The following are examples of utilizing ddb. In these examples the system is a C3820. All information after the spu> and [DDB]-> prompts is entered by you.

1. This example shows how to run selected subtests.

```
spu> ddb
cpu:0,cir:  0, tid:  0
[DDB] -> $test cpu4332 0
loading p0r0_4332.ldr at logical offset 0x00000000
loading cpu4332.ldr at logical offset 0x00022000
Test loading completed, initializing Comm Regs
Comm Reg initialization completed, setting up SDRs
cpu:0,cir: 0,tid:  0
[DDB] -> $status              (to obtain test status menu)

        +--------------------------------+
        | Memory        Stop     Bkpt    |
        |  Mode         Mode     Mode    |
        +--------------------------------+
        | LOGICAL      NORMAL   DIAG     |
        +--------------------------------+
        +--------------------------------+
        | CPUID      CPUSTATE  BKPTMODE  |
        +--------------------------------+
        |  CPU0       HALTED    IDLE     |
        |  CPU1       HALTED    IDLE     |
        |  CPU2       UNINST   PARKED    |
        |  CPU3       UNINST   PARKED    |
        |  CPU4       UNINST   PARKED    |
        |  CPU5       UNINST   PARKED    |
        |  CPU6       UNINST   PARKED    |
        |  CPU7       UNINST   PARKED    |
        +--------------------------------+
     Test name = 'cpu4332';
  data_ffaults = 'OFF'           ip_ffaults = 'OFF'
  enable_dcache = 'ON'           dcache_resize = 'ON'
  seq_mode = 'OFF'               chain_mode = 'OFF'
  parallel_mode = 'OFF'          multi_cirs = 'ON'
  scn_ovr = 'ON'                 vl_count = '16'
  secure_mode = 'OFF'
cpu:0,cir: 0,tid:  0
[DDB] -> $park 1          (This will prevent cpu1 from being tested)

$park: CPU1 placed into parked mode
cpu:0,cir:  0,tid:  0
[DDB] -> $run 0           (CPU0 will be tested)
$run:  CPU0 placed into run mode
cpu:0,cir:  0,tid:  0
[DDB] -> st_100:r     (To start subtest 100 of this test)
CPU0 halt:  a1=00000100
        cputest_complete+0xe: halt      #0x100,a1
cpu:0,cir: 0,tid:0
[DDB]-> st_271:r      (To start subtest 271 of this test)
CPU0 halt:  a1=00000100
        cputest_complete+0xe: halt      #0x100,a1
cpu:0,cir: 0, tid: 0
[DDB]-> $q                  (To exit test)
spu>
```

(Note: If a1=00000100, the the test passed. If it is equal to anyother value, the test failed and at this point you can look at the cpureg by: [DDB]-> $R).

2. The following example shows how to initiate the complete test for cpu1, in the chained mode.

```
spu> ddb
cpu:0, cir: 0,tid: 0
[DDB]-> $test cpu4332 0
loading p0r0_4332.ldr at logical offset 0x00000000
loading cpu4332.ldr at logical offset 0x00022000
Test loading completed, initializing Comm Regs
Comm Reg initialization completed, setting up SDRs
cpu:0,cir: 0,tid: 0
[DDB]-> $run 1           (CPU1 will be tested)
$run; CPU1 placed into run mode
cpu:0,cir: 0,tid: 0
[DDB]-> $park 0          (CPU0 will not be tested)
$park: CPU0 placed into parked mode
cpu:0,cir: 0,tid 0
[DDB]-> $chain_mode 1    (activates chain mode)
chain_mode = 'ON'
cpu:0,cir: 0,tid: 0
[DDB]-> Chain_entrypoint:r    (Note: no $ - C is upper case)

NOTE: This test in the chained mode should have completed within
five minutes.  As it did not, the test was stopped and a get
was performed on hard_err1.

\032[1] + Stopped            ddb
spu> get hard_err1
hard_err1
VALUE: 16#00004000
 | 31 | 30 | 29 | 28 # 27 | 26 | 25 | 24 # 23 | 22 | 21 | 20 # 19 | 18 | 17 | 1
6
 |reserved |          BAY 3              #          BAY 2             | Base I
/O

 | 15 | 14 | 13 | 12 # 11 | 10 |  9 |  8 #  7 |  6 |  5 |  4 #  3 |  2 |  1 |
0
 |   Base I/O LSB    #          BAY 1             |        BAY 0
```

3. The following is an example of a cpu4030 failure. The test was run only on one cpu with normal defaults.

```
spu> ddb
cpu:6,cir: 6,tid: 0
[DDB]-> $test cpu4030 0
loading p0r0_4030.ldr at logical offset 0x00000000
                .
                .
loading wrapl_4030.ldr at logical offset 0xe0000000
Test loading completed, initializing Comm Regs
Comm Reg initialization completed, setting up SDRs
cpu:6,cir: 6,tid: 0
[DDB]-> $park 7
$park: CPU7 placed into parked mode
cpu:6,cir: 6,tid: 0
[DDB]-> $run 6
$run: CPU6 placed into run mode
cpu:6,cir: 6,tid: 0
[DDB]-> :r
****
CPU6 halt:  a1=00005008  FAILED
        l_st_305+0x2: halt       #0x5008,a1
```

```
cpu:6,cir: 6,tid: 0
[DDB]-> $r
Register state for CPU 6
pc=000000e0 upc=00000bfd cir=00000006 tid=00000000 ccr=80000020
ps=00000000  (XF)
sp=00002f90 a1=00005008 a2=0000aabb a3=0000ccdd
a4=0000eeff a5=00000000 ap=00008899 fp=00002f90
s0=000000a000000020 s1=0000000043800000 s2=000014484a742404 s3=000000004a742403
s4=33333333ca742404 s5=0000000055555555 s6=00000000aaaaaaaa s7=000000000000ffff
t0=00000000 t1=00000000 t2=80000020 t3=00000000
t4=00000000 t5=00000001 t6=00000008 t7=00000002
v1=00000000 vs=00000000 vv=00000001
vm=00000000000000000000000000000000
cpu:6,cir: 6,tid: 0
[DDB]-> $q
spu> get hard_err1
hard_err1
VALUE: 16#00000000
| 31 | 30 | 29 | 28 # 27 | 26 | 25 | 24 # 23 | 22 | 21 | 20 # 19 | 18 | 17 | 16
|reserved |          BAY 3              #          BAY 2              | Base I/
O

| 15 | 14 | 13 | 12 # 11 | 10 |  9 |  8 #  7 |  6 |  5 |  4 #  3 |  2 |  1 |  0
|   Base I/O LSB    #           BAY 1             |        BAY 0

spu> get hard_err2
hard_err2
VALUE: 16#00000000
| 31 | 30 | 29 | 28 # 27 | 26 | 25 | 24 # 23 | 22 | 21 | 20 # 19 | 18 | 17 | 16
|                 Reserved          |        IA Soft Errors

| 15 | 14 | 13 | 12 # 11 | 10 |  9 |  8 #  7 |  6 |  5 |  4 #  3 |  2 |  1 |  0
|               MB Soft Errors          #Rsvd|           CROSSBAR

spu> get scalar_halt
scalar_halt
VALUE: 16#00400040
| 31 | 30 | 29 | 28 # 27 | 26 | 25 | 24 # 23 | 22 | 21 | 20 # 19 | 18 | 17 | 16
|               Reserved                #        SCALAR HW HUNG

| 15 | 14 | 13 | 12 # 11 | 10 |  9 |  8 #  7 |  6 |  5 |  4 #  3 |  2 |  1 |  0
|               HALT MASK               #             HALT


*SP6 hung in the above test.*


4. The following is an example of a cpu4041 test in the chained mode.

spu> ddb
cpu:6,cir: 6,tid: 0
[DDB]-> $test cpu4041 0
loading p0r0_4041.ldr at logical offset 0x00000000
loading cpu4041.ldr at logical offset 0x00008000
Test loading completed, initializing Comm Regs
Comm Reg initialization completed, setting up SDRs
cpu:6,cir: 6,tid: 0
[DDB]-> $run 6
$run: CPU6 placed into run mode
cpu:6,cir: 6,tid: 0
[DDB]-> $park 7
$park: CPU7 placed into parked mode
cpu:6,cir: 6,tid: 0
[DDB]-> $chain_mode 1
chain_mode = 'ON'
cpu:6,cir: 6,tid: 0
[DDB]-> Chain_entrypoint:r
CPU6 halt:  a1=00000100  PASSED
        Chain_entrypoint_end: halt       #0x100,a1
cpu:6,cir: 6,tid: 0
[DDB]-> $q
* * * * * Last command returned status 1 * * * * *
spu>
```