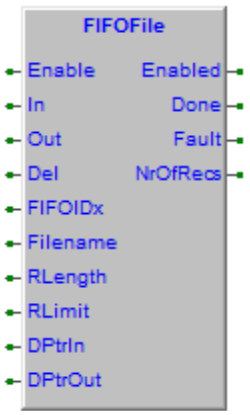
1.1.3 FIFOFile, gestisce registro FIFO su file

Type	Library
FB	eLogLib_B100



Questo blocco funzione effettua la gestione di un registro FIFO su di un file (***Può essere eseguito in un solo loop di programma come una funzione***). Attivando ***In*** i dati nel buffer puntato da ***DptrIn*** sono inseriti nel registro FIFO ed il valore di ***NrOfRecs*** verrà incrementato.

Attivando l'ingresso ***Out*** il primo record inserito nel registro viene trasferito nel buffer puntato da ***DptrOut***. L'ingresso ***Del*** permette di eliminare i records secondo l'ordine di inserimento, attivando l'ingresso il record viene eliminato ed il valore di ***NrOfRecs*** viene decrementato.

Raggiunto il numero di records indicato da ***RLimit*** verranno sovrascritti i records più datati presenti nel registro FIFO.

In ***FIFOIDx*** occorre fornire l'indirizzo di un array di 2 UDINT che devono essere allocati dal programma. Se si desidera che la situazione dei dati nel FIFO sia mantenuta allo spegnimento del sistema occorre allocare questo array in una memoria tampone.

L'uscita ***Done*** si attiva per un loop di programma ad ogni esecuzione di un comando se l'esecuzione ha esito positivo. In caso di errore esecuzione viene attivata per un loop l'uscita ***Fault***.

Enable (BOOL)	Comando di abilitazione blocco funzione.
In (BOOL)	Comando di inserimento record nel registro FIFO.
Out (BOOL)	Comando di lettura record dal registro FIFO.
Del (BOOL)	Comando di cancellazione record dal registro FIFO.
FIFOIDx (@UDINT)	Pointer alla variabile usata come indice di gestione FIFO.
Filename (STRING[32])	Percorso e nome del file in cui fare log (es.: 'SDCard/MyFile.txt').
RLength (UDINT)	Lunghezza del record da gestire con il registro FIFO.
RLimit (UDINT)	Numero di records unici nel registro FIFO. Superato il valore i records sono sovrascritti.
DptrIn (@USINT)	Pointer ai dati da inserire nel registro FIFO.
DptrOut (@USINT)	Pointer ai dati letti dal registro FIFO.
Enabled (BOOL)	Attivo su abilitazione blocco funzione.
Done (BOOL)	Attivo per un loop se la scrittura nel file è riuscita.
Fault (BOOL)	Attivo per un loop su errore esecuzione del comando.
NrOfRecs (UDINT)	Numero di records inseriti nel registro FIFO.

Codici di errore

In caso di errore si attiva l'uscita ***Fault*** e con ***SysGetLastError*** è possibile rilevare il codice di errore.

10055100	FB usata in task fast o slow.
1005520 (0~9)	Errori nell'inserimento del record nel registro.
1005530 (0~9)	Errori nella lettura del record dal registro.

Esempi

Nel seguente esempio sul fronte di attivazione dell'input ***Di00CPU*** viene inserito un record nel registro FIFO. Ad ogni attivazione dell'ingresso Di01CPU il record viene estratto ed eliminato dal registro FIFO.

Definizione variabili

	Name	Type	Address	Array	Init value	Attribute	Description
1	FIFO	FIFOFile	Auto	No		..	FIFO file FB instance
2	j	INT	Auto	No		..	Auxiliary variable
3	DataIn	RECORDDATASTRUCT	Auto	No		..	FIFO input data
4	DateTime	SysETimeToDate	Auto	No		..	Actual Date/Time
5	Year	USINT	Auto	No		..	Year value (00-99)
6	Pulse	BOOL	Auto	[0..1]		..	Pulse flag
7	DataOut	RECORDDATASTRUCT	Auto	No		..	FIFO output data

Esempio ST

```
   (* Execute the program initialization. *)

  IF (SysFirstLoop) THEN
    FIFO.Filename:='Storage/FIFO.bin'; (* Filename *)
    FIFO.RLength:=SIZEOF(DataIn); (* Record length *)
    FIFO.DPtrIn:=ADR(DataIn); (* Data pointer (In) *)
    FIFO.DPtrOut:=ADR(DataOut); (* Data pointer (Out) *)
    FIFO.RLimit:=50000/32; (* Records limit *)
  END_IF;

  (* Calculate the actual Date/Time and manage the FIFO. *)

  DateTime(EpochTime:=SysDateTime); (* Actual Date/Time *)
  Year:=TO_USINT(DateTime.Year-2000); (* Year value (00-99) *)
  FIFO(Enable:=TRUE); (* FIFO file FB instance *)
  FIFO.FIFOIDx:=ADR(FIFOIDx); (* FIFO indexes *)
  FIFO.In:=FALSE; (* In command *)
  FIFO.Out:=FALSE; (* Out command *)
  FIFO.Del:=FALSE; (* Delete command *)

  (* Manage the data record and insert it on FIFO register. *)
  (* The Date/Time is in the format "14/05/15 12.24.04". *)

  IF (Di00CPU <> Pulse[0]) THEN
    Pulse[0]:=Di00CPU; (* Pulse flag *)

    IF (Pulse[0]) THEN
      j:=SysVarsnprintf(ADR(DataIn.DateTime), 3+1, '%02d/', USINT_TYPE, ADR(DateTime.Day));
      j:=SysVarsnprintf(ADR(DataIn.DateTime)+LEN(DataIn.DateTime), 3+1, '%02d/', USINT_TYPE, ADR(DateTime.Month));
      j:=SysVarsnprintf(ADR(DataIn.DateTime)+LEN(DataIn.DateTime), 3+1, '%02d ', USINT_TYPE, ADR(Year));

      j:=SysVarsnprintf(ADR(DataIn.DateTime)+LEN(DataIn.DateTime), 3+1, '%02d.', USINT_TYPE, ADR(DateTime.Hour));
      j:=SysVarsnprintf(ADR(DataIn.DateTime)+LEN(DataIn.DateTime), 3+1, '%02d.', USINT_TYPE, ADR(DateTime.Minute));
      j:=SysVarsnprintf(ADR(DataIn.DateTime)+LEN(DataIn.DateTime), 2+1, '%02d', USINT_TYPE, ADR(DateTime.Second));

      DataIn.Voltage[0]:=DataIn.Voltage[0]+1.0;
      DataIn.Voltage[1]:=DataIn.Voltage[1]+2.0;
      FIFO.In:=TRUE; (* In command *)
    END_IF;
  END_IF;

  (* Extact and delete record from FIFO. *)

  IF (Di01CPU <> Pulse[1]) THEN
    Pulse[1]:=Di01CPU; (* Pulse flag *)

    IF (Pulse[1]) THEN
      FIFO.Out:=TRUE; (* Out command *)
      FIFO.Del:=TRUE; (* Delete command *)
    END_IF;
  END_IF;

(* [End of file] *)
```