OneUp Wi-11 Simulator

Generated by Doxygen 1.7.2

Fri Jan 21 2011 01:47:00

## **Contents**

1	Main	Page										1
2	Data 2.1	Structur Class H	e Index Hierarchy				 	 				3
3	<b>Data</b> 3.1	Structur Data S	re Index tructures				 	 				5 5
4	<b>File I</b> 4.1	ndex File Lis	t				 	 				7 7
5	Data 5.1 5.2 5.3 5.4 5.5 5.6 5.7 5.8 5.9 5.10	iDecod iInterpr iLoader iMemor Instruct iObjPar iRegist iSimula iWi11 C	re Documel er Class Reter Class Reter Class Reter Class Retion Struct reer Class Retion Class Reter Class Refer Class Refer Class Refer Class Refer Detailed   Member   5.10.2.1   5.10.2.2   5.10.2.3   5.10.2.4   5.10.2.5   5.10.2.6   5.10.2.7   5.10.2.1   5.10.2.11   5.10.2.12   5.10.2.13   5.10.2.14   5.10.2.15   5.10.2.16	eference Reference ference Reference Reference Reference Reference rence rence Descriptio Function E toInt toInt2Cont toStr toHex fromInt fromStr fromHex Add operator+ Subtract operator- And	e	ntatio						9 9 9 10 10 11 12 13 14 15 15 16 16 17 17 18 18 19 19 20

ii CONTENTS

			5.10.2.17	operator[]				 		 						. 20
	5.11	Object	Data Struc	t Reference	э			 		 						. 20
	5.12	Registe	er Class Re	eference .				 		 						. 21
	5.13	Result	Decoder C	ass Refere	ence			 		 						. 22
	5.14	Word C	Class Refer	ence				 		 						. 23
		5.14.1	Member I	unction D	ocume	enta	tion			 						. 25
			5.14.1.1	toInt				 		 						. 25
			5.14.1.2	toInt2Com	olemen	t.		 		 						. 25
			5.14.1.3													
			5.14.1.4	toHex				 		 						. 26
			5.14.1.5	fromInt .				 		 						. 26
			5.14.1.6	fromStr .				 		 						. 26
			5.14.1.7	fromHex .				 		 						. 27
			5.14.1.8	Add				 		 						. 27
			5.14.1.9	operator+				 		 						. 28
			5.14.1.10	Subtract .												
			5.14.1.11	operator-				 		 						. 28
			5.14.1.12	And												_
			5.14.1.13	copy												
			5.14.1.14	operator=												
			5.14.1.15	operator++												
			5.14.1.16	operator++												
			5.14.1.17	operator[]												
			5.14.1.17	operator[]			• •	 	•	 	•	•	•	 •	•	. 00
6	File [	Oocumer	ntation													31
	6.1	iWord.h	n File Refe	rence				 		 						. 31
		6.1.1		Description												

Chapter 1

Main Page

2 Main Page

# Chapter 2

# **Data Structure Index**

## 2.1 Class Hierarchy

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

iDecoder																		9
iInterpreter																		9
iLoader																		9
iMemory																		10
Instruction																		10
iObjParser																		
iRegister																		
iSimulator																		
iWi11																		
$iWord  . \ . \ . \ .$																		13
Word																		23
ObjectData .																		20
Register																		21
ResultDecoder	٠.																	22

Data Structure Index

# **Chapter 3**

# **Data Structure Index**

## 3.1 Data Structures

Here are the data structures with brief descriptions:

iDecoder	9
iInterpreter	9
iLoader	9
iMemory	0
Instruction	0
iObjParser	0
iRegister	0
iSimulator	1
iWi11 1	2
iWord (The iWord interface class defines the a "word" of data on the Wi-11	
Machine )	3
ObjectData	0
Register	1
ResultDecoder	2
Word	3

6 Data Structure Index

# Chapter 4

# File Index

## 4.1 File List

Here is a list of all documented files with brief descriptions:

iDecoder.h																																??
ilnterpreter.h																																??
iLoader.h .																																??
iMemory.h																																??
iObjParser.h																																
iRegister.h																																
iSimulator.h																																
iWi11.h																																
iWord.h (Th	ne	ir	nte	erf	fac	се	ir	nr	ole	m	eı	٦te	ed	b	٧	th	е	٣V	۷o	ro	"	cla	as	s	)							31
Register.h								•							-										•							
ResultCodes																																
Word h			Ī		Ī	Ī			-	-	Ī	Ī	Ī	Ī	Ī				-					-	-					-	-	22

8 File Index

## **Chapter 5**

## **Data Structure Documentation**

#### 5.1 iDecoder Class Reference

#### **Public Member Functions**

• virtual **DecodeInstruction** (const iWord &) const =0

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

· iDecoder.h

## 5.2 iInterpreter Class Reference

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

· iInterpreter.h

## 5.3 iLoader Class Reference

#### **Public Member Functions**

- virtual iLoader (iMemory \*)=0
- virtual Codes::RESULT Load (const char \*filename, iWord &PC\_address)=0

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

· iLoader.h

## 5.4 iMemory Class Reference

#### **Public Member Functions**

- virtual Codes::RESULT Reserve (const iWord &initial\_address, const iWord &length)
   const =0
- virtual Codes::RESULT Load (const iWord &) const =0
- virtual Codes::RESULT Store (const iWord &address, const iWord &value)=0

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

· iMemory.h

#### 5.5 Instruction Struct Reference

#### **Data Fields**

- INSTRUCTION TYPE type
- std::vector< iWord > data

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

· iDecoder.h

## 5.6 iObjParser Class Reference

#### **Public Member Functions**

- virtual Codes::Result Initialize (const char \*)=0
- virtual ObjectData GetNext ()=0

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

· iObjParser.h

## 5.7 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

#### 5.8 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

#### 5.9 iWi11 Class Reference

#### **Public Member Functions**

- virtual bool LoadObj (const char \*)=0
- virtual void DisplayMemory () const =0
- virtual void DisplayRegisters () const =0
- virtual bool ExecuteNext (bool verbose=false)=0

#### **Private Member Functions**

- virtual Codes::RESULT\_Add (const Decoder::REGISTER\_ID DR, const Decoder::REGISTER\_-ID SR1, const Decoder::REGISTER\_ID SR2)=0
- virtual Codes::RESULT\_Add (const Decoder::REGISTER\_ID DR, const Decoder::REGISTER\_-ID SR1, const iWord &immediate)=0
- virtual Codes::RESULT\_And (const Decoder::REGISTER\_ID DR, const Decoder::REGISTER\_-ID SR1, const Decoder::REGISTER\_ID SR2)=0
- virtual Codes::RESULT\_And (const Decoder::REGISTER\_ID DR, const Decoder::REGISTER\_-ID SR1, const iWord &immediate)=0
- virtual Codes::RESULT \_Branch (const iWord &address)=0
- virtual Codes::RESULT \_Debug ()=0
- virtual Codes::RESULT \_JSR (const iWord &)=0
- virtual Codes::RESULT \_JSRR (const iWord &baseR, const iWord &address)=0
- virtual Codes::RESULT \_Load (const Decoder::REGISTER\_ID DR, const iWord &address)=0
- virtual Codes::RESULT \_LoadI (const Decoder::REGISTER\_ID DR, const iWord &address)=0
- virtual Codes::RESULT\_LoadR (const Decoder::REGISTER\_ID DR, Decoder::REGISTER\_-ID baseR, const iWord &address)=0
- virtual Codes::RESULT\_Not (const Decoder::REGISTER\_ID DR, const Decoder::REGISTER\_-ID SR)=0
- virtual Codes::RESULT \_Ret ()=0
- virtual Codes::RESULT \_Store (const Decoder::REGISTER\_ID SR1, const iWord &address)=0

- virtual Codes::RESULT \_STI (const Decoder::REGISTER\_ID SR1, const iWord &address)=0
- virtual Codes::RESULT\_STR (const Decoder::REGISTER\_ID SR1, const Decoder::REGISTER\_-ID baseR, const iWord &address)=0
- virtual Codes::RESULT \_Trap (const iWord &code)=0

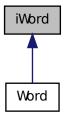
The documentation for this class was generated from the following file:

• iWi11.h

#### 5.10 iWord Class Reference

The iWord interface class defines the a "word" of data on the Wi-11 Machine.

Inheritance diagram for iWord:



#### **Public Member Functions**

- virtual int tolnt () const =0
  - "To non-negative Integer"
- virtual int toInt2Complement () const =0

"To Integer as 2's Complement"

- virtual std::string toStr () const =0
   "To String"
- virtual std::string toHex () const =0
   "To Hexadecimal"
- virtual bool fromInt (int)=0

"From Integer"

- virtual bool fromStr (const std::string &)=0
   "From String"
- virtual bool fromHex (const std::string &)=0
   "From Hexadecimal"
- virtual Word Add (const iWord &) const =0
   Adds two words.
- virtual Word operator+ (const iWord &) const =0
   A standard addition operator.
- virtual Word Subtract (const iWord &) const =0
   Subtracts two words.
- virtual Word operator- (const iWord &) const =0
   A standard subtraction operator.
- virtual Word And (const iWord &) const =0
   "And"s the bits of two words.
- 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.

#### 5.10.1 Detailed Description

The iWord interface class defines the a "word" of data on the Wi-11 Machine. The methods present in this inteface are meant to mimic the functionality of the Wi-11 machine, allowing for simplified execution of the instructions therein. As the size of a "word" depends on the architecture, classes implementing this interface should define the word length to be 16 bits in length.

5.10 iWord Class Reference 15

#### 5.10.2 Member Function Documentation

```
5.10.2.1 virtual int iWord::tolnt() const [pure virtual]
```

"To non-negative Integer"

#### Postcondition

The value of the word is not changed.

#### Returns

The bits of the word interpreted as a positive integer value.

Implemented in Word.

```
5.10.2.2 virtual int iWord::tolnt2Complement() const [pure virtual]
```

"To Integer as 2's Complement"

#### Postcondition

The value of the word is not changed.

#### Returns

The bits of the word interpreted as a signed (2's complement) integer value.

Implemented in Word.

```
5.10.2.3 virtual std::string iWord::toStr() const [pure virtual]
```

"To String"

#### Postcondition

The value of the word is not changed.

#### Returns

```
16 characters: each either a 1 or 0
```

### **Examples:**

```
If the object holds a (2's comp.) value 4: "0000000000000000000" If the object holds a (2's comp.) value -1: "11111111111111"
```

Implemented in Word.

#### 5.10.2.4 virtual std::string iWord::toHex() const [pure virtual]

"To Hexadecimal"

#### Postcondition

The value of the word is not changed.

#### Returns

```
"0x" + <4 characters in the range [0-9],[A-F]>
```

#### **Examples:**

```
If the object holds (2's comp.) value 8: "0x0008" If the object holds (2's comp.) value -2: "0xFFFE"
```

Implemented in Word.

#### 5.10.2.5 virtual bool iWord::fromInt(int) [pure virtual]

"From Integer"

#### **Parameters**

in	value	The value to be stored into the word.
----	-------	---------------------------------------

#### Postcondition

"value" is not changed.

#### Returns

True if and only if "value" can be represented in 16 bits

When this function returns "False", the value of the word is unchanged.

Otherwise, the word now holds the value "value".

Implemented in Word.

#### 5.10.2.6 virtual bool iWord::fromStr ( const std::string & ) [pure virtual]

"From String"

#### **Parameters**

in	str	A string of characters meant to represent a "word" to be stored.	1
	00	, A during or orial actors mount to represent a word to be clored.	П

#### Postcondition

"str" is not changed.

#### Returns

True if and only if "str" is well-formed (as defined in toStr()).

When this function returns "False", the value of the word is unchanged.

Otherwise, the word now holds the value "str".

Implemented in Word.

### **5.10.2.7 virtual bool iWord::fromHex ( const std::string & )** [pure virtual]

"From Hexadecimal"

#### **Parameters**

in	str A string of characters meant to represent a "word" to be stored.	str	

#### Postcondition

"str" is not changed.

#### Returns

True if and only if "str" is well-formed (as defined in toHex()).

When this function returns "False", the value of the word is unchanged.

Otherwise, the word now holds the value "str".

Implemented in Word.

#### 5.10.2.8 virtual Word iWord::Add ( const iWord & ) const [pure virtual]

Adds two words.

#### **Parameters**

in	W	A word value to be added.
----	---	---------------------------

#### Postcondition

Both "w" and the calling object do not change.

#### Returns

A new "Word" object containing result of adding "w" and the calling object.

#### Note

The addition is carried out with no regard to logical overflow.

Implemented in Word.

#### 5.10.2.9 virtual Word iWord::operator+ ( const iWord & ) const [pure virtual]

A standard addition operator.

#### Note

"result = p + w" is equivalent to "result = p.Add(w)".

Implemented in Word.

#### **5.10.2.10 virtual Word iWord::Subtract ( const iWord & ) const** [pure virtual]

Subtracts two words.

#### **Parameters**

#### Postcondition

Both "w" and the calling object do not change.

#### Returns

A new "Word" object containing the result of subtracting "w" from the calling object.

#### Note

The subtraction is carried out with no regard for logical overflow.

Implemented in Word.

#### 5.10.2.11 virtual Word iWord::operator-( const iWord & ) const [pure virtual]

A standard subtraction operator.

#### Note

"result = p - w" is equivalent to "result = p.Subtract(w)".

Implemented in Word.

#### 5.10.2.12 virtual Word iWord::And ( const iWord & ) const [pure virtual]

"And"s the bits of two words.

#### **Parameters**

in	w A word value to be "and"ed.

#### **Postcondition**

Both "w" and the calling object do not change.

#### Returns

A new "Word" object containing the result of performing a bit-wise and on "w" and the calling object.

Implemented in Word.

5.10.2.13 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.

**5.10.2.14 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.

5.10.2.15 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.

```
5.10.2.16 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.

```
5.10.2.17 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 \ll 1, False \ll 0.
```

The number of the bits starts at zero and rises into the more significant bits.

#### **Examples:**

```
If the object holds a value of 4 (0...100 in binary): num[2] = 1.

If it holds a value of 1 (0...001 in binary): num[0] = 1.

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

## 5.11 ObjectData Struct Reference

#### **Data Fields**

· char type

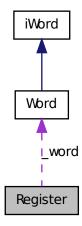
• std::vector< std::string > data

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

· iObjParser.h

## 5.12 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

#### 5.13 ResultDecoder Class Reference

#### **Public Member Functions**

- std::string Find (const Codes::RESULT &) const

#### **Static Private Attributes**

static std::map< Codes::Result, std::string > \_codes

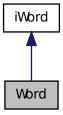
The documentation for this class was generated from the following file:

· ResultCodes.h

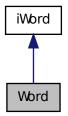
5.14 Word Class Reference 23

## 5.14 Word Class Reference

Inheritance diagram for Word:



Collaboration diagram for Word:



### **Public Member Functions**

- int toInt () const
  "To non-negative Integer"
- int toInt2Complement () const
  "To Integer as 2's Complement"
- std::string toStr () const "To String"

• std::string toHex () const
"To Hexadecimal"

• bool fromInt (int)

"From Integer"

bool fromStr (const std::string &)
 "From String"

• bool fromHex (const std::string &)

"From Hexadecimal"

 Word Add (const iWord &) const Adds two words.

Word operator+ (const iWord &) const
 A standard addition operator.

• Word Subtract (const iWord &) const Subtracts two words.

Word operator- (const iWord &) const
 A standard subtraction operator.

• Word And (const iWord &) const "And"s the bits of two words.

- 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.

#### **Private Member Functions**

• bool \_hasBit (int) const

5.14 Word Class Reference 25

#### **Private Attributes**

· unsigned short \_value

#### 5.14.1 Member Function Documentation

```
5.14.1.1 int Word::tolnt() const [virtual]
```

"To non-negative Integer"

#### **Postcondition**

The value of the word is not changed.

#### Returns

The bits of the word interpreted as a positive integer value.

Implements iWord.

```
5.14.1.2 int Word::tolnt2Complement ( ) const [virtual]
```

"To Integer as 2's Complement"

#### **Postcondition**

The value of the word is not changed.

#### Returns

The bits of the word interpreted as a signed (2's complement) integer value.

Implements iWord.

```
5.14.1.3 string Word::toStr() const [virtual]
```

"To String"

#### Postcondition

The value of the word is not changed.

#### Returns

```
16 characters: each either a 1 or 0
```

#### **Examples:**

```
If the object holds a (2's comp.) value 4: "000000000000000000" If the object holds a (2's comp.) value -1: "111111111111111"
```

Implements iWord.

#### 5.14.1.4 string Word::toHex()const [virtual]

"To Hexadecimal"

#### Postcondition

The value of the word is not changed.

#### Returns

```
"0x" + <4 characters in the range [0-9],[A-F]>
```

#### **Examples:**

```
If the object holds (2's comp.) value 8: "0x0008" If the object holds (2's comp.) value -2: "0xFFFE"
```

Implements iWord.

#### 5.14.1.5 bool Word::fromInt(int) [virtual]

"From Integer"

#### **Parameters**

in	value	The value to be stored into the word.
----	-------	---------------------------------------

#### Postcondition

"value" is not changed.

#### Returns

True if and only if "value" can be represented in 16 bits

When this function returns "False", the value of the word is unchanged.

Otherwise, the word now holds the value "value".

Implements iWord.

#### 5.14.1.6 bool Word::fromStr ( const std::string & ) [virtual]

"From String"

#### **Parameters**

in	str A strin	g of characters meant to represent a "word" to be stored.
		g

#### Postcondition

"str" is not changed.

#### Returns

True if and only if "str" is well-formed (as defined in toStr()).

When this function returns "False", the value of the word is unchanged.

Otherwise, the word now holds the value "str".

Implements iWord.

### **5.14.1.7** bool Word::fromHex ( const std::string & ) [virtual]

"From Hexadecimal"

#### **Parameters**

in	str A string of characters meant to represent a "word" to be stored.	to be stored.	

#### Postcondition

"str" is not changed.

#### Returns

True if and only if "str" is well-formed (as defined in toHex()).

When this function returns "False", the value of the word is unchanged.

Otherwise, the word now holds the value "str".

Implements iWord.

#### **5.14.1.8** Word Word::Add ( const iWord & ) const [virtual]

Adds two words.

#### **Parameters**

in	W	A word value to be added.
----	---	---------------------------

#### Postcondition

Both "w" and the calling object do not change.

#### Returns

A new "Word" object containing result of adding "w" and the calling object.

#### Note

The addition is carried out with no regard to logical overflow.

Implements iWord.

#### 5.14.1.9 Word Word::operator+(constiWord & )const [virtual]

A standard addition operator.

#### Note

"result = p + w" is equivalent to "result = p.Add(w)".

Implements iWord.

#### 5.14.1.10 Word Word::Subtract ( const iWord & ) const [virtual]

Subtracts two words.

#### **Parameters**

in	W	A word value to be subtracted.
----	---	--------------------------------

#### Postcondition

Both "w" and the calling object do not change.

#### Returns

A new "Word" object containing the result of subtracting "w" from the calling object.

#### Note

The subtraction is carried out with no regard for logical overflow.

Implements iWord.

#### 5.14.1.11 Word Word::operator-(constiWord & )const [virtual]

A standard subtraction operator.

#### Note

"result = p - w" is equivalent to "result = p.Subtract(w)".

Implements iWord.

#### 5.14.1.12 Word Word::And ( const iWord & ) const [virtual]

"And"s the bits of two words.

#### **Parameters**

in w A word value to be "and"ed.
----------------------------------

#### **Postcondition**

Both "w" and the calling object do not change.

#### Returns

A new "Word" object containing the result of performing a bit-wise and on "w" and the calling object.

Implements iWord.

```
5.14.1.13 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.

#### **5.14.1.14** 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.

```
5.14.1.15 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.

```
5.14.1.16 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.

```
5.14.1.17 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 holds a value of 4 (0...100 in binary): num[2] = 1.

If it holds a value of 1 (0...001 in binary): num[0] = 1.

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:

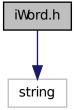
- Word.h
- Word.cpp

## **Chapter 6**

# **File Documentation**

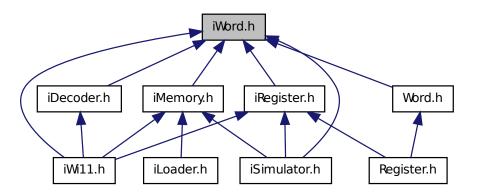
## 6.1 iWord.h File Reference

The interface implemented by the "Word" class. Include dependency graph for iWord.h:



32 File Documentation

This graph shows which files directly or indirectly include this file:



#### **Data Structures**

· class iWord

The iWord interface class defines the a "word" of data on the Wi-11 Machine.

### 6.1.1 Detailed Description

The interface implemented by the "Word" class.

#### Author

Joshua Green Andrew Groot

Defines the operations and signatures by which the "Word" class should operate. The signatures, while intended to be coded to the interface, are done as to this as C++ allows.

# Index

Add iWord, 17	Subtract, 18 toHex, 15
Word, 27	tolnt, 15
And	toInt2Complement, 15
iWord, 18	toStr, 15
Word, 28	iWord.h, 31
сору	ObjectData, 20
iWord, 19	operator+
Word, 29	iWord, 17
	Word, 27
fromHex	operator++
iWord, 17	iWord, 19, 20
Word, 27	Word, 29, 30
fromInt	operator-
iWord, 16	iWord, 18
Word, 26	Word, 28
fromStr	operator=
iWord, 16	iWord, 19
Word, 26	Word, 29
iDecoder 9	Register, 21
iDecoder, 9	Register, 21 ResultDecoder, 22
iInterpreter, 9	_
iInterpreter, 9 iLoader, 9	_
iInterpreter, 9 iLoader, 9 iMemory, 10	ResultDecoder, 22
iInterpreter, 9 iLoader, 9 iMemory, 10 Instruction, 10	ResultDecoder, 22 Subtract
iInterpreter, 9 iLoader, 9 iMemory, 10 Instruction, 10 iObjParser, 10	ResultDecoder, 22 Subtract iWord, 18 Word, 28
iInterpreter, 9 iLoader, 9 iMemory, 10 Instruction, 10 iObjParser, 10 iRegister, 10	ResultDecoder, 22  Subtract iWord, 18 Word, 28  toHex
iInterpreter, 9 iLoader, 9 iMemory, 10 Instruction, 10 iObjParser, 10 iRegister, 10 iSimulator, 11	ResultDecoder, 22  Subtract iWord, 18 Word, 28  toHex iWord, 15
iInterpreter, 9 iLoader, 9 iMemory, 10 Instruction, 10 iObjParser, 10 iRegister, 10 iSimulator, 11 iWi11, 12	ResultDecoder, 22  Subtract iWord, 18 Word, 28  toHex iWord, 15 Word, 25
iInterpreter, 9 iLoader, 9 iMemory, 10 Instruction, 10 iObjParser, 10 iRegister, 10 iSimulator, 11 iWi11, 12 iWord, 13	ResultDecoder, 22  Subtract iWord, 18 Word, 28  toHex iWord, 15 Word, 25  toInt
iInterpreter, 9 iLoader, 9 iMemory, 10 Instruction, 10 iObjParser, 10 iRegister, 10 iSimulator, 11 iWi11, 12 iWord, 13 Add, 17	ResultDecoder, 22  Subtract iWord, 18 Word, 28  toHex iWord, 15 Word, 25  toInt iWord, 15
iInterpreter, 9 iLoader, 9 iMemory, 10 Instruction, 10 iObjParser, 10 iRegister, 10 iSimulator, 11 iWi11, 12 iWord, 13 Add, 17 And, 18	ResultDecoder, 22  Subtract iWord, 18 Word, 28  toHex iWord, 15 Word, 25  toInt iWord, 15 Word, 25
iInterpreter, 9 iLoader, 9 iMemory, 10 Instruction, 10 iObjParser, 10 iRegister, 10 iSimulator, 11 iWi11, 12 iWord, 13 Add, 17 And, 18 copy, 19	ResultDecoder, 22  Subtract iWord, 18 Word, 28  toHex iWord, 15 Word, 25  toInt iWord, 15 Word, 25  toInt2Complement
iInterpreter, 9 iLoader, 9 iMemory, 10 Instruction, 10 iObjParser, 10 iRegister, 10 iSimulator, 11 iWi11, 12 iWord, 13 Add, 17 And, 18 copy, 19 fromHex, 17	ResultDecoder, 22  Subtract iWord, 18 Word, 28  toHex iWord, 15 Word, 25  toInt iWord, 15 Word, 25  toInt2Complement iWord, 15
iInterpreter, 9 iLoader, 9 iMemory, 10 Instruction, 10 iObjParser, 10 iRegister, 10 iSimulator, 11 iWi11, 12 iWord, 13	ResultDecoder, 22  Subtract iWord, 18 Word, 28  toHex iWord, 15 Word, 25  toInt iWord, 15 Word, 25  toInt2Complement iWord, 15 Word, 25
iInterpreter, 9 iLoader, 9 iMemory, 10 Instruction, 10 iObjParser, 10 iRegister, 10 iSimulator, 11 iWi11, 12 iWord, 13 Add, 17 And, 18 copy, 19 fromHex, 17 fromInt, 16 fromStr, 16	ResultDecoder, 22  Subtract iWord, 18 Word, 28  toHex iWord, 15 Word, 25  toInt iWord, 15 Word, 25  toInt2Complement iWord, 15
iInterpreter, 9 iLoader, 9 iMemory, 10 Instruction, 10 iObjParser, 10 iRegister, 10 iSimulator, 11 iWi11, 12 iWord, 13     Add, 17     And, 18     copy, 19     fromHex, 17     fromInt, 16     fromStr, 16     operator+, 17	ResultDecoder, 22  Subtract iWord, 18 Word, 28  toHex iWord, 15 Word, 25  toInt iWord, 15 Word, 25  toInt2Complement iWord, 15 Word, 25  toStr iWord, 15
iInterpreter, 9 iLoader, 9 iMemory, 10 Instruction, 10 iObjParser, 10 iRegister, 10 iSimulator, 11 iWi11, 12 iWord, 13     Add, 17     And, 18     copy, 19     fromHex, 17     fromInt, 16     fromStr, 16     operator++, 17     operator++, 19, 20	ResultDecoder, 22  Subtract iWord, 18 Word, 28  toHex iWord, 15 Word, 25  toInt iWord, 15 Word, 25  toInt2Complement iWord, 15 Word, 25  toInt2Complement iWord, 25  toStr
iInterpreter, 9 iLoader, 9 iMemory, 10 Instruction, 10 iObjParser, 10 iRegister, 10 iSimulator, 11 iWi11, 12 iWord, 13     Add, 17     And, 18     copy, 19     fromHex, 17     fromInt, 16     fromStr, 16     operator+, 17	ResultDecoder, 22  Subtract iWord, 18 Word, 28  toHex iWord, 15 Word, 25  toInt iWord, 15 Word, 25  toInt2Complement iWord, 15 Word, 25  toStr iWord, 15

34 INDEX

```
Add, 27
And, 28
copy, 29
fromHex, 27
fromInt, 26
fromStr, 26
operator+, 27
operator++, 29, 30
operator-, 28
operator-, 28
operator-, 29
Subtract, 28
toHex, 25
toInt, 25
toInt2Complement, 25
toStr, 25
```