Programmable Logic Array Simulator -
Alice Plebe, Matteo Cavallaro

API Documentation

July 18, 2018

Contents

1 Module circuits

Libreria di circuiti predefiniti.

Author: Alice Plebe, Matteo Cavallaro

Version: 3.0

1.1 Variables

Name	Description
circ_h	1 bit half adder **Value:** `Circuit(2, 2, 3)`
circ_a	1 bit full adder **Value:** `Circuit(3, 2, 7)`
circ_b	2 bit adder **Value:** `Circuit(4, 3, 13)`
circ_e	priority encoder **Value:** `Circuit(4, 3, 7)`
circ_m	multiplexer **Value:** `Circuit(6, 1, 4)`
circ_g	majority **Value:** `Circuit(3, 1, 4)`
circ_d	decoder **Value:** `Circuit(3, 8, 8)`
circ_s	shift register **Value:** `Circuit(6, 5, 8)`
circ_c	comparator **Value:** `Circuit(4, 3, 10)`
circ_mlg	multiple logic gate **Value:** `Circuit(2, 6, 10)`
circ_mult2	2 bit multiplicator **Value:** `Circuit(4, 4, 9)`
circ_r32	reductor 3-2 **Value:** `Circuit(3, 2, 7)`
circ_bcd	decoder 7 segmenti **Value:** `Circuit(4, 7, 16)`
circ_sr	one step flip-flop sr **Value:** `Circuit(4, 2, 5)`
circ_t	one step flip-flop t **Value:** `Circuit(2, 2, 5)`
circ_jk	one step flip-flop jk **Value:** `Circuit(3, 2, 5)`
circ_compl1	6-bit ones' complement **Value:** `Circuit(6, 6, 6)`
circ_pc	parity check **Value:** `Circuit(4, 1, 8)`
circ_crc3	crc-3-gsm **Value:** `Circuit(4, 3, 14)`
circ_compl2	4-bit two' complement **Value:** `Circuit(4, 4, 13)`

continued on next page

Name	Description
circ_sqrt	6 bit square root floor **Value:** `Circuit(6, 3, 16)`
circs	lista dei circuiti predefiniti disponibili nella libreria **Value:** `[circ_h, circ_a, circ_b, circ_compl1, circ_compl2, circ_m...`
__package__	**Value:** `None`

1.2 Class Circuit

object
└ **circuits.Circuit**

Classe dei circuiti predefiniti. Ogni istanza di questa classe è composta dalle due matrici di connessione delle porte AND e OR del PLA, inizializzate tutte con nodi non connessi.

Note: il numero di input, output, porte AND non deve superare quello del simulatore.

1.2.1 Methods

__init__(*self*, *n_in*, *n_out*, *n_and*)

Istanzia un circuito predefinito.

Parameters

`n_in`: numero di input del circuito

`n_out`: numero di output del circuito

`n_and`: numero di porte AND del circuito

Overrides: object.__init__

generate_code(*name*, *description*, *function*, *input_names*, *output_names*)

Genera il codice necessario per creare un circuito data una funzione logica. Nota: questa funzione non genera una rete combinatoria minimale.

Parameters

`name`: nome della variabile

`description`: nome del circuito

`function`: funzione booleana da replicare

`input_names`: denominazioni degli input del circuito

`output_names`: denominazioni degli input del circuito

generate_obj(*description*, *function*, *input_names*, *output_names*)

Genera un circuito data una funzione logica. Nota: questa funzione non genera una rete combinatoria minimale.

Parameters

`description:`	nome del circuito
`function:`	funzione booleana da replicare
`input_names:`	denominazioni degli input del circuito
`output_names:`	denominazioni degli input del circuito

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

1.2.2 Properties

Name	Description
Inherited from object	
__class__	

1.2.3 Instance Variables

Name	Description
n_inputs	numero di ingressi del circuito **Value:** `0`
n_outputs	numero di uscite del circuito, equivalente al numero di porte OR presenti **Value:** `0`
n_and	numero di porte AND del circuito **Value:** `0`
and_matrix	matrice di connessione tra ingressi e porte AND **Value:** `None`
or_matrix	matrice di connessione tra porte AND e porte OR **Value:** `None`
description	il nome del circuito predefinito **Value:** `''`
labels_i	denominazioni degli input del circuito **Value:** `[]`
labels_o	denominazioni degli output del circuito **Value:** `[]`

2 Module component

Classi di tutti i componenti elettronici del simulatore.

Author: Alice Plebe, Matteo Cavallaro

Version: 3.0

2.1 Variables

Name	Description
__package__	**Value:** `None`

2.2 Class Component

object
└ **component.Component**

Known Subclasses: component.Port, component.Fuse, component.InPin, component.OutPin, component.Wire

Classe astratta di tutti i componenti circuitali nella loro forma grafica.

Ogni sottoclasse va istanziata con primo argomento un oggetto di classe Pla. Tuti i valori geometrici sono trattati in coordinate normalizzate, e solo nel momento di chiamare le primitive grafiche sono trasformati in valori assoluti (pixel).

Per tutti i componenti, Wire e pin di ingresso e uscita esclusi, viene validato l'attributo *tags* di Tkinter con una stringa composta dal nome del componente e il suo numero progressivo.

2.2.1 Methods

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __init__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

2.2.2 Properties

Name	Description
Inherited from object	

continued on next page

Name	Description
__class__	

2.3 Class Port

object
component.Component
component.Port

Known Subclasses: component.And, component.Not, component.Or

Classe delle porte logiche.

Tutte le sottoclassi avranno un loro attributo di classe 'count' che contiene il numero di porte correntemente istanziate di quel tipo. Tutte le sottoclassi avranno un loro attributo di classe 'name' con il prefisso del rispettivo tipo di porta.

2.3.1 Methods

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __init__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

2.3.2 Properties

Name	Description
Inherited from object __class__	

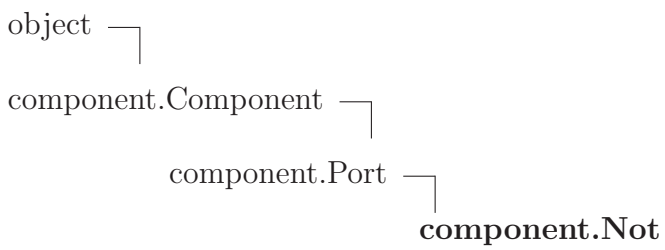
2.3.3 Class Variables

Name	Description
size	grandezza complessiva di una porta **Value:** `0.04` *(type=dimensione normalizzata 0..1)*
thick	spessore dei contorni di una porta **Value:** `1`

continued on next page

Name	Description
color	colore dei bordi di una porta **Value:** `'black'`

2.4 Class Not



Classe della porta logica NOT.

La porta viene disegnata in orientamento verticale con ingresso in alto, quindi con un triangolo equilatero al cui vertice è\ aggiunto un piccolo cerchio.

2.4.1 Methods

__init__(*self*, *pla*, *x*)

Istanzia una porta NOT.

Parameters

`pla`: il simulatore
(type=Pla)

`x`: coordinata orizzontale del centro della porta
(type=dimensione normalizzata 0..1)

Overrides: object.__init__

pin_in(*self*)

Calcola le coordinate del punto di ingresso della porta.

Return Value

coordinate normalizzate del punto di ingresso della porta

pin_out(*self*)

Calcola le coordinate del punto di uscita della porta.

Return Value

coordinate normalizzate del punto di uscita della porta

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

2.4.2 Properties

Name	Description
Inherited from object __class__	

2.4.3 Class Variables

Name	Description
name	prefisso della porta NOT **Value:** `'not_'`
c_size	diametro del cerchio da cui è50 composta la porta NOT **Value:** `0.2` *(type=frazione della dimensione totale della porta)*
Inherited from component.Port (Section 2.3) color, size, thick	

2.4.4 Instance Variables

Name	Description
count	numero di porte NOT correntemente istanziate **Value:** `0`
y_not	coordinata verticale del centro della porta NOT **Value:** `0.0` *(type=dimensione normalizzata 0..1)*
tag	tag dell'oggetto grafico delineante la porta *(type=Tkinter widget tag)*
x	coordinata orizzontale del centro della porta *(type=dimensione normalizzata 0..1)*

continued on next page

Name	Description
y_in	coordinata verticale del pin di ingresso della porta *(type=dimensione normalizzata 0..1)*
y_out	coordinata verticale del pin di uscita della porta *(type=dimensione normalizzata 0..1)*

2.5 Class Or

object
component.Component
component.Port
component.Or

Classe della porta logica OR.

La porta viene disegnata in orientamento verticale con ingresso in alto, ed è composta mediante:

- arco di cerchio con centro sull'asse, per la sua base
- due linee verticali parallele
- due archi di cerchio con centri sfasati rispetto all'asse, per la punta

2.5.1 Methods

draw_lines(*self*, *xa_0*, *xa_2*, *ya_0*, *ya_2*)

Traccia le linee verticali della porta

Il metodo non restituisce nulla, inserisce gli oggetti widget nella lista self.lines

Parameters

`xa_0`: coordinata orizzontale della linea sinistra

`xa_0`: coordinata orizzontale della linea destra

`ya_0`: coordinata verticale del vertice superiore delle linee

`ya_2`: coordinata verticale del vertice inferiore delle linee

draw_bottom(*self*, *ra*, *theta*)

Disegna l'arco per la base della porta

Il metodo non restituisce nulla, inserisce gli oggetti widget nella lista self.arcs

Parameters

`ra:` raggio dell'arco

`theta:` semiangolo del vertice superiore del triangolo isoscele su cui insiste l'arco

draw_top(*self*, *xa_delta*, *ya_c*, *ra*, *theta*, *delta*)

Disegna i due archi per la punta della porta

Il metodo non restituisce nulla, inserisce gli oggetti widget nella lista self.arcs

Parameters

`xa_delta:` offset tra il centro orizzontale della porta, e i centri delle circonferenze

`ya_c:` coordinata verticale dei centri delle circonferenze

`ra:` raggio degli archi

`theta:` angolo formato tra il segmento che congiunge il vertice inferiore della linea sinistra (che è anche vertice superiore dell'arco sinistro) con il centro della circonferenza per l'arco sinsitro, e l'asse orizzontale

`delta:` angolo formato tra il segmento che congiunge il vertice inferiore degli archi con il centro della circonferenza per l'arco sinsitro, e l'asse verticale

__**init**__(*self*, *pla*, *x*)

Istanzia una porta OR.

Parameters

`pla:` il simulatore

(type=Pla)

`x:` coordinata orizzontale del centro della porta

(type=dimensione normalizzata 0..1)

Overrides: object.__init__

pin_in(*self*)

Calcola le coordinate del punto di ingresso della porta.

Return Value
coordinate normalizzate del punto di ingresso della porta

pin_out(*self*)

Calcola le coordinate del punto di uscita della porta.

Return Value
coordinate normalizzate del punto di uscita della porta

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

2.5.2 Properties

Name	Description
Inherited from object	
__class__	

2.5.3 Class Variables

Name	Description
name	prefisso della porta OR **Value:** `'or_'`
line_s	dimensione delle linee costituenti la porta **Value:** `0.4` *(type=frazione della dimensione totale della porta)*
elong	rapporto tra lunghezza (verticale) e larghezza (orizzontale) della porta OR **Value:** `1.7`
r_top	rapporto tra raggio dei cerchi usati per la punta e dimensione della porta **Value:** `0.5`
r_bottom	rapporto tra raggio del cerchio usato per la base e dimensione della porta **Value:** `0.7`
top_sharp	inverso dell'acutezza della punta **Value:** `0.2`

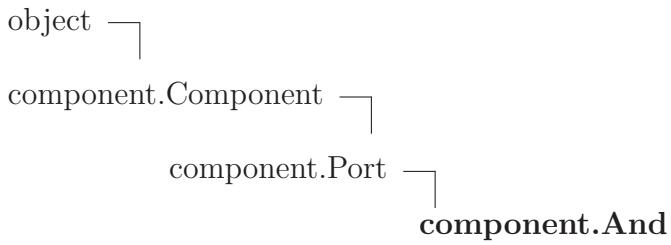
continued on next page

Name	Description
adjust	fattore correttivo della dimensione della porta OR per equipararla alla porta AND **Value:** `1.4`
Inherited from component.Port (Section 2.3) color, size, thick	

2.5.4 Instance Variables

Name	Description
count	numero di porte OR correntemente istanziate **Value:** `0`
y_or	coordinata verticale del centro della porta OR **Value:** `0.0` *(type=dimensione normalizzata 0..1)*
pla	il simulatore **Value:** `None` *(type=Pla)*
tag	tag dell'oggetto grafico delineante la porta *(type=Tkinter widget tag)*
x	coordinata orizzontale del centro della porta *(type=dimensione normalizzata 0..1)*
y_in	coordinata verticale del pin di ingresso della porta *(type=dimensione normalizzata 0..1)*
y_out	coordinata verticale del pin di uscita della porta *(type=dimensione normalizzata 0..1)*

2.6 Class And



Classe della porta logica AND.

La porta viene disegnata in orientamento orizzontale con ingresso a sinistra, ed è composta mediante:

- poligonale con due linee orizzontali parallele e una verticale

- arco di cerchio con centro sull'asse, per la sua parte anteriore

2.6.1 Methods

__init__(*self*, *pla*, *y*)

Istanzia una porta AND.

Parameters

`pla`: il simulatore

(type=Pla)

`y`: coordinata verticale del centro della porta

(type=dimensione normalizzata 0..1)

Overrides: object.__init__

value(*self*, *pla*, *status*)

Aggiorna *status* e *text* della porta in base al valore indicato.

Parameters

`pla`: il simulatore

(type=Pla)

`status`: nuovo stato logico della porta

disable(*self*, *pla*)

Disattiva una porta AND.

Parameters

`pla`: il simulatore

(type=Pla)

reset(*self*, *pla*)

Resetta una porta AND al suo stato iniziale.

Parameters

`pla`: il simulatore

(type=Pla)

pin_in(*self*)

Calcola le coordinate del punto di entrata della porta.

Return Value
coordinate normalizzate del punto di entrata della porta

pin_out(*self*)

Calcola le coordinate del punto di uscita della porta.

Return Value
coordinate normalizzate del punto di uscita della porta

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

2.6.2 Properties

Name	Description
Inherited from object __class__	

2.6.3 Class Variables

Name	Description
name	prefisso della porta AND **Value:** `'and_'`
line_s	dimensione delle linee costituenti la porta **Value:** `0.6` *(type=frazione della dimensione totale della porta)*
Inherited from component.Port (Section 2.3) color, size, thick	

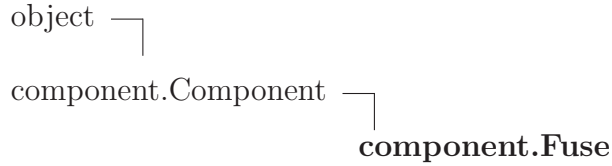
2.6.4 Instance Variables

Name	Description
count	numero di porte AND correntemente istanziate **Value:** `0`

continued on next page

Name	Description
x_and	coordinata orizzontale del centro della porta AND **Value:** `0.0` *(type=dimensione normalizzata 0..1)*
status	stato logico della porta **Value:** `0`
locked	variabile indicante se la porta è inattiva **Value:** `False` *(type=boolean)*
text	area di testo dove viene visualizzato lo *status* della porta **Value:** `None` *(type=Tkinter.Canvas object ID)*
tag	tag dell'oggetto grafico delineante la porta *(type=Tkinter widget tag)*
x_in	coordinata orizzontale del pin di ingresso della porta *(type=dimensione normalizzata 0..1)*
x_out	coordinata orizzontale del pin di uscita della porta *(type=dimensione normalizzata 0..1)*
y	coordinata verticale del centro della porta *(type=dimensione normalizzata 0..1)*

2.7 Class Fuse



Classe dei fusibili.

Vengono considerate più categorie di fusibili, pertanto il nome di un tipo di fusibile è composto dall'attributo *name*, più un suffisso specificato come argomento, ed è conservato nell'attributo *category*.

2.7.1 Methods

__init__(*self*, *pla*, *x*, *y*, *suffix=''*)

Istanzia un fusibile.

Parameters

`pla:` il simulatore
(type=Pla)

`x:` coordinata orizzontale del centro del fusibile
(type=dimensione normalizzata 0..1)

`y:` coordinata verticale del centro del fusibile
(type=dimensione normalizzata 0..1)

`suffix:` suffisso indicante la categoria di appartenenza del fusibile

Overrides: object.__init__

toggle(*self*, *pla*)

Scambia lo stato attuale del fusibile.

Parameters

`pla:` il simulatore
(type=Pla)

deset(*self*, *pla*)

Setta lo stato scollegato a un fusibile.

Parameters

`pla:` il simulatore
(type=Pla)

reset(*self*, *pla*)

Resetta lo stato collegato a un fusibile.

Parameters

`pla:` il simulatore
(type=Pla)

pin_in(*self*)

Calcola le coordinate del pin a ovest del fusibile.

Return Value
coordinate normalizzate del pin a ovest del fusibile

pin_out(*self*)

Calcola le coordinate del pin a est del fusibile.

Return Value
coordinate normalizzate del pin a est del fusibile

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

2.7.2 Properties

Name	Description
Inherited from object	
__class__	

2.7.3 Class Variables

Name	Description
name	prefisso del fusibile **Value:** `'fuse_'`
size	grandezza complessiva di un fusibile **Value:** `0.009` *(type=dimensione normalizzata 0..1)*
thick	spessore dei contorni di un fusibile **Value:** `1`
color	colore dei bordi di un fusibile **Value:** `'black'`

2.7.4 Instance Variables

Name	Description
count	numero di fusibili correntemente istanziati per ogni categoria **Value:** {}

continued on next page

Name	Description
category	nome completo del fusibile, comprendente il suffisso della categoria a cui appartiene **Value:** ’’
status	stato del fusibile (è *False* quando esso risulta interrotto) **Value:** `True` *(type=boolean)*
tag	tag dell’oggetto grafico delineante la porta *(type=Tkinter widget tag)*
x_in	coordinata orizzontale del pin di ingresso del fusibile *(type=dimensione normalizzata 0..1)*
x_out	coordinata orizzontale del pin di uscita del fusibile *(type=dimensione normalizzata 0..1)*
y	coordinata verticale del centro del fusibile *(type=dimensione normalizzata 0..1)*

2.8 Class Wire

object
└ component.Component
  └ **component.Wire**

Classe dei fili di collegamento

Un filo viene inizializzata specificando gli oggetti di classe Component di partenza e di arrivo del collegamento, e con l’argomento opzionale *placement* se il collegamento deve essere al pin del Component oppure al suo centro.

2.8.1 Methods

__init__(*self*, *pla*, *c_from*, *c_to*, *placement*=('pin', 'pin'))

Istanzia un filo di collegamento.

Parameters

`pla:`	il simulatore *(type=Pla)*
`c_from:`	componente collegato alla partenza del filo *(type=Component)*
`c_to:`	componente collegato all'arrivo del filo *(type=Component)*
`placement:`	indica se il collegamento va effettuato al pin del componente (default), o al suo centro

Overrides: object.__init__

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

2.8.2 Properties

Name	Description
Inherited from object	
__class__	

2.8.3 Class Variables

Name	Description
thick	spessore della linea del filo di collegamento **Value:** `1`
color	colore della linea del filo di collegamento **Value:** `'black'`

2.9 Class InPin

object
component.Component
component.InPin

Classe dei pin di ingresso del PLA realizzati mediante pulsanti.

La classe comprende nell'attributo *var* un oggetto Tkinter.IntVar, utile per propagare nel programma lo stato dei pin di ingresso.

Al costruttore Button è assegnato l'argomento *command* al metodo *toggle*, che crea la gestione dell'evento del mouse che clicca sul pulsante, senza necessità di un'esplicita chiamata canvas.bind()

2.9.1 Methods

__init__(*self*, *pla*, *x*, *v*)

Istanzia un pin di ingresso.

Parameters

`pla:` il simulatore
(type=Pla)

`x:` coordinata orizzontale del punto di uscita del pin

`v:` valore dello stato logico del pin
(type=Tkinter.IntVar)

Overrides: object.__init__

set_label(*self*, *pla*, *t*)

Setta l'etichetta dell'input con la stringa passata come parametro.

Parameters

`pla:` il simulatore
(type=Pla)

`t:` stringa della nuova etichetta dell'input

reset_label(*self*, *pla*)

Resetta l'etichetta dell'input a stringa vuota.

Parameters

`pla`: il simulatore

(type=Pla)

toggle(*self*)

Scambia lo stato logico del pin

enable(*self*)

Abilita il pin.

disable(*self*)

Disabilita il pin.

reset(*self*, *pla*)

Resetta il pin di input al suo stato iniziale.

Parameters

`pla`: il simulatore

(type=Pla)

wire_not(*self*, *pla*, *c_not*)

Funzione ausiliaria per disegnare il collegamento tra pin di ingresso del PLA e gli ingressi delle porte NOT.

Parameters

`pla`: il simulatore

(type=Pla)

`c_not`: porta logica NOT

(type=Component.Not)

pin_out(*self*)

Calcola le coordinate del punto di uscita del pin.

Return Value

coordinate normalizzate del punto di uscita del pin

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

2.9.2 Properties

Name	Description
Inherited from object	
__class__	

2.9.3 Instance Variables

Name	Description
y_in	coordinata verticale del centro del pin di input **Value:** `0.0` *(type=dimensione normalizzata 0..1)*
label	area di testo dove viene visualizzata l'etichetta dell'input **Value:** `None` *(type=Tkinter.Canvas object ID)*
y_lab	fattore di posizionamento della label nella finestra **Value:** `540.0` *(type=pixel)*
v	valore dello stato logico del pin *(type=Tkinter.IntVar)*
x	coordinata orizzontale del centro del pin di input *(type=dimensione normalizzata 0..1)*

2.10 Class OutPin

object
component.Component
component.OutPin

Classe dei pin di uscita del PLA.

Il pin viene rappresentato da due cerchi concentrici.

2.10.1 Methods

__init__(*self*, *pla*, *x*)

Istanzia un pin di ingresso.

Parameters

`pla:` il simulatore

(type=Pla)

`x:` coordinata orizzontale del punto di uscita del pin

(type=dimensione normalizzata 0..1)

`x:` coordinata orizzontale del punto di uscita del pin

(type=dimensione normalizzata 0..1)

Overrides: object.__init__

set_label(*self*, *pla*, *t*)

Setta l'etichetta dell'output con la stringa passata come parametro.

Parameters

`pla:` il simulatore

(type=Pla)

`t:` stringa della nuova etichetta del pin

reset_label(*self*, *pla*)

Resetta l'etichetta dell'output a stringa vuota.

Parameters

`pla:` il simulatore

(type=Pla)

value(*self*, *pla*, *status*)

Aggiorna *text* del pin in base al valore di stato indicato.

Parameters

`pla:` il simulatore

(type=Pla)

`status:` nuovo stato logico della porta

disable(*self*, *pla*)

Disabilita il pin.

Parameters

`pla`: il simulatore

(type=Pla)

reset(*self*, *pla*)

Resetta il pin di output al suo stato iniziale.

Parameters

`pla`: il simulatore

(type=Pla)

pin_in(*self*)

Calcola le coordinate del punto di entrata del pin.

Return Value

coordinate normalizzate del punto di entrata del pin

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

2.10.2 Properties

Name	Description
Inherited from object	
__class__	

2.10.3 Class Variables

Name	Description
size	grandezza complessiva di un pin **Value:** `0.04` *(type=dimensione normalizzata 0..1)*
thick	spessore dei contorni di un fusibile **Value:** `1`
color	colore dei bordi di un fusibile **Value:** `'black'`

2.10.4 Instance Variables

Name	Description
y_in	coordinata verticale del centro del pin di output **Value:** `0.0` *(type=dimensione normalizzata 0..1)*
locked	variabile indicante se il pin è inattivo **Value:** `False` *(type=boolean)*
text	area di testo dove viene visualizzato lo stato del pin **Value:** `None` *(type=Tkinter.Canvas object ID)*
label	area di testo dove viene visualizzata l'etichetta dell'output **Value:** `None` *(type=Tkinter.Canvas object ID)*

3 Module pla

Il simulatore di Programmable Logic Array.

Author: Alice Plebe, Matteo Cavallaro

Version: 3.0

3.1 Functions

options(a)

Definisce le opzioni accettate dal programma nella linea di comando. Questa funzione utilizza il metodo OptionParser.add_option() per specificare gli attributi di tutte le optioni accettate dal programma

Parameters

`a:` istanza di OptionParser

(type=oggetto OptionParser)

3.2 Variables

Name	Description
args	**Value:** `OptionParser(Pla.usage)`
sim	istanza base di Tkinter **Value:** `Tk()`
pla	istanza della classe Pla, che contiene il simulatore corrente **Value:** `Pla(sim)`

3.3 Class Pla

object
└ **pla.Pla**

La classe del circuito PLA. Comprende i parametri definitori del layout del circuito, include nei suoi attributi le liste di tutti gli oggetti di classe Component che costituiscono il circuito.

Un oggetto Pla viene inizializzato con argomento il Tk root widget.

Notare i tre livelli di coordinate che coesistono nel simulatore:

1. normalizzate [0...1]
2. assolute [pixel]
3. in unità di griglia, corrispondente alla distanza fra due fusibili adiacenti uguale nelle due dimensioni

Tutti i tre sistemi di riferimento hanno origine nell'angolo superiore sinistro.

3.3.1 Methods

__init__(*self*, *root*)

Inizializza il Programmable Logic Array

Parameters

`root:` la Tk root widget

(type=Tkinter object)

Overrides: object.__init__

nor_to_abs(*self*, *n*)

Converte coordinate normalizzate in dimensioni assolute.

Parameters

`n:` coordinata iniziale

(type=dimensione normalizzata 0..1)

Return Value

coordinata in dimensioni assolute [pixel]

Note: la conversione si basa solamente sulla dimensione verticale.

add_and(*self*, *y*)

Crea una porta AND e aggiunge l'oggetto grafico nell'array corrispondente.

Parameters

`y:` coordinata verticale del centro della porta

(type=dimensione normalizzata 0..1)

place_and(*self*)

Crea tutte le porte AND nelle posizioni stabilite dal layout.

add_or(*self*, *x*)

Crea una porta OR e aggiunge l'oggetto grafico nell'array corrispondente.

Parameters

`x`: coordinata orizzontale del centro della porta

(type=dimensione normalizzata 0..1)

place_or(*self*)

Crea tutte le porte OR nelle posizioni stabilite dal layout.

add_not(*self*, *x*)

Crea una porta NOT e aggiunge l'oggetto grafico nell'array corrispondente.

Parameters

`x`: coordinata orizzontale del centro della porta

(type=dimensione normalizzata 0..1)

place_not(*self*)

Crea tutte le porte NOT nelle posizioni stabilite dal layout.

place_inputs(*self*)

Crea tutti i pin di input e conserva gli oggetti grafici negli array. Genera inoltre i relativi collegamenti con le porte NOT.

place_outputs(*self*)

Crea tutti i pin di output e conserva gli oggetti grafici negli array. Genera inoltre i relativi collegamenti con le porte OR.

place_wire_in(*self*)

Realizza i collegamenti tra i fusibili nella matrice di collegamenti tra input e porte AND.

place_wire_out(*self*)

Realizza i collegamenti tra i fusibili nella matrice di collegamenti tra porte AND e OR.

place_fuse_in(*self*)

Crea i fusibili relativi alla matrice IN-AND.

place_fuse_out(*self*)

Crea i fusibili relativi alla matrice AND-OR.

switch_fuse_in(*self*, *tag*)

Realizza lo switch di un fusibile della matrice IN-AND.

Parameters

`tag`: il tag del fusibile.

switch_fuse_out(*self*, *tag*)

Realizza lo switch di un fusibile della matrice AND-OR.

Parameters

`tag`: il tag del fusibile.

get_tag(*self*, *x*, *y*, *comp*=`None`)

Cerca il componente più vicino alla coordinata specificata, e ne restituisce il *tag*.

Se viene specificato l'argomento *comp* la ricerca è limitata a quel tipo di componente i componenti vengono rilevati in un intervallo rettangolare di dimensioni *halo*.

Parameters

`x`: coordinata orizzontale del punto di inizio della ricerca

`y`: coordinata verticale del punto di inizio della ricerca

`comp`: categoria di componente a cui viene limitata la ricerca

(type=Component)

handler(*self*, *event*)

Identifica il componente su cui si è cliccato e esegue l'opportuna operazione.

Tipicamente viene alternato lo stato di un fusibile.

Parameters

`event`: evento catturato da Tkinter.bind, utilizzato per identificare la posizione del mouse

(type=instance)

place_components(*self*)

Posiziona l'intero set di componenti nel layout, e avvia la gestione del mouse.

compute_and(*self*, *r*)

Computa la funzione di una porta logica AND.

Parameters

`r:` la riga di fusibili a cui è legata la porta.

compute_ands(*self*)

Calcola gli output ottenuti dalle porte AND.

compute_out(*self*, *c*)

Computa la funzione di una porta logica OR.

Parameters

`c:` la colonna di fusibili a cui è legata la porta.

compute_outs(*self*)

Calcola gli output ottenuti dalle porte OR, e quindi dell'intero PLA.

run(*self*)

Avvia la computazione dell'output del PLA.

fuse_all(*self*)

Resetta i componenti grafici come al momento di avvio del programma, con i fusibili tutti non collegati.

reset(*self*)

Resetta i componenti grafici come al momento di avvio del programma, con i fusibili tutti collegati.

load(*self*, *circ*)

Carica uno dei circuiti disponibili in libreria.

Parameters

`circ:` circuito da caricare

(type=Circuit)

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

3.3.2 Properties

Name	Description
Inherited from object __class__	

3.3.3 Class Variables

Name	Description
debug	livello di debug, dev'essere = 0 in produzione **Value:** `0`
usage	**Value:** `"""%prog [-x x_size][-i n_inputs][-o n_outputs][-a n_and]"""`
title	titolo della finestra in cui è contenuto il simulatore **Value:** `'PLA simulator'`
halo	gittata del click nel selezionare fusibili **Value:** `0.025` *(type=frazione della lunghezza della finestra)*
l_and	spazio orizzontale dedicato alle AND **Value:** `2.7` *(type=# grid_delta)*
l_or	spazio verticale dedicato alle OR **Value:** `1.3` *(type=# grid_delta)*
upper_block	coordinata verticale del centro delle NOT **Value:** `3.6` *(type=# grid_delta)*
lower_block	minima spaziatura verticale tra outputs e bordo inferiore **Value:** `5` *(type=# grid_delta)*
grid_cols	numero di colonne virtuali nella Tkinter.grid **Value:** `10`
grid_but_extra	spazio aggiuntivo per il posizionamento del pulsante di run **Value:** `1.8` *(type=range [1.0-2.0])*
x_size	dimensione in pixel della lunghezza della finestra del simulatore **Value:** `opts.x_size`
n_inputs	numero di ingressi del circuito, corrispondente al numero di porte NOT **Value:** `opts.n_inputs`
n_outputs	numero di uscite del circuito, corrispondente al numero di OR **Value:** `opts.n_outputs`

continued on next page

Name	Description
n_and	numero di porte AND **Value:** `opts.n_and`

3.3.4 Instance Variables

Name	Description
g_inputs	lista dei componenti grafici della classe Component.InPin istanziati **Value:** `None`
g_outputs	lista dei componenti grafici della classe Component.OutPin istanziati **Value:** `None`
g_and	lista dei componenti grafici della classe Component.Port.And istanziati **Value:** `None`
g_not	lista dei componenti grafici della classe Component.Port.Not istanziati **Value:** `None`
g_or	lista dei componenti grafici della classe Component.Port.Or istanziati **Value:** `None`
g_fuse_in	lista dei componenti grafici della classe Component.Fuse istanziati **Value:** `None`
g_fuse_out	lista dei componenti grafici della classe Component.Fuse istanziati **Value:** `None`
inputs	lista di variabili di classe IntVar usate per gli input **Value:** `[]`
n_or	numero di porte OR **Value:** `0`
n_not	numero di porte NOT **Value:** `0`
grid_delta	passo di griglia del layout del circuito **Value:** `0`
a_ratio	aspect ratio della finestra del simulatore **Value:** `0`

Index