Zynq-7000 Driver Pack

Generato da Doxygen 1.8.8

Sab 20 Mag 2017 00:00:30

Indice

Capitolo 1

Indice dei moduli

1.1 Moduli

Questo è l'elenco di tutti i moduli:

Capitolo 2

Indice delle strutture dati

2.1 Strutture dati

Queste sono le strutture dati con una loro breve descrizione:

Capitolo 3

Indice dei file

3.1 Elenco dei file

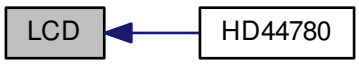
Questo è un elenco di tutti i file con una loro breve descrizione:

Capitolo 4

Documentazione dei moduli

4.1 LCD

Diagramma di collaborazione per LCD:



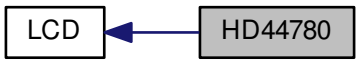
Moduli

- HD44780

4.1.1 Descrizione dettagliata

4.2 HD44780

Diagramma di collaborazione per HD44780:



Strutture dati

- struct HD44780_LCD_t

  Struttura opaca che astrae un device Display LCD con cntroller Hitachi HD44780, o compatibile. Un oggetto di tipo HD44780_LCD_t rappresenta un device lcd HD44780. Il modulo e' pensato per permettere la gestione di piu' display da parte dello stesso processore, agendo su oggetti HD44780_LCD_t diversi. Il modulo permette di utilizzare sia l'interfacciamento ad otto bit che quello a quattro bit, inizializzando il device opportunamente, attraverso l'uso delle funzioni HD44780_Init8 eHD44780_Init4. Il modulo fornisce anche semplici funzioni per la stampa di un carattere o di una stringa null-terminated di caratteri. Si veda la documentazione delle funzioni HD44780_Printc e HD44780_Print. Inoltre sono presenti diverse funzioni di utilita' generica, come quelle per la pulizia del display, per lo spostamento del cursore di un posto in avanti o indietro, alla riga in basso o in alto.

Tipi enumerati (enum)

- enum HD44780_InterfaceMode_t { HD44780_INTERFACE_4bit, HD44780_INTERFACE_8bit }
- enum HD44780_Direction_t { HD44780_CursorLeft, HD44780_CursorRight }

  Direzioni di spostamento del cursore.

Funzioni

- void HD44780_Init8 (HD44780_LCD_t *lcd, GPIO_t *gpio, GPIO_mask RS, GPIO_mask RW, GPIO_mask E, GPIO_mask Data7, GPIO_mask Data6, GPIO_mask Data5, GPIO_mask Data4, GPIO_mask Data3, GPIO_mask Data2, GPIO_mask Data1, GPIO_mask Data0)

  Inizializza un display lcd HD44780 con interfacciamento ad 8 bit.
- void HD44780_Init4 (HD44780_LCD_t *lcd, GPIO_t *gpio, GPIO_mask RS, GPIO_mask RW, GPIO_mask E, GPIO_mask Data7, GPIO_mask Data6, GPIO_mask Data5, GPIO_mask Data4)

  Inizializza un oggetto display lcd HD44780 affinche' si utilizzi l'interfaccia a 4 bit.
- void HD44780_Printc (HD44780_LCD_t *lcd, char c)

  Stampa un carattere.
- void HD44780_Print (HD44780_LCD_t *lcd, const char *s)

  Stampa una stringa null-terminated di caratteri.
- void HD44780_printBinary8 (HD44780_LCD_t *lcd, uint8_t b)

  Stampa un byte in binario. (bit piu' significativo a sinistra)
- void HD44780_printBinary32 (HD44780_LCD_t *lcd, uint32_t w)

  Stampa una word di 32 bit in binario. (bit piu' significativo a sinistra)
- void HD44780_printBinary64 (HD44780_LCD_t *lcd, uint64_t b)

  Stampa un blocco di 64 bit in binario. (bit piu' significativo a sinistra)
- void HD44780_printHex8 (HD44780_LCD_t *lcd, uint8_t b)

  Stampa un byte in esadecimale. (bit piu' significativo a sinistra)

- void HD44780_printHex32 (HD44780_LCD_t *lcd, uint32_t w)

  Stampa una word di 32 bit in esadecimale. (bit piu' significativo a sinistra)
- void HD44780_printHex64 (HD44780_LCD_t *lcd, uint64_t b)

  Stampa un blocco di 64 bit in esadecimale. (bit piu' significativo a sinistra)
- void HD44780_Clear (HD44780_LCD_t *lcd)

  Pulisce il display e sposta il cursore all'inizio della prima riga.
- void HD44780_Home (HD44780_LCD_t *lcd)

  Sposta il cursore all'inizio della prima riga.
- void HD44780_MoveToRow1 (HD44780_LCD_t *lcd)

  Sposta il cursore all'inizio della prima riga.
- void HD44780_MoveToRow2 (HD44780_LCD_t *lcd)

  Sposta il cursore all'inizio della seconda riga.
- void HD44780_MoveCursor (HD44780_LCD_t *lcd, HD44780_Direction_t dir)

  Sposta il cursore di una posizione a destra o sinistra.
- void HD44780_DisplayOff (HD44780_LCD_t *lcd)

  Disattiva il display.
- void HD44780_CursorOff (HD44780_LCD_t *lcd)

  Disattiva la visualizzazione del cursore.
- void HD44780_CursorOn (HD44780_LCD_t *lcd)

  Attiva la visualizzazione del cursore.
- void HD44780_CursorBlink (HD44780_LCD_t *lcd)

  Attiva il cursore lampeggiante.

4.2.1 Descrizione dettagliata

Un oggetto di tipo HD44780_LCD_t rappresenta un device lcd HD44780. Il modulo e' pensato per permettere la gestione di piu' display da parte dello stesso processore, agendo su oggetti HD44780_LCD_t diversi.
La struttura HD44780_LCD_t specifica quali siano i pin del microcontrollore che pilotano un determinato segnale del device. Ciascuno dei pin, cosi' come previsto dalla libreria STMCube, e' identificato attraverso una coppia porta-pin (ad esempio la coppia GPIOD-GPIO_PIN_9 si riferisce al pin 9 della porta D, quindi PD9). L'assegnazione segnale-coppia, quindi l' inizializzazione della struttura HD44780_LCD_t relativa ad un device lcd, DEVE essere effettuata tassativamente utilizzando le funzioni

- HD44780_Init4()
- HD44780_Init4_v2()
- HD44780_Init8()
- HD44780_Init8_v2()

le quali provvedono anche ad effettuare un test di connessione volto ad individuare eventuali segnali erroneamente associati.
Tali funzioni restituiscono un codice di errore, il quale puo' essere utilizzato per identificare la problematica sorta durante l'inizializzazione e provvedere alla sua gestione. Per i dettagli si rimanda alla documentazione delle specifiche funzioni.

Oltre alle funzioni di inizializzazione, il modulo fornisce anche funzioni basilari per la stampa su display lcd di

- caratteri, con la funzione HD44780_Printc()

- stringhe null-terminated di caratteri, con la funzione HD44780_Print()

Sono disponibili, inoltre, anche funzioni specifiche per inviare comandi al device:

- HD44780_Clear()
- HD44780_Home()
- HD44780_MoveToRow1()
- HD44780_MoveToRow2()
- HD44780_MoveCursor()
- HD44780_DisplayOff()
- HD44780_CursorOff()
- HD44780_CursorOn()
- HD44780_CursorBlink()

Per ulteriori dettagli si rimanda alla documentazione delle specifiche funzioni ed alla documentazione esterna che accompagna il modulo, reperibile nella cartella Doc.

4.2.2 Documentazione dei tipi enumerati

4.2.2.1 enum HD44780_Direction_t

Direzioni di spostamento del cursore.

Valori del tipo enumerato

HD44780_CursorLeft left sposta il cursore a sinistra

HD44780_CursorRight right sposta il cursore a destra

4.2.2.2 enum HD44780_InterfaceMode_t

Valori del tipo enumerato

HD44780_INTERFACE_4bit

HD44780_INTERFACE_8bit

4.2.3 Documentazione delle funzioni

4.2.3.1 void HD44780_Clear (HD44780_LCD_t * *lcd*)

Pulisce il display e sposta il cursore all'inizio della prima riga.

Parametri

lcd	display da pilotare;

4.2.3.2 void HD44780_CursorBlink (HD44780_LCD_t * *lcd*)

Attiva il cursore lampeggiante.

Parametri

lcd	display da pilotare;

4.2.3.3 void HD44780_CursorOff (HD44780_LCD_t * *lcd*)

Disattiva la visualizzazione del cursore.

Parametri

lcd	display da pilotare;

4.2.3.4 void HD44780_CursorOn (HD44780_LCD_t * *lcd*)

Attiva la visualizzazione del cursore.

Parametri

lcd	display da pilotare;

4.2.3.5 void HD44780_DisplayOff (HD44780_LCD_t * *lcd*)

Disattiva il display.

Parametri

lcd	display da pilotare;

4.2.3.6 void HD44780_Home (HD44780_LCD_t * *lcd*)

Sposta il cursore all'inizio della prima riga.

Parametri

lcd	display da pilotare;

4.2.3.7 void HD44780_Init4 (HD44780_LCD_t * *lcd*, GPIO_t * *gpio*, GPIO_mask *RS*, GPIO_mask *RW*, GPIO_mask *E*, GPIO_mask *Data7*, GPIO_mask *Data6*, GPIO_mask *Data5*, GPIO_mask *Data4*)

Inizializza un oggetto display lcd HD44780 affinche' si utilizzi l'interfaccia a 4 bit.

Inizializza un oggetto HD44780_LCD_t verificando la validita' delle coppie porta-pin per l' interfacciamento, configurando i pin GPIO e inizializzando il device.

Avvertimento

Se i pin associati ai segnali di pilotaggio del device non sono correttamente configurati come pin di output, il dispositivo non funzionera' correttamente.
Non modificare i campi della struttura HD44780_LCD_t dopo che essa sia stata inizializzata.
La struttura GPIO_t, a cui fa riferimento il parametro gpio, va inizializzata a parte.

Parametri

lcd	puntatore a struttura di tipo HD44780_LCD_t che descrive un display HD44780 da inizializzare;
gpio	puntatore alla struttura GPIO_t che astrae il device GPIO a cui il display e' connesso. Non viene inizializzato dalla funziona, sara' necessario inizializzarlo preventivamente;
RS	pin del device GPIO a cui e' associato il segnale RS (data/command) del display LCD;
RW	pin del device GPIO a cui e' associato il segnale RW (read/write) del display LCD;
E	pin del device GPIO a cui e' associato il segnale E (Enable) del display LCD;
Data7	pin del device GPIO a cui e' associato il segnale Data7 del display LCD;
Data6	pin del device GPIO a cui e' associato il segnale Data6 del display LCD;
Data5	pin del device GPIO a cui e' associato il segnale Data5 del display LCD;
Data4	pin del device GPIO a cui e' associato il segnale Data4 del display LCD;

```
1 GPIO_t gpioDisplay;
2 GPIO_init(&gpioDisplay, XPAR_MYGPIO_3_S00_AXI_BASEADDR, 11, 0, 4, 8);
3 HD44780_LCD_t lcd;
4 HD44780_Init4(&lcd, &gpioDisplay,   GPIO_pin10, GPIO_pin9, GPIO_pin8,
5                                         GPIO_pin0, GPIO_pin1, GPIO_pin2, GPIO_pin3);
6 HD44780_Print(&lcd, "Ciao! Come va");
7 HD44780_MoveToRow2(&lcd);
8 HD44780_Print(&lcd, "lo studio?");
```

4.2.3.8 void HD44780_Init8 (HD44780_LCD_t * *lcd*, GPIO_t * *gpio*, GPIO_mask *RS*, GPIO_mask *RW*, GPIO_mask *E*, GPIO_mask *Data7*, GPIO_mask *Data6*, GPIO_mask *Data5*, GPIO_mask *Data4*, GPIO_mask *Data3*, GPIO_mask *Data2*, GPIO_mask *Data1*, GPIO_mask *Data0*)

Inizializza un display lcd HD44780 con interfacciamento ad 8 bit.

Avvertimento

Se i pin associati ai segnali di pilotaggio del device non sono correttamente configurati come pin di output, il dispositivo non funzionera' correttamente.
Non modificare i campi della struttura dopo che essa sia stata inizializzata.
La struttura GPIO_t, a cui fa riferimento il parametro gpio, va inizializzata a parte.

Parametri

lcd	puntatore a struttura di tipo HD44780_LCD_t che descrive un display HD44780 da inizializzare;
gpio	puntatore alla struttura GPIO_t che astrae il device GPIO a cui il display e' connesso. Non viene inizializzato dalla funziona, sara' necessario inizializzarlo preventivamente;
RS	pin del device GPIO a cui e' associato il segnale RS (data/command) del display LCD;
RW	pin del device GPIO a cui e' associato il segnale RW (read/write) del display LCD;
E	pin del device GPIO a cui e' associato il segnale E (Enable) del display LCD;
Data7	pin del device GPIO a cui e' associato il segnale Data7 del display LCD;
Data6	pin del device GPIO a cui e' associato il segnale Data6 del display LCD;
Data5	pin del device GPIO a cui e' associato il segnale Data5 del display LCD;
Data4	pin del device GPIO a cui e' associato il segnale Data4 del display LCD;
Data3	pin del device GPIO a cui e' associato il segnale Data3 del display LCD;
Data2	pin del device GPIO a cui e' associato il segnale Data2 del display LCD;
Data1	pin del device GPIO a cui e' associato il segnale Data1 del display LCD;
Data0	pin del device GPIO a cui e' associato il segnale Data0 del display LCD;

```
1 GPIO_t gpioDisplay;
2 GPIO_init(&gpioDisplay, XPAR_MYGPIO_3_S00_AXI_BASEADDR, 11, 0, 4, 8);
3 HD44780_LCD_t lcd;
4 HD44780_Init8(&lcd, &gpioDisplay,   GPIO_pin10, GPIO_pin9, GPIO_pin8,
5                                         GPIO_pin0, GPIO_pin1, GPIO_pin2, GPIO_pin3,
6                                     GPIO_pin4, GPIO_pin5, GPIO_pin6, GPIO_pin7);
7 HD44780_Print(&lcd, "Ciao! Come va");
8 HD44780_MoveToRow2(&lcd);
9 HD44780_Print(&lcd, "lo studio?");
```

4.2.3.9 void HD44780_MoveCursor (HD44780_LCD_t * *lcd,* HD44780_Direction_t *dir*)

Sposta il cursore di una posizione a destra o sinistra.

Parametri

lcd	display da pilotare;
dir	direzione in cui spostare il cursore,

Si veda anche

direction_t;

4.2.3.10 void HD44780_MoveToRow1 (HD44780_LCD_t * *lcd*)

Sposta il cursore all'inizio della prima riga.

Parametri

lcd	display da pilotare;

4.2.3.11 void HD44780_MoveToRow2 (HD44780_LCD_t * *lcd*)

Sposta il cursore all'inizio della seconda riga.

Parametri

lcd	display da pilotare;

4.2.3.12 void HD44780_Print (HD44780_LCD_t * *lcd,* const char * *s*)

Stampa una stringa null-terminated di caratteri.

La funzione può essere utilizzata per stampare anche numeri interi e floating point. Si veda gli esempi di cui sotto.

Parametri

lcd	display da pilotare;
s	puntatore alla stringa null-terminated da stampare sul display;

```
1 // stampa di un intero
2 #include <stdlib.h>
3     ...
4 char str[10];   // assicurarsi di allocare sufficiente spazio per la stampa del numero
5 sprintf(str,"%d", integer_number);
6 error = HD44780_Print(lcd, str);
7 // stampa di un intero
8 #include <stdlib.h>
9 ...
10 char str[10];
11 snprintf(str, 10,"%d", integer_number);
12 error = HD44780_Print(lcd, str);
13 // stampa di un float
14 #include <stdlib.h>
15 ...
16 char str[20];   // assicurarsi di allocare sufficiente spazio per la stampa del numero
17 sprintf(str,"%f", float_number);
18 error = HD44780_Print(lcd, str);
19 // stampa di un float, nel caso in cui la soluzione precedente dovesse non funzionare
20 #include <stdlib.h>
21 ...
22 char str[20];
23 int parte_intera, parte_decimale, moltiplicatore = 1000;
24 // se si desiderano piu' di tre cifre decimali basta aumentare la potenza del
25 // moltiplicatore
26 // es. cinque cifre decimali ==> moltiplicatore = 100000
```

```
27 // si sconsiglia di stampare piu' di quattro cifre decimali per non causare overflow
28 // nelle istruzioni che seguono
29 parte_intera = (int) float_number;
30 parte_decimale = (int)(float_number * moltiplicatore) - (parte_intera * moltiplicatore);
31 snprintf(str, 20,"%d.%d", parte_intera, parte_decimale);
32 error = HD44780_Print(lcd, str);
```

4.2.3.13 void HD44780_printBinary32 (HD44780_LCD_t * *lcd*, uint32_t *w*)

Stampa una word di 32 bit in binario. (bit piu' significativo a sinistra)

Parametri

lcd	
w	word da stampare

4.2.3.14 void HD44780_printBinary64 (HD44780_LCD_t * *lcd*, uint64_t *b*)

Stampa un blocco di 64 bit in binario. (bit piu' significativo a sinistra)

Parametri

lcd	
b	blocco da stampare

4.2.3.15 void HD44780_printBinary8 (HD44780_LCD_t * *lcd*, uint8_t *b*)

Stampa un byte in binario. (bit piu' significativo a sinistra)

Parametri

lcd	
b	byte da stampare

4.2.3.16 void HD44780_Printc (HD44780_LCD_t * *lcd*, char *c*)

Stampa un carattere.

Parametri

lcd	display da pilotare;
c	carattere da stampare sul display;

4.2.3.17 void HD44780_printHex32 (HD44780_LCD_t * *lcd*, uint32_t *w*)

Stampa una word di 32 bit in esadecimale. (bit piu' significativo a sinistra)

Parametri

lcd	
w	word da stampare

4.2.3.18 void HD44780_printHex64 (HD44780_LCD_t * *lcd*, uint64_t *b*)

Stampa un blocco di 64 bit in esadecimale. (bit piu' significativo a sinistra)

Parametri

lcd	
b	blocco da stampare

4.2.3.19 void HD44780_printHex8 (HD44780_LCD_t * *lcd,* uint8_t *b*)

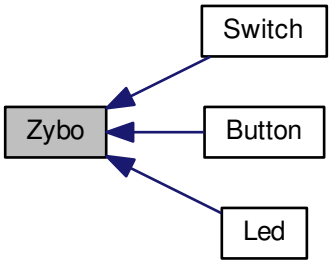
Stampa un byte in esadecimale. (bit piu' significativo a sinistra)

Parametri

lcd	
b	byte da stampare

4.3 Zybo

Diagramma di collaborazione per Zybo:



Moduli

- Button
- Led
- Switch

4.3.1 Descrizione dettagliata

4.4 Button

Diagramma di collaborazione per Button:

Strutture dati

- struct ZyboButton_t

  Struttura opaca che astrae l'insieme dei button presenti sulla board Digilent Zybo;.

Definizioni

- #define ZyboButton(i) ((uint32_t)(1<<i))

  Metodo alternativo per la specifica di uno dei button presenti sulla board Digilent Zybo.
- #define ZyboButton_DebounceWait 50

  Tempo di attesa (in millisecondi) usato per prevenire il fenomeno del bouncing. Il valore di default è 50, determinato empiricamente. Puo' essere modificato a piacimento cambiando il valore alla macro seguente.

Tipi enumerati (enum)

- enum ZyboButton_mask_t { ZyboButton3 = 0x8, ZyboButton2 = 0x4, ZyboButton1 = 0x2, ZyboButton0 = 0x1 }

  Maschere di selezione dei PushButton.
- enum ZyboButton_status_t { ZyboButton_off, ZyboButton_on }

  Status di attivo/inattivo dei PushButton.

Funzioni

- void ZyboButton_init (ZyboButton_t *buttons, GPIO_t *gpio, GPIO_mask Button3_pin, GPIO_mask Button2_pin, GPIO_mask Button1_pin, GPIO_mask Button0_pin)

  Inizializza un oggetto di tipo ZyboButton_t.
- void ZyboButton_waitWhileIdle (ZyboButton_t *buttons)

  Permettere di mettere il programma in attesa attiva finche' i button restano inattivi;.
- void ZyboButton_waitWhileBusy (ZyboButton_t *buttons)

  Permettere di mettere il programma in attesa attiva finche' i button restano attivi;.
- ZyboButton_status_t ZyboButton_getStatus (ZyboButton_t *buttons, ZyboButton_mask_t mask)

  Permette la lettura dello stato dei button presenti sulla board.

4.4.1 Descrizione dettagliata

4.4.2 Documentazione delle definizioni

4.4.2.1 #define ZyboButton(*i*) ((uint32_t)(1<<i))

Metodo alternativo per la specifica di uno dei button presenti sulla board Digilent Zybo.

Parametri

i	indice del button da selezionare, da 0 a 3

Restituisce

maschera di selezione del button i-esimo

4.4.2.2 #define ZyboButton_DebounceWait 50

Tempo di attesa (in millisecondi) usato per prevenire il fenomeno del bouncing. Il valore di default è 50, determinato empiricamente. Puo' essere modificato a piacimento cambiando il valore alla macro seguente.

4.4.3 Documentazione dei tipi enumerati

4.4.3.1 enum ZyboButton_mask_t

Maschere di selezione dei PushButton.

Valori del tipo enumerato

ZyboButton3 ZyboButton3, seleziona il button 3 sulla board Digilent Zybo;.

ZyboButton2 ZyboButton2, seleziona il button 2 sulla board Digilent Zybo;.

ZyboButton1 ZyboButton1, seleziona il button 1 sulla board Digilent Zybo;.

ZyboButton0 ZyboButton0, seleziona il button 0 sulla board Digilent Zybo;.

4.4.3.2 enum ZyboButton_status_t

Status di attivo/inattivo dei PushButton.

Valori del tipo enumerato

ZyboButton_off ZyboButton_off, corrisponde al valore logico '0', indica che un pushbutton e' inattivo;.

ZyboButton_on ZyboButton_on, corrisponde al valore logico '1', indica che un pushbutton e' attivo;.

4.4.4 Documentazione delle funzioni

4.4.4.1 ZyboButton_status_t ZyboButton_getStatus (ZyboButton_t * *buttons*, ZyboButton_mask_t *mask*)

Permette la lettura dello stato dei button presenti sulla board.

Parametri

buttons	puntatore a struttura ZyboButton_t, che astrae l'insieme dei button presenti sulla board Digilent Zybo;
mask	maschera di selezione dei button, quelli non selezionati non vengono tenuti in considerazione

Restituisce

status status dei button

Valori di ritorno

ZyboButton_on	se uno dei button selezionati e' attivo;
ZyboButton_off	altrimenti

```
1 ZyboButton_status_t button_status = ZyboButton_getStatus(&buttons, ZyboButton3);                // leggo lo
      stato ddi button 3
2 ZyboLed_status_t led_status = (button_status == ZyboButton_on ? ZyboLed_on : ZyboLed_off);  // se lo stato
      e' attivo accendo il led 3
3 ZyboLed_setStatus(&leds, ZyboLed3, led_status);                                             //
      accendo/spengo led 3
```

Avvertimento

Usa la macro assert per verificare che:

- buttons non sia un puntatore nullo;
- buttons->gpio non sia un puntatore nullo

4.4.4.2 void ZyboButton_init (ZyboButton_t * *buttons,* GPIO_t * *gpio,* GPIO_mask *Button3_pin,* GPIO_mask *Button2_pin,* GPIO_mask *Button1_pin,* GPIO_mask *Button0_pin*)

Inizializza un oggetto di tipo ZyboButton_t.

Parametri

buttons	puntatore a struttura ZyboButton_t, che astrae l'insieme dei button presenti sulla board Digilent Zybo;
gpio	puntatore a struttura GPIO_t, che astrae un device GPIO; non viene inizializzato dalla funzione, sara' necessario inizializzarlo preventivamente; si faccia riferimento all'esempio riportato di seguito
Button3_pin	pin del device GPIO a cui e' associato il button 3 della board Digilent Zybo;
Button2_pin	pin del device GPIO a cui e' associato il button 2 della board Digilent Zybo;
Button1_pin	pin del device GPIO a cui e' associato il button 1 della board Digilent Zybo;
Button0_pin	pin del device GPIO a cui e' associato il button 0 della board Digilent Zybo;

```
1 GPIO_t gpioButton;
2 GPIO_init(&gpioButton, XPAR_MYGPIO_1_S00_AXI_BASEADDR, 4, 0, 4, 8);
3 ZyboButton_t buttons;
4 ZyboButton_init(&buttons, &gpioButton, GPIO_pin3, GPIO_pin2, GPIO_pin1, GPIO_pin0);
```

Avvertimento

Usa la macro assert per verificare che:

- buttons non sia un puntatore nullo;
- gpio non sia un puntatore nullo
- ButtonN_pin siano tutti pin differenti

4.4.4.3 void ZyboButton_waitWhileBusy (ZyboButton_t * *buttons*)

Permettere di mettere il programma in attesa attiva finche' i button restano attivi;.

Avvertimento

La funzione integra le funzionalita' di debouncing. Il tempo di attesa e' determinato sulla base del valore della macro ZyboButton_DebounceWait

Parametri

buttons	puntatore a struttura ZyboButton_t, che astrae l'insieme dei button presenti sulla board Digilent Zybo;

```
1 ZyboButton_waitWhileIdle(&buttons);
2 ZyboButton_status_t button3_status = ZyboButton_getStatus(&buttons, ZyboButton3);              // leggo lo
      stato ddi button 3
3 ZyboButton_status_t button2_status = ZyboButton_getStatus(&buttons, ZyboButton2);              // leggo lo
      stato ddi button 2
4 ZyboButton_status_t button1_status = ZyboButton_getStatus(&buttons, ZyboButton1);              // leggo lo
      stato ddi button 1
5 ZyboButton_status_t button0_status = ZyboButton_getStatus(&buttons, ZyboButton0);              // leggo lo
      stato ddi button 0
6 ZyboButton_waitWhileBusy(&buttons);
```

Avvertimento

Usa la macro assert per verificare che:

- buttons non sia un puntatore nullo;
- buttons->gpio non sia un puntatore nullo

4.4.4.4 void ZyboButton_waitWhileIdle (ZyboButton_t * *buttons*)

Permettere di mettere il programma in attesa attiva finche' i button restano inattivi;.

Avvertimento

La funzione integra le funzionalita' di debouncing. Il tempo di attesa e' determinato sulla base del valore della macro ZyboButton_DebounceWait

Parametri

buttons	puntatore a struttura ZyboButton_t, che astrae l'insieme dei button presenti sulla board Digilent Zybo;

```
1 ZyboButton_waitWhileIdle(&buttons);
2 ZyboButton_status_t button3_status = ZyboButton_getStatus(&buttons, ZyboButton3);              // leggo lo
      stato ddi button 3
3 ZyboButton_status_t button2_status = ZyboButton_getStatus(&buttons, ZyboButton2);              // leggo lo
      stato ddi button 2
4 ZyboButton_status_t button1_status = ZyboButton_getStatus(&buttons, ZyboButton1);              // leggo lo
      stato ddi button 1
5 ZyboButton_status_t button0_status = ZyboButton_getStatus(&buttons, ZyboButton0);              // leggo lo
      stato ddi button 0
6 ZyboButton_waitWhileBusy(&buttons);
```

Avvertimento

Usa la macro assert per verificare che:

- buttons non sia un puntatore nullo;
- buttons->gpio non sia un puntatore nullo

4.5 Led

Diagramma di collaborazione per Led:

Strutture dati

- struct ZyboLed_t

  Struttura opaca che astrae l'insieme dei Led presenti sulla board Digilent Zybo;.

Definizioni

- #define ZyboLed(i) ((uint32_t)(1<<i))

  Metodo alternativo per la specifica di uno dei led presenti sulla board Digilent Zybo.

Tipi enumerati (enum)

- enum ZyboLed_mask_t { ZyboLed3 = 0x8, ZyboLed2 = 0x4, ZyboLed1 = 0x2, ZyboLed0 = 0x1 }

  Maschere di selezione dei led.
- enum ZyboLed_status_t { ZyboLed_off, ZyboLed_on }

  Status di accensione/spegnimento dei led.

Funzioni

- void ZyboLed_init (ZyboLed_t *leds, GPIO_t *gpio, GPIO_mask Led3_pin, GPIO_mask Led2_pin, GPIO_↩ mask Led1_pin, GPIO_mask Led0_pin)

  Inizializza un oggetto di tipo ZyboLed_t.
- void ZyboLed_setStatus (ZyboLed_t *leds, ZyboLed_mask_t mask, ZyboLed_status_t status)

  Permette di accendere/spegnere i Led sulla board.
- void ZyboLed_toggle (ZyboLed_t *leds, ZyboLed_mask_t mask)

  Permette di accendere/spegnere i Led sulla board, invertendone il valore.

4.5.1 Descrizione dettagliata

4.5.2 Documentazione delle definizioni

4.5.2.1 #define ZyboLed(*i*) ((uint32_t)(1<<i))

Metodo alternativo per la specifica di uno dei led presenti sulla board Digilent Zybo.

Parametri

i	indice del led da selezionare, da 0 a 3

Restituisce

maschera di selezione del led i-esimo

4.5.3 Documentazione dei tipi enumerati

4.5.3.1 enum ZyboLed_mask_t

Maschere di selezione dei led.

Valori del tipo enumerato

ZyboLed3 ZyboLed3, seleziona il led 3 sulla board Digilent Zybo;.

ZyboLed2 ZyboLed2, seleziona il led 2 sulla board Digilent Zybo;.

ZyboLed1 ZyboLed1, seleziona il led 1 sulla board Digilent Zybo;.

ZyboLed0 ZyboLed0, seleziona il led 0 sulla board Digilent Zybo;.

4.5.3.2 enum ZyboLed_status_t

Status di accensione/spegnimento dei led.

Valori del tipo enumerato

ZyboLed_off ZyboLed_off, corrisponde al valore logico '0', per lo spegnimento dei Led.

ZyboLed_on ZyboLed_on, corrisponde al valore logico '1', per l'accensione dei Led.

4.5.4 Documentazione delle funzioni

4.5.4.1 void ZyboLed_init (ZyboLed_t * *leds*, GPIO_t * *gpio*, GPIO_mask *Led3_pin*, GPIO_mask *Led2_pin*, GPIO_mask *Led1_pin*, GPIO_mask *Led0_pin*)

Inizializza un oggetto di tipo ZyboLed_t.

Parametri

leds	puntatore a struttura ZyboLed_t, che astrae l'insieme dei Led presenti sulla board Digilent Zybo;
gpio	puntatore a struttura GPIO_t, che astrae un device GPIO; non viene inizializzato dalla funzione, sara' necessario inizializzarlo preventivamente; si faccia riferimento all'esempio riportato di seguito
Led3_pin	pin del device GPIO a cui e' associato il Led3 della board Digilent Zybo;
Led2_pin	pin del device GPIO a cui e' associato il Led2 della board Digilent Zybo;
Led1_pin	pin del device GPIO a cui e' associato il Led1 della board Digilent Zybo;
Led0_pin	pin del device GPIO a cui e' associato il Led0 della board Digilent Zybo;

```
1 GPIO_t gpioLed;
2 GPIO_init(&gpioLed, XPAR_MYGPIO_0_S00_AXI_BASEADDR, 4, 0, 4, 8);               // inizializzazione del
      device GPIO
3 ZyboLed_t leds;
4 ZyboLed_init(&leds, &gpioLed, GPIO_pin3, GPIO_pin2, GPIO_pin1, GPIO_pin0);  // inzializzazione della
      struttura ZyboLed_t
```

Avvertimento

Usa la macro assert per verificare che:

- leds non sia un puntatore nullo;
- gpio non sia un puntatore nullo
- LedN_pin siano tutti pin differenti

4.5.4.2 void ZyboLed_setStatus (ZyboLed_t * *leds*, ZyboLed_mask_t *mask*, ZyboLed_status_t *status*)

Permette di accendere/spegnere i Led sulla board.

Parametri

leds	puntatore a struttura ZyboLed_t, che astrae l'insieme dei Led presenti sulla board Digilent Zybo;
mask	maschera di selezione dei led, quelli non selezionati vengono lasciati inalterati
status	status dei led, ZyboLed_on per accendere, ZyboLed_off per spegnere

```
1 ZyboLed_setStatus(&leds, ZyboLed3 | ZyboLed1, ZyboLed_on);  // accensione dei Led 3 ed 1
2 ZyboLed_setStatus(&leds, ZyboLed3 | ZyboLed1, ZyboLed_off); // spegnimento dei Led 3 ed 1
3 ZyboLed_setStatus(&leds, ZyboLed2 | ZyboLed0, ZyboLed_on);  // accendsione dei Led 2 ed 0
4 ZyboLed_setStatus(&leds, ZyboLed2 | ZyboLed0, ZyboLed_off); // spegnimento dei Led 2 ed 0
5 ZyboLed_setStatus(&leds, ZyboLed3 | ZyboLed1 | ZyboLed2 | ZyboLed0, ZyboLed_on);        // accensone di
      tutti i led
6 ZyboLed_setStatus(&leds, ZyboLed3 | ZyboLed1 | ZyboLed2 | ZyboLed0, ZyboLed_off);   // spegnimento di tutti
      i led
```

Avvertimento

Usa la macro assert per verificare che:

- leds non sia un puntatore nullo;
- leds->gpio non sia un puntatore nullo

4.5.4.3 void ZyboLed_toggle (ZyboLed_t * *leds*, ZyboLed_mask_t *mask*)

Permette di accendere/spegnere i Led sulla board, invertendone il valore.

Parametri

leds	puntatore a struttura ZyboLed_t, che astrae l'insieme dei Led presenti sulla board Digilent Zybo;
mask	maschera di selezione dei led, quelli non selezionati vengono lasciati inalterati

```
1 ZyboLed_toggle(&leds, ZyboLed3 | ZyboLed1); // accensione/spegnimento dei Led 3 ed 1
```

Avvertimento

Usa la macro assert per verificare che:

- leds non sia un puntatore nullo;
- leds->gpio non sia un puntatore nullo

4.6 Switch

Diagramma di collaborazione per Switch:

Strutture dati

- struct ZyboSwitch_t

  Struttura opaca che astrae l'insieme degli switch presenti sulla board Digilent Zybo;.

Definizioni

- #define ZyboSwitch(i) ((uint32_t)(1<<i))

  Metodo alternativo per la specifica di uno degli switch presenti sulla board Digilent Zybo.

Tipi enumerati (enum)

- enum ZyboSwitch_mask_t { ZyboSwitch3 = 0x8, ZyboSwitch2 = 0x4, ZyboSwitch1 = 0x2, ZyboSwitch0 = 0x1 }

  Maschere di selezione degli switch.
- enum ZyboSwitch_status_t { ZyboSwitch_off, ZyboSwitch_on }

  Status di attivo/inattivo degli switch.

Funzioni

- void ZyboSwitch_init (ZyboSwitch_t *switches, GPIO_t *gpio, GPIO_mask Switch3_pin, GPIO_mask Switch2_pin, GPIO_mask Switch1_pin, GPIO_mask Switch0_pin)

  Inizializza un oggetto di tipo ZyboSwitch_t.
- ZyboSwitch_status_t ZyboSwitch_getStatus (ZyboSwitch_t *switches, ZyboSwitch_mask_t mask)

  Permette la lettura dello stato degli switch presenti sulla board.

4.6.1 Descrizione dettagliata

4.6.2 Documentazione delle definizioni

4.6.2.1 #define ZyboSwitch(*i*) ((uint32_t)(1<<i))

Metodo alternativo per la specifica di uno degli switch presenti sulla board Digilent Zybo.

Parametri

i	indice dello switch da selezionare, da 0 a 3

Restituisce

maschera di selezione dello switch i-esimo

4.6.3 Documentazione dei tipi enumerati

4.6.3.1 enum ZyboSwitch_mask_t

Maschere di selezione degli switch.

Valori del tipo enumerato

ZyboSwitch3 ZyboSwitch3, seleziona lo switch 3 sulla board Digilent Zybo;.
ZyboSwitch2 ZyboSwitch2, seleziona lo switch 2 sulla board Digilent Zybo;.
ZyboSwitch1 ZyboSwitch1, seleziona lo switch 1 sulla board Digilent Zybo;.
ZyboSwitch0 ZyboSwitch0, seleziona lo switch 0 sulla board Digilent Zybo;.

4.6.3.2 enum ZyboSwitch_status_t

Status di attivo/inattivo degli switch.

Valori del tipo enumerato

ZyboSwitch_off ZyboSwitch_off, corrisponde al valore logico '0', indica che lo switch e' inattivo;.
ZyboSwitch_on ZyboSwitch_on, corrisponde al valore logico '1', indica che lo switch e' attivo;.

4.6.4 Documentazione delle funzioni

4.6.4.1 ZyboSwitch_status_t ZyboSwitch_getStatus (ZyboSwitch_t * *switches*, ZyboSwitch_mask_t *mask*)

Permette la lettura dello stato degli switch presenti sulla board.

Parametri

switches	puntatore a struttura ZyboSwitch_t, che astrae l'insieme degli switch presenti sulla board Digilent Zybo;
mask	maschera di selezione degli switch, quelli non selezionati non vengono tenuti in considerazione

Restituisce

status status degli switch

Valori di ritorno

ZyboSwitch_on	se uno degli switch selezionati e' attivo;
ZyboSwitch_off	altrimenti

```
1 ZyboSwitch_status_t switch_status = ZyboSwitch_getStatus(&switches, ZyboSwitch3);          // leggo lo
      stato dello switch 3
2 ZyboLed_status_t led_status = (switch_status == ZyboSwitch_on ? ZyboLed_on : ZyboLed_off); // se lo switch
      3 e' attivo accendo il led 3
3 ZyboLed_setStatus(&leds, ZyboLed3, led_status);                                           //
      accendo/spengo led 3
```

Avvertimento

Usa la macro assert per verificare che:

- switches non sia un puntatore nullo;
- switches->gpio non sia un puntatore nullo

4.6.4.2 void ZyboSwitch_init (ZyboSwitch_t * *switches,* GPIO_t * *gpio,* GPIO_mask *Switch3_pin,* GPIO_mask *Switch2_pin,* GPIO_mask *Switch1_pin,* GPIO_mask *Switch0_pin*)

Inizializza un oggetto di tipo ZyboSwitch_t.

Parametri

switches	puntatore a struttura ZyboSwitch_t, che astrae l'insieme degli switch presenti sulla board Digilent Zybo;
gpio	puntatore a struttura GPIO_t, che astrae un device GPIO; non viene inizializzato dalla funzione, sara' necessario inizializzarlo preventivamente; si faccia riferimento all'esempio riportato di seguito
Switch3_pin	pin del device GPIO a cui e' associato lo switch 3 della board Digilent Zybo;
Switch2_pin	pin del device GPIO a cui e' associato lo switch 2 della board Digilent Zybo;
Switch1_pin	pin del device GPIO a cui e' associato lo switch 1 della board Digilent Zybo;
Switch0_pin	pin del device GPIO a cui e' associato lo switch 0 della board Digilent Zybo;

```
1 GPIO_t gpioSwitch;
2 GPIO_init(&gpioSwitch, XPAR_MYGPIO_1_S00_AXI_BASEADDR, 4, 0, 4, 8);
3 ZyboSwitch_t switches;
4 ZyboSwitch_init(&switches, &gpioSwitch, GPIO_pin3, GPIO_pin2, GPIO_pin1, GPIO_pin0);
```

Avvertimento

Usa la macro assert per verificare che:

- switches non sia un puntatore nullo;
- gpio non sia un puntatore nullo
- SwitchN_pin siano tutti pin differenti

4.7 GPIO

Strutture dati

- struct GPIO_t

  Struttura che astrae un device GPIO.

Definizioni

- #define GPIO_pin(i) ((uint32_t)(1<<i))

  Metodo alternativo per la specifica di uno dei pin di un device GPIO.

Tipi enumerati (enum)

- enum GPIO_mask {
  GPIO_pin0 = 0x1, GPIO_pin1 = 0x2, GPIO_pin2 = 0x4, GPIO_pin3 = 0x8,
  GPIO_pin4 = 0x10, GPIO_pin5 = 0x20, GPIO_pin6 = 0x40, GPIO_pin7 = 0x80,
  GPIO_pin8 = 0x100, GPIO_pin9 = 0x200, GPIO_pin10 = 0x400, GPIO_pin11 = 0x800,
  GPIO_pin12 = 0x1000, GPIO_pin13 = 0x2000, GPIO_pin14 = 0x4000, GPIO_pin15 = 0x8000,
  GPIO_pin16 = 0x10000, GPIO_pin17 = 0x20000, GPIO_pin18 = 0x40000, GPIO_pin19 = 0x80000,
  GPIO_pin20 = 0x100000, GPIO_pin21 = 0x200000, GPIO_pin22 = 0x400000, GPIO_pin23 = 0x800000,
  GPIO_pin24 = 0x1000000, GPIO_pin25 = 0x2000000, GPIO_pin26 = 0x4000000, GPIO_pin27 = 0x8000000,
  GPIO_pin28 = 0x10000000, GPIO_pin29 = 0x20000000, GPIO_pin30 = 0x40000000, GPIO_pin31 = 0x80000000,
  GPIO_byte0 = 0x000000ff, GPIO_byte1 = 0x0000ff00, GPIO_byte2 = 0x00ff0000, GPIO_byte3 = 0xff000000
  }

  Maschere di selezione dei pin di un device GPIO.
- enum GPIO_mode { GPIO_read, GPIO_write }

  GPIO_mode, modalita' di funzionamento (lettura/scrittura) di un device GPIO.
- enum GPIO_value { GPIO_reset, GPIO_set }

  GPIO_value.

Funzioni

- void GPIO_init (GPIO_t *gpio, uint32_t *base_address, uint8_t width, uint8_t enable_offset, uint8_t write_↩ offset, uint8_t read_offset)

  Inizializza un device GPIO.
- void GPIO_setMode (GPIO_t *gpio, GPIO_mask mask, GPIO_mode mode)

  Permette di settare la modalita' lettura/scrittura dei pin di un device GPIO;.
- void GPIO_setValue (GPIO_t *gpio, GPIO_mask mask, GPIO_value value)

  Permette di settare il valore dei pin di un device GPIO;.
- void GPIO_toggle (GPIO_t *gpio, GPIO_mask mask)

  Permette di invertire il valore dei pin di un device GPIO;.
- GPIO_value GPIO_getValue (GPIO_t *gpio, GPIO_mask mask)

  Permette di leggere il valore dei pin di un device GPIO;.

4.7.1 Descrizione dettagliata

4.7.2 Documentazione delle definizioni

4.7.2.1 #define GPIO_pin(*i*) ((uint32_t)(1<<i))

Metodo alternativo per la specifica di uno dei pin di un device GPIO.

Parametri

i	indice del bit da selezionare, da 0 (bit meno significativo) a 31 (bit piu' significativo)

Restituisce

maschera di selezione del pin i-esimo

4.7.3 Documentazione dei tipi enumerati

4.7.3.1 enum GPIO_mask

Maschere di selezione dei pin di un device GPIO.

Valori del tipo enumerato

GPIO_pin0 GPIO pin0.
GPIO_pin1 GPIO pin1.
GPIO_pin2 GPIO pin2.
GPIO_pin3 GPIO pin3.
GPIO_pin4 GPIO pin4.
GPIO_pin5 GPIO pin5.
GPIO_pin6 GPIO pin6.
GPIO_pin7 GPIO pin7.
GPIO_pin8 GPIO pin8.
GPIO_pin9 GPIO pin9.
GPIO_pin10 GPIO pin10.
GPIO_pin11 GPIO pin11.
GPIO_pin12 GPIO pin12.
GPIO_pin13 GPIO pin13.
GPIO_pin14 GPIO pin14.
GPIO_pin15 GPIO pin15.
GPIO_pin16 GPIO pin16.
GPIO_pin17 GPIO pin17.
GPIO_pin18 GPIO pin18.
GPIO_pin19 GPIO pin19.
GPIO_pin20 GPIO pin20.
GPIO_pin21 GPIO pin21.
GPIO_pin22 GPIO pin22.
GPIO_pin23 GPIO pin23.
GPIO_pin24 GPIO pin24.
GPIO_pin25 GPIO pin25.
GPIO_pin26 GPIO pin26.
GPIO_pin27 GPIO pin27.
GPIO_pin28 GPIO pin28.
GPIO_pin29 GPIO pin29.
GPIO_pin30 GPIO pin30.
GPIO_pin31 GPIO pin31.
GPIO_byte0 GPIO byte0.
GPIO_byte1 GPIO byte1.
GPIO_byte2 GPIO byte2.
GPIO_byte3 GPIO byte3.

4.7.3.2 enum GPIO_mode

GPIO_mode, modalita' di funzionamento (lettura/scrittura) di un device GPIO.

Valori del tipo enumerato

GPIO_read GPIO_read modalita' lettura.

GPIO_write GPIO_write modalita' scrittura.

4.7.3.3 enum GPIO_value

GPIO_value.

Valori del tipo enumerato

GPIO_reset GPIO_reset, corrisponde al valore logico '0'.

GPIO_set GPIO_set, corrisponde al valore logico '1'.

4.7.4 Documentazione delle funzioni

4.7.4.1 GPIO_value GPIO_getValue (GPIO_t * *gpio,* GPIO_mask *mask*)

Permette di leggere il valore dei pin di un device GPIO;.

```
1 // legge il valore del pin 0 di un device GPIO.
2 GPIO_value value = gpio_getValue(gpio, GPIO_pin0);
3
4 // legge il valore del pin 0, 3 e 5 di un device GPIO.
5 // Verra' restituita la OR tra i valori dei pin
6 GPIO_value value = gpio_getValue(gpio, GPIO_pin0 | GPIO_pin3 | GPIO_pin5);
```

Parametri

gpio	puntatore a GPIO_t, che astrae un device GPIO;
mask	maschera dei pin su cui agire;

Restituisce

restituisce la OR dei pin letti

Valori di ritorno

GPIO_set	se uno dei pin letti e' GPIO_set,
GPIO_reset	se TUTTI i pin sono GPIO_reset

Avvertimento

Usa la macro assert per verificare che gpio non sia un puntatore nullo

4.7.4.2 void GPIO_init (GPIO_t * *gpio,* uint32_t * *base_address,* uint8_t *width,* uint8_t *enable_offset,* uint8_t *write_offset,* uint8_t *read_offset*)

Inizializza un device GPIO.

Inizializza una struttura di tipo GPIO_t, che astrae u device GPIO, controllando che l'inizializzazione vada a buon fine, effettuando diversi test sui parametri di inizializzazione e restituendo un codice di errore.

Parametri

gpio	puntatore a GPIO_t, che astrae un device GPIO;
base_address	indirizzo di memoria a cui e' mappato il device GPIO;
width	numero di pin di ingresso/uscita del device GPIO; Deve essere un numero compreso tra 1 e 32;
enable_offset	offset in byte del registro "enable" del device GPIO; il registro permette di impostare la modalita' di funzionamento del device GPIO;
write_offset	offset in byte del registro "write" del device GPIO; il registro permette di scrivere sui pin del device GPIO;
read_offset	offset in byte del registro "read" del device GPIO; il registro permette di leggere dai pin del device GPIO;

```
1 GPIO_t gpio;
2 GPIO_init(&gpio, BASE_ADDRESS, WIDTH, EN_OFFSET; WR_OFFSET, RD_OFFSET);
```

Avvertimento

Usa la macro assert per verificare che gpio non sia un puntatore nullo e che gli offset siano tutti differenti

4.7.4.3 void GPIO_setMode (GPIO_t * *gpio*, GPIO_mask *mask*, GPIO_mode *mode*)

Permette di settare la modalita' lettura/scrittura dei pin di un device GPIO;.

```
1 // setta i pin 0 ed 1 di un device GPIO come pin di uscita, gli altri restano invariati
2 GPIO_setMode(gpio, GPIO_pin0 | GPIO_pin1, GPIO_write);
3
4 // setta i pin 19, 20 e 21 di un device GPIO come pin di ingresso, gli altri restano invariati
5 GPIO_setMode(gpio, GPIO_pin19 | GPIO_pin20 | GPIO_pin21, GPIO_read);
```

Parametri

gpio	puntatore a GPIO_t, che astrae un device GPIO;
mask	maschera dei pin su cui agire;
mode	modalita' di funzionamento dei pin;

Avvertimento

Usa la macro assert per verificare che gpio non sia un puntatore nullo

4.7.4.4 void GPIO_setValue (GPIO_t * *gpio*, GPIO_mask *mask*, GPIO_value *value*)

Permette di settare il valore dei pin di un device GPIO;.

```
1 // setta i pin 0 ed 1 di un device GPIO a livello logico '1', gli altri restano invariati
2 GPIO_setValue(gpio, GPIO_pin0 | GPIO_pin1, GPIO_set);
3
4 // setta i pin 19, 20 e 21 di un device GPIO a livello logico '0', gli altri restano invariati
5 GPIO_setValue(gpio, GPIO_pin19 | GPIO_pin20 | GPIO_pin21, GPIO_reset);
```

Parametri

gpio	puntatore a GPIO_t, che astrae un device GPIO;

mask	maschera dei pin su cui agire;
value	valore dei pin

Avvertimento

Usa la macro assert per verificare che gpio non sia un puntatore nullo

4.7.4.5 void GPIO_toggle (GPIO_t * *gpio,* GPIO_mask *mask*)

Permette di invertire il valore dei pin di un device GPIO;.

```
1 // inverte i pin 0 ed 1 di un device GPIO a livello logico '1', gli altri restano invariati
2 GPIO_toggle(gpio, GPIO_pin0 | GPIO_pin1);
```

Parametri

gpio	puntatore a GPIO_t, che astrae un device GPIO;
mask	maschera dei pin su cui agire;

Avvertimento

Usa la macro assert per verificare che gpio non sia un puntatore nullo

Capitolo 5

Documentazione delle classi

5.1 Riferimenti per la struct GPIO_t

Struttura che astrae un device GPIO.

`#include <gpio.h>`

Campi

- uint32_t * base_address
- uint8_t width
- uint8_t enable_offset
- uint8_t write_offset
- uint8_t read_offset

5.1.1 Descrizione dettagliata

Struttura che astrae un device GPIO.

5.1.2 Documentazione dei campi

5.1.2.1 uint32_t* base_address

5.1.2.2 uint8_t enable_offset

5.1.2.3 uint8_t read_offset

5.1.2.4 uint8_t width

5.1.2.5 uint8_t write_offset

La documentazione per questa struct è stata generata a partire dal seguente file:

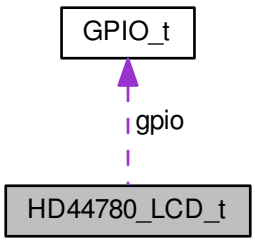
- GPIO/gpio.h

5.2 Riferimenti per la struct HD44780_LCD_t

Struttura opaca che astrae un device Display LCD con cntroller Hitachi HD44780, o compatibile. Un oggetto di tipo HD44780_LCD_t rappresenta un device lcd HD44780. Il modulo e' pensato per permettere la gestione di piu' display da parte dello stesso processore, agendo su oggetti HD44780_LCD_t diversi. Il modulo permette di utilizzare sia l'interfacciamento ad otto bit che quello a quattro bit, inizializzando il device opportunamente, attraverso l'uso delle funzioni HD44780_Init8 eHD44780_Init4. Il modulo fornisce anche semplici funzioni per la stampa di un carattere o di una stringa null-terminated di caratteri. Si veda la documentazione delle funzioni HD44780_Printc e HD44780_Print. Inoltre sono presenti diverse funzioni di utilita' generica, come quelle per la pulizia del display, per lo spostamento del cursore di un posto in avanti o indietro, alla riga in basso o in alto.

`#include <hd44780.h>`

Diagramma di collaborazione per HD44780_LCD_t:



Campi

- GPIO_t * gpio
- GPIO_mask RS
- GPIO_mask RW
- GPIO_mask E
- GPIO_mask Data7
- GPIO_mask Data6
- GPIO_mask Data5
- GPIO_mask Data4
- GPIO_mask Data3
- GPIO_mask Data2
- GPIO_mask Data1
- GPIO_mask Data0
- HD44780_InterfaceMode_t iface_mode

5.2.1 Descrizione dettagliata

Struttura opaca che astrae un device Display LCD con cntroller Hitachi HD44780, o compatibile. Un oggetto di tipo HD44780_LCD_t rappresenta un device lcd HD44780. Il modulo e' pensato per permettere la gestione di piu' display da parte dello stesso processore, agendo su oggetti HD44780_LCD_t diversi. Il modulo permette di utilizzare sia l'interfacciamento ad otto bit che quello a quattro bit, inizializzando il device opportunamente, attraverso l'uso delle funzioni HD44780_Init8 eHD44780_Init4. Il modulo fornisce anche semplici funzioni per la stampa di un carattere o di una stringa null-terminated di caratteri. Si veda la documentazione delle funzioni HD44780_Printc e HD44780_Print. Inoltre sono presenti diverse funzioni di utilita' generica, come quelle per la pulizia del display, per lo spostamento del cursore di un posto in avanti o indietro, alla riga in basso o in alto.

5.2.2 Documentazione dei campi

5.2.2.1 GPIO_mask Data0

5.2.2.2 GPIO_mask Data1

5.2.2.3 GPIO_mask Data2

5.2.2.4 GPIO_mask Data3

5.2.2.5 GPIO_mask Data4

5.2.2.6 GPIO_mask Data5

5.2.2.7 GPIO_mask Data6

5.2.2.8 GPIO_mask Data7

5.2.2.9 GPIO_mask E

5.2.2.10 GPIO_t* gpio

5.2.2.11 HD44780_InterfaceMode_t iface_mode

5.2.2.12 GPIO_mask RS

5.2.2.13 GPIO_mask RW

La documentazione per questa struct è stata generata a partire dal seguente file:

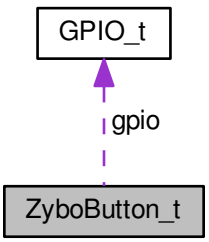
- Lcd/hd44780.h

5.3 Riferimenti per la struct ZyboButton_t

Struttura opaca che astrae l'insieme dei button presenti sulla board Digilent Zybo;.

`#include <ZyboButton.h>`

Diagramma di collaborazione per ZyboButton_t:



Campi

- GPIO_t ∗ gpio
- GPIO_mask Button3_pin
- GPIO_mask Button2_pin
- GPIO_mask Button1_pin
- GPIO_mask Button0_pin

5.3.1 Descrizione dettagliata

Struttura opaca che astrae l'insieme dei button presenti sulla board Digilent Zybo;.

5.3.2 Documentazione dei campi

5.3.2.1 GPIO_mask Button0_pin

5.3.2.2 GPIO_mask Button1_pin

5.3.2.3 GPIO_mask Button2_pin

5.3.2.4 GPIO_mask Button3_pin

5.3.2.5 GPIO_t∗ gpio

La documentazione per questa struct è stata generata a partire dal seguente file:

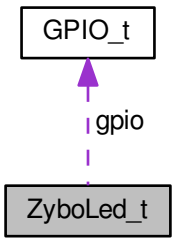
- Zybo/ZyboButton.h

5.4 Riferimenti per la struct ZyboLed_t

Struttura opaca che astrae l'insieme dei Led presenti sulla board Digilent Zybo;.

`#include <ZyboLed.h>`

Diagramma di collaborazione per ZyboLed_t:



Campi

- GPIO_t ∗ gpio

- GPIO_mask Led3_pin
- GPIO_mask Led2_pin
- GPIO_mask Led1_pin
- GPIO_mask Led0_pin

5.4.1 Descrizione dettagliata

Struttura opaca che astrae l'insieme dei Led presenti sulla board Digilent Zybo;.

5.4.2 Documentazione dei campi

5.4.2.1 GPIO_t* gpio

5.4.2.2 GPIO_mask Led0_pin

5.4.2.3 GPIO_mask Led1_pin

5.4.2.4 GPIO_mask Led2_pin

5.4.2.5 GPIO_mask Led3_pin

La documentazione per questa struct è stata generata a partire dal seguente file:

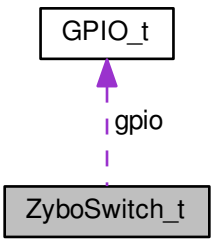
- Zybo/ZyboLed.h

5.5 Riferimenti per la struct ZyboSwitch_t

Struttura opaca che astrae l'insieme degli switch presenti sulla board Digilent Zybo;.

`#include <ZyboSwitch.h>`

Diagramma di collaborazione per ZyboSwitch_t:



Campi

- GPIO_t * gpio
- GPIO_mask Switch3_pin
- GPIO_mask Switch2_pin

- GPIO_mask Switch1_pin
- GPIO_mask Switch0_pin

5.5.1 Descrizione dettagliata

Struttura opaca che astrae l'insieme degli switch presenti sulla board Digilent Zybo;.

5.5.2 Documentazione dei campi

5.5.2.1 GPIO_t* gpio

5.5.2.2 GPIO_mask Switch0_pin

5.5.2.3 GPIO_mask Switch1_pin

5.5.2.4 GPIO_mask Switch2_pin

5.5.2.5 GPIO_mask Switch3_pin

La documentazione per questa struct è stata generata a partire dal seguente file:

- Zybo/ZyboSwitch.h

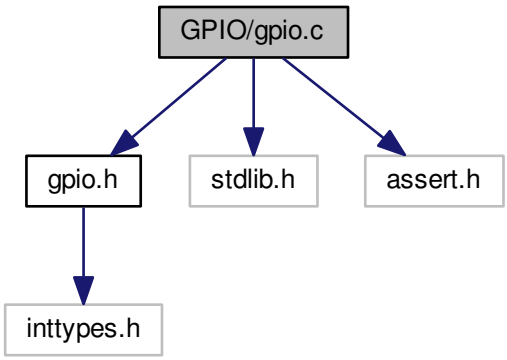
Capitolo 6

Documentazione dei file

6.1 Riferimenti per il file GPIO/gpio.c

```
#include "gpio.h"
#include <stdlib.h>
#include <assert.h>
```

Grafo delle dipendenze di inclusione per gpio.c:



Funzioni

- void GPIO_init (GPIO_t *gpio, uint32_t *base_address, uint8_t width, uint8_t enable_offset, uint8_t write_↩ offset, uint8_t read_offset)

  Inizializza un device GPIO.
- void GPIO_setMode (GPIO_t *gpio, GPIO_mask mask, GPIO_mode mode)

  Permette di settare la modalita' lettura/scrittura dei pin di un device GPIO;.
- void GPIO_setValue (GPIO_t *gpio, GPIO_mask mask, GPIO_value value)

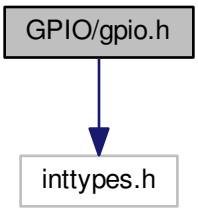
  Permette di settare il valore dei pin di un device GPIO;.
- void GPIO_toggle (GPIO_t *gpio, GPIO_mask mask)

  Permette di invertire il valore dei pin di un device GPIO;.
- GPIO_value GPIO_getValue (GPIO_t *gpio, GPIO_mask mask)

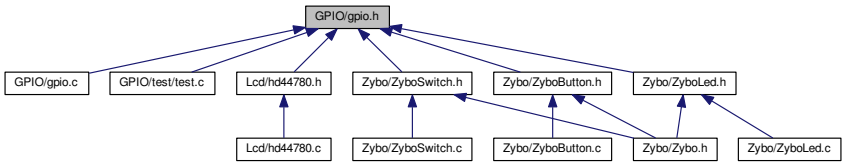
  Permette di leggere il valore dei pin di un device GPIO;.

6.2 Riferimenti per il file GPIO/gpio.h

`#include <inttypes.h>`
Grafo delle dipendenze di inclusione per gpio.h:



Questo grafo mostra quali altri file includono direttamente o indirettamente questo file:



Strutture dati

- struct GPIO_t

  Struttura che astrae un device GPIO.

Definizioni

- #define GPIO_pin(i) ((uint32_t)(1<<i))

  Metodo alternativo per la specifica di uno dei pin di un device GPIO.

Tipi enumerati (enum)

- enum GPIO_mask {
  GPIO_pin0 = 0x1, GPIO_pin1 = 0x2, GPIO_pin2 = 0x4, GPIO_pin3 = 0x8,
  GPIO_pin4 = 0x10, GPIO_pin5 = 0x20, GPIO_pin6 = 0x40, GPIO_pin7 = 0x80,
  GPIO_pin8 = 0x100, GPIO_pin9 = 0x200, GPIO_pin10 = 0x400, GPIO_pin11 = 0x800,
  GPIO_pin12 = 0x1000, GPIO_pin13 = 0x2000, GPIO_pin14 = 0x4000, GPIO_pin15 = 0x8000,
  GPIO_pin16 = 0x10000, GPIO_pin17 = 0x20000, GPIO_pin18 = 0x40000, GPIO_pin19 = 0x80000,
  GPIO_pin20 = 0x100000, GPIO_pin21 = 0x200000, GPIO_pin22 = 0x400000, GPIO_pin23 = 0x800000,
  GPIO_pin24 = 0x1000000, GPIO_pin25 = 0x2000000, GPIO_pin26 = 0x4000000, GPIO_pin27 = 0x8000000,
  GPIO_pin28 = 0x10000000, GPIO_pin29 = 0x20000000, GPIO_pin30 = 0x40000000, GPIO_pin31 =

0x80000000,
GPIO_byte0 = 0x000000ff, GPIO_byte1 = 0x0000ff00, GPIO_byte2 = 0x00ff0000, GPIO_byte3 = 0xff000000
}

Maschere di selezione dei pin di un device GPIO.

- enum GPIO_mode { GPIO_read, GPIO_write }

  GPIO_mode, modalita' di funzionamento (lettura/scrittura) di un device GPIO.
- enum GPIO_value { GPIO_reset, GPIO_set }

  GPIO_value.

Funzioni

- void GPIO_init (GPIO_t *gpio, uint32_t *base_address, uint8_t width, uint8_t enable_offset, uint8_t write_offset, uint8_t read_offset)

  Inizializza un device GPIO.
- void GPIO_setMode (GPIO_t *gpio, GPIO_mask mask, GPIO_mode mode)

  Permette di settare la modalita' lettura/scrittura dei pin di un device GPIO;.
- void GPIO_setValue (GPIO_t *gpio, GPIO_mask mask, GPIO_value value)

  Permette di settare il valore dei pin di un device GPIO;.
- void GPIO_toggle (GPIO_t *gpio, GPIO_mask mask)

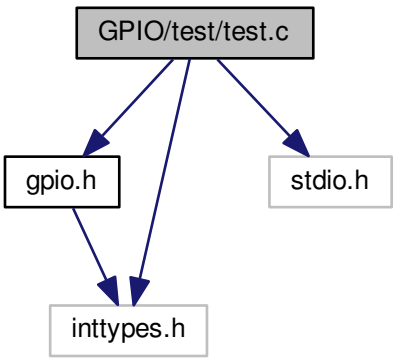
  Permette di invertire il valore dei pin di un device GPIO;.
- GPIO_value GPIO_getValue (GPIO_t *gpio, GPIO_mask mask)

  Permette di leggere il valore dei pin di un device GPIO;.

6.3 Riferimenti per il file GPIO/test/test.c

```
#include "gpio.h"
#include <inttypes.h>
#include <stdio.h>
```

Grafo delle dipendenze di inclusione per test.c:



Funzioni

- int main ()

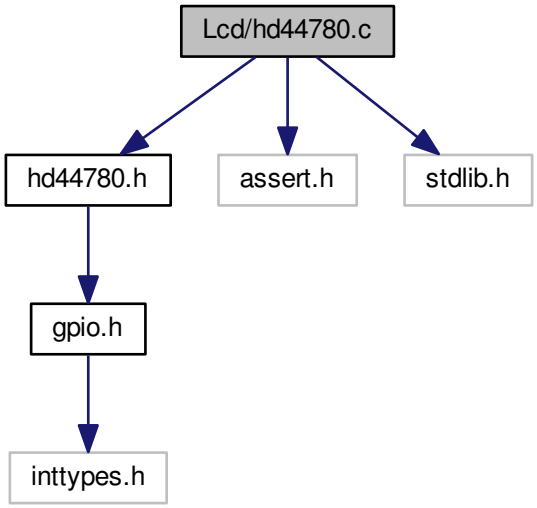
6.3.1 Documentazione delle funzioni

6.3.1.1 int main ( )

6.4 Riferimenti per il file Lcd/hd44780.c

```
#include "hd44780.h"
#include <assert.h>
#include <stdlib.h>
```

Grafo delle dipendenze di inclusione per hd44780.c:



Definizioni

- #define HD44780_clear 0x01
- #define HD44780_home 0x02
- #define HD44780_row1 0x80
- #define HD44780_row2 0xC0
- #define HD44780_cursor_r 0x14
- #define HD44780_cursor_l 0x10
- #define HD44780_display_off 0x08
- #define HD44780_cursor_off 0x0C
- #define HD44780_cursor_on 0x0E
- #define HD44780_cursor_blink 0x0F
- #define HD44780_clear 0x01
- #define HD44780_dec_no_shift 0x04
- #define HD44780_dec_shift 0x05
- #define HD44780_inc_no_shift 0x06
- #define HD44780_inc_shift 0x07
- #define timer_wait_ms(ms) usleep(ms<<10)
- #define timer_wait_us(us) usleep(us)
- #define lcd_command(lcd) GPIO_setValue(lcd->gpio, lcd->RS, GPIO_reset)

- #define lcd_data(lcd) GPIO_setValue(lcd->gpio, lcd->RS, GPIO_set)
- #define lcd_write(lcd) GPIO_setValue(lcd->gpio, lcd->RW, GPIO_reset)
- #define lcd_read(lcd) GPIO_setValue(lcd->gpio, lcd->RW, GPIO_set)
- #define lcd_enable(lcd)

Funzioni

- void HD44780_SetByte (HD44780_LCD_t *lcd, uint8_t byte)
- void HD44780_WriteCommand (HD44780_LCD_t *lcd, uint8_t command)
- void HD44780_WriteData (HD44780_LCD_t *lcd, uint8_t data)
- int HD44780_ValidatePair (HD44780_LCD_t *lcd)
- void HD44780_ConfigurePin (HD44780_LCD_t *lcd)
- void HD44780_Init8 (HD44780_LCD_t *lcd, GPIO_t *gpio, GPIO_mask RS, GPIO_mask RW, GPIO_mask E, GPIO_mask Data7, GPIO_mask Data6, GPIO_mask Data5, GPIO_mask Data4, GPIO_mask Data3, GPIO_mask Data2, GPIO_mask Data1, GPIO_mask Data0)

  Inizializza un display lcd HD44780 con interfacciamento ad 8 bit.
- void HD44780_Init4 (HD44780_LCD_t *lcd, GPIO_t *gpio, GPIO_mask RS, GPIO_mask RW, GPIO_mask E, GPIO_mask Data7, GPIO_mask Data6, GPIO_mask Data5, GPIO_mask Data4)

  Inizializza un oggetto display lcd HD44780 affinche' si utilizzi l'interfaccia a 4 bit.
- void HD44780_Printc (HD44780_LCD_t *lcd, char c)

  Stampa un carattere.
- void HD44780_Print (HD44780_LCD_t *lcd, const char *s)

  Stampa una stringa null-terminated di caratteri.
- void HD44780_printBinary8 (HD44780_LCD_t *lcd, uint8_t b)

  Stampa un byte in binario. (bit piu' significativo a sinistra)
- void HD44780_printBinary32 (HD44780_LCD_t *lcd, uint32_t w)

  Stampa una word di 32 bit in binario. (bit piu' significativo a sinistra)
- void HD44780_printBinary64 (HD44780_LCD_t *lcd, uint64_t b)

  Stampa un blocco di 64 bit in binario. (bit piu' significativo a sinistra)
- void HD44780_printHex8 (HD44780_LCD_t *lcd, uint8_t b)

  Stampa un byte in esadecimale. (bit piu' significativo a sinistra)
- void HD44780_printHex32 (HD44780_LCD_t *lcd, uint32_t w)

  Stampa una word di 32 bit in esadecimale. (bit piu' significativo a sinistra)
- void HD44780_printHex64 (HD44780_LCD_t *lcd, uint64_t b)

  Stampa un blocco di 64 bit in esadecimale. (bit piu' significativo a sinistra)
- void HD44780_Clear (HD44780_LCD_t *lcd)

  Pulisce il display e sposta il cursore all'inizio della prima riga.
- void HD44780_Home (HD44780_LCD_t *lcd)

  Sposta il cursore all'inizio della prima riga.
- void HD44780_MoveToRow1 (HD44780_LCD_t *lcd)

  Sposta il cursore all'inizio della prima riga.
- void HD44780_MoveToRow2 (HD44780_LCD_t *lcd)

  Sposta il cursore all'inizio della seconda riga.
- void HD44780_MoveCursor (HD44780_LCD_t *lcd, HD44780_Direction_t dir)

  Sposta il cursore di una posizione a destra o sinistra.
- void HD44780_DisplayOff (HD44780_LCD_t *lcd)

  Disattiva il display.
- void HD44780_CursorOff (HD44780_LCD_t *lcd)

  Disattiva la visualizzazione del cursore.
- void HD44780_CursorOn (HD44780_LCD_t *lcd)

  Attiva la visualizzazione del cursore.
- void HD44780_CursorBlink (HD44780_LCD_t *lcd)

  Attiva il cursore lampeggiante.

6.4.1 Documentazione delle definizioni

6.4.1.1 #define HD44780_clear 0x01

6.4.1.2 #define HD44780_clear 0x01

6.4.1.3 #define HD44780_cursor_blink 0x0F

6.4.1.4 #define HD44780_cursor_l 0x10

6.4.1.5 #define HD44780_cursor_off 0x0C

6.4.1.6 #define HD44780_cursor_on 0x0E

6.4.1.7 #define HD44780_cursor_r 0x14

6.4.1.8 #define HD44780_dec_no_shift 0x04

6.4.1.9 #define HD44780_dec_shift 0x05

6.4.1.10 #define HD44780_display_off 0x08

6.4.1.11 #define HD44780_home 0x02

6.4.1.12 #define HD44780_inc_no_shift 0x06

6.4.1.13 #define HD44780_inc_shift 0x07

6.4.1.14 #define HD44780_row1 0x80

6.4.1.15 #define HD44780_row2 0xC0

6.4.1.16 #define lcd_command(*lcd*) GPIO_setValue(lcd->gpio, lcd->RS, GPIO_reset)

6.4.1.17 #define lcd_data(*lcd*) GPIO_setValue(lcd->gpio, lcd->RS, GPIO_set)

6.4.1.18 #define lcd_enable(*lcd*)

Valore:

```
GPIO_setValue(lcd->gpio, lcd->E, GPIO_set); \
                          timer_wait_us(100); \
                          GPIO_setValue(lcd->gpio, lcd->E,
      GPIO_reset)
```

6.4.1.19 #define lcd_read(*lcd*) GPIO_setValue(lcd->gpio, lcd->RW, GPIO_set)

6.4.1.20 #define lcd_write(*lcd*) GPIO_setValue(lcd->gpio, lcd->RW, GPIO_reset)

6.4.1.21 #define timer_wait_ms(*ms*) usleep(ms<<10)

6.4.1.22 #define timer_wait_us(*us*) usleep(us)

6.4.2 Documentazione delle funzioni

6.4.2.1 void HD44780_ConfigurePin (HD44780_LCD_t * *lcd*)

6.4.2.2 void HD44780_SetByte (HD44780_LCD_t * *lcd*, uint8_t *byte*)

6.4.2.3 int HD44780_ValidatePair (HD44780_LCD_t * *lcd*)

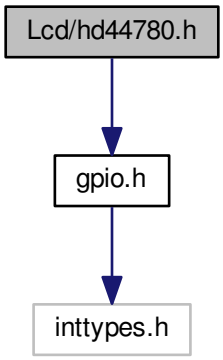
6.4.2.4 void HD44780_WriteCommand (HD44780_LCD_t * *lcd*, uint8_t *command*)

6.4.2.5 void HD44780_WriteData (HD44780_LCD_t * *lcd*, uint8_t *data*)

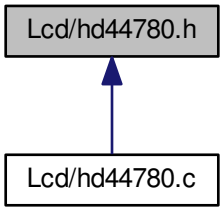
6.5 Riferimenti per il file Lcd/hd44780.h

```
#include "gpio.h"
```

Grafo delle dipendenze di inclusione per hd44780.h:



Questo grafo mostra quali altri file includono direttamente o indirettamente questo file:



Strutture dati

- struct HD44780_LCD_t

  Struttura opaca che astrae un device Display LCD con cntroller Hitachi HD44780, o compatibile. Un oggetto di tipo HD44780_LCD_t rappresenta un device lcd HD44780. Il modulo e' pensato per permettere la gestione di piu' display da parte dello stesso processore, agendo su oggetti HD44780_LCD_t diversi. Il modulo permette di utilizzare sia

l'interfacciamento ad otto bit che quello a quattro bit, inizializzando il device opportunamente, attraverso l'uso delle funzioni HD44780_Init8 eHD44780_Init4. Il modulo fornisce anche semplici funzioni per la stampa di un carattere o di una stringa null-terminated di caratteri. Si veda la documentazione delle funzioni HD44780_Printc e HD44780_Print. Inoltre sono presenti diverse funzioni di utilita' generica, come quelle per la pulizia del display, per lo spostamento del cursore di un posto in avanti o indietro, alla riga in basso o in alto.

Tipi enumerati (enum)

- enum HD44780_InterfaceMode_t { HD44780_INTERFACE_4bit, HD44780_INTERFACE_8bit }
- enum HD44780_Direction_t { HD44780_CursorLeft, HD44780_CursorRight }

  Direzioni di spostamento del cursore.

Funzioni

- void HD44780_Init8 (HD44780_LCD_t *lcd, GPIO_t *gpio, GPIO_mask RS, GPIO_mask RW, GPIO_mask E, GPIO_mask Data7, GPIO_mask Data6, GPIO_mask Data5, GPIO_mask Data4, GPIO_mask Data3, GPIO_mask Data2, GPIO_mask Data1, GPIO_mask Data0)

  Inizializza un display lcd HD44780 con interfacciamento ad 8 bit.
- void HD44780_Init4 (HD44780_LCD_t *lcd, GPIO_t *gpio, GPIO_mask RS, GPIO_mask RW, GPIO_mask E, GPIO_mask Data7, GPIO_mask Data6, GPIO_mask Data5, GPIO_mask Data4)

  Inizializza un oggetto display lcd HD44780 affinche' si utilizzi l'interfaccia a 4 bit.
- void HD44780_Printc (HD44780_LCD_t *lcd, char c)

  Stampa un carattere.
- void HD44780_Print (HD44780_LCD_t *lcd, const char *s)

  Stampa una stringa null-terminated di caratteri.
- void HD44780_printBinary8 (HD44780_LCD_t *lcd, uint8_t b)

  Stampa un byte in binario. (bit piu' significativo a sinistra)
- void HD44780_printBinary32 (HD44780_LCD_t *lcd, uint32_t w)

  Stampa una word di 32 bit in binario. (bit piu' significativo a sinistra)
- void HD44780_printBinary64 (HD44780_LCD_t *lcd, uint64_t b)

  Stampa un blocco di 64 bit in binario. (bit piu' significativo a sinistra)
- void HD44780_printHex8 (HD44780_LCD_t *lcd, uint8_t b)

  Stampa un byte in esadecimale. (bit piu' significativo a sinistra)
- void HD44780_printHex32 (HD44780_LCD_t *lcd, uint32_t w)

  Stampa una word di 32 bit in esadecimale. (bit piu' significativo a sinistra)
- void HD44780_printHex64 (HD44780_LCD_t *lcd, uint64_t b)

  Stampa un blocco di 64 bit in esadecimale. (bit piu' significativo a sinistra)
- void HD44780_Clear (HD44780_LCD_t *lcd)

  Pulisce il display e sposta il cursore all'inizio della prima riga.
- void HD44780_Home (HD44780_LCD_t *lcd)

  Sposta il cursore all'inizio della prima riga.
- void HD44780_MoveToRow1 (HD44780_LCD_t *lcd)

  Sposta il cursore all'inizio della prima riga.
- void HD44780_MoveToRow2 (HD44780_LCD_t *lcd)

  Sposta il cursore all'inizio della seconda riga.
- void HD44780_MoveCursor (HD44780_LCD_t *lcd, HD44780_Direction_t dir)

  Sposta il cursore di una posizione a destra o sinistra.
- void HD44780_DisplayOff (HD44780_LCD_t *lcd)

  Disattiva il display.
- void HD44780_CursorOff (HD44780_LCD_t *lcd)

  Disattiva la visualizzazione del cursore.

- void HD44780_CursorOn (HD44780_LCD_t *lcd)

  Attiva la visualizzazione del cursore.

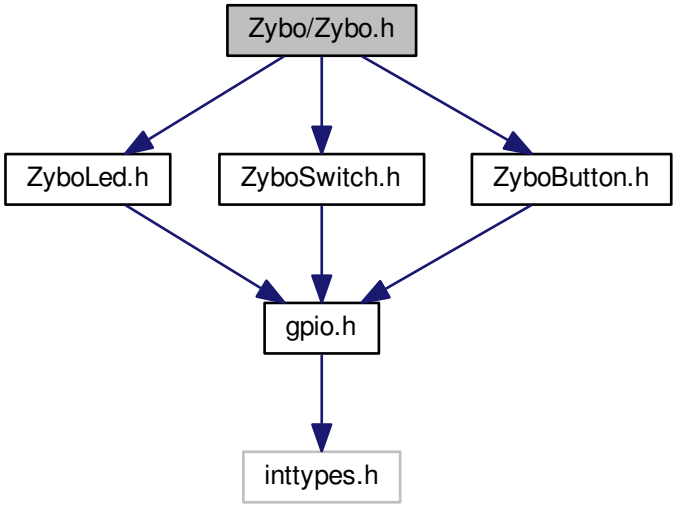
- void HD44780_CursorBlink (HD44780_LCD_t *lcd)

  Attiva il cursore lampeggiante.

6.6 Riferimenti per il file Zybo/Zybo.h

```
#include "ZyboLed.h"
#include "ZyboSwitch.h"
#include "ZyboButton.h"
```

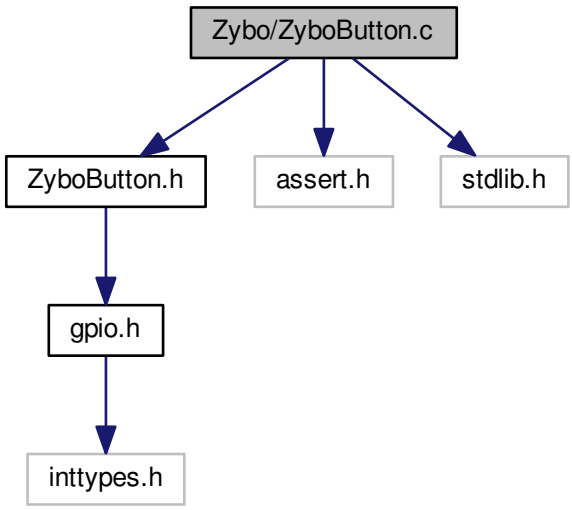
Grafo delle dipendenze di inclusione per Zybo.h:



6.7 Riferimenti per il file Zybo/ZyboButton.c

```
#include "ZyboButton.h"
#include <assert.h>
#include <stdlib.h>
```

Grafo delle dipendenze di inclusione per ZyboButton.c:



Definizioni

- #define timer_wait_ms(ms) usleep(ms<<10)

Funzioni

- void ZyboButton_init (ZyboButton_t *buttons, GPIO_t *gpio, GPIO_mask Button3_pin, GPIO_mask Button2_pin, GPIO_mask Button1_pin, GPIO_mask Button0_pin)

  Inizializza un oggetto di tipo ZyboButton_t.

- void ZyboButton_waitWhileIdle (ZyboButton_t *buttons)

  Permettere di mettere il programma in attesa attiva finche' i button restano inattivi;.

- void ZyboButton_waitWhileBusy (ZyboButton_t *buttons)

  Permettere di mettere il programma in attesa attiva finche' i button restano attivi;.

- ZyboButton_status_t ZyboButton_getStatus (ZyboButton_t *buttons, ZyboButton_mask_t mask)

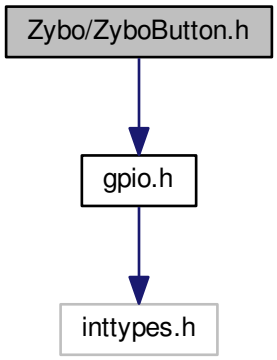
  Permette la lettura dello stato dei button presenti sulla board.

6.7.1 Documentazione delle definizioni

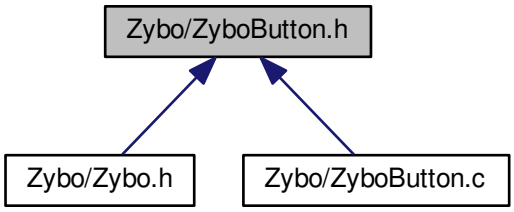
6.7.1.1 #define timer_wait_ms(*ms*) usleep(ms<<10)

6.8 Riferimenti per il file Zybo/ZyboButton.h

`#include "gpio.h"`
Grafo delle dipendenze di inclusione per ZyboButton.h:



Questo grafo mostra quali altri file includono direttamente o indirettamente questo file:



Strutture dati

- struct ZyboButton_t

  Struttura opaca che astrae l'insieme dei button presenti sulla board Digilent Zybo;.

Definizioni

- #define ZyboButton(i) ((uint32_t)(1<<i))

  Metodo alternativo per la specifica di uno dei button presenti sulla board Digilent Zybo.
- #define ZyboButton_DebounceWait 50

  Tempo di attesa (in millisecondi) usato per prevenire il fenomeno del bouncing. Il valore di default è 50, determinato empiricamente. Puo' essere modificato a piacimento cambiando il valore alla macro seguente.

Tipi enumerati (enum)

- enum ZyboButton_mask_t { ZyboButton3 = 0x8, ZyboButton2 = 0x4, ZyboButton1 = 0x2, ZyboButton0 = 0x1 }

  Maschere di selezione dei PushButton.
- enum ZyboButton_status_t { ZyboButton_off, ZyboButton_on }

  Status di attivo/inattivo dei PushButton.

Funzioni

- void ZyboButton_init (ZyboButton_t *buttons, GPIO_t *gpio, GPIO_mask Button3_pin, GPIO_mask Button2_pin, GPIO_mask Button1_pin, GPIO_mask Button0_pin)

  Inizializza un oggetto di tipo ZyboButton_t.
- void ZyboButton_waitWhileIdle (ZyboButton_t *buttons)

  Permettere di mettere il programma in attesa attiva finche' i button restano inattivi;.
- void ZyboButton_waitWhileBusy (ZyboButton_t *buttons)

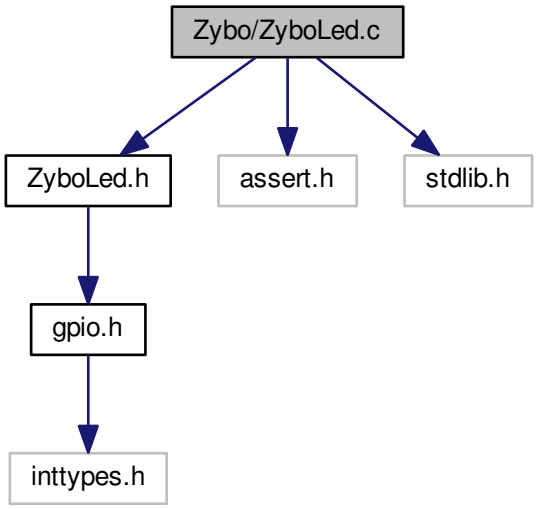
  Permettere di mettere il programma in attesa attiva finche' i button restano attivi;.
- ZyboButton_status_t ZyboButton_getStatus (ZyboButton_t *buttons, ZyboButton_mask_t mask)

  Permette la lettura dello stato dei button presenti sulla board.

6.9 Riferimenti per il file Zybo/ZyboLed.c

```
#include "ZyboLed.h"
#include <assert.h>
#include <stdlib.h>
```

Grafo delle dipendenze di inclusione per ZyboLed.c:



Funzioni

- void ZyboLed_init (ZyboLed_t *leds, GPIO_t *gpio, GPIO_mask Led3_pin, GPIO_mask Led2_pin, GPIO_mask Led1_pin, GPIO_mask Led0_pin)

  Inizializza un oggetto di tipo ZyboLed_t.
- void ZyboLed_setStatus (ZyboLed_t *leds, ZyboLed_mask_t mask, ZyboLed_status_t status)

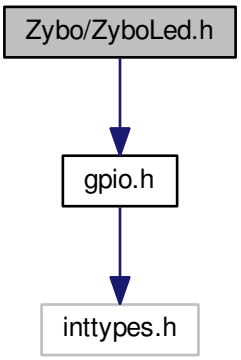
  Permette di accendere/spegnere i Led sulla board.
- void ZyboLed_toggle (ZyboLed_t *leds, ZyboLed_mask_t mask)

  Permette di accendere/spegnere i Led sulla board, invertendone il valore.

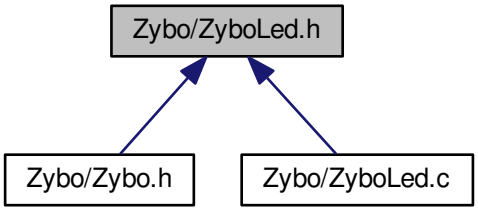
6.10 Riferimenti per il file Zybo/ZyboLed.h

```
#include "gpio.h"
```

Grafo delle dipendenze di inclusione per ZyboLed.h:



Questo grafo mostra quali altri file includono direttamente o indirettamente questo file:



Strutture dati

- struct ZyboLed_t

Struttura opaca che astrae l'insieme dei Led presenti sulla board Digilent Zybo;.

Definizioni

- #define ZyboLed(i) ((uint32_t)(1<<i))

  Metodo alternativo per la specifica di uno dei led presenti sulla board Digilent Zybo.

Tipi enumerati (enum)

- enum ZyboLed_mask_t { ZyboLed3 = 0x8, ZyboLed2 = 0x4, ZyboLed1 = 0x2, ZyboLed0 = 0x1 }

  Maschere di selezione dei led.

- enum ZyboLed_status_t { ZyboLed_off, ZyboLed_on }

  Status di accensione/spegnimento dei led.

Funzioni

- void ZyboLed_init (ZyboLed_t *leds, GPIO_t *gpio, GPIO_mask Led3_pin, GPIO_mask Led2_pin, GPIO_mask Led1_pin, GPIO_mask Led0_pin)

  Inizializza un oggetto di tipo ZyboLed_t.

- void ZyboLed_setStatus (ZyboLed_t *leds, ZyboLed_mask_t mask, ZyboLed_status_t status)

  Permette di accendere/spegnere i Led sulla board.

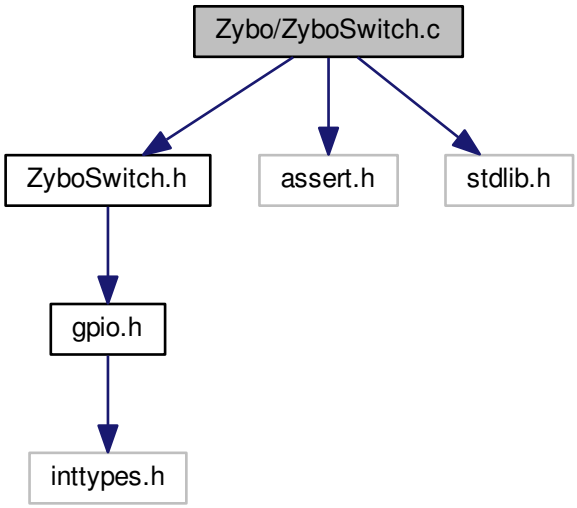
- void ZyboLed_toggle (ZyboLed_t *leds, ZyboLed_mask_t mask)

  Permette di accendere/spegnere i Led sulla board, invertendone il valore.

6.11 Riferimenti per il file Zybo/ZyboSwitch.c

```
#include "ZyboSwitch.h"
#include <assert.h>
#include <stdlib.h>
```

Grafo delle dipendenze di inclusione per ZyboSwitch.c:



Funzioni

- void ZyboSwitch_init (ZyboSwitch_t *switches, GPIO_t *gpio, GPIO_mask Switch3_pin, GPIO_mask Switch2_pin, GPIO_mask Switch1_pin, GPIO_mask Switch0_pin)

  Inizializza un oggetto di tipo ZyboSwitch_t.

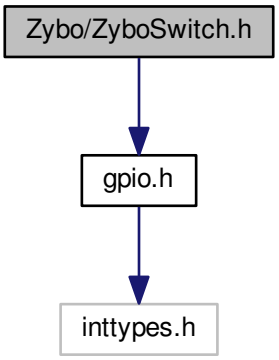
- ZyboSwitch_status_t ZyboSwitch_getStatus (ZyboSwitch_t *switches, ZyboSwitch_mask_t mask)

  Permette la lettura dello stato degli switch presenti sulla board.

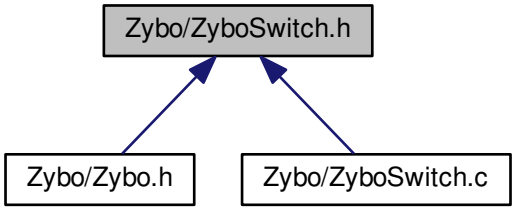
6.12 Riferimenti per il file Zybo/ZyboSwitch.h

```
#include "gpio.h"
```

Grafo delle dipendenze di inclusione per ZyboSwitch.h:



Questo grafo mostra quali altri file includono direttamente o indirettamente questo file:



Strutture dati

- struct ZyboSwitch_t

  Struttura opaca che astrae l'insieme degli switch presenti sulla board Digilent Zybo;.

Definizioni

- #define ZyboSwitch(i) ((uint32_t)(1<<i))

  Metodo alternativo per la specifica di uno degli switch presenti sulla board Digilent Zybo.

Tipi enumerati (enum)

- enum ZyboSwitch_mask_t { ZyboSwitch3 = 0x8, ZyboSwitch2 = 0x4, ZyboSwitch1 = 0x2, ZyboSwitch0 = 0x1 }

  Maschere di selezione degli switch.
- enum ZyboSwitch_status_t { ZyboSwitch_off, ZyboSwitch_on }

  Status di attivo/inattivo degli switch.

Funzioni

- void ZyboSwitch_init (ZyboSwitch_t *switches, GPIO_t *gpio, GPIO_mask Switch3_pin, GPIO_mask Switch2_pin, GPIO_mask Switch1_pin, GPIO_mask Switch0_pin)

  Inizializza un oggetto di tipo ZyboSwitch_t.
- ZyboSwitch_status_t ZyboSwitch_getStatus (ZyboSwitch_t *switches, ZyboSwitch_mask_t mask)

  Permette la lettura dello stato degli switch presenti sulla board.

Indice analitico