

carry\_save

Corso di ASE anno 18/19

Gruppo 14

PREVITERA GABRIELE

PENNONE MIRKO

PENNA SIMONE



# Contents

<b>1</b>	<b>Class Index</b>	<b>1</b>
1.1	Class List . . . . .	1
<b>2</b>	<b>Class Documentation</b>	<b>3</b>
2.1	carry_save Entity Reference . . . . .	3
2.1.1	Member Data Documentation . . . . .	3
2.1.1.1	IEEE . . . . .	3
2.1.1.2	STD_LOGIC_1164 . . . . .	4
2.2	carry_save_logic Entity Reference . . . . .	4
2.2.1	Detailed Description . . . . .	4
2.2.2	Member Data Documentation . . . . .	4
2.2.2.1	IEEE . . . . .	5
2.2.2.2	STD_LOGIC_1164 . . . . .	5
2.3	carry_save_timing Entity Reference . . . . .	5
2.3.1	Detailed Description . . . . .	5
2.3.2	Member Data Documentation . . . . .	6
2.3.2.1	IEEE . . . . .	6
2.3.2.2	STD_LOGIC_1164 . . . . .	6
2.4	d_edge_in Entity Reference . . . . .	6
2.5	d_edge_out Entity Reference . . . . .	7
2.6	full_adder Entity Reference . . . . .	7
2.6.1	Detailed Description . . . . .	8
2.6.2	Member Data Documentation . . . . .	8
2.6.2.1	IEEE . . . . .	8

---

2.6.2.2	STD_LOGIC_1164 . . . . .	8
2.7	ripple_carry_adder Entity Reference . . . . .	8
2.7.1	Detailed Description . . . . .	9
2.7.2	Member Data Documentation . . . . .	9
2.7.2.1	c_in . . . . .	9
2.7.2.2	c_out . . . . .	9
2.7.2.3	IEEE . . . . .	9
2.7.2.4	S . . . . .	9
2.7.2.5	STD_LOGIC_1164 . . . . .	9
2.7.2.6	width . . . . .	9
2.7.2.7	Y . . . . .	9
	<b>Index</b>	<b>11</b>

# Chapter 1

## Class Index

### 1.1 Class List

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

entity <a href="#">carry_save</a> . . . . .	3
entity <a href="#">carry_save_logic</a> . . . . .	4
entity <a href="#">carry_save_timing</a>	
Uncomment the following library declaration if instantiating any Xilinx primitives in this code . . .	5
entity <a href="#">d_edge_in</a> . . . . .	6
entity <a href="#">d_edge_out</a> . . . . .	7
entity <a href="#">full_adder</a> . . . . .	7
entity <a href="#">ripple_carry_adder</a> . . . . .	8



## Chapter 2

# Class Documentation

### 2.1 carry\_save 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** )

#### 2.1.1 Member Data Documentation

##### 2.1.1.1 IEEE

[IEEE](#) [Library]

### 2.1.1.2 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`

## 2.2 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*

### 2.2.1 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

### 2.2.2 Member Data Documentation



## 2.2.2.1 IEEE

[IEEE](#) [Library]

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

## 2.2.2.2 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`

## 2.3 carry\_save\_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 := 32`

### 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 )`

### 2.3.1 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

## 2.3.2 Member Data Documentation

### 2.3.2.1 IEEE

IEEE [Library]

Company: Engineer:

Create Date: 14:00:52 02/16/2019 Design Name: Module Name: [carry\\_save\\_timing](#) - Behavioral Project Name:  
Target Devices: Tool versions: Description:

### 2.3.2.2 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](#)

## 2.4 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**)

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

- [d\\_edge\\_in.vhd](#)

## 2.5 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** )

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

- [d\\_edge\\_out.vhd](#)

## 2.6 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*

### 2.6.1 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.

### 2.6.2 Member Data Documentation

#### 2.6.2.1 IEEE

[IEEE](#) [Library]

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

#### 2.6.2.2 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](#)

## 2.7 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, 9
- c\_out
  - ripple\_carry\_adder, 9
- carry\_save, 3
  - IEEE, 3
  - STD\_LOGIC\_1164, 3
- carry\_save\_logic, 4
  - IEEE, 4
  - STD\_LOGIC\_1164, 5
- carry\_save\_timing, 5
  - IEEE, 6
  - STD\_LOGIC\_1164, 6
- d\_edge\_in, 6
- d\_edge\_out, 7
- full\_adder, 7
  - IEEE, 8
  - STD\_LOGIC\_1164, 8
- IEEE
  - carry\_save, 3
  - carry\_save\_logic, 4
  - carry\_save\_timing, 6
  - full\_adder, 8
  - ripple\_carry\_adder, 9
- ripple\_carry\_adder, 8
  - c\_in, 9
  - c\_out, 9
  - IEEE, 9
  - S, 9
  - STD\_LOGIC\_1164, 9
  - width, 9
  - Y, 9
- S
  - ripple\_carry\_adder, 9
- STD\_LOGIC\_1164
  - carry\_save, 3
  - carry\_save\_logic, 5
  - carry\_save\_timing, 6
  - full\_adder, 8
  - ripple\_carry\_adder, 9
- width
  - ripple\_carry\_adder, 9
- Y
  - ripple\_carry\_adder, 9