Nucleus DBUG+

Reference Manual

0001048-001

Accelerated Technology, Inc.
720 Oak Circle Dr. E.
Mobile, AL 36609
(334) 661-5770

Related Documentation

Nucleus PLUS Reference Manual, by Accelerated Technology, describes the operation and usage of the Nucleus PLUS kernel.

Nucleus PLUS Internals, by Accelerated Technology, describes, in considerable detail, the implementation of the Nucleus PLUS kernel.

Style and Symbol Conventions

Program listings, program examples, filenames, menu items/buttons and interactive displays are each shown in a special font.

Program listings and program examples - `Courier New`
Filenames - `COURIER NEW, ALL CAPS`
Interactive Command Lines - **`Courier New, Bold`**
Menu Items/Buttons – *Times New Roman Italic*

Trademarks

MS-DOS is a trademark of Microsoft Corporation
UNIX is a trademark of X/Open
IBM PC is a trademark of International Business Machines, Inc.

Additional Assistance

For additional assistance, please contact us at the following:

Accelerated Technology
720 Oak Circle Drive, East
Mobile, AL 36609
800-468-6853
334-661-5770
334-661-5788 (fax)
support@atinucleus.com
http://www.atinucleus.com

Document Part Number : 0001048-001
Last Revised: August 11, 1999

Accelerated Technology

INCORPORATED

Contents

1

Getting Started

Introduction

This chapter provides an introduction to the Nucleus DEBUG+. Additionally, this chapter provides a high-level introduction to the debugger concepts.

About Nucleus DBUG+

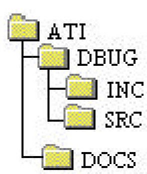
Nucleus DBUG+ provides operational insight into real-time applications running under the Nucleus PLUS multi-tasking executive. Information pertaining to Nucleus PLUS tasks, mailboxes, queues, pipes, semaphores, events, signals, partitions, dynamic memory, interrupts, and timers is provided. Nucleus DBUG+ also allows the user to initiate various Nucleus PLUS services from the command line. This feature enables the user to simulate interaction from Interrupt Service Routines (ISRs) and/or tasks that are not fully operational.

This chapter describes the Nucleus DBUG+ environment, installation, and files. Additionally, this chapter provides an explanation of the debugger execution environment, and provides a high-level introduction to the debugger concepts.

Installing Nucleus DBUG+

Nucleus DBUG+ is designed to operate as a task along with the actual tasks of the real-time application. It is comprised of several files and is designed for quick portation to any target environment.

To install the Nucleus Debug software, execute `SETUP.EXE` from the distribution software. The Nucleus Debug files will be in the Debug directory as shown below. After installation, the directory structure will be as follows:



NOTE: Instructions do not apply to Nucleus EDE. If you purchased EDE, the files are only included into the workspace as source code files. They are not meant to be compiled as is. Please refer to your Nucleus EDE Online Help for more information.

Debugger Files:

The Nucleus DBUG+ is comprised of four files that contain roughly 8,000 lines of C source. The actual file names are listed below:

```
DB_DEFS.H
DB_EXTR.H
DBC.C
DBT.C
```

The file `DB_DEFS.H` contains all of the constants used by the Nucleus DBUG+.

The file `DB_EXTR.H` contains external function references between the two C files and resolves forward references in DBC.C in particular.

The file `DBC.C` contains the portable portion of the Nucleus Debugger. This file should require very little, if any, modification to run on a given target platform.

The file `DBT.C` contains routines for string comparison, string conversion, and character input and output. The routines in this file may require modification to run on different target platforms.

In order to build a system with the Nucleus debugger the object files associated with the compilation of `DBT.C` and `DBC.C` must be linked in with the rest of the application.

The source files contain "`#include`" paths that may need modifying. Also, "`#include <conio.h>`" may need to be commented out, if that file isn't available with your toolset.

How to Use Nucleus DBUG+

To use Nucleus PLUS debugger, a task must be added to the system to control the debugger. This task must contain the entry point of the `DBC_Debug` module. The user must adjust the debugger task's priority to a level that allows an acceptable amount of interaction between the debugger and the system.

Target System Considerations

Nucleus DBUG+ is implemented as a task. The entry point of the Nucleus DBUG+ task is at `DBC_Debug`. Hence, this is the start address that should be specified as `task_entry` in the `NU_Create_Task` definition structure in `Application_Initialize` for a debugger task.

The priority of the Nucleus DBUG+ is application dependent. Generally, the Nucleus DBUG+ should have the lowest priority possible that still permits reasonable execution.

Although Nucleus DBUG+ avoids allocating large data structures on its run-time stack, a stack size permitting 4-5 levels of function nesting with moderate local variable and parameter passing is required.

On CISC processors, the actual program memory requirements of Nucleus DBUG+ are on the order of 15 to 25 KBytes. For RISC platforms, the program memory requirements may double. Data memory requirements generally range from 2-4 KBytes on all platforms.

Configuration Options

There are two basic configuration options. The switches `-DNO_SERVICES` and `-DNO_STATUS` are used to reduce the size of the debugger. Without these switches the full system will be compiled.

2

Functional Description

Physical Input/Output

C - Library Usage

Display Parameters

This chapter describes Nucleus DBUG+ functional operations. The debugger performs operations based on single-line commands entered by the user at the debugger prompt. Decimal is the default display format. However, some data items are more naturally displayed/entered in hexadecimal. In such cases, the field name or prompt contains a (H) to indicate a hexadecimal format.

Physical Input/Output

Nucleus DBUG+ uses `DBT_Get_Char` to input characters from the user and `DBT_Put_Char` to output characters to the user. Both routines are defined in the file `DBT.C`.

`DBT_Get_Char`, by default, contains code to poll the serial port using calls to `NU_Sleep` and the `DOS C` library call "kbhit" to wait for a character from the user. Once "kbhit" indicates a character is present, "getch" is used to retrieve the character. In most `non-DOS` targets, `DBT_Get_Char` must be modified.

`DBT_Put_Char`, by default, contains code to send a character to the user using a call to "putch." In most non-DOS targets, `DBT_Put_Char` must be modified.

C Library Usage

In addition to the input/output references mentioned in the previous section, Nucleus DBUG+ also references several other standard C library functions. The services "atoi," "strncmp," and "strcat" are referenced in the `DBT.C` file and "sprintf" is referenced in the `DBC.C` file. This information is provided in case there are problems with a cross-compiler's libraries.

Display Parameters

The display of Nucleus `DBUG+` information is controlled by several constants defined in the `DB_DEFS.H` file.

The constant ROW defines the number of rows available on the display. ROW must be greater than 9. The constant COL defines the number of columns available on the display. COL should be greater than 50. Nucleus uses these constants to determine how much information to display at one time.

The constant DUPLEX determines whether or not the user's terminal is FULL or HALF duplex. A value of 1 indicates that FULL duplex is in use, i.e. all characters entered must also be sent back to the user.

3

Nucleus DBUG+ Services

- User Help
- Task Status
- Mailbox Status
- Queue Status
- Pipe Status
- Semaphore Status
- Event Group Status
- Signal Status
- Timer Status
- Partition Timer Status
- Dynamic Memory Status
- Display Memory
- Set Memory

User Help

This command provides a brief list of all available Nucleus DBUG+ commands and their meaning. This list is automatically displayed when the debugger starts.

The following is an example of a help command entered by the user:

```
DBUG+> help

          ***** Nucleus PLUS Tasking Debugger *****

                       Version 1.5

ts [name]        Display Task Status  [task name]
ms [name]        Display Mailbox Status [mailbox name]
qs [name]        Display Queue Status [queue name]
ps [name]        Display Pipe Status [pipe name]
ss [name]        Display Semaphore Status [semaphore name]
es [name]        Display Event Status [event group name]
si [name]        Display Signal Status [task name]
ti [name]        Display Timer Status [timer name]
pm [name]        Display Partition Status [partition name]
dm               Display Dynamic Memory Status
m [a]            Display Memory [(H) address]
sm  a            Set Memory a = (H) address
help             Display this menu again
```

Available Nucleus Service Routines:

Service Routine	Description
NU_Resume_Task	Resumes a task
NU_Suspend_Task	Suspends a task
NU_Terminate_Task	Terminates execution of a task
NU_Reset_Task	Resets terminated task
NU_Change_Priority	Changes a task's priority
NU_Broadcast_To_Mailbox	Broadcasts message to mailbox
NU_Receive_From_Mailbox	Receives message from mailbox
NU_Reset_Mailbox	Resets mailbox
NU_Send_To_Mailbox	Sends message to mailbox
NU_Broadcast_To_Queue	Broadcasts message to queue
NU_Receive_From_Queue	Receives message from queue
NU_Reset_Queue	Resets queue
NU_Send_To_Front_Of_Queue	Sends message to front of a queue
NU_Send_To_Queue	Sends message to queue
NU_Broadcast_To_Pipe	Broadcasts message to pipe
NU_Receive_From_Pipe	Receives message from pipe
NU_Reset_Pipe	Resets pipe
NU_Send_To_Front_Of_Pipe	Sends message to front of a pipe
NU_Send_To_Pipe	Sends message to pipe
NU_Obtain_Semaphore	Obtains semaphore
NU_Release_Semaphore	Releases semaphore
NU_Reset_Semaphore	Resets a semaphore
NU_Retrieve_Events	Retrieves events
NU_Set_Events	Sets events
NU_Send_Signals	Sends signals
NU_Control_Timer	Controls timer
NU_Reset_Timer	Resets a timer
NU_Retrieve_Clock	Retrieves the Nucleus PLUS time
NU_Set_Clock	Sets the Nucleus PLUS time
NU_Allocate_Partition	Allocates a memory partition
NU_Deallocate_Partition	Deallocates a memory partition
NU_Allocate_Memory	Allocates dynamic memory
NU_Deallocate_Memory	Deallocates dynamic memory

Task Status

`ts` *`[task name]`*

This command provides information about Nucleus PLUS tasks. If information on a specific task is required, `"ts"` must be followed by the corresponding task name. Information on all tasks is displayed if a task name is not specified.

The task status service is comprised of several fields, including task status, scheduled count, priority, preempt, time slice, stack base, size, and minimum stack.

The task status field contains the current status of the task, including: `READY`, `FINISHED`, `SLEEP`, `MAILBOX`, `QUEUE`, `PIPE`, `EVENTS`, `SEMA`, `MEMORY`, `PARTITION`, and `DRIVER`.

The scheduled count contains the total number of times the task has been scheduled.

The priority field contains the task priority value (0-255). Lower priority numbers indicate higher priority.

The preempt field contains the task preempt value: YES/NO.

The time slice field contains the task time slicing value. If this field contains a zero value the time slicing is disabled.

The stack base field contains the starting address of the task's stack.

The size field contains the total number of bytes in the task's stack.

The minimum stack field contains the minimum amount of bytes available in the task's stack.

The following is an example of a user request to display the status of Task 1:

```
DBUG+> ts Task 1

               ***** Task Status Display *****

  Task Name:                     Task 1
  Task Status:                   READY
  Scheduled Count (H):           26
  Priority:                      1
  Preemptable:                   YES
  Time Slice:                    DISABLED
  Stack Base (H):                278A0008
  Stack Size:                    2000
  Minimum Stack:                 500
```

Mailbox Status

`ms [`*`mailbox name`*`]`

This command provides information about Nucleus PLUS mailboxes. If information on a specific mailbox is required, "ms" must be followed by the corresponding mailbox name. Information on all mailboxes is displayed if a mailbox name is not specified.

Mailbox Status information is comprised of various fields, including the suspend type, message present, tasks waiting, and first task.

The suspend type field contains one of the Nucleus PLUS suspend types: FIFO (First-In-First-Out) or PRIORITY (based on task priority).

The message present field contains TRUE if a message is present or FALSE if no messages are present.

The tasks waiting field contains the total number of tasks waiting on this mailbox.

The first task field contains a pointer to the first task waiting.

The following is an example of a user request to display the status of Mail 0:

```
DBUG+> ms Mail 0

               ***** Mailbox Status Display *****

   Mailbox Name:                      Mail 0
   Suspend Type:                      FIFO
   Message Present:                   TRUE
   Tasks Waiting:                     0
   First Task Waiting:                NONE
```

Queue Status

`qs [`*`queue name`*`]`

This command provides information about Nucleus PLUS queues. If information on a specific queue is required, "qs" must be followed by the corresponding queue name. Information on all queues is displayed if a queue name is not specified.

Queue Status information is comprised of various fields, including the start address, queue size, available, messages, message type, message size, suspend type, tasks waiting, and first task.

The start address field contains the starting memory address of the queue.

The queue size field contains the total number of UNSIGNED data elements in the queue.

The available field contains the total number of available UNSIGNED data elements in the queue.

The messages field contains the total number of messages currently in the queue.

The message type field contains the value of the message type. Valid message types are FIXED, and VARIABLE.

The message size field contains the number of UNSIGNED data elements in each message. If the queue is set to VARIABLE this number will reflect the maximum message size.

The suspend type field contains the FIFO (First-In-First-Out) or the PRIORITY (task priority) value.

The tasks waiting field contains the total number of tasks waiting on the queue.

The first task field contains a pointer to the first task waiting.

The following is an example of a user request to display the status of queue 0:

```
DBUG+> qs Queue 0

                 ***** Queue Status Display *****

  Queue name:                    Queue 0
  Start Address:                 2B5C0008
  Queue Size:                    100
  Available:                     95
  Messages:                      5
  Message Type:                  FIXED
  Message Size:                  1
  Suspend Type:                  FIFO
  Tasks Waiting:                 0
  First Task Waiting:            NONE
```

Pipe Status

`ps [pipe name]`

This command provides information about Nucleus PLUS pipes. If information on a specific pipe is required, "ps" must be followed by the corresponding pipe name. Information on all pipes is displayed if a pipe name is not specified.

Pipe Status information is comprised of various fields, including the start address, pipe size, available, messages, message type, message size, suspend type, tasks waiting, and first task.

The start address field contains the starting memory address of the pipe.

The pipe size field contains the total number of bytes in the pipe.

The available field contains the total number of available bytes in the pipe.

The messages field contains the total number of message currently in the pipe.

The message type field contains the value of the message type. Valid message types are FIXED, and VARIABLE.

The message size field contains the number of bytes in each message. If the pipe is set to VARIABLE this number will reflect the maximum message size.

The suspend type field contains the FIFO (First-In-First-Out) or the PRIORITY (task priority) value.

The tasks waiting field contains the total number of tasks waiting on the pipe.

The first task field contains a pointer to the first task waiting.

The following is an example of a user request to display the status of Pipe 0:

```
DBUG+> ps Pipe 0

               ***** Pipe Status Display *****
  Pipe name:                             Pipe 0
  Start Address:                         2B5C0008
  Pipe Size:                             100
  Available:                             95
  Messages:                              5
  Message Type:                          NU_FIXED_SIZE
  Message Size:                          1
  Suspend Type:                          FIFO
  Tasks Waiting:                         0
  First Task Waiting:                    NONE
```

Semaphore Status

`ss [`*`semaphore name`*`]`

This command provides information about Nucleus PLUS semaphores. If information on a specific semaphore is required, `"ss"` must be followed by the corresponding semaphore name. Information on all semaphores is displayed if a semaphore name is not specified.

Semaphore Status information is comprised of various fields, including the current count, suspend type, tasks waiting, and first task fields.

The current count field contains the current instance count of the semaphore.

The suspend type field contains the FIFO (First-In-First-Out) or the PRIORITY (task priority) value.

The tasks waiting field contains the total number of tasks waiting on the semaphore.

The first task field contains a pointer to the first task waiting.

The following is an example of a user request to display the status of Sem 0:

```
DBUG+> ss Sem 0

       ***** Semaphore Status Display *****

   Semaphore name:                     Sem 0
   Current Count:                      2
   Suspend Type:                       FIFO
   Tasks Waiting:                      4
   First Task:                         Task3
```

Event Group Status

`es [event group name]`

This command provides information about Nucleus PLUS event-flag groups. If information on a specific group is required, "es" must be followed by the corresponding event-flag group name. Information on all event-flag groups is displayed if a group name is not specified.

Event Status information is comprised of various fields, including the event flag, tasks waiting, and first task fields.

The event flag field contains the current event flag settings of the corresponding event-flag group.

The tasks waiting field contains the total number of tasks waiting on the event group.

The first task field contains a pointer to the first task waiting.

The following is an example of a user request to display the status of event-flag group 0:

DBUG+> **es Event 0**

```
                 ***** Event Status Display *****
Event Name:                          Event 0
Event Flag (H):                      00000000
Tasks Waiting:                       5
First Task:                          Task5
```

Signal Status

`si [`*`task name`*`]`

This command provides information about Nucleus PLUS signals. If information on a specific signal is required, `"si"` must be followed by the corresponding task name. Information on all task's signals is displayed if a task name is not specified.

Signal Status information is comprised of various fields, including the signal handler address, signal mask, and signals present.

The Signal Handler address indicates the address location of the signal handler control block.

Signal Mask indicates the signal mask that is currently enabled.

Signals Present represents the current signal mask that has been receiving values.

The following is an example of a user request to display the status of signals for Task 0:

```
DBUG+> si Task 0

                 ***** Signal Status Display *****
  Task Name:                                   Task 0
  Signal Handler (H):                          2B5C0008
  Signal Mask (H):                             00000000
  Signals Present (H):                         NONE
```

Timer Status

`ti [`*`timer name`*`]`

This command provides information about Nucleus PLUS timers. If information on a specific timer is required, "`ti`" must be followed by the corresponding timer name. Information on all timers is displayed if a timer name is not specified.

Timer Status information is comprised of various fields, including the enable, expirations, id, initial time, and reschedule time.

The enable field contains the timer's current enable state, either ENABLE or DISABLE.

The expirations field contains the total number of times the timer has expired.

The id field contains the user supplied id.

The initial time field contains the initial timer expiration value.

The reschedule time field contains the timers reschedule value.

The following is an example of a user request to display the status of Timer 0:

DBUG+> **ti Timer 0**

```
               ***** Timer Status Display *****

  Timer Name:                           Timer 0
  Enable State:                         NABLE
  Expirations:                          23
  Timer ID:                             0
  Initial Time:                         4
  Reschedule Time:                      10
```

Partition Memory Status

`pm [`*`partition name`*`]`

This command provides information about Nucleus PLUS memory partitions. If information on a specific partition is required, "pm" must be followed by the corresponding partition name. Information on all partitions is displayed if a partition name is not specified.

Partition Status information is comprised of various fields, including the start address, pool size, partition size, available, allocated, suspend type, tasks waiting, and first task fields.

The start address field contains the starting address of the pool.

The pool size field contains the total number of bytes in the pool.

The partition size field contains the total number of bytes in the partition.

The available field contains the total number of partitions in the pool.

The allocated field contains the number of allocated partitions.

The suspend type field contains the FIFO (First-In-First-Out) or the PRIORITY (task priority) value.

The tasks waiting field contains the total number of tasks waiting on the partition pool.

The first task field contains a pointer to the first task waiting.

The following is an example of a user request to display the status of part 0:

DBUG+> **pm Part 0**

```
              ***** Partition Status Display *****
Partition Name:                      Part 0
Start Address:                       2B5C0008
Pool Size:                           8900
Partition Size:                      100
Available:                           89
Allocated:                           0
First Task Waiting:                  NONE
Last Task Waiting:                   NONE
```

Dynamic Memory Status

`dm [`*`memory pool name`*`]`

This command provides information about Nucleus PLUS memory pools. If information on a specific memory pool is required, `"dm"` must be followed by the corresponding memory pool name. Information on all memory pools is displayed if a memory pool name is not specified.

Dynamic Memory Status information is comprised of various fields, including the start address, pool size, min allocation, available, suspend type, tasks waiting, and first task fields.

The start address field contains the starting address of the pool.

The pool size field contains the total number of bytes in the pool.

The min allocation field contains the minimum number of bytes for each allocation from this pool.

The available field contains the total number of bytes available in the pool.

The suspend type field contains the FIFO (First-In-First-Out) or the PRIORITY (task priority) value.

The tasks waiting field contains the total number of tasks waiting on the dynamic memory pool.

The first task field contains a pointer to the first task waiting.

The following is an example of a user request to display the status of Nucleus PLUS dynamic memory Pool 1:

```
DBUG+> dm Pool 1

         ***** Dynamic Memory Status Display *****

  Memory Pool Name:                   Pool 1
  Start Address (H):                  2BC3001C
  Pool Size:                          59970
  Min. Allocation:                    50
  Available:                          59970
  Suspend Type:                       FIFO
  Task Waiting:                       0
  First Task Waiting:                 NONE
```

Display Memory "m"

This command displays memory in "unsigned long" format. If an address is supplied, the memory starting at that address is displayed. Otherwise, if an address is not specified, the memory display continues where the previous `"m"` command left-off.

The following is an example of a user request to display memory starting at the address 2BAA0008:

```
DBUG+> m 2BAA0008

2BAA0008    00081234
2BAA000C    00002BAF
2BAA0010    FFFF0000
2BAA0014    00000000
2BAA0018    00005678
2BAA001C    00000000
2BAA0020    00000000
2BAA0024    00000000
2BAA0028    00000000
2BAA002C    00000000
2BAA0030    00000000
2BAA0034    00000000
2BAA0038    00000000
2BAA003C    00000000
2BAA0040    00000000
2BAA0044    00000000
2BAA0048    00000000
2BAA004C    00000000
2BAA0050    00000051
2BAA0054    2BAA0000
2BAA0058    00081234
2BAA005C    00082BB4
```

Set Memory

This command modifies memory in "unsigned long" format starting at the address specified. The Nucleus PLUS debugger displays the address, the current contents of the memory location, and then waits for the user to enter new data. If new data is entered, the memory is modified, the address is incremented, and the Nucleus PLUS debugger prepares for modification of the new address. If no new data is entered, processing of the command is terminated.

The following is an example of a user request to place the value 1234 into location 2BAA0030:

```
DBUG+> sm 2BAA0030

2BAA0030  00000000  1234
2BAA0034  00000000  <cr>
```

NU_Resume_Task

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected task name. See the Nucleus PLUS Reference manual for more information about the `NU_Resume_Task` command.

The following is an example of a user request to resume Task 1:

```
DBUG+> NU_Resume_Task

Name of task to resume:                 Task 1
Return Status:                          NU_SUCCESS
```

NU_Suspend_Task

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected task name. See the Nucleus PLUS Reference manual for more information about the `NU_Suspend_Task` command.

The following is an example of a user request to suspend task 1:

```
DBUG+> NU_Suspend_Task

Name of task to suspend:                Task 1
Return Status:                          NU_SUCCESS
```

NU_Terminate_Task

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected task name. See the Nucleus PLUS Reference manual for more information about the NU_Terminate_Task command.

The following is an example of a user request to terminate task 1:

```
DBUG+> NU_Terminate_Task

Name of task to terminate:      Task 1
Return Status:                  NU_SUCCESS
```

NU_Reset_Task

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected task name. See the Nucleus PLUS Reference manual for more information about the NU_Reset_Task command.

The following is an example of a user request to reset task 1:

```
DBUG+> NU_Reset_Task

Name of task to reset:          Task 1
Return Status:                  NU_SUCCESS
```

NU_Change_Priority

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected task name and the new priority. See the Nucleus PLUS Reference manual for more information about the `NU_Change_Priority` command.

The following is an example of a user request to change task 1's priority to 9:

```
DBUG+> NU_Change_Priority

Name of task to change priority of:          Task 1
Enter the new priority for the task:
Return Status:                               NU_SUCCESS
```

NU_Broadcast_To_Mailbox

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected mailbox name and the message to send. Note: services initiated from Nucleus DBUG+ are not allowed to suspend. See the Nucleus PLUS Reference manual for more information about the `NU_Broadcast_To_Mailbox` command.

The following is an example of a user request to send a 4 word message to Mbox 1:

```
DBUG+> NU_Broadcast_To_Mailbox

Name of mailbox to broadcast to:             Mbox 1
Enter the message below (H):
Word  0:                                     12345678
Word  1:                                     9ABCDE
Word  2:                                     56780F00
Word  3:                                     87650000
Return Status:                               NU_SUCCESS
```

NU_Receive_From_Mailbox

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected mailbox name. Note: services initiated from Nucleus DBUG+ are not allowed to suspend. See the Nucleus PLUS Reference manual for more information about the `NU_Receive_From_Mailbox` command.

The following is an example of a user request that receives a 4-word message from Mbox 1:

```
DBUG+> NU_Receive_From_Mailbox

Name of mailbox to receive from:            Mbox 1
The following message was received (H):
Word  0:                                    1234000
Word  1:                                    4321000
Word  2:                                    5678000
Word  3:                                    8765000
Return Status:                              NU_SUCCESS
```

NU_Reset_Mailbox

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected mailbox name. See the Nucleus PLUS Reference manual for more information about the `NU_Reset_Mailbox` command.

The following is an example of a user request to reset Mbox 1:

```
DBUG+> NU_Reset_Mailbox

Name of mailbox to reset:                   Mbox 1
Return Status:                              NU_SUCCESS
```

NU_Send_To_Mailbox

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected mailbox name and the message to send. Note: services initiated from Nucleus DBUG+ are not allowed to suspend. See the Nucleus PLUS Reference manual for more information about the `NU_Send_To_Mailbox` command.

The following is an example of a user request to send a 4 word message to Mbox 1:

```
DBUG+> NU_Send_To_Mailbox

Name of mailbox to send to:                        Mbox 1
Enter the message below (H):
Word  0:                                           12340000
Word  1:                                           43210000
Word  2:                                           56780000
Word  3:                                           87650000
Return Status:                                     NU_SUCCESS
```

NU_Broadcast_To_Queue

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected queue name and the message to send. Note: services initiated from Nucleus `DBUG+` are not allowed to suspend. See the Nucleus PLUS Reference manual for more information about the `NU_Broadcast_To_Queue` command.

The following is an example of a user request to send a 1 word message containing 12340000 to Queue 1:

```
DBUG+> NU_Broadcast_To_Queue

Name of queue to broadcast to:                     Queue 1
Enter the message size:                            1
Enter the message below (H):
Word  0:                                           12340000
Return Status:                                     NU_SUCCESS
```

NU_Receive_From_Queue

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected queue name. Note: services initiated from Nucleus DBUG+ are not allowed to suspend. See the Nucleus PLUS Reference manual for more information about the `NU_Receive_From_Queue` command.

The following is an example of a user request that retrieves a 1-word message that contains 12340000 from queue 1:

```
DBUG+> NU_Receive_From_Queue

Name of queue to receive from:                  Queue 1
The following message was received (H):
Word  0:                                        12340000
Return Status:                                  NU_SUCCESS
```

NU_Reset_Queue

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected queue name. See the Nucleus PLUS Reference manual for more information about the `NU_Reset_Queue` command.

The following is an example of a user request to reset Queue 1:

```
DBUG+> NU_Reset_Queue

Name of queue to reset:                         Queue 1
Return Status:                                  NU_SUCCESS
```

NU_Send_To_Front_Of_Queue

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected queue name and the message to send. Note: services initiated from Nucleus DBUG+ are not allowed to suspend. See the Nucleus PLUS Reference manual for more information about the `NU_Send_To_Front_Of_Queue` command.

The following is an example of a user request to send a 1 word message containing 12340000 to Queue 1:

```
DBUG+> NU_Send_To_Front_Of_Queue

Name of queue to send to:                    Queue 1
Enter message size:                          1
Enter the message below (H):
Word  0:                                     12340000
Return Status:                               NU_SUCCESS
```

NU_Send_To_Queue

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected queue name and the message to send. Note: services initiated from Nucleus DBUG+ are not allowed to suspend. See the Nucleus PLUS Reference manual for more information about the `NU_Send_To_Queue` command.

The following is an example of a user request to send a 1 word message containing 12340000 to Queue 1:

```
DBUG+> NU_Send_To_Queue

Name of queue to send to:                    Queue 1
Enter message size:                          1
Enter the message below (H):
Word  0:                                     12340000
Return Status:                               NU_SUCCESS
```

NU_Broadcast_To_Pipe

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected pipe name and the message to send. Note: services initiated from Nucleus DBUG+ are not allowed to suspend. See the Nucleus PLUS Reference manual for more information about the `NU_Broadcast_To_Pipe` command.

The following is an example of a user request to send a 1-byte message containing 12 to Pipe 1:

```
DBUG+> NU_Broadcast_To_Pipe

Name of pipe to broadcast to:                 Pipe 1
Enter message size:                           1
Enter the message below (H):
Btye  0:                                      12
Return Status:                                NU_SUCCESS
```

NU_Receive_From_Pipe

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected pipe name. Note: services initiated from Nucleus DBUG+ are not allowed to suspend. See the Nucleus PLUS Reference manual for more information about the `NU_Retrieve_From_Pipe` command.

The following is an example of a user request that receives a 1-byte message that contains 12 from Pipe 1:

```
DBUG+> NU_Receive_From_Pipe

Name of pipe to receive from:                 Pipe 1
Enter message size:                           1
The following message was received (H):
Byte  0:                                      12
Return Status:                                NU_SUCCESS
```

NU_Reset_Pipe

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected pipe name. See the Nucleus PLUS Reference manual for more information about the `NU_Reset_Pipe` command.

The following is an example of a user request to reset Pipe 1:

```
DBUG+> NU_Reset_Pipe

Name of pipe to reset:                              Pipe 1
Return Status:                                      NU_SUCCESS
```

NU_Send_To_Front_Of_Pipe

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected pipe name and the message to send. Note: services initiated from Nucleus DBUG+ are not allowed to suspend. See the Nucleus PLUS Reference manual for more information about the `NU_Send_To_Front_Of_Pipe` command.

The following is an example of a user request to send a 1-byte message containing 12 to Pipe 1:

```
DBUG+> NU_Send_To_Front_Of_Pipe

Name of pipe to send to:                            Pipe 1
Enter message size:                                 1
Enter the message below (H):
Byte  0:                                            12
Return Status:                                      NU_SUCCESS
```

NU_Send_To_Pipe

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected pipe name and the message to send. Note: services initiated from Nucleus DBUG+ are not allowed to suspend. See the Nucleus PLUS Reference manual for more information about the `NU_Send_To_Pipe` command.

The following is an example of a user request to send a 1-byte message containing 12 to Pipe 1:

```
DBUG+> NU_Send_To_Pipe

Name of pipe to send to:                     Pipe 1
Enter message size:                          1
Enter the message below (H):
Byte  0:                                     12
Return Status:                               NU_SUCCESS
```

NU_Obtain_Semaphore

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected semaphore name. Note: services initiated from Nucleus DBUG+ are not allowed to suspend. See the Nucleus PLUS Reference manual for more information about the `NU_Obtain_Semaphore` command.

The following is an example of a user request to obtain Sem 1:

```
DBUG+> NU_Obtain_Semaphore

Name of the semaphore:                       Sem 1
Return Status:                               NU_SUCCESS
```

NU_Release_Semaphore

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected semaphore name. See the Nucleus PLUS Reference manual for more information about the `NU_Release_Semaphore` command.

The following is an example of a user request to release Sem 1:

```
DBUG+> NU_Release_Semaphore

Name of the semaphore:                        Sem 1
Return Status:                                NU_SUCCESS
```

NU_Reset_Semaphore

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected semaphore name. See the Nucleus PLUS Reference manual for more information about the `NU_Reset_Semaphore` command.

The following is an example of a user request to reset Sem 1:

```
DBUG+> NU_Reset_Semaphore

Name of semaphore to reset:                   Sem 1
Return Status:                                NU_SUCCESS
```

NU_Retrieve_Events

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected event name, the event flag operation, and the actual event flags required. Note that services initiated from Nucleus DBUG+ are not allowed to suspend. See the Nucleus PLUS Reference manual for more information about the `NU_Retrieve_Events` command.

The following is an example of a user request to get all the event flags (bits) represented by `0000FFEA` in Event 1:

```
DBUG+> NU_Retrieve_Events

Name of the event group:                    Event 1
Enter operation - AND, OR,
AND_CONSUME, OR_CONSUME:                    AND
Enter the Event Flag value (H):             0000FFEA
Return Status:                              NU_SUCCESS
```

NU_Set_Events

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected event name, the event flag operation, and the actual event flags. See the Nucleus PLUS Reference manual for more information about the `NU_Set_Events` command.

The following is an example of a user request to set the event flags (bits) represented by `0000FFEA` in Event 1:

```
DBUG+> NU_Set_Events

Name of the event group:                    Event 1
Enter operation (AND or OR):                OR
Enter the Event Flag value (H):             0000FFEA
Return Status:                              NU_SUCCESS
```

NU_Send_Signals

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected task name and the signal to send. See the Nucleus PLUS Reference manual for more information about the `NU_Send_Signals` command.

The following is an example of a user request to send a 1 word of signals, represented by 12340000 to Task 1:

```
DBUG+> NU_Send_Signals

Name of task to send to:                     Task 1
Enter signal (H):                            12340000
Return Status:                               NU_SUCCESS
```

NU_Control_Timer

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected timer name, and the operation. See the Nucleus PLUS Reference manual for more information about the `NU_Control_Timer` command.

The following is an example of a user request to control Timer 1:

```
DBUG+> NU_Control_Timer
```

Name of timer to control:	`Timer 1`
Enter the enable value:	`NU_DISABLE_TIMER`
Return Status:	`NU_SUCCESS`

NU_Reset_Timer

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected timer name. See the Nucleus PLUS Reference manual for more information about the `NU_Reset_Timer` command.

The following is an example of a user request to reset Timer 1:

```
DBUG+> NU_Reset_Timer

Name of timer to reset:                      Timer 1
Return Status:                               NU_SUCCESS
```

NU_Retrieve_Clock

This command executes the Nucleus PLUS service of the same name. See the Nucleus PLUS Reference manual for more information about the `NU_Retrieve_Clock` command.

The following is an example of a user request to read the current time (number of ticks):

```
DBUG+> NU_Retrieve_Clock

Nucleus PLUS time is (in ticks):             76
```

NU_Set_Clock

This command executes the Nucleus PLUS service of the same name after prompting the user for the time (number of ticks). See the Nucleus PLUS Reference manual for more information about the `NU_Set_Clock` command.

The following is an example of a user request to set the time (number of ticks) to 1:

```
DBUG+> NU_Set_Clock

Enter the new system time (in ticks):             1
```

NU_Allocate_Partition

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected memory partition name. Note: services initiated from Nucleus DBUG+ are not allowed to suspend. See the Nucleus PLUS Reference manual for more information about the `NU_Allocate_Partition` command.

The following is an example of a user request to allocate a memory partition from partition named Part 0:

```
DBUG+> NU_Allocate_Partition

Name of partition:                          Part 0
Partition Start Address (H):                2BAA001A
Return Status:                              NU_SUCCESS
```

NU_Deallocate_Partition

This command executes the Nucleus PLUS service of the same name after prompting the user for the selected memory partition absolute address. See the Nucleus PLUS Reference manual for more information about the `NU_Deallocate_Partition` command.

The following is an example of a user request to deallocate a memory partition previously allocated from partition named `Part0` with an absolute address of `2BAA001A`:

```
DBUG+> NU_Deallocate_Partition

Enter the partition address (H):                 2BAA001A
Return Status:                                   NU_SUCCESS
```

NU_Allocate_Memory

This command executes the Nucleus PLUS service of the same name after prompting the user for the number of bytes of memory needed. Note: services initiated from Nucleus DBUG+ are not allowed to suspend. See the Nucleus PLUS Reference manual for more information about the `NU_Allocate_Memory` command.

The following is an example of a user request to allocate a memory block of 200 bytes:

```
DBUG+> NU_Allocate_Memory

Name of memory pool:                            Sysmem 1
Enter the number of bytes req:                  200
Memory Start Address (H):                       2BC4000C
Return Status:                                  NU_SUCCESS
```

NU_Deallocate_Memory

This command executes the Nucleus PLUS service of the same name after prompting the user for the absolute address of the memory block to deallocate. See the Nucleus PLUS Reference manual for more information about the `NU_Deallocate_Memory` command.

The following is an example of a user request to deallocate a memory block at absolute address 2BAA001A:

```
DBUG+> NU_Deallocate_Memory

Enter the memory address (H):                   2BAA001A
Return Status:                                  NU_SUCCESS
```