RSX-11D SPEC

TO: RSX-11D Distribution

FROM: R. McLean

DATE: 22 June 72

SUBJ: TASK INSTALLATION

DOC: 130-101-038-00

Unless specified otherwise, the terms "RSX" and "RSX-11" imply "RSX-11D".

INTRODUCTION

Task Installation will consist of two task's. The Task Builder will perform the linking of the task and optionally install the task. Install will install a task into the system that has been built by the Task Builder.

The Task Builder is an interactive MCR Function that relocates and links object modules, user library's and, Global Common's. The Task Builder also allocates the user ASR'S and establishes the user's privileges, default priority and, default partitions.

Install is an mcr function that enters a task that has been built by the task builder into the rsx-11d system. Install optionally establishes the user's default partition and default priority.

Task Building of an RSX-11D Task is the process of creating a contiguous Task Image on a disk and optionally an entry in the System Task Directory, from relocatable object modules. These task's can be User Programs, MCR Function Task's or I/O Handler Task's.

The following is a list of the major operations required to build a task. They are listed in the order in which they are performed:

1. Allocates disk storage for the user task with the task name as the file name and finds a free System Task Directory entry if the task is to be installed.

2. Allocates ASR's for Global Common and System Lists*.

3. Reads object modules and relocates generating a memory image on a disk.

4. Conditionally Relocates routines from the User Library.

5. Conditionally Links to memory Resident Library routines.

---

* -- only privileged task's are provided access to List's, Tables, System Subroutines, and External Page.

6, Conditionally relocates routines from the System Library,

(Relocation is in is increasing virtual memory,)

CONTROL STATMENTS

The following is a description of the Task Builder Control Statments. The syntax is defined in modified BNF using the following conventions and definitions:

```
Angle brackets delimit metallinguistic variables,
Quote marks delimit a character string,
A slash (/) indicates alternation (OR),
No operator indicates concatenation,
Parens indicate factoring,
 CARRAGE RETURN implies line continuation necessary
 ALTMODE implies no continuation necessary
'$' indicates any number [including zero] of,
'NUL' indicates the empty set,
<BC> ::= Space or Blank or comma

<DIGIT> ::= "0"/"1"/"2"/"3"/"4"/"5"/"6"/"7"/"8"/"9";
<LETTER> ::= "A"/"B"/"C"  .,.,.,.,.,.   "X"/"Y"/"Z";
<NAME> ::= [1,6](<LETTER>/<DIGIT>);
<CR> ::= CARRAGE RETURN
<AM> ::= ALTMODE
<TC> ::= <CR>/<AM>;
<NBC> ::= Non Break Character
```

Install is an MCR Function task that inserts a task into the System Task Directory, The Task Name is the file name, The Task Name, Default Partition, and Default priority are specified in the following syntax:

```
  MCR SYNTAX ::= "INS"$<NBC> <BC> <TASK NAME> <BC>
   <DEFAULT PARTITION NAME> <BC> <DEFAULT PRIORITY> <TC>;
    <TASK NAME > ::= <NAME>;
    <DEFAULT PARTITION> ::= <NAME>;
    <DEFAULT PRIORITY> ::= <DECIMAL VALUE>;
```

EXAMPLE:

```
  MCR; INS JOE,PAR,400
```

The Task Builder is an MCR Function Task* the task name, default partition and default priority are specified in the following MCR Function syntax:

```
  MCR SYNTAX ::= "TKB"$<NBC> <BC> <TASK NAME> <BC>
     <DEFAULT PARTITION NAME> <BC> <DEFAULT PRIORITY NUMBER>

     <TC>;
      <TASK NAME> ::= <NAME>;
      <DEFAULT PARTITION NAME> ::= <NAME>;
      <DEFAULT PRIORITY NUMBER> ::=   <DECIMAL VALUE>;
```

```
  EXAMPLE:
    MCR, TKB JOE,PAR,400
```

TASK NAME -- RSX Task's are identified by Names that are specified when the Tasks are Installed, These names are independent of program or file names, Task Names are the names of the tasks to be Installed, This will be the name that is inserted in the System Task Directory (STD),

DEFAULT PARTITION -- Default Partition is the memory partition in which the task will run unless specifically overridden when the task's execution is requested,

DEFAULT PRIORITY -- Default Priority is the priority under which the Task will run unless a partition is specified when the Task's execution is requested,

After The Task Builder has validated the MCR command line it will will output prompting symbols and accept responses as described below,

OPTIONS

The Task Builder will type an "OPTIONS:" prompting symbol and accept responses of the following syntax,

```
  OPTIONS ::= $( <OPTION> / <TC> );
    <OPTION> ::= <OPTION SYMBOL> <TERMINATOR> ;
```

----

* -- WHEN BATCH IS IMPLEMENTED, THE TASK BUILDER WILL OPERATE UNDER BATCH AND NOT UNDER MCR,

```
<option symbol> ::= "pt"/"nm"/"sz"/"cp"/"nf"/"in";
<terminator> ::= <tc> / <space> <option>;
```

example:

```
options:
options:pt,nm,sz,cp
```

The option symbols have the following meanings:

IN -- Insert an entry for this task in the System Task Directory (STD).

NF-- declare this task to be a task that will not use the floating point unit.

PT -- A Privileged Task is a task that has Executive capabilities, and is trusted not to endanger the system. I/O Handler Task's and some MCR Function task's require this privilegge in order to access the PDP-11 External Page and System Subroutines, or System Tables and List's.

NM -- No load map prevents the load map from being printed as the task is being linked. The Task Builder program normally prints the load map.

SZ -- Size of task. This option cause the size of the user task to be printed after it has been linked.

CP -- Checkpointable task. This options causes the disk allocation for checkpointing and permits the task to request checkpointing.

STACK AREA

Stack space is allocated in increasing memory from virtual locaton two (virtual zero is always used for Directive status), with the Task's Stack Pointer set to the maximum address +2. The Stack Area is free memory and need NOT be used exclusively for Stack storage.

The Task Builder outputs a "STACK" prompting symbol and accepts a response of the following syntax:

```
STACK AREA ::= (<POSITIVE DECIMAL VALUE>/NUL) <TC>;
```

If no Stack size is specified, an area of 31 words is allocated.

Examples:

```
STACK:75
STACK:
STACK:0
```

POOL LIMIT

Pool Limit is the number of Pool Nodes that may be in use by a Task at any one time. If no Pool Limit is specified a default pool limit of five nodes will be allocated. The Task Builder outputs a "POOL:" prompting symbol and accepts a response of the following syntax:

```
POOL LIMIT ::= (<POSITIVE DECIMAL VALUE>/NUL) <TC>;
```

```
EXAMPLE:
  POOL:10
  POOL:
```

USER LIB

User Lib requests the input of the name of the user library, if any, to be specified. The Task Builder will type "USER LIB:" and accept a User Library name in the following syntax:

```
USER LIB NAM ::= (<LIBRARY NAME> /NUL) <TC> ;
  <LIBRARY NAME> ::= <NAME>;
```

```
EXAMPLE:
  USER LIB:LIBR
  USER LIB:
```

The User Library Name is optional and it will cause an indicated user provided library to be scanned (to satisfy unresolved global symbol references) before any other library is scanned. Routines from a User Library are relocated and included as a part of the Task load image.

GLOBAL COM

The Global Common is used for Inter-Task communications and extra-Task data storage. Install may be instructed to map a Task's references to elements of a named Common into a Global Common of the same name, with either read-only or read/write access. The Task Builder will type "GLOBAL COM:" and accept Global Common names in the following syntax:

```
  GLOBAL COM ::= <COMMON NAME> ('/'<COMMON NAME>)
       (<ACCESS CODE> /NUL) <TC>;
    <COMMON NAME> ::= <NAME>;
    <ACCESS CODE> ::= "RO"/"RW";


      EXAMPLE:
        GLOBAL COM:COMA/RW
        GLOBAL COM:
```

Each Global Common referenced requires one ASR for each 4K of storage area.

NUM LUNS

The request Num LUNS specifies the number of LUNs that will be allocated to a task. The Task Builder will type "NUM LUNS:" and accept the number of LUNs in the following syntax:

```
    NUM LUNS ::= <OCTAL VALUE> <TC>;


      EXAMPLE:
        NUM LUNS:5
        NUM LUNS:
```

DEFAULT ASSIGNMENTS

Default Assignments allows a Task's LUNs to be preassigned. Any LUNs not assigned will default to "none" (unassigned).

Any number of Assignments may be used. If a LUN is assigned to more than one physical device-unit, the latter assignments are considered re-assignments, and the last assignment is used as the default assignment. The Task Builder will type "DEFAULT ASSIGNMENTS:" and accept the default assignments in the following syntax:

```
 DEFAULT ASSIGNMENTS ::= <ASSIGNMENT>;
   <ASSIGNMENT> ::= <DEVICE UNIT>":" <LUNS>;
   <DEVICE UNIT> ::= <DEVICE NAME> <UNIT NUMBER>;
     <DEVICE NAME> ::= <LETTER> <LETTER>;
     <UNIT NUMBER> ::= <OCTAL VALUE>;
   <LUNS> ::= <LUN> <TERM>;
     <LUN> ::= <DECIMAL VALUE>;
     <TERM> ::= <TC>/(<BC> <LUNS>);


     EXAMPLE:
      DK0:1,2
      TT0:4,5,3
      TT1:6
```

RESIDENT CODE

Resident code defines the file names of the object modules that are to be linked to form the user task image. These object modules will be linked in the order in which they are specified. The Task Builder will type "resident code:" and accept the resident code modules in the following syntax:

```
 RESIDENT CODE ::= <MODULE FILE NAMES> <TC>;
   <MODULE FILE NAMES> ::= (<MODULE NAME>/NUL>;
     <MODULE NAME> ::= <NAME>;
```

GENERAL DESCRIPTION

The Task Builder is a single task that contains a linker which relocates and links Object Modules, relocates and links references to Global Common, and Links to the Resident Re-entrant Core Library. The output of this linker will be placed directly on an allocated area on the specified disk. The Task Builder program will allocate the ASR's and System Task Directory entry for each task.

Install is a single task that installs a task into the system by making an entry in the System Task Directory (STD). The task must have been previously built by the Task Builder and the file name must be the same as the task name.

DISK ALLOCATION:

The Task Builder allocates and zeroes contiguous space on the specified disk for the resident code (excluding Global Common, re-entrant library and system list's) of a user task. If the task is Checkpointable The Task Builder will also allocate additional space to save the checkpointed task image.

ASR ALLOCATION

For all Privileged Tasks, use of the virtual address space from 060000 thru 177777 (ASR3-ASR7) is pre-defined. Address space 160000-177777 (ASR7) is mapped into the PDP-11 External Page. Address space 100000-157777 (ASRs 4, 5 & 6) is mapped into the executive's space for access to system tables, lists, pool, and executive subroutines. Address space 060000-077777 (ASR3) is used by the Task to access all memory (via run-time alteration of its contents), and its contents are not prescribed by The Task Builder.

Address space 000000-057777 (ASRs 0, 1 & 2) of Privileged Tasks, and all memory address space of non-Privileged Tasks is allocated as follows:

Memory space is allocated for access to GLOBAL COMMONs from the top (starting with ASR7 for non-Privileged Tasks & ASR2 for Privileged Tasks). One ASR is used for each GLOBAL COMMON (less than 4K) used. Additional ASRs are used for GLOBAL COMMONs that are greater than 4K words.

If references to the Resident Library are made, sufficient ASRs to map into the entire library are allocated from the top down.

ASRs for the Task execution (including routines from a user and system medium library) are allocated from the bottom up. ASR'S are completely setup for the task's execution by The Task Builder. The only changes that the system needs to make at request time is to set up the physical location ofthe user code. This is determined at execution by the partition that is specified when the task is requested.

RELOCATION PROCEDURE

All task's will be relocated in physical core by the segmentation unit (KT11). The user code and free core must be contiguous but it may be anywhere in physical core. The task will appear to start at location 0 and expand upwards in Virtual memory (as specified in ASR Allocation). A user task may be any size to a maximum of 32K. The Global Common and user library are included in the size of the task and may not be contiguous in virtual or physical memory.

Memory is allocated in the following sequence:
If the task is a Privilledged Task ASR7 is initalized for the External Page, ASR'S 6,5,4 are then used as necessary to permit access to the system list's and re-entrant I/O Subroutines.

Global Common is allocated first since it's size, location, and characteristics are known before the Object Modules are linked. The Global Common loads using the first free ASR (ASR7 if not a privilledged task).

After the Global Common is allocated the User Task is linked. The base of the user task starts just above the Free Core and must be physically continuous. The User Task and Free Core always start from Virtual 0 (ASR0) and expand upwards in Virtual memory.
The Re-entrant Position Independent System Library (FORTRAN LIBRARY, ETC.) is the last segment to be linked. It will use the remaining ASR'S. The Library will have read only privileges to protect it against accidental destruction.

SYSTEM TASK DIRECTORY ENTRY:

The Task Builder and Install will make an entry in the System Task Directory. This entry will include the Task Name, Default Partition, and Default Priority as specified in the System List's Spec. The Task Installation will insure that there are no conflicts in Task Names or other parameters of the task.