Guida NIRS Box DLL

- **NIRS_OPEN: apre la comunicazione con la NIRS Box.** Deve essere eseguita per prima. Richiede come parametro un puntatore a uint32, un puntatore ad un array composto da 128 uint32 e la lunghezza di tale array in formato uint32 che viene passato alla funzione come valore. Restituisce in uscita un indicatore di stato, uguale a 1 se OK, altrimenti uguale a 0.

  PROTOTIPO

  ```
  uint32 NIRS_OPEN(uint32 *Handle, uint32 Registers_Out[], int32
  Registers_Out_Length);
  ```

  ESEMPIO MATLAB

  status_open = calllib('NIRS_DLL', 'NIRS_OPEN', Handle_Ptr, Registers_Ptr, int32(128))

- **NIRS_ON: Accende i laser, il modulo SiPM, i controlli termici ed il clock che fornisce il SYNC.** Il clock parte a 40 MHz ed il tempo di integrazione a 1 secondo. Deve essere eseguita dopo la NIRS_OPEN. Richiede come parametro un puntatore a uint32 per l'Handle, un puntatore ad un array composto da 128 uint32 e la lunghezza di tale array in formato uint32 che viene passato alla funzione come valore (questi ultimi 2 parametri sono ripetuti due volte). Restituisce in uscita un indicatore di stato, uguale a 1 se OK, altrimenti uguale a 0. N.B.: Registers_In[] e Registers_Out[] richiedono lo stesso puntatore.

  PROTOTIPO

  ```
  uint32 NIRS_ON(uint32 *Handle, uint32 Registers_In[], uint32
  Registers_Out[], int32 Registers_In_Length, int32
  Registers_Out_Length);
  ```

  ESEMPIO MATLAB

  status_on = calllib('NIRS_DLL', 'NIRS_ON', Handle_Ptr, Registers_Ptr, Registers_Ptr, int32(128), int32(128))

- **NIRS_SET: Permette di variare frequenza del clock ed il tempo di integrazione.** Può essere eseguita in qualsiasi momento dopo la NIRS_ON. Richiede come parametro un puntatore a uint32 per l'Handle, un puntatore ad un array composto da 128 uint32 due volte, la lunghezza di tale array in formato uint32 che viene passato alla funzione come valore due volte, la frequenza in MHz desiderata in formato uint32 ed il tempo di integrazione in millisecondi in formato uint32. Restituisce in uscita un indicatore di stato, uguale a 1 se OK, altrimenti uguale a 0. N.B.: Registers_In[] e Registers_Out[] richiedono lo stesso puntatore.

  PROTOTIPO

  ```
  uint32 NIRS_SET(uint32 *Handle, uint32 Registers_In[], uint32
  Registers_Out[], int32 Registers_In_Length, int32
  Registers_Out_Length, uint32 Frequency, uint32 Time);
  ```

ESEMPIO MATLAB

status_set = calllib('NIRS_DLL', 'NIRS_SET', Handle_Ptr, Registers_Ptr, Registers_Ptr, int32(128), int32(128), int32(20), int32(2000))

- **NIRS_ACQ: Permette l'acquisizione dell'istogramma.** Deve essere eseguita in un processo di polling. Quando una misura è completa restituisce il valore 1 ed i dati della misura sono resi disponibili nelle relative variabili. E' quindi possibile leggere l'istogramma ed i dati dei contatori (SYNC counts, SiPM counts, TDC conversions) nell'intervallo di integrazione. Richiede come parametro un puntatore a uint32 per l'Handle, un puntatore ad un array composto da 8192 uint32, un puntatore ad un array composto da 3 uint32, le lunghezze di questi array (rispettivamente 8192 e 3) in formato uint32 che vengono passate alla funzione come valore ed un puntatore a uint32 per il banco di memoria letto (corrispondente al laser utilizzato, nella misura appena scaricata, ha valore 1 o 2). Per avere il numero di conteggi validi occorre sommare tutti i conteggi presenti nell'istogramma. Restituisce in uscita un indicatore di stato della misura, uguale a 1 se è completa, altrimenti uguale a 0. N.B.: Registers_In[] e Registers_Out[] richiedono lo stesso puntatore.

PROTOTIPO

```
uint32 NIRS_ACQ(uint32 *Handle, uint32 Histogram[], uint32
Stats[], int32 Histogram_Length, int32 Stats_Length, uint32
*Bank);
```

ESEMPIO MATLAB

meas_done = calllib('NIRS_DLL', 'NIRS_ACQ', Handle_Ptr, Histogram_Ptr, Stats_Ptr, int32(8192), int32(3), Mem_bank_Ptr);

- **NIRS_OFF: Spegne i laser, il modulo SiPM, i controlli termici ed il clock che fornisce il SYNC.** Deve essere eseguita prima della NIRS_CLOSE. Richiede come parametro un puntatore a uint32 per l'Handle, un puntatore ad un array composto da 128 uint32 due volte e la lunghezza di tale array in formato uint32 che viene passato alla funzione come valore due volte. Restituisce in uscita un indicatore di stato, uguale a 1 se OK, altrimenti uguale a 0. N.B.: Registers_In[] e Registers_Out[] richiedono lo stesso puntatore.

PROTOTIPO

```
uint32 NIRS_OFF(uint32 *Handle, uint32 Registers_In[], uint32
Registers_Out[], int32 Registers_In_Length, int32
Registers_Out_Length);
```

ESEMPIO MATLAB

status_off = calllib('NIRS_DLL', 'NIRS_OFF', Handle_Ptr, Registers_Ptr, Registers_Ptr, int32(128), int32(128))

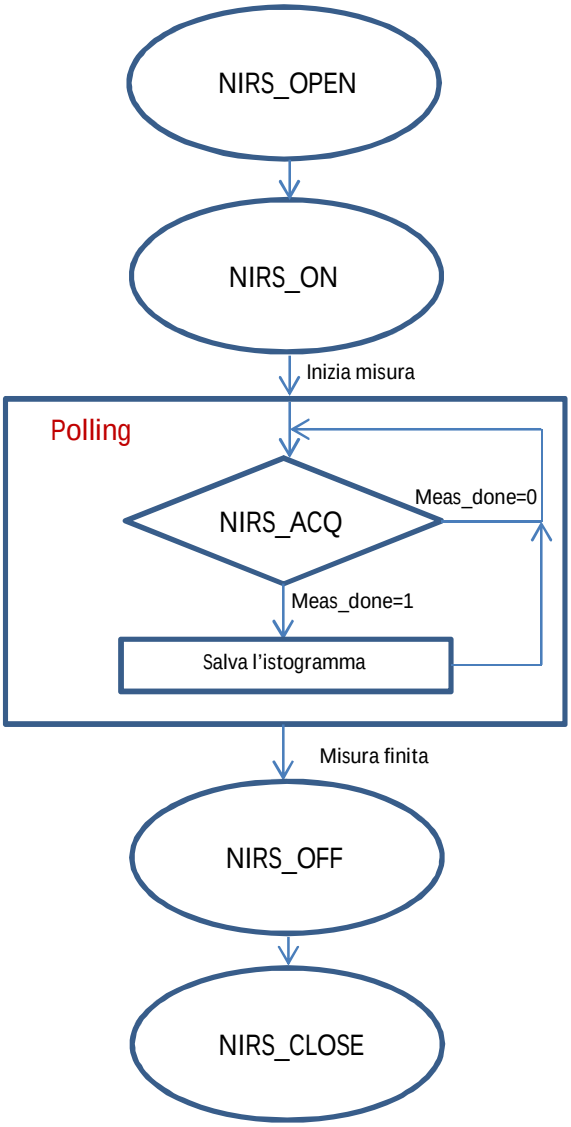
- **NIRS_CLOSE: chiude la comunicazione con la NIRS Box.** Deve essere eseguita per ultima. Richiede come parametro un puntatore a uint32 per l'Handle. Restituisce in uscita un indicatore di stato, uguale a 1 se OK, altrimenti uguale a 0.

  PROTOTIPO

  ```
  uint32 NIRS_CLOSE(uint32 *Handle);
  ```

  ESEMPIO MATLAB

  status_close = calllib('NIRS_DLL', 'NIRS_CLOSE', Handle_Ptr)



Utilizzando le dll con un polling a 100 ms è possibile ridurre l'integration time fino a 300 ms garantendo misure real-time.