My Project

Generated by Doxygen 1.8.3

Mon Mar 11 2013 10:51:46

Contents

1	Hiera	archical Index	1
	1.1	Class Hierarchy	1
2	Clas	s Index	3
	2.1	Class List	3
3	File	Index	5
	3.1	File List	5
4	Clas	s Documentation	7
	4.1	Arc_t Struct Reference	7
	4.2	asfig Struct Reference	7
	4.3	asisc Struct Reference	7
	4.4	asiss Struct Reference	8
	4.5	asobj Struct Reference	8
	4.6	asosc Struct Reference	8
	4.7	Basic_block Class Reference	9
		4.7.1 Detailed Description	10
	4.8	Cfg Class Reference	10
		4.8.1 Detailed Description	11
	4.9	dep Struct Reference	11
	4.10		11
		4.10.1 Detailed Description	12
	4.11	Directive Class Reference	12
		4.11.1 Detailed Description	13
	4.12	Function Class Reference	13
		4.12.1 Detailed Description	14
	4.13	Instruction Class Reference	14
		4.13.1 Detailed Description	17
		4.13.2 Member Function Documentation	17
		4.13.2.1 get_nbOp	17
		4.13.2.2 is den MEM	17

ii CONTENTS

		4.13.2.3 is_dep_RAW	17
		4.13.2.4 is_dep_RAW1	17
		4.13.2.5 is_dep_RAW2	17
		4.13.2.6 is_dep_WAR	18
		4.13.2.7 is_dep_WAR1	18
		4.13.2.8 is_dep_WAR2	18
		4.13.2.9 is_dep_WAW	18
		4.13.2.10 is_dependant	18
4.14	Label C	Class Reference	19
	4.14.1	Detailed Description	19
4.15	Line CI	ass Reference	19
	4.15.1	Detailed Description	20
	4.15.2	Member Function Documentation	20
		4.15.2.1 to_string	20
4.16	Node C	Class Reference	20
	4.16.1	Detailed Description	21
4.17	Node_d	dfg Class Reference	21
	4.17.1	Detailed Description	22
4.18	Operar	nd Class Reference	22
	4.18.1	Detailed Description	23
	4.18.2	Member Function Documentation	23
		4.18.2.1 get_op_type	23
		4.18.2.2 to_string	23
4.19	OPExp	ression Class Reference	23
	4.19.1	Detailed Description	24
	4.19.2	Member Function Documentation	24
		4.19.2.1 get_op	24
		4.19.2.2 get_op_type	24
		4.19.2.3 to_string	24
4.20		nediate Class Reference	25
	4.20.1	Detailed Description	25
	4.20.2	Member Function Documentation	25
		4.20.2.1 get_op	25
		4.20.2.2 get_op_type	26
		4.20.2.3 to_string	26
4.21		el Class Reference	26
		Detailed Description	27
	4.21.2	Member Function Documentation	27
		4.21.2.1 get_op_type	27
		4.21.2.2 to_string	27

CONTENTS

4.22	OPRegister Class Reference	27
	4.22.1 Detailed Description	28
	4.22.2 Member Function Documentation	28
	4.22.2.1 get_op	28
	4.22.2.2 get_op_type	28
	4.22.2.3 get_reg	28
	4.22.2.4 to_string	29
4.23	Program Class Reference	29
	4.23.1 Detailed Description	30
	4.23.2 Member Function Documentation	30
	4.23.2.1 in_file	30
4.24	s_Profile Struct Reference	30
	4.24.1 Detailed Description	30
4.25	TestOPLabel Class Reference	31
4.26	utchn Struct Reference	31
4.27	utdat Union Reference	31
4.28	utdic Struct Reference	31
4.29	utdit Struct Reference	32
4.30	uttdc Struct Reference	32
4.31	uttpd Struct Reference	32
4.32	uttyp Struct Reference	33
4.33	YYSTYPE Union Reference	33
Eile	Documentation	35
5.1	_	35
E 0	•	35
5.2	•	35
E 0	·	36 36
5.3	•	
E A	•	36 36
5.4		36 37
5.5	·	37
5.5		37 37
E C	and the second pro-	37 37
5.6		
F 7	·	38
5.7		38
F 0	•	38
5.8		38
	5.8.1 Detailed Description	39

5

iv CONTENTS

5.9	Node.h File Reference	39
	5.9.1 Detailed Description	39
5.10	Node_dfg.h File Reference	39
	5.10.1 Detailed Description	40
5.11	Operand.h File Reference	40
	5.11.1 Detailed Description	40
5.12	OPExpression.h File Reference	40
	5.12.1 Detailed Description	41
5.13	OPImmediate.h File Reference	41
	5.13.1 Detailed Description	41
5.14	OPLabel.h File Reference	41
	5.14.1 Detailed Description	41
5.15	OPRegister.h File Reference	42
	5.15.1 Detailed Description	42
5.16	Program.h File Reference	42
	5.16.1 Detailed Description	42
Index		42

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Arc_t	. 7
asfig	. 7
asisc	. 7
asiss	. 8
asobj	
asosc	
Basic_block	
Cfg	
dep	
Dfg	
Function	_
Line	
Directive	
Instruction	
Label	19
Node	. 20
Node_dfg	. 21
Operand	. 22
OPExpression	23
OPImmediate	25
OPLabel	26
OPRegister	27
Program	. 29
s_Profile	. 30
TestFixture	
TestOPLabel	31
utchn	. 31
utdat	. 31
utdic	. 31
utdit	. 32
uttdc	. 32
uttpd	. 32
uttyp	. 33
YYSTYPE	33

2 **Hierarchical Index**

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief description	Here are the classes.	structs.	unions a	and interfaces	with brid	ef descri	ptions:
---	-----------------------	----------	----------	----------------	-----------	-----------	---------

Arc_t .		7
asfig		7
asisc .		7
asiss .		8
asobj .		8
asosc .		8
Basic_bl		
	Class representing a Basic_block of a fonction	9
Cfg		
	Class representing control flow graph	10
dep Dfg		11
	Class representing a Dfg of a Basic block, a data flow graph that is to be used to calculate the critical path and schedule code	11
Directive	•	
	Class representing an Directive herited by Line	12
Function	· · · · · · · · · · · · · · · · · · ·	
	Class representing a Function on a program	13
Instruction	on .	
	Class representing an instruction which herited by Line	14
Label		
	Class representing an Label herited by Line	19
Line		
	Abstract class representing an Line	19
Node		
	Class representing a Node in list	20
Node_df		
	Class representing a node of data flow graph	21
Operand		
	Abstract class representing an operand	22
OPExpre		
	Class representing an expression herited by Operand	23
OPImme		
001 1	Class representing an Immediate herited by Operand	25
OPLabel		00
ODD- :	Class representing a Label herited by Operand	26
OPRegis		07
	Class representing a Register herited by Operand	27

4 Class Index

Program
Class representing a program as list of lines
s_Profile
Structure allowing to add caracteristics to an operator
TestOPLabel 3
utchn
utdat
utdic
utdit
uttdc
uttpd
uttyp
YYSTYPE

Chapter 3

File Index

3.1 File List

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

asm200.h	
asm_mipsyac.h	??
Basic_block.h	
Basic_block class	35
Cfg.h	
Cfg class	35
Dfg.h	
Dfg class	36
Directive.h	
Directive class	
Enum_type.h	??
Function.h	0.
Function class	37
Instruction.h Instruction class	37
Label.h	31
Label class	38
Line.h	30
Line class	38
Node.h	
Node class	39
Node dfg.h	
Node dfg class	39
Operand.h	
Operand class	40
OPExpression.h	
OPExpression class	40
OPImmediate.h	
OPImmediate class	41
OPLabel.h	
OPLabel class	41
OPRegister.h	
OPRegister class	42
Program.h	
Program class	
TestOPLabel.h	
utl200.h	??

6 File Index

Chapter 4

Class Documentation

4.1 Arc_t Struct Reference

Public Attributes

- int delai
- t_Dep dep
- Node_dfg * next

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

• Node_dfg.h

4.2 asfig Struct Reference

Public Attributes

- struct utdic * GLB_DIC
- struct uttyp * GLB_SYM
- struct uttyp * MEM_TAB
- struct asosc * OUT_SEC
- struct asisc * IN_SEC
- struct asobj * OBJECTS
- unsigned int FLAG

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

· asm200.h

4.3 asisc Struct Reference

Public Attributes

- struct asisc * NEXT
- char * IDENT
- struct asosc * OUT_SEC
- unsigned int POSITION

· unsigned int FLAG

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

asm200.h

4.4 asiss Struct Reference

Public Attributes

- struct asiss * NEXT
- · unsigned int ADDR
- unsigned int SIZE
- unsigned int FLAG

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

· asm200.h

4.5 asobj Struct Reference

Public Attributes

- struct asobj * NEXT
- char * IDENT
- struct utdic * SYM_DIC
- struct uttyp * SEC_SYM
- unsigned int FLAG

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

asm200.h

4.6 asosc Struct Reference

Public Attributes

- struct asosc * NEXT
- char * IDENT
- unsigned int INS_NBR
- struct asiss ** CUR_ISS
- struct asiss ** SUB SEC
- unsigned int ADDR
- unsigned int SIZE
- unsigned int FLAG

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

· asm200.h

4.7 Basic block Class Reference

```
class representing a Basic block of a fonction
#include <Basic_block.h>
Public Member Functions
    · Basic block ()
          Constructor of a Basic Block.

    ∼Basic block ()

          Destructor of a basic block.
    void set_head (Node *)
          setter of the head of the basic block
    void set_end (Node *)
          setter of the end of the basic block
    Node * get_head ()
          get the head of the basic block
    Node * get_end ()
          get the end of the basic block
    void set_branch (Node *)
          setter of Node corresponding to the branch
    Node * get_branch ()
          get the Node corresponding to the branch

    bool is_labeled ()

          Return true if the first line of the block is a label.
    • void set index (int i)
          set the index of the basic block
    int get_index ()
          get the index of the basic block
    • int size ()
          returns the size (in nodes) of the basic block
    int get_nb_succ ()
          returns/gets the number of successors of the basic block
    • int get nb pred ()
          returns/gets the number of predecessors of the basic block

    void set_successor1 (Basic_block *BB)

          setter of the successor of the basic block

    Basic_block * get_successor1 ()

          get the successor of the basic block

    void set_successor2 (Basic_block *BB)

          setter of the successor of the basic block

    Basic_block * get_successor2 ()

          get the successor of the basic block

    void set predecessor (Basic block *BB)

          setter of the predecessor of the basic block

    Basic_block * get_predecessor (int)

          get the ith predecessor of the basic block
    int get_nb_inst ()
          returns the number of instructions
```

Node * get_first_node_instruction ()

```
    Instruction * get_first_instruction ()

      return the first instruction of the basic block, NULL if any

    Instruction * get_last_instruction ()

    Instruction * get_instruction_at_index (int)

      returns the instruction at the given index, NULL if any

    void link_instructions ()

      link instructions in the order they appear in the code

    void comput pred succ dep ()

      computes dependances predecessors and successors of each instructions in the BB
· void display ()
      to display the basic block
· void restitution (string const)
      restitutes the basic block in a file

    void set_link_succ_pred (Basic_block *)

      sets the parameter as successor and this as predecessor of the parameter

    bool is_delayed_slot (Instruction *)

      tests if the instruction is in the delayed slots of the branch terminating the BB if any
int nb_cycles ()
      gives the number of cycles to execute all instruction in this

    void test ()

      this method is to be used to test other methods
```

Static Public Member Functions

• static void **show_dependances** (Instruction *, Instruction *)

4.7.1 Detailed Description

class representing a Basic_block of a fonction

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

• Basic_block.h

4.8 Cfg Class Reference

```
class representing control flow graph
```

```
#include <Cfg.h>
```

Public Member Functions

```
    Cfg (Basic_block *, int)
    Constructor of Cfg.
```

• ∼Cfg ()

Destructor of Cfg.

• Basic block * get head ()

get the head of the cfg

void display (Basic_block *)

Display cfg, when you call this method you have to affect the fisrt parameter to NULL.

void restitution (Basic_block *, string const)

Restitut the cfg in file with DOT, when you call this method you have to affect the first parameter to NULL.

4.8.1 Detailed Description

class representing control flow graph

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

• Cfg.h

4.9 dep Struct Reference

Public Attributes

- Instruction * inst
- t_Dep type

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

· Instruction.h

4.10 Dfg Class Reference

class representing a Dfg of a Basic block, a data flow graph that is to be used to calculate the critical path and schedule code

```
#include <Dfg.h>
```

Public Member Functions

Dfg (Basic_block *)

Constructor of Dfg given a basic block.

• ∼Dfg ()

Destructor of Dfg.

void build_dfg (Node_dfg *, bool)

Build the Dfg, when you call this method you have to affect the fisrt parameter to NULL and the second to true.

void display (Node_dfg *, bool)

Display the Dfg, when you call this method you have to affect the first parameter to NULL and the second to true.

void restitute (Node_dfg *, string const, bool)

restitute the Dfg, when you call this method you have to affect the fisrt parameter to NULL and the third to true

• bool read_test ()

tests if all node have been read

void comput_critical_path ()

comput the node weight needed for critical path computation of the Dfg

- void compute_nb_descendant ()
- void scheduling (bool)

order the instructions in the basic block according to an algorithm list

- void apply_scheduling ()
- int get_critical_path ()

returns the highest weigth of nodes

void display_sheduled_instr ()

4.10.1 Detailed Description

class representing a Dfg of a Basic block, a data flow graph that is to be used to calculate the critical path and schedule code

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

• Dfg.h

4.11 Directive Class Reference

class representing an Directive herited by Line

```
#include <Directive.h>
```

Inheritance diagram for Directive:



Public Member Functions

· Directive (string)

Constructor of the Directive.

• Directive (string, string)

Constructor of the Directive with directive, content and an boolean.

Directive (string, string, bool)

Constructor of the Directive with directive, content and an boolean.

virtual ∼Directive ()

Destructor of the Directive.

• virtual t_Line type_line ()

get the type of the line

virtual string to_string ()

get the string of the Directive

• virtual string get_content ()

get the string of the Directive

• virtual void set_content (string)

set the string of the Directive

• bool is_function ()

return true if the directive indicate a function

virtual t_Inst get_type ()

return the type of the instruction

Public Attributes

- string _dir
- · string _value
- bool _isfunction

Additional Inherited Members

4.11.1 Detailed Description

class representing an Directive herited by Line

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

· Directive.h

4.12 Function Class Reference

```
class representing a Function on a program
```

```
#include <Function.h>
```

Public Member Functions

```
• Function ()
```

Constructor of a function.

• ∼Function ()

Destructor of a function.

void set_head (Node *)

setter of the head of the function

void set_end (Node *)

setter of the end of the function

Node * get_head ()

get the head of the function

- Basic_block * get_firstBB ()
- Node * get_end ()

get the end of the function

• void display ()

display the function

• int size ()

get the size of the function

void restitution (string const)

restitute the function in a file

- void add_BB (Node *, Node *, int)
- void comput_basic_block ()

Calculate the basics bolck of the function.

• int nbr_BB ()

get the number of Basic block in the function

Basic_block * get_BB (int)

get the Basic Block in the list

- list< Basic_block * >::iterator bb_list_begin ()
- list< Basic_block * >::iterator bb_list_end ()
- void comput_label ()

comput labels of the function in list

Label * get_label (int)

get all labels of the function

• int nbr_label ()

get the size of the list label

• Basic_block * find_label_BB (OPLabel *)

Get the basic block corresponding to the label.

void comput_succ_pred_BB ()

Associate for each Basic block its successors.

• void test ()

method to test other methods

4.12.1 Detailed Description

class representing a Function on a program

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

Function.h

4.13 Instruction Class Reference

class representing an instruction which herited by Line

```
#include <Instruction.h>
```

Inheritance diagram for Instruction:



Public Member Functions

• Instruction (string, t_Operator, t_Inst, Operand *, Operand *, Operand *)

Constructor of the class instruction.

Instruction (t_Operator, Operand *, Operand *, Operand *)

Constructor with 3 Operands of the class instruction.

Instruction (t_Operator, Operand *, Operand *)

Constructor with 2 Operands of the class instruction.

Instruction (t_Operator, Operand *)

Constructor with 1 Operand of the class instruction.

Instruction (t_Operator)

Constructor without Operands of the class instruction.

virtual ∼Instruction ()

Destructor of the class instruction.

Operand * get_op1 ()

Get the first operand value accessor of the operand.

void set_op1 (Operand *o)

set the first operand value setter of the operand

Operand * get_op2 ()

Get the second operand value accessor of the operand.

void set op2 (Operand *o)

set the second operand value setter of the operand

• Operand * get_op3 ()

```
Get the third operand value accessor of the operand.
void set_op3 (Operand *o)
      set the third operand value setter of the operand
• t_Operator get_opcode ()
      get the Opcode value accessor of the opcode
• string string_opcode ()
      get the string Opcode value accessor of the string opcode

    void set opcode (t Operator newop)

      set the opcode value setter of the opcode

    t_Format get_format ()

      get the format of the Instruction accessor of the format
virtual t_Inst get_type ()
      get the Type of the Instruction accessor of the Type

    virtual t_Line type_line ()

      get the type of the line

    virtual string to_string ()

      get the name string instruction

    virtual string get content ()

      get the string of the instruction

    virtual void set_content (string)

      set the string of the instruction
• string string_form ()
      set the string format
string_type ()
      set the string Type of instruction
• bool reads_in (int dst)
• bool writes in (int dst)

    t_Dep is_dependant (Instruction *i2)

      get the dependance between the current instruction and i2

    bool is dep RAW1 (Instruction *i2)

      get the information if there is dependance RAW between the current instruction and i2

    bool is_dep_RAW2 (Instruction *i2)

      get the information if there is dependance RAW between the current instruction and the first source operand of i2

    bool is_dep_RAW (Instruction *i2)

      get the information if there is dependance RAW between the current instruction and i2

    bool is_dep_WAR1 (Instruction *i2)

      test if there is dependance WAR between the first source operande of the current instruction if any and the destination
      register operande i2 if any

    bool is dep WAR2 (Instruction *i2)

      test if there is dependance WAR between the second source operande of the current instruction if any and the
      destination register operande i2 if any
• bool is_dep_WAR (Instruction *i2)
      get the information if there is dependance WAR between the current instruction and i2

    bool is_dep_WAW (Instruction *i2)

      get the information if there is dependance WAW between the current instruction and i2

    bool is dep MEM (Instruction *i2)

      test if there is dependance MEMDEP between the current instruction and i2
int get_nbOp ()
      get the number of operand

    void set number oper (int)

      set the number of operand
```

```
    OPRegister * get_reg_dst ()

      get the regiter destination of the instruction

    OPRegister * get reg src1 ()

      get the first register source of the instruction

    OPRegister * get_reg_src2 ()

      get the second register source of the instruction

    void set next (Instruction *)

      get the successor of the Instruction

    void set_link_succ_pred (Instruction *)

      set the parameter as successor and this as predecessor of the parameter

    Instruction * get_next ()

      get the successor of the Instruction

    void set prev (Instruction *)

      setter of the predecessor of the Instruction
Instruction * get_prev ()
      get the predecessor of the Instruction
void add_pred_dep (dep *)
      add a type of a dep with a predecessor instruction to the dependance type list
dep * get_pred_dep (int i)
      get the dependance type with the ith predecessor instruction of the current instruction
void add_succ_dep (dep *)
      add a type of a dep with a successor instruction to list of the dependance type of successors
• list< dep * >::iterator succ_begin ()

    list < dep * >::iterator succ_end ()

dep * get_succ_dep (int i)
      get the dependance type with ith successor instruction of the current instruction
• int get_nb_succ ()
      get the number of successor of the Instruction
int get_nb_pred ()
      get the number of predecessor of the Instruction
• int get index ()
      get the index of instruction

    void set index (int)

      set the index of instruction
• bool is branch ()
      test if the instruction is a branch
• bool is_call ()
      test if the instruction is a call
• bool is cond branch ()
      test if the instruction is a conditionnal branch

    bool is_indirect_branch ()

      test if the instruction a branch and the target adress is in a register

    bool is_mem ()

    bool is_mem_load ()

      test if the instruction is a memory access that reads a value
• bool is mem store ()
      test if the instruction is a memory access that writes a value
• int get_latency ()
      test if the instruction is a memory access that writes a value
```

void print_succ_dep ()

Static Public Member Functions

• static bool is_writed_between (int dst, Instruction *i1, Instruction *i2exclu)

Additional Inherited Members

4.13.1 Detailed Description

class representing an instruction which herited by Line

4.13.2 Member Function Documentation

4.13.2.1 int Instruction::get_nbOp()

get the number of operand

Returns

return the number of operand

4.13.2.2 bool Instruction::is_dep_MEM (Instruction *i2)

test if there is dependance MEMDEP between the current instruction and i2

Returns

return true if there is a MEMDEP dependance

4.13.2.3 bool Instruction::is_dep_RAW (Instruction * i2)

get the information if there is dependance RAW between the current instruction and i2

Returns

return true if there is a RAW dependance

4.13.2.4 bool Instruction::is_dep_RAW1 (Instruction * i2)

get the information if there is dependance RAW between the current instruction and i2

Returns

return true if there is a RAW dependance between the current instruction and i2

4.13.2.5 bool Instruction::is_dep_RAW2 (Instruction * i2)

get the information if there is dependance RAW between the current instruction and the first source operand of i2

Returns

return true if there is a RAW dependance between the current instruction and the first source register operand of i2

```
4.13.2.6 bool Instruction::is_dep_WAR ( Instruction * i2 )
```

get the information if there is dependance WAR between the current instruction and i2

Returns

return true if there is a WAR dependance

```
4.13.2.7 bool Instruction::is_dep_WAR1 ( Instruction * i2 )
```

test if there is dependance WAR between the first source operande of the current instruction if any and the destination register operande i2 if any

Returns

return true if there is a WAR dependance between the first source operande of the current instruction if any and the destination register operande i2 if any

```
4.13.2.8 bool Instruction::is_dep_WAR2 ( Instruction * i2 )
```

test if there is dependance WAR between the second source operande of the current instruction if any and the destination register operande i2 if any

Returns

return true if there is a WAR dependance between the second source operande of the current instruction if any and the destination register operande i2 if any

```
4.13.2.9 bool Instruction::is_dep_WAW ( Instruction * i2 )
```

get the information if there is dependance WAW between the current instruction and i2

Returns

return true if there is a WAW dependance

```
4.13.2.10 t_Dep Instruction::is_dependant ( Instruction * i2 )
```

get the dependance between the current instruction and i2

Returns

```
return "RAW", "WAR", "WAW", "MEMDEP" or "not dependant" in format enum
```

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

· Instruction.h

4.14 Label Class Reference 19

4.14 Label Class Reference

class representing an Label herited by Line

#include <Label.h>

Inheritance diagram for Label:



Public Member Functions

• Label (string)

Constructor of the Label.

virtual ~Label ()

Destructor of the Label.

• virtual t_Line type_line ()

get the type of the line

• virtual string to_string ()

get the string of Label

• virtual string get_content ()

get the string of the Label

· virtual void set_content (string)

set the string of the Label

• virtual t_Inst get_type ()

return the type of the instruction

Additional Inherited Members

4.14.1 Detailed Description

class representing an Label herited by Line

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

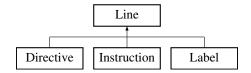
· Label.h

4.15 Line Class Reference

Abstract class representing an Line.

#include <Line.h>

Inheritance diagram for Line:



Public Member Functions

virtual ~Line ()

Virtual destructor.

• virtual string get_content ()=0

get the string of the line virtual getter

• virtual void set_content (string)=0

set the string of the line virtual setter

• virtual t_Line type_line ()=0

get the type of the line virtual accessor of the type

• virtual string to_string ()=0

get the name string accessor of the type line

• virtual t_Inst get_type ()=0

return the type of the instruction

• bool islnst ()

tests if the line is an instruction

• bool isLabel ()

tests if the line is a label

· bool isDirective ()

tests if the line is a directive

Protected Attributes

• string _line

4.15.1 Detailed Description

Abstract class representing an Line.

4.15.2 Member Function Documentation

```
4.15.2.1 virtual string Line::to_string() [pure virtual]
```

get the name string accessor of the type line

Implemented in Instruction, Directive, and Label.

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

· Line.h

4.16 Node Class Reference

class representing a Node in list

#include <Node.h>

Public Member Functions

```
Node (Line *content)
    Node constructor.
~Node ()
    Node destructor.
Node * get_next ()
    get the next node
void set_next (Node *)
    set the next node
Node * get_prev ()
    get the previous node
void set_prev (Node *)
    set the previous node
Line * get_line ()
    get the current line
```

void set_line (Line *newline)
 set the current line
 string get_lineContent ()

get the content of the line

4.16.1 Detailed Description

class representing a Node in list

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

Node.h

4.17 Node_dfg Class Reference

```
class representing a node of data flow graph
```

```
#include <Node_dfg.h>
```

Public Member Functions

```
    void add_predecesseur (Node_dfg *)

• int nb_preds ()

    list < Node_dfg * >::iterator pred_begin ()

list< Node_dfg * >::iterator pred_end ()
void set_instruction (Instruction *)
     set the Instruction
• int compute_weight ()

    void set_weight (int)

     set the weight
· int get_weight ()
     get the weight
• int compute_nb_descendant (int nb_instr, int *deja_comptes)

    void set_nb_descendant (int)

     set the number of descendant
• int get_nb_descendant ()
     get the number of descendant
void set_tready (int t)
• int get_tready ()
```

4.17.1 Detailed Description

class representing a node of data flow graph

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

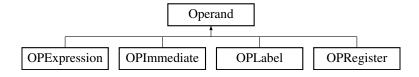
• Node_dfg.h

4.18 Operand Class Reference

Abstract class representing an operand.

```
#include <Operand.h>
```

Inheritance diagram for Operand:



Public Member Functions

```
• virtual \simOperand ()
```

Virtual destructor.

virtual string get_op ()=0

Get the operand value virtual accessor of the operand.

• virtual void set_op (string)=0

set the operand value virtual setter of the operand

• virtual t_OpType get_op_type ()=0

get the operator type virtual accessor of accessor

• virtual string to_string ()=0

virtual tostring

Protected Attributes

· string _oper

4.18.1 Detailed Description

Abstract class representing an operand.

4.18.2 Member Function Documentation

```
4.18.2.1 virtual t_OpType Operand::get_op_type( ) [pure virtual]
```

get the operator type virtual accessor of accessor

Returns

return the Operand type as enum

Implemented in OPRegister, OPImmediate, OPExpression, and OPLabel.

```
4.18.2.2 virtual string Operand::to_string() [pure virtual]
```

virtual tostring

Returns

return the Object as string

Implemented in OPRegister, OPImmediate, OPExpression, and OPLabel.

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

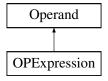
· Operand.h

4.19 OPExpression Class Reference

class representing an expression herited by Operand

```
#include <OPExpression.h>
```

Inheritance diagram for OPExpression:



Public Member Functions

OPExpression (string)

Constructor of the Expression class.

virtual ∼OPExpression ()

Destructor of the Expression class.

```
virtual string get_op ()
          Get the operand value.
    virtual t_OpType get_op_type ()
         get the operator type
    • virtual string to_string ()
         tostring
    • virtual void set_op (string)
         set the operand value setter of the operand
Additional Inherited Members
4.19.1 Detailed Description
class representing an expression herited by Operand
4.19.2 Member Function Documentation
4.19.2.1 virtual string OPExpression::get_op( ) [virtual]
Get the operand value.
Returns
    return the string of the Expression
Implements Operand.
4.19.2.2 virtual t_OpType OPExpression::get_op_type() [virtual]
get the operator type
Returns
    return the Operand type as enum
Implements Operand.
4.19.2.3 virtual string OPExpression::to_string() [virtual]
tostring
Returns
    return the Object as string
Implements Operand.
The documentation for this class was generated from the following file:
```

· OPExpression.h

4.20 OPImmediate Class Reference

class representing an Immediate herited by Operand

#include <OPImmediate.h>

Inheritance diagram for OPImmediate:



Public Member Functions

• OPImmediate (string)

Constructor of the Immediate Class.

OPImmediate (int)

Constructor of the Immediate Class.

• virtual \sim OPImmediate ()

Destructor of the Immediate Class.

virtual string get_op ()

Get the string of the operand.

virtual t_OpType get_op_type ()

get the operator type

• virtual string to_string ()

tostring

virtual void set_op (string)

set the string of the operand setter of the operand

Additional Inherited Members

4.20.1 Detailed Description

class representing an Immediate herited by Operand

4.20.2 Member Function Documentation

4.20.2.1 virtual string OPImmediate::get_op() [virtual]

Get the string of the operand.

Returns

return the string of the Immediate

Implements Operand.

```
4.20.2.2 virtual t_OpType OPImmediate::get_op_type( ) [virtual]
get the operator type

Returns
    return the Operand type as enum

Implements Operand.

4.20.2.3 virtual string OPImmediate::to_string( ) [virtual]

tostring

Returns
    return the name of the Object as string
```

Implements Operand.

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

• OPImmediate.h

4.21 OPLabel Class Reference

class representing a Label herited by Operand

```
#include <OPLabel.h>
```

Inheritance diagram for OPLabel:



Public Member Functions

• OPLabel (string)

Constructor of the Label Class.

virtual ∼OPLabel ()

Destructor of the Label Class.

virtual string get_op ()

Get the string of the operand accessor of the operand.

• virtual t_OpType get_op_type ()

get the operator type

virtual string to_string ()

tostring

virtual void set_op (string)

set the operand value setter of the operand

Additional Inherited Members

4.21.1 Detailed Description

class representing a Label herited by Operand

4.21.2 Member Function Documentation

```
4.21.2.1 virtual t_OpType OPLabel::get_op_type( ) [virtual]
```

get the operator type

Returns

return the Operand type as enum

Implements Operand.

```
4.21.2.2 virtual string OPLabel::to_string() [virtual]
```

tostring

Returns

return the name of the Object as string

Implements Operand.

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

OPLabel.h

4.22 OPRegister Class Reference

class representing a Register herited by Operand

```
#include <OPRegister.h>
```

Inheritance diagram for OPRegister:



Public Member Functions

- OPRegister (string, t_Src_Dst)
 - Constructor of the Register class.
- OPRegister (string, int, t_Src_Dst)

Constructor of the Register class.

- OPRegister (int, t_Src_Dst)
- virtual \sim OPRegister ()

```
int get_reg ()
          Get the Register value.
    void set_reg (int)
          set the Register value setter of the Register
    virtual string get_op ()
          Get the operand value.
    virtual t_OpType get_op_type ()
          get the operator type

    virtual string to_string ()

          tostring

    virtual void set_op (string)

          set the operand value setter of the operand
    void set_type (t_Src_Dst)
          set the type of the register setter of the register type
    • t_Src_Dst get_type ()
          get the type of the register getter of the register type
Additional Inherited Members
4.22.1 Detailed Description
class representing a Register herited by Operand
4.22.2 Member Function Documentation
4.22.2.1 virtual string OPRegister::get_op( ) [virtual]
Get the operand value.
Returns
    return the string of the register
Implements Operand.
4.22.2.2 virtual t_OpType OPRegister::get_op_type( ) [virtual]
get the operator type
Returns
    return the Operand type as enum
Implements Operand.
4.22.2.3 int OPRegister::get_reg ( )
Get the Register value.
Returns
    return the number of the Register
```

Destructor of the Register class.

```
virtual string OPRegister::to_string( ) [virtual]
tostring
Returns
    return the Object as string
Implements Operand.
The documentation for this class was generated from the following file:
    · OPRegister.h
4.23
         Program Class Reference
class representing a program as list of lines
#include <Program.h>
Public Member Functions
    • Program ()
          Empty constructor of a program.
    • Program (Program const &otherprogram)
          Copy constructor of a program.
    • Program (string const file)
           Constructor with the input file of program.
    • ∼Program ()
          Destructor of program.

    void add_line (Line *newline)

          Add a line at the end of the program.

    int add_line_at (Line *newline, int position)

          Add a line to the program with position as index.

    void exchange_line (int line1, int line2)

          Reverse two lines which are at the index line1 and line2.
    · void display ()
          display the program
    • void del_line (int index)
          Delete the line at the given index in the program.

    Line * find_line (int index)

          gives the line that corresponds to the index
    • int size ()
          get the length of the program

    void in_file (string const filename)

          returns the dependance betwen the two given instructions
    • bool is empty ()
          return true if the program is Empty

    void comput_function ()

          calculate the functions of the program
```

get the number of functions in the program

• int nbr_func ()

Function * get_function (int index)

returns the function of index index in the list _myfunc

- list< Function * >::iterator function_list_begin ()
- list < Function * >::iterator function_list_end ()
- void flush ()

empty the program

void comput_CFG ()

calculate the CFG associated with each function of the program

Cfg * get_CFG (int index)

returns the CFG of index index in the list _myCFG

4.23.1 Detailed Description

class representing a program as list of lines

4.23.2 Member Function Documentation

```
4.23.2.1 void Program::in_file ( string const filename )
```

returns the dependance betwen the two given instructions

Returns

returns the dependance in the enum formatwrite the programme into a file

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

· Program.h

4.24 s Profile Struct Reference

Structure allowing to add caracteristics to an operator.

```
#include <Enum_type.h>
```

Public Attributes

- t_Operator op
- std::string nom
- t Format format
- t_Inst type
- int nb_oper

4.24.1 Detailed Description

Structure allowing to add caracteristics to an operator.

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

· Enum_type.h

4.25 TestOPLabel Class Reference

Inheritance diagram for TestOPLabel:



Public Member Functions

- void setUp (void)
- void tearDown (void)

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

· TestOPLabel.h

4.26 utchn Struct Reference

Public Attributes

- struct utchn * NEXT
- union utdat DATA

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

utl200.h

4.27 utdat Union Reference

Public Attributes

- void * VPNT
- float FLOT
- unsigned int **UINT**
- int SINT
- char CHAR
- unsigned char UCHR

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

utl200.h

4.28 utdic Struct Reference

Public Attributes

struct utdic * NEXT

32 Class Documentation

- struct utdit * TABLE
- void *(* ADD_K)()
- void(* FRE_K)()
- int(* CMP_K)()
- void *(* ADD_D)()
- void(* FRE_D)()
- unsigned int(* HSH_K)()
- unsigned short SIZE
- unsigned short SPEED
- · unsigned int INIT
- unsigned int STATUS
- · unsigned int FLAG

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

utl200.h

4.29 utdit Struct Reference

Public Attributes

struct uttyp * ITEM

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

utl200.h

4.30 uttdc Struct Reference

Public Attributes

- struct uttdc * NEXT
- union utdat DAT1
- union utdat DAT2
- union utdat DAT3

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

utl200.h

4.31 uttpd Struct Reference

Public Attributes

- struct uttpd * NEXT
- union utdat DAT1
- double DAT2

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

utl200.h

4.32 uttyp Struct Reference

Public Attributes

- struct uttyp * NEXT
- union utdat DAT1
- union utdat DAT2

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

utl200.h

4.33 YYSTYPE Union Reference

Public Attributes

- struct utchn * pchn
- · unsigned int uval
- char * text

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

• asm_mipsyac.h

34 Class Documentation

Chapter 5

File Documentation

5.1 Basic_block.h File Reference

Basic_block class.

```
#include <Node.h>
#include <Instruction.h>
#include <string>
#include <stdio.h>
#include <Enum_type.h>
#include <fstream>
#include <list>
#include <Dfg.h>
#include <Node_dfg.h>
```

Classes

• class Basic_block

class representing a Basic_block of a fonction

Macros

• #define NB_REGISTRES 32

5.1.1 Detailed Description

Basic_block class.

Author

Hajjem

5.2 Cfg.h File Reference

Cfg class.

```
#include <Basic_block.h>
#include <string>
#include <stdio.h>
#include <Label.h>
#include <Enum_type.h>
#include <list>
#include <fstream>
```

Classes

• class Cfg

class representing control flow graph

5.2.1 Detailed Description

Cfg class.

Author

Hajjem

5.3 Dfg.h File Reference

Dfg class.

```
#include <Node_dfg.h>
#include <Instruction.h>
#include <Enum_type.h>
#include <fstream>
#include <list>
#include <boost/graph/adjacency_list.hpp>
#include <boost/graph/astar_search.hpp>
```

Classes

· class Dfg

class representing a Dfg of a Basic block, a data flow graph that is to be used to calculate the critical path and schedule code

5.3.1 Detailed Description

Dfg class.

Author

Hajjem

5.4 Directive.h File Reference

Directive class.

```
#include <iostream>
#include <string>
#include <Enum_type.h>
#include <Line.h>
```

Classes

class Directive

class representing an Directive herited by Line

5.4.1 Detailed Description

Directive class.

Author

Hajjem

5.5 Function.h File Reference

Function class.

```
#include <Node.h>
#include <Basic_block.h>
#include <Instruction.h>
#include <string>
#include <stdio.h>
#include <Label.h>
#include <Enum_type.h>
#include <list>
#include <Cfg.h>
#include <fstream>
```

Classes

class Function

class representing a Function on a program

5.5.1 Detailed Description

Function class.

Author

Hajjem

5.6 Instruction.h File Reference

Instruction class.

```
#include <Operand.h>
#include <string>
#include <OPExpression.h>
#include <OPImmediate.h>
#include <OPLabel.h>
#include <Line.h>
#include <OPRegister.h>
#include <Enum_type.h>
#include <list>
```

Classes

- struct dep
- · class Instruction

class representing an instruction which herited by Line

5.6.1 Detailed Description

Instruction class.

Author

Hajjem - Heydemann - Girault

5.7 Label.h File Reference

Label class.

```
#include <iostream>
#include <string>
#include <Enum_type.h>
#include <Line.h>
```

Classes

class Label

class representing an Label herited by Line

5.7.1 Detailed Description

Label class.

Author

Hajjem

5.8 Line.h File Reference

Line class.

5.9 Node.h File Reference 39

```
#include <iostream>
#include <string>
#include <Enum_type.h>
```

Classes

• class Line

Abstract class representing an Line.

5.8.1 Detailed Description

Line class.

Author

Hajjem

5.9 Node.h File Reference

Node class.

```
#include <Line.h>
#include <string>
#include <Enum_type.h>
```

Classes

• class Node

class representing a Node in list

5.9.1 Detailed Description

Node class.

Author

Hajjem

5.10 Node_dfg.h File Reference

Node_dfg class.

```
#include <Basic_block.h>
#include <string>
#include <stdio.h>
#include <Label.h>
#include <Enum_type.h>
```

Classes

- struct Arc_t
- class Node_dfg

class representing a node of data flow graph

5.10.1 Detailed Description

```
Node_dfg class.
```

Author

Hajjem

5.11 Operand.h File Reference

Operand class.

```
#include <iostream>
#include <string>
#include <Enum_type.h>
```

Classes

class Operand

Abstract class representing an operand.

5.11.1 Detailed Description

Operand class.

Author

Hajjem

5.12 OPExpression.h File Reference

OPExpression class.

```
#include <iostream>
#include <string>
#include <Operand.h>
#include <Enum_type.h>
```

Classes

class OPExpression

class representing an expression herited by Operand

5.12.1 Detailed Description

OPExpression class.

Author

Hajjem

5.13 OPImmediate.h File Reference

OPImmediate class.

```
#include <iostream>
#include <string>
#include <Operand.h>
#include <Enum_type.h>
```

Classes

· class OPImmediate

class representing an Immediate herited by Operand

5.13.1 Detailed Description

OPImmediate class.

Author

Hajjem

5.14 OPLabel.h File Reference

OPLabel class.

```
#include <iostream>
#include <Operand.h>
#include <Enum_type.h>
#include <string>
```

Classes

class OPLabel

class representing a Label herited by Operand

5.14.1 Detailed Description

OPLabel class.

Author

Hajjem

5.15 OPRegister.h File Reference

OPRegister class.

```
#include <iostream>
#include <string>
#include <Operand.h>
#include <Enum_type.h>
```

Classes

· class OPRegister

class representing a Register herited by Operand

5.15.1 Detailed Description

OPRegister class.

Author

Hajjem

5.16 Program.h File Reference

Program class.

```
#include <Node.h>
#include <Function.h>
#include <Basic_block.h>
#include <Instruction.h>
#include <Directive.h>
#include <Cfg.h>
#include <string>
#include <stdio.h>
#include <Enum_type.h>
#include <fstream>
#include <list>
```

Classes

· class Program

class representing a program as list of lines

5.16.1 Detailed Description

Program class.

Author

Hajjem

Index

Arc_t, 7	is_dep_MEM
asfig, 7	Instruction, 17
asisc, 7	is_dep_RAW
asiss, 8	Instruction, 17
asobj, 8	is_dep_RAW1
asosc, 8	Instruction, 17
	is_dep_RAW2
Basic_block, 9	Instruction, 17
Basic_block.h, 35	is_dep_WAR
	Instruction, 17
Cfg, 10	is_dep_WAR1
Cfg.h, 35	
,	Instruction, 18
dep, 11	is_dep_WAR2
Dfg, 11	Instruction, 18
Dfg.h, 36	is_dep_WAW
Directive, 12	Instruction, 18
	is_dependant
Directive.h, 36	Instruction, 18
Function 40	
Function, 13	Label, 19
Function.h, 37	Label.h, 38
get_nbOp	Line, 19
Instruction, 17	to_string, 20
get_op	Line.h, 38
OPExpression, 24	
OPImmediate, 25	Node, 20
OPImmediate, 25	Node, 20 Node.h, 39
OPImmediate, 25 OPRegister, 28	Node.h, 39
OPImmediate, 25 OPRegister, 28 get_op_type	Node.h, 39 Node_dfg, 21
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23	Node.h, 39
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24 OPImmediate, 25	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39 OPExpression, 23
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24 OPImmediate, 25 OPLabel, 27	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39 OPExpression, 23 get_op, 24
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24 OPImmediate, 25	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39 OPExpression, 23 get_op, 24 get_op_type, 24
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24 OPImmediate, 25 OPLabel, 27	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39 OPExpression, 23 get_op, 24 get_op_type, 24 to_string, 24
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24 OPImmediate, 25 OPLabel, 27 OPRegister, 28 get_reg	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39 OPExpression, 23 get_op, 24 get_op_type, 24 to_string, 24 OPExpression.h, 40
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24 OPImmediate, 25 OPLabel, 27 OPRegister, 28	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39 OPExpression, 23 get_op, 24 get_op_type, 24 to_string, 24
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24 OPImmediate, 25 OPLabel, 27 OPRegister, 28 get_reg	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39 OPExpression, 23 get_op, 24 get_op_type, 24 to_string, 24 OPExpression.h, 40 OPImmediate, 25
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24 OPImmediate, 25 OPLabel, 27 OPRegister, 28 get_reg OPRegister, 28 in_file	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39 OPExpression, 23 get_op, 24 get_op_type, 24 to_string, 24 OPExpression.h, 40 OPImmediate, 25 get_op, 25
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24 OPImmediate, 25 OPLabel, 27 OPRegister, 28 get_reg OPRegister, 28 in_file Program, 30	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39 OPExpression, 23 get_op, 24 get_op_type, 24 to_string, 24 OPExpression.h, 40 OPImmediate, 25 get_op, 25 get_op_type, 25
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24 OPImmediate, 25 OPLabel, 27 OPRegister, 28 get_reg OPRegister, 28 in_file Program, 30 Instruction, 14	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39 OPExpression, 23 get_op, 24 get_op_type, 24 to_string, 24 OPExpression.h, 40 OPImmediate, 25 get_op, 25 get_op_type, 25 to_string, 26
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24 OPImmediate, 25 OPLabel, 27 OPRegister, 28 get_reg OPRegister, 28 in_file Program, 30 Instruction, 14 get_nbOp, 17	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39 OPExpression, 23 get_op, 24 get_op_type, 24 to_string, 24 OPExpression.h, 40 OPImmediate, 25 get_op, 25 get_op_type, 25 to_string, 26 OPImmediate.h, 41
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24 OPImmediate, 25 OPLabel, 27 OPRegister, 28 get_reg OPRegister, 28 in_file Program, 30 Instruction, 14 get_nbOp, 17 is_dep_MEM, 17	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39 OPExpression, 23 get_op, 24 get_op_type, 24 to_string, 24 OPExpression.h, 40 OPImmediate, 25 get_op, 25 get_op_type, 25 to_string, 26 OPImmediate.h, 41 OPLabel, 26
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24 OPImmediate, 25 OPLabel, 27 OPRegister, 28 get_reg OPRegister, 28 in_file Program, 30 Instruction, 14 get_nbOp, 17 is_dep_MEM, 17 is_dep_RAW, 17	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39 OPExpression, 23 get_op, 24 get_op_type, 24 to_string, 24 OPExpression.h, 40 OPImmediate, 25 get_op, 25 get_op_type, 25 to_string, 26 OPImmediate.h, 41 OPLabel, 26 get_op_type, 27
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24 OPImmediate, 25 OPLabel, 27 OPRegister, 28 get_reg OPRegister, 28 in_file Program, 30 Instruction, 14 get_nbOp, 17 is_dep_MEM, 17 is_dep_RAW1, 17	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39 OPExpression, 23 get_op, 24 get_op_type, 24 to_string, 24 OPExpression.h, 40 OPImmediate, 25 get_op, 25 get_op_type, 25 to_string, 26 OPImmediate.h, 41 OPLabel, 26 get_op_type, 27 to_string, 27
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24 OPImmediate, 25 OPLabel, 27 OPRegister, 28 get_reg OPRegister, 28 in_file Program, 30 Instruction, 14 get_nbOp, 17 is_dep_MEM, 17 is_dep_RAW1, 17 is_dep_RAW2, 17	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39 OPExpression, 23 get_op, 24 get_op_type, 24 to_string, 24 OPExpression.h, 40 OPImmediate, 25 get_op, 25 get_op_type, 25 to_string, 26 OPImmediate.h, 41 OPLabel, 26 get_op_type, 27 to_string, 27 OPLabel.h, 41
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24 OPImmediate, 25 OPLabel, 27 OPRegister, 28 get_reg OPRegister, 28 in_file Program, 30 Instruction, 14 get_nbOp, 17 is_dep_MEM, 17 is_dep_RAW, 17 is_dep_RAW1, 17 is_dep_RAW2, 17 is_dep_WAR, 17	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39 OPExpression, 23 get_op, 24 get_op_type, 24 to_string, 24 OPExpression.h, 40 OPImmediate, 25 get_op, 25 get_op_type, 25 to_string, 26 OPImmediate.h, 41 OPLabel, 26 get_op_type, 27 to_string, 27 OPLabel.h, 41 OPRegister, 27
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24 OPImmediate, 25 OPLabel, 27 OPRegister, 28 get_reg OPRegister, 28 in_file Program, 30 Instruction, 14 get_nbOp, 17 is_dep_MEM, 17 is_dep_RAW1, 17 is_dep_RAW2, 17	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39 OPExpression, 23 get_op, 24 get_op_type, 24 to_string, 24 OPExpression.h, 40 OPImmediate, 25 get_op, 25 get_op_type, 25 to_string, 26 OPImmediate.h, 41 OPLabel, 26 get_op_type, 27 to_string, 27 OPLabel.h, 41
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24 OPImmediate, 25 OPLabel, 27 OPRegister, 28 get_reg OPRegister, 28 in_file Program, 30 Instruction, 14 get_nbOp, 17 is_dep_MEM, 17 is_dep_RAW, 17 is_dep_RAW1, 17 is_dep_RAW2, 17 is_dep_WAR, 17	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39 OPExpression, 23 get_op, 24 get_op_type, 24 to_string, 24 OPExpression.h, 40 OPImmediate, 25 get_op, 25 get_op_type, 25 to_string, 26 OPImmediate.h, 41 OPLabel, 26 get_op_type, 27 to_string, 27 OPLabel.h, 41 OPRegister, 27
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24 OPImmediate, 25 OPLabel, 27 OPRegister, 28 get_reg OPRegister, 28 in_file Program, 30 Instruction, 14 get_nbOp, 17 is_dep_MEM, 17 is_dep_RAW, 17 is_dep_RAW1, 17 is_dep_RAW2, 17 is_dep_WAR1, 18 is_dep_WAR2, 18	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39 OPExpression, 23 get_op, 24 get_op_type, 24 to_string, 24 OPExpression.h, 40 OPImmediate, 25 get_op, 25 get_op_type, 25 to_string, 26 OPImmediate.h, 41 OPLabel, 26 get_op_type, 27 to_string, 27 OPLabel.h, 41 OPRegister, 27 get_op, 28 get_op_type, 28
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24 OPImmediate, 25 OPLabel, 27 OPRegister, 28 get_reg OPRegister, 28 in_file Program, 30 Instruction, 14 get_nbOp, 17 is_dep_MEM, 17 is_dep_RAW, 17 is_dep_RAW1, 17 is_dep_RAW2, 17 is_dep_WAR1, 18 is_dep_WAR2, 18 is_dep_WAW, 18	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39 OPExpression, 23 get_op, 24 get_op_type, 24 to_string, 24 OPExpression.h, 40 OPImmediate, 25 get_op, 25 get_op_type, 25 to_string, 26 OPImmediate.h, 41 OPLabel, 26 get_op_type, 27 to_string, 27 OPLabel.h, 41 OPRegister, 27 get_op, 28 get_op_type, 28 get_reg, 28
OPImmediate, 25 OPRegister, 28 get_op_type Operand, 23 OPExpression, 24 OPImmediate, 25 OPLabel, 27 OPRegister, 28 get_reg OPRegister, 28 in_file Program, 30 Instruction, 14 get_nbOp, 17 is_dep_MEM, 17 is_dep_RAW, 17 is_dep_RAW1, 17 is_dep_RAW2, 17 is_dep_WAR1, 18 is_dep_WAR2, 18	Node.h, 39 Node_dfg, 21 Node_dfg.h, 39 OPExpression, 23 get_op, 24 get_op_type, 24 to_string, 24 OPExpression.h, 40 OPImmediate, 25 get_op, 25 get_op_type, 25 to_string, 26 OPImmediate.h, 41 OPLabel, 26 get_op_type, 27 to_string, 27 OPLabel.h, 41 OPRegister, 27 get_op, 28 get_op_type, 28

44 INDEX

```
Operand, 22
    get_op_type, 23
    to_string, 23
Operand.h, 40
Program, 29
    in_file, 30
Program.h, 42
s_Profile, 30
TestOPLabel, 31
to_string
    Line, 20
    Operand, 23
    OPExpression, 24
    OPImmediate, 26
    OPLabel, 27
    OPRegister, 28
utchn, 31
utdat, 31
utdic, 31
utdit, 32
uttdc, 32
uttpd, 32
uttyp, 33
YYSTYPE, 33
```