Pic-Simulator

Generated by Doxygen 1.8.18

Chapter 1

Deprecated List

Member Controller.setCodeViewLabel (int line, String label)

??????????

Member Controller.setCodeViewAdress (int line, int adress)

??????????

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Class Documentation

5.1 Controller Class Reference

class Controller

Public Member Functions

- Controller (Simulator_Window pGui)
- void inizializeMemory ()
- void startMemoryUpdateThread ()
- void openMnemonicView ()
- void showError (String title, String text)
- void startSimu ()
- void stopSimu ()
- void setTextActive (int row) throws BadLocationException
- String[ ] searchJumperMarks (String[ ] pCode)
- String[ ] searchEQUMarks (String[ ] pCode)
- String getEQUValue (String equName)
- void outputToConsole (String in)
- void setSegment (int c1, int c2, int c3, int c4)
- void loadFile (File pFile) throws IOException
- void saveMnemonicCode (String text)
- void setCodeViewCounter (int oldC, int newC)
- void setCodeViewLabel (int line, String label)
- void setCodeViewAdress (int line, int adress)
- void compileCode ()
- void setColumnWidth ()
- void inizializeTables ()
- void executeCommand (String command)
- void closeMnemonicWindow ()
- void clearCodeTable ()
- void refreshIO ()
- void updateFrequency (String selectedItem)
- void saveSRCFile (File fileToSave)
- void loadSRCFile (File file)
- void saveLSTFile (File fileToSave)

Protected Member Functions

- void updateMemoryTable (String value, int x, int y)
- void updateSpecialRegTable (String value, int x, int y)
- void loadMnemonicFromTable ()
- String getIOAnalog_OUT ()
- String getIOAnalog_IN ()
- void setIOAnalog_OUT (int number)
- void setIOAnalog_IN (int number)
- int getJumpersCount ()

Protected Attributes

- Processor proc

  Processor object used to work each code step.
- Memory memory

  memory object used to store the data of microprocessor
- MnemonicParser parser

  Parser Object to parse Mnemonic Code into Binary Code.
- boolean isCompiled = false

  Displaying if the code is compiled.
- String[ ][ ] tableData = new String[32][9]

  The data model to initialize the data table.
- String[ ] jumpers = new String[512]

  An array holding the jumper line number and the mnemonic code.
- String[ ] equ = new String[256]

  An array holding the EQUs.
- String[ ] mnemonicLines

  The mnemonic code.
- int[ ] programCounterList = new int[1024]

  A list of the program counter as key with the dedicated code line as value.
- int programmCounter

  The current position of the program in the code.
- String[ ] code

  The program code as string array. Every code line is one string.
- int codeLength = 0

  The length of the compiled code.
- int frequency = 1000

  The Quartz frequency.

5.1.1 Detailed Description

class Controller

This class is the heart of this Simulator. It is the connection between all classes. Each interaction by a user on the GUI is executed in the controller class. Here are objects of processor, memory, timer, interrupt and watchdog

5.1.2 Constructor & Destructor Documentation

5.1.2.1 Controller()

```
Controller.Controller (
            Simulator_Window pGui )
```

The Constructor, creating a new Memory and MnemonicParser.

Parameters

pGui	Is an Object of Simulator_Window

5.1.3 Member Function Documentation

5.1.3.1 clearCodeTable()

```
void Controller.clearCodeTable ( )
```

Method to clear the code view Table.

5.1.3.2 closeMnemonicWindow()

```
void Controller.closeMnemonicWindow ( )
```

Method to close the Window of the Mnemonic editor.

5.1.3.3 compileCode()

```
void Controller.compileCode ( )
```

Method to compile the code needs to be reworked

5.1.3.4 executeCommand()

```
void Controller.executeCommand (
            String command )
```

This method selects the command which must be executed

Parameters

command	the command to execute as a String

5.1.3.5 getEQUValue()

```
String Controller.getEQUValue (
            String equName )
```

Method to get the EQU value. When no matching EQU is found an empty string will be returned.

Parameters

equName	is the name of the EQU as a String.

Returns

The EQU equivilant as a String.

5.1.3.6 getIOAnalog_IN()

```
String Controller.getIOAnalog_IN ( )  [protected]
```

Method to get the values of the analog IO input pins. For every active pin a 1 will be appended to a string, for every inactive a 0.

Returns

the 8 pin values as a String

5.1.3.7 getIOAnalog_OUT()

```
String Controller.getIOAnalog_OUT ( )  [protected]
```

Method to get the values of the analog IO output pins. For every active pin a 1 will be appended to a string, for every inactive a 0.

Returns

the 8 pin values as a String

5.1.3.8 getJumpersCount()

```
int Controller.getJumpersCount ( )  [protected]
```

Method to get get the amount of jumpers.

Returns

The number of jump marks as integer

5.1.3.9 inizializeMemory()

```
void Controller.inizializeMemory ( )
```

Method to initialize the Memory.

5.1.3.10 inizializeTables()

```
void Controller.inizializeTables ( )
```

Method to initialize the labels of all tables.

5.1.3.11 loadFile()

```
void Controller.loadFile (
            File pFile ) throws IOException
```

Method to load a file. The code table and program counter list will be cleared.

Parameters

pFile	is the File to be loaded.

5.1.3.12 loadMnemonicFromTable()

```
void Controller.loadMnemonicFromTable ( )  [protected]
```

Method to load the program code from the code view table. If labels are found in the 5th column they are added. Otherwise the blank mnemonic code from the 6th column is checked for EQUs. If it contains EQUs the line is added, otherwise the line is added with a space as prefix. Then the mnemonic code string is loaded into the mnemonic editor.

5.1.3.13 loadSRCFile()

```
void Controller.loadSRCFile (
            File file )
```

Method to load a SRC File into the mnemonic view editor.

Parameters

file	The File to load.

5.1.3.14 openMnemonicView()

```
void Controller.openMnemonicView ( )
```

Try to open a new MnemonicView and display it. Then try to load the Mnemonic code from the code view table.

Exceptions

Exception	Is catched and printed.

5.1.3.15 outputToConsole()

```
void Controller.outputToConsole (
            String in )
```

Method to set the values of the analog IO input pins.

Parameters

in	is a String

5.1.3.16 refreshIO()

```
void Controller.refreshIO ( )
```

Method to refresh the Analog IOs. The selected Port of the analog output is read and written to the digital output. The analog input is read and written to the selected Port.

5.1.3.17 saveLSTFile()

```
void Controller.saveLSTFile (
            File fileToSave )
```

Method to save the current code to a selected or new lst file.

Parameters

fileToSave	The File to save.

5.1.3.18 saveMnemonicCode()

```
void Controller.saveMnemonicCode (
```

```
            String text )
```

Method to save the mnemonic code of the opened editor. If the code is not compiled, compileCode will be called.

Parameters

text	is the File to be loaded.

5.1.3.19 saveSRCFile()

```
void Controller.saveSRCFile (
            File fileToSave )
```

Method to save a SRC File.

Parameters

fileToSave	the File to save.

5.1.3.20 searchEQUMarks()

```
String [] Controller.searchEQUMarks (
            String[] pCode )
```

Method to search the EQU marks in the code.

Parameters

pCode	is a String array which holds the program code

Returns

the given program code

5.1.3.21 searchJumperMarks()

```
String [] Controller.searchJumperMarks (
            String[] pCode )
```

Method to check if there are any jump marks in the code, adding to jumpers list and delete from code

Parameters

pCode	is a String array which holds the program code

Returns

the given program code

5.1.3.22 setCodeViewAdress()

```
void Controller.setCodeViewAdress (
            int line,
            int adress )
```

Deprecated ??????????

5.1.3.23 setCodeViewCounter()

```
void Controller.setCodeViewCounter (
            int oldC,
            int newC )
```

Method to set the arrow which displays the active step to a new row.

Parameters

oldC	the old Counter row
newC	the new Counter row

5.1.3.24 setCodeViewLabel()

```
void Controller.setCodeViewLabel (
            int line,
            String label )
```

Deprecated ??????????

5.1.3.25 setColumnWidth()

```
void Controller.setColumnWidth ( )
```

Method to set the column width of the code view table

5.1.3.26 setIOAnalog_IN()

```
void Controller.setIOAnalog_IN (
            int number )  [protected]
```

Method to set the values of the analog IO input pins.

Parameters

number	is the integer (0-255) to set

5.1.3.27 setIOAnalog_OUT()

```
void Controller.setIOAnalog_OUT (
            int number )  [protected]
```

Method to set the values of the analog IO output pins.

Parameters

number	is the integer (0-255) to set

5.1.3.28 setSegment()

```
void Controller.setSegment (
            int c1,
            int c2,
            int c3,
            int c4 )
```

Method to set the 4 values of the 7-Segment display.

Parameters

c1	is the first integer number (most left).
c2	is the second integer number.
c3	is the third integer number.
c4	is the fourth integer number (most right).

5.1.3.29 setTextActive()

```
void Controller.setTextActive (
            int row ) throws BadLocationException
```

Method to highlight the text where the programmcounter points to in Mnemonic Editor

have to be changed to codeTable

5.1.3.30 showError()

```
void Controller.showError (
            String title,
            String text )
```

Method to create a new ErrorDialog with variable title and text and display it.

Parameters

title	is a String and the title of the dialog
text	is a String and the text of the dialog

5.1.3.31 startMemoryUpdateThread()

```
void Controller.startMemoryUpdateThread ( )
```

Starts a thread to cyclic update the memory table.

5.1.3.32 startSimu()

```
void Controller.startSimu ( )
```

Method to start the simulation. If the code is compiled (isCompiles) a new Processor will be created and started. Otherwise and an error window will be displayed

See also

showError

5.1.3.33 stopSimu()

```
void Controller.stopSimu ( )
```

Method to stop the Simulation. The active processor will be stopped via stopThread.

5.1.3.34 updateFrequency()

```
void Controller.updateFrequency (
            String selectedItem )
```

Method to update the selected quarz frequency.

Parameters

selectedItem	The selected Item from the drop down menu.

5.1.3.35 updateMemoryTable()

```
void Controller.updateMemoryTable (
            String value,
            int x,
            int y )  [protected]
```

Method to input a value into a specific cell of the memory table.

Parameters

value	is an String which is put in the cell
x	is an integer referencing to the column
y	is an integer referencing to the row + 1, because the first row are the labels.

5.1.3.36 updateSpecialRegTable()

```
void Controller.updateSpecialRegTable (
            String value,
            int x,
            int y )  [protected]
```

Method to input a value into a specific cell of the special register table.

Parameters

value	is an String which is put in the cell
x	is an integer referencing to the column
y	is an integer referencing to the row

5.1.4 Member Data Documentation

5.1.4.1 code

```
String [] Controller.code  [protected]
```

The program code as string array. Every code line is one string.

5.1.4.2 codeLength

```
int Controller.codeLength = 0  [protected]
```

The length of the compiled code.

5.1.4.3 equ

```
String [] Controller.equ = new String[256]  [protected]
```

An array holding the EQUs.

One entry holds the "original" before the EQU, followed by an ':' appended with the value after the EQU.

5.1.4.4 frequency

```
int Controller.frequency = 1000  [protected]
```

The Quartz frequency.

5.1.4.5 isCompiled

```
boolean Controller.isCompiled = false  [protected]
```

Displaying if the code is compiled.

5.1.4.6 jumpers

```
String [] Controller.jumpers = new String[512]  [protected]
```

An array holding the jumper line number and the mnemonic code.

The jumpers are holding the mnemonic code line with an ':' and the program counter appended. They are listed starting at 0.

5.1.4.7 memory

```
Memory Controller.memory  [protected]
```

memory object used to store the data of microprocessor

5.1.4.8 mnemonicLines

```
String [] Controller.mnemonicLines  [protected]
```

The mnemonic code.

5.1.4.9 parser

```
MnemonicParser Controller.parser  [protected]
```

Parser Object to parse Mnemonic Code into Binary Code.

5.1.4.10 proc

```
Processor Controller.proc  [protected]
```

Processor object used to work each code step.

5.1.4.11 programCounterList

```
int [] Controller.programCounterList = new int[1024]  [protected]
```

A list of the program counter as key with the dedicated code line as value.

5.1.4.12 programmCounter

```
int Controller.programmCounter  [protected]
```

The current position of the program in the code.

5.1.4.13 tableData

```
String [][] Controller.tableData = new String[32][9]  [protected]
```

The data model to initialize the data table.

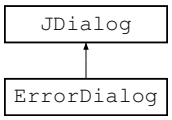
The documentation for this class was generated from the following file:

- F:/Coding/DH/Pic_Simulator/Simulator/src/Controller.java

5.2 ErrorDialog Class Reference

class ErrorDialog

Inheritance diagram for ErrorDialog:



Public Member Functions

- ErrorDialog ()

Static Public Member Functions

- static void main (String[ ] args)

Protected Attributes

- JLabel lbl_ErrorTitle
- JLabel lbl_ErrorText

5.2.1 Detailed Description

class ErrorDialog

An Error Window

5.2.2 Constructor & Destructor Documentation

5.2.2.1 ErrorDialog()

```
ErrorDialog.ErrorDialog ( )
```

Create the dialog.

5.2.3 Member Function Documentation

5.2.3.1 main()

```
static void ErrorDialog.main (
            String[] args )  [static]
```

Launch the application.

5.2.4 Member Data Documentation

5.2.4.1 lbl_ErrorText

```
JLabel ErrorDialog.lbl_ErrorText  [protected]
```

5.2.4.2 lbl_ErrorTitle

```
JLabel ErrorDialog.lbl_ErrorTitle  [protected]
```

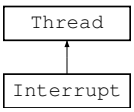
The documentation for this class was generated from the following file:

- F:/Coding/DH/Pic_Simulator/Simulator/src/ErrorDialog.java

5.3 Interrupt Class Reference

class Interrupt

Inheritance diagram for Interrupt:



Public Member Functions

- Interrupt (Controller pCtr)
- void run ()

5.3.1 Detailed Description

class Interrupt

This class is a basic implementation of the interrupt functionality

If the interrupt is enabled, this class checks the interrupt IO pins and jumps to the specific adress 0x04

5.3.2 Constructor & Destructor Documentation

5.3.2.1 Interrupt()

```
Interrupt.Interrupt (
            Controller pCtr )
```

5.3.3 Member Function Documentation

5.3.3.1 run()

```
void Interrupt.run ( )
```

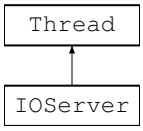
The documentation for this class was generated from the following file:

- F:/Coding/DH/Pic_Simulator/Simulator/src/Interrupt.java

5.4 IOServer Class Reference

class IOServer

Inheritance diagram for IOServer:



Public Member Functions

- IOServer (Controller pCtr)

Protected Member Functions

- void startServer ()
- java.net.Socket waitForClient () throws IOException
- void readMessage ()
- void sendMessage ()

Protected Attributes

- int serverPort

5.4.1 Detailed Description

class IOServer

used to connect simulator to a outsourced software with hardware communication

5.4.2 Constructor & Destructor Documentation

5.4.2.1 IOServer()

```
IOServer.IOServer (
            Controller pCtr )
```

5.4.3 Member Function Documentation

5.4.3.1 readMessage()

```
void IOServer.readMessage ( )  [protected]
```

5.4.3.2 sendMessage()

```
void IOServer.sendMessage ( )  [protected]
```

5.4.3.3 startServer()

```
void IOServer.startServer ( )  [protected]
```

5.4.3.4 waitForClient()

```
java.net.Socket IOServer.waitForClient ( ) throws IOException  [protected]
```

5.4.4 Member Data Documentation

5.4.4.1 serverPort

```
int IOServer.serverPort  [protected]
```

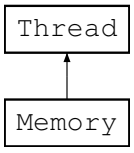
The documentation for this class was generated from the following file:

- F:/Coding/DH/Pic_Simulator/Simulator/src/IOServer.java

5.5 Memory Class Reference

class Memory

Inheritance diagram for Memory:



Public Member Functions

- Memory (Controller pCtr)
- void run ()
- void clearProgMem ()

Protected Member Functions

- void set_INDF (int bit, int value)
- void set_TMR0 (int bit, int value)
- void set_PCL (int bit, int value)
- void set_STATUS (int bit, int value)
- void set_FSR (int bit, int value)
- void set_PORTA (int bit, int value)
- void set_PORTB (int bit, int value)
- void set_EEDATA (int bit, int value)
- void set_EEADR (int bit, int value)
- void set_PCLATH (int bit, int value)
- void set_INTCON (int bit, int value)
- void set_OPTION_REG (int bit, int value)
- void set_TRISA (int bit, int value)
- void set_TRISB (int bit, int value)
- void set_EECON1 (int bit, int value)
- void set_EECON2 (int bit, int value)
- void set_SRAM (int fileaddress, int bit, int value)
- void set_SRAM (int fileaddress, int value)
- void set_CARRYFLAG (int c)
- int get_CARRYFLAG ()
- int get_Memory (int fileaddress, int bit)
- int get_Memory (int fileaddress)
- void set_PROGRAMMCOUNTER (int counter)
- int get_PROGRAMMCOUNTER ()
- void set_WREGISTER (int bit, int value)
- void set_WREGISTER (int value)
- int get_WREGISTER (int bit)
- int get_WREGISTER ()
- void pushToStack (int adr)
- int popFromStack ()

Protected Attributes

- int[ ][ ] dataMemory = new int[256][8]
- int[ ] programMemory = new int[1024]
- int programmcounter
- int[ ] w_register = new int[8]

  w_register storage for operations
- Stack< Integer > stack = new Stack<Integer>()

  the stack is used to store the pushed adresses by a call command

5.5.1 Detailed Description

class Memory

This class is the basic implementation of the controller memory There a various variables for w_register, carry flag or the stack

5.5.2 Constructor & Destructor Documentation

5.5.2.1 Memory()

```
Memory.Memory (
            Controller pCtr )
```

5.5.3 Member Function Documentation

5.5.3.1 clearProgMem()

```
void Memory.clearProgMem ( )
```

- used to clear the programMemory reset value is ff

5.5.3.2 get_CARRYFLAG()

```
int Memory.get_CARRYFLAG ( )  [protected]
```

5.5.3.3 get_Memory() [1/2]

```
int Memory.get_Memory (
            int fileaddress )  [protected]
```

5.5.3.4 get_Memory() [2/2]

```
int Memory.get_Memory (
            int fileaddress,
            int bit )  [protected]
```

5.5.3.5 get_PROGRAMMCOUNTER()

```
int Memory.get_PROGRAMMCOUNTER ( )  [protected]
```

5.5.3.6 get_WREGISTER() [1/2]

```
int Memory.get_WREGISTER ( )  [protected]
```

5.5.3.7 get_WREGISTER() [2/2]

```
int Memory.get_WREGISTER (
            int bit )  [protected]
```

- function to get ther specific bit of w register

Parameters

bit	bit index

Returns

value of the w register specific bit

5.5.3.8 popFromStack()

```
int Memory.popFromStack ( )  [protected]
```

- used to pop the needed programm counter from stack

Returns

is the popped adress

5.5.3.9 pushToStack()

```
void Memory.pushToStack (
            int adr )  [protected]
```

- used to push the current programmcounter+1 on the stack

Parameters

adr	is the adress to push

5.5.3.10 run()

```
void Memory.run ( )
```

5.5.3.11 set_CARRYFLAG()

```
void Memory.set_CARRYFLAG (
            int c )  [protected]
```

5.5.3.12 set_EEADR()

```
void Memory.set_EEADR (
            int bit,
            int value )  [protected]
```

5.5.3.13 set_EECON1()

```
void Memory.set_EECON1 (
            int bit,
            int value )  [protected]
```

5.5.3.14 set_EECON2()

```
void Memory.set_EECON2 (
            int bit,
            int value )  [protected]
```

5.5.3.15 set_EEDATA()

```
void Memory.set_EEDATA (
             int bit,
             int value )  [protected]
```

5.5.3.16 set_FSR()

```
void Memory.set_FSR (
             int bit,
             int value )  [protected]
```

5.5.3.17 set_INDF()

```
void Memory.set_INDF (
             int bit,
             int value )  [protected]
```

5.5.3.18 set_INTCON()

```
void Memory.set_INTCON (
             int bit,
             int value )  [protected]
```

5.5.3.19 set_OPTION_REG()

```
void Memory.set_OPTION_REG (
             int bit,
             int value )  [protected]
```

5.5.3.20 set_PCL()

```
void Memory.set_PCL (
             int bit,
             int value )  [protected]
```

5.5.3.21 set_PCLATH()

```
void Memory.set_PCLATH (
            int bit,
            int value )  [protected]
```

5.5.3.22 set_PORTA()

```
void Memory.set_PORTA (
            int bit,
            int value )  [protected]
```

5.5.3.23 set_PORTB()

```
void Memory.set_PORTB (
            int bit,
            int value )  [protected]
```

5.5.3.24 set_PROGRAMMCOUNTER()

```
void Memory.set_PROGRAMMCOUNTER (
            int counter )  [protected]
```

5.5.3.25 set_SRAM() [1/2]

```
void Memory.set_SRAM (
            int fileaddress,
            int bit,
            int value )  [protected]
```

5.5.3.26 set_SRAM() [2/2]

```
void Memory.set_SRAM (
            int fileaddress,
            int value )  [protected]
```

5.5.3.27 set_STATUS()

```
void Memory.set_STATUS (
            int bit,
            int value )  [protected]
```

5.5.3.28 set_TMR0()

```
void Memory.set_TMR0 (
            int bit,
            int value )  [protected]
```

5.5.3.29 set_TRISA()

```
void Memory.set_TRISA (
            int bit,
            int value )  [protected]
```

5.5.3.30 set_TRISB()

```
void Memory.set_TRISB (
            int bit,
            int value )  [protected]
```

5.5.3.31 set_WREGISTER() [1/2]

```
void Memory.set_WREGISTER (
            int bit,
            int value )  [protected]
```

5.5.3.32 set_WREGISTER() [2/2]

```
void Memory.set_WREGISTER (
            int value )  [protected]
```

5.5.4 Member Data Documentation

5.5.4.1 dataMemory

```
int [][] Memory.dataMemory = new int[256][8]  [protected]
```

this memory will store the data memory 00 to 7F is the first bank 80 to FF is the second bank

5.5.4.2 programmcounter

```
int Memory.programmcounter  [protected]
```

counter on which line the processor is default es 0 to indicate a reset

5.5.4.3 programMemory

```
int [] Memory.programMemory = new int[1024]  [protected]
```

this is the storage for the program code 0000 is the reset and 0004 is the interrupt value

5.5.4.4 stack

```
Stack<Integer> Memory.stack = new Stack<Integer>()  [protected]
```

the stack is used to store the pushed adresses by a call command

5.5.4.5 w_register

```
int [] Memory.w_register = new int[8]  [protected]
```

w_register storage for operations

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

- F:/Coding/DH/Pic_Simulator/Simulator/src/Memory.java

5.6 MnemonicParser Class Reference

class MnemonicParser

Public Member Functions

- MnemonicParser (Controller pCtr)
  constructor
- String fromMnemToHex (String c, int line)
- String hexToBinary (String hex)

5.6.1 Detailed Description

class MnemonicParser

This class is used to parse mnemonic code into binary assembler code

5.6.2 Constructor & Destructor Documentation

5.6.2.1 MnemonicParser()

```
MnemonicParser.MnemonicParser (
            Controller pCtr )
```

constructor

The constructor is used to set the parameter given by Controller class with creation to the local Controller object

Parameters

pCtr	temporary object of Controller

5.6.3 Member Function Documentation

5.6.3.1 fromMnemToHex()

```
String MnemonicParser.fromMnemToHex (
            String c,
            int line )
```

function to parse a line of mnemonic assembler code into binary code

Parameters

c	the line to parse
line	to display the line where an error occured

Returns

the binary string of c

5.6.3.2 hexToBinary()

```
String MnemonicParser.hexToBinary (
            String hex )
```

- Converter from hex to bin string

Parameters

hex	

Returns

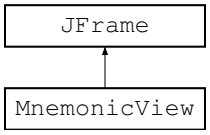
The documentation for this class was generated from the following file:

- F:/Coding/DH/Pic_Simulator/Simulator/src/MnemonicParser.java

5.7 MnemonicView Class Reference

class MnemonicView

Inheritance diagram for MnemonicView:



Public Member Functions

- MnemonicView (Controller pCtr)

Protected Attributes

- JTextArea txtArea_mnemonic

5.7.1 Detailed Description

class MnemonicView

The Viewer to Display and edit Mnemonic Text

5.7.2 Constructor & Destructor Documentation

5.7.2.1 MnemonicView()

```
MnemonicView.MnemonicView (
            Controller pCtr )
```

Create the frame.

5.7.3 Member Data Documentation

5.7.3.1 txtArea_mnemonic

```
JTextArea MnemonicView.txtArea_mnemonic  [protected]
```

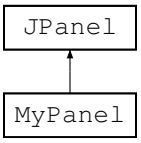
The documentation for this class was generated from the following file:

- F:/Coding/DH/Pic_Simulator/Simulator/src/MnemonicView.java

5.8 MyPanel Class Reference

class MyPanel

Inheritance diagram for MyPanel:



Public Member Functions

- MyPanel ()
- void setChars (int c1, int c2, int c3, int c4)
- void paintFirst (Graphics g, int Char)
- void paintSecond (Graphics g, int Char)
- void paintThird (Graphics g, int Char)
- void paintFourth (Graphics g, int Char)
- Dimension getPreferredSize ()

Protected Member Functions

- void paintComponent (Graphics g)

5.8.1 Detailed Description

class MyPanel

The Panel to display the 7-segment Display.

5.8.2 Constructor & Destructor Documentation

5.8.2.1 MyPanel()

```
MyPanel.MyPanel ( )
```

5.8.3 Member Function Documentation

5.8.3.1 getPreferredSize()

```
Dimension MyPanel.getPreferredSize ( )
```

5.8.3.2 paintComponent()

```
void MyPanel.paintComponent (
            Graphics g )  [protected]
```

5.8.3.3 paintFirst()

```
void MyPanel.paintFirst (
            Graphics g,
            int Char )
```

5.8.3.4 paintFourth()

```
void MyPanel.paintFourth (
            Graphics g,
            int Char )
```

5.8.3.5 paintSecond()

```
void MyPanel.paintSecond (
            Graphics g,
            int Char )
```

5.8.3.6 paintThird()

```
void MyPanel.paintThird (
            Graphics g,
            int Char )
```

5.8.3.7 setChars()

```
void MyPanel.setChars (
            int c1,
            int c2,
            int c3,
            int c4 )
```

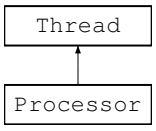
The documentation for this class was generated from the following file:

- F:/Coding/DH/Pic_Simulator/Simulator/src/MyPanel.java

5.9 Processor Class Reference

class Processor

Inheritance diagram for Processor:



Public Member Functions

- Processor (Controller pC)
- void run ()
- void stopThread ()

5.9.1 Detailed Description

class Processor

This class is the main computing unit. if the thread is started, the programm walks through the program memory. if there is a command to execute it will be executed and the Counter arrow is set to the next one

5.9.2 Constructor & Destructor Documentation

5.9.2.1 Processor()

```
Processor.Processor (
            Controller pC )
```

5.9.3 Member Function Documentation

5.9.3.1 run()

```
void Processor.run ( )
```

5.9.3.2 stopThread()

```
void Processor.stopThread ( )
```

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

- F:/Coding/DH/Pic_Simulator/Simulator/src/Processor.java

5.10 Simulator_Window Class Reference

class Simulator_Window

Public Member Functions

- Simulator_Window ()
- void SetData (Object obj, int row_index, int col_index)
- String getData (int row, int col)
- void setSegment (int c1, int c2, int c3, int c4)
- void setSpecialData (Object obj, int row_index, int col_index)
- void openHelp ()

Static Public Member Functions

- static void main (String[ ] args)

Protected Attributes

- TableModel dataModel
- MyPanel panel_segmentCanvas
- JTable table_Code
- DefaultTableModel tbl_code
- DefaultTableModel tbl_memory
- DefaultTableModel tbl_special
- DefaultTableModel tbl_status
- DefaultTableModel tbl_pcl
- JTable table_Memory
- JTable table_special_regs
- JTextArea txtArea_Console
- JComboBox comboBox_AnalogIn_PortSelector
- JRadioButton rb_io_out_1
- JRadioButton rb_io_out_2
- JRadioButton rb_io_out_3
- JRadioButton rb_io_out_4
- JRadioButton rb_io_out_5
- JRadioButton rb_io_out_6
- JRadioButton rb_io_out_7
- JRadioButton rb_io_out_8
- JComboBox comboBox_AnalogOUT_PortSelector
- JRadioButton rb_io_in_1
- JRadioButton rb_io_in_2
- JRadioButton rb_io_in_3
- JRadioButton rb_io_in_4
- JRadioButton rb_io_in_5
- JRadioButton rb_io_in_6
- JRadioButton rb_io_in_7
- JRadioButton rb_io_in_8
- JComboBox comboBox_quarzFrequency

5.10.1 Detailed Description

class Simulator_Window

Grafical user interface for Pic Simualtor.

REWORK NEEDED

5.10.2 Constructor & Destructor Documentation

5.10.2.1 Simulator_Window()

```
Simulator_Window.Simulator_Window ( )
```

Create the application.

5.10.3 Member Function Documentation

5.10.3.1 getData()

```
String Simulator_Window.getData (
            int row,
            int col )
```

5.10.3.2 main()

```
static void Simulator_Window.main (
            String[] args )  [static]
```

Launch the application.

5.10.3.3 openHelp()

```
void Simulator_Window.openHelp ( )
```

5.10.3.4 SetData()

```
void Simulator_Window.SetData (
            Object obj,
            int row_index,
            int col_index )
```

5.10.3.5 setSegment()

```
void Simulator_Window.setSegment (
             int c1,
             int c2,
             int c3,
             int c4 )
```

5.10.3.6 setSpecialData()

```
void Simulator_Window.setSpecialData (
             Object obj,
             int row_index,
             int col_index )
```

5.10.4 Member Data Documentation

5.10.4.1 comboBox_AnalogIn_PortSelector

```
JComboBox Simulator_Window.comboBox_AnalogIn_PortSelector  [protected]
```

5.10.4.2 comboBox_AnalogOUT_PortSelector

```
JComboBox Simulator_Window.comboBox_AnalogOUT_PortSelector  [protected]
```

5.10.4.3 comboBox_quarzFrequency

```
JComboBox Simulator_Window.comboBox_quarzFrequency  [protected]
```

5.10.4.4 dataModel

```
TableModel Simulator_Window.dataModel  [protected]
```

5.10.4.5 panel_segmentCanvas

```
MyPanel Simulator_Window.panel_segmentCanvas  [protected]
```

5.10.4.6 rb_io_in_1

```
JRadioButton Simulator_Window.rb_io_in_1  [protected]
```

5.10.4.7 rb_io_in_2

```
JRadioButton Simulator_Window.rb_io_in_2  [protected]
```

5.10.4.8 rb_io_in_3

```
JRadioButton Simulator_Window.rb_io_in_3  [protected]
```

5.10.4.9 rb_io_in_4

```
JRadioButton Simulator_Window.rb_io_in_4  [protected]
```

5.10.4.10 rb_io_in_5

```
JRadioButton Simulator_Window.rb_io_in_5  [protected]
```

5.10.4.11 rb_io_in_6

```
JRadioButton Simulator_Window.rb_io_in_6  [protected]
```

5.10.4.12 rb_io_in_7

```
JRadioButton Simulator_Window.rb_io_in_7  [protected]
```

5.10.4.13 rb_io_in_8

```
JRadioButton Simulator_Window.rb_io_in_8  [protected]
```

5.10.4.14 rb_io_out_1

```
JRadioButton Simulator_Window.rb_io_out_1  [protected]
```

5.10.4.15 rb_io_out_2

```
JRadioButton Simulator_Window.rb_io_out_2  [protected]
```

5.10.4.16 rb_io_out_3

```
JRadioButton Simulator_Window.rb_io_out_3  [protected]
```

5.10.4.17 rb_io_out_4

```
JRadioButton Simulator_Window.rb_io_out_4  [protected]
```

5.10.4.18 rb_io_out_5

```
JRadioButton Simulator_Window.rb_io_out_5  [protected]
```

5.10.4.19 rb_io_out_6

```
JRadioButton Simulator_Window.rb_io_out_6  [protected]
```

5.10.4.20 rb_io_out_7

```
JRadioButton Simulator_Window.rb_io_out_7  [protected]
```

5.10.4.21 rb_io_out_8

```
JRadioButton Simulator_Window.rb_io_out_8  [protected]
```

5.10.4.22 table_Code

```
JTable Simulator_Window.table_Code  [protected]
```

5.10.4.23 table_Memory

```
JTable Simulator_Window.table_Memory  [protected]
```

5.10.4.24 table_special_regs

```
JTable Simulator_Window.table_special_regs  [protected]
```

5.10.4.25 tbl_code

```
DefaultTableModel Simulator_Window.tbl_code  [protected]
```

5.10.4.26 tbl_memory

```
DefaultTableModel Simulator_Window.tbl_memory  [protected]
```

5.10.4.27 tbl_pcl

```
DefaultTableModel Simulator_Window.tbl_pcl  [protected]
```

5.10.4.28 tbl_special

```
DefaultTableModel Simulator_Window.tbl_special  [protected]
```

5.10.4.29 tbl_status

```
DefaultTableModel Simulator_Window.tbl_status  [protected]
```

5.10.4.30 txtArea_Console

```
JTextArea Simulator_Window.txtArea_Console  [protected]
```

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

- F:/Coding/DH/Pic_Simulator/Simulator/src/Simulator_Window.java

5.11 Timer Class Reference

class Timer

5.11.1 Detailed Description

class Timer

This class is used for implementation of a simple Timer

If the Timer is active, the Timer Register is incremented each tick by specific Time

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

- F:/Coding/DH/Pic_Simulator/Simulator/src/Timer.java

5.12 Watchdog Class Reference

class Watchdog

5.12.1 Detailed Description

class Watchdog

This class is a basic implementation of a watchdog timer

The Watchdog checks wether the processor is in a deadlock. If the watchdog found a deadlock or an error, he automaticly sets the reset flag

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

- F:/Coding/DH/Pic_Simulator/Simulator/src/Watchdog.java

Chapter 6

File Documentation

6.1 F:/Coding/DH/Pic_Simulator/Simulator/src/Controller.java File Reference

Classes

- class Controller
  class Controller

6.2 F:/Coding/DH/Pic_Simulator/Simulator/src/ErrorDialog.java File Reference

Classes

- class ErrorDialog
  class ErrorDialog

6.3 F:/Coding/DH/Pic_Simulator/Simulator/src/Interrupt.java File Reference

Classes

- class Interrupt
  class Interrupt

6.4 F:/Coding/DH/Pic_Simulator/Simulator/src/IOServer.java File Reference

Classes

- class IOServer
  class IOServer

6.5 F:/Coding/DH/Pic_Simulator/Simulator/src/Memory.java File Reference

Classes

- class Memory
  class Memory

6.6 F:/Coding/DH/Pic_Simulator/Simulator/src/MnemonicParser.java File Reference

Classes

- class MnemonicParser
  class MnemonicParser

6.7 F:/Coding/DH/Pic_Simulator/Simulator/src/MnemonicView.java File Reference

Classes

- class MnemonicView
  class MnemonicView

6.8 F:/Coding/DH/Pic_Simulator/Simulator/src/MyPanel.java File Reference

Classes

- class MyPanel
  class MyPanel

6.9 F:/Coding/DH/Pic_Simulator/Simulator/src/Processor.java File Reference

Classes

- class Processor
  class Processor

6.10 F:/Coding/DH/Pic_Simulator/Simulator/src/Simulator_Window.java File Reference

Classes

- class Simulator_Window

  class Simulator_Window

6.11 F:/Coding/DH/Pic_Simulator/Simulator/src/Timer.java File Reference

Classes

- class Timer

  class Timer

6.12 F:/Coding/DH/Pic_Simulator/Simulator/src/Watchdog.java File Reference

Classes

- class Watchdog

  class Watchdog

Index