Cuttle覆盖分析报告
改进的条件判定覆盖分析报告(MCDC), DO-178B A级
Cuttle报告生成器版本: ver4.1
报告生成时间: Mon Jun 24 19:53:44 CST 2013
IDB工程文件目录:
E:\UIP-Kernel\uipkernel\ct-A.idb
Cuttle项目
|
+------ 总体概要(MCDC): 共5个文件
| 106个真-假判定
| 覆 盖 数: 95
| 部分覆盖数: 11
| 未 覆 盖: 0
| 120个条件判定
| 覆 盖 数: 108
| 部分覆盖数: 12
| 未 覆 盖: 0
| 0个条件分支
| 覆 盖 数: 0
| 未 覆 盖: 0
| 116个覆盖事件
| 覆 盖 数: 116
| 未 覆 盖: 0
|
+------ 1. 文件 uip_kernel_alarm.c 目录位置 D:\UIP-Kernel\uipkernel\uipkernel_original\src
| | 最后更新时间: Mon Jun 24 09:43:26 CST 2013
| | 文件校验和: XXXXXX
| |
| +------ 文件概要(MCDC): 共9个函数
| | 29个真-假判定
| | 覆 盖 数: 28
| | 部分覆盖数: 1
| | 未 覆 盖: 0
| | 34个条件判定
| | 覆 盖 数: 33
| | 部分覆盖数: 1
| | 未 覆 盖: 0
| | 0个条件分支
| | 覆 盖 数: 0
| | 未 覆 盖: 0
| | 33个覆盖事件
| | 覆 盖 数: 33
| | 未 覆 盖: 0
| |
| +------ 1.1 函数 UIP_GetAlarmBase
| | | StatusType UIP_GetAlarmBase(AlarmType AlarmID, AlarmBaseRefType Info)
| | |
| | +------ 函数概要(MCDC):
| | | 1个真-假判定
| | | 覆 盖 数: 1
| | | 部分覆盖数: 0

```
|     |     |                 未 覆 盖: 0
|     |     |           1个条件判定
|     |     |                 覆 盖 数: 1
|     |     |                 部分覆盖数: 0
|     |     |                 未 覆 盖: 0
|     |     |           0个条件分支
|     |     |                 覆 盖 数: 0
|     |     |                 未 覆 盖: 0
|     |     |           3个覆盖事件
|     |     |                 覆 盖 数: 3
|     |     |                 未 覆 盖: 0
|     |     |
|     |     +------ coverage行 459: 函数入口
|     |     |           {
|     |     |           覆盖
|     |     |
|     |     +------ decision行 462: if语句
|     |     |           AlarmID > uipAlarmNumber
|     |     |
|     |     |                 T       F
|     |     |           01:    *t      *f(AlarmID > uipAlarmNumber)覆盖
|     |     |
|     |     +------ coverage行 464: 函数返回语句
|     |     |           return (E_OS_ID);
|     |     |           覆盖
|     |     |
|     |     +------ coverage行 471: 函数返回语句
|     |           return (E_OK);
|     |           覆盖
|     |
|     +------ 1.2 函数 UIP_GetAlarm
|     |     |     StatusType UIP_GetAlarm(AlarmType AlarmID, TickRefType Tick)
|     |     |
|     |     +------ 函数概要(MCDC):
|     |     |           3个真-假判定
|     |     |                 覆 盖 数: 3
|     |     |                 部分覆盖数: 0
|     |     |                 未 覆 盖: 0
|     |     |           3个条件判定
|     |     |                 覆 盖 数: 3
|     |     |                 部分覆盖数: 0
|     |     |                 未 覆 盖: 0
|     |     |           0个条件分支
|     |     |                 覆 盖 数: 0
|     |     |                 未 覆 盖: 0
|     |     |           4个覆盖事件
|     |     |                 覆 盖 数: 4
|     |     |                 未 覆 盖: 0
|     |     |
|     |     +------ coverage行 530: 函数入口
|     |     |           {
```

```
|       |       |           覆盖
|       |       |
|       |       +------ decision行 534: if语句
|       |       |           AlarmID > uipAlarmNumber
|       |       |
|       |       |                T       F
|       |       |           01:    *t      *f(AlarmID > uipAlarmNumber)覆盖
|       |       |
|       |       +------ coverage行 536: 函数返回语句
|       |       |           return (E_OS_ID);
|       |       |           覆盖
|       |       |
|       |       +------ decision行 544: if语句
|       |       |           (alarmPtr -> uipAlarmState) == uip_ALARM_OFF
|       |       |
|       |       |                T       F
|       |       |           01:    *t      *f((alarmPtr -> uipAlarmState) == uip_ALARM_OFF)覆盖
|       |       |
|       |       +------ coverage行 549: 函数返回语句
|       |       |           return (E_OS_NOFUNC);
|       |       |           覆盖
|       |       |
|       |       +------ decision行 553: if语句
|       |       |           (alarmPtr -> uipAlarmValue) >= ((alarmPtr -> uipCounterPtr) -> uipCtrValue)
|       |       |
|       |       |                T       F
|       |       |           01:    *t      *f((alarmPtr -> uipAlarmValue) >= ((alarmPtr -> uipCounterPtr) ->
uipCtrValue))覆盖
|       |       |
|       |       +------ coverage行 568: 函数返回语句
|       |                   return (E_OK);
|       |                   覆盖
|       |
|       +------ 1.3 函数 UIP_SetRelAlarm
|       |       |       StatusType UIP_SetRelAlarm(AlarmType AlarmID, TickType increment, TickType cycle)
|       |       |
|       |       +------ 函数概要(MCDC):
|       |       |           6个真-假判定
|       |       |                覆 盖 数: 6
|       |       |                部分覆盖数: 0
|       |       |                未 覆 盖: 0
|       |       |           8个条件判定
|       |       |                覆 盖 数: 8
|       |       |                部分覆盖数: 0
|       |       |                未 覆 盖: 0
|       |       |           0个条件分支
|       |       |                覆 盖 数: 0
|       |       |                未 覆 盖: 0
|       |       |           6个覆盖事件
|       |       |                覆 盖 数: 6
|       |       |                未 覆 盖: 0
```

```
|     |     |
|     |     +------ coverage行 650: 函数入口
|     |     |           {
|     |     |           覆盖
|     |     |
|     |     +------ decision行 655: if语句
|     |     |           AlarmID > uipAlarmNumber
|     |     |
|     |     |                 T       F
|     |     |           01:    *t      *f(AlarmID > uipAlarmNumber)覆盖
|     |     |
|     |     +------ coverage行 657: 函数返回语句
|     |     |           return (E_OS_ID);
|     |     |           覆盖
|     |     |
|     |     +------ decision行 666: if语句
|     |     |           (alarmPtr -> uipAlarmState) == uip_ALARM_ON
|     |     |
|     |     |                 T       F
|     |     |           01:    *t      *f((alarmPtr -> uipAlarmState) == uip_ALARM_ON)覆盖
|     |     |
|     |     +------ coverage行 671: 函数返回语句
|     |     |           return (E_OS_STATE);
|     |     |           覆盖
|     |     |
|     |     +------ decision行 675: if语句
|     |     |           increment > (((alarmPtr -> uipCounterPtr)
                        -> uipCtrBase) -> maxAllowedValue)
|     |     |
|     |     |                 T       F
|     |     |           01:    *t      *f(increment > (((alarmPtr -> uipCounterPtr)
                        -> uipCtrBase) -> maxAllowedValue))覆盖
|     |     |
|     |     +------ coverage行 681: 函数返回语句
|     |     |           return (E_OS_VALUE);
|     |     |           覆盖
|     |     |
|     |     +------ decision行 685: if语句中的条件表达式
|     |     |           (cycle!=0) && ((cycle < ((alarmPtr -> uipCounterPtr) -> uipCtrBase) -> minCycle) ||
              (cycle > ((alarmPtr -> uipCounterPtr) -> uipCtrBase) -> maxAllowedValue))
|     |     |
|     |     |                 T          F
|     |     |           01:    *ttx        *fxx((cycle!=0))覆盖
|     |     |                 *tft        *fxx
|     |     |           02:    *ttx        *tff((cycle < ((alarmPtr -> uipCounterPtr) -> uipCtrBase) ->
minCycle))覆盖
|     |     |           03:    *tft        *tff((cycle > ((alarmPtr -> uipCounterPtr) -> uipCtrBase) ->
maxAllowedValue))覆盖
|     |     |
|     |     +------ coverage行 691: 函数返回语句
|     |     |           return (E_OS_VALUE);
```

```
|       |       |           覆盖
|       |       |
|       |       +------ decision行 694: if语句
|       |       |           increment > ((((alarmPtr -> uipCounterPtr) -> uipCtrBase) -> maxAllowedValue)
                            - ((alarmPtr -> uipCounterPtr) -> uipCtrValue))
|       |       |
|       |       |                T      F
|       |       |           01:   *t     *f(increment > ((((alarmPtr -> uipCounterPtr) -> uipCtrBase) ->
maxAllowedValue)
                            - ((alarmPtr -> uipCounterPtr) -> uipCtrValue)))覆盖
|       |       |
|       |       +------ decision行 712: if语句
|       |       |           (alarmPtr -> uipAlarmValue) <= ((alarmPtr -> uipCounterPtr) -> uipCtrValue)
|       |       |
|       |       |                T      F
|       |       |           01:   *t     *f((alarmPtr -> uipAlarmValue) <= ((alarmPtr -> uipCounterPtr) ->
uipCtrValue))覆盖
|       |       |
|       |       +------ coverage行 727: 函数返回语句
|       |                   return (E_OK);
|       |                   覆盖
|       |
|       +------ 1.4 函数 UIP_SetAbsAlarm
|       |       |       StatusType UIP_SetAbsAlarm(AlarmType AlarmID, TickType start, TickType cycle)
|       |       |
|       |       +------ 函数概要(MCDC):
|       |       |           5个真-假判定
|       |       |                覆 盖 数: 5
|       |       |                部分覆盖数: 0
|       |       |                未 覆 盖: 0
|       |       |           7个条件判定
|       |       |                覆 盖 数: 7
|       |       |                部分覆盖数: 0
|       |       |                未 覆 盖: 0
|       |       |           0个条件分支
|       |       |                覆 盖 数: 0
|       |       |                未 覆 盖: 0
|       |       |           6个覆盖事件
|       |       |                覆 盖 数: 6
|       |       |                未 覆 盖: 0
|       |       |
|       |       +------ coverage行 804: 函数入口
|       |       |           {
|       |       |           覆盖
|       |       |
|       |       +------ decision行 809: if语句
|       |       |           AlarmID > uipAlarmNumber
|       |       |
|       |       |                T      F
|       |       |           01:   *t     *f(AlarmID > uipAlarmNumber)覆盖
|       |       |
```

```
|       |       +------ coverage行 811: 函数返回语句
|       |       |             return (E_OS_ID);
|       |       |             覆盖
|       |       |
|       |       +------ decision行 820: if语句
|       |       |             (alarmPtr -> uipAlarmState) == uip_ALARM_ON
|       |       |
|       |       |                   T       F
|       |       |             01:    *t      *f((alarmPtr -> uipAlarmState) == uip_ALARM_ON)覆盖
|       |       |
|       |       +------ coverage行 825: 函数返回语句
|       |       |             return (E_OS_STATE);
|       |       |             覆盖
|       |       |
|       |       +------ decision行 829: if语句
|       |       |             start > (((alarmPtr -> uipCounterPtr) -> uipCtrBase) -> maxAllowedValue)
|       |       |
|       |       |                   T       F
|       |       |             01:    *t      *f(start > (((alarmPtr -> uipCounterPtr) -> uipCtrBase) ->
maxAllowedValue))覆盖
|       |       |
|       |       +------ coverage行 834: 函数返回语句
|       |       |             return (E_OS_VALUE);
|       |       |             覆盖
|       |       |
|       |       +------ decision行 838: if语句中的条件表达式
|       |       |             (cycle != 0) && ((cycle < ((alarmPtr -> uipCounterPtr) -> uipCtrBase) -> minCycle)
        || (cycle > ((alarmPtr -> uipCounterPtr) -> uipCtrBase) -> maxAllowedValue))
|       |       |
|       |       |                   T           F
|       |       |             01:    *ttx        *fxx((cycle != 0))覆盖
|       |       |                   *tft        *fxx
|       |       |             02:    *ttx        *tff((cycle < ((alarmPtr -> uipCounterPtr) -> uipCtrBase) ->
minCycle))覆盖
|       |       |             03:    *tft        *tff((cycle > ((alarmPtr -> uipCounterPtr) -> uipCtrBase) ->
maxAllowedValue))覆盖
|       |       |
|       |       +------ coverage行 844: 函数返回语句
|       |       |             return (E_OS_VALUE);
|       |       |             覆盖
|       |       |
|       |       +------ decision行 855: if语句
|       |       |             (alarmPtr -> uipAlarmValue) <= ((alarmPtr -> uipCounterPtr) -> uipCtrValue)
|       |       |
|       |       |                   T       F
|       |       |             01:    *t      *f((alarmPtr -> uipAlarmValue) <= ((alarmPtr -> uipCounterPtr) ->
uipCtrValue))覆盖
|       |       |
|       |       +------ coverage行 869: 函数返回语句
|       |                     return (E_OK);
|       |                     覆盖
```

```
|      |
|      +------ 1.5 函数 UIP_CancelAlarm
|      |      |      StatusType UIP_CancelAlarm(AlarmType AlarmID)
|      |      |
|      |      +------ 函数概要(MCDC):
|      |      |      |      2个真-假判定
|      |      |      |            覆 盖 数: 2
|      |      |      |            部分覆盖数: 0
|      |      |      |            未 覆 盖: 0
|      |      |      |      2个条件判定
|      |      |      |            覆 盖 数: 2
|      |      |      |            部分覆盖数: 0
|      |      |      |            未 覆 盖: 0
|      |      |      |      0个条件分支
|      |      |      |            覆 盖 数: 0
|      |      |      |            未 覆 盖: 0
|      |      |      |      4个覆盖事件
|      |      |      |            覆 盖 数: 4
|      |      |      |            未 覆 盖: 0
|      |      |      |
|      |      +------ coverage行 925: 函数入口
|      |      |      |      {
|      |      |      |      覆盖
|      |      |      |
|      |      +------ decision行 930: if语句
|      |      |      |      AlarmID > uipAlarmNumber
|      |      |      |
|      |      |      |            T      F
|      |      |      |      01:    *t     *f(AlarmID > uipAlarmNumber)覆盖
|      |      |      |
|      |      +------ coverage行 932: 函数返回语句
|      |      |      |      return (E_OS_ID);
|      |      |      |      覆盖
|      |      |      |
|      |      +------ decision行 940: if语句
|      |      |      |      (alarmPtr -> uipAlarmState) == uip_ALARM_OFF
|      |      |      |
|      |      |      |            T      F
|      |      |      |      01:    *t     *f((alarmPtr -> uipAlarmState) == uip_ALARM_OFF)覆盖
|      |      |      |
|      |      +------ coverage行 945: 函数返回语句
|      |      |      |      return (E_OS_NOFUNC);
|      |      |      |      覆盖
|      |      |      |
|      |      +------ coverage行 953: 函数返回语句
|      |      |            return (E_OK);
|      |      |            覆盖
|      |      |
|      +------ 1.6 函数 uipSystemCounterHandler
|      |      |      VOID uipSystemCounterHandler(VOID)
|      |      |
```

```
|       |       +------ 函数概要(MCDC):
|       |       |             3个真-假判定
|       |       |                   覆 盖 数: 3
|       |       |                   部分覆盖数: 0
|       |       |                   未 覆 盖: 0
|       |       |             3个条件判定
|       |       |                   覆 盖 数: 3
|       |       |                   部分覆盖数: 0
|       |       |                   未 覆 盖: 0
|       |       |             0个条件分支
|       |       |                   覆 盖 数: 0
|       |       |                   未 覆 盖: 0
|       |       |             2个覆盖事件
|       |       |                   覆 盖 数: 2
|       |       |                   未 覆 盖: 0
|       |       |
|       |       +------ coverage行 1004: 函数入口
|       |       |             {
|       |       |             覆盖
|       |       |
|       |       +------ decision行 1010: for语句
|       |       |             alarm_index < uipAlarmNumber
|       |       |
|       |       |                   T       F
|       |       |             01:    *t      *f(alarm_index < uipAlarmNumber)覆盖
|       |       |
|       |       +------ decision行 1014: if语句
|       |       |             alarmPtr -> uipAlarmState == uip_ALARM_ON
|       |       |
|       |       |                   T       F
|       |       |             01:    *t      *f(alarmPtr -> uipAlarmState == uip_ALARM_ON)覆盖
|       |       |
|       |       +------ decision行 1018: if语句
|       |       |             (alarmPtr -> uipCounterPtr) -> uipCtrValue == alarmPtr -> uipAlarmValue
|       |       |
|       |       |                   T       F
|       |       |             01:    *t      *f((alarmPtr -> uipCounterPtr) -> uipCtrValue == alarmPtr ->
uipAlarmValue)覆盖
|       |       |
|       |       +------ coverage行 1025: 函数结束
|       |       |                   }
|       |       |                   覆盖
|       |       |
|       |      +------ 1.7 函数 uipAlarmHandler
|       |       |       VOID uipAlarmHandler(uip_UINT alarm_index)
|       |       |
|       |       +------ 函数概要(MCDC):
|       |       |             5个真-假判定
|       |       |                   覆 盖 数: 4
|       |       |                   部分覆盖数: 1
|       |       |                   未 覆 盖: 0
```

```
|     |     |          5个条件判定
|     |     |               覆 盖 数: 4
|     |     |               部分覆盖数: 1
|     |     |               未 覆 盖: 0
|     |     |          0个条件分支
|     |     |               覆 盖 数: 0
|     |     |               未 覆 盖: 0
|     |     |          2个覆盖事件
|     |     |               覆 盖 数: 2
|     |     |               未 覆 盖: 0
|     |     |
|     |     +------ coverage行 1076: 函数入口
|     |     |          {
|     |     |          覆盖
|     |     |
|     |     +------ decision行 1083: if语句
|     |     |          alarmPtr -> uipCycle == 0
|     |     |
|     |     |               T      F
|     |     |          01:    *t     *f(alarmPtr -> uipCycle == 0)覆盖
|     |     |
|     |     +------ decision行 1090: if语句
|     |     |          (alarmPtr -> uipCycle) > ((((alarmPtr -> uipCounterPtr) -> uipCtrBase) ->
maxAllowedValue) -
                         ((alarmPtr -> uipCounterPtr) -> uipCtrValue))
|     |     |
|     |     |               T      F
|     |     |          01:    *t     *f((alarmPtr -> uipCycle) > ((((alarmPtr -> uipCounterPtr) -> uipCtrBase)
-> maxAllowedValue) -
                         ((alarmPtr -> uipCounterPtr) -> uipCtrValue)))覆盖
|     |     |
|     |     +------ decision行 1105: if语句
|     |     |          alarmPtr -> uipCallBack != 0
|     |     |
|     |     |               T      F
|     |     |          01:    *t     *f(alarmPtr -> uipCallBack != 0)覆盖
|     |     |
|     |     +------ decision行 1111: if语句
|     |     |          alarmPtr -> uipEvent != 0
|     |     |
|     |     |               T      F
|     |     |          01:    *t     *f(alarmPtr -> uipEvent != 0)覆盖
|     |     |
|     |     +------ decision行 1117: if语句
|     |     |          alarmPtr -> uipAlarmActTaskID != uipIdleTaskID
|     |     |
|     |     |               T      F
|     |     |          01:    *t      f(alarmPtr -> uipAlarmActTaskID != uipIdleTaskID)部分覆盖
|     |     |
|     |     +------ coverage行 1121: 函数结束
|     |               }
```

```
|     |                 覆盖
|     |
|     +------ 1.8 函数 uipGeneralCounterHandler
|     |     |      VOID uipGeneralCounterHandler(AlarmType AlarmID)
|     |     |
|     |     +------ 函数概要(MCDC):
|     |     |           2个真-假判定
|     |     |                覆 盖 数: 2
|     |     |                部分覆盖数: 0
|     |     |                未 覆 盖: 0
|     |     |           3个条件判定
|     |     |                覆 盖 数: 3
|     |     |                部分覆盖数: 0
|     |     |                未 覆 盖: 0
|     |     |           0个条件分支
|     |     |                覆 盖 数: 0
|     |     |                未 覆 盖: 0
|     |     |           2个覆盖事件
|     |     |                覆 盖 数: 2
|     |     |                未 覆 盖: 0
|     |     |
|     |     +------ coverage行 1171: 函数入口
|     |     |           {
|     |     |           覆盖
|     |     |
|     |     +------ decision行 1179: if语句中的条件表达式
|     |     |           (AlarmID <= uipAlarmNumber) && ((alarmPtr -> uipAlarmState) == uip_ALARM_ON)
|     |     |
|     |     |                T         F
|     |     |           01:   *tt       *fx((AlarmID <= uipAlarmNumber))覆盖
|     |     |           02:   *tt       *tf(((alarmPtr -> uipAlarmState) == uip_ALARM_ON))覆盖
|     |     |
|     |     +------ decision行 1182: if语句
|     |     |           (alarmPtr -> uipCounterPtr) -> uipCtrValue == alarmPtr -> uipAlarmValue
|     |     |
|     |     |                T      F
|     |     |           01:   *t     *f((alarmPtr -> uipCounterPtr) -> uipCtrValue == alarmPtr ->
uipAlarmValue)覆盖
|     |     |
|     |     +------ coverage行 1188: 函数结束
|     |                 }
|     |                 覆盖
|     |
|     +------ 1.9 函数 uipGetCtrCurrentValue
|           |      StatusType uipGetCtrCurrentValue(AlarmType AlarmID, TickRefType Tick)
|           |
|           +------ 函数概要(MCDC):
|           |           2个真-假判定
|           |                覆 盖 数: 2
|           |                部分覆盖数: 0
|           |                未 覆 盖: 0
```

```
|           |           2个条件判定
|           |                覆 盖 数: 2
|           |                部分覆盖数: 0
|           |                未 覆 盖: 0
|           |           0个条件分支
|           |                覆 盖 数: 0
|           |                未 覆 盖: 0
|           |           4个覆盖事件
|           |                覆 盖 数: 4
|           |                未 覆 盖: 0
|           |
|           +------ coverage行 1238: 函数入口
|           |           {
|           |           覆盖
|           |
|           +------ decision行 1242: if语句
|           |           AlarmID > uipAlarmNumber
|           |
|           |                T      F
|           |           01:   *t     *f(AlarmID > uipAlarmNumber)覆盖
|           |
|           +------ coverage行 1244: 函数返回语句
|           |           return (E_OS_ID);
|           |           覆盖
|           |
|           +------ decision行 1252: if语句
|           |           (alarmPtr -> uipAlarmState) == uip_ALARM_OFF
|           |
|           |                T      F
|           |           01:   *t     *f((alarmPtr -> uipAlarmState) == uip_ALARM_OFF)覆盖
|           |
|           +------ coverage行 1257: 函数返回语句
|           |           return (E_OS_NOFUNC);
|           |           覆盖
|           |
|           +------ coverage行 1266: 函数返回语句
|                       return (E_OK);
|                       覆盖
|
+------ 2. 文件 uip_kernel_event.c 目录位置 D:\UIP-Kernel\uipkernel\uipkernel_original\src
|       |       最后更新时间: Mon Jun 24 09:43:22 CST 2013
|       |       文件校验和: XXXXXX
|       |
|       +------ 文件概要(MCDC): 共4个函数
|       |           17个真-假判定
|       |                覆 盖 数: 17
|       |                部分覆盖数: 0
|       |                未 覆 盖: 0
|       |           19个条件判定
|       |                覆 盖 数: 19
|       |                部分覆盖数: 0
```

```
|     |                未 覆 盖: 0
|     |           0个条件分支
|     |                覆 盖 数: 0
|     |                未 覆 盖: 0
|     |           22个覆盖事件
|     |                覆 盖 数: 22
|     |                未 覆 盖: 0
|     |
|     +------ 2.1 函数 UIP_SetEvent
|     |     |      StatusType UIP_SetEvent(TaskType TaskID, EventMaskType Mask)
|     |     |
|     |     +------ 函数概要(MCDC):
|     |     |           6个真-假判定
|     |     |                覆 盖 数: 6
|     |     |                部分覆盖数: 0
|     |     |                未 覆 盖: 0
|     |     |           7个条件判定
|     |     |                覆 盖 数: 7
|     |     |                部分覆盖数: 0
|     |     |                未 覆 盖: 0
|     |     |           0个条件分支
|     |     |                覆 盖 数: 0
|     |     |                未 覆 盖: 0
|     |     |           6个覆盖事件
|     |     |                覆 盖 数: 6
|     |     |                未 覆 盖: 0
|     |     |
|     |     +------ coverage行 468: 函数入口
|     |     |           {
|     |     |           覆盖
|     |     |
|     |     +------ decision行 474: if语句
|     |     |           uipSchedulerLock != 0
|     |     |
|     |     |                T      F
|     |     |           01:   *t     *f(uipSchedulerLock != 0)覆盖
|     |     |
|     |     +------ coverage行 477: 函数返回语句
|     |     |           return (uip_SCHEDULER_LOCKED);
|     |     |           覆盖
|     |     |
|     |     +------ decision行 486: if语句中的条件表达式
|     |     |           (TaskID < uip_TASK_MAX) && (TaskID >= 0)
|     |     |
|     |     |                T         F
|     |     |           01:   *tt       *fx((TaskID < uip_TASK_MAX))覆盖
|     |     |           02:   *tt       *tf((TaskID >= 0))覆盖
|     |     |
|     |     +------ decision行 492: if语句
|     |     |           ((taskPtr -> uipTaskState) & uip_TASK_STATE_MASK) == SUSPENDED_STATE
|     |     |
```

```
|     |     |                 T     F
|     |     |          01:   *t     *f(((taskPtr -> uipTaskState) & uip_TASK_STATE_MASK) ==
SUSPENDED_STATE)覆盖
|     |     |
|     |     +------ coverage行 498: 函数返回语句
|     |     |          return (E_OS_STATE);
|     |     |          覆盖
|     |     |
|     |     +------ decision行 502: if语句
|     |     |          ((taskPtr -> uipTaskState) & uip_TASK_EXTENDED) == 0
|     |     |
|     |     |               T     F
|     |     |          01:   *t     *f(((taskPtr -> uipTaskState) & uip_TASK_EXTENDED) == 0)覆盖
|     |     |
|     |     +------ coverage行 508: 函数返回语句
|     |     |          return (E_OS_ACCESS);
|     |     |          覆盖
|     |     |
|     |     +------ decision行 514: if语句
|     |     |          ((taskPtr -> uipTaskState) & uip_TASK_STATE_MASK) == WAITING_STATE
|     |     |
|     |     |               T     F
|     |     |          01:   *t     *f(((taskPtr -> uipTaskState) & uip_TASK_STATE_MASK) ==
WAITING_STATE)覆盖
|     |     |
|     |     +------ decision行 517: if语句
|     |     |          ((taskPtr -> uipWaitEvent) & (taskPtr -> uipSetEvent)) != 0
|     |     |
|     |     |               T     F
|     |     |          01:   *t     *f(((taskPtr -> uipWaitEvent) & (taskPtr -> uipSetEvent)) != 0)覆盖
|     |     |
|     |     +------ coverage行 547: 函数返回语句
|     |     |          return (E_OK);
|     |     |          覆盖
|     |     |
|     |     +------ coverage行 554: 函数返回语句
|     |     |          return (E_OS_ID);
|     |     |          覆盖
|     |     |
|     +------ 2.2 函数 UIP_ClearEvent
|     |     |     StatusType UIP_ClearEvent(EventMaskType Mask)
|     |     |
|     |     +------ 函数概要(MCDC):
|     |     |          3个真-假判定
|     |     |               覆 盖 数: 3
|     |     |               部分覆盖数: 0
|     |     |               未 覆 盖: 0
|     |     |          3个条件判定
|     |     |               覆 盖 数: 3
|     |     |               部分覆盖数: 0
|     |     |               未 覆 盖: 0
```

```
|     |     |           0个条件分支
|     |     |                覆 盖 数: 0
|     |     |                未 覆 盖: 0
|     |     |           5个覆盖事件
|     |     |                覆 盖 数: 5
|     |     |                未 覆 盖: 0
|     |     |
|     |     +------ coverage行 611: 函数入口
|     |     |           {
|     |     |           覆盖
|     |     |
|     |     +------ decision行 616: if语句
|     |     |           uipNestedInterrupts != 0
|     |     |
|     |     |                T      F
|     |     |           01:   *t     *f(uipNestedInterrupts != 0)覆盖
|     |     |
|     |     +------ coverage行 619: 函数返回语句
|     |     |           return (E_OS_CALLEVEL);
|     |     |           覆盖
|     |     |
|     |     +------ decision行 623: if语句
|     |     |           uipSchedulerLock != 0
|     |     |
|     |     |                T      F
|     |     |           01:   *t     *f(uipSchedulerLock != 0)覆盖
|     |     |
|     |     +------ coverage行 626: 函数返回语句
|     |     |           return (uip_SCHEDULER_LOCKED);
|     |     |           覆盖
|     |     |
|     |     +------ decision行 635: if语句
|     |     |           ((taskCurrentPtr -> uipTaskState) & uip_TASK_EXTENDED) == 0
|     |     |
|     |     |                T      F
|     |     |           01:   *t     *f(((taskCurrentPtr -> uipTaskState) & uip_TASK_EXTENDED) == 0)覆盖
|     |     |
|     |     +------ coverage行 641: 函数返回语句
|     |     |           return (E_OS_ACCESS);
|     |     |           覆盖
|     |     |
|     |     +------ coverage行 650: 函数返回语句
|     |     |          return (E_OK);
|     |     |          覆盖
|     |
|     +------ 2.3 函数 UIP_GetEvent
|     |     |      StatusType UIP_GetEvent(TaskType TaskID, EventMaskRefType Mask)
|     |     |
|     |     +------ 函数概要(MCDC):
|     |     |           4个真-假判定
|     |     |                覆 盖 数: 4
```

```
|     |     |                 部分覆盖数: 0
|     |     |                 未 覆 盖: 0
|     |     |           5个条件判定
|     |     |                 覆 盖 数: 5
|     |     |                 部分覆盖数: 0
|     |     |                 未 覆 盖: 0
|     |     |           0个条件分支
|     |     |                 覆 盖 数: 0
|     |     |                 未 覆 盖: 0
|     |     |           6个覆盖事件
|     |     |                 覆 盖 数: 6
|     |     |                 未 覆 盖: 0
|     |     |
|     |     +------ coverage行 712: 函数入口
|     |     |           {
|     |     |           覆盖
|     |     |
|     |     +------ decision行 717: if语句
|     |     |           uipSchedulerLock != 0
|     |     |
|     |     |                 T       F
|     |     |           01:    *t      *f(uipSchedulerLock != 0)覆盖
|     |     |
|     |     +------ coverage行 720: 函数返回语句
|     |     |           return (uip_SCHEDULER_LOCKED);
|     |     |           覆盖
|     |     |
|     |     +------ decision行 729: if语句中的条件表达式
|     |     |           (TaskID < uip_TASK_MAX) && (TaskID >= 0)
|     |     |
|     |     |                 T         F
|     |     |           01:    *tt       *fx((TaskID < uip_TASK_MAX))覆盖
|     |     |           02:    *tt       *tf((TaskID >= 0))覆盖
|     |     |
|     |     +------ decision行 735: if语句
|     |     |           ((taskPtr -> uipTaskState) & uip_TASK_STATE_MASK) == SUSPENDED_STATE
|     |     |
|     |     |                 T       F
|     |     |           01:    *t      *f(((taskPtr -> uipTaskState) & uip_TASK_STATE_MASK) ==
SUSPENDED_STATE)覆盖
|     |     |
|     |     +------ coverage行 741: 函数返回语句
|     |     |           return (E_OS_STATE);
|     |     |           覆盖
|     |     |
|     |     +------ decision行 745: if语句
|     |     |           ((taskPtr -> uipTaskState) & uip_TASK_EXTENDED) == 0
|     |     |
|     |     |                 T       F
|     |     |           01:    *t      *f(((taskPtr -> uipTaskState) & uip_TASK_EXTENDED) == 0)覆盖
|     |     |
```

```
|       |       +------ coverage行 751: 函数返回语句
|       |       |             return (E_OS_ACCESS);
|       |       |             覆盖
|       |       |
|       |       +------ coverage行 761: 函数返回语句
|       |       |             return (E_OK);
|       |       |             覆盖
|       |       |
|       |       +------ coverage行 769: 函数返回语句
|       |                     return (E_OS_ID);
|       |                     覆盖
|       |
|       +------ 2.4 函数 UIP_WaitEvent
|               |       StatusType UIP_WaitEvent(EventMaskType Mask)
|               |
|               +------ 函数概要(MCDC):
|               |             4个真-假判定
|               |                   覆 盖 数: 4
|               |                   部分覆盖数: 0
|               |                   未 覆 盖: 0
|               |             4个条件判定
|               |                   覆 盖 数: 4
|               |                   部分覆盖数: 0
|               |                   未 覆 盖: 0
|               |             0个条件分支
|               |                   覆 盖 数: 0
|               |                   未 覆 盖: 0
|               |             5个覆盖事件
|               |                   覆 盖 数: 5
|               |                   未 覆 盖: 0
|               |
|               +------ coverage行 831: 函数入口
|               |             {
|               |             覆盖
|               |
|               +------ decision行 836: if语句
|               |             uipNestedInterrupts != 0
|               |
|               |                   T       F
|               |             01:    *t      *f(uipNestedInterrupts != 0)覆盖
|               |
|               +------ coverage行 839: 函数返回语句
|               |             return (E_OS_CALLEVEL);
|               |             覆盖
|               |
|               +------ decision行 843: if语句
|               |             uipSchedulerLock != 0
|               |
|               |                   T       F
|               |             01:    *t      *f(uipSchedulerLock != 0)覆盖
|               |
```

```
|           +------ coverage行 846: 函数返回语句
|           |           return (uip_SCHEDULER_LOCKED);
|           |           覆盖
|           |
|           +------ decision行 855: if语句
|           |           ((taskCurrentPtr -> uipTaskState) & uip_TASK_EXTENDED) == 0
|           |
|           |                 T       F
|           |           01:    *t      *f(((taskCurrentPtr -> uipTaskState) & uip_TASK_EXTENDED) == 0)覆盖
|           |
|           +------ coverage行 861: 函数返回语句
|           |           return (E_OS_ACCESS);
|           |           覆盖
|           |
|           +------ decision行 870: if语句
|           |           ((taskCurrentPtr -> uipSetEvent) & Mask) == 0
|           |
|           |                 T       F
|           |           01:    *t      *f(((taskCurrentPtr -> uipSetEvent) & Mask) == 0)覆盖
|           |
|           +------ coverage行 893: 函数返回语句
|                       return (E_OK);
|                       覆盖
|
+------ 3. 文件 uip_kernel_message.c 目录位置 D:\UIP-Kernel\uipkernel\uipkernel_original\src
|       |       最后更新时间: Mon Jun 24 09:43:18 CST 2013
|       |       文件校验和: XXXXXX
|       |
|       +------ 文件概要(MCDC): 共8个函数
|       |           30个真-假判定
|       |                 覆 盖 数: 27
|       |                 部分覆盖数: 3
|       |                 未 覆 盖: 0
|       |           32个条件判定
|       |                 覆 盖 数: 29
|       |                 部分覆盖数: 3
|       |                 未 覆 盖: 0
|       |           0个条件分支
|       |                 覆 盖 数: 0
|       |                 未 覆 盖: 0
|       |           30个覆盖事件
|       |                 覆 盖 数: 30
|       |                 未 覆 盖: 0
|       |
|       +------ 3.1 函数 UIP_SendMessage
|       |       |       StatusType UIP_SendMessage(MsgType MsgID, AppDataRef DataRef)
|       |       |
|       |       +------ 函数概要(MCDC):
|       |       |           14个真-假判定
|       |       |                 覆 盖 数: 13
|       |       |                 部分覆盖数: 1
```

```
|     |     |                未 覆 盖: 0
|     |     |          14个条件判定
|     |     |                覆 盖 数: 13
|     |     |                部分覆盖数: 1
|     |     |                未 覆 盖: 0
|     |     |          0个条件分支
|     |     |                覆 盖 数: 0
|     |     |                未 覆 盖: 0
|     |     |          7个覆盖事件
|     |     |                覆 盖 数: 7
|     |     |                未 覆 盖: 0
|     |     |
|     |     +------ coverage行 461: 函数入口
|     |     |          {
|     |     |          覆盖
|     |     |
|     |     +------ decision行 468: if语句
|     |     |          MsgID > uipMsgNumber
|     |     |
|     |     |                T      F
|     |     |          01:    *t     *f(MsgID > uipMsgNumber)覆盖
|     |     |
|     |     +------ coverage行 470: 函数返回语句
|     |     |          return (E_OS_ID);
|     |     |          覆盖
|     |     |
|     |     +------ decision行 476: if语句
|     |     |          msgPtr -> uipQueueWriteHead == (uip_UCHAR_PTR)DataRef
|     |     |
|     |     |                T      F
|     |     |          01:    *t     *f(msgPtr -> uipQueueWriteHead == (uip_UCHAR_PTR)DataRef)覆盖
|     |     |
|     |     +------ coverage行 479: 函数返回语句
|     |     |          return (uip_COM_BUFFER);
|     |     |          覆盖
|     |     |
|     |     +------ decision行 483: if语句
|     |     |          (msgPtr -> uipMsgStatus & uip_MSG_LOCKED) != 0
|     |     |
|     |     |                T      F
|     |     |          01:    *t     *f((msgPtr -> uipMsgStatus & uip_MSG_LOCKED) != 0)覆盖
|     |     |
|     |     +------ coverage行 486: 函数返回语句
|     |     |          return (uip_COM_LOCKED);
|     |     |          覆盖
|     |     |
|     |     +------ decision行 500: if语句
|     |     |          msgPtr -> uipMsgType == uip_MSG_UNQUEUED
|     |     |
|     |     |                T      F
|     |     |          01:    *t     *f(msgPtr -> uipMsgType == uip_MSG_UNQUEUED)覆盖
```

```
|     |     |
|     |     +------ decision行 502: for语句
|     |     |           data_size < msgPtr -> uipMsgSize
|     |     |
|     |     |                 T       F
|     |     |           01:    *t      *f(data_size < msgPtr -> uipMsgSize)覆盖
|     |     |
|     |     +------ decision行 513: if语句
|     |     |           ((msgPtr -> uipMsgStatus) & uip_MSG_OVERFLOW) == 0
|     |     |
|     |     |                 T       F
|     |     |           01:    *t      *f(((msgPtr -> uipMsgStatus) & uip_MSG_OVERFLOW) == 0)覆盖
|     |     |
|     |     +------ decision行 515: for语句
|     |     |           data_size < msgPtr -> uipMsgSize
|     |     |
|     |     |                 T       F
|     |     |           01:    *t      *f(data_size < msgPtr -> uipMsgSize)覆盖
|     |     |
|     |     +------ decision行 523: if语句
|     |     |           (uip_UCHAR_PTR)(msgPtr -> uipQueueWriteHead) >=
                        (uip_UCHAR_PTR)(msgPtr -> uipQueueEnd)
|     |     |
|     |     |                 T       F
|     |     |           01:    *t      *f((uip_UCHAR_PTR)(msgPtr -> uipQueueWriteHead) >=
                        (uip_UCHAR_PTR)(msgPtr -> uipQueueEnd))覆盖
|     |     |
|     |     +------ decision行 539: if语句
|     |     |           msgPtr -> uipMsgNotifyType == uip_COM_NOTIFICATION_CALLBACK
|     |     |
|     |     |                 T       F
|     |     |           01:    *t      *f(msgPtr -> uipMsgNotifyType ==
uip_COM_NOTIFICATION_CALLBACK)覆盖
|     |     |
|     |     +------ decision行 545: if语句
|     |     |           msgPtr -> uipMsgNotifyType == uip_COM_NOTIFICATION_FLAG
|     |     |
|     |     |                 T       F
|     |     |           01:    *t      *f(msgPtr -> uipMsgNotifyType == uip_COM_NOTIFICATION_FLAG)覆盖
|     |     |
|     |     +------ decision行 550: if语句
|     |     |           msgPtr -> uipMsgNotifyType == uip_COM_NOTIFICATION_SETEVENT
|     |     |
|     |     |                 T       F
|     |     |           01:    *t      *f(msgPtr -> uipMsgNotifyType ==
uip_COM_NOTIFICATION_SETEVENT)覆盖
|     |     |
|     |     +------ decision行 553: if语句
|     |     |           status != E_OK
|     |     |
|     |     |                 T       F
```

```
|       |       |           01:    *t      *f(status != E_OK)覆盖
|       |       |
|       |       +------ coverage行 559: 函数返回语句
|       |       |           return (status);
|       |       |           覆盖
|       |       |
|       |       +------ decision行 563: if语句
|       |       |           msgPtr -> uipMsgNotifyType == uip_COM_NOTIFICATION_ACTIVATETASK
|       |       |
|       |       |                T       F
|       |       |           01:    *t       f(msgPtr -> uipMsgNotifyType ==
uip_COM_NOTIFICATION_ACTIVATETASK)部分覆盖
|       |       |
|       |       +------ decision行 566: if语句
|       |       |           status != E_OK
|       |       |
|       |       |                T       F
|       |       |           01:    *t      *f(status != E_OK)覆盖
|       |       |
|       |       +------ coverage行 572: 函数返回语句
|       |       |           return (status);
|       |       |           覆盖
|       |       |
|       |       +------ coverage行 580: 函数返回语句
|       |                   return (E_OK);
|       |                   覆盖
|       |
|       +------ 3.2 函数 UIP_ReceiveMessage
|       |       |       StatusType UIP_ReceiveMessage(MsgType MsgID, AppDataRef DataRef)
|       |       |
|       |       +------ 函数概要(MCDC):
|       |       |           9个真-假判定
|       |       |                覆 盖 数: 9
|       |       |                部分覆盖数: 0
|       |       |                未 覆 盖: 0
|       |       |           10个条件判定
|       |       |                覆 盖 数: 10
|       |       |                部分覆盖数: 0
|       |       |                未 覆 盖: 0
|       |       |           0个条件分支
|       |       |                覆 盖 数: 0
|       |       |                未 覆 盖: 0
|       |       |           6个覆盖事件
|       |       |                覆 盖 数: 6
|       |       |                未 覆 盖: 0
|       |       |
|       |       +------ coverage行 642: 函数入口
|       |       |           {
|       |       |           覆盖
|       |       |
|       |       +------ decision行 650: if语句
```

```
|     |     |           MsgID > uipMsgNumber
|     |     |
|     |     |                T      F
|     |     |           01:   *t     *f(MsgID > uipMsgNumber)覆盖
|     |     |
|     |     +------ coverage行 652: 函数返回语句
|     |     |           return (E_OS_ID);
|     |     |           覆盖
|     |     |
|     |     +------ decision行 658: if语句
|     |     |           (uip_UCHAR_PTR)(msgPtr -> uipQueueReadHead) == (uip_UCHAR_PTR)DataRef
|     |     |
|     |     |                T      F
|     |     |           01:   *t     *f((uip_UCHAR_PTR)(msgPtr -> uipQueueReadHead) ==
(uip_UCHAR_PTR)DataRef)覆盖
|     |     |
|     |     +------ coverage行 661: 函数返回语句
|     |     |           return (uip_COM_BUFFER);
|     |     |           覆盖
|     |     |
|     |     +------ decision行 665: if语句
|     |     |           ((msgPtr -> uipMsgStatus) & uip_MSG_LOCKED) != 0
|     |     |
|     |     |                T      F
|     |     |           01:   *t     *f(((msgPtr -> uipMsgStatus) & uip_MSG_LOCKED) != 0)覆盖
|     |     |
|     |     +------ coverage行 668: 函数返回语句
|     |     |           return (uip_COM_LOCKED);
|     |     |           覆盖
|     |     |
|     |     +------ decision行 682: if语句
|     |     |           msgPtr -> uipMsgType == uip_MSG_UNQUEUED
|     |     |
|     |     |                T      F
|     |     |           01:   *t     *f(msgPtr -> uipMsgType == uip_MSG_UNQUEUED)覆盖
|     |     |
|     |     +------ decision行 684: for语句
|     |     |           data_size < msgPtr -> uipMsgSize
|     |     |
|     |     |                T      F
|     |     |           01:   *t     *f(data_size < msgPtr -> uipMsgSize)覆盖
|     |     |
|     |     +------ decision行 697: if语句中的条件表达式
|     |     |           (((msgPtr -> uipMsgStatus) & uip_MSG_OVERFLOW) == 0) &&
        ((msgPtr -> uipQueueReadHead) == (msgPtr -> uipQueueWriteHead))
|     |     |
|     |     |                T        F
|     |     |           01:   *tt      *fx((((msgPtr -> uipMsgStatus) & uip_MSG_OVERFLOW) == 0))覆盖
|     |     |           02:   *tt      *tf(((msgPtr -> uipQueueReadHead) == (msgPtr ->
uipQueueWriteHead)))覆盖
|     |     |
```

```
|     |     +------ coverage行 704: 函数返回语句
|     |     |           return (uip_COM_BUFFER_EMPTY);
|     |     |           覆盖
|     |     |
|     |     +------ decision行 707: if语句
|     |     |           ((msgPtr -> uipMsgStatus) & uip_MSG_OVERFLOW) != 0
|     |     |
|     |     |                T      F
|     |     |           01:   *t     *f(((msgPtr -> uipMsgStatus) & uip_MSG_OVERFLOW) != 0)覆盖
|     |     |
|     |     +------ decision行 714: for语句
|     |     |           data_size < msgPtr -> uipMsgSize
|     |     |
|     |     |                T      F
|     |     |           01:   *t     *f(data_size < msgPtr -> uipMsgSize)覆盖
|     |     |
|     |     +------ decision行 722: if语句
|     |     |           (uip_UCHAR_PTR)(msgPtr -> uipQueueReadHead) >=
|                         (uip_UCHAR_PTR)(msgPtr -> uipQueueEnd)
|     |     |
|     |     |                T      F
|     |     |           01:   *t     *f((uip_UCHAR_PTR)(msgPtr -> uipQueueReadHead) >=
|                         (uip_UCHAR_PTR)(msgPtr -> uipQueueEnd))覆盖
|     |     |
|     |     +------ coverage行 733: 函数返回语句
|     |               return (status);
|     |               覆盖
|     |
|     +------ 3.3 函数 UIP_GetMessageStatus
|     |     |     StatusType UIP_GetMessageStatus(MsgType MsgID)
|     |     |
|     |     +------ 函数概要(MCDC):
|     |     |           5个真-假判定
|     |     |                覆 盖 数: 3
|     |     |                部分覆盖数: 2
|     |     |                未 覆 盖: 0
|     |     |           6个条件判定
|     |     |                覆 盖 数: 4
|     |     |                部分覆盖数: 2
|     |     |                未 覆 盖: 0
|     |     |           0个条件分支
|     |     |                覆 盖 数: 0
|     |     |                未 覆 盖: 0
|     |     |           6个覆盖事件
|     |     |                覆 盖 数: 6
|     |     |                未 覆 盖: 0
|     |     |
|     |     +------ coverage行 792: 函数入口
|     |     |           {
|     |     |           覆盖
|     |     |
```

```
|       |       +------ decision行 796: if语句
|       |       |             MsgID > uipMsgNumber
|       |       |
|       |       |                   T       F
|       |       |             01:    *t      *f(MsgID > uipMsgNumber)覆盖
|       |       |
|       |       +------ coverage行 798: 函数返回语句
|       |       |             return (E_OS_ID);
|       |       |             覆盖
|       |       |
|       |       +------ decision行 804: if语句
|       |       |             ((msgPtr -> uipMsgStatus) & uip_MSG_LOCKED) != 0
|       |       |
|       |       |                   T       F
|       |       |             01:    *t      *f(((msgPtr -> uipMsgStatus) & uip_MSG_LOCKED) != 0)覆盖
|       |       |
|       |       +------ coverage行 807: 函数返回语句
|       |       |             return (uip_COM_LOCKED);
|       |       |             覆盖
|       |       |
|       |       +------ decision行 810: if语句
|       |       |             msgPtr -> uipMsgType == uip_MSG_QUEUED
|       |       |
|       |       |                   T       F
|       |       |             01:    *t      *f(msgPtr -> uipMsgType == uip_MSG_QUEUED)覆盖
|       |       |
|       |       +------ decision行 814: if语句中的条件表达式
|       |       |             (((msgPtr -> uipMsgStatus) & uip_MSG_OVERFLOW) == 0) &&
        (msgPtr -> uipQueueReadHead == msgPtr -> uipQueueWriteHead)
|       |       |
|       |       |                   T         F
|       |       |             01:    *tt       *fx((((msgPtr -> uipMsgStatus) & uip_MSG_OVERFLOW) == 0))覆盖
|       |       |             02:    *tt        tf((msgPtr -> uipQueueReadHead == msgPtr ->
uipQueueWriteHead))部分覆盖
|       |       |
|       |       +------ coverage行 818: 函数返回语句
|       |       |             return (uip_COM_BUFFER_EMPTY);
|       |       |             覆盖
|       |       |
|       |       +------ decision行 821: if语句
|       |       |             ((msgPtr -> uipMsgStatus) & uip_MSG_OVERFLOW) != 0
|       |       |
|       |       |                   T       F
|       |       |             01:    *t       f(((msgPtr -> uipMsgStatus) & uip_MSG_OVERFLOW) != 0)部分覆盖
|       |       |
|       |       +------ coverage行 823: 函数返回语句
|       |       |             return (uip_COM_OVERFLOW);
|       |       |             覆盖
|       |       |
|       |       +------ coverage行 827: 函数返回语句
|       |                     return (E_OK);
```

```
|     |                 覆盖
|     |
|     +------ 3.4 函数 UIP_ReadFlag
|     |     |     FlagType UIP_ReadFlag(MsgType MsgID)
|     |     |
|     |     +------ 函数概要(MCDC):
|     |     |           0个真-假判定
|     |     |                 覆 盖 数: 0
|     |     |                 部分覆盖数: 0
|     |     |                 未 覆 盖: 0
|     |     |           0个条件判定
|     |     |                 覆 盖 数: 0
|     |     |                 部分覆盖数: 0
|     |     |                 未 覆 盖: 0
|     |     |           0个条件分支
|     |     |                 覆 盖 数: 0
|     |     |                 未 覆 盖: 0
|     |     |           2个覆盖事件
|     |     |                 覆 盖 数: 2
|     |     |                 未 覆 盖: 0
|     |     |
|     |     +------ coverage行 873: 函数入口
|     |     |           {
|     |     |           覆盖
|     |     |
|     |     +------ coverage行 881: 函数返回语句
|     |                 return (value);
|     |                 覆盖
|     |
|     +------ 3.5 函数 UIP_ResetFlag
|     |     |     StatusType UIP_ResetFlag(MsgType MsgID)
|     |     |
|     |     +------ 函数概要(MCDC):
|     |     |           0个真-假判定
|     |     |                 覆 盖 数: 0
|     |     |                 部分覆盖数: 0
|     |     |                 未 覆 盖: 0
|     |     |           0个条件判定
|     |     |                 覆 盖 数: 0
|     |     |                 部分覆盖数: 0
|     |     |                 未 覆 盖: 0
|     |     |           0个条件分支
|     |     |                 覆 盖 数: 0
|     |     |                 未 覆 盖: 0
|     |     |           2个覆盖事件
|     |     |                 覆 盖 数: 2
|     |     |                 未 覆 盖: 0
|     |     |
|     |     +------ coverage行 931: 函数入口
|     |     |           {
|     |     |           覆盖
```

```
|       |       |
|       |       +------ coverage行 945: 函数返回语句
|       |       |               return (E_OK);
|       |       |               覆盖
|       |       |
|       +------ 3.6 函数 UIP_StartCOM
|       |       |       StatusType UIP_StartCOM(void)
|       |       |
|       |       +------ 函数概要(MCDC):
|       |       |       |       0个真-假判定
|       |       |       |               覆 盖 数: 0
|       |       |       |               部分覆盖数: 0
|       |       |       |               未 覆 盖: 0
|       |       |       |       0个条件判定
|       |       |       |               覆 盖 数: 0
|       |       |       |               部分覆盖数: 0
|       |       |       |               未 覆 盖: 0
|       |       |       |       0个条件分支
|       |       |       |               覆 盖 数: 0
|       |       |       |               未 覆 盖: 0
|       |       |       |       2个覆盖事件
|       |       |       |               覆 盖 数: 2
|       |       |       |               未 覆 盖: 0
|       |       |       |
|       |       |       +------ coverage行 998: 函数入口
|       |       |       |       {
|       |       |       |       覆盖
|       |       |       |
|       |       |       +------ coverage行 1004: 函数返回语句
|       |       |               return (returnVal);
|       |       |               覆盖
|       |       |
|       +------ 3.7 函数 UIP_CloseCOM
|       |       |       StatusType UIP_CloseCOM(void)
|       |       |
|       |       +------ 函数概要(MCDC):
|       |       |       |       0个真-假判定
|       |       |       |               覆 盖 数: 0
|       |       |       |               部分覆盖数: 0
|       |       |       |               未 覆 盖: 0
|       |       |       |       0个条件判定
|       |       |       |               覆 盖 数: 0
|       |       |       |               部分覆盖数: 0
|       |       |       |               未 覆 盖: 0
|       |       |       |       0个条件分支
|       |       |       |               覆 盖 数: 0
|       |       |       |               未 覆 盖: 0
|       |       |       |       2个覆盖事件
|       |       |       |               覆 盖 数: 2
|       |       |       |               未 覆 盖: 0
|       |       |       |
```

```
|       |       +------ coverage行 1053: 函数入口
|       |       |             {
|       |       |             覆盖
|       |       |
|       |       +------ coverage行 1057: 函数返回语句
|       |                     return (E_OK);
|       |                     覆盖
|       |
|       +------ 3.8 函数 UIP_StopCOM
|               |       StatusType UIP_StopCOM(uip_UCHAR ShutdownMode)
|               |
|               +------ 函数概要(MCDC):
|               |             2个真-假判定
|               |                   覆 盖 数: 2
|               |                   部分覆盖数: 0
|               |                   未 覆 盖: 0
|               |             2个条件判定
|               |                   覆 盖 数: 2
|               |                   部分覆盖数: 0
|               |                   未 覆 盖: 0
|               |             0个条件分支
|               |                   覆 盖 数: 0
|               |                   未 覆 盖: 0
|               |             3个覆盖事件
|               |                   覆 盖 数: 3
|               |                   未 覆 盖: 0
|               |
|               +------ coverage行 1113: 函数入口
|               |             {
|               |             覆盖
|               |
|               +------ decision行 1117: for语句
|               |             msgNumber < uipMsgNumber
|               |
|               |                   T       F
|               |             01:    *t      *f(msgNumber < uipMsgNumber)覆盖
|               |
|               +------ decision行 1121: if语句
|               |             ((msgPtr -> uipMsgStatus) & (uip_MSG_LOCKED)) != 0
|               |
|               |                   T       F
|               |             01:    *t      *f(((msgPtr -> uipMsgStatus) & (uip_MSG_LOCKED)) != 0)覆盖
|               |
|               +------ coverage行 1123: 函数返回语句
|               |             return (uip_COM_LOCKED);
|               |             覆盖
|               |
|               +------ coverage行 1127: 函数返回语句
|                           return (E_OK);
|                           覆盖
|
```

```
+------ 4. 文件 uip_kernel_resource.c 目录位置 D:\UIP-Kernel\uipkernel\uipkernel_original\src
|      |      最后更新时间: Mon Jun 24 09:43:13 CST 2013
|      |      文件校验和: XXXXXX
|      |
|      +------ 文件概要(MCDC): 共2个函数
|      |            9个真-假判定
|      |                  覆 盖 数: 8
|      |                  部分覆盖数: 1
|      |                  未 覆 盖: 0
|      |            10个条件判定
|      |                  覆 盖 数: 9
|      |                  部分覆盖数: 1
|      |                  未 覆 盖: 0
|      |            0个条件分支
|      |                  覆 盖 数: 0
|      |                  未 覆 盖: 0
|      |            12个覆盖事件
|      |                  覆 盖 数: 12
|      |                  未 覆 盖: 0
|      |
|      +------ 4.1 函数 uipGetResource
|      |      |      StatusType uipGetResource(ResourceType ResID)
|      |      |
|      |      +------ 函数概要(MCDC):
|      |      |            4个真-假判定
|      |      |                  覆 盖 数: 4
|      |      |                  部分覆盖数: 0
|      |      |                  未 覆 盖: 0
|      |      |            4个条件判定
|      |      |                  覆 盖 数: 4
|      |      |                  部分覆盖数: 0
|      |      |                  未 覆 盖: 0
|      |      |            0个条件分支
|      |      |                  覆 盖 数: 0
|      |      |                  未 覆 盖: 0
|      |      |            6个覆盖事件
|      |      |                  覆 盖 数: 6
|      |      |                  未 覆 盖: 0
|      |      |
|      |      +------ coverage行 479: 函数入口
|      |      |            {
|      |      |            覆盖
|      |      |
|      |      +------ decision行 491: if语句
|      |      |            uipNestedInterrupts != 0
|      |      |
|      |      |                  T      F
|      |      |            01:    *t     *f(uipNestedInterrupts != 0)覆盖
|      |      |
|      |      +------ coverage行 494: 函数返回语句
|      |      |            return (E_OS_CALLEVEL);
```

```
|       |       |           覆盖
|       |       |
|       |       +------ decision行 499: if语句
|       |       |           ResID >= uipResNumber
|       |       |
|       |       |                T       F
|       |       |           01:    *t      *f(ResID >= uipResNumber)覆盖
|       |       |
|       |       +------ coverage行 502: 函数返回语句
|       |       |           return (E_OS_ID);
|       |       |           覆盖
|       |       |
|       |       +------ decision行 512: if语句
|       |       |           (resourcePtr -> uipLock) == 1
|       |       |
|       |       |                T       F
|       |       |           01:    *t      *f((resourcePtr -> uipLock) == 1)覆盖
|       |       |
|       |       +------ coverage行 518: 函数返回语句
|       |       |           return (E_OS_ACCESS);
|       |       |           覆盖
|       |       |
|       |       +------ decision行 526: if语句
|       |       |           uipCurrentTaskPriority < (resourcePtr -> uipCeilingPrio)
|       |       |
|       |       |                T       F
|       |       |           01:    *t      *f(uipCurrentTaskPriority < (resourcePtr -> uipCeilingPrio))覆盖
|       |       |
|       |       +------ coverage行 532: 函数返回语句
|       |       |           return (E_OS_ACCESS);
|       |       |           覆盖
|       |       |
|       |       +------ coverage行 560: 函数返回语句
|       |       |           return (E_OK);
|       |       |           覆盖
|       |       |
|       |       +------ 4.2 函数 uipReleaseResource
|       |       |       StatusType uipReleaseResource(ResourceType ResID)
|       |       |
|       |       +------ 函数概要(MCDC):
|       |       |           5个真-假判定
|       |       |                覆 盖 数: 4
|       |       |                部分覆盖数: 1
|       |       |                未 覆 盖: 0
|       |       |           6个条件判定
|       |       |                覆 盖 数: 5
|       |       |                部分覆盖数: 1
|       |       |                未 覆 盖: 0
|       |       |           0个条件分支
|       |       |                覆 盖 数: 0
|       |       |                未 覆 盖: 0
```

```
|           |           6个覆盖事件
|           |                覆 盖 数: 6
|           |                未 覆 盖: 0
|           |
|           +------ coverage行 624: 函数入口
|           |           {
|           |           覆盖
|           |
|           +------ decision行 635: if语句
|           |           uipNestedInterrupts != 0
|           |
|           |                T      F
|           |           01:   *t     *f(uipNestedInterrupts != 0)覆盖
|           |
|           +------ coverage行 641: 函数返回语句
|           |           return (E_OS_CALLEVEL);
|           |           覆盖
|           |
|           +------ decision行 646: if语句
|           |           ResID >= uipResNumber
|           |
|           |                T      F
|           |           01:   *t     *f(ResID >= uipResNumber)覆盖
|           |
|           +------ coverage行 649: 函数返回语句
|           |           return (E_OS_ID);
|           |           覆盖
|           |
|           +------ decision行 661: if语句中的条件表达式
|           |           ((resourcePtr -> uipLock) == 0) || ((taskCurrentPtr -> uipResources) != resourcePtr)
|           |
|           |                T         F
|           |           01:   *tx       *ff(((resourcePtr -> uipLock) == 0))覆盖
|           |           02:   *ft       *ff(((taskCurrentPtr -> uipResources) != resourcePtr))覆盖
|           |
|           +------ coverage行 667: 函数返回语句
|           |           return (E_OS_NOFUNC);
|           |           覆盖
|           |
|           +------ decision行 673: if语句
|           |           uipCurrentTaskPriority < (resourcePtr -> uipCeilingPrio)
|           |
|           |                T      F
|           |           01:   *t     *f(uipCurrentTaskPriority < (resourcePtr -> uipCeilingPrio))覆盖
|           |
|           +------ coverage行 679: 函数返回语句
|           |           return (E_OS_ACCESS);
|           |           覆盖
|           |
|           +------ decision行 696: if语句
|           |           uipPrioGrpArray[resourcePtr -> uipResCeilingGrpArrayIndex] == 0x00
```

```
|           |
|           |                T      F
|           |           01:    *t       f(uipPrioGrpArray[resourcePtr -> uipResCeilingGrpArrayIndex] ==
0x00)部分覆盖
|           |
|           +------ coverage行 713: 函数返回语句
|                       return (E_OK);
|                       覆盖
|
+------ 5. 文件 uip_kernel_task.c 目录位置 D:\UIP-Kernel\uipkernel\uipkernel_original\src
      |     最后更新时间: Mon Jun 24 09:43:07 CST 2013
      |     文件校验和: XXXXXX
      |
      +------ 文件概要(MCDC): 共5个函数
      |           21个真-假判定
      |                 覆 盖 数: 15
      |                 部分覆盖数: 6
      |                 未 覆 盖: 0
      |           25个条件判定
      |                 覆 盖 数: 18
      |                 部分覆盖数: 7
      |                 未 覆 盖: 0
      |           0个条件分支
      |                 覆 盖 数: 0
      |                 未 覆 盖: 0
      |           19个覆盖事件
      |                 覆 盖 数: 19
      |                 未 覆 盖: 0
      |
      +------ 5.1 函数 UIP_ActivateTask
      |     |      StatusType UIP_ActivateTask(TaskType TaskID)
      |     |
      |     +------ 函数概要(MCDC):
      |     |           4个真-假判定
      |     |                 覆 盖 数: 3
      |     |                 部分覆盖数: 1
      |     |                 未 覆 盖: 0
      |     |           6个条件判定
      |     |                 覆 盖 数: 4
      |     |                 部分覆盖数: 2
      |     |                 未 覆 盖: 0
      |     |           0个条件分支
      |     |                 覆 盖 数: 0
      |     |                 未 覆 盖: 0
      |     |           4个覆盖事件
      |     |                 覆 盖 数: 4
      |     |                 未 覆 盖: 0
      |     |
      |     +------ coverage行 465: 函数入口
      |     |           {
      |     |           覆盖
```

```
|   |
|   +------ decision行 475: if语句中的条件表达式
|   |           (TaskID < uip_TASK_MAX) && (TaskID >= 0)
|   |
|   |               T        F
|   |           01:   *tt       *fx((TaskID < uip_TASK_MAX))覆盖
|   |           02:   *tt       *tf((TaskID >= 0))覆盖
|   |
|   +------ decision行 482: if语句
|   |           ((taskPtr -> uipTaskState) & uip_TASK_STATE_MASK) == SUSPENDED_STATE
|   |
|   |               T      F
|   |           01:   *t     *f(((taskPtr -> uipTaskState) & uip_TASK_STATE_MASK) ==
SUSPENDED_STATE)覆盖
|   |
|   +------ decision行 503: if语句中的条件表达式
|   |           (uipNestedInterrupts == 0) && (uipSchedulerLock == 0)
|   |
|   |               T        F
|   |           01:   *tt       fx((uipNestedInterrupts == 0))部分覆盖
|   |           02:   *tt       tf((uipSchedulerLock == 0))部分覆盖
|   |
|   +------ decision行 512: if语句
|   |           (taskPtr -> uipActivatedNumber) < ACTIVATED_NUM_MAX
|   |
|   |               T      F
|   |           01:   *t     *f((taskPtr -> uipActivatedNumber) < ACTIVATED_NUM_MAX)覆盖
|   |
|   +------ coverage行 523: 函数返回语句
|   |           return (E_OS_LIMIT);
|   |           覆盖
|   |
|   +------ coverage行 531: 函数返回语句
|   |           return (E_OK);
|   |           覆盖
|   |
|   +------ coverage行 538: 函数返回语句
|               return (E_OS_ID);
|               覆盖
|
+------ 5.2 函数 UIP_TerminateTask
|   |     StatusType UIP_TerminateTask (VOID)
|   |
|   +------ 函数概要(MCDC):
|   |           6个真-假判定
|   |               覆 盖 数: 4
|   |               部分覆盖数: 2
|   |               未 覆 盖: 0
|   |           6个条件判定
|   |               覆 盖 数: 4
|   |               部分覆盖数: 2
```

```
|     |                未 覆 盖: 0
|     |          0个条件分支
|     |                覆 盖 数: 0
|     |                未 覆 盖: 0
|     |          4个覆盖事件
|     |                覆 盖 数: 4
|     |                未 覆 盖: 0
|     |
|     +------ coverage行 601: 函数入口
|     |          {
|     |          覆盖
|     |
|     +------ decision行 607: if语句
|     |          uipNestedInterrupts != 0
|     |
|     |                T      F
|     |          01:   *t     *f(uipNestedInterrupts != 0)覆盖
|     |
|     +------ coverage行 610: 函数返回语句
|     |          return (E_OS_CALLEVEL);
|     |          覆盖
|     |
|     +------ decision行 613: if语句
|     |          uipSchedulerLock != 0
|     |
|     |                T      F
|     |          01:   *t     *f(uipSchedulerLock != 0)覆盖
|     |
|     +------ coverage行 616: 函数返回语句
|     |          return (uip_SCHEDULER_LOCKED);
|     |          覆盖
|     |
|     +------ decision行 626: if语句
|     |          (taskPtr -> uipResources) != 0
|     |
|     |                T      F
|     |          01:   *t     *f((taskPtr -> uipResources) != 0)覆盖
|     |
|     +------ coverage行 632: 函数返回语句
|     |          return (E_OS_RESOURCE);
|     |          覆盖
|     |
|     +------ decision行 651: if语句
|     |          (taskPtr -> uipActivatedNumber) == 0
|     |
|     |                T      F
|     |          01:   *t     *f((taskPtr -> uipActivatedNumber) == 0)覆盖
|     |
|     +------ decision行 668: if语句
|     |          (taskPtr -> uipDispatchPrio) != (taskPtr -> uipReadyPrio)
|     |
```

```
|     |                 T      F
|     |           01:   *t      f((taskPtr -> uipDispatchPrio) != (taskPtr -> uipReadyPrio))部分覆盖
|     |
|     +------ decision行 673: if语句
|               uipPrioGrpArray[taskPtr -> uipDispPrioGrpArrayIndex] == 0x00
|
|                     T      F
|               01:   *t      f(uipPrioGrpArray[taskPtr -> uipDispPrioGrpArrayIndex] == 0x00)部分覆
盖
|
+------ 5.3 函数 UIP_ChainTask
|     |     StatusType UIP_ChainTask (TaskType TaskID)
|     |
|     +------ 函数概要(MCDC):
|     |           9个真-假判定
|     |                 覆 盖 数: 6
|     |                 部分覆盖数: 3
|     |                 未 覆 盖: 0
|     |           10个条件判定
|     |                 覆 盖 数: 7
|     |                 部分覆盖数: 3
|     |                 未 覆 盖: 0
|     |           0个条件分支
|     |                 覆 盖 数: 0
|     |                 未 覆 盖: 0
|     |           6个覆盖事件
|     |                 覆 盖 数: 6
|     |                 未 覆 盖: 0
|     |
|     +------ coverage行 780: 函数入口
|     |           {
|     |           覆盖
|     |
|     +------ decision行 786: if语句
|     |           uipNestedInterrupts != 0
|     |
|     |                 T      F
|     |           01:   *t     *f(uipNestedInterrupts != 0)覆盖
|     |
|     +------ coverage行 789: 函数返回语句
|     |           return (E_OS_CALLEVEL);
|     |           覆盖
|     |
|     +------ decision行 792: if语句
|     |           uipSchedulerLock != 0
|     |
|     |                 T      F
|     |           01:   *t     *f(uipSchedulerLock != 0)覆盖
|     |
|     +------ coverage行 795: 函数返回语句
|     |           return (uip_SCHEDULER_LOCKED);
```

```
|     |           覆盖
|     |
|     +------ decision行 804: if语句中的条件表达式
|     |           (TaskID < uip_TASK_MAX) && (TaskID >= 0)
|     |
|     |                T         F
|     |           01:   *tt       *fx((TaskID < uip_TASK_MAX))覆盖
|     |           02:   *tt       *tf((TaskID >= 0))覆盖
|     |
|     +------ decision行 810: if语句
|     |           (taskCurrentPtr -> uipResources) != 0
|     |
|     |                T      F
|     |           01:   *t     *f((taskCurrentPtr -> uipResources) != 0)覆盖
|     |
|     +------ coverage行 816: 函数返回语句
|     |           return (E_OS_RESOURCE);
|     |           覆盖
|     |
|     +------ decision行 824: if语句
|     |           taskPtr != uipCurrentTaskTCBPtr
|     |
|     |                T      F
|     |           01:   *t     *f(taskPtr != uipCurrentTaskTCBPtr)覆盖
|     |
|     +------ decision行 827: if语句
|     |           ((taskPtr -> uipTaskState) & uip_TASK_STATE_MASK) == SUSPENDED_STATE
|     |
|     |                T      F
|     |           01:   *t     *f(((taskPtr -> uipTaskState) & uip_TASK_STATE_MASK) ==
SUSPENDED_STATE)覆盖
|     |
|     +------ coverage行 853: 函数返回语句
|     |           return (E_OS_LIMIT);
|     |           覆盖
|     |
|     +------ decision行 885: if语句
|     |           (taskCurrentPtr -> uipDispatchPrio) != (taskCurrentPtr -> uipReadyPrio)
|     |
|     |                T      F
|     |           01:   *t      f((taskCurrentPtr -> uipDispatchPrio) != (taskCurrentPtr ->
uipReadyPrio))部分覆盖
|     |
|     +------ decision行 890: if语句
|     |           uipPrioGrpArray[taskCurrentPtr -> uipDispPrioGrpArrayIndex] == 0x00
|     |
|     |                T      F
|     |           01:   *t      f(uipPrioGrpArray[taskCurrentPtr -> uipDispPrioGrpArrayIndex] ==
0x00)部分覆盖
|     |
|     +------ decision行 915: if语句
```

```
|      |            (taskCurrentPtr -> uipActivatedNumber) != 0
|      |
|      |                 T      F
|      |            01:   *t      f((taskCurrentPtr -> uipActivatedNumber) != 0)部分覆盖
|      |
|      +------ coverage行 930: 函数返回语句
|                 return (E_OS_ID);
|                 覆盖
|
+------ 5.4 函数 UIP_GetTaskID
|      |      StatusType UIP_GetTaskID (TaskRefType TaskID)
|      |
|      +------ 函数概要(MCDC):
|      |            1个真-假判定
|      |                 覆 盖 数: 1
|      |                 部分覆盖数: 0
|      |                 未 覆 盖: 0
|      |            1个条件判定
|      |                 覆 盖 数: 1
|      |                 部分覆盖数: 0
|      |                 未 覆 盖: 0
|      |            0个条件分支
|      |                 覆 盖 数: 0
|      |                 未 覆 盖: 0
|      |            2个覆盖事件
|      |                 覆 盖 数: 2
|      |                 未 覆 盖: 0
|      |
|      +------ coverage行 986: 函数入口
|      |            {
|      |            覆盖
|      |
|      +------ decision行 992: if语句
|      |            taskCurrentPtr == uip_NULL
|      |
|      |                 T      F
|      |            01:   *t     *f(taskCurrentPtr == uip_NULL)覆盖
|      |
|      +------ coverage行 1002: 函数返回语句
|                 return (E_OK);
|                 覆盖
|
+------ 5.5 函数 UIP_GetTaskState
       |      StatusType UIP_GetTaskState (TaskType TaskID, TaskStateRefType State)
       |
      +------ 函数概要(MCDC):
       |            1个真-假判定
       |                 覆 盖 数: 1
       |                 部分覆盖数: 0
       |                 未 覆 盖: 0
       |            2个条件判定
```

```
|                   覆 盖 数: 2
|                   部分覆盖数: 0
|                   未 覆 盖: 0
|             0个条件分支
|                   覆 盖 数: 0
|                   未 覆 盖: 0
|             3个覆盖事件
|                   覆 盖 数: 3
|                   未 覆 盖: 0
|
+------ coverage行 1063: 函数入口
|             {
|             覆盖
|
+------ decision行 1073: if语句中的条件表达式
|             (TaskID < uip_TASK_MAX) && (TaskID >= 0)
|
|                   T         F
|             01:    *tt        *fx((TaskID < uip_TASK_MAX))覆盖
|             02:    *tt        *tf((TaskID >= 0))覆盖
|
+------ coverage行 1084: 函数返回语句
|             return (E_OK);
|             覆盖
|
+------ coverage行 1091: 函数返回语句
            return (E_OS_ID);
            覆盖
```