OneUp Wi' 11 Simulator

Generated by Doxygen 1.7.2

Thu Jan 20 2011 00:26:43

Contents

Chapter 1

Class Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

Class Documentation

3.1 iInterpreter Class Reference

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

- iInterpreter.h

3.2 iMemory Class Reference

Public Member Functions

- virtual void **setAddress** (const iWord &) const =0
- virtual void **setSize** (const int lowerbound, const int upperbound) const =0
- virtual bool **Initialize** () const =0
- virtual Word **Load** (const iWord &) const =0
- virtual bool **Store** (const iWord &address, const iWord &value)=0

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

- iMemory.h

3.3 iRegister Class Reference

Public Member Functions

- virtual Word **getValue** () const =0
- virtual void **Add** (const iWord &)=0
- virtual Register **Add** (const iRegister &) const =0
- virtual void **operator+** (const iWord &)=0

- virtual Register **operator+** (const iRegister &) const =0
- virtual void **Subtract** (const iWord &)=0
- virtual Register **Subtract** (const iRegister &) const =0
- virtual void **operator-** (const iWord &)=0
- virtual Register **operator-** (const iRegister &) const =0
- virtual void **And** (const iWord &)=0
- virtual Register **And** (const iRegister &) const =0
- virtual void **Or** (const iWord &)=0
- virtual Register **Or** (const iRegister &) const =0
- virtual void **Not** ()=0
- virtual Register **Not** () const =0
- virtual void **Store** (const iWord &)=0
- virtual void **Store** (const iRegister &)=0
- virtual Register & **operator=** (const iWord &)=0
- virtual Register & **operator=** (const Register)=0
- virtual Register & **operator++** ()=0
- virtual Register & **operator++** (int)=0

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

- iRegister.h

3.4 iSimulator Class Reference

Public Member Functions

- virtual bool **Initialize** (const char ∗)=0
- virtual bool **Add** (const REGISTER_ID DR, const REGISTER_ID SR1, const REGISTER_-ID SR2)=0
- virtual bool **Add** (const REGISTER_ID DR, const REGISTER_ID SR1, const iWord &immediate)=0
- virtual bool **And** (const REGISTER_ID DR, const REGISTER_ID SR1, const REGISTER_-ID SR2)=0
- virtual bool **And** (const REGISTER_ID DR, const REGISTER_ID SR1, const iWord &immediate)=0
- virtual bool **Branch** (const iWord &address)=0
- virtual bool **Debug** ()=0
- virtual bool **JSR** (const iWord &)=0
- virtual bool **JSRR** (const iWord &baseR, const iWord &address)=0
- virtual bool **Load** (const REGISTER_ID DR, const iWord &address)=0
- virtual bool **LDI** (const REGISTER_ID DR, const iWord &address)=0
- virtual bool **LDR** (const REGISTER_ID DR, const iWord &baseR, const iWord &address)=0
- virtual bool **Not** (const REGISTER_ID DR, const REGISTER_ID SR)=0
- virtual bool **Ret** ()=0

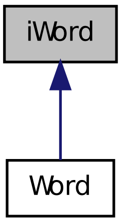
- virtual bool **Store** (const REGISTER_ID DR, const iWord &address)=0
- virtual bool **STI** (const REGISTER_ID DR, const iWord &address)=0
- virtual bool **STR** (const REGISTER_ID DR, const iWord &baseR, const iWord &address)=0
- virtual bool **Trap** (const iWord &address)=0

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

- iSimulator.h

3.5 iWord Class Reference

Inheritance diagram for iWord:



Public Member Functions

- virtual int **toInt** () const =0
- virtual int **toInt2Complement** () const =0
- virtual std::string **toStr** () const =0
- virtual std::string **toHex** () const =0
- virtual bool **fromInt** (int)=0
- virtual bool **fromStr** (const std::string &)=0
- virtual bool **fromHex** (const std::string &)=0
- virtual Word **Add** (const iWord &) const =0
- virtual Word **operator+** (const iWord &) const =0
- virtual Word **Subtract** (const iWord &) const =0
- virtual Word **operator-** (const iWord &) const =0
- virtual Word **And** (const iWord &) const =0
- virtual Word **Or** (const iWord &) const =0
- virtual Word **Not** () const =0

- virtual void copy (const iWord &)=0

  Copies a word.

- virtual Word & operator= (const Word)=0

  A standard assignment operator.

- virtual iWord & operator++ ()=0
- virtual iWord & operator++ (int)=0

  A standard post-increment operator.

- virtual bool operator[] (int) const =0

  An accessor to the "i"th bit of the value.

- virtual void **print** () const =0

3.5.1 Member Function Documentation

3.5.1.1 virtual void iWord::copy (const iWord &) `[pure virtual]`

Copies a word.

Parameters

`out`	*The*	value to be copied.

Postcondition

> The caller equals that parameter.

Equivalent to the assignment "caller = parameter".

Implemented in Word.

3.5.1.2 virtual iWord& iWord::operator++ ( ) `[pure virtual]`

A standard pre-increment operator.

Returns

> A reference to itself.

The object increments its value BEFORE the execution of the current line.

Implemented in Word.

3.5.1.3 virtual iWord& iWord::operator++ (int) `[pure virtual]`

A standard post-increment operator.

Returns

A reference to itself.

The object increments its value AFTER the execution of the current line.

Implemented in Word.

3.5.1.4 virtual Word& iWord::operator= (const *Word*) [pure virtual]

A standard assignment operator.

Parameters

`in`	*The*	value to be copied.

Returns

A copy of the parameter.

The return value and parameter here must be declared as "Word"s as C++ does not work well with polymorphic assignment operators.

Implemented in Word.

3.5.1.5 virtual bool iWord::operator[] (int) const [pure virtual]

An accessor to the "i"th bit of the value.

Parameters

`in`	*The*	index of the bit in question.

Precondition

The index must be less than the size of a word, ie. 16.

Returns

True <=> 1, False <=> 0.

The number of the bits starts at zero and rises into the more significant bits. Examples: If the object "num" holds a value of 4 (0...100 in binary), num[0] = 0, num[1] = 0, num[2] = 1. If it holds a value of 1 (0...001 in binary) num[0] = 1, num[1] = 0, num[2] = 0, etc. If it holds a negative value (Starting with a 1 in 2's complement), num[15] = 1.

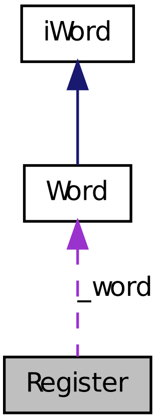
Implemented in Word.

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

- iWord.h

3.6 Register Class Reference

Collaboration diagram for Register:



Public Member Functions

- **Register** (const Word w)
- Word **getValue** () const
- void **Add** (const iWord &)
- Register **Add** (const iRegister &) const
- void **operator+** (const iWord &)
- Register **operator+** (const iRegister &) const
- void **Subtract** (const iWord &)
- Register **Subtract** (const iRegister &) const
- void **operator-** (const iWord &)
- Register **operator-** (const iRegister &) const
- void **And** (const iWord &)
- Register **And** (const iRegister &) const
- void **Or** (const iWord &)
- Register **Or** (const iRegister &) const
- void **Not** ()
- Register **Not** () const
- void **Store** (const iWord &)
- void **Store** (const iRegister &)
- Register & **operator=** (const iWord &)
- Register & **operator=** (const Register)

- Register & **operator++** ()
- Register & **operator++** (int)

Private Attributes

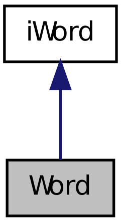
- Word **_word**

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

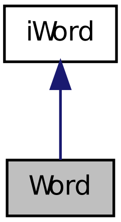
- Register.h
- Register.cpp

3.7 Word Class Reference

Inheritance diagram for Word:



Collaboration diagram for Word:



Public Member Functions

- int **toInt** () const
- int **toInt2Complement** () const
- std::string **toStr** () const
- std::string **toHex** () const
- bool **fromInt** (int)
- bool **fromStr** (const std::string &)
- bool **fromHex** (const std::string &)
- Word **Add** (const iWord &) const
- Word **operator+** (const iWord &) const
- Word **Subtract** (const iWord &) const
- Word **operator-** (const iWord &) const
- Word **And** (const iWord &) const
- Word **Or** (const iWord &) const
- Word **Not** () const
- void copy (const iWord &)

  Copies a word.
- Word & operator= (const Word)

  A standard assignment operator.
- iWord & operator++ ()
- iWord & operator++ (int)

  A standard post-increment operator.
- bool operator[] (const int) const

  An accessor to the "i"th bit of the value.
- void **print** () const

Private Member Functions

- bool **_hasBit** (int) const

Private Attributes

- unsigned short **_value**

3.7.1 Member Function Documentation

3.7.1.1 void Word::copy (const iWord &) [virtual]

Copies a word.

Parameters

out	*The*	value to be copied.

Postcondition

The caller equals that parameter.

Equivalent to the assignment "caller = parameter".

Implements iWord.

3.7.1.2 iWord & Word::operator++ (int) `[virtual]`

A standard post-increment operator.

Returns

A reference to itself.

The object increments its value AFTER the execution of the current line.

Implements iWord.

3.7.1.3 iWord & Word::operator++ ( ) `[virtual]`

A standard pre-increment operator.

Returns

A reference to itself.

The object increments its value BEFORE the execution of the current line.

Implements iWord.

3.7.1.4 Word & Word::operator= (const *Word*) `[virtual]`

A standard assignment operator.

Parameters

in	*The*	value to be copied.

Returns

A copy of the parameter.

The return value and parameter here must be declared as "Word"s as C++ does not work well with polymorphic assignment operators.

Implements iWord.

3.7.1.5 bool Word::operator[] (const) const [virtual]

An accessor to the "i"th bit of the value.

Parameters

in	*The*	index of the bit in question.

Precondition

The index must be less than the size of a word, ie. 16.

Returns

True <=> 1, False <=> 0.

The number of the bits starts at zero and rises into the more significant bits. Examples: If the object "num" holds a value of 4 (0...100 in binary), num[0] = 0, num[1] = 0, num[2] = 1. If it holds a value of 1 (0...001 in binary) num[0] = 1, num[1] = 0, num[2] = 0, etc. If it holds a negative value (Starting with a 1 in 2's complement), num[15] = 1.

Implements iWord.

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

- Word.h
- Word.cpp