```
# ------------------------------------------------------------
# File testplan.tpl
# ------------------------------------------------------------
Version 1.0;

# Imports of OTPL sources
Import OASISResources.rsc;
Import testcondition.tcg;
Import asicbins.bdefs;

Import DatalogSetupTest.ph;
Import FunctionalTest.ph;
Import SearchTest.ph;
Import DCParametricPowerSupplyTest.ph;
Import DCParametricPMUTest.ph;

# Import of Runtime files
Import pindesc.pin;

#------------------------------------------------------------
# Start of the test plan
#------------------------------------------------------------

# Name of the TestPlan
TestPlan testplan;

# The type of DUT
DUTType "DiagPB";

# This block defines Pattern Lists file-qualified names and
# Pattern List variables that are used in Test declarations.
# The file-qualified names refer to pattern lists in the named
# files.  The variables refer to String variables which will
# hold the pattern list names at run time.  User defined
# Flowable objects could set the values of these variables
# through an API provided by OASIS.
#PListDefs
#{
#  # File qualified pattern list names
#  pattern.plist:DiagPBPat,
#  wireModel_fixedCycle.plist:wireModel_fixedCycle,
#  wireModel_Alpg.plist:wireModel_PList_AlpgTiming,
#  wireModel_Scan.plist:wireModel_PList_ScanTiming
#}


# SocketDef, UserVars declaration as before ...
SocketDef = socket.soc;

# Offline configuration file
OfflineDef = offline.cfg;

# Declarations of TestConditions TC1Min, TC1Typ, TC1Max,
# TC2Min, TC2Typ, TC2Max as before ...
```

```
55: TestCondition TC1Min
56: {
57:     TestConditionGroup = DiagPBTCG;
58:     Selector = min;
59: }
60:
61: TestCondition TC1Max
62: {
63:     TestConditionGroup = DiagPBTCG;
64:     Selector = max;
65: }
66:
67: TestCondition TC1Typ
68: {
69:     TestConditionGroup = DiagPBTCG;
70:     Selector = typ;
71: }
72:
73: TestCondition TC1LevelMin
74: {
75:     TestConditionGroup = DiagPBTCG_Level;
76:     Selector = min;
77: }
78:
79: TestCondition TC1TimingMax
80: {
81:     TestConditionGroup = DiagPBTCG_Timing;
82:     Selector = max;
83: }
84:
85: TestCondition RoffSeq
86: {
87:     TestConditionGroup = RoffSeqCondition;
88: }
89:
90: #
91: # Test condition for Vdd Short test
92: #-------------------------------------------------
93: TestCondition Idd_Short_Levels
94: {
95:     TestConditionGroup = wireModel_Vdd_Short_Levels;
96: }
97: TestCondition Idd_Short_DCParams
98: {
99:     TestConditionGroup = wireModel_Vdd_Short_DCParam;
100: }
101:
102: #
103: # Test condition for Open Test
104: #-------------------------------------------------
105: TestCondition Contact_Open_Levels
106: {
107:    TestConditionGroup = wireModel_Open_Levels;
108: }
```

```
109: TestCondition Contact_Open_DCParams
110: {
111:    TestConditionGroup = wireModel_Open_DCParam;
112: }
113:
114: #
115: # Test condition for Input Leakage Low test
116: #-------------------------------------------------
117: TestCondition Input_Leakage_Low_Levels
118: {
119:    TestConditionGroup = wireModel_Leakage_Levels;
120:    Selector = iil;
121: }
122: TestCondition Input_Leakage_Low_DCParams
123: {
124:    TestConditionGroup = wireModel_Leakage_DCParam;
125:    Selector = vinl;
126: }
127:
128: #
129: # Test condition for Idd static vdd_max test
130: #-------------------------------------------------
131: TestCondition Idd_Static_Levels_Max
132: {
133:    TestConditionGroup = wireModel_Idd_Levels;
134:    Selector = max;
135: }
136: TestCondition Idd_Static_DCParams_Max
137: {
138:    TestConditionGroup = wireModel_Idd_DCParam;
139:    Selector = max;
140: }
141:
142: # Minimum Voltage Group
143: #-------------------------------------------------
144: TestCondition Func_Min_Dps
145: {
146:    TestConditionGroup = wireModel_Func_Dps_Levels;
147:    Selector = vmin;
148: }
149: TestCondition Func_Min_Sig_VT
150: {
151:    TestConditionGroup = wireModel_Func_SDR_VT_Levels;
152:    Selector = vmin;
153: }
154:
155: # Timing Condition
156: #-------------------------------------------------
157:
158: TestCondition Func_Timing_SDR_Scan
159: {
160:    TestConditionGroup = wireModel_Func_SDR_Scan_Timings;
161: }
162: TestCondition Func_Timing_SDR_Alpg
```

```
163: {
164:    TestConditionGroup = wireModel_Func_SDR_Alpg_Timings;
165: }
166:
167: # Pin Open Group
168: #-----------------------------------------------
169: TestCondition Pin_5_8_OPEN
170: {
171:    TestConditionGroup = wireModel_Func_Pin_Open;
172: }
173:
174: # Test condition for Idd test
175: #-----------------------------------------------
176: TestCondition Idd_Static_Function_Min
177: {
178:    TestConditionGroup = wireModel_Idd_Static_Function;
179:    Selector = min;
180: }
181: TestCondition Idd_Static_DCParams_Min
182: {
183:    TestConditionGroup = wireModel_Idd_TraceMode_DCParam;
184:    Selector = min;
185: }
186:
187: ##########################################################################
Tests for Datalog Setup
188: Test DatalogSetupTest DatalogSetup
189: {
190:    DatalogSetupFileName = "DatalogStreamSetup.dls";
191:    DatalogSetupFileName = "FuncDatalogSetup.dls";
192: }
193:
194: ##########################################################################
Tests for DC
195: Test DCParametricPowerSupplyTest DiagPB_Vdd_Short
196: {
197:    TestConditionParam= Idd_Short_DCParams;
198:    TestConditionParam= Idd_Short_Levels;
199:    PowerSupply= "VDD";
200:    SettlingTimePreMeasure= "3mS";
201:    SettlingTimePostMeasure= "0mS";
202: }
203: Test DCParametricPMUTest DiagPB_Contact_Open
204: {
205:    TestConditionParam= Contact_Open_Levels;
206:    TestConditionParam= Contact_Open_DCParams;
207:    MeasurementPin= "allpins";
208:    PMURelayControl= "On";
209:    MeasMode= "Simultaneous";
210:    SettlingTimePreMeasure= "3mS";
211:    SettlingTimePostMeasure= "0mS";
212: }
213:
214: #
```

```
215: # Declare a FunctionalTest.  "FunctionalTest" refers to a C++
216: # test class that runs the test, and returns a 0, 1 or 2 as
217: # a Result.  The Test Condition Group TCG1 is selected with
218: # the "min" selector by referring to the TC1Min TestCondition.
219: #
220: # Note that OTPL can compile this because of the imported
221: # FunctionalTest.ph file.
222: #
223: ##########################################################################
Tests for Function
224: Test FunctionalTest DiagPBFunctionalTestMin
225: {
226:    PListParam = DiagPBPat;
227:    TestConditionParam = TC1Min;
228: }
229:
230: Test FunctionalTest DiagPBFunctionalTestMax
231: {
232:    PListParam = DiagPBPat;
233:    TestConditionParam = TC1Max;
234: }
235:
236: Test FunctionalTest DiagPBFunctionalTestTyp
237: {
238:    PListParam = DiagPBPat;
239:    TestConditionParam = TC1Typ;
240: }
241:
242: ##########################################################################
Tests for Search
243: Test SearchTest Margin
244: {
245:    PListParam = DiagPBPat;
246:    TestConditionParam = TC1LevelMin;
247:    TestConditionParam = TC1TimingMax;
248:    Algorithm = "margin";
249:    SearchVarParam = ",TC1TimingMax,tStrbH,0nS,10nS,10";
250: }
251:
252: Test SearchTest Shmoo
253: {
254:    PListParam = DiagPBPat;
255:    TestConditionParam = TC1LevelMin;
256:    TestConditionParam = TC1TimingMax;
257:    Algorithm = "shmoo";
258:    SearchVarParam = "x,TC1TimingMax,tStrbH,0nS,10nS,10";
259:    SearchVarParam = "y,TC1LevelMin,voh,1V,0.5V,12";
260:    DebugMode = 0;
261: }
262:
263: ##########################################################################
Tests for DC
264: Test DCParametricPMUTest DiagPB_Leakage_Low
265: {
```

```
266:    TestConditionParam= Input_Leakage_Low_Levels;
267:    TestConditionParam= Input_Leakage_Low_DCParams;
268:    MeasurementPin= "inpins";
269:    PMURelayControl= "On";
270:    MeasMode= "Serial";
271:    SettlingTimePreMeasure= "3mS";
272:    SettlingTimePostMeasure= "0mS";
273: }
274:
275: Test DCParametricPowerSupplyTest DiagPB_Idd_Static_Max
276: {
277:    TestConditionParam= Idd_Static_DCParams_Max;
278:    TestConditionParam= Idd_Static_Levels_Max;
279:    PowerSupply= "VDD";
280:    PinRelay= "Open";
281:    SettlingTimePreMeasure= "3mS";
282:    SettlingTimePostMeasure= "0mS";
283: }
284:
285: ##########################################################################
Tests for SCAN
286:
287: Test FunctionalTest DiagPB_Func_SCAN_VT_Min
288: {
289:    PListParam= wireModel_PList_ScanTiming;
290:    TestConditionParam= Func_Min_Dps;
291:    TestConditionParam= Func_Min_Sig_VT;
292:    TestConditionParam= Func_Timing_SDR_Scan;
293: }
294:
295: ##########################################################################
Tests for ALPG
296: Test FunctionalTest DiagPB_Func_ALPG_VT_Min
297: {
298:    PListParam= wireModel_PList_AlpgTiming;
299:    TestConditionParam= Func_Min_Dps;
300:    TestConditionParam= Func_Min_Sig_VT;
301:    TestConditionParam= Pin_5_8_OPEN;
302:    TestConditionParam= Func_Timing_SDR_Alpg;
303: }
304:
305: ##########################################################################
Test for Module Trigger
306: Test DCParametricPowerSupplyTest DiagPB_Idd_with_ModuleTrigger
307: {
308:    TestConditionParam= Idd_Static_Function_Min;
309:    TestConditionParam= Idd_Static_DCParams_Min;
310:    PListParam = wireModel_fixedCycle;
311:    PowerSupply= "VDD";
312:    ModuleTrigger= "PMDTR2,0xf";
313: }
314:
315: EndSequence
316: {
```

```
317:    RoffSeq
318: }
319:
320: # Counters
321: Counters {PassCount, FailCount}
322:
323:
324: # FlowMain is the main flow.
325: Flow FlowMain
326: {
327:    #
328:    # The first declared flow is the initial flow to be
329:    # executed.  It goes to FlowMain_Max on success, and
330:    # returns 1 on failure.
331:
332:    FlowItem DatalogSetupFlow DatalogSetup
333:    {
334:      Result 0
335:      {
336:        Property PassFail = "Pass";
337:        GoTo FlowMain_Vdd_Short;
338:      }
339:    }
340:    FlowItem FlowMain_Vdd_Short DiagPB_Vdd_Short
341:    {
342:      Result 0
343:      {
344:        Property PassFail = "Pass";
345:        IncrementCounters PassCount;
346:        GoTo FlowMain_Open;
347:      }
348:      Result 1
349:      {
350:        Property PassFail = "Fail";
351:        IncrementCounters FailCount;
352:        SetBin SoftBins.FailVddShortTest;
353:        GoTo FlowMain_Open;
354:      }
355:    }
356:    FlowItem FlowMain_Open DiagPB_Contact_Open
357:    {
358:      Result 0
359:      {
360:        Property PassFail = "Pass";
361:        IncrementCounters PassCount;
362:        GoTo FlowMain_IIL;
363:      }
364:      Result 1
365:      {
366:        Property PassFail = "Fail";
367:        IncrementCounters FailCount;
368:        SetBin SoftBins.FailOpenTest;
369:        GoTo FlowMain_IIL;
370:      }
```

```
371:    }
372:    FlowItem FlowMain_IIL DiagPB_Leakage_Low
373:    {
374:     Result 0
375:     {
376:      Property PassFail = "Pass";
377:      IncrementCounters PassCount;
378:      GoTo FlowMain_Min;
379:     }
380:     Result 1
381:     {
382:      Property PassFail = "Fail";
383:      IncrementCounters FailCount;
384:      SetBin SoftBins.FailLeakageL;
385:      GoTo FlowMain_Min;
386:     }
387:    }
388:    FlowItem FlowMain_Min DiagPBFunctionalTestMin
389:    {
390:     Result 0
391:     {
392:      Property PassFail = "Pass";
393:      IncrementCounters PassCount;
394:      GoTo FlowMain_Max;
395:     }
396:
397:     Result 1
398:     {
399:      Property PassFail = "Fail";
400:      IncrementCounters FailCount;
401:      SetBin SoftBins.FailCache3GHz;
402:      Return 1;
403:     }
404:    }
405:    FlowItem FlowMain_Max DiagPBFunctionalTestMax
406:    {
407:     Result 0
408:     {
409:      Property PassFail = "Pass";
410:      IncrementCounters PassCount;
411:      GoTo FlowMain_Typ;
412:     }
413:
414:     Result 1
415:     {
416:      Property PassFail = "Fail";
417:      IncrementCounters FailCount;
418:      SetBin SoftBins.FailSBFT3GHz;
419:      Return 1;
420:     }
421:    }
422:    FlowItem FlowMain_Typ DiagPBFunctionalTestTyp
423:    {
424:     Result 0
```

```
425:     {
426:       Property PassFail = "Pass";
427:       IncrementCounters PassCount;
428:       GoTo MarginFlow;
429:     }
430:
431:     Result 1
432:     {
433:       Property PassFail = "Fail";
434:       IncrementCounters FailCount;
435:       SetBin SoftBins.FailLeakage3GHz;
436:       Return 1;
437:     }
438:   }
439:   FlowItem MarginFlow Margin
440:   {
441:     Result 0
442:     {
443:       Property PassFail = "Pass";
444:       GoTo ShmooFlow;
445:     }
446:   }
447:   FlowItem ShmooFlow Shmoo
448:   {
449:     Result 0
450:     {
451:       Property PassFail = "Pass";
452:       GoTo FlowMain_Idd_Static_Max;
453:     }
454:   }
455:   FlowItem FlowMain_Idd_Static_Max DiagPB_Idd_Static_Max
456:   {
457:     Result 0
458:     {
459:       Property PassFail = "Pass";
460:       IncrementCounters PassCount;
461:       GoTo FlowMain_Func_S_SCAN_Min;
462:     }
463:
464:     Result 1
465:     {
466:       Property PassFail = "Fail";
467:       IncrementCounters FailCount;
468:       SetBin SoftBins.FailIddMax;
469:       GoTo FlowMain_Func_S_SCAN_Min;
470:     }
471:   }
472:   FlowItem FlowMain_Func_S_SCAN_Min DiagPB_Func_SCAN_VT_Min
473:   {
474:     Result 0
475:     {
476:       Property PassFail = "Pass";
477:       IncrementCounters PassCount;
478:       GoTo FlowMain_Func_S_ALPG_Min;
```

```
479:     }
480:     Result 1
481:     {
482:       Property PassFail = "Fail";
483:       IncrementCounters FailCount;
484:       SetBin SoftBins.FailFuncMin;
485:       Return 1;
486:     }
487:    }
488:    FlowItem FlowMain_Func_S_ALPG_Min DiagPB_Func_ALPG_VT_Min
489:    {
490:     Result 0
491:     {
492:       Property PassFail = "Pass";
493:       IncrementCounters PassCount;
494:       GoTo FlowMain_Idd_with_ModuleTrigger;
495:     }
496:     Result 1
497:     {
498:       Property PassFail = "Fail";
499:       IncrementCounters FailCount;
500:       SetBin SoftBins.FailFuncMin;
501:       Return 1;
502:     }
503:    }
504:    FlowItem FlowMain_Idd_with_ModuleTrigger DiagPB_Idd_with_ModuleTrigger
505:    {
506:     Result 0
507:     {
508:       Property PassFail = "Pass";
509:       IncrementCounters PassCount;
510:       SetBin SoftBins.PassAll3GHz;
511:       Return 0;
512:     }
513:     Result 1
514:     {
515:       Property PassFail = "Fail";
516:       IncrementCounters FailCount;
517:       SetBin SoftBins.FailIddMin;
518:       Return 1;
519:     }
520:    }
521: }
522:
523: # Specifies some special flows, including the flow that
524: # is the main flow.  This can be used to specify other
525: # flows that can be launched when a test starts, etc.
526: FlowDefs
527: {
528:    MainFlow = FlowMain;
529: }
```