1 Main Page

2 File Index
   2.1 File List ........................................................................................................ 11

3 File Documentation
   3.1 biddy.h File Reference ..................................................................................... 13
       3.1.1 Detailed Description .................................................................................. 15
       3.1.2 Macro Definition Documentation ............................................................... 16
           3.1.2.1 Biddy_IsNull ...................................................................................... 16
           3.1.2.2 Biddy_IsConstant .............................................................................. 16
           3.1.2.3 Biddy_IsEqvPointer .......................................................................... 16
           3.1.2.4 Biddy_GetMark .................................................................................... 16
           3.1.2.5 Biddy_SetMark .................................................................................... 16
           3.1.2.6 Biddy_ClearMark .................................................................................. 16
           3.1.2.7 Biddy_InvertMark ................................................................................. 17
           3.1.2.8 Biddy_Not ............................................................................................ 17
           3.1.2.9 Biddy_NotCond .................................................................................... 17
           3.1.2.10 Biddy_Regular .................................................................................... 17
           3.1.2.11 Biddy_Complement .............................................................................. 17
           3.1.2.12 Biddy_String2Variable ....................................................................... 17
           3.1.2.13 Biddy_GetConstantZero ....................................................................... 17
           3.1.2.14 Biddy_GetConstantOne ....................................................................... 17
           3.1.2.15 Biddy_GetVariableName ..................................................................... 17
           3.1.2.16 Biddy_GetTopVariableName ................................................................ 18
           3.1.2.17 Biddy_GetTopVariableChar ................................................................ 18
           3.1.2.18 Biddy_ResetVariablesValue ................................................................. 18
           3.1.2.19 Biddy_SetVariableValue ..................................................................... 18
           3.1.2.20 Biddy_IsSmaller ................................................................................... 18
           3.1.2.21 Biddy_AddVariable .............................................................................. 18
           3.1.2.22 Biddy_AddVariableBelow ................................................................... 18
           3.1.2.23 Biddy_AddVariableAbove ................................................................... 18
3.1.2.64 Biddy_VariableTableNum ........................................... 23
3.1.2.65 Biddy_NodeTableSize .................................................. 23
3.1.2.66 Biddy_NodeTableMax ................................................... 23
3.1.2.67 Biddy_NodeTableNum .................................................. 23
3.1.2.68 Biddy_NodeTableNumVar ............................................. 23
3.1.2.69 Biddy_NodeTableNumF .................................................. 23
3.1.2.70 Biddy_NodeTableFOA ................................................... 24
3.1.2.71 Biddy_NodeTableCompare .......................................... 24
3.1.2.72 Biddy_NodeTableAdd .................................................. 24
3.1.2.73 Biddy_NodeTableGarbage ........................................... 24
3.1.2.74 Biddy_NodeTableGenerated .......................................... 24
3.1.2.75 Biddy_NodeTableBlockNumber ..................................... 24
3.1.2.76 Biddy_NodeTableSwapNumber ...................................... 24
3.1.2.77 Biddy_NodeTableSiftingNumber ................................... 24
3.1.2.78 Biddy_ListUsed ........................................................ 24
3.1.2.79 Biddy_ListMaxLength ............................................... 25
3.1.2.80 Biddy_ListAvgLength ................................................ 25
3.1.2.81 Biddy_IteCacheSearch .............................................. 25
3.1.2.82 Biddy_IteCacheFind .................................................. 25
3.1.2.83 Biddy_IteCacheOverwrite .......................................... 25
3.1.2.84 Biddy_NodeNumberPlain ........................................... 25
3.1.2.85 Biddy_VariableNumber .............................................. 25
3.1.2.86 Biddy_NodeVarNumber ............................................... 25
3.1.2.87 Biddy_CountMinterm ................................................ 25
3.1.2.88 Biddy_ReadMemoryInUse ........................................... 26
3.1.2.89 Biddy_PrintInfo ........................................................ 26
3.1.2.90 Biddy_Eval0 ............................................................ 26
3.1.2.91 Biddy_Eval1 ............................................................ 26
3.1.2.92 Biddy_Eval1x ............................................................ 26
3.1.2.93 Biddy_Eval2 ............................................................. 26
3.1.2.94 Biddy_WriteBDD ........................................................ 26
3.1.2.95 Biddy_WriteBDDx ....................................................... 26
3.1.2.96 Biddy_WriteDot ........................................................ 26
3.1.2.97 Biddy_WriteDotx ........................................................ 27
3.1.2.98 Biddy_WriteTable ...................................................... 27

3.1.3 Typedef Documentation ................................................. 27
3.1.3.1 Biddy_Boolean .......................................................... 27
3.1.3.2 Biddy_String ............................................................. 27
3.1.3.3 Biddy_Manager ........................................................... 27
3.1.3.4 Biddy_Cache .............................................................. 27
3.1.3.5  Biddy_Variable  .................................................. 27
3.1.3.6  Biddy_Edge  ..................................................... 27
3.1.3.7  Biddy_GCFunction  .............................................. 28
3.1.3.8  Biddy_LookupFunction  ....................................... 28

3.2  biddyInOut.c File Reference ........................................... 28

3.2.1  Detailed Description .................................................. 28

3.2.2  Function Documentation ............................................. 29

3.2.2.1  Biddy_Managed_Eval0 ........................................... 29
3.2.2.2  Biddy_Managed_Eval1 ........................................... 29
3.2.2.3  Biddy_Managed_Eval2 ........................................... 29
3.2.2.4  Biddy_Managed_WriteBDD ....................................... 30
3.2.2.5  Biddy_Managed_WriteDot ...................................... 30
3.2.2.6  Biddy_Managed_WriteTable .................................... 30

3.3  biddyInt.h File Reference ............................................. 31

3.3.1  Detailed Description .................................................. 31

3.4  biddyMain.c File Reference ........................................... 31

3.4.1  Detailed Description .................................................. 34

3.4.2  Function Documentation ............................................. 35

3.4.2.1  Biddy_Init ......................................................... 35
3.4.2.2  Biddy_Exit ....................................................... 35
3.4.2.3  Biddy_About ...................................................... 35
3.4.2.4  Biddy_GetThen ................................................... 36
3.4.2.5  Biddy_GetElse ................................................... 36
3.4.2.6  Biddy_GetTopVariable .......................................... 36
3.4.2.7  Biddy_SelectNode ............................................... 37
3.4.2.8  Biddy_DeselectNode ............................................ 38
3.4.2.9  Biddy_IsSelected ............................................... 38
3.4.2.10 Biddy_NodeSelect ............................................... 39
3.4.2.11 Biddy_NodeRepair .............................................. 39
3.4.2.12 Biddy_NOT ....................................................... 40
3.4.2.13 Biddy_TransferMark ............................................. 40
3.4.2.14 Biddy_Managed_GetConstantZero ................................ 40
3.4.2.15 Biddy_Managed_GetConstantOne ................................ 41
3.4.2.16 Biddy_Managed_GetVariableName ................................ 41
3.4.2.17 Biddy_Managed_GetTopVariableName .......................... 41
3.4.2.18 Biddy_Managed_GetTopVariableChar .......................... 42
3.4.2.19 Biddy_Managed_ResetVariablesValue ........................ 42
3.4.2.20 Biddy_Managed_SetVariableValue ............................ 42
3.4.2.21 Biddy_Managed_IsSmaller ..................................... 42
3.4.2.22 Biddy_Managed_AddVariable .................................. 43
<table>
<thead>
<tr>
<th>Section</th>
<th>Function Name</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.4.2.23</td>
<td>Biddy_Managed_AddVariableBelow</td>
<td>43</td>
</tr>
<tr>
<td>3.4.2.24</td>
<td>Biddy_Managed_AddVariableAbove</td>
<td>44</td>
</tr>
<tr>
<td>3.4.2.25</td>
<td>Biddy_Managed_FoVarVariable</td>
<td>44</td>
</tr>
<tr>
<td>3.4.2.26</td>
<td>Biddy_Managed_FoVarNode</td>
<td>45</td>
</tr>
<tr>
<td>3.4.2.27</td>
<td>Biddy_Managed_ITE</td>
<td>45</td>
</tr>
<tr>
<td>3.4.2.28</td>
<td>Biddy_Managed_And</td>
<td>46</td>
</tr>
<tr>
<td>3.4.2.29</td>
<td>Biddy_Managed_Or</td>
<td>47</td>
</tr>
<tr>
<td>3.4.2.30</td>
<td>Biddy_Managed_Nand</td>
<td>47</td>
</tr>
<tr>
<td>3.4.2.31</td>
<td>Biddy_Managed_Nor</td>
<td>47</td>
</tr>
<tr>
<td>3.4.2.32</td>
<td>Biddy_Managed_Xor</td>
<td>47</td>
</tr>
<tr>
<td>3.4.2.33</td>
<td>Biddy_Managed_Xnor</td>
<td>48</td>
</tr>
<tr>
<td>3.4.2.34</td>
<td>Biddy_Managed_Leq</td>
<td>48</td>
</tr>
<tr>
<td>3.4.2.35</td>
<td>Biddy_Managed_Intersect</td>
<td>48</td>
</tr>
<tr>
<td>3.4.2.36</td>
<td>Biddy_Managed_Restrict</td>
<td>49</td>
</tr>
<tr>
<td>3.4.2.37</td>
<td>Biddy_Managed_Compose</td>
<td>49</td>
</tr>
<tr>
<td>3.4.2.38</td>
<td>Biddy_Managed_E</td>
<td>49</td>
</tr>
<tr>
<td>3.4.2.39</td>
<td>Biddy_Managed_A</td>
<td>50</td>
</tr>
<tr>
<td>3.4.2.40</td>
<td>Biddy_Managed_IsVariableDependent</td>
<td>50</td>
</tr>
<tr>
<td>3.4.2.41</td>
<td>Biddy_Managed_ExistAbstract</td>
<td>51</td>
</tr>
<tr>
<td>3.4.2.42</td>
<td>Biddy_Managed_UnivAbstract</td>
<td>51</td>
</tr>
<tr>
<td>3.4.2.43</td>
<td>Biddy_Managed_AndAbstract</td>
<td>52</td>
</tr>
<tr>
<td>3.4.2.44</td>
<td>Biddy_Managed_Constrain</td>
<td>52</td>
</tr>
<tr>
<td>3.4.2.45</td>
<td>Biddy_Managed_Simplify</td>
<td>52</td>
</tr>
<tr>
<td>3.4.2.46</td>
<td>Biddy_Managed_Support</td>
<td>52</td>
</tr>
<tr>
<td>3.4.2.47</td>
<td>Biddy_Managed_Replace</td>
<td>53</td>
</tr>
<tr>
<td>3.4.2.48</td>
<td>Biddy_Managed_IsOK</td>
<td>53</td>
</tr>
<tr>
<td>3.4.2.49</td>
<td>Biddy_Managed_Garbage</td>
<td>54</td>
</tr>
<tr>
<td>3.4.2.50</td>
<td>Biddy_Managed_Clean</td>
<td>54</td>
</tr>
<tr>
<td>3.4.2.51</td>
<td>Biddy_Managed_ClearAll</td>
<td>55</td>
</tr>
<tr>
<td>3.4.2.52</td>
<td>Biddy_Managed_AddCache</td>
<td>55</td>
</tr>
<tr>
<td>3.4.2.53</td>
<td>Biddy_Managed_FindFormula</td>
<td>55</td>
</tr>
<tr>
<td>3.4.2.54</td>
<td>Biddy_Managed_AddFormula</td>
<td>56</td>
</tr>
<tr>
<td>3.4.2.55</td>
<td>Biddy_Managed_DeleteFormula</td>
<td>56</td>
</tr>
<tr>
<td>3.4.2.56</td>
<td>Biddy_Managed_DeleteIthFormula</td>
<td>56</td>
</tr>
<tr>
<td>3.4.2.57</td>
<td>Biddy_Managed_GetIthFormula</td>
<td>57</td>
</tr>
<tr>
<td>3.4.2.58</td>
<td>Biddy_Managed_GetIthFormulaName</td>
<td>57</td>
</tr>
<tr>
<td>3.4.2.59</td>
<td>Biddy_Managed_SwapWithHigher</td>
<td>58</td>
</tr>
<tr>
<td>3.4.2.60</td>
<td>Biddy_Managed_SwapWithLower</td>
<td>58</td>
</tr>
<tr>
<td>3.4.2.61</td>
<td>Biddy_Managed_Sifting</td>
<td>58</td>
</tr>
<tr>
<td>3.4.2.62</td>
<td>Biddy_Managed_Purge</td>
<td>59</td>
</tr>
</tbody>
</table>
Chapter 1

Main Page

Biddy is a multi-platform academic Binary Decision Diagrams package.
Biddy is maintained by Robert Meolic (robert.meolic@um.si) at University of Maribor, Slovenia.
Homepage: http://biddy.meolic.com/

1. AN OVERVIEW

Biddy is free software released under GPL.
Biddy includes automatic garbage collection with a formulae counter, node management with formulae tagging, and variable swapping. Sifting algorithm is implemented.
Biddy is optimized for efficiency, but it is mainly oriented towards readable and comprehensible source code in C.
Biddy is currently used in the following projects:

- BDD Scout, demo project
- Efficient Symbolic Tools (EST), http://est.meolic.com/

2. SOURCE CODE

The short name of Biddy package is 'biddy'. This name is placed in front of all filenames and external identifiers. It may appear in all lowercase, or with its first letter capitalized, written as 'biddy' and 'Biddy', respectively.

There are three categories of C functions.

- Exported functions are visible outside the package.
- Internal functions are visible to all files within the package.
- Static functions are visible to the file only.

There are two types of C functions.

- General functions which operates on a particular BDD and considering only graph properties (i.e. changing edge's mark, selecting nodes, counting nodes etc.). These functions are the same for different type of decision diagrams (BDD, ZDD, etc.). Functions, which add or delete nodes or those which needs info about variables (e.g. name) are not general functions. Exported general functions have prefix Biddy_.
• Managed functions, which operates on a global properties of a BDD system (e.g. node table, variable table, formula table, various caches, etc.) or consider a BDD as a Boolean function (e.g. Boolean operations, counting minterms, etc.). These functions need info stored in a manager. Exported managed functions have prefix Biddy_Managed_.

Biddy consists of the following files:

• README.md (this file)
• CHANGES (history of changes)
• COPYING (license file)
• VERSION (project’s version)
• Makefile (used to produce binary code)
• Makefile.Linux (Makefile definitions for GNU/Linux)
• Makefile.MINGW (Makefile definitions for MS Windows)
• Makefile.Darwin (Makefile definitions for MacOS)
• biddy.h (header)
• biddyInt.h (header)
• biddyMain.c (main functions)
• biddyStat.c (functions for statistic)
• biddyInOut.c (parsers and generators for Boolean functions)
• package-source (script used to build distribution)
• package-bin (script used to build distribution)
• package-bin.bat (script used to build distribution)
• package-tgz (script used to build distribution)
• package-rpm (script used to build distribution)
• debian/* (files used when creating deb package)
• rpm/* (files used when creating rpm package)
• biddy-example-8queens.c (8 Queens example)

There are two C headers, external and internal. The external header file, named biddy.h, defines features visible from outside the package. The internal header file, named biddyInt.h defines features used in multiple files inside the package, but not outside.

There are two additional packages included into Biddy distribution: bddview and Bdd Scout.

bddview is a pure Tcl/Tk script for visualization of BDDs.

BDD Scout is a demo application demonstrating the capability of Biddy and bddview.

To compile Biddy library, you can use:

```
biddy> make dynamic "BINDIR = ./bin"
biddy> make clean "BINDIR = ./bin"
```

More details about building are given in Section 4.
3. USING BIDDY LIBRARY

Precompiled packages include dynamically linked library (i.e. *.so on GNU/Linux, *.dll on MS Windows, *.dylib on Mac OS X), and the appropriate C header biddy.h. Currently, there are no interfaces for other programming languages.

For linking with Biddy library you have to use:

```
-lbiddy -lgmp
```

Biddy is capable of all the typical operations regarding Boolean functions and BDDs.

The following code is an example of usage. Please note, that functions for node management are not shown. Moreover, Biddy has a manager but its usage is optional and it is also not shown in the given example.

IMPORTANT: You should define UNIX, MACOSX, or MINGW. You should define USE_BIDDY iff you are using Biddy via dynamic library.

```
#define UNIX
#define USE_BIDDY
#include "/path/to/biddy.h"
#define Str2Var(x) (Biddy_GetTopVariable(Biddy_FoaVariable((Biddy_String)x)))

int main() {
    Biddy_Edge f, g, h, r;
    Biddy_Init();

    f = Biddy_Eval1((Biddy_String)"(OR H E L L O)"); /* PREFIX INPUT */
    g = Biddy_Eval1((Biddy_String)"(AND W O R L D)"); /* PREFIX INPUT */
    h = Biddy_Eval2((Biddy_String)"(H * E * L * L)"); /* INFIX INPUT */
    /* BASIC OPERATION */
    r = Biddy_Xor(f, g);
    /* REPLACE SOME VARIABLES */
    Biddy_ResetVariablesValue();
    Biddy_SetVariableValue(Str2Var("H"), Biddy_FoaVariable((Biddy_String)"L"));
    Biddy_SetVariableValue(Str2Var("K"), Biddy_FoaVariable((Biddy_String)"L"));
    Biddy_SetVariableValue(Str2Var("W"), Biddy_FoaVariable((Biddy_String)"L"));
    r = Biddy_Replace(r);
    /* SIMPLE RESTRICTION */
    r = Biddy_Restrict(r, Str2Var("E"), Biddy_GetConstantZero());
    /* COUDERT AND MADRE’S RESTRICT FUNCTION */
    r = Biddy_Simplify(r);
    /* SOME STATS */
    printf("Function r depends on %u variables.\n", Biddy_VariableNumber(r));
    printf("Function r has %.0f minterms.\n", Biddy_CountMinterm(r, 0));
    /* TRUTH TABLE */
    printf("Here is a truth table for function r:\n");
    Biddy_WriteTable(r);
    /* GRAPHVIZ/DOT OUTPUT */
    Biddy_WriteDot("biddy.dot", r, "Function r");
    printf("USE ‘dot -y -Tpng -o biddy.dot’ to visualize BDD for function r.\n");
    Biddy_Exit();
}
```

3.1 NODE MANAGEMENT WITH FORMULAE TAGGING

Biddy includes powerful node management based on formulae tagging. There are seven user functions to maintain nodes.
**Biddy_AddFormula(name,bdd,count)**

Nodes of the given BDD will be preserved for the given number of cleanings. If (name != NULL) then formula is accessible by its name. If formula with a given name already exists it is overwritten. If (count == 0) then formula is persistently preserved and you have to use Biddy_DeleteFormula and Biddy_ClearAll to remove its nodes. There are two macros defined to simplify formulae management. Macro Biddy_AddTmpFormula(bdd,count) is defined as Biddy_AddFormula(NULL,bdd,count) and macro Biddy_AddPersistentFormula(name,bdd) is defined as Biddy_AddFormula(name,bdd,0).

**Biddy_DeleteFormula(name)**

Nodes of the given formula are tagged as not needed. Formula is not accessible by its name anymore. Regular cleaning with Biddy_Clean is not considered this tag. Obsolete nodes from preserved formulae can be immediately removed in explicite call to Biddy_ClearAll (this is the only way to remove obsolete nodes from permanently preserved formulae).

**Biddy_Clean()**

Performs cleaning. Discard all nodes which were not preserved or which are not preserved anymore. Obsolete nodes are not immediately removed, they will be removed during the first garbage collection. Use Biddy_Purge or Biddy_PurgeAndReorder to immediately remove all nodes which are not preserved.

**Biddy_Purge()**

Immediately remove all nodes which were not preserved or which are not preserved anymore. Call to Biddy_Purge does not count as cleaning and thus all preserved formulae remains preserved for the same number of cleanings.

**Biddy_PurgeAndReorder(bdd)**

The same as Biddy_Purge but also trigger reordering on function (if BDD is given) or global reordering (if NULL is given).

**Biddy_Refresh(bdd)**

All obsolete nodes become fresh nodes. This is an external variant of internal function BiddyRefresh. It is needed to implement user caches.

**Biddy_ClearAll()**

Immediately remove all nodes except the permanently preserved ones which were not made obsolete by Biddy_DeleteFormula. Please note, that regularly use of Biddy_Clean will trigger automatic garbage collection, thus Biddy_ClearAll is needed only if you need more extensive cleaning, e.g. to remove all obsolete nodes after a call to Biddy_DeleteFormula.

### 3.2. EXAMPLES OF NODE MANAGEMENT WITH BIDDY

The first example is a straightforward calculation.

```c
f1 = op(...);
f2 = op(...);
g1 = op(f1,f2,...);
Biddy_AddTmpFormula(g1,1);  /* g1 is preserved for next cleaning */
f1 = op(...);
```

f2 = op(...);
g2 = op(f1,f2,...);
Biddy_AddTmpFormula(g2,1); /* g2 is preserved for next cleaning */
Biddy_Clean(); /* g1 and g2 are still usable */
result = op(g1,g2,...);
Biddy_AddPersistentFormula("result",result); /* final result is permanently preserved */
Biddy_Clean(); /* tmp result are not needed, anymore */

""
If garbage collection is needed also after the calculation of g1, you must add the following code after the calculation of g1:
""
Biddy_AddTmpFormula(g1,2); /* g1 is preserved for next two cleanings */
Biddy_Clean(); /* g1 remains preserved for one additional cleaning */

""
To trigger reordering in the first example, you should add calls to Biddy_PurgeAndReorder to get the following code:
""
f1 = op(...);
f2 = op(...);
g1 = op(f1,f2,...);
Biddy_AddTmpFormula(g1,2); /* g1 is preserved for next two cleanings */
Biddy_PurgeAndReorder(g1); /* keep only preserved formulae (g1), perform reordering on g1 */
Biddy_Clean(); /* g1 remains preserved for one additional cleaning */
f1 = op(...);
f2 = op(...);
g2 = op(f1,f2,...);
Biddy_AddTmpFormula(g2,1); /* g2 is preserved for next cleaning */
Biddy_PurgeAndReorder(g2); /* keep only preserved formulae (g1, g2), perform reordering on g2 */
Biddy_Clean(); /* g1 and g2 are still usable but not preserved */
result = op(g1,g2,...);
Biddy_AddPersistentFormula("result",result); /* result is permanently preserved */
Biddy_PurgeAndReorder(result); /* keep only preserved formulae (result), perform reordering on result */
Biddy_Clean(); /* g1 and g2 are still usable but not preserved */

""
The second example is an iterative calculation:
""
f = op(...);
result = op(f,...);
while (!finish) {
    Biddy_AddTmpFormula(result,1); /* result is preserved for next cleaning */
    Biddy_Clean(); /* result is still usable but not preserved */
    f = op(...);
    g = op(...);
    result = op(result,g,...);
}]
Biddy_AddPersistentFormula("result",result); /* final result is permanently preserved */
Biddy_Clean(); /* tmp results are not needed, anymore */

""
If garbage collection is needed also after the calculation of g, you must use the following code:
""
f = op(...);
result = op(f,...);
while (!finish) {
    Biddy_AddTmpFormula(result,2); /* result is preserved for next two cleanings */
    Biddy_Clean(); /* result remains preserved for one additional cleaning */
    f = op(...);
    g = op(...);
    Biddy_AddTmpFormula(g,1); /* g is preserved for next cleaning */
    Biddy_Clean(); /* result and g are still usable but not preserved */
    result = op(result,g,...);
}]
Biddy_AddPersistentFormula("result",result); /* final result is permanently preserved */
Biddy_Clean(); /* tmp results are not needed, anymore */

""
result = op(result, g, ...);
}
Biddy_AddPersistentFormula("result", result); /* final result is permanently preserved */
Biddy_Clean(); /* tmp results are not needed, anymore */

==

To trigger reordering in the second example, you should change code in the following way:

==

f = op(...);
result = op(f, ...);
while (!finish) {
    Biddy_AddTmpFormula(result, 2); /* result is preserved for next two cleanings */
    Biddy_PurgeAndReorder(result); /* keep only preserved formulae (result), perform
    reordering on result */
    Biddy_Clean(); /* result remains preserved for one additional cleaning */
    f = op(...);
    g = op(f, ...);
    Biddy_AddTmpFormula(g, 1); /* g is preserved for next cleaning */
    Biddy_Clean(); /* result and g are still usable but not preserved */
    result = op(result, g, ...);
}
Biddy_AddPersistentFormula("result", result); /* final result is permanently preserved */
Biddy_PurgeAndReorder(result); /* keep only preserved formulae (result), perform
    reordering on result */

==

The third example is an outline of an implementation of model checking where we are trying to benefit from regularly
reordering:

==

sup = Prepare(...);
Biddy_AddPersistentFormula("sup", sup) /* sup is permanently preserved */
Z = 0;
last = 1;
while (Z!=last) {
    Biddy_AddTmpFormula(Z, 1); /* Z is preserved for next cleaning */
    Biddy_PurgeAndReorder(Z); /* keep only preserved formulae (sup, Z), perform
    reordering on Z */
    Biddy_Clean(); /* Z remains usable but not preserved */
    last = Z;
    Z = NextSet(Z, sup, ...);
}
result = Z;
Biddy_AddPersistentFormula("result", result); /* final result is permanently preserved */
Biddy_DeleteFormula("sup");
Biddy_ClearAll(); /* optional, if you really need to immediately remove sup */

==

The fourth example is an outline of an implementation of bisimulation where we are trying to benefit from regularly
reordering:

==

init = AND(init_p, init_q)
Biddy_AddPersistentFormula("init", init) /* init is permanently preserved */
eq = InitialEq(init_p, tr_p, init_q, tr_q, ...);
do {
    Biddy_AddTmpFormula(eq, 1); /* eq is preserved for next cleaning */
    Biddy_PurgeAndReorder(eq); /* keep only preserved formulae (init, eq), perform
    reordering on eq */
    Biddy_Clean(); /* eq remains usable but not preserved */
    last = eq;
    eq1 = NextEqPart1(eq, tr_p, tr_q, ...);
    eq2 = NextEqPart2(eq, tr_p, tr_q, ...);
    eq = AND(eq1, eq2);
    while (AND(init, eq)!0 && eq!=last)
        if (AND(init, eq)!0) return false; else return true;
    Biddy_DeleteFormula("init");
    Biddy_ClearAll(); /* optional, if you really need to immediately remove init */
}

==
The fifth example is an outline of an implementation of parallel composition where we are trying to benefit from intensive GC:

```
sacc = snew = AND(init_1, init_2, ..., init_N);
for (i=1;i<=N;i++) di[i] = 0;
for (i=1;i<=N;i++) for (j=1;i<=N;j++) dij[i,j] = 0;
do {
    Biddy_AddTmpFormula(snew, N*(N+1)); /* snew is preserved just long enough */
    Biddy_AddTmpFormula(sacc, N*(N+1)); /* sacc is preserved just long enough */
    new1 = 0;
    for (i=1;i<=N;i++) {
        sup = OneStep(snew, tr_i, ...);
        di[i] = OR(d[i], sup);
        new1 = OR(new1, NextState(sup, ...));
        Biddy_AddTmpFormula(d[i], N*(N+1)); /* di[i] is preserved just long enough */
        Biddy_Clean(); /* new1 remains usable but not preserved */
    }
    Biddy_AddTmpFormula(new1, N*N); /* new1 is preserved just long enough */
    new2 = 0;
    for (i=1;i<=N;i++) for (j=1;i<=N;j++) {
        sup = OneStep(snew, tr_i, tr_j, ...);
        dij[i,j] = OR(dij[i,j], sup);
        new2 = OR(new2, NextState(sup, ...));
        Biddy_AddTmpFormula(dij[i,j], N*(N+1)); /* dij[i,j] is preserved just long enough */
        Biddy_Clean(); /* new2 remains usable but not preserved */
    }
snew = AND(OR(new1, new2), NOT(sacc));
sacc = OR(sacc, snew);
} while (snew!=0)
```

```
result = OR(tr1, tr2);
Biddy_AddPersistentFormula("result", result); /* final result is permanently preserved */
Biddy_ClearAll(); /* optional, tmp result are not needed, anymore */
```

3.3 GARBAGE COLLECTION WITH A FORMULAE COUNTER

Garbage collection is automatically triggered if nodes from all reserved blocks of nodes are used. Garbage collection will remove as many obsolete nodes as possible.

Biddy does not use reference counter. We call the implemented algorithm "GC with a formulae counter". Please note, that formulae tagging as presented above can be implemented via reference counter or via formulae counter, in both case it relies on additional structures, e.g. formulae table.

While using formulae counter instead of reference counter is less efficient (i.e. it requires more work to maintain nodes) it has some advantages. It allows GC to be started in any time without breaking the ongoing calculation. Thus, there is no need to taking care of repeating broken calculations. Moreover, the usage of formulae counter instead of reference counter removes all the hassle of referencing and dereferencing nodes and thus it is favorable in an academic package oriented towards simple and readable source code.

There are four classes of nodes. Every node belongs to one of these classes:

- **fortified** node (count = 0);
- **fresh** node (count = biddyCount);
- **prolonged** node (count > biddyCount);
- **obsolete** node (0 < count < biddyCount).

All successors of a fortified node are fortified nodes. All successors of a prolonged node are prolonged or fortified nodes. All successors of a fresh node are fresh, prolonged, or fortified nodes.

There are four internal functions to maintain nodes.

- **BiddyFortify** (all nodes become fortified nodes)
- **BiddyRefresh** (all obsolete nodes become fresh nodes)
- **BiddyProlong** (all obsolete and fresh nodes become prolonged nodes)
- **BiddyIncCounter** (all prolonged nodes become fresh nodes, all fresh nodes become obsolete nodes)

There are four functions which can be used by an expert user.

- **Biddy_Garbage** (explicit garbage collection, remove all obsolete nodes)
- **Biddy_SwapWithLower** (explicit swap of two variables, it removes all obsolete and fresh nodes)
- **Biddy_SwapWithHigher** (explicit swap of two variables, it removes all obsolete and fresh nodes)
- **Biddy_Sifting** (explicit sifting, it removes all obsolete and fresh nodes)

These four functions will keep all nodes preserved by Biddy_AddFormula they will not change the class of any node. Please note, that Biddy_Garbage can be started in any time whilst Biddy_SwapWithLower, Biddy_SwapWithHigher, and Biddy_Sifting will break an ongoing calculation (because they remove fresh nodes).

### 3.4 MORE DETAILS OF MEMORY MANAGEMENT (NODE CHAINING)

Biddy relies on a single hash table for all variables. However, it supports chaining of nodes to form different lists (using an extra pointer in each node). This facility is used to improve efficiency of garbage collection and sifting.

Please note, that node chaining is not determined or limited by using formulae tagging or a formulae counter, it is an independent mechanism.

### 4. BUILDING PACKAGES

**Compiling Biddy library**

```
biddy> make dynamic "BINDIR = ./bin"
biddy> make clean "BINDIR = ./bin"
```

Alternatively, you can use:

```
biddy> make static "BINDIR = ./bin"
biddy> make debug "BINDIR = ./bin"
biddy> make profile "BINDIR = ./bin"
```

On MS Windows, we are using MSYS2. Moreover, we are using MPIR library (project dll_mpir_core2) which we obtained from [http://mpir.org/](http://mpir.org/).

We have used pacman to prepare the environment:

```
MSYS shell> update-core
(restart MSYS shell)
MSYS shell> pacman -S mingw-w64-x86_64-gcc
MSYS shell> pacman -S mingw-w64-x86_64-gcc
MSYS shell> pacman -S make
MSYS shell> pacman -S bison
MSYS shell> pacman -S gdb
MSYS shell> pacman -S nano
MSYS shell> pacman -S subversion
```
Alternatively, you can use Visual Studio for building. A prepared solution consisting of many projects is ./VS/Biddy.sln . You need to adapt include and lib folders.

To produce nice setup files, projects based on Advanced Installer (http://www.advancedinstaller.-com/) are integrated into Visual Studio solution. We have been granted a free licence. MANY THANKS!

Creating Biddy library as a zip package

biddy> ./package-bin

You need a zip program.

On MS Windows, you need 7-Zip (http://7-zip.org/) - and it has a strange use of -x! You also need file 7zsd_All_x64.sfx that you should download as part of "7z SFX Tools" from http://7zsfx.info/en/ and put in the directory containing 7z.exe.

You install the resulting package by extracting libraries to the appropriate directory (may be local, e.g. user's home directory).

When using this package on GNU/Linux, you have to tell bash about the library:

export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/absolute/path/to/biddy/library

Create create zip file with source code of complete Biddy project

biddy> ./package-source

If available, source code of bddview and BDD Scout will be included, too.

Creating packages for GNU/Linux

biddy> ./package-tgz
biddy> ./package-deb
biddy> ./package-rpm

These scripts needs release number as an argument. Script package-tgz must be invoked before running package-deb. Debian packages must be created before RPM packages.

./package-tgz should create orig.tar.gz file and prepare directories for creating debian and RPM packages. You can run ./package-tgz only if version changes.

./package-deb should create debian packages (libbiddy and libbiddy-dev). They are tested on Ubuntu system.

./package-rpm should create RPM packages (libbiddy and libbiddy-devel). They are tested on openSUSE system.

Creating demo application bddscout

You need complete sources for biddy, bddview, and bddscout. Scripts are located in biddy/bddscout.

bddscout> ./package-bin
bddscout> ./package-tgz
bddscout> ./package-deb
bddscout> ./package-rpm

package-bin should create BDD Scout (statically linked with Biddy library). The script will produce a zip file. You install BDD Scout by simply unzip to the target directory.

./package-tgz should create orig.tar.gz file and prepare directories for creating debian and RPM packages. You can run ./package-tgz only if version changes.

./package-deb should create debian packages (bddscout, bddscout-bra, bddscout-ifip, bddscout-bddtrace, bddscout-ifip-data, and bddscout-bddtrace-data). They are tested on Ubuntu system.

./package-rpm should create RPM packages (bddscout, bddscout-bra, bddscout-ifip, bddscout-bddtrace, bddscout-ifip-data, and bddscout-bddtrace-data). They are tested on openSUSE system.
5. HISTORY

Biddy is based on a BDD package written in Pascal in 1992 as a student project. At that time, it was a great work and the paper about it won a second place at IEEE Region Student Paper Contest (Paris-Evry, 1993). The paper was published by IEEE as "A. Casar, R. Meolic. Representation of Boolean functions with ROBDDs. IEEE Student paper contest : regional contest winners 1990-1997 : prize-winning papers demonstrating student excellence worldwide, Piscataway, 2000" and can be obtained from http://research.meolic.com/papers/robdd.pdf

In 1995, this BDD package was rewritten in C. Later, this BDD package became an integral part of EST package, a prototype tool for formal verification of concurrent systems. The homepage for EST project is http://est.meolic.com/


In 2006, BDD package in EST got the name Biddy.

In 2007, a main part of Biddy package was extracted from EST forming a separate package called Biddy. The code has been reorganized in such a way, that EST is not using internal structures (e.g. node table) directly but using the provided API only.

In 2007, we created local svn repository for maintaining the source code (not live, anymore).

On May 15, 2008, Biddy v1.0 was released containing also bddview v0.95 (Tcl/Tk BDD viewer) and Bdd Scout v0.90 (demo application).

In 2009, 2010, and 2012 an updated version of Biddy v1.0 was released which added support for debian packaging, support for RPM packaging, fix error in Biddy_E and Biddy_A, and fix and improve details of documentation, packaging, and Tcl/Tk GUI.


In 2013, Biddy v1.1 was released. Biddy_Edge became a pointer instead of structure and all other structures were optimized.

In 2014, Biddy v1.2 was released. Variable swapping and sifting algorithm were the most significant additions.

In 2014, svn repositories for biddy, bddview and bddscout are moved to Savannah. http://svn.savannah.nongnu.org/viewvc/?root=biddy

In 2015, Biddy v1.3, v.1.4 and v1.5 was released. Various input/output methods have been added. Support for 64-bit architectures and support for Visual Studio projects were improved. Biddy got a manager. Many CUDD-like functions have been added. Comment's style changed to support doxygen. HTML and PDF documentation were produced.

Also in 2015, Personal Package Archive ppa:meolic/biddy has been created https://launchpad.net/~meolic/archive/ubuntu/biddy

Also in 2015, sources became available on GitHub https://github.com/meolic/biddy

6. PUBLICATIONS

If you find our work useful, please, cite us.


- Robert Meolic. **Biddy: An academic BDD package.** We are preparing a paper for SCP.
Chapter 2

File Index

2.1 File List

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

- **biddy-example-8queens.c**
  - File contains declaration of all external data structures
  - Page 13

- **biddy.h**
  - File contains declaration of all external data structures
  - Page 13

- **biddyInOut.c**
  - File contains various parsers and generators
  - Page 28

- **biddyInt.h**
  - File contains declaration of internal data structures
  - Page 31

- **biddyMain.c**
  - File contains main functions for representation and manipulation of boolean functions with ROBDDs
  - Page 31

- **biddyStat.c**
  - File contains statistical functions
  - Page 60
Chapter 3

File Documentation

3.1  biddy.h File Reference

File biddy.h contains declaration of all external data structures.

```c
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <string.h>
#include <ctype.h>
#include <stdarg.h>
#include <gmp.h>
```

Macros

- `#define Biddy_IsNull(f) (f == NULL)
- `#define Biddy_IsConstant(f) ((((void **)((uintptr_t) f & (!(uintptr_t) 1)))[2] == NULL) & ((void **)((uintptr_t) f & (!(uintptr_t) 1)))[3] == NULL)
- `#define Biddy_IsEqvPointer(f, g) (((uintptr_t) f & (!(uintptr_t) 1)) == ((uintptr_t) g & (!(uintptr_t) 1)))
- `#define Biddy_GetMark(f) (((uintptr_t) f & (uintptr_t) 1) != 0)
- `#define Biddy_SetMark(f) (f = (void *) ((uintptr_t) f | (uintptr_t) 1))
- `#define Biddy_ClearMark(f) (f = (void *) ((uintptr_t) f & (!(uintptr_t) 1)))
- `#define Biddy_InvertMark(f) (f = (void *) ((uintptr_t) f ^ (uintptr_t) 1))
- `#define Biddy_Not(f) ((void *) ((uintptr_t) f ^ (uintptr_t) 1))
- `#define Biddy_NotCond(f, c) (c ? ((void *) ((uintptr_t) f ^ (uintptr_t) 1)) : f)
- `#define Biddy-Regular(f) ((void **) ((uintptr_t) f & (!(uintptr_t) 1)))
- `#define Biddy_Complement(f) ((void **) ((uintptr_t) f | (uintptr_t) 1))
- `#define Biddy_String2Variable(MNG, x) (Biddy_GetTopVariable(Biddy_Managed_FoaVariable(MNG,(Biddy_String)x)))
- `#define Biddy_GetConstantZero() Biddy_Managed_GetConstantZero(NULL)
- `#define Biddy_GetConstantOne() Biddy_Managed_GetConstantOne(NULL)
- `#define Biddy_GetVariableName(v) Biddy_Managed_GetVariableName(NULL,v)
- `#define Biddy_GetTopVariableName(f) Biddy_Managed_GetTopVariableName(NULL,f)
- `#define Biddy_GetTopVariableChar(f) Biddy_Managed_GetTopVariableChar(NULL,f)
- `#define Biddy_ResetVariablesValue() Biddy_Managed_ResetVariablesValue(NULL)
- `#define Biddy_SetVariableValue(v, f) Biddy_Managed_SetVariableValue(NULL,v,f)
- `#define Biddy_IsSmaller(fv, gv) Biddy_Managed_IsSmaller(NULL,fv,gv)
- `#define Biddy_AddVariable() Biddy_Managed_AddVariable(NULL)
- `#define Biddy_AddVariableBelow(v) Biddy_Managed_AddVariableBelow(NULL,v)
• #define Biddy_AddVariableAbove(v) Biddy_Managed_AddVariableAbove(NULL,v)
• #define Biddy_FoaVariable(x) Biddy_Managed_FoaVariable(NULL,x)
• #define Biddy_FoaNode(v, pf, pt, garbageAllowed) Biddy_Managed_FoaNode(NULL,v, pf, pt, garbageAllowed)
• #define Biddy_ITE(f, g, h) Biddy_Managed_ITE(NULL,f, g, h)
• #define Biddy_And(f, g) Biddy_Managed_And(NULL,f, g)
• #define Biddy_Or(f, g) Biddy_Managed_Or(NULL,f, g)
• #define Biddy_Nand(f, g) Biddy_Managed_Nand(NULL,f, g)
• #define Biddy_Nor(f, g) Biddy_Managed_Nor(NULL,f, g)
• #define Biddy_Xor(f, g) Biddy_Managed_Xor(NULL,f, g)
• #define Biddy_Xnor(f, g) Biddy_Managed_Xnor(NULL,f, g)
• #define Biddy_Leq(f, g) Biddy_Managed_Leq(NULL,f, g)
• #define Biddy_Intersect(f, g) Biddy_Managed_Intersect(NULL,f, g)
• #define Biddy_Restrict(f, v, value) Biddy_Managed_Restrict(NULL,f, v, value)
• #define Biddy_Compose(f, v, g) Biddy_Managed_Compose(NULL,f, v, g)
• #define Biddy_E(f, v) Biddy_Managed_E(NULL,f, v)
• #define Biddy_A(f, v) Biddy_Managed_A(NULL,f, v)
• #define Biddy_IsVariableDependent(f, v) Biddy_Managed_IsVariableDependent(NULL,f, v)
• #define Biddy_ExistAbstract(f, cube) Biddy_Managed_ExistAbstract(NULL,f, cube)
• #define Biddy_UnivAbstract(f, cube) Biddy_Managed_UnivAbstract(NULL,f, cube)
• #define Biddy_AndAbstract(f, g, cube) Biddy_Managed_AndAbstract(NULL,f, g, cube)
• #define Biddy_Constrain(f, c) Biddy_Managed_Constrain(NULL,f, c)
• #define Biddy_Simplify(f, c) Biddy_Managed_Simplify(NULL,f, c)
• #define Biddy_Support(f) Biddy_Managed_Support(NULL,f)
• #define Biddy_Replace(f) Biddy_Managed_Replace(NULL,f)
• #define Biddy_IsOK(f) Biddy_Managed_IsOK(NULL,f)
• #define Biddy_Garbage() Biddy_Managed_Garbage(NULL)
• #define Biddy_Clear() Biddy_Managed_Clear(NULL)
• #define Biddy_Purge() Biddy_Managed_Purge(NULL)
• #define Biddy_PurgeAndReorder(f) Biddy_Managed_PurgeAndReorder(NULL,f)
• #define Biddy_Refresh(f) Biddy_Managed_Refresh(NULL,f)
• #define Biddy_ClearAll() Biddy_Managed_ClearAll(NULL)
• #define Biddy_AddCache(gc) Biddy_Managed_AddCache(NULL, gc)
• #define Biddy_AddFormula(x, f, count) Biddy_Managed_AddFormula(NULL, x, f, count)
• #define Biddy_DeleteFormula(x, f) Biddy_Managed_DeleteFormula(NULL, x, f)
• #define Biddy_DeleteFormula(x) Biddy_Managed_DeleteFormula(NULL, x)
• #define Biddy_GetthFormula(i) Biddy_Managed_GetthFormula(NULL, i)
• #define Biddy_GetthFormulaName(i) Biddy_Managed_GetthFormulaName(NULL, i)
• #define Biddy_SwapWithHigher(v) Biddy_Managed_SwapWithHigher(NULL, v)
• #define Biddy_SwapWithLower(v) Biddy_Managed_SwapWithLower(NULL, v)
• #define Biddy_Sifting(f) Biddy_Managed_Sifting(NULL, f)
• #define Biddy_VariableTableNum() Biddy_Managed_VariableTableNum(NULL)
• #define Biddy_NodeTableSize() Biddy_Managed_NodeTableSize(NULL)
• #define Biddy_NodeTableMax() Biddy_Managed_NodeTableMax(NULL)
• #define Biddy_NodeTableNum() Biddy_Managed_NodeTableNum(NULL)
• #define Biddy_NodeTableNumVar(v) Biddy_Managed_NodeTableNumVar(NULL, v)
• #define Biddy_NodeTableNumF() Biddy_Managed_NodeTableNumF(NULL)
• #define Biddy_NodeTableFOA() Biddy_Managed_NodeTableFOA(NULL)
• #define Biddy_NodeTableCompare() Biddy_Managed_NodeTableCompare(NULL)
• #define Biddy_NodeTableAdd() Biddy_Managed_NodeTableAdd(NULL)
• #define Biddy_NodeTableGarbage() Biddy_Managed_NodeTableGarbage(NULL)
• #define Biddy_NodeTableGenerated() Biddy_Managed_NodeTableGenerated(NULL)
• #define Biddy_NodeTableBlockNumber() Biddy_Managed_NodeTableBlockNumber(NULL)
• #define Biddy_NodeTableSwapNumber() Biddy_Managed_NodeTableSwapNumber(NULL)
• #define Biddy_NodeTableSiftingNumber() Biddy_Managed_NodeTableSiftingNumber(NULL)
3.1 biddy.h File Reference

- `#define Biddy_ListUsed() Biddy_Managed_ListUsed(NULL)`
- `#define Biddy_ListMaxLength() Biddy_Managed_ListMaxLength(NULL)`
- `#define Biddy_ListAvgLength() Biddy_Managed_ListAvgLength(NULL)`
- `#define Biddy_IteCacheSearch() Biddy_Managed_IteCacheSearch(NULL)`
- `#define Biddy_IteCacheFind() Biddy_Managed_IteCacheFind(NULL)`
- `#define Biddy_IteCacheOverwrite() Biddy_Managed_IteCacheOverwrite(NULL)`
- `#define Biddy_NodeNumberPlain(f) Biddy_Managed_NodeNumberPlain(NULL,f)`
- `#define Biddy_VariableNumber(f) Biddy_Managed_VariableNumber(NULL,f)`
- `#define Biddy_NodeVarNumber(f, n, v) Biddy_Managed_NodeVarNumber(NULL,f,n,v)`
- `#define Biddy_CountMinterm(f, nvars) Biddy_Managed_CountMinterm(NULL,f,nvars)`
- `#define Biddy_ReadMemoryInUse() Biddy_Managed_ReadMemoryInUse(NULL)`
- `#define Biddy_PrintInfo(f) Biddy_Managed_PrintInfo(NULL,f)`
- `#define Biddy_Eval0(s) Biddy_Managed_Eval0(NULL,s)`
- `#define Biddy_Eval1(s) Biddy_Managed_Eval1(NULL,s)`
- `#define Biddy_Eval1x(s, f) Biddy_Managed_Eval1x(NULL,s,f)`
- `#define Biddy_Eval2(boolFunc) Biddy_Managed_Eval2(NULL,boolFunc)`
- `#define Biddy_WriteBDD(f) Biddy_Managed_WriteBDD(NULL,f)`
- `#define Biddy_WriteBDDx(filename, f, label) Biddy_Managed_WriteBDDx(NULL,filename,f,label)`
- `#define Biddy_WriteDot(filename, f, label) Biddy_Managed_WriteDot(NULL,filename,f,label)`
- `#define Biddy_WriteDotx(filename, f, label, id) Biddy_Managed_WriteDotx(NULL,filename,f,label,id)`
- `#define Biddy_WriteTable(f) Biddy_Managed_WriteTable(NULL,f)`

**Typedefs**

- `typedef char Biddy_Boolean`
- `typedef char * Biddy_String`
- `typedef void ** Biddy_Manager`
- `typedef void * Biddy_Cache`
- `typedef unsigned int Biddy_Variable`
- `typedef void (Biddy_GCFunction)(Biddy_Manager)`
- `typedef Biddy_Boolean (Biddy_LookupFunction)(Biddy_String, Biddy_Edge *)`

### 3.1.1 Detailed Description

File `biddy.h` contains declaration of all external data structures.

**Description**

**PackageName** [Biddy]

**Synopsis** [Biddy provides data structures and algorithms for the representation and manipulation of Boolean functions with ROBDDs. A hash table is used for quick search of nodes. Complement edges decreases the number of nodes. An automatic garbage collection with a formulae counter is implemented. Variable swapping and sifting are implemented.]

**FileName** [biddy.h]

**Revision** [Revision: 124]

**Date** [Date: 2015-12-30 17:27:05 +0100 (sre, 30 dec 2015)]

**Authors** [Robert Meolic (robert.meolic@um.si), Ales Casar (ales@homemade.net)]

---

Generated on Sat Jan 2 2016 11:00:42 for Biddy by Doxygen
Copyright

Copyright (C) 2006, 2015 UM-FERI, Smetanova ulica 17, SI-2000 Maribor, Slovenia

Biddy is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

Biddy is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied
warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public
License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the
Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

More info

See also: biddyInt.h

Definition in file biddy.h.

3.1.2 Macro Definition Documentation

3.1.2.1 #define Biddy_IsNull( f ) (f == NULL)

Biddy_IsNull returns TRUE iff given BDD is a null edge

Definition at line 114 of file biddy.h.

3.1.2.2 #define Biddy_IsConstant( f ) (((void *)((uintptr_t) f & ~((uintptr_t) 1)))[2] == NULL) && (((void *)((uintptr_t) f & ~((uintptr_t) 1)))[3] == NULL)

Biddy_IsConstant returns TRUE iff given BDD is a constant 0 or 1

Definition at line 119 of file biddy.h.

3.1.2.3 #define Biddy_IsEqvPointer( f, g ) (((uintptr_t) f & ~((uintptr_t) 1)) == ((uintptr_t) g & ~((uintptr_t) 1)))

Biddy_IsEqvPointer returns TRUE iff given BDDs are equal or inverted.

Definition at line 122 of file biddy.h.

3.1.2.4 #define Biddy_GetMark( f ) (((uintptr_t) f & (uintptr_t) 1) != 0)

Biddy_GetMark returns TRUE iff given edge is complemented.

Definition at line 125 of file biddy.h.

3.1.2.5 #define Biddy_SetMark( f ) (f = (void *)((uintptr_t) f | (uintptr_t) 1))

Biddy_SetMark makes given edge complemented.

Definition at line 128 of file biddy.h.

3.1.2.6 #define Biddy_ClearMark( f ) (f = (void *)((uintptr_t) f & ~((uintptr_t) 1)))

Biddy_ClearMark makes given edge not-complemented.

Definition at line 131 of file biddy.h.
3.1.2.7  #define Biddy_InvertMark( f ) (f = (void *) ((uintptr_t) f ^ (uintptr_t) 1))

Biddy_InvertMark changes complement bit of the given edge.
Definition at line 134 of file biddy.h.

3.1.2.8  #define Biddy_Not( f ) ((void *) ((uintptr_t) f ^ (uintptr_t) 1))

Biddy_Not returns edge with changed complemented bit, since Biddy v1.4.
Definition at line 137 of file biddy.h.

3.1.2.9  #define Biddy_NotCond( f, c ) (c ? ((void *) ((uintptr_t) f ^ (uintptr_t) 1)) : f)

Biddy_NotCond returns edge with conditionally changed complemented bit, since Biddy v1.4.
Definition at line 140 of file biddy.h.

3.1.2.10 #define Biddy_Regular( f ) ((void *) ((uintptr_t) f & ~((uintptr_t) 1)))

Biddy_Regular returns not-complemented version of edge, since Biddy v1.4.
Definition at line 144 of file biddy.h.

3.1.2.11 #define Biddy_Complement( f ) ((void *) ((uintptr_t) f | (uintptr_t) 1))

Biddy_Complement returns complemented version of edge, since Biddy v1.4.
Definition at line 147 of file biddy.h.

3.1.2.12 #define Biddy_String2Variable( MNG, x ) (Biddy_GetTopVariable(Biddy_Managed_FoaVariable(MNG,(Biddy_String)x)))

Biddy_String2Variable finds or adds variable with a given name, since Biddy v1.4.
Definition at line 150 of file biddy.h.

3.1.2.13 #define Biddy_GetConstantZero( ) Biddy_Managed_GetConstantZero(NULL)

Macro Biddy_GetConstantZero is defined for use with anonymous manager.
Definition at line 258 of file biddy.h.

3.1.2.14 #define Biddy_GetConstantOne( ) Biddy_Managed_GetConstantOne(NULL)

Macro Biddy_GetConstantOne is defined for use with anonymous manager.
Definition at line 263 of file biddy.h.

3.1.2.15 #define Biddy_GetVariableName( v ) Biddy_Managed_GetVariableName(NULL,v)

Macro Biddy_GetVariableName is defined for use with anonymous manager.
Definition at line 268 of file biddy.h.
3.1.2.16  
#define Biddy_GetTopVariableName(  f ) Biddy_Managed_GetTopVariableName(NULL,f)

Macro Biddy_GetTopVariableName is defined for use with anonymous manager.
Definition at line 273 of file biddy.h.

3.1.2.17  
#define Biddy_GetTopVariableChar(  f ) Biddy_Managed_GetTopVariableChar(NULL,f)

Macro Biddy_GetTopVariableChar is defined for use with anonymous manager.
Definition at line 278 of file biddy.h.

3.1.2.18  
#define Biddy_ResetVariablesValue( ) Biddy_Managed_ResetVariablesValue(NULL)

Macro Biddy_ResetVariablesValue is defined for use with anonymous manager.
Definition at line 283 of file biddy.h.

3.1.2.19  
#define Biddy_SetVariableValue( v, f ) Biddy_Managed_SetVariableValue(NULL,v,f)

Macro Biddy_SetVariableValue is defined for use with anonymous manager.
Definition at line 288 of file biddy.h.

3.1.2.20  
#define Biddy_IsSmaller( fv, gv ) Biddy_Managed_IsSmaller(NULL,fv,gv)

Macro Biddy_IsSmaller is defined for use with anonymous manager.
Definition at line 293 of file biddy.h.

3.1.2.21  
#define Biddy_AddVariable( ) Biddy_Managed_AddVariable(NULL)

Macro Biddy_AddVariable is defined for use with anonymous manager.
Definition at line 298 of file biddy.h.

3.1.2.22  
#define Biddy_AddVariableBelow( v ) Biddy_Managed_AddVariableBelow(NULL,v)

Macro Biddy_AddVariableBelow is defined for use with anonymous manager.
Definition at line 303 of file biddy.h.

3.1.2.23  
#define Biddy_AddVariableAbove( v ) Biddy_Managed_AddVariableAbove(NULL,v)

Macro Biddy_AddVariableAbove is defined for use with anonymous manager.
Definition at line 308 of file biddy.h.

3.1.2.24  
#define Biddy_FoaVariable( x ) Biddy_Managed_FoaVariable(NULL,x)

Macro Biddy_FoaVariable is defined for use with anonymous manager.
Definition at line 313 of file biddy.h.
3.1.2.25 \texttt{#define Biddy\_FoaNode( v, pf, pt, garbageAllowed ) Biddy\_Managed\_FoaNode(NULL,v,pf,pt,garbageAllowed)}

Macro \texttt{Biddy\_FoaNode} is defined for use with anonymous manager.  
Definition at line 318 of file biddy.h.

3.1.2.26 \texttt{#define Biddy\_ITE( f, g, h ) Biddy\_Managed\_ITE(NULL,f,g,h)}

Macro \texttt{Biddy\_ITE} is defined for use with anonymous manager.  
Definition at line 323 of file biddy.h.

3.1.2.27 \texttt{#define Biddy\_And( f, g ) Biddy\_Managed\_And(NULL,f,g)}

Macro \texttt{Biddy\_And} is defined for use with anonymous manager.  
Definition at line 328 of file biddy.h.

3.1.2.28 \texttt{#define Biddy\_Or( f, g ) Biddy\_Managed\_Or(NULL,f,g)}

Macro \texttt{Biddy\_Or} is defined for use with anonymous manager.  
Definition at line 333 of file biddy.h.

3.1.2.29 \texttt{#define Biddy\_Nand( f, g ) Biddy\_Managed\_Nand(NULL,f,g)}

Macro \texttt{Biddy\_Nand} is defined for use with anonymous manager.  
Definition at line 338 of file biddy.h.

3.1.2.30 \texttt{#define Biddy\_Nor( f, g ) Biddy\_Managed\_Nor(NULL,f,g)}

Macro \texttt{Biddy\_Nor} is defined for use with anonymous manager.  
Definition at line 343 of file biddy.h.

3.1.2.31 \texttt{#define Biddy\_Xor( f, g ) Biddy\_Managed\_Xor(NULL,f,g)}

Macro \texttt{Biddy\_Xor} is defined for use with anonymous manager.  
Definition at line 348 of file biddy.h.

3.1.2.32 \texttt{#define Biddy\_Xnor( f, g ) Biddy\_Managed\_Xnor(NULL,f,g)}

Macro \texttt{Biddy\_Xnor} is defined for use with anonymous manager.  
Definition at line 353 of file biddy.h.

3.1.2.33 \texttt{#define Biddy\_Leq( f, g ) Biddy\_Managed\_Leq(NULL,f,g)}

Macro \texttt{Biddy\_Leq} is defined for use with anonymous manager.  
Definition at line 358 of file biddy.h.
3.1.2.34 #define Biddy_Intersect( f, g ) Biddy_Managed_Intersect(NULL,f,g)

Macro Biddy_Intersect is defined for use with anonymous manager.
Definition at line 363 of file biddy.h.

3.1.2.35 #define Biddy_Restrict( f, v, value ) Biddy_Managed_Restrict(NULL,f,v,value)

Macro Biddy_Restrict is defined for use with anonymous manager.
Definition at line 368 of file biddy.h.

3.1.2.36 #define Biddy_Compose( f, v, g ) Biddy_Managed_Compose(NULL,f,v,g)

Macro Biddy_Compose is defined for use with anonymous manager.
Definition at line 373 of file biddy.h.

3.1.2.37 #define Biddy_E( f, v ) Biddy_Managed_E(NULL,f,v)

Macro Biddy_E is defined for use with anonymous manager.
Definition at line 378 of file biddy.h.

3.1.2.38 #define Biddy_A( f, v ) Biddy_Managed_A(NULL,f,v)

Macro Biddy_A is defined for use with anonymous manager.
Definition at line 383 of file biddy.h.

3.1.2.39 #define Biddy_IsVariableDependent( f, v ) Biddy_Managed_IsVariableDependent(NULL,f,v)

Macro Biddy_IsVariableDependent is defined for use with anonymous manager.
Definition at line 388 of file biddy.h.

3.1.2.40 #define Biddy_ExistAbstract( f, cube ) Biddy_Managed_ExistAbstract(NULL,f,cube)

Macro Biddy_ExistAbstract is defined for use with anonymous manager.
Definition at line 393 of file biddy.h.

3.1.2.41 #define Biddy_UnivAbstract( f, cube ) Biddy_Managed_UnivAbstract(NULL,f,cube)

Macro Biddy_UnivAbstract is defined for use with anonymous manager.
Definition at line 398 of file biddy.h.

3.1.2.42 #define Biddy_AndAbstract( f, g, cube ) Biddy_Managed_AndAbstract(NULL,f,g,cube)

Macro Biddy_AndAbstract is defined for use with anonymous manager.
Definition at line 403 of file biddy.h.
#define Biddy_Constrain(f, c) Biddy_Managed_Constrain(NULL,f,c)

Macro Biddy_Constrain is defined for use with anonymous manager.
Definition at line 408 of file biddy.h.

#define Biddy_Simplify(f, c) Biddy_Managed_Simplify(NULL,f,c)

Macro Biddy_Simplify is defined for use with anonymous manager.
Definition at line 413 of file biddy.h.

#define Biddy_Support(f) Biddy_Managed_Support(NULL,f)

Macro Biddy_Support is defined for use with anonymous manager.
Definition at line 418 of file biddy.h.

#define Biddy_Replace(f) Biddy_Managed_Replace(NULL,f)

Macro Biddy_Replace is defined for use with anonymous manager.
Definition at line 423 of file biddy.h.

#define Biddy_IsOK(f) Biddy_Managed_IsOK(NULL,f)

Macro Biddy_IsOK is defined for use with anonymous manager.
Definition at line 428 of file biddy.h.

#define Biddy_Garbage() Biddy_Managed_Garbage(NULL)

Macro Biddy_Garbage is defined for use with anonymous manager.
Definition at line 433 of file biddy.h.

#define Biddy_Clean() Biddy_Managed_Clean(NULL)

Macro Biddy_Clean is defined for use with anonymous manager.
Definition at line 438 of file biddy.h.

#define Biddy_Purge() Biddy_Managed_Purge(NULL)

Macro Biddy_Purge is defined for use with anonymous manager.
Definition at line 443 of file biddy.h.

#define Biddy_PurgeAndReorder(f) Biddy_Managed_PurgeAndReorder(NULL,f)

Macro Biddy_PurgeAndReorder is defined for use with anonymous manager.
Definition at line 448 of file biddy.h.
3.1.2.52  
#define Biddy_Refresh( f ) Biddy_Managed_Refresh(NULL,f)

Macro Biddy_Refresh is defined for use with anonymous manager.
Definition at line 453 of file biddy.h.

3.1.2.53  
#define Biddy_ClearAll( ) Biddy_Managed_ClearAll(NULL)

Macro Biddy_ClearAll is defined for use with anonymous manager.
Definition at line 458 of file biddy.h.

3.1.2.54  
#define Biddy_AddCache( gc ) Biddy_Managed_AddCache(NULL,gc)

Macro Biddy_AddCache is defined for use with anonymous manager.
Definition at line 463 of file biddy.h.

3.1.2.55  
#define Biddy_AddFormula( x, f, count ) Biddy_Managed_AddFormula(NULL,x,f,count)

Macro Biddy_AddFormula is defined for use with anonymous manager.
Definition at line 468 of file biddy.h.

3.1.2.56  
#define Biddy_FindFormula( x, f ) Biddy_Managed_FindFormula(NULL,x,f)

Macro Biddy_FindFormula is defined for use with anonymous manager.
Definition at line 477 of file biddy.h.

3.1.2.57  
#define Biddy_DeleteFormula( x ) Biddy_Managed_DeleteFormula(NULL,x)

Macro Biddy_DeleteFormula is defined for use with anonymous manager.
Definition at line 482 of file biddy.h.

3.1.2.58  
#define Biddy_DeleteIthFormula( x ) Biddy_Managed_DeleteIthFormula(NULL,x)

Macro Biddy_DeleteIthFormula is defined for use with anonymous manager.
Definition at line 487 of file biddy.h.

3.1.2.59  
#define Biddy_GetIthFormula( i ) Biddy_Managed_GetIthFormula(NULL,i)

Macro Biddy_GetIthFormula is defined for use with anonymous manager.
Definition at line 492 of file biddy.h.

3.1.2.60  
#define Biddy_GetIthFormulaName( i ) Biddy_Managed_GetIthFormulaName(NULL,i)

Macro Biddy_GetIthFormulaName is defined for use with anonymous manager.
Definition at line 497 of file biddy.h.
3.1.2.61  
#define Biddy_SwapWithHigher( v ) Biddy_Managed_SwapWithHigher(NULL,v)

Macro Biddy_SwapWithHigher is defined for use with anonymous manager.
Definition at line 502 of file biddy.h.

3.1.2.62  
#define Biddy_SwapWithLower( v ) Biddy_Managed_SwapWithLower(NULL,v)

Macro Biddy_SwapWithLower is defined for use with anonymous manager.
Definition at line 507 of file biddy.h.

3.1.2.63  
#define Biddy_Sifting( f ) Biddy_Managed_Sifting(NULL,f)

Macro Biddy_Sifting is defined for use with anonymous manager.
Definition at line 512 of file biddy.h.

3.1.2.64  
#define Biddy_VariableTableNum( ) Biddy_Managed_VariableTableNum(NULL)

Macro Biddy_VariableTableNum is defined for use with anonymous manager.
Definition at line 542 of file biddy.h.

3.1.2.65  
#define Biddy_NodeTableSize( ) Biddy_Managed_NodeTableSize(NULL)

Macro Biddy_NodeTableSize is defined for use with anonymous manager.
Definition at line 547 of file biddy.h.

3.1.2.66  
#define Biddy_NodeTableMax( ) Biddy_Managed_NodeTableMax(NULL)

Macro Biddy_NodeTableMax is defined for use with anonymous manager.
Definition at line 552 of file biddy.h.

3.1.2.67  
#define Biddy_NodeTableNum( ) Biddy_Managed_NodeTableNum(NULL)

Macro Biddy_NodeTableNum is defined for use with anonymous manager.
Definition at line 557 of file biddy.h.

3.1.2.68  
#define Biddy_NodeTableNumVar( v ) Biddy_Managed_NodeTableNumVar(NULL,v)

Macro Biddy_NodeTableNumVar is defined for use with anonymous manager.
Definition at line 562 of file biddy.h.

3.1.2.69  
#define Biddy_NodeTableNumF( ) Biddy_Managed_NodeTableNumF(NULL)

Macro Biddy_NodeTableNumF is defined for use with anonymous manager.
Definition at line 567 of file biddy.h.
3.1.2.70  \#define Biddy_NodeTableFOA( ) Biddy_Managed_NodeTableFOA(NULL)

Macro Biddy_NodeTableFOA is defined for use with anonymous manager.  
Definition at line 572 of file biddy.h.

3.1.2.71  \#define Biddy_NodeTableCompare( ) Biddy_Managed_NodeTableCompare(NULL)

Macro Biddy_NodeTableCompare is defined for use with anonymous manager.  
Definition at line 577 of file biddy.h.

3.1.2.72  \#define Biddy_NodeTableAdd( ) Biddy_Managed_NodeTableAdd(NULL)

Macro Biddy_NodeTableAdd is defined for use with anonymous manager.  
Definition at line 582 of file biddy.h.

3.1.2.73  \#define Biddy_NodeTableGarbage( ) Biddy_Managed_NodeTableGarbage(NULL)

Macro Biddy_NodeTableGarbage is defined for use with anonymous manager.  
Definition at line 587 of file biddy.h.

3.1.2.74  \#define Biddy_NodeTableGenerated( ) Biddy_Managed_NodeTableGenerated(NULL)

Macro Biddy_NodeTableGenerated is defined for use with anonymous manager.  
Definition at line 592 of file biddy.h.

3.1.2.75  \#define Biddy_NodeTableBlockNumber( ) Biddy_Managed_NodeTableBlockNumber(NULL)

Macro Biddy_NodeTableBlockNumber is defined for use with anonymous manager.  
Definition at line 597 of file biddy.h.

3.1.2.76  \#define Biddy_NodeTableSwapNumber( ) Biddy_Managed_NodeTableSwapNumber(NULL)

Macro Biddy_NodeTableSwapNumber is defined for use with anonymous manager.  
Definition at line 602 of file biddy.h.

3.1.2.77  \#define Biddy_NodeTableSiftingNumber( ) Biddy_Managed_NodeTableSiftingNumber(NULL)

Macro Biddy_NodeTableSiftingNumber is defined for use with anonymous manager.  
Definition at line 607 of file biddy.h.

3.1.2.78  \#define Biddy_ListUsed( ) Biddy_Managed_ListUsed(NULL)

Macro Biddy_ListUsed is defined for use with anonymous manager.  
Definition at line 612 of file biddy.h.
3.1.2.79  

#define Biddy_ListMaxLength() Biddy_Managed_ListMaxLength(NULL)

Macro Biddy_ListMaxLength is defined for use with anonymous manager. 
Definition at line 617 of file biddy.h.

3.1.2.80  

#define Biddy_ListAvgLength() Biddy_Managed_ListAvgLength(NULL)

Macro Biddy_ListAvgLength is defined for use with anonymous manager. 
Definition at line 622 of file biddy.h.

3.1.2.81  

#define Biddy_IteCacheSearch() Biddy_Managed_IteCacheSearch(NULL)

Macro Biddy_IteCacheSearch is defined for use with anonymous manager. 
Definition at line 627 of file biddy.h.

3.1.2.82  

#define Biddy_IteCacheFind() Biddy_Managed_IteCacheFind(NULL)

Macro Biddy_IteCacheFind is defined for use with anonymous manager. 
Definition at line 632 of file biddy.h.

3.1.2.83  

#define Biddy_IteCacheOverwrite() Biddy_Managed_IteCacheOverwrite(NULL)

Macro Biddy_IteCacheOverwrite is defined for use with anonymous manager. 
Definition at line 637 of file biddy.h.

3.1.2.84  

#define Biddy_NodeNumberPlain( f ) Biddy_Managed_NodeNumberPlain(NULL,f)

Macro Biddy_NodeNumberPlain is defined for use with anonymous manager. 
Definition at line 642 of file biddy.h.

3.1.2.85  

#define Biddy_VariableNumber( f ) Biddy_Managed_VariableNumber(NULL,f)

Macro Biddy_VariableNumber is defined for use with anonymous manager. 
Definition at line 647 of file biddy.h.

3.1.2.86  

#define Biddy_NodeVarNumber( f, n, v ) Biddy_Managed_NodeVarNumber(NULL,f,n,v)

Macro Biddy_NodeVarNumber is defined for use with anonymous manager. 
Definition at line 652 of file biddy.h.

3.1.2.87  

#define Biddy_CountMinterm( f, nvars ) Biddy_Managed_CountMinterm(NULL,f,nvars)

Macro Biddy_CountMinterm is defined for use with anonymous manager. 
Definition at line 657 of file biddy.h.

Generated on Sat Jan 2 2016 11:00:42 for Biddy by Doxygen
3.1.2.88  
#define Biddy_ReadMemoryInUse( ) Biddy_Managed_ReadMemoryInUse(NULL)

Macro Biddy_ReadMemoryInUse is defined for use with anonymous manager.
Definition at line 662 of file biddy.h.

3.1.2.89  
#define Biddy_PrintInfo( f ) Biddy_Managed_PrintInfo(NULL,f)

Macro Biddy_PrintInfo is defined for use with anonymous manager.
Definition at line 667 of file biddy.h.

3.1.2.90  
#define Biddy_Eval0( s ) Biddy_Managed_Eval0(NULL,s)

Macro Biddy_Eval0 is defined for use with anonymous manager.
Definition at line 684 of file biddy.h.

3.1.2.91  
#define Biddy_Eval1( s ) Biddy_Managed_Eval1(NULL,s)

Macro Biddy_Eval1 is defined for use with anonymous manager.
Definition at line 689 of file biddy.h.

3.1.2.92  
#define Biddy_Eval1x( s, f ) Biddy_Managed_Eval1x(NULL,s,f)

Macro Biddy_Eval1x is defined for use with anonymous manager.
Definition at line 694 of file biddy.h.

3.1.2.93  
#define Biddy_Eval2( boolFunc ) Biddy_Managed_Eval2(NULL,boolFunc)

Macro Biddy_Eval2 is defined for use with anonymous manager.
Definition at line 699 of file biddy.h.

3.1.2.94  
#define Biddy_WriteBDD( f ) Biddy_Managed_WriteBDD(NULL,f)

Macro Biddy_WriteBDD is defined for use with anonymous manager.
Definition at line 704 of file biddy.h.

3.1.2.95  
#define Biddy_WriteBDDx( filename, f, label ) Biddy_Managed_WriteBDDx(NULL,filename,f,label)

Macro Biddy_WriteBDDx is defined for use with anonymous manager.
Definition at line 709 of file biddy.h.

3.1.2.96  
#define Biddy_WriteDot( filename, f, label ) Biddy_Managed_WriteDot(NULL,filename,f,label)

Macro Biddy_WriteDot is defined for use with anonymous manager.
Definition at line 714 of file biddy.h.
3.1.2.97  
#define Biddy_WriteDotx( filename, f, label, id ) Biddy_Managed_WriteDotx(NULL, filename, f, label, id) 

Macro Biddy_WriteDotx is defined for use with anonymous manager.
Definition at line 719 of file biddy.h.

3.1.2.98  
#define Biddy_WriteTable( f ) Biddy_Managed_WriteTable(NULL, f) 

Macro Biddy_WriteTable is defined for use with anonymous manager.
Definition at line 724 of file biddy.h.

3.1.3  
Typedef Documentation

3.1.3.1  
typedef char Biddy_Boolean 

Biddy_Boolean is used for boolean values.
Definition at line 157 of file biddy.h.

3.1.3.2  
typedef char* Biddy_String 

Biddy_String is used for strings.
Definition at line 160 of file biddy.h.

3.1.3.3  
typedef void** Biddy_Manager 

Biddy_Manager is used to specify manager. Manager is a pointer to BiddyManager. A manager includes Node Table, Variable Table, Formulae Table, Ordering Table, three basic caches (ITE Cache, EA Cache and RC Cache), list of user's caches, formulae counter and some other structures needed for memory management. Internal structure of BiddyManager is not exported but must be imitated to create user's managers.
Definition at line 169 of file biddy.h.

3.1.3.4  
typedef void* Biddy_Cache 

Biddy_Cache is used to specify user's cache table. Caches for different operations are different and the user is responsible for the correct internal structure.
Definition at line 174 of file biddy.h.

3.1.3.5  
typedef unsigned int Biddy_Variable 

Biddy_Variable is used for indices in variable table.
Definition at line 178 of file biddy.h.

3.1.3.6  
typedef void* Biddy_Edge 

Biddy_Edge is a marked edge (i.e., a marked pointer to BiddyNode). Mark is encoded as the value of the last significant bit. Internal structure of BiddyNode is not visible to the user.
Definition at line 183 of file biddy.h.
3.1.3.7 typedef void (Biddy_GCFunction)(Biddy_Manager)

Biddy_GCFunction is used in Biddy_AddCache to specify user's function which will performs garbage collection. Definition at line 187 of file biddy.h.

3.1.3.8 typedef Biddy_Boolean (Biddy_LookupFunction)(Biddy_String, Biddy_Edge)

Biddy_LookupFunction is used in Biddy_Eval1x to specify user's function which will lookups in a user's formula table. Definition at line 191 of file biddy.h.

3.2 biddyInOut.c File Reference

File biddyInOut.c contains various parsers and generators.

#include "biddyInt.h"

Functions

- Biddy_String Biddy_Managed_Eval0 (Biddy_Manager MNG, Biddy_String s)
  
  Function Biddy_Managed_Eval0 evaluates raw format.

- Biddy_Edge Biddy_Managed_Eval1 (Biddy_Manager MNG, Biddy_String s)
  
  Function Biddy_Managed_Eval1 evaluates prefix AND-OR-EXOR-NOT format.

- Biddy_Edge Biddy_Managed_Eval2 (Biddy_Manager MNG, Biddy_String boolFunc)
  
  Function Biddy_Managed_Eval2 evaluates infix & | ^ < > format.

- void Biddy_Managed_WriteBDD (Biddy_Manager MNG, Biddy_Edge f)
  
  Function Biddy_Managed_WriteBDD writes raw format.

- unsigned int Biddy_Managed_WriteDot (Biddy_Manager MNG, const char filename[], Biddy_Edge f, const char label[])
  
  Function Biddy_Managed_WriteDot writes dot/graphviz format.

- void Biddy_Managed_WriteTable (Biddy_Manager MNG, Biddy_Edge f)
  
  Function Biddy_Managed_WriteTable writes truth table.

3.2.1 Detailed Description

File biddyInOut.c contains various parsers and generators.

Description

PackageName [Biddy]

Synopsis [Biddy provides data structures and algorithms for the representation and manipulation of Boolean functions with ROBDDs. A hash table is used for quick search of nodes. Complement edges decreases the number of nodes. An automatic garbage collection with a formulae counter is implemented. Variable swapping and sifting are implemented.]

FileName [biddyInOut.c]

Revision [$Revision: 124 $]

Date [$Date: 2015-12-30 17:27:05 +0100 (sre, 30 dec 2015) $]

Authors [Robert Meolic (robert.meolic@um.si), Alex Casar (alex@homemade.net), Volodymyr Mihav (mihav.volodymyr@gmail.com), Jan Kraner (jankristian.kraner@student.um.si), Ziga Kobale (ziga.kobale@student.um.si) ]
Copyright

Copyright (C) 2006, 2015 UM-FERI, Smetanova ulica 17, SI-2000 Maribor, Slovenia
Biddy is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.
Biddy is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied
warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public
License for more details.
You should have received a copy of the GNU General Public License along with this program; if not, write to the
Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

More info

See also: biddy.h, biddyInt.h
Definition in file biddyInOut.c.

3.2.2 Function Documentation

3.2.2.1 Biddy_String Biddy_Managed_Eval0 ( Biddy_Manager MNG, Biddy_String s )

Function Biddy_Managed_Eval0 evaluates raw format.

Description

Side effects

Not reentrant.

More info

Macro Biddy_Eval0(s) is defined for use with anonymous manager.
Definition at line 138 of file biddyInOut.c.

3.2.2.2 Biddy_Edge Biddy_Managed_Eval1 ( Biddy_Manager MNG, Biddy_String s )

Function Biddy_Managed_Eval1 evaluates prefix AND-OR-EXOR-NOT format.

Description

Side effects

Not reentrant.

More info

Macro Biddy_Eval1(s) is defined for use with anonymous manager.
Definition at line 192 of file biddyInOut.c.

3.2.2.3 Biddy_Edge Biddy_Managed_Eval2 ( Biddy_Manager MNG, Biddy_String boolFunc )

Function Biddy_Managed_Eval2 evaluates infix &|^~>< format.
Description

Boolean constants are '0' and '1'. Parenthesis are implemented. Operators' priority is implemented. Formula Tree is supported (global table, only). Operators '∗' and '+' are also allowed for conjunction/disjunction.

Side effects

Variable names must be one letter a-zA-Z optionally followed by int number.

More info

Author: Volodymyr Mihav (mihaw.wolodymyr@gmail.com) Original implementation of this function is on https://github.com/sungmaster/liBDD. Macro Biddy_Eval2(boolFunc) is defined for use with anonymous manager.

Definition at line 262 of file biddyInOut.c.

3.2.2.4 void Biddy_Managed_WriteBDD (Biddy_Manager MNG, Biddy_Edge f)

Function Biddy_Managed_WriteBDD writes raw format.

Description

Side effects

More info

Macro Biddy_WriteBDD(f) is defined for use with anonymous manager.

Definition at line 427 of file biddyInOut.c.

3.2.2.5 unsigned int Biddy_Managed_WriteDot (Biddy_Manager MNG, const char filename[], Biddy_Edge f, const char label[])

Function Biddy_Managed_WriteDot writes dot/graphviz format.

Description

Side effects

More info

Macro Biddy_WriteDot(filename,f,label) is defined for use with anonymous manager.

Definition at line 483 of file biddyInOut.c.

3.2.2.6 void Biddy_Managed_WriteTable (Biddy_Manager MNG, Biddy_Edge f)

Function Biddy_Managed_WriteTable writes truth table.

Description

Side effects

More info

Author: Jan Kraner (jankristian.kraner@student.um.si) Author: Ziga Kobale (ziga-kobale@student.um.si) Macro Biddy_WriteTable(f) is defined for use with anonymous manager.

Definition at line 609 of file biddyInOut.c.
3.3  biddyInt.h File Reference

File biddyInt.h contains declaration of internal data structures.

#include "biddy.h"

3.3.1  Detailed Description

File biddyInt.h contains declaration of internal data structures.

Description

PackageName [Biddy]
Synopsis [Biddy provides data structures and algorithms for the
representation and manipulation of Boolean functions with
ROBDDs. A hash table is used for quick search of nodes.
Complement edges decreases the number of nodes. An automatic
garbage collection with a formulae counter is implemented.
Variable swapping and sifting are implemented.]

FileName  [biddyInt.h]
Revision  [$Revision: 124 $]
Date  [$Date: 2015-12-30 17:27:05 +0100 (sre, 30 dec 2015) $]
Authors  [Robert Meolic (robert.meolic@um.si),
Ales Casar (ales@homemade.net)]

Copyright

Copyright (C) 2006, 2015 UM-FERI, Smetanova ulica 17, SI-2000 Maribor, Slovenia

Biddy is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

Biddy is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied
warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public
License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the
Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

More info

See also: biddy.h
Definition in file biddyInt.h.

3.4  biddyMain.c File Reference

File biddyMain.c contains main functions for representation and manipulation of boolean functions with ROBDDs.

#include "biddyInt.h"

Functions

- void Biddy_Init ()
  
  Function Biddy_Init initialize BDD package.

- void Biddy_Exit ()
  
  Function Biddy_Exit deletes anonymous manager.

Generated on Sat Jan 2 2016 11:00:42 for Biddy by Doxygen
• **Biddy_String Biddy_About ()**
  Function Biddy_About reports version of Biddy package.

• **Biddy_Edge Biddy_GetThen (Biddy_Edge f)**
  Function Biddy_GetThen returns THEN successor.

• **Biddy_Edge Biddy_GetElse (Biddy_Edge f)**
  Function Biddy_GetElse returns ELSE successor.

• **Biddy_Variable Biddy_GetTopVariable (Biddy_Edge f)**
  Function Biddy_GetTopVariable returns the top variable.

• **void Biddy_SelectNode (Biddy_Edge f)**
  Function Biddy_SelectNode selects the top node of the given function.

• **void Biddy_DeselectNode (Biddy_Edge f)**
  Function Biddy_DeselectNode deselects the top node of the given function.

• **Biddy_Boolean Biddy_IsSelected (Biddy_Edge f)**
  Function Biddy_IsSelected returns TRUE iff the top node of the given function is selected.

• **void Biddy_NodeSelect (Biddy_Edge f)**
  Function Biddy_NodeSelect recursively selects all nodes of a given function.

• **void Biddy_NodeRepair (Biddy_Edge f)**
  Function Biddy_NodeRepair recursively repairs (deselects) all nodes of a given function.

• **Biddy_Edge Biddy_NOT (Biddy_Edge f)**
  Function Biddy_NOT complements Boolean function.

• **Biddy_Edge Biddy_TransferMark (Biddy_Edge f, Biddy_Boolean mark)**
  Function Biddy_TransferMark complements Boolean function if given parameter mark is TRUE.

• **Biddy_Edge Biddy_Managed_GetConstantZero (Biddy_Manager MNG)**
  Function Biddy_Managed_GetConstantZero returns constant 0.

• **Biddy_Edge Biddy_Managed_GetConstantOne (Biddy_Manager MNG)**
  Function Biddy_Managed_GetConstantOne returns constant 1.

• **Biddy_String Biddy_Managed_GetVariableName (Biddy_Manager MNG, Biddy_Variable v)**
  Function Biddy_Managed_GetVariableName returns the name of a variable.

• **Biddy_String Biddy_Managed_GetTopVariableName (Biddy_Manager MNG, Biddy_Edge f)**
  Function Biddy_Managed_GetTopVariableName returns the name of top variable.

• **char Biddy_Managed_GetTopVariableChar (Biddy_Manager MNG, Biddy_Edge f)**
  Function Biddy_Managed_GetTopVariableChar returns the first character in the name of top variable.

• **void Biddy_Managed_ResetVariablesValue (Biddy_Manager MNG)**
  Function Biddy_Managed_ResetVariablesValue sets all variable’s value to biddyZero, except constant node which gets value biddyOne.

• **void Biddy_Managed_SetVariableValue (Biddy_Manager MNG, Biddy_Variable v, Biddy_Edge f)**
  Function Biddy_Managed_SetVariableValue sets variable’s value.

• **Biddy_Boolean Biddy_Managed_IsSmaller (Biddy_Manager MNG, Biddy_Variable fv, Biddy_Variable gv)**
  Function Biddy_Managed_IsSmaller returns TRUE if the first variable is smaller (= lower = topmore).

• **Biddy_Edge Biddy_Managed_AddVariable (Biddy_Manager MNG)**
  Function Biddy_Managed_AddVariable adds a numbered variable.

• **Biddy_Edge Biddy_Managed_AddVariableBelow (Biddy_Manager MNG, Biddy_Variable v)**
  Function Biddy_Managed_AddVariableBelow adds a numbered variable.

• **Biddy_Edge Biddy_Managed_AddVariableAbove (Biddy_Manager MNG, Biddy_Variable v)**
  Function Biddy_Managed_AddVariableAbove adds a numbered variable.

• **Biddy_Edge Biddy_Managed_FoaVariable (Biddy_Manager MNG, Biddy_String x)**
  Function Biddy_Managed_FoaVariable finds or adds variable edge.

• **Biddy_Edge Biddy_Managed_FoaNode (Biddy_Manager MNG, Biddy_Variable v, Biddy_Edge pf, Biddy_Edge pt, Biddy_Boolean garbageAllowed)**
  Function Biddy_Managed_FoaNode finds or adds new node with the given variable and successors.

• **Biddy_Edge Biddy_Managed_ITE (Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g, Biddy_Edge h)**
Function Biddy_Managed ITE calculates ITE operation of three Boolean functions.

- Biddy_Edge Biddy_Managed_And (Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)
  Function Biddy_Managed_And calculates Boolean function AND (conjunction).

- Biddy_Edge Biddy_Managed_Or (Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)
  Function Biddy_Managed_Or calculates Boolean function OR (disjunction).

- Biddy_Edge Biddy_Managed_Nand (Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)
  Function Biddy_Managed_Nand calculates Boolean function NAND (Sheffer).

- Biddy_Edge Biddy_Managed_Nor (Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)
  Function Biddy_Managed_Nor calculates Boolean function NOR (Peirce).

- Biddy_Edge Biddy_Managed_Xor (Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)
  Function Biddy_Managed_Xor calculates Boolean function XOR.

- Biddy_Edge Biddy_Managed_Xnor (Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)
  Function Biddy_Managed_Xnor calculates Boolean function XNOR.

- Biddy_Edge Biddy_Managed_Leq (Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)
  Function Biddy_Managed_Leq returns TRUE iff function f is included in function g.

- Biddy_Edge Biddy_Managed_Intersect (Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)
  Biddy_Managed_Intersect calculates a function included in the intersection of f and g.

- Biddy_Edge Biddy_Managed_Restrict (Biddy_Manager MNG, Biddy_Edge f, Biddy_Variable v, Biddy_Edge value)
  Function Biddy_Managed_Restrict calculates a restriction of Boolean function.

- Biddy_Edge Biddy_Managed_Compose (Biddy_Manager MNG, Biddy_Edge f, Biddy_Variable v, Biddy_Edge g)
  Function Biddy_Managed_Compose calculates a composition of two Boolean functions.

- Biddy_Edge Biddy_Managed_E (Biddy_Manager MNG, Biddy_Edge f, Biddy_Variable v)
  Function Biddy_Managed_E calculates an existential quantification of Boolean function.

- Biddy_Edge Biddy_Managed_A (Biddy_Manager MNG, Biddy_Edge f, Biddy_Variable v)
  Function Biddy_Managed_A calculates an universal quantification of Boolean function.

- Biddy_Boolean Biddy_Managed_IsVariableDependent (Biddy_Manager MNG, Biddy_Edge f, Biddy_Variable v)
  Function Biddy_Managed_IsVariableDependent returns TRUE iff variable is dependent on others in a function.

- Biddy_Edge Biddy_Managed_ExistAbstract (Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge cube)
  Function Biddy_Managed_ExistAbstract existentially abstracts all the variables in cube from f.

- Biddy_Edge Biddy_Managed_UnivAbstract (Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge cube)
  Function Biddy_Managed_UnivAbstract universally abstracts all the variables in cube from f.

- Biddy_Edge Biddy_Managed_AndAbstract (Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g, Biddy_Edge cube)
  Function Biddy_Managed_AndAbstract calculates the AND of two BDDs and simultaneously (existentially) abstracts the variables in cube.

- Biddy_Edge Biddy_Managed_Constrain (Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge c)
  Function Biddy_Managed_Constrain calculates Coudert and Madre’s constrain function.

- Biddy_Edge Biddy_Managed_Simplify (Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge c)
  Function Biddy_Managed_Simplify calculates Coudert and Madre’s restrict function.

- Biddy_Edge Biddy_Managed_Support (Biddy_Manager MNG, Biddy_Edge f)
  Function Biddy_Managed_Support calculates a product of all dependent variables.

- Biddy_Edge Biddy_Managed_Replace (Biddy_Manager MNG, Biddy_Edge f)
  Function Biddy_Managed_Replace calculates BDD with one or more variables replaced.

- Biddy_Boolean Biddy_Managed_IsOK (Biddy_Manager MNG, Biddy_Edge f)
  Function Biddy_Managed_IsOK returns TRUE iff given node is not obsolete.

- void Biddy_Managed_Garbage (Biddy_Manager MNG)
  Function Biddy_Managed_Garbage performs garbage collection.

- void Biddy_Managed_Clean (Biddy_Manager MNG)
Function `Biddy_Managed_Clean` performs cleaning.

- void `Biddy_Managed_ClearAll (Biddy_Manager MNG)`
  
  Function `Biddy_Managed_ClearAll` performs exhaustive garbage collection.

- void `Biddy_Managed_AddCache (Biddy_Manager MNG, Biddy_GCFunction gc)`
  
  Function `Biddy_Managed_AddCache` adds cache to the end of Cache list.

- `Biddy_Boolean Biddy_Managed_FindFormula (Biddy_Manager MNG, Biddy_String x, Biddy_Edge *f)`
  
  Function `Biddy_Managed_FindFormula` finds formula in Formula table.

- void `Biddy_Managed_AddFormula (Biddy_Manager MNG, Biddy_String x, Biddy_Edge f, int count)`
  
  Function `Biddy_Managed_AddFormula` adds formula to Formula table.

- `Biddy_Boolean Biddy_Managed_DeleteFormula (Biddy_Manager MNG, Biddy_String x)`
  
  Function `Biddy_Managed_DeleteFormula` deletes formula from Formula table.

- `Biddy_Boolean Biddy_Managed_DeleteIthFormula (Biddy_Manager MNG, unsigned int i)`
  
  Function `Biddy_Managed_DeleteIthFormula` deletes formula from the table.

- `Biddy_Edge Biddy_Managed_GetIthFormula (Biddy_Manager MNG, unsigned int i)`
  
  Function `Biddy_Managed_GetIthFormula` returns ith formula in a Formula table.

- `Biddy_String Biddy_Managed_GetIthFormulaName (Biddy_Manager MNG, unsigned int i)`
  
  Function `Biddy_Managed_GetIthFormulaName` returns name of the ith formula in a Formula table.

- `Biddy_Variable Biddy_Managed_SwapWithHigher (Biddy_Manager MNG, Biddy_Variable v)`
  
  Function `Biddy_Managed_SwapWithHigher` swaps two adjacent variables.

- `Biddy_Variable Biddy_Managed_SwapWithLower (Biddy_Manager MNG, Biddy_Variable v)`
  
  Function `Biddy_Managed_SwapWithLower` swaps two adjacent variables.

- `Biddy_Boolean Biddy_Managed_Sifting (Biddy_Manager MNG, Biddy_Edge f)`
  
  Function `Biddy_Managed_Sifting` reorders variables to minimize node number for the whole system (if f = NULL) or for the given function (if f != NULL) using Rudell's sifting algorithm.

- void `Biddy_Managed_Purge (Biddy_Manager MNG)`
  
  Function `Biddy_Managed_Purge` denotes that one step of calculation is finished.

- void `Biddy_Managed_PurgeAndReorder (Biddy_Manager MNG, Biddy_Edge f)`
  
  Function `Biddy_Managed_PurgeAndReorder` denotes that one step of calculation is finished.

- void `Biddy_Managed_Refresh (Biddy_Manager MNG, Biddy_Edge f)`
  
  Function `Biddy_Managed_Refresh` refreshes all obsolete nodes in a function.

### 3.4.1 Detailed Description

File `biddyMain.c` contains main functions for representation and manipulation of boolean functions with ROBDDs.

#### Description

**PackageName** [Biddy]

**Synopsis** [Biddy provides data structures and algorithms for the representation and manipulation of Boolean functions with ROBDDs. A hash table is used for quick search of nodes. Complement edges decreases the number of nodes. An automatic garbage collection with a formulae counter is implemented. Variable swapping and sifting are implemented.]

**FileName** [biddyMain.c]

**Revision** [$Revision: 131$]

**Date** [$Date: 2016-01-02 09:42:28 +0100 (sob, 02 jan 2016) $]

**Authors** [Robert Meolic (robert.meolic@um.si), Ales Casar (ales@homemade.net)]
Copyright

Copyright (C) 2006, 2015 UM-FERI, Smetanova ulica 17, SI-2000 Maribor, Slovenia

Biddy is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

Biddy is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied
warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public
License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the
Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

More info

See also: biddy.h, biddyInt.h

Definition in file biddyMain.c.

3.4.2 Function Documentation

3.4.2.1 void Biddy_Init ( )

Function Biddy_Init initialize BDD package.

Description

Biddy_Init creates anonymous manager which consists of creating node table (biddyNodeTable), variable table
(biddyVariableTable), formula table (biddyFormulaTable), three basic caches (biddyIteCache, biddyEACache and
biddyRCCache), and cache list (biddyCacheList). Biddy_Init also initializes constant nodes (biddyOne, biddyZero),
memory management and automatic garbage collection.

Side effects

Allocate a lot of memory.

More info

Definition at line 189 of file biddyMain.c.

3.4.2.2 void Biddy_Exit ( )

Function Biddy_Exit deletes anonymous manager.

Description

Deallocates all memory allocated by Biddy_Init, Biddy_FoaVariable, Biddy_FoaNode etc.

Side effects

More info

Definition at line 588 of file biddyMain.c.

3.4.2.3 Biddy_String Biddy_About ( )

Function Biddy_About reports version of Biddy package.
3.4.2.4  Biddy_Edge Biddy_GetThen ( Biddy_Edge f )

Function Biddy_GetThen returns THEN successor.

Description
Input mark is not considered! External use, only.

Side effects

More info
Macro BiddyT(f) is defined for internal use.
Definition at line 759 of file biddyMain.c.

3.4.2.5  Biddy_Edge Biddy_GetElse ( Biddy_Edge f )

Function Biddy_GetElse returns ELSE successor.

Description
Input mark is not considered! External use, only.

Side effects

More info
Macro BiddyE(f) is defined for internal use.
Definition at line 783 of file biddyMain.c.

3.4.2.6  Biddy_Variable Biddy_GetTopVariable ( Biddy_Edge f )

Function Biddy_GetTopVariable returns the top variable.

Description

Side effects

More info
Definition at line 805 of file biddyMain.c.
Here is the caller graph for this function:

![Caller Graph](image)

### 3.4.2.7 void Biddy_SelectNode ( Biddy_Edge f )

Function Biddy_SelectNode selects the top node of the given function.

**Description**

**Side effects**

This would be faster but more danger to implement without checking if already selected.

**More info**

Definition at line 829 of file biddyMain.c.
3.4.2.8 void Biddy_DeselectNode ( Biddy_Edge f )

Function Biddy_DeselectNode deselects the top node of the given function.

Description

Side effects

This would be faster but more danger to implement without checking if already deselected.

More info

Definition at line 855 of file biddyMain.c.

Here is the caller graph for this function:

3.4.2.9 Biddy_Boolean Biddy_IsSelected ( Biddy_Edge f )

Function Biddy_IsSelected returns TRUE iff the top node of the given function is selected.

Description

Side effects
3.4 biddyMain.c File Reference

More info

Definition at line 880 of file biddyMain.c.
Here is the caller graph for this function:

```
Biddy_IsSelected
Biddy_NodeSelect
Biddy_NodeRepair

3.4.2.10 void Biddy_NodeSelect ( Biddy_Edge f )

Function Biddy_NodeSelect recursively selects all nodes of a given function.

Description

Side effects

Constant node must be selected before starting this function!

More info

Definition at line 904 of file biddyMain.c.

3.4.2.11 void Biddy_NodeRepair ( Biddy_Edge f )

Function Biddy_NodeRepair recursively repairs (deselects) all nodes of a given function.

Description

Selection of nodes (used for statistics, reordering and garbage collection techniques etc.) creates invalid counts that have to be repaired before any operation on BDDs.

Side effects

Constant node must not be selected before starting this function!

More info

Definition at line 935 of file biddyMain.c.
Here is the caller graph for this function:

![Caller Graph]

3.4.2.12  **Biddy_Edge Biddy_NOT ( Biddy_Edge f )**

Function Biddy_NOT complements Boolean function.

**Description**

Deprecated! Use macro Biddy_Not.

**Side effects**

**More info**

Definition at line 962 of file biddyMain.c.

3.4.2.13  **Biddy_Edge Biddy_TransferMark ( Biddy_Edge f, Biddy_Boolean mark )**

Function Biddy_TransferMark complements Boolean function if given parameter mark is TRUE.

**Description**

Deprecated! Use macro Biddy_NotCond.

**Side effects**

**More info**

Definition at line 987 of file biddyMain.c.

3.4.2.14  **Biddy_Edge Biddy_Managed_GetConstantZero ( Biddy_Manager MNG )**

Function Biddy_Managed_GetConstantZero returns constant 0.

**Description**

Constants 0 and 1 depend on a manager.
Side effects

More info

Internally, use macro biddyZero. Macro `Biddy_GetConstantZero()` is defined for use with anonymous manager. Definition at line 1009 of file biddyMain.c.

3.4.2.15  **Biddy_Edge Biddy_Managed_GetConstantOne ( Biddy_Manager MNG )**

Function `Biddy_Managed_GetConstantOne` returns constant 1.

Description

Constants 0 and 1 depend on a manager.

Side effects

More info

Internally, use macro biddyOne. Macro `Biddy_GetConstantOne()` is defined for use with anonymous manager. Definition at line 1028 of file biddyMain.c.

3.4.2.16  **Biddy_String Biddy_Managed_GetVariableName ( Biddy_Manager MNG, Biddy_Variable v )**

Function `Biddy_Managed_GetVariableName` returns the name of a variable.

Description

Side effects

More info

Macro `Biddy_GetVariableName(v)` is defined for use with anonymous manager. Definition at line 1049 of file biddyMain.c.

Here is the caller graph for this function:

![Caller Graph](image)

3.4.2.17  **Biddy_String Biddy_Managed_GetTopVariableName ( Biddy_Manager MNG, Biddy_Edge f )**

Function `Biddy_Managed_GetTopVariableName` returns the name of top variable.

Description

Side effects

More info

Macro `Biddy_GetTopVariableName(f)` is defined for use with anonymous manager.
3.4.2.18 char Biddy_Managed_GetTopVariableChar ( Biddy_Manager MNG, Biddy_Edge f )

Function Biddy_Managed_GetTopVariableChar returns the first character in the name of top variable.

Description

Side effects

More info

Macro Biddy_GetTopVariableChar(f) is defined for use with anonymous manager.

Definition at line 1075 of file biddyMain.c.

3.4.2.19 void Biddy_Managed_ResetVariablesValue ( Biddy_Manager MNG )

Function Biddy_Managed_ResetVariablesValue sets all variable’s value to biddyZero, except constant node which gets value biddyOne.

Description

Side effects

Only active (used) variables are reinitialized.

More info

Macro Biddy_ResetVariablesValue() is defined for use with anonymous manager.

Definition at line 1101 of file biddyMain.c.

3.4.2.20 void Biddy_Managed_SetVariableValue ( Biddy_Manager MNG, Biddy_Variable v, Biddy_Edge f )

Function Biddy_Managed_SetVariableValue sets variable’s value.

Description

Side effects

More info

Macro Biddy_SetVariableValue(v,f) is defined for use with anonymous manager.

Definition at line 1128 of file biddyMain.c.

3.4.2.21 Biddy_Boolean Biddy_Managed_IsSmaller ( Biddy_Manager MNG, Biddy_Variable fv, Biddy_Variable gv )

Function Biddy_Managed_IsSmaller returns TRUE if the first variable is smaller ( = lower = topmore).

Description

Side effects

More info

Macro Biddy_IsSmaller(fv,gv) is defined for use with anonymous manager.

Definition at line 1158 of file biddyMain.c.
3.4 biddyMain.c File Reference

Here is the caller graph for this function:

```
Biddy_Managed_AddVariable
Biddy_Managed_AddVariable
Below
Biddy_Managed_AddVariable
Above
```

3.4.2.22 Biddy_Edge Biddy_Managed_AddVariable ( Biddy_Manager MNG )

Function Biddy_Managed_AddVariable adds a numbered variable.

Description

Numbered variables have only digits in its name. The current number of numbered variables is stored in numnum. Function increments numnum and always creates non-existing variable. Function returns variable edge.

Side effects

More info

Macro Biddy_AddVariable() is defined for use with anonymous manager.

Definition at line 1213 of file biddyMain.c.

Here is the caller graph for this function:

3.4.2.23 Biddy_Edge Biddy_Managed_AddVariableBelow ( Biddy_Manager MNG, Biddy_Variable v )

Function Biddy_Managed_AddVariableBelow adds a numbered variable.
Description

Biddy_Managed_AddVariableBelow uses Biddy_Managed_AddVariable to add numbered variable. Then, the order of the new variable is changed to become immediately below the given variable (below = topmore in BDD) Function returns variable edge.

Side effects

More info

Macro Biddy_AddVariableBelow(v) is defined for use with anonymous manager.
Definition at line 1255 of file biddyMain.c.

3.4.2.24  Biddy_Edge Biddy_Managed_AddVariableAbove ( Biddy_Manager MNG, Biddy_Variable v )

Function Biddy_Managed_AddVariableAbove adds a numbered variable.

Description

Biddy_Managed_AddVariableAbove uses Biddy_Managed_AddVariable to add numbered variable. Then, the order of the new variable is changed to become immediately above the given variable (above = bottommore in BDD) Function returns variable edge.

Side effects

More info

Macro Biddy_AddVariableAbove(v) is defined for use with anonymous manager.
Definition at line 1301 of file biddyMain.c.

3.4.2.25  Biddy_Edge Biddy_Managed_FoaVariable ( Biddy_Manager MNG, Biddy_String x )

Function Biddy_Managed_FoaVariable finds or adds variable edge.

Description

Variable edge points to a node with successors biddyZero and biddyOne. If such variable edge already exists, function returns it and does not create the new one.

Side effects

Inefficient! We should use some kind of searching tree.

More info

Macro Biddy_FoaVariable(x) is defined for use with anonymous manager.
Definition at line 1352 of file biddyMain.c.

Here is the caller graph for this function:
Function Biddy_Managed_FoaNode finds or adds new node with the given variable and successors.

**Description**

If such node already exists, function returns it and does not create the new one.

**Side effects**

Using Biddy_Managed_FoaNode you can create node with arbitrary ordering. It is much more safe to use Biddy_Managed_ITE. To enable efficient implementation of e.g. sifting the function started with the returned node is not refreshed!

**More info**

Macro `Biddy_FoaNode(v, pf, pt, garbageAllowed)` is defined for use with anonymous manager.

Definition at line 1445 of file biddyMain.c.

Here is the caller graph for this function:

![Caller Graph](image)

3.4.2.27  Biddy_Edge Biddy_Managed_ITE ( Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g, Biddy_Edge h )

Function Biddy_Managed_ITE calculates ITE operation of three Boolean functions.

**Description**

**Side Effects**

**More info**

Macro `Biddy_ITE(f, g, h)` is defined for use with anonymous manager.

Definition at line 1596 of file biddyMain.c.
Here is the caller graph for this function:

![Caller Graph](image)

3.4.2.28 Biddy_Edge Biddy_Managed_And ( Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g )

Function Biddy_Managed_And calculates Boolean function AND (conjunction).

Description

Side Effects

Uses ITE. Less efficient as direct implementation in CUDD.

More Info

Macro Biddy_And(f,g) is defined for use with anonymous manager.

Definition at line 1891 of file biddyMain.c.
3.4.2.29  Biddy_Edge Biddy_Managed_Or ( Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g )

Function Biddy_Managed_Or calculates Boolean function OR (disjunction).

Description

Side Effects

Uses ITE. Less efficient as direct implementation in CUDD.

More Info

Macro Biddy_Or(f,g) is defined for use with anonymous manager.
Definition at line 1917 of file biddyMain.c.

3.4.2.30  Biddy_Edge Biddy_Managed_Nand ( Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g )

Function Biddy_Managed_Nand calculates Boolean function NAND (Sheffer).

Description

Side Effects

Uses ITE. Less efficient as direct implementation in CUDD.

More Info

Macro Biddy_Nand(f,g) is defined for use with anonymous manager.
Definition at line 1943 of file biddyMain.c.

3.4.2.31  Biddy_Edge Biddy_Managed_Nor ( Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g )

Function Biddy_Managed_Nor calculates Boolean function NOR (Peirce).

Description

Side Effects

Uses ITE. Less efficient as direct implementation in CUDD.

More Info

Macro Biddy_Nor(f,g) is defined for use with anonymous manager.
Definition at line 1969 of file biddyMain.c.

3.4.2.32  Biddy_Edge Biddy_Managed_Xor ( Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g )

Function Biddy_Managed_Xor calculates Boolean function XOR.

Description

Side Effects

Uses ITE. Less efficient as direct implementation in CUDD.
Macro **Biddy_Xor(f,g)** is defined for use with anonymous manager.
Definition at line 1995 of file biddyMain.c.

3.4.2.33 **Biddy_Edge Biddy_Managed_Xnor ( Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g )**

Function **Biddy_Managed_Xnor** calculates Boolean function XNOR.

Description

Side Effects

Uses ITE. Less efficient as direct implementation in CUDD.

More Info

Macro **Biddy_Xnor(f,g)** is defined for use with anonymous manager.
Definition at line 2021 of file biddyMain.c.

3.4.2.34 **Biddy_Boolean Biddy_Managed_Leq ( Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g )**

Function **Biddy_Managed_Leq** returns TRUE iff function f is included in function g.

Description

Side Effects

Implemented by calculating full implication which is less efficient as implementation in CUDD.

More Info

Macro **Biddy_Leq(f,g)** is defined for use with anonymous manager.
Definition at line 2049 of file biddyMain.c.

3.4.2.35 **Biddy_Edge Biddy_Managed_Intersect ( Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g )**

**Biddy_Managed_Intersect** calculates a function included in the intersection of f and g.

Description

If the result is not constant 0 then it is a witness that the intersection is not empty. The result should be calculated with as few new nodes as possible, and the result may not be the same as conjunction between functions! If the only result of interest is whether f and g intersect, **Biddy_Leq** should be used instead (return \((f \leq g')\)).

Side Effects

Implemented by calculating full conjunction which is less efficient as implementation in CUDD.

More Info

Macro **Biddy_Intersect(f,g)** is defined for use with anonymous manager.
Definition at line 2085 of file biddyMain.c.
3.4.2.36  Biddy_Edge Biddy_Managed_Restrict ( Biddy_Manager MNG, Biddy_Edge f, Biddy_Variable v,
        Biddy_Edge value )

Function Biddy_Managed_Restrict calculates a restriction of Boolean function.

Description

Original BDD is not changed! Value must be constant 0 or constant 1. It uses optimization: F(a=x) == NOT((NOT
F)(a=x)).

Side effects

This is not Coudert and Madre’s restrict function. Use Biddy_Simplify if you need that one. Cache table is not used,
yet.

More info

Macro Biddy_Restrict(t,v.value) is defined for use with anonymous manager.

Definition at line 2117 of file biddyMain.c.

Here is the caller graph for this function:

3.4.2.37  Biddy_Edge Biddy_Managed_Compose ( Biddy_Manager MNG, Biddy_Edge f, Biddy_Variable v,
        Biddy_Edge g )

Function Biddy_Managed_Compose calculates a composition of two Boolean functions.

Description

Original BDDs are not changed! It uses optimization: F(a=G) == NOT((NOT F)(a=G)).

Side effects

Cache table is not used, yet.

More info

Macro Biddy_Compose(f,v,g) is defined for use with anonymous manager.

Definition at line 2196 of file biddyMain.c.

3.4.2.38  Biddy_Edge Biddy_Managed_E ( Biddy_Manager MNG, Biddy_Edge f, Biddy_Variable v )

Function Biddy_Managed_E calculates an existential quantification of Boolean function.

Description

Original BDD is not changed! Be careful: ExA F != NOT(ExA (NOT F)). Counterexample: Exb (AND (NOT a) b c).
Side effects

More info

Macro `Biddy_E(f,v)` is defined for use with anonymous manager.
Definition at line 2281 of file `biddyMain.c`.
Here is the caller graph for this function:

```
Biddy_Managed_E  Biddy_Managed_A  Biddy_Managed_Simplify
```

3.4.2.39 `Biddy_Edge Biddy_Managed_A ( Biddy_Manager MNG, Biddy_Edge f, Biddy_Variable v )`

Function `Biddy_Managed_A` calculates an universal quantification of Boolean function.

Description

Original BDD is not changed! Be careful: AxA F !⇒ NOT(AxA (NOT F)). Counterexample: Axb (AND (NOT a) b c).

Side effects

More info

Macro `Biddy_A(f,v)` is defined for use with anonymous manager.
Definition at line 2365 of file `biddyMain.c`.
Here is the caller graph for this function:

```
Biddy_Managed_A  Biddy_Managed_IsVariableDependent
```

3.4.2.40 `Biddy_Boolean Biddy_Managed_IsVariableDependent ( Biddy_Manager MNG, Biddy_Edge f, Biddy_Variable v )`

Function `Biddy_Managed_IsVariableDependent` returns TRUE iff variable is dependent on others in a function.
3.4 biddyMain.c File Reference

Description
A variable is dependent on others in a function iff universal quantification of this variable returns constant FALSE.

Side effects
Implemented by calculating full universal quantification which is less efficient as direct implementation in CUDD.

More info
Macro Biddy_IsVariableDependent(f,v) is defined for use with anonymous manager.
Definition at line 2417 of file biddyMain.c.

3.4.2.41 Biddy_Edge Biddy_Managed_ExistAbstract ( Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge cube )
Function Biddy_Managed_ExistAbstract existentially abstracts all the variables in cube from f.

Description

Side effects
Original BDD is not changed!

More info
Macro Biddy_ExistAbstract(f,cube) is defined for use with anonymous manager.
Definition at line 2447 of file biddyMain.c.
Here is the caller graph for this function:

3.4.2.42 Biddy_Edge Biddy_Managed_UnivAbstract ( Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge cube )
Function Biddy_Managed_UnivAbstract universally abstracts all the variables in cube from f.

Description

Side effects
Original BDD is not changed!

More info
Macro Biddy_UnivAbstract(f,cube) is defined for use with anonymous manager.
Definition at line 2541 of file biddyMain.c.
Function Biddy_Managed_AndAbstract calculates the AND of two BDDs and simultaneously (existentially) abstracts the variables in cube.

Description

Side effects

More info

Macro Biddy_AndAbstract(f,g,cube) is defined for use with anonymous manager.
Definition at line 2588 of file biddyMain.c.

Function Biddy_Managed_Constrain calculates Coudert and Madre's constrain function.

Description

Coudert and Madre's constrain function is also called a generalized cofactor of function f with respect to function c.

Side effects

Cache table is not implemented, yet.

More info

Macro Biddy_Constrain(f,c) is defined for use with anonymous manager.
Definition at line 2742 of file biddyMain.c.

Function Biddy_Managed_Simplify calculates Coudert and Madre's restrict function.

Description

Coudert and Madre's restrict function tries to simplify function f by restricting it to the domain covered by function c.

Side effects

No checks are done to see if the result is actually smaller than the input. Cache table is not implemented, yet.

More info

Macro Biddy_Simplify(f,c) is defined for use with anonymous manager.
Definition at line 2835 of file biddyMain.c.

Function Biddy_Managed_Support calculates a product of all dependent variables.

Description

Side effects
Macro `Biddy_Support(f)` is defined for use with anonymous manager.
Definition at line 2928 of file biddyMain.c.

Here is the caller graph for this function:

```
Biddy_Managed_Support
    Biddy_Managed_WriteTable

Biddy_Managed_Sifting
```

### 3.4.2.47 Biddy_Edge Biddy_Managed_Replace ( Biddy_Manager MNG, Biddy_Edge f )

Function `Biddy_Managed_Replace` calculates BDD with one or more variables replaced.

**Description**

Original BDD is not changed! Replacing is controlled by variable’s values (which are edges!). Use `Biddy_Managed_ResetVariablesValue` and `Biddy_Managed_SetVariableValue` to prepare replacing. Current and new variables should be disjoint sets.

**Side effects**

Cache table is not implemented, yet.

**More info**

Macro `Biddy_Replace(f)` is defined for use with anonymous manager.
Definition at line 2998 of file biddyMain.c.

### 3.4.2.48 Biddy_Boolean Biddy_Managed_IsOK ( Biddy_Manager MNG, Biddy_Edge f )

Function `Biddy_Managed_IsOK` returns TRUE iff given node is not obsolete.

**Description**

External use, only.

**Side effects**

**More info**

Macro `BiddyIsOK(f)` is defined for internal use. Macro `Biddy_IsOK(f)` is defined for use with anonymous manager.
Definition at line 3063 of file biddyMain.c.
Function Biddy_Managed_Garbage performs garbage collection.

Description

Part of List of new nodes is checked (until freshNodes), bad nodes are deleted whilst fortified nodes in this part are moved to List of fortified nodes.

Side effects

The first element of each chain in a node table should have a special value for its ‘prev’ element to allow tricky but efficient deleting. Moreover, ‘prev’ and ‘next’ should be the first and the second element in the structure BiddyNode, respectively. Not all fortified nodes are moved to List of fortified nodes, the ones after ‘freshNode’ remains! Garbage collection is reported by biddyNodeTable.garbage only if some bad nodes are purged!

More info

Macro Biddy_Garbage() is defined for use with anonymous manager.

Definition at line 3102 of file biddyMain.c.

Here is the caller graph for this function:

![Caller Graph](image)

Function Biddy_Managed_Clean performs cleaning.

Description

Discard all nodes which are not preserved or which are not preserved anymore. Obsolete nodes are not immediately removed, they will be removed during the first garbage collection.

Side effects

Tag deleted is not considered and thus no fortified node is removed.
Macro \texttt{Biddy\_Clean()} is defined for use with anonymous manager.
Definition at line 3268 of file \textit{biddyMain.c}.

\subsection{3.4.2.51 \texttt{void Biddy\_Managed\_ClearAll ( Biddy\_Manager MNG )}}

Function \texttt{Biddy\_Managed\_ClearAll} performs exhaustive garbage collection.

\textbf{Description}
Immediately remove all nodes except the permanently preserved ones which were not made obsolete by \texttt{Biddy\_DeleteFormula}.

\textbf{Side effects}

Macro \texttt{Biddy\_ClearAll()} is defined for use with anonymous manager.
Definition at line 3329 of file \textit{biddyMain.c}.

\subsection{3.4.2.52 \texttt{void Biddy\_Managed\_AddCache ( Biddy\_Manager MNG, Biddy\_GCFunction gc )}}

Function \texttt{Biddy\_Managed\_AddCache} adds cache to the end of Cache list.

\textbf{Description}
If Cache list does not exist, function creates it.

\textbf{Side effects}

Macro \texttt{Biddy\_AddCache(gc)} is defined for use with anonymous manager.
Definition at line 3439 of file \textit{biddyMain.c}.

Here is the caller graph for this function:

```
Biddy\_Managed\_AddCache Biddy\_Init
```

\subsection{3.4.2.53 \texttt{Biddy\_Boolean Biddy\_Managed\_FindFormula ( Biddy\_Manager MNG, Biddy\_String x, Biddy\_Edge \* f )}}

Function \texttt{Biddy\_Managed\_FindFormula} find formula in Formula table.

\textbf{Description}

\textbf{Side effects}
More info

Macro `Biddy_FindFormula(x,f)` is defined for use with anonymous manager.
Definition at line 3487 of file biddyMain.c.

### 3.4.2.54 `void Biddy_Managed_AddFormula ( Biddy_Manager MNG, Biddy_String x, Biddy_Edge f, int count )`

Function `Biddy_Managed_AddFormula` adds formula to Formula table.

**Description**

Nodes of the given BDD will be preserved for the given number of clearings. If (name != NULL) then formula is accessible by its name. If (count == 0) then formula is persistently preserved and you have to use `Biddy_DeleteFormula` and `Biddy_ClearAll` to remove its nodes. There are two macros defined to simplify formulae management. Macro `Biddy_Managed_AddTmpFormula(mng,bdd,count)` is defined as `Biddy_Manager_AddFormula(mng,NULL,bdd,count)` and macro `Biddy_Managed_AddPersistentFormula(mng,name,bdd)` is defined as `Biddy_Manager_AddFormula(mng,name,bdd,0)`.

**Side effects**

Function is prolonged or fortified. Formulae with name are ordered by name. If formula with same name already exists, it will be overwritten (preserved and persistently preserved formulae, too)! Obsolete nodes of preserved and persistently preserved formulae will not be immediately removed.

More info

Macro `Biddy_AddFormula(x,f)` is defined for use with anonymous manager.
Definition at line 3571 of file biddyMain.c.

### 3.4.2.55 `Biddy_Boolean Biddy_Managed_DeleteFormula ( Biddy_Manager MNG, Biddy_String x )`

Function `Biddy_Managed_DeleteFormula` delete formula from Formula table.

**Description**

Nodes of the given formula are tagged as not needed. Formula is not accessible by its name anymore. Regular cleaning with `Biddy_Clean` is not considered this tag. Obsolete nodes from preserved formulae can be immediately removed in explicit call to `Biddy_ClearAll` (this is the only way to remove obsolete nodes from permanently preserved formulae).

**Side effects**

Function representing a variable cannot be removed. Formula is labelled and not immediately removed. Nodes are not immediately removed.

More info

Macro `Biddy_DeleteFormula(x)` is defined for use with anonymous manager.
Definition at line 3746 of file biddyMain.c.

### 3.4.2.56 `Biddy_Boolean Biddy_Managed_DeleteIthFormula ( Biddy_Manager MNG, unsigned int i )`

Function `Biddy_Managed_DeleteIthFormula` deletes formula from the table.
Description

Nodes of the given formula are tagged as not needed. Formula is not accessible by its name anymore. Regular cleaning with Biddy_Clean is not considered this tag. Obsolete nodes from preserved formulae can be immediately removed in explicit call to Biddy_ClearAll (this is the only way to remove obsolete nodes from permanently preserved formulae).

Side effects

Function representing a variable cannot be removed. Formula is labelled and not immediately removed. Nodes are not immediately removed.

More info

Macro Biddy_DeleteIthFormula(x) is defined for use with anonymous manager.
Definition at line 3813 of file biddyMain.c.
Here is the caller graph for this function:

3.4.2.57  Biddy_Edge Biddy_Managed_GetIthFormula ( Biddy_Manager MNG, unsigned int i )

Function Biddy_Managed_GetIthFormula returns ith formula in a Formula table.

Description

Return biddyNull if ith formulae does not exist.

Side effects

After adding new formula the index of others may change!

More info

Macro Biddy_GetIthFormula(i) is defined for use with anonymous manager.
Definition at line 3873 of file biddyMain.c.

3.4.2.58  Biddy_String Biddy_Managed_GetIthFormulaName ( Biddy_Manager MNG, unsigned int i )

Function Biddy_Managed_GetIthFormulaName returns name of the ith formula in a Formula table.

Description

Return NULL if ith formulae does not exist.

Side effects

After adding new formula the index of others may change!
Macro **Biddy_GetIthFormulaName(i)** is defined for use with anonymous manager.
Definition at line 3904 of file biddyMain.c.

### 3.4.2.59 Biddy_Variable Biddy_Managed_SwapWithHigher ( Biddy_Manager MNG, Biddy_Variable v )

Function **Biddy_Managed_SwapWithHigher** swaps two adjacent variables.

**Description**

Higher (greater) variable is the bottommore one! The highest element is constant "1". Constant '1' has global ordering numUsedVariables (not smaller than anyone). Global ordering is the number of 0 in corresponding line of orderingTable.

**Side effects**

More info

Macro **Biddy_SwapWithHigher(v)** is defined for use with anonymous manager.
Definition at line 3944 of file biddyMain.c.

### 3.4.2.60 Biddy_Variable Biddy_Managed_SwapWithLower ( Biddy_Manager MNG, Biddy_Variable v )

Function **Biddy_Managed_SwapWithLower** swaps two adjacent variables.

**Description**

Lower (smaller) variable is the topmore one! The lowest (topmost) element is not fixed. Topmost variable has global ordering 1 (smaller than all except itself). Global ordering is the number of 0 in corresponding line of orderingTable.

**Side effects**

More info

Macro **Biddy_SwapWithLower(v)** is defined for use with anonymous manager.
Definition at line 3983 of file biddyMain.c.

### 3.4.2.61 Biddy_Boolean Biddy_Managed_Sifting ( Biddy_Manager MNG, Biddy_Edge f )

Function **Biddy_Managed_Sifting** reorders variables to minimize node number for the whole system (if f = NULL) or for the given function (if f != NULL) using Rudell's sifting algorithm.

**Description**

Variables are reordered globally! All fresh nodes are removed

**Side effects**

It is hard to implement sifting if fresh nodes cannot be removed.

More info

Macro **Biddy_Sifting(f)** is defined for use with anonymous manager.
Definition at line 4023 of file biddyMain.c.
Here is the caller graph for this function:

![Caller Graph Diagram]

3.4.2.62 void Biddy_Managed_Purge ( Biddy_Manager MNG )

Function Biddy_Managed_Purge denotes that one step of calculation is finished.

Description

Immediately remove all nodes which were not preserved or which are not preserved anymore. Call to Biddy_Purge does not count as clearing and thus all preserved formulae remains preserved for the same number of clearings.

Side effects

Tag deleted is not considered and thus no fortified node is removed.

More info

Macro Biddy_Purge(f) is defined for use with anonymous manager.

Definition at line 4790 of file biddyMain.c.

3.4.2.63 void Biddy_Managed_PurgeAndReorder ( Biddy_Manager MNG, Biddy_Edge f )

Function Biddy_Managed_PurgeAndReorder denotes that one step of calculation is finished.

Description

Immediately remove all nodes which were not preserved or which are not preserved anymore. Call to Biddy_PurgeAndReorder does not count as clearing and thus all preserved formulae remains preserved for the same number of clearings. Moreover, Biddy_PurgeAndReorder trigger reordering on function (if BDD is given) or global reordering (if NULL is given).

Side effects

Tag deleted is not considered and thus no fortified node is removed. Removing nodes is done during sifting.

More info

Macro Biddy_PurgeAndReorder(f) is defined for use with anonymous manager.

Definition at line 4825 of file biddyMain.c.

3.4.2.64 void Biddy_Managed_Refresh ( Biddy_Manager MNG, Biddy_Edge f )

Function Biddy_Managed_Refresh refreshes all obsolete nodes in a function.

Description

It is supposed that childrens of not-obsolete nodes are all not obsolete.
Side effects

This is an external variant of internal function BiddyRefresh. It is needed to implement user caches.

More info

Definition at line 4853 of file biddyMain.c.

3.5 biddyStat.c File Reference

File biddyStat.c contains statistical functions.

#include "biddyInt.h"

Functions

- unsigned int Biddy_NodeNumber (Biddy_Edge f)
  Function Biddy_NodeNumber.
- unsigned int Biddy_NodeMaxLevel (Biddy_Edge f)
  Function Biddy_NodeMaxLevel.
- float Biddy_NodeAvgLevel (Biddy_Edge f)
  Function Biddy_NodeAvgLevel.
- Biddy_Variable Biddy_Managed_VariableTableNum (Biddy_Manager MNG)
  Function Biddy_Managed_VariableTableNum returns number of used variables.
- unsigned int Biddy_Managed_NodeTableSize (Biddy_Manager MNG)
  Function Biddy_Managed_NodeTableSize returns the size of node table.
- unsigned int Biddy_Managed_NodeTableMax (Biddy_Manager MNG)
  Function Biddy_Managed_NodeTableMax.
- unsigned int Biddy_Managed_NodeTableNum (Biddy_Manager MNG)
  Function Biddy_Managed_NodeTableNum.
- unsigned int Biddy_Managed_NodeTableNumVar (Biddy_Manager MNG, Biddy_Variable v)
  Function Biddy_Managed_NodeTableNumVar.
- unsigned int Biddy_Managed_NodeTableNumF (Biddy_Manager MNG)
  Function Biddy_Managed_NodeTableNumF.
- unsigned long long int Biddy_Managed_NodeTableFOA (Biddy_Manager MNG)
  Function Biddy_Managed_NodeTableFOA.
- unsigned long long int Biddy_Managed_NodeTableCompare (Biddy_Manager MNG)
  Function Biddy_Managed_NodeTableCompare.
- unsigned long long int Biddy_Managed_NodeTableAdd (Biddy_Manager MNG)
  Function Biddy_Managed_NodeTableAdd.
- unsigned int Biddy_Managed_NodeTableGarbage (Biddy_Manager MNG)
  Function Biddy_Managed_NodeTableGarbage.
- unsigned int Biddy_Managed_NodeTableGenerated (Biddy_Manager MNG)
  Function Biddy_Managed_NodeTableGenerated.
- unsigned int Biddy_Managed_NodeTableBlockNumber (Biddy_Manager MNG)
  Function Biddy_Managed_NodeTableBlockNumber.
- unsigned int Biddy_Managed_NodeTableSwapNumber (Biddy_Manager MNG)
  Function Biddy_Managed_NodeTableSwapNumber.
- unsigned int Biddy_Managed_NodeTableSiftingNumber (Biddy_Manager MNG)
  Function Biddy_Managed_NodeTableSiftingNumber.
3.5 biddyStat.c File Reference

- unsigned int Biddy_Managed_ListUsed (Biddy_Manager MNG)
  
  Function Biddy_Managed_ListUsed.

- unsigned int Biddy_Managed_ListMaxLength (Biddy_Manager MNG)
  
  Function Biddy_Managed_ListMaxLength.

- float Biddy_Managed_ListAvgLength (Biddy_Manager MNG)
  
  Function Biddy_Managed_ListAvgLength.

- unsigned long long int Biddy_Managed_IteCacheSearch (Biddy_Manager MNG)
  
  Function Biddy_Managed_IteCacheSearch.

- unsigned long long int Biddy_Managed_IteCacheFind (Biddy_Manager MNG)
  
  Function Biddy_Managed_IteCacheFind.

- unsigned long long int Biddy_Managed_IteCacheOverwrite (Biddy_Manager MNG)
  
  Function Biddy_Managed_IteCacheOverwrite.

- unsigned int Biddy_Managed_NodeNumberPlain (Biddy_Manager MNG, Biddy_Edge f)
  
  Function Biddy_Managed_NodeNumberPlain.

- unsigned int Biddy_Managed_VariableNumber (Biddy_Manager MNG, Biddy_Edge f)
  
  Function Biddy_Managed_VariableNumber.

- void Biddy_Managed_NodeVarNumber (Biddy_Manager MNG, Biddy_Edge f, unsigned int *n, unsigned int *v)
  
  Function Biddy_Managed_NodeVarNumber.

- double Biddy_Managed_CountMinterm (Biddy_Manager MNG, Biddy_Edge f, unsigned int nvars)
  
  Function Biddy_Managed_CountMinterm.

- unsigned long long int Biddy_Managed_ReadMemoryInUse (Biddy_Manager MNG)
  
  Function Biddy_Managed_ReadMemoryInUse report memory consumption of main data structures (nodes, node table, variable table, ordering table, formula table, ITE cache, EA cache, RC cache) in bytes.

- void Biddy_Managed_PrintInfo (Biddy_Manager MNG, FILE *f)
  
  Function Biddy_Managed_PrintInfo prepare a file with stats.

3.5.1 Detailed Description

File biddyStat.c contains statistical functions.

Description

PackageName [Biddy]

Synopsis [Biddy provides data structures and algorithms for the representation and manipulation of Boolean functions with ROBDDs. A hash table is used for quick search of nodes. Complement edges decreases the number of nodes. An automatic garbage collection with a formulae counter is implemented. Variable swapping and sifting are implemented.]

FileName [biddyStat.c]

Revision [$Revision: 132 $]

Date [$Date: 2016-01-02 09:59:22 +0100 (sob, 02 jan 2016) $]

Authors [Robert Meolic (robert.meolic@um.si), Ales Casar (ales@homemade.net)]

Copyright

Copyright (C) 2006, 2015 UM-FERI, Smetanova ulica 17, SI-2000 Maribor, Slovenia

Biddy is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

Biddy is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

Generated on Sat Jan 2 2016 11:00:42 for Biddy by Doxygen
3.5.2 Function Documentation

3.5.2.1 unsigned int Biddy_NodeNumber ( Biddy_Edge f )

Function Biddy_NodeNumber.

Description

Count number of nodes in a BDD.

Side effects

Definition at line 75 of file biddyStat.c.

3.5.2.2 unsigned int Biddy_NodeMaxLevel ( Biddy_Edge f )

Function Biddy_NodeMaxLevel.

Description

Side effects

Definition at line 113 of file biddyStat.c.

3.5.2.3 float Biddy_NodeAvgLevel ( Biddy_Edge f )

Function Biddy_NodeAvgLevel.

Description

Side effects

Definition at line 141 of file biddyStat.c.

3.5.2.4 Biddy_Variable Biddy_Managed_VariableTableNum ( Biddy_Manager MNG )

Function Biddy_Managed_VariableTableNum returns number of used variables.

Description

Side effects

Variable '1' is included.
3.5 biddyStat.c File Reference

More info

Macro `Biddy_VariableTableNum()` is defined for use with anonymous manager.
Definition at line 173 of file biddyStat.c.
Here is the caller graph for this function:

```
Biddy_Managed_VariableTableNum
    Biddy_Managed_Sifting
    Biddy_Managed_PrintInfo
```

3.5.2.5 `unsigned int Biddy_Managed_NodeTableSize ( Biddy_Manager MNG )`

Function `Biddy_Managed_NodeTableSize` returns the size of node table.

Description

Side effects

More info

Macro `Biddy_NodeTableSize()` is defined for use with anonymous manager.
Definition at line 198 of file biddyStat.c.
Here is the caller graph for this function:

```
Biddy_Managed_NodeTableSize
    Biddy_Managed_PrintInfo
```

3.5.2.6 `unsigned int Biddy_Managed_NodeTableMax ( Biddy_Manager MNG )`

Function `Biddy_Managed_NodeTableMax`.

Description

Side effects

More info

Macro `Biddy_NodeTableMax()` is defined for use with anonymous manager.
Definition at line 227 of file biddyStat.c.
Here is the caller graph for this function:

```
Biddy_Managed_NodeTableMax -> Biddy_Managed_PrintInfo
```

3.5.2.7 unsigned int Biddy_Managed_NodeTableNum ( Biddy_Manager MNG )

Function Biddy_Managed_NodeTableNum.

Description

Side effects

More info

Macro Biddy_NodeTableNum() is defined for use with anonymous manager.

Definition at line 252 of file biddyStat.c.

Here is the caller graph for this function:

```
Biddy_Managed_NodeTableNum -> Biddy_Managed_PrintInfo
```

3.5.2.8 unsigned int Biddy_Managed_NodeTableNumVar ( Biddy_Manager MNG, Biddy_Variable v )

Function Biddy_Managed_NodeTableNumVar.

Description

Side effects

More info

Macro Biddy_NodeTableNumVar(v) is defined for use with anonymous manager.

Definition at line 277 of file biddyStat.c.

3.5.2.9 unsigned int Biddy_Managed_NodeTableNumF ( Biddy_Manager MNG )

Function Biddy_Managed_NodeTableNumF.

Description
Macro `Biddy_NodeTableNumF()` is defined for use with anonymous manager. Definition at line 302 of file biddyStat.c. Here is the caller graph for this function:

![Caller graph of Biddy_NodeTableNumF](image)

3.5.2.10  `unsigned long long int Biddy_Managed_NodeTableFOA ( Biddy_Manager MNG )`

Function `Biddy_Managed_NodeTableFOA`. Description

Side effects

More info

Macro `Biddy_NodeTableFOA()` is defined for use with anonymous manager. Definition at line 327 of file biddyStat.c. Here is the caller graph for this function:

![Caller graph of Biddy_NodeTableFOA](image)

3.5.2.11  `unsigned long long int Biddy_Managed_NodeTableCompare ( Biddy_Manager MNG )`

Function `Biddy_Managed_NodeTableCompare`. Description

Side effects

More info

Macro `Biddy_NodeTableCompare()` is defined for use with anonymous manager.
3.5.2.12 unsigned long long int Biddy_Managed_NodeTableAdd ( Biddy_Manager MNG )

Function Biddy_Managed_NodeTableAdd.

Description

Side effects

More info

Macro Biddy_NodeTableAdd() is defined for use with anonymous manager.

Definition at line 377 of file biddyStat.c.

Here is the caller graph for this function:

3.5.2.13 unsigned int Biddy_Managed_NodeTableGarbage ( Biddy_Manager MNG )

Function Biddy_Managed_NodeTableGarbage.

Description

Side effects

More info

Macro Biddy_NodeTableGarbage() is defined for use with anonymous manager.

Definition at line 402 of file biddyStat.c.
Here is the caller graph for this function:

```
Biddy_Managed_NodeTableGarbage  Biddy_Managed_PrintInfo
```

3.5.2.14  unsigned int Biddy_Managed_NodeTableGenerated ( Biddy_Manager MNG )

Function Biddy_Managed_NodeTableGenerated.

Description

Side effects

More info

Macro Biddy_NodeTableGenerated() is defined for use with anonymous manager.

Definition at line 427 of file biddyStat.c.

Here is the caller graph for this function:

```
Biddy_Managed_NodeTableGenerated  Biddy_Managed_PrintInfo
```

3.5.2.15  unsigned int Biddy_Managed_NodeTableBlockNumber ( Biddy_Manager MNG )

Function Biddy_Managed_NodeTableBlockNumber.

Description

Side effects

More info

Macro Biddy_NodeTableBlockNumber() is defined for use with anonymous manager.

Definition at line 453 of file biddyStat.c.
Here is the caller graph for this function:

3.5.2.16  unsigned int Biddy_Managed_NodeTableSwapNumber ( Biddy_Manager MNG )

Function Biddy_Managed_NodeTableSwapNumber.

Description

Side effects

More info

Macro Biddy_NodeTableSwapNumber() is defined for use with anonymous manager.

Definition at line 478 of file biddyStat.c.

Here is the caller graph for this function:

3.5.2.17  unsigned int Biddy_Managed_NodeTableSiftingNumber ( Biddy_Manager MNG )

Function Biddy_Managed_NodeTableSiftingNumber.

Description

Side effects

More info

Macro Biddy_NodeTableSiftingNumber() is defined for use with anonymous manager.

Definition at line 504 of file biddyStat.c.
Here is the caller graph for this function:

3.5.2.18 unsigned int Biddy_Managed_ListUsed ( Biddy_Manager MNG )

Function Biddy_Managed_ListUsed.

Description

Side effects

More info

Macro Biddy_ListUsed() is defined for use with anonymous manager.
Definition at line 529 of file biddyStat.c.
Here is the caller graph for this function:

3.5.2.19 unsigned int Biddy_Managed_ListMaxLength ( Biddy_Manager MNG )

Function Biddy_Managed_ListMaxLength.

Description

Side effects

More info

Macro Biddy_ListMaxLength() is defined for use with anonymous manager.
Definition at line 562 of file biddyStat.c.
Here is the caller graph for this function:

![Caller Graph]

### 3.5.2.20 float Biddy_Managed_ListAvgLength ( Biddy_Manager MNG )

Function Biddy_Managed_ListAvgLength.

**Description**

**Side effects**

**More info**

Macro `Biddy_ListAvgLength()` is defined for use with anonymous manager.

Definition at line 625 of file `biddyStat.c`.

Here is the caller graph for this function:

![Caller Graph]

### 3.5.2.21 unsigned long long int Biddy_Managed_IteCacheSearch ( Biddy_Manager MNG )

Function Biddy_Managed_IteCacheSearch.

**Description**

**Side effects**

**More info**

Macro `Biddy_IteCacheSearch()` is defined for use with anonymous manager.

Definition at line 666 of file `biddyStat.c`.
3.5.2.22  unsigned long long int Biddy_Managed_IteCacheFind ( Biddy_Manager MNG )

Function Biddy_Managed_IteCacheFind.

Description

Side effects

More info

Macro Biddy_IteCacheFind() is defined for use with anonymous manager.

Definition at line 691 of file biddyStat.c.

Here is the caller graph for this function:

3.5.2.23  unsigned long long int Biddy_Managed_IteCacheOverwrite ( Biddy_Manager MNG )

Function Biddy_Managed_IteCacheOverwrite.

Description

Side effects

More info

Macro Biddy_IteCacheOverwrite() is defined for use with anonymous manager.

Definition at line 716 of file biddyStat.c.
Here is the caller graph for this function:

![Caller Graph]

3.5.2.24 unsigned int Biddy_Managed_NodeNumberPlain ( Biddy_Manager MNG, Biddy_Edge f )

Function Biddy_Managed_NodeNumberPlain.

Description

Count number of nodes in a corresponding BDD without complement edges.

More info

Macro Biddy_NodeNumberPlain(f) is defined for use with anonymous manager.
Definition at line 742 of file biddyStat.c.

3.5.2.25 unsigned int Biddy_Managed_VariableNumber ( Biddy_Manager MNG, Biddy_Edge f )

Function Biddy_Managed_VariableNumber.

Description

Count number of dependent variables in a function.

More info

Macro Biddy_VariableNumber(f) is defined for use with anonymous manager.
Definition at line 779 of file biddyStat.c.

Here is the caller graph for this function:

![Caller Graph]
3.5.2.26  void Biddy_Managed_NodeVarNumber ( Biddy_Manager MNG, Biddy_Edge f, unsigned int * n, unsigned int * v )

Function Biddy_Managed_NodeVarNumber.

Description
Count number of nodes and number of dependent variables in a function.

Side effects
This is faster as counting separately.

More info
Macro Biddy_NodeVarNumber(f,n,v) is defined for use with anonymous manager.
Definition at line 832 of file biddyStat.c.

3.5.2.27  double Biddy_Managed_CountMinterm ( Biddy_Manager MNG, Biddy_Edge f, unsigned int nvars )

Function Biddy_Managed_CountMinterm.

Description
If nvars == 0 then number of dependent variables is used.

Side effects
We are using GNU Multiple Precision Arithmetic Library (GMP).

More info
Macro Biddy_CountMinterm(f,nvars) is defined for use with anonymous manager.
Definition at line 886 of file biddyStat.c.

3.5.2.28  unsigned long long int Biddy_Managed_ReadMemoryInUse ( Biddy_Manager MNG )

Function Biddy_Managed_ReadMemoryInUse report memory consumption of main data structures (nodes, node table, variable table, ordering table, formula table, ITE cache, EA cache, RC cache) in bytes.

Description
Side effects

More info
Macro Biddy_ReadMemoryInUse() is defined for use with anonymous manager.
Definition at line 940 of file biddyStat.c.
Here is the caller graph for this function:

```
Biddy_Managed_ReadMemory
    InUse
```

3.5.2.29  void Biddy_Managed_PrintInfo ( Biddy_Manager MNG, FILE * f )

Function Biddy_Managed_PrintInfo prepare a file with stats.

Description

Side effects

More info

Macro Biddy_PrintInfo(f) is defined for use with anonymous manager.

Definition at line 1025 of file biddyStat.c.
Index

biddy.h, 13
  Biddy_A, 20
  Biddy_AddCache, 22
  Biddy_AddFormula, 22
  Biddy_AddVariable, 18
  Biddy_AddVariableAbove, 18
  Biddy_AddVariableBelow, 18
  Biddy_And, 19
  Biddy_AndAbstract, 20
  Biddy_Bool, 18
  Biddy_Cache, 27
  Biddy_ClearAll, 22
  Biddy_ClearMark, 16
  Biddy_Complement, 17
  Biddy_Compose, 20
  Biddy_Constrain, 20
  Biddy_CountMinterm, 25
  Biddy_DeleteFormula, 22
  Biddy_DeleteIthFormula, 22
  Biddy_E, 20
  Biddy_Edge, 27
  Biddy_Eval, 26
  Biddy_Eval1, 26
  Biddy_Eval1x, 26
  Biddy_Eval2, 26
  Biddy_ExistAbstract, 20
  Biddy_FindFormula, 22
  Biddy_FoaNode, 18
  Biddy_FoaVariable, 18
  Biddy_GCFunction, 27
  Biddy_Garbage, 21
  Biddy_GetConstantOne, 17
  Biddy_GetConstantZero, 17
  Biddy_GetIthFormula, 22
  Biddy_GetIthFormulaName, 22
  Biddy_GetMark, 16
  Biddy_GetTopFormulaChar, 18
  Biddy_GetTopFormulaName, 17
  Biddy_GetVariableName, 17
  Biddy_ITE, 19
  Biddy_Intersect, 19
  Biddy_InvertMark, 16
  Biddy_IsConstant, 16
  Biddy_IsEqvPointer, 16
  Biddy_IsNull, 16
  Biddy_IsOK, 21
  Biddy_IsSmaller, 18
  Biddy_IsVariableDependent, 20
  Biddy_IteCacheFind, 25
  Biddy_IteCacheOverwrite, 25
  Biddy_IteCacheSearch, 25
  Biddy_Leg, 19
  Biddy_ListAvgLength, 25
  Biddy_ListMaxLength, 24
  Biddy_ListUsed, 24
  Biddy_LookupFunction, 28
  Biddy_Manager, 27
  Biddy_Nand, 19
  Biddy_NodeNumberPlain, 25
  Biddy_NodeTableAdd, 24
  Biddy_NodeTableBlockNumber, 24
  Biddy_NodeTableCompare, 24
  Biddy_NodeTableFOA, 23
  Biddy_NodeTableGarbage, 24
  Biddy_NodeTableGenerated, 24
  Biddy_NodeTableMax, 23
  Biddy_NodeTableNum, 23
  Biddy_NodeTableNumF, 23
  Biddy_NodeTableNumVar, 23
  Biddy_NodeTableSiftingNumber, 24
  Biddy_NodeTableSize, 23
  Biddy_NodeTableSwapNumber, 24
  Biddy_NodeVarNumber, 25
  Biddy_Nor, 19
  Biddy_Not, 17
  Biddy_NotCond, 17
  Biddy_Or, 19
  Biddy_PrintInfo, 26
  Biddy_Purge, 21
  Biddy_PurgeAndReorder, 21
  Biddy_ReadMemoryInUse, 25
  Biddy_Refresh, 21
  Biddy_Regular, 17
  Biddy_Replace, 21
  Biddy_ResetVariablesValue, 18
  Biddy_Restrict, 20
  Biddy_SetMark, 16
  Biddy_SetVariableValue, 18
  Biddy_Sifting, 23
  Biddy_Simplify, 21
  Biddy_String, 27
  Biddy_String2Variable, 17
  Biddy_Support, 21
  Biddy_SwapWithHigher, 22
  Biddy_SwapWithLower, 23
  Biddy_UnivAbstract, 20
  Biddy_Variable, 27
<table>
<thead>
<tr>
<th>Function</th>
<th>File</th>
<th>Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>biddyStat.c, 65</td>
<td>Biddy_Managed_NodeTableFOA</td>
<td></td>
</tr>
<tr>
<td>biddyStat.c, 65</td>
<td>Biddy_Managed_NodeTableGarbage</td>
<td></td>
</tr>
<tr>
<td>biddyStat.c, 65</td>
<td>Biddy_Managed_NodeTableGenerated</td>
<td></td>
</tr>
<tr>
<td>biddyStat.c, 66</td>
<td>Biddy_Managed_NodeTableMax</td>
<td></td>
</tr>
<tr>
<td>biddyStat.c, 64</td>
<td>Biddy_Managed_NodeTableNum</td>
<td></td>
</tr>
<tr>
<td>biddyStat.c, 64</td>
<td>Biddy_Managed_NodeTableNumF</td>
<td></td>
</tr>
<tr>
<td>biddyStat.c, 64</td>
<td>Biddy_Managed_NodeTableNumVar</td>
<td></td>
</tr>
<tr>
<td>biddyStat.c, 68</td>
<td>Biddy_Managed_NodeTableSiftingNumber</td>
<td></td>
</tr>
<tr>
<td>biddyStat.c, 63</td>
<td>Biddy_Managed_NodeTableSize</td>
<td></td>
</tr>
<tr>
<td>biddyStat.c, 68</td>
<td>Biddy_Managed_NodeTableSwapNumber</td>
<td></td>
</tr>
<tr>
<td>biddyStat.c, 72</td>
<td>Biddy_Managed NguyTableNumF</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 47</td>
<td>Biddy_Managed Nor</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 47</td>
<td>Biddy_Managed Or</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 48</td>
<td>Biddy_Managed PrintInfo</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 59</td>
<td>Biddy_Managed Purge</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 59</td>
<td>Biddy_Managed PurgeAndReorder</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 73</td>
<td>Biddy_Managed ReadMemoryInUse</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 59</td>
<td>Biddy_Managed Refresh</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 53</td>
<td>Biddy_Managed Replace</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 42</td>
<td>Biddy_Managed ResetVariablesValue</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 48</td>
<td>Biddy_Managed Restrict</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 42</td>
<td>Biddy_Managed SetVariableValue</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 58</td>
<td>Biddy_Managed Sifting</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 52</td>
<td>Biddy_Managed Simplify</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 52</td>
<td>Biddy_Managed Support</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 58</td>
<td>Biddy_Managed SwapWithHigher</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 58</td>
<td>Biddy_Managed SwapWithLower</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 51</td>
<td>Biddy_Managed UnivAbstract</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 52</td>
<td>Biddy_Managed VariableNumber</td>
<td></td>
</tr>
<tr>
<td>biddyStat.c, 72</td>
<td>Biddy_Managed VariableTableNum</td>
<td></td>
</tr>
<tr>
<td>biddyStat.c, 62</td>
<td>Biddy_Managed WriteBDD</td>
<td></td>
</tr>
<tr>
<td>biddyInOut.c, 30</td>
<td>Biddy_Managed WriteDot</td>
<td></td>
</tr>
<tr>
<td>biddyInOut.c, 30</td>
<td>Biddy_Managed WriteTable</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 40</td>
<td>Biddy_Managed Xnor</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 47</td>
<td>Biddy_Managed_Xor</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 27</td>
<td>Biddy_Manager</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 40</td>
<td>Biddy_NOT</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 19</td>
<td>Biddy_Nand</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 62</td>
<td>Biddy_NodeAvgLevel</td>
<td></td>
</tr>
<tr>
<td>biddyStat.c, 62</td>
<td>Biddy_NodeMaxLevel</td>
<td></td>
</tr>
<tr>
<td>biddyStat.c, 62</td>
<td>Biddy_NodeNumber</td>
<td></td>
</tr>
<tr>
<td>biddyStat.c, 62</td>
<td>Biddy_NodeNumberPlain</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 25</td>
<td>Biddy_NodeRepair</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 39</td>
<td>Biddy_NodeSelect</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 39</td>
<td>Biddy_NodeTableAdd</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 24</td>
<td>Biddy_NodeTableBlockNumber</td>
<td></td>
</tr>
<tr>
<td>biddy.Main.c, 24</td>
<td>Biddy_NodeTableCompare</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 24</td>
<td>Biddy_NodeTableFOA</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 23</td>
<td>Biddy_NodeTableGarbage</td>
<td></td>
</tr>
<tr>
<td>biddy.Main.c, 23</td>
<td>Biddy_NodeTableGenerated</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 23</td>
<td>Biddy_NodeTableMax</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 23</td>
<td>Biddy_NodeTableNum</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 23</td>
<td>Biddy_NodeTableNumF</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 23</td>
<td>Biddy_NodeTableNumVar</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 23</td>
<td>Biddy_NodeTableSiftingNumber</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 23</td>
<td>Biddy_NodeTableSize</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 24</td>
<td>Biddy_NodeTableSwapNumber</td>
<td></td>
</tr>
<tr>
<td>biddyMain.c, 24</td>
<td>Biddy_NodeVarNumber</td>
<td></td>
</tr>
</tbody>
</table>
INDEX

biddy.h, 25
Biddy_Nor
  biddy.h, 19
Biddy_Not
  biddy.h, 17
Biddy_NotCond
  biddy.h, 17
Biddy_Or
  biddy.h, 19
Biddy_PrintInfo
  biddy.h, 26
Biddy_Purge
  biddy.h, 21
Biddy_PurgeAndReorder
  biddy.h, 21
Biddy_ReadMemoryInUse
  biddy.h, 25
Biddy_Refresh
  biddy.h, 21
Biddy-Regular
  biddy.h, 17
Biddy_Replace
  biddy.h, 21
Biddy_ResetVariablesValue
  biddy.h, 18
Biddy_Restrict
  biddy.h, 20
Biddy_SelectNode
  biddyMain.c, 37
Biddy_SetMark
  biddy.h, 16
Biddy_SetVariableValue
  biddy.h, 18
Biddy_Sifting
  biddy.h, 23
Biddy_Simplify
  biddy.h, 21
Biddy_String
  biddy.h, 27
Biddy_String2Variable
  biddy.h, 17
Biddy_Support
  biddy.h, 21
Biddy_SwapWithHigher
  biddy.h, 22
Biddy_SwapWithLower
  biddy.h, 23
Biddy_TransferMark
  biddyMain.c, 40
Biddy_UnivAbstract
  biddy.h, 20
Biddy_Variable
  biddy.h, 27
Biddy_VariableNumber
  biddy.h, 25
Biddy_VariableTableNum
  biddy.h, 23
Biddy_WriteBDD

biddy.h, 26
Biddy_WriteBDDx
  biddy.h, 26
Biddy_WriteDot
  biddy.h, 26
Biddy_WriteDotx
  biddy.h, 26
Biddy_WriteTable
  biddy.h, 27
Biddy_Xnor
  biddy.h, 19
Biddy_Xor
  biddy.h, 19
biddyInOut.c, 28
Biddy_Managed_Eval0, 29
Biddy_Managed_Eval1, 29
Biddy_Managed_Eval2, 29
Biddy_Managed_WriteBDD, 30
Biddy_Managed_WriteDot, 30
Biddy_Managed_WriteTable, 30
biddyInt.h, 31
biddyMain.c, 31
Biddy_About, 35
Biddy_DeselectNode, 38
Biddy_Exit, 35
Biddy_GetElse, 36
Biddy_GetThen, 36
Biddy_GetTopVariable, 36
Biddy_Init, 35
Biddy_IsSelected, 38
Biddy_Managed_A, 50
Biddy_Managed_AddCache, 55
Biddy_Managed_AddFormula, 56
Biddy_Managed_AddVariable, 43
Biddy_Managed_AddVariableAbove, 44
Biddy_Managed_AddVariableBelow, 43
Biddy_Managed_And, 46
Biddy_Managed_AndAbstract, 51
Biddy_Managed_ClearAll, 55
Biddy_Managed_Compose, 49
Biddy_Managed_Constrain, 52
Biddy_Managed_DeleteAllFormula, 56
Biddy_Managed_DeleteIthFormula, 56
Biddy_Managed_E, 49
Biddy_Managed_ExistAbstract, 51
Biddy_Managed_FindFormula, 55
Biddy_Managed_FoaNode, 45
Biddy_Managed_FoaVariable, 44
Biddy_Managed_Garbage, 53
Biddy_Managed_GetConstantOne, 41
Biddy_Managed_GetConstantZero, 40
Biddy_Managed_GetIthFormulaName, 57
Biddy_Managed_GetTopVariableChar, 42
Biddy_Managed_GetTopVariableName, 41
Biddy_Managed_GetVariableName, 41
Biddy_Managed_ITE, 45
Biddy_Managed_Intersect, 48
Biddy_Managed_IsOK, 53
Biddy_Managed_IsSmaller, 42
Biddy_Managed_IsVariableDependent, 50
Biddy_Managed_Leq, 48
Biddy_Managed_Nand, 47
Biddy_Managed_Nor, 47
Biddy_Managed_Or, 46
Biddy_Managed_Purge, 59
Biddy_Managed_PurgeAndReorder, 59
Biddy_Managed_Refresh, 59
Biddy_Managed_Replace, 53
Biddy_Managed_ResetVariablesValue, 42
Biddy_Managed_Restrict, 48
Biddy_Managed_SetVariableValue, 42
Biddy_Managed_Sifting, 58
Biddy_Managed_Simplify, 52
Biddy_Managed_Support, 52
Biddy_Managed_SwapWithHigher, 58
Biddy_Managed_SwapWithLower, 58
Biddy_Managed_UnivAbstract, 51
Biddy_Managed_Xnor, 48
Biddy_Managed_Xor, 47
Biddy_NOT, 40
Biddy_NodeRepair, 39
Biddy_NodeSelect, 39
Biddy_SelectNode, 37
Biddy_TransferMark, 40
biddyStat.c, 60
  Biddy_Managed_CountMinterm, 73
  Biddy_Managed_IteCacheFind, 71
  Biddy_Managed_IteCacheOverwrite, 71
  Biddy_Managed_IteCacheSearch, 70
  Biddy_Managed_ListAvgLength, 70
  Biddy_Managed_ListMaxLength, 69
  Biddy_Managed_ListUsed, 69
  Biddy_Managed_NodeNumberPlain, 72
  Biddy_Managed_NodeTableAdd, 66
  Biddy_Managed_NodeTableBlockNumber, 67
  Biddy_Managed_NodeTableCompare, 65
  Biddy_Managed_NodeTableFOA, 65
  Biddy_Managed_NodeTableGarbage, 66
  Biddy_Managed_NodeTableGenerated, 67
  Biddy_Managed_NodeTableMax, 63
  Biddy_Managed_NodeTableNum, 64
  Biddy_Managed_NodeTableNumF, 64
  Biddy_Managed_NodeTableNumVar, 64
  Biddy_Managed_NodeTableSiftingNumber, 68
  Biddy_Managed_NodeTableSize, 63
  Biddy_Managed_NodeTableSwapNumber, 68
  Biddy_Managed_NodeVarNumber, 72
  Biddy_Managed_PrintInfo, 74
  Biddy_Managed_ReadMemoryInUse, 73
  Biddy_Managed_VariableNumber, 72
  Biddy_Managed_VariableTableNum, 62
  Biddy_NodeAvgLevel, 62
  Biddy_NodeMaxLevel, 62
  Biddy_NodeNumber, 62