Circuits-Project

1

Generated by Doxygen 1.8.13

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Namespace Documentation

4.1 colorwin Namespace Reference

Classes

- class color
- class withcolor

Enumerations

- enum CW_COLORS {
  red = FOREGROUND_RED | FOREGROUND_INTENSITY, yellow = FOREGROUND_RED | FOREGROUND_GREEN | FOREGROUND_INTENSITY, green = FOREGROUND_GREEN | FOREGROUND_INTENSITY, cyan = FOREGROUND_GREEN | FOREGROUND_BLUE | FOREGROUND_INTENSITY,
  blue = FOREGROUND_BLUE | FOREGROUND_INTENSITY, magenta = FOREGROUND_BLUE | FOREGROUND_RED | FOREGROUND_INTENSITY, white = FOREGROUND_RED | FOREGROUND_GREEN | FOREGROUND_BLUE | FOREGROUND_INTENSITY, gray = FOREGROUND_RED | FOREGROUND_GREEN | FOREGROUND_BLUE,
  grey = FOREGROUND_RED | FOREGROUND_GREEN | FOREGROUND_BLUE, dark_gray = FOREGROUND_INTENSITY, dark_grey = FOREGROUND_INTENSITY }

Functions

- template<typename charT , typename traits >
  std::basic_ostream< charT, traits > & operator<< (std::basic_ostream< charT, traits > &lhs, colorwin::color const &rhs)

4.1.1 Enumeration Type Documentation

4.1.1.1 CW_COLORS

```
enum colorwin::CW_COLORS
```

Enumerator

red	
yellow	
green	
cyan	
blue	
magenta	
white	
gray	
grey	
dark_gray	
dark_grey	

4.1.2 Function Documentation

4.1.2.1 operator<<()

```
template<typename charT , typename traits >
std::basic_ostream<charT, traits>& colorwin::operator<< (
             std::basic_ostream< charT, traits > & lhs,
             colorwin::color const & rhs )
```

Chapter 5

Class Documentation

5.1 Circuit Class Reference

```
#include <Data.h>
```

Public Member Functions

- void Add (Node ∗n)
- bool Remove (Node ∗n)
- bool Remove (const double &val, SEARCH_BY type=ID)
- Node ∗ GetLastNode ()
- Node ∗ GetFirstNode ()
- int GetNumOfNodes ()
- void Read (bool start_with_printing_help=true)
- ∼Circuit ()
- Circuit ()
- Circuit (Circuit ∗c)
- Circuit (Circuit &c)
- Node ∗ GetNode (const double &val, SEARCH_BY type=ID)
- bool HasNode (const double &val, SEARCH_BY type=ID)
- Element ∗ GetElement (char type, const int &id)
- bool HasElement (char type, const int &id)
- bool IsEmpty ()
- Circuit & operator= (Circuit &c)
- Circuit ∗ Copy ()
- Node ∗∗ GetTerminals (Element ∗e)
- Node ∗∗ GetTerminals (Element ∗e, Node ∗&n1, Node ∗&n2)
- void Print ()

5.1.1 Constructor & Destructor Documentation

5.1.1.1 ∼Circuit()

```
Circuit::∼Circuit ( )
```

5.1.1.2 Circuit() [1/3]

```
Circuit::Circuit ( )
```

5.1.1.3 Circuit() [2/3]

```
Circuit::Circuit (
             Circuit * c )
```

5.1.1.4 Circuit() [3/3]

```
Circuit::Circuit (
             Circuit & c )
```

5.1.2 Member Function Documentation

5.1.2.1 Add()

```
void Circuit::Add (
             Node * n )
```

5.1.2.2 Copy()

```
Circuit * Circuit::Copy ( )
```

5.1.2.3 GetElement()

```
Element * Circuit::GetElement (
             char type,
             const int & id )
```

5.1.2.4 GetFirstNode()

```
Node * Circuit::GetFirstNode ( )
```

5.1.2.5 GetLastNode()

```
Node * Circuit::GetLastNode ( )
```

5.1.2.6 GetNode()

```
Node * Circuit::GetNode (
             const double & val,
             SEARCH_BY type = ID )
```

5.1.2.7 GetNumOfNodes()

```
int Circuit::GetNumOfNodes ( )
```

5.1.2.8 GetTerminals() [1/2]

```
Node ** Circuit::GetTerminals (
             Element * e )
```

5.1.2.9 GetTerminals() [2/2]

```
Node ** Circuit::GetTerminals (
             Element * e,
             Node *& n1,
             Node *& n2 )
```

5.1.2.10 HasElement()

```
bool Circuit::HasElement (
             char type,
             const int & id )
```

5.1.2.11 HasNode()

```
bool Circuit::HasNode (
             const double & val,
             SEARCH_BY type = ID )
```

5.1.2.12 IsEmpty()

```
bool Circuit::IsEmpty ( )
```

5.1.2.13 operator=()

```
Circuit & Circuit::operator= (
             Circuit & c )
```

5.1.2.14 Print()

```
void Circuit::Print ( )
```

5.1.2.15 Read()

```
void Circuit::Read (
             bool start_with_printing_help = true )
```

5.1.2.16 Remove() [1/2]

```
bool Circuit::Remove (
             Node * n )
```

5.1.2.17 Remove() [2/2]

```
bool Circuit::Remove (
             const double & val,
             SEARCH_BY type = ID )
```

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

- Headers/Data.h
- Sources/_Circuit.cpp
- Sources/Circuit.cpp

5.2 colorwin::color Class Reference

```
#include <colorwin.hpp>
```

Public Member Functions

- color (CW_COLORS color)
- ∼color ()

Friends

- class withcolor
- template<typename charT , typename traits >
  std::basic_ostream< charT, traits > & operator<< (std::basic_ostream< charT, traits > &lhs, colorwin::color const &rhs)

5.2.1 Constructor & Destructor Documentation

5.2.1.1 color()

```
colorwin::color::color (
            CW_COLORS color )  [inline]
```

5.2.1.2 ∼color()

```
colorwin::color::∼color ( )  [inline]
```

5.2.2 Friends And Related Function Documentation

5.2.2.1 operator<<

```
template<typename charT , typename traits >
std::basic_ostream<charT, traits>& operator<< (
            std::basic_ostream< charT, traits > & lhs,
            colorwin::color const & rhs )  [friend]
```

5.2.2.2 withcolor

```
friend class withcolor  [friend]
```

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

- Headers/colorwin.hpp

5.3 Element Class Reference

```
#include <Data.h>
```

Public Member Functions

- Element (const char &type, const int &id, const double &val)
- Element (const char &type, const int &id, const double &val, const int &node_id)
- char GetType ()
- void ChangeType (const char &c)
- Element ∗ GetNext ()
- Element ∗ GetPrev ()
- int GetId ()
- void ChangeId (const int &num)
- int GetNodeId ()
- double GetValue ()
- void ChangeValue (const double &num)
- Element ∗ Copy ()
- bool operator== (Element &)

Friends

- class Node
- class Circuit

5.3.1 Constructor & Destructor Documentation

5.3.1.1 Element() [1/2]

```
Element::Element (
            const char & type,
            const int & id,
            const double & val )
```

5.3.1.2 Element() [2/2]

```
Element::Element (
             const char & type,
             const int & id,
             const double & val,
             const int & node_id )
```

5.3.2 Member Function Documentation

5.3.2.1 ChangeId()

```
void Element::ChangeId (
             const int & num )
```

5.3.2.2 ChangeType()

```
void Element::ChangeType (
             const char & c )
```

5.3.2.3 ChangeValue()

```
void Element::ChangeValue (
             const double & num )
```

5.3.2.4 Copy()

```
Element * Element::Copy ( )
```

5.3.2.5 GetId()

```
int Element::GetId ( )
```

5.3.2.6 GetNext()

```
Element * Element::GetNext ( )
```

5.3.2.7 GetNodeId()

```
int Element::GetNodeId ( )
```

5.3.2.8 GetPrev()

```
Element * Element::GetPrev ( )
```

5.3.2.9 GetType()

```
char Element::GetType ( )
```

5.3.2.10 GetValue()

```
double Element::GetValue ( )
```

5.3.2.11 operator==()

```
bool Element::operator== (
             Element & e )
```

5.3.3 Friends And Related Function Documentation

5.3.3.1 Circuit

```
friend class Circuit  [friend]
```

5.3.3.2 Node

```
friend class Node  [friend]
```

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

- Headers/Data.h
- Sources/_Element.cpp
- Sources/Element.cpp

5.4 Node Class Reference

```
#include <Data.h>
```

Public Member Functions

- int GetId ()
- void ChangeVolt (const double &v)
- double GetVolt ()
- int GetNumOfElements ()
- bool IsEssential ()
- bool IsEmpty ()
- Element ∗ GetFirstElement ()
- Node ∗ GetNext ()
- Node ∗ GetPrev ()
- void Add (Element ∗e)
- bool Remove (Element ∗e)
- bool Remove (char type, const int &id)
- Element ∗ GetElement (char type, const int &id)
- bool HasElement (char type, const int &id)
- Node (const int &id)
- ∼Node ()
- Node ∗ Copy ()

Friends

- class Circuit

5.4.1 Constructor & Destructor Documentation

5.4.1.1 Node()

```
Node::Node (
             const int & id )
```

5.4.1.2 ∼Node()

```
Node::∼Node ( )
```

5.4.2 Member Function Documentation

5.4.2.1 Add()

```
void Node::Add (
             Element * e )
```

5.4.2.2 ChangeVolt()

```
void Node::ChangeVolt (
             const double & v )
```

5.4.2.3 Copy()

```
Node * Node::Copy ( )
```

5.4.2.4 GetElement()

```
Element * Node::GetElement (
             char type,
             const int & id )
```

5.4.2.5 GetFirstElement()

```
Element * Node::GetFirstElement ( )
```

5.4.2.6 GetId()

```
int Node::GetId ( )
```

5.4.2.7 GetNext()

```
Node * Node::GetNext ( )
```

5.4.2.8 GetNumOfElements()

```
int Node::GetNumOfElements ( )
```

5.4.2.9 GetPrev()

```
Node * Node::GetPrev ( )
```

5.4.2.10 GetVolt()

```
double Node::GetVolt ( )
```

5.4.2.11 HasElement()

```
bool Node::HasElement (
             char type,
             const int & id )
```

5.4.2.12 IsEmpty()

```
bool Node::IsEmpty ( )
```

5.4.2.13 IsEssential()

```
bool Node::IsEssential ( )
```

5.4.2.14 Remove() [1/2]

```
bool Node::Remove (
            Element * e )
```

5.4.2.15 Remove() [2/2]

```
bool Node::Remove (
            char type,
            const int & id )
```

5.4.3 Friends And Related Function Documentation

5.4.3.1 Circuit

```
friend class Circuit  [friend]
```

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

- Headers/Data.h
- Sources/Node.cpp

5.5 colorwin::withcolor Class Reference

```
#include <colorwin.hpp>
```

Public Member Functions

- withcolor (CW_COLORS color)
- int printf (const char ∗format,...)

5.5.1 Constructor & Destructor Documentation

5.5.1.1 withcolor()

```
colorwin::withcolor::withcolor (
            CW_COLORS color )  [inline]
```

5.5.2 Member Function Documentation

5.5.2.1 printf()

```
int colorwin::withcolor::printf (
            const char * format,
             ...  )  [inline]
```

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

- Headers/colorwin.hpp

Chapter 6

File Documentation

6.1 Headers/___.h File Reference

```
#include "Colors.h"
```

Macros

- #define CREDITS YELLOW << "Credits:\n> Hadi Maher\tMahmoud Youssri\n> Mohammed Magdi\tKhaled Sabri\tMahmoud Adas\n\n" << WHITE
- #define PROMPT BLUE << ">>> " << CYAN
- #define HELP GREEN << "Enter the elements node by node \n\nValid Commands:-\n----> x \tend the node\n----> xx \tend all nodes\n----> h \thelp\n----> c \tcredits\n----> p \tprint the completed nodes\n----> Ctrl+z or Ctrl+c\texit program\n----> # \tmake a comment\n\nValid Types:-\n----> r \tResistance\n----> e \tVoltage Source\n----> j \tCurrent Source\n\nAll commands/types could be written in upper or lower case\n\n" << CREDITS

6.1.1 Macro Definition Documentation

6.1.1.1 CREDITS

```
#define CREDITS YELLOW << "Credits:\n> Hadi Maher\tMahmoud Youssri\n> Mohammed Magdi\tKhaled
Sabri\tMahmoud Adas\n\n" << WHITE
```

6.1.1.2 HELP

```
#define HELP GREEN << "Enter the elements node by node \n\nValid Commands:-\n----> x \tend
the node\n----> xx \tend all nodes\n----> h \thelp\n----> c \tcredits\n----> p \tprint the
completed nodes\n----> Ctrl+z or Ctrl+c\texit program\n----> # \tmake a comment\n\nValid
Types:-\n----> r \tResistance\n----> e \tVoltage Source\n----> j \tCurrent Source\n\nAll
commands/types could be written in upper or lower case\n\n" << CREDITS
```

6.1.1.3 PROMPT

```
#define PROMPT BLUE << ">>> " << CYAN
```

6.2 Headers/Colors.h File Reference

Macros

- #define RED "\033[0;31m"
- #define BLUE "\033[0;34m"
- #define GREEN "\033[0;32m"
- #define YELLOW "\033[1;33m"
- #define CYAN "\033[0;36m"
- #define WHITE "\033[0m"

6.2.1 Macro Definition Documentation

6.2.1.1 BLUE

```
#define BLUE "\033[0;34m"
```

6.2.1.2 CYAN

```
#define CYAN "\033[0;36m"
```

6.2.1.3 GREEN

```
#define GREEN "\033[0;32m"
```

6.2.1.4 RED

```
#define RED "\033[0;31m"
```

6.2.1.5 WHITE

```
#define WHITE "\033[0m"
```

6.2.1.6 YELLOW

```
#define YELLOW "\033[1;33m"
```

6.3 Headers/colorwin.hpp File Reference

```
#include <Windows.h>
#include <iostream>
#include <stack>
```

Classes

- class colorwin::color
- class colorwin::withcolor

Namespaces

- colorwin

Enumerations

- enum colorwin::CW_COLORS {
  colorwin::red = FOREGROUND_RED | FOREGROUND_INTENSITY, colorwin::yellow = FOREGROUND_RED | FOREGROUND_GREEN | FOREGROUND_INTENSITY, colorwin::green = FOREGROUND_GREEN | FOREGROUND_INTENSITY, colorwin::cyan = FOREGROUND_GREEN | FOREGROUND_BLUE | FOREGROUND_INTENSITY,
  colorwin::blue = FOREGROUND_BLUE | FOREGROUND_INTENSITY, colorwin::magenta = FOREGROUND_BLUE | FOREGROUND_RED | FOREGROUND_INTENSITY, colorwin::white = FOREGROUND_RED | FOREGROUND_GREEN | FOREGROUND_BLUE | FOREGROUND_INTENSITY, colorwin::gray = FOREGROUND_RED | FOREGROUND_GREEN | FOREGROUND_BLUE,
  colorwin::grey = FOREGROUND_RED | FOREGROUND_GREEN | FOREGROUND_BLUE, colorwin::dark_gray = FOREGROUND_INTENSITY, colorwin::dark_grey = FOREGROUND_INTENSITY }

Functions

- template<typename charT , typename traits >
  std::basic_ostream< charT, traits > & colorwin::operator<< (std::basic_ostream< charT, traits > &lhs, colorwin::color const &rhs)

6.4 Headers/Data.h File Reference

```
#include <iostream>
#include <string>
#include <vector>
#include <list>
#include <tuple>
#include "Errors.h"
#include "___.h"
```

Classes

- class Element
- class Node
- class Circuit

Macros

- #define NULL nullptr

Enumerations

- enum SEARCH_BY { ID, VOLT }
- enum Type { R, E, J }
- enum Command {
  Help, Print_Credits, Print_Circuit, EndNode,
  EndAll, InvalidCommand }

6.4.1 Macro Definition Documentation

6.4.1.1 NULL

```
#define NULL nullptr
```

6.4.2 Enumeration Type Documentation

6.4.2.1 Command

```
enum Command
```

Enumerator

Help	
Print_Credits	
Print_Circuit	
EndNode	
EndAll	
InvalidCommand	

6.4.2.2 SEARCH_BY

```
enum SEARCH_BY
```

Enumerator

ID	
VOLT	

6.4.2.3 Type

```
enum Type
```

Enumerator

R	
E	
J	

6.5 Headers/Errors.h File Reference

```
#include "Colors.h"
#include <cassert>
#include <iostream>
```

Macros

- #define FOR_DEBUGGING 0
- #define HANDLE_SAME_POLARITY RED << "\tTheir is two source elements with the same polarity, deleting the last one\n" << WHITE
- #define HANDLE_DUPLICATE_ELEMENT RED << "\tDuplicate element, deleting it\n" << WHITE

- #define HANDLE_DUPLICATE_WITH_DIFF_VALUES RED << "\tFound two elements having same ID and differnt values, deleting the last one\n" << WHITE
- #define HANDLE_BAD_TYPE_NAME RED << "\tInvalid Type, please re-enter it\n" << WHITE
- #define HANDLE_NEGATIVE_RESISTANCE RED << "\tResistance can't be negative, deleted it\n" << WHITE
- #define HANDLE_EMPTY_NODE RED << "\tLast node is empty, deleting it\n" << WHITE
- #define HANDLE_NODE_WITH_ONE_ELEM RED << "\tFound node with one element, deleting the node and the element\n" << WHITE
- #define HANDLE_INVALID_INPUT RED << "\tInvalid input, type h to see valid commands\n" << WHITE
- #define HANDLE_PARALLEL_DIFF_VOLTAGES RED << "\tFound two parallel voltage sources with different values, deleteing both of them\n" << WHITE
- #define HANDLE_SERIES_DIFF_CURRENTS RED << "\tFound two current sources with different values in series, deleteing both of them\n" << WHITE

Enumerations

- enum error {
  SAME_POLARITY, DUPLICATE_WITH_DIFF_VALUES, DUPLICATE_ELEMENT, BAD_TYPE_NAME,
  NEGATIVE_RESISTANCE, INVALID_STORED_TYPE, LONELY_ELEMENT, DEREF_NULL_PTR,
  DEL_ELEMENT_FROM_WRONG_NODE, NODE_ID_IN_ELEM_UNASSIGNED, INVALID_NODE_ID, IN↩
  VALID_INPUT,
  PARALLEL_DIFF_VOLTAGES, SERIES_DIFF_CURRENTS }

Functions

- void HandleError (const error &err)

6.5.1 Macro Definition Documentation

6.5.1.1 FOR_DEBUGGING

```
#define FOR_DEBUGGING 0
```

6.5.1.2 HANDLE_BAD_TYPE_NAME

```
#define HANDLE_BAD_TYPE_NAME RED << "\tInvalid Type, please re-enter it\n" << WHITE
```

6.5.1.3 HANDLE_DUPLICATE_ELEMENT

```
#define HANDLE_DUPLICATE_ELEMENT RED << "\tDuplicate element, deleting it\n" << WHITE
```

6.5.1.4 HANDLE_DUPLICATE_WITH_DIFF_VALUES

```
#define HANDLE_DUPLICATE_WITH_DIFF_VALUES RED << "\tFound two elements having same ID and
differnt values, deleting the last one\n" << WHITE
```

6.5.1.5 HANDLE_EMPTY_NODE

```
#define HANDLE_EMPTY_NODE RED << "\tLast node is empty, deleting it\n" << WHITE
```

6.5.1.6 HANDLE_INVALID_INPUT

```
#define HANDLE_INVALID_INPUT RED << "\tInvalid input, type h to see valid commands\n" << WH←
ITE
```

6.5.1.7 HANDLE_NEGATIVE_RESISTANCE

```
#define HANDLE_NEGATIVE_RESISTANCE RED << "\tResistance can't be negative, deleted it\n" <<
WHITE
```

6.5.1.8 HANDLE_NODE_WITH_ONE_ELEM

```
#define HANDLE_NODE_WITH_ONE_ELEM RED << "\tFound node with one element, deleting the node and
the element\n" << WHITE
```

6.5.1.9 HANDLE_PARALLEL_DIFF_VOLTAGES

```
#define HANDLE_PARALLEL_DIFF_VOLTAGES RED << "\tFound two parallel voltage sources with different
values, deleteing both of them\n" << WHITE
```

6.5.1.10 HANDLE_SAME_POLARITY

```
#define HANDLE_SAME_POLARITY RED << "\tTheir is two source elements with the same polarity,
deleting the last one\n" << WHITE
```

6.5.1.11 HANDLE_SERIES_DIFF_CURRENTS

```
#define HANDLE_SERIES_DIFF_CURRENTS RED << "\tFound two current sources with different values
in series, deleteing both of them\n" << WHITE
```

6.5.2 Enumeration Type Documentation

6.5.2.1 error

```
enum error
```

Enumerator

SAME_POLARITY	
DUPLICATE_WITH_DIFF_VALUES	
DUPLICATE_ELEMENT	
BAD_TYPE_NAME	
NEGATIVE_RESISTANCE	
INVALID_STORED_TYPE	
LONELY_ELEMENT	
DEREF_NULL_PTR	
DEL_ELEMENT_FROM_WRONG_NODE	
NODE_ID_IN_ELEM_UNASSIGNED	
INVALID_NODE_ID	
INVALID_INPUT	
PARALLEL_DIFF_VOLTAGES	
SERIES_DIFF_CURRENTS	

6.5.3 Function Documentation

6.5.3.1 HandleError()

```
void HandleError (
             const error & err )
```

6.6 Headers/Solving.h File Reference

```
#include "Data.h"
#include "Errors.h"
#include <iostream>
#include <string>
#include <cmath>
```

Functions

- void insert (double ∗∗arr, double ∗arr2, int)
- void print (double ∗∗arr, double ∗arr2, int)
- void swapMat (double arr[ ][50], double arr2[ ][1], int, int, double arr3[ ][50])
- void intElements (double arr[ ][50], int)
- double detMat (double arr[ ][50], int)
- void DoublingMat (double ∗∗arr, int)
- void init (double ∗∗arr, int)
- double ∗ SolvingMat (double ∗∗arr, double ∗arr2, int)
- int sizeOfTheMat (Circuit ∗c)
- int countCircuit (Circuit ∗c)
- double SearchNodeByElement (Element ∗e, Circuit ∗c)
- Node ∗ SearchNodeNon (Element ∗&e, Circuit ∗c, int ID)
- Node ∗ SearchNode (Circuit ∗c, int ID)
- Element ∗ SearchElement (Element ∗e, int, Circuit ∗c)
- void print (Circuit ∗c)
- void voltageTransformation (Circuit ∗&c)
- void solve (Circuit ∗&c)
- void VoltageBack (Circuit ∗in, Circuit ∗&out)
- void SolveNonEss (Circuit ∗&c)
- double Get_Total_Supplied_Power (Circuit ∗circuit)
- double Get_Total_Dissipated_Power (Circuit ∗circuit)
- bool Circuit_Is_Power_Balanced (Circuit ∗circuit)
- double Get_Res_Max (Circuit ∗circuit, Element ∗resistance)
- double Get_Pow_Max (Circuit ∗circuit, Element ∗resistance)
- double Get_Current (Circuit ∗circuit, Element ∗element)
- double Get_Current (Circuit ∗circuit, Element ∗element, Element ∗due_to_element)
- double Get_Power (Circuit ∗circuit, Element ∗element)
- double Get_Power (Circuit ∗circuit, Element ∗element, Element ∗due_to_element)
- double Get_VoltDiff (Circuit ∗circuit, const int node1_id, const int node2_id)
- double Get_VoltDiff (Circuit ∗circuit, const int node1_id, const int node2_id, Element ∗&due_to_element)

6.6.1 Function Documentation

6.6.1.1 Circuit_Is_Power_Balanced()

```
bool Circuit_Is_Power_Balanced (
            Circuit ∗ circuit )
```

6.6.1.2 countCircuit()

```
int countCircuit (
            Circuit ∗ c )
```

6.6.1.3 detMat()

```
double detMat (
             double arr[ ][50],
             int   )
```

6.6.1.4 DoublingMat()

```
void DoublingMat (
             double ** arr,
             int   )
```

6.6.1.5 Get_Current() [1/2]

```
double Get_Current (
             Circuit * circuit,
             Element * element )
```

6.6.1.6 Get_Current() [2/2]

```
double Get_Current (
             Circuit * circuit,
             Element * element,
             Element * due_to_element )
```

6.6.1.7 Get_Pow_Max()

```
double Get_Pow_Max (
             Circuit * circuit,
             Element * resistance )
```

6.6.1.8 Get_Power() [1/2]

```
double Get_Power (
             Circuit * circuit,
             Element * element )
```

6.6.1.9 Get_Power() [2/2]

```
double Get_Power (
            Circuit * circuit,
            Element * element,
            Element * due_to_element )
```

6.6.1.10 Get_Res_Max()

```
double Get_Res_Max (
            Circuit * circuit,
            Element * resistance )
```

6.6.1.11 Get_Total_Dissipated_Power()

```
double Get_Total_Dissipated_Power (
            Circuit * circuit )
```

6.6.1.12 Get_Total_Supplied_Power()

```
double Get_Total_Supplied_Power (
            Circuit * circuit )
```

6.6.1.13 Get_VoltDiff() [1/2]

```
double Get_VoltDiff (
            Circuit * circuit,
            const int node1_id,
            const int node2_id )
```

6.6.1.14 Get_VoltDiff() [2/2]

```
double Get_VoltDiff (
            Circuit * circuit,
            const int node1_id,
            const int node2_id,
            Element *& due_to_element )
```

6.6.1.15 init()

```
void init (
            double ** arr,
            int  )
```

6.6.1.16 insert()

```
void insert (
            double ** arr,
            double * arr2,
            int  )
```

6.6.1.17 intElements()

```
void intElements (
            double arr[ ][50],
            int  )
```

6.6.1.18 print() [1/2]

```
void print (
            double ** arr,
            double * arr2,
            int  )
```

6.6.1.19 print() [2/2]

```
void print (
            Circuit * c )
```

6.6.1.20 SearchElement()

```
Element* SearchElement (
            Element * e,
            int ,
            Circuit * c )
```

6.6.1.21 SearchNode()

```
Node* SearchNode (
             Circuit * c,
             int ID )
```

6.6.1.22 SearchNodeByElement()

```
double SearchNodeByElement (
             Element * e,
             Circuit * c )
```

6.6.1.23 SearchNodeNon()

```
Node* SearchNodeNon (
             Element *& e,
             Circuit * c,
             int ID )
```

6.6.1.24 sizeOfTheMat()

```
int sizeOfTheMat (
             Circuit * c )
```

6.6.1.25 solve()

```
void solve (
             Circuit *& c )
```

6.6.1.26 SolveNonEss()

```
void SolveNonEss (
             Circuit *& c )
```

6.6.1.27 SolvingMat()

```
double* SolvingMat (
             double ** arr,
             double * arr2,
             int   )
```

6.6.1.28 swapMat()

```
void swapMat (
             double arr[ ][50],
             double arr2[ ][1],
             int ,
             int ,
             double arr3[ ][50] )
```

6.6.1.29 VoltageBack()

```
void VoltageBack (
             Circuit * in,
             Circuit *& out )
```

6.6.1.30 voltageTransformation()

```
void voltageTransformation (
             Circuit *& c )
```

6.7 Headers/Test.h File Reference

```
#include "Data.h"
#include "Solving.h"
#include <cstdio>
```

Macros

- #define __RELEASE__
- #define INPUT_FILE "./Test/input"
- #define OUTPUT_FILE "./Test/result"

Functions

- void Redirect_IO (bool redirect)
- void test_hadi_solving2 ()
- int main ()

6.7.1 Macro Definition Documentation

6.7.1.1 __RELEASE__

```
#define __RELEASE__
```

6.7.1.2 INPUT_FILE

```
#define INPUT_FILE "./Test/input"
```

6.7.1.3 OUTPUT_FILE

```
#define OUTPUT_FILE "./Test/result"
```

6.7.2 Function Documentation

6.7.2.1 main()

```
int main ( )
```

6.7.2.2 Redirect_IO()

```
void Redirect_IO (
             bool redirect )
```

6.7.2.3 test_hadi_solving2()

```
void test_hadi_solving2 ( )
```

6.8 Sources/_Circuit.cpp File Reference

```
#include "Data.h"
```

6.9 Sources/_Element.cpp File Reference

```
#include "Data.h"
```

6.10 Sources/_Input.cpp File Reference

```
#include "Data.h"
```

6.11 Sources/_List.cpp File Reference

```
#include "Data.h"
```

6.12 Sources/Circuit.cpp File Reference

```
#include "Data.h"
```

6.13 Sources/Element.cpp File Reference

```
#include "Data.h"
```

6.14 Sources/Errors.cpp File Reference

```
#include "Errors.h"
```

Functions

- void HandleError (const error &err)

6.14.1 Function Documentation

6.14.1.1 HandleError()

```
void HandleError (
            const error & err )
```

6.15 Sources/K&Y_Solving.cpp File Reference

```
#include "Solving.h"
```

Functions

- void Disable_Sources (Circuit ∗c)
- double Voltage (Node ∗n1, Node ∗n2)
- double Ampere (Node ∗n1, Node ∗n2, Element ∗e, Circuit ∗c)
- double Power (Element ∗e, Circuit ∗c)
- void Get_2_Nodes (Element ∗e, Node ∗&n1, Node ∗&n2, Circuit ∗c)
- Circuit ∗ Disable_Sources_superpostion (Circuit ∗c, Element ∗e_temp)
- double Get_Total_Dissipated_Power (Circuit ∗c)
- double Get_Total_Supplied_Power (Circuit ∗c)
- bool Circuit_Is_Power_Balanced (Circuit ∗c)
- double Get_Res_Max (Circuit ∗circuit, Element ∗resistance)
- double Get_Pow_Max (Circuit ∗circuit, Element ∗resistance)
- double Get_Current (Circuit ∗circuit, Element ∗element)
- double Get_Current (Circuit ∗circuit, Element ∗element, Element ∗due_to_element)
- double Get_Power (Circuit ∗circuit, Element ∗element)
- double Get_VoltDiff (Circuit ∗circuit, const int node1_id, const int node2_id)
- double Get_VoltDiff (Circuit ∗circuit, const int node1_id, const int node2_id, Element ∗&due_to_element)

6.15.1 Function Documentation

6.15.1.1 Ampere()

```
double Ampere (
            Node ∗ n1,
            Node ∗ n2,
            Element ∗ e,
            Circuit ∗ c )
```

6.15.1.2 Circuit_Is_Power_Balanced()

```
bool Circuit_Is_Power_Balanced (
            Circuit * c )
```

6.15.1.3 Disable_Sources()

```
void Disable_Sources (
            Circuit * c )
```

6.15.1.4 Disable_Sources_superpostion()

```
Circuit * Disable_Sources_superpostion (
            Circuit * c,
            Element * e_temp )
```

6.15.1.5 Get_2_Nodes()

```
void Get_2_Nodes (
            Element * e,
            Node *& n1,
            Node *& n2,
            Circuit * c )
```

6.15.1.6 Get_Current() [1/2]

```
double Get_Current (
            Circuit * circuit,
            Element * element )
```

6.15.1.7 Get_Current() [2/2]

```
double Get_Current (
            Circuit * circuit,
            Element * element,
            Element * due_to_element )
```

6.15.1.8 Get_Pow_Max()

```
double Get_Pow_Max (
            Circuit * circuit,
            Element * resistance )
```

6.15.1.9 Get_Power()

```
double Get_Power (
            Circuit * circuit,
            Element * element )
```

6.15.1.10 Get_Res_Max()

```
double Get_Res_Max (
            Circuit * circuit,
            Element * resistance )
```

6.15.1.11 Get_Total_Dissipated_Power()

```
double Get_Total_Dissipated_Power (
            Circuit * c )
```

6.15.1.12 Get_Total_Supplied_Power()

```
double Get_Total_Supplied_Power (
            Circuit * c )
```

6.15.1.13 Get_VoltDiff() [1/2]

```
double Get_VoltDiff (
            Circuit * circuit,
            const int node1_id,
            const int node2_id )
```

6.15.1.14 Get_VoltDiff() [2/2]

```
double Get_VoltDiff (
            Circuit * circuit,
            const int node1_id,
            const int node2_id,
            Element *& due_to_element )
```

6.15.1.15 Power()

```
double Power (
            Element * e,
            Circuit * c )
```

6.15.1.16 Voltage()

```
double Voltage (
            Node * n1,
            Node * n2 )
```

6.16 Sources/Main.cpp File Reference

```
#include <iostream>
#include <string>
#include "Data.h"
#include "Test.h"
```

Functions

- int main ()

6.16.1 Function Documentation

6.16.1.1 main()

```
int main ( )
```

6.17 Sources/Node.cpp File Reference

```
#include "Data.h"
```

6.18 Sources/NodeSolving.cpp File Reference

```
#include "data.h"
#include "Solving.h"
```

Functions

- void voltageTransformation (Circuit ∗&C)
- void print (Circuit ∗C)
- void insert (double ∗∗arr, double ∗arr2, int size)
- void print (double ∗∗arr, double ∗arr2, int size)
- void swapMat (double ∗∗arr, double ∗arr2, int size, int numCol, double ∗∗arr3)
- void intElements (double ∗∗arr, int size)
- double detMat (double ∗∗arr, int size)
- int sizeOfTheMat (Circuit ∗c)
- void init (double ∗∗arr, int size)
- void DoublingMat (double ∗∗arr, int size)
- double ∗ SolvingMat (double ∗∗arr, double ∗arr2, int size)
- Element ∗ SearchElement (Element ∗e, int NodeID, Circuit ∗c)
- int countCircuit (Circuit ∗c)
- void solve (Circuit ∗&c)
- void VoltageBack (Circuit ∗in, Circuit ∗&out)
- Node ∗ SearchNode (Circuit ∗c, int ID)
- double SearchNodeByElement (Element ∗e, Circuit ∗c)
- int sizeofNon (Circuit ∗c)
- Node ∗ SearchNodeNon (Element ∗&e, Circuit ∗c, int ID)
- void SolveNonEss (Circuit ∗&c)

6.18.1 Function Documentation

6.18.1.1 countCircuit()

```
int countCircuit (
            Circuit * c )
```

6.18.1.2 detMat()

```
double detMat (
             double ** arr,
             int size )
```

6.18.1.3 DoublingMat()

```
void DoublingMat (
             double ** arr,
             int size )
```

6.18.1.4 init()

```
void init (
             double ** arr,
             int size )
```

6.18.1.5 insert()

```
void insert (
             double ** arr,
             double * arr2,
             int size )
```

6.18.1.6 intElements()

```
void intElements (
             double ** arr,
             int size )
```

6.18.1.7 print() [1/2]

```
void print (
             Circuit * C )
```

6.18.1.8 print() [2/2]

```
void print (
              double ** arr,
              double * arr2,
              int size )
```

6.18.1.9 SearchElement()

```
Element* SearchElement (
              Element * e,
              int NodeID,
              Circuit * c )
```

6.18.1.10 SearchNode()

```
Node* SearchNode (
              Circuit * c,
              int ID )
```

6.18.1.11 SearchNodeByElement()

```
double SearchNodeByElement (
              Element * e,
              Circuit * c )
```

6.18.1.12 SearchNodeNon()

```
Node* SearchNodeNon (
              Element *& e,
              Circuit * c,
              int ID )
```

6.18.1.13 sizeofNon()

```
int sizeofNon (
              Circuit * c )
```

6.18.1.14 sizeOfTheMat()

```
int sizeOfTheMat (
             Circuit * c )
```

6.18.1.15 solve()

```
void solve (
             Circuit *& c )
```

6.18.1.16 SolveNonEss()

```
void SolveNonEss (
             Circuit *& c )
```

6.18.1.17 SolvingMat()

```
double* SolvingMat (
             double ** arr,
             double * arr2,
             int size )
```

6.18.1.18 swapMat()

```
void swapMat (
             double ** arr,
             double * arr2,
             int size,
             int numCol,
             double ** arr3 )
```

6.18.1.19 VoltageBack()

```
void VoltageBack (
             Circuit * in,
             Circuit *& out )
```

6.18.1.20 voltageTransformation()

```
void voltageTransformation (
             Circuit *& C )
```

Index