

Biddy

1.7.4

Generated by Doxygen 1.8.11



# Contents

- 1 USER MANUAL** **1**
  
- 2 Data Structure Index** **13**
  - 2.1 Data Structures . . . . . 13
  
- 3 File Index** **15**
  - 3.1 File List . . . . . 15
  
- 4 Data Structure Documentation** **17**
  - 4.1 Biddy\_XY Struct Reference . . . . . 17
    - 4.1.1 Detailed Description . . . . . 17
  
- 5 File Documentation** **19**
  - 5.1 biddy.h File Reference . . . . . 19
    - 5.1.1 Detailed Description . . . . . 23
    - 5.1.2 Macro Definition Documentation . . . . . 23
      - 5.1.2.1 Biddy\_IsNull . . . . . 23
      - 5.1.2.2 Biddy\_IsTerminal . . . . . 23
      - 5.1.2.3 Biddy\_IsEqvPointer . . . . . 24
      - 5.1.2.4 Biddy\_GetMark . . . . . 24
      - 5.1.2.5 Biddy\_SetMark . . . . . 24
      - 5.1.2.6 Biddy\_ClearMark . . . . . 24
      - 5.1.2.7 Biddy\_InvertMark . . . . . 24
      - 5.1.2.8 Biddy\_Inv . . . . . 24
      - 5.1.2.9 Biddy\_InvCond . . . . . 24

---

5.1.2.10	Biddy_Regular	24
5.1.2.11	Biddy_Complement	25
5.1.2.12	Biddy_GetTag	25
5.1.2.13	Biddy_SetTag	25
5.1.2.14	Biddy_ClearTag	25
5.1.2.15	Biddy_Untagged	25
5.1.2.16	Biddy_Init	25
5.1.2.17	Biddy_Exit	25
5.1.2.18	Biddy_GetManagerType	25
5.1.2.19	Biddy_GetManagerName	26
5.1.2.20	Biddy_SetManagerParameters	26
5.1.2.21	Biddy_Managed_GetThen	26
5.1.2.22	Biddy_Managed_GetElse	26
5.1.2.23	Biddy_Managed_GetTopVariable	26
5.1.2.24	Biddy_IsEqv	26
5.1.2.25	Biddy_SelectNode	26
5.1.2.26	Biddy_DeselectNode	27
5.1.2.27	Biddy_IsSelected	27
5.1.2.28	Biddy_SelectFunction	27
5.1.2.29	Biddy_DeselectAll	27
5.1.2.30	Biddy_GetTerminal	27
5.1.2.31	Biddy_GetConstantZero	27
5.1.2.32	Biddy_GetConstantOne	27
5.1.2.33	Biddy_GetBaseSet	27
5.1.2.34	Biddy_GetVariable	28
5.1.2.35	Biddy_GetLowestVariable	28
5.1.2.36	Biddy_GetlthVariable	28
5.1.2.37	Biddy_GetPrevVariable	28
5.1.2.38	Biddy_GetNextVariable	28
5.1.2.39	Biddy_GetVariableEdge	28

---

5.1.2.40	Biddy_GetElementEdge	28
5.1.2.41	Biddy_GetVariableName	28
5.1.2.42	Biddy_GetTopVariableEdge	29
5.1.2.43	Biddy_GetTopVariableName	29
5.1.2.44	Biddy_GetTopVariableChar	29
5.1.2.45	Biddy_ResetVariablesValue	29
5.1.2.46	Biddy_SetVariableValue	29
5.1.2.47	Biddy_GetVariableValue	29
5.1.2.48	Biddy_ClearVariablesData	29
5.1.2.49	Biddy_SetVariableData	29
5.1.2.50	Biddy_GetVariableData	30
5.1.2.51	Biddy_IsSmaller	30
5.1.2.52	Biddy_IsLowest	30
5.1.2.53	Biddy_IsHighest	30
5.1.2.54	Biddy_FoaVariable	30
5.1.2.55	Biddy_ChangeVariableName	30
5.1.2.56	Biddy_AddVariableByName	30
5.1.2.57	Biddy_AddElementByName	30
5.1.2.58	Biddy_AddVariableBelow	31
5.1.2.59	Biddy_AddVariableAbove	31
5.1.2.60	Biddy_TransferMark	31
5.1.2.61	Biddy_IncTag	31
5.1.2.62	Biddy_TaggedFoaNode	31
5.1.2.63	Biddy_IsOK	31
5.1.2.64	Biddy_GC	31
5.1.2.65	Biddy_Clean	32
5.1.2.66	Biddy_Purge	32
5.1.2.67	Biddy_PurgeAndReorder	32
5.1.2.68	Biddy_Refresh	32
5.1.2.69	Biddy_AddCache	32

---

---

5.1.2.70	Biddy_AddFormula	32
5.1.2.71	Biddy_FindFormula	33
5.1.2.72	Biddy_DeleteFormula	33
5.1.2.73	Biddy_DeletelthFormula	33
5.1.2.74	Biddy_GetlthFormula	33
5.1.2.75	Biddy_GetlthFormulaName	33
5.1.2.76	Biddy_SwapWithHigher	33
5.1.2.77	Biddy_SwapWithLower	33
5.1.2.78	Biddy_Sifting	33
5.1.2.79	Biddy_MinimizeBDD	34
5.1.2.80	Biddy_MaximizeBDD	34
5.1.2.81	Biddy_Copy	34
5.1.2.82	Biddy_CopyFormula	34
5.1.2.83	Biddy_Eval	34
5.1.2.84	Biddy_Not	34
5.1.2.85	Biddy_ITE	34
5.1.2.86	Biddy_And	35
5.1.2.87	Biddy_Or	35
5.1.2.88	Biddy_Nand	35
5.1.2.89	Biddy_Nor	35
5.1.2.90	Biddy_Xor	35
5.1.2.91	Biddy_Xnor	35
5.1.2.92	Biddy_Leq	35
5.1.2.93	Biddy_Gt	36
5.1.2.94	Biddy_IsLeq	36
5.1.2.95	Biddy_Restrict	36
5.1.2.96	Biddy_Compose	36
5.1.2.97	Biddy_E	36
5.1.2.98	Biddy_A	36
5.1.2.99	Biddy_IsVariableDependent	36

---

5.1.2.100 Bidy_ExistAbstract . . . . .	36
5.1.2.101 Bidy_UnivAbstract . . . . .	37
5.1.2.102 Bidy_AndAbstract . . . . .	37
5.1.2.103 Bidy_Constrain . . . . .	37
5.1.2.104 Bidy_Simplify . . . . .	37
5.1.2.105 Bidy_Support . . . . .	37
5.1.2.106 Bidy_ReplaceByKeyword . . . . .	37
5.1.2.107 Bidy_Change . . . . .	37
5.1.2.108 Bidy_Subset . . . . .	38
5.1.2.109 Bidy_CreateMinterm . . . . .	38
5.1.2.110 Bidy_CreateFunction . . . . .	38
5.1.2.111 Bidy_RandomFunction . . . . .	38
5.1.2.112 Bidy_RandomSet . . . . .	38
5.1.2.113 Bidy_CountNodes . . . . .	38
5.1.2.114 Bidy_Managed_MaxLevel . . . . .	38
5.1.2.115 Bidy_Managed_AvgLevel . . . . .	38
5.1.2.116 Bidy_VariableTableNum . . . . .	39
5.1.2.117 Bidy_NodeTableSize . . . . .	39
5.1.2.118 Bidy_NodeTableBlockNumber . . . . .	39
5.1.2.119 Bidy_NodeTableGenerated . . . . .	39
5.1.2.120 Bidy_NodeTableMax . . . . .	39
5.1.2.121 Bidy_NodeTableNum . . . . .	39
5.1.2.122 Bidy_NodeTableNumVar . . . . .	39
5.1.2.123 Bidy_NodeTableResizeNumber . . . . .	39
5.1.2.124 Bidy_NodeTableFoaNumber . . . . .	40
5.1.2.125 Bidy_NodeTableFindNumber . . . . .	40
5.1.2.126 Bidy_NodeTableCompareNumber . . . . .	40
5.1.2.127 Bidy_NodeTableAddNumber . . . . .	40
5.1.2.128 Bidy_NodeTableGCNumber . . . . .	40
5.1.2.129 Bidy_NodeTableGCTime . . . . .	40

---

---

5.1.2.130 Bidly_NodeTableGCObsoleteNumber	40
5.1.2.131 Bidly_NodeTableSwapNumber	40
5.1.2.132 Bidly_NodeTableSiftingNumber	41
5.1.2.133 Bidly_NodeTableDRTime	41
5.1.2.134 Bidly_NodeTableITENumber	41
5.1.2.135 Bidly_NodeTableITERRecursiveNumber	41
5.1.2.136 Bidly_NodeTableANDORNumber	41
5.1.2.137 Bidly_NodeTableANDORRecursiveNumber	41
5.1.2.138 Bidly_NodeTableXORNumber	41
5.1.2.139 Bidly_NodeTableXORRecursiveNumber	42
5.1.2.140 Bidly_FormulaTableNum	42
5.1.2.141 Bidly_ListUsed	42
5.1.2.142 Bidly_ListMaxLength	42
5.1.2.143 Bidly_ListAvgLength	42
5.1.2.144 Bidly_OPCCacheSearch	42
5.1.2.145 Bidly_OPCCacheFind	42
5.1.2.146 Bidly_OPCCacheInsert	43
5.1.2.147 Bidly_OPCCacheOverwrite	43
5.1.2.148 Bidly_CountNodesPlain	43
5.1.2.149 Bidly_DependentVariableNumber	43
5.1.2.150 Bidly_CountComplementedEdges	43
5.1.2.151 Bidly_CountPaths	43
5.1.2.152 Bidly_CountMinterms	43
5.1.2.153 Bidly_DensityOfFunction	43
5.1.2.154 Bidly_DensityOfBDD	44
5.1.2.155 Bidly_ReadMemoryInUse	44
5.1.2.156 Bidly_PrintInfo	44
5.1.2.157 Bidly_Eval0	44
5.1.2.158 Bidly_Eval1x	44
5.1.2.159 Bidly_Eval2	44



---

5.1.2.160	Biddy_ReadVerilogFile	44
5.1.2.161	Biddy_PrintfBDD	44
5.1.2.162	Biddy_WriteBDD	45
5.1.2.163	Biddy_PrintfTable	45
5.1.2.164	Biddy_WriteTable	45
5.1.2.165	Biddy_PrintfSOP	45
5.1.2.166	Biddy_WriteSOP	45
5.1.2.167	Biddy_WriteDot	45
5.1.2.168	Biddy_WriteBddview	45
5.1.3	Typedef Documentation	46
5.1.3.1	Biddy_Boolean	46
5.1.3.2	Biddy_String	46
5.1.3.3	Biddy_Manager	46
5.1.3.4	Biddy_Cache	46
5.1.3.5	Biddy_Variable	46
5.1.3.6	Biddy_Edge	46
5.1.3.7	Biddy_GCFunction	47
5.1.3.8	Biddy_LookupFunction	47
5.2	biddyInOut.c File Reference	47
5.2.1	Detailed Description	48
5.2.2	Function Documentation	48
5.2.2.1	Biddy_Managed_Eval0(Biddy_Manager MNG, Biddy_String s)	48
5.2.2.2	Biddy_Managed_Eval1x(Biddy_Manager MNG, Biddy_String s, Biddy_Lookup↔ Function lf)	49
5.2.2.3	Biddy_Managed_Eval2(Biddy_Manager MNG, Biddy_String boolFunc)	49
5.2.2.4	Biddy_Managed_ReadVerilogFile(Biddy_Manager MNG, const char filename[], Biddy_String prefix)	49
5.2.2.5	Biddy_Managed_PrintfBDD(Biddy_Manager MNG, Biddy_Edge f)	50
5.2.2.6	Biddy_Managed_WriteBDD(Biddy_Manager MNG, const char filename[], Biddy_Edge f, Biddy_String label)	50
5.2.2.7	Biddy_Managed_PrintfTable(Biddy_Manager MNG, Biddy_Edge f)	50

5.2.2.8	Bidly_Managed_WriteTable(Bidly_Manager MNG, const char filename[], Bidly_Edge f) . . . . .	51
5.2.2.9	Bidly_Managed_PrintfSOP(Bidly_Manager MNG, Bidly_Edge f) . . . . .	51
5.2.2.10	Bidly_Managed_WriteSOP(Bidly_Manager MNG, const char filename[], Bidly_Edge f) . . . . .	51
5.2.2.11	Bidly_Managed_WriteDot(Bidly_Manager MNG, const char filename[], Bidly↔_Edge f, const char label[], int id, Bidly_Boolean cudd) . . . . .	52
5.2.2.12	Bidly_Managed_WriteBddview(Bidly_Manager MNG, const char filename[], Bidly_Edge f, const char label[], Bidly_XY *table) . . . . .	52
5.3	bidlyInt.h File Reference . . . . .	53
5.3.1	Detailed Description . . . . .	53
5.4	bidlyMain.c File Reference . . . . .	53
5.4.1	Detailed Description . . . . .	57
5.4.2	Function Documentation . . . . .	58
5.4.2.1	Bidly_InitMNG(Bidly_Manager *mng, int gddtype) . . . . .	58
5.4.2.2	Bidly_ExitMNG(Bidly_Manager *mng) . . . . .	59
5.4.2.3	Bidly_About() . . . . .	59
5.4.2.4	Bidly_Managed_GetManagerType(Bidly_Manager MNG) . . . . .	59
5.4.2.5	Bidly_Managed_GetManagerName(Bidly_Manager MNG) . . . . .	60
5.4.2.6	Bidly_Managed_SetManagerParameters(Bidly_Manager MNG, float gcr, float gcrF, float gcrX, float rr, float rrF, float rrX, float st, float cst) . . . . .	60
5.4.2.7	Bidly_GetThen(Bidly_Edge fun) . . . . .	61
5.4.2.8	Bidly_GetElse(Bidly_Edge fun) . . . . .	61
5.4.2.9	Bidly_GetTopVariable(Bidly_Edge fun) . . . . .	62
5.4.2.10	Bidly_Managed_IsEqv(Bidly_Manager MNG1, Bidly_Edge f1, Bidly_Manager MNG2, Bidly_Edge f2) . . . . .	62
5.4.2.11	Bidly_Managed_SelectNode(Bidly_Manager MNG, Bidly_Edge f) . . . . .	63
5.4.2.12	Bidly_Managed_DeselectNode(Bidly_Manager MNG, Bidly_Edge f) . . . . .	63
5.4.2.13	Bidly_Managed_IsSelected(Bidly_Manager MNG, Bidly_Edge f) . . . . .	64
5.4.2.14	Bidly_Managed_SelectFunction(Bidly_Manager MNG, Bidly_Edge f) . . . . .	64
5.4.2.15	Bidly_Managed_DeselectAll(Bidly_Manager MNG) . . . . .	65
5.4.2.16	Bidly_Managed_GetTerminal(Bidly_Manager MNG) . . . . .	65
5.4.2.17	Bidly_Managed_GetConstantZero(Bidly_Manager MNG) . . . . .	66

---

5.4.2.18	Biddy_Managed_GetConstantOne(Biddy_Manager MNG)	66
5.4.2.19	Biddy_Managed_GetBaseSet(Biddy_Manager MNG)	67
5.4.2.20	Biddy_Managed_GetVariable(Biddy_Manager MNG, Biddy_String x)	67
5.4.2.21	Biddy_Managed_GetLowestVariable(Biddy_Manager MNG)	68
5.4.2.22	Biddy_Managed_GetIthVariable(Biddy_Manager MNG, Biddy_Variable i)	68
5.4.2.23	Biddy_Managed_GetPrevVariable(Biddy_Manager MNG, Biddy_Variable v)	68
5.4.2.24	Biddy_Managed_GetNextVariable(Biddy_Manager MNG, Biddy_Variable v)	69
5.4.2.25	Biddy_Managed_GetVariableEdge(Biddy_Manager MNG, Biddy_Variable v)	69
5.4.2.26	Biddy_Managed_GetElementEdge(Biddy_Manager MNG, Biddy_Variable v)	70
5.4.2.27	Biddy_Managed_GetVariableName(Biddy_Manager MNG, Biddy_Variable v)	70
5.4.2.28	Biddy_Managed_GetTopVariableEdge(Biddy_Manager MNG, Biddy_Edge f)	71
5.4.2.29	Biddy_Managed_GetTopVariableName(Biddy_Manager MNG, Biddy_Edge f)	71
5.4.2.30	Biddy_Managed_GetTopVariableChar(Biddy_Manager MNG, Biddy_Edge f)	72
5.4.2.31	Biddy_Managed_ResetVariablesValue(Biddy_Manager MNG)	72
5.4.2.32	Biddy_Managed_SetVariableValue(Biddy_Manager MNG, Biddy_Variable v, Biddy_Edge f)	73
5.4.2.33	Biddy_Managed_GetVariableValue(Biddy_Manager MNG, Biddy_Variable v)	73
5.4.2.34	Biddy_Managed_ClearVariablesData(Biddy_Manager MNG)	74
5.4.2.35	Biddy_Managed_SetVariableData(Biddy_Manager MNG, Biddy_Variable v, void *x)	74
5.4.2.36	Biddy_Managed_GetVariableData(Biddy_Manager MNG, Biddy_Variable v)	75
5.4.2.37	Biddy_Managed_IsSmaller(Biddy_Manager MNG, Biddy_Variable fv, Biddy_↔ Variable gv)	75
5.4.2.38	Biddy_Managed_IsLowest(Biddy_Manager MNG, Biddy_Variable v)	75
5.4.2.39	Biddy_Managed_IsHighest(Biddy_Manager MNG, Biddy_Variable v)	76
5.4.2.40	Biddy_Managed_FoaVariable(Biddy_Manager MNG, Biddy_String x, Biddy_↔ Boolean varelem)	77
5.4.2.41	Biddy_Managed_ChangeVariableName(Biddy_Manager MNG, Biddy_Variable v, Biddy_String x)	77
5.4.2.42	Biddy_Managed_AddVariableByName(Biddy_Manager MNG, Biddy_String x)	78
5.4.2.43	Biddy_Managed_AddElementByName(Biddy_Manager MNG, Biddy_String x)	78
5.4.2.44	Biddy_Managed_AddVariableBelow(Biddy_Manager MNG, Biddy_Variable v)	79
5.4.2.45	Biddy_Managed_AddVariableAbove(Biddy_Manager MNG, Biddy_Variable v)	79

---

5.4.2.46	Bidly_Managed_TransferMark(Bidly_Manager MNG, Bidly_Edge f, Bidly_↔ Boolean mark, Bidly_Boolean leftright) . . . . .	80
5.4.2.47	Bidly_Managed_IncTag(Bidly_Manager MNG, Bidly_Edge f) . . . . .	80
5.4.2.48	Bidly_Managed_TaggedFoaNode(Bidly_Manager MNG, Bidly_Variable v, Bidly_Edge pf, Bidly_Edge pt, Bidly_Variable ptag, Bidly_Boolean garbage↔ Allowed) . . . . .	81
5.4.2.49	Bidly_Managed_IsOK(Bidly_Manager MNG, Bidly_Edge f) . . . . .	82
5.4.2.50	Bidly_Managed_GC(Bidly_Manager MNG, Bidly_Variable targetLT, Bidly_↔ Variable targetGEQ, Bidly_Boolean purge, Bidly_Boolean total) . . . . .	82
5.4.2.51	Bidly_Managed_Clean(Bidly_Manager MNG) . . . . .	83
5.4.2.52	Bidly_Managed_Purge(Bidly_Manager MNG) . . . . .	84
5.4.2.53	Bidly_Managed_PurgeAndReorder(Bidly_Manager MNG, Bidly_Edge f, Bidly_Boolean converge) . . . . .	85
5.4.2.54	Bidly_Managed_Refresh(Bidly_Manager MNG, Bidly_Edge f) . . . . .	85
5.4.2.55	Bidly_Managed_AddCache(Bidly_Manager MNG, Bidly_GCFunction gc) . . . . .	85
5.4.2.56	Bidly_Managed_AddFormula(Bidly_Manager MNG, Bidly_String x, Bidly_↔ Edge f, int c) . . . . .	86
5.4.2.57	Bidly_Managed_FindFormula(Bidly_Manager MNG, Bidly_String x, unsigned int *idx, Bidly_Edge *f) . . . . .	87
5.4.2.58	Bidly_Managed_DeleteFormula(Bidly_Manager MNG, Bidly_String x) . . . . .	88
5.4.2.59	Bidly_Managed_DeletelthFormula(Bidly_Manager MNG, unsigned int i) . . . . .	88
5.4.2.60	Bidly_Managed_GetlthFormula(Bidly_Manager MNG, unsigned int i) . . . . .	88
5.4.2.61	Bidly_Managed_GetlthFormulaName(Bidly_Manager MNG, unsigned int i) . . . . .	89
5.4.2.62	Bidly_Managed_SwapWithHigher(Bidly_Manager MNG, Bidly_Variable v) . . . . .	89
5.4.2.63	Bidly_Managed_SwapWithLower(Bidly_Manager MNG, Bidly_Variable v) . . . . .	90
5.4.2.64	Bidly_Managed_Sifting(Bidly_Manager MNG, Bidly_Edge f, Bidly_Boolean converge) . . . . .	91
5.4.2.65	Bidly_Managed_MinimizeBDD(Bidly_Manager MNG, Bidly_String name) . . . . .	91
5.4.2.66	Bidly_Managed_MaximizeBDD(Bidly_Manager MNG, Bidly_String name) . . . . .	92
5.5	bidlyMainLegacy.c File Reference . . . . .	92
5.5.1	Detailed Description . . . . .	97
5.5.2	Function Documentation . . . . .	97
5.5.2.1	Bidly_InitMNG(Bidly_Manager *mng, int gddtype) . . . . .	97
5.5.2.2	Bidly_ExitMNG(Bidly_Manager *mng) . . . . .	98

5.5.2.3	Biddy_About()	98
5.5.2.4	Biddy_Managed_GetManagerType(Biddy_Manager MNG)	98
5.5.2.5	Biddy_Managed_SetManagerParameters(Biddy_Manager MNG, float gcr, float gcrF, float gcrX, float rr, float rrF, float rrX, float st, float cst)	99
5.5.2.6	Biddy_GetThen(Biddy_Edge fun)	99
5.5.2.7	Biddy_GetElse(Biddy_Edge fun)	100
5.5.2.8	Biddy_GetTopVariable(Biddy_Edge fun)	100
5.5.2.9	Biddy_Managed_IsEqv(Biddy_Manager MNG1, Biddy_Edge f1, Biddy_Manager MNG2, Biddy_Edge f2)	100
5.5.2.10	Biddy_Managed_SelectNode(Biddy_Manager MNG, Biddy_Edge f)	101
5.5.2.11	Biddy_Managed_DeselectNode(Biddy_Manager MNG, Biddy_Edge f)	101
5.5.2.12	Biddy_Managed_IsSelected(Biddy_Manager MNG, Biddy_Edge f)	102
5.5.2.13	Biddy_Managed_SelectFunction(Biddy_Manager MNG, Biddy_Edge f)	102
5.5.2.14	Biddy_Managed_DeselectAll(Biddy_Manager MNG)	103
5.5.2.15	Biddy_Managed_GetTerminal(Biddy_Manager MNG)	103
5.5.2.16	Biddy_Managed_GetConstantZero(Biddy_Manager MNG)	104
5.5.2.17	Biddy_Managed_GetConstantOne(Biddy_Manager MNG)	104
5.5.2.18	Biddy_Managed_GetBaseSet(Biddy_Manager MNG)	105
5.5.2.19	Biddy_Managed_GetVariable(Biddy_Manager MNG, Biddy_String x)	105
5.5.2.20	Biddy_Managed_GetPrevVariable(Biddy_Manager MNG, Biddy_Variable v)	106
5.5.2.21	Biddy_Managed_GetNextVariable(Biddy_Manager MNG, Biddy_Variable v)	106
5.5.2.22	Biddy_Managed_GetVariableEdge(Biddy_Manager MNG, Biddy_Variable v)	107
5.5.2.23	Biddy_Managed_GetElementEdge(Biddy_Manager MNG, Biddy_Variable v)	107
5.5.2.24	Biddy_Managed_GetVariableName(Biddy_Manager MNG, Biddy_Variable v)	107
5.5.2.25	Biddy_Managed_GetTopVariableEdge(Biddy_Manager MNG, Biddy_Edge f)	108
5.5.2.26	Biddy_Managed_GetTopVariableName(Biddy_Manager MNG, Biddy_Edge f)	108
5.5.2.27	Biddy_Managed_GetTopVariableChar(Biddy_Manager MNG, Biddy_Edge f)	109
5.5.2.28	Biddy_Managed_ResetVariablesValue(Biddy_Manager MNG)	109
5.5.2.29	Biddy_Managed_SetVariableValue(Biddy_Manager MNG, Biddy_Variable v, Biddy_Edge f)	110
5.5.2.30	Biddy_Managed_IsSmaller(Biddy_Manager MNG, Biddy_Variable fv, Biddy_↔ Variable gv)	110

5.5.2.31	Biddy_Managed_FoaVariable(Biddy_Manager MNG, Biddy_String x, Biddy_↔ Boolean varelem)	111
5.5.2.32	Biddy_Managed_AddVariableByName(Biddy_Manager MNG, Biddy_String x)	112
5.5.2.33	Biddy_Managed_AddElementByName(Biddy_Manager MNG, Biddy_String x)	113
5.5.2.34	Biddy_Managed_AddVariableBelow(Biddy_Manager MNG, Biddy_Variable v)	113
5.5.2.35	Biddy_Managed_AddVariableAbove(Biddy_Manager MNG, Biddy_Variable v)	114
5.5.2.36	Biddy_Managed_TransferMark(Biddy_Manager MNG, Biddy_Edge f, Biddy_↔ Boolean mark, Biddy_Boolean leftright)	114
5.5.2.37	Biddy_Managed_IncTag(Biddy_Manager MNG, Biddy_Edge f)	115
5.5.2.38	Biddy_Managed_TaggedFoaNode(Biddy_Manager MNG, Biddy_Variable v, Biddy_Edge pf, Biddy_Edge pt, Biddy_Variable ptag, Biddy_Boolean garbage↔ Allowed)	115
5.5.2.39	Biddy_Managed_Not(Biddy_Manager MNG, Biddy_Edge f)	116
5.5.2.40	Biddy_Managed_ITE(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g, Biddy_Edge h)	116
5.5.2.41	Biddy_Managed_And(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)	117
5.5.2.42	Biddy_Managed_Or(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)	117
5.5.2.43	Biddy_Managed_Nand(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)	118
5.5.2.44	Biddy_Managed_Nor(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)	118
5.5.2.45	Biddy_Managed_Xor(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)	118
5.5.2.46	Biddy_Managed_Xnor(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)	119
5.5.2.47	Biddy_Managed_Leq(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)	119
5.5.2.48	Biddy_Managed_Gt(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)	119
5.5.2.49	Biddy_Managed_IsLeq(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)	119
5.5.2.50	Biddy_Managed_SubIntersect(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)	120
5.5.2.51	Biddy_Managed_Restrict(Biddy_Manager MNG, Biddy_Edge f, Biddy_Variable v, Biddy_Boolean value)	120
5.5.2.52	Biddy_Managed_Compose(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g, Biddy_Variable v)	121
5.5.2.53	Biddy_Managed_E(Biddy_Manager MNG, Biddy_Edge f, Biddy_Variable v)	121
5.5.2.54	Biddy_Managed_A(Biddy_Manager MNG, Biddy_Edge f, Biddy_Variable v)	121
5.5.2.55	Biddy_Managed_IsVariableDependent(Biddy_Manager MNG, Biddy_Edge f, Biddy_Variable v)	122

5.5.2.56	Biddy_Managed_ExistAbstract(Biddy_Manager MNG, Biddy_Edge f, Biddy_↔ Edge cube) . . . . .	122
5.5.2.57	Biddy_Managed_UnivAbstract(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge cube) . . . . .	122
5.5.2.58	Biddy_Managed_AndAbstract(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g, Biddy_Edge cube) . . . . .	123
5.5.2.59	Biddy_Managed_Constrain(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge c) . . . . .	123
5.5.2.60	Biddy_Managed_Simplify(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge c) . . . . .	123
5.5.2.61	Biddy_Managed_Support(Biddy_Manager MNG, Biddy_Edge f) . . . . .	124
5.5.2.62	Biddy_Managed_Replace(Biddy_Manager MNG, Biddy_Edge f) . . . . .	124
5.5.2.63	Biddy_Managed_Change(Biddy_Manager MNG, Biddy_Edge f, Biddy_Variable v) . . . . .	125
5.5.2.64	Biddy_Managed_Subset(Biddy_Manager MNG, Biddy_Edge f, Biddy_Variable v, Biddy_Boolean value) . . . . .	125
5.5.2.65	Biddy_Managed_IsOK(Biddy_Manager MNG, Biddy_Edge f) . . . . .	125
5.5.2.66	Biddy_Managed_GC(Biddy_Manager MNG, Biddy_Variable target, Biddy_↔ Boolean purge, Biddy_Boolean total) . . . . .	126
5.5.2.67	Biddy_Managed_Clean(Biddy_Manager MNG) . . . . .	127
5.5.2.68	Biddy_Managed_Purge(Biddy_Manager MNG) . . . . .	127
5.5.2.69	Biddy_Managed_PurgeAndReorder(Biddy_Manager MNG, Biddy_Edge f, Biddy_Boolean converge) . . . . .	128
5.5.2.70	Biddy_Managed_Refresh(Biddy_Manager MNG, Biddy_Edge f) . . . . .	128
5.5.2.71	Biddy_Managed_AddCache(Biddy_Manager MNG, Biddy_GCFunction gc) . . . . .	128
5.5.2.72	Biddy_Managed_AddFormula(Biddy_Manager MNG, Biddy_String x, Biddy_↔ Edge f, int c) . . . . .	129
5.5.2.73	Biddy_Managed_FindFormula(Biddy_Manager MNG, Biddy_String x, Biddy_↔ Edge *f) . . . . .	130
5.5.2.74	Biddy_Managed_DeleteFormula(Biddy_Manager MNG, Biddy_String x) . . . . .	130
5.5.2.75	Biddy_Managed_DeletelthFormula(Biddy_Manager MNG, unsigned int i) . . . . .	130
5.5.2.76	Biddy_Managed_GetlthFormula(Biddy_Manager MNG, unsigned int i) . . . . .	131
5.5.2.77	Biddy_Managed_GetlthFormulaName(Biddy_Manager MNG, unsigned int i) . . . . .	131
5.5.2.78	Biddy_Managed_SwapWithHigher(Biddy_Manager MNG, Biddy_Variable v) . . . . .	132
5.5.2.79	Biddy_Managed_SwapWithLower(Biddy_Manager MNG, Biddy_Variable v) . . . . .	132
5.5.2.80	Biddy_Managed_Sifting(Biddy_Manager MNG, Biddy_Edge f, Biddy_Boolean converge) . . . . .	133

5.5.2.81	Biddy_Managed_Random(Biddy_Manager MNG, Biddy_Edge support, double r)	133
5.5.2.82	Biddy_Managed_RandomSet(Biddy_Manager MNG, Biddy_Edge unit, double r)	134
5.6	biddyOp.c File Reference	134
5.6.1	Detailed Description	136
5.6.2	Function Documentation	136
5.6.2.1	Biddy_Managed_Not(Biddy_Manager MNG, Biddy_Edge f)	136
5.6.2.2	Biddy_Managed_ITE(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g, Biddy_Edge h)	137
5.6.2.3	Biddy_Managed_And(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)	137
5.6.2.4	Biddy_Managed_Or(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)	137
5.6.2.5	Biddy_Managed_Nand(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)	138
5.6.2.6	Biddy_Managed_Nor(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)	138
5.6.2.7	Biddy_Managed_Xor(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)	139
5.6.2.8	Biddy_Managed_Xnor(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)	139
5.6.2.9	Biddy_Managed_Leq(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)	139
5.6.2.10	Biddy_Managed_Gt(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)	140
5.6.2.11	Biddy_Managed_IsLeq(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g)	140
5.6.2.12	Biddy_Managed_Restrict(Biddy_Manager MNG, Biddy_Edge f, Biddy_Variable v, Biddy_Boolean value)	140
5.6.2.13	Biddy_Managed_Compose(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g, Biddy_Variable v)	141
5.6.2.14	Biddy_Managed_E(Biddy_Manager MNG, Biddy_Edge f, Biddy_Variable v)	141
5.6.2.15	Biddy_Managed_A(Biddy_Manager MNG, Biddy_Edge f, Biddy_Variable v)	142
5.6.2.16	Biddy_Managed_IsVariableDependent(Biddy_Manager MNG, Biddy_Edge f, Biddy_Variable v)	142
5.6.2.17	Biddy_Managed_ExistAbstract(Biddy_Manager MNG, Biddy_Edge f, Biddy_↔ Edge cube)	142
5.6.2.18	Biddy_Managed_UnivAbstract(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge cube)	143
5.6.2.19	Biddy_Managed_AndAbstract(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge g, Biddy_Edge cube)	143
5.6.2.20	Biddy_Managed_Constrain(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge c)	143
5.6.2.21	Biddy_Managed_Simplify(Biddy_Manager MNG, Biddy_Edge f, Biddy_Edge c)	144



---

5.6.2.22	Bidly_Managed_Support(Bidly_Manager MNG, Bidly_Edge f)	144
5.6.2.23	Bidly_Managed_ReplaceByKeyword(Bidly_Manager MNG, Bidly_Edge f, Bidly_String keyword)	145
5.6.2.24	Bidly_Managed_Change(Bidly_Manager MNG, Bidly_Edge f, Bidly_Variable v)	145
5.6.2.25	Bidly_Managed_Subset(Bidly_Manager MNG, Bidly_Edge f, Bidly_Variable v, Bidly_Boolean value)	145
5.6.2.26	Bidly_Managed_CreateMinterm(Bidly_Manager MNG, Bidly_Edge support, long long unsigned int x)	146
5.6.2.27	Bidly_Managed_CreateFunction(Bidly_Manager MNG, Bidly_Edge support, long long unsigned int x)	146
5.6.2.28	Bidly_Managed_RandomFunction(Bidly_Manager MNG, Bidly_Edge support, double r)	147
5.6.2.29	Bidly_Managed_RandomSet(Bidly_Manager MNG, Bidly_Edge unit, double r)	147
5.7	bidlyStat.c File Reference	147
5.7.1	Detailed Description	149
5.7.2	Function Documentation	150
5.7.2.1	Bidly_Managed_CountNodes(Bidly_Manager MNG, Bidly_Edge f)	150
5.7.2.2	Bidly_MaxLevel(Bidly_Edge f)	151
5.7.2.3	Bidly_AvgLevel(Bidly_Edge f)	151
5.7.2.4	Bidly_Managed_VariableTableNum(Bidly_Manager MNG)	152
5.7.2.5	Bidly_Managed_NodeTableSize(Bidly_Manager MNG)	152
5.7.2.6	Bidly_Managed_NodeTableBlockNumber(Bidly_Manager MNG)	153
5.7.2.7	Bidly_Managed_NodeTableGenerated(Bidly_Manager MNG)	153
5.7.2.8	Bidly_Managed_NodeTableMax(Bidly_Manager MNG)	153
5.7.2.9	Bidly_Managed_NodeTableNum(Bidly_Manager MNG)	153
5.7.2.10	Bidly_Managed_NodeTableNumVar(Bidly_Manager MNG, Bidly_Variable v)	154
5.7.2.11	Bidly_Managed_NodeTableResizeNumber(Bidly_Manager MNG)	154
5.7.2.12	Bidly_Managed_NodeTableFoaNumber(Bidly_Manager MNG)	155
5.7.2.13	Bidly_Managed_NodeTableFindNumber(Bidly_Manager MNG)	155
5.7.2.14	Bidly_Managed_NodeTableCompareNumber(Bidly_Manager MNG)	155
5.7.2.15	Bidly_Managed_NodeTableAddNumber(Bidly_Manager MNG)	156
5.7.2.16	Bidly_Managed_NodeTableGCNumber(Bidly_Manager MNG)	156
5.7.2.17	Bidly_Managed_NodeTableGCTime(Bidly_Manager MNG)	156

5.7.2.18	Biddy_Managed_NodeTableGCObssoleteNumber(Biddy_Manager MNG)	156
5.7.2.19	Biddy_Managed_NodeTableSwapNumber(Biddy_Manager MNG)	157
5.7.2.20	Biddy_Managed_NodeTableSiftingNumber(Biddy_Manager MNG)	157
5.7.2.21	Biddy_Managed_NodeTableDRTTime(Biddy_Manager MNG)	158
5.7.2.22	Biddy_Managed_NodeTableITENumber(Biddy_Manager MNG)	158
5.7.2.23	Biddy_Managed_NodeTableITERRecursiveNumber(Biddy_Manager MNG)	158
5.7.2.24	Biddy_Managed_NodeTableANDORNumber(Biddy_Manager MNG)	158
5.7.2.25	Biddy_Managed_NodeTableANDORRecursiveNumber(Biddy_Manager MNG)	159
5.7.2.26	Biddy_Managed_NodeTableXORNumber(Biddy_Manager MNG)	159
5.7.2.27	Biddy_Managed_NodeTableXORRecursiveNumber(Biddy_Manager MNG)	159
5.7.2.28	Biddy_Managed_FormulaTableNum(Biddy_Manager MNG)	160
5.7.2.29	Biddy_Managed_ListUsed(Biddy_Manager MNG)	160
5.7.2.30	Biddy_Managed_ListMaxLength(Biddy_Manager MNG)	161
5.7.2.31	Biddy_Managed_ListAvgLength(Biddy_Manager MNG)	161
5.7.2.32	Biddy_Managed_OPCacheSearch(Biddy_Manager MNG)	162
5.7.2.33	Biddy_Managed_OPCacheFind(Biddy_Manager MNG)	162
5.7.2.34	Biddy_Managed_OPCacheInsert(Biddy_Manager MNG)	162
5.7.2.35	Biddy_Managed_OPCacheOverwrite(Biddy_Manager MNG)	162
5.7.2.36	Biddy_Managed_CountNodesPlain(Biddy_Manager MNG, Biddy_Edge f)	163
5.7.2.37	Biddy_Managed_DependentVariableNumber(Biddy_Manager MNG, Biddy_Edge f)	163
5.7.2.38	Biddy_Managed_CountComplementedEdges(Biddy_Manager MNG, Biddy_↔ Edge f)	164
5.7.2.39	Biddy_Managed_CountPaths(Biddy_Manager MNG, Biddy_Edge f)	164
5.7.2.40	Biddy_Managed_CountMinterms(Biddy_Manager MNG, Biddy_Edge f, unsigned int nvars)	164
5.7.2.41	Biddy_Managed_DensityOfFunction(Biddy_Manager MNG, Biddy_Edge f, un- signed int nvars)	165
5.7.2.42	Biddy_Managed_DensityOfBDD(Biddy_Manager MNG, Biddy_Edge f, unsigned int nvars)	165
5.7.2.43	Biddy_Managed_ReadMemoryInUse(Biddy_Manager MNG)	166
5.7.2.44	Biddy_Managed_PrintInfo(Biddy_Manager MNG, FILE *f)	166

# Chapter 1

## USER MANUAL

### ### TL;DR

Biddy is a multi-platform academic Binary Decision Diagrams package. It supports plain ROBDDs, ROBDDs with complemented edges, 0-sup-BDDs with complemented edges, and plain Tagged 0-sup-BDDs.

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

Biddy is a library to be included in your C and C++ projects:

```
1 #include "/path/to/biddy.h"
```

To compile Biddy library use "make static" or "make dynamic" or "make debug". There is no configuration script, you should edit Makefiles to adapt system configuration.

```
1 biddy> make dynamic
2 biddy> make clean
```

Dependencies:

- on GNU/Linux, you need libgmp (<https://gmplib.org/>).
- on MS Windows, you need MPIR library (<http://mpir.org/>).

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

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

There are two additional packages included into Biddy distribution:

- bddview is a pure Tcl/Tk script for visualization of BDDs,
- BDD Scout is a demo application demonstrating the capability of Biddy and bddview.

Biddy is free software maintained by Robert Meolic ([robert.meolic@um.si](mailto:robert.meolic@um.si)) at University of Maribor, Slovenia.

Homepage: <http://biddy.meolic.com/>

### ### 1. AN OVERVIEW

Biddy supports ROBDDs as described in "K. S. Brace, R. L. Rudell, R. E. Bryant. Efficient Implementation of a BDD Package. 27th ACM/IEEE DAC, pages 40-45, 1990".

Biddy supports 0-sup-BDDs as described in "S. Minato. Zero-Suppressed BDDs for Set Manipulation in Combinatorial Problems, 30th ACM/IEEE DAC, pages 272-277, 1993".

Biddy supports Tagged 0-sup-BDDs (also denoted TZDDs or TZBDDs) as introduced in "R. Meolic. Implementation aspects of a BDD package supporting general decision diagrams, <https://dk.um.si/lzpisGradiva.php?id=68831>" and described in "T. van Dijk, R. Wille, R. Meolic. Tagged BDDs: Combining Reduction Rules from Different Decision Diagram Types, 17th FMCAD, pages 108-115, 2017".

Biddy includes:

- automatic garbage collection with a system age (a variant of a mark-and-sweep approach),
- node management through formulae protecting,
- variable swapping and sifting algorithm for all supported BDD types.

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 which allows visualization of BDDs and also includes some benchmarks
- Efficient Symbolic Tools (EST), model checking and other algorithms for formal verification of systems (<http://est.meolic.com/>)

### ### 2. SOURCE CODE

Biddy is free software released under GPL.

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. a name or a value) 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 core 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)
- [biddyOp.c](#) (functions for operations on BDDs)
- [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-deb (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)

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.

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:

```
1 -lbiddy -lgmp
```

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.

```
1 /* compile with gcc -o program.exe source.c -I. -L. -lbiddy -lgmp */
2 #define UNIX
3 #include "biddy.h"
4
5 #define Str2Var(x) (Biddy_GetVariable((Biddy_String)x))
6
7 int main() {
8     Biddy_Edge f,g,h,r;
9
10    Biddy_Init(); /* use default, i.e. ROBDD WITH COMPLEMENTED EDGES */
11    printf("Biddy is using %s.\n",Biddy_GetManagerName());
12
13    f = Biddy_Evall((Biddy_String)"(OR H E L L O)"); /* PREFIX INPUT */
14    g = Biddy_Evall((Biddy_String)"(AND W O R L D)"); /* PREFIX INPUT */
15    h = Biddy_Eval2((Biddy_String)"~(H * E * L * L)"); /* INFIX INPUT */
16
17    /* BASIC OPERATION */
18    r = Biddy_Xor(f,g);
19
20    /* REPLACE SOME VARIABLES */
21    Biddy_ResetVariablesValue();
22    Biddy_SetVariableValue(Str2Var("H"),Biddy_GetVariableEdge(Str2Var("L")));
23    Biddy_SetVariableValue(Str2Var("K"),Biddy_GetVariableEdge(Str2Var("L")));
24    Biddy_SetVariableValue(Str2Var("W"),Biddy_GetVariableEdge(Str2Var("L")));
25    r = Biddy_Replace(r);
26
27    /* SIMPLE RESTRICTION */
28    r = Biddy_Restrict(r,Str2Var("E"),FALSE);
29
30    /* COUDERT AND MADRE'S RESTRICT FUNCTION */
31    r = Biddy_Simplify(r,h);
32
33    /* SOME STATS */
34    printf("Function r depends on %u variables.\n",Biddy_DependentVariableNumber(r));
35    printf("Function r has %.0f minterms.\n",Biddy_CountMinterms(r,0));
36    printf("BDD for function r has %u nodes.\n",Biddy_CountNodes(r));
37
38    /* TRUTH TABLE */
39    printf("Here is a truth table for function r\n");
40    Biddy_PrintfTable(r);
41
42    /* GRAPHVIZ/DOT OUTPUT */
43    Biddy_WriteDot("biddy.dot",r,"r",-1,FALSE);
44    printf("USE 'dot -y -Tpng -O biddy.dot' to visualize BDD for function r.\n");
45
46    Biddy_Exit();
47 }
```

If you do not want to use the default BDD type you simply change the initialization call. Supported BDD types are BIDDYTYPEOBDD, BIDDYTYPEOBDDC, BIDDYTYPEZBDDC, and BIDDYTYPETZBDD.

```
1 Biddy_InitAnonymous(BIDDYTYPEZBDDC);
```

---

### 3.1 NODE MANAGEMENT THROUGH FORMULAE PROTECTING

Biddy includes powerful node management based on formulae tagging. There are six user functions to maintain nodes.

#### **Biddy\_AddFormula(name,bdd,c)**

Given BDD becomes a formula. Its nodes 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 (c == -1) then formula is refreshed but not preserved. If (c == 0) then formula is persistently preserved. There are five macros defined to simplify adding formulae: `Biddy_AddTmpFormula(name,bdd)`, `Biddy_AddPersistentFormula(name,bdd)`, `Biddy_KeepFormula(bdd)`, `Biddy_KeepFormulaUntilPurge(bdd)`, and `Biddy_KeepFormulaProlonged(bdd,c)`.

#### **Biddy\_DeleteFormula(name)**

Nodes of the given formula are marked as deleted. Formula is not accessible by its name anymore. Nodes of deleted formula which are preserved or persistently preserved will not be removed by regular GC whilst `Biddy_Purge` will immediately remove all of them.

#### **Biddy\_Clean()**

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 obsolete nodes.

#### **Biddy\_Purge()**

Immediately removes all nodes which were not preserved or which are not preserved anymore. Moreover, all formulae without a name are deleted! All nodes from all deleted formulae are removed if they are not needed by other formulae. 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 in the given bdd become fresh nodes. Formula is not created and no information about the existing formulae is changed. This function is needed to implement user's operation caches, only.

### 3.2. EXAMPLES OF NODE MANAGEMENT WITH BIDDY

The first example is a straightforward calculation.

```

1 f1 = op(...);
2 f2 = op(...);
3 g1 = op(f1,f2,...);
4 Biddy_KeepFormula(g1); /* g1 is preserved for next cleaning */
5 f1 = op(...);
6 f2 = op(...);
7 g2 = op(f1,f2,...);
8 Biddy_KeepFormula(g2); /* g2 is preserved for next cleaning */
9 Biddy_Clean(); /* g1 and g2 are still usable, f1 and f2 are obsolete */
10 result = op(g1,g2,...);
11 Biddy_KeepFormulaUntilPurge(result); /* result is permanently preserved */
12 Biddy_Clean(); /* all nodes not belonging to result are immediately removed */

```

If additional garbage collection is needed also after the calculation of g1, you can use the following code after the calculation of g1:

```

1 Biddy_KeepFormulaProlonged(g1,2); /* g1 is preserved for next two cleanings */
2 Biddy_Clean(); /* g1 remains preserved for next cleaning */

```

In this approach, f1 and f2 become obsolete after Biddy\_Clean, but their nodes are not immediately removed (automatic garbage collection is only started when there are no free nodes in the system).

Alternatively, you can use the following code which is simpler but somehow less efficient because Biddy\_Purge() always starts garbage collection:

```

1 f1 = op(...);
2 f2 = op(...);
3 g1 = op(f1,f2,...);
4 Biddy_AddTmpFormula("g1",g1); /* g1 is named */
5 Biddy_Purge(); /* keep only nodes from named formulae */
6 f1 = op(...);
7 f2 = op(...);
8 g2 = op(f1,f2,...);
9 Biddy_AddTmpFormula("g2",g2); /* g2 is named */
10 Biddy_Purge(); /* keep only nodes from named formulae */
11 result = op(g1,g2,...);
12 Biddy_AddPersistentFormula("result",result); /* result is permanently preserved */
13 Biddy_Clean(); /* tmp formulae becomes obsolete */
14 Biddy_Purge(); /* all nodes not belonging to result are immediately removed */

```

The second example is an iterative calculation:

```

1 result = op(...);
2 while (!finish) {
3   Biddy_KeepFormula(result); /* result is preserved for next cleaning */
4   Biddy_Clean(); /* result is still usable, f and g are obsolete */
5   f = op(...);
6   g = op(f,...);
7   result = op(result,g,...);
8 }
9 Biddy_KeepFormulaUntilPurge(result); /* result is permanently preserved */
10 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:

```

1 result = op(...);
2 while (!finish) {
3   Biddy_KeepFormulaProlonged(result,2); /* result is preserved for next two cleanings */
4   Biddy_Clean(); /* result remains preserved, f and g are obsolete */
5   f = op(...);
6   g = op(f,...);
7   Biddy_KeepFormula(g); /* g is preserved for next cleaning */
8   Biddy_Clean(); /* result and g are still usable, f is obsolete */
9   result = op(result,g,...);
10 }
11 Biddy_KeepFormulaUntilPurge(result); /* final result is permanently preserved */
12 Biddy_Clean(); /* tmp results are not needed, anymore */

```



The third example is an outline of an implementation of bisimulation:

```

1 init = AND(init_p,init_q)
2 Bidy_KeepFormulaUntilPurge(init) /* init is permanently preserved */
3 eq = InitialEq(init_p,tr_p,init_q,tr_q,...);
4 do {
5   Bidy_KeepFormula(eq); /* eq is preserved for next cleaning */
6   Bidy_Clean(); /* eq remains usable */
7   last = eq;
8   eq1 = NextEqPart1(eq,tr_p,tr_q,...);
9   eq2 = NextEqPart2(eq,tr_p,tr_q,...);
10  eq = AND(eq1,eq2);
11 } while (AND(init,eq)!=0 && eq!=last)
12 if (AND(init,eq)!=0) result=false; else result=true;
13 Bidy_Clean();
14 Bidy_Purge(); /* immediately remove all nodes created during the calculation */

```

The fourth example is an outline of an implementation of model checking where we are trying to benefit from regularly reordering (in contrast to `Bidy_Purge()`, `Bidy_PurgeAndReorder()` will delete named tmp formulae):

```

1 sup = Prepare(...);
2 Bidy_AddPersistentFormula("sup",sup) /* sup is permanently preserved */
3 Z = 0;
4 last = 1;
5 while (Z!=last) {
6   Bidy_AddPersistentFormula("Z",Z); /* old Z is marked as deleted */
7   Bidy_PurgeAndReorder(Z); /* perform reordering to optimize Z */
8   last = Z;
9   Z = NextSet(Z,sup,...);
10 }
11 result = Z;
12 Bidy_AddPersistentFormula("result",result); /* final result is permanently preserved */
13 Bidy_DeleteFormula("sup"); /* sup is marked as deleted */
14 Bidy_DeleteFormula("Z"); /* Z is marked as deleted */
15 Bidy_Purge(); /* remove nodes not belonging to result */

```

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

```

1 sacc = snew = AND(init_1,init_2,...,init_N);
2 for (i=1;i<=N;i++) di[i] = 0;
3 for (i=1;i<=N;i++) for (j=1;j<=N;j++) dij[i,j] = 0;
4 do {
5   Bidy_KeepFormulaProlonged(snew,N*(N+1)); /* snew is preserved just long enough */
6   Bidy_KeepFormulaProlonged(sacc,N*(N+1)); /* sacc is preserved just long enough */
7   new1 = 0;
8   for (i=1;i<=N;i++) {
9     sup = OneStep(snew,tr_i,...);
10    di[i] = OR(d[i],sup);
11    new1 = OR(new1,NextState(sup,...));
12    Bidy_KeepFormulaProlonged(di[i],N*(N+1)); /* di[i] is preserved just long enough */
13    Bidy_KeepFormulaProlonged(new1,1); /* new1 is preserved for next cleaning, only */
14    Bidy_Clean(); /* new1 remains usable */
15  }
16  Bidy_KeepFormulaProlonged(new1,N*N); /* new1 is preserved just long enough */
17  new2 = 0;
18  for (i=1;i<=N;i++) for (j=1;j<=N;j++) {
19    sup = OneStep(snew,tr_i,tr_j,...);
20    dij[i,j] = OR(d[i,j],sup);
21    new2 = OR(new2,NextState(sup,...));
22    Bidy_KeepFormulaProlonged(dij[i,j],N*(N+1)); /* dij[i,j] is preserved just long enough */
23    Bidy_KeepFormulaProlonged(new2,1); /* new2 is preserved for next cleaning, only */
24    Bidy_Clean(); /* new2 remains usable */
25  }
26  snew = AND(OR(new1,new2),NOT(sacc));
27  sacc = OR(sacc,snew);
28 } while (snew!=0)
29 tr1 = 0;
30 for (i=1;i<=N;i++) {
31  sup = AddStab(di[i],...);
32  tr1 = OR(tr1,sup);
33  Bidy_KeepFormulaProlonged(tr1,1); /* tr1 is preserved for next cleaning, only */
34  Bidy_Clean(); /* tr1 remains usable */
35 }
36 Bidy_KeepFormulaProlonged(tr1,N*N); /* tr1 is preserved just long enough */
37 tr2 = 0;
38 for (i=1;i<=N;i++) for (j=1;j<=N;j++) {

```

```

39  sup = AddStab(dij[i,j],...);
40  tr2 = OR(tr2,sup);
41  Biddy_KeepFormulaProlonged(tr2,1); /* tr2 is preserved for next cleaning, only */
42  Biddy_Clean(); /* tr2 remains usable */
43 }
44 result = OR(tr1,tr2);
45 Biddy_KeepFormulaUntilPurge(result); /* final result is permanently preserved */
46 Biddy_Clean(); /* tmp results are not needed, anymore */

```

### 3.3 MORE DETAILS ON MEMORY MANAGEMENT (SYSTEM AGE AND NODE CHAINING)

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 but a different approach. We call the implemented algorithm "GC with a system age". It is a variant of a mark-and-sweep approach.

Using system age instead of reference counter 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 system age 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 (expiry value == 0);
- **fresh** node (expiry value == biddySystemAge);
- **prolonged** node (expiry value > biddySystemAge);
- **obsolete** node (0 < expiry value < biddySystemAge).

Before each GC, nodes must be refreshed in such a way that no successor of a non-obsolete node is obsolete. This is achieved with a relatively simple loop which check every nodes at most once.

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 protecting schema or a system age approach, it is an independent mechanism.

## ### 4. BUILDING PACKAGES

### Compiling Biddy library

On GNU/Linux, we are using gcc.

```

1 biddy> make dynamic
2 biddy> make clean

```

Alternatively, you can use:

```

1 biddy> make static
2 biddy> make debug
3 biddy> make profile

```

You can use specify the target folder:

```
1 bidy> make dynamic "BINDIR = ./bin"
2 bidy> make clean "BINDIR = ./bin"
```

On MS Windows, we are using MSYS2. We use pacman to prepare the environment:

```
1 MSYS shell> pacman -Syuu
2 MSYS shell> pacman -S mingw-w64-i686-gcc
3 MSYS shell> pacman -S mingw-w64-x86_64-gcc
4 MSYS shell> pacman -S make
5 MSYS shell> pacman -S bison
6 MSYS shell> pacman -S gdb
7 MSYS shell> pacman -S nano
8 MSYS shell> pacman -S tar
9 MSYS shell> pacman -S subversion
```

Alternatively, you can use Visual Studio for building. There is a prepared solution consisting of many projects. You need to adapt include and lib folders.

```
1 ./VS/Biddy.sln
```

To produce nice setup files, we use Advanced Installer (<http://www.advancedinstaller.com/>). We have been granted a free licence. MANY THANKS!

## Dependencies

On GNU/Linux, we are using libgmp (<https://gmplib.org/>).

On MS Windows, we are using MPIR library (<http://mpir.org/>).

## Creating Bidy library as a zip package

```
1 bidy> ./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:

```
1 export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/absolute/path/to/bidy/library
```

## Creating zip file with source code of a complete Bidy project

```
1 bidy> ./package-source
```

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

## Creating packages for GNU/Linux

```
1 biddy> ./package-tgz
2 biddy> ./package-deb
3 biddy> ./package-rpm
```

These scripts are intended to be used on Ubuntu. These scripts need 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`.

```
1 bddscout> ./package-bin
2 bddscout> ./package-tgz
3 bddscout> ./package-deb
4 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 unzipping 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 2003, BDD package from EST was included in the report presented at 16th Symposium on Integrated Circuits and Systems Design (SBCCI'03). The report is available as a paper "G. Janssen. A Consumer Report on BDD Packages. IBM T.J. Watson Research Center. 2003". Get it from <http://doi.ieeecomputersociety.org/10.1109/SBCCI.2003.1232832>

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 updated versions of Biddy v1.0 were released which added support for debian packaging, support for RPM packaging, fix errors, and improve documentation, packaging, and Tcl/Tk GUI.

In 2012, a research paper about Biddy library appears in Journal of Software (doi:10.4304/jsw.7.6.1358-1366) <http://ojs.academypublisher.com/index.php/jsw/article/view/jsw070613581366/>

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, v1.4 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>

In 2016, Biddy v1.6 was released. Formulae are not recursively refreshed all the time, anymore. The size of Node table became resizable.

In 2017, Biddy v1.7 was released (there were 4 minor releases, the last one was Biddy v1.7.4). Terminology has changed a lot, e.g. "formulae counter" became "system age". Added support for ROBDDs without complemented edges, 0-sup-BDDs and Tagged 0-sup-BDDs. Implemented creation and manipulation of non-anonymous managers. Added manipulation of combination sets. Improved many functionalities, e.g sifting. Many new CUDD-like functions have been added. Moreover, bddview and BDD Scout have been significantly improved.

## ### 6. PUBLICATIONS

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

- Robert Meolic. **Biddy - a multi-platform academic BDD package**. Journal of Software, 7(6), pp. 1358-1366, 2012. <http://ojs.academypublisher.com/index.php/jsw/article/view/jsw070613581366>
- Robert Meolic. **Implementation aspects of a BDD package supporting general decision diagrams**. Technical report, University of Maribor, 2016. <https://dk.um.si/IzpisGradiva.php?id=68831>
- Robert Meolic. **Biddy**. We are preparing a paper for SCP.



# Chapter 2

## Data Structure Index

### 2.1 Data Structures

Here are the data structures with brief descriptions:

[Bidy\\_XY](#) ..... 17





# Chapter 3

## File Index

### 3.1 File List

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

<b>bidly-cudd.c</b>	??
<b>bidly-cudd.h</b>	??
<b>bidly-example-8queens.c</b>	??
<b>bidly-example-bra.c</b>	??
<b>bidly-example-dictionary.c</b>	??
<b>bidly-example-hanoi.c</b>	??
<b>bidly-example-independence-europe.c</b>	??
<b>bidly-example-independence-test.c</b>	??
<b>bidly-example-independence-usa.c</b>	??
<b>bidly-example-independence.c</b>	??
<b>bidly-example-random.c</b>	??
<b>bidly.h</b>	
File <b>bidly.h</b> contains declaration of all external data structures	19
<b>bidlyInOut.c</b>	
File <b>bidlyInOut.c</b> contains various parsers and generators	47
<b>bidlyInt.h</b>	
File <b>bidlyInt.h</b> contains declaration of internal data structures	53
<b>bidlyMain.c</b>	
File <b>bidlyMain.c</b> contains main functions for representation and manipulation of boolean functions with various types of Binary Decision Diagrams (GDD = general decision diagrams)	53
<b>bidlyMainLegacy.c</b>	
File <b>bidlyMainLegacy.c</b> contains main functions for representation and manipulation of boolean functions with ROBDDs	92
<b>bidlyOp.c</b>	
File <b>bidlyOp.c</b> contains functions for operations on various types of Binary Decision Diagrams	134
<b>bidlyStat.c</b>	
File <b>bidlyStat.c</b> contains statistical functions	147



## Chapter 4

# Data Structure Documentation

### 4.1 Biddy\_XY Struct Reference

```
#include <biddy.h>
```

#### 4.1.1 Detailed Description

[Biddy\\_XY](#) is used in `Biddy_WriteBddview` to pass node coordinates

Definition at line 277 of file `biddy.h`.

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

- [biddy.h](#)



# Chapter 5

## File Documentation

### 5.1 biddy.h File Reference

File [biddy.h](#) contains declaration of all external data structures.

```
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <stdarg.h>
#include <time.h>
```

#### Data Structures

- struct [Biddy\\_XY](#)

#### Macros

- `#define Biddy_IsNull(f) (f == NULL)`
- `#define Biddy_IsTerminal(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 = (Biddy_Edge) ((uintptr_t) f | (uintptr_t) 1))`
- `#define Biddy_ClearMark(f) (f = (Biddy_Edge) ((uintptr_t) f & ~((uintptr_t) 1)))`
- `#define Biddy_InvertMark(f) (f = (Biddy_Edge) ((uintptr_t) f ^ (uintptr_t) 1))`
- `#define Biddy_Inv(f) ((Biddy_Edge) ((uintptr_t) f ^ (uintptr_t) 1))`
- `#define Biddy_InvCond(f, c) (c ? ((Biddy_Edge) ((uintptr_t) f ^ (uintptr_t) 1)) : f)`
- `#define Biddy_Regular(f) ((Biddy_Edge) ((uintptr_t) f & ~((uintptr_t) 1)))`
- `#define Biddy_Complement(f) ((Biddy_Edge) ((uintptr_t) f | (uintptr_t) 1))`
- `#define Biddy_GetTag(f) 0`
- `#define Biddy_SetTag(f, t) 0`
- `#define Biddy_ClearTag(f) 0`
- `#define Biddy_Untagged(f) 0`
- `#define Biddy_Init() Biddy_InitMNG(NULL, BIDDYTYPEOBDDC)`
- `#define Biddy_Exit() Biddy_ExitMNG(NULL)`

- #define Bidly\_GetManagerType() Bidly\_Managed\_GetManagerType(NULL)
- #define Bidly\_GetManagerName() Bidly\_Managed\_GetManagerName(NULL)
- #define Bidly\_SetManagerParameters(gcr, gcrF, gcrX, rr, rrF, rrX, st, cst) Bidly\_Managed\_SetManagerParameters(NULL,gcr,gcrF,gcrX,rr,rrF,rrX,st,cst)
- #define Bidly\_Managed\_GetThen(MNG, f) Bidly\_GetThen(f)
- #define Bidly\_Managed\_GetElse(MNG, f) Bidly\_GetElse(f)
- #define Bidly\_Managed\_GetTopVariable(MNG, f) Bidly\_GetTopVariable(f)
- #define Bidly\_IsEqv(f1, MNG2, f2) Bidly\_Managed\_IsEqv(NULL,f1,MNG2,f2)
- #define Bidly\_SelectNode(f) Bidly\_Managed\_SelectNode(NULL,f)
- #define Bidly\_DeselectNode(f) Bidly\_Managed\_DeselectNode(NULL,f)
- #define Bidly\_IsSelected(f) Bidly\_Managed\_IsSelected(NULL,f)
- #define Bidly\_SelectFunction(f) Bidly\_Managed\_SelectFunction(NULL,f)
- #define Bidly\_DeselectAll() Bidly\_Managed\_DeselectAll(NULL)
- #define Bidly\_GetTerminal() Bidly\_Managed\_GetTerminal(NULL)
- #define Bidly\_GetConstantZero() Bidly\_Managed\_GetConstantZero(NULL)
- #define Bidly\_GetConstantOne() Bidly\_Managed\_GetConstantOne(NULL)
- #define Bidly\_GetBaseSet() Bidly\_Managed\_GetBaseSet(NULL)
- #define Bidly\_GetVariable(x) Bidly\_Managed\_GetVariable(NULL,x)
- #define Bidly\_GetLowestVariable() Bidly\_Managed\_GetLowestVariable(NULL)
- #define Bidly\_GetIthVariable(i) Bidly\_Managed\_GetIthVariable(NULL,i)
- #define Bidly\_GetPrevVariable(v) Bidly\_Managed\_GetPrevVariable(NULL,v)
- #define Bidly\_GetNextVariable(v) Bidly\_Managed\_GetNextVariable(NULL,v)
- #define Bidly\_GetVariableEdge(v) Bidly\_Managed\_GetVariableEdge(NULL,v)
- #define Bidly\_GetElementEdge(v) Bidly\_Managed\_GetElementEdge(NULL,v)
- #define Bidly\_GetVariableName(v) Bidly\_Managed\_GetVariableName(NULL,v)
- #define Bidly\_GetTopVariableEdge(f) Bidly\_Managed\_GetTopVariableEdge(NULL,f)
- #define Bidly\_GetTopVariableName(f) Bidly\_Managed\_GetTopVariableName(NULL,f)
- #define Bidly\_GetTopVariableChar(f) Bidly\_Managed\_GetTopVariableChar(NULL,f)
- #define Bidly\_ResetVariablesValue() Bidly\_Managed\_ResetVariablesValue(NULL)
- #define Bidly\_SetVariableValue(v, f) Bidly\_Managed\_SetVariableValue(NULL,v,f)
- #define Bidly\_GetVariableValue(v) Bidly\_Managed\_GetVariableValue(NULL,v)
- #define Bidly\_ClearVariablesData() Bidly\_Managed\_ClearVariablesData(NULL)
- #define Bidly\_SetVariableData(v, x) Bidly\_Managed\_SetVariableData(NULL,v,x)
- #define Bidly\_GetVariableData(v) Bidly\_Managed\_GetVariableData(NULL,v)
- #define Bidly\_IsSmaller(fv, gv) Bidly\_Managed\_IsSmaller(NULL,fv,gv)
- #define Bidly\_IsLowest(v) Bidly\_Managed\_IsLowest(NULL,v)
- #define Bidly\_IsHighest(v) Bidly\_Managed\_IsHighest(NULL,v)
- #define Bidly\_FoaVariable(x, varelem) Bidly\_Managed\_FoaVariable(NULL,x,varelem)
- #define Bidly\_ChangeVariableName(v, x) Bidly\_Managed\_ChangeVariableName(NULL,v,x)
- #define Bidly\_AddVariableByName(x) Bidly\_Managed\_AddVariableByName(NULL,x)
- #define Bidly\_AddElementByName(x) Bidly\_Managed\_AddElementByName(NULL,x)
- #define Bidly\_AddVariableBelow(v) Bidly\_Managed\_AddVariableBelow(NULL,v)
- #define Bidly\_AddVariableAbove(v) Bidly\_Managed\_AddVariableAbove(NULL,v)
- #define Bidly\_TransferMark(f, mark, leftright) Bidly\_Managed\_TransferMark(NULL,f,mark,leftright)
- #define Bidly\_IncTag(f) Bidly\_Managed\_IncTag(NULL,f)
- #define Bidly\_TaggedFoaNode(v, pf, pt, ptag, garbageAllowed) Bidly\_Managed\_TaggedFoaNode(NULL,v,pf,pt,ptag,garbageAllowed)
- #define Bidly\_IsOK(f) Bidly\_Managed\_IsOK(NULL,f)
- #define Bidly\_GC(targetLT, targetGEQ, purge, total) Bidly\_Managed\_GC(NULL,targetLT,targetGEQ,purge,total)
- #define Bidly\_Clean() Bidly\_Managed\_Clean(NULL)
- #define Bidly\_Purge() Bidly\_Managed\_Purge(NULL)
- #define Bidly\_PurgeAndReorder(f, c) Bidly\_Managed\_PurgeAndReorder(NULL,f,c)
- #define Bidly\_Refresh(f) Bidly\_Managed\_Refresh(NULL,f)
- #define Bidly\_AddCache(gc) Bidly\_Managed\_AddCache(NULL,gc)

- `#define Biddy_AddFormula(x, f, c) Biddy_Managed_AddFormula(NULL,x,f,c)`
- `#define Biddy_FindFormula(x, idx, f) Biddy_Managed_FindFormula(NULL,x,idx,f)`
- `#define Biddy_DeleteFormula(x) Biddy_Managed_DeleteFormula(NULL,x)`
- `#define Biddy_DeletelthFormula(x) Biddy_Managed_DeletelthFormula(NULL,x)`
- `#define Biddy_GetlthFormula(i) Biddy_Managed_GetlthFormula(NULL,i)`
- `#define Biddy_GetlthFormulaName(i) Biddy_Managed_GetlthFormulaName(NULL,i)`
- `#define Biddy_SwapWithHigher(v) Biddy_Managed_SwapWithHigher(NULL,v)`
- `#define Biddy_SwapWithLower(v) Biddy_Managed_SwapWithLower(NULL,v)`
- `#define Biddy_Sifting(f, c) Biddy_Managed_Sifting(NULL,f,c)`
- `#define Biddy_MinimizeBDD(f) Biddy_Managed_MinimizeBDD(NULL,f)`
- `#define Biddy_MaximizeBDD(f) Biddy_Managed_MaximizeBDD(NULL,f)`
- `#define Biddy_Copy(MNG2, f) Biddy_Managed_Copy(NULL,MNG2,f)`
- `#define Biddy_CopyFormula(MNG2, x) Biddy_Managed_CopyFormula(NULL,MNG2,x)`
- `#define Biddy_Eval(f) Biddy_Managed_Eval(NULL,f)`
- `#define Biddy_Not(f) Biddy_Managed_Not(NULL,f)`
- `#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_Gt(f, g) Biddy_Managed_Gt(NULL,f,g)`
- `#define Biddy_IsLeq(f, g) Biddy_Managed_IsLeq(NULL,f,g)`
- `#define Biddy_Restrict(f, v, value) Biddy_Managed_Restrict(NULL,f,v,value)`
- `#define Biddy_Compose(f, g, v) Biddy_Managed_Compose(NULL,f,g,v)`
- `#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_ReplaceByKeyword(f, keyword) Biddy_Managed_ReplaceByKeyword(NULL,f,keyword)`
- `#define Biddy_Change(f, v) Biddy_Managed_Change(NULL,f,v)`
- `#define Biddy_Subset(f, v, value) Biddy_Managed_Subset(NULL,f,v,value)`
- `#define Biddy_CreateMinterm(support, x) Biddy_Managed_CreateMinterm(NULL,support,x)`
- `#define Biddy_CreateFunction(support, x) Biddy_Managed_CreateFunction(NULL,support,x)`
- `#define Biddy_RandomFunction(support, r) Biddy_Managed_RandomFunction(NULL,support,r)`
- `#define Biddy_RandomSet(unit, r) Biddy_Managed_RandomSet(NULL,unit,r)`
- `#define Biddy_CountNodes(f) Biddy_Managed_CountNodes(NULL,f)`
- `#define Biddy_Managed_MaxLevel(MNG, f) Biddy_MaxLevel(f)`
- `#define Biddy_Managed_AvgLevel(MNG, f) Biddy_AvgLevel(f)`
- `#define Biddy_VariableTableNum() Biddy_Managed_VariableTableNum(NULL)`
- `#define Biddy_NodeTableSize() Biddy_Managed_NodeTableSize(NULL)`
- `#define Biddy_NodeTableBlockNumber() Biddy_Managed_NodeTableBlockNumber(NULL)`
- `#define Biddy_NodeTableGenerated() Biddy_Managed_NodeTableGenerated(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_NodeTableResizeNumber() Biddy_Managed_NodeTableResizeNumber(NULL)`
- `#define Biddy_NodeTableFoaNumber() Biddy_Managed_NodeTableFoaNumber(NULL)`

- #define Biddy\_NodeTableFindNumber() Biddy\_Managed\_NodeTableFindNumber(NULL)
- #define Biddy\_NodeTableCompareNumber() Biddy\_Managed\_NodeTableCompareNumber(NULL)
- #define Biddy\_NodeTableAddNumber() Biddy\_Managed\_NodeTableAddNumber(NULL)
- #define Biddy\_NodeTableGCNumber() Biddy\_Managed\_NodeTableGCNumber(NULL)
- #define Biddy\_NodeTableGCTime() Biddy\_Managed\_NodeTableGCTime(NULL)
- #define Biddy\_NodeTableGCObsoleteNumber() Biddy\_Managed\_NodeTableGCObsoleteNumber(NULL)
- #define Biddy\_NodeTableSwapNumber() Biddy\_Managed\_NodeTableSwapNumber(NULL)
- #define Biddy\_NodeTableSiftingNumber() Biddy\_Managed\_NodeTableSiftingNumber(NULL)
- #define Biddy\_NodeTableDRTTime() Biddy\_Managed\_NodeTableDRTTime(NULL)
- #define Biddy\_NodeTableITENumber() Biddy\_Managed\_NodeTableITENumber(NULL)
- #define Biddy\_NodeTableITERRecursiveNumber() Biddy\_Managed\_NodeTableITERRecursiveNumber(NULL)
- #define Biddy\_NodeTableANDORNumber() Biddy\_Managed\_NodeTableANDORNumber(NULL)
- #define Biddy\_NodeTableANDORRecursiveNumber() Biddy\_Managed\_NodeTableANDORRecursiveNumber(NULL)
- #define Biddy\_NodeTableXORNumber() Biddy\_Managed\_NodeTableXORNumber(NULL)
- #define Biddy\_NodeTableXORRecursiveNumber() Biddy\_Managed\_NodeTableXORRecursiveNumber(NULL)
- #define Biddy\_FormulaTableNum() Biddy\_Managed\_FormulaTableNum(NULL)
- #define Biddy\_ListUsed() Biddy\_Managed\_ListUsed(NULL)
- #define Biddy\_ListMaxLength() Biddy\_Managed\_ListMaxLength(NULL)
- #define Biddy\_ListAvgLength() Biddy\_Managed\_ListAvgLength(NULL)
- #define Biddy\_OPCacheSearch() Biddy\_Managed\_OPCacheSearch(NULL)
- #define Biddy\_OPCacheFind() Biddy\_Managed\_OPCacheFind(NULL)
- #define Biddy\_OPCacheInsert() Biddy\_Managed\_OPCacheInsert(NULL)
- #define Biddy\_OPCacheOverwrite() Biddy\_Managed\_OPCacheOverwrite(NULL)
- #define Biddy\_CountNodesPlain(f) Biddy\_Managed\_CountNodesPlain(NULL,f)
- #define Biddy\_DependentVariableNumber(f) Biddy\_Managed\_DependentVariableNumber(NULL,f)
- #define Biddy\_CountComplementedEdges(f) Biddy\_Managed\_CountComplementedEdges(NULL,f)
- #define Biddy\_CountPaths(f) Biddy\_Managed\_CountPaths(NULL,f)
- #define Biddy\_CountMinterms(f, nvars) Biddy\_Managed\_CountMinterms(NULL,f,nvars)
- #define Biddy\_DensityOfFunction(f, nvars) Biddy\_Managed\_DensityOfFunction(NULL,f,nvars)
- #define Biddy\_DensityOfBDD(f, nvars) Biddy\_Managed\_DensityOfBDD(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\_Eval1x(s, lf) Biddy\_Managed\_Eval1x(NULL,s,lf)
- #define Biddy\_Eval2(boolFunc) Biddy\_Managed\_Eval2(NULL,boolFunc)
- #define Biddy\_ReadVerilogFile(filename, prefix) Biddy\_Managed\_ReadVerilogFile(NULL,filename,prefix)
- #define Biddy\_PrintfBDD(f) Biddy\_Managed\_PrintfBDD(NULL,f)
- #define Biddy\_WriteBDD(filename, f, label) Biddy\_Managed\_WriteBDD(NULL,filename,f,label)
- #define Biddy\_PrintfTable(f) Biddy\_Managed\_PrintfTable(NULL,f)
- #define Biddy\_WriteTable(filename, f) Biddy\_Managed\_WriteTable(NULL,filename,f)
- #define Biddy\_PrintfSOP(f) Biddy\_Managed\_PrintfSOP(NULL,f)
- #define Biddy\_WriteSOP(filename, f) Biddy\_Managed\_WriteSOP(NULL,filename,f)
- #define Biddy\_WriteDot(filename, f, label, id, cudd) Biddy\_Managed\_WriteDot(NULL,filename,f,label,id,cudd);
- #define Biddy\_WriteBddview(filename, f, label, table) Biddy\_Managed\_WriteBddview(NULL,filename,f,label,table);

## Typedefs

- typedef char Biddy\_Boolean
- typedef char \* Biddy\_String
- typedef void \*\* Biddy\_Manager
- typedef void \* Biddy\_Cache
- typedef unsigned short int Biddy\_Variable
- typedef void \* Biddy\_Edge
- typedef void(\* Biddy\_GCFunction) (Biddy\_Manager)
- typedef Biddy\_Boolean(\* Biddy\_LookupFunction) (Biddy\_String, Biddy\_Edge \*)



### 5.1.1 Detailed Description

File [bidly.h](#) contains declaration of all external data structures.

#### Description

```

PackageName [Bidly]
Synopsis [Bidly provides data structures and algorithms for the
representation and manipulation of Boolean functions with
ROBDDs, 0-sup-BDDs, and TZBDDs. A hash table is used for quick
search of nodes. Complement edges decreases the number of
nodes. An automatic garbage collection with a system age is
implemented. Variable swapping and sifting are implemented.]

FileName [bidly.h]
Revision [${Revision}: 365 $]
Date [Date: 2017-12-18 13:01:40 +0100 (pon, 18 dec 2017) $]
Authors [Robert Meolic (robert.meolic@um.si),
Ales Casar (ales@homemade.net)]

```

#### Copyright

Copyright (C) 2006, 2017 UM FERi, Koroska cesta 46, SI-2000 Maribor, Slovenia

Bidly 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.

Bidly 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: [bidlyInt.h](#)

### 5.1.2 Macro Definition Documentation

#### 5.1.2.1 `#define Bidly_IsNull( f ) (f == NULL)`

`Bidly_IsNull` returns TRUE iff given BDD is a null edge.

Definition at line 150 of file `bidly.h`.

#### 5.1.2.2 `#define Bidly_IsTerminal( f ) (((void*)((uintptr_t) f & ~((uintptr_t) 1))) [2] == NULL) && (((void*)((uintptr_t) f & ~((uintptr_t) 1))) [3] == NULL))`

`Bidly_IsTerminal` returns TRUE iff given edge points to the terminal node.

Definition at line 160 of file `bidly.h`.

```
5.1.2.3 #define Biddy_IsEqPointer( f, g ) (((uintptr_t) f & ~((uintptr_t) 1)) == ((uintptr_t) g & ~((uintptr_t) 1)))
```

Biddy\_IsEqPointer returns TRUE iff given edges points to the same node.

Definition at line 165 of file biddy.h.

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

Biddy\_GetMark returns TRUE iff given edge is complemented.

Definition at line 168 of file biddy.h.

```
5.1.2.5 #define Biddy_SetMark( f ) (f = (Biddy_Edge) ((uintptr_t) f | (uintptr_t) 1))
```

Biddy\_SetMark makes given edge complemented.

Definition at line 171 of file biddy.h.

```
5.1.2.6 #define Biddy_ClearMark( f ) (f = (Biddy_Edge) ((uintptr_t) f & ~((uintptr_t) 1)))
```

Biddy\_ClearMark makes given edge not-complemented.

Definition at line 174 of file biddy.h.

```
5.1.2.7 #define Biddy_InvertMark( f ) (f = (Biddy_Edge) ((uintptr_t) f ^ (uintptr_t) 1))
```

Biddy\_InvertMark changes complement bit of the given edge.

Definition at line 177 of file biddy.h.

```
5.1.2.8 #define Biddy_Inv( f ) ((Biddy_Edge) ((uintptr_t) f ^ (uintptr_t) 1))
```

Biddy\_Inv returns edge with changed complement bit.

Definition at line 181 of file biddy.h.

```
5.1.2.9 #define Biddy_InvCond( f, c ) (c ? ((Biddy_Edge) ((uintptr_t) f ^ (uintptr_t) 1)) : f)
```

Biddy\_InvCond returns edge with conditionally changed complement bit.

Definition at line 184 of file biddy.h.

```
5.1.2.10 #define Biddy_Regular( f ) ((Biddy_Edge) ((uintptr_t) f & ~((uintptr_t) 1)))
```

Biddy\_Regular returns not-complemented version of edge, since Biddy v1.4.

Definition at line 187 of file biddy.h.

5.1.2.11 `#define Bidly_Complement( f ) ((Bidly_Edge) ((uintptr_t) f | (uintptr_t) 1))`

`Bidly_Complement` returns complemented version of edge, since Bidly v1.4.

Definition at line 190 of file `bidly.h`.

5.1.2.12 `#define Bidly_GetTag( f ) 0`

`Bidly_GetTag` returns tag used for the given edge, since Bidly v1.7.

Definition at line 199 of file `bidly.h`.

5.1.2.13 `#define Bidly_SetTag( f, t ) 0`

`Bidly_SetTag` adds tag to the given edge, since Bidly v1.7.

Definition at line 208 of file `bidly.h`.

5.1.2.14 `#define Bidly_ClearTag( f ) 0`

`Bidly_ClearTag` removes tag from the given edge, since Bidly v1.7.

Definition at line 216 of file `bidly.h`.

5.1.2.15 `#define Bidly_Untagged( f ) 0`

`Bidly_Untagged` returns untagged version of edge, since Bidly v1.7.

Definition at line 224 of file `bidly.h`.

5.1.2.16 `#define Bidly_Init( ) Bidly_InitMNG(NULL,BIDDYTYPEOBDDC)`

Macros `Bidly_Init` and `Bidly_InitAnonymous` will initialize anonymous manager.

Definition at line 307 of file `bidly.h`.

5.1.2.17 `#define Bidly_Exit( ) Bidly_ExitMNG(NULL)`

Macro `Bidly_Exit` will delete anonymous manager.

Definition at line 313 of file `bidly.h`.

5.1.2.18 `#define Bidly_GetManagerType( ) Bidly_Managed_GetManagerType(NULL)`

Macro `Bidly_GetManagerType` is defined for use with anonymous manager.

Definition at line 321 of file `bidly.h`.

```
5.1.2.19 #define Biddy_GetManagerName( ) Biddy_Managed_GetManagerName(NULL)
```

Macro Biddy\_GetManagerName is defined for use with anonymous manager.

Definition at line 326 of file biddy.h.

```
5.1.2.20 #define Biddy_SetManagerParameters( gcr, gcrF, gcrX, rr, rrF, rrX, st, cst
        ) Biddy_Managed_SetManagerParameters(NULL,gcr,gcrF,gcrX,rr,rrF,rrX,st,cst)
```

Macro Biddy\_SetManagerParameters is defined for use with anonymous manager.

Definition at line 331 of file biddy.h.

```
5.1.2.21 #define Biddy_Managed_GetThen( MNG, f ) Biddy_GetThen(f)
```

Macro Biddy\_Managed\_GetThen is defined for your convenience.

Definition at line 336 of file biddy.h.

```
5.1.2.22 #define Biddy_Managed_GetElse( MNG, f ) Biddy_GetElse(f)
```

Macro Biddy\_Managed\_GetElse is defined for your convenience.

Definition at line 341 of file biddy.h.

```
5.1.2.23 #define Biddy_Managed_GetTopVariable( MNG, f ) Biddy_GetTopVariable(f)
```

Macro Biddy\_Managed\_GetTopVariable is defined for your convenience.

Definition at line 346 of file biddy.h.

```
5.1.2.24 #define Biddy_IsEqv( f1, MNG2, f2 ) Biddy_Managed_IsEqv(NULL,f1,MNG2,f2)
```

Macro Biddy\_IsEqv is defined for use with anonymous manager.

Definition at line 351 of file biddy.h.

```
5.1.2.25 #define Biddy_SelectNode( f ) Biddy_Managed_SelectNode(NULL,f)
```

Macro Biddy\_SelectNode is defined for use with anonymous manager.

Definition at line 356 of file biddy.h.

5.1.2.26 **#define Bidly\_DeselectNode( f ) Bidly\_Managed\_DeselectNode(NULL,f)**

Macro Bidly\_DeselectNode is defined for use with anonymous manager.

Definition at line 361 of file bidly.h.

5.1.2.27 **#define Bidly\_IsSelected( f ) Bidly\_Managed\_IsSelected(NULL,f)**

Macro Bidly\_IsSelected is defined for use with anonymous manager.

Definition at line 366 of file bidly.h.

5.1.2.28 **#define Bidly\_SelectFunction( f ) Bidly\_Managed\_SelectFunction(NULL,f)**

Macro Bidly\_SelectFunction is defined for use with anonymous manager.

Definition at line 371 of file bidly.h.

5.1.2.29 **#define Bidly\_DeselectAll( ) Bidly\_Managed\_DeselectAll(NULL)**

Macro Bidly\_DeselectAll is defined for use with anonymous manager.

Definition at line 376 of file bidly.h.

5.1.2.30 **#define Bidly\_GetTerminal( ) Bidly\_Managed\_GetTerminal(NULL)**

Macro Bidly\_GetTerminal is defined for use with anonymous manager.

Definition at line 381 of file bidly.h.

5.1.2.31 **#define Bidly\_GetConstantZero( ) Bidly\_Managed\_GetConstantZero(NULL)**

Macro Bidly\_GetConstantZero is defined for use with anonymous manager.

Definition at line 386 of file bidly.h.

5.1.2.32 **#define Bidly\_GetConstantOne( ) Bidly\_Managed\_GetConstantOne(NULL)**

Macro Bidly\_GetConstantOne is defined for use with anonymous manager.

Definition at line 393 of file bidly.h.

5.1.2.33 **#define Bidly\_GetBaseSet( ) Bidly\_Managed\_GetBaseSet(NULL)**

Macro Bidly\_GetBaseSet is defined for use with anonymous manager.

Definition at line 400 of file bidly.h.

5.1.2.34 `#define Biddy_GetVariable( x ) Biddy_Managed_GetVariable(NULL,x)`

Macro `Biddy_GetVariable` is defined for use with anonymous manager.

Definition at line 405 of file `biddy.h`.

5.1.2.35 `#define Biddy_GetLowestVariable( ) Biddy_Managed_GetLowestVariable(NULL)`

Macro `Biddy_GetLowestVariable` is defined for use with anonymous manager.

Definition at line 410 of file `biddy.h`.

5.1.2.36 `#define Biddy_GetlthVariable( i ) Biddy_Managed_GetlthVariable(NULL,i)`

Macro `Biddy_GetlthVariable` is defined for use with anonymous manager.

Definition at line 415 of file `biddy.h`.

5.1.2.37 `#define Biddy_GetPrevVariable( v ) Biddy_Managed_GetPrevVariable(NULL,v)`

Macro `Biddy_GetPrevVariable` is defined for use with anonymous manager.

Definition at line 420 of file `biddy.h`.

5.1.2.38 `#define Biddy_GetNextVariable( v ) Biddy_Managed_GetNextVariable(NULL,v)`

Macro `Biddy_GetNextVariable` is defined for use with anonymous manager.

Definition at line 425 of file `biddy.h`.

5.1.2.39 `#define Biddy_GetVariableEdge( v ) Biddy_Managed_GetVariableEdge(NULL,v)`

Macro `Biddy_GetVariableEdge` is defined for use with anonymous manager.

Definition at line 430 of file `biddy.h`.

5.1.2.40 `#define Biddy_GetElementEdge( v ) Biddy_Managed_GetElementEdge(NULL,v)`

Macro `Biddy_GetElementEdge` is defined for use with anonymous manager.

Definition at line 435 of file `biddy.h`.

5.1.2.41 `#define Biddy_GetVariableName( v ) Biddy_Managed_GetVariableName(NULL,v)`

Macro `Biddy_GetVariableName` is defined for use with anonymous manager.

Definition at line 440 of file `biddy.h`.

5.1.2.42 `#define Bidly_GetTopVariableEdge( f ) Bidly_Managed_GetTopVariableEdge(NULL,f)`

Macro `Bidly_GetTopVariableEdge` is defined for use with anonymous manager.

Definition at line 445 of file `bidly.h`.

5.1.2.43 `#define Bidly_GetTopVariableName( f ) Bidly_Managed_GetTopVariableName(NULL,f)`

Macro `Bidly_GetTopVariableName` is defined for use with anonymous manager.

Definition at line 450 of file `bidly.h`.

5.1.2.44 `#define Bidly_GetTopVariableChar( f ) Bidly_Managed_GetTopVariableChar(NULL,f)`

Macro `Bidly_GetTopVariableChar` is defined for use with anonymous manager.

Definition at line 455 of file `bidly.h`.

5.1.2.45 `#define Bidly_ResetVariablesValue( ) Bidly_Managed_ResetVariablesValue(NULL)`

Macro `Bidly_ResetVariablesValue` is defined for use with anonymous manager.

Definition at line 460 of file `bidly.h`.

5.1.2.46 `#define Bidly_SetVariableValue( v, f ) Bidly_Managed_SetVariableValue(NULL,v,f)`

Macro `Bidly_SetVariableValue` is defined for use with anonymous manager.

Definition at line 465 of file `bidly.h`.

5.1.2.47 `#define Bidly_GetVariableValue( v ) Bidly_Managed_GetVariableValue(NULL,v)`

Macro `Bidly_GetVariableValue` is defined for use with anonymous manager.

Definition at line 470 of file `bidly.h`.

5.1.2.48 `#define Bidly_ClearVariablesData( ) Bidly_Managed_ClearVariablesData(NULL)`

Macro `Bidly_ClearVariablesData` is defined for use with anonymous manager.

Definition at line 475 of file `bidly.h`.

5.1.2.49 `#define Bidly_SetVariableData( v, x ) Bidly_Managed_SetVariableData(NULL,v,x)`

Macro `Bidly_SetVariableData` is defined for use with anonymous manager.

Definition at line 480 of file `bidly.h`.

5.1.2.50 **#define Bidly\_GetVariableData( v ) Bidly\_Managed\_GetVariableData(NULL,v)**

Macro Bidly\_GetVariableData is defined for use with anonymous manager.

Definition at line 485 of file bidly.h.

5.1.2.51 **#define Bidly\_IsSmaller( fv, gv ) Bidly\_Managed\_IsSmaller(NULL,fv,gv)**

Macro Bidly\_IsSmaller is defined for use with anonymous manager.

Definition at line 490 of file bidly.h.

5.1.2.52 **#define Bidly\_IsLowest( v ) Bidly\_Managed\_IsLowest(NULL,v)**

Macro Bidly\_IsLowest is defined for use with anonymous manager.

Definition at line 495 of file bidly.h.

5.1.2.53 **#define Bidly\_IsHighest( v ) Bidly\_Managed\_IsHighest(NULL,v)**

Macro Bidly\_IsHighest is defined for use with anonymous manager.

Definition at line 500 of file bidly.h.

5.1.2.54 **#define Bidly\_FoaVariable( x, varelem ) Bidly\_Managed\_FoaVariable(NULL,x,varelem)**

Macro Bidly\_FoaVariable is defined for use with anonymous manager.

Definition at line 505 of file bidly.h.

5.1.2.55 **#define Bidly\_ChangeVariableName( v, x ) Bidly\_Managed\_ChangeVariableName(NULL,v,x)**

Macro Bidly\_ChangeVariableName is defined for use with anonymous manager.

Definition at line 510 of file bidly.h.

5.1.2.56 **#define Bidly\_AddVariableByName( x ) Bidly\_Managed\_AddVariableByName(NULL,x)**

Macro Bidly\_AddVariableByName is defined for use with anonymous manager.

Definition at line 515 of file bidly.h.

5.1.2.57 **#define Bidly\_AddElementByName( x ) Bidly\_Managed\_AddElementByName(NULL,x)**

Macro Bidly\_AddElementByName is defined for use with anonymous manager.

Definition at line 522 of file bidly.h.



5.1.2.58 `#define Bidly_AddVariableBelow( v ) Bidly_Managed_AddVariableBelow(NULL,v)`

Macro `Bidly_AddVariableBelow` is defined for use with anonymous manager.

Definition at line 529 of file `bidly.h`.

5.1.2.59 `#define Bidly_AddVariableAbove( v ) Bidly_Managed_AddVariableAbove(NULL,v)`

Macro `Bidly_AddVariableAbove` is defined for use with anonymous manager.

Definition at line 534 of file `bidly.h`.

5.1.2.60 `#define Bidly_TransferMark( f, mark, leftright ) Bidly_Managed_TransferMark(NULL,f,mark,leftright)`

Macro `Bidly_TransferMark` is defined for use with anonymous manager.

For OBDD, use macro `Bidly_InvCond`.

Definition at line 540 of file `bidly.h`.

5.1.2.61 `#define Bidly_IncTag( f ) Bidly_Managed_IncTag(NULL,f)`

Macro `Bidly_IncTag` is defined for use with anonymous manager.

Definition at line 545 of file `bidly.h`.

5.1.2.62 `#define Bidly_TaggedFoaNode( v, pf, pt, ptag, garbageAllowed ) Bidly_Managed_TaggedFoaNode(NULL,v,pf,pt,ptag,garbageAllowed)`

Macro `Bidly_TaggedFoaNode` is defined for use with anonymous manager.

Definition at line 550 of file `bidly.h`.

5.1.2.63 `#define Bidly_IsOK( f ) Bidly_Managed_IsOK(NULL,f)`

Macro `Bidly_IsOK` is defined for use with anonymous manager.

Definition at line 557 of file `bidly.h`.

5.1.2.64 `#define Bidly_GC( targetLT, targetGEQ, purge, total ) Bidly_Managed_GC(NULL,targetLT,targetGEQ,purge,total)`

Macro `Bidly_GC` is defined for use with anonymous manager.

Macros `Bidly_Managed_FullGC` and `Bidly_FullGC` are useful variants.

Definition at line 563 of file `bidly.h`.

**5.1.2.65 #define Bidy\_Clean( ) Biddy\_Managed\_Clean(NULL)**

Macro Bidy\_Clean is defined for use with anonymous manager.

Definition at line 570 of file bidy.h.

**5.1.2.66 #define Bidy\_Purge( ) Biddy\_Managed\_Purge(NULL)**

Macro Bidy\_Purge is defined for use with anonymous manager.

Definition at line 575 of file bidy.h.

**5.1.2.67 #define Bidy\_PurgeAndReorder( f, c ) Biddy\_Managed\_PurgeAndReorder(NULL,f,c)**

Macro Bidy\_PurgeAndReorder is defined for use with anonymous manager.

Definition at line 580 of file bidy.h.

**5.1.2.68 #define Bidy\_Refresh( f ) Biddy\_Managed\_Refresh(NULL,f)**

Macro Bidy\_Refresh is defined for use with anonymous manager.

Definition at line 585 of file bidy.h.

**5.1.2.69 #define Bidy\_AddCache( gc ) Biddy\_Managed\_AddCache(NULL,gc)**

Macro Bidy\_AddCache is defined for use with anonymous manager.

Definition at line 590 of file bidy.h.

**5.1.2.70 #define Bidy\_AddFormula( x, f, c ) Biddy\_Managed\_AddFormula(NULL,x,f,c)**

Macro Bidy\_AddFormula is defined for use with anonymous manager.

Macros Bidy\_Managed\_AddTmpFormula, Bidy\_AddTmpFormula,

Bidy\_Managed\_AddPreservedFormula, Bidy\_AddPreservedFormula,

Bidy\_Managed\_AddPersistentFormula, Bidy\_AddPersistentFormula,

Bidy\_Managed\_KeepFormula, Bidy\_KeepFormula,

Bidy\_Managed\_KeepFormulaUntilDelete, and Bidy\_KeepFormulaUntilDelete

are defined to simplify formulae management.

Definition at line 601 of file bidy.h.

5.1.2.71 **#define Bidly\_FindFormula( x, idx, f ) Bidly\_Managed\_FindFormula(NULL,x,idx,f)**

Macro Bidly\_FindFormula is defined for use with anonymous manager.

Definition at line 616 of file bidly.h.

5.1.2.72 **#define Bidly\_DeleteFormula( x ) Bidly\_Managed\_DeleteFormula(NULL,x)**

Macro Bidly\_DeleteFormula is defined for use with anonymous manager.

Definition at line 621 of file bidly.h.

5.1.2.73 **#define Bidly\_DeletelthFormula( x ) Bidly\_Managed\_DeletelthFormula(NULL,x)**

Macro Bidly\_DeletelthFormula is defined for use with anonymous manager.

Definition at line 626 of file bidly.h.

5.1.2.74 **#define Bidly\_GetlthFormula( i ) Bidly\_Managed\_GetlthFormula(NULL,i)**

Macro Bidly\_GetlthFormula is defined for use with anonymous manager.

Definition at line 631 of file bidly.h.

5.1.2.75 **#define Bidly\_GetlthFormulaName( i ) Bidly\_Managed\_GetlthFormulaName(NULL,i)**

Macro Bidly\_GetlthFormulaName is defined for use with anonymous manager.

Definition at line 636 of file bidly.h.

5.1.2.76 **#define Bidly\_SwapWithHigher( v ) Bidly\_Managed\_SwapWithHigher(NULL,v)**

Macro Bidly\_SwapWithHigher is defined for use with anonymous manager.

Definition at line 641 of file bidly.h.

5.1.2.77 **#define Bidly\_SwapWithLower( v ) Bidly\_Managed\_SwapWithLower(NULL,v)**

Macro Bidly\_SwapWithLower is defined for use with anonymous manager.

Definition at line 646 of file bidly.h.

5.1.2.78 **#define Bidly\_Sifting( f, c ) Bidly\_Managed\_Sifting(NULL,f,c)**

Macro Bidly\_Sifting is defined for use with anonymous manager.

Definition at line 651 of file bidly.h.

5.1.2.79 `#define Biddy_MinimizeBDD( f ) Biddy_Managed_MinimizeBDD(NULL,f)`

Macro Biddy\_MinimizeBDD is defined for use with anonymous manager.

Definition at line 656 of file biddy.h.

5.1.2.80 `#define Biddy_MaximizeBDD( f ) Biddy_Managed_MaximizeBDD(NULL,f)`

Macro Biddy\_MaximizeBDD is defined for use with anonymous manager.

Definition at line 661 of file biddy.h.

5.1.2.81 `#define Biddy_Copy( MNG2, f ) Biddy_Managed_Copy(NULL,MNG2,f)`

Macro Biddy\_Copy is defined for use with anonymous manager.

Definition at line 666 of file biddy.h.

5.1.2.82 `#define Biddy_CopyFormula( MNG2, x ) Biddy_Managed_CopyFormula(NULL,MNG2,x)`

Macro Biddy\_CopyFormula is defined for use with anonymous manager.

Definition at line 671 of file biddy.h.

5.1.2.83 `#define Biddy_Eval( f ) Biddy_Managed_Eval(NULL,f)`

Macro Biddy\_Eval is defined for use with anonymous manager.

Definition at line 676 of file biddy.h.

5.1.2.84 `#define Biddy_Not( f ) Biddy_Managed_Not(NULL,f)`

Macro Biddy\_Not is defined for use with anonymous manager.

For OBDD and OFDD, use macro Biddy\_Inv.

Definition at line 694 of file biddy.h.

5.1.2.85 `#define Biddy_ITE( f, g, h ) Biddy_Managed_ITE(NULL,f,g,h)`

Macro Biddy\_ITE is defined for use with anonymous manager.

Definition at line 699 of file biddy.h.

5.1.2.86 `#define Bidly_And( f, g ) Bidly_Managed_And(NULL,f,g)`

Macro `Bidly_And` is defined for use with anonymous manager.

Macros `Bidly_Managed_Intersect` and `Bidly_Intersect` are defined for set manipulation.

Definition at line 705 of file `bidly.h`.

5.1.2.87 `#define Bidly_Or( f, g ) Bidly_Managed_Or(NULL,f,g)`

Macro `Bidly_Or` is defined for use with anonymous manager.

Macros `Bidly_Managed_Union` and `Bidly_Union` are defined for set manipulation.

Definition at line 713 of file `bidly.h`.

5.1.2.88 `#define Bidly_Nand( f, g ) Bidly_Managed_Nand(NULL,f,g)`

Macro `Bidly_Nand` is defined for use with anonymous manager.

Definition at line 720 of file `bidly.h`.

5.1.2.89 `#define Bidly_Nor( f, g ) Bidly_Managed_Nor(NULL,f,g)`

Macro `Bidly_Nor` is defined for use with anonymous manager.

Definition at line 725 of file `bidly.h`.

5.1.2.90 `#define Bidly_Xor( f, g ) Bidly_Managed_Xor(NULL,f,g)`

Macro `Bidly_Xor` is defined for use with anonymous manager.

Definition at line 730 of file `bidly.h`.

5.1.2.91 `#define Bidly_Xnor( f, g ) Bidly_Managed_Xnor(NULL,f,g)`

Macro `Bidly_Xnor` is defined for use with anonymous manager.

Definition at line 735 of file `bidly.h`.

5.1.2.92 `#define Bidly_Leq( f, g ) Bidly_Managed_Leq(NULL,f,g)`

Macro `Bidly_Leq` is defined for use with anonymous manager.

Definition at line 740 of file `bidly.h`.

5.1.2.93 `#define Biddy_Gt( f, g ) Biddy_Managed_Gt(NULL,f,g)`

Macro Biddy\_Gt is defined for use with anonymous manager.

Definition at line 745 of file biddy.h.

5.1.2.94 `#define Biddy_IsLeq( f, g ) Biddy_Managed_IsLeq(NULL,f,g)`

Macro Biddy\_IsLeq is defined for use with anonymous manager.

Definition at line 752 of file biddy.h.

5.1.2.95 `#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 758 of file biddy.h.

5.1.2.96 `#define Biddy_Compose( f, g, v ) Biddy_Managed_Compose(NULL,f,g,v)`

Macro Biddy\_Compose is defined for use with anonymous manager.

Definition at line 763 of file biddy.h.

5.1.2.97 `#define Biddy_E( f, v ) Biddy_Managed_E(NULL,f,v)`

Macro Biddy\_E is defined for use with anonymous manager.

Definition at line 768 of file biddy.h.

5.1.2.98 `#define Biddy_A( f, v ) Biddy_Managed_A(NULL,f,v)`

Macro Biddy\_A is defined for use with anonymous manager.

Definition at line 773 of file biddy.h.

5.1.2.99 `#define Biddy_IsVariableDependent( f, v ) Biddy_Managed_IsVariableDependent(NULL,f,v)`

Macro Biddy\_IsVariableDependent is defined for use with anonymous manager.

Definition at line 778 of file biddy.h.

5.1.2.100 `#define Biddy_ExistAbstract( f, cube ) Biddy_Managed_ExistAbstract(NULL,f,cube)`

Macro Biddy\_ExistAbstract is defined for use with anonymous manager.

Definition at line 783 of file biddy.h.

5.1.2.101 `#define Bidly_UnivAbstract( f, cube ) Bidly_Managed_UnivAbstract(NULL,f,cube)`

Macro `Bidly_UnivAbstract` is defined for use with anonymous manager.

Definition at line 788 of file `bidly.h`.

5.1.2.102 `#define Bidly_AndAbstract( f, g, cube ) Bidly_Managed_AndAbstract(NULL,f,g,cube)`

Macro `Bidly_AndAbstract` is defined for use with anonymous manager.

Definition at line 793 of file `bidly.h`.

5.1.2.103 `#define Bidly_Constrain( f, c ) Bidly_Managed_Constrain(NULL,f,c)`

Macro `Bidly_Constrain` is defined for use with anonymous manager.

Definition at line 798 of file `bidly.h`.

5.1.2.104 `#define Bidly_Simplify( f, c ) Bidly_Managed_Simplify(NULL,f,c)`

Macro `Bidly_Simplify` is defined for use with anonymous manager.

Definition at line 804 of file `bidly.h`.

5.1.2.105 `#define Bidly_Support( f ) Bidly_Managed_Support(NULL,f)`

Macro `Bidly_Support` is defined for use with anonymous manager.

Definition at line 809 of file `bidly.h`.

5.1.2.106 `#define Bidly_ReplaceByKeyword( f, keyword ) Bidly_Managed_ReplaceByKeyword(NULL,f,keyword)`

Macro `Bidly_ReplaceByKeyword` is defined for use with anonymous manager.

Macros `Bidly_Managed_Replace` and `Bidly_Replace` are variants

with less effective cache table

Definition at line 816 of file `bidly.h`.

5.1.2.107 `#define Bidly_Change( f, v ) Bidly_Managed_Change(NULL,f,v)`

Macro `Bidly_Change` is defined for use with anonymous manager.

Definition at line 823 of file `bidly.h`.

5.1.2.108 `#define Biddy_Subset( f, v, value ) Biddy_Managed_Subset(NULL,f,v,value)`

Macro `Biddy_Subset` is defined for use with anonymous manager.

Definition at line 829 of file `biddy.h`.

5.1.2.109 `#define Biddy_CreateMinterm( support, x ) Biddy_Managed_CreateMinterm(NULL,support,x)`

Macro `Biddy_CreateMinterm` is defined for use with anonymous manager.

Definition at line 838 of file `biddy.h`.

5.1.2.110 `#define Biddy_CreateFunction( support, x ) Biddy_Managed_CreateFunction(NULL,support,x)`

Macro `Biddy_CreateFunction` is defined for use with anonymous manager.

Definition at line 843 of file `biddy.h`.

5.1.2.111 `#define Biddy_RandomFunction( support, r ) Biddy_Managed_RandomFunction(NULL,support,r)`

Macro `Biddy_RandomFunction` is defined for use with anonymous manager.

Definition at line 848 of file `biddy.h`.

5.1.2.112 `#define Biddy_RandomSet( unit, r ) Biddy_Managed_RandomSet(NULL,unit,r)`

Macro `Biddy_RandomSet` is defined for use with anonymous manager.

Definition at line 853 of file `biddy.h`.

5.1.2.113 `#define Biddy_CountNodes( f ) Biddy_Managed_CountNodes(NULL,f)`

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

Definition at line 874 of file `biddy.h`.

5.1.2.114 `#define Biddy_Managed_MaxLevel( MNG, f ) Biddy_MaxLevel(f)`

Macro `Biddy_Managed_MaxLevel(MNG,f)` is defined for your convenience.

Definition at line 879 of file `biddy.h`.

5.1.2.115 `#define Biddy_Managed_AvgLevel( MNG, f ) Biddy_AvgLevel(f)`

Macro `Biddy_Managed_AvgLevel(MNG,f)` is defined for your convenience.

Definition at line 884 of file `biddy.h`.



5.1.2.116 **#define Bidly\_VariableTableNum( ) Bidly\_Managed\_VariableTableNum(NULL)**

Macro Bidly\_VariableTableNum is defined for use with anonymous manager.

Definition at line 889 of file bidly.h.

5.1.2.117 **#define Bidly\_NodeTableSize( ) Bidly\_Managed\_NodeTableSize(NULL)**

Macro Bidly\_NodeTableSize is defined for use with anonymous manager.

Definition at line 894 of file bidly.h.

5.1.2.118 **#define Bidly\_NodeTableBlockNumber( ) Bidly\_Managed\_NodeTableBlockNumber(NULL)**

Macro Bidly\_NodeTableBlockNumber is defined for use with anonymous manager.

Definition at line 899 of file bidly.h.

5.1.2.119 **#define Bidly\_NodeTableGenerated( ) Bidly\_Managed\_NodeTableGenerated(NULL)**

Macro Bidly\_NodeTableGenerated is defined for use with anonymous manager.

Definition at line 904 of file bidly.h.

5.1.2.120 **#define Bidly\_NodeTableMax( ) Bidly\_Managed\_NodeTableMax(NULL)**

Macro Bidly\_NodeTableMax is defined for use with anonymous manager.

Definition at line 909 of file bidly.h.

5.1.2.121 **#define Bidly\_NodeTableNum( ) Bidly\_Managed\_NodeTableNum(NULL)**

Macro Bidly\_NodeTableNum is defined for use with anonymous manager.

Definition at line 914 of file bidly.h.

5.1.2.122 **#define Bidly\_NodeTableNumVar( v ) Bidly\_Managed\_NodeTableNumVar(NULL,v)**

Macro Bidly\_NodeTableNumVar is defined for use with anonymous manager.

Definition at line 919 of file bidly.h.

5.1.2.123 **#define Bidly\_NodeTableResizeNumber( ) Bidly\_Managed\_NodeTableResizeNumber(NULL)**

Macro Bidly\_NodeTableResizeNumber is defined for use with anonymous manager.

Definition at line 924 of file bidly.h.

5.1.2.124 `#define Biddy_NodeTableFoaNumber( ) Biddy_Managed_NodeTableFoaNumber(NULL)`

Macro `Biddy_NodeTableFoaNumber` is defined for use with anonymous manager.

Definition at line 929 of file `biddy.h`.

5.1.2.125 `#define Biddy_NodeTableFindNumber( ) Biddy_Managed_NodeTableFindNumber(NULL)`

Macro `Biddy_NodeTableFindNumber` is defined for use with anonymous manager.

Definition at line 934 of file `biddy.h`.

5.1.2.126 `#define Biddy_NodeTableCompareNumber( ) Biddy_Managed_NodeTableCompareNumber(NULL)`

Macro `Biddy_NodeTableCompareNumber` is defined for use with anonymous manager.

Definition at line 939 of file `biddy.h`.

5.1.2.127 `#define Biddy_NodeTableAddNumber( ) Biddy_Managed_NodeTableAddNumber(NULL)`

Macro `Biddy_NodeTableAddNumber` is defined for use with anonymous manager.

Definition at line 944 of file `biddy.h`.

5.1.2.128 `#define Biddy_NodeTableGCNumber( ) Biddy_Managed_NodeTableGCNumber(NULL)`

Macro `Biddy_NodeTableGCNumber` is defined for use with anonymous manager.

Definition at line 949 of file `biddy.h`.

5.1.2.129 `#define Biddy_NodeTableGCTime( ) Biddy_Managed_NodeTableGCTime(NULL)`

Macro `Biddy_NodeTableGCTime` is defined for use with anonymous manager.

Definition at line 954 of file `biddy.h`.

5.1.2.130 `#define Biddy_NodeTableGCObsoleteNumber( ) Biddy_Managed_NodeTableGCObsoleteNumber(NULL)`

Macro `Biddy_NodeTableGCObsoleteNumber` is defined for use with anonymous manager.

Definition at line 959 of file `biddy.h`.

5.1.2.131 `#define Biddy_NodeTableSwapNumber( ) Biddy_Managed_NodeTableSwapNumber(NULL)`

Macro `Biddy_NodeTableSwapNumber` is defined for use with anonymous manager.

Definition at line 964 of file `biddy.h`.

5.1.2.132 `#define Bidly_NodeTableSiftingNumber( ) Bidly_Managed_NodeTableSiftingNumber(NULL)`

Macro `Bidly_NodeTableSiftingNumber` is defined for use with anonymous manager.

Definition at line 969 of file `bidly.h`.

5.1.2.133 `#define Bidly_NodeTableDRTime( ) Bidly_Managed_NodeTableDRTime(NULL)`

Macro `Bidly_NodeTableDRTime` is defined for use with anonymous manager.

Definition at line 974 of file `bidly.h`.

5.1.2.134 `#define Bidly_NodeTableITENumber( ) Bidly_Managed_NodeTableITENumber(NULL)`

Macro `Bidly_NodeTableITENumber` is defined for use with anonymous manager.

Definition at line 979 of file `bidly.h`.

5.1.2.135 `#define Bidly_NodeTableITERRecursiveNumber( ) Bidly_Managed_NodeTableITERRecursiveNumber(NULL)`

Macro `Bidly_NodeTableITERRecursiveNumber` is defined for use with anonymous manager.

Definition at line 984 of file `bidly.h`.

5.1.2.136 `#define Bidly_NodeTableANDORNumber( ) Bidly_Managed_NodeTableANDORNumber(NULL)`

Macro `Bidly_NodeTableANDORNumber` is defined for use with anonymous manager.

Definition at line 989 of file `bidly.h`.

5.1.2.137 `#define Bidly_NodeTableANDORRecursiveNumber( ) Bidly_Managed_NodeTableANDORRecursiveNumber(NULL)`

Macro `Bidly_NodeTableANDORRecursiveNumber` is defined for use with anonymous manager.

Definition at line 994 of file `bidly.h`.

5.1.2.138 `#define Bidly_NodeTableXORNumber( ) Bidly_Managed_NodeTableXORNumber(NULL)`

Macro `Bidly_NodeTableXORNumber` is defined for use with anonymous manager.

Definition at line 999 of file `bidly.h`.

5.1.2.139 **#define Biddy\_NodeTableXORRecursiveNumber( ) Biddy\_Managed\_NodeTableXORRecursiveNumber(NULL)**

Macro Biddy\_NodeTableXORRecursiveNumber is defined for use with anonymous manager.

Definition at line 1004 of file biddy.h.

5.1.2.140 **#define Biddy\_FormulaTableNum( ) Biddy\_Managed\_FormulaTableNum(NULL)**

Macro Biddy\_FormulaTableNum is defined for use with anonymous manager.

Definition at line 1009 of file biddy.h.

5.1.2.141 **#define Biddy\_ListUsed( ) Biddy\_Managed\_ListUsed(NULL)**

Macro Biddy\_ListUsed is defined for use with anonymous manager.

Definition at line 1014 of file biddy.h.

5.1.2.142 **#define Biddy\_ListMaxLength( ) Biddy\_Managed\_ListMaxLength(NULL)**

Macro Biddy\_ListMaxLength is defined for use with anonymous manager.

Definition at line 1019 of file biddy.h.

5.1.2.143 **#define Biddy\_ListAvgLength( ) Biddy\_Managed\_ListAvgLength(NULL)**

Macro Biddy\_ListAvgLength is defined for use with anonymous manager.

Definition at line 1024 of file biddy.h.

5.1.2.144 **#define Biddy\_OPCCacheSearch( ) Biddy\_Managed\_OPCCacheSearch(NULL)**

Macro Biddy\_OPCCacheSearch is defined for use with anonymous manager.

Definition at line 1029 of file biddy.h.

5.1.2.145 **#define Biddy\_OPCCacheFind( ) Biddy\_Managed\_OPCCacheFind(NULL)**

Macro Biddy\_OPCCacheFind is defined for use with anonymous manager.

Definition at line 1034 of file biddy.h.

5.1.2.146 **#define Bidly\_OPCCacheInsert( ) Bidly\_Managed\_OPCCacheInsert(NULL)**

Macro Bidly\_OPCCacheInsert is defined for use with anonymous manager.

Definition at line 1039 of file bidly.h.

5.1.2.147 **#define Bidly\_OPCCacheOverwrite( ) Bidly\_Managed\_OPCCacheOverwrite(NULL)**

Macro Bidly\_OPCCacheOverwrite is defined for use with anonymous manager.

Definition at line 1044 of file bidly.h.

5.1.2.148 **#define Bidly\_CountNodesPlain( f ) Bidly\_Managed\_CountNodesPlain(NULL,f)**

Macro Bidly\_CountNodesPlain is defined for use with anonymous manager.

Definition at line 1049 of file bidly.h.

5.1.2.149 **#define Bidly\_DependentVariableNumber( f ) Bidly\_Managed\_DependentVariableNumber(NULL,f)**

Macro Bidly\_DependentVariableNumber is defined for use with anonymous manager.

Definition at line 1054 of file bidly.h.

5.1.2.150 **#define Bidly\_CountComplementedEdges( f ) Bidly\_Managed\_CountComplementedEdges(NULL,f)**

Macro Bidly\_CountComplementedEdges is defined for use with anonymous manager.

Definition at line 1059 of file bidly.h.

5.1.2.151 **#define Bidly\_CountPaths( f ) Bidly\_Managed\_CountPaths(NULL,f)**

Macro Bidly\_CountPaths is defined for use with anonymous manager.

Definition at line 1064 of file bidly.h.

5.1.2.152 **#define Bidly\_CountMinterms( f, nvars ) Bidly\_Managed\_CountMinterms(NULL,f,nvars)**

Macro Bidly\_CountMinterms is defined for use with anonymous manager.

Definition at line 1069 of file bidly.h.

5.1.2.153 **#define Bidly\_DensityOfFunction( f, nvars ) Bidly\_Managed\_DensityOfFunction(NULL,f,nvars)**

Macro Bidly\_DensityOfFunction is defined for use with anonymous manager.

Definition at line 1076 of file bidly.h.

5.1.2.154 `#define Biddy_DensityOfBDD( f, nvars ) Biddy_Managed_DensityOfBDD(NULL,f,nvars)`

Macro Biddy\_DensityOfBDD is defined for use with anonymous manager.

Definition at line 1081 of file biddy.h.

5.1.2.155 `#define Biddy_ReadMemoryInUse( ) Biddy_Managed_ReadMemoryInUse(NULL)`

Macro Biddy\_ReadMemoryInUse is defined for use with anonymous manager.

Definition at line 1086 of file biddy.h.

5.1.2.156 `#define Biddy_PrintInfo( f ) Biddy_Managed_PrintInfo(NULL,f)`

Macro Biddy\_PrintInfo is defined for use with anonymous manager.

Definition at line 1091 of file biddy.h.

5.1.2.157 `#define Biddy_Eval0( s ) Biddy_Managed_Eval0(NULL,s)`

Macro Biddy\_Eval0 is defined for use with anonymous manager.

Definition at line 1108 of file biddy.h.

5.1.2.158 `#define Biddy_Eval1x( s, lf ) Biddy_Managed_Eval1x(NULL,s,lf)`

Macro Biddy\_Eval1x is defined for use with anonymous manager.

Definition at line 1113 of file biddy.h.

5.1.2.159 `#define Biddy_Eval2( boolFunc ) Biddy_Managed_Eval2(NULL,boolFunc)`

Macro Biddy\_Eval2 is defined for use with anonymous manager.

Definition at line 1120 of file biddy.h.

5.1.2.160 `#define Biddy_ReadVerilogFile( filename, prefix ) Biddy_Managed_ReadVerilogFile(NULL,filename,prefix)`

Macro Biddy\_ReadVerilogFile is defined for use with anonymous manager.

Definition at line 1125 of file biddy.h.

5.1.2.161 `#define Biddy_PrintfBDD( f ) Biddy_Managed_PrintfBDD(NULL,f)`

Macro Biddy\_PrintfBDD is defined for use with anonymous manager.

Definition at line 1130 of file biddy.h.

5.1.2.162 `#define Bidly_WriteBDD( filename, f, label ) Bidly_Managed_WriteBDD(NULL,filename,f,label)`

Macro `Bidly_WriteBDD` is defined for use with anonymous manager.

Definition at line 1135 of file `bidly.h`.

5.1.2.163 `#define Bidly_PrintfTable( f ) Bidly_Managed_PrintfTable(NULL,f)`

Macro `Bidly_PrintfTable` is defined for use with anonymous manager.

Definition at line 1140 of file `bidly.h`.

5.1.2.164 `#define Bidly_WriteTable( filename, f ) Bidly_Managed_WriteTable(NULL,filename,f)`

Macro `Bidly_WriteTable` is defined for use with anonymous manager.

Definition at line 1145 of file `bidly.h`.

5.1.2.165 `#define Bidly_PrintfSOP( f ) Bidly_Managed_PrintfSOP(NULL,f)`

Macro `Bidly_PrintfSOP` is defined for use with anonymous manager.

Definition at line 1150 of file `bidly.h`.

5.1.2.166 `#define Bidly_WriteSOP( filename, f ) Bidly_Managed_WriteSOP(NULL,filename,f)`

Macro `Bidly_WriteSOP` is defined for use with anonymous manager.

Definition at line 1155 of file `bidly.h`.

5.1.2.167 `#define Bidly_WriteDot( filename, f, label, id, cudd ) Bidly_Managed_WriteDot(NU↔  
LL,filename,f,label,id,cudd);`

Macro `Bidly_WriteDot` is defined for use with anonymous manager.

Definition at line 1160 of file `bidly.h`.

5.1.2.168 `#define Bidly_WriteBddview( filename, f, label, table ) Bidly_Managed_WriteBddview(NU↔  
LL,filename,f,label,table);`

Macro `Bidly_WriteBddview` is defined for use with anonymous manager.

Definition at line 1165 of file `bidly.h`.

### 5.1.3 Typedef Documentation

#### 5.1.3.1 typedef char **Biddy\_Boolean**

Biddy\_Boolean is used for boolean values.

Definition at line 232 of file biddy.h.

#### 5.1.3.2 typedef char\* **Biddy\_String**

Biddy\_String is used for strings.

Definition at line 235 of file biddy.h.

#### 5.1.3.3 typedef void\*\* **Biddy\_Manager**

Biddy\_Manager is used to specify manager. Manager is a pointer to BidyManager. 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, system age and some other structures needed for memory management. Internal structure of BidyManager is not exported but must be imitated to create user's managers

Definition at line 244 of file biddy.h.

#### 5.1.3.4 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 249 of file biddy.h.

#### 5.1.3.5 typedef unsigned short int **Biddy\_Variable**

Biddy\_Variable is used for indices in variable table.

Definition at line 253 of file biddy.h.

#### 5.1.3.6 typedef void\* **Biddy\_Edge**

Biddy\_Edge is a marked edge (i.e. a marked pointer to BidyNode). Mark is encoded as the value of the last significant bit. For TZBDDs and TZFDDs, edges are tagged. Tag is a 16 bit number (unsigned short int) which is stored in the highest part of the pointer (this is safe because only 48 bits are used). TZBDDs and TZFDDs are supported only on 64-bits architectures. Internal structure of BidyNode is not visible to the user.

Definition at line 262 of file biddy.h.



### 5.1.3.7 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 266 of file biddy.h.

### 5.1.3.8 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 270 of file biddy.h.

## 5.2 biddlyInOut.c File Reference

File [biddlyInOut.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\\_Eval1x](#) (Biddy\_Manager MNG, Biddy\_String s, Biddy\_LookupFunction lf)  
*Function Biddy\_Managed\_Eval1x 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\\_ReadVerilogFile](#) (Biddy\_Manager MNG, const char filename[], Biddy\_String prefix)  
*Function Biddy\_Managed\_ReadVerilogFile reads Verilog file and creates variables for all primary inputs and Boolean functions for all primary outputs.*
- void [Biddy\\_Managed\\_PrintfBDD](#) (Biddy\_Manager MNG, Biddy\_Edge f)  
*Function Biddy\_Managed\_PrintfBDD writes raw format using printf.*
- void [Biddy\\_Managed\\_WriteBDD](#) (Biddy\_Manager MNG, const char filename[], Biddy\_Edge f, Biddy\_String label)  
*Function Biddy\_Managed\_WriteBDD writes raw format using fprintf.*
- void [Biddy\\_Managed\\_PrintfTable](#) (Biddy\_Manager MNG, Biddy\_Edge f)  
*Function Biddy\_Managed\_PrintfTable writes truth table using printf.*
- void [Biddy\\_Managed\\_WriteTable](#) (Biddy\_Manager MNG, const char filename[], Biddy\_Edge f)  
*Function Biddy\_Managed\_WriteTable writes truth table using fprintf.*
- void [Biddy\\_Managed\\_PrintfSOP](#) (Biddy\_Manager MNG, Biddy\_Edge f)  
*Function Biddy\_Managed\_PrintfSOP writes SOP using printf.*
- void [Biddy\\_Managed\\_WriteSOP](#) (Biddy\_Manager MNG, const char filename[], Biddy\_Edge f)  
*Function Biddy\_Managed\_WriteSOP writes SOP using fprintf.*
- unsigned int [Biddy\\_Managed\\_WriteDot](#) (Biddy\_Manager MNG, const char filename[], Biddy\_Edge f, const char label[], int id, Biddy\_Boolean cudd)  
*Function Biddy\_Managed\_WriteDot writes dot/graphviz format using fprintf.*
- unsigned int [Biddy\\_Managed\\_WriteBddview](#) (Biddy\_Manager MNG, const char filename[], Biddy\_Edge f, const char label[], Biddy\_XY \*table)  
*Function Biddy\_Managed\_WriteBDDView writes bddview format using fprintf.*

## 5.2.1 Detailed Description

File [bidlyInOut.c](#) contains various parsers and generators.

### Description

```

PackageName [Bidly]
Synopsis [Bidly provides data structures and algorithms for the
         representation and manipulation of Boolean functions with
         ROBDDs, 0-sup-BDDs, and TZBDDs. A hash table is used for quick
         search of nodes. Complement edges decreases the number of
         nodes. An automatic garbage collection with a system age is
         implemented. Variable swapping and sifting are implemented.]

FileName [bidlyInOut.c]
Revision [${Revision: 355 $}]
Date [${Date: 2017-12-12 22:57:11 +0100 (tor, 12 dec 2017) $}]
Authors [Robert Meolic (robert.meolic@um.si),
        Ales Casar (ales@homemade.net),
        Jan Kraner (jankristian.kraner@student.um.si),
        Ziga Kobale (ziga.kobale@student.um.si),
        Volodymyr Mihav (mihaw.wolodymyr@gmail.com),
        David Kebo Houngninou (dhoungninou@smu.edu)]

```

### Copyright

Copyright (C) 2006, 2017 UM FERi, Koroska cesta 46, SI-2000 Maribor, Slovenia

Bidly 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.

Bidly 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: [bidly.h](#), [bidlyInt.h](#)

## 5.2.2 Function Documentation

### 5.2.2.1 Bidly\_String Bidly\_Managed\_Eval0 ( Bidly\_Manager MNG, Bidly\_String s )

Function Bidly\_Managed\_Eval0 evaluates raw format.

#### Description

First word is a name. It is followed by raw format. Function return name of the formula.

#### Side effects

All variables should already exists in the correct ordering! Not reentrant.

**More info**

Macro [Biddy\\_Eval0\(s\)](#) is defined for use with anonymous manager.

Definition at line 241 of file biddyInOut.c.

**5.2.2.2 Biddy\_Edge Biddy\_Managed\_Eval1x ( Biddy\_Manager *MNG*, Biddy\_String *s*, Biddy\_LookupFunction *lf* )**

Function `Biddy_Managed_Eval1x` evaluates prefix AND-OR-EXOR-NOT format.

**Description**

Parameter *lf* is a lookup function in the user-defined cache table.

**Side effects**

Not reentrant.

**More info**

Macro [Biddy\\_Eval1x\(s,lf\)](#) is defined for use with anonymous manager. Macros `Biddy_Managed_Eval1(s)` and `Biddy_Eval1(s)` are defined for use without searching in the user-defined cache.

Definition at line 318 of file biddyInOut.c.

**5.2.2.3 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**

Original author: Volodymyr Mihav ([mihaw.wolodymyr@gmail.com](mailto: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 417 of file biddyInOut.c.

**5.2.2.4 void Biddy\_Managed\_ReadVerilogFile ( Biddy\_Manager *MNG*, const char *filename*[], Biddy\_String *prefix* )**

Function `Biddy_Managed_ReadVerilogFile` reads Verilog file and creates variables for all primary inputs and Boolean functions for all primary outputs.

**Description**

If (prefix != NULL) then the created BDD variables and formulae will get it.

**Side effects****More info**

Original author: David Kebo Houngrinou, Southern Methodist University Original implementation of this function is on <https://github.com/davidkebo/verilog-parser> Macro [Bidly\\_ReadVerilogFile\(filename,prefix\)](#) is defined for use with anonymous manager.

Definition at line 610 of file `bidlyInOut.c`.

**5.2.2.5 void Bidly\_Managed\_PrintfBDD ( Bidly\_Manager MNG, Bidly\_Edge f )**

Function `Bidly_Managed_PrintfBDD` writes raw format using `printf`.

**Description**

This function is intended for writing to an output channel via macro which overrides the meaning of standard `printf` calls. For writing raw format into the file, use `Bidly_Managed_WriteBDD`.

**Side effects****More info**

Macro [Bidly\\_PrintfBDD\(f\)](#) is defined for use with anonymous manager.

Definition at line 711 of file `bidlyInOut.c`.

**5.2.2.6 void Bidly\_Managed\_WriteBDD ( Bidly\_Manager MNG, const char filename[], Bidly\_Edge f, Bidly\_String label )**

Function `Bidly_Managed_WriteBDD` writes raw format using `fprintf`.

**Description****Side effects****More info**

Macro [Bidly\\_WriteBDD\(f\)](#) is defined for use with anonymous manager.

Definition at line 741 of file `bidlyInOut.c`.

**5.2.2.7 void Bidly\_Managed\_PrintfTable ( Bidly\_Manager MNG, Bidly\_Edge f )**

Function `Bidly_Managed_PrintfTable` writes truth table using `printf`.

### Description

This function is intended for writing to an output channel via macro which overrides the meaning of standard printf calls. For writing truth table into the file, use `Biddy_Managed_WriteTable`.

### Side effects

### More info

Thanks to Jan Kraner ([jankristian.kraner@student.um.si](mailto:jankristian.kraner@student.um.si)) and Ziga Kobale ([ziga.kobale@student.um.si](mailto:ziga.kobale@student.um.si)). Macro `Biddy_PrintfTable(f)` is defined for use with anonymous manager.

Definition at line 788 of file `biddlyInOut.c`.

#### 5.2.2.8 void `Biddy_Managed_WriteTable` ( `Biddy_Manager MNG`, `const char filename[]`, `Biddy_Edge f` )

Function `Biddy_Managed_WriteTable` writes truth table using `fprintf`.

### Description

### Side effects

### More info

Thanks to Jan Kraner ([jankristian.kraner@student.um.si](mailto:jankristian.kraner@student.um.si)) and Ziga Kobale ([ziga.kobale@student.um.si](mailto:ziga.kobale@student.um.si)). Macro `Biddy_WriteTable(f)` is defined for use with anonymous manager.

Definition at line 896 of file `biddlyInOut.c`.

#### 5.2.2.9 void `Biddy_Managed_PrintfSOP` ( `Biddy_Manager MNG`, `Biddy_Edge f` )

Function `Biddy_Managed_PrintfSOP` writes SOP using `printf`.

### Description

### Side effects

### More info

Definition at line 918 of file `biddlyInOut.c`.

#### 5.2.2.10 void `Biddy_Managed_WriteSOP` ( `Biddy_Manager MNG`, `const char filename[]`, `Biddy_Edge f` )

Function `Biddy_Managed_WriteSOP` writes SOP using `fprintf`.

### Description

### Side effects

### More info

Definition at line 996 of file biddyInOut.c.

5.2.2.11 `unsigned int Biddy_Managed_WriteDot ( Biddy_Manager MNG, const char filename[], Biddy_Edge f, const char label[], int id, Biddy_Boolean cudd )`

Function `Biddy_Managed_WriteDot` writes dot/graphviz format using `fprintf`.

### Description

Output dot format. Two approaches are implemented. The CUDD-like implementation is copied from CUDD 3.0.

### Side effects

If (`id != -1`) then `id` is used instead of `<...>` for variable names. If (`filename == NULL`) then output is to `stdout`. Function resets all variables value.

### More info

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

Definition at line 1111 of file biddyInOut.c.

5.2.2.12 `unsigned int Biddy_Managed_WriteBddview ( Biddy_Manager MNG, const char filename[], Biddy_Edge f, const char label[], Biddy_XY * table )`

Function `Biddy_Managed_WriteBDDView` writes bddview format using `fprintf`.

### Description

Output bddview format.

### Side effects

Parameter `table` is optional, if not `NULL` then it must contain node names and coordinates. If (`filename == NULL`) then output is to `stdout`.

### More info

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

Definition at line 1330 of file biddyInOut.c.

## 5.3 biddyInt.h File Reference

File [biddyInt.h](#) contains declaration of internal data structures.

```
#include "biddy.h"  
#include <assert.h>  
#include <string.h>  
#include <ctype.h>  
#include <gmp.h>
```

### 5.3.1 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, 0-sup-BDDs, and TZBDDs. A hash table is used for quick  
search of nodes. Complement edges decreases the number of  
nodes. An automatic garbage collection with a system age is  
implemented. Variable swapping and sifting are implemented.]  
  
FileName [biddyInt.h]  
Revision [${Revision: 360 $}]  
Date [${Date: 2017-12-16 09:23:48 +0100 (sob, 16 dec 2017) $}]  
Authors [Robert Meolic (robert.meolic@um.si),  
Ales Casar (ales@homemade.net)]
```

#### Copyright

Copyright (C) 2006, 2017 UM FERi, Koroska cesta 46, 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](#)

## 5.4 biddyMain.c File Reference

File [biddyMain.c](#) contains main functions for representation and manipulation of boolean functions with various types of Binary Decision Diagrams (GDD = general decision diagrams).

```
#include "biddyInt.h"
```

## Functions

- void [Biddy\\_InitMNG](#) ([Biddy\\_Manager](#) \*mng, int gddtype)  
*Function Biddy\_InitMNG initialize a manager.*
- void [Biddy\\_ExitMNG](#) ([Biddy\\_Manager](#) \*mng)  
*Function Biddy\_ExitMNG deletes a manager.*
- [Biddy\\_String Biddy\\_About](#) ()  
*Function Biddy\_About reports version of Biddy package.*
- int [Biddy\\_Managed\\_GetManagerType](#) ([Biddy\\_Manager](#) MNG)  
*Function Biddy\_Managed\_GetManagerType reports BDD type used in the manager.*
- [Biddy\\_String Biddy\\_Managed\\_GetManagerName](#) ([Biddy\\_Manager](#) MNG)  
*Function Biddy\_Managed\_GetManagerName reports the name of the BDD type used in the manager.*
- void [Biddy\\_Managed\\_SetManagerParameters](#) ([Biddy\\_Manager](#) MNG, float gcr, float gcrF, float gcrX, float rr, float rrF, float rrX, float st, float cst)  
*Function Biddy\_Managed\_SetManagerParameters set modifiable parameters.*
- [Biddy\\_Edge Biddy\\_GetThen](#) ([Biddy\\_Edge](#) fun)  
*Function Biddy\_GetThen returns THEN successor.*
- [Biddy\\_Edge Biddy\\_GetElse](#) ([Biddy\\_Edge](#) fun)  
*Function Biddy\_GetElse returns ELSE successor.*
- [Biddy\\_Variable Biddy\\_GetTopVariable](#) ([Biddy\\_Edge](#) fun)  
*Function Biddy\_GetTopVariable returns the top variable.*
- [Biddy\\_Boolean Biddy\\_Managed\\_IsEqv](#) ([Biddy\\_Manager](#) MNG1, [Biddy\\_Edge](#) f1, [Biddy\\_Manager](#) MNG2, [Biddy\\_Edge](#) f2)  
*Function Biddy\_Managed\_IsEqv returns TRUE iff two BDDs are equal.*
- void [Biddy\\_Managed\\_SelectNode](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Edge](#) f)  
*Function Biddy\_Managed\_SelectNode selects the top node of the given function.*
- void [Biddy\\_Managed\\_DeselectNode](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Edge](#) f)  
*Function Biddy\_Managed\_DeselectNode deselects the top node of the given function.*
- [Biddy\\_Boolean Biddy\\_Managed\\_IsSelected](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Edge](#) f)  
*Function Biddy\_Managed\_IsSelected returns TRUE iff the top node of the given function is selected.*
- void [Biddy\\_Managed\\_SelectFunction](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Edge](#) f)  
*Function Biddy\_Managed\_SelectFunction recursively selects all nodes of a given function.*
- void [Biddy\\_Managed\\_DeselectAll](#) ([Biddy\\_Manager](#) MNG)  
*Function Biddy\_Managed\_DeselectAll deselects all nodes.*
- [Biddy\\_Edge Biddy\\_Managed\\_GetTerminal](#) ([Biddy\\_Manager](#) MNG)  
*Function Biddy\_Managed\_GetTerminal returns unmarked and untagged edge pointing to terminal node 1.*
- [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\\_Edge Biddy\\_Managed\\_GetBaseSet](#) ([Biddy\\_Manager](#) MNG)  
*Function Biddy\_Managed\_GetBaseSet returns set containing only a null combination, i.e. it returns {{{}}.*
- [Biddy\\_Variable Biddy\\_Managed\\_GetVariable](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_String](#) x)  
*Function Biddy\_Managed\_GetVariable returns variable with the given name.*
- [Biddy\\_Variable Biddy\\_Managed\\_GetLowestVariable](#) ([Biddy\\_Manager](#) MNG)  
*Function Biddy\_Managed\_GetLowestVariable returns the lowest variable in the current ordering.*
- [Biddy\\_Variable Biddy\\_Managed\\_GetIthVariable](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Variable](#) i)  
*Function Biddy\_Managed\_GetIthVariable returns ith variable in the current global ordering.*
- [Biddy\\_Variable Biddy\\_Managed\\_GetPrevVariable](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Variable](#) v)  
*Function Biddy\_Managed\_GetPrevVariable returns previous variable in the global ordering (lower, topmore).*
- [Biddy\\_Variable Biddy\\_Managed\\_GetNextVariable](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Variable](#) v)



- Function Bidy\_Managed\_GetNextVariable returns next variable in the global ordering (higher, bottommore).*

  - [Bidy\\_Edge Bidy\\_Managed\\_GetVariableEdge](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Variable](#) v)

*Function Bidy\_Managed\_GetVariableEdge returns variable's edge.*
- [Bidy\\_Edge Bidy\\_Managed\\_GetElementEdge](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Variable](#) v)

*Function Bidy\_Managed\_GetElementEdge returns element's edge.*
- [Bidy\\_String Bidy\\_Managed\\_GetVariableName](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Variable](#) v)

*Function Bidy\_Managed\_GetVariableName returns the name of a variable.*
- [Bidy\\_Edge Bidy\\_Managed\\_GetTopVariableEdge](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f)

*Function Bidy\_Managed\_GetTopVariableEdge returns variable's edge of top variable.*
- [Bidy\\_String Bidy\\_Managed\\_GetTopVariableName](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f)

*Function Bidy\_Managed\_GetTopVariableName returns the name of top variable.*
- [char Bidy\\_Managed\\_GetTopVariableChar](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f)

*Function Bidy\_Managed\_GetTopVariableChar returns the first character in the name of top variable.*
- [void Bidy\\_Managed\\_ResetVariablesValue](#) ([Bidy\\_Manager](#) MNG)

*Function Bidy\_Managed\_ResetVariablesValue sets all variable's value to bidyZero.*
- [void Bidy\\_Managed\\_SetVariableValue](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Variable](#) v, [Bidy\\_Edge](#) f)

*Function Bidy\_Managed\_SetVariableValue sets variable's value.*
- [Bidy\\_Edge Bidy\\_Managed\\_GetVariableValue](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Variable](#) v)

*Function Bidy\_Managed\_GetVariableValue gets variable's value.*
- [void Bidy\\_Managed\\_ClearVariablesData](#) ([Bidy\\_Manager](#) MNG)

*Function Bidy\_Managed\_ClearVariablesData free memory used for all variable's data.*
- [void Bidy\\_Managed\\_SetVariableData](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Variable](#) v, [void \\*x](#))

*Function Bidy\_Managed\_SetVariableData sets variable's data.*
- [void \\* Bidy\\_Managed\\_GetVariableData](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Variable](#) v)

*Function Bidy\_Managed\_GetVariableData gets variable's data.*
- [Bidy\\_Boolean Bidy\\_Managed\\_IsSmaller](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Variable](#) fv, [Bidy\\_Variable](#) gv)

*Function Bidy\_Managed\_IsSmaller returns TRUE if the first variable is smaller (= lower = previous = above = topmore).*
- [Bidy\\_Boolean Bidy\\_Managed\\_IsLowest](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Variable](#) v)

*Function Bidy\_Managed\_IsLowest returns TRUE if the variable is the lowest one (lowest == topmost).*
- [Bidy\\_Boolean Bidy\\_Managed\\_IsHighest](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Variable](#) v)

*Function Bidy\_Managed\_IsHighest returns TRUE if the variable is the highest one if terminal node is ignored (highest == bottommost).*
- [Bidy\\_Variable Bidy\\_Managed\\_FoaVariable](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_String](#) x, [Bidy\\_Boolean](#) varelem)

*Function Bidy\_Managed\_FoaVariable finds variable/element or adds new variable (i.e. Boolean function f = x) and new element (i.e. it creates set {{x}}).*
- [void Bidy\\_Managed\\_ChangeVariableName](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Variable](#) v, [Bidy\\_String](#) x)

*Function Bidy\_Managed\_ChangeVariableName set new name to the given variable/element.*
- [Bidy\\_Edge Bidy\\_Managed\\_AddVariableByName](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_String](#) x)

*Function Bidy\_Managed\_AddVariableByName adds variable.*
- [Bidy\\_Edge Bidy\\_Managed\\_AddElementByName](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_String](#) x)

*Function Bidy\_Managed\_AddElementByName adds element.*
- [Bidy\\_Edge Bidy\\_Managed\\_AddVariableBelow](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Variable](#) v)

*Function Bidy\_Managed\_AddVariableBelow adds variable.*
- [Bidy\\_Edge Bidy\\_Managed\\_AddVariableAbove](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Variable](#) v)

*Function Bidy\_Managed\_AddVariableAbove adds variable.*
- [Bidy\\_Edge Bidy\\_Managed\\_TransferMark](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f, [Bidy\\_Boolean](#) mark, [Bidy\\_Boolean](#) leftright)

*Function Bidy\_Managed\_TransferMark returns edge with inverted complement bit iff the second parameter is TRUE and normalization rules require this.*
- [Bidy\\_Edge Bidy\\_Managed\\_IncTag](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f)

- Function Bidly\_Managed\_IncTag returns edge with an incremented tag.*

  - [Bidly\\_Edge Bidly\\_Managed\\_TaggedFoaNode](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Variable](#) v, [Bidly\\_Edge](#) pf, [Bidly\\_Edge](#) pt, [Bidly\\_Variable](#) ptag, [Bidly\\_Boolean](#) garbageAllowed)

*Function Bidly\_Managed\_TaggedFoaNode finds or adds new node with the given variable and successors.*

  - [Bidly\\_Boolean Bidly\\_Managed\\_IsOK](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f)

*Function Bidly\_Managed\_IsOK returns TRUE iff given node is not obsolete.*

  - void [Bidly\\_Managed\\_GC](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Variable](#) targetLT, [Bidly\\_Variable](#) targetGEQ, [Bidly\\_Boolean](#) purge, [Bidly\\_Boolean](#) total)

*Function Bidly\_Managed\_GC performs garbage collection.*

  - void [Bidly\\_Managed\\_Clean](#) ([Bidly\\_Manager](#) MNG)

*Function Bidly\_Managed\_Clean performs cleaning.*

  - void [Bidly\\_Managed\\_Purge](#) ([Bidly\\_Manager](#) MNG)

*Function Bidly\_Managed\_Purge immediately removes all nodes which were not preserved or which are not preserved anymore.*

  - void [Bidly\\_Managed\\_PurgeAndReorder](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f, [Bidly\\_Boolean](#) converge)

*Function Bidly\_Managed\_PurgeAndReorder immediately removes non-preserved nodes and triggers reordering on function.*

  - void [Bidly\\_Managed\\_Refresh](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f)

*Function Bidly\_Managed\_Refresh refreshes top node in a given function.*

  - void [Bidly\\_Managed\\_AddCache](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_GCFunction](#) gc)

*Function Bidly\_Managed\_AddCache adds cache to the end of Cache list.*

  - unsigned int [Bidly\\_Managed\\_AddFormula](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_String](#) x, [Bidly\\_Edge](#) f, int c)

*Function Bidly\_Managed\_AddFormula adds formula to Formula table.*

  - [Bidly\\_Boolean Bidly\\_Managed\\_FindFormula](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_String](#) x, unsigned int \*idx, [Bidly\\_Edge](#) \*f)

*Function Bidly\_Managed\_FindFormula find formula in Formula table.*

  - [Bidly\\_Boolean Bidly\\_Managed\\_DeleteFormula](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_String](#) x)

*Function Bidly\_Managed\_DeleteFormula delete formula from Formula table.*

  - [Bidly\\_Boolean Bidly\\_Managed\\_DeletelthFormula](#) ([Bidly\\_Manager](#) MNG, unsigned int i)

*Function Bidly\_Managed\_DeletelthFormula deletes formula from the table.*

  - [Bidly\\_Edge Bidly\\_Managed\\_GetlthFormula](#) ([Bidly\\_Manager](#) MNG, unsigned int i)

*Function Bidly\_Managed\_GetlthFormula returns ith formula in a Formula table.*

  - [Bidly\\_String Bidly\\_Managed\\_GetlthFormulaName](#) ([Bidly\\_Manager](#) MNG, unsigned int i)

*Function Bidly\_Managed\_GetlthFormulaName returns name of the ith formula in a Formula table.*

  - [Bidly\\_Variable Bidly\\_Managed\\_SwapWithHigher](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Variable](#) v)

*Function Bidly\_Managed\_SwapWithHigher swaps two adjacent variables.*

  - [Bidly\\_Variable Bidly\\_Managed\\_SwapWithLower](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Variable](#) v)

*Function Bidly\_Managed\_SwapWithLower swaps two adjacent variables.*

  - [Bidly\\_Boolean Bidly\\_Managed\\_Sifting](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f, [Bidly\\_Boolean](#) converge)

*Function Bidly\_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 [Bidly\\_Managed\\_MinimizeBDD](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_String](#) name)

*Function Bidly\_Managed\_MinimizeBDD reorders variables to minimize the node number of the given formula using an exhaustive search over all possible orderings.*

  - void [Bidly\\_Managed\\_MaximizeBDD](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_String](#) name)

*Function Bidly\_Managed\_MaximizeBDD reorders variables to maximize the node number of the given function using an exhaustive search over all possible orderings.*

  - [Bidly\\_Edge Bidly\\_Managed\\_Copy](#) ([Bidly\\_Manager](#) MNG1, [Bidly\\_Manager](#) MNG2, [Bidly\\_Edge](#) f)

*Function Bidly\_Managed\_Copy copies a graph from one manager to another manager which can use the same or different BDD type.*

**Description**

The function takes a graph from one manager and creates the same graph in another manager. If the managers do not use the same BDD type then a graph is converted. The resulting graph will represent the same Boolean function assuming the domain from the target manager. If `f = biddyZero` then only the domain is copied.

**Side effects**

If source and target manager are the same then function does nothing. The variable ordering of the created BDD is trying to follow the original ordering, but if some variables already exist in the target manager then the final ordering is adapted to the target manager. Please note, that indices of variables in the target manager may not be the same as in the source manager (for example, if source manager does not use initial ordering the indices in the target manager will follow the variable's ordering and not variable's original indices)

**More info**

Macro `Biddy_Copy(MNG2,f)` is defined for use with anonymous manager.

- void `Biddy_Managed_CopyFormula` (`Biddy_Manager` MNG1, `Biddy_Manager` MNG2, `Biddy_String` x)

Function `Biddy_Managed_CopyFormula` uses `Biddy_Managed_Copy` to copy a graph from one manager to another manager which can use the same or different BDD type.

**Description**

See `Biddy_Managed_Copy`.

**Side effects**

If source and target manager are the same then function does nothing. The variable ordering of created BDD is adapted to the target manager. The created formula is not preserved.

**More info**

Macro `Biddy_CopyFormula(MNG2,x)` is defined for use with anonymous manager.

- `Biddy_Boolean Biddy_Managed_Eval` (`Biddy_Manager` MNG, `Biddy_Edge` f)

Function `Biddy_Managed_Eval` returns the value of a Boolean function for a given variable assignment.

**Description****Side effects**

Variables must have values assigned. Variable is considered to be `FALSE` iff `variable.value == biddyZero`, whilst it is considered to be `TRUE`, otherwise.

**More info**

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

### 5.4.1 Detailed Description

File `biddyMain.c` contains main functions for representation and manipulation of boolean functions with various types of Binary Decision Diagrams (GDD = general decision diagrams).

**Description**

```

PackageName [Biddy]
Synopsis [Biddy provides data structures and algorithms for the
representation and manipulation of Boolean functions with
ROBDDs, 0-sup-BDDs, and TZBDDs. A hash table is used for quick
search of nodes. Complement edges decreases the number of
nodes. An automatic garbage collection with a system age is
implemented. Variable swapping and sifting are implemented.]

FileName [biddyMain.c]
Revision [${Revision: 362 $}]
Date [${Date: 2017-12-17 17:01:16 +0100 (ned, 17 dec 2017) $}]
Authors [Robert Meolic (robert.meolic@um.si)]

```

## Copyright

Copyright (C) 2006, 2017 UM FERi, Koroska cesta 46, 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](#)

## 5.4.2 Function Documentation

### 5.4.2.1 void Bidy\_InitMNG ( Bidy\_Manager \* mng, int gddtype )

Function Bidy\_InitMNG initialize a manager.

#### Description

Bidy\_InitMNG creates and initializes a manager. Initialization consists of creating manager structure (MNG), node table (biddyNodeTable), variable table (biddyVariableTable), formula table (biddyFormulaTable), four basic caches (biddyOPCache, biddyEACache, biddyRCCache, and biddyReplaceCache), and cache list (biddyCacheList). Bidy\_InitMNG also initializes constant edges (biddyOne, biddyZero), memory management and automatic garbage collection.

#### Side effects

Allocates a lot of memory.

#### More info

Macro Bidy\_InitAnonymous() will initialize anonymous manager. Macro [Bidy\\_Init\(\)](#) will initialize anonymous manager for ROBDDs.

Definition at line 178 of file biddyMain.c.

Here is the caller graph for this function:



#### 5.4.2.2 void Bidy\_ExitMNG ( Biddy\_Manager \* mng )

Function Bidy\_ExitMNG deletes a manager.

##### Description

Deallocates all memory allocated by Bidy\_InitMNG, Bidy\_FoaVariable, Bidy\_FoaNode etc.

##### Side effects

##### More info

Macro [Bidy\\_Exit\(\)](#) will delete anonymous manager.

Definition at line 854 of file biddyMain.c.

Here is the caller graph for this function:



#### 5.4.2.3 Bidy\_String Bidy\_About ( )

Function Bidy\_About reports version of Bidy package.

##### Description

##### Side effects

##### More info

Definition at line 1077 of file biddyMain.c.

#### 5.4.2.4 int Bidy\_Managed\_GetManagerType ( Biddy\_Manager MNG )

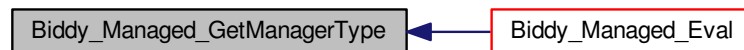
Function Bidy\_Managed\_GetManagerType reports BDD type used in the manager.

**Description****Side effects****More info**

Macro [Bidly\\_GetManagerType\(\)](#) is defined for use with anonymous manager.

Definition at line 1100 of file `biddyMain.c`.

Here is the caller graph for this function:



#### 5.4.2.5 Bidly\_String Bidly\_Managed\_GetManagerName ( Bidly\_Manager MNG )

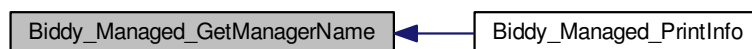
Function `Bidly_Managed_GetManagerName` reports the name of the BDD type used in the manager.

**Description****Side effects****More info**

Macro [Bidly\\_GetManagerName\(\)](#) is defined for use with anonymous manager.

Definition at line 1126 of file `biddyMain.c`.

Here is the caller graph for this function:



#### 5.4.2.6 void Bidly\_Managed\_SetManagerParameters ( Bidly\_Manager MNG, float gcr, float gcrF, float gcrX, float rr, float rrF, float rrX, float st, float cst )

Function `Bidly_Managed_SetManagerParameters` set modifiable parameters.

**Description**

Function expect 6 float values. If the value is  $< 0$  then the parameter is not modified. The parameters are: `biddyNodeTable.gcratio` (do not delete nodes if the effect is to small), `biddyNodeTable.gcratioF` (do not delete nodes if the effect is to small), `biddyNodeTable.gcratioX` (do not delete nodes if the effect is to small), `biddyNodeTable.resizeratio` (resize Node table if there are to many nodes), `biddyNodeTable.resizeratioF` (resize Node table if there are to many nodes), `biddyNodeTable.resizeratioX` (resize Node table if there are to many nodes), `biddyNodeTable.siftingtreshold` (stop sifting if the size of the system grows to much), `biddyNodeTable.fsiftingtreshold` (stop sifting if the size of the function grows to much), `biddyNodeTable.convergesiftingtreshold` (stop one step of converging sifting if the size of the system grows to much), `biddyNodeTable.fconvergesiftingtreshold` (stop one step of converging sifting if the size of the function grows to much).

**Side effects**

Initial values are given in `Biddy_InitMNG`.

**More info**

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

Definition at line 1169 of file `biddyMain.c`.

**5.4.2.7 Biddy\_Edge Biddy\_GetThen ( Biddy\_Edge fun )**

Function `Biddy_GetThen` returns THEN successor.

**Description**

Input mark is not transfered! External use, only.

**Side effects****More info**

Macro `BiddyT(fun)` is defined for internal use.

Definition at line 1213 of file `biddyMain.c`.

**5.4.2.8 Biddy\_Edge Biddy\_GetElse ( Biddy\_Edge fun )**

Function `Biddy_GetElse` returns ELSE successor.

**Description**

Input mark is not transfered! External use, only.

**Side effects****More info**

Macro `BiddyE(fun)` is defined for internal use.

Definition at line 1249 of file `biddyMain.c`.

**5.4.2.9 `Biddy_Variable` `Biddy_GetTopVariable` ( `Biddy_Edge fun` )**

Function `Biddy_GetTopVariable` returns the top variable.

**Description**

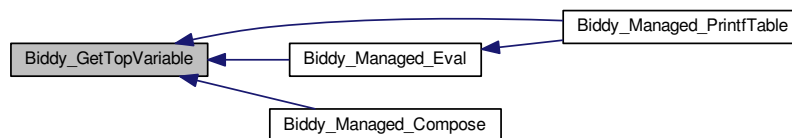
External use, only.

**Side effects****More info**

Macro `BiddyV(fun)` is defined for internal use.

Definition at line 1285 of file `biddyMain.c`.

Here is the caller graph for this function:

**5.4.2.10 `Biddy_Boolean` `Biddy_Managed_IsEqv` ( `Biddy_Manager MNG1`, `Biddy_Edge f1`, `Biddy_Manager MNG2`, `Biddy_Edge f2` )**

Function `Biddy_Managed_IsEqv` returns TRUE iff two BDDs are equal.

**Description**

It is assumed that `f1` and `f2` have the same ordering.

**Side effects****More info**

Macro `Biddy_IsEqv(f1,MNG2,f2)` is defined for use with anonymous manager.

Definition at line 1309 of file `biddyMain.c`.



#### 5.4.2.11 void Biddy\_Managed\_SelectNode ( Biddy\_Manager *MNG*, Biddy\_Edge *f* )

Function Biddy\_Managed\_SelectNode selects the top node of the given function.

**Description**

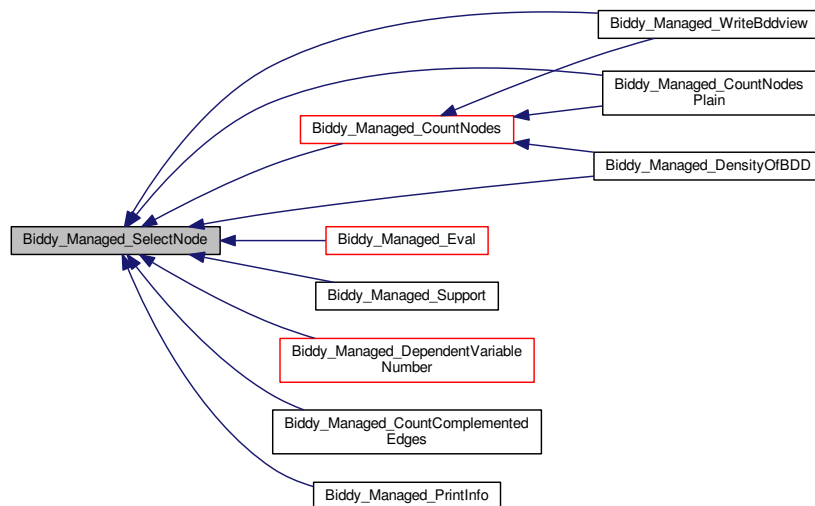
**Side effects**

**More info**

Macro [Biddy\\_SelectNode\(f\)](#) is defined for use with anonymous manager.

Definition at line 1344 of file biddyMain.c.

Here is the caller graph for this function:



#### 5.4.2.12 void Biddy\_Managed\_DeselectNode ( Biddy\_Manager *MNG*, Biddy\_Edge *f* )

Function Biddy\_Managed\_DeselectNode deselects the top node of the given function.

**Description**

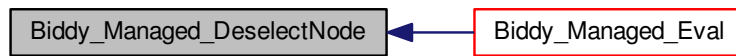
**Side effects**

**More info**

Macro [Biddy\\_DeselectNode\(f\)](#) is defined for use with anonymous manager.

Definition at line 1370 of file biddyMain.c.

Here is the caller graph for this function:



#### 5.4.2.13 Bidly\_Boolean Bidly\_Managed\_IsSelected ( Bidly\_Manager MNG, Bidly\_Edge f )

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

**Description**

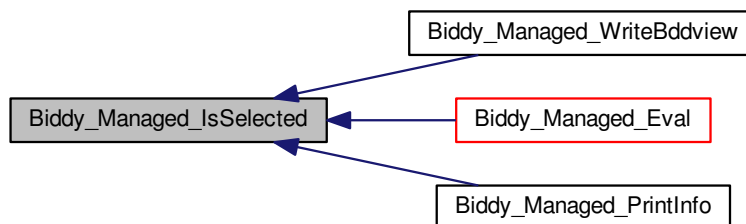
**Side effects**

**More info**

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

Definition at line 1396 of file `bidlyMain.c`.

Here is the caller graph for this function:



#### 5.4.2.14 void Bidly\_Managed\_SelectFunction ( Bidly\_Manager MNG, Bidly\_Edge f )

Function `Bidly_Managed_SelectFunction` recursively selects all nodes of a given function.

**Description**

**Side effects**

Terminal node must be selected before starting this function!

**More info**

Macro [Biddy\\_SelectFunction\(f\)](#) is defined for use with anonymous manager.

Definition at line 1423 of file biddyMain.c.

**5.4.2.15 void Biddy\_Managed\_DeselectAll ( Biddy\_Manager *MNG* )**

Function Biddy\_Managed\_DeselectAll deselects all nodes.

**Description**

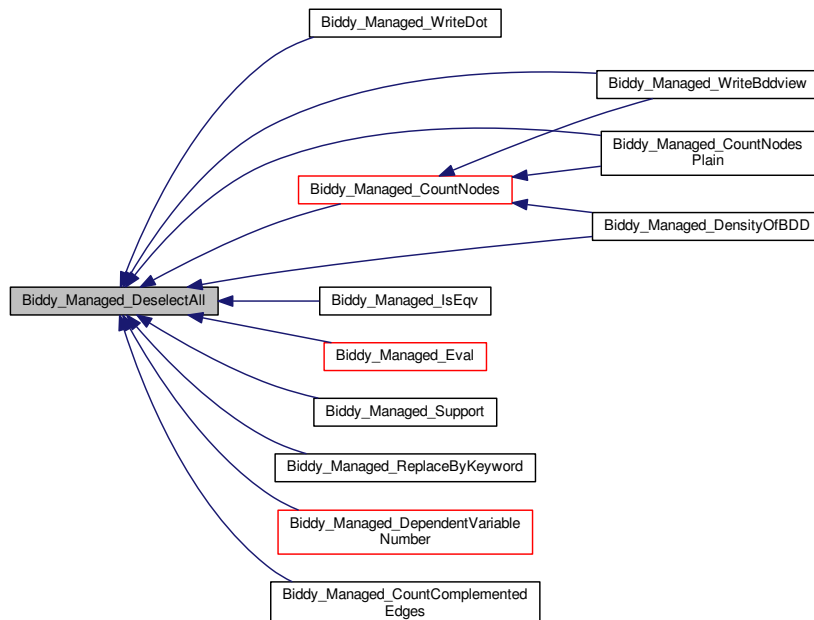
Deselect all nodes.

**Side effects****More info**

Macro [Biddy\\_DeselectAll\(\)](#) is defined for use with anonymous manager.

Definition at line 1480 of file biddyMain.c.

Here is the caller graph for this function:

**5.4.2.16 Biddy\_Edge Biddy\_Managed\_GetTerminal ( Biddy\_Manager *MNG* )**

Function Biddy\_Managed\_GetTerminal returns unmarked and untagged edge pointing to terminal node 1.

### Description

Terminal node depends on a manager.

### Side effects

### More info

Internally, use macro `biddyTerminal`. Macro `Biddy_GetTerminal()` is defined for use with anonymous manager.

Definition at line 1524 of file `biddyMain.c`.

#### 5.4.2.17 `Biddy_Edge Biddy_Managed_GetConstantZero ( Biddy_Manager MNG )`

Function `Biddy_Managed_GetConstantZero` returns constant 0.

### Description

Constants 0 and 1 depend on a manager. For combination sets, constant 0 coincides with empty set.

### Side effects

### More info

Internally, use macro `biddyZero`. Macro `Biddy_GetConstantZero()` is defined for use with anonymous manager. Macros `Biddy_Managed_GetEmptySet(MNG)` and `Biddy_GetEmptySet()` are defined for manipulation of combination sets.

Definition at line 1554 of file `biddyMain.c`.

#### 5.4.2.18 `Biddy_Edge Biddy_Managed_GetConstantOne ( Biddy_Manager MNG )`

Function `Biddy_Managed_GetConstantOne` returns constant 1.

### Description

Constants 0 and 1 depend on a manager. For combination sets, constant 1 coincides with universal set.

### Side effects

For ZBDDs and ZFDDs, you should always obtain constant 1 through the call of this function!

### More info

Internally, use macro `biddyOne` (also for ZBDDs and ZFDDs!). Macro `Biddy_GetConstantOne()` is defined for use with anonymous manager. Macros `Biddy_Managed_GetUniversalSet(MNG)` and `Biddy_GetUniversalSet()` are defined for manipulation of combination sets.

Definition at line 1586 of file `biddyMain.c`.

#### 5.4.2.19 Bidly\_Edge Bidly\_Managed\_GetBaseSet ( Bidly\_Manager MNG )

Function Bidly\_Managed\_GetBaseSet returns set containing only a null combination, i.e. it returns {{}}.

**Description**

**Side effects**

**More info**

Macro [Bidly\\_GetBaseSet\(\)](#) is defined for use with anonymous manager.

Definition at line 1612 of file biddyMain.c.

Here is the caller graph for this function:



#### 5.4.2.20 Bidly\_Variable Bidly\_Managed\_GetVariable ( Bidly\_Manager MNG, Bidly\_String x )

Function Bidly\_Managed\_GetVariable returns variable with the given name.

**Description**

**Side effects**

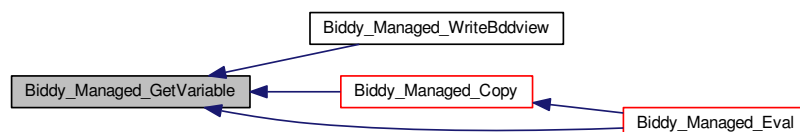
If variable is not found function returns 0!

**More info**

Macro [Bidly\\_GetVariable\(x\)](#) is defined for use with anonymous manager.

Definition at line 1677 of file biddyMain.c.

Here is the caller graph for this function:



#### 5.4.2.21 Bidly\_Variable Bidly\_Managed\_GetLowestVariable ( Bidly\_Manager *MNG* )

Function Bidly\_Managed\_GetLowestVariable returns the lowest variable in the current ordering.

##### Description

The lowest variable is the tompost variable.

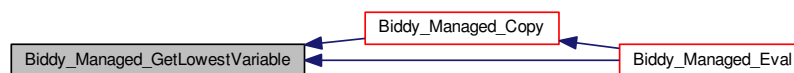
##### Side effects

##### More info

Macro [Bidly\\_GetLowestVariable\(x\)](#) is defined for use with anonymous manager.

Definition at line 1729 of file bidlyMain.c.

Here is the caller graph for this function:



#### 5.4.2.22 Bidly\_Variable Bidly\_Managed\_GetIthVariable ( Bidly\_Manager *MNG*, Bidly\_Variable *i* )

Function Bidly\_Managed\_GetIthVariable returns ith variable in the current global ordering.

##### Description

The lowest (topmost) variable has global ordering 1. The highest (bottommost) variable is '1' and has global ordering equal to numUsedVariables.

##### Side effects

If argument is 0, function returns 0. If argument is larger than the number of variables, function returns 0.

##### More info

Macro [Bidly\\_GetIthVariable\(x\)](#) is defined for use with anonymous manager.

Definition at line 1767 of file bidlyMain.c.

#### 5.4.2.23 Bidly\_Variable Bidly\_Managed\_GetPrevVariable ( Bidly\_Manager *MNG*, Bidly\_Variable *v* )

Function Bidly\_Managed\_GetPrevVariable returns previous variable in the global ordering (lower, topmore).

Description

Side effects

More info

Macro [Biddy\\_GetPrevVariable\(v\)](#) is defined for use with anonymous manager.

Definition at line 1805 of file biddyMain.c.

Here is the caller graph for this function:



#### 5.4.2.24 `Biddy_Variable` `Biddy_Managed_GetNextVariable` ( `Biddy_Manager MNG`, `Biddy_Variable v` )

Function `Biddy_Managed_GetNextVariable` returns next variable in the global ordering (higher, bottommore).

Description

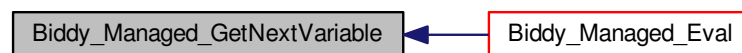
Side effects

More info

Macro [Biddy\\_GetNextVariable\(v\)](#) is defined for use with anonymous manager.

Definition at line 1835 of file biddyMain.c.

Here is the caller graph for this function:



#### 5.4.2.25 `Biddy_Edge` `Biddy_Managed_GetVariableEdge` ( `Biddy_Manager MNG`, `Biddy_Variable v` )

Function `Biddy_Managed_GetVariableEdge` returns variable's edge.

Description

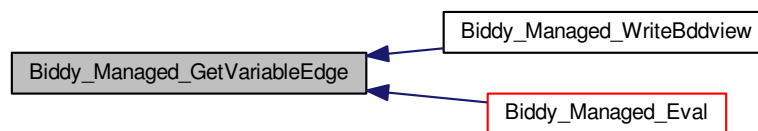
Side effects

More info

Macro [Bidly\\_GetVariableEdge\(v\)](#) is defined for use with anonymous manager.

Definition at line 1864 of file biddyMain.c.

Here is the caller graph for this function:



#### 5.4.2.26 `Bidly_Edge Bidly_Managed_GetElementEdge ( Bidly_Manager MNG, Bidly_Variable v )`

Function `Bidly_Managed_GetElementEdge` returns element's edge.

Description

Side effects

More info

Macro [Bidly\\_GetElementEdge\(v\)](#) is defined for use with anonymous manager.

Definition at line 1889 of file biddyMain.c.

#### 5.4.2.27 `Bidly_String Bidly_Managed_GetVariableName ( Bidly_Manager MNG, Bidly_Variable v )`

Function `Bidly_Managed_GetVariableName` returns the name of a variable.



Description

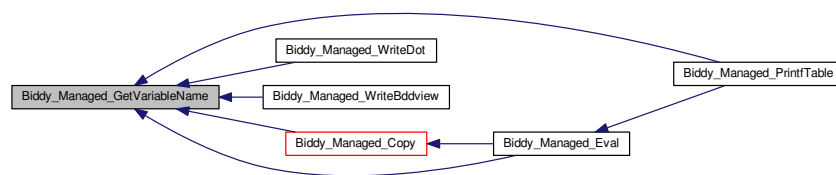
Side effects

More info

Macro [Biddy\\_GetVariableName\(v\)](#) is defined for use with anonymous manager.

Definition at line 1914 of file biddyMain.c.

Here is the caller graph for this function:



#### 5.4.2.28 `Biddy_Edge` `Biddy_Managed_GetTopVariableEdge ( Biddy_Manager MNG, Biddy_Edge f )`

Function `Biddy_Managed_GetTopVariableEdge` returns variable's edge of top variable.

Description

Side effects

TO DO: For ZBDDs, element edge is sometimes preferred over variable edge.

More info

Macro [Biddy\\_GetTopVariableEdge\(f\)](#) is defined for use with anonymous manager.

Definition at line 1942 of file biddyMain.c.

#### 5.4.2.29 `Biddy_String` `Biddy_Managed_GetTopVariableName ( Biddy_Manager MNG, Biddy_Edge f )`

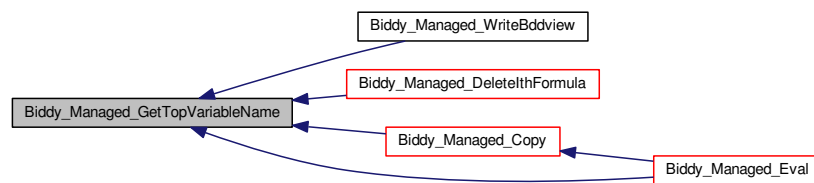
Function `Biddy_Managed_GetTopVariableName` returns the name of top variable.

**Description****Side effects****More info**

Macro [Bidly\\_GetTopVariableName\(f\)](#) is defined for use with anonymous manager.

Definition at line 1970 of file bidlyMain.c.

Here is the caller graph for this function:



#### 5.4.2.30 char Bidly\_Managed\_GetTopVariableChar ( Bidly\_Manager MNG, Bidly\_Edge f )

Function `Bidly_Managed_GetTopVariableChar` returns the first character in the name of top variable.

**Description****Side effects****More info**

Macro [Bidly\\_GetTopVariableChar\(f\)](#) is defined for use with anonymous manager.

Definition at line 1998 of file bidlyMain.c.

#### 5.4.2.31 void Bidly\_Managed\_ResetVariablesValue ( Bidly\_Manager MNG )

Function `Bidly_Managed_ResetVariablesValue` sets all variable's value to `bidlyZero`.

**Description****Side effects**

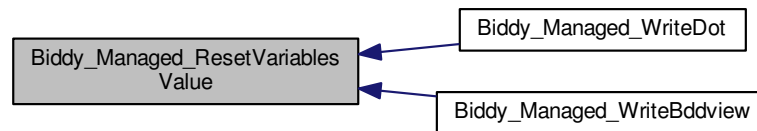
Only active (used) variables are reinitialized.

**More info**

Macro [Biddy\\_ResetVariablesValue\(\)](#) is defined for use with anonymous manager.

Definition at line 2027 of file biddyMain.c.

Here is the caller graph for this function:



#### 5.4.2.32 void Biddy\_Managed\_SetVariableValue ( Biddy\_Manager *MNG*, Biddy\_Variable *v*, Biddy\_Edge *f* )

Function Biddy\_Managed\_SetVariableValue sets variable's value.

**Description****Side effects**

It is not checked that the given variable is valid.

**More info**

Macro [Biddy\\_SetVariableValue\(v,f\)](#) is defined for use with anonymous manager.

Definition at line 2057 of file biddyMain.c.

#### 5.4.2.33 Biddy\_Edge Biddy\_Managed\_GetVariableValue ( Biddy\_Manager *MNG*, Biddy\_Variable *v* )

Function Biddy\_Managed\_GetVariableValue gets variable's value.

**Description****Side effects**

It is not checked that the given variable is valid.

**More info**

Macro [Biddy\\_GetVariableValue\(v\)](#) is defined for use with anonymous manager.

Definition at line 2083 of file biddyMain.c.

#### 5.4.2.34 void Biddy\_Managed\_ClearVariablesData ( Biddy\_Manager MNG )

Function Biddy\_Managed\_ClearVariablesData free memory used for all variable's data.

##### Description

##### Side effects

Only active (used) variables are considered.

##### More info

Macro [Biddy\\_ClearVariablesData\(\)](#) is defined for use with anonymous manager.

Definition at line 2110 of file biddyMain.c.

Here is the caller graph for this function:



#### 5.4.2.35 void Biddy\_Managed\_SetVariableData ( Biddy\_Manager MNG, Biddy\_Variable v, void \* x )

Function Biddy\_Managed\_SetVariableData sets variable's data.

##### Description

##### Side effects

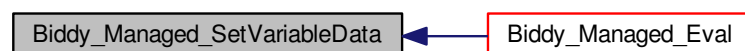
It is not checked that the given variable is valid.

##### More info

Macro [Biddy\\_SetVariableData\(v,x\)](#) is defined for use with anonymous manager.

Definition at line 2143 of file biddyMain.c.

Here is the caller graph for this function:



**5.4.2.36** void\* Biddy\_Managed\_GetVariableData ( Biddy\_Manager *MNG*, Biddy\_Variable *v* )

Function Biddy\_Managed\_GetVariableData gets variable's data.

**Description****Side effects**

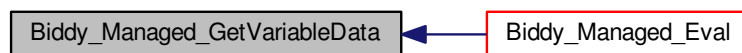
It is not checked that the given variable is valid.

**More info**

Macro [Biddy\\_GetVariableData\(v\)](#) is defined for use with anonymous manager.

Definition at line 2169 of file biddyMain.c.

Here is the caller graph for this function:

**5.4.2.37** 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 = previous = above = topmore).

**Description****Side effects****More info**

Macro BiddyIsSmaller(fv,gv) is defined for internal use. Macro [Biddy\\_IsSmaller\(fv,gv\)](#) is defined for use with anonymous manager.

Definition at line 2196 of file biddyMain.c.

**5.4.2.38** Biddy\_Boolean Biddy\_Managed\_IsLowest ( Biddy\_Manager *MNG*, Biddy\_Variable *v* )

Function Biddy\_Managed\_IsLowest returns TRUE if the variable is the lowest one (lowest == topmost).

**Description****Side effects****More info**

Macro `BiddyIsLowest(v)` is defined for internal use. Macro `Biddy_IsLowest(v)` is defined for use with anonymous manager.

Definition at line 2223 of file `biddyMain.c`.

Here is the caller graph for this function:

**5.4.2.39 Biddy\_Boolean Biddy\_Managed\_IsHighest ( Biddy\_Manager MNG, Biddy\_Variable v )**

Function `Biddy_Managed_IsHighest` returns TRUE if the variable is the highest one if terminal node is ignored (highest == bottommost).

**Description****Side effects****More info**

Macro `BiddyIsHighest(v)` is defined for internal use. Macro `Biddy_IsHighest(v)` is defined for use with anonymous manager.

Definition at line 2260 of file `biddyMain.c`.

Here is the caller graph for this function:



#### 5.4.2.40 Bidly\_Managed\_FoaVariable ( Bidly\_Manager MNG, Bidly\_String x, Bidly\_Boolean varelem )

Function Bidly\_Managed\_FoaVariable finds variable/element or adds new variable (i.e. Boolean function  $f = x$ ) and new element (i.e. it creates set  $\{\{x\}\}$ ).

##### Description

If variable/element already exists, function returns the existing one. If  $x == \text{NULL}$  then numbered variable/element is added. Numbered variables/elements have only digits in its name. The current number of numbered variables/elements is stored in numnum. If numbered variable/element is requested then function increments numnum and creates a new (non-existing) variable/element. Parameter varelem is used to determine how to adapt the existing BDD base to keep the current formula valid (use varelem = TRUE if formulae represent Boolean functions and varelem = FALSE if they represent combination sets). The ordering of the new variable/element is determined in Bidly\_InitMNG. Function always returns variable.

##### Side effects

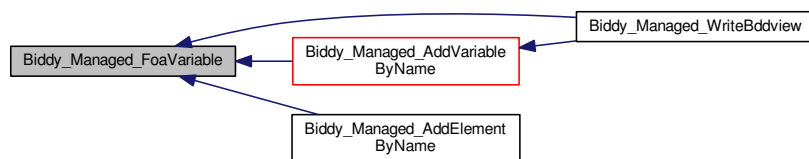
Adding new variable/element may change the meaning of the existing BDDs. Variables and elements are repaired. Moreover, formulae are repaired with regards to the parameter varelem. BDDs without external references are not repaired. For OBDDs, OFDDs, TZBDDs, and TZFDDs, it is safe to add new variables/elements if BDDs are used to represent Boolean functions. For ZBDDs and ZFDDs, it is safe to add new variables/elements if BDDs are used to represent combination sets. User should not add numbered variables/elements with some other function. TO DO: Formulae in user's formula tables are not repaired, yet!

##### More info

Macro [Bidly\\_FoaVariable\(x\)](#) is defined for use with anonymous manager.

Definition at line 2318 of file biddyMain.c.

Here is the caller graph for this function:



#### 5.4.2.41 void Bidly\_Managed\_ChangeVariableName ( Bidly\_Manager MNG, Bidly\_Variable v, Bidly\_String x )

Function Bidly\_Managed\_ChangeVariableName set new name to the given variable/element.

##### Description

##### Side effects

It is not checked that the same variable/element does not already exist. The ordering of the variable is not changed.

**More info**

Macro [Bidly\\_ChangeVariableName\(v,x\)](#) is defined for use with anonymous manager.

Definition at line 2586 of file bidlyMain.c.

**5.4.2.42 Bidly\_Edge Bidly\_Managed\_AddVariableByName ( Bidly\_Manager MNG, Bidly\_String x )**

Function Bidly\_Managed\_AddVariableByName adds variable.

**Description**

Bidly\_Managed\_AddVariableByName uses Bidly\_Managed\_FoaVariable to find or add variable. Function returns variable edge. If variable already exists, function returns the existing variable edge. For more details see Bidly\_Managed\_FoaVariable.

**Side effects**

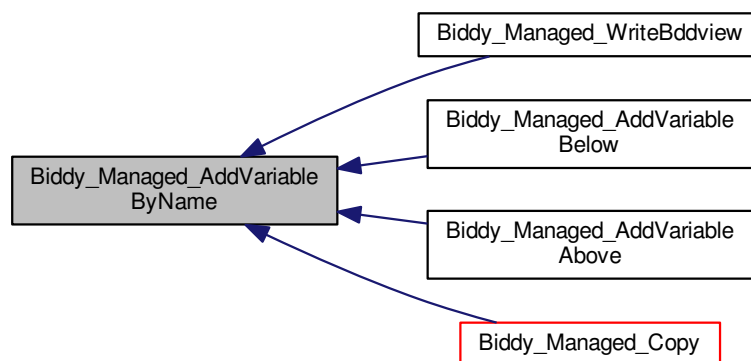
See Bidly\_Managed\_FoaVariable.

**More info**

Macro [Bidly\\_AddVariableByName\(x\)](#) is defined for use with anonymous manager. Macros Bidly\_Managed\_AddVariable(MNG) and Bidly\_AddVariable() are defined for creating numbered variables.

Definition at line 2622 of file bidlyMain.c.

Here is the caller graph for this function:

**5.4.2.43 Bidly\_Edge Bidly\_Managed\_AddElementByName ( Bidly\_Manager MNG, Bidly\_String x )**

Function Bidly\_Managed\_AddElementByName adds element.



### Description

Biddy\_Managed\_AddElementByName uses Biddy\_Managed\_FoaVariable to find or add element. Function returns element edge. If element already exists, function returns the existing element edge. For more details see Biddy\_Managed\_FoaVariable.

### Side effects

See Biddy\_Managed\_FoaVariable.

### More info

Macro [Biddy\\_AddElementByName\(x\)](#) is defined for use with anonymous manager. Macros Biddy\_Managed\_AddElement(MNG) and Biddy\_AddElement() are defined for creating numbered elements.

Definition at line 2657 of file biddyMain.c.

#### 5.4.2.44 Biddy\_Edge Biddy\_Managed\_AddVariableBelow ( Biddy\_Manager MNG, Biddy\_Variable v )

Function Biddy\_Managed\_AddVariableBelow adds variable.

### Description

Biddy\_Managed\_AddVariableBelow uses Biddy\_Managed\_AddVariableByName to add new variable. Then, the order of the new variable is changed to become immediately below the given variable (below = next = bottommore in BDD) Function returns variable edge.

### Side effects

### More info

Macro [Biddy\\_AddVariableBelow\(v\)](#) is defined for use with anonymous manager.

Definition at line 2689 of file biddyMain.c.

#### 5.4.2.45 Biddy\_Edge Biddy\_Managed\_AddVariableAbove ( Biddy\_Manager MNG, Biddy\_Variable v )

Function Biddy\_Managed\_AddVariableAbove adds variable.

### Description

Biddy\_Managed\_AddVariableAbove uses Biddy\_Managed\_AddVariableByName to add new variable. Then, the order of the new variable is changed to become immediately above the given variable (above = previous = topmore in BDD) Function returns variable edge.

### Side effects

### More info

Macro [Biddy\\_AddVariableAbove\(v\)](#) is defined for use with anonymous manager.

Definition at line 2772 of file biddyMain.c.

#### 5.4.2.46 Biddy\_Edge Biddy\_Managed\_TransferMark ( Biddy\_Manager *MNG*, Biddy\_Edge *f*, Biddy\_Boolean *mark*, Biddy\_Boolean *leftright* )

Function Biddy\_Managed\_TransferMark returns edge with inverted complement bit iff the second parameter is T↔RUE and normalization rules require this.

##### Description

Parameter leftright should be TRUE for left and FALSE for right. For OBDDC, it is better to use macro Biddy\_Inv↔Cond. For OBDDC, parameter leftright is ignored.

##### Side effects

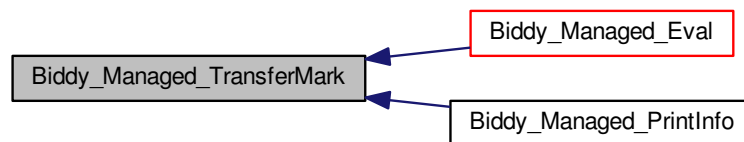
TO DO: swap the meaning of parameter leftright (left should be FALSE)

##### More info

Macro Biddy\_TransferMark() is defined for use with anonymous manager.

Definition at line 2853 of file biddyMain.c.

Here is the caller graph for this function:



#### 5.4.2.47 Biddy\_Edge Biddy\_Managed\_IncTag ( Biddy\_Manager *MNG*, Biddy\_Edge *f* )

Function Biddy\_Managed\_IncTag returns edge with an incremented tag.

##### Description

Used for TZBDDs and TZFDDs, only.

##### Side effects

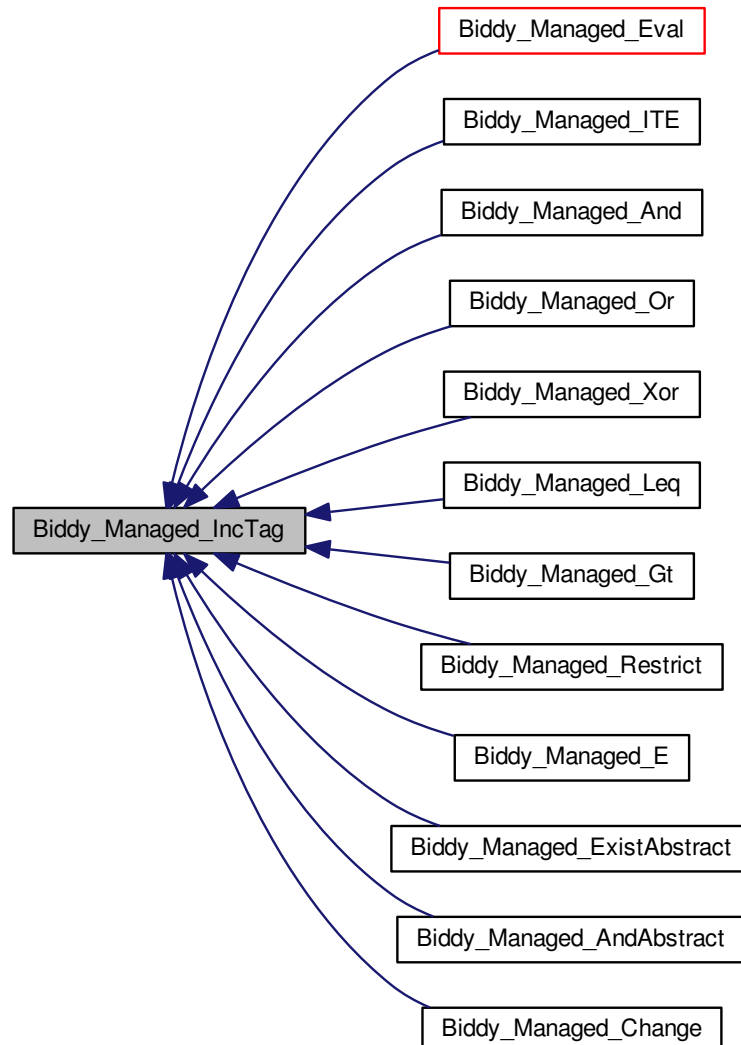
It is not checked, that the resulting tag is not greater than top variable. Function may return non-fresh node even if *f* is fresh.

More info

Macro [Biddy\\_IncTag\(\)](#) is defined for use with anonymous manager.

Definition at line 2919 of file biddyMain.c.

Here is the caller graph for this function:



**5.4.2.48** `Biddy_Edge Biddy_Managed_TaggedFoaNode ( Biddy_Manager MNG, Biddy_Variable v, Biddy_Edge pf, Biddy_Edge pt, Biddy_Variable ptag, Biddy_Boolean garbageAllowed )`

Function `Biddy_Managed_TaggedFoaNode` 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. If  $pf = pt = \text{NULL}$  (and  $ptag = v$ ) then new variable (for OBDD, OFDD, TZBDD, and TZFDD) or new element (for ZBDD and ZFDD) is created. For OBDD, ZBDD, OFDD, and ZFDD, parameter  $ptag$  is ignored. For TZBDD and TZFDD, the returned edge is tagged with the given  $ptag$ . This function should not be called directly to add new variables and elements, you must use `Biddy_Managed_FoaVariable`, `Biddy_Managed_AddVariableByName`, or `Biddy_Managed_AddElementByName`.

### Side effects

Using `Biddy_Managed_TaggedFoaNode` you can create node with an arbitrary ordering. It is much more safe to use Boolean operators, e.g. `Biddy_Managed_ITE`.

### More info

Macro `Biddy_Managed_FoaNode(MNG,v,pf,pt,garbageAllowed)` is defined for use with implicit tags or without tags. Macros `Biddy_TaggedFoaNode(v,pf,pt,tag,garbageAllowed)` and `Biddy_FoaNode(v,pf,pt,garbageAllowed)` are defined for use with anonymous manager.

Definition at line 2997 of file `biddyMain.c`.

#### 5.4.2.49 `Biddy_Boolean Biddy_Managed_IsOK ( Biddy_Manager MNG, Biddy_Edge f )`

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

### Description

This is needed for implementation of user caches.

### Side effects

### More info

Macro `BiddyIsOK(f)` is defined for debugging. It will check more properties and not only the expiry value. Macro `Biddy_IsOK(f)` is defined for use with anonymous manager.

Definition at line 3419 of file `biddyMain.c`.

#### 5.4.2.50 `void Biddy_Managed_GC ( Biddy_Manager MNG, Biddy_Variable targetLT, Biddy_Variable targetGEQ, Biddy_Boolean purge, Biddy_Boolean total )`

Function `Biddy_Managed_GC` performs garbage collection.

### Description

All obsolete nodes are deleted. Iff parameter `purge` is true then all formulae without name are deleted. Iff parameter `purge` is true then all nodes which are not part of any non-obsolete non-deleted formulae are removed even if they are fresh or fortified (this should not be used during automatic garbage collection!). If parameter `total` is true then unnecessary nodes are guaranteed to be deleted, otherwise they are deleted only when there are enough of them. If  $(\text{targetLT} \neq 0)$  then node table resizing is disabled. If  $(\text{targetLT} \neq 0)$  then there should not exist obsolete formulae. If  $(\text{targetLT} \neq 0)$  then there should not exist obsolete nodes which are not part of any non-obsolete non-deleted formulae. If  $(\text{targetLT} \neq 0)$  then there should not exist obsolete nodes with variable equal or higher (bottom-more) than `target` and smaller (top-more) than `targetGEQ`.

**Side effects**

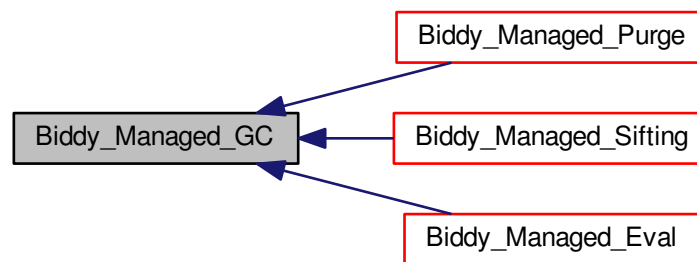
The first element of each chain in a node table should have a special value for its 'prev' element to allow tricky but efficient deleting. Moreover, 'prev' and 'next' should be the first and the second element in the structure Bidy↔Node, respectively. Garbage collection is reported by bidyNodeTable.garbage only if some bad nodes are purged! Parameters targetLT and targetGEQ are used during sifting, only, in all other cases 0 is used.

**More info**

Macro `Biddy_GC(target,purge,total)` is defined for use with anonymous manager. Macros `Biddy_Managed_Auto↔GC(MNG)` and `Biddy_AutoGC()` are useful variants with `target = 0`, `purge = FALSE`, and `total = FALSE`.

Definition at line 3468 of file biddyMain.c.

Here is the caller graph for this function:

**5.4.2.51 void Bidy\_Managed\_Clean ( Bidy\_Manager MNG )**

Function `Bidy_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**

Field deleted is not considered and thus no fortified node and no prolonged node is discarded.

**More info**

Macro [Bidly\\_Clean\(\)](#) is defined for use with anonymous manager.

Definition at line 3958 of file biddyMain.c.

Here is the caller graph for this function:

**5.4.2.52 void Bidly\_Managed\_Purge ( Bidly\_Manager MNG )**

Function `Bidly_Managed_Purge` immediately removes all nodes which were not preserved or which are not preserved anymore.

**Description**

All fresh and obsolete nodes are immediately removed. All formulae without name are deleted. Moreover, nodes from deleted prolonged formulae and nodes from deleted fortified formulae are removed if they are not needed by other formulae. Call to `Bidly_Purge` does not count as clearing and thus all preserved formulae remains preserved for the same number of clearings.

**Side effects**

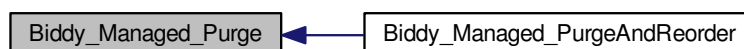
Removes all fresh nodes!

**More info**

Macro [Bidly\\_Purge\(f\)](#) is defined for use with anonymous manager.

Definition at line 4008 of file biddyMain.c.

Here is the caller graph for this function:



5.4.2.53 void `Biddy_Managed_PurgeAndReorder` ( `Biddy_Manager MNG`, `Biddy_Edge f`, `Biddy_Boolean converge` )

Function `Biddy_Managed_PurgeAndReorder` immediately removes non-preserved nodes and triggers reordering on function.

#### Description

All obsolete nodes are immediately removed. Moreover, nodes from deleted prolonged formulae and nodes from deleted fortified formulae are removed if they are not needed by other formulae. If BDD is given (`f != NULL`), reordering on function is used. Otherwise (`f == NULL`) global reordering is used. Call to `Biddy_PurgeAndReorder` does not count as clearing and thus all preserved formulae remains preserved for the same number of clearings.

#### Side effects

Removes all fresh nodes.

#### More info

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

Definition at line 4056 of file `biddyMain.c`.

5.4.2.54 void `Biddy_Managed_Refresh` ( `Biddy_Manager MNG`, `Biddy_Edge f` )

Function `Biddy_Managed_Refresh` refreshes top node in a given function.

#### Description

This is an external variant of internal macro `BiddyRefresh`. This is needed for implementing user caches.

#### Side effects

#### More info

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

Definition at line 4085 of file `biddyMain.c`.

5.4.2.55 void `Biddy_Managed_AddCache` ( `Biddy_Manager MNG`, `Biddy_GCFunction gc` )

Function `Biddy_Managed_AddCache` adds cache to the end of Cache list.

#### Description

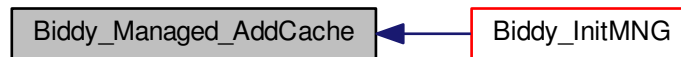
If Cache list does not exist, function creates it.

**Side effects****More info**

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

Definition at line 4111 of file `biddyMain.c`.

Here is the caller graph for this function:



#### 5.4.2.56 unsigned int Bidly\_Managed\_AddFormula ( Bidly\_Manager MNG, Bidly\_String x, Bidly\_Edge f, int c )

Function `Bidly_Managed_AddFormula` adds formula to Formula table.

**Description**

Given BDD becomes a formula. If ( $x \neq \text{NULL}$ ) then formula is accessible by its name. If ( $x \neq \text{NULL}$ ) then index of the formula in the Formulae Table is returned, otherwise function returns 0. Nodes of the given BDD will be preserved for the given number of clearings. If ( $c == -1$ ) then formula is refreshed but not preserved. If ( $c == 0$ ) then formula is persistently preserved. You have to use `Bidly_DeleteFormula` and `Bidly_Purge` to remove nodes of persistently preserved formulae. There are five macros defined to simplify formulae management: `Bidly_Managed_AddTmpFormula(mng,name,bdd) := Bidly_Managed_AddFormula(mng,name,bdd,-1)` `Bidly_Managed_AddPreservedFormula(mng,bdd,c) := Bidly_Managed_AddFormula(mng,NULL,bdd,c)` `Bidly_Managed_AddPersistentFormula(mng,name,bdd) := Bidly_Managed_AddFormula(mng,name,bdd,0)` `Bidly_Managed_KeepFormula(mng,bdd) := Bidly_Managed_AddFormula(mng,NULL,bdd,1)` `Bidly_Managed_KeepFormulaUntilDelete(mng,bdd) := Bidly_Managed_AddFormula(mng,NULL,bdd,0)`

**Side effects**

Function is prolonged or fortified. Formulae with name are ordered by name. If formula with the same name already exists, it will be overwritten - preserved (i.e. not obsolete and not fresh) and persistently preserved formulae will be deleted at the original index and recreated at new index!

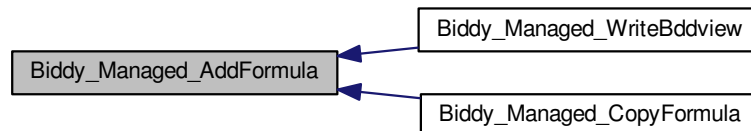


**More info**

Macros [Biddy\\_AddFormula\(x,f\)](#), [Biddy\\_AddTmpFormula\(f\)](#), [Biddy\\_AddPreservedFormula\(f,c\)](#), [Biddy\\_AddPersistentFormula\(x,f\)](#), [Biddy\\_KeepFormula\(f\)](#), and [Biddy\\_KeepFormulaUntilDelete\(f\)](#) are defined for use with anonymous manager.

Definition at line 4185 of file biddyMain.c.

Here is the caller graph for this function:



#### 5.4.2.57 Biddy\_Boolean Biddy\_Managed\_FindFormula ( Biddy\_Manager MNG, Biddy\_String x, unsigned int \* idx, Biddy\_Edge \* f )

Function `Biddy_Managed_FindFormula` find formula in Formula table.

**Description**

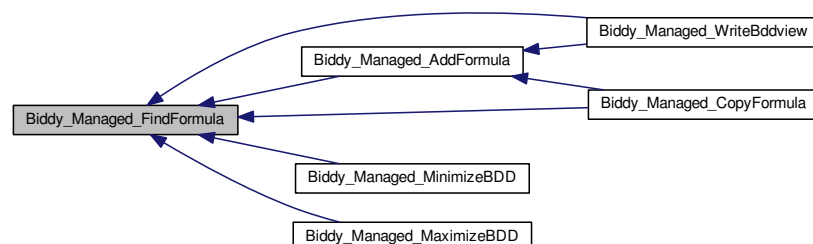
Return TRUE/FALSE, index, and the formula. If formula is constant or variable then `idx = 0` and `f != biddyNull`. If formula is not found then `idx` is a position where the formulae should exist and `f == biddyNull`.

**Side effects****More info**

Macro [Biddy\\_FindFormula\(x,f\)](#) is defined for use with anonymous manager.

Definition at line 4441 of file biddyMain.c.

Here is the caller graph for this function:



#### 5.4.2.58 **Biddy\_Boolean** Biddy\_Managed\_DeleteFormula ( Biddy\_Manager *MNG*, Biddy\_String *x* )

Function Biddy\_Managed\_DeleteFormula delete formula from Formula table.

##### Description

Formula is labelled but not immediately removed. Nodes of the given formula are not immediately removed.

##### Side effects

Formula is not accessible by its name anymore. Formulae representing constants and variables will not be deleted.

##### More info

Macro [Biddy\\_DeleteFormula\(x\)](#) is defined for use with anonymous manager.

Definition at line 4615 of file biddyMain.c.

#### 5.4.2.59 **Biddy\_Boolean** Biddy\_Managed\_DeletelthFormula ( Biddy\_Manager *MNG*, unsigned int *i* )

Function Biddy\_Managed\_DeletelthFormula deletes formula from the table.

##### Description

Formula is labelled but not immediately removed. Nodes of the given formula are not immediately removed.

##### Side effects

Formula is not accessible by its name anymore. The first two formulae ("0" and "1") will not be deleted. Formulae representing variables will not be deleted.

##### More info

Macro [Biddy\\_DeletelthFormula\(x\)](#) is defined for use with anonymous manager.

Definition at line 4679 of file biddyMain.c.

Here is the caller graph for this function:



#### 5.4.2.60 **Biddy\_Edge** Biddy\_Managed\_GetlthFormula ( Biddy\_Manager *MNG*, unsigned int *i* )

Function Biddy\_Managed\_GetlthFormula 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\\_GetlthFormula\(i\)](#) is defined for use with anonymous manager.

Definition at line 4739 of file biddyMain.c.

**5.4.2.61 Biddy\_String Biddy\_Managed\_GetlthFormulaName ( Biddy\_Manager MNG, unsigned int i )**

Function Biddy\_Managed\_GetlthFormulaName 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!

**More info**

Macro [Biddy\\_GetlthFormulaName\(i\)](#) is defined for use with anonymous manager.

Definition at line 4770 of file biddyMain.c.

**5.4.2.62 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 variable is constant variable "1". Global ordering is number of zeros in corresponding line of orderingTable. Constant variable '1' has global ordering greater than all others.

**Side effects**

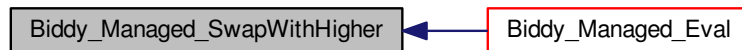
All obsolete nodes will be removed.

**More info**

Macro [Bidly\\_SwapWithHigher\(v\)](#) is defined for use with anonymous manager.

Definition at line 4811 of file biddyMain.c.

Here is the caller graph for this function:



#### 5.4.2.63 Bidly\_Variable Bidly\_Managed\_SwapWithLower ( Bidly\_Manager *MNG*, Bidly\_Variable *v* )

Function `Bidly_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 zeros in corresponding line of orderingTable.

**Side effects**

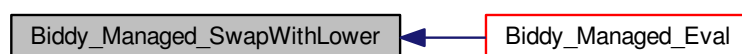
All obsolete nodes will be removed.

**More info**

Macro [Bidly\\_SwapWithLower\(v\)](#) is defined for use with anonymous manager.

Definition at line 4875 of file biddyMain.c.

Here is the caller graph for this function:



#### 5.4.2.64 Bidly\_Boolean Bidly\_Managed\_Sifting ( Bidly\_Manager *MNG*, Bidly\_Edge *f*, Bidly\_Boolean *converge* )

Function Bidly\_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 obsolete nodes will be removed.

##### Side effects

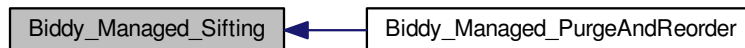
For TZBDD, all unreferenced nodes (not part of registered formulae) will be removed. For TZBDD, sifting may change top edge or even a top node of any function/formula - this is a problem, because functions referenced by local variables only may become wrong. Consequently, for TZBDDs, sifting is not safe to start automatically!

##### More info

Macro [Bidly\\_Sifting\(f\)](#) is defined for use with anonymous manager.

Definition at line 4943 of file biddyMain.c.

Here is the caller graph for this function:



#### 5.4.2.65 void Bidly\_Managed\_MinimizeBDD ( Bidly\_Manager *MNG*, Bidly\_String *name* )

Function Bidly\_Managed\_MinimizeBDD reorders variables to minimize the node number of the given formula using an exhaustive search over all possible orderings.

##### Description

Steinhaus–Johnson–Trotter algorithm is used to generate all possible permutations. An optimized version of Bubble Sort is used to setup the final ordering. Variables are reordered globally. All obsolete nodes will be removed.

##### Side effects

Indeed, this function may take a lot of time! For TZBDD, all unreferenced nodes (not part of registered formulae) will be removed. For TZBDD, this function may change top edge or even a top node of any function/formula - this is a problem, because functions referenced by local variables only may become wrong. Consequently, for TZBDDs, sifting is not safe to start automatically!

### More info

Macro [Biddy\\_MinimizeBDD\(f\)](#) is defined for use with anonymous manager.

Definition at line 5127 of file biddyMain.c.

#### 5.4.2.66 void Biddy\_Managed\_MaximizeBDD ( Biddy\_Manager *MNG*, Biddy\_String *name* )

Function `Biddy_Managed_MaximizeBDD` reorders variables to maximize the node number of the given function using an exhaustive search over all possible orderings.

### Description

Steinhaus–Johnson–Trotter algorithm is used to generate all possible permutations. An optimized version of Bubble Sort is used to setup the final ordering. Variables are reordered globally. All obsolete nodes will be removed.

### Side effects

Indeed, this function may take a lot of time! For TZBDD, all unreferenced nodes (not part of registered formulae) will be removed. For TZBDD, this function may change top edge or even a top node of any function/formula - this is a problem, because functions referenced by local variables only may become wrong. Consequently, for TZBDDs, sifting is not safe to start automatically!

### More info

Macro [Biddy\\_MaximizeBDD\(f\)](#) is defined for use with anonymous manager.

Definition at line 5222 of file biddyMain.c.

## 5.5 biddyMainLegacy.c File Reference

File [biddyMainLegacy.c](#) contains main functions for representation and manipulation of boolean functions with R↔OBDDs.

```
#include "biddyInt.h"
```

## Functions

- void [Biddy\\_InitMNG](#) ([Biddy\\_Manager](#) \*mng, int gddtype)  
*Function Biddy\_InitMNG initialize a manager.*
- void [Biddy\\_ExitMNG](#) ([Biddy\\_Manager](#) \*mng)  
*Function Biddy\_ExitMNG deletes a manager.*
- [Biddy\\_String](#) [Biddy\\_About](#) ()  
*Function Biddy\_About reports version of Biddy package.*
- int [Biddy\\_Managed\\_GetManagerType](#) ([Biddy\\_Manager](#) MNG)  
*Function Biddy\_Managed\_GetManagerType reports BDD type used in the manager.*
- void [Biddy\\_Managed\\_SetManagerParameters](#) ([Biddy\\_Manager](#) MNG, float gcr, float gcrF, float gcrX, float rr, float rrF, float rrX, float st, float cst)  
*Function Biddy\_Managed\_SetManagerParameters set modifiable parameters.*
- [Biddy\\_Edge](#) [Biddy\\_GetThen](#) ([Biddy\\_Edge](#) fun)  
*Function Biddy\_GetThen returns THEN successor.*
- [Biddy\\_Edge](#) [Biddy\\_GetElse](#) ([Biddy\\_Edge](#) fun)  
*Function Biddy\_GetElse returns ELSE successor.*
- [Biddy\\_Variable](#) [Biddy\\_GetTopVariable](#) ([Biddy\\_Edge](#) fun)  
*Function Biddy\_GetTopVariable returns the top variable.*
- [Biddy\\_Boolean](#) [Biddy\\_Managed\\_IsEqv](#) ([Biddy\\_Manager](#) MNG1, [Biddy\\_Edge](#) f1, [Biddy\\_Manager](#) MNG2, [Biddy\\_Edge](#) f2)  
*Function Biddy\_Managed\_IsEqv returns TRUE iff two BDDs are equal.*
- void [Biddy\\_Managed\\_SelectNode](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Edge](#) f)  
*Function Biddy\_Managed\_SelectNode selects the top node of the given function.*
- void [Biddy\\_Managed\\_DeselectNode](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Edge](#) f)  
*Function Biddy\_Managed\_DeselectNode deselects the top node of the given function.*
- [Biddy\\_Boolean](#) [Biddy\\_Managed\\_IsSelected](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Edge](#) f)  
*Function Biddy\_Managed\_IsSelected returns TRUE iff the top node of the given function is selected.*
- void [Biddy\\_Managed\\_SelectFunction](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Edge](#) f)  
*Function Biddy\_Managed\_SelectFunction recursively selects all nodes of a given function.*
- void [Biddy\\_Managed\\_DeselectAll](#) ([Biddy\\_Manager](#) MNG)  
*Function Biddy\_Managed\_DeselectAll deselects all nodes.*
- [Biddy\\_Edge](#) [Biddy\\_Managed\\_GetTerminal](#) ([Biddy\\_Manager](#) MNG)  
*Function Biddy\_Managed\_GetTerminal returns unmarked edge pointing to the constant node 1.*
- [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\\_Edge](#) [Biddy\\_Managed\\_GetBaseSet](#) ([Biddy\\_Manager](#) MNG)  
*Function Biddy\_Managed\_GetBaseSet returns set containing only a null combination, i.e. it returns {}.*
- [Biddy\\_Variable](#) [Biddy\\_Managed\\_GetVariable](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_String](#) x)  
*Function Biddy\_Managed\_GetVariable returns variable with the given name.*
- [Biddy\\_Variable](#) [Biddy\\_Managed\\_GetPrevVariable](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Variable](#) v)  
*Function Biddy\_Managed\_GetPrevVariable returns previous variable in the global ordering (lower, topmore).*
- [Biddy\\_Variable](#) [Biddy\\_Managed\\_GetNextVariable](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Variable](#) v)  
*Function Biddy\_Managed\_GetNextVariable returns next variable in the global ordering (higher, bottommore).*
- [Biddy\\_Edge](#) [Biddy\\_Managed\\_GetVariableEdge](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Variable](#) v)  
*Function Biddy\_Managed\_GetVariableEdge returns variable's edge.*
- [Biddy\\_Edge](#) [Biddy\\_Managed\\_GetElementEdge](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Variable](#) v)  
*Function Biddy\_Managed\_GetElementEdge returns element's edge.*
- [Biddy\\_String](#) [Biddy\\_Managed\\_GetVariableName](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Variable](#) v)

- Function Bidly\_Managed\_GetVariableName returns the name of a variable.*

  - [Bidly\\_Edge Bidly\\_Managed\\_GetTopVariableEdge](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f)

*Function Bidly\_Managed\_GetTopVariableEdge returns variable's edge of top variable.*
- [Bidly\\_String Bidly\\_Managed\\_GetTopVariableName](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f)

*Function Bidly\_Managed\_GetTopVariableName returns the name of top variable.*
- char [Bidly\\_Managed\\_GetTopVariableChar](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f)

*Function Bidly\_Managed\_GetTopVariableChar returns the first character in the name of top variable.*
- void [Bidly\\_Managed\\_ResetVariablesValue](#) ([Bidly\\_Manager](#) MNG)

*Function Bidly\_Managed\_ResetVariablesValue sets all variable's value to bidlyZero.*
- void [Bidly\\_Managed\\_SetVariableValue](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Variable](#) v, [Bidly\\_Edge](#) f)

*Function Bidly\_Managed\_SetVariableValue sets variable's value.*
- [Bidly\\_Boolean Bidly\\_Managed\\_IsSmaller](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Variable](#) fv, [Bidly\\_Variable](#) gv)

*Function Bidly\_Managed\_IsSmaller returns TRUE if the first variable is smaller (= lower = previous = above = top-more).*
- [Bidly\\_Variable Bidly\\_Managed\\_FoaVariable](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_String](#) x, [Bidly\\_Boolean](#) varelem)

*Function Bidly\_Managed\_FoaVariable finds variable/element or adds new variable (i.e. Boolean function  $f = x$ ) and new element (i.e. it creates set  $\{\{x\}\}$ ).*
- [Bidly\\_Edge Bidly\\_Managed\\_AddVariableByName](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_String](#) x)

*Function Bidly\_Managed\_AddVariableByName adds variable.*
- [Bidly\\_Edge Bidly\\_Managed\\_AddElementByName](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_String](#) x)

*Function Bidly\_Managed\_AddElementByName adds element.*
- [Bidly\\_Edge Bidly\\_Managed\\_AddVariableBelow](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Variable](#) v)

*Function Bidly\_Managed\_AddVariableBelow adds a numbered variable.*
- [Bidly\\_Edge Bidly\\_Managed\\_AddVariableAbove](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Variable](#) v)

*Function Bidly\_Managed\_AddVariableAbove adds a numbered variable.*
- [Bidly\\_Edge Bidly\\_Managed\\_TransferMark](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f, [Bidly\\_Boolean](#) mark, [Bidly\\_Boolean](#) leftright)

*Function Bidly\_Managed\_TransferMark returns edge with inverted complement bit iff the second parameter is TRUE and normalization rules require this.*
- [Bidly\\_Edge Bidly\\_Managed\\_IncTag](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f)

*Function Bidly\_Managed\_IncTag returns edge with an incremented tag.*
- [Bidly\\_Edge Bidly\\_Managed\\_TaggedFoaNode](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Variable](#) v, [Bidly\\_Edge](#) pf, [Bidly\\_Edge](#) pt, [Bidly\\_Variable](#) ptag, [Bidly\\_Boolean](#) garbageAllowed)

*Function Bidly\_Managed\_TaggedFoaNode finds or adds new node with the given variable and successors.*
- [Bidly\\_Edge Bidly\\_Managed\\_Not](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f)

*Function Bidly\_Managed\_Not calculates Boolean function NOT.*
- [Bidly\\_Edge Bidly\\_Managed\\_ITE](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f, [Bidly\\_Edge](#) g, [Bidly\\_Edge](#) h)

*Function Bidly\_Managed\_ITE calculates ITE operation of three Boolean functions.*
- [Bidly\\_Edge Bidly\\_Managed\\_And](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f, [Bidly\\_Edge](#) g)

*Function Bidly\_Managed\_And calculates Boolean function AND (conjunction).*
- [Bidly\\_Edge Bidly\\_Managed\\_Or](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f, [Bidly\\_Edge](#) g)

*Function Bidly\_Managed\_Or calculates Boolean function OR (disjunction).*
- [Bidly\\_Edge Bidly\\_Managed\\_Nand](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f, [Bidly\\_Edge](#) g)

*Function Bidly\_Managed\_Nand calculates Boolean function NAND (Sheffer).*
- [Bidly\\_Edge Bidly\\_Managed\\_Nor](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f, [Bidly\\_Edge](#) g)

*Function Bidly\_Managed\_Nor calculates Boolean function NOR (Peirce).*
- [Bidly\\_Edge Bidly\\_Managed\\_Xor](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f, [Bidly\\_Edge](#) g)

*Function Bidly\_Managed\_Xor calculates Boolean function XOR.*
- [Bidly\\_Edge Bidly\\_Managed\\_Xnor](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f, [Bidly\\_Edge](#) g)

*Function Bidly\_Managed\_Xnor calculates Boolean function XNOR.*
- [Bidly\\_Edge Bidly\\_Managed\\_Leq](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f, [Bidly\\_Edge](#) g)



- Function `Biddy_Managed_Leq` calculates Boolean implication.*

  - [Biddy\\_Edge](#) [Biddy\\_Managed\\_Gt](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Edge](#) f, [Biddy\\_Edge](#) g)

*Function `Biddy_Managed_Gt` calculates the negation of Boolean implication.*

  - [Biddy\\_Boolean](#) [Biddy\\_Managed\\_IsLeq](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Edge](#) f, [Biddy\\_Edge](#) g)

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

  - [Biddy\\_Edge](#) [Biddy\\_Managed\\_SubIntersect](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Edge](#) f, [Biddy\\_Edge](#) g)

*`Biddy_Managed_SubIntersect` 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\\_↔](#) Boolean value)

*Function `Biddy_Managed_Restrict` calculates a restriction of Boolean function.*

  - [Biddy\\_Edge](#) [Biddy\\_Managed\\_Compose](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Edge](#) f, [Biddy\\_Edge](#) g, [Biddy\\_↔](#) Variable v)

*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\\_Edge](#) [Biddy\\_Managed\\_Change](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Edge](#) f, [Biddy\\_Variable](#) v)

*Function `Biddy_Managed_Change` change the form of the given variable (positive literal becomes negative and vice versa).*

  - [Biddy\\_Edge](#) [Biddy\\_Managed\\_Subset](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Edge](#) f, [Biddy\\_Variable](#) v, [Biddy\\_↔](#) Boolean value)

*Function `Biddy_Managed_Subset` calculates a division of Boolean function with a literal.*

  - [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\\_GC](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Variable](#) target, [Biddy\\_Boolean](#) purge, [Biddy\\_↔](#) Boolean total)

*Function `Biddy_Managed_GC` performs garbage collection.*

  - void [Biddy\\_Managed\\_Clean](#) ([Biddy\\_Manager](#) MNG)

*Function `Biddy_Managed_Clean` performs cleaning.*

  - void [Biddy\\_Managed\\_Purge](#) ([Biddy\\_Manager](#) MNG)

*Function `Biddy_Managed_Purge` immediately removes all nodes which were not preserved or which are not preserved anymore.*

- void [Bidly\\_Managed\\_PurgeAndReorder](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f, [Bidly\\_Boolean](#) converge)  
*Function Bidly\_Managed\_PurgeAndReorder immediately removes non-preserved nodes and triggers reordering on function.*
- void [Bidly\\_Managed\\_Refresh](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f)  
*Function Bidly\_Managed\_Refresh refreshes top node in a given function.*
- void [Bidly\\_Managed\\_AddCache](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_GCFunction](#) gc)  
*Function Bidly\_Managed\_AddCache adds cache to the end of Cache list.*
- unsigned int [Bidly\\_Managed\\_AddFormula](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_String](#) x, [Bidly\\_Edge](#) f, int c)  
*Function Bidly\_Managed\_AddFormula adds formula to Formula table.*
- [Bidly\\_Boolean](#) [Bidly\\_Managed\\_FindFormula](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_String](#) x, [Bidly\\_Edge](#) \*f)  
*Function Bidly\_Managed\_FindFormula find formula in Formula table.*
- [Bidly\\_Boolean](#) [Bidly\\_Managed\\_DeleteFormula](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_String](#) x)  
*Function Bidly\_Managed\_DeleteFormula delete formula from Formula table.*
- [Bidly\\_Boolean](#) [Bidly\\_Managed\\_DeletelthFormula](#) ([Bidly\\_Manager](#) MNG, unsigned int i)  
*Function Bidly\_Managed\_DeletelthFormula deletes formula from the table.*
- [Bidly\\_Edge](#) [Bidly\\_Managed\\_GetlthFormula](#) ([Bidly\\_Manager](#) MNG, unsigned int i)  
*Function Bidly\_Managed\_GetlthFormula returns ith formula in a Formula table.*
- [Bidly\\_String](#) [Bidly\\_Managed\\_GetlthFormulaName](#) ([Bidly\\_Manager](#) MNG, unsigned int i)  
*Function Bidly\_Managed\_GetlthFormulaName returns name of the ith formula in a Formula table.*
- [Bidly\\_Variable](#) [Bidly\\_Managed\\_SwapWithHigher](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Variable](#) v)  
*Function Bidly\_Managed\_SwapWithHigher swaps two adjacent variables.*
- [Bidly\\_Variable](#) [Bidly\\_Managed\\_SwapWithLower](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Variable](#) v)  
*Function Bidly\_Managed\_SwapWithLower swaps two adjacent variables.*
- [Bidly\\_Boolean](#) [Bidly\\_Managed\\_Sifting](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f, [Bidly\\_Boolean](#) converge)  
*Function Bidly\_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.*
- [Bidly\\_Edge](#) [Bidly\\_Managed\\_Copy](#) ([Bidly\\_Manager](#) MNG1, [Bidly\\_Manager](#) MNG2, [Bidly\\_Edge](#) f)  
*Function Bidly\_Managed\_Copy copies a graph from one manager to another manager.*

**Description**

The function takes a graph from one manager and creates the same graph in another manager. The resulting graph will represent the same Boolean function assuming the domain from the target manager. If f = `bidlyZero` then only the domain is copied.

**Side effects**

If source and target manager are the same then function does nothing. The variable ordering of created BDD is adapted to the target manager.

**More info**

Macro `Bidly_Copy(MNG2,f)` is defined for use with anonymous manager.

- void [Bidly\\_Managed\\_CopyFormula](#) ([Bidly\\_Manager](#) MNG1, [Bidly\\_Manager](#) MNG2, [Bidly\\_String](#) x)  
*Function Bidly\_Managed\_CopyFormula uses Bidly\_Managed\_Copy to copy a graph from one manager to another manager.*

**Description**

See `Bidly_Managed_Copy`.

**Side effects**

If source and target manager are the same then function does nothing. The variable ordering of created BDD is adapted to the target manager. The created formula is not preserved.

**More info**

Macro `Bidly_CopyFormula(MNG2,x)` is defined for use with anonymous manager.

- [Bidly\\_Boolean](#) [Bidly\\_Managed\\_Eval](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f)  
*Function Bidly\_Managed\_Eval returns the value of a Boolean function for a given variable assignment.*

**Description****Side effects****More info**

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

- [Bidly\\_Edge Bidly\\_Managed\\_Random](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) support, double r)  
Function `Bidly_Managed_Random` generates a random BDD.
- [Bidly\\_Edge Bidly\\_Managed\\_RandomSet](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) unit, double r)  
Function `Bidly_Managed_RandomSet` generates a random BDD.

### 5.5.1 Detailed Description

File `bidlyMainLegacy.c` contains main functions for representation and manipulation of boolean functions with R↔OBDDs.

**Description**

```

PackageName [Bidly]
Synopsis [Bidly 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 system age is implemented.
Variable swapping and sifting are implemented.]

FileName [bidlyMainLegacy.c]
Revision [${Revision: 356 $}]
Date [${Date: 2017-12-13 22:30:46 +0100 (sre, 13 dec 2017) $}]
Authors [Robert Meolic (robert.meolic@um.si),
Ales Casar (ales@homemade.net)]

```

**Copyright**

Copyright (C) 2006, 2017 UM FERi, Koroska cesta 46, SI-2000 Maribor, Slovenia

Bidly 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.

Bidly 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: [bidly.h](#), [bidlyInt.h](#)

### 5.5.2 Function Documentation

#### 5.5.2.1 `void Bidly_InitMNG ( Bidly_Manager * mng, int gdtype )`

Function `Bidly_InitMNG` initialize a manager.

**Description**

Bidly\_InitMNG creates and initializes a manager. Initialization consists of creating manager structure (MNG), node table (bidlyNodeTable), variable table (bidlyVariableTable), formula table (bidlyFormulaTable), three basic caches (bidlyOPCache, bidlyEACache and bidlyRCCache), and cache list (bidlyCacheList). Bidly\_InitMNG also initializes constant edges (bidlyOne, bidlyZero), memory management and automatic garbage collection.

**Side effects**

Allocates a lot of memory. Parameter `gdtype` is ignored.

**More info**

Macro [Bidly\\_Init\(\)](#) will initialize anonymous manager.

Definition at line 210 of file `bidlyMainLegacy.c`.

**5.5.2.2 void Bidly\_ExitMNG ( Bidly\_Manager \* mng )**

Function `Bidly_ExitMNG` deletes a manager.

**Description**

Deallocates all memory allocated by `Bidly_InitMNG`, `Bidly_FoaVariable`, `Bidly_FoaNode` etc.

**Side effects****More info**

Macro [Bidly\\_Exit\(\)](#) will delete anonymous manager.

Definition at line 711 of file `bidlyMainLegacy.c`.

**5.5.2.3 Bidly\_String Bidly\_About ( )**

Function `Bidly_About` reports version of Bidly package.

**Description****Side effects****More info**

Definition at line 912 of file `bidlyMainLegacy.c`.

**5.5.2.4 int Bidly\_Managed\_GetManagerType ( Bidly\_Manager MNG )**

Function `Bidly_Managed_GetManagerType` reports BDD type used in the manager.

**Description****Side effects****More info**

Macro [Biddy\\_GetManagerType\(\)](#) is defined for use with anonymous manager.

Definition at line 934 of file biddyMainLegacy.c.

#### 5.5.2.5 void Biddy\_Managed\_SetManagerParameters ( Biddy\_Manager MNG, float gcr, float gcrF, float gcrX, float rr, float rrF, float rrX, float st, float cst )

Function Biddy\_Managed\_SetManagerParameters set modifiable parameters.

**Description**

Function expect 6 float values. If the value is  $< 0$  then the parameter is not modified. The parameters are: biddyNodeTable.gcratio (do not delete nodes if the effect is to small), biddyNodeTable.gcratioF (do not delete nodes if the effect is to small), biddyNodeTable.gcratioX (do not delete nodes if the effect is to small), biddyNodeTable.resizeratio (resize Node table if there are to many nodes), biddyNodeTable.resizeratioF (resize Node table if there are to many nodes), biddyNodeTable.resizeratioX (resize Node table if there are to many nodes), biddyNodeTable.siftingreshold (stop sifting if the size of the system grows to much), biddyNodeTable.fsiftingreshold (stop sifting if the size of the function grows to much), biddyNodeTable.convergesiftingreshold (stop one step of converging sifting if the size of the system grows to much), biddyNodeTable.fconvergesiftingreshold (stop one step of converging sifting if the size of the function grows to much).

**Side effects**

Initial values are given in Biddy\_InitMNG.

**More info**

Macro [Biddy\\_SetManagerParameters\(\)](#) is defined for use with anonymous manager.

Definition at line 977 of file biddyMainLegacy.c.

#### 5.5.2.6 Biddy\_Edge Biddy\_GetThen ( Biddy\_Edge fun )

Function Biddy\_GetThen returns THEN successor.

**Description**

Input mark is not transfered! External use, only.

**Side effects****More info**

Macro BiddyT(fun) is defined for internal use.

Definition at line 1011 of file biddyMainLegacy.c.

### 5.5.2.7 Biddy\_Edge Biddy\_GetElse ( Biddy\_Edge fun )

Function Biddy\_GetElse returns ELSE successor.

#### Description

Input mark is not transfered! External use, only.

#### Side effects

#### More info

Macro BiddyE(fun) is defined for internal use.

Definition at line 1047 of file biddyMainLegacy.c.

### 5.5.2.8 Biddy\_Variable Biddy\_GetTopVariable ( Biddy\_Edge fun )

Function Biddy\_GetTopVariable returns the top variable.

#### Description

External use, only.

#### Side effects

#### More info

Macro BiddyV(fun) is defined for internal use.

Definition at line 1083 of file biddyMainLegacy.c.

Here is the caller graph for this function:



### 5.5.2.9 Biddy\_Boolean Biddy\_Managed\_IsEqv ( Biddy\_Manager MNG1, Biddy\_Edge f1, Biddy\_Manager MNG2, Biddy\_Edge f2 )

Function Biddy\_Managed\_IsEqv returns TRUE iff two BDDs are equal.

**Description**

It is assumed that `f1` and `f2` have the same ordering.

**Side effects****More info**

Macro `Biddy_IsEqv(f1,MNG2,f2)` is defined for use with anonymous manager.

Definition at line 1107 of file `biddyMainLegacy.c`.

**5.5.2.10 void Biddy\_Managed\_SelectNode ( Biddy\_Manager MNG, Biddy\_Edge f )**

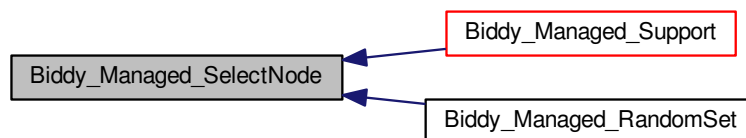
Function `Biddy_Managed_SelectNode` selects the top node of the given function.

**Description****Side effects****More info**

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

Definition at line 1142 of file `biddyMainLegacy.c`.

Here is the caller graph for this function:

**5.5.2.11 void Biddy\_Managed\_DeselectNode ( Biddy\_Manager MNG, Biddy\_Edge f )**

Function `Biddy_Managed_DeselectNode` deselects the top node of the given function.

**Description****Side effects****More info**

Macro [Bidy\\_DeselectNode\(f\)](#) is defined for use with anonymous manager.

Definition at line 1168 of file `bidyMainLegacy.c`.

Here is the caller graph for this function:

**5.5.2.12 Bidy\_Boolean Bidy\_Managed\_IsSelected ( Bidy\_Manager MNG, Bidy\_Edge f )**

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

**Description****Side effects****More info**

Macro [Bidy\\_IsSelected\(f\)](#) is defined for use with anonymous manager.

Definition at line 1194 of file `bidyMainLegacy.c`.

Here is the caller graph for this function:

**5.5.2.13 void Bidy\_Managed\_SelectFunction ( Bidy\_Manager MNG, Bidy\_Edge f )**

Function `Bidy_Managed_SelectFunction` recursively selects all nodes of a given function.



**Description****Side effects**

Constant node must be selected before starting this function!

**More info**

Macro [Biddy\\_SelectFunction\(f\)](#) is defined for use with anonymous manager.

Definition at line 1221 of file biddyMainLegacy.c.

**5.5.2.14 void Biddy\_Managed\_DeselectAll ( Biddy\_Manager MNG )**

Function Biddy\_Managed\_DeselectAll deselects all nodes.

**Description**

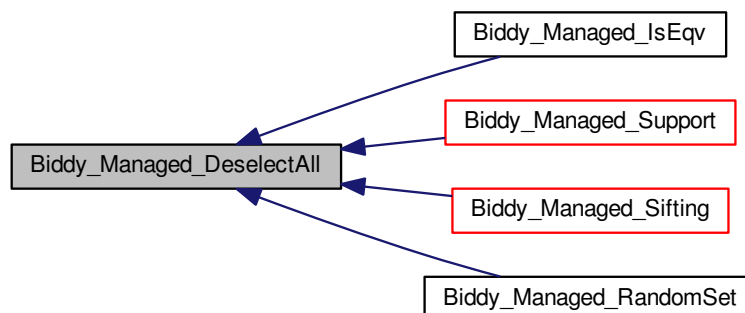
Deselect all nodes.

**Side effects****More info**

Macro [Biddy\\_DeselectAll\(\)](#) is defined for use with anonymous manager.

Definition at line 1278 of file biddyMainLegacy.c.

Here is the caller graph for this function:

**5.5.2.15 Biddy\_Edge Biddy\_Managed\_GetTerminal ( Biddy\_Manager MNG )**

Function `Biddy_Managed_GetTerminal` returns unmarked edge pointing to the constant node 1.

**Description**

Terminal node depends on a manager.

**Side effects****More info**

Internally, use macro `biddyTerminal`. Macro `Biddy_GetTerminal()` is defined for use with anonymous manager.

Definition at line 1312 of file `biddyMainLegacy.c`.

**5.5.2.16 Biddy\_Edge Biddy\_Managed\_GetConstantZero ( Biddy\_Manager MNG )**

Function `Biddy_Managed_GetConstantZero` returns constant 0.

**Description**

Constants 0 and 1 depend on a manager. For combination sets, constant 0 coincides with empty set.

**Side effects****More info**

Internally, use macro `biddyZero`. Macro `Biddy_GetConstantZero()` is defined for use with anonymous manager. Macros `Biddy_Managed_GetEmptySet(MNG)` and `Biddy_GetEmptySet()` are defined for manipulation of combination sets.

Definition at line 1342 of file `biddyMainLegacy.c`.

Here is the caller graph for this function:

**5.5.2.17 Biddy\_Edge Biddy\_Managed\_GetConstantOne ( Biddy\_Manager MNG )**

Function `Biddy_Managed_GetConstantOne` returns constant 1.

**Description**

Constants 0 and 1 depend on a manager. For combination sets, constant 1 coincides with universal set.

**Side effects****More info**

Internally, use macro `biddyOne`. Macro [Biddy\\_GetConstantOne\(\)](#) is defined for use with anonymous manager. Macros `Biddy_Managed_GetUniversalSet(MNG)` and `Biddy_GetUniversalSet()` are defined for manipulation of combination sets.

Definition at line 1372 of file `biddyMainLegacy.c`.

Here is the caller graph for this function:

**5.5.2.18 Biddy\_Edge Biddy\_Managed\_GetBaseSet ( Biddy\_Manager MNG )**

Function `Biddy_Managed_GetBaseSet` returns set containing only a null combination, i.e. it returns `{{}}`.

**Description****Side effects****More info**

Macro [Biddy\\_GetBaseSet\(\)](#) is defined for use with anonymous manager.

Definition at line 1398 of file `biddyMainLegacy.c`.

Here is the caller graph for this function:

**5.5.2.19 Biddy\_Variable Biddy\_Managed\_GetVariable ( Biddy\_Manager MNG, Biddy\_String x )**

Function `Biddy_Managed_GetVariable` returns variable with the given name.

**Description****Side effects**

If variable is not found function returns 0!

**More info**

Macro [Biddy\\_GetVariable\(x\)](#) is defined for use with anonymous manager.

Definition at line 1433 of file biddyMainLegacy.c.

Here is the caller graph for this function:

**5.5.2.20 Bidly\_Variable Bidly\_Managed\_GetPrevVariable ( Bidly\_Manager MNG, Bidly\_Variable v )**

Function Bidly\_Managed\_GetPrevVariable returns previous variable in the global ordering (lower, topmore).

**Description****Side effects****More info**

Macro [Biddy\\_GetPrevVariable\(v\)](#) is defined for use with anonymous manager.

Definition at line 1478 of file biddyMainLegacy.c.

**5.5.2.21 Bidly\_Variable Bidly\_Managed\_GetNextVariable ( Bidly\_Manager MNG, Bidly\_Variable v )**

Function Bidly\_Managed\_GetNextVariable returns next variable in the global ordering (higher, bottommore).

**Description****Side effects****More info**

Macro [Biddy\\_GetNextVariable\(v\)](#) is defined for use with anonymous manager.

Definition at line 1508 of file biddyMainLegacy.c.

#### 5.5.2.22 Biddy\_Edge Biddy\_Managed\_GetVariableEdge ( Biddy\_Manager *MNG*, Biddy\_Variable *v* )

Function Biddy\_Managed\_GetVariableEdge returns variable's edge.

Description

Side effects

More info

Macro [Biddy\\_GetVariableEdge\(v\)](#) is defined for use with anonymous manager.

Definition at line 1537 of file biddyMainLegacy.c.

Here is the caller graph for this function:



#### 5.5.2.23 Biddy\_Edge Biddy\_Managed\_GetElementEdge ( Biddy\_Manager *MNG*, Biddy\_Variable *v* )

Function Biddy\_Managed\_GetElementEdge returns element's edge.

Description

Side effects

More info

Macro [Biddy\\_GetElementEdge\(v\)](#) is defined for use with anonymous manager.

Definition at line 1562 of file biddyMainLegacy.c.

#### 5.5.2.24 Biddy\_String Biddy\_Managed\_GetVariableName ( Biddy\_Manager *MNG*, Biddy\_Variable *v* )

Function Biddy\_Managed\_GetVariableName returns the name of a variable.

Description

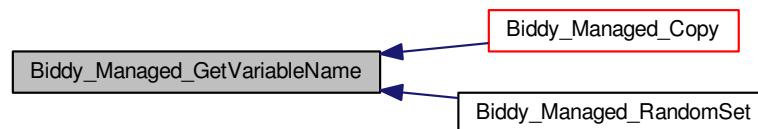
Side effects

More info

Macro [Bidly\\_GetVariableName\(v\)](#) is defined for use with anonymous manager.

Definition at line 1586 of file `bidlyMainLegacy.c`.

Here is the caller graph for this function:



#### 5.5.2.25 `Bidly_Edge Bidly_Managed_GetTopVariableEdge ( Bidly_Manager MNG, Bidly_Edge f )`

Function `Bidly_Managed_GetTopVariableEdge` returns variable's edge of top variable.

Description

Side effects

TO DO: For ZBDDs, element edge is sometimes preferred over variable edge.

More info

Macro [Bidly\\_GetTopVariableEdge\(f\)](#) is defined for use with anonymous manager.

Definition at line 1614 of file `bidlyMainLegacy.c`.

#### 5.5.2.26 `Bidly_String Bidly_Managed_GetTopVariableName ( Bidly_Manager MNG, Bidly_Edge f )`

Function `Bidly_Managed_GetTopVariableName` returns the name of top variable.

Description

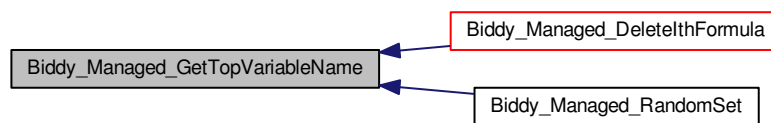
Side effects

More info

Macro [Biddy\\_GetTopVariableName\(f\)](#) is defined for use with anonymous manager.

Definition at line 1642 of file biddyMainLegacy.c.

Here is the caller graph for this function:



#### 5.5.2.27 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 1670 of file biddyMainLegacy.c.

#### 5.5.2.28 void Biddy\_Managed\_ResetVariablesValue ( Biddy\_Manager *MNG* )

Function `Biddy_Managed_ResetVariablesValue` sets all variable's value to `biddyZero`.

Description

Side effects

Only active (used) variables are reinitialized.

**More info**

Macro [Bidly\\_ResetVariablesValue\(\)](#) is defined for use with anonymous manager.

Definition at line 1699 of file bidlyMainLegacy.c.

5.5.2.29 `void Bidly_Managed_SetVariableValue ( Bidly_Manager MNG, Bidly_Variable v, Bidly_Edge f )`

Function `Bidly_Managed_SetVariableValue` sets variable's value.

**Description****Side effects****More info**

Macro [Bidly\\_SetVariableValue\(v,f\)](#) is defined for use with anonymous manager.

Definition at line 1728 of file bidlyMainLegacy.c.

5.5.2.30 `Bidly_Boolean Bidly_Managed_IsSmaller ( Bidly_Manager MNG, Bidly_Variable fv, Bidly_Variable gv )`

Function `Bidly_Managed_IsSmaller` returns TRUE if the first variable is smaller (= lower = previous = above = topmore).

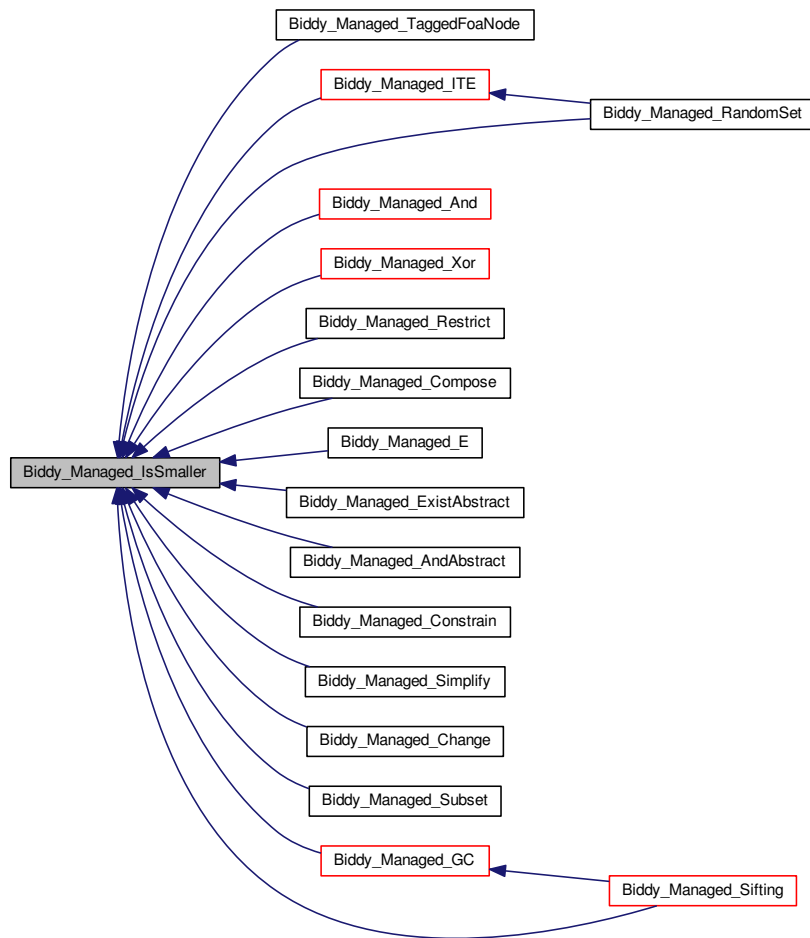
**Description****Side effects****More info**

Macro [Bidly\\_IsSmaller\(fv,gv\)](#) is defined for use with anonymous manager.

Definition at line 1754 of file bidlyMainLegacy.c.



Here is the caller graph for this function:



#### 5.5.2.31 Bidly\_Variable Bidly\_Managed\_FoaVariable ( Bidly\_Manager MNG, Bidly\_String x, Bidly\_Boolean varelem )

Function Bidly\_Managed\_FoaVariable finds variable/element or adds new variable (i.e. Boolean function  $f = x$ ) and new element (i.e. it creates set  $\{\{x\}\}$ ).

#### Description

If variable/element already exists, function returns the existing one. If  $x == \text{NULL}$  then numbered variable/element is added. Numbered variables/elements have only digits in its name. The current number of numbered variables/elements is stored in numnum. If numbered variable/element is requested then function increments numnum and creates a new (non-existing) variable/element. Parameter varelem is used to determine how to adapt the existing BDD base to keep the current formula valid (use varelem = TRUE if formulae represent Boolean functions and varelem = FALSE if they represent combination sets). The ordering of the new variable/element is determined in Bidly\_InitMNG. Function always returns variable.

**Side effects**

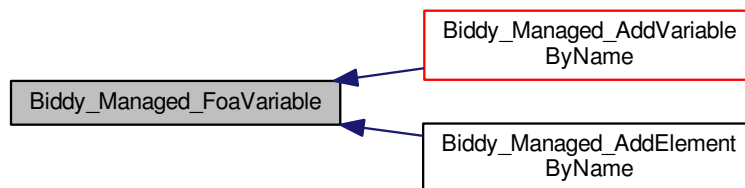
Adding new variable/element may change the meaning of the existing BDDs. Variables and elements are repaired. Moreover, formulae are repaired with regards to the parameter varelem. For OBDDs, it is safe to add new variables/elements if BDDs are used to represent Boolean functions. User should not add numbered variables/elements with some other function. TO DO: Formulae in user's formula tables are not repaired, yet!

**More info**

Macro [Biddy\\_FoaVariable\(x\)](#) is defined for use with anonymous manager.

Definition at line 1799 of file biddyMainLegacy.c.

Here is the caller graph for this function:



### 5.5.2.32 `Biddy_Edge Biddy_Managed_AddVariableByName ( Biddy_Manager MNG, Biddy_String x )`

Function `Biddy_Managed_AddVariableByName` adds variable.

**Description**

`Biddy_Managed_AddVariableByName` uses `Biddy_Managed_FoaVariable` to find or add variable. Function returns variable edge. If variable already exists, function returns the existing variable edge. For more details see `Biddy_Managed_FoaVariable`.

**Side effects**

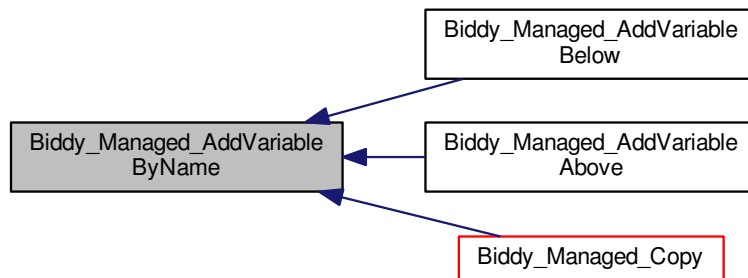
See `Biddy_Managed_FoaVariable`.

**More info**

Macro [Biddy\\_AddVariableByName\(x\)](#) is defined for use with anonymous manager. Macros [Biddy\\_Managed\\_AddVariable\(MNG\)](#) and [Biddy\\_AddVariable\(\)](#) are defined for creating numbered variables.

Definition at line 1915 of file biddyMainLegacy.c.

Here is the caller graph for this function:



### 5.5.2.33 **Biddy\_Edge** Biddy\_Managed\_AddElementByName ( Biddy\_Manager MNG, Biddy\_String x )

Function [Biddy\\_Managed\\_AddElementByName](#) adds element.

**Description**

[Biddy\\_Managed\\_AddElementByName](#) uses [Biddy\\_Managed\\_FoaVariable](#) to find or add element. Function returns element edge. If element already exists, function returns the existing element edge. For more details see [Biddy\\_Managed\\_FoaVariable](#).

**Side effects**

See [Biddy\\_Managed\\_FoaVariable](#).

**More info**

Macro [Biddy\\_AddElementByName\(x\)](#) is defined for use with anonymous manager. Macros [Biddy\\_Managed\\_AddElement\(MNG\)](#) and [Biddy\\_AddElement\(\)](#) are defined for creating numbered elements.

Definition at line 1950 of file biddyMainLegacy.c.

### 5.5.2.34 **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\_AddVariableByName to add numbered variable. Then, the order of the new variable is changed to become immediately below the given variable (below = next = bottom-more in BDD) Function returns variable edge.

**Side effects****More info**

Macro [Biddy\\_AddVariableBelow\(v\)](#) is defined for use with anonymous manager.

Definition at line 1982 of file biddyMainLegacy.c.

### 5.5.2.35 **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\_AddVariableByName to add numbered variable. Then, the order of the new variable is changed to become immediately above the given variable (above = previous = topmore in BDD) Function returns variable edge.

**Side effects****More info**

Macro [Biddy\\_AddVariableAbove\(v\)](#) is defined for use with anonymous manager.

Definition at line 2049 of file biddyMainLegacy.c.

### 5.5.2.36 **Biddy\_Edge** Biddy\_Managed\_TransferMark ( **Biddy\_Manager** *MNG*, **Biddy\_Edge** *f*, **Biddy\_Boolean** *mark*, **Biddy\_Boolean** *leftright* )

Function Biddy\_Managed\_TransferMark returns edge with inverted complement bit iff the second parameter is T↔RUE and normalization rules require this.

**Description**

It is better to use macro Biddy\_InvCond. Parameter leftright is ignored.

Side effects

More info

Macro [Biddy\\_TransferMark\(\)](#) is defined for use with anonymous manager.

Definition at line 2112 of file biddyMainLegacy.c.

Here is the caller graph for this function:



#### 5.5.2.37 `Biddy_Edge Biddy_Managed_IncTag ( Biddy_Manager MNG, Biddy_Edge f )`

Function `Biddy_Managed_IncTag` returns edge with an incremented tag.

Description

This function is not used for OBDDs.

Side effects

More info

Macro [Biddy\\_IncTag\(\)](#) is defined for use with anonymous manager.

Definition at line 2140 of file biddyMainLegacy.c.

#### 5.5.2.38 `Biddy_Edge Biddy_Managed_TaggedFoaNode ( Biddy_Manager MNG, Biddy_Variable v, Biddy_Edge pf, Biddy_Edge pt, Biddy_Variable ptag, Biddy_Boolean garbageAllowed )`

Function `Biddy_Managed_TaggedFoaNode` 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. If `pf = pt = NULL` then new variable is created. This function should not be called directly to add new variables, you must use `Biddy_Managed_FoaVariable` and `Biddy_Managed_AddVariableByName`.

Side effects

Parameter `ptag` is ignored. 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 sifting the function started with the returned node is not refreshed!

**More info**

Macro `Bidly_Managed_FoaNode(MNG,v,pf,pt,garbageAllowed)` is defined for use without tags. Macros [Bidly\\_↔ TaggedFoaNode\(v,pf,pt,tag,garbageAllowed\)](#) and `Bidly_FoaNode(v,pf,pt,garbageAllowed)` are defined for use with anonymous manager.

Definition at line 2178 of file `bidlyMainLegacy.c`.

**5.5.2.39 Bidly\_Edge Bidly\_Managed\_Not ( Bidly\_Manager MNG, Bidly\_Edge f )**

Function `Bidly_Managed_Not` calculates Boolean function NOT.

**Description**

It is better to use macro `Bidly_Inv`.

**Side effects****More info**

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

Definition at line 2394 of file `bidlyMainLegacy.c`.

Here is the caller graph for this function:

**5.5.2.40 Bidly\_Edge Bidly\_Managed\_ITE ( Bidly\_Manager MNG, Bidly\_Edge f, Bidly\_Edge g, Bidly\_Edge h )**

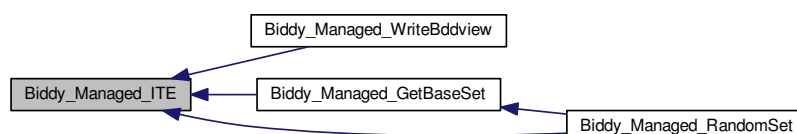
Function `Bidly_Managed_ITE` calculates ITE operation of three Boolean functions.

**Description****Side Effects****More info**

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

Definition at line 2428 of file `bidlyMainLegacy.c`.

Here is the caller graph for this function:



#### 5.5.2.41 Biddy\_Edge Biddy\_Managed\_And ( Biddy\_Manager *MNG*, Biddy\_Edge *f*, Biddy\_Edge *g* )

Function Biddy\_Managed\_And calculates Boolean function AND (conjunction).

##### Description

For combination sets, this function coincides with Intersection.

##### Side Effects

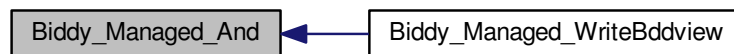
Used by ITE.

##### More Info

Macro [Biddy\\_And\(f,g\)](#) is defined for use with anonymous manager. Macros Biddy\_Managed\_Intersect(MNG,f,g) and Biddy\_Intersect(f,g) are defined for manipulation of combination sets.

Definition at line 2722 of file biddyMainLegacy.c.

Here is the caller graph for this function:



#### 5.5.2.42 Biddy\_Edge Biddy\_Managed\_Or ( Biddy\_Manager *MNG*, Biddy\_Edge *f*, Biddy\_Edge *g* )

Function Biddy\_Managed\_Or calculates Boolean function OR (disjunction).

##### Description

For combination sets, this function coincides with Union.

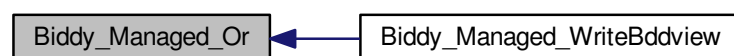
##### Side Effects

##### More Info

Macro [Biddy\\_Or\(f,g\)](#) is defined for use with anonymous manager. Macros Biddy\_Managed\_Union(MNG,f,g) and Biddy\_Union(f,g) are defined for manipulation of combination sets.

Definition at line 2938 of file biddyMainLegacy.c.

Here is the caller graph for this function:



#### 5.5.2.43 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

More Info

Macro [Biddy\\_Nand\(f,g\)](#) is defined for use with anonymous manager.

Definition at line 2993 of file biddyMainLegacy.c.

#### 5.5.2.44 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

More Info

Macro [Biddy\\_Nor\(f,g\)](#) is defined for use with anonymous manager.

Definition at line 3041 of file biddyMainLegacy.c.

#### 5.5.2.45 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

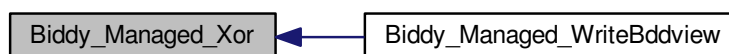
Used by ITE.

More Info

Macro [Biddy\\_Xor\(f,g\)](#) is defined for use with anonymous manager.

Definition at line 3089 of file biddyMainLegacy.c.

Here is the caller graph for this function:





**5.5.2.46 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

More Info

Macro [Biddy\\_Xnor\(f,g\)](#) is defined for use with anonymous manager.

Definition at line 3315 of file biddyMainLegacy.c.

**5.5.2.47 Biddy\_Edge Biddy\_Managed\_Leq ( Biddy\_Manager *MNG*, Biddy\_Edge *f*, Biddy\_Edge *g* )**

Function Biddy\_Managed\_Leq calculates Boolean implication.

Description

Side Effects

More Info

Macro [Biddy\\_Leq\(f,g\)](#) is defined for use with anonymous manager.

Definition at line 3362 of file biddyMainLegacy.c.

**5.5.2.48 Biddy\_Edge Biddy\_Managed\_Gt ( Biddy\_Manager *MNG*, Biddy\_Edge *f*, Biddy\_Edge *g* )**

Function Biddy\_Managed\_Gt calculates the negation of Boolean implication.

Description

For combination sets, this function coincides with Diff.

Side Effects

More Info

Macro [Biddy\\_Gt\(f,g\)](#) is defined for use with anonymous manager. Macros [Biddy\\_Managed\\_Diff\(MNG,f,g\)](#) and [Biddy\\_Diff\(f,g\)](#) are defined for manipulation of combination sets.

Definition at line 3411 of file biddyMainLegacy.c.

**5.5.2.49 Biddy\_Boolean Biddy\_Managed\_IsLeq ( Biddy\_Manager *MNG*, Biddy\_Edge *f*, Biddy\_Edge *g* )**

Function Biddy\_Managed\_IsLeq 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 [Bidly\\_IsLeq\(f,g\)](#) is defined for use with anonymous manager.

Definition at line 3460 of file biddyMainLegacy.c.

**5.5.2.50 Bidly\_Edge Bidly\_Managed\_SubIntersect ( Bidly\_Manager *MNG*, Bidly\_Edge *f*, Bidly\_Edge *g* )**

`Bidly_Managed_SubIntersect` 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, `Bidly_IsLeq` should be used (which returns TRUE iff  $f * g' == 0$ ).

**Side Effects**

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

**More Info**

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

Definition at line 3497 of file biddyMainLegacy.c.

**5.5.2.51 Bidly\_Edge Bidly\_Managed\_Restrict ( Bidly\_Manager *MNG*, Bidly\_Edge *f*, Bidly\_Variable *v*, Bidly\_Boolean *value* )**

Function `Bidly_Managed_Restrict` calculates a restriction of Boolean function.

**Description**

Original BDD is not changed. This is not Coudert and Madre's restrict function (use `Bidly_Simplify` if you need that one).

**Side effects**

Recursive calls use optimization:  $F(a=x) == \text{NOT}(\text{NOT } F(a=x))$ .

**More info**

Macro [Bidly\\_Restrict\(f,v,value\)](#) is defined for use with anonymous manager.

Definition at line 3531 of file biddyMainLegacy.c.

**5.5.2.52 Biddy\_Edge Biddy\_Managed\_Compose ( Biddy\_Manager *MNG*, Biddy\_Edge *f*, Biddy\_Edge *g*, Biddy\_Variable *v* )**

Function Biddy\_Managed\_Compose calculates a composition of two Boolean functions.

**Description**

Original BDDs are not changed.

**Side effects**

It uses optimization:  $F(a=G) == \text{NOT}(\text{NOT } F(a=G))$ .

**More info**

Macro [Biddy\\_Compose\(f,g,v\)](#) is defined for use with anonymous manager.

Definition at line 3622 of file biddyMainLegacy.c.

**5.5.2.53 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.

**Side effects**

Be careful:  $\text{ExA } F \neq \text{NOT}(\text{ExA } (\text{NOT } F))$ . Counterexample:  $\text{Exb } (\text{AND } (\text{NOT } a) b c)$ .

**More info**

Macro [Biddy\\_E\(f,v\)](#) is defined for use with anonymous manager.

Definition at line 3715 of file biddyMainLegacy.c.

**5.5.2.54 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.

**Side effects**

Be careful:  $\text{AxA } F \neq \text{NOT}(\text{AxA } (\text{NOT } F))$ . Counterexample:  $\text{Axb } (\text{AND } (\text{NOT } a) b c)$ .

**More info**

Macro [Biddy\\_A\(f,v\)](#) is defined for use with anonymous manager.

Definition at line 3808 of file biddyMainLegacy.c.

#### 5.5.2.55 **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.

**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 3865 of file biddyMainLegacy.c.

#### 5.5.2.56 **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**

Original BDD is not changed.

**Side effects****More info**

Macro [Biddy\\_ExistAbstract\(f,cube\)](#) is defined for use with anonymous manager.

Definition at line 3896 of file biddyMainLegacy.c.

#### 5.5.2.57 **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**

Original BDD is not changed.

Side effects

More info

Macro [Biddy\\_UnivAbstract\(f,cube\)](#) is defined for use with anonymous manager.

Definition at line 4001 of file biddyMainLegacy.c.

#### 5.5.2.58 **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`.

Description

Side effects

More info

Macro [Biddy\\_AndAbstract\(f,g,cube\)](#) is defined for use with anonymous manager.

Definition at line 4056 of file biddyMainLegacy.c.

#### 5.5.2.59 **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.

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 4215 of file biddyMainLegacy.c.

#### 5.5.2.60 **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.

**Description**

Coudert and Madre's restrict function tries to simplify function  $f$  by restricting it to the domain covered by function  $c$ . No checks are done to see if the result is actually smaller than the input.

**Side effects**

Cache table is not implemented, yet.

**More info**

Macro [Bidy\\_Simplify\(f,c\)](#) is defined for use with anonymous manager.

Definition at line 4308 of file biddyMainLegacy.c.

**5.5.2.61 Bidy\_Edge Bidy\_Managed\_Support ( Bidy\_Manager MNG, Bidy\_Edge  $f$  )**

Function Bidy\_Managed\_Support calculates a product of all dependent variables.

**Description****Side effects****More info**

Macro [Bidy\\_Support\(f\)](#) is defined for use with anonymous manager.

Definition at line 4401 of file biddyMainLegacy.c.

Here is the caller graph for this function:

**5.5.2.62 Bidy\_Edge Bidy\_Managed\_Replace ( Bidy\_Manager MNG, Bidy\_Edge  $f$  )**

Function Bidy\_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 Bidy\_Managed\_ResetVariablesValue and Bidy\_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 4471 of file `biddyMainLegacy.c`.

**5.5.2.63 Biddy\_Edge Biddy\_Managed\_Change ( Biddy\_Manager *MNG*, Biddy\_Edge *f*, Biddy\_Variable *v* )**

Function `Biddy_Managed_Change` change the form of the given variable (positive literal becomes negative and vice versa).

**Description****Side effects****More info**

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

Definition at line 4539 of file `biddyMainLegacy.c`.

**5.5.2.64 Biddy\_Edge Biddy\_Managed\_Subset ( Biddy\_Manager *MNG*, Biddy\_Edge *f*, Biddy\_Variable *v*, Biddy\_Boolean *value* )**

Function `Biddy_Managed_Subset` calculates a division of Boolean function with a literal.

**Description**

Original BDD is not changed. For combination sets, this function coincides with `Subset0` and `Subset1`.

**Side effects****More info**

Macro `Biddy_Subset(f,v,value)` is defined for use with anonymous manager. Macros `Biddy_Managed_Subset0(MNG,f,v)`, `Biddy_Subset0(f,v)`, `Biddy_Managed_Subset1(MNG,f,v)`, and `Biddy_Subset1(f,v)` are defined for manipulation of combination sets.

Definition at line 4608 of file `biddyMainLegacy.c`.

**5.5.2.65 Biddy\_Boolean Biddy\_Managed\_IsOK ( Biddy\_Manager *MNG*, Biddy\_Edge *f* )**

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

**Description**

This is needed for implementation of user caches.

**Side effects****More info**

Macro `BiddyIsOK(f)` is defined for debugging. It will check more properties and not only the expiry value. Macro `Biddy_IsOK(f)` is defined for use with anonymous manager.

Definition at line 4714 of file `biddyMainLegacy.c`.

**5.5.2.66** `void Biddy_Managed_GC ( Biddy_Manager MNG, Biddy_Variable target, Biddy_Boolean purge, Biddy_Boolean total )`

Function `Biddy_Managed_GC` performs garbage collection.

**Description**

All obsolete nodes are deleted. Parameter `target` is used during sifting. If parameter `total` is true than all obsolete nodes are deleted, otherwise nodes are deleted only if there are enough obsolete nodes. Nodes from deleted non-obsolete formulae are immediately removed only if parameter `purge` is true (this should not be used during the automatic garbage collection), otherwise these nodes only become fresh.

**Side effects**

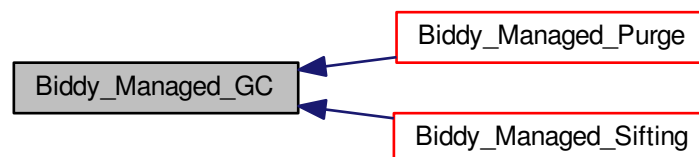
The first element of each chain in a node table should have a special value for its 'prev' element to allow tricky but efficient deleting. Moreover, 'prev' and 'next' should be the first and the second element in the structure `BiddyNode`, respectively. Garbage collection is reported by `biddyNodeTable.garbage` only if some bad nodes are purged!

**More info**

Macro `Biddy_GC(target,purge,total)` is defined for use with anonymous manager. Macros `Biddy_Managed_AutoGC(MNG)` and `Biddy_AutoGC()` are useful variants with `target = 0`, `purge = FALSE`, and `total = FALSE`.

Definition at line 4754 of file `biddyMainLegacy.c`.

Here is the caller graph for this function:





### 5.5.2.67 void Biddy\_Managed\_Clean ( Biddy\_Manager MNG )

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 and no prolonged node is discarded. Constants and variables should be fortified! Restoring elements is not implemented, yet.

#### More info

Macro [Biddy\\_Clean\(\)](#) is defined for use with anonymous manager.

Definition at line 5097 of file biddyMainLegacy.c.

### 5.5.2.68 void Biddy\_Managed\_Purge ( Biddy\_Manager MNG )

Function Biddy\_Managed\_Purge immediately removes all nodes which were not preserved or which are not preserved anymore.

#### Description

All fresh and obsolete nodes are immediately removed. Moreover, nodes from deleted prolonged formulae and nodes from deleted fortified formulae are removed if they are not needed by other formulae. Call to Biddy\_Purge does not count as clearing and thus all preserved formulae remains preserved for the same number of clearings.

#### Side effects

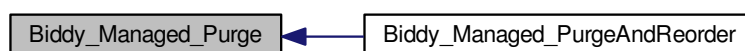
Removes all fresh nodes!

#### More info

Macro [Biddy\\_Purge\(f\)](#) is defined for use with anonymous manager.

Definition at line 5131 of file biddyMainLegacy.c.

Here is the caller graph for this function:



5.5.2.69 void `Biddy_Managed_PurgeAndReorder` ( `Biddy_Manager MNG`, `Biddy_Edge f`, `Biddy_Boolean converge` )

Function `Biddy_Managed_PurgeAndReorder` immediately removes non-preserved nodes and triggers reordering on function.

#### Description

All obsolete nodes are immediately removed. Moreover, nodes from deleted prolonged formulae and nodes from deleted fortified formulae are removed if they are not needed by other formulae. If BDD is given (`f != NULL`), reordering on function is used. Otherwise (`f == NULL`) global reordering is used. Call to `Biddy_PurgeAndReorder` does not count as clearing and thus all preserved formulae remains preserved for the same number of clearings.

#### Side effects

Removes all fresh nodes.

#### More info

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

Definition at line 5167 of file `biddyMainLegacy.c`.

5.5.2.70 void `Biddy_Managed_Refresh` ( `Biddy_Manager MNG`, `Biddy_Edge f` )

Function `Biddy_Managed_Refresh` refreshes top node in a given function.

#### Description

This is an external variant of internal macro `BiddyRefresh` This is needed for implementing user caches.

#### Side effects

#### More info

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

Definition at line 5196 of file `biddyMainLegacy.c`.

5.5.2.71 void `Biddy_Managed_AddCache` ( `Biddy_Manager MNG`, `Biddy_GCFunction gc` )

Function `Biddy_Managed_AddCache` adds cache to the end of Cache list.

#### Description

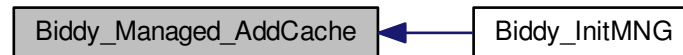
If Cache list does not exist, function creates it.

**Side effects****More info**

Macro [Biddy\\_AddCache\(gc\)](#) is defined for use with anonymous manager.

Definition at line 5222 of file biddyMainLegacy.c.

Here is the caller graph for this function:



#### 5.5.2.72 unsigned int Biddy\_Managed\_AddFormula ( Biddy\_Manager MNG, Biddy\_String x, Biddy\_Edge f, int c )

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 ( $x \neq \text{NULL}$ ) then formula is accessible by its name. If ( $c == -1$ ) then formula is not preserved. If ( $c == 0$ ) then formula is persistently preserved and you have to use `Biddy_DeleteFormula` to remove its nodes. There are two macros defined to simplify formulae management. Macro `Biddy_Managed_AddTmpFormula(mng,bdd,c)` is defined as `Biddy_Managed_AddFormula(mng, NULL,bdd,c)` and macro `Biddy_Managed_AddPersistentFormula(mng,name,bdd)` is defined as `Biddy_Managed_AddFormula(mng,name,bdd,0)`.

**Side effects**

Function is prolonged or fortified. Formulae with name are ordered by name. If formula with the same name already exists, it will be overwritten (preserved and persistently preserved formulae, too)!

**More info**

Macros [Biddy\\_AddFormula\(x,f\)](#), `Biddy_AddTmpFormula(f,c)`, and `Biddy_AddPersistentFormula(x,f)` are defined for use with anonymous manager.

Definition at line 5283 of file biddyMainLegacy.c.

Here is the caller graph for this function:



### 5.5.2.73 Biddy\_Boolean Biddy\_Managed\_FindFormula ( Biddy\_Manager *MNG*, Biddy\_String *x*, Biddy\_Edge \* *f* )

Function Biddy\_Managed\_FindFormula find formula in Formula table.

#### Description

#### Side effects

#### More info

Macro [Biddy\\_FindFormula\(x,f\)](#) is defined for use with anonymous manager.

Definition at line 5476 of file biddyMainLegacy.c.

Here is the caller graph for this function:



### 5.5.2.74 Biddy\_Boolean Biddy\_Managed\_DeleteFormula ( Biddy\_Manager *MNG*, Biddy\_String *x* )

Function Biddy\_Managed\_DeleteFormula delete formula from Formula table.

#### Description

Formula is labelled but not immediately removed. Nodes of the given formula are not immediately removed.

#### Side effects

Formula is not accessible by its name anymore. Formulae representing constants and variables will not be deleted.

#### More info

Macro [Biddy\\_DeleteFormula\(x\)](#) is defined for use with anonymous manager.

Definition at line 5577 of file biddyMainLegacy.c.

### 5.5.2.75 Biddy\_Boolean Biddy\_Managed\_DeletelthFormula ( Biddy\_Manager *MNG*, unsigned int *i* )

Function Biddy\_Managed\_DeletelthFormula deletes formula from the table.

**Description**

Formula is labelled but not immediately removed. Nodes of the given formula are not immediately removed.

**Side effects**

Formula is not accessible by its name anymore. The first two formulae ("0" and "1") will not be deleted. Formulae representing variables will not be deleted.

**More info**

Macro [Biddy\\_DeleteIthFormula\(x\)](#) is defined for use with anonymous manager.

Definition at line 5640 of file biddyMainLegacy.c.

Here is the caller graph for this function:

**5.5.2.76 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 5700 of file biddyMainLegacy.c.

**5.5.2.77 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!

**More info**

Macro [Bidly\\_GetIthFormulaName\(i\)](#) is defined for use with anonymous manager.

Definition at line 5731 of file bidlyMainLegacy.c.

**5.5.2.78 Bidly\_Variable Bidly\_Managed\_SwapWithHigher ( Bidly\_Manager *MNG*, Bidly\_Variable *v* )**

Function Bidly\_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 zeros in corresponding line of orderingTable.

**Side effects**

All obsolete nodes will be removed.

**More info**

Macro [Bidly\\_SwapWithHigher\(v\)](#) is defined for use with anonymous manager.

Definition at line 5772 of file bidlyMainLegacy.c.

**5.5.2.79 Bidly\_Variable Bidly\_Managed\_SwapWithLower ( Bidly\_Manager *MNG*, Bidly\_Variable *v* )**

Function Bidly\_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 zeros in corresponding line of orderingTable.

**Side effects**

All obsolete nodes will be removed.

**More info**

Macro [Biddy\\_SwapWithLower\(v\)](#) is defined for use with anonymous manager.

Definition at line 5810 of file biddyMainLegacy.c.

**5.5.2.80 Biddy\_Boolean Biddy\_Managed\_Sifting ( Biddy\_Manager MNG, Biddy\_Edge f, Biddy\_Boolean converge )**

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**

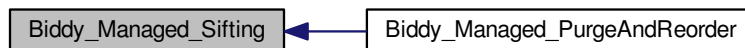
Variables are reordered globally. All obsolete nodes will be removed.

**Side effects****More info**

Macro [Biddy\\_Sifting\(f\)](#) is defined for use with anonymous manager.

Definition at line 5847 of file biddyMainLegacy.c.

Here is the caller graph for this function:

**5.5.2.81 Biddy\_Edge Biddy\_Managed\_Random ( Biddy\_Manager MNG, Biddy\_Edge support, double r )**

Function `Biddy_Managed_Random` generates a random BDD.

**Description**

The represented Boolean function depends on the variables given with parameter `support` whilst the parameter `r` determines the ratio between the number of function's minterms and the number of all possible minterms. Parameter `support` is a product of positive variables.

**Side effects**

Parameter `r` must be a number from `[0,1]`. Otherwise, function returns `biddyNull`.

**More info**

Macro `Biddy_Random(support,r)` is defined for use with anonymous manager.

Definition at line 6942 of file `biddyMainLegacy.c`.

**5.5.2.82 Biddy\_Edge Biddy\_Managed\_RandomSet ( Biddy\_Manager MNG, Biddy\_Edge unit, double r )**

Function `Biddy_Managed_RandomSet` generates a random BDD.

**Description**

The represented set is a random combination set determined by the parameter `unit` whilst the parameter `r` determines the ratio between the number of set's subsets and the number of all possible subsets. Parameter `set` is a set containing only one subset which consist of all elements, i.e. it is a set  $\{x_1, x_2, \dots, x_n\}$ .

**Side effects**

Parameter `r` must be a number from  $[0,1]$ . Otherwise, function returns `biddyNull`.

**More info**

Macro `Biddy_RandomSet(unit,r)` is defined for use with anonymous manager.

Definition at line 7054 of file `biddyMainLegacy.c`.

**5.6 biddyOp.c File Reference**

File `biddyOp.c` contains functions for operations on various types of Binary Decision Diagrams.

```
#include "biddyInt.h"
```

**Functions**

- [Biddy\\_Edge Biddy\\_Managed\\_Not](#) ([Biddy\\_Manager MNG](#), [Biddy\\_Edge f](#))  
*Function `Biddy_Managed_Not` calculates Boolean function NOT.*
- [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 Bidy\_Managed\_Xor calculates Boolean function XOR.*

  - [Bidy\\_Edge Bidy\\_Managed\\_Xnor](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f, [Bidy\\_Edge](#) g)

*Function Bidy\_Managed\_Xnor calculates Boolean function XNOR.*
- [Bidy\\_Edge Bidy\\_Managed\\_Leq](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f, [Bidy\\_Edge](#) g)

*Function Bidy\_Managed\_Leq calculates Boolean implication.*
- [Bidy\\_Edge Bidy\\_Managed\\_Gt](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f, [Bidy\\_Edge](#) g)

*Function Bidy\_Managed\_Gt calculates the negation of Boolean implication.*
- [Bidy\\_Boolean Bidy\\_Managed\\_IsLeq](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f, [Bidy\\_Edge](#) g)

*Function Bidy\_Managed\_IsLeq returns TRUE iff function f is included in function g.*
- [Bidy\\_Edge Bidy\\_Managed\\_Restrict](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f, [Bidy\\_Variable](#) v, [Bidy\\_↔ Boolean](#) value)

*Function Bidy\_Managed\_Restrict calculates a restriction of Boolean function.*
- [Bidy\\_Edge Bidy\\_Managed\\_Compose](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f, [Bidy\\_Edge](#) g, [Bidy\\_↔ Variable](#) v)

*Function Bidy\_Managed\_Compose calculates a composition of two Boolean functions.*
- [Bidy\\_Edge Bidy\\_Managed\\_E](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f, [Bidy\\_Variable](#) v)

*Function Bidy\_Managed\_E calculates an existential quantification of Boolean function.*
- [Bidy\\_Edge Bidy\\_Managed\\_A](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f, [Bidy\\_Variable](#) v)

*Function Bidy\_Managed\_A calculates an universal quantification of Boolean function.*
- [Bidy\\_Boolean Bidy\\_Managed\\_IsVariableDependent](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f, [Bidy\\_Variable](#) v)

*Function Bidy\_Managed\_IsVariableDependent returns TRUE iff variable is dependent on others in a function.*
- [Bidy\\_Edge Bidy\\_Managed\\_ExistAbstract](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f, [Bidy\\_Edge](#) cube)

*Function Bidy\_Managed\_ExistAbstract existentially abstracts all the variables in cube from f.*
- [Bidy\\_Edge Bidy\\_Managed\\_UnivAbstract](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f, [Bidy\\_Edge](#) cube)

*Function Bidy\_Managed\_UnivAbstract universally abstracts all the variables in cube from f.*
- [Bidy\\_Edge Bidy\\_Managed\\_AndAbstract](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f, [Bidy\\_Edge](#) g, [Bidy\\_↔ Edge](#) cube)

*Function Bidy\_Managed\_AndAbstract calculates the AND of two BDDs and simultaneously (existentially) abstracts the variables in cube.*
- [Bidy\\_Edge Bidy\\_Managed\\_Constrain](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f, [Bidy\\_Edge](#) c)

*Function Bidy\_Managed\_Constrain calculates Coudert and Madre's constrain function.*
- [Bidy\\_Edge Bidy\\_Managed\\_Simplify](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f, [Bidy\\_Edge](#) c)

*Function Bidy\_Managed\_Simplify calculates Coudert and Madre's restrict function.*
- [Bidy\\_Edge Bidy\\_Managed\\_Support](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f)

*Function Bidy\_Managed\_Support calculates a product of all dependent variables (OBDD and TZBDD) or the combination set containing a subset which includes all dependent variables (ZBDD).*
- [Bidy\\_Edge Bidy\\_Managed\\_ReplaceByKeyword](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f, [Bidy\\_String](#) keyword)

*Function Bidy\_Managed\_ReplaceByKeyword calculates Boolean function with one or more variables replaced.*
- [Bidy\\_Edge Bidy\\_Managed\\_Change](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f, [Bidy\\_Variable](#) v)

*Function Bidy\_Managed\_Change change the form of the given variable (positive literal becomes negative and vice versa).*
- [Bidy\\_Edge Bidy\\_Managed\\_Subset](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) f, [Bidy\\_Variable](#) v, [Bidy\\_↔ Boolean](#) value)

*Function Bidy\_Managed\_Subset calculates a division of Boolean function with a literal.*
- [Bidy\\_Edge Bidy\\_Managed\\_CreateMinterm](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) support, long long unsigned int x)

*Function Bidy\_Managed\_CreateMinterm generates one minterm.*
- [Bidy\\_Edge Bidy\\_Managed\\_CreateFunction](#) ([Bidy\\_Manager](#) MNG, [Bidy\\_Edge](#) support, long long unsigned int x)

*Function Bidy\_Managed\_CreateFunction generates one Boolean function.*

- [Biddy\\_Edge Biddy\\_Managed\\_RandomFunction](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Edge](#) support, double r)  
*Function Biddy\_Managed\_RandomFunction generates a random BDD.*
- [Biddy\\_Edge Biddy\\_Managed\\_RandomSet](#) ([Biddy\\_Manager](#) MNG, [Biddy\\_Edge](#) unit, double r)  
*Function Biddy\_Managed\_RandomSet generates a random BDD.*

### 5.6.1 Detailed Description

File [biddyOp.c](#) contains functions for operations on various types of Binary Decision Diagrams.

#### Description

```

PackageName [Biddy]
Synopsis    [Biddy provides data structures and algorithms for the
             representation and manipulation of Boolean functions with
             ROBDDs, 0-sup-BDDs, and TZBDDs. A hash table is used for quick
             search of nodes. Complement edges decreases the number of
             nodes. An automatic garbage collection with a system age is
             implemented. Variable swapping and sifting are implemented.]

FileName    [biddyOp.c]
Revision    [${Revision: 365 $}]
Date        [${Date: 2017-12-18 13:01:40 +0100 (pon, 18 dec 2017) $}]
Authors     [Robert Meolic (robert.meolic@um.si)]

```

#### Copyright

Copyright (C) 2006, 2017 UM FERl, Koroska cesta 46, 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](#)

### 5.6.2 Function Documentation

#### 5.6.2.1 Biddy\_Edge Biddy\_Managed\_Not ( Biddy\_Manager MNG, Biddy\_Edge f )

Function Biddy\_Managed\_Not calculates Boolean function NOT.

#### Description

#### Side effects

Implemented for OBDD, OBDDC, ZBDDC, and TZBDD. For OBDDC and OFDDC, it is better to use macro Biddy↔\_Inv. For OBDDC, cache table is not needed. For ZBDDC, recursive calls are via Xor and thus its cache table is used. For OBDD and TZBDD, results are cached as (f,biddyZero,biddyOne).

**More info**

Macro [Biddy\\_Not\(\)](#) is defined for use with anonymous manager.

Definition at line 100 of file bidyOp.c.

### 5.6.2.2 [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**

Implemented for OBDD, OBDDC, ZBDDC, and TzBDD. For OBDDC, results are cached as parameters to  $ITE(F, \leftrightarrow G, H) = F * G \text{ XOR } F' * H$ . For all other BDD types, results are cached as  $(f, g, h)$  where  $f, g, h \neq 0$ ,  $f \neq g$ ,  $f \neq h$ , and  $g \neq h$ .

**More info**

Macro [Biddy\\_ITE\(f,g,h\)](#) is defined for use with anonymous manager.

Definition at line 366 of file bidyOp.c.

### 5.6.2.3 [Biddy\\_Edge Biddy\\_Managed\\_And](#) ( [Biddy\\_Manager MNG](#), [Biddy\\_Edge f](#), [Biddy\\_Edge g](#) )

Function [Biddy\\_Managed\\_And](#) calculates Boolean function AND (conjunction).

**Description**

For combination sets, this function coincides with Intersection.

**Side Effects**

Used by ITE (for OBDD). Implemented for OBDD, OBDDC, ZBDDC, and TzBDD. For OBDDC, results are cached as parameters to  $ITE(F, G, H) = F * G \text{ XOR } F' * H$ . For all other BDD types, results are cached as  $(f, g, \text{bidyZero})$ .

**More Info**

Macro [Biddy\\_And\(f,g\)](#) is defined for use with anonymous manager. Macros [Biddy\\_Managed\\_Intersect\(MNG,f,g\)](#) and [Biddy\\_Intersect\(f,g\)](#) are defined for manipulation of combination sets.

Definition at line 804 of file bidyOp.c.

### 5.6.2.4 [Biddy\\_Edge Biddy\\_Managed\\_Or](#) ( [Biddy\\_Manager MNG](#), [Biddy\\_Edge f](#), [Biddy\\_Edge g](#) )

Function [Biddy\\_Managed\\_Or](#) calculates Boolean function OR (disjunction).

**Description**

For combination sets, this function coincides with Union.

**Side Effects**

Implemented for OBDD, OBDDC, ZBDDC, and TzBDD. For OBDDC, results are cached as parameters to  $\text{ITE}(F, \leftarrow G, H) = F * G \text{ XOR } F' * H$ . For all other BDD types, results are cached as  $(\text{bidlyZero}, f, g)$ .

**More Info**

Macro [Bidly\\_Or\(f,g\)](#) is defined for use with anonymous manager. Macros [Bidly\\_Managed\\_Union\(MNG,f,g\)](#) and [Bidly\\_Union\(f,g\)](#) are defined for manipulation of combination sets.

Definition at line 1230 of file `bidlyOp.c`.

**5.6.2.5 Bidly\_Edge Bidly\_Managed\_Nand ( Bidly\_Manager MNG, Bidly\_Edge f, Bidly\_Edge g )**

Function `Bidly_Managed_Nand` calculates Boolean function NAND (Sheffer).

**Description****Side Effects**

Implemented for OBDDC. Prototyped for OBDD. Prototyped for ZBDDC and TzBDD (via `and-not`). For OBDDC, results are cached as parameters to  $\text{ITE}(F, G, H) = F * G \text{ XOR } F' * H$ . For OBDD, ZBDDC and TzBDD, results could be cached as  $(f, g, \text{bidlyNull})$ .

**More Info**

Macro [Bidly\\_Nand\(f,g\)](#) is defined for use with anonymous manager.

Definition at line 1693 of file `bidlyOp.c`.

**5.6.2.6 Bidly\_Edge Bidly\_Managed\_Nor ( Bidly\_Manager MNG, Bidly\_Edge f, Bidly\_Edge g )**

Function `Bidly_Managed_Nor` calculates Boolean function NOR (Peirce).

**Description****Side Effects**

Implemented for OBDDC. Prototyped for OBDD. Prototyped for ZBDDC and TzBDD (via `or-not`). For OBDDC, results are cached as parameters to  $\text{ITE}(F, G, H) = F * G \text{ XOR } F' * H$ . For OBDD, ZBDDC and TzBDD, results could be cached as  $(\text{bidlyNull}, f, g)$ .

**More Info**

Macro [Bidly\\_Nor\(f,g\)](#) is defined for use with anonymous manager.

Definition at line 1788 of file `bidlyOp.c`.

### 5.6.2.7 Bidy\_Edge Bidy\_Managed\_Xor ( Bidy\_Manager *MNG*, Bidy\_Edge *f*, Bidy\_Edge *g* )

Function Bidy\_Managed\_Xor calculates Boolean function XOR.

#### Description

#### Side Effects

Used by ITE (for OBDDC). Implemented for OBDD, OBDDC, ZBDDC, and TZBDD. For OBDDC, results are cached as parameters to  $ITE(F,G,H) = F * G \text{ XOR } F' * H$ . For all other BDD types, results are cached as  $(f, \text{bidyZero}, g)$ .

#### More Info

Macro [Bidy\\_Xor\(f,g\)](#) is defined for use with anonymous manager.

Definition at line 1882 of file bidyOp.c.

### 5.6.2.8 Bidy\_Edge Bidy\_Managed\_Xnor ( Bidy\_Manager *MNG*, Bidy\_Edge *f*, Bidy\_Edge *g* )

Function Bidy\_Managed\_Xnor calculates Boolean function XNOR.

#### Description

#### Side Effects

Implemented for OBDDC. Prototyped for OBDD. Prototyped for ZBDDC and TZBDD (via xor-not). For OBDDC, results are cached as parameters to  $ITE(F,G,H) = F * G \text{ XOR } F' * H$ . For OBDD, ZBDDC and TZBDD, results could be cached as  $(f, \text{bidyNull}, g)$ .

#### More Info

Macro [Bidy\\_Xnor\(f,g\)](#) is defined for use with anonymous manager.

Definition at line 2319 of file bidyOp.c.

### 5.6.2.9 Bidy\_Edge Bidy\_Managed\_Leq ( Bidy\_Manager *MNG*, Bidy\_Edge *f*, Bidy\_Edge *g* )

Function Bidy\_Managed\_Leq calculates Boolean implication.

#### Description

Boolean function  $\text{leq}(f,g) = \text{or}(\text{not}(f),g) = \text{not}(\text{gt}(f,g))$ . This function coincides with implication  $f \rightarrow g$ .

#### Side Effects

Implemented for OBDD, OBDDC, ZBDDC, and TZBDD. For OBDDC, results are cached as parameters to  $ITE(F,\leftrightarrow G,H) = F * G \text{ XOR } F' * H$ . For all other BDD types, results are cached as  $(f,f,g)$ .

**More Info**

Macro [Bidly\\_Leq\(f,g\)](#) is defined for use with anonymous manager.

Definition at line 2414 of file bidlyOp.c.

**5.6.2.10 Bidly\_Edge Bidly\_Managed\_Gt ( Bidly\_Manager MNG, Bidly\_Edge f, Bidly\_Edge g )**

Function Bidly\_Managed\_Gt calculates the negation of Boolean implication.

**Description**

Boolean function  $gt(f,g) = \text{and}(f, \text{not}(g))$ . For combination sets, this function coincides with Diff.

**Side Effects**

Implemented for OBDD, OBDDC, ZBDDC, and TZBDD. For OBDDC, results are cached as parameters to  $\text{ITE}(F, \leftrightarrow G, H) = F * G \text{ XOR } F' * H$ . For all other BDD types, results are cached as (f,g,g).

**More Info**

Macro [Bidly\\_Gt\(f,g\)](#) is defined for use with anonymous manager. Macros Bidly\_Managed\_Diff(MNG,f,g) and Bidly\_Diff(f,g) are defined for manipulation of combination sets.

Definition at line 2726 of file bidlyOp.c.

**5.6.2.11 Bidly\_Boolean Bidly\_Managed\_IsLeq ( Bidly\_Manager MNG, Bidly\_Edge f, Bidly\_Edge g )**

Function Bidly\_Managed\_IsLeq returns TRUE iff function f is included in function g.

**Description****Side Effects**

Prototyped for OBDDs, ZBDDs, and TZBDDs (via calculating full implication, this is less efficient as implementation in CUDD).

**More Info**

Macro [Bidly\\_IsLeq\(f,g\)](#) is defined for use with anonymous manager.

Definition at line 3045 of file bidlyOp.c.

**5.6.2.12 Bidly\_Edge Bidly\_Managed\_Restrict ( Bidly\_Manager MNG, Bidly\_Edge f, Bidly\_Variable v, Bidly\_Boolean value )**

Function Bidly\_Managed\_Restrict calculates a restriction of Boolean function.

**Description**

This is not Coudert and Madre's restrict function (use `Bidly_Simplify` if you need that one).

**Side effects**

Original BDD is not changed. Implemented for OBDD, OBDDC, ZBDDC, and TZBDD. For OBDDs, recursive calls use optimization:  $F(a=x) == \text{NOT}(\text{NOT } F)(a=x)$ .

**More info**

Macro `Bidly_Restrict(f,v,value)` is defined for use with anonymous manager.

Definition at line 3114 of file `bidlyOp.c`.

### 5.6.2.13 `Bidly_Edge Bidly_Managed_Compose ( Bidly_Manager MNG, Bidly_Edge f, Bidly_Edge g, Bidly_Variable v )`

Function `Bidly_Managed_Compose` calculates a composition of two Boolean functions.

**Description****Side effects**

Original BDD is not changed. Implemented for OBDD, OBDDC, ZBDDC, and TZBDD. For OBDDs, recursive calls use optimization:  $F(a=G) == \text{NOT}(\text{NOT } F)(a=G)$ .

**More info**

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

Definition at line 3327 of file `bidlyOp.c`.

### 5.6.2.14 `Bidly_Edge Bidly_Managed_E ( Bidly_Manager MNG, Bidly_Edge f, Bidly_Variable v )`

Function `Bidly_Managed_E` calculates an existential quantification of Boolean function.

**Description****Side effects**

Original BDD is not changed. Implemented for OBDD, OBDDC, ZBDDC, and TZBDD. Be careful:  $\text{ExA } F \neq \text{NOT}(\text{ExA } \text{NOT } F)$ . Counterexample:  $\text{Exb } (\text{AND } (\text{NOT } a) b c)$ .

**More info**

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

Definition at line 3551 of file `bidlyOp.c`.

#### 5.6.2.15 **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

##### Side effects

Original BDD is not changed. Implemented for OBDDC. Prototyped for OBDD. Prototyped for ZBDDC and TZBDD. Be careful:  $Ax A F \neq \text{NOT}(Ax A (\text{NOT } F))$ . Counterexample:  $Axb (\text{AND } (\text{NOT } a) b c)$ .

##### More info

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

Definition at line 3752 of file `biddyOp.c`.

#### 5.6.2.16 **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.

##### Description

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

##### Side effects

Prototyped for OBDDs (via  $xA$ , calculating full universal quantification 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 3844 of file `biddyOp.c`.

#### 5.6.2.17 **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. Implemented for OBDD, OBDDC, ZBDDC, and TZBDD.



**More info**

Macro `Bidly_ExistAbstract(f,cube)` is defined for use with anonymous manager.

Definition at line 3907 of file `bidlyOp.c`.

**5.6.2.18 Bidly\_Edge Bidly\_Managed\_UnivAbstract ( Bidly\_Manager *MNG*, Bidly\_Edge *f*, Bidly\_Edge *cube* )**

Function `Bidly_Managed_UnivAbstract` universally abstracts all the variables in `cube` from `f`.

**Description****Side effects**

Original BDD is not changed. Implemented for OBDDC. Prototyped for OBDD. Prototyped for ZBDDC and TZBDD.

**More info**

Macro `Bidly_UnivAbstract(f,cube)` is defined for use with anonymous manager.

Definition at line 4156 of file `bidlyOp.c`.

**5.6.2.19 Bidly\_Edge Bidly\_Managed\_AndAbstract ( Bidly\_Manager *MNG*, Bidly\_Edge *f*, Bidly\_Edge *g*, Bidly\_Edge *cube* )**

Function `Bidly_Managed_AndAbstract` calculates the AND of two BDDs and simultaneously (existentially) abstracts the variables in `cube`.

**Description****Side effects**

Original BDD is not changed. Implemented for OBDD, OBDDC, ZBDDC, and TZBDD.

**More info**

Macro `Bidly_AndAbstract(f,g,cube)` is defined for use with anonymous manager.

Definition at line 4248 of file `bidlyOp.c`.

**5.6.2.20 Bidly\_Edge Bidly\_Managed\_Constrain ( Bidly\_Manager *MNG*, Bidly\_Edge *f*, Bidly\_Edge *c* )**

Function `Bidly_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**

Original BDD is not changed. Implemented for OBDD and OBDDC. Cache table is not used.

**More info**

Macro [Biddy\\_Constrain\(f,c\)](#) is defined for use with anonymous manager.

Definition at line 4672 of file biddyOp.c.

**5.6.2.21 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.

**Description**

Coudert and Madre's restrict function tries to simplify function *f* by restricting it to the domain covered by function *c*. No checks are done to see if the result is actually smaller than the input.

**Side effects**

Original BDD is not changed. Implemented for OBDD and OBDDC. Cache table is not used.

**More info**

Macro [Biddy\\_Simplify\(f,c\)](#) is defined for use with anonymous manager.

Definition at line 4805 of file biddyOp.c.

**5.6.2.22 Biddy\_Edge Biddy\_Managed\_Support ( Biddy\_Manager *MNG*, Biddy\_Edge *f* )**

Function `Biddy_Managed_Support` calculates a product of all dependent variables (OBDD and TZBDD) or the combination set containing a subset which includes all dependent variables (ZBDD).

**Description**

Implemented for OBDD, OBDDC, ZBDDC, and TZBDD. For OBDDs, dependent variables are all variables existing in the graph. For ZBDDs and TZBDDs, this is not true.

**Side effects****More info**

Macro [Biddy\\_Support\(f\)](#) is defined for use with anonymous manager.

Definition at line 4943 of file biddyOp.c.

### 5.6.2.23 Biddy\_Edge Biddy\_Managed\_ReplaceByKeyword ( Biddy\_Manager *MNG*, Biddy\_Edge *f*, Biddy\_String *keyword* )

Function Biddy\_Managed\_ReplaceByKeyword calculates Boolean function with one or more variables replaced.

#### Description

Original BDD is not changed. Implemented for OBDD and OBDDC. Prototyped for ZBDDC and TZBDD (via And-↔Xor-Not-Restrict). The sets of current and new variables should be disjoint. Replacing is controlled by variable's values (which are edges!). Use Biddy\_Managed\_ResetVariablesValue and Biddy\_Managed\_SetVariableValue to prepare replacing. Parameter keyword is used to maintain cache table. If (keyword == NULL) then entries in the cache table from previous calculations are deleted.

#### Side effects

#### More info

Macro [Biddy\\_ReplaceByKeyword\(f,keyword\)](#) is defined for use with anonymous manager. Macros Biddy\_↔Managed\_Replace(MNG,f) and Biddy\_Replace(f) are variants with less effective cache table.

Definition at line 5120 of file bidyOp.c.

### 5.6.2.24 Biddy\_Edge Biddy\_Managed\_Change ( Biddy\_Manager *MNG*, Biddy\_Edge *f*, Biddy\_Variable *v* )

Function Biddy\_Managed\_Change change the form of the given variable (positive literal becomes negative and vice versa).

#### Description

#### Side effects

Original BDD is not changed. Implemented for OBDD, OBDDC, ZBDDC, and TZBDD. RC Cache is used with parameters (f,biddyNull,v).

#### More info

Macro [Biddy\\_Change\(\)](#) is defined for use with anonymous manager.

Definition at line 5379 of file bidyOp.c.

### 5.6.2.25 Biddy\_Edge Biddy\_Managed\_Subset ( Biddy\_Manager *MNG*, Biddy\_Edge *f*, Biddy\_Variable *v*, Biddy\_Boolean *value* )

Function Biddy\_Managed\_Subset calculates a division of Boolean function with a literal.

#### Description

For combination sets, this function coincides with Subset0 and Subset1.

**Side effects**

Original BDD is not changed. Implemented for OBDD, OBDDC, ZBDDC, and TZBDD. Cache table for AND is used.

**More info**

Macro [Bidly\\_Subset\(f,v,value\)](#) is defined for use with anonymous manager. Macros [Bidly\\_Managed\\_Subset0\(←MNG,f,v\)](#), [Bidly\\_Subset0\(f,v\)](#), [Bidly\\_Managed\\_Subset1\(MNG,f,v\)](#), and [Bidly\\_Subset1\(f,v\)](#) are defined for manipulation of combination sets.

Definition at line 5560 of file `bidlyOp.c`.

#### 5.6.2.26 **Bidly\_Edge Bidly\_Managed\_CreateMinterm ( Bidly\_Manager MNG, Bidly\_Edge support, long long unsigned int x )**

Function `Bidly_Managed_CreateMinterm` generates one minterm.

**Description**

The represented Boolean function depends on the variables given with parameter support whilst the parameter `n` determines the generated minterm.

**Side effects**

Implemented for OBDD, OBDDC, ZBDDC, and TZBDD.

**More info**

Macro [Bidly\\_CreateMinterm\(support,x\)](#) is defined for use with anonymous manager.

Definition at line 5804 of file `bidlyOp.c`.

#### 5.6.2.27 **Bidly\_Edge Bidly\_Managed\_CreateFunction ( Bidly\_Manager MNG, Bidly\_Edge support, long long unsigned int x )**

Function `Bidly_Managed_CreateFunction` generates one Boolean function.

**Description**

The represented Boolean function depends on the variables given with parameter support whilst the parameter `n` determines the generated function.

**Side effects**

Implemented for OBDD, OBDDC, ZBDDC, and TZBDD.

**More info**

Macro [Bidly\\_CreateFunction\(support,x\)](#) is defined for use with anonymous manager.

Definition at line 5882 of file `bidlyOp.c`.

#### 5.6.2.28 `Bidly_Edge Bidly_Managed_RandomFunction ( Bidly_Manager MNG, Bidly_Edge support, double r )`

Function `Bidly_Managed_RandomFunction` generates a random BDD.

##### Description

The represented Boolean function depends on the variables given with parameter `support` whilst the parameter `r` determines the ratio between the number of function's minterms and the number of all possible minterms. Parameter `support` is a product of positive variables.

##### Side effects

Implemented for OBDD, OBDDC, ZBDDC, and TzBDD. Parameter `r` must be a number from  $[0,1]$ . Otherwise, function returns `bidlyNull`.

##### More info

Macro `Bidly_RandomFunction(support,r)` is defined for use with anonymous manager.

Definition at line 5976 of file `bidlyOp.c`.

#### 5.6.2.29 `Bidly_Edge Bidly_Managed_RandomSet ( Bidly_Manager MNG, Bidly_Edge unit, double r )`

Function `Bidly_Managed_RandomSet` generates a random BDD.

##### Description

The represented set is a random combination set determined by the parameter `unit` whilst the parameter `r` determines the ratio between the number of set's subsets and the number of all possible subsets. Parameter `set` is a set containing only one subset which consist of all elements, i.e. it is a set  $\{x_1, x_2, \dots, x_n\}$ .

##### Side effects

Implemented for OBDD, OBDDC, ZBDDC, and TzBDD. Parameter `r` must be a number from  $[0,1]$ . Otherwise, function returns `bidlyNull`.

##### More info

Macro `Bidly_RandomSet(unit,r)` is defined for use with anonymous manager.

Definition at line 6107 of file `bidlyOp.c`.

## 5.7 `bidlyStat.c` File Reference

File `bidlyStat.c` contains statistical functions.

```
#include "bidlyInt.h"
```

## Functions

- unsigned int [Bidly\\_Managed\\_CountNodes](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f)  
*Function Bidly\_Managed\_CountNodes.*
- unsigned int [Bidly\\_MaxLevel](#) ([Bidly\\_Edge](#) f)  
*Function Bidly\_MaxLevel.*
- float [Bidly\\_AvgLevel](#) ([Bidly\\_Edge](#) f)  
*Function Bidly\_AvgLevel.*
- [Bidly\\_Variable Bidly\\_Managed\\_VariableTableNum](#) ([Bidly\\_Manager](#) MNG)  
*Function Bidly\_Managed\_VariableTableNum returns number of used variables.*
- unsigned int [Bidly\\_Managed\\_NodeTableSize](#) ([Bidly\\_Manager](#) MNG)  
*Function Bidly\_Managed\_NodeTableSize returns the size of node table.*
- unsigned int [Bidly\\_Managed\\_NodeTableBlockNumber](#) ([Bidly\\_Manager](#) MNG)  
*Function Bidly\_Managed\_NodeTableBlockNumber.*
- unsigned int [Bidly\\_Managed\\_NodeTableGenerated](#) ([Bidly\\_Manager](#) MNG)  
*Function Bidly\_Managed\_NodeTableGenerated.*
- unsigned int [Bidly\\_Managed\\_NodeTableMax](#) ([Bidly\\_Manager](#) MNG)  
*Function Bidly\_Managed\_NodeTableMax returns maximal (peek) number of nodes in node table.*
- unsigned int [Bidly\\_Managed\\_NodeTableNum](#) ([Bidly\\_Manager](#) MNG)  
*Function Bidly\_Managed\_NodeTableNum returns number of all nodes currently in node table.*
- unsigned int [Bidly\\_Managed\\_NodeTableNumVar](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Variable](#) v)  
*Function Bidly\_Managed\_NodeTableNumVar returns number of nodes with a given variable currently in node table.*
- unsigned int [Bidly\\_Managed\\_NodeTableResizeNumber](#) ([Bidly\\_Manager](#) MNG)  
*Function Bidly\_Managed\_NodeTableResizeNumber.*
- unsigned long long int [Bidly\\_Managed\\_NodeTableFoaNumber](#) ([Bidly\\_Manager](#) MNG)  
*Function Bidly\_Managed\_NodeTableFoaNumber.*
- unsigned long long int [Bidly\\_Managed\\_NodeTableFindNumber](#) ([Bidly\\_Manager](#) MNG)  
*Function Bidly\_Managed\_NodeTableFindNumber.*
- unsigned long long int [Bidly\\_Managed\\_NodeTableCompareNumber](#) ([Bidly\\_Manager](#) MNG)  
*Function Bidly\_Managed\_NodeTableCompareNumber.*
- unsigned long long int [Bidly\\_Managed\\_NodeTableAddNumber](#) ([Bidly\\_Manager](#) MNG)  
*Function Bidly\_Managed\_NodeTableAddNumber.*
- unsigned int [Bidly\\_Managed\\_NodeTableGCNumber](#) ([Bidly\\_Manager](#) MNG)  
*Function Bidly\_Managed\_NodeTableGCNumber.*
- unsigned int [Bidly\\_Managed\\_NodeTableGCTime](#) ([Bidly\\_Manager](#) MNG)  
*Function Bidly\_Managed\_NodeTableGCTime.*
- unsigned long long int [Bidly\\_Managed\\_NodeTableGCObsoleteNumber](#) ([Bidly\\_Manager](#) MNG)  
*Function Bidly\_Managed\_NodeTableGCObsoleteNumber.*
- unsigned int [Bidly\\_Managed\\_NodeTableSwapNumber](#) ([Bidly\\_Manager](#) MNG)  
*Function Bidly\_Managed\_NodeTableSwapNumber.*
- unsigned int [Bidly\\_Managed\\_NodeTableSiftingNumber](#) ([Bidly\\_Manager](#) MNG)  
*Function Bidly\_Managed\_NodeTableSiftingNumber.*
- unsigned int [Bidly\\_Managed\\_NodeTableDRTTime](#) ([Bidly\\_Manager](#) MNG)  
*Function Bidly\_Managed\_NodeTableDRTTime.*
- unsigned int [Bidly\\_Managed\\_NodeTableITENumber](#) ([Bidly\\_Manager](#) MNG)  
*Function Bidly\_Managed\_NodeTableITENumber.*
- unsigned long long int [Bidly\\_Managed\\_NodeTableITERecursiveNumber](#) ([Bidly\\_Manager](#) MNG)  
*Function Bidly\_Managed\_NodeTableITERecursiveNumber.*
- unsigned int [Bidly\\_Managed\\_NodeTableANDORNumber](#) ([Bidly\\_Manager](#) MNG)  
*Function Bidly\_Managed\_NodeTableANDORNumber.*
- unsigned long long int [Bidly\\_Managed\\_NodeTableANDORRecursiveNumber](#) ([Bidly\\_Manager](#) MNG)

- Function Bidly\_Managed\_NodeTableANDORRecursiveNumber.*

  - unsigned int [Bidly\\_Managed\\_NodeTableXORNumber](#) ([Bidly\\_Manager](#) MNG)

*Function Bidly\_Managed\_NodeTableXORNumber.*
- unsigned long long int [Bidly\\_Managed\\_NodeTableXORRecursiveNumber](#) ([Bidly\\_Manager](#) MNG)

*Function Bidly\_Managed\_NodeTableXORRecursiveNumber.*
- unsigned int [Bidly\\_Managed\\_FormulaTableNum](#) ([Bidly\\_Manager](#) MNG)

*Function Bidly\_Managed\_FormulaTableNum returns number of known formulae.*
- unsigned int [Bidly\\_Managed\\_ListUsed](#) ([Bidly\\_Manager](#) MNG)

*Function Bidly\_Managed\_ListUsed.*
- unsigned int [Bidly\\_Managed\\_ListMaxLength](#) ([Bidly\\_Manager](#) MNG)

*Function Bidly\_Managed\_ListMaxLength.*
- float [Bidly\\_Managed\\_ListAvgLength](#) ([Bidly\\_Manager](#) MNG)

*Function Bidly\_Managed\_ListAvgLength.*
- unsigned long long int [Bidly\\_Managed\\_OPCCacheSearch](#) ([Bidly\\_Manager](#) MNG)

*Function Bidly\_Managed\_OPCCacheSearch.*
- unsigned long long int [Bidly\\_Managed\\_OPCCacheFind](#) ([Bidly\\_Manager](#) MNG)

*Function Bidly\_Managed\_OPCCacheFind.*
- unsigned long long int [Bidly\\_Managed\\_OPCCacheInsert](#) ([Bidly\\_Manager](#) MNG)

*Function Bidly\_Managed\_OPCCacheInsert.*
- unsigned long long int [Bidly\\_Managed\\_OPCCacheOverwrite](#) ([Bidly\\_Manager](#) MNG)

*Function Bidly\_Managed\_OPCCacheOverwrite.*
- unsigned int [Bidly\\_Managed\\_CountNodesPlain](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f)

*Function Bidly\_Managed\_CountNodesPlain.*
- unsigned int [Bidly\\_Managed\\_DependentVariableNumber](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f)

*Function Bidly\_Managed\_DependentVariableNumber.*
- unsigned int [Bidly\\_Managed\\_CountComplementedEdges](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f)

*Function Bidly\_Managed\_CountComplementedEdges count the number of complemented edges.*
- unsigned long long int [Bidly\\_Managed\\_CountPaths](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f)

*Function Bidly\_Managed\_CountPaths count the number of 1-paths.*
- double [Bidly\\_Managed\\_CountMinterms](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f, unsigned int nvars)

*Function Bidly\_Managed\_CountMinterms.*
- double [Bidly\\_Managed\\_DensityOfFunction](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f, unsigned int nvars)

*Function Bidly\_Managed\_DensityOfFunction calculates the ratio of the number of on-set minterms to the number of all minterms.*
- double [Bidly\\_Managed\\_DensityOfBDD](#) ([Bidly\\_Manager](#) MNG, [Bidly\\_Edge](#) f, unsigned int nvars)

*Function Bidly\_Managed\_DensityOfBDD calculates the ratio of the number of on-set minterms to the number of nodes.*
- unsigned long long int [Bidly\\_Managed\\_ReadMemoryInUse](#) ([Bidly\\_Manager](#) MNG)

*Function Bidly\_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 [Bidly\\_Managed\\_PrintInfo](#) ([Bidly\\_Manager](#) MNG, FILE \*f)

*Function Bidly\_Managed\_PrintInfo prepare a file with stats.*

### 5.7.1 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, 0-sup-BDDs, and TZBDDs. A hash table is used for quick
search of nodes. Complement edges decreases the number of
nodes. An automatic garbage collection with a system age is
implemented. Variable swapping and sifting are implemented.]

FileName [biddyStat.c]
Revision [${Revision: 360 $}]
Date [${Date: 2017-12-16 09:23:48 +0100 (sob, 16 dec 2017) $}]
Authors [Robert Meolic (robert.meolic@um.si),
Ales Casar (ales@homemade.net)]
```

## Copyright

Copyright (C) 2006, 2017 UM FERi, Koroska cesta 46, 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](#)

## 5.7.2 Function Documentation

### 5.7.2.1 unsigned int Biddy\_Managed\_CountNodes ( Biddy\_Manager *MNG*, Biddy\_Edge *f* )

Function Biddy\_Managed\_CountNodes.

#### Description

Count number of nodes in a BDD.

#### Side effects

This function must be managed because node selection is used.

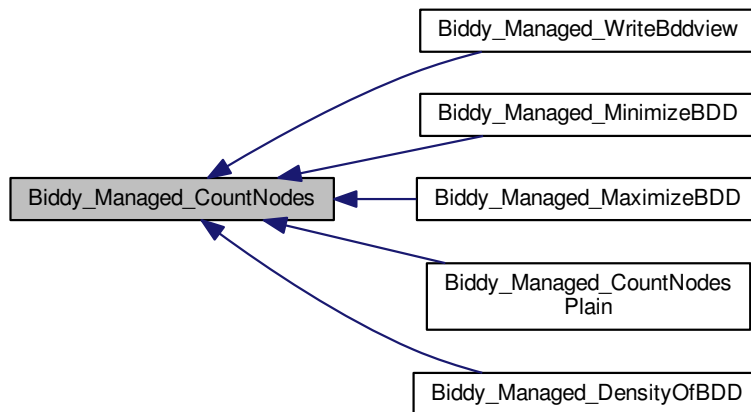


**More info**

Macro [Biddy\\_CountNodes\(f\)](#) is defined for use with anonymous manager.

Definition at line 85 of file biddyStat.c.

Here is the caller graph for this function:



### 5.7.2.2 unsigned int Biddy\_MaxLevel ( Biddy\_Edge f )

Function `Biddy_MaxLevel`.

**Description****Side effects****More info**

Macro [Biddy\\_Managed\\_MaxLevel\(f\)](#) is defined for user convenience.

Definition at line 164 of file biddyStat.c.

### 5.7.2.3 float Biddy\_AvgLevel ( Biddy\_Edge f )

Function `Biddy_AvgLevel`.

**Description****Side effects**

The result may not be compatible with your definition of Average Level for DAG. The result is especially problematic if there exists a node with two equal descendants (e.g for ZBDDs and TZBDDs).

**More info**

Macro [Bidly\\_Managed\\_AvgLevel\(f\)](#) is defined for user convenience.

Definition at line 196 of file biddyStat.c.

**5.7.2.4 Bidly\_Managed\_VariableTableNum ( Bidly\_Manager MNG )**

Function Bidly\_Managed\_VariableTableNum returns number of used variables.

**Description****Side effects**

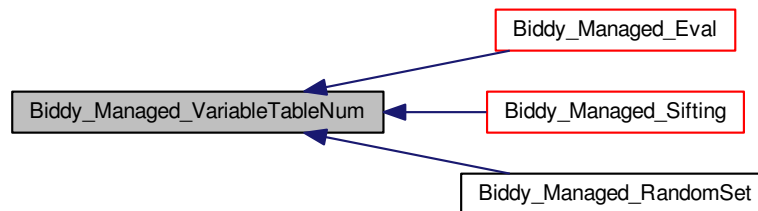
Variable '1' is included.

**More info**

Macro [Bidly\\_VariableTableNum\(\)](#) is defined for use with anonymous manager.

Definition at line 228 of file biddyStat.c.

Here is the caller graph for this function:

**5.7.2.5 unsigned int Bidly\_Managed\_NodeTableSize ( Bidly\_Manager MNG )**

Function Bidly\_Managed\_NodeTableSize returns the size of node table.

**Description****Side effects****More info**

Macro [Bidly\\_NodeTableSize\(\)](#) is defined for use with anonymous manager.

Definition at line 253 of file biddyStat.c.

#### 5.7.2.6 `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 283 of file `biddyStat.c`.

#### 5.7.2.7 `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 308 of file `biddyStat.c`.

#### 5.7.2.8 `unsigned int Biddy_Managed_NodeTableMax ( Biddy_Manager MNG )`

Function `Biddy_Managed_NodeTableMax` returns maximal (peek) number of nodes in node table.

**Description**

**Side effects**

**More info**

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

Definition at line 334 of file `biddyStat.c`.

#### 5.7.2.9 `unsigned int Biddy_Managed_NodeTableNum ( Biddy_Manager MNG )`

Function `Biddy_Managed_NodeTableNum` returns number of all nodes currently in node table.

Description

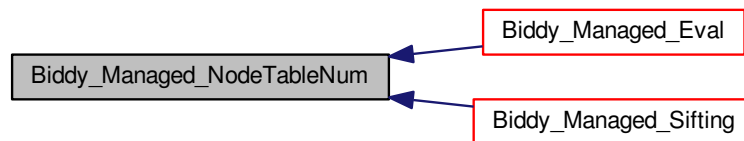
Side effects

More info

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

Definition at line 360 of file `biddyStat.c`.

Here is the caller graph for this function:



5.7.2.10 `unsigned int Biddy_Managed_NodeTableNumVar ( Biddy_Manager MNG, Biddy_Variable v )`

Function `Biddy_Managed_NodeTableNumVar` returns number of nodes with a given variable currently in node table.

Description

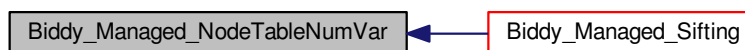
Side effects

More info

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

Definition at line 386 of file `biddyStat.c`.

Here is the caller graph for this function:



5.7.2.11 `unsigned int Biddy_Managed_NodeTableResizeNumber ( Biddy_Manager MNG )`

Function `Biddy_Managed_NodeTableResizeNumber`.

Description

Side effects

More info

Macro [Biddy\\_NodeTableResizeNumber\(\)](#) is defined for use with anonymous manager.

Definition at line 412 of file biddyStat.c.

#### 5.7.2.12 unsigned long long int Biddy\_Managed\_NodeTableFoaNumber ( Biddy\_Manager *MNG* )

Function Biddy\_Managed\_NodeTableFoaNumber.

Description

Side effects

More info

Macro [Biddy\\_NodeTableFoaNumber\(\)](#) is defined for use with anonymous manager.

Definition at line 438 of file biddyStat.c.

#### 5.7.2.13 unsigned long long int Biddy\_Managed\_NodeTableFindNumber ( Biddy\_Manager *MNG* )

Function Biddy\_Managed\_NodeTableFindNumber.

Description

Side effects

More info

Macro [Biddy\\_NodeTableFindNumber\(\)](#) is defined for use with anonymous manager.

Definition at line 470 of file biddyStat.c.

#### 5.7.2.14 unsigned long long int Biddy\_Managed\_NodeTableCompareNumber ( Biddy\_Manager *MNG* )

Function Biddy\_Managed\_NodeTableCompareNumber.

Description

Side effects

More info

Macro [Biddy\\_NodeTableCompareNumber\(\)](#) is defined for use with anonymous manager.

Definition at line 502 of file biddyStat.c.

#### 5.7.2.15 unsigned long long int Bidly\_Managed\_NodeTableAddNumber ( Bidly\_Manager MNG )

Function Bidly\_Managed\_NodeTableAddNumber.

Description

Side effects

More info

Macro [Bidly\\_NodeTableAddNumber\(\)](#) is defined for use with anonymous manager.

Definition at line 534 of file bidlyStat.c.

#### 5.7.2.16 unsigned int Bidly\_Managed\_NodeTableGCNumber ( Bidly\_Manager MNG )

Function Bidly\_Managed\_NodeTableGCNumber.

Description

Side effects

More info

Macro [Bidly\\_NodeTableGCNumber\(\)](#) is defined for use with anonymous manager.

Definition at line 565 of file bidlyStat.c.

#### 5.7.2.17 unsigned int Bidly\_Managed\_NodeTableGCTime ( Bidly\_Manager MNG )

Function Bidly\_Managed\_NodeTableGCTime.

Description

Side effects

More info

Macro [Bidly\\_NodeTableGCTime\(\)](#) is defined for use with anonymous manager.

Definition at line 590 of file bidlyStat.c.

#### 5.7.2.18 unsigned long long int Bidly\_Managed\_NodeTableGCObsoleteNumber ( Bidly\_Manager MNG )

Function Bidly\_Managed\_NodeTableGCObsoleteNumber.

**Description**

Return the number of nodes deleted by GC.

**Side effects**

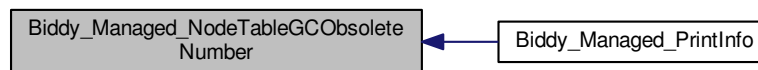
Obsolete nodes deleted by GC are counted only if Biddy is compiled using directive `BIDDYEXTENDEDSTATS_↔` YES.

**More info**

Macro [Biddy\\_NodeTableGCObsoleteNumber\(\)](#) is defined for use with anonymous manager.

Definition at line 619 of file `biddyStat.c`.

Here is the caller graph for this function:

**5.7.2.19 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 658 of file `biddyStat.c`.

**5.7.2.20 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 684 of file `biddyStat.c`.

#### 5.7.2.21 unsigned int Bidly\_Managed\_NodeTableDRTime ( Bidly\_Manager MNG )

Function Bidly\_Managed\_NodeTableDRTime.

Description

Side effects

More info

Macro [Bidly\\_NodeTableDRTime\(\)](#) is defined for use with anonymous manager.

Definition at line 709 of file bidlyStat.c.

#### 5.7.2.22 unsigned int Bidly\_Managed\_NodeTableITENumber ( Bidly\_Manager MNG )

Function Bidly\_Managed\_NodeTableITENumber.

Description

Side effects

More info

Macro [Bidly\\_NodeTableITENumber\(\)](#) is defined for use with anonymous manager.

Definition at line 734 of file bidlyStat.c.

#### 5.7.2.23 unsigned long long int Bidly\_Managed\_NodeTableITERecursiveNumber ( Bidly\_Manager MNG )

Function Bidly\_Managed\_NodeTableITERecursiveNumber.

Description

Side effects

Recursive ITE calls are counted only if Bidly is compiled using directive BIDDYEXTENDEDSTATS\_YES.

More info

Macro [Bidly\\_NodeTableITERecursiveNumber\(\)](#) is defined for use with anonymous manager.

Definition at line 762 of file bidlyStat.c.

#### 5.7.2.24 unsigned int Bidly\_Managed\_NodeTableANDORNumber ( Bidly\_Manager MNG )

Function Bidly\_Managed\_NodeTableANDORNumber.



Description

Side effects

More info

Macro [Biddy\\_NodeTableANDORNumber\(\)](#) is defined for use with anonymous manager.

Definition at line 794 of file biddyStat.c.

#### 5.7.2.25 unsigned long long int Biddy\_Managed\_NodeTableANDORRecursiveNumber ( Biddy\_Manager MNG )

Function Biddy\_Managed\_NodeTableANDORRecursiveNumber.

Description

Side effects

Recursive AND/OR calls are counted only if Biddy is compiled using directive BIDDYEXTENDEDSTATS\_YES.

More info

Macro [Biddy\\_NodeTableANDORRecursiveNumber\(\)](#) is defined for use with anonymous manager.

Definition at line 822 of file biddyStat.c.

#### 5.7.2.26 unsigned int Biddy\_Managed\_NodeTableXORNumber ( Biddy\_Manager MNG )

Function Biddy\_Managed\_NodeTableXORNumber.

Description

Side effects

More info

Macro [Biddy\\_NodeTableXORNumber\(\)](#) is defined for use with anonymous manager.

Definition at line 853 of file biddyStat.c.

#### 5.7.2.27 unsigned long long int Biddy\_Managed\_NodeTableXORRecursiveNumber ( Biddy\_Manager MNG )

Function Biddy\_Managed\_NodeTableXORRecursiveNumber.

Description

Side effects

Recursive XOR calls are counted only if Biddy is compiled using directive BIDDYEXTENDEDSTATS\_YES.

**More info**

Macro [Bidly\\_NodeTableXORRecursiveNumber\(\)](#) is defined for use with anonymous manager.

Definition at line 881 of file biddyStat.c.

**5.7.2.28 unsigned int Bidly\_Managed\_FormulaTableNum ( Bidly\_Manager MNG )**

Function Bidly\_Managed\_FormulaTableNum returns number of known formulae.

**Description****Side effects**

Formulae '0' and '1' are included.

**More info**

Macro [Bidly\\_FormulaTableNum\(\)](#) is defined for use with anonymous manager.

Definition at line 913 of file biddyStat.c.

**5.7.2.29 unsigned int Bidly\_Managed\_ListUsed ( Bidly\_Manager MNG )**

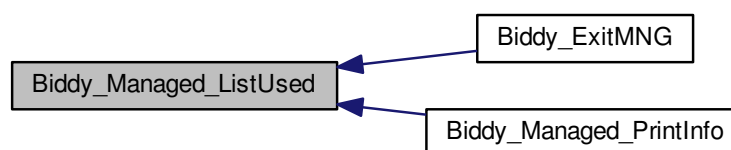
Function Bidly\_Managed\_ListUsed.

**Description****Side effects****More info**

Macro [Bidly\\_ListUsed\(\)](#) is defined for use with anonymous manager.

Definition at line 938 of file biddyStat.c.

Here is the caller graph for this function:



### 5.7.2.30 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 971 of file biddyStat.c.

Here is the caller graph for this function:



### 5.7.2.31 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 1035 of file biddyStat.c.

Here is the caller graph for this function:



### 5.7.2.32 unsigned long long int Bidly\_Managed\_OPCCacheSearch ( Bidly\_Manager MNG )

Function Bidly\_Managed\_OPCCacheSearch.

**Description**

**Side effects**

**More info**

Macro [Bidly\\_OPCCacheSearch\(\)](#) is defined for use with anonymous manager.

Definition at line 1077 of file bidlyStat.c.

### 5.7.2.33 unsigned long long int Bidly\_Managed\_OPCCacheFind ( Bidly\_Manager MNG )

Function Bidly\_Managed\_OPCCacheFind.

**Description**

**Side effects**

**More info**

Macro [Bidly\\_OPCCacheFind\(\)](#) is defined for use with anonymous manager.

Definition at line 1102 of file bidlyStat.c.

### 5.7.2.34 unsigned long long int Bidly\_Managed\_OPCCacheInsert ( Bidly\_Manager MNG )

Function Bidly\_Managed\_OPCCacheInsert.

**Description**

**Side effects**

**More info**

Macro [Bidly\\_OPCCacheInsert\(\)](#) is defined for use with anonymous manager.

Definition at line 1127 of file bidlyStat.c.

### 5.7.2.35 unsigned long long int Bidly\_Managed\_OPCCacheOverwrite ( Bidly\_Manager MNG )

Function Bidly\_Managed\_OPCCacheOverwrite.

**Description****Side effects****More info**

Macro [Biddy\\_OPCacheOverwrite\(\)](#) is defined for use with anonymous manager.

Definition at line 1158 of file biddyStat.c.

**5.7.2.36 unsigned int Biddy\_Managed\_CountNodesPlain ( Biddy\_Manager *MNG*, Biddy\_Edge *f* )**

Function Biddy\_Managed\_CountNodesPlain.

**Description**

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

**Side effects****More info**

Macro Biddy\_Managed\_CountNodesPlain(*f*) is defined for use with anonymous manager.

Definition at line 1191 of file biddyStat.c.

**5.7.2.37 unsigned int Biddy\_Managed\_DependentVariableNumber ( Biddy\_Manager *MNG*, Biddy\_Edge *f* )**

Function Biddy\_Managed\_DependentVariableNumber.

**Description**

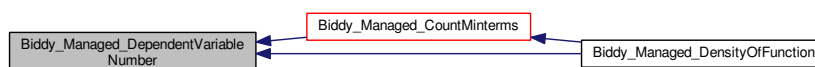
Count number of dependent variables. For OBDD, the number of dependent variables is the same as the number of variables in the graph. For ZBDD and TZBDD, this is not true.

**Side effects****More info**

Macro [Biddy\\_DependentVariableNumber\(\*f\*\)](#) is defined for use with anonymous manager.

Definition at line 1275 of file biddyStat.c.

Here is the caller graph for this function:



### 5.7.2.38 unsigned int Bidly\_Managed\_CountComplementedEdges ( Bidly\_Manager *MNG*, Bidly\_Edge *f* )

Function Bidly\_Managed\_CountComplementedEdges count the number of complemented edges.

#### Description

Count number of complemented edges in a given BDD.

#### Side effects

Terminal 0 is represented by complemented edge to terminal 1 with all BDD types but this edge is counted as a complemented one only if complemented edges are explicitly used.

#### More info

Macro Bidly\_Managed\_CountComplementedEdges(*f*) is defined for use with anonymous manager.

Definition at line 1389 of file bidlyStat.c.

### 5.7.2.39 unsigned long long int Bidly\_Managed\_CountPaths ( Bidly\_Manager *MNG*, Bidly\_Edge *f* )

Function Bidly\_Managed\_CountPaths count the number of 1-paths.

#### Description

#### Side effects

Implemented for OBDD, ZBDD, and TZBDD. TO DO: implement this using GNU Multiple Precision Arithmetic Library (GMP).

#### More info

Macro Bidly\_CountPaths(*f*) is defined for use with anonymous manager.

Definition at line 1452 of file bidlyStat.c.

### 5.7.2.40 double Bidly\_Managed\_CountMinterms ( Bidly\_Manager *MNG*, Bidly\_Edge *f*, unsigned int *nvars* )

Function Bidly\_Managed\_CountMinterms.

#### Description

Parameter *nvars* is a user-defined number of dependent variables. If *nvars* == 0 then number of variables existing in the graph is used. For combination sets, this function coincides with combination counting.

#### Side effects

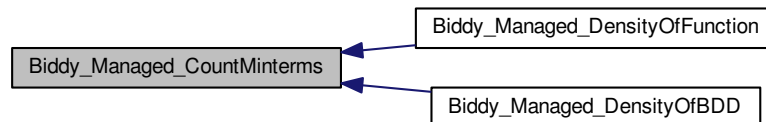
We are using GNU Multiple Precision Arithmetic Library (GMP). For ZBDDs, this function coincides with the 1-path count.

**More info**

Macro [Biddy\\_CountMinterms\(f,nvars\)](#) is defined for use with anonymous manager. Macros [Biddy\\_Managed\\_CountCombination\(MNG,f,nvars\)](#) and [Biddy\\_CountCombinations\(f,nvars\)](#) are defined for use with combination sets.

Definition at line 1520 of file biddyStat.c.

Here is the caller graph for this function:



#### 5.7.2.41 `double Biddy_Managed_DensityOfFunction ( Biddy_Manager MNG, Biddy_Edge f, unsigned int nvars )`

Function `Biddy_Managed_DensityOfFunction` calculates the ratio of the number of on-set minterms to the number of all minterms.

**Description**

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

**Side effects****More info**

Macro [Biddy\\_DensityOfFunction\(f,nvars\)](#) is defined for use with anonymous manager.

Definition at line 1663 of file biddyStat.c.

#### 5.7.2.42 `double Biddy_Managed_DensityOfBDD ( Biddy_Manager MNG, Biddy_Edge f, unsigned int nvars )`

Function `Biddy_Managed_DensityOfBDD` calculates the ratio of the number of on-set minterms to the number of nodes.

**Description**

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

**Side effects****More info**

Macro [Biddy\\_DensityOfBDD\(f,nvars\)](#) is defined for use with anonymous manager.

Definition at line 1735 of file biddyStat.c.

#### 5.7.2.43 unsigned long long int Bidly\_Managed\_ReadMemoryInUse ( Bidly\_Manager *MNG* )

Function Bidly\_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

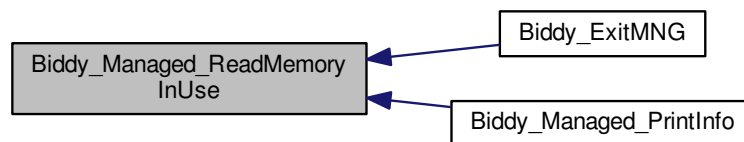
##### Side effects

##### More info

Macro [Bidly\\_ReadMemoryInUse\(\)](#) is defined for use with anonymous manager.

Definition at line 1828 of file bidlyStat.c.

Here is the caller graph for this function:



#### 5.7.2.44 void Bidly\_Managed\_PrintInfo ( Bidly\_Manager *MNG*, FILE \* *f* )

Function Bidly\_Managed\_PrintInfo prepare a file with stats.

##### Description

##### Side effects

##### More info

Macro [Bidly\\_PrintInfo\(f\)](#) is defined for use with anonymous manager.

Definition at line 1914 of file bidlyStat.c.



# Index

[biddy.h](#), 19

- [Biddy\\_AddCache](#), 32
- [Biddy\\_AddElementByName](#), 30
- [Biddy\\_AddFormula](#), 32
- [Biddy\\_AddVariableAbove](#), 31
- [Biddy\\_AddVariableBelow](#), 30
- [Biddy\\_AddVariableByName](#), 30
- [Biddy\\_And](#), 34
- [Biddy\\_AndAbstract](#), 37
- [Biddy\\_Boolean](#), 46
- [Biddy\\_Cache](#), 46
- [Biddy\\_Change](#), 37
- [Biddy\\_ChangeVariableName](#), 30
- [Biddy\\_Clean](#), 31
- [Biddy\\_ClearMark](#), 24
- [Biddy\\_ClearTag](#), 25
- [Biddy\\_ClearVariablesData](#), 29
- [Biddy\\_Complement](#), 24
- [Biddy\\_Compose](#), 36
- [Biddy\\_Constrain](#), 37
- [Biddy\\_Copy](#), 34
- [Biddy\\_CopyFormula](#), 34
- [Biddy\\_CountComplementedEdges](#), 43
- [Biddy\\_CountMinterms](#), 43
- [Biddy\\_CountNodes](#), 38
- [Biddy\\_CountNodesPlain](#), 43
- [Biddy\\_CountPaths](#), 43
- [Biddy\\_CreateFunction](#), 38
- [Biddy\\_CreateMinterm](#), 38
- [Biddy\\_DeleteFormula](#), 33
- [Biddy\\_DeleteIthFormula](#), 33
- [Biddy\\_DensityOfBDD](#), 43
- [Biddy\\_DensityOfFunction](#), 43
- [Biddy\\_DependentVariableNumber](#), 43
- [Biddy\\_DeselectAll](#), 27
- [Biddy\\_DeselectNode](#), 26
- [Biddy\\_Edge](#), 46
- [Biddy\\_Eval](#), 34
- [Biddy\\_Eval0](#), 44
- [Biddy\\_Eval1x](#), 44
- [Biddy\\_Eval2](#), 44
- [Biddy\\_ExistAbstract](#), 36
- [Biddy\\_Exit](#), 25
- [Biddy\\_FindFormula](#), 32
- [Biddy\\_FoaVariable](#), 30
- [Biddy\\_FormulaTableNum](#), 42
- [Biddy\\_GCFunction](#), 46
- [Biddy\\_GC](#), 31
- [Biddy\\_GetBaseSet](#), 27
- [Biddy\\_GetConstantOne](#), 27
- [Biddy\\_GetConstantZero](#), 27
- [Biddy\\_GetElementEdge](#), 28
- [Biddy\\_GetIthFormula](#), 33
- [Biddy\\_GetIthFormulaName](#), 33
- [Biddy\\_GetIthVariable](#), 28
- [Biddy\\_GetLowestVariable](#), 28
- [Biddy\\_GetManagerName](#), 25
- [Biddy\\_GetManagerType](#), 25
- [Biddy\\_GetMark](#), 24
- [Biddy\\_GetNextVariable](#), 28
- [Biddy\\_GetPrevVariable](#), 28
- [Biddy\\_GetTag](#), 25
- [Biddy\\_GetTerminal](#), 27
- [Biddy\\_GetTopVariableChar](#), 29
- [Biddy\\_GetTopVariableEdge](#), 28
- [Biddy\\_GetTopVariableName](#), 29
- [Biddy\\_GetVariable](#), 27
- [Biddy\\_GetVariableData](#), 29
- [Biddy\\_GetVariableEdge](#), 28
- [Biddy\\_GetVariableName](#), 28
- [Biddy\\_GetVariableValue](#), 29
- [Biddy\\_Gt](#), 35
- [Biddy\\_ITE](#), 34
- [Biddy\\_IncTag](#), 31
- [Biddy\\_Init](#), 25
- [Biddy\\_Inv](#), 24
- [Biddy\\_InvCond](#), 24
- [Biddy\\_InvertMark](#), 24
- [Biddy\\_IsEqv](#), 26
- [Biddy\\_IsEqvPointer](#), 23
- [Biddy\\_IsHighest](#), 30
- [Biddy\\_IsLeq](#), 36
- [Biddy\\_IsLowest](#), 30
- [Biddy\\_IsNull](#), 23
- [Biddy\\_IsOK](#), 31
- [Biddy\\_IsSelected](#), 27
- [Biddy\\_IsSmaller](#), 30
- [Biddy\\_IsTerminal](#), 23
- [Biddy\\_IsVariableDependent](#), 36
- [Biddy\\_Leq](#), 35
- [Biddy\\_ListAvgLength](#), 42
- [Biddy\\_ListMaxLength](#), 42
- [Biddy\\_ListUsed](#), 42
- [Biddy\\_LookupFunction](#), 47
- [Biddy\\_Managed\\_AvgLevel](#), 38
- [Biddy\\_Managed\\_GetElse](#), 26
- [Biddy\\_Managed\\_GetThen](#), 26
- [Biddy\\_Managed\\_GetTopVariable](#), 26

Bidly\_Managed\_MaxLevel, 38  
 Bidly\_Manager, 46  
 Bidly\_MaximizeBDD, 34  
 Bidly\_MinimizeBDD, 33  
 Bidly\_Nand, 35  
 Bidly\_NodeTableANDORNumber, 41  
 Bidly\_NodeTableANDORRecursiveNumber, 41  
 Bidly\_NodeTableAddNumber, 40  
 Bidly\_NodeTableBlockNumber, 39  
 Bidly\_NodeTableCompareNumber, 40  
 Bidly\_NodeTableDRTTime, 41  
 Bidly\_NodeTableFindNumber, 40  
 Bidly\_NodeTableFoaNumber, 39  
 Bidly\_NodeTableGCNumber, 40  
 Bidly\_NodeTableGCObsoleteNumber, 40  
 Bidly\_NodeTableGCTime, 40  
 Bidly\_NodeTableGenerated, 39  
 Bidly\_NodeTableITENumber, 41  
 Bidly\_NodeTableITERRecursiveNumber, 41  
 Bidly\_NodeTableMax, 39  
 Bidly\_NodeTableNum, 39  
 Bidly\_NodeTableNumVar, 39  
 Bidly\_NodeTableResizeNumber, 39  
 Bidly\_NodeTableSiftingNumber, 40  
 Bidly\_NodeTableSize, 39  
 Bidly\_NodeTableSwapNumber, 40  
 Bidly\_NodeTableXORNumber, 41  
 Bidly\_NodeTableXORRecursiveNumber, 41  
 Bidly\_Nor, 35  
 Bidly\_Not, 34  
 Bidly\_OPCacheFind, 42  
 Bidly\_OPCacheInsert, 42  
 Bidly\_OPCacheOverwrite, 43  
 Bidly\_OPCacheSearch, 42  
 Bidly\_Or, 35  
 Bidly\_PrintInfo, 44  
 Bidly\_PrintfBDD, 44  
 Bidly\_PrintfSOP, 45  
 Bidly\_PrintfTable, 45  
 Bidly\_Purge, 32  
 Bidly\_PurgeAndReorder, 32  
 Bidly\_RandomFunction, 38  
 Bidly\_RandomSet, 38  
 Bidly\_ReadMemoryInUse, 44  
 Bidly\_ReadVerilogFile, 44  
 Bidly\_Refresh, 32  
 Bidly\_Regular, 24  
 Bidly\_ReplaceByKeyword, 37  
 Bidly\_ResetVariablesValue, 29  
 Bidly\_Restrict, 36  
 Bidly\_SelectFunction, 27  
 Bidly\_SelectNode, 26  
 Bidly\_SetManagerParameters, 26  
 Bidly\_SetMark, 24  
 Bidly\_SetTag, 25  
 Bidly\_SetVariableData, 29  
 Bidly\_SetVariableValue, 29  
 Bidly\_Sifting, 33  
 Bidly\_Simplify, 37  
 Bidly\_String, 46  
 Bidly\_Subset, 37  
 Bidly\_Support, 37  
 Bidly\_SwapWithHigher, 33  
 Bidly\_SwapWithLower, 33  
 Bidly\_TaggedFoaNode, 31  
 Bidly\_TransferMark, 31  
 Bidly\_UnivAbstract, 36  
 Bidly\_Untagged, 25  
 Bidly\_Variable, 46  
 Bidly\_VariableTableNum, 38  
 Bidly\_WriteBDD, 44  
 Bidly\_WriteBddview, 45  
 Bidly\_WriteDot, 45  
 Bidly\_WriteSOP, 45  
 Bidly\_WriteTable, 45  
 Bidly\_Xnor, 35  
 Bidly\_Xor, 35  
 Bidly\_A, 36  
 Bidly\_E, 36  
 Bidly\_About  
     bidlyMain.c, 59  
     bidlyMainLegacy.c, 98  
 Bidly\_AddCache  
     bidly.h, 32  
 Bidly\_AddElementByName  
     bidly.h, 30  
 Bidly\_AddFormula  
     bidly.h, 32  
 Bidly\_AddVariableAbove  
     bidly.h, 31  
 Bidly\_AddVariableBelow  
     bidly.h, 30  
 Bidly\_AddVariableByName  
     bidly.h, 30  
 Bidly\_And  
     bidly.h, 34  
 Bidly\_AndAbstract  
     bidly.h, 37  
 Bidly\_AvgLevel  
     bidlyStat.c, 151  
 Bidly\_Boolean  
     bidly.h, 46  
 Bidly\_Cache  
     bidly.h, 46  
 Bidly\_Change  
     bidly.h, 37  
 Bidly\_ChangeVariableName  
     bidly.h, 30  
 Bidly\_Clean  
     bidly.h, 31  
 Bidly\_ClearMark  
     bidly.h, 24  
 Bidly\_ClearTag  
     bidly.h, 25  
 Bidly\_ClearVariablesData  
     bidly.h, 29

Bidddy\_Complement  
bidddy.h, 24

Bidddy\_Compose  
bidddy.h, 36

Bidddy\_Constrain  
bidddy.h, 37

Bidddy\_Copy  
bidddy.h, 34

Bidddy\_CopyFormula  
bidddy.h, 34

Bidddy\_CountComplementedEdges  
bidddy.h, 43

Bidddy\_CountMinterms  
bidddy.h, 43

Bidddy\_CountNodes  
bidddy.h, 38

Bidddy\_CountNodesPlain  
bidddy.h, 43

Bidddy\_CountPaths  
bidddy.h, 43

Bidddy\_CreateFunction  
bidddy.h, 38

Bidddy\_CreateMinterm  
bidddy.h, 38

Bidddy\_DeleteFormula  
bidddy.h, 33

Bidddy\_DeletelthFormula  
bidddy.h, 33

Bidddy\_DensityOfBDD  
bidddy.h, 43

Bidddy\_DensityOfFunction  
bidddy.h, 43

Bidddy\_DependentVariableNumber  
bidddy.h, 43

Bidddy\_DeselectAll  
bidddy.h, 27

Bidddy\_DeselectNode  
bidddy.h, 26

Bidddy\_Edge  
bidddy.h, 46

Bidddy\_Eval  
bidddy.h, 34

Bidddy\_Eval0  
bidddy.h, 44

Bidddy\_Eval1x  
bidddy.h, 44

Bidddy\_Eval2  
bidddy.h, 44

Bidddy\_ExistAbstract  
bidddy.h, 36

Bidddy\_Exit  
bidddy.h, 25

Bidddy\_ExitMNG  
bidddyMain.c, 58  
bidddyMainLegacy.c, 98

Bidddy\_FindFormula  
bidddy.h, 32

Bidddy\_FoaVariable  
bidddy.h, 30

Bidddy\_FormulaTableNum  
bidddy.h, 42

Bidddy\_GCFunction  
bidddy.h, 46

Bidddy\_GC  
bidddy.h, 31

Bidddy\_GetBaseSet  
bidddy.h, 27

Bidddy\_GetConstantOne  
bidddy.h, 27

Bidddy\_GetConstantZero  
bidddy.h, 27

Bidddy\_GetElementEdge  
bidddy.h, 28

Bidddy\_GetElse  
bidddyMain.c, 61  
bidddyMainLegacy.c, 99

Bidddy\_GetlthFormula  
bidddy.h, 33

Bidddy\_GetlthFormulaName  
bidddy.h, 33

Bidddy\_GetlthVariable  
bidddy.h, 28

Bidddy\_GetLowestVariable  
bidddy.h, 28

Bidddy\_GetManagerName  
bidddy.h, 25

Bidddy\_GetManagerType  
bidddy.h, 25

Bidddy\_GetMark  
bidddy.h, 24

Bidddy\_GetNextVariable  
bidddy.h, 28

Bidddy\_GetPrevVariable  
bidddy.h, 28

Bidddy\_GetTag  
bidddy.h, 25

Bidddy\_GetTerminal  
bidddy.h, 27

Bidddy\_GetThen  
bidddyMain.c, 61  
bidddyMainLegacy.c, 99

Bidddy\_GetTopVariable  
bidddyMain.c, 62  
bidddyMainLegacy.c, 100

Bidddy\_GetTopVariableChar  
bidddy.h, 29

Bidddy\_GetTopVariableEdge  
bidddy.h, 28

Bidddy\_GetTopVariableName  
bidddy.h, 29

Bidddy\_GetVariable  
bidddy.h, 27

Bidddy\_GetVariableData  
bidddy.h, 29

Bidddy\_GetVariableEdge  
bidddy.h, 28

- Bidly\_GetVariableName  
bidly.h, 28
- Bidly\_GetVariableValue  
bidly.h, 29
- Bidly\_Gt  
bidly.h, 35
- Bidly\_ITE  
bidly.h, 34
- Bidly\_IncTag  
bidly.h, 31
- Bidly\_Init  
bidly.h, 25
- Bidly\_InitMNG  
bidlyMain.c, 58  
bidlyMainLegacy.c, 97
- Bidly\_Inv  
bidly.h, 24
- Bidly\_InvCond  
bidly.h, 24
- Bidly\_InvertMark  
bidly.h, 24
- Bidly\_IsEqv  
bidly.h, 26
- Bidly\_IsEqvPointer  
bidly.h, 23
- Bidly\_IsHighest  
bidly.h, 30
- Bidly\_IsLeq  
bidly.h, 36
- Bidly\_IsLowest  
bidly.h, 30
- Bidly\_IsNull  
bidly.h, 23
- Bidly\_IsOK  
bidly.h, 31
- Bidly\_IsSelected  
bidly.h, 27
- Bidly\_IsSmaller  
bidly.h, 30
- Bidly\_IsTerminal  
bidly.h, 23
- Bidly\_IsVariableDependent  
bidly.h, 36
- Bidly\_Leq  
bidly.h, 35
- Bidly\_ListAvgLength  
bidly.h, 42
- Bidly\_ListMaxLength  
bidly.h, 42
- Bidly\_ListUsed  
bidly.h, 42
- Bidly\_LookupFunction  
bidly.h, 47
- Bidly\_Managed\_AddCache  
bidlyMain.c, 85  
bidlyMainLegacy.c, 128
- Bidly\_Managed\_AddElementByName  
bidlyMain.c, 78  
bidlyMainLegacy.c, 113
- Bidly\_Managed\_AddFormula  
bidlyMain.c, 86  
bidlyMainLegacy.c, 129
- Bidly\_Managed\_AddVariableAbove  
bidlyMain.c, 79  
bidlyMainLegacy.c, 114
- Bidly\_Managed\_AddVariableBelow  
bidlyMain.c, 79  
bidlyMainLegacy.c, 113
- Bidly\_Managed\_AddVariableByName  
bidlyMain.c, 78  
bidlyMainLegacy.c, 112
- Bidly\_Managed\_And  
bidlyMainLegacy.c, 116  
bidlyOp.c, 137
- Bidly\_Managed\_AndAbstract  
bidlyMainLegacy.c, 123  
bidlyOp.c, 143
- Bidly\_Managed\_AvgLevel  
bidly.h, 38
- Bidly\_Managed\_Change  
bidlyMainLegacy.c, 125  
bidlyOp.c, 145
- Bidly\_Managed\_ChangeVariableName  
bidlyMain.c, 77
- Bidly\_Managed\_Clean  
bidlyMain.c, 83  
bidlyMainLegacy.c, 126
- Bidly\_Managed\_ClearVariablesData  
bidlyMain.c, 73
- Bidly\_Managed\_Compose  
bidlyMainLegacy.c, 120  
bidlyOp.c, 141
- Bidly\_Managed\_Constrain  
bidlyMainLegacy.c, 123  
bidlyOp.c, 143
- Bidly\_Managed\_CountComplementedEdges  
bidlyStat.c, 163
- Bidly\_Managed\_CountMinterms  
bidlyStat.c, 164
- Bidly\_Managed\_CountNodes  
bidlyStat.c, 150
- Bidly\_Managed\_CountNodesPlain  
bidlyStat.c, 163
- Bidly\_Managed\_CountPaths  
bidlyStat.c, 164
- Bidly\_Managed\_CreateFunction  
bidlyOp.c, 146
- Bidly\_Managed\_CreateMinterm  
bidlyOp.c, 146
- Bidly\_Managed\_DeleteFormula  
bidlyMain.c, 87  
bidlyMainLegacy.c, 130
- Bidly\_Managed\_DeletelthFormula  
bidlyMain.c, 88  
bidlyMainLegacy.c, 130
- Bidly\_Managed\_DensityOfBDD

- bidlyStat.c, [165](#)
- Bidly\_Managed\_DensityOfFunction
  - bidlyStat.c, [165](#)
- Bidly\_Managed\_DependentVariableNumber
  - bidlyStat.c, [163](#)
- Bidly\_Managed\_DeselectAll
  - bidlyMain.c, [65](#)
  - bidlyMainLegacy.c, [103](#)
- Bidly\_Managed\_DeselectNode
  - bidlyMain.c, [63](#)
  - bidlyMainLegacy.c, [101](#)
- Bidly\_Managed\_Eval0
  - bidlyInOut.c, [48](#)
- Bidly\_Managed\_Eval1x
  - bidlyInOut.c, [49](#)
- Bidly\_Managed\_Eval2
  - bidlyInOut.c, [49](#)
- Bidly\_Managed\_ExistAbstract
  - bidlyMainLegacy.c, [122](#)
  - bidlyOp.c, [142](#)
- Bidly\_Managed\_FindFormula
  - bidlyMain.c, [87](#)
  - bidlyMainLegacy.c, [129](#)
- Bidly\_Managed\_FoaVariable
  - bidlyMain.c, [76](#)
  - bidlyMainLegacy.c, [111](#)
- Bidly\_Managed\_FormulaTableNum
  - bidlyStat.c, [160](#)
- Bidly\_Managed\_GC
  - bidlyMain.c, [82](#)
  - bidlyMainLegacy.c, [126](#)
- Bidly\_Managed\_GetBaseSet
  - bidlyMain.c, [66](#)
  - bidlyMainLegacy.c, [105](#)
- Bidly\_Managed\_GetConstantOne
  - bidlyMain.c, [66](#)
  - bidlyMainLegacy.c, [104](#)
- Bidly\_Managed\_GetConstantZero
  - bidlyMain.c, [66](#)
  - bidlyMainLegacy.c, [104](#)
- Bidly\_Managed\_GetElementEdge
  - bidlyMain.c, [70](#)
  - bidlyMainLegacy.c, [107](#)
- Bidly\_Managed\_GetElse
  - bidly.h, [26](#)
- Bidly\_Managed\_GetlthFormula
  - bidlyMain.c, [88](#)
  - bidlyMainLegacy.c, [131](#)
- Bidly\_Managed\_GetlthFormulaName
  - bidlyMain.c, [89](#)
  - bidlyMainLegacy.c, [131](#)
- Bidly\_Managed\_GetlthVariable
  - bidlyMain.c, [68](#)
- Bidly\_Managed\_GetLowestVariable
  - bidlyMain.c, [67](#)
- Bidly\_Managed\_GetManagerName
  - bidlyMain.c, [60](#)
- Bidly\_Managed\_GetManagerType
  - bidlyMain.c, [59](#)
- bidlyMainLegacy.c, [98](#)
- Bidly\_Managed\_GetNextVariable
  - bidlyMain.c, [69](#)
  - bidlyMainLegacy.c, [106](#)
- Bidly\_Managed\_GetPrevVariable
  - bidlyMain.c, [68](#)
  - bidlyMainLegacy.c, [106](#)
- Bidly\_Managed\_GetTerminal
  - bidlyMain.c, [65](#)
  - bidlyMainLegacy.c, [103](#)
- Bidly\_Managed\_GetThen
  - bidly.h, [26](#)
- Bidly\_Managed\_GetTopVariable
  - bidly.h, [26](#)
- Bidly\_Managed\_GetTopVariableChar
  - bidlyMain.c, [72](#)
  - bidlyMainLegacy.c, [109](#)
- Bidly\_Managed\_GetTopVariableEdge
  - bidlyMain.c, [71](#)
  - bidlyMainLegacy.c, [108](#)
- Bidly\_Managed\_GetTopVariableName
  - bidlyMain.c, [71](#)
  - bidlyMainLegacy.c, [108](#)
- Bidly\_Managed\_GetVariable
  - bidlyMain.c, [67](#)
  - bidlyMainLegacy.c, [105](#)
- Bidly\_Managed\_GetVariableData
  - bidlyMain.c, [74](#)
- Bidly\_Managed\_GetVariableEdge
  - bidlyMain.c, [69](#)
  - bidlyMainLegacy.c, [106](#)
- Bidly\_Managed\_GetVariableName
  - bidlyMain.c, [70](#)
  - bidlyMainLegacy.c, [107](#)
- Bidly\_Managed\_GetVariableValue
  - bidlyMain.c, [73](#)
- Bidly\_Managed\_Gt
  - bidlyMainLegacy.c, [119](#)
  - bidlyOp.c, [140](#)
- Bidly\_Managed\_ITE
  - bidlyMainLegacy.c, [116](#)
  - bidlyOp.c, [137](#)
- Bidly\_Managed\_IncTag
  - bidlyMain.c, [80](#)
  - bidlyMainLegacy.c, [115](#)
- Bidly\_Managed\_IsEqv
  - bidlyMain.c, [62](#)
  - bidlyMainLegacy.c, [100](#)
- Bidly\_Managed\_IsHighest
  - bidlyMain.c, [76](#)
- Bidly\_Managed\_IsLeq
  - bidlyMainLegacy.c, [119](#)
  - bidlyOp.c, [140](#)
- Bidly\_Managed\_IsLowest
  - bidlyMain.c, [75](#)
- Bidly\_Managed\_IsOK
  - bidlyMain.c, [82](#)

- bidlyMainLegacy.c, [125](#)
- Bidly\_Managed\_IsSelected
  - bidlyMain.c, [64](#)
  - bidlyMainLegacy.c, [102](#)
- Bidly\_Managed\_IsSmaller
  - bidlyMain.c, [75](#)
  - bidlyMainLegacy.c, [110](#)
- Bidly\_Managed\_IsVariableDependent
  - bidlyMainLegacy.c, [122](#)
  - bidlyOp.c, [142](#)
- Bidly\_Managed\_Leq
  - bidlyMainLegacy.c, [119](#)
  - bidlyOp.c, [139](#)
- Bidly\_Managed\_ListAvgLength
  - bidlyStat.c, [161](#)
- Bidly\_Managed\_ListMaxLength
  - bidlyStat.c, [160](#)
- Bidly\_Managed\_ListUsed
  - bidlyStat.c, [160](#)
- Bidly\_Managed\_MaxLevel
  - bidly.h, [38](#)
- Bidly\_Managed\_MaximizeBDD
  - bidlyMain.c, [92](#)
- Bidly\_Managed\_MinimizeBDD
  - bidlyMain.c, [91](#)
- Bidly\_Managed\_Nand
  - bidlyMainLegacy.c, [117](#)
  - bidlyOp.c, [138](#)
- Bidly\_Managed\_NodeTableANDORNumber
  - bidlyStat.c, [158](#)
- Bidly\_Managed\_NodeTableANDORRecursiveNumber
  - bidlyStat.c, [159](#)
- Bidly\_Managed\_NodeTableAddNumber
  - bidlyStat.c, [155](#)
- Bidly\_Managed\_NodeTableBlockNumber
  - bidlyStat.c, [152](#)
- Bidly\_Managed\_NodeTableCompareNumber
  - bidlyStat.c, [155](#)
- Bidly\_Managed\_NodeTableDRTime
  - bidlyStat.c, [157](#)
- Bidly\_Managed\_NodeTableFindNumber
  - bidlyStat.c, [155](#)
- Bidly\_Managed\_NodeTableFoaNumber
  - bidlyStat.c, [155](#)
- Bidly\_Managed\_NodeTableGCNumber
  - bidlyStat.c, [156](#)
- Bidly\_Managed\_NodeTableGCObsoleteNumber
  - bidlyStat.c, [156](#)
- Bidly\_Managed\_NodeTableGCTime
  - bidlyStat.c, [156](#)
- Bidly\_Managed\_NodeTableGenerated
  - bidlyStat.c, [153](#)
- Bidly\_Managed\_NodeTableITENumber
  - bidlyStat.c, [158](#)
- Bidly\_Managed\_NodeTableITERRecursiveNumber
  - bidlyStat.c, [158](#)
- Bidly\_Managed\_NodeTableMax
  - bidlyStat.c, [153](#)
- Bidly\_Managed\_NodeTableNum
  - bidlyStat.c, [153](#)
- Bidly\_Managed\_NodeTableNumVar
  - bidlyStat.c, [154](#)
- Bidly\_Managed\_NodeTableResizeNumber
  - bidlyStat.c, [154](#)
- Bidly\_Managed\_NodeTableSiftingNumber
  - bidlyStat.c, [157](#)
- Bidly\_Managed\_NodeTableSize
  - bidlyStat.c, [152](#)
- Bidly\_Managed\_NodeTableSwapNumber
  - bidlyStat.c, [157](#)
- Bidly\_Managed\_NodeTableXORNumber
  - bidlyStat.c, [159](#)
- Bidly\_Managed\_NodeTableXORRecursiveNumber
  - bidlyStat.c, [159](#)
- Bidly\_Managed\_Nor
  - bidlyMainLegacy.c, [118](#)
  - bidlyOp.c, [138](#)
- Bidly\_Managed\_Not
  - bidlyMainLegacy.c, [116](#)
  - bidlyOp.c, [136](#)
- Bidly\_Managed\_OPCacheFind
  - bidlyStat.c, [162](#)
- Bidly\_Managed\_OPCacheInsert
  - bidlyStat.c, [162](#)
- Bidly\_Managed\_OPCacheOverwrite
  - bidlyStat.c, [162](#)
- Bidly\_Managed\_OPCacheSearch
  - bidlyStat.c, [161](#)
- Bidly\_Managed\_Or
  - bidlyMainLegacy.c, [117](#)
  - bidlyOp.c, [137](#)
- Bidly\_Managed\_PrintInfo
  - bidlyStat.c, [166](#)
- Bidly\_Managed\_PrintfBDD
  - bidlyInOut.c, [50](#)
- Bidly\_Managed\_PrintfSOP
  - bidlyInOut.c, [51](#)
- Bidly\_Managed\_PrintfTable
  - bidlyInOut.c, [50](#)
- Bidly\_Managed\_Purge
  - bidlyMain.c, [84](#)
  - bidlyMainLegacy.c, [127](#)
- Bidly\_Managed\_PurgeAndReorder
  - bidlyMain.c, [84](#)
  - bidlyMainLegacy.c, [127](#)
- Bidly\_Managed\_Random
  - bidlyMainLegacy.c, [133](#)
- Bidly\_Managed\_RandomFunction
  - bidlyOp.c, [146](#)
- Bidly\_Managed\_RandomSet
  - bidlyMainLegacy.c, [134](#)
  - bidlyOp.c, [147](#)
- Bidly\_Managed\_ReadMemoryInUse
  - bidlyStat.c, [165](#)
- Bidly\_Managed\_ReadVerilogFile
  - bidlyInOut.c, [49](#)

- Bidly\_Managed\_Refresh
  - bidlyMain.c, [85](#)
  - bidlyMainLegacy.c, [128](#)
- Bidly\_Managed\_Replace
  - bidlyMainLegacy.c, [124](#)
- Bidly\_Managed\_ReplaceByKeyword
  - bidlyOp.c, [144](#)
- Bidly\_Managed\_ResetVariablesValue
  - bidlyMain.c, [72](#)
  - bidlyMainLegacy.c, [109](#)
- Bidly\_Managed\_Restrict
  - bidlyMainLegacy.c, [120](#)
  - bidlyOp.c, [140](#)
- Bidly\_Managed\_SelectFunction
  - bidlyMain.c, [64](#)
  - bidlyMainLegacy.c, [102](#)
- Bidly\_Managed\_SelectNode
  - bidlyMain.c, [62](#)
  - bidlyMainLegacy.c, [101](#)
- Bidly\_Managed\_SetManagerParameters
  - bidlyMain.c, [60](#)
  - bidlyMainLegacy.c, [99](#)
- Bidly\_Managed\_SetVariableData
  - bidlyMain.c, [74](#)
- Bidly\_Managed\_SetVariableValue
  - bidlyMain.c, [73](#)
  - bidlyMainLegacy.c, [110](#)
- Bidly\_Managed\_Sifting
  - bidlyMain.c, [90](#)
  - bidlyMainLegacy.c, [133](#)
- Bidly\_Managed\_Simplify
  - bidlyMainLegacy.c, [123](#)
  - bidlyOp.c, [144](#)
- Bidly\_Managed\_SubIntersect
  - bidlyMainLegacy.c, [120](#)
- Bidly\_Managed\_Subset
  - bidlyMainLegacy.c, [125](#)
  - bidlyOp.c, [145](#)
- Bidly\_Managed\_Support
  - bidlyMainLegacy.c, [124](#)
  - bidlyOp.c, [144](#)
- Bidly\_Managed\_SwapWithHigher
  - bidlyMain.c, [89](#)
  - bidlyMainLegacy.c, [132](#)
- Bidly\_Managed\_SwapWithLower
  - bidlyMain.c, [90](#)
  - bidlyMainLegacy.c, [132](#)
- Bidly\_Managed\_TaggedFoaNode
  - bidlyMain.c, [81](#)
  - bidlyMainLegacy.c, [115](#)
- Bidly\_Managed\_TransferMark
  - bidlyMain.c, [79](#)
  - bidlyMainLegacy.c, [114](#)
- Bidly\_Managed\_UnivAbstract
  - bidlyMainLegacy.c, [122](#)
  - bidlyOp.c, [143](#)
- Bidly\_Managed\_VariableTableNum
  - bidlyStat.c, [152](#)
- Bidly\_Managed\_WriteBDD
  - bidlyInOut.c, [50](#)
- Bidly\_Managed\_WriteBddview
  - bidlyInOut.c, [52](#)
- Bidly\_Managed\_WriteDot
  - bidlyInOut.c, [52](#)
- Bidly\_Managed\_WriteSOP
  - bidlyInOut.c, [51](#)
- Bidly\_Managed\_WriteTable
  - bidlyInOut.c, [51](#)
- Bidly\_Managed\_Xnor
  - bidlyMainLegacy.c, [118](#)
  - bidlyOp.c, [139](#)
- Bidly\_Managed\_Xor
  - bidlyMainLegacy.c, [118](#)
  - bidlyOp.c, [138](#)
- Bidly\_Managed\_A
  - bidlyMainLegacy.c, [121](#)
  - bidlyOp.c, [141](#)
- Bidly\_Managed\_E
  - bidlyMainLegacy.c, [121](#)
  - bidlyOp.c, [141](#)
- Bidly\_Manager
  - bidly.h, [46](#)
- Bidly\_MaxLevel
  - bidlyStat.c, [151](#)
- Bidly\_MaximizeBDD
  - bidly.h, [34](#)
- Bidly\_MinimizeBDD
  - bidly.h, [33](#)
- Bidly\_Nand
  - bidly.h, [35](#)
- Bidly\_NodeTableANDORNumber
  - bidly.h, [41](#)
- Bidly\_NodeTableANDORRecursiveNumber
  - bidly.h, [41](#)
- Bidly\_NodeTableAddNumber
  - bidly.h, [40](#)
- Bidly\_NodeTableBlockNumber
  - bidly.h, [39](#)
- Bidly\_NodeTableCompareNumber
  - bidly.h, [40](#)
- Bidly\_NodeTableDRTIME
  - bidly.h, [41](#)
- Bidly\_NodeTableFindNumber
  - bidly.h, [40](#)
- Bidly\_NodeTableFoaNumber
  - bidly.h, [39](#)
- Bidly\_NodeTableGCNumber
  - bidly.h, [40](#)
- Bidly\_NodeTableGCObsoleteNumber
  - bidly.h, [40](#)
- Bidly\_NodeTableGCTime
  - bidly.h, [40](#)
- Bidly\_NodeTableGenerated
  - bidly.h, [39](#)
- Bidly\_NodeTableITENumber
  - bidly.h, [41](#)



- Bidly\_NodeTableITERecursiveNumber  
bidly.h, 41
- Bidly\_NodeTableMax  
bidly.h, 39
- Bidly\_NodeTableNum  
bidly.h, 39
- Bidly\_NodeTableNumVar  
bidly.h, 39
- Bidly\_NodeTableResizeNumber  
bidly.h, 39
- Bidly\_NodeTableSiftingNumber  
bidly.h, 40
- Bidly\_NodeTableSize  
bidly.h, 39
- Bidly\_NodeTableSwapNumber  
bidly.h, 40
- Bidly\_NodeTableXORNumber  
bidly.h, 41
- Bidly\_NodeTableXORRecursiveNumber  
bidly.h, 41
- Bidly\_Nor  
bidly.h, 35
- Bidly\_Not  
bidly.h, 34
- Bidly\_OPCCacheFind  
bidly.h, 42
- Bidly\_OPCCacheInsert  
bidly.h, 42
- Bidly\_OPCCacheOverwrite  
bidly.h, 43
- Bidly\_OPCCacheSearch  
bidly.h, 42
- Bidly\_Or  
bidly.h, 35
- Bidly\_PrintInfo  
bidly.h, 44
- Bidly\_PrintfBDD  
bidly.h, 44
- Bidly\_PrintfSOP  
bidly.h, 45
- Bidly\_PrintfTable  
bidly.h, 45
- Bidly\_Purge  
bidly.h, 32
- Bidly\_PurgeAndReorder  
bidly.h, 32
- Bidly\_RandomFunction  
bidly.h, 38
- Bidly\_RandomSet  
bidly.h, 38
- Bidly\_ReadMemoryInUse  
bidly.h, 44
- Bidly\_ReadVerilogFile  
bidly.h, 44
- Bidly\_Refresh  
bidly.h, 32
- Bidly\_Regular  
bidly.h, 24
- Bidly\_ReplaceByKeyword  
bidly.h, 37
- Bidly\_ResetVariablesValue  
bidly.h, 29
- Bidly\_Restrict  
bidly.h, 36
- Bidly\_SelectFunction  
bidly.h, 27
- Bidly\_SelectNode  
bidly.h, 26
- Bidly\_SetManagerParameters  
bidly.h, 26
- Bidly\_SetMark  
bidly.h, 24
- Bidly\_SetTag  
bidly.h, 25
- Bidly\_SetVariableData  
bidly.h, 29
- Bidly\_SetVariableValue  
bidly.h, 29
- Bidly\_Sifting  
bidly.h, 33
- Bidly\_Simplify  
bidly.h, 37
- Bidly\_String  
bidly.h, 46
- Bidly\_Subset  
bidly.h, 37
- Bidly\_Support  
bidly.h, 37
- Bidly\_SwapWithHigher  
bidly.h, 33
- Bidly\_SwapWithLower  
bidly.h, 33
- Bidly\_TaggedFoaNode  
bidly.h, 31
- Bidly\_TransferMark  
bidly.h, 31
- Bidly\_UnivAbstract  
bidly.h, 36
- Bidly\_Untagged  
bidly.h, 25
- Bidly\_Variable  
bidly.h, 46
- Bidly\_VariableTableNum  
bidly.h, 38
- Bidly\_WriteBDD  
bidly.h, 44
- Bidly\_WriteBddview  
bidly.h, 45
- Bidly\_WriteDot  
bidly.h, 45
- Bidly\_WriteSOP  
bidly.h, 45
- Bidly\_WriteTable  
bidly.h, 45
- Bidly\_Xnor  
bidly.h, 35



- Biddy\_Xor
  - biddy.h, 35
- Biddy\_XY, 17
- Biddy\_A
  - biddy.h, 36
- Biddy\_E
  - biddy.h, 36
- biddyInOut.c, 47
  - Biddy\_Managed\_Eval0, 48
  - Biddy\_Managed\_Eval1x, 49
  - Biddy\_Managed\_Eval2, 49
  - Biddy\_Managed\_PrintfBDD, 50
  - Biddy\_Managed\_PrintfSOP, 51
  - Biddy\_Managed\_PrintfTable, 50
  - Biddy\_Managed\_ReadVerilogFile, 49
  - Biddy\_Managed\_WriteBDD, 50
  - Biddy\_Managed\_WriteBddview, 52
  - Biddy\_Managed\_WriteDot, 52
  - Biddy\_Managed\_WriteSOP, 51
  - Biddy\_Managed\_WriteTable, 51
- biddyInt.h, 53
- biddyMain.c, 53
  - Biddy\_About, 59
  - Biddy\_ExitMNG, 58
  - Biddy\_GetElse, 61
  - Biddy\_GetThen, 61
  - Biddy\_GetTopVariable, 62
  - Biddy\_InitMNG, 58
  - Biddy\_Managed\_AddCache, 85
  - Biddy\_Managed\_AddElementByName, 78
  - Biddy\_Managed\_AddFormula, 86
  - Biddy\_Managed\_AddVariableAbove, 79
  - Biddy\_Managed\_AddVariableBelow, 79
  - Biddy\_Managed\_AddVariableByName, 78
  - Biddy\_Managed\_ChangeVariableName, 77
  - Biddy\_Managed\_Clean, 83
  - Biddy\_Managed\_ClearVariablesData, 73
  - Biddy\_Managed\_DeleteFormula, 87
  - Biddy\_Managed\_DeletelthFormula, 88
  - Biddy\_Managed\_DeselectAll, 65
  - Biddy\_Managed\_DeselectNode, 63
  - Biddy\_Managed\_FindFormula, 87
  - Biddy\_Managed\_FoaVariable, 76
  - Biddy\_Managed\_GC, 82
  - Biddy\_Managed\_GetBaseSet, 66
  - Biddy\_Managed\_GetConstantOne, 66
  - Biddy\_Managed\_GetConstantZero, 66
  - Biddy\_Managed\_GetElementEdge, 70
  - Biddy\_Managed\_GetlthFormula, 88
  - Biddy\_Managed\_GetlthFormulaName, 89
  - Biddy\_Managed\_GetlthVariable, 68
  - Biddy\_Managed\_GetLowestVariable, 67
  - Biddy\_Managed\_GetManagerName, 60
  - Biddy\_Managed\_GetManagerType, 59
  - Biddy\_Managed\_GetNextVariable, 69
  - Biddy\_Managed\_GetPrevVariable, 68
  - Biddy\_Managed\_GetTerminal, 65
  - Biddy\_Managed\_GetTopVariableChar, 72
  - Biddy\_Managed\_GetTopVariableEdge, 71
  - Biddy\_Managed\_GetTopVariableName, 71
  - Biddy\_Managed\_GetVariable, 67
  - Biddy\_Managed\_GetVariableData, 74
  - Biddy\_Managed\_GetVariableEdge, 69
  - Biddy\_Managed\_GetVariableName, 70
  - Biddy\_Managed\_GetVariableValue, 73
  - Biddy\_Managed\_IncTag, 80
  - Biddy\_Managed\_IsEqv, 62
  - Biddy\_Managed\_IsHighest, 76
  - Biddy\_Managed\_IsLowest, 75
  - Biddy\_Managed\_IsOK, 82
  - Biddy\_Managed\_IsSelected, 64
  - Biddy\_Managed\_IsSmaller, 75
  - Biddy\_Managed\_MaximizeBDD, 92
  - Biddy\_Managed\_MinimizeBDD, 91
  - Biddy\_Managed\_Purge, 84
  - Biddy\_Managed\_PurgeAndReorder, 84
  - Biddy\_Managed\_Refresh, 85
  - Biddy\_Managed\_ResetVariablesValue, 72
  - Biddy\_Managed\_SelectFunction, 64
  - Biddy\_Managed\_SelectNode, 62
  - Biddy\_Managed\_SetManagerParameters, 60
  - Biddy\_Managed\_SetVariableData, 74
  - Biddy\_Managed\_SetVariableValue, 73
  - Biddy\_Managed\_Sifting, 90
  - Biddy\_Managed\_SwapWithHigher, 89
  - Biddy\_Managed\_SwapWithLower, 90
  - Biddy\_Managed\_TaggedFoaNode, 81
  - Biddy\_Managed\_TransferMark, 79
- biddyMainLegacy.c, 92
  - Biddy\_About, 98
  - Biddy\_ExitMNG, 98
  - Biddy\_GetElse, 99
  - Biddy\_GetThen, 99
  - Biddy\_GetTopVariable, 100
  - Biddy\_InitMNG, 97
  - Biddy\_Managed\_AddCache, 128
  - Biddy\_Managed\_AddElementByName, 113
  - Biddy\_Managed\_AddFormula, 129
  - Biddy\_Managed\_AddVariableAbove, 114
  - Biddy\_Managed\_AddVariableBelow, 113
  - Biddy\_Managed\_AddVariableByName, 112
  - Biddy\_Managed\_And, 116
  - Biddy\_Managed\_AndAbstract, 123
  - Biddy\_Managed\_Change, 125
  - Biddy\_Managed\_Clean, 126
  - Biddy\_Managed\_Compose, 120
  - Biddy\_Managed\_Constrain, 123
  - Biddy\_Managed\_DeleteFormula, 130
  - Biddy\_Managed\_DeletelthFormula, 130
  - Biddy\_Managed\_DeselectAll, 103
  - Biddy\_Managed\_DeselectNode, 101
  - Biddy\_Managed\_ExistAbstract, 122
  - Biddy\_Managed\_FindFormula, 129
  - Biddy\_Managed\_FoaVariable, 111
  - Biddy\_Managed\_GC, 126
  - Biddy\_Managed\_GetBaseSet, 105

- Biddy\_Managed\_GetConstantOne, 104
- Biddy\_Managed\_GetConstantZero, 104
- Biddy\_Managed\_GetElementEdge, 107
- Biddy\_Managed\_GetIthFormula, 131
- Biddy\_Managed\_GetIthFormulaName, 131
- Biddy\_Managed\_GetManagerType, 98
- Biddy\_Managed\_GetNextVariable, 106
- Biddy\_Managed\_GetPrevVariable, 106
- Biddy\_Managed\_GetTerminal, 103
- Biddy\_Managed\_GetTopVariableChar, 109
- Biddy\_Managed\_GetTopVariableEdge, 108
- Biddy\_Managed\_GetTopVariableName, 108
- Biddy\_Managed\_GetVariable, 105
- Biddy\_Managed\_GetVariableEdge, 106
- Biddy\_Managed\_GetVariableName, 107
- Biddy\_Managed\_Gt, 119
- Biddy\_Managed\_ITE, 116
- Biddy\_Managed\_IncTag, 115
- Biddy\_Managed\_IsEqv, 100
- Biddy\_Managed\_IsLeq, 119
- Biddy\_Managed\_IsOK, 125
- Biddy\_Managed\_IsSelected, 102
- Biddy\_Managed\_IsSmaller, 110
- Biddy\_Managed\_IsVariableDependent, 122
- Biddy\_Managed\_Leq, 119
- Biddy\_Managed\_Nand, 117
- Biddy\_Managed\_Nor, 118
- Biddy\_Managed\_Not, 116
- Biddy\_Managed\_Or, 117
- Biddy\_Managed\_Purge, 127
- Biddy\_Managed\_PurgeAndReorder, 127
- Biddy\_Managed\_Random, 133
- Biddy\_Managed\_RandomSet, 134
- Biddy\_Managed\_Refresh, 128
- Biddy\_Managed\_Replace, 124
- Biddy\_Managed\_ResetVariablesValue, 109
- Biddy\_Managed\_Restrict, 120
- Biddy\_Managed\_SelectFunction, 102
- Biddy\_Managed\_SelectNode, 101
- Biddy\_Managed\_SetManagerParameters, 99
- Biddy\_Managed\_SetVariableValue, 110
- Biddy\_Managed\_Sifting, 133
- Biddy\_Managed\_Simplify, 123
- Biddy\_Managed\_SubIntersect, 120
- Biddy\_Managed\_Subset, 125
- Biddy\_Managed\_Support, 124
- Biddy\_Managed\_SwapWithHigher, 132
- Biddy\_Managed\_SwapWithLower, 132
- Biddy\_Managed\_TaggedFoaNode, 115
- Biddy\_Managed\_TransferMark, 114
- Biddy\_Managed\_UnivAbstract, 122
- Biddy\_Managed\_Xnor, 118
- Biddy\_Managed\_Xor, 118
- Biddy\_Managed\_A, 121
- Biddy\_Managed\_E, 121
- biddyOp.c, 134
  - Biddy\_Managed\_And, 137
  - Biddy\_Managed\_AndAbstract, 143
  - Biddy\_Managed\_Change, 145
  - Biddy\_Managed\_Compose, 141
  - Biddy\_Managed\_Constrain, 143
  - Biddy\_Managed\_CreateFunction, 146
  - Biddy\_Managed\_CreateMinterm, 146
  - Biddy\_Managed\_ExistAbstract, 142
  - Biddy\_Managed\_Gt, 140
  - Biddy\_Managed\_ITE, 137
  - Biddy\_Managed\_IsLeq, 140
  - Biddy\_Managed\_IsVariableDependent, 142
  - Biddy\_Managed\_Leq, 139
  - Biddy\_Managed\_Nand, 138
  - Biddy\_Managed\_Nor, 138
  - Biddy\_Managed\_Not, 136
  - Biddy\_Managed\_Or, 137
  - Biddy\_Managed\_RandomFunction, 146
  - Biddy\_Managed\_RandomSet, 147
  - Biddy\_Managed\_ReplaceByKeyword, 144
  - Biddy\_Managed\_Restrict, 140
  - Biddy\_Managed\_Simplify, 144
  - Biddy\_Managed\_Subset, 145
  - Biddy\_Managed\_Support, 144
  - Biddy\_Managed\_UnivAbstract, 143
  - Biddy\_Managed\_Xnor, 139
  - Biddy\_Managed\_Xor, 138
  - Biddy\_Managed\_A, 141
  - Biddy\_Managed\_E, 141
- biddyStat.c, 147
  - Biddy\_AvgLevel, 151
  - Biddy\_Managed\_CountComplementedEdges, 163
  - Biddy\_Managed\_CountMinterms, 164
  - Biddy\_Managed\_CountNodes, 150
  - Biddy\_Managed\_CountNodesPlain, 163
  - Biddy\_Managed\_CountPaths, 164
  - Biddy\_Managed\_DensityOfBDD, 165
  - Biddy\_Managed\_DensityOfFunction, 165
  - Biddy\_Managed\_DependentVariableNumber, 163
  - Biddy\_Managed\_FormulaTableNum, 160
  - Biddy\_Managed\_ListAvgLength, 161
  - Biddy\_Managed\_ListMaxLength, 160
  - Biddy\_Managed\_ListUsed, 160
  - Biddy\_Managed\_NodeTableANDORNumber, 158
  - Biddy\_Managed\_NodeTableANDORRecursive↔  
Number, 159
  - Biddy\_Managed\_NodeTableAddNumber, 155
  - Biddy\_Managed\_NodeTableBlockNumber, 152
  - Biddy\_Managed\_NodeTableCompareNumber, 155
  - Biddy\_Managed\_NodeTableDRTime, 157
  - Biddy\_Managed\_NodeTableFindNumber, 155
  - Biddy\_Managed\_NodeTableFoaNumber, 155
  - Biddy\_Managed\_NodeTableGCNumber, 156
  - Biddy\_Managed\_NodeTableGCObsoleteNumber,  
156
  - Biddy\_Managed\_NodeTableGCTime, 156
  - Biddy\_Managed\_NodeTableGenerated, 153
  - Biddy\_Managed\_NodeTableITENumber, 158
  - Biddy\_Managed\_NodeTableITERecursiveNumber,  
158

Biddy\_Managed\_NodeTableMax, [153](#)  
Biddy\_Managed\_NodeTableNum, [153](#)  
Biddy\_Managed\_NodeTableNumVar, [154](#)  
Biddy\_Managed\_NodeTableResizeNumber, [154](#)  
Biddy\_Managed\_NodeTableSiftingNumber, [157](#)  
Biddy\_Managed\_NodeTableSize, [152](#)  
Biddy\_Managed\_NodeTableSwapNumber, [157](#)  
Biddy\_Managed\_NodeTableXORNumber, [159](#)  
Biddy\_Managed\_NodeTableXORRecursive↔  
Number, [159](#)  
Biddy\_Managed\_OPCacheFind, [162](#)  
Biddy\_Managed\_OPCacheInsert, [162](#)  
Biddy\_Managed\_OPCacheOverwrite, [162](#)  
Biddy\_Managed\_OPCacheSearch, [161](#)  
Biddy\_Managed\_PrintInfo, [166](#)  
Biddy\_Managed\_ReadMemoryInUse, [165](#)  
Biddy\_Managed\_VariableTableNum, [152](#)  
Biddy\_MaxLevel, [151](#)