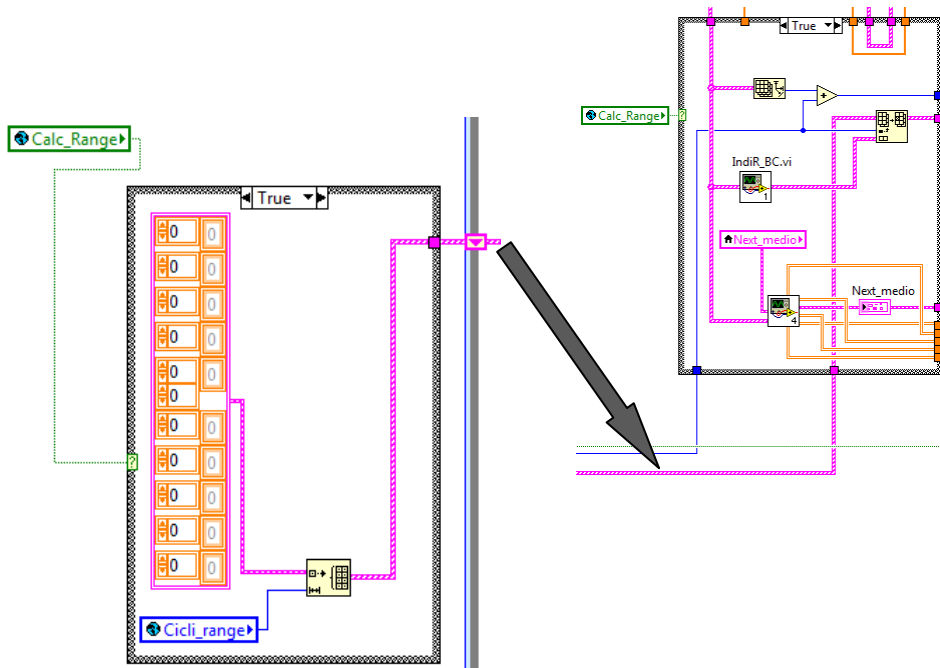
Questo blocco, come si vede in Figura 4.1.13, sostituisce uno o più elementi di un vettore di partenza, usato come base, con dei nuovi elementi a partire dall'indice specificato:



**Figura 4.1.13:** *Context Help del blocco Replace Array Subset*

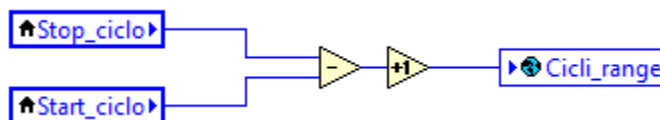
Pertanto, i diversi vettori contenenti un numero di cicli variabile vengono “montati” uno di seguito all'altro, fino al raggiungimento del numero di cicli desiderato. Per poter fare questo, come detto è necessario un vettore di base da cui partire. Bisogna allora inizializzare il registro a scorrimento con un vettore vuoto, avente

però almeno un numero di elementi pari al range di cicli selezionato (Figura 4.1.14):



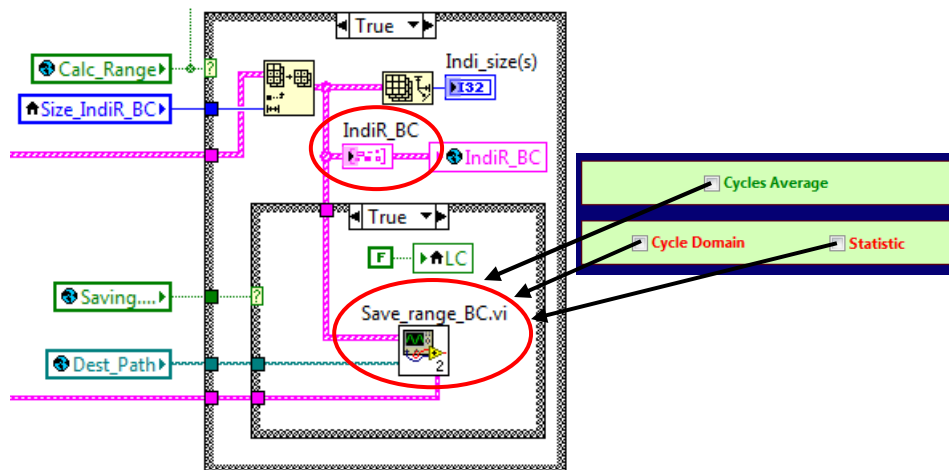
**Figura 4.1.14:** *Inizializzazione Shift Register*

Il blocco *Initialize Array* in questo caso serve appunto per preparare un vettore vuoto, con un numero di elementi pari a (Figura 4.1.15):



**Figura 4.1.15:** *Determinazione range di cicli selezionato*

Le grandezze medie e il vettore contenente i valori delle grandezze in base ciclo per tutto il range selezionato possono infine essere salvate nell'ultimo frame della struttura *Flat sequence* esterna (**Frame 3**). Qui è possibile individuare una struttura *Case* comandata ancora una volta dalla variabile *Calc\_Range* (Figura 4.1.16):



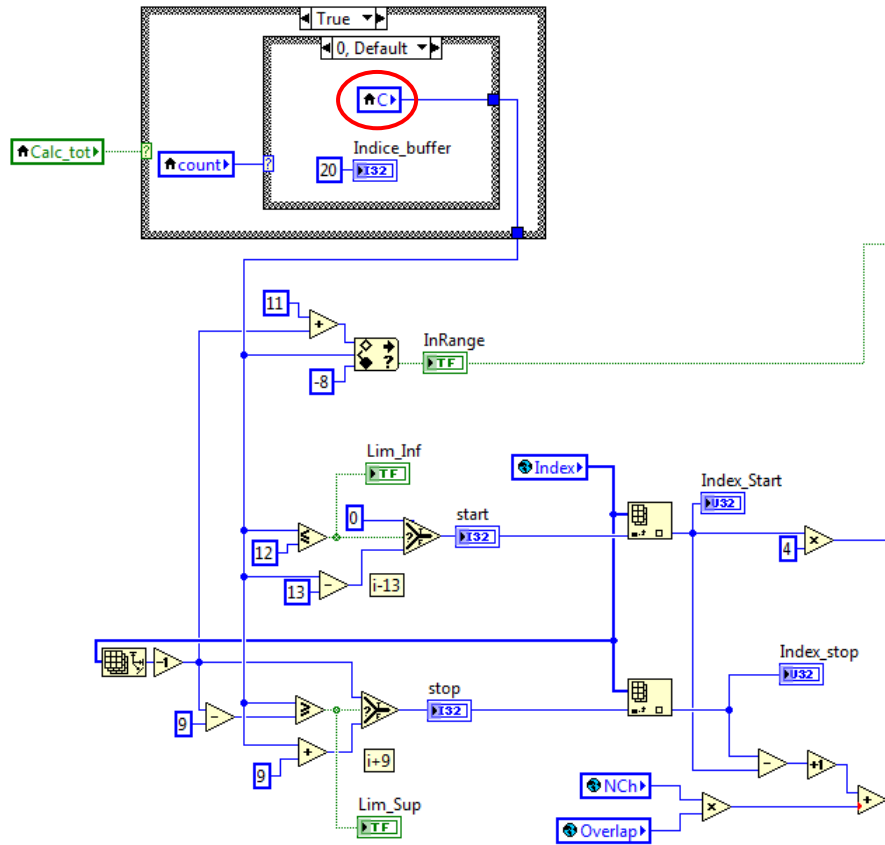
**Figura 4.1.16:** Salvataggio dei parametri raffigurati nel pannello e indicatore per le grandezze in base ciclo

Nel capitolo precedente si parlava della visualizzazione delle grandezze in base ciclo sul diagramma principale (paragrafo 3.4). Nella struttura *Case* di Figura 4.1.16 viene costruito proprio il vettore *IndiR\_BC*, richiamato poi con una variabile locale nella parte finale del Loop di visualizzazione.

Inoltre, è possibile notare che gli input del VI *Save\_range\_BC* sono le grandezze in base ciclo e le grandezze medie (il calcolo dei valori statistici viene effettuato direttamente all'interno del VI stesso).

## 4.2–Calcolo per un vettore di elementi centrato esattamente sul ciclo selezionato con il marker

Andando subito a vedere quello che succede nel **Frame 1** della struttura *Flat Sequence*, se *Calc\_Range=False*, è possibile notare che l'offset da cui iniziare a leggere il file grezzo di partenza per poter fare i calcoli e il numero di bytes da leggere cambiano in questo modo (Figura 4.2.1):



**Figura 4.2.1:** Nuova logica per la determinazione dell'offset di partenza e del numero di elementi da leggere il file

In particolare, compare una nuova variabile locale denominata *Calc\_tot*: questa variabile monitora lo stato di un vettore di 60 elementi, che viene costruito per velocizzare le operazioni di visualizzazione delle grandezze sullo strumento *Graph Angle*, quando si posiziona il marker in una determinata zona del diagramma principale. Infatti, quando il marker viene collocato sul ciclo di interesse, non vengono calcolate e visualizzate solo le grandezze riferite a quel ciclo, perché se poi l'utente desidera spostarsi al ciclo immediatamente successivo (o precedente) con le frecce disponibili sulla tastiera del PC, l'andamento ad esempio della pressione nel cilindro verrebbe mostrato con un certo ritardo, dovuto al fatto che le grandezze devono essere ricalcate per il nuovo ciclo. Quando si seleziona il ciclo di interesse sul diagramma principale, si è allora

deciso di non effettuare i calcoli solo per quel ciclo, ma di eseguirli per un vettore contenente 60 elementi, dove il ciclo di interesse è collocato esattamente in mezzeria. In questo modo, se l'utente si sposta anche solo di un ciclo a destra o a sinistra con le frecce della tastiera, non ci sarà nessun ritardo nella visualizzazione dei diversi andamenti sullo strumento *Graph Angle*. Tuttavia, bisogna monitorare la posizione in cui si trova il ciclo di interesse, perché se ci si sposta troppo verso gli estremi del vettore fino ad uscirne, è necessario calcolarne uno nuovo. La variabile *Calc\_tot* serve proprio a questo: se *Calc\_tot* è *True*, significa che si è fuori dal vettore di 60 elementi e quindi deve essere riempito un nuovo vettore sempre di 60 cicli, centrato in *C* (vedi Figura 4.2.1). Partono allora 3 iterazioni, in ciascuna delle quali viene calcolato un sottovettore da 20 cicli. La posizione in cui inserire i sottovettori all'interno del vettore di base è stabilita dall'*Indice\_buffer* che, in ordine, assume i seguenti valori:

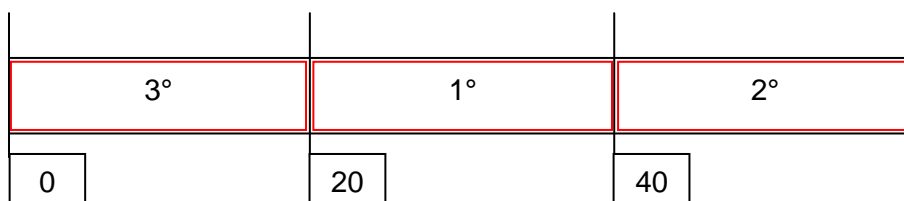
20→40→0

In pratica, la costruzione del vettore completo avviene in questo modo:

- 1) si parte da un vettore vuoto con 60 elementi:



- 2) si riempie questo vettore con 3 sottovettori da 20 elementi ciascuno, seguendo l'ordine raffigurato sotto:

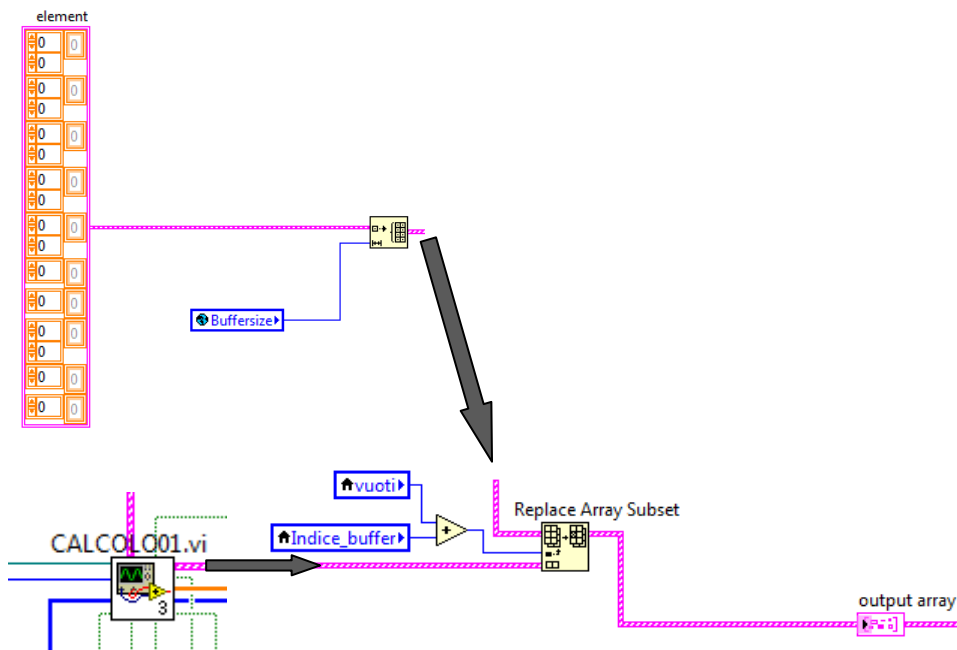




Per processare solo i 20 cicli di interesse, viene aperto il file grezzo in corrispondenza dell'indice  $C-13$  (che determina l'inizio del ciclo  $C-12$ ) fino all'indice  $C+9$  (che determina la fine del ciclo  $C+9$ ), in modo da avere 10 cicli prima e 9 dopo il ciclo  $C$  selezionato.

Siccome nel Loop di calcolo i primi due cicli considerati vengono eliminati, i cicli realmente a disposizione sono  $n-2$  (dove  $n = \text{size del vettore Index}-1$ ). Ciò significa che il 1° ciclo selezionato dall'utente, in realtà è il 3° ciclo contenuto nei dati. Facendo riferimento alla modalità di calcolo, è evidente che, se  $C \leq 12$ , non si hanno a disposizione i 10 cicli precedenti, per cui si rende necessario porre a zero l'indice di apertura del file. Analogamente per l'estremo superiore, per  $C \geq (\text{size del vettore Index}-1)-9$ , non si avranno a disposizione i 9 cicli successivi, per cui si pone come indice di chiusura del file l'ultimo indice del vettore *Index*.

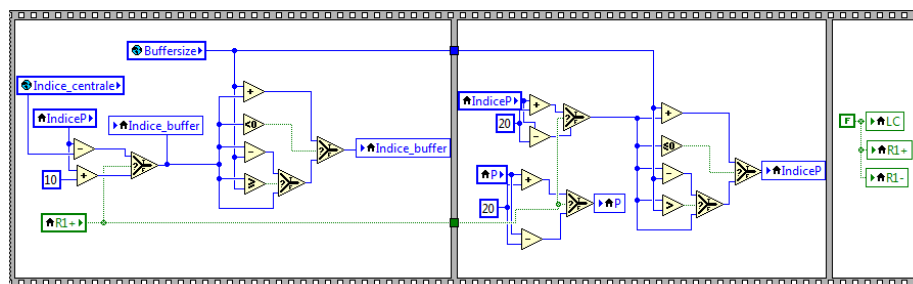
La costruzione del vettore completo si ottiene nel **Frame 2** della struttura *Flat Sequence* che si trova nel *Timed Loop* più interno. In questo caso infatti, se *Calc\_Range=False*, vengono svolte le seguenti operazioni (Figura 4.2.2):



**Figura 4.2.2:** Operazioni di Replace

Il vettore di base viene inizializzato nel *Timed Loop* più esterno e contiene un numero di elementi pari a *Buffersize* (di default impostato a 60). Con il blocco *Replace Array Subset* vengono sovrascritti gli elementi del vettore di base con quelli del vettore *Indi* e l'indice da cui partire varia seguendo *Indice\_buffer*. In uscita si ottiene allora il vettore *output array* già analizzato quando si parlava della visualizzazione delle grandezze in base angolo per il ciclo selezionato con il marker (vedi capitolo 3, paragrafo 3.3).

Nel **Frame 2** è anche presente il calcolo dei valori necessari per determinare il parametro *indice ciclo* (vedi sempre capitolo 3, paragrafo 3.3), come mostrato in Figura 4.2.3:



**Figura 4.2.3:** *Calcolo dei valori necessari per determinare il parametro indice ciclo*

e il calcolo della variabile locale *count* (=0,1,2) che controlla le 3 iterazioni nelle quali vengono valutati i sottovettori da 20 cicli e gestisce il valore dell'*Indice\_buffer* (Figura 4.2.4):

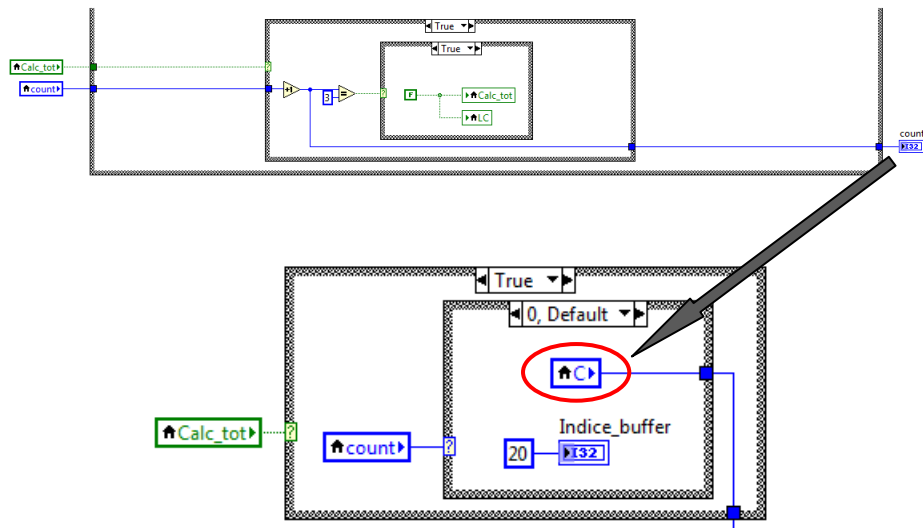


Figura 4.2.4: Calcolo della variabile locale count

### 4.3–Disattivazione Loop interno ed esterno

Il *Timed Loop* più interno termina l'esecuzione delle proprie operazioni quando l'uscita dell'*OR* logico raffigurato in Figura 4.3.1 è uguale a *True*:

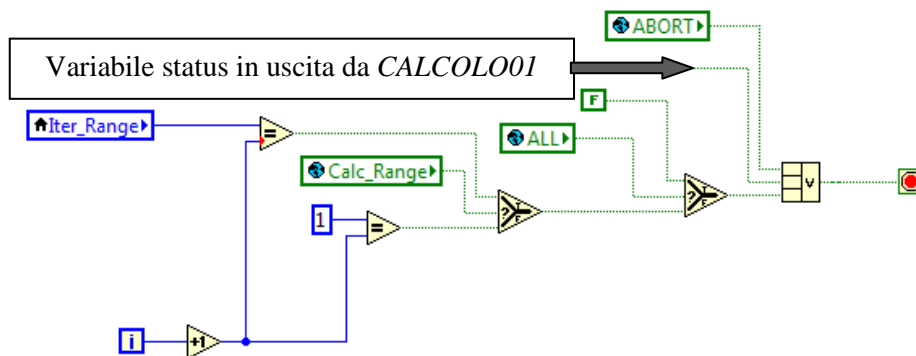
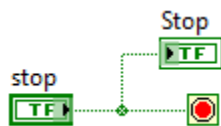


Figura 4.3.1: Terminale condizionale collegato ad un OR logico di diverse variabili

Invece, il controllo *stop* termina l'esecuzione del *Timed Loop* più esterno una volta selezionato il percorso *File* → *Close* (Figura 4.3.2):



**Figura 4.3.2:** *Terminale condizionale collegato alla variabile stop*

# CAPITOLO 5

## WHILE LOOP DI SALVATAGGIO

In questo capitolo viene riportata una descrizione del *While Loop* dedicato al salvataggio delle grandezze in base angolo (Figura 5.1):

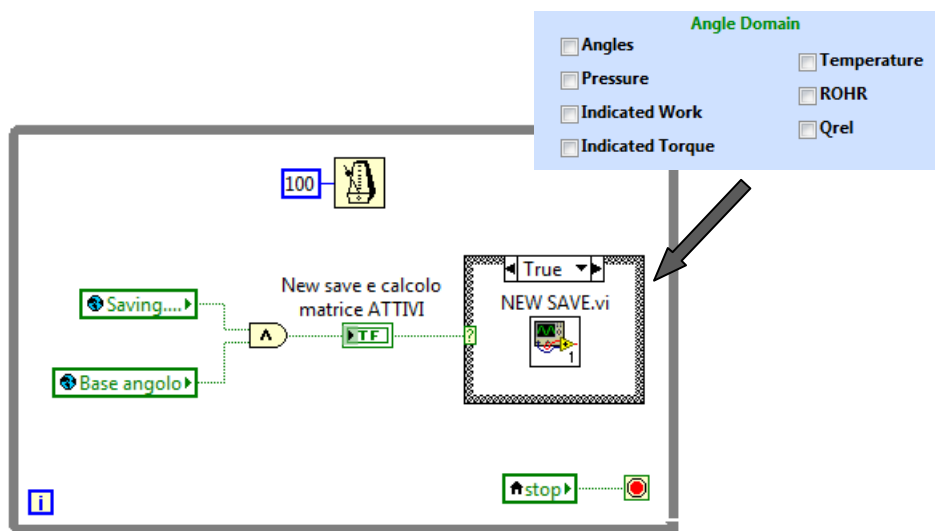


Figura 5.1: VI per il salvataggio Angle Domain

### 5.1– Vecchia logica di salvataggio

La vecchia logica di salvataggio (al momento disabilitata) era inclusa nel *Timed Loop* di calcolo ed era caratterizzata da diversi problemi relativi:

- alla tempistica di esecuzione;
- all'occupazione della RAM disponibile.

L'origine di questi problemi è stata individuata dopo un'analisi approfondita del codice che era alla base di questa logica: il VI, infatti, aspettava che venisse costruito tutto il vettore con le grandezze calcolate per un numero di cicli pari al *Cycle4Pack* e solo successivamente salvava questo vettore "in blocco". Andando a monitorare le prestazioni del PC, in ordine si osservava:

- 1) una crescita repentina della RAM utilizzata, dovuta alla costruzione in corso del vettore con un numero di cicli pari al *Cycle4Pack*;
- 2) una stabilizzazione della RAM ad un valore costante piuttosto elevato (scrittura su disco dei dati);
- 3) un decremento della RAM.

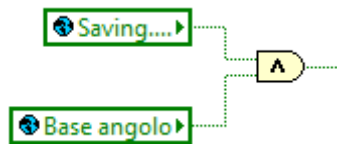
L'incremento del valore numerico della barra di progresso, che indicava l'andamento del salvataggio, proseguiva quindi a scatti. In particolare, era possibile notare una crescita del valore numerico della barra di progresso fino al raggiungimento del *Cycle4Pack*, poi una sosta di diversi minuti dovuta alla scrittura sul disco fisso e infine la barra ripartiva per salvare il vettore successivo.

Questo problemi erano tanto maggiori, quanto maggiore era:

- il *Cycle4Pack*;
- il numero di grandezze da salvare;
- il numero di cilindri per i quali si volevano salvare le grandezze;
- la risoluzione angolare adottata.

## 5.2–Nuova logica di salvataggio

La nuova logica di salvataggio si trova nel VI *NEW SAVE*, quindi è separata dal Loop di calcolo. La struttura *Case* di Figura 5.1 serve a garantirne l'attivazione solo quando l'AND logico tra le due variabili globali *Saving...* e *Base angolo* (che si attiva se anche solo una delle grandezze del pannello di Figura 5.1 viene spuntata) fornisce in uscita il valore booleano *True* (Figura 5.2.1):



**Figura 5.2.1:** AND logico

Questo procedimento è stato pensato per impedire l'attivazione del salvataggio quando:

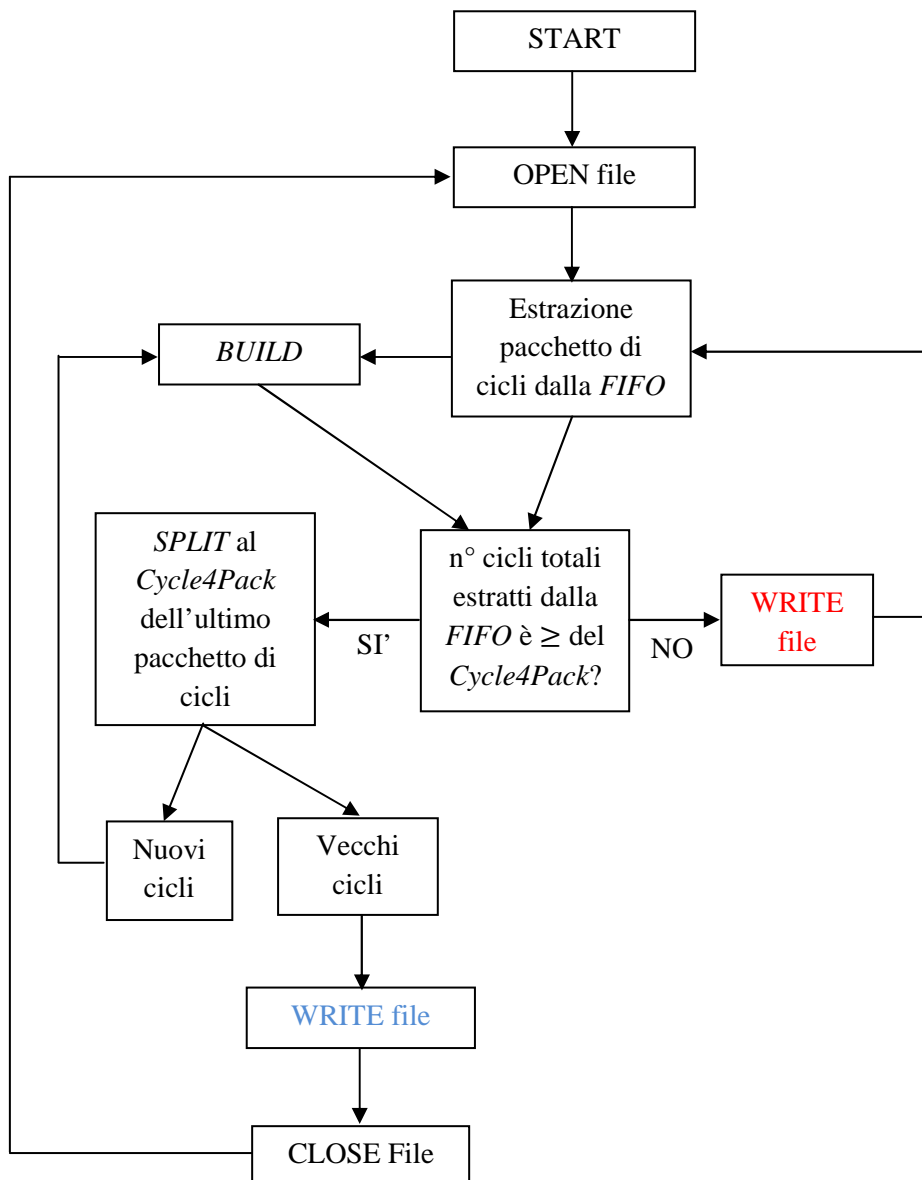
- 1) si vuole eseguire solo il calcolo e la visualizzazione delle grandezze:

$$\left. \begin{array}{l} \textit{Saving...=False} \\ \textit{Base angolo=False} \end{array} \right\} \rightarrow \textit{AND=False}$$

- 2) si vuole eseguire il salvataggio delle grandezze in base ciclo o delle grandezze medie:

$\left. \begin{array}{l} Saving...=True \\ Base\ angolo=False \end{array} \right\} \rightarrow AND=False$

La nuova logica di salvataggio funziona in maniera completamente diversa rispetto a quella scritta precedentemente nel software. Lo schema seguito per poterla costruire è riportato in Figura 5.2.2:



**Figura 5.2.2:** Schema a blocchi della nuova logica di salvataggio



Una volta avviato il salvataggio del range di cicli di interesse (START), viene aperto il file per la scrittura dei dati e si cominciano ad estrarre i pacchetti di cicli dalla *FIFO*, acronimo inglese di First In First Out (il primo elemento ad entrare è il primo elemento ad uscire), in questo caso impiegata come buffer di memoria fra il calcolo dei dati ed il loro salvataggio.

I dati appena scaricati vengono confrontati con il *Cycle4Pack*: se il *Cycle4Pack* è maggiore del numero di cicli totali estratti, si passa alla scrittura dei dati direttamente sul disco fisso (**WRITE file**).

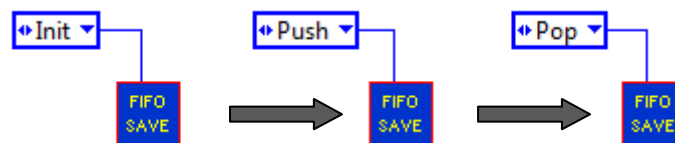
Si rimane all'interno di questo Loop finché il numero di cicli totali scaricati non diventa maggiore o uguale al *Cycle4Pack*: in questo caso, l'ultimo pacchetto di elementi viene "tagliato" (*SPLIT*) con una logica opportuna e i vecchi dati vengono scritti nel file attualmente aperto (**WRITE file**).

Tale file contiene ora un numero di cicli pari al *Cycle4Pack* e può quindi essere chiuso. Tuttavia, per continuare il salvataggio del range selezionato, è necessario aprire un nuovo file su cui poter scrivere i dati: gli elementi in ingresso saranno l'unione (*BUILD*) dei nuovi cicli e di quelli appena estratti dalla *FIFO*.

L'idea alla base è quindi quella di scrivere direttamente su disco i dati relativi ad un pacchetto di cicli, non appena questo viene calcolato. Di conseguenza, non si aspetta più di costruire tutto il vettore con un numero di cicli pari al *Cycle4Pack* per poi salvarlo in blocco, perché lo schema a blocchi di Figura 5.2.2 è stato inserito in un *Timed Loop* che permette di realizzare il *Cycle4Pack* iterazione dopo iterazione. I tempi di salvataggio si riducono notevolmente (dipende comunque dalla mole dei dati che si desidera salvare) e la RAM viene occupata solo per il tempo di stazionamento dei dati nella *FIFO*.

È stato inserito un buffer di memoria tra il calcolo ed il salvataggio, perché si vuole evitare la perdita di dati. Questa opzione è necessaria, perché la logica di salvataggio risulta comunque più lenta rispetto alla parte di calcolo affrontata nel capitolo precedente, soprattutto se si vuole salvare una mole notevole di dati. Si ha quindi a che fare con due processi in comunicazione, ma che lavorano a velocità differenti. Per questo motivo è stato necessario inserire un buffer di memoria (la *FIFO*) dove il VI di calcolo può scrivere i dati, continuando così a lavorare alla sua velocità, mentre il VI di salvataggio può leggere i dati senza interrompere il programma di calcolo.

In particolare, la *FIFO* utilizzata nel VI *NEW SAVE* è a sua volta un VI che può essere impostato in 3 modalità differenti (Figura 5.2.3):



**Figura 5.2.3:** Modalità del VI *FIFO SAVE*

Le 3 modalità di funzionamento devono essere utilizzate in sequenza:

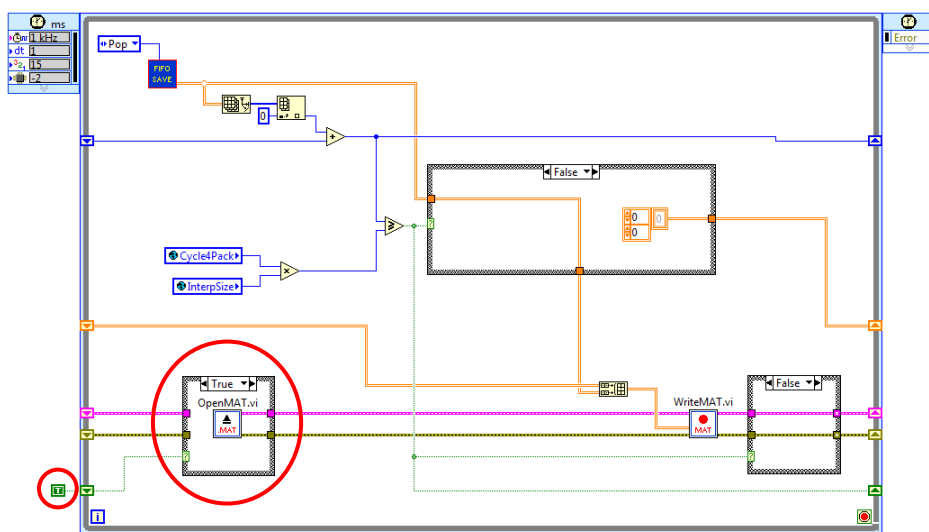
- 1) *Init* → inizializza il *size* della *FIFO*, ovvero imposta il numero di elementi che può contenere. In questo caso, la *FIFO* è stata inizializzata con un *size* pari ad 1000000 nello schema a blocchi del pop up *ConfigSave*: tale valore non è attualmente modificabile ed è un ordine di grandezza superiore rispetto al numero di elementi che si sono accumulati all'interno del buffer di memoria durante le prove effettuate;
- 2) *Push* → consente l'introduzione dei dati nella *FIFO*. L'introduzione dei dati avviene nel Loop di calcolo, in un VI denominato *Indicating\_02*. Per evitare che la *FIFO* si riempia

( $Full=True$ ) e che quindi i dati vengano persi, sempre in questo VI è stato predisposto un sistema di controllo che blocca il *Push* della *FIFO*, finché la parte di salvataggio non ha scaricato tutti i dati in essa contenuti;

- 3) *Pop* → consente l'estrazione dei dati dalla *FIFO*. Il numero di dati estratto è consigliabile essere sempre pari al *size* della *FIFO* stessa, in modo tale da svuotare completamente il buffer ad ogni iterazione del Loop di salvataggio.

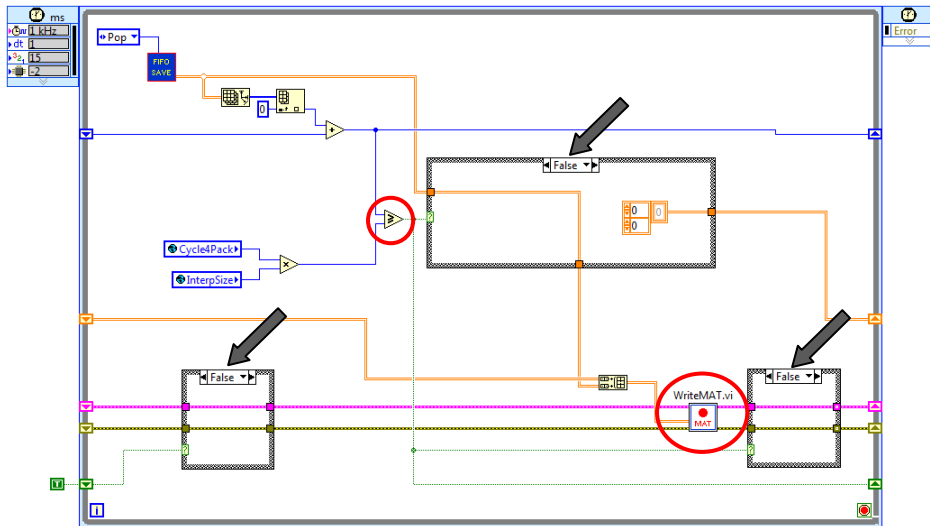
Di seguito viene riportato uno schema semplificato del funzionamento della nuova logica di salvataggio:

- 1) *START* → la struttura *Case* nella parte inferiore sinistra del *Timed Loop* (evidenziata in rosso) riceve in input il valore *True*, perché il registro di scorrimento ha come primo valore impostato proprio il booleano *True*. Di conseguenza, viene abilitato il VI per l'apertura del file su cui scrivere i dati (Figura 5.2.4):



**Figura 5.2.4:** Inizio salvataggio

- 2) I dati estratti dalla *FIFO* vengono scritti sul disco fisso. Le 3 strutture *Case* evidenziate dalle frecce in grigio sono sicuramente impostate a *False*, perché si è appena incominciato ad estrarre i dati, quindi la disuguaglianza evidenziata in Figura 5.2.5 fornisce in uscita il valore *False*:



**Figura 5.2.5:** Scrittura dati su disco

- 3) Quando la disuguaglianza fornisce in uscita il valore *True*, vuol dire che è stato superato il *Cycle4Pack*, quindi si attiva la logica di taglio (i vecchi elementi vengono scritti, mentre i nuovi entrano in uno *Shift Register* che li mette a disposizione della scrittura su disco per l'iterazione successiva), viene chiuso il file attuale e aperto un nuovo file (Figura 5.2.6):

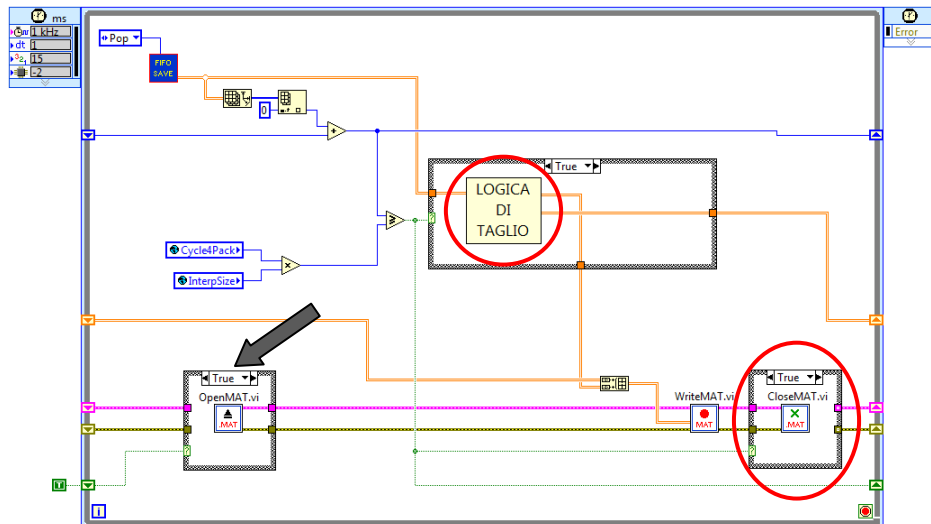


Figura 5.2.6: Operazioni eseguite nel caso in cui venga superato il Cycle4Pack

### 5.3– Validazione della nuova logica di salvataggio

Per validare la nuova logica di salvataggio è stato realizzato uno script Matlab (Figura 5.3.1), che illustra due file salvati con la vecchia e con la nuova logica. In particolare, sono stati considerati due andamenti della pressione nel cilindro:

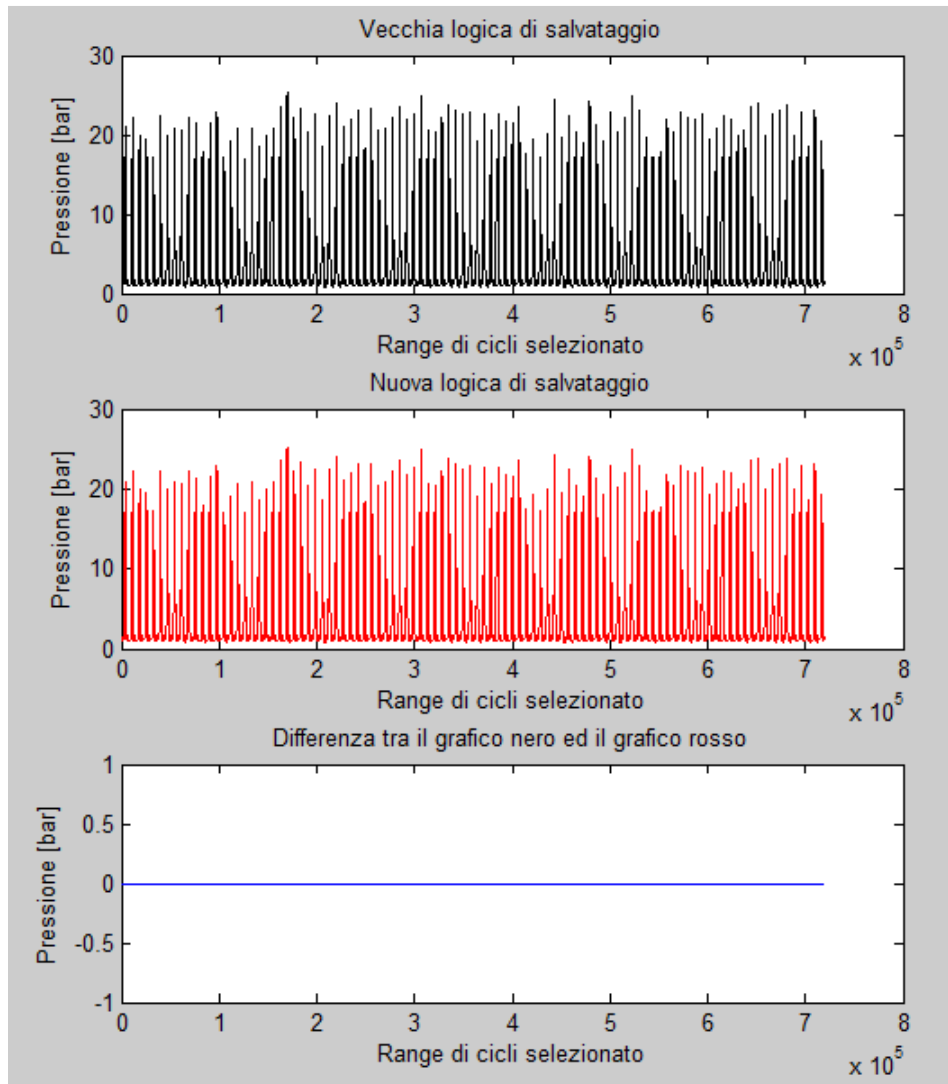
```

%caricare le matrici A e Angle_Domain

a=A(:,2); %tutte le righe, 2^ colonna della matrice A
a_d=Angle_Domain(2,:); %2^ riga, tutte le colonne della matrice Angle_Domain
subplot(221)
plot(a,'k')
title('Vecchia logica di salvataggio')
xlabel('Range di cicli selezionato')
ylabel('Pressione [bar]')
subplot(222)
plot(a_d,'r')
title('Nuova logica di salvataggio')
xlabel('Range di cicli selezionato')
ylabel('Pressione [bar]')

a_d_Trasp=a_d'; %vettore trasposto
c=a-a_d_Trasp; %differenza tra i due vettori
subplot(223)
plot(c)
title('Differenza tra il grafico nero ed il grafico rosso')
xlabel('Range di cicli selezionato')
ylabel('Pressione [bar]')

```



**Figura 5.3.1:** Script Matlab e grafici in output

Come è possibile vedere dall'andamento blu del grafico più in basso, la differenza tra i due andamenti di pressione è nulla. Questo significa che i dati salvati con la nuova logica sono esattamente gli stessi della vecchia logica.

Il nuovo procedimento scritto nel VI *NEW SAVE* quindi funziona correttamente e consente un risparmio di tempo fino all'80%.

# CAPITOLO 6

## TIMED LOOP DI GESTIONE DEGLI EVENTI

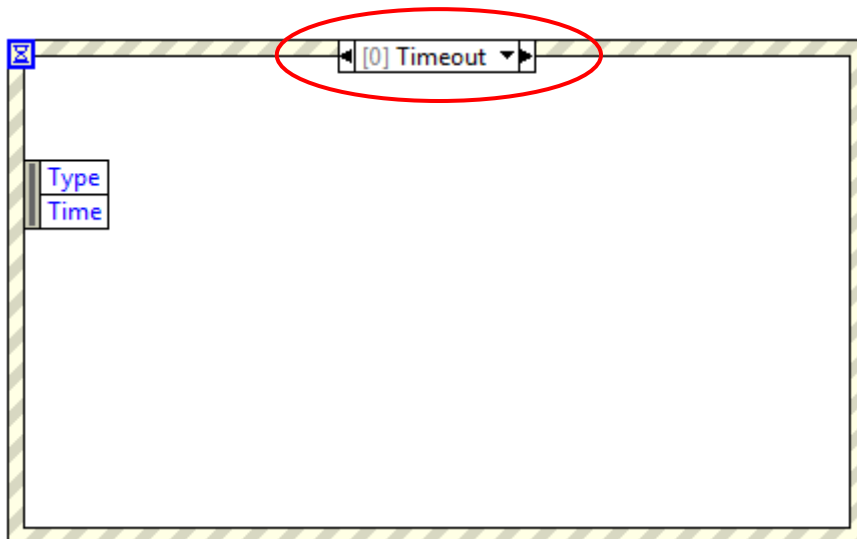
In questo capitolo viene riportata una descrizione della serie di operazioni che vengono eseguite nell'ultimo *Timed Loop*. Si tratta di una struttura ad elevata priorità, perché il parametro *Priority* è impostato a 100.

Dato il numero limitato di elementi presenti all'interno di quest'ultimo *Timed Loop*, non si è ritenuto necessario realizzare uno schema semplificato dello stesso. Sono infatti presenti due sole strutture:

- 1) una struttura *Event*;
- 2) una struttura *Flat Sequence*.

### 6.1–Struttura Event

La struttura *Event* (Figura 6.1.1) effettua le operazioni specificate al suo interno solo nel verificarsi dell'evento prestabilito da un'ampia lista (Figura 6.1.2) e indicato nel selettore superiore:



**Figura 6.1.1: Struttura Event**

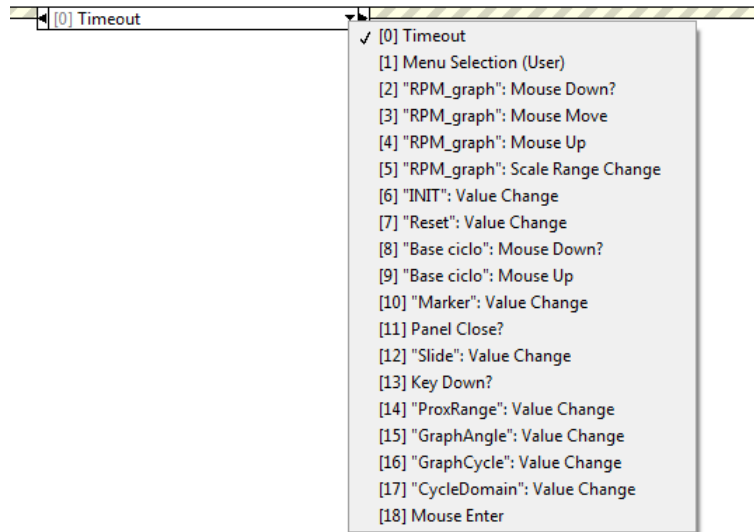
Event	Description
Drag Ended	Generated after a drag and drop operation completes. <a href="#">Details</a>
Drag Enter	Generated when there is a drag operation pending and the cursor enters the bounds of a control. <a href="#">Details</a>
Drag Leave	Generated when there is a drag operation pending and the mouse leaves a control that can accept a drag, or if a user cancels a drag and drop operation when hovering over a control. <a href="#">Details</a>
Drag Over	Generated when there is a drag and drop operation pending, as the mouse moves over a control. <a href="#">Details</a>
Drag Source Update	Generated when the mouse moves or a key state changes during a drag and drop operation. <a href="#">Details</a>
Drop	Generated when the user lifts the mouse when hovering over a control when there is a drag and drop operation pending. <a href="#">Details</a>
Key Down	Generated on a control that has keyboard focus. If a key press matches a keyboard shortcut in the VI menu, such as Ctrl-C or Ctrl-V, LabVIEW does not generate a Key Down event, regardless of whether the menu item is enabled. <a href="#">Details</a>
Key Down?	Generated on a control that has keyboard focus. If a key press matches a keyboard shortcut in the VI menu, such as Ctrl-C or Ctrl-V, LabVIEW does not generate a Key Down event, regardless of whether the menu item is enabled. <a href="#">Details</a>
Key Repeat	Generated at regular intervals when the user presses and holds a key in a front panel control. <a href="#">Details</a>
Key Repeat?	Generated when the user presses and holds a key in a front panel control. <a href="#">Details</a>
Key Up	Generated when the user releases a key on the keyboard in a specific control on the front panel. <a href="#">Details</a>
Mouse Down	Generated when you click the mouse button on a specific control. <a href="#">Details</a>
Mouse Down?	Generated when you click the mouse button on a specific control. <a href="#">Details</a>
Mouse Enter	Generated when the cursor enters the bounds of the front panel object. <a href="#">Details</a>
Mouse Leave	Generated when the cursor leaves the bounds of the front panel object. <a href="#">Details</a>
Mouse Move	Generated when you move the mouse over a control. <a href="#">Details</a>
Mouse Up	Generated when you release the mouse button on a specific control. LabVIEW does not generate this event if a shortcut menu appears when you click the mouse button. <a href="#">Details</a>
Shortcut Menu Activation?	Generated when the user right-clicks a control to display the shortcut menu. <a href="#">Details</a>
Shortcut Menu Selection (App)	Generated when the user selects an <a href="#">application item</a> from the shortcut menu of a control. Use the <a href="#">Shortcut Menu Selection (User)</a> event to generate an event when the user selects a user-defined menu item. This event is posted after the built-in shortcut menu item is processed by LabVIEW. <a href="#">Details</a>
Shortcut Menu Selection (User)	Generated when the user selects a <a href="#">user-defined</a> item from the shortcut menu. Use the <a href="#">Shortcut Menu Selection (App)</a> event to generate an event when the user selects an application item from the shortcut menu. <a href="#">Details</a>
Shortcut Menu Selection? (App)	Generated when the user selects an <a href="#">application item</a> from the shortcut menu. This event is posted before the application item is processed by LabVIEW. <a href="#">Details</a>
Value Change	Generated when the user changes the value of a control. You must <a href="#">read the terminal of a latched Boolean control</a> in its Value Change event case. <a href="#">Details</a>

**Figura 6.1.2: Lista di eventi disponibili**

Selezionato un evento, nel nodo *Event Data* che compare nella parte interna sinistra della cornice della struttura vengono visualizzate le operazioni disponibili.

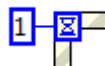
In questo caso, gli eventi disponibili sono 19 (Figura 6.1.3):





**Figura 6.1.3:** *Eventi presenti nella struttura Event di HeatITOff*

Il primo evento viene denominato *Timeout* e consiste nell'interruzione delle operazioni durante l'attesa della notifica di un evento. Il valore collegato al terminale nell'angolo in alto a sinistra (Figura 6.1.4) specifica il numero di millisecondi che la struttura deve attendere prima del time out (in questo caso, 1 ms):



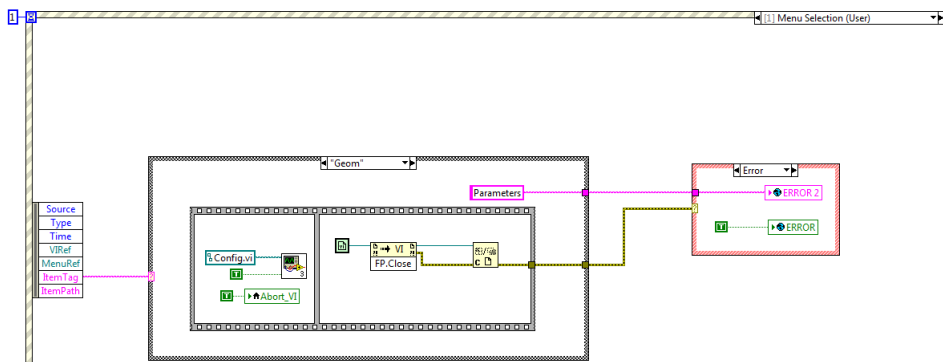
**Figura 6.1.4:** *Terminale Timeout*

Il secondo evento viene denominato *Menu Selection (User)* ed è interfacciato direttamente con la barra dei menu inserita nella parte superiore dell'interfaccia principale di HeatITOff. I comandi presenti in questa barra sono:

- *File* → *Open*  
           *Advanced*  
           *Close*

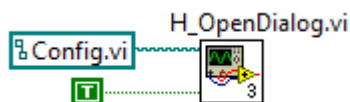
- *Configuration* → *Parameters*  
*Save Configuration*  
*Load Configuration*  
*Header*
- *Save* → *Multisave*  
*Configuration File*
- *Info*

Se si entra più nel dettaglio delle operazioni eseguite quando capita questo evento, è possibile vedere che ognuna delle voci sopraelencate è stata identificata con un *ItemTag* (Figura 6.1.5):



**Figura 6.1.5:** *ItemTag* riferito alla voce *Parameters*

Quando l'utente seleziona una delle voci riportate nella barra dei menu, viene attivato il VI *H\_OpenDialog* (Figura 6.1.6):



**Figura 6.1.6:** *Input VI H\_OpenDialog*

Tale VI apre a sua volta il VI riportato nel *path* che ha in ingresso, attivando il pannello frontale e mandandolo in esecuzione. Per capire quale dei VI aprire, l'applicazione fa appunto riferimento all'*ItemTag* della voce su cui l'utente ha cliccato.

Gli eventi:

- *Mouse Down?*;
- *Mouse Move*;
- *Mouse Up*;
- *Mouse Enter*;
- *Scale Range Change*.

eseguono alcune operazioni sul grafico degli RPM, mentre l'evento *Value Change* viene attivato quando uno dei seguenti controlli cambia valore, andando ad aprire lo strumento corrispondente (Figura 6.1.7):

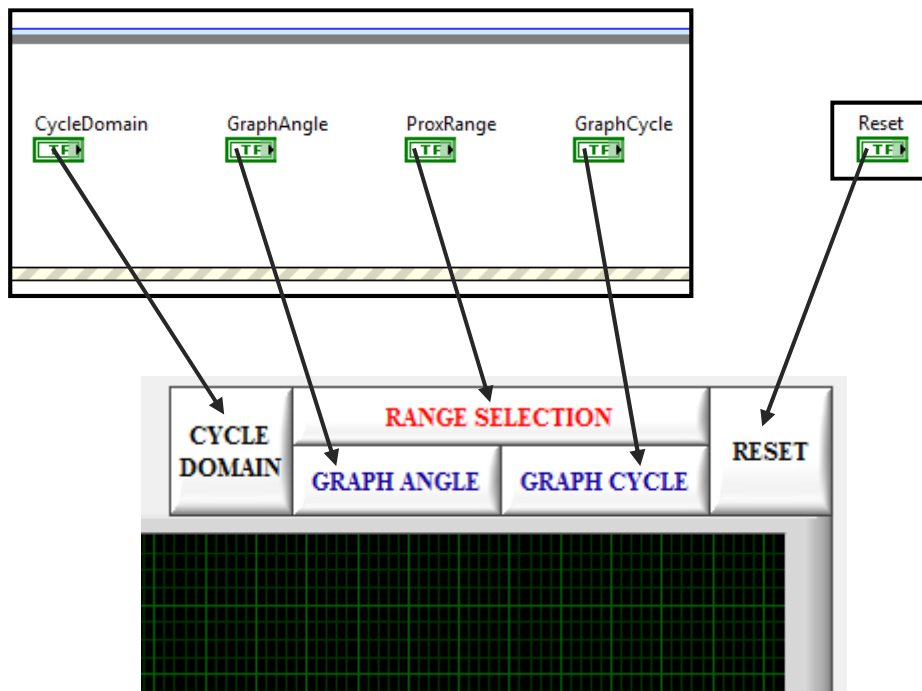
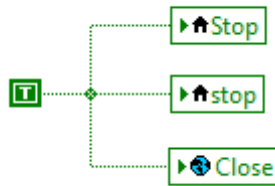


Figura 6.1.7: Pulsanti per l'apertura degli strumenti descritti nel manuale utente

Infine, l'evento *Panel Close?* imposta a *True* tutte le variabili collegate ai terminali condizionali dei Loop che eseguono le operazioni per il funzionamento di HeatITOff, terminando la loro esecuzione, e chiude l'interfaccia principale del software (Figura 6.1.8):



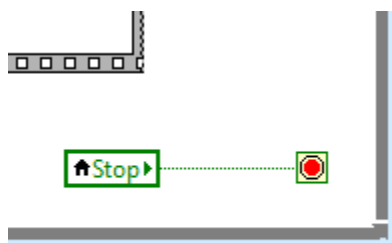
**Figura 6.1.8:** *Disattivazione di tutti i Loop (prime due variabili) e chiusura interfaccia principale (terza variabile)*

## 6.2–Struttura Flat Sequence e disattivazione Loop di gestione degli eventi

La struttura *Flat Sequence* contiene i passaggi necessari per permettere di evidenziare in blu, nel diagramma principale, i cicli selezionati con lo strumento *Range Selection*, quando:

- si attiva l'opzione *Range* e si clicca con il tasto sinistro del mouse sul primo ciclo di interesse nel grafico degli RPM, trascinando il cursore fino all'ultimo ciclo di interesse;
- si attiva l'opzione *Range* e si clicca sul pulsante *Start range*, digitando il primo ciclo di interesse nel controllo *Start Cycle*; si prosegue, cliccando sul pulsante *Stop range* e digitando l'ultimo ciclo di interesse nel controllo *Stop Cycle*;
- si clicca sul pulsante *ALL*.

In conclusione, la variabile locale *Stop* termina l'esecuzione del *Timed Loop* di gestione degli eventi una volta selezionato il percorso *File* → *Close* (Figura 6.2.1):



**Figura 6.2.1:** Terminale condizionale collegato alla variabile *Stop*



# CONCLUSIONI

Come accennato nell'introduzione, l'azienda Alma Automotive ha sviluppato uno strumento che permette il calcolo delle grandezze indicate, sulla base dei dati grezzi acquisiti e organizzati dal sistema real-time. In questo modo è possibile svolgere uno studio approfondito delle prestazioni di un motore a combustione interna.

Durante lo studio del software sono state apportate diverse modifiche per incrementare le velocità di calcolo e salvataggio. In particolare, per quanto riguarda la velocità di calcolo ci si è concentrati soprattutto nell'alleggerimento dell'intero software, mediante il raggruppamento di porzioni di codice in VI che non devono essere richiamati più volte, perché non sono collocati all'interno di Loop che devono svolgere un certo numero di iterazioni. Inoltre, sono state rimosse le linee di codice scritte nei *MathScript Node* e corretti alcuni errori riguardanti la stima della coppia indicata e il calcolo del volume occupato dal fluido in funzione dell'angolo di manovella.

La logica di salvataggio è stata invece completamente rivista: l'introduzione di un VI in grado di salvare il dato su disco immediatamente dopo il calcolo, ha permesso un guadagno di tempo dell'80% rispetto alla precedente versione. In più, grazie all'utilizzo di una *FIFO* è possibile evitare la perdita di dati nel caso più critico in cui si vogliono salvare tutte le grandezze con la massima risoluzione angolare impostabile.

Attualmente, il sistema si presenta piuttosto rapido e funzionale.

Tuttavia, lascia ancora spazio a diversi miglioramenti, fra i quali i più importanti sono:

- 1) il salvataggio dello streaming dei dati mediante l'utilizzo delle librerie *MATIO*. L'attuale conversione in Matlab, infatti, è piuttosto inefficiente perché consente di salvare solo una matrice e nessuna stringa;
- 2) l'eliminazione del VI per l'interpolazione in base angolo, inserito prima dello svolgimento dei calcoli indicating, ed il suo utilizzo in base alle necessità. Questo serve per migliorare la precisione del calcolo della singola grandezza (non c'è più l'approssimazione dovuta proprio all'interpolazione) e perché il flusso di dati su base tempo può servire per altri tipi di analisi, come il calcolo dello spettro di un segnale;
- 3) la conversione dei file in altri formati, quale *ifiles*, per consentire l'analisi con altri strumenti software.

Una volta aggiunte anche queste ultime modifiche, HeatITOff potrà essere uno strumento utile per l'ottimizzazione del processo di combustione e la ricerca delle massime prestazioni di un motore a combustione interna, nel rispetto comunque dei limiti sempre più restringenti sulle emissioni inquinanti.



## BIBLIOGRAFIA

- G. Ferrari, *Motori a combustione interna*, Edizioni Il Capitello, Torino
- G. Minelli, *Motori endotermici alternativi*, Pitagora Editrice, Bologna
- L. Solieri, *Sviluppo di algoritmi avanzati di analisi e diagnosi combustione in tempo reale per motori endotermici alternativi*, Tesi di dottorato, Università di Bologna
- L. Fiumana, *Sviluppo di un sistema per l'analisi indicating offline nei motori a combustione interna*, Tesi di laurea, Università di Bologna
- LabVIEW User Manual
- E. Corti, *Dispense Macchine L*
- E. Corti, *Dispense Controllo dei motori a combustione interna LM*
- D. Moro, *Dispense Motori a combustione interna LM*



# INDICE

<b>INTRODUZIONE</b> .....	pag. 3
<b>CAPITOLO 1 – Manuale utente</b>	
<b>1.1</b> – Interfaccia principale .....	pag. 5
<b>1.2</b> – Menu Save .....	pag. 19
<b>1.3</b> – Menu File .....	pag. 23
<b>1.4</b> – Menu Configuration .....	pag. 24
<b>1.5</b> – Menu Info .....	pag. 34
<b>CAPITOLO 2 – Schema generale del codice sorgente</b>	
<b>2.1</b> – Labview .....	pag. 35
<b>2.2</b> – Organizzazione del progetto HeatITOff .....	pag. 37
<b>CAPITOLO 3 – Timed Loop di visualizzazione</b>	
<b>3.1</b> – Caricamento o salvataggio della configurazione del file grezzo .....	pag. 50
<b>3.2</b> – Visualizzazione dell’anteprima giri motore in base ciclo .....	pag. 53
<b>3.3</b> – Visualizzazione delle grandezze in base angolo sullo strumento Graph Angle .....	pag. 55
<b>3.4</b> – Visualizzazione delle grandezze in base ciclo sul diagramma principale .....	pag. 60
<b>3.5</b> – Attivazione Loop di calcolo, disattivazione Loop di visualizzazione .....	pag. 62
<b>CAPITOLO 4 – Timed Loop di calcolo</b>	
<b>4.1</b> – Calcolo per un numero di cicli compreso tra Start Cycle e Stop Cycle .....	pag. 66

4.2– Calcolo per un vettore di elementi centrato esattamente sul ciclo selezionato con il marker .....	pag. 78
4.3– Disattivazione Loop interno ed esterno.....	pag. 83
<b>CAPITOLO 5 – While Loop di salvataggio</b>	
5.1– Vecchia logica di salvataggio .....	pag. 85
5.2– Nuova logica di salvataggio .....	pag. 87
5.3– Validazione della nuova logica di salvataggio .....	pag. 93
<b>CAPITOLO 6 – Timed Loop di gestione degli eventi</b>	
6.1– Struttura Event .....	pag. 95
6.2– Struttura Flat Sequence e disattivazione Loop di gestione degli eventi .....	pag. 100
<b>CONCLUSIONI</b> .....	pag. 103
<b>BIBLIOGRAFIA</b> .....	pag. 105