Codice Sistemi Embedded

Generated by Doxygen 1.8.13

Contents

Chapter 1

Documentazione codice sistemi embedded

Table of Contents

1.2.2 Hardware

- Controlla la generazione dell' interrupt UART_v1_0_S00_AXI.vhd
- Top level entity del componente UART_v1_0_S00_AXI UART_v1_0.vhd

1.3 CRC

@ F3

- gestione dell' invio, calcolo e check del CRC main.c

Chapter 2

GPIO

Permette di avere una serie di GPIO sotto lo stesso device.

Inizializza il driver kernel ed espone le funzionalità del modulo.

Funzioni utilizzate per interagire con la singola entità GPIO permette la gestione dell' interrupt

Inizializza il driver kernel ed espone le funzionalità del modulo

Permette di avere una serie di GPIO sotto lo stesso device

permette la gestione del GPIO utilizzando un driver di tipo UIO

Chapter 3

Design Unit Index

3.1 Design Unit Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 4

Design Unit Index

4.1 Design Unit List

Here is a list of all design unit members with links to the Entities they belong to:

Chapter 5

File Index

5.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 6

Data Structure Documentation

6.1 arch_imp Architecture Reference

Components

- GPIO_v1_0_S00_AXI

Instantiations

- gpio_v1_0_s00_axi_inst **GPIO_v1_0_S00_AXI**

The documentation for this class was generated from the following file:

- /media/saverio/OS/Users/Saverio/Desktop/SE/git/codici_da_mandare/FPGA/GPIO/Hardware/GPIO_1.↩ 0/hdl/GPIO_v1_0.vhd

6.2 arch_imp Architecture Reference

Processes

- PROCESS_0(**S_AXI_ACLK**)
- PROCESS_1(**S_AXI_ACLK**)
- PROCESS_2(**S_AXI_ACLK**)
- PROCESS_3(**S_AXI_ACLK**)
- PROCESS_4(**S_AXI_ACLK**)
- PROCESS_5(**S_AXI_ACLK**)
- PROCESS_6(**S_AXI_ACLK**)
- PROCESS_7(**slv_reg0 , slv_reg1 , gpio_read , slv_reg3 , slv_reg4 , slv_reg5 , status_reg_out , slv_**↩ **reg7_out , axi_araddr , S_AXI_ARESETN , slv_reg_rden**)
- PROCESS_8(**S_AXI_ACLK**)
- gpio_read_sampling(**S_AXI_ACLK , gpio_read**)

  Campiona i segnali di cui si vuole verificare la generazione di un interrupt.
- intr_pending(**S_AXI_ACLK , change_detected , ack_intr**)

  Gestisce il registro pending.
- inst_irq(**S_AXI_ACLK , pending_intr**)

Components

- GPIO_Array

Constants

- ADDR_LSB **integer:=(C_S_AXI_DATA_WIDTH/ 32)+ 1**
- OPT_MEM_ADDR_BITS **integer:= 2**

Signals

- axi_awaddr **std_logic_vector(C_S_AXI_ADDR_WIDTH- 1 downto 0)**
- axi_awready **std_logic**
- axi_wready **std_logic**
- axi_bresp **std_logic_vector(1 downto 0)**
- axi_bvalid **std_logic**
- axi_araddr **std_logic_vector(C_S_AXI_ADDR_WIDTH- 1 downto 0)**
- axi_arready **std_logic**
- axi_rdata **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- axi_rresp **std_logic_vector(1 downto 0)**
- axi_rvalid **std_logic**
- slv_reg0 **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- slv_reg1 **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- slv_reg2 **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- slv_reg3 **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- slv_reg4 **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- slv_reg5 **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- slv_reg6 **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- slv_reg7 **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- slv_reg7_out **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- slv_reg_rden **std_logic**
- slv_reg_wren **std_logic**
- reg_data_out **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- byte_index **integer**
- aw_en **std_logic**
- gpio_read **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- status_reg_out **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- status_reg **std_logic_vector(width - 1 downto 0)**

  determina se un segnale ha generato interrupt
- status_reg_tmp **std_logic_vector(width - 1 downto 0)**
- detected_intr **std_logic_vector(width - 1 downto 0)**
- pending_intr **std_logic_vector(width - 1 downto 0)**

  determina se una interruzione è pendente
- pending_intr_tmp **std_logic_vector(width - 1 downto 0)**

  salva interruzioni pendenti precedenti
- changed_bits **std_logic_vector(width - 1 downto 0)**

  identifica quale pin del GPIO sono abilitati descrive se è stato campionanto un interrupt
- last_stage **std_logic_vector(width - 1 downto 0)**

  registro primario del campionatore
- current_stage **std_logic_vector(width - 1 downto 0)**

  registro secondario del campionatore
- change_detected **std_logic**

  determina se è avvenuta una condizione che ha generato un interrupt

Instantiations

- inst_gpio_array **gpio_array**

Aliases

- global_intr **std_logicisslv_reg3(0)**
- intr_mask **std_logic_vector(width - 1 downto 0)isslv_reg4(width - 1 downto 0)**

  determina se le interruzioni globali sono attive
- ack_intr **std_logic_vector(width - 1 downto 0)isslv_reg7(width - 1 downto 0)**

  determina quali interrupt sono attivi
- gpio_enable **std_logic_vector(width - 1 downto 0)isslv_reg0(width - 1 downto 0)**

  determina quali segnali di interrupt pendenti sono stati catturati dal driver

6.2.1 Member Function Documentation

6.2.1.1 gpio_read_sampling()

```
gpio_read_sampling (
              S_AXI_ACLK,
              gpio_read )
```

Campiona i segnali di cui si vuole verificare la generazione di un interrupt.

Parameters

in	*S_AXI_ACLK*	clock del bus AXI
in	*gpio_read*	valori del GPIO da campionare

6.2.1.2 inst_irq()

```
inst_irq(
              S_AXI_ACLK  ,
              pending_intr   )  [Process]
```

Per la descrizione del componente riferirsi alla documentazione dell' intero design

6.2.1.3 intr_pending()

```
intr_pending(
              S_AXI_ACLK  ,
              change_detected   ,
              ack_intr   )  [Process]
```

Gestisce il registro pending.

Per la descrizione del componente riferirsi alla documentazione dell' intero design

Parameters

in	*S_AXI_ACLK*	clock del bus AXI
in	*change_detected*	identifica l' avvenimento dell' interruput su un segnale abilitato
in	*ack_intr*	cattura un segnale di ack generato dal driver che gestisce l' eccezione

The documentation for this class was generated from the following file:

- /media/saverio/OS/Users/Saverio/Desktop/SE/git/codici_da_mandare/FPGA/GPIO/Hardware/GPIO_1.↩ 0/hdl/GPIO_v1_0_S00_AXI.vhd

6.3 arch_imp Architecture Reference

Processes

- PROCESS_9**(S_AXI_ACLK)**

  dato ricevuto
- PROCESS_10**(S_AXI_ACLK)**

  segnale il cui valore alto indica che un nuovo dato ricevuto è dispobile
- PROCESS_11**(S_AXI_ACLK)**
- PROCESS_12**(S_AXI_ACLK)**
- PROCESS_13**(S_AXI_ACLK)**
- PROCESS_14**(S_AXI_ACLK)**
- PROCESS_15**(S_AXI_ACLK)**
- PROCESS_16**(slv_reg0 , slv_reg1 , uart_status_reg , slv_reg3_out , slv_reg4 , slv_reg5 , slv_reg6 , slv_reg7_out , axi_araddr , S_AXI_ARESETN , slv_reg_rden)**
- PROCESS_17**(S_AXI_ACLK)**
- status_reg_sampling**(S_AXI_ACLK , uart_status_reg)**

  Campiona i segnali di cui si vuole verificare la generazione di un interrupt.
- intr_pending**(S_AXI_ACLK , change_detected , ack_intr)**

  Gestisce il registro pending.
- inst_irq**(S_AXI_ACLK , pending_intr)**

  Disabilita l' interrupt nel caso di reset del bus e tiene alto il segnale di interrupt finchè rimane pendente.

Components

- UART

  UART.

Constants

- ADDR_LSB **integer:=(C_S_AXI_DATA_WIDTH/ 32)+ 1**
- OPT_MEM_ADDR_BITS **integer:= 2**

Signals

- axi_awaddr **std_logic_vector(C_S_AXI_ADDR_WIDTH- 1 downto 0)**
- axi_awready **std_logic**
- axi_wready **std_logic**
- axi_bresp **std_logic_vector(1 downto 0)**
- axi_bvalid **std_logic**
- axi_araddr **std_logic_vector(C_S_AXI_ADDR_WIDTH- 1 downto 0)**
- axi_arready **std_logic**
- axi_rdata **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- axi_rresp **std_logic_vector(1 downto 0)**
- axi_rvalid **std_logic**
- slv_reg0 **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- slv_reg1 **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- slv_reg2 **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- slv_reg3 **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- slv_reg4 **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- slv_reg5 **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- slv_reg6 **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0):=(others=>' 0 ')**
- slv_reg7 **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- slv_reg_rden **std_logic**
- slv_reg_wren **std_logic**
- reg_data_out **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- byte_index **integer**
- aw_en **std_logic**
- uart_status_reg **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- slv_reg3_out **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- slv_reg7_out **std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- reset **std_logic**
- pending_intr **std_logic_vector(1 downto 0)**

  interruzioni pendenti
- pending_intr_tmp **std_logic_vector(1 downto 0)**

  delay intr_pending
- changed_bits **std_logic_vector(1 downto 0)**
- tx_busy_falling_detect **std_logic**

  vale 1 quando viene rilevato il falling_edge di tx_busy
- rx_rising_detect **std_logic**

  alto quando viene rilevato il rising_edge di RDA
- last_stage **std_logic_vector(1 downto 0)**
- current_stage **std_logic_vector(1 downto 0)**
- change_detected **std_logic**

Instantiations

- inst_uart **uart**

Aliases

- global_intr **std_logicisslv_reg4(0)**
- intr_mask **std_logic_vector(1 downto 0)isslv_reg5(1 downto 0)**

  enable interruzioni IP CORE
- ack_intr **std_logic_vector(1 downto 0)isslv_reg7(1 downto 0)**

6.3.1 Member Function Documentation

6.3.1.1 inst_irq()

```
inst_irq(
            S_AXI_ACLK  ,
            pending_intr   )  [Process]
```

Disabilita l' interrupt nel caso di reset del bus e tiene alto il segnale di interrupt finchè rimane pendente.

Per la descrizione del componente riferirsi alla documentazione dell' intero design

Parameters

in	*S_AXI_ACLK*	clock del bus AXI
in	*pending_intr*	registro che identifica le interruzioni pendenti

6.3.1.2 intr_pending()

```
intr_pending(
            S_AXI_ACLK  ,
            change_detected  ,
            ack_intr   )  [Process]
```

Gestisce il registro pending.

Per la descrizione del componente riferirsi alla documentazione dell' intero design

Parameters

in	*S_AXI_ACLK*	clock del bus AXI
in	*change_detected*	identifica l' avvenimento dell' interruput su un segnale abilitato
in	*ack_intr*	cattura un segnale di ack generato dal driver che gestisce l' eccezione

6.3.1.3 status_reg_sampling()

```
status_reg_sampling (
            S_AXI_ACLK,
            uart_status_reg )
```

Campiona i segnali di cui si vuole verificare la generazione di un interrupt.

Parameters

in	*S_AXI_ACLK*	clock del bus AXI
in	*uart_status_reg*	valori del UART da campionare

6.3.2 Field Documentation

6.3.2.1 ack_intr

```
ack_intr std_logic_vector( 1 downto 0 )isslv_reg7( 1 downto 0 )   [Alias]
```

maschera interruzioni rda(1) e tx_busy(0). Mettendo il relativo bit ad uno si abilita la lina di interruzione

6.3.2.2 changed_bits

```
changed_bits std_logic_vector( 1 downto 0 )   [Signal]
```

segnale di ack. Il bit 0 da ack all'interuzione della trasmissione, il bit 1 a quello dela ricezione. Logica 1 attiva

6.3.2.3 UART

```
UART    [Component]
```

UART.

componente contenente un ricevitore e un trasmettitore che implementano il protocollo UART. Consulatare documentazione esterna.

The documentation for this class was generated from the following file:

- /media/saverio/OS/Users/Saverio/Desktop/SE/git/Andrea/FPGA/ip_repo/UART_1.0/hdl/UART_v1_0_S00↩_AXI.vhd

6.4 arch_imp Architecture Reference

componente UART_AXI_S00 componente nel quale è incapsulato il componente UART e la logica di gestione delle interruzioni.

Components

- UART_v1_0_S00_AXI

Instantiations

- uart_v1_0_s00_axi_inst **UART_v1_0_S00_AXI**

6.4.1 Detailed Description

componente UART_AXI_S00 componente nel quale è incapsulato il componente UART e la logica di gestione delle interruzioni.

The documentation for this class was generated from the following file:

- /media/saverio/OS/Users/Saverio/Desktop/SE/git/Andrea/FPGA/ip_repo/UART_1.0/hdl/UART_v1_0.vhd

6.5 GPIO Struct Reference

Stuttura che astrae un device GPIO in kernel-mode. Contiene ciò che è necessario al funzionamento del driver.

`#include <GPIO.h>`

Data Fields

6.5.1 Detailed Description

Stuttura che astrae un device GPIO in kernel-mode. Contiene ciò che è necessario al funzionamento del driver.

The documentation for this struct was generated from the following file:

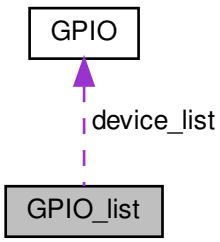
- /media/saverio/OS/Users/Saverio/Desktop/SE/git/codici_da_mandare/FPGA/GPIO/Driver/Kernel_Mode/GPIO.h

6.6 GPIO_list Struct Reference

Struttura dati per la gestione di più device GPIO da parte del driver.

`#include <GPIO_list.h>`

Collaboration diagram for GPIO_list:



Data Fields

6.6.1 Detailed Description

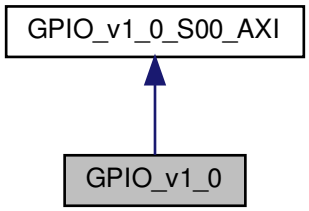
Struttura dati per la gestione di più device GPIO da parte del driver.

The documentation for this struct was generated from the following file:

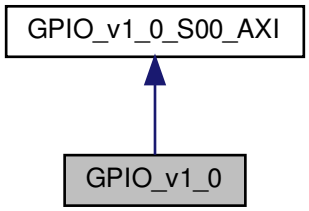
- /media/saverio/OS/Users/Saverio/Desktop/SE/git/codici_da_mandare/FPGA/GPIO/Driver/Kernel_Mode/GPIO_list.h

6.7 GPIO_v1_0 Entity Reference

Inheritance diagram for GPIO_v1_0:



Collaboration diagram for GPIO_v1_0:



Entities

- arch_imp architecture

Libraries

- ieee

  Viene utilizzata la libreria IEEE.

Use Clauses

- std_logic_1164

  Sono utilizzati i segnali della standard logic.
- numeric_std

  Vengono utilizzate le funzioni numeriche.

Generics

- width **integer:=** 4

  determina il numero di GPIO da controllare
- C_S00_AXI_DATA_WIDTH **integer:=** 32
- C_S00_AXI_ADDR_WIDTH **integer:=** 5

Ports

- pads **inout std_logic_vector(width - 1 downto 0)**

  se GPIO in modalità lettura mostra il valore letto, altrimenti forza un valore in uscita
- interrupt **out std_logic**

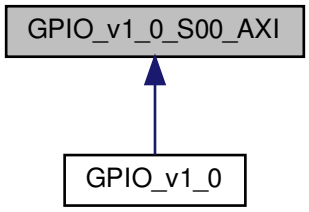
  segnale di interrupt
- s00_axi_aclk **in std_logic**
- s00_axi_aresetn **in std_logic**
- s00_axi_awaddr **in std_logic_vector(C_S00_AXI_ADDR_WIDTH- 1 downto 0)**
- s00_axi_awprot **in std_logic_vector(2 downto 0)**
- s00_axi_awvalid **in std_logic**
- s00_axi_awready **out std_logic**
- s00_axi_wdata **in std_logic_vector(C_S00_AXI_DATA_WIDTH- 1 downto 0)**
- s00_axi_wstrb **in std_logic_vector((C_S00_AXI_DATA_WIDTH/ 8)- 1 downto 0)**
- s00_axi_wvalid **in std_logic**
- s00_axi_wready **out std_logic**
- s00_axi_bresp **out std_logic_vector(1 downto 0)**
- s00_axi_bvalid **out std_logic**
- s00_axi_bready **in std_logic**
- s00_axi_araddr **in std_logic_vector(C_S00_AXI_ADDR_WIDTH- 1 downto 0)**
- s00_axi_arprot **in std_logic_vector(2 downto 0)**
- s00_axi_arvalid **in std_logic**
- s00_axi_arready **out std_logic**
- s00_axi_rdata **out std_logic_vector(C_S00_AXI_DATA_WIDTH- 1 downto 0)**
- s00_axi_rresp **out std_logic_vector(1 downto 0)**
- s00_axi_rvalid **out std_logic**
- s00_axi_rready **in std_logic**

The documentation for this class was generated from the following file:

- /media/saverio/OS/Users/Saverio/Desktop/SE/git/codici_da_mandare/FPGA/GPIO/Hardware/GPIO_1.↩
  0/hdl/GPIO_v1_0.vhd

6.8 GPIO_v1_0_S00_AXI Entity Reference

Inheritance diagram for GPIO_v1_0_S00_AXI:



Entities

- arch_imp architecture

Libraries

- ieee

  Viene utilizzato la libreria IEEE.

Use Clauses

- std_logic_1164

  Sono utilizzati i segnali della standard logic.
- numeric_std

  Vengono utilizzate le funzioni numeriche.
- std_logic_misc

  Viene utilizzata la libreria misc di utility.

Generics

- width **integer:=** 4

  determina il numero di GPIO da controllare
- C_S_AXI_DATA_WIDTH **integer:=** 32
- C_S_AXI_ADDR_WIDTH **integer:=** 5

Ports

- pads **inout std_logic_vector(width - 1 downto 0)**

  se GPIO in modalità lettura mostra il valore letto, altrimenti forza un valore in uscita
- interrupt **out std_logic**

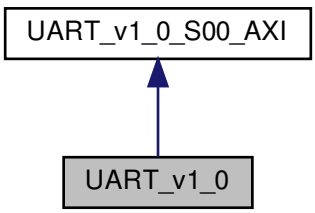
  segnale di interrupt
- S_AXI_ACLK **in std_logic**
- S_AXI_ARESETN **in std_logic**
- S_AXI_AWADDR **in std_logic_vector(C_S_AXI_ADDR_WIDTH- 1 downto 0)**
- S_AXI_AWPROT **in std_logic_vector(2 downto 0)**
- S_AXI_AWVALID **in std_logic**
- S_AXI_AWREADY **out std_logic**
- S_AXI_WDATA **in std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- S_AXI_WSTRB **in std_logic_vector((C_S_AXI_DATA_WIDTH/ 8)- 1 downto 0)**
- S_AXI_WVALID **in std_logic**
- S_AXI_WREADY **out std_logic**
- S_AXI_BRESP **out std_logic_vector(1 downto 0)**
- S_AXI_BVALID **out std_logic**
- S_AXI_BREADY **in std_logic**
- S_AXI_ARADDR **in std_logic_vector(C_S_AXI_ADDR_WIDTH- 1 downto 0)**
- S_AXI_ARPROT **in std_logic_vector(2 downto 0)**
- S_AXI_ARVALID **in std_logic**
- S_AXI_ARREADY **out std_logic**
- S_AXI_RDATA **out std_logic_vector(C_S_AXI_DATA_WIDTH- 1 downto 0)**
- S_AXI_RRESP **out std_logic_vector(1 downto 0)**
- S_AXI_RVALID **out std_logic**
- S_AXI_RREADY **in std_logic**

The documentation for this class was generated from the following file:

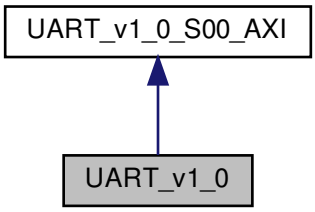
- /media/saverio/OS/Users/Saverio/Desktop/SE/git/codici_da_mandare/FPGA/GPIO/Hardware/GPIO_1.↩ 0/hdl/GPIO_v1_0_S00_AXI.vhd

6.9 UART_v1_0 Entity Reference

Inheritance diagram for UART_v1_0:



Collaboration diagram for UART_v1_0:



Entities

- arch_imp architecture

  componente UART_AXI_S00 componente nel quale è incapsulato il componente UART e la logica di gestione delle interruzioni.

Libraries

- ieee

  Viene utilizzata la libreria IEEE.

Use Clauses

- std_logic_1164

  Sono utilizzati i segnali della standard logic.
- numeric_std

  Vengono utilizzate le funzioni numeriche.

Generics

- baudrate **integer:=** 9600

  baudare trasmissione
- clock_freq **integer:=** 50_000_000

  frequenza clock ingresso
- C_S00_AXI_DATA_WIDTH **integer:=** 32
- C_S00_AXI_ADDR_WIDTH **integer:=** 5

Ports

- tx **out std_logic**

  linea uscita per la trasmissione
- rx **in std_logic**

  linea ingresso per la ricezione
- interrupt **out std_logic**

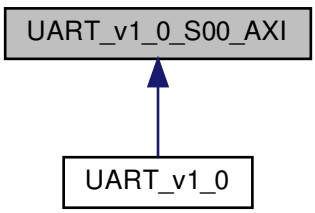
  segnale per richiede l'interrupt
- s00_axi_aclk **in std_logic**
- s00_axi_aresetn **in std_logic**
- s00_axi_awaddr **in std_logic_vector(C_S00_AXI_ADDR_WIDTH- 1 downto 0)**
- s00_axi_awprot **in std_logic_vector(2 downto 0)**
- s00_axi_awvalid **in std_logic**
- s00_axi_awready **out std_logic**
- s00_axi_wdata **in std_logic_vector(C_S00_AXI_DATA_WIDTH- 1 downto 0)**
- s00_axi_wstrb **in std_logic_vector((C_S00_AXI_DATA_WIDTH/ 8)- 1 downto 0)**
- s00_axi_wvalid **in std_logic**
- s00_axi_wready **out std_logic**
- s00_axi_bresp **out std_logic_vector(1 downto 0)**
- s00_axi_bvalid **out std_logic**
- s00_axi_bready **in std_logic**
- s00_axi_araddr **in std_logic_vector(C_S00_AXI_ADDR_WIDTH- 1 downto 0)**
- s00_axi_arprot **in std_logic_vector(2 downto 0)**
- s00_axi_arvalid **in std_logic**
- s00_axi_arready **out std_logic**
- s00_axi_rdata **out std_logic_vector(C_S00_AXI_DATA_WIDTH- 1 downto 0)**
- s00_axi_rresp **out std_logic_vector(1 downto 0)**
- s00_axi_rvalid **out std_logic**
- s00_axi_rready **in std_logic**

The documentation for this class was generated from the following file:

- /media/saverio/OS/Users/Saverio/Desktop/SE/git/Andrea/FPGA/ip_repo/UART_1.0/hdl/UART_v1_0.vhd

6.10 UART_v1_0_S00_AXI Entity Reference

Inheritance diagram for UART_v1_0_S00_AXI:



Entities

- arch_imp architecture

Libraries

- ieee

  Viene utilizzata la libreria IEEE.

Use Clauses

- std_logic_1164

  Sono utilizzati i segnali della standard logic.
- numeric_std

  Vengono utilizzate le funzioni numeriche.
- std_logic_misc

  libreria necessaria per la funzione or_reduce

Generics

- baudrate **integer:=** 9600
- clock_freq **integer:=** 50_000_000
- C_S_AXI_DATA_WIDTH **integer:=** 32
- C_S_AXI_ADDR_WIDTH **integer:=** 5

Ports

- tx **out std_logic**
- rx **in std_logic**
- interrupt **out std_logic**
- S_AXI_ACLK **in std_logic**
- S_AXI_ARESETN **in std_logic**
- S_AXI_AWADDR **in std_logic_vector(C_S_AXI_ADDR_WIDTH-** 1 **downto** 0 **)**
- S_AXI_AWPROT **in std_logic_vector(** 2 **downto** 0 **)**
- S_AXI_AWVALID **in std_logic**
- S_AXI_AWREADY **out std_logic**
- S_AXI_WDATA **in std_logic_vector(C_S_AXI_DATA_WIDTH-** 1 **downto** 0 **)**
- S_AXI_WSTRB **in std_logic_vector((C_S_AXI_DATA_WIDTH/** 8 **)-** 1 **downto** 0 **)**
- S_AXI_WVALID **in std_logic**
- S_AXI_WREADY **out std_logic**
- S_AXI_BRESP **out std_logic_vector(** 1 **downto** 0 **)**
- S_AXI_BVALID **out std_logic**
- S_AXI_BREADY **in std_logic**
- S_AXI_ARADDR **in std_logic_vector(C_S_AXI_ADDR_WIDTH-** 1 **downto** 0 **)**
- S_AXI_ARPROT **in std_logic_vector(** 2 **downto** 0 **)**
- S_AXI_ARVALID **in std_logic**
- S_AXI_ARREADY **out std_logic**
- S_AXI_RDATA **out std_logic_vector(C_S_AXI_DATA_WIDTH-** 1 **downto** 0 **)**
- S_AXI_RRESP **out std_logic_vector(** 1 **downto** 0 **)**
- S_AXI_RVALID **out std_logic**
- S_AXI_RREADY **in std_logic**

The documentation for this class was generated from the following file:

- /media/saverio/OS/Users/Saverio/Desktop/SE/git/Andrea/FPGA/ip_repo/UART_1.0/hdl/UART_v1_0_S00_AXI.vhd

Chapter 7

File Documentation

7.1 /media/saverio/OS/Users/Saverio/Desktop/SE/git/Andrea/FPGA/ip_repo/UART_↩ 1.0/hdl/UART_v1_0.vhd File Reference

UART AXI IPCORE with interrupt.

Entities

- UART_v1_0 entity
- arch_imp architecture

  componente UART_AXI_S00 componente nel quale è incapsulato il componente UART e la logica di gestione delle interruzioni.

7.1.1 Detailed Description

UART AXI IPCORE with interrupt.

7.2 /media/saverio/OS/Users/Saverio/Desktop/SE/git/Andrea/FPGA/ip_repo/UART_↩ 1.0/hdl/UART_v1_0_S00_AXI.vhd File Reference

UART AXI IPCORE with interrupt.

Entities

- UART_v1_0_S00_AXI entity
- arch_imp architecture

7.2.1 Detailed Description

UART AXI IPCORE with interrupt.

7.3 /media/saverio/OS/Users/Saverio/Desktop/SE/git/codici_da_mandare/FPGA/GPIO/↩ Driver/Kernel_Mode/GPIO.c File Reference

7.3.1 Function Documentation

7.3.1.1 GPIO_Destroy()

```
void GPIO_Destroy (
            GPIO* device )
```

Rimuove un device GPIO con le relative strutture kernel allocate per il suo funzionamento.

Parameters

device	puntatore a struttura GPIO che indica l'istanza GPIO da rimuovere

7.3.1.2 GPIO_GetDeviceAddress()

```
void* GPIO_GetDeviceAddress (
            GPIO* device )
```

Restituisce l'indirizzo virtuale di memoria cui è mappato un device.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare

7.3.1.3 GPIO_GetPollMask()

```
unsigned GPIO_GetPollMask (
            GPIO * device,
            struct file * file_ptr,
            struct poll_table_struct * wait )
```

Verifica che le operazioni di lettura risultino non-bloccanti.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare
file	puntatore al descrittore file del device
wait	puntatore alla struttura poll_table

Returns

maschera di bit che indica se sia possibile effettuare operazioni di lettura non bloccanti.

Back-end di tre diverse sys-calls: poll, epoll e select,

7.3.1.4 GPIO_GlobalInterruptDisable()

```
void GPIO_GlobalInterruptDisable (
            GPIO* device )
```

Disabilitazione interrupt globali;.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare

7.3.1.5 GPIO_GlobalInterruptEnable()

```
void GPIO_GlobalInterruptEnable (
            GPIO* device )
```

Abilitazione interrupt globali;.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare

7.3.1.6 GPIO_Init()

```
int GPIO_Init (
            GPIO* GPIO_device,
            struct module * owner,
            struct platform_device * pdev,
            struct class* class,
            const char* driver_name,
            const char* device_name,
            uint32_t serial,
            struct file_operations * f_ops,
            irq_handler_t irq_handler,
            uint32_t irq_mask )
```

Inizializza una struttura GPIO per il corrispondente device.

Parameters

GPIO_device	puntatore a struttura GPIO, corrispondente al device su cui operare
owner	puntatore a struttura struct module, proprietario del device (THIS_MODULE)
pdev	puntatore a struct platform_device
driver_name	nome del driver
device_name	nome del device
serial	numero seriale del device
f_ops	puntatore a struttura struct file_operations, specifica le funzioni che agiscono sul device
irq_handler	puntatore irq_handler_t alla funzione che gestisce gli interrupt generati dal device
irq_mask	maschera delle interruzioni attive del device

Return values

0	se non si è verificato nessun errore

Inizializzazione della wait-queue per la system-call read() e poll()

Inizializzazione degli spinlock

Abilitazione degli interrupt del device

7.3.1.7 GPIO_PendingPinInterrupt()

```
unsigned GPIO_PendingPinInterrupt (
            GPIO* device )
```

Fornisce una maschera che indica quali interrupt non sono ancora stati serviti e che quindi risultano pending.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare

Returns

maschera riportante i pin per i quali gli interrupt non sono stati ancora serviti

7.3.1.8 GPIO_PinInterruptAck()

```
void GPIO_PinInterruptAck (
            GPIO* device,
            unsigned mask )
```

Invia al device notifica di servizio di un interrupt;.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare
mask	maschera di selezione degli interrupt da notificare

7.3.1.9 GPIO_PinInterruptDisable()

```
void GPIO_PinInterruptDisable (
            GPIO* device,
            unsigned mask )
```

Disabilitazione interrupt per i singoli pin del device.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare
mask	maschera di selezione degli interrupt da disabilitare

7.3.1.10 GPIO_PinInterruptEnable()

```
void GPIO_PinInterruptEnable (
            GPIO* device,
            unsigned mask )
```

Abilitazione interrupt per i singoli pin del device.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare
mask	maschera di selezione degli interrupt da abilitare

7.3.1.11 GPIO_ResetCanRead()

```
void GPIO_ResetCanRead (
            GPIO* device )
```

Utilizzata per resettare il flag "can_read" di uno specifico device GPIO.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare

7.3.1.12 GPIO_SetCanRead()

```
void GPIO_SetCanRead (
            GPIO* device )
```

Utilizzata per asserire il flag "can_read" di uno specifico device GPIO.

Parameters

device	puntatore a struttura GPIO, device su cui operare

7.3.1.13 GPIO_TestCanReadAndSleep()

```
void GPIO_TestCanReadAndSleep (
            GPIO* device )
```

Testa il valore del flag "can_read". Se è uguale a 0, ovvero non è possibile effettuare una lettura, mette in sleep il processo.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare

7.3.1.14 GPIO_WakeUp()

```
void GPIO_WakeUp (
            GPIO* device )
```

Risveglia i processi in attesa sulle code di read e poll.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare

7.4 /media/saverio/OS/Users/Saverio/Desktop/SE/git/codici_da_mandare/FPGA/GPIO/↩ Driver/Kernel_Mode/GPIO.h File Reference

Data Structures

- struct GPIO

Stuttura che astrae un device GPIO in kernel-mode. Contiene ciò che è necessario al funzionamento del driver.

7.4.1 Function Documentation

7.4.1.1 GPIO_Destroy()

```
void GPIO_Destroy (
            GPIO* device )
```

Rimuove un device GPIO con le relative strutture kernel allocate per il suo funzionamento.

Parameters

device	puntatore a struttura GPIO che indica l'istanza GPIO da rimuovere

7.4.1.2 GPIO_GetDeviceAddress()

```
void* GPIO_GetDeviceAddress (
            GPIO* device )
```

Restituisce l'indirizzo virtuale di memoria cui è mappato un device.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare

7.4.1.3 GPIO_GetPollMask()

```
unsigned GPIO_GetPollMask (
            GPIO * device,
            struct file * file_ptr,
            struct poll_table_struct * wait )
```

Verifica che le operazioni di lettura risultino non-bloccanti.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare
file	puntatore al descrittore file del device
wait	puntatore alla struttura poll_table

Returns

maschera di bit che indica se sia possibile effettuare operazioni di lettura non bloccanti.

Back-end di tre diverse sys-calls: poll, epoll e select,

7.4.1.4 GPIO_GlobalInterruptDisable()

```
void GPIO_GlobalInterruptDisable (
            GPIO* device )
```

Disabilitazione interrupt globali;.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare

7.4.1.5 GPIO_GlobalInterruptEnable()

```
void GPIO_GlobalInterruptEnable (
            GPIO* device )
```

Abilitazione interrupt globali;.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare

7.4.1.6 GPIO_Init()

```
int GPIO_Init (
            GPIO* GPIO_device,
            struct module * owner,
            struct platform_device * pdev,
            struct class* class,
            const char* driver_name,
            const char* device_name,
            uint32_t serial,
            struct file_operations * f_ops,
            irq_handler_t irq_handler,
            uint32_t irq_mask )
```

Inizializza una struttura GPIO per il corrispondente device.

Parameters

GPIO_device	puntatore a struttura GPIO, corrispondente al device su cui operare
owner	puntatore a struttura struct module, proprietario del device (THIS_MODULE)
pdev	puntatore a struct platform_device
driver_name	nome del driver
device_name	nome del device
serial	numero seriale del device
f_ops	puntatore a struttura struct file_operations, specifica le funzioni che agiscono sul device
irq_handler	puntatore irq_handler_t alla funzione che gestisce gli interrupt generati dal device
irq_mask	maschera delle interruzioni attive del device

Return values

0	se non si è verificato nessun errore

Inizializzazione della wait-queue per la system-call read() e poll()

Inizializzazione degli spinlock

Abilitazione degli interrupt del device

7.4.1.7 GPIO_PendingPinInterrupt()

```
unsigned GPIO_PendingPinInterrupt (
            GPIO* device )
```

Fornisce una maschera che indica quali interrupt non sono ancora stati serviti e che quindi risultano pending.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare

Returns

maschera riportante i pin per i quali gli interrupt non sono stati ancora serviti

7.4.1.8 GPIO_PinInterruptAck()

```
void GPIO_PinInterruptAck (
            GPIO* device,
            unsigned mask )
```

Invia al device notifica di servizio di un interrupt;.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare
mask	maschera di selezione degli interrupt da notificare

7.4.1.9 GPIO_PinInterruptDisable()

```
void GPIO_PinInterruptDisable (
            GPIO* device,
            unsigned mask )
```

Disabilitazione interrupt per i singoli pin del device.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare
mask	maschera di selezione degli interrupt da disabilitare

7.4.1.10 GPIO_PinInterruptEnable()

```
void GPIO_PinInterruptEnable (
            GPIO* device,
            unsigned mask )
```

Abilitazione interrupt per i singoli pin del device.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare
mask	maschera di selezione degli interrupt da abilitare

7.4.1.11 GPIO_ResetCanRead()

```
void GPIO_ResetCanRead (
            GPIO* device )
```

Utilizzata per resettare il flag "can_read" di uno specifico device GPIO.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare

7.4.1.12 GPIO_SetCanRead()

```
void GPIO_SetCanRead (
            GPIO* device )
```

Utilizzata per asserire il flag "can_read" di uno specifico device GPIO.

Parameters

device	puntatore a struttura GPIO, device su cui operare

7.4.1.13 GPIO_TestCanReadAndSleep()

```
void GPIO_TestCanReadAndSleep (
            GPIO* device )
```

Testa il valore del flag "can_read". Se è uguale a 0, ovvero non è possibile effettuare una lettura, mette in sleep il processo.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare

7.4.1.14 GPIO_WakeUp()

```
void GPIO_WakeUp (
            GPIO* device )
```

Risveglia i processi in attesa sulle code di read e poll.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare

7.5 /media/saverio/OS/Users/Saverio/Desktop/SE/git/codici_da_mandare/FPGA/GPIO/↩ Driver/Kernel_Mode/GPIO_kernel_main.c File Reference

7.5.1 Function Documentation

7.5.1.1 GPIO_irq_handler()

```
static irqreturn_t GPIO_irq_handler (
            int irq,
            struct pt_regs * regs )  [static]
```

Interrupt-handler.

Parameters

irq	Interrupt-number a cui il device è connesso
regs	registri sullo stack alla system call entry

Return values

IRQ_HANDLED	dopo aver servito l'interruzione

7.5.1.2 GPIO_llseek()

```
static loff_t GPIO_llseek (
            struct file * file_ptr,
            loff_t off,
            int whence )  [static]
```

Implementa le system-call lseek() e llseek().

Parameters

file_ptr	puntatore al descrittore file del device
off	offset da aggiungere al parametro whence per il posizionamento
whence	può assumere i valori SEEK_SET, SEEK_CUR o SEEK_END per specificare rispettivamente il riferimento dall'inizio file, dalla posizione corrente o dalla fine.

Returns

Nuova posizione della "testina" di lettura/scrittura

7.5.1.3 GPIO_open()

```
static int GPIO_open (
            struct inode * inode,
            struct file * file_ptr )  [static]
```

Invocata all'apertura del file corrispondente al device.

Return values

0	se non si verifica nessun errore

7.5.1.4 GPIO_poll()

```
static unsigned int GPIO_poll (
            struct file * file_ptr,
            struct poll_table_struct * wait )  [static]
```

Verifica che le operazioni di lettura risultino non-bloccanti.

Parameters

device	puntatore a struttura GPIO, che si riferisce al device su cui operare
file_ptr	puntatore al descrittore file del device
wait	puntatore alla struttura poll_table

Returns

maschera di bit che indica se sia possibile effettuare operazioni di lettura non bloccanti.

Back-end di tre diverse sys-calls: poll, epoll e select,

7.5.1.5 GPIO_probe()

```
static int GPIO_probe (
            struct platform_device * pdev )  [static]
```

Inizializzazione del driver

7.5.1.6 GPIO_read()

```
static ssize_t GPIO_read (
            struct file * file_ptr,
            char * buf,
            size_t count,
            loff_t * off )  [static]
```

Legge dati dal device.

Parameters

file_ptr	puntatore al descrittore file del device
buf	puntatore all'area di memoria dove verranno copiati i count bytes letti
count	numeri di bytes da trasferire
off	long offset type che indica la posizione alla quale si sta effettuando l'accesso

Note

l'aggiunta del flag O_NONBLOCK all'apertura del file descriptor associato al device farà sì che il processo chiamante non verrà bloccato se alla chiamata di una lettura non troverà dati disponibili

7.5.1.7 GPIO_release()

```
static int GPIO_release (
            struct inode * inode,
            struct file * file_ptr )  [static]
```

Invocata alla chiusura del file corrispondente al device.

Parameters

inode	struttura dati sul file system che archivia e descrive attributi base su file, directory o qualsiasi altro oggetto
file_ptr	puntatore al descrittore file del device

Return values

0	se non si verifica nessun errore

7.5.1.8 GPIO_remove()

```
static int GPIO_remove (
            struct platform_device * pdev )  [static]
```

Viene chiamata automaticamente alla rimozione del modulo.

Parameters

pdev	

Return values

0	se non si verifica nessun errore

Dealloca tutta la memoria utilizzata dal driver, de-inizializzando il device e disattivando gli interrupt per il device, effettuando tutte le operazioni inverse della funzione GPIO_probe().

7.5.1.9 GPIO_write()

```
static ssize_t GPIO_write (
            struct file * file_ptr,
```

```
            const char * buf,
            size_t size,
            loff_t * off )  [static]
```

Invia dati al device.

Parameters

file_ptr	puntatore al descrittore file del device
buf	puntatore all'area di memoria dalla quale verranno copiati i count bytes
count	numeri di bytes da trasferire
off	long offset type che indica la posizione alla quale si sta effettuando l'accesso

7.5.1.10 module_platform_driver()

```
module_platform_driver (
            GPIO_driver  )
```

la macro module_platform_driver() prende in input la struttura platform_driver ed implementa le funzioni module↩
_init() e module_close() standard, chiamate quando il modulo viene caricato o rimosso dal kernel.

Parameters

GPIO_driver	struttura platform_driver associata al driver

7.5.2 Variable Documentation

7.5.2.1 __test_int_driver_id

```
const struct of_device_id __test_int_driver_id[]  [static]
```

Initial value:

```
={
    {.compatible = "GPIO"},
    {}
}
```

Identifica il device all'interno del device tree.

7.5.2.2 GPIO_driver

```
struct platform_driver GPIO_driver  [static]
```

Initial value:

```
= {
    .driver = {
                .name = DRIVER_NAME,
                .owner = THIS_MODULE,
                .of_match_table = of_match_ptr(__test_int_driver_id),
        },
    .probe = GPIO_probe,
    .remove = GPIO_remove
}
```

Definisce le funzioni probe() e remove() da chiamare al caricamento del driver.

7.5.2.3 GPIO_fops

```
struct file_operations GPIO_fops  [static]
```

Initial value:

```
= {
        .owner      = THIS_MODULE,
        .llseek     = GPIO_llseek,
        .read       = GPIO_read,
        .write      = GPIO_write,
        .poll       = GPIO_poll,
        .open       = GPIO_open,
        .release    = GPIO_release
}
```

Struttura che specifica le funzioni che agiscono sul device.

7.6 /media/saverio/OS/Users/Saverio/Desktop/SE/git/codici_da_mandare/FPGA/GPIO/↩ Driver/Kernel_Mode/GPIO_list.c File Reference

7.6.1 Function Documentation

7.6.1.1 GPIO_list_add()

```
int GPIO_list_add (
            GPIO_list * list,
            GPIO * device )
```

Aggiunge un oggetto GPIO alla lista.

Parameters

list	puntatore a GPIO_list, lista a cui aggiungere l'oggetto
device	puntatore a GPIO, oggetto da aggiungere alla lista

Return values

-1	se è stsato già inserito il numero massimo di device
0	se non si manifesta nessun errore

7.6.1.2 GPIO_list_Destroy()

```
void GPIO_list_Destroy (
            GPIO_list* list )
```

Dealloca gli oggetti internamente contenuti nella GPIO_list.

Parameters

list	puntatore a GPIO_list, lista da distruggere

7.6.1.3 GPIO_list_device_count()

```
uint32_t GPIO_list_device_count (
            GPIO_list * list )
```

Restituisce il numero di device presenti nella lista.

Parameters

list	puntatore a GPIO_list, lista di cui si intende conoscere il numero di oggetti GPIO contenuti

Returns

numero di device presenti nella lista

7.6.1.4 GPIO_list_find_by_minor()

```
GPIO* GPIO_list_find_by_minor (
            GPIO_list * list,
            dev_t dev )
```

Ricerca un oggetto GPIO all'interno della lista tramite il minor number associato al device.

Parameters

list	puntatore a GPIO_list, lista in cui effettuare la ricerca
dev	major/minor number associato al device, parametro con cui viene invocata la open() o la release()

Returns

indirizzo dell'oggetto GPIO, se è presente nella lista, NULL altrimenti

7.6.1.5 GPIO_list_find_by_pdev()

```
GPIO* GPIO_list_find_by_pdev (
            GPIO_list * list,
            struct platform_device * pdev )
```

Ricerca un oggetto GPIO all'interno della lista tramite il campo pdev.

Parameters

list	puntatore a GPIO_list in cui effettuare la ricerca
pdev	puntatore a struct platform_device

Returns

indirizzo dell'oggetto GPIO, se è contenuto nella lista, NULL altrimenti

7.6.1.6 GPIO_list_find_irq_line()

```
GPIO* GPIO_list_find_irq_line (
            GPIO_list * list,
            int irq_line )
```

Ricerca un oggetto GPIO all'interno della lista tramite l' interrupt-number.

Parameters

list	puntatore a GPIO_list, lista in cui effettuare la ricerca
irq_line	linea di interruzione alla quale il device è connesso

Returns

indirizzo dell'oggetto GPIO, se è presente nella lista, NULL altrimenti

7.6.1.7 GPIO_list_Init()

```
int GPIO_list_Init (
            GPIO_list * list,
            uint32_t list_size )
```

Inizializza una struttura dati GPIO_list.

Parameters

list	puntatore a lista da inizializzare
list_size	numero massimo di device che la struttra dati potrà contenere

Return values

-ENOMEM	nel caso in cui la struttura non possa essere allocata in memoria
0	se non si manifestano errori

7.7 /media/saverio/OS/Users/Saverio/Desktop/SE/git/codici_da_mandare/FPGA/GPIO/↩ Driver/Kernel_Mode/GPIO_list.h File Reference

Data Structures

- struct GPIO_list

  Struttura dati per la gestione di più device GPIO da parte del driver.

7.7.1 Function Documentation

7.7.1.1 GPIO_list_add()

```
int GPIO_list_add (
            GPIO_list * list,
            GPIO * device )
```

Aggiunge un oggetto GPIO alla lista.

Parameters

list	puntatore a GPIO_list, lista a cui aggiungere l'oggetto
device	puntatore a GPIO, oggetto da aggiungere alla lista

Return values

-1	se è ststo già inserito il numero massimo di device

Return values

0	se non si manifesta nessun errore

7.7.1.2 GPIO_list_Destroy()

```
void GPIO_list_Destroy (
            GPIO_list* list )
```

Dealloca gli oggetti internamente contenuti nella GPIO_list.

Parameters

list	puntatore a GPIO_list, lista da distruggere

7.7.1.3 GPIO_list_device_count()

```
uint32_t GPIO_list_device_count (
            GPIO_list * list )
```

Restituisce il numero di device presenti nella lista.

Parameters

list	puntatore a GPIO_list, lista di cui si intende conoscere il numero di oggetti GPIO contenuti

Returns

numero di device presenti nella lista

7.7.1.4 GPIO_list_find_by_minor()

```
GPIO* GPIO_list_find_by_minor (
            GPIO_list * list,
            dev_t dev )
```

Ricerca un oggetto GPIO all'interno della lista tramite il minor number associato al device.

Parameters

list	puntatore a GPIO_list, lista in cui effettuare la ricerca
dev	major/minor number associato al device, parametro con cui viene invocata la open() o la release()

Returns

indirizzo dell'oggetto GPIO, se è presente nella lista, NULL altrimenti

7.7.1.5 GPIO_list_find_by_pdev()

```
GPIO* GPIO_list_find_by_pdev (
            GPIO_list * list,
            struct platform_device * pdev )
```

Ricerca un oggetto GPIO all'interno della lista tramite il campo pdev.

Parameters

list	puntatore a GPIO_list in cui effettuare la ricerca
pdev	puntatore a struct platform_device

Returns

indirizzo dell'oggetto GPIO, se è contenuto nella lista, NULL altrimenti

7.7.1.6 GPIO_list_find_irq_line()

```
GPIO* GPIO_list_find_irq_line (
            GPIO_list * list,
            int irq_line )
```

Ricerca un oggetto GPIO all'interno della lista tramite l' interrupt-number.

Parameters

list	puntatore a GPIO_list, lista in cui effettuare la ricerca
irq_line	linea di interruzione alla quale il device è connesso

Returns

indirizzo dell'oggetto GPIO, se è presente nella lista, NULL altrimenti

7.7.1.7 GPIO_list_Init()

```
int GPIO_list_Init (
            GPIO_list * list,
            uint32_t list_size )
```

Inizializza una struttura dati GPIO_list.

Parameters

list	puntatore a lista da inizializzare
list_size	numero massimo di device che la struttra dati potrà contenere

Return values

-ENOMEM	nel caso in cui la struttura non possa essere allocata in memoria
0	se non si manifestano errori

7.8 /media/saverio/OS/Users/Saverio/Desktop/SE/git/codici_da_mandare/FPGA/GPIO/↩ Driver/UIO/GPIO_interrupt_uio_poll.c File Reference

7.8.1 Function Documentation

7.8.1.1 read_reg()

```
unsigned int read_reg (
            void * addr,
            unsigned int offset )
```

Dealloca gli oggetti internamente contenuti nella GPIO_list.

Parameters

addr,puntatore	all' indirizzo da voler leggere
offset,offset	a partire dall' indirizzo a cui vogliamo scrivere

7.8.1.2 wait_for_interrupt()

```
void wait_for_interrupt (
            int fd0,
            int fd1,
            int fd2,
            void * addr_0,
            void * addr_1,
            void * addr_2 )
```

Dealloca gli oggetti internamente contenuti nella GPIO_list.

Parameters

fd0,valore	del file descriptor del primo GPIO
fd1,valore	del file descriptor del secondo GPIO
fd2,valore	del file descriptor del terzo GPIO
addr_0,indirizzo	base della prima periferica GPIO
addr_1,indirizzo	base della seconda periferica GPIO
addr_2,indirizzo	base della terza periferica GPIO

7.8.1.3 write_reg()

```
void write_reg (
            void * addr,
            unsigned int offset,
            unsigned int value )
```

Dealloca gli oggetti internamente contenuti nella GPIO_list.

Parameters

addr,puntatore	all' indirizzo da voler scrivere
offset,offset	a partire dall' indirizzo a cui vogliamo scrivere
value,valore	da voler scrivere

7.9 /media/saverio/OS/Users/Saverio/Desktop/SE/git/codici_da_mandare/FPGA/GPIO/↩ Driver/UIO/GPIO_interrupt_uio_poll.h File Reference

7.9.1 Function Documentation

7.9.1.1 read_reg()

```
unsigned int read_reg (
            void * addr,
            unsigned int offset )
```

Dealloca gli oggetti internamente contenuti nella GPIO_list.

Parameters

addr,puntatore	all' indirizzo da voler leggere
offset,offset	a partire dall' indirizzo a cui vogliamo scrivere

7.9.1.2 wait_for_interrupt()

```
void wait_for_interrupt (
             int fd0,
             int fd1,
             int fd2,
             void * addr_0,
             void * addr_1,
             void * addr_2 )
```

Dealloca gli oggetti internamente contenuti nella GPIO_list.

Parameters

fd0,valore	del file descriptor del primo GPIO
fd1,valore	del file descriptor del secondo GPIO
fd2,valore	del file descriptor del terzo GPIO
addr_0,indirizzo	base della prima periferica GPIO
addr_1,indirizzo	base della seconda periferica GPIO
addr_2,indirizzo	base della terza periferica GPIO

7.9.1.3 write_reg()

```
void write_reg (
             void * addr,
             unsigned int offset,
             unsigned int value )
```

Dealloca gli oggetti internamente contenuti nella GPIO_list.

Parameters

addr,puntatore	all' indirizzo da voler scrivere
offset,offset	a partire dall' indirizzo a cui vogliamo scrivere
value,valore	da voler scrivere

7.10 /media/saverio/OS/Users/Saverio/Desktop/SE/git/codici_da_mandare/FPGA/GPIO/↩ Hardware/GPIO_1.0/hdl/GPIO_v1_0.vhd File Reference

Top level entity del custom IP core GPIO_V1_0_S00_AXI.VHD.

Entities

- GPIO_v1_0 entity
- arch_imp architecture

7.10.1 Detailed Description

Top level entity del custom IP core GPIO_V1_0_S00_AXI.VHD.

7.11 /media/saverio/OS/Users/Saverio/Desktop/SE/git/codici_da_mandare/FPGA/GPIO/↩ Hardware/GPIO_1.0/hdl/GPIO_v1_0_S00_AXI.vhd File Reference

Componente utilizzato collegare il GPIO al bus AXI e gestire le interruzioni.

Entities

- GPIO_v1_0_S00_AXI entity
- arch_imp architecture

7.11.1 Detailed Description

Componente utilizzato collegare il GPIO al bus AXI e gestire le interruzioni.

Index