rca_tre_operandi

Corso di ASE anno 18/19

Gruppo 14
PREVITERA GABRIELE
PENNONE MIRKO
PENNA SIMONE

Contents

1 Class Index **1**

1.1 Class List 1

2 Class Documentation **3**

2.1 carry_save_logic Entity Reference 3

2.1.1 Detailed Description 3

2.1.2 Member Data Documentation 3

2.1.2.1 IEEE 4

2.1.2.2 STD_LOGIC_1164 4

2.2 d_edge_in Entity Reference 4

2.2.1 Member Data Documentation 4

2.2.1.1 ieee 4

2.2.1.2 std_logic_1164 5

2.3 d_edge_out Entity Reference 5

2.3.1 Member Data Documentation 5

2.3.1.1 ieee 5

2.3.1.2 std_logic_1164 6

2.4 full_adder Entity Reference 6

2.4.1 Detailed Description 6

2.4.2 Member Data Documentation 6

2.4.2.1 IEEE 7

2.4.2.2 STD_LOGIC_1164 7

2.5 rca_tre_operandi Entity Reference 7

2.5.1 Member Data Documentation 7

2.5.1.1 IEEE . . . 7
2.5.1.2 STD_LOGIC_1164 . . . 8
2.6 rca_tre_operandi_timing Entity Reference . . . 8
2.6.1 Detailed Description . . . 8
2.6.2 Member Data Documentation . . . 8
2.6.2.1 IEEE . . . 9
2.6.2.2 STD_LOGIC_1164 . . . 9
2.7 ripple_carry_adder Entity Reference . . . 9
2.7.1 Detailed Description . . . 9
2.7.2 Member Data Documentation . . . 10
2.7.2.1 c_in . . . 10
2.7.2.2 c_out . . . 10
2.7.2.3 IEEE . . . 10
2.7.2.4 S . . . 10
2.7.2.5 STD_LOGIC_1164 . . . 10
2.7.2.6 width . . . 10
2.7.2.7 Y . . . 10

Index **11**

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

entity carry_save_logic 3
entity d_edge_in 4
entity d_edge_out 5
entity full_adder 6
entity rca_tre_operandi 7
entity rca_tre_operandi_timing
Uncomment the following library declaration if instantiating any Xilinx primitives in this code . . 8
entity ripple_carry_adder 9

Chapter 2

Class Documentation

2.1 carry_save_logic Entity Reference

Libraries

- IEEE

Use Clauses

- STD_LOGIC_1164

Generics

- width **NATURAL:=** 4

Ports

- X **in STD_LOGIC_VECTOR(width- 1 downto 0)**
- Y **in STD_LOGIC_VECTOR(width- 1 downto 0)**
- Z **in STD_LOGIC_VECTOR(width- 1 downto 0)**
- T **out STD_LOGIC_VECTOR(width- 1 downto 0)**

  somme dei 3 bit di stesso peso di X, Y e Z
- CS **out STD_LOGIC_VECTOR(width- 1 downto 0)**

  riporti uscenti dai carry save blocks

2.1.1 Detailed Description

Descrizione Somma tre stringhe di bit per poter realizzare la logica di un carry save e viene posto prima del ripple carry in un carry save adder

2.1.2 Member Data Documentation

2.1.2.1 IEEE

IEEE [Library]

FEDERICO II , CORSO DI ASE 18/19, Gruppo 14 –

2.1.2.2 STD_LOGIC_1164

STD_LOGIC_1164 [Package]

last changes: <14/11/2018> <13/11/2018> <log> create

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

- carry_save_logic.vhd

2.2 d_edge_in Entity Reference

Libraries

- ieee

Use Clauses

- std_logic_1164
- all

Generics

- width **natural:=** 8

Ports

- clock **in STD_LOGIC**
- D **in STD_LOGIC_VECTOR(width-** 1 **downto** 0 **)**
- Q **out STD_LOGIC_VECTOR(width-** 1 **downto** 0 **)**

2.2.1 Member Data Documentation

2.2.1.1 ieee

ieee [Library]

D Flip-Flop (ESD book Chapter 2.3.1) by Weijun Zhang, 04/2001

2.2.1.2 std_logic_1164

std_logic_1164 [Package]

Flip-flop is the basic component in sequential logic design we assign input signal to the output at the clock rising edge

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

- d_edge_in.vhd

2.3 d_edge_out Entity Reference

Libraries

- ieee

Use Clauses

- std_logic_1164
- all

Generics

- width **natural:=** 8

Ports

- clock **in STD_LOGIC**
- D **in STD_LOGIC_VECTOR(width-** 1 **downto** 0 **)**
- Q **out STD_LOGIC_VECTOR(width-** 1 **downto** 0 **)**

2.3.1 Member Data Documentation

2.3.1.1 ieee

ieee [Library]

D Flip-Flop (ESD book Chapter 2.3.1) by Weijun Zhang, 04/2001

2.3.1.2 std_logic_1164

`std_logic_1164` `[Package]`

Flip-flop is the basic component in sequential logic design we assign input signal to the output at the clock rising edge

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

- d_edge_out.vhd

2.4 full_adder Entity Reference

Libraries

- IEEE

Use Clauses

- STD_LOGIC_1164

Ports

- x **in STD_LOGIC**
  full_adder input : addendo
- y **in STD_LOGIC**
  full_adder input : addendo
- c_in **in STD_LOGIC**
  full_adder input : carry in ingresso
- s **out STD_LOGIC**
  full_adder output : somma
- c_out **out STD_LOGIC**
  full_adder output : carry

2.4.1 Detailed Description

Descrizione Somma i 3 bit in ingresso (2 addendi e 1 carry in ingresso).
In uscita abbiamo il risultato della somma sul bit S e il riporto sul bit C.

2.4.2 Member Data Documentation

2.4.2.1 IEEE

IEEE [Library]

FEDERICO II , CORSO DI ASE 18/19, Gruppo 14 –

2.4.2.2 STD_LOGIC_1164

STD_LOGIC_1164 [Package]

last changes: <11/11/2018> <15/10/2018> <log> Aggiunta doc doxygen

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

- full_adder.vhd

2.5 rca_tre_operandi Entity Reference

Libraries

- IEEE

Use Clauses

- STD_LOGIC_1164

Generics

- width **natural:=** 128

Ports

- X **in STD_LOGIC_VECTOR(width-** 1 **downto** 0 **)**
- Y **in STD_LOGIC_VECTOR(width-** 1 **downto** 0 **)**
- Z **in STD_LOGIC_VECTOR(width-** 1 **downto** 0 **)**
- S **out STD_LOGIC_VECTOR(width+** 1 **downto** 0 **)**

2.5.1 Member Data Documentation

2.5.1.1 IEEE

IEEE [Library]

FEDERICO II , CORSO DI ASE 18/19, Gruppo 14 –

2.5.1.2 STD_LOGIC_1164

STD_LOGIC_1164 [Package]

last changes: <14/11/2018> <13/11/2018> <log> create

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

- carry_save.vhd

2.6 rca_tre_operandi_timing Entity Reference

Uncomment the following library declaration if instantiating any Xilinx primitives in this code.

Libraries

- IEEE

Use Clauses

- STD_LOGIC_1164

Generics

- width **natural:=** 4

Ports

- clock **in STD_LOGIC**
- X **in STD_LOGIC_VECTOR(width-** 1 **downto** 0 **)**
- Y **in STD_LOGIC_VECTOR(width-** 1 **downto** 0 **)**
- Z **in STD_LOGIC_VECTOR(width-** 1 **downto** 0 **)**
- S **out STD_LOGIC_VECTOR(width+** 1 **downto** 0 **)**

2.6.1 Detailed Description

Uncomment the following library declaration if instantiating any Xilinx primitives in this code.

Uncomment the following library declaration if using arithmetic functions with Signed or Unsigned values

2.6.2 Member Data Documentation

2.6.2.1 IEEE

IEEE [Library]

Company: Engineer:

Create Date: 14:00:52 02/16/2019 Design Name: Module Name: rca_tre_operandi_timing - Behavioral Project Name: Target Devices: Tool versions: Description:

2.6.2.2 STD_LOGIC_1164

STD_LOGIC_1164 [Package]

Revision: Revision 0.01 - File Created Additional Comments:

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

- carry_save_timing.vhd

2.7 ripple_carry_adder Entity Reference

Libraries

- IEEE

Use Clauses

- STD_LOGIC_1164

Generics

- width **NATURAL:=** 8

Ports

- X **in STD_LOGIC_VECTOR(width - 1 downto 0)**
- Y **in STD_LOGIC_VECTOR(width - 1 downto 0)**
- c_in **in STD_LOGIC**
- S **out STD_LOGIC_VECTOR(width - 1 downto 0)**
- c_out **out STD_LOGIC**

  ripple_carry_adder output: carry

2.7.1 Detailed Description

Descrizione Somma le 2 stringe di bit in ingresso (2 addendi) e 1 bit (carry in ingresso). Caratterizzato da una serie di full_adder in cascata che propagano il riporto.
In uscita abbiamo il risultato della somma sul bit S e il riporto sul bit C.

2.7.2 Member Data Documentation

2.7.2.1 c_in

`c_in` **in STD_LOGIC** [Port]

ripple_carry_adder input: addendo

2.7.2.2 c_out

`c_out` **out STD_LOGIC** [Port]

ripple_carry_adder output: carry

ripple_carry_adder output: somma

2.7.2.3 IEEE

`IEEE` [Library]

FEDERICO II , CORSO DI ASE 18/19, Gruppo 14 –

2.7.2.4 S

`S` **out STD_LOGIC_VECTOR(width - 1 downto 0)** [Port]

ripple_carry_adder input : carry in ingresso

2.7.2.5 STD_LOGIC_1164

`STD_LOGIC_1164` [Package]

last changes: <11/11/2018> <15/10/2018> <log> Aggiunta doc doxygen

2.7.2.6 width

`width` **NATURAL:= 8** [Generic]

usato per definire il parallelismo del ripple_carry_adder

2.7.2.7 Y

`Y` **in STD_LOGIC_VECTOR(width - 1 downto 0)** [Port]

ripple_carry_adder input: addendo

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

- ripple_carry_adder.vhd

Index

c_in
    ripple_carry_adder, 10
c_out
    ripple_carry_adder, 10
carry_save_logic, 3
    IEEE, 3
    STD_LOGIC_1164, 4

d_edge_in, 4
    ieee, 4
    std_logic_1164, 4
d_edge_out, 5
    ieee, 5
    std_logic_1164, 5

full_adder, 6
    IEEE, 6
    STD_LOGIC_1164, 7

IEEE
    carry_save_logic, 3
    full_adder, 6
    rca_tre_operandi, 7
    rca_tre_operandi_timing, 8
    ripple_carry_adder, 10
ieee
    d_edge_in, 4
    d_edge_out, 5

rca_tre_operandi, 7
    IEEE, 7
    STD_LOGIC_1164, 7
rca_tre_operandi_timing, 8
    IEEE, 8
    STD_LOGIC_1164, 9
ripple_carry_adder, 9
    c_in, 10
    c_out, 10
    IEEE, 10
    S, 10
    STD_LOGIC_1164, 10
    width, 10
    Y, 10

S
    ripple_carry_adder, 10
STD_LOGIC_1164
    carry_save_logic, 4
    full_adder, 7
    rca_tre_operandi, 7
    rca_tre_operandi_timing, 9
    ripple_carry_adder, 10
std_logic_1164
    d_edge_in, 4
    d_edge_out, 5

width
    ripple_carry_adder, 10

Y
    ripple_carry_adder, 10