A/UX: Changing Kernel Parameters (6/93)

Article Created: 14 June 1990
Article Reviewed/Updated: 25 June 1993

TOPIC ---

Where can I find information on kernel parameters? How can I change kernel parameters?

DISCUSSION ---

Beyond minor references in the A/UX documentation suite, there are two places to find information on kernel parameters:

- See the man page for kconfig(1M).

- See kconfig(1M) in the "A/UX System Administration's Reference"

Type this command to get a list of current Kernel parameters:

```
kconfig -av /unix
```

The MSGxxx parameters (the ones that deal with messages) and SHMxxx parameters (the ones that deal with shared memory) cannot be viewed or modified via the "kconfig" command, because they are defined and initialized separately inside the kernel. But, by using "adb" (a kernel debugger), you may be able to change these parameters. The last part of this article describes how to change the kernel parameters for the Shared Memory, Messaging Operations, and Semaphores within A/UX 1.0.

The Shared Memory, Semaphore, and Message Operation structures are defined in:

```
/usr/include/sys/shm.h    Shared Memory
/usr/include/sys/sem.h    Semaphore
/usr/include/sys/msg.h    Message Operations
```

The structure members are initialized in /usr/include/sys/space.h. The values used for initialization are defined in two locations:

```
/usr/include/sys/config.h
../psn/cf/bnetconfig.h
```

The bnetconfig.h file is found only in the source code and is the last file to be read in the build process, so it "overrides" the values in the config.h file. The values listed within this article are those found in bnetconfig.h. In the original version of this article, we verified them by looking at the virgin A/UX 1.0 binary (/newunix).

If you have a version of A/UX prior to Release 3.0.1, you can change parameters with ADB. In A/UX 3.0.1, you can use kconig to set share memory parameters.

Using adb to Change Parameters

Here is how to change the shmseg value from 6 to 12:

```
$ adb -w /unix                      # you type the adb command
cannot open core                    # system displays
ready                               # system displays
shminfo+c?D                         # you type to display the value
                                    # in Decimal
shminfo+0xC:             6          # system displays
.?W 0d12                            # you type to change the value
                                    # in decimal
shminfo+0xC:           0x6  =  0xC # system displays
$q                                  # to quit adb, you type
```

To determine the offset and value, look at the tables below, i.e. shmseg is at shminfo+C.

Shared Memory

```
shminfo+0:   262144
  int      shmmax     /* max shared memory segment size */
  #define  SHMMAX      262144
shminfo+4:   1
  int      shmmin     /* min shared memory segment size */
  #define  SHMMIN      1
shminfo+8:   100
  int      shmmni     /* # of shared memory identifiers */
  #define  SHMNMI      100
shminfo+C:   6
  int         shmseg  /* max attached shared memory segments per process */
  #define  SHMSEG      6
shminfo+10:    0
  int      shmbrk     /* gap (in clicks) used between data and shared
                         memory */
               0
shminfo+14:   512
  int      shmall     /* max total shared memory system wide (in clicks) */
  #define  SHMALL      512
```

Semaphores

```
seminfo+0:   50
```

```
  int      semmap     /* # of entries in semaphore map */
  #define  SEMMAP     50
seminfo+4:   50
  int      semmni     /* # of semaphore identifiers */
  #define  SEMMNI     50
seminfo+8:   300
  int      semmns     /* # of semaphores in system */
  #define  SEMMNS     300
seminfo+C:   30
  int      semmnu     /* # of undo structures in system */
  #define  SEMMNU     30
seminfo+10:  25
  int      semmsl     /* max # of semaphores per id */
  #define  SEMMSL     25
seminfo+14:  10
  int      semopm     /* max # of operations per semop call */
  #define  SEMOPM     10
seminfo+18:  10
  int      semume     /* max # of undo entries per process */
  #define  SEMUME     10
seminfo+1C:  94
  int      semusz     /* size in bytes of undo structure */
  #define   SEMUSZ    (sizeof(struct sem_undo)+sizeof(struct undo)*SEMUME)
seminfo+20:  32767
  int      semvmx     /* semaphore maximum value */
  #define  SEMVMX     32767
seminfo+24:  16384
  int      semaem     /* adjust on exit max value */
  #define  SEMAEM     16384

Message Operations
------------------
msginfo+0:    100
  int       msgmap    /* # of entries in msg map */
  #define   MSGMAP    100
msginfo+4:    8192
  int       msgmax    /* max message size */
  #define   MSGMAX    8192
msginfo+8:    16384
  int       msgmnb    /* max # bytes on queue */
  #define   MSGMNB    16384
msginfo+C:    50
  int       msgmni    /* # of message queue identifiers */
  #define   MSGMNI    50
msginfo+10:   8
  int       msgssz    /* msg segment size (should be word size multiple) */
  #define   MSGSSZ    8
msginfo+14:   200
  int       msgtql    /* # of system message headers */
  #define   MSGTQL    200
msginfo+18:   8192
  ushort    msgseg    /* # of msg segments (MUST BE < 32768) */
  #define   MSGSEG    8192
```

Article Change History:
25 Jun 1993 - Revised for clarity.
31 Aug 1993 - Reviewed for technical accuracy.

Keywords: <None>

==========

19960215 11:05:19.00

Tech Info Library Article Number: 2815