SPANEX™

Restart and Job Networking Guide

Span Software Consultants Limited

Version: 06.0 Product Number: SPOS-001

Revision: 1st March 2015 Manual Ref: SPX-03-019

CONTENTS Page

This page intentionally left blank.

1 Introduction

Note: This manual describes in detail the usage of the SPANEX Job Networking and Job Restart facilities, including such information as SPANEX definition statements and command formats. New readers and those intending to perform the simplest and quickest implementation are recommended to gain initial familiarity with SPANEX by working through the SPANEX Scheduling Beginning User's Guide, manual ref SPX-12. Subsequent reading should continue with Section 10 on page 151, the description of the SPANEX Quicknet feature. General information concerning SPANEX options, parameters, etc, will be found in the SPANEX General Usage manual, ref SPX-02, and details on the installation of the product are in the SPANEX Installation and Maintenance manual, ref SPX-09.

SPANEX, when used to monitor the execution of batch user programs and utilities, can determine the results, success or failure, of the execution of all Jobsteps in all job suites that are under its control. Definition of the criteria by which success or failure is determined is extremely flexible and powerful, and this information may be put to use by implementing the SPANEX Automatic Job Restart Facility based upon user-defined criteria for the resumed execution of failed jobs. Such is the power of this facility that it alone justifies the adoption of SPANEX as an installation standard.

The SPANEX Restart Facility provides automatic (with optional operator override) restart at the jobstep level for all multi-step jobs, and in addition will ensure that dependent jobs are run only when pre-requisite jobs have completed successfully (this latter function may be independent of the SPANEX Job Networking facility if automatic job scheduling is not required for a particular job stream). It is possible to define the logic of the relationships between the steps of a job, so as, for example, to specify that a dataset restore utility be run before a second attempt at an update step be made; this restore step could be run only in the situation where the update step had previously failed.

The SPANEX Job Networking Facility extends this capability to the relationship between the jobs of a suite. Inter-dependencies of any complexity may be defined to SPANEX, and SPANEX will automatically submit Jobs for execution at the correct point in the processing of the suite, always ensuring the successful completion of pre-requisite jobs and other events beforehand. Control of a SPANEX Job Network is provided via the SPANEX Utility, which may be run as a batch job, via a time-sharing or dedicated SPANEX 3270 terminal, or from an MCS operator console. Facilities are also available for the dynamic control of a Job Network from within application programs which are processing as part of the same Job Network, thus enabling decisions to be made, according to the application program logic, concerning which jobs should run and which should not in any given combination of circumstances. SPANEX at all times gives access to all the relevant information, either via internal application program macro calls or by the use of the SPANEX Utility, by operator, time-sharing user or other personnel.

Standard SPANEX facilities provide the ability to define for each step of each job what constitutes a failure, and a set of generation macros provides the user with the means to define to the SPANEX Restart and Job Networking facilities the steps and jobs for a suite and their interrelationships.

Changes for Version 06.0

SPANEX is undergoing a continuous cycle of development and enhancement. The new and improved features of Version 06.0 of SPANEX over the previous Version (05.1) are listed here.

- New options of the LINKAGE parameter of #SPANTRY allow the specification of the storage location of dynamic save areas.
- New CPRIGHT2 option of the #SPZFLD macro for a mixed-case copyright notice value.
- New format option for SPZPARSE fields that allows format-checking of character name values (ie alphabetic first character, followed by alpha-numeric characters).
- All SPANEX modules can now be located above the 16-Megabyte line.

Changes for Version 05.1

SPANEX is undergoing a continuous cycle of development and enhancement. The new and improved features of Version 05.1 of SPANEX over the previous Version (05.0) are listed here.

- Support for z/OS.
- Support for JES job numbers up to J0999999.
- New dump-formatting service routine for printed output (SPZFDUMP).
- Support for 64-bit applications is included in the #SPANTRY, #SPANXIT, #SPAMODE and related macros.
- Support for relative branching, and assembler modules without base registers, is included in the #SPANTRY, #SPANXIT and related macros.
- All software maintenance has been consolidated into the SPANEX code, and many reliability and service improvements have been made throughout the product.
- Minor editorial and technical changes have been made throughout the SPANEX manuals.

Changes for Version 05.0

The new and improved features of Version 05.0 of SPANEX over the previous Version (04.6) are listed here.

- Support for "Year 2000 compliance". All SPANEX reports, displays and date formats now use 4-digit year indications. All scheduling and calendar functions support the running of jobs across the boundary from 1999 to the year 2000.

- SPANEX Restart and Networking features now allow a SPANEX KSDS Catalog to be defined with JCL DD statements. This allows individual users or groups to have their own SPANEX Catalog datasets, if the VSAM KSDS Catalog is implemented when SPANEX is installed.

- SPANEX Quicknet now propagates SPXCAT1 and SPXCAT2 DD statements to submitted jobs, in the same manner as previously supported for TASKLIB and STEPCAT DD statements. This allows the use of user-defined SPANEX VSAM KSDS catalogs, in addition to the optional system-wide catalog specification.

- SPANEX now supports RACF checking of all MVS and JES commands issued from within SPANEX job control features. Authorisation to issue commands can be given to SPANEX without allowing the user assigned to a job to have that authority.

- SPANEX Quicknet now detects and reports Abend codes from all jobsteps, rather than simply reporting that an Abend occurred.

- SPANEX Quicknet retrospective condition code checking now allows specific Abend codes to be interrogated when determining whether a jobstep ended successfully.

- An MVS dump is no longer produced if the "Cancel" option is given to the SPANEX OPT=D jobstep start-up message.

- SPZPARSE now supports TSO- or IDCAMS-style command syntax, as an alternative to the traditional "keyword=value" style. Command and parameter format is dynamically specified via a new parameter of the #SPZPARS macro.

- The features provided in the #SPANTRY and #SPANXIT macros for MVS/ESA and OS/390 programming using the Linkage Stack may now be executed is ASC AR mode.

- New DATE4 field type in the #SPZFLD macro supports for 4-digit year notation for Year 2000 compliance.

- New SYSID field type in the #SPZFLD macro supports inclusion of the SMF System ID in program output.

- New USERID field type in the #SPZFLD macro supports inclusion of the owning user ID in program output.

- New version of the SOB used by SPOUTPUT eliminates all 3-byte addresses from the SPOUTPUT interface. All modules sharing a SOB must be assembled with the same version of the SPANEX macros, but SPOUTPUT supports existing modules compiled with all previous versions of the macros.

- Minor editorial and technical changes have been made throughout the SPANEX manuals.

Changes for Version 04.6

The new and improved features of Version 04.6 of SPANEX over the previous Version (04.5) are listed here.

- SPANEX installation, and all SPANEX macros now support the High-Level Assembler. The High-level Assembler is now the default assembler for the SPANEX installation process (#SPXGEN macro).

- SPANEX Quicknet now propagates STEPCAT DD statements to submitted jobs, in the same manner as previously supported for TASKLIB DD statements. This allows SPANEX restart catalog information to be held in an MVS user catalog.

- Arbitrary internal limits for the number of jobs and jobsteps in a SPANEX Network have been substantially increased. Limits are now 4000 jobs and 16000 jobsteps in a Network. These limits can be increased further if required by means of a small user modification to SPANEX.

- SPZSORT can now sort in-storage records up to 32767 bytes in length.

- New features are provided in the #SPANTRY and #SPANXIT macros for MVS/ESA programming using the Linkage Stack.

- The #SPANTRY macro now support ESA-style module entry and exit linkage, and allows user-defined identification text to be added to a module entry point.

- All SPANEX executable macros for Span Service Routine functions now support user programs running in AR mode on MVS/ESA systems.

- New “Double-underline” feature is provided by SPOUTPUT formatting.

- Minor editorial and technical changes have been made throughout the SPANEX manuals.

SPANEX Manuals	Order No
SPANEX General Usage Manual	SPX-02
SPANEX Restart and Job Networking Guide	SPX-03
SPANEX Scheduling Beginning User's Guide	SPX-12
SPANEX Automated Data Areas Manual	SPX-04
SPANEX Messages and Codes Manual	SPX-05
SPANEX Terminal User's Guide	SPX-07
SPANEX Installation and Maintenance Manual	SPX-09
SPANEX Documentation Index	SPX-10
Span Macros Manual	SPZ-02
Span Service Routines Manual	SPZ-03
SPSMFINF User Manual	SPI-01

This page intentionally left blank.

2 SPANEX Concepts

2.1 The Problem of Restart

All computer application systems, whether simple or sophisticated, are inevitably liable to unpredictable failure, from which recovery needs to be made as quickly and with as little operational complexity as possible. Processing interruptions may be caused by random failure of hardware, either of the mainframe CPU or of one or more peripheral devices; by software error, either of the System Control Program or within the application code; or even by a human error on the part of operations staff or of those responsible for the preparation or maintenance of the data that is used by or input to the system.

When such an unexpected failure occurs, processing needs to be restarted as soon as possible, but frequently the personnel responsible for doing this have neither a sufficiently detailed understanding of the processing logic of the application system, nor sufficient time available to ensure that the best and fastest solution is adopted. Obviously this difficulty can be considerably reduced by means of detailed operating instructions for each application system, but such written documentation is notoriously unreliable, with the inevitable tendency for people to guess or "remember" what the documentation instructs them to do.

The ideal solution to the problem of restart is to take the decision-making process out of the hands of those responsible for running the computer machinery and give it to the application development or implementation team, who can design and implement solutions away from the urgency of a busy production environment. The SPANEX Restart Facility provides the software to realize this concept.

2.2 Why Multi-Job Suites ?

Even though the SPANEX Restart Facility permits the use of extremely complex logic in the definition of the various steps of a job, it is desirable for a number of reasons to keep the complexity of each job to a minimum. This will aid maintenance and reduce the difficulty of making alterations and additions to the Job Control Language. It is also helpful to all involved if one job performs one logically separate function in the processing of a suite; this permits differing combinations of jobs to be run on different days, allowing daily, weekly, monthly, annual jobs to be created, and so on. Splitting a suite into a number of jobs can also help to speed up overall processing, as perhaps at some stages of the work there are some jobs that may be executed in parallel, while at other times everything may be dependent upon one key job such as a master file update process. SPANEX users have reported reductions in elapsed run times of up to 75% merely by implementing SPANEX Job Networking on a single application system.

The SPANEX Job Networking Facility provides the software to control any number of job inter-dependencies; it will automatically schedule jobs at the appropriate stage of execution of the suite according to the definition of job relationships prepared when the suite is designed. SPANEX will also allow any jobs to be omitted on any particular occasion, so that processing may be varied as desired, and this may be controlled dynamically, while the suite of jobs is executing, by means of simple application program calls to the SPANEX Utility, or by manual use of the same Utility. The many other facilities included for job scheduling functions permit the control of application systems of virtually unlimited complexity, with complete flexibility of control provided at all times.

3 SPANEX Solutions

3.1 The SPANEX Restart Solution

When the SPANEX Restart Monitoring option is used, SPANEX will maintain, transparently to the application system, status information pertaining to the progress of execution of each job in the system. On each occasion that the job begins execution, SPANEX will interrogate this control information to determine if an earlier attempt at running the job failed to complete successfully. If this is a clean start, processing will begin normally. If, however, this is a restart of the job, the SPANEX Restart Initialization process terminates with a code that specifies the point at which restart is to occur.

Selection of Jobsteps to be executed in both restart and clean start situations is performed by means of standard JCL Condition Code processing, thus permitting uninterrupted use of dataset disposition parameters. The interaction between these JCL parameters and the Condition Codes issued by the SPANEX Restart Initialization processor allows a considerable amount of logic to be incorporated into the architecture of jobs, so as to minimize the effort involved in recovering and restarting after a failure at any stage of application processing.

Thus, all the effort involved in providing an automatic restart capability for any job is exerted when the job is designed, during the planning of the JCL and the generation of the SPANEX Restart Control Module (the SPANEX definition of Job Restart and Network logic) for the job. Other options are also available in the Restart Control Module, such as whether or not the operator should have the option of overriding the restart point selected by SPANEX, whether or not the SPANEX Utility facility is permitted to update the restart status, whether or not job and step beginning and ending messages are required, and so on.

3.2 The SPANEX Job Networking Facility

The SPANEX Job Networking Facility is fully integrated with the SPANEX Job Restart Facility, and all the information required by SPANEX for controlling the suite of Jobs is contained in the same Restart Control Module. Some additional parameters specify that Job Networking is to be supported, and define any optional user parameters or access restrictions to the functions of Job Network control.

As SPANEX detects the successful completion of each job, all post-requisite jobs are scheduled, provided, of course, that any other pre-requisite jobs of these post-requisite jobs have also completed successfully. If a job fails at any stage, the SPANEX Restart Facility can be used to rerun from the point of failure; the remainder of the Job Network will then be processed when this job completes successfully. Actual submission of jobs for execution is performed by a SPANEX exit routine, which may be supplied by the user or may be one of the many standard routines supplied with SPANEX. This provides considerable flexibility and permits any desired technique to be used for the creation of jobs and their JCL.

When a SPANEX Job Network is initiated (or at any other time that fits in with specific user operational techniques), an optional "Job Check Sheet" is printed by SPANEX, tailored to this particular run of the Network, and this may be used by Data Control or Schedulers or operations personnel to keep an additional manual check and log of the execution of the jobs in the Network.

Information about the status of the whole Job Network and about individual jobs is available by use of a number of SPANEX Utility commands, so that manual control or override may be performed if necessary. Although all SPANEX Job Restart and Job Networking functions are fully automatic, installation personnel always have the capability of controlling the processing as an ultimate safeguard.

The addition of the SPANEX RCM Map facility further enhances the flexibility of a SPANEX Job Networking environment by providing additional display capabilities of real-time data concerning the progress of execution of the job suite. (For time-sharing users and users with a dedicated SPANEX terminal, this data is available as a full-screen display on 3270-type terminals.) A "Wall Chart" facility is also included, permitting easily-understood graphic display of the relationships between jobs in a Network.

Control of SPANEX Job Networks, and powerful enquiry commands, are provided by means of the SPANEX Utility. Extended TP support routines supply full 3270 support for use of the facilities of this Utility, with full-screen displays, on-line HELP data, and definable program function keys to minimize user keystrokes.

3.3 The SPANEX Automatic Calendar Facility

SPANEX includes an Automatic Calendar Facility. This is a significant enhancement to the SPANEX job scheduling features, but is implemented in a completely compatible way with existing SPANEX facilities. The Customer's existing jobs and job networks will execute unchanged, with no effect on their operation, and the manual calendar facilities of SPANEX 4.4 will continue to be fully supported. SPANEX Job Networks (either new networks or modified existing networks) can also now be defined with calendar processing, on a date basis, for both the job network or application as a whole and also for individual jobs within the network. As always with SPANEX, this is implemented in a completely flexible way, so that as much or as little work as desired can be input to designing the system.

SPANEX defines a Calendar day-type as a series of days within a calendar year. A SPANEX Calendar day-type may contain as few as one day, or as many as all the days in the year. For example, a Calendar day-type called “MONDAY” could be defined to include all the Mondays in the year.

Calendars are defined to SPANEX in an offline process similar to the existing definitions of Restart Control Modules (RCMs), and a SPANEX Calendar Table can contain up to 1000 individual Calendars. There is one “System” Calendar Table, and any number of “User” Calendar Tables. Each job network can use its own local User Calendar Table, and the System Calendar Table is available to all job networks.

The installation process for SPANEX Release 4.6 builds a default System Calendar Table for the current calendar year (and for the following year if the installation process is performed in December). This default System Calendar Table contains a reasonable selection of standard Calendars; these standard Calendars can be edited and augmented using the SPANEX Calendar definition process. Obviously, company-specific dates such as financial month-ends, and country-specific dates such as public holidays, can not be included automatically. The fact that the total number of available calendar definitions is unlimited means that all requirements can be catered for.

Usage of these defined Calendars is specified both at the Job Network level, and, within that, at the individual job level. Up to 10 Run Calendars can be specified for a Job Network; for example, an application may be specified to run on Mondays, Fridays and at month-end. Also, up to 10 Non-run Calendars may be specified, and these override the Run Calendars. For example, an application may be specified to run on Mondays, Fridays and at month-end, but not if one of those days is also a public holiday.

Within a Job Network, each job may also have up to 10 Run Calendars and 10 Non-run Calendars specified. Thus, for an application that is run only on Mondays and Fridays, a particular job may be run only on Fridays. The great flexibility of these specifications, combined with the ability to define an unlimited number of user-designed Calendars, means that all customer requirements can be met.

These new automatic calendar features of SPANEX exist alongside all the other facilities that were previously available. There are a very few minor logical conflicts between job definition options (for example, the EXCLUDE=NO job option, which states that the job must always be run as part of the application, is

incompatible with the concept of Non-run Calendar days for the same job, but not with Non-run Calendar days for other jobs or for the Job Network as a whole). Any such conflicts are resolved at the time the Job Network RCM is defined, and only valid logically consistent definitions are permitted to be passed to SPANEX.

In order to cope with the real world, SPANEX offers a flexible method of determining the current date. Many DP installations run batch work overnight following a business day, and the processing mix is based on the date of the business day, rather than changing promptly at midnight. SPANEX has a concept of Notional Start of Day, which is the time of day at which today is deemed to begin. This might be, for example, 6am, just before the start of the new day's online service. This Notional Start of Day time is defined globally as a default within the Calendar Tables, and can also be defined individually within each Job Network. Furthermore, the NETSTART command allows a specific run date to be entered, so that, for example, last Tuesday's Payroll application can be run at the weekend after a program amendment has been made.

Various SPANEX facilities have also been enhanced in line with the addition of Automatic Calendars. The Calendar Table definition process produces printed calendar maps for checking and for documentation purposes. Documentation produced by RCM generations now includes lists of Calendar specifications, and the NETSTART command can optionally list all jobs that have been included or excluded from the network run as a result of calendar processing. A new facility of the CSHEET (Job Check Sheet) command allows the production of Job Confirmation Sheets, which consist of lists of jobs (including averaged run times) within a Job Network that will be or would have been run on any given day of the year. Since Calendar Tables can be defined for any year from 1989 to 2087, predictions of application executions at any time in the future can be made. Output from various other SPANEX display and action commands is also enhanced to support SPANEX Automatic Calendars.

The SPANEX Automatic Calendar Facility can be used for all Job Networks and for all Jobs, or just for those that have particular date-based requirements. Since calendar usage is individually defined, no changes need to be made to existing SPANEX Job Networks.

If Calendar specifications are included in RCMs, then, for example, a standard daily series of SPANEX NETSTART commands could be issued for all Job Networks. NETSTARTs for Job Networks that do not run on the day in question will be ignored. For a Job Network that does run, the job mix will be automatically configured according to the calendar specifications.

Job scheduling staff can plan schedules for any time in the future. Naturally, there is an initial data-collection task to be performed for each application system, and for the customer installation as a whole, to determine the calendar definitions that will be required. Once this is done and established, it is, typically, a simple annual task to prepare the Calendar Tables for the following calendar year (this can actually be done at any stage, and the Calendar Tables made available to SPANEX).

4 Guide to the use of SPANEX Restart and Job Networking Features

In order to make use of the SPANEX Job Restart and/or Job Networking facilities, the user defines jobs to SPANEX in a control block known as a Restart Control Module (RCM). This is a Load Module, generated by assembling, by means of the Operating System Assembler, a set of SPANEX Macro statements (SPXJOB to define each Job, SPXSTEP to define each Jobstep within each Job, SPXRCM to define global options for all jobs). The format of these macro statements is explained in Section 5 of this manual, and examples of RCM generation are given in Section 11. No knowledge of Assembler Language programming is required in order to perform RCM generation.

4.1 SPANEX Automatic Job Restart

Automatic Restart using the SPANEX Job Restart facility can be performed only within each job of a suite, and this should be borne in mind when the division of a suite into jobs is being planned. The additional capability to support separate "recovery" jobs is also provided as part of the SPANEX Job Network facility.

Each job should be planned so as to include steps which will recover any datasets, databases, etc, that may be corrupted by an unpredictable failure at any point in the processing of the job. These extra steps (if any are necessary) may or may not be executed during a "Clean Start" of the job, depending on the JCL Condition Code processing specified when the JCL for the job is written.

The first step of each SPANEX Restart-controlled job must have the option "OPT=I" specified, with the name of the Restart Control Module specified by means of the "NET=" or "RCM=" parameter (this is optional if SPANEX Job Networking is in use), these values being part of the SPANEX JCL EXEC statement "PARM" field. Execution of a user program can be combined within the "OPT=I" step, where that user program may be restart-monitored (by specifying "OPT=IM" instead of just "OPT=I") or not, as required. If the execution of a user program is combined with the "OPT=I" step, it is treated as a logically separate step by SPANEX for restart purposes, and the user program is never executed in cases where the restart logic results in a non-zero return code from SPANEX "OPT=I" processing (see Section 4.6.5 on page 30 below).

Control of the restart of a job is performed by means of the Return Code from the SPANEX "OPT=I" (Restart Initialization) process. This Return Code is always zero for a Clean Start of the job. For a restart, the Return Code used is obtained from the Restart Control Module, based on the point at which failure occurred during the previous execution of the job. This Return Code can be modified after selection either by a user exit routine (see Section 6.1 on page 59 of this manual) or as a result of an operator response to WTO and WTOR messages issued if the "CONFIRM=YES" option is in effect. Actual execution of jobsteps during both restart and clean start situations is controlled by standard Operating System condition code processing, and it is the responsibility of the user when generating the JCL for the job to ensure the correct alignment of the codes specified in the

SPANEX Restart Control Module and in the JCL. SPANEX will issue error messages and ABEND if JCL errors of this nature are detected.

4.2 SPANEX Automatic Job Networking

Some additional optional parameters on the SPXRCM macro specify that SPANEX Job Networking is to be supported for a given Restart Control Module. When a Restart Control Module generated with the Job Networking option is in use, SPANEX will automatically provide network processing with no further action being required on the part of operations or programming staff. All jobs required for a given run of a job suite are automatically generated or scheduled by SPANEX when all defined predecessor jobs have completed successfully. There are no JCL parameter differences between the use of SPANEX Job Restart and the use of full SPANEX Job Networking.

Control of a SPANEX Job Network is provided by means of various commands of the SPANEX Utility. The "NETSTART" command is used to initiate the execution of a suite of jobs; all jobs defined with no pre-requisites will be scheduled to the Operating System, and an optional "Job Check Sheet" may be printed for manual control of the network. The "SCHEDULE" command is used to force the initiation of a specific job, either to override the job dependencies defined in the RCM, or to re-submit a job that failed and is now ready to be executed a further time. The "EXCLUDE" command makes possible the tailoring of a suite of jobs to a specific occasion; normally all possible jobs in a suite should be defined in the RCM, and EXCLUDE commands issued for all jobs not required for this particular run of the suite. All desired EXCLUDE commands should normally be issued before the NETSTART command, so that the entire job mix is defined before execution is begun, but the facility to EXCLUDE jobs while the Network is running is provided. For commonly used job combinations a control statement dataset should be created containing a sequence of EXCLUDE commands followed by the NETSTART command; this dataset can then be used as input to a batch execution of the SPANEX Utility, or can be dynamically invoked during any form of execution of the Utility by means of the SPANEX "INPUT" command, to cause the initiation of the job suite.

The "HALT" and "PROCEED" commands together provide the facility to suspend execution of an entire Job Network, and to resume execution of the Network, each with a single command. Execution of jobs can be interrupted either at the end of the current jobstep or at the end of the current job, and SPANEX preserves all necessary status information to enable the suite to be restarted. All suites of jobs are supported by this facility, no matter how complex or how many jobs are involved.

The "HOLD" and "POST" commands manage the implementation of user-specified events that may delay the processing of jobs. These may be used to include time dependencies or to permit any level of inter-dependency between different jobs, any dependencies on events or functions external to the processing of mainframe applications, and inter-dependencies between different job Networks.

The “CSHEET” command adds flexibility to the availability of SPANEX Job Check Sheets; these are available for printing either before or during the execution of the Job Network.

Actual submission of jobs for execution is performed by an installation-written Job Submit Routine, whose name is specified for each Job Network in the RCM. A large number of different sample submit routines is provided with the SPANEX system, each sample routine dealing with a particular combination of Operating System and job control technique. These routines may be used as supplied (the great majority of installations will find that their requirements are exactly matched by one or more of the sample routines), or may be modified or replaced at the installation's option. User submit routines, and the supplied sample routines, are described in Section 6.2 on page 62 of this manual.

4.3 The SPXRCM Macro

The SPXRCM macro terminates the input to the SPANEX RCM generation process and is also used to specify global options that apply to the whole suite of jobs being defined. Options available include the use of a user restart exit routine for all jobs in the suite (unless overridden by the specification of a user restart exit on an SPXJOB macro) as discussed in Section 6.1 on page 59 of this manual; the specification of the “CONFIRM” value as discussed below; the marking of the RCM as for testing purposes; the disabling of the Utility “UPDATE” facility for jobs defined by this RCM; whether or not SPANEX Job Networking is required for the job suite defined by this RCM; SPANEX Calendar definitions; and many other security and functional options.

All of the options of the SPXRCM macro may alternatively be specified on the QUICKNET macro if the SPANEX Quicknet feature is being used to define the network.

The “CONFIRM” option specifies whether or not the operator should be involved with the decision regarding the restart point of a failed job. This parameter may be specified on both the SPXRCM macro and the SPXJOB macro - the SPXJOB macro specification overrides the SPXRCM macro specification for one job only. If “CONFIRM=NO” is in effect, no operator intervention is invited during SPANEX Restart Initialization; if “CONFIRM=YES” is in effect, messages are issued to the operator, who has the option to permit the restart to continue as selected by SPANEX, to select a different restart point, to cancel the restart altogether or to start the job from the beginning.

By specifying “JOBNET=YES”, the use of SPANEX Job Networking is enabled, and the “NETRTN” parameter is used to define the “Job Submit Routine” to be used by SPANEX to generate jobs as required during the execution of the job suite. Job Networking may be specified for an RCM where there are no job dependencies, if it is desired to use the SPANEX Utility commands to generate jobs or to configure the network for specific occasions. The “ERASEON=YES” and “EONRTN=” parameters request particular functions to be invoked when SPANEX detects the end of processing for a Network; this is discussed further below. The “TITLE=” option of the SPXRCM macro permits the storing of a title for the Job Network in the RCM for the use of SPANEX or of any user program that requires access to RCM information.

Control of a SPANEX Job Network is exercised by means of the “EXCLUDE”, “HALT”, “HOLD”, “INCLUDE”, “NETSTART”, “POST”, “PROCEED”, “SCHEDULE” and “STATUS” commands of the SPANEX Utility, and the use of these can sometimes be sensitive. The “NETPASS=” option of the SPXRCM macro can be used to specify a password that must be included as a parameter on each of these commands for a particular Job Network. This can be used either as a security feature or as a safeguard against accidental errors in the running of the network.

The “USE=TEST” option causes the RCM to be marked as for testing purposes. When each job starts, a message is issued to the operator informing him that this is only a test run of the job. User programs may be replaced by dummy modules if desired and failure can be simulated by means of Job CANCEL commands or by forcing SPANEX abnormal termination action in order to test the action of the restart capability of the job. Thus the perhaps complex interaction of the Restart Control Module and the JCL “COND” parameters can be tested and corrected before the final installation of the job. Also any errors detected by SPANEX in the RCM definition or in the JCL can be corrected at this stage. The “USE=TEST” option also has the effect of overriding the default “ACK” option, if this was selected during the SPANEX installation process (see the SPANEX Installation and Maintenance manual for more details). For SPANEX Quicknet Networks, generated JCL steps will have a SYSUDUMP DD statement automatically added, if the “USE=TEST” option is specified on the QUICKNET statement.

The “OPTU=NO” option is available for security purposes, and will prevent any action of the SPANEX Utility facility “UPDATE” command to update the status of any job described by this Restart Control Module. Display functions of the Utility will still be permitted. See the description of the SPANEX Utility in this manual.

The “TSUPD=YES” and “TSNET=YES” options permit the use of update/delete commands and Job Network control commands, respectively, to be issued from a Time Sharing terminal (TSO support is standard with SPANEX) for this Restart Control Module. These commands will not be permitted from Time Sharing terminals unless these options of the SPXRCM macro are specified. Note that these functions are always supported from dedicated SPANEX terminals, and that the password and user security exit facilities are expected to be used to provide security for these users.

Calendar definitions specified on the SPXRCM or QUICKNET macro control the dates on which the Job Network as a whole is run. See Section 9 starting on page 135 of this manual for full details.

4.4 The SPXJOB Macro

The SPXJOB macro describes a job to the RCM generation process, and is also used to specify options that apply to one job in particular. Options that are available both on the SPXJOB and the SPXRCM macros are taken from the SPXJOB macro in preference to the SPXRCM macro. Options available are: the use of a User Restart Exit Routine for the restart of this job (as discussed in Section 6.1 on page 59 of this manual); the specification of the “CONFIRM” value as discussed below; the definition of any pre-requisite jobs to this one

within the Job Network (note that post-requisite and co-requisite jobs are defined by implication, and that any complexity of job inter-relationship can be defined with just the pre-requisite parameter); the definition of any "mutually exclusive" jobs (jobs that cannot, for whatever reason, run concurrently with this job); the specification of any "HOLD" events that must occur before this job is scheduled; the specification of the text of optional messages to be issued to the operator upon the start and/or successful end of the job; the specification of a descriptive title for this job, which will be used in informative messages to the operator, and also in a comment field on the SPANEX Job Check Sheet, allowing a job to be recognized by function as well as by jobname; the specification of any special processing to be performed by SPANEX for this job.

Calendar definitions specified on the SPXJOB or QUICKJOB macro control the dates on which one specific Job is run, providing that the Job Network as a whole is defined to run on those dates. See Section 9 starting on page 135 of this manual for full details.

All of the options of the SPXJOB macro may alternatively be specified on the QUICKJOB macro if the SPANEX Quicknet feature is being used to define the network.

4.4.1 SPXJOB Macro - Discussion of Operands

The "CONFIRM" option specifies whether or not the operator should be involved with the decision regarding the restart point of a failed job. This parameter may be specified on both the SPXRCM macro and the SPXJOB macro - the SPXJOB macro specification overrides the SPXRCM macro specification for one job only. If "CONFIRM=NO" is in effect, no operator intervention is invited during SPANEX Restart Initialization; if "CONFIRM=YES" is in effect, messages are issued to the operator, who has the option to permit the restart to continue as selected by SPANEX; to select a different restart point; to cancel the restart altogether, or to start the job from the beginning. CONFIRM=NO should be specified if the SPANEX Job Restart facility is not being used.

The "PREREQ" parameter permits the specification of one or more jobs that must be successfully executed before the job described by this SPXJOB macro is to be allowed to continue. Used in conjunction with the "JOBNET=YES" parameter of the SPXRCM macro (SPANEX Job Networking), this parameter defines the relationship between all the jobs of the suite; combinations of the "PREREQ" parameter on the various SPXJOB macros in the RCM allow any complexity of multiple pre- and post-requisite jobs to be defined. If the "JOBNET=YES" option of the SPXRCM macro is not specified, SPANEX will merely ensure that these pre-requisite jobs are not executing and do not have restarts pending on each occasion that this job begins; SPANEX will not check (unless Job Networking is in effect) whether the pre-requisite jobs have been executed and have already completed and vanished from the system, as this would prevent the specification of a pre-requisite job that does not always have to be run as part of the suite of jobs.

The "MUTEXCL" parameter permits the specification of one or more jobs that cannot execute concurrently with this job. This feature may be used for jobs that are not dependent on each other to the extent that they must be run in a particular order, but which may update the same dataset or use a large number

of mountable devices, such that they cannot be run at the same time. It is quite permissible to combine "PREREQ" and "MUTEXCL" relationships for the same job; although it is not logically sound to specify the same jobname as both a pre-requisite and mutually-exclusive. SPANEX will not allow mutually exclusive jobs to execute at the same time, but will suspend the second and subsequent jobs until they can run without conflict.

The "HOLD" option specifies a list of up to eight external events that must occur before this job is to be scheduled by SPANEX. The actual meaning of each event is arbitrary and user-determined. When all of a job's pre-requisite jobs have completed, SPANEX checks for any outstanding HOLD events before allowing the job to process. If a job's pre-requisite jobs have all completed, the last HOLD event to occur will cause the scheduling of the job.

The "TITLE" option permits a descriptive title for the job to be included in the RCM. This title will be used by the SPANEX Utility when describing job status; and also by SPANEX Job Networking, when the job is submitted for execution, and as a comment field on the SPANEX Job Check Sheet. SPANEX message editing is not performed on job titles. The job title value may be used for any purpose (such as Tape or Disk requirement notification to the operator) as the contents are not significant to SPANEX.

The "STRTMSG" and "ENDMSG" options allow the specification of the text of messages that will be issued at the start (either Clean Start or restart), and successful completion, respectively, of the job being described. The text of these messages may include any of the defined character strings by which the standard SPANEX message editing (string substitution) is performed (see the SPANEX General Usage Manual), so that variable data may be included.

The "EXCLUDE=NO" option is a security feature that allows a job to be retained in a Job Network in spite of attempts to issue an "EXCLUDE" SPANEX Utility command for it. This implies that the job is essential to the running of the network and must always be included in any subset of the jobs that is defined to be executed as a run of this Network. An "EXCLUDE" command for a job with this option will be rejected.

The "PROCESS=" parameter, which describes optional SPANEX special processing, is described in Section 4.4.2 below.

Note that, for a SPANEX Job Network (JOBNET=YES specified on the SPXRCM macro), SPANEX will schedule jobs that become eligible for scheduling at the same time in the order that they are defined in the RCM. Thus jobs which have common pre-requisites should be defined in priority order.

4.4.2 SPXJOB Macro - SPANEX Job Process Options

SPANEX Job Process Options permit the user to specify, for individual jobs, special processing that is to be performed by SPANEX to alter or augment the normal functions involved in a SPANEX Job Network. Process options are specified (if required) by means of the "PROCESS=" operand of the SPXJOB (or QUICKJOB) macro, and may be specified either singly or in any combination.

DELAY: this option permits a break in a SPANEX Job Network before the execution of the "delayed" job. Network execution will stop when scheduling of the job with the "DELAY" option is encountered, and this job must be scheduled manually by means of the SPANEX Utility "SCHEDULE" command.

WAIT-FOR-INPUT: this option permits a pause before execution of a job that requires input of any kind from a source other than jobs within its own network (eg manual user input, tapes from other locations, etc). If the input is available when the job becomes eligible for scheduling it will be scheduled by SPANEX; if the input has not arrived, the job should be scheduled manually by means of the SPANEX Utility "SCHEDULE" command. Messages will be issued to the operator to enquire as to whether or not input is available.

MULTIPLE-EXECUTION: this option permits a job within a SPANEX Job Network to be run successfully multiple times before its post-requisite jobs are scheduled. This feature may be used, for example, for a job which processes several batches of user input during a day, before the remainder of the suite is run. Messages will be issued to the operator, during the job termination process, to enquire as to whether further executions of the job are required. The SPANEX #SPXRSTU macro may be issued by an application program to inform SPANEX whether or not further executions of the job are required - this avoids the necessity of an operator interaction during the job termination process.

IGNORE-ERROR: this option permits scheduling of post-requisite jobs to continue even if a failure occurs in the last defined step of the job with this option. This function is similar to the "#SPXRSTU TYPE=SETOK" SPANEX macro instruction, but takes effect only for the last SPANEX step in the job. There is a "PROCESS=IGNERR" option on the SPXSTEP macro for ignoring errors in other steps of a job for SPANEX restart purposes.

RECOVERY: this option specifies that this job is a "recovery" job that is executed only when recovery of a different job is required. This job will be automatically "EXCLUDEd" whenever this Network is started. This job should be defined in the RCM as a pre-requisite of the job for which it is to recover. If that job fails, the "recovery" job must be manually scheduled by means of the "SCHEDULE" SPANEX Utility command; successful completion of the "recovery" job will cause the failed job to be automatically scheduled again by SPANEX.

EXCLUDE: this option specifies that, by default, this job is to be "EXCLUDEd" from the network. The NETSTART process will always automatically exclude this job; if it is required to be run, it should be reinstated by means of an

"INCLUDE" SPANEX Utility command, or by use of the #SPXRSTU macro within a user program.

4.5 The SPXSTEP Macro

The SPXSTEP macro describes a Jobstep to the RCM generation process, and is also used to specify options that apply to this Jobstep in particular. Options that are available are: the specification of the values of the Condition Codes by which the SPANEX restart process is to be driven, as described below; the specification of a Return Code threshold for this step, beyond which the user program is taken to have failed; the specification of the text of optional messages to be issued to the operator upon the start and/or successful end of the step; any SPANEX special processing required for this step.

All of the options of the SPXSTEP macro may alternatively be specified on the QUICKSTP macro if the SPANEX Quicknet feature is being used to define the network.

The "CODE" parameter has two sub-parameters which specify the Return Code to be issued from the SPANEX Restart Initialization process in order to restart this job at the correct point. The first CODE sub-parameter is the Return Code for failure within the execution of this step, such as an ABEND of the user application program. The second CODE sub-parameter is the Return Code for failure after the successful completion of this step but before any other SPANEX restart-controlled step has begun execution, such as an ABEND during a non-SPANEX-restart-controlled step or a failure during the Initiator/Terminator processing (eg allocation error, insufficient DASD space, etc). The values for these codes should be considered carefully in conjunction with the values to be specified in the JCL "COND" parameters for the job. It is suggested that, when initially allocated, these codes should be in ascending order through the SPXSTEP macros for each SPXJOB macro, perhaps in multiples of 4 or 10 to allow for insertions of additional steps at a later date. SPANEX makes the assumption that codes are in ascending order when building the text of informational messages to issue to the operator, although this has no other significance and should not be considered a restriction. SPANEX also detects when the same value has been specified for a code as one defined for an earlier step, and makes the assumption in this case (also for the building of message text) that this implies that a restart point for the step which has the second or subsequent occurrence of the code value is at a relatively early point in the job (at the point at which that code value is first specified). Again this should not be considered a restriction as ample information is provided to the operator to enable a realistic assessment of the situation to be performed.

The "ACCRC" option is used to specify the highest acceptable return code from the user program is this step. Any code higher than the value specified here, if returned by the user program, is taken by SPANEX to indicate that the user program has failed. The default acceptable return code is zero. Note that this value can also be specified on the SPANEX JCL parameter for each step; any value specified in the JCL, including the value zero, will over-ride the specification on the SPXSTEP macro. The advantage of specifying the acceptable return code value on the SPXSTEP macro, rather than in the JCL, is that, if

SPANEX Quicknet or the SPXRCCC0 routine is being used, the value is available when making a retrospective decision as to the success or failure of the step.

The “STRTMSG”, “ENDMSG” and “FAILMSG” options allow the specification of the text of messages that will be issued at the start, successful completion, and unsuccessful completion, respectively, of the step being described. The text of these messages may include any of the defined character strings by which the standard SPANEX message editing (string substitution) is performed (see SPANEX General Usage Manual), so that variable data may be included.

Process Options available on the SPXSTEP macro include “PROCESS=IGNERR” and “PROCESS=NOGLOG”. The “PROCESS=IGNERR” option causes a failure of the user program in this step to be ignored for SPANEX restart purposes. Note that any defined SPANEX Abnormal Termination Action will still be taken by SPANEX (see SPANEX General Usage Manual), but that the restart status of this step will be set as normal. This function is similar to the “#SPXRSTU TYPE=SETOK” SPANEX macro instruction. The “PROCESS=IGNERR” option should be specified on the SPXJOB macro in order to control SPANEX Job Networking dependent job scheduling in the last step of a job. The “PROCESS=NOGLOG” option suppresses the entry of records on the network Global Log dataset for the logging of the start and normal end of this step (other Global Log records and failures in this step are always logged).

4.6 SPANEX Implementation Considerations

4.6.1 Specifying a SPANEX Catalog

SPANEX Restart and Job Network processing uses a dataset known as the “Catalog” for the storage of Job and Network status information. This Catalog dataset may be a system or user catalog (CVOL, VSAM or ICF), or may be a native VSAM KSDS. The Catalog dataset, whatever its access method and organization, must be shared by all CPUs in a multi-CPU or multi-access spool environment. When installing SPANEX for the first time, a decision must be taken whether or not to use SPANEX cross-CPU serialization. This option is specified by means of the RESVOL parameter of the #SPXGEN installation macro when SPANEX is installed and specifies the Volume Serial Number of a disk volume to be used via RESERVE to serialize access to the SPANEX Catalog, and, in the case where a CVOL Catalog is being used, will normally be the serial number of the volume containing this Catalog. The RESVOL parameter may be omitted or specified as “000000” if shared DASD is not to be used. SPANEX uses a high-level index of “#SPANEX1” on all its Catalog operations. If a CVOL Catalog is specified (by the use of the CATALOG=CVOL option of the #SPXGEN macro), each batch use of the SPANEX Restart or Job Networking facilities will optionally (dependent on a further SPANEX generation option) force the SPANEX Catalog to be connected to the system Master Catalog (by invoking IDCAMS for MVS). If a RESVOL is specified, the SPANEX Job Network processor will RESERVE this volume for a short period at each invocation (if the volume is mounted on a shared device) to ensure serialization of the Network scheduling process. Note that a RESVOL specification is essential in a shared DASD environment. The implications of the selection of the type of Catalog dataset to be used for SPANEX are fully explored in the SPANEX Installation

and Maintenance manual. If a system or user Catalog is defined for SPANEX, the Catalog may be shared by other users without concern.

Note that, for a given SPANEX system, all physical access to the SPANEX Catalog is performed by means of parameter-driven calls to a single SPANEX module (SPXM0210 for CVOL, VSAM or ICF Catalogs, SPXM0370 for native VSAM KSDS Catalogs). Thus, if desired, the user installation can easily implement any desired alternative input/output technique. The SPXM0210 and SPXM0370 source modules supplied with the SPANEX system have comments at all places that would need to be altered if any alternative catalog dataset technique were to be adopted. The use of a system or user Catalog has the advantages of automatic indexing and the absence of JCL or dynamic allocation requirements. The use of a native VSAM KSDS Catalog has the advantage of the support for dual copies of the Catalog for integrity and recovery purposes; the CATMAINT SPANEX command is also supplied for the maintenance of the dual copies of the Catalog data. "Local" (ie user-specific) KSDS Catalogs may be implemented by adding SPXCAT1 (and, optionally, SPXCAT2) DD statements to each SPANEX jobstep or utility execution.

4.6.2 Selection of Job names

It is important to understand the mechanism that SPANEX uses to record status information in the Catalog. Job Network information is held keyed on Network name, and Job information is held keyed on Jobname only. If two Job Networks containing Jobs with the same Jobname are run simultaneously, the Job status information stored by one Job Network will clash with that stored by the other. The results can be unpredictable if this happens.

Thus a naming convention for Jobs should be developed to ensure that this cannot happen. It is quite acceptable for more than one Job Network to contain any given Job Name, provided that these Networks will not be executed at the same time.

4.6.3 Job and Jobstep Organization

For a job using the SPANEX Job Restart facility, it is essential that the last step defined for this job in the Restart Control Module is always executed in the situation where the job has completed normally; if this does not fit in with the logic of the job, an additional dummy step with a program name of "IEFBR14" should be defined at the end of the job to permit SPANEX normal-end-of-job processing to be performed.

Conversely, it is important that the last SPANEX step does not execute successfully in the situation where an earlier step of the job failed. In the simplest case this can be achieved by specifying the JCL parameter "COND=(0,NE)" on this last step, which will cause any non-zero completion codes of earlier steps to prevent the execution of the last step. Alternatively, the use of the Quicknet or Retrospective Condition Code features will allow SPANEX itself to check back through jobsteps, and to ensure that the correct scheduling decisions are taken.

For a job that is part of a SPANEX Job Network, at least one SPANEX restart-controlled step must be defined for the job even if this job does not require the SPANEX Restart facility. Again, this step must always be executed if the job completes successfully, to permit SPANEX end-of-job processing to be performed.

4.6.4 Executing a SPANEX Job Network

Jobs which are part of a SPANEX Job Network should not be manually submitted for execution - SPANEX will automatically schedule dependent jobs, and the SPANEX Utility "NETSTART" and "SCHEDULE" commands should be used to invoke the whole network or an individual job, respectively. Before entering the "NETSTART" command to initiate a run of the entire Job Network, the "EXCLUDE" command should be entered for each job that is not to be run for this execution of the suite; the "EXCLUDE" command with the "NOW" option may be issued for a job while the network is executing (provided that job has not yet begun execution) but it is not recommended that this is made normal practice. The effect of an "EXCLUDE" command can be undone, if this is required, by means of the "INCLUDE" command for the same job, which will submit the job for execution if all prerequisite conditions have been satisfied. The effect of excluding and including jobs can also be performed dynamically by means of assembler-language application program calls to SPANEX using the #SPXRSTU macro (see Section 6.5 on page 81 of this manual).

The NETSTART, EXCLUDE and INCLUDE commands also support the "NEXT" parameter, which allows these commands to be entered for a Network that is already active. The fact that the Network is to be NETSTARTed a further time, with appropriate jobs excluded or included, is remembered by SPANEX until the present Network execution is complete, and then the new run is automatically initiated.

The "NETSTART" command will initiate the execution of all jobs in the network that are not "EXCLUDEd", and have no defined pre-requisite jobs, or whose pre-requisites are all "EXCLUDEd". If SPANEX automatic scheduling of a dependent job is to be overridden, the "SCHEDULE" command will force the execution of a single job. The "SCHEDULE" command should also be used to re-submit a job that failed and is to be re-run.

SPANEX Job Check Sheets, which permit a manual record of the execution of the jobs in a network, may be produced as a result of the "NETSTART" command, and may also be produced either in advance or after the start of the network by means of the "CSHEET" command to the SPANEX Utility. For MVS users, the Job Check Sheets may be "spun-off" to any SYSOUT class required, so that they are immediately available for printing.

The "HALT" and "PROCEED" commands are provided to take account of the probably unusual situation where execution of one or more entire job networks is to be temporarily suspended. The use of these commands should be evaluated by each user installation, but they may have particular application, for example, if the CPU has to be given over to some other purpose, perhaps for engineering work, at short notice, or when work is to be transferred to another CPU in a multi-CPU installation. These commands permit a graceful close-down of the work of a job network, and a rapid resumption of the work by means of a single SPANEX command.

4.6.5 Use of OPT=I and OPT=M

Optional processing by SPANEX is requested by means of the "OPT=" parameter option. Two options are used by the SPANEX Restart and Job Networking facilities, option I which is used on the first SPANEX step in a job, and option M which is used on every step in a SPANEX job which performs useful work and at which a restart may be performed by SPANEX in the event of a failure.

OPT=I must always be specified on the first step defined in the RCM to request initialization processing. This process includes the analysis of whether a rerun of the job is required because of a previous failure, and, for a SPANEX Job Network, this process checks that all pre-requisite jobs of this job have completed successfully. If all pre-requisites have not completed, at this point authority is requested from the operator for the job to continue. Thus, in a SPANEX Job Network, it is recommended that the SPANEX OPT=I step is always the first in the job, so that the job can be immediately terminated if it is run out of order.

The OPT=M process involves the running of an operational program under the control of SPANEX, and must be specified at least once in a SPANEX job that either uses the SPANEX Restart facility or is part of a SPANEX Job Network. When the step that is run is the last step as defined in the Restart Control Module for this job, then, for a network, dependent jobs are scheduled at the completion of the user program. The options I and M can be combined in the same step (by specifying OPT=IM), which means that real work can be performed within the jobstep that is used for restart analysis, in order to reduce the total number of jobsteps, or to use SPANEX networking for a single-step job. When OPT=IM is specified and a SPANEX restart of the job is performed, the user program specified in the OPT=IM step is never executed when the restart condition code selected is greater than zero.

Steps which use either of the options I and M must be defined in the RCM. Note that if the job "STRTMSG" option of the SPXJOB macro is used, this message will be issued from the first OPT=M step in which the user program is executed, for both clean start and restart conditions. Thus it may be of advantage always to specify option M when option I is specified and to specify a program name of "IEFBR14" to ensure this message is issued at the start of a job.

To take an existing job suite and install SPANEX Job Networking, although the minimum requirement is that one step be defined per job, the recommended approach is to define two steps in each job, and add these steps to each job, one at the beginning (specifying OPT=I or OPT=IM) and one at the end, executing program "IEFBR14" and specifying OPT=M. In this way, no change at all need be made to existing and well-tried JCL and SPANEX job networking can be implemented in a very short time. For the fastest possible implementation of SPANEX into an existing job suite, the Quicknet feature is probably the optimum technique, as this does not require any JCL changes at all. The Quicknet feature is fully described in Section 10 of this manual.

4.6.6 The SPANEX Global Log

The SPANEX Global Log is an optional feature of the SPANEX Job Networking system that records messages defining all significant events in the processing of the network. Such events include the starting and ending of jobs and jobsteps, and any SPANEX Utility commands which cause any change in the status of any job or of the whole network. The Global Log is a single sequential dataset, to which a record is added for each logged event. There may be one Global Log dataset per network, or one dataset may be shared between multiple networks. The NETSTART processor clears the dataset (except for shared Global Logs) and inserts a single network start log record and a list of jobs excluded from this run at the beginning, but the optional NETSTART user exit is called before this is done so that the contents of the dataset may be saved by this user routine if required. Note that for the first use of a Global Log dataset a null end-of-file mark should be placed in the dataset before invoking SPANEX. SPANEX will place a null end-of-file mark at the start of a non-shared Global Log dataset during the NETSTART process.

The Global Log option is selected by means of parameters on the SPXRCM macro when the job network is defined. Note, however, that there is a SPANEX generation option that may make the use of a Global Log compulsory for all Job Networks in an installation. Parameters that may be specified include the DDNAME to be used for the Global Log dataset by the NETSTART processor (which may need to access several Global Logs for different job networks from within the same SPANEX Utility execution), the dataset name of the Global Log dataset for this network (MVS only, but may be specified for other Operating Systems for compatibility and to permit easier migration), and whether or not this Global Log dataset is shared by other SPANEX Networks.

A unique DDNAME must be defined for each SPANEX Global Log dataset in the installation, as this is the means used by SPANEX for serializing updates to the dataset.

For MVS, Dynamic Allocation is used for the Global Log dataset (which must be catalogued and mounted), unless a DD statement is provided for the dataset, in which case the DD statement will be used for compatibility with the other Operating Systems. For MVS users, however, it is recommended that no DD statements are supplied and that SPANEX Dynamic Allocation should be allowed to occur for the Global Log.

The SPANEX Utility LOG command is provided to permit user data records to be added to the Global Log, and the TRACE command supplies search and display capabilities for Global Log records by jobname, time range, or by any string value such as message identifier, ABEND code, etc.

Although the SPANEX Global Log is an extremely powerful and useful feature, the possible overheads should be borne in mind, particularly by users who have a small surplus machine capacity. For example, a typical MVS Global Log message would require the following processing by SPANEX: the formatting of the message, dynamic allocation (with the FREE=CLOSE option) of the Global Log dataset, a RESERVE on the Global Log device (if on shared DASD), an OPEN (with the EXTEND option), the writing of the message, a CLOSE, a RELEASE of the Global Log device. This processing will occur twice for each jobstep plus twice for each job plus once for each SPANEX Utility command that alters the status of the network or of any job. The “PROCESS=NOGLOG”

operand of the SPXSTEP macro can suppress the routine step-start and step-end Global Log records for individual steps if required, although errors will always be logged.

Note that it is essential that enough Direct Access space is allocated to the Global Log dataset. SPANEX writes unblocked 133-byte records, and no recovery is attempted if an out-of-space condition (eg SD37 or SB37 Abend) occurs. Such an error in a Global Log operation will effectively stop the execution of the Job Network using that dataset. I/O errors in a SPANEX Global Log dataset are reported to the master console.

4.6.7 SPANEX End-of-Network Detection

At various points during the processing of a Job Network, SPANEX will perform a function known as end-of-network detection. This occurs at the successful end of each job, during the execution of an EXCLUDE SPANEX Utility command with the "NOW" option, and when an UPDATE SPANEX Utility command is issued which sets the status of a network job to "SUCC" (successfully completed). The end-of-network detection function checks the status of all other jobs in the network in order to determine if all jobs in the network have completed successfully. When all jobs have completed, the appropriate user-defined functions are performed, the Network status record is updated to show that the network is complete, and the Global Log is terminated and is available for printing or saving. End-of-network processing also detects a pending Netstart (caused by the use of the NEXT option of the NETSTART command), and will automatically start the next run of the Network. A user end-of-network exit routine may be used to print or save the Global Log dataset for a network, as this is not invoked until the Global Log has been terminated. A sample end-of-network exit routine to perform this function for MVS users is provided in the SPANEX source library with a name of SAMPEONR; this routine as provided will dynamically allocate a SYSOUT=A dataset and print the entire contents of the Global Log.

4.6.8 The SPANEX Command Library

The SPANEX Utility supports a command library so that pre-determined sequences of commands may be defined and stored and called-up when required. The SPANEX Utility "INPUT" command is used to invoke a series of commands. Thus it is possible to define, for example, a sequence of "EXCLUDE" commands followed by a "NETSTART" command in order to start a frequently-used combination of jobs in a SPANEX network. This library may be a standard source Partitioned Dataset, with 80-byte logical records, or may be a CA-PANVALET or CA-LIBRARIAN library. A PDS is accessed by the SPANEX Utility by means of the SPXUTLIB DD statement, a CA-PANVALET library is accessed by means of the SPXUTPAN DD statement, and a CA-LIBRARIAN Master is accessed by means of the SPXUTMST DD statement. One or more of these DD statements may be supplied; if more than one is present, the PDS only is always searched for the requested member, unless the LIBTYPE parameter of the "INPUT" command is used to override this.

4.6.9 SPANEX Scheduling Logic

This section briefly describes the logic used by SPANEX when determining the next job to schedule. This logic is invoked at the successful completion of each job in a SPANEX Job Network.

Starting with the terminating job, SPANEX looks down the network for any jobs that are post-requisites (ie jobs that define this job as a pre-requisite). Taking each of these jobs in turn, all pre-requisites of the job are checked for successful completion by means of an upward scan, and, if this is true, the job is scheduled. During this upward scan, if a completed job is encountered before any incomplete jobs are found in this chain, then the upward scan is satisfied at this point and does not continue searching for further incomplete jobs. Before any job is actually submitted, a check is made for any other active jobs with which execution is mutually exclusive; if any are found, the job is suspended until the conflict is resolved.

After the completion of the scheduling of post-requisite jobs that can now be run, SPANEX checks for any other jobs in the network that have been held up because of a mutual exclusion relationship with this job. If any are found, they are then processed again to determine if they can now be run.

"Excluded" jobs are treated as follows: When searching downwards through the network (ie to find a post-requisite job), excluded jobs are treated as though they have just completed, and this entire piece of logic is executed again recursively, with the excluded job being taken as the starting point for the network scans. When searching upwards through the network (ie checking completion of pre-requisites of a post-requisite job), excluded jobs are treated as though they didn't exist at all, and their dependencies are propagated to the job being processed.

4.6.10 SPANEX Inter-Network dependencies

This section discusses how dependency relationships between separate SPANEX Job Networks may be implemented. The technique uses the "Event Hold" feature, which may be defined in the RCM for a job, or which may be specified dynamically by means of the "HOLD" SPANEX Utility command.

Each job in a SPANEX network may be defined with up to eight arbitrary "events" which must have occurred, in addition to all other dependencies, before the job will be scheduled by SPANEX. An event is signalled complete by means of the "POST" SPANEX Utility command; an event may be signalled incomplete by means of the "HOLD" SPANEX Utility command. The HOLD and POST commands may be issued for a network by jobs in a different network, by executing the SPANEX Utility as a jobstep. The HOLD and POST functions can also be issued by means of an application program macro call to SPANEX, but in this case they are limited to operating on jobs within the same Network as the application program making the call.

A considerable level of complexity in the inter-dependency of separate Networks may be achieved by the use of this feature, and, since the inter-dependency is at the job level, entire Networks may be intertwined with one another if required.

The eight arbitrary events supported for each job may be assigned individual meanings by the user installation, although the convention is to reserve event 1 to signify a time dependency - this event can be set or reset at certain times of the day in order to restrict when a job is to be run.

4.7 Job Control Language Recommendations

This section contains some recommendations for JCL standards in order to make the best use of the SPANEX Restart and Job Networking facilities.

4.7.1 Generation Data Groups

A problem can exist with Generation Data Groups regarding the restart and re-running of jobs that create a new level of GDGs (by referring to level +1, etc). It is the Span Software Consultants recommendation (whether or not SPANEX is being used) that new GDG levels are not created in jobs which run for a significant length of time or which may ever need to be restarted. A separate dummy job should be created, executing perhaps program IEFBR14, that runs before the job that creates the new generation, and whose sole function is to allocate the new level by referring to it as level +1.

For disk GDGs this dummy job must also allocate the direct access space; for tape or disk GDGs the Catalog must be updated, via JCL, to reflect the new latest level. The processing job can now refer to the new output level as level +0 and may be run many times without requiring any manual Catalog adjustments; these two jobs should ideally be connected within a SPANEX Job Network.

4.7.2 Temporary or Short-term Datasets

In the same way as above, new disk datasets should not be unconditionally allocated in long-running processing jobs. A similar dummy job should be run that merely allocates the space required for such datasets so that the “real” job can refer to these as “OLD”. Similarly these short-term datasets should not be deleted within a processing job unless it can be guaranteed that the data is no longer required regardless of re-run conditions; it is far better to schedule a post-requisite dummy job to perform such deletes, connected via a SPANEX Job Network so as to ensure correct synchrony of execution.

4.8 The SPANEX Quicknet Feature

The SPANEX Quicknet feature is an additional feature of SPANEX designed to make the implementation of SPANEX Job Networking simpler. Its use applies in the main to existing applications suites where SPANEX is not already included, and to testing work, where a full-scale SPANEX implementation can not always be justified.

SPANEX Quicknet consists of a set of RCM generation macros which are used instead of the normal SPXJOB, SPXSTEP and SPXRCM macros, and which

provide simplified parameters to minimize the effort required to define a Job network. There are also some additional supplied submit routines whose use is automatically specified within the new RCM generation macros.

The result of this is that, providing all Job JCL is held on one of the SPANEX-supported JCL library formats, SPANEX Networking is implemented with no additional or modified JCL for SPANEX, and with optional condition code checking for all steps of each job (MVS only). All SPANEX Job Networking facilities are supported, including Wall Charts, Job Check Sheets, Global Log, etc.

Full details of the SPANEX Quicknet feature can be found in Section 10 of this manual.

4.9 SPANEX Retrospective Condition Code Checking

SPANEX provides a function for checking the validity of condition codes returned by all jobsteps in a job. This is accomplished by executing a special SPANEX module as the last step of the job, and the JCL to do this can be automatically generated when the SPANEX Quicknet facility is in use (see Section 10 of this manual).

However, this facility may also be used by jobs that are set up not using Quicknet, and the JCL to be used for the last step of the job is shown below. Note that the special checking program must run in a step defined in the RCM for the job, and must run as a SPANEX execution with OPT=M specified as a SPANEX option. The jobsteps checked may be SPANEX-controlled or not, but any steps that can validly return condition codes greater than zero must be defined in the RCM as such. The ACCRC parameter of the SPXSTEP macro can be used to define valid condition codes for all jobsteps.

```
//stepname EXEC PGM=SPANEX,
//          PARM='SPXRCCC0,OPT=M/SCANOPT=n'
```

The EXEC statement for the last jobstep in the job must be coded as shown above, where the “stepname” is as defined in the RCM for the last step of the job, and “n” is the condition code scan option, as described below:

SCANOPT=0 is a null option and causes no checking to be performed;

SCANOPT=1 specifies that all jobsteps in the job are to be checked, and all condition codes greater than zero are to represent an error;

SCANOPT=2 specifies that only jobsteps defined in the RCM are to be checked, and that condition codes greater than zero are to represent an error except where the ACCRC parameter was specified, when this value is used for the checking (see the description of the SPXSTEP macro on page 46 of this manual);

SCANOPT=3 specifies that all jobsteps in the job are to be checked. Condition codes greater than zero are assumed to represent errors, except in cases where a jobstep is defined in the RCM and has an ACCRC parameter specified.

SCANOPT=4 specifies that all jobsteps in the job are to be checked. Condition codes greater than zero are assumed to represent errors, except in cases where a jobstep is defined in the RCM and has an ACCRC parameter specified. SPANEX Automatic Step Restart is also supported, using the restart condition codes specified via the CODE= parameter of the SPXSTEP or QUICKSTP macro (see the description of the SPXSTEP macro on page 46 of this manual for details of the CODE= parameter).

Note that ABENDs are always recognized as errors except in the cases for scan options 2 and 3 where the PROCESS=IGNERR option is specified on the SPXSTEP macro for the abending jobstep.

5 Generating an RCM or SPANEX Job Network

The generation process for a SPANEX RCM consists of compiling, by means of the system Assembler, a series of SPANEX macro statements which define to SPANEX the jobs and jobsteps which make up the application system. Each RCM requires one or more SPXJOB macros (each describing a job within the job suite or network), each one followed by one or more SPXSTEP macros, each of these describing a jobstep within the job described by the preceding SPXJOB macro. When all jobs and steps have been defined, the Assembler input is terminated by means of an SPXRCM macro, which delimits the input and describes to SPANEX parameters which apply to the RCM as a whole. These SPANEX macros are described in this section of this manual. No knowledge of Assembler language is required in order to perform SPANEX RCM generation. See also Section 10 of this manual for a discussion of the Quicknet feature, which provides higher-level alternatives to the SPXJOB, SPXSTEP and SPXRCM macros.

Output from the Assembler consists of an object module for the RCM (if no severe errors were found in the generation) and an Assembler listing which contains messages describing any errors or warnings encountered during the generation, together with a map of the jobs defined, which may be used as operating documentation and as a guide to the coding of the JCL for these jobs. All accepted parameters of the SPXRCM macro are listed at the start of the output listing from the RCM generation process, and these should be checked to ensure there were no syntax errors in the user input. Error and warning messages are documented in the SPANEX Messages and Codes manual.

Notation for Macros in this Section

Square brackets, [ ], denote (1) that a macro parameter is optional: if an entire parameter with its options is enclosed in square brackets, then that parameter is optional; (2) that a range of values is permissible for a given parameter: if a series of possible values for a parameter is shown in a vertical manner, all surrounded by additional square brackets, then choose one from the values shown.

Normal parentheses, ( ), signify that parentheses should appear when the macro is coded, denoting, for example, a list of sub-parameters.

Underlining, ___, denotes default values for parameters.

Coding Conventions

Standard Assembler language coding conventions are used for SPANEX macros:

- Labels (ie Jobnames, Stepnames, RCMname) must begin in column 1;
- Macro names and operands may be placed anywhere on the statement but are conventionally in columns 10 and 20 respectively for SPANEX;

- Continuations are indicated by a non-blank character in column 72 of the continued statement;
- Continuation statements must begin in column 16.

5.1 Sample JCL for RCM Generation

```
//RCMGEN    PROC  RCM=
//RCMASM    EXEC  PGM=SPANEX,
//          PARM='ASMA90,4/DECK,NOOBJ,LIST,XREF'
//SYSPRINT  DD SYSOUT=A
//SYSUT1    DD UNIT=SYSDA,SPACE=(CYL,(5,1))
//SYSUT2    DD UNIT=SYSDA,SPACE=(CYL,(5,1))
//SYSUT3    DD UNIT=SYSDA,SPACE=(CYL,(5,1))
//SYSLIB    DD DSN=SPANEX.SOURCE,DISP=SHR
//SYSPUNCH  DD DSN=&&LOAD,DISP=(,PASS,DELETE),
//          SPACE=(TRK,(5,1)),
//
UNIT=SYSDA,DCB=(BLKSIZE=400,LRECL=80,RECFM=FB)
//RCMLKED   EXEC  PGM=SPANEX,
//          PARM='IEWL/LIST,XREF,OL'
//SYSPRINT  DD SYSOUT=A
//SYSUT1    DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSLIN    DD DSN=&&LOAD,DISP=(OLD,DELETE)
//SYSLMOD   DD DSN=user.load(&RCM),DISP=SHR
//          PEND

//USERRCM   EXEC  RCMGEN,RCM=rcmname
//RCMASM.SYSIN DD  *
       .             .     .
       .             .     .
  enter user input (SPXJOB, SPXSTEP, SPXRCM, QUICKJOB
              QUICKSTP, QUICKNET macros) here
       .             .     .
       .             .     .
/*
//
```


5.2 SPXJOB Macro - Define a Job to SPANEX RCM

The SPXJOB macro should be used to define each job in a suite of one or more jobs that is described by an RCM generation. For each job there should be one SPXJOB macro, followed by one SPXSTEP macro for each SPANEX restart-controlled step in this job, specified in the order that these steps appear in the job JCL. Non-restart-controlled jobs may be included in a suite of jobs that is to use the SPANEX Restart facility, but SPXJOB macros are not required for these unless they are part of a SPANEX Job Network, in which case the SPXJOB macro must be specified, followed by at least one SPXSTEP macro. There may be no duplicate job names within an RCM generation. Note that for SPANEX Quicknet (a quicker and simplified method of generating SPANEX RCMs) the QUICKJOB macro may be used as an alternative to the SPXJOB macro; all options of the SPXJOB macro may also be specified on the QUICKJOB macro - see Section 10 in this manual.

format:

```
jobname   SPXJOB   [STRTMSG=mmm]   [,ENDMSG=mmm]

                   [,PREREQ=(jjj,jjj,...)]

                   [,MUTEXCL=(jjj,jjj,...)]

                   [,TITLE=ttt]

                   [          [YES]]
                   [,CONFIRM=[   ]]   [,EXIT=rtnname]
                   [          [NO ]]

                   [,EXCLUDE=NO]   [,HOLD=(n,n, ...)]

                   [          [DELAY,    ...]]
                   [,PROCESS=[WFI,       ...]]
                   [          [MULT,     ...]]
                   [          [IGNERR,   ...]]
                   [          [RECOVERY,...]]
                   [          [EXCLUDE,  ...]]

                   [,RUNDAYS=(dddd,dddd, ...)]

                   [,NONDAYS=(dddd,dddd, ...)]

                   [,MEMBER=membername]
```

where:

jobname - JCL jobname, max=8 bytes, required
- specifies the jobname for the job described by this SPXJOB macro. This jobname must be the jobname that appears in the JCL "JOB" statement for this job, and must be unique in this RCM.

STRTMSG= - character text enclosed in single quotes
- specifies an optional message to be issued to the console operator when this job begins execution. This message will also appear on the Global Log dataset if one is defined. If a quotation mark appears in the message it should be entered as two consecutive quotation marks; if an ampersand (&) appears in the message it should be entered as two consecutive ampersands. SPANEX will prefix the message with the identifier "SPX899JS" and will process the text of the message with the SPANEX message text editing facility to enable string substitution to be performed (see SPANEX General Usage Manual).

ENDMSG= - character text enclosed in single quotes
- specifies an optional message to be issued to the console operator when this job finishes execution. This message will also appear on the Global Log dataset if one is defined, and will appear there instead of the normal SPX905I message issued for a successful job end. If a quotation mark appears in the message it should be entered as two consecutive quotation marks; if an ampersand (&) appears in the message it should be entered as two consecutive ampersands. SPANEX will prefix the message with the identifier "SPX899JE" and will process the text of the message with the SPANEX message text editing facility to enable string substitution to be performed (see SPANEX General Usage Manual).

PREREQ= - one or more jobname symbols, each max 8 bytes
- specifies a list of one or more jobs which must have completed successfully before this job is to begin execution. Since multiple jobs may be specified, and the same pre-requisite job may be specified by multiple SPXJOB macros, a Job Network of any complexity may be defined in the RCM. The "PREREQ" parameter is valid whether or not SPANEX Job Networking is specified on the SPXRCM macro. If more than one job is specified, enclose the list of jobs, separated by commas, in parentheses. All jobs specified must be defined by SPXJOB macros in this RCM generation.

MUTEXCL= - one or more jobname symbols, each max 8 bytes
- specifies a list of one or more jobs which may not execute at the same time as this job. Multiple jobs may be specified, and the same mutually-exclusive job may be specified by multiple SPXJOB macros. When several mutually-exclusive jobs become eligible for scheduling at the same time, the job that is defined first in the RCM will be the one to be scheduled. It is not

necessary to define mutual-exclusion relationships in both directions, ie if JobA is mutually exclusive with JobB, it is not necessary also to specify that JobB is mutually exclusive with JobA, although it is not an error to do so. The "MUTEXCL" parameter is valid only when SPANEX Job Networking is specified on the SPXRCM macro. If more than one job is specified, enclose the list of jobs, separated by commas, in parentheses. All jobs specified must be defined by SPXJOB macros in this RCM generation.

TITLE= - character text enclosed in single quotes
- specifies an optional title for the job described by this SPXJOB macro. This title will appear in a SPX996I message issued to the operator when this job is scheduled by SPANEX, or in response to SPANEX Utility enquiries, and is printed on the SPANEX Job Check Sheet. If a quotation mark appears in the message it should be entered as two consecutive quotation marks; if an ampersand (&) appears in the message it should be entered as two consecutive ampersands.

CONFIRM= - fixed keyword value
- specifies the "CONFIRM" option to be used for restart of this job. This overrides the option specified on the SPXRCM or QUICKNET macro, for this job only. If "CONFIRM=YES" is specified, operator authority will be requested for a restart of this job after a failure. If "CONFIRM=NO" is specified, SPANEX will restart in this job after a failure without allowing the operator to override the restart point or to cancel the restart.

EXIT= - Load Module name, max=8 bytes
- specifies the name of a User Exit routine load module that is to be fetched and executed by the SPANEX Restart facility each time this job begins to execute. This overrides any exit routine specified in the SPXRCM or QUICKNET macro, for this job only. This routine has the ability to request or override a restart of this job. See Section 6.1 on page 59 of this manual for details of user restart exit routines.

EXCLUDE= - fixed keyword value
- "EXCLUDE=NO" specifies that an "EXCLUDE" command for this job must not be accepted by the SPANEX Utility. This implies that this job must always be executed when this SPANEX Job Network is run, whatever the configuration of the network for any occasion. Note that EXCLUDE=NO will be ignored if the Job Process Options "RECOVERY" or "EXCLUDE" are specified (see below).

HOLD= - one or more numeric event identifiers
- specifies up to 8 external events that must have occurred before this job is eligible for scheduling by SPANEX. The exact nature of these events is arbitrarily decided by the user, and events are designated to SPANEX by single-digit numbers 1-8. Events are signalled complete by means of the SPANEX Utility POST command, which may be entered by the OS operator, a remote workstation operator, TSO user, from a batch job, etc, or by means of the #SPXRSTU macro in an application program. If more than one event is to specified for a single job, the event numbers should be enclosed in parentheses and separated by commas.

PROCESS= - fixed keyword value(s)
One or more of the following Process Options may be specified for any job in a SPANEX Job Network.
- "PROCESS=DELAY" specifies that a delay or break in the processing of this SPANEX Job Network is to occur before this job is run. This job must always be manually invoked by means of the "SCHEDULE" command to the SPANEX Utility.
"PROCESS=WFI" specifies that this job must wait for an external event (such as user input arrival) before it is run - SPANEX will issue WTOR messages to the console to determine if the job may run immediately it becomes eligible for scheduling, or whether it must wait for manual intervention. If the job must wait, it must be manually invoked by means of the "SCHEDULE" command to the SPANEX Utility.
"PROCESS=MULT" specifies that this job may need to be run multiple times before its post-requisites are run; as the job completes, SPANEX will issue WTOR messages to ask if this is the last run - if it is not, or if information concerning multiple runs is not available, scheduling of post-requisite jobs will be suspended. The WTOR messages may be bypassed by use of the #SPXRSTU macro in an application program to inform SPANEX of when the last run of this job occurs (see Section 6.5 of this manual).
"PROCESS=IGNERR" specifies that if this job fails in the last SPANEX step, scheduling of post-requisite jobs will continue normally. Failures in earlier steps of this job are not affected by this option.
"PROCESS=RECOVERY" specifies that this job is executed only in order to recover from the failure of some other job in this network. SPANEX never schedules this job automatically, and the job is treated as if it is always EXCLUDEd from the network at NETSTART time. When the job is

required to be run, it should be submitted by means of a SPANEX INCLUDE or SCHEDULE command. The job whose failure is recovered by the job defined with the "PROCESS=RECOVERY" option should specify the recovery job as a prerequisite (by means of the PREREQ parameter), so that when the recovery job is successfully completed the failing job will be automatically scheduled again by SPANEX. "PROCESS=EXCLUDE" specifies that this job is always automatically EXCLUDEd from the network at NETSTART time. This option is used for a job that is not normally needed to be run, or to effect a system of "including" jobs that are to be run from a selection of "excluded" jobs. If the job is required to be run, then it can be made eligible for scheduling by means of the "INCLUDE" command, or forced to execute immediately by means of the "SCHEDULE" command with the "FORCE" option. The job can also be submitted from within a user application program by use of the #SPXRSTU macro (see Section 6.5 of this manual).

RUNDAYS= - list of calendar day-type names
- specifies one or more calendar day-type names that define days on which this Job is run. These names refer to calendar tables that are defined either in the SPANEX System Calendar tables, or in a user calendar table whose identifier is specified by the USERCAL= parameter of the SPXRCM or QUICKNET macro. Each name specifies symbolically a day or a group of days when this job is to be run. For example, if the Job is run on Mondays and Fridays, the parameter might be specified as "RUNDAYS=(MONDAY,FRIDAY)". Note that if calendar specifications are also included on the SPXRCM or QUICKNET macro for this RCM, then this job will be run only on days for which the Job Network as a whole is defined to be run.

NONDAYS= - list of calendar day-type names
- specifies one or more calendar day-type names that define days on which this Job is not to be run. These names refer to calendar tables that are defined either in the SPANEX System Calendar tables, or in a user calendar table whose identifier is specified by the USERCAL= parameter of the SPXRCM or QUICKNET macro. Each name specifies symbolically a day or a group of days when this job is not to be run. These days may be a subset of the calendar days specified by the RUNDAYS= parameter. For example, if the Job is run on all Mondays except the first Monday in each month, the parameters might be specified as

		"RUNDAYS=MONDAY", and "NONDAYS=1STMON". Note that if calendar specifications are also included on the SPXRCM or QUICKNET macro for this RCM, then this job will be run only on days for which the Job Network as a whole is defined to be run.
MEMBER=	-	library member name, max=8 characters
	-	specifies the library member name of the member which contains the JCL for this job. This parameter is meaningful only for RCMs whose job submission technique involves searching for the JCL on a library. This parameter allows the library member name to differ from the job name, although the default, if this parameter is omitted, is for the member to be the same as the jobname. This feature is supported by SPANEX Quicknet, and by all the standard SPANEX Job Submit modules that use Partitioned Datasets, CA-PANVALET or CA-LIBRARIAN.

5.3 SPXSTEP Macro - Define a Jobstep to SPANEX RCM

The SPXSTEP macro should be used to define each restart-controlled step in the job defined by the previous SPXJOB or QUICKJOB macro in this RCM generation. Steps that are not to be restart-controlled (ie executed with SPANEX "OPT=M" specified) need not be defined by SPXSTEP macros. SPXSTEP macros must appear in the RCM generation input in the order in which the steps occur in the corresponding job JCL. Duplicate step names within a job are supported if there are differing procedure step names, which must be specified via the PROCSTP= parameter of the SPXSTEP macro. There may be no duplicate stepname/procstepname combinations within any one job. Note that for SPANEX Quicknet (a quicker and simplified method of generating SPANEX RCMs) the QUICKSTP macro may be used as an alternative to the SPXSTEP macro; all options of the SPXSTEP macro may also be specified on the QUICKSTP macro - see Section 10 in this manual.

format:

```
  stepname SPXSTEP CODE=(aaa,bbb) [,PROCSTP=procstepname]

                   [,STRTMSG=mmm]   [,ENDMSG=mmm]

                   [,FAILMSG=mmm]

                   [          [IGNERR]]   [        [nnnn ]]
                   [,PROCESS=[       ]]   [,ACCRC=[Unnnn]]
                   [          [NOGLOG]]   [        [Sxxx ]]

                   [,CHKEXIT=modname]
```

where:

stepname - JCL stepname, max=8 bytes, required
- specifies the stepname for the jobstep described by this SPXSTEP macro. This stepname must be the stepname that appears in the JCL "EXEC" statement for this step.

CODE= - two sub-parameters, numeric, max=4095
- specifies the Condition Codes to be returned from the SPANEX Restart Initialization processor when job failure occurred in a previous execution DURING this step (aaa) or AFTER this step and before the start of the following SPANEX restart-controlled step (bbb). If the CODE parameter is specified, both these sub-parameters must be included. The CODE parameter is always required, except for the first step in a job when the first step is to be executed with SPANEX OPT=I only (ie no user program execution in the first step).

PROCSTP= - JCL procedure stepname, max=8 bytes
- specifies the procedure stepname for this step. The specification of procedure stepname is optional, and is required only if the stepname for this step is a duplicate within the same job. This may occur, for example, if a JCL procedure is executed more than once within the job.

STRTMSG= - character text enclosed in single quotes
- specifies an optional message to be issued to the console operator when this jobstep begins execution. This message will also appear on the Global Log dataset if one is defined, and will appear there instead of the normal SPX902I message issued at the start of a step. If a quotation mark appears in the message it should be entered as two consecutive quotation marks; if an ampersand (&) appears in the message it should be entered as two consecutive ampersands. SPANEX will prefix the message with the identifier "SPX899SS" and will process the text of the message with the SPANEX message text editing facility to enable string substitution to be performed (see SPANEX General Usage Manual).

ENDMSG= - character text enclosed in single quotes
- specifies an optional message to be issued to the console operator when this jobstep finishes execution. This message will also appear on the Global Log dataset if one is defined, and will appear there instead of the normal SPX903I message issued for a successful step end. If a quotation mark appears in the message it should be entered as two consecutive quotation marks; if an ampersand (&) appears in the message it should be entered as two consecutive ampersands. SPANEX will prefix the message with the identifier "SPX899SE" and will process the text of the message with the SPANEX message text editing facility to enable string substitution to be performed (see SPANEX General Usage Manual).

FAILMSG= - character text enclosed in single quotes
- specifies an optional message to be issued to the console operator when this jobstep fails. This message will also appear on the Global Log dataset if one is defined, and will appear there in addition to any other SPANEX messages indicating that an error occurred in processing. This facility can be used to issue a message explaining recovery action that may need to be taken as a result of a failure in this step. If a quotation mark appears in the message it should be entered as two consecutive quotation marks; if an ampersand (&) appears in the message it should be entered as two consecutive ampersands.

SPANEX will prefix the message with the identifier "SPX899SF" and will process the text of the message with the SPANEX message text editing facility to enable string substitution to be performed (see SPANEX General Usage Manual).

PROCESS= - fixed keyword value(s)
- "PROCESS=IGNERR" specifies that if this jobstep fails for any reason, the failure is to be ignored for the purpose of determining whether or not a restart is to be performed (the step is set as normally-completed in the SPANEX catalog of job status).
"PROCESS=NOGLOG" specifies that routine Global Log messages (step-start and normal step-end) are not to be recorded for this jobstep.

ACCRC= - numeric condition code value, max=4095, or System or User abend code specification
- specifies an acceptable condition code or abend code value for the user program in this jobstep. An acceptable condition code value is the highest condition code from this program that does not constitute an error that should cause suspension of dependent jobs. An acceptable abend code is the single System or User abend code from this step that does not constitute an error that should cause suspension of dependent jobs; if an acceptable abend code is specified, and the jobstep does not abend, then all non-zero condition codes are recognized as errors. Note that the acceptable abend code feature is not supported for Retrospective Condition Code checking - if abends are to be accepted in this case, the only method is the use of the IGNERR option (see the description of the PROCESS parameter above). If the ACCRC parameter is omitted, any condition code from this jobstep that is greater than zero will be recognized as an error. If the ACCRC value is specified as 4095, the highest legal condition code from a user program, then all condition code values will be treated as a good end of the user program. Note that this value can also be specified in the SPANEX JCL parameter, and that any JCL specification overrides the SPXSTEP macro specification. Multiple conditions for acceptable errors can be implemented by means of a user exit (see the description of the CHKEXIT parameter).

CHKEXIT= - Load module name, max=8 bytes
- specifies the name of a user module that is to be called to check the execution of the program that runs in this jobstep. This "user check exit" facility is required if more than one acceptable abend code is required, or if the combination of an acceptable

abend code and acceptable non-zero condition codes is required. Note that this value can also be specified in the SPANEX JCL parameter, and that any JCL specification overrides the SPXSTEP macro specification. The definition of the interface to user check exit modules is contained in the SPANEX General Usage manual.

5.4 SPXRCM Macro - Generate a SPANEX RCM

The SPXRCM macro should be used to terminate the input (SPXJOB and SPXSTEP macros) to the SPANEX RCM generation process. It should be the last statement in the input to the Assembler. Note that for SPANEX Quicknet (a quicker and simplified method of generating SPANEX RCMs) the QUICKNET macro may be used as an alternative to the SPXRCM macro - see Section 10 in this manual.

format:

```
rcmname   SPXRCM   [CONFIRM=NO ]   [,OPTU=NO]
                   [        YES]

                   [,JOBNET=YES,NETRTN=rtnname]

                   [,USE=TEST]   [,EXIT=rtnname]

                   [,TSUPD=YES]   [,TSNET=YES]

                   [,ERASEON=YES]   [,EONRTN=rtnname]

                   [,TITLE=title]   [,NETPASS=password]

                   [,STREXIT=rtnname]

                   [,GLOGDD=ddname]   [,GLOGDSN=dsname]

                   [,GLOGOPT=SHR]

                   [,RUNDAYS=(dddd,dddd, ...)]

                   [,NONDAYS=(dddd,dddd, ...)]

                   [,NEWDAY=nnnn]   [,USERCAL=xxx]

                   [          [RUNDAYS]          ]
                   [,NSLIST=([        ] [, ...])]
                   [          [NONDAYS]          ]

                   [,ASMLIST=YES]
```

where:

rcmname - RCM load module name, max=8 bytes, required
- specifies the load module name to be used for this RCM. This name is included in the generated RCM for checking and debugging purposes.

CONFIRM= - fixed keyword value
- specifies the "CONFIRM" option to be used for restart of jobs defined by this RCM. This can be overridden by the "CONFIRM" option if specified on the SPXJOB or QUICKJOB macro, for that job only. If "CONFIRM=YES" is specified, operator authority will be requested for a restart after a failure of a job. If "CONFIRM=NO" is specified, SPANEX will restart the failed job as defined in the RCM without allowing the operator to override the restart point or to cancel the restart. If "CONFIRM" is not specified on either the SPXRCM or the SPXJOB/QUICKJOB macro, the default is "CONFIRM=NO".

OPTU= - fixed keyword value
- "OPTU=NO" specifies that jobs defined by this RCM are not to have their Restart Status updated by means of the SPANEX Utility "UPDATE" command.

JOBNET= - fixed keyword value
- "JOBNET=YES" specifies that the jobs defined in this RCM constitute a SPANEX Job Network, and automatic scheduling of jobs is to be performed.

NETRTN= - load module name, max=8 bytes
- specifies the name of a routine to be invoked by SPANEX to process the scheduling of a job. Automatic scheduling, or scheduling invoked by means of any of the job networking commands of the SPANEX Utility, will be performed by means of a call to this routine. See Section 6.2 on User Submit modules in this manual.

USE= - fixed keyword value
- "USE=TEST" specifies that this RCM is to be used for testing the SPANEX Restart and/or Networking facilities, or for the testing of the job or suite of jobs that is defined by this RCM. If the "ACK" SPANEX EXEC statement parameter option was made an installation default when SPANEX was installed this will be negated for jobs run under a "USE=TEST" RCM. If the ACK parameter option is required, it should be specified in the JCL EXEC PARM field.

EXIT= - load module name, max=8 bytes
- specifies the name of a user exit routine load module that is to be fetched and executed by SPANEX each time each job defined in this RCM begins to execute. This routine has the ability to request or override a restart of a job. The name of the restart exit routine can be overridden for individual jobs by specifying the EXIT= parameter on the SPXJOB or QUICKJOB macro. See

Section 6.1 on User Restart Exit routines in this manual.

TSUPD= - fixed keyword value
- "TSUPD=YES" specifies that jobs defined by this RCM are able to have their restart status updated or deleted by the SPANEX Utility "UPDATE" and "DELETE" commands executed from a time-sharing terminal. The default is "TSUPD=NO" and RCMs that do not specify "TSUPD=YES" will not have update or delete operations performed upon them in foreground. Dedicated SPANEX terminals are not affected by his option, and are not restricted in the commands they can issue except by the use of the SPANEX network access control exit routine.

TSNET= - fixed keyword value
- "TSNET=YES" specifies that the job network defined by this RCM may be controlled by the SPANEX Utility commands executed from a time-sharing terminal. The default is "TSNET=NO" and RCMs that do not specify "TSNET=YES" will not have network control commands ("NETSTART", "EXCLUDE", "SCHEDULE", "INCLUDE", "CSHEET", "HALT", "PROCEED", "HOLD", "POST", "STATUS" (some options)) accepted in foreground. Dedicated SPANEX terminals are not affected by his option, and are not restricted in the commands they can issue except by the use of the SPANEX network access control routine.

ERASEON= - fixed keyword value
- "ERASEON=YES" specifies that SPANEX is required to detect when all processing for the job network defined by this RCM has successfully completed execution. When end-of-network is detected, SPANEX will notify the operator and will delete the recorded status of all jobs in the network. This will have the effect of saving space in the SPANEX Catalog but will prevent the normal re-execution of a single job of the network after end-of-network has been detected.

EONRTN= - load module name, max=8 bytes
- specifies the name of a user load module to be given control when execution of the whole of this job network completes successfully. This routine will be invoked by the last defined step of the last job of the network to complete successfully. See Section 6.3 on page 79 of this manual.

TITLE= - character text enclosed in single quotes
- specifies an optional title for the job network described by this RCM. This title will appear

after a SPX887I message issued to the operator when the network begins or ends execution, and is also printed on SPANEX Job Check Sheets. If a quotation mark appears in the message it should be entered as two consecutive quotation marks; if an ampersand (&) appears in the message it should be entered as two consecutive ampersands.

NETPASS= - network password, max=8 bytes
- specifies a password to protect this job network against unintentional or illegal use of the "CSHEET", "EXCLUDE", "HALT", "HOLD", "INCLUDE", "NETSTART", "POST", "PROCEED", "SCHEDULE" or "STATUS" commands. For each of these commands this password must be specified via the "PASSWORD=" parameter in order that the command will be accepted by the SPANEX Utility. If the "NETPASS=" parameter of the SPXRCM macro is not specified, then there is no password protection for this RCM.

STREXIT= - load module name, max=8 bytes
- specifies the name of a user load module to be given control when a SPANEX Utility "NETSTART" command is about to be processed. The routine has the ability to permit or disallow a Netstart of this network, and could be used, for example, to ensure that the Global Log for the previous network execution has been printed or saved.

GLOGDD= - DDNAME, max=8 bytes
- specifies the DDNAME to be used by the SPANEX Utility command processors to access the optional Global Log dataset for this RCM. This DDNAME must be unique for Global Log datasets within an installation but may be used by multiple networks if they are to share a Global Log dataset (see the "GLOGOPT=" parameter below). This DDNAME must be provided in the Job Control Language for the SPANEX Utility and must refer to the correct Global Log dataset. Note that this parameter must be specified for MVS, even if Dynamic Allocation is to be used (see the "GLOGDSN=" parameter below and Section 4.6.6 on page 31 for a description of the Global Log facility), both for compatibility with other operating systems and because it is the GLOGDD parameter which triggers the inclusion of the Global Log facility in a job network and is used for serialization of updates to the Global Log dataset. For all executions of SPANEX apart from the SPANEX Utility, the fixed DDNAME "SPXGLOG" is used to access the Global Log dataset. Note that although the Global Log, and thus the "GLOGDD="

parameter, are normally optional, there is a SPANEX generation option that may make a Global Log compulsory for all use in an installation. In this case, RCM generation will fail if this parameter is omitted.

GLOGDSN= - Global Log Dataset name, max=44 bytes
- specifies the fully-qualified dataset name of the optional Global Log dataset for this RCM. This parameter is meaningful only for MVS when Dynamic Allocation is to be used for the Global Log, and is recognized only if the GLOGDD parameter is also specified. The Global Log dataset must be catalogued, and there must be a separate Global Log dataset for each RCM that is in use, except when the shared Global Log option is used (see the "GLOGOPT=" parameter below). See Section 4.6.6 on page 31 of this manual for a full description of the Global Log facility.

GLOGOPT= - fixed keyword value
- "GLOGOPT=SHR" specifies that the Global Log to be used for this Network, and specified by the "GLOGDD=" and "GLOGDSN=" parameters, is shared between this network and one or more other networks. If this option is specified, it is the user's responsibility to ensure that the correct values are specified for the "GLOGDD=" and "GLOGDSN=" parameters in all the RCMs that share this Global Log. Note that the value specified for the "GLOGDD=" parameter is the mechanism used by SPANEX for serializing updates to the Global Log, and so this value is critical even if MVS Dynamic Allocation is to be used. SPANEX will never clear a Global Log dataset that has the "GLOGOPT=SHR" option. A possible solution to this is to specify this option in all RCMs that share this Global Log dataset, except the RCM for the Network that is run first. The NETSTART for this first network will then clear the dataset as with non-shared Global Logs. Otherwise it is the user's responsibility periodically to clear the dataset.

RUNDAYS= - list of calendar day-type names
- specifies one or more calendar day-type names that define days on which this Job Network is run. These names refer to calendar tables that are defined either in the SPANEX System Calendar tables, or in a user calendar table whose identifier is specified by the USERCAL= parameter (see below). Each name specifies symbolically a day or a group of days when the application defined by this RCM is to be run. For example, if the Network is run on Mondays and Fridays, the

parameter might be specified as "RUNDAYS=(MONDAY,FRIDAY)".

NONDAYS= - list of calendar day-type names
- specifies one or more calendar day-type names that define days on which this Job Network is not run. These names refer to calendar tables that are defined either in the SPANEX System Calendar tables, or in a user calendar table whose identifier is specified by the USERCAL= parameter (see below). Each name specifies symbolically a day or a group of days when the application defined by this RCM is not to be run. These days may be a subset of the calendar days specified by the RUNDAYS= parameter. For example, if the Network is run on all Mondays except the first Monday in each month, the parameters might be specified as "RUNDAYS=MONDAY", and "NONDAYS=1STMON".

NEWDAY= - time-of-day in hhmm format
- specifies a notional start-of-day time for this Job Network. This value overrides any NEWDAY value specified in the SPANEX System Calendar Tables, or in the user Calendar identified by the USERCAL= parameter below, for this Network only. The notional start-of-day value is the time of day at which the calendar is deemed to change from one date to the next. This is designed to handle the situation where, for example, batch processing for Monday is performed between 6pm on Monday evening and 5am on Tuesday morning; in this case, the NEWDAY value might be specified as "0600" so that a NETSTART command for this Job Network issued before 6am is treated as scheduling the Network for Monday's processing. The NEWDAY= parameter must be four decimal digits representing a time in the format "hhmm" using the 24-hour clock.

USERCAL= - SPANEX user calendar identifier
- specifies the 3-character identifier of a SPANEX User Calendar table. The User Calendar table should be defined using the SPANEX Calendar Definition process. Processing of this Job Network will search the User Calendar table for any calendar names specified in the RUNDAYS= or NONDAYS= parameters before searching the SPANEX System Calendar table. This allows individual calendars to be defined or overridden for specific Job Networks or for specific user groups. Any specification of the NEWDAY value in the User Calendar will override the NEWDAY value in the SPANEX System Calendar; both of these are overridden by the NEWDAY=

parameter of the SPXRCM macro described above. The USERCAL= value may be any unique 3-character string, excluding some reserved SPANEX strings (such as “CAL”, “M0x” and “NJS”). An error message will be issued if an attempt is made to use one of the reserved strings.

NSLIST= - specific keyword value(s)
- specifies that jobname lists are to be produced by the NETSTART command to document the jobs that are included and/or excluded from the Job Network as a result of SPANEX Calendar processing. This list is output both to the SPANEX Message LOG (SPXPRINT DD statement) and to the Global Log dataset for the Job Network. NSLIST=RUNJOBS specifies that a list should be produced of jobs that are to be run; NSLIST=NONJOBS specifies that a list should be produced of jobs that are not to be run. If both options are required, they should be placed in parentheses and separated by a comma; the NETSTART command will produce the list of excluded jobs first, followed by the list of included jobs.

ASMLIST= - fixed keyword value
- “ASMLIST=YES” specifies that a complete assembly list of the RCM generation output should be produced. This may be useful for debugging purposes. If this parameter is not specified, only the SPANEX run documentation will be printed from the RCM generation job.

5.5 #SPXRDEF Macro - Generate SPANEX Restart/Networking DSECTS

The #SPXRDEF macro should be used in modules (eg SPANEX RCM User Exit Routines, end-of-network exit routines, NETSTART exit routines) that require access to the internal control blocks associated with the SPANEX Restart and Networking facilities. Control blocks mapped by the #SPXRDEF macro are: RCM (RCM header section, DSECT name SPXRCM), JRCB (Job Restart Control Block, DSECT name SPXJRCB), SRCB (Step Restart Control Block, DSECT name SPXSRCB), RSB (Restart Status Block, DSECT name SPXRSB), RUP (Restart User Exit Parameter, DSECT name SPXRUP).

format:

```
#SPXRDEF  DSECT=YES
```

where:

DSECT= - fixed keyword value
- "DSECT=YES" specifies that this is a request to generate the DSECTs for the SPANEX Restart and Networking facilities. The "DSECT=YES" operand is required.

This page intentionally left blank.

6 Installation-written Routines for SPANEX

6.1 Coding a Restart User Exit Routine

A SPANEX Restart Facility User Exit Routine is a user-supplied load module that may be invoked by SPANEX before beginning either a clean start or a restart of a SPANEX restart-controlled job. The name of the load module is specified on the SPXRCM or QUICKNET macro ("EXIT=" parameter) for all jobs in the suite or network, or on the SPXJOB or QUICKJOB macro ("EXIT=" parameter) for individual jobs; an SPXJOB/QUICKJOB specification overrides any SPXRCM/QUICKNET specification.

The "TASKLIB" SPANEX DD statement library or concatenation is searched first for the user exit routine load module; if it is found there it is used, otherwise the JOBLIB/STEPLIB or Linklist is searched. If the user exit routine cannot be found at all, then the user exit function is bypassed and processing continues as if the user exit option was not specified in the RCM. Appropriate messages appear on the SPANEX Message Log (SPXPRINT DD statement).

The user exit routine is given control after SPANEX has decided on the restart action to be taken (if any) and before the operator is asked to confirm any restart action (if "CONFIRM=YES" is in operation as a result of the SPXRCM/QUICKNET or SPXJOB/QUICKJOB macros). The exit routine may alter or override any of the actions that SPANEX may recommend, and may force or disable operator communication before the action is taken.

On entry to the user exit routine, Register 1 points to the following parameter list:

```
A(SPANEX ICB)
A(RCM)
A(JRCB for this Job)
A(SRCB for failed step if restart, otherwise for this
step)
A(Restart Status Block)
```

This parameter list is known as the RUP (Restart User exit Parameter) and is mapped by the #SPXRDEF macro. For an explanation of the SPANEX control blocks see the SPANEX Automated Data Areas manual.

Format of RSB (Restart Status Block) (RSB is mapped by #SPXRDEF macro):

+0	RSBSTAT:	(Last known job status)
	'INIT'	Job failed during Restart Init
	'STST'	Start of Step
	'ABNU'	User Abend (code in RSBCODE)
	'ABNS'	System Abend (code in RSBCODE)
	'ABNX'	Unknown Abend
	'ABPF'	Operator stopped job
	'ABRC'	Abnormal RC (code in RSBCODE)
	'RSTR'	'RSTR' Restart attempted (code in RSBCODE)
	'UCHK'	User Check Exit detected error
	'URST'	User Exit restart attempted (code in RSBCODE)
	'USRR'	User Program called internal utility interface
	'STEN'	Step Ended normally (code in RSBCODE)
	'PREX'	Clean Start for network job
	'SUCC'	Job has already completed
	' ' (blank)	Clean Start
+4	RSBSTEP	Name of Step described by RSBSTAT field
+12	RSBDATE	Date of last status, or blank format: X'00YYDDDF'
+16	RSBTIME	Time of last status, or blank format: X'HHMMSS0F'
+20	RSBCODE	Code according to RSBSTAT See explanation below
+24	RSBSCODE	Code to be used by Restart Initialization step as Return Code to effect this restart, or zero)
+28	RSBUCODE	Field for code to be returned by user exit

Operating upon this information passed from SPANEX, the user exit routine may perform any function, and should terminate by returning to SPANEX with a return code (in Register 15) specifying the action that is to be taken by SPANEX. Any SPANEX or system macro, including

ABEND, may be issued by the user exit routine. If ABEND is issued during user exit processing, or if an invalid return code is passed to SPANEX from the user exit, the job will be abnormally terminated with no alteration having been made to the restart status as it was on initial entry to the user exit routine.

Valid return codes from User Restart Exit routines:

0	Continue with processing, actioning current "CONFIRM" option
4	Continue with processing, forcing "CONFIRM=YES"
8	Continue with processing, forcing "CONFIRM=NO"
12	Cancel this job (no change to restart status)
16	Issue restart condition code passed in RSBUCODE field
20	Issue restart condition code passed in RSBUCODE field but confirm with the operator first: if a user restart exit routine returns a code of 20 to SPANEX, SPANEX will verify the value that is passed in the RSBUCODE field to ensure that it appears in the RCM - if it does not, Abend U0048 will be issued; if it does, the stepname for which the code is first defined will be used in the SPX816I message issued to the operator for his confirmation of the restart point.

Explanation of RSBCODE field and relation to other fields

If RSBSTAT = "INIT", "RSTR" or "URST", and RSBCODE byte 1 = "?", then RSBCODE low-order 3 bytes contain the Restart Init condition code chosen by SPANEX for the previous restart attempt, in binary.

If RSBSTAT = "STST" and RSBCODE byte 1 = "?", then RSBCODE low-order 3 bytes are not valid.

If RSBSTAT = "STEN" and RSBCODE byte 1 = "?", then RSBCODE low-order 3 bytes contain the return code from the step that ended, in binary.

If RSBSTAT = "ABRC", then RSBCODE byte 1 = X'FF' and RSBCODE low-order 3 bytes contain the abnormal return code issued by the user program, in binary.

If RSBSTAT = "ABNS", then RSBCODE byte 1 = C'S' and RSBCODE low-order 3 bytes contain the System Abend code issued by the user program, in character format.

If RSBSTAT = "ABNU", then RSBCODE bytes 1-4 contain the User Abend code issued by the user program, in character format.

If RSBSTAT = "ABPF", then RSBCODE byte 1 = C'F' and RSBCODE low-order 3 bytes are not valid.

If RSBSTAT = "USRR", then RSBCODE byte 1 = C'U' and RSBCODE low-order 3 bytes are not valid.

Other combinations of these field values are not currently supported by SPANEX and may be ignored.

6.2 SPANEX Job Networking User Submit Routines

A SPANEX Job Networking User Submit Routine is a user-supplied load module that is invoked by SPANEX in order to schedule a job for execution to the Operating System. The name of the user submit routine load module is specified on the SPXRCM macro ("NETRTN=" parameter) for all jobs in the network.

In order to simplify the implementation of SPANEX Job Networking for the user installation, many different sample user submit routines are supplied with the SPANEX release. The supplied routines cover virtually all requirements and system configurations. Routines are provided using a number of different techniques for the submission of jobs. All Operating Systems supported by SPANEX are catered for by one or more of these routines. These routines may be used as supplied or may be modified in any way. Any other technique preferred by the user installation may be employed, and different submit routines may be used for different networks of jobs. The supplied sample submit routines are documented below. See also the description of the SPANEX Quicknet feature in Section 10 of this manual for the additional submit routines supplied for the simplest possible implementation of job networking.

The "TASKLIB" SPANEX DD statement library or concatenation is searched first for the user submit routine load module; if it is found there it is used, otherwise the JOBLIB/STEPLIB or Linklist is searched. If the user submit module cannot be found at all, then job scheduling will not be performed and SPANEX will terminate abnormally after issuing messages documenting the name(s) of the job(s) that were to be scheduled at the time of the failure to locate the routine. Note that SPANEX user submit routine load modules should be installed on an APF authorized library, or an Abend may result when SPANEX attempts to fetch the module into main storage.

The user submit routine is given control by SPANEX Job Networking when SPANEX has decided to submit a job for processing. It is the responsibility of the routine to perform any and all actions necessary to cause the invocation of the subject job, and to notify SPANEX of whether or not the submission of the job was successful.

On entry to the user submit routine, the following parameter registers are supplied by SPANEX:

Register 0	➜	JRCB for the job to be submitted (the JRCB is mapped by the #SPXRDEF macro)
Register 1	➜	SPANEX ICB, which contains or points to all information used by SPANEX (the SPANEX ICB is mapped by the #SPXICB macro)

The user submit routine must return to SPANEX with one of the following codes in Register 15:

0	Job submitted successfully, normal SPANEX messages are to be issued to the operator concerning the submission of this job
4	Job submitted successfully, no SPANEX messages are to be issued to the operator concerning the submission of this job (submit routine has already issued messages)
8	Job not submitted, normal SPANEX messages are to be issued to the operator concerning the submission of this job
12	Job not submitted, no SPANEX messages are to be issued to the operator concerning the submission of this job (submit routine has already issued messages)

6.2.1 Sample Submit Routine 1 -- SPXNJS01

Operating System: MVS only, with JES2 or JES3.

Submit Technique: JCL for submitted jobs must be held on a partitioned dataset, with fixed or fixed-blocked 80-byte records, with one PDS member per job, where the member name is the job name or the name specified by the MEMBER= parameter of the SPXJOB macro statement.

Module usage: A JOBPDS DD statement must be included in the JCL for the use of this routine, specifying the datasetname of the PDS containing the JCL for all jobs in the network. A TEMPPDS DD statement may optionally also be included - this will be searched first for each job's JCL, allowing temporary JCL overrides without modifying the production JCL allocated to the JOBPDS DD statement. The routine will be called by the last defined processing jobstep of each job in the network that has post-requisite job(s), and by the SPANEX Utility for “INCLUDE”, “NETSTART”, “SCHEDULE”, “POST” or “PROCEED” command processing.

Processing logic: An MVS Internal Reader is dynamically allocated and dynamic VSAM control blocks are used for writing out the JCL. The Span Software “QPAM” queued PDS access method (SPZQPAM routine) is used to read the JCL. A maximum of ten attempts, ten seconds apart, will be made to allocate an Internal Reader.

Return Codes: Return codes 4 and 12 are issued by this routine as all necessary messages are issued from within the routine. Abend U0999 is issued if this routine is run under any Operating System other than MVS.

6.2.2 Sample Submit Routine 2 -- SPXNJS02

Operating System: MVS with JES2 or JES3.

Submit Technique: JCL for submitted jobs must be held on a partitioned dataset, with fixed or fixed-blocked 80-byte records, with one PDS member per job, where the member name is the job name or the name specified by the MEMBER= parameter of the SPXJOB macro statement.

Module usage: A JOBPDS DD statement must be included in the JCL for the use of this routine, specifying the datasetname of the PDS containing the JCL for all jobs in the network, and must specify DISP=SHR. A TEMPPDS DD statement may optionally also be included - this will be searched first for each job's JCL, allowing temporary JCL overrides without modifying the production JCL allocated to the JOBPDS DD statement. The routine will be called by the last defined processing jobstep of each job in the network that has post-requisite job(s), and by the SPANEX Utility for “INCLUDE”, “NETSTART”, “SCHEDULE”, “POST” or “PROCEED” command processing.

Processing logic: An Operating System “START RDR” command is issued, using the job PDS member as input to the Reader. A check is made that the member exists before issuing the command.

Return Codes: Return codes 0 and 8 are issued by this routine and normal SPANEX messages will inform the operator whether or not the job has been submitted. Abend U0999 is issued if this routine is run under any Operating System other than MVS.

6.2.3 Sample Submit Routine 3 -- SPXNJS03

Operating System: VS/1 only.

Submit Technique: JCL for all possible submitted jobs must be placed on the Operating System Job Queue, with each job specifying the "TYPRUN=HOLD" parameter on the "JOB" statement, by the operator, before the NETSTART command is issued for this network.

Module usage: The routine will be called by the last defined processing jobstep of each job in the network that has post-requisite job(s), and by the SPANEX Utility for "INCLUDE", "NETSTART", "SCHEDULE", "POST" or "PROCEED" command processing.

Processing logic: An Operating System "RELEASE jobname" command is issued for the job. For VS/1 the Job Queue is first scanned to ensure that the job exists and is in HOLD status.

Return Codes: Return codes 0 and 12 are issued: standard SPANEX messages appear for a successful job submission; all submit error messages are issued from within this module. Abend U0999 is issued if this routine is run under any Operating System other than VS/1.

6.2.4 Sample Submit Routine 4 -- SPXNJS04

Operating System:	MVS only, with JES2 or JES3.
Submit Technique:	JCL for all possible submitted jobs must be placed on the JES Job Queue, with each job specifying the "TYPRUN=HOLD" parameter on the "JOB" statement, by the operator, before the NETSTART command is issued for this network.
Module usage:	The routine will be called by the last defined processing jobstep of each job in the network that has post-requisite job(s), and by the SPANEX Utility for "INCLUDE", "NETSTART", "SCHEDULE", "POST" or "PROCEED" command processing.
Processing logic:	The MVS Subsystem Interface is used to communicate with the active subsystem (JES2 or JES3) firstly to ensure that the job is found, is in HOLD status, and to obtain its JES JOBID, and then to issue the appropriate subsystem command to release the job for execution. The #SPXSVC macro "TYPE=OWNCODE" option is used to gain authority to use the Subsystem Interface. Explanatory error messages are issued if any difficulty is encountered with communication with the Job Entry Sub-system.
Return Codes:	Return codes 4 and 12 are issued by this routine as all necessary messages are issued from within the routine. Abend U0999 is issued if this routine is run under any Operating System other than MVS.

6.2.5 Sample Submit Routine 5 -- SPXNJS05

Operating System: MVS only, with JES2 or JES3.

Submit Technique: JCL for submitted jobs must be held on a partitioned dataset, with fixed or fixed-blocked 80-byte records, with one PDS member per job, where the member name is the job name or the name specified by the MEMBER= parameter of the SPXJOB macro statement.

Module usage: A JOBPDS DD statement must be included in the JCL for the use of this routine, specifying the datasetname of the PDS containing the JCL for all jobs in the network. A TEMPPDS DD statement may optionally also be included - this will be searched first for each job's JCL, allowing temporary JCL overrides without modifying the production JCL allocated to the JOBPDS DD statement. The routine will be called by the last defined processing jobstep of each job in the network that has post-requisite job(s), and by the SPANEX Utility for “INCLUDE”, “NETSTART”, “SCHEDULE”, “POST” or “PROCEED” command processing. An Internal Reader must also be allocated to the jobstep with a DDNAME of “INTRDR”.

Processing logic: The JCL is copied from the JOBPDS (or TEMPPDS) to the Internal Reader. The Span Software “QPAM” queued PDS access method (SPZQPAM routine) is used to read the JCL.

Return Codes: Return codes 0 and 12 are issued by this routine - all necessary messages are issued from within the routine for errors, standard SPANEX messages are issued if the job is successfully submitted. Abend U0999 is issued if this routine is run under any Operating System other than MVS.

6.2.6 Sample Submit Routine 6 -- SPXNJS06

Operating System: MVS only, with JES2 or JES3.

Submit Technique: JCL for submitted jobs must be held on a partitioned dataset, with fixed or fixed-blocked 80-byte records, with one PDS member per job, where the member name is the job name or the name specified by the MEMBER= parameter of the SPXJOB macro statement. JCL may also be held on the JES Job Queue with the "TYPRUN=HOLD" parameter specified on the JOB statement.

Module usage: A JOBPDS DD statement must be included in the JCL for the use of this routine, specifying the datasetname of the PDS containing the JCL for all jobs in the network. A TEMPPDS DD statement may optionally also be included - this will be searched first for each job's JCL, allowing temporary JCL overrides without modifying the production JCL allocated to the JOBPDS DD statement. The routine will be called by the last defined processing jobstep of each job in the network that has post-requisite job(s), and by the SPANEX Utility for "INCLUDE", "NETSTART", "SCHEDULE", "POST" or "PROCEED" command processing.

Processing logic: This routine is logically a combination of routines SPXNJS04 and SPXNJS01: the JES Job Queue is scanned, and the job is released if it is found on the queue in HOLD status. If not, an MVS Internal Reader is dynamically allocated and dynamic VSAM control blocks are used for writing out the JCL. The Span Software "QPAM" queued PDS access method (SPZQPAM routine) is used to read the JCL. The Job Queue is then re-checked and the job is released if it is found to be in HOLD status when it appears on the queue (up to 25 seconds is allowed for the job to arrive on the Job Queue). This routine permits multiple runs of a job to be performed without manual intervention to place the job on the JES queue. A maximum of ten attempts, ten seconds apart, will be made to allocate an Internal Reader.

Return Codes: Return codes 4 and 12 are issued by this routine as all necessary messages are issued from within the routine. Abend U0999 is issued if this routine is run under any Operating System other than MVS.

6.2.7 Sample Submit Routine 7 -- SPXNJS07

Operating System: VS/1 only.

Submit Technique: JCL for all possible submitted jobs may be placed on the Operating System JES Job Queue, with each job specifying the "TYPRUN=HOLD" parameter on the JOB statement, by the operator, before the NETSTART command is issued for the network; and a JOBPDS DD statement must be supplied for a dataset containing each job's JCL as a PDS member. A TEMPPDS DD statement may optionally also be included - this will be searched first for each job's JCL, allowing temporary JCL overrides without modifying the production JCL allocated to the JOBPDS DD statement. The PDS member must be the same as the jobname or the name specified by the MEMBER= parameter of the SPXJOB macro statement.

Module usage: The routine will be called by the last defined processing jobstep of each job in the network that has post-requisite job(s), and by the SPANEX Utility for "INCLUDE", "NETSTART", "SCHEDULE", "POST" or "PROCEED" command processing.

Processing logic: The JES Job Queue is scanned to check that the job exists and is in HOLD status - if found it is released. If not found, a START RDR command is issued against the appropriate member of the job PDS to place the JCL on the Job Queue. (The VS/1 system should be generated with enough Readers specified to ensure that the START command issued will be successful, as this routine waits for notification from the Reader that it has completed.) The queue is then checked to ensure that the job has arrived, and the job is released if it appears in HOLD status. Up to 25 seconds is allowed for the job to reach the queue.

Return Codes: Return codes 4 and 12 are issued by this routine as all necessary messages are issued from within the routine. Abend U0999 is issued if this routine is run under any Operating System other than VS/1.

6.2.8 Sample Submit Routine 8 -- SPXNJS08

Operating System: MVS only, with JES2 or JES3.

Submit Technique: JCL for submitted jobs must be held on a CA-PANVALET library, with fixed 80-byte records, with one CA-PANVALET member per job, where the member name is the job name or the name specified by the MEMBER= parameter of the SPXJOB macro statement.

Module usage: A PANVALET DD statement must be included in the JCL for the use of this routine, allocated to the CA-PANVALET JCL library. A TEMPPANV DD statement may optionally also be included - this will be searched first for each job's JCL, allowing temporary JCL overrides without modifying the production JCL allocated to the PANVALET DD statement. The routine will be called by the last defined processing jobstep of each job in the network that has post-requisite job(s), and by the SPANEX Utility for “INCLUDE”, “NETSTART”, “SCHEDULE”, “POST” or “PROCEED” command processing.

Processing logic: An MVS Internal Reader is dynamically allocated and dynamic VSAM control blocks are used for writing out the JCL. The CA-PANVALET Access Method (PAM) is used to read the job JCL. A maximum of ten attempts, ten seconds apart, will be made to allocate an Internal Reader.

Return Codes: Return codes 4 and 12 are issued by this routine as all necessary messages are issued from within the routine. Abend U0999 is issued if this routine is run under any Operating System other than MVS.

6.2.9 Sample Submit Routine 9 -- SPXNJS09

Operating System: MVS only, with JES2 or JES3.

Submit Technique: JCL for submitted jobs must be held on a CA-PANVALET library, with fixed 80-byte records, with one CA-PANVALET member per job, where the member name is the job name or the name specified by the MEMBER= parameter of the SPXJOB macro statement. JCL may also be held on the JES Job Queue with the "TYPRUN=HOLD" parameter specified on the JOB statement.

Module usage: A PANVALET DD statement must be included in the JCL for the use of this routine, allocated to the CA-PANVALET JCL library. A TEMPPANV DD statement may optionally also be included - this will be searched first for each job's JCL, allowing temporary JCL overrides without modifying the production JCL allocated to the PANVALET DD statement. The routine will be called by the last defined processing jobstep of each job in the network that has post-requisite job(s), and by the SPANEX Utility for "INCLUDE", "NETSTART", "SCHEDULE", "POST" or "PROCEED" command processing.

Processing logic: This routine is logically a combination of routines SPXNJS04 and SPXNJS08: the JES Job Queue is scanned, and the job is released if it is found on the queue in HOLD status. If not, an MVS Internal Reader is dynamically allocated and dynamic VSAM control blocks are used for writing out the JCL. The CA-PANVALET Access Method (PAM) is used to read the job JCL. The Job Queue is then re-checked and the job is released if it is found to be in HOLD status when it appears on the queue (up to 25 seconds is allowed for the job to arrive on the Job Queue). This routine permits multiple runs of a job to be performed without manual intervention to place the job on the JES queue. A maximum of ten attempts, ten seconds apart, will be made to allocate an Internal Reader.

Return Codes: Return codes 4 and 12 are issued by this routine as all necessary messages are issued from within the routine. Abend U0999 is issued if this routine is run under any Operating System other than MVS.

6.2.10 Sample Submit Routine 10 -- SPXNJS10

Operating System: MVS with JES2 or JES3.

Submit Technique: JCL for submitted jobs must be held on a CA-PANVALET library, with fixed 80-byte records, with one CA-PANVALET member per job, where the member name is the job name or the name specified by the MEMBER= parameter of the SPXJOB macro statement.

Module usage: A PANVALET DD statement must be included in the JCL for the use of this routine, allocated to the CA-PANVALET JCL library. A TEMPPANV DD statement may optionally also be included - this will be searched first for each job's JCL, allowing temporary JCL overrides without modifying the production JCL allocated to the PANVALET DD statement. A “SYSUT1” DD statement must also be supplied, referencing a sequential DASD work dataset sufficiently large to contain the JCL for any one job in the network; the DD statement must specify “DISP=SHR”. This routine will be called by the last defined processing jobstep of each job in the network that has post-requisite job(s), and by the SPANEX Utility for “INCLUDE”, “NETSTART”, “SCHEDULE”, “POST” or “PROCEED” command processing.

Processing logic: The CA-PANVALET Access Method (PAM) is used to read the job JCL from the CA-PANVALET library. The JCL is then copied to the dataset referenced by the SYSUT1 DD statement, and an Operating System “START RDR” command is issued, using the SYSUT1 work dataset as input to the Reader. The VS/1 system should be generated with enough Readers specified to ensure that the START command issued will be successful, as this routine waits for notification from the Reader that it has completed.

Return Codes: Return codes 4 and 12 are issued by this routine as all necessary messages are issued from within the routine. Abend U0999 is issued if this routine is run under any Operating System other than MVS.

6.2.11 Sample Submit Routine 11 -- SPXNJS11

Operating System: VS/1 only.

Submit Technique: JCL for all possible submitted jobs may be placed on the Operating System JES Job Queue, with each job specifying the "TYPRUN=HOLD" parameter on the JOB statement, by the operator, before the NETSTART command is issued for the network. A PANVALET DD statement must be supplied for a CA-PANVALET library containing each job's JCL as a CA-PANVALET member. A TEMPPANV DD statement may optionally also be included - this will be searched first for each job's JCL, allowing temporary JCL overrides without modifying the production JCL allocated to the PANVALET DD statement. The CA-PANVALET member name must be the same as the jobname or the name specified by the MEMBER= parameter of the SPXJOB macro statement. A "SYSUT1" DD statement must also be supplied, referencing a sequential DASD work dataset sufficiently large to contain the JCL for any one job in the network; the DD statement must specify "DISP=SHR".

Module usage: The routine will be called by the last defined processing jobstep of each job in the network that has post-requisite job(s), and by the SPANEX Utility for "INCLUDE", "NETSTART", "SCHEDULE", "POST" or "PROCEED" command processing.

Processing logic: The JES Job Queue is scanned to check that the job exists and is in HOLD status - if found it is released. If not found, the JCL is obtained from CA-PANVALET and is copied to the work dataset referenced by the SYSUT1 DD statement. A START RDR command is issued against the SYSUT1 dataset the JCL on the Job Queue. (The VS/1 system should be generated with enough Readers specified to ensure that the START command issued will be successful, as this routine waits for notification from the Reader that it has completed.) The queue is then checked to ensure that the job has arrived, and the job is released if it appears in HOLD status. Up to 25 seconds is allowed for the job to reach the queue.

Return Codes: Return codes 4 and 12 are issued by this routine as all necessary messages are issued from within the routine. Abend U0999 is issued if this routine is run under any Operating System other than VS/1.

6.2.12 Sample Submit Routine 12 -- SPXNJS12

Operating System: MVS only, with JES2 or JES3.

Submit Technique: JCL for submitted jobs must be held on a CA-LIBRARIAN Master, with fixed 80-byte records, with one CA-LIBRARIAN member per job, where the member name is the job name or the name specified by the MEMBER= parameter of the SPXJOB macro statement.

Module usage: A LIBRMAST DD statement must be included in the JCL for the use of this routine, allocated to the CA-LIBRARIAN Master JCL library. A LIBRTEMP DD statement may optionally also be included - this will be searched first for each job's JCL, allowing temporary JCL overrides without modifying the production JCL allocated to the LIBRMAST DD statement. The routine will be called by the last defined processing jobstep of each job in the network that has post-requisite job(s), and by the SPANEX Utility for “INCLUDE”, “NETSTART”, “SCHEDULE”, “POST” or “PROCEED” command processing.

Processing logic: An MVS Internal Reader is dynamically allocated and dynamic VSAM control blocks are used for writing out the JCL. The CA-LIBRARIAN File Access Interface Routine (FAIR) is used to read the job JCL. A maximum of ten attempts, ten seconds apart, will be made to allocate an Internal Reader.

Return Codes: Return codes 4 and 12 are issued by this routine as all necessary messages are issued from within the routine. Abend U0999 is issued if this routine is run under any Operating System other than MVS.

6.2.13 Sample Submit Routine 13 -- SPXNJS13

Operating System: MVS only, with JES2 or JES3.

Submit Technique: JCL for submitted jobs must be held on a CA-LIBRARIAN Master, with fixed 80-byte records, with one CA-LIBRARIAN member per job, where the member name is the job name or the name specified by the MEMBER= parameter of the SPXJOB macro statement. JCL may also be held on the JES Job Queue with the "TYPRUN=HOLD" parameter specified on the JOB statement.

Module usage: A LIBRMAST DD statement must be included in the JCL for the use of this routine, allocated to the CA-LIBRARIAN Master JCL library. A LIBRTEMP DD statement may optionally also be included - this will be searched first for each job's JCL, allowing temporary JCL overrides without modifying the production JCL allocated to the LIBRMAST DD statement. The routine will be called by the last defined processing jobstep of each job in the network that has post-requisite job(s), and by the SPANEX Utility for "INCLUDE", "NETSTART", "SCHEDULE", "POST" or "PROCEED" command processing.

Processing logic: This routine is logically a combination of routines SPXNJS04 and SPXNJS12: the JES Job Queue is scanned, and the job is released if it is found on the queue in HOLD status. If not, an MVS Internal Reader is dynamically allocated and dynamic VSAM control blocks are used for writing out the JCL. The CA-LIBRARIAN File Access Interface Routine (FAIR) is used to read the job JCL. The Job Queue is then re-checked and the job is released if it is found to be in HOLD status when it appears on the queue (up to 25 seconds is allowed for the job to arrive on the Job Queue). This routine permits multiple runs of a job to be performed without manual intervention to place the job on the JES queue. A maximum of ten attempts, ten seconds apart, will be made to allocate an Internal Reader.

Return Codes: Return codes 4 and 12 are issued by this routine as all necessary messages are issued from within the routine. Abend U0999 is issued if this routine is run under any Operating System other than MVS.

6.2.14 Sample Submit Routine 14 -- SPXNJS14

Operating System: MVS with JES2 or JES3.

Submit Technique: JCL for submitted jobs must be held on a CA-LIBRARIAN Master, with fixed 80-byte records, with one CA-LIBRARIAN member per job, where the member name is the job name or the name specified by the MEMBER= parameter of the SPXJOB macro statement.

Module usage: A LIBRMAST DD statement must be included in the JCL for the use of this routine, allocated to the CA-LIBRARIAN Master JCL library. A LIBRTEMP DD statement may optionally also be included - this will be searched first for each job's JCL, allowing temporary JCL overrides without modifying the production JCL allocated to the LIBRMAST DD statement. A "SYSUT1" DD statement must also be supplied, referencing a sequential DASD work dataset sufficiently large to contain the JCL for any one job in the network; the DD statement must specify "DISP=SHR". This routine will be called by the last defined processing jobstep of each job in the network that has post-requisite job(s), and by the SPANEX Utility for "INCLUDE", "NETSTART", "SCHEDULE", "POST" or "PROCEED" command processing.

Processing logic: The CA-LIBRARIAN File Access Interface Routine (FAIR) is used to read the job JCL from the CA-LIBRARIAN Master. The JCL is then copied to the dataset referenced by the SYSUT1 DD statement, and an Operating System "START RDR" command is issued, using the SYSUT1 work dataset as input to the Reader. The VS/1 system should be generated with enough Readers specified to ensure that the START command issued will be successful, as this routine waits for notification from the Reader that it has completed.

Return Codes: Return codes 4 and 12 are issued by this routine as all necessary messages are issued from within the routine. Abend U0999 is issued if this routine is run under any Operating System other than MVS.

6.2.15 Sample Submit Routine 15 -- SPXNJS15

Operating System: VS/1 only.

Submit Technique: JCL for all possible submitted jobs may be placed on the Operating System JES Job Queue, with each job specifying the "TYPRUN=HOLD" parameter on the JOB statement, by the operator, before the NETSTART command is issued for the network. A LIBRMAST DD statement must be supplied for a CA-LIBRARIAN Master containing each job's JCL as a CA-LIBRARIAN member. A LIBRTEMP DD statement may optionally also be included - this will be searched first for each job's JCL, allowing temporary JCL overrides without modifying the production JCL allocated to the LIBRMAST DD statement. The CA-LIBRARIAN member name must be the same as the jobname or the name specified by the MEMBER= parameter of the SPXJOB macro statement. A "SYSUT1" DD statement must also be supplied, referencing a sequential DASD work dataset sufficiently large to contain the JCL for any one job in the network; the DD statement must specify "DISP=SHR".

Module usage: The routine will be called by the last defined processing jobstep of each job in the network that has post-requisite job(s), and by the SPANEX Utility for "INCLUDE", "NETSTART", "SCHEDULE", "POST" or "PROCEED" command processing.

Processing logic: The JES Job Queue is scanned to check that the job exists and is in HOLD status - if found it is released. If not found, the JCL is obtained from CA-LIBRARIAN and is copied to the work dataset referenced by the SYSUT1 DD statement. A START RDR command is issued against the SYSUT1 dataset the JCL on the Job Queue. (The VS/1 system should be generated with enough Readers specified to ensure that the START command issued will be successful, as this routine waits for notification from the Reader that it has completed.) The queue is then checked to ensure that the job has arrived, and the job is released if it appears in HOLD status. Up to 25 seconds is allowed for the job to reach the queue.

Return Codes: Return codes 4 and 12 are issued by this routine as all necessary messages are issued from within the routine. Abend U0999 is issued if this routine is run under any Operating System other than VS/1.

6.3 User End-of-Network Exit Routines

SPANEX will optionally invoke a user exit routine to handle the condition of successful completion of all processing in one run of a SPANEX Job Network. The name of this routine, if any, is specified during the RCM generation process by means of the "EONRTN=" parameter of the SPXRCM or QUICKNET macro.

Upon successful completion of the Job Network, after all status has been erased from the SPANEX Catalog (if "ERASEON=YES" is specified on the SPXRCM macro), the user end-of-network routine will be loaded (the TASKLIB dataset, if any, will be searched first) and invoked by SPANEX with registers set as follows:

Register 0	➜	SPANEX Catalog Work Area (mapped by the #SPXCWA macro)
Register 1	➜	SPANEX ICB, which contains or points to all information used by SPANEX (the SPANEX ICB is mapped by the #SPXICB macro)

The user routine is entered under the control of the SPANEX major task, and has the authority to issue any SPANEX macros. Any required processing may be performed. A typical use of the end-of-network exit may be to print the contents of the SPANEX Global Log, and a sample routine to perform this function for MVS installations is provided in the SPANEX source library with a member name of SAMPEONR.

6.4 User NETSTART Exit Routines

SPANEX will optionally invoke a user exit routine during the execution of a NETSTART command for a job network. The name of this routine, if any, is specified during the RCM generation process by means of the "STREXIT=" parameter of the SPXRCM or QUICKNET macro.

Before any action has been taken by the NETSTART processor, this user routine is invoked, and the routine can determine whether a start of the network at this time is permissible (status information held elsewhere may be checked, for example, to prevent multiple runs of a network on the same day). The routine can also perform other processing, such as ensuring that the Global Log data from the previous network execution has been printed or saved (this function may also be performed by the end-of-network user exit, but that routine will not be invoked if, for any reason, the previous network execution was abandoned by use of the "STATUS COMPLETE" or "STATUS DELETE" command). The NETSTART user exit routine is entered with registers set as follows:

Register 0	➜	SPANEX Catalog Work Area (mapped by the #SPXCWA macro)
Register 1	➜	SPANEX ICB, which contains or points to all information used by SPANEX (the SPANEX ICB is mapped by the #SPXICB macro)

The user routine is entered under the control of the SPANEX major task, and has the authority to issue any SPANEX macros. Any required processing may be performed, and the routine should return to SPANEX with Register 15 containing one of the following values:

0	-	NETSTART may proceed;
4	-	NETSTART is to be terminated with no change being made to the status of the network.

6.5 Application Program Options

No changes to existing application programs are necessary in order to make full use of the SPANEX Restart and Job Networking facilities, and no special coding techniques are required. Any programming language may be used for application programs.

SPANEX does, however, provide the #SPXRSTU macro for assembler language programs, which permits access to and control of certain aspects of the running of a SPANEX restart-controlled job and of the processing of a SPANEX job network. The coding of this macro is described in detail in the Span Macros Manual, and the concepts of its facilities are described below.

6.5.1 Status Inquiry/Update Option

The #SPXRSTU macro options SETERR, SETOK and INQ can be used in a restart-controlled application program to force a SPANEX restart of the current job, to prevent a SPANEX restart of the current job, or to inquire as to any previous SETERR or SETOK macro calls, respectively.

The “TYPE=SETERR” macro call will force a restart of this job the next time the job is executed, will set the completion of this step as an error regardless of the return code from the application program, and will reset any previous “TYPE=SETOK” macro call.

The “TYPE=SETOK” macro call will set the completion of this step to appear successful, and will reset any previous “TYPE=SETERR” macro call. This will prevent SPANEX restart from recovering if there is any failure during this step, even if the application program abends or issues an error return code.

The “TYPE=INQ” macro call will return a code to the application program informing it of whether or not SPANEX currently considers an error to have occurred during the processing of this step, as a result either of a program error or of a “TYPE=SETERR” macro call.

The #SPXRSTU macro may be issued any number of times during the execution of an application program.

6.5.2 Job Network Control

Selection of jobs to run in a SPANEX job network can be performed by dynamic decision-making during the execution of application programs which form part of the job network suite itself. Control of jobs defined in a different Job Network is not supported from within an application program, but jobsteps may be included at any point to perform a batch execution of the SPANEX Utility to issue commands against any job in any Network.

Subject to the restrictions shown below, the user program may issue one or more “TYPE=CANCEL“ and “TYPE=SCHEDULE” macro calls to delete or release, respectively, other jobs within its own job network. The special

"TYPE=(SCHEDULE,FORCE)" macro call may be used to schedule a job that is currently EXCLUDEd from the network. Job "Hold Events" for other jobs within the same Network may also be manipulated by the use of the "TYPE=HOLD" and "TYPE=POST" options. It should be understood that SPANEX will not action any job scheduling which becomes necessary as a result of these macro calls until the termination of the user application program, which is treated as the synchronization point for this purpose; however, for some options the changed status of the subject jobs will be recorded immediately, preventing any possible interlock.

The restrictions governing the use of the #SPXRSTU macro for dynamic job network control are as follows:

- program issuing #SPXRSTU macro must be executing under the control of SPANEX with the "OPT=M" execution option.
- the RCM must be generated as a job network ("JOBNET=YES" specified on the SPXRCM macro).
- the job in which the program is executing must be part of the same SPANEX job network as the job named in the #SPXRSTU macro call.
- a job to be cancelled, scheduled or held must not yet have begun execution.

Return codes as documented in the Span Macros Manual are provided after the execution of the #SPXRSTU macro.

6.6 Utility Access Control Exit Routines

SPANEX provides an optional user exit facility in order to provide access control for the commands of the SPANEX Utility. This facility is in addition to other access control options that may be defined on the SPXRCM or QUICKNET macro ("OPTU=", "TSUPD=" and "TSNET=" parameters). A sample routine is provided in the SPANEX source library with a member name of SPXURCMV.

If used, this routine must have a CSECT name of SPXURCMV, and should be link-edited with the SPANEX SPXM0130 load module. It will be invoked by SPANEX each time an RCM load module is fetched into storage by the Utility. The routine has access to all control information available from SPANEX or Operating System control blocks. At the end of the processing of this user exit routine, a code must be returned to SPANEX indicating the authority of this execution of the Utility against the RCM just loaded. These return codes are listed below. The Utility Access Control user exit routine is entered with registers set as follows:

Register 0	➜	RCM header (mapped by the #SPXRDEF macro)
Register 1	➜	SPANEX ICB, which contains or points to all information used by SPANEX (the SPANEX ICB is mapped by the #SPXICB macro)

The user routine is entered under the control of the SPANEX major task, and has the authority to issue any SPANEX macros. Any required processing may be performed, and the routine should return to SPANEX with Register 15 containing one of the following values:

0	-	this Utility execution has all authority, except as defined during the RCM generation process;
4	-	this Utility execution may not perform UPDATE or DELETE commands, but has all other authority except as defined during the RCM generation process;
8	-	this Utility execution may not perform job networking control commands, but has all other authority except as defined during the RCM generation process;
12	-	this Utility execution may not perform UPDATE or DELETE commands, or job networking control commands;
16	-	this Utility execution may not access this RCM at all.

Note that the coding of this routine must be reentrant. It is recommended that the Span Software #SPANTRY and #SPANXIT macros are used to provide reentrant module linkage (see the Span Macros manual). Other macros that may be required are #SPXICB to map the SPANEX ICB, and #SPXRDEF to map the various blocks of the RCM. The SPANEX Automated Data Areas manual should be studied to show the layout of these control blocks.

Some control block fields that may be of use to this routine are:

SPXURCMN (SPXICB DSECT)	-	name of the RCM to be examined
SPXATIOT (SPXICB DSECT)	-	address of TIOT (Jobname/TSO userid)
SPXTSFLG (SPXICB DSECT)	-	address of a 1-byte field, whose high-order bit (X'80') is set if this is a TSO execution
SPXFLG12 (SPXICB DSECT)	-	bit SPXFCPRE in this byte is set if this is an extended TP execution (using SPANEX full-screen 3270 terminal support).

7 The SPANEX Utility Facility

The SPANEX Utility is invoked when "OPT=U" is specified as a SPANEX parameter in a batch or TSO execution, or via one of the Extended TP support modules. The SPANEX Utility provides the ability for the user or the system operator to display or modify the status of any SPANEX-controlled job, and also provides the main vehicle for exercising control over SPANEX job networks. The modification of restart status can be prevented for jobs within a given RCM by specifying "OPTU=NO" on the SPXRCM or QUICKNET macro, although this does not prevent the displaying of status information.

The SPANEX Utility will accept command input from the operator (MCS) console, via the SPANEX Extended TP support (from a 3270-type terminal), or via 80-byte control statements (which may also be input from a time-sharing terminal); if a SPXRSCTL DD statement is provided (mandatory for standard TSO support), control statement input (80-byte records) is expected; if no SPXRSCTL DD statement is provided in batch mode, WTOR messages are sent to the MCS console requesting command input.

The SPANEX Utility currently accepts nineteen control commands: CSHEET, DELETE, DISPLAY, EXCLUDE, HALT, HOLD, INCLUDE, INPUT, LOG, MAP, NETSTART, POST, PRINT, PROCEED, SCHEDULE, SHOW, STATUS, TRACE, UPDATE. These operate upon the SPANEX status that is maintained for all SPANEX-controlled jobs in the SPANEX Catalog. All commands are accepted from each of the available input sources. Command parsing is performed by the standard Span Software service routine SPZPARSE, and syntax error messages are issued by this routine where appropriate - these error messages are documented in the SPANEX Messages and Codes manual. The special CATMAINT command is provided by the SPANEX Utility for the maintenance of SPANEX Catalog datasets when using the native VSAM KSDS option; this command is documented in the SPANEX Installation and Maintenance manual.

Additionally, certain functions of the SPANEX Utility are accessible from a user program or SPANEX Installation Exit routine by means of the #SPXRSTU macro. See Section 6.5 on page 81 of this manual, and the description of the #SPXRSTU macro in the Span Macros manual.

Note that the "PASSWORD=" parameter may be specified on all of the SPANEX Utility commands, even though it is not required and has no use on most of the display-type commands. This is to permit these commands to be enclosed in a SPANEX catalogued command member, and to be called up with an overall "PASSWORD" parameter specified on the "INPUT" command.

7.1 SPANEX Utility - CSHEET Command

The CSHEET command (no abbreviation) is used to produce a customized SPANEX Job Check Sheet for a SPANEX job network. When used with the DATE= parameter, the CSHEET command produces a Job Confirmation Sheet, which lists jobs that are expected to run on any date in the future (or which might have run on any date in the past), using SPANEX Calendar definitions. The Job Check Sheet may alternatively optionally be produced by the NETSTART command. If the Sheet is to be printed, a SPXNETCS DD statement is required (unless the MVS spin-off option is requested). The Job Confirmation Sheet will be produced as a full-screen display if the command is entered from a 3270-type terminal.

syntax:

```
          [NET=rcmname]   [,PRENS ]
CSHEET    [           ]   [       ]   [,SPIN=sysoutclass]
          [RCM=rcmname]   [,POSTNS]

                [       [TODAY          ]]
                [       [TOMORROW       ]]
                [,DATE=[YESTERDAY       ]]
                [       [(MMDD,mmdd)    ]]
                [       [(YYMMDD,yymmdd)]]
                [       [(JULIAN,ddd)   ]]
```

where:

NET= - specifies the name of the RCM (or Network) for which a customized Job Check Sheet is to be printed. This parameter is required, unless the "NET=" or "RCM=" operand was specified in the SPANEX EXEC statement PARM or in a SPANEX VDU "SET" command, in which case it will not be accepted and the EXEC PARM network name will be used. The "NET=" and the "RCM=" parameters are synonymous.

PRENS - specifies that the network status is pre-NETSTART (all EXCLUDE commands have been issued but the network is not yet started) and the job check sheet is not to contain jobs for which EXCLUDE commands are pending for the forthcoming network execution. This is the default if neither "PRENS" nor "POSTNS" is specified.

POSTNS - specifies that the NETSTART command has already been issued and that the job check sheet is not to contain jobs which are EXCLUDEd from the execution of the network that has already begun.

SPIN= - specifies MVS spin-off for the printed Job Check Sheets (this operand is ignored if the Operating

System is not MVS). This parameter specifies the SYSOUT Class to be used for the printed sheets and should be a single character in the range A-Z or 0-9. The Job Check Sheet will appear on the SYSOUT queue immediately upon completion of the command.

DATE= - specifies that a Job Confirmation Sheet is to be produced instead of a Job Check Sheet. This parameter will be ignored if there are no calendar definitions in the RCM. The use of the DATE= parameter is to use a calendar date (either in the future or in the past) for which a hypothetical list of executed jobs is to built.
DATE=TODAY specifies that the current date, (not modified by any NEWDAY value), is to be used to produce a Job Confirmation Sheet.
DATE=TOMORROW specifies that the date of the day after the day on which the CSHEET command is issued is to be used to produce a Job Confirmation Sheet.
DATE=YESTERDAY specifies that the date of the day before the day on which the CSHEET command is issued is to be used to produce a Job Confirmation Sheet.
DATE=(MMDD,mmdd) specifies an explicit date as four decimal digits, representing the calendar month and the day within the month. This is the date, within the current calendar year, that is to be used to produce a Job Confirmation Sheet.
DATE=(YYMMDD,yymmdd) specifies an explicit date as six decimal digits, representing the low-order two digits of the year, the calendar month within that year, and the day within that month. This is the date that is to be used to produce a Job Confirmation Sheet.
DATE=(JULIAN,ddd) specifies an explicit date as three decimal digits, representing the julian day number within the current calendar year, of the date that is to be used to produce a Job Confirmation Sheet.

7.2 SPANEX Utility - DELETE Command

The DELETE command (abbreviation "DEL") is used to remove all trace of a given job from the SPANEX Catalog. By default, job status is "logically deleted" from the Catalog, so that it does not appear in SPANEX status reports, but so that SPANEX job statistics information is not lost. The Delete capability may be specifically permitted for time-sharing users for a particular network by means of the "TSUPD=YES" parameter of the SPXRCM or QUICKNET macro. This command may be used to cancel the effect of an EXCLUDE command for a job, prior to issuing the NETSTART command, if the EXCLUDE command was issued in error.

syntax:

```
[DELETE]   [NET=rcmname]        [jobname]
[      ]   [            ]  ,JOB=[       ]  [,PURGE]
[DEL   ]   [RCM=rcmname]        [/ALL   ]
```

where:

NET= - specifies the name of the RCM (or Network) which contains the specification of the job whose restart status is to be deleted. This parameter is required, unless the "NET=" or "RCM=" operand was specified in the SPANEX EXEC statement PARM or in a SPANEX VDU "SET" command, in which case it will not be accepted and the EXEC PARM network name will be used. The "NET=" and the "RCM=" parameters are synonymous.

JOB= - specifies the name of the job whose restart status is to be deleted. This parameter is required. Specifying "JOB=/ALL" will cause the status of all jobs defined in this RCM to be deleted, and this may be used as an equivalent to the "ERASEON=YES" parameter of the SPXRCM macro to clear out job status after network completion.

PURGE - specifies that the job status information is to be physically deleted from the SPANEX Catalog. All job statistics data will be lost if this option is used. Use of this option is the only way to cause the physical removal of SPANEX job information from the Catalog; all other methods cause a logical delete to be performed.

7.3 SPANEX Utility - DISPLAY Command

The DISPLAY command (abbreviation "D") is used to display to the operator the SPANEX status of a particular job or all jobs in a SPANEX job network.

syntax:

```
[DISPLAY]   [NET=rcmname]       [jobname]
[       ]   [            ]  ,JOB=[       ]  [,PREQS]
[D      ]   [RCM=rcmname]       [/ALL   ]
```

where:

NET= - specifies the name of the RCM (or Network) which contains the specification of the job or jobs whose status is to be displayed on the operator console. This parameter is required, unless the "NET=" or "RCM= operand was specified in the SPANEX EXEC statement PARM or in a SPANEX VDU "SET" command, in which case it will not be accepted and the EXEC PARM network name will be used. The "NET=" and the "RCM=" parameters are synonymous.

JOB= - specifies the name of the job whose restart status is to be displayed. This parameter is required. Specifying "JOB=/ALL" will cause the status of all jobs defined in this RCM to be displayed. The abbreviation "J=" is accepted for the "JOB" parameter.

PREQS - specifies that all pre- and post-requisite jobs, and any outstanding "Hold Events", are to be listed for each job whose status is displayed.

7.4 SPANEX Utility - EXCLUDE Command

The EXCLUDE command (abbreviation "EXC") is used to exclude from a SPANEX job network one job, for a particular execution of the network only. Any desired EXCLUDE commands should be entered before the NETSTART command for the network is issued (although the "NOW" option permits job exclusion whilst the network is running). The capability to use this command may be specifically permitted for time-sharing users for a particular RCM by means of the "TSNET=YES" parameter of the SPXRCM or QUICKNET macro. If an EXCLUDE command has been issued for a job and it is later determined that the job is required to run, the effect of the EXCLUDE command can be undone by means of an "INCLUDE" command for the job. It is also possible to undo an EXCLUDE command by means of an application program call to SPANEX (see Section 6.5 of this manual).

syntax:

```
[EXCLUDE] [NET=rcmname]                [,PASSWORD=password]
[       ] [           ] ,JOB=jobname   [                  ]
[EXC    ] [RCM=rcmname]                [,PASSW=password   ]

                        [,NOW ]
                        [     ]
                        [,NEXT]
```

where:

NET= - specifies the name of the RCM (or Network) which contains the definition of the job network containing the specification of the job that is to be EXCLUDEd. The "JOBNET=YES" option must have been specified on the SPXRCM macro when the RCM generation process was performed (this is default if the QUICKNET macro was used). This parameter is required, unless the "NET=" or "RCM=" operand was specified in the SPANEX EXEC statement PARM or in a SPANEX VDU "SET" command, in which case it will not be accepted and the EXEC PARM network name will be used. The "NET=" and the "RCM=" parameters are synonymous.

JOB= - specifies the name of the job which is to be EXCLUDEd from this execution of the network. This parameter is required. The abbreviation "J=" is accepted for the "JOB" parameter.

PASSWORD= - specifies the password (maximum 8 characters) for network commands for this network. If the requirement for a password was defined when the RCM was generated, then this operand is required, and the command will be rejected if the password is omitted or incorrect.

NOW	-	specifies that the job should be EXCLUDEd from the execution of the network that has already begun, ie the EXCLUDE command has been issued after the successful completion of NETSTART for this network and while the network is active.
NEXT	-	specifies that the job should be EXCLUDEd from the next execution of the network. If the network is not currently active, the NEXT option has no effect and, as normal, the EXCLUDE applies to the forthcoming NETSTART of the network. If the network is active, the NEXT option will cause the job to be EXCLUDEd from the following run of the network, and the EXCLUDE command has no effect on the current network run.

7.5 SPANEX Utility - HALT Command

The HALT command (abbreviation "Z") is used to suspend the execution of a SPANEX job network. When the appropriate stage of each executing job is reached, processing of that job will stop. Network execution can be resumed by means of the "PROCEED" SPANEX command. The capability to use this command may be specifically permitted for time-sharing users for a particular RCM by means of the "TSNET=YES" parameter of the SPXRCM or QUICKNET macro.

syntax:

```
[HALT]   [NET=rcmname]   [,EOJ]   [,PASSWORD=password]
[    ]   [           ]   [    ]   [                    ]
[Z   ]   [RCM=rcmname]   [,EOS]   [,PASSW=password    ]
```

where:

NET= - specifies the name of the RCM (or Network) whose processing is to be halted temporarily. The "JOBNET=YES" option must have been specified on the SPXRCM macro when the RCM generation process was performed (this is the default if the QUICKNET macro was used). This parameter is required, unless the "NET=" or "RCM=" operand was specified in the SPANEX EXEC statement PARM or in a SPANEX VDU "SET" command, in which case it will not be accepted and the EXEC PARM network name will be used. The "NET=" and the "RCM=" parameters are synonymous.

EOJ - specifies that jobs that are currently executing are to be HALTed at end-of-job. No interruption of individual job processing is to be performed. This is the default option if neither "EOJ" or "EOS" is specified.

EOS - specifies that jobs that are currently executing are to be HALTed at end-of-step. As the currently-executing step (or the next SPANEX-controlled step) of each active job terminates, an internal CANCEL command will be issued for the job. Normal SPANEX restart processing will be performed for all jobs cancelled in this way when the PROCEED command is issued to resume processing of the network. This option should be used only for networks whose jobs have SPANEX restart fully implemented.

PASSWORD= - specifies the password (maximum 8 characters) for network commands for this network. If the requirement for a password was defined when the RCM was generated, then this operand is required, and the command will be rejected if the password is omitted or incorrect.

7.6 SPANEX Utility - HOLD Command

The HOLD command (no abbreviation) is used to set a specific job in a SPANEX job network to "hold" status on one of its 8 external events. Each job in a SPANEX network may have up to 8 events specified which must be complete before the job will be executed. These events are assigned arbitrary numbers 1-8, and the meaning of each event may be assigned by the user. A job may be set into "hold" status for one of its events by the use of the HOLD command, or it may be defined with outstanding events on the SPXJOB or QUICKJOB macro when the RCM is generated. An event is set complete by means of the POST SPANEX command, or via an application program call to the SPANEX Utility. The HOLD command is not accepted after the job has begun execution. The capability to use this command may be specifically permitted for time-sharing users for a particular RCM by means of the "TSNET=YES" parameter of the SPXRCM or QUICKNET macro.

syntax:

```
       [NET=rcmname]   [,JOB=jobname]
HOLD   [            ]  [             ]  ,EVENT=n
       [RCM=rcmname]   [,J=jobname  ]

                           [,PASSWORD=password]
                           [                  ]
                           [,PWORD=password   ]
```

where:

NET= - specifies the name of the RCM (or Network) which contains the job whose event is to be set incomplete. The "JOBNET=YES" option must have been specified on the SPXRCM macro when the RCM generation process was performed (this is the default if the QUICKNET macro was used). This parameter is required, unless the "NET=" or "RCM=" operand was specified in the SPANEX EXEC statement PARM or in a SPANEX VDU "SET" command, in which case it will not be accepted and the EXEC PARM network name will be used. The "NET=" and the "RCM=" parameters are synonymous.

JOB= - specifies the name of the job whose Event is to be set incomplete. This parameter is required. The abbreviation "J=" is accepted for the "JOB" parameter.

EVENT= - specifies the event identifier (single digit, 1-8) of the event for this job that is to be set incomplete. The job will not now execute until this event is signalled complete by means of the POST SPANEX command, or via an application program call.

PASSWORD=	-	specifies the password (maximum 8 characters) for network commands for this network. If the requirement for a password was defined when the RCM was generated, then this operand is required, and the command will be rejected if the password is omitted or incorrect.

7.7 SPANEX Utility - INCLUDE Command

The INCLUDE command (abbreviation "INC") is used to undo the effect of an EXCLUDE command or of the "PROCESS=EXCLUDE" option of the SPXJOB or QUICKJOB macro. If, at the time the INCLUDE command is issued, the job would have run had it not been EXCLUDEd, it will be submitted for execution by the INCLUDE command. Otherwise, the effect of the INCLUDE command is merely to remove the "EXCLUDEd" status for the job. The capability to use this command may be specifically permitted for time-sharing users for a particular RCM by means of the "TSNET=YES" parameter of the SPXRCM or QUICKNET macro.

syntax:

```
[INCLUDE] [NET=rcmname]                [,PASSWORD=password]
[       ] [           ] ,JOB=jobname   [                  ]
[INC    ] [RCM=rcmname]                [,PASSW=password   ]

                          [,NOW  ]
                          [      ]
                          [,NEXT]
```

where:

NET= - specifies the name of the RCM (or Network) which contains the definition of the job network containing the specification of the job that is to be INCLUDEd. The "JOBNET=YES" option must have been specified on the SPXRCM macro when the RCM generation process was performed (this is the default if the QUICKNET macro was used). This parameter is required, unless the "NET=" or "RCM=" operand was specified in the SPANEX EXEC statement PARM or in a SPANEX VDU "SET" command, in which case it will not be accepted and the EXEC PARM network name will be used. The "NET=" and the "RCM=" parameters are synonymous.

JOB= - specifies the name of the job which is currently EXCLUDEd from this execution of the network and which is now to be considered part of this run. This parameter is required. The abbreviation "J=" is accepted for the "JOB" parameter.

PASSWORD= - specifies the password (maximum 8 characters) for network commands for this network. If the requirement for a password was defined when the RCM was generated, then this operand is required, and the command will be rejected if the password is omitted or incorrect.

NOW	-	is the default option of INCLUDE, and specifies that the command applies to the current run of the network. An INCLUDE command with the NOW option will be accepted only whilst the network is active.
NEXT	-	specifies that the job should be INCLUDEd into the next execution of the network. If the network is not currently active, the NEXT option of INCLUDE reverses the effect of an EXCLUDE command, or nullifies the PROCESS=EXCLUDE option for the forthcoming NETSTART of the network. If the network is active, the NEXT option will cause the job to be INCLUDEd into the following run of the network in the same way, reversing the effect of an "EXCLUDE NEXT" command or nullifying the PROCESS=EXCLUDE option, and the INCLUDE command has no effect on the current network run.

7.8 SPANEX Utility - INPUT Command

The INPUT command (no abbreviation) is used to invoke a sequence of SPANEX Utility commands from the SPANEX command library. The SPANEX command library is either a partitioned dataset of 80-byte logical records, where each PDS member is a set of SPANEX commands that are to be issued in sequence, or a CA-PANVALET library or CA-LIBRARIAN master, where each member is a set of SPANEX commands. The member name of a set of commands is entered as an operand of the INPUT command in order to execute all the commands contained in that member. The SPANEX command library is accessed by the SPANEX Utility via a fixed DDNAME of "SPXUTLIB" if it is a PDS, via a fixed DDNAME of "SPXUTPAN" if it is a CA-PANVALET library, or via a fixed DDNAME of "SPXUTMST" if it is a CA-LIBRARIAN master. Both a PDS and a CA-PANVALET or CA-LIBRARIAN library may be supplied to a single execution of the SPANEX Utility, but the LIBTYPE parameter will need to be specified if both a PDS and one of the other library types is supplied, and the member required is not in the PDS.

syntax:

```
        [NET=rcmname]   [,MEMBER=membername]
INPUT   [           ]   [                  ]
        [RCM=rcmname]   [,MEM=membername   ]

              [,PASSWORD=password]  [          [PDS ]]
              [                  ]  [,LIBTYPE=[PANV]]
              [,PASSW=password   ]  [          [LIBR]]

              [,NOCONFRM]
              [         ]
              [,NOCO    ]
```

where:

NET= - specifies the name of the RCM (or Network) against which all the commands in this member of the SPANEX command library are to be executed. This parameter may be specified in one of three places: (1) via the "NET=" or "RCM=" operand of the SPANEX EXEC statement PARM or in a SPANEX VDU "SET" command (in which case it will not be accepted as an operand of the INPUT command and the EXEC PARM network name will be used for all commands in the input member); (2) via this operand of the INPUT command (in which case this network name will be used for all commands in the input member); or (3) on the individual commands in the input member, thus allowing different networks to be accessed from a single INPUT command. The "NET=" and the "RCM=" parameters are synonymous.

MEMBER= - specifies the PDS or CA-PANVALET or CA-LIBRARIAN member name in the SPANEX command library of the sequence of SPANEX Utility commands to be executed. This parameter is required. The abbreviation "MEM=" is accepted for the MEMBER parameter.

PASSWORD= - specifies the password (maximum 8 characters) for network commands contained in this INPUT member. This password will be used for all the commands in this member, unless the individual commands have the PASSWORD operand specified - in which case the password on the statement within the input member is used in preference to the one specified here. The use of this parameter avoids the possible security exposure associated with including network command passwords within the SPXRSCTL dataset.

LIBTYPE= - specifies the type of library (PDS, CA-PANVALET or CA-LIBRARIAN) in which the requested member resides. This parameter is required only if multiple DDNAMEs are supplied to the SPANEX Utility. The PDS, if present, is always searched if this parameter is not specified.

NOCONFRM - See the description of the NOCONFRM parameter of the NETSTART command. The purpose of the NOCONFRM parameter of the INPUT command is to allow the NOCONFRM option to be propagated to a NETSTART command that is contained within a SPANEX command library input member.

7.9 SPANEX Utility - LOG Command

The LOG (or GLOG) command is used to add user data to the SPANEX Global Log dataset for a SPANEX network. This may be useful for documentation purposes, or for adding operator comments to explain or clarify abnormal conditions.

syntax:

```
[LOG ]    [NET=rcmname]
[    ]    [           ]   ,TEXT='user data'
[GLOG]    [RCM=rcmname]
```

where:

NET= - specifies the name of the RCM (or Network) whose Global Log dataset is to be updated with user information. This parameter is required, unless the "NET=" or "RCM=" operand was specified in the SPANEX EXEC statement PARM or in a SPANEX VDU "SET" command, in which case it will not be accepted and the EXEC PARM network name will be used. The "NET=" and the "RCM=" parameters are synonymous.

TEXT= - specifies the user information that is to be added to the Global Log dataset. Up to 100 bytes of data may be specified, enclosed in single quotes if blanks or special characters are included in the data. The "TEXT=" parameter is mandatory for the LOG command.

7.10 SPANEX Utility - MAP Command

The MAP command (no abbreviation) is used to invoke the SPANEX Utility RCM Map sub-function. The RCM Map routine is used to produce full-screen on-line displays of the status of all jobs in a network, together with optional hard-copy, and some other functions such as a report on the generation options used in a particular RCM, or a SPANEX Wall Chart giving a pictorial representation of the relationships between jobs of a network. Certain extra DD statements may be required for the printing of some of the output of the MAP command (unless the MVS spin-off parameter option is requested - see Section 8.3 on page 121 of this manual).

syntax:

```
      [NET=rcmname]   [       ([NOVDU,    ....]
MAP   [            ]   [,PARM=([ABBR,     ....]
      [RCM=rcmname]   [       ([OPTS,     ....]
                      [       ([CALENDAR, ...]
                      [       ([RESETP,   ....]

                    [          [,[SIZE1]]          ]
                    [,WALLCHRT[,[     ]] [,STATUS]]
                    [          [,[SIZE2]]          ]

                    [,TABLE=(calendar [,day-type] )]

                    [,SPIN=sysoutclass], ....)]
```

where:

NET= - specifies the name of the RCM (or Network) whose job information is to be accessed by the RCM Map function. This parameter is required, unless the "NET=" or "RCM=" operand was specified in the SPANEX EXEC statement PARM or in a SPANEX VDU "SET" command, in which case it will not be accepted and the EXEC PARM network name will be used. The "NET=" and the "RCM=" parameters are synonymous.

PARM= - specifies the optional parameter information to be passed to the RCM Map program to request the various functions of this routine. If more than one option is required, enclose all options, separated by commas with no embedded blanks, in either quotes or parentheses. If the "PARM=" parameter is not specified, a full-screen status display of all jobs defined in the RCM will be produced.
"NOVDU" requests that the RCM Map Status and Options reports are produced as hard copy (RCMAPOUT DD statement or spin-off) only, and not on the user's terminal.
"ABBR" (abbreviation "AB" is also accepted) specifies that the RCM Map Status report is to be produced in abbreviated form (any jobs that are

currently EXCLUDEd, which have already completed successfully or which have not yet been scheduled are not listed); this permits a more manageable size of report or display to be produced for an RCM that describes a very large number of jobs.
"OPTS" requests the RCM Map Options report (note that the "STATUS" option must also be requested if it is required; the "STATUS" option should also be requested if a Wall Chart is requested within the same command).
"CALENDAR" (abbreviation "CAL" is also accepted), requests a list of the Jobs in the RCM that have Calendar definitions associated with them; the Calendar day-type names are given for each job. Note that Network Calendar definitions are given as part of the "OPTS" report.
"RESETP" (abbreviation "REP" is also accepted) is used if either or both of the RCMAPOUT and WALLCHRT DD statements are allocated to disk datasets, and the new data created by this command is not to be added to the end of the datasets but written to the beginning; if the "RESETP" option is not specified the datasets will be treated as though the "DISP=MOD" JCL parameter were used.
"WALLCHRT" (abbreviation "WALL" is also accepted) requests a SPANEX RCM Map Wall Chart. Default size is "SIZE1", the smaller of the two available sizes.
"SIZE2" (abbreviation "S2" is also accepted), in combination with the "WALLCHRT" option, requests a large-size Wall Chart.
"STATUS" (abbreviation "ST" is also accepted), in combination with the "WALLCHRT" option, requests a Status report (with optional Options report) as well as a Wall Chart.
"TABLE=(calendar[,day-type])" (abbreviation "TAB=" is also accepted) is used to request a display of the contents of a SPANEX Calendar module. The calendar value is the name of the Calendar load module ("SPXCALnn" for the SPANEX System Calendar, or "SPXxxxnn" for a SPANEX User Calendar, where "nn" is the two low-order digits of the calendar year, and "xxx" is the User Calendar identifier). If there is no day-type parameter, a list of the day-types defined within the calendar module is given. If a day-type parameter is specified, a calendar display is produced showing the days in the year that are part of the day-type. If a day-type of "/ALL" is specified, then calendar displays of all the day-types in the Calendar module are produced. If a day-type parameter is specified, the operands of the TABLE= parameter must be enclosed in parentheses.

"SPIN=sysoutclass" (MVS only) specifies that hard-copy output of all the reports produced by this command should be printed by means of a Dynamically Allocated SYSOUT dataset. Specify a single-character SYSOUT class in the range A-Z or 0-9, and the report will be available for printing immediately upon completion of the MAP command processor.

7.11 SPANEX Utility - NETSTART Command

The NETSTART command (no abbreviation) is used to initiate the execution of an entire suite of jobs that is defined as a SPANEX job network. The NETSTART command should normally be issued after all necessary EXCLUDE commands have been issued for jobs not required. When the SPANEX Utility is executed as a batch job or as a Started Task, the console operator will be asked to confirm that a NETSTART is to be performed for this network. The NETSTART command capability may be specifically permitted for time-sharing users for a particular RCM by means of the "TSNET=YES" parameter of the SPXRCM or QUICKNET macro. If the "SPXNETCS" DD statement is supplied (allocated to any output dataset, eg SYSOUT), a SPANEX Job Check Sheet will be printed for all non-EXCLUDEd jobs in the network, unless the "NOSHEET" parameter is specified.

syntax:

```
            [NET=rcmname]   [,PASSWORD=password]   [,NOSHEET]
NETSTART    [            ]   [                 ]   [        ]
            [RCM=rcmname]   [,PASSW=password   ]   [,NOSH   ]

                  [,NOCONFRM]
                  [          ]   [,RESET]   [,NEXT]
                  [,NOCO     ]

                  [       [TODAY          ]]
                  [       [TOMORROW       ]]
                  [,DATE=[YESTERDAY       ]]
                  [       [(MMDD,mmdd)    ]]
                  [       [(YYMMDD,yymmdd)]]
                  [       [(JULIAN,ddd)   ]]
```

where:

NET= - specifies the name of the RCM (or Network) which contains the definition of the job network which is to be initiated. The "JOBNET=YES" option must have been specified on the SPXRCM macro when the RCM generation process was performed (this is the default if the QUICKNET macro was used). This parameter is required, unless the "NET=" or "RCM=" operand was specified in the SPANEX EXEC statement PARM or in a SPANEX VDU "SET" command, in which case it will not be accepted and the EXEC PARM network name will be used. The "NET=" and the "RCM=" parameters are synonymous.

PASSWORD= - specifies the password (maximum 8 characters) for network commands for this network. If the requirement for a password was defined when the RCM was generated, then this operand is required, and the command will be rejected if the password is omitted or incorrect.

NOSHEET - specifies that the NETSTART processor is not to produce a SPANEX Job Check Sheet even if there is a SPXNETCS DD statement provided. The CSHEET command can be used to produce Job Check sheets when required. The abbreviation "NOSH" is accepted for the NOSHEET parameter.

NOCONFRM - specifies that the NETSTART processor is not to request confirmation before proceeding with the NETSTART operation. Without this option, an SPX857A message is issued to the operator to request permission to continue with the NETSTART process. The abbreviation "NOCO" is accepted for the NOCONFRM parameter.

RESET - specifies that this NETSTART is to reset the execution statistics maintained by SPANEX for this Job Network. The average run-time, last run duration and date, and execution count data will be lost.

NEXT - specifies that this NETSTART is to be queued until the currently active execution of the network is complete. When the current execution completes, a new NETSTART will be automatically initiated by SPANEX, with the network job mix being tailored by means of any "EXCLUDE NEXT" or "INCLUDE NEXT" commands that have been previously issued. The use of the NEXT option of the NETSTART, EXCLUDE and INCLUDE commands is to allow a further network execution to be set up whilst the current run is still in progress. This is particularly valuable in the situation where a problem, which is still under investigation, occurred in the application jobs of a network; the NEXT option allows the planning and scheduling of the next day's work to be continued as normal.

DATE= - specifies a date override for the calendar processing included in this Job Network. This parameter will be ignored if there are no calendar definitions in the RCM. The use of the DATE= parameter is to determine the calendar date that will be used to build a list of jobs that are to be run in this execution. The default date is "TODAY", which implies the date on which the NETSTART command is issued, possibly qualified by the use of the NEWDAY value obtained either from the SPANEX Calendar of from the RCM. The NEWDAY value is the time-of-day at which the calendar is deemed to change from one date to the next; the default time is midnight. Note that, if the DATE= parameter is specified for a NETSTART command that also includes the NEXT option, the date will be used to configure

the next execution of the Job Network at the time the automatic Netstart process is begun.
DATE=TODAY specifies that the current date, modified by the NEWDAY value, is to be used to determine whether or not this Job Network is to be run at all, and, if it is, the job combination to be run.
DATE=TOMORROW specifies that the date of the day after the day on which the NETSTART command is issued is to be used to determine whether or not this Job Network is to be run at all, and, if it is, the job combination to be run.
DATE=YESTERDAY specifies that the date of the day before the day on which the NETSTART command is issued is to be used to determine whether or not this Job Network is to be run at all, and, if it is, the job combination to be run.
DATE=(MMDD,mmdd) specifies an explicit date as four decimal digits, representing the calendar month and the day within the month. This is the date, within the current calendar year, that is to be used to determine whether or not this Job Network is to be run at all, and, if it is, the job combination to be run.
DATE=(YYMMDD,yymmdd) specifies an explicit date as six decimal digits, representing the low-order two digits of the year, the calendar month within that year, and the day within that month. This is the date that is to be used to determine whether or not this Job Network is to be run at all, and, if it is, the job combination to be run.
DATE=(JULIAN,ddd) specifies an explicit date as three decimal digits, representing the julian day number within the current calendar year, of the date that is to be used to determine whether or not this Job Network is to be run at all, and, if it is, the job combination to be run.

7.12 SPANEX Utility - POST Command

The POST command (no abbreviation) is used to signal the completion of an event for a specific job in a SPANEX job network. Each job in a SPANEX network may have up to 8 events specified which must be complete before the job will be executed. These events are assigned arbitrary numbers 1-8, and the meaning of each event may be assigned by the user. A job may be set into "hold" status for one of its events by the use of the HOLD command, or it may be defined with outstanding events on the SPXJOB or QUICKJOB macro when the RCM is generated. An event is set complete by means of this command, or via an application program call to the SPANEX Utility. The POST command is not accepted after the job has begun execution. The capability to use this command may be specifically permitted for time-sharing users for a particular RCM by means of the "TSNET=YES" parameter of the SPXRCM or QUICKJOB macro.

syntax:

```
        [NET=rcmname]   [,JOB=jobname]
POST    [            ]   [            ]   ,EVENT=n
        [RCM=rcmname]   [,J=jobname  ]

              [,PASSWORD=password]
              [                  ]
              [,PWORD=password   ]
```

where:

NET= - specifies the name of the RCM (or Network) which contains the job whose event is to be set complete. The "JOBNET=YES" option must have been specified on the SPXRCM macro when the RCM generation process was performed (this is the default if the QUICKNET macro was used). This parameter is required, unless the "NET=" or "RCM=" operand was specified in the SPANEX EXEC statement PARM or in a SPANEX VDU "SET" command, in which case it will not be accepted and the EXEC PARM network name will be used. The "NET=" and the "RCM=" parameters are synonymous.

JOB= - specifies the name of the job whose Event is now complete. This parameter is required. The abbreviation "J=" is accepted for the "JOB" parameter.

EVENT= - specifies the event identifier (single digit, 1-8) of the event for this job that has now completed. The job will now execute if all events are signalled complete and if all prerequisite jobs have also completed successfully.

PASSWORD= - specifies the password (maximum 8 characters) for network commands for this network. If the requirement for a password was defined when the RCM was generated, then this operand is

required, and the command will be rejected if the password is omitted or incorrect.

7.13 SPANEX Utility - PRINT Command

The PRINT command (abbreviation "P") is used to print (on the SPANEX Message Log - the SPXPRINT DD statement) the SPANEX Restart Status of one or all jobs in a given suite or network (as defined by one RCM). Note that the output from this command appears on the SPANEX Message Log only, even if the command was entered from an MCS console.

syntax:

```
[PRINT]   [NET=rcmname]
[     ]   [           ]   [,JOB=jobname]   [,PREQS]
[P    ]   [RCM=rcmname]
```

where:

NET= - specifies the name of the RCM (or Network) which contains the specification of the job or jobs whose status is to be printed on the SPANEX Log. This parameter is required, unless the "NET=" or "RCM=" operand was specified in the SPANEX EXEC statement PARM or in a SPANEX VDU "SET" command, in which case it will not be accepted and the EXEC PARM network name will be used. The "NET=" and the "RCM=" parameters are synonymous.

JOB= - specifies the name of a single job whose Restart Status is to be printed. If this parameter is omitted the Restart Status of all jobs defined in this RCM will be printed. The abbreviation "J=" is accepted for the "JOB" parameter.

PREQS - specifies that all pre- and post-requisite jobs, and any outstanding "Hold Events", are to be listed for each job whose status is printed.

7.14 SPANEX Utility - PROCEED Command

The PROCEED command (abbreviation "PROC") is used to resume the execution of a SPANEX Job Network that was previously stopped by means of the HALT command. The capability to use this command may be specifically permitted for time-sharing users for a particular RCM by means of the "TSNET=YES" parameter of the SPXRCM or QUICKNET macro.

syntax:

```
[PROCEED]   [NET=rcmname]   [,PASSWORD=password]
[       ]   [           ]   [                   ]
[PROC   ]   [RCM=rcmname]   [,PASSW=password    ]
```

where:

NET= - specifies the name of the RCM (or Network) whose processing is to be restarted. The "JOBNET=YES" option must have been specified on the SPXRCM macro when the RCM generation process was performed (this is the default if the QUICKNET macro was used). This parameter is required, unless the "NET=" or "RCM=" operand was specified in the SPANEX EXEC statement PARM or in a SPANEX VDU "SET" command, in which case it will not be accepted and the EXEC PARM network name will be used. The "NET=" and the "RCM=" parameters are synonymous.

PASSWORD= - specifies the password (maximum 8 characters) for network commands for this network. If the requirement for a password was defined when the RCM was generated, then this operand is required, and the command will be rejected if the password is omitted or incorrect.

7.15 SPANEX Utility - SCHEDULE Command

The SCHEDULE command (abbreviation "SCH") is used to force the execution of a particular job that is part of a SPANEX job network. The SCHEDULE command should be used if it is necessary to override the job dependencies defined in the RCM, or if it is required to re-submit a job that failed to complete successfully at an earlier execution. The SCHEDULE command with the FORCE option may be used to cause the execution of an "EXCLUDEd" job, although the INCLUDE command should normally be used for his purpose. The capability to use this command may be specifically permitted for time-sharing users for a particular RCM by means of the "TSNET=YES" parameter of the SPXRCM or QUICKNET macro.

syntax:

```
[SCHEDULE] [NET=rcmname]              [,PASSWORD=password]
[         ] [           ],JOB=jobname [                   ]
[SCH      ] [RCM=rcmname]             [,PASSW=password    ]

                         [,FORCE]
```

where:

NET= - specifies the name of the RCM (or Network) which contains the definition of the job network containing the specification of the job that is to be SCHEDULEd. The "JOBNET=YES" option must have been specified on the SPXRCM macro when the RCM generation process was performed (this is the default if the QUICKNET macro was used). This parameter is required, unless the "NET=" or "RCM=" operand was specified in the SPANEX EXEC statement PARM or in a SPANEX VDU "SET" command, in which case it will not be accepted and the EXEC PARM network name will be used. The "NET=" and the "RCM=" parameters are synonymous.

JOB= - specifies the name of the job which is to be SCHEDULEd for execution or for re-execution. The SPANEX Job Process Count (printed by the PRINT command) for this job will be incremented by 1. This parameter is required. The abbreviation "J=" is accepted for the "JOB" parameter.

PASSWORD= - specifies the password (maximum 8 characters) for network commands for this network. If the requirement for a password was defined when the RCM was generated, then this operand is required, and the command will be rejected if the password is omitted or incorrect.

FORCE	-	the “FORCE” option specifies that this job is to be scheduled even though it has already completed successfully according to its SPANEX status. The “FORCE” option must also be used if the job to be scheduled is EXCLUDEd. The “FORCE” option will be ignored if the job is not EXCLUDEd or has not already completed successfully. The “FORCE” option causes the loss of all previous status information for the job so that a clean start of the job will occur.

7.16 SPANEX Utility - SHOW Command

The SHOW command (no abbreviation) is used to display access statistics for the SPANEX Catalog datasets. This is particularly of use in an online SPANEX Utility session, where Catalog performance is being evaluated. The same statistics may be obtained in batch jobs or batch utility executions by the use of SPANEX OPT=L (see the SPANEX General Usage Manual for details.

syntax:

```
SHOW  STATS  [,NET=rcmname]
```

where:

NET=	-	specifies the name of an RCM (or Network). This parameter is not required for the SHOW command, but is permitted for compatibility with other SPANEX commands and to facilitate the use of the SHOW command from a SPANEX terminal session where the NET= parameter may be automatically added to user input.
STATS	-	specifies that SPANEX Catalog access statistics for this session are to be displayed. No action will be performed by the SHOW command if the STATS parameter is omitted.

7.17 SPANEX Utility - STATUS Command

The STATUS command (abbreviation "ST") is used to display or modify the execution status of a SPANEX job network. The STATUS command should be used if it is necessary to change or delete the execution status of a SPANEX Network so that an execution that is incomplete may be abandoned. SPANEX will normally disallow a new execution of a Network whose previous execution has not completed. The STATUS command with neither the "DELETE" or the "COMPLETE" operand merely displays information concerning the state of this Network, and does not require the use of the PASSWORD parameter. As a safety measure, both the "DELETE" and the "COMPLETE" operands require that there be no active jobs in the network (ie each job status in the SPANEX Catalog must be "complete" or must have been deleted) before they will function; if a run of a network is to be abandoned part-way through by the use of the STATUS command with one of these options, the DELETE command may first be used to delete the status for all jobs in the network (eg use the "JOB=/ALL" option of the DELETE command). The capability to use the STATUS command may be specifically permitted for time-sharing users for a particular RCM by means of the "TSNET=YES" parameter of the SPXRCM or QUICKNET macro.

syntax:

```
          [NET=rcmname]
[STATUS]  [           ]   [,DELETE   ]   [,PASSWORD=password]
[      ]  [RCM=rcmname]   [          ]   [                   ]
[ST    ]  [           ]   [,COMPLETE]   [,PASSW=password    ]
          [NET=/ALL    ]
```

where:

NET= - specifies the name of the RCM (or Network) whose status is to be examined or modified. The "JOBNET=YES" option must have been specified on the SPXRCM macro when the RCM generation process was performed (this is the default if the QUICKNET macro was used). This parameter is required, unless the "NET=" or "RCM=" operand was specified in the SPANEX EXEC statement PARM or in a SPANEX VDU "SET" command, in which case it will not be accepted and the EXEC PARM network name will be used. The special value "NET=/ALL" is provided, for the display option of the STATUS command only, so that the status of all defined networks can be examined with a single command; this function is supported for all SPANEX Catalog dataset types except CVOL System Catalogs (see the SPANEX Installation and Maintenance manual for details). The "NET=" and the "RCM=" parameters are synonymous.

DELETE - specifies that the status of this Network is to be deleted from the SPANEX Catalog. This will cause the loss of all status for this network, and of all historical information concerning execution times, and should be used only if the network is

no longer run or if it is desirable that all previous run statistics should be lost. The DELETE option will not be permitted if there is any active job status in the SPANEX Catalog for jobs in this Network. If a PASSWORD is defined for this RCM it will be required for the use of this option.

COMPLETE - specifies that the status of this Network is to be set as complete. This will permit a new execution of this Network to be begun without the loss of status for this network. The COMPLETE option will not be permitted if there is any active job status in the SPANEX Catalog for jobs in this Network. If a PASSWORD is defined for this RCM it will be required for the use of this option.

PASSWORD= - specifies the password (maximum 8 characters) for network commands for this network. If the requirement for a password was defined when the RCM was generated, and an update-type STATUS command is being executed, then this operand is required, and the command will be rejected if the password is omitted or incorrect.

7.18 SPANEX Utility - TRACE Command

The TRACE command (abbreviation "TRA") is used to scan a SPANEX Global Log dataset and display records found there. The Global Log may be scanned for all occurrences of a particular Job, for messages within a certain time range, for messages containing a certain string of characters (eg a message identifier), or all of these in combination. If no search parameters are provided, the entire contents of the Global Log dataset are displayed. Messages retrieved from the Global Log are displayed with a "TRACE==>" prefix to distinguish them from real-time messages.

syntax:

```
[TRACE]   [NET=rcmname]
[     ]   [           ]   [,JOB=jobname]
[TRA  ]   [RCM=rcmname]

            [,STRING='search string']
            [                       ]
            [,S='search string'     ]

            [,START=hh:mm]   [,END=hh:mm  ]
            [            ]   [            ]
            [,STIME=hh:mm]   [,ETIME=hh:mm]
```

where:

NET= - specifies the name of the RCM (or Network) whose Global Log is to be scanned. This parameter is required, unless the "NET=" or "RCM=" operand was specified in the SPANEX EXEC statement PARM or in a SPANEX VDU "SET" command, in which case it will not be accepted and the EXEC PARM network name will be used. The "NET=" and the "RCM=" parameters are synonymous.

JOB= - specifies the name of a job which is to be searched for in the Global Log. All messages issued by or referring to this job will be retrieved. This parameter is optional and may be combined with the "STRING=" and time parameters. The abbreviation "J=" is accepted for the "JOB" parameter.

STRING= - specifies any string of characters (enclosed in single quotes if it contains blanks or special characters) which is to be searched for in the Global Log. A string of up to 60 characters may be specified. Searching begins at the start of message text in the Global Log records, and does not include record prefix information such as RCM name, date and time. This parameter is optional and may be combined with the "JOB=" and time parameters. The abbreviation "S=" is accepted for the "STRING" parameter.

START= - the “START” parameter specifies a time (in hours and minutes) to begin searching in the Global Log dataset. This parameter must be specified as two digits of hours and two digits of minutes, with a colon (:) or a full stop (period) in between. This parameter is optional and may be combined with the “JOB=”, “STRING=” and “END=” parameters. The alternative “STIME=” parameter is synonymous with the “START=” parameter.

END= - the “END” parameter specifies a time (in hours and minutes) to end searching in the Global Log dataset. This parameter must be specified as two digits of hours and two digits of minutes, with a colon (:) or a full stop (period) in between. SPANEX does not begin searching for the end time value until the start time value (if specified) has been found, so that time ranges spanning midnight may validly be specified for the TRACE command. This parameter is optional and may be combined with the “JOB=”, “STRING=” and “START=” parameters. The alternative “ETIME=” parameter is synonymous with the “END=” parameter.

7.19 SPANEX Utility - UPDATE Command

The UPDATE command (no abbreviation) is used to update (or add to the SPANEX Catalog) the SPANEX Restart Status for a particular job. This is used to overcome any unpredictable problems associated with the processing of a job, or to recover from errors in the definition of the RCM or in the JCL for the job, or to enable the portability of work to a different machine or computer centre without losing the results of partial processing of jobs. The UPDATE command is also useful for the testing of job restart logic, as error conditions can be simulated by the creation of any desired job error status. The capability to use this command may be disallowed altogether for a particular RCM by means of the "OPTU=NO" parameter of the SPXRCM or QUICKNET macro, or may be specifically permitted for time-sharing users for a particular RCM by means of the "TSUPD=YES" parameter of the SPXRCM or QUICKNET macro.

syntax:

```
          [NET=rcmname]
UPDATE    [            ]  ,JOB=jobname  ,STEP=stepname
          [RCM=rcmname]

                          [,PROCSTEP=procstepname]

                                        [         [INIT]]
                                        [         [PREX]]
                                        [         [STST]]
                                        [         [ABNU]]
                                        [         [ABNS]]
                                        [         [ABNX]]
                          [        0 ]  [         [ABPF]]
                          [,CODE=    ]  [,STATUS=[ABRC]]
                          [      nnn]  [         [RSTR]]
                                        [         [UCHK]]
                                        [         [URST]]
                                        [         [USRR]]
                                        [         [STEN]]
                                        [         [SUCC]]
                                        [         [BLNK]]
```

where:

NET= - specifies the name of the RCM (or Network) which contains the specification of the job whose Restart Status is to be updated or added to the SPANEX Catalog. This parameter is required, unless the "NET=" or "RCM=" operand was specified in the SPANEX EXEC statement PARM or in a SPANEX VDU "SET" command, in which case it will not be accepted and the EXEC PARM network name will be used. The "NET=" and the "RCM=" parameters are synonymous.

JOB= - specifies the jobname of the job whose status is to be updated or added to the SPANEX Catalog. This parameter is required. The abbreviation "J=" is accepted for the "JOB" parameter.

STEP= - specifies the stepname that is to be set for the job specified by the "JOB=" parameter. Depending upon the values specified for the "CODE=" and "STATUS=" parameters, this stepname will be used by SPANEX to restart this job according to the definitions in the RCM. The "STEP=" parameter is required.

PROCSTEP= - specifies the procedure stepname for a job where the stepname specified by the "STEP=" parameter is not unique. This parameter is used as a further qualification of the "STEP=" parameter. If specified, the procedure stepname must be one of the procedure stepnames defined in the RCM for this job and must match the "STEP=" parameter. If the job is defined with duplicate stepnames, and the "PROCSTEP=" parameter of the "UPDATE" command is omitted, then the step set into the job's restart status will be the first in the RCM for which the "STEP=" parameter matches.

CODE= - specifies the numeric value to be set for the "last known code" field in the SPANEX Catalog entry for this job and step. Correct specification of this value in combination with the specification of the "STATUS=" parameter will permit the automatic SPANEX restart of this job at any desired point. Careful consideration of the RCM definition should be given in order to arrive at the value required for this operand. The "CODE=" parameter is optional and has a default value of zero.

STATUS= - specifies the character value to be set for the "status" field in the SPANEX Catalog entry for this job and step. Correct specification of this value in combination with the specification of the "CODE=" parameter will permit the automatic SPANEX restart of this job at any desired point. Careful consideration of the RCM definition should be given in order to arrive at the value required for this operand. The "STATUS=" parameter is optional and has a default value of "BLNK" (which represents a blank status field). The values which are acceptable for the "STATUS=" parameter are shown in the syntax table above, and each corresponds to one of the possible values in the RSBSTAT field of the Restart Status Block (see Section 6.1 on page 59 of this manual). Appendix A of this manual also gives an explanation of the values of this field.

8 The SPANEX RCM Map Feature

8.1 RCM Map Introduction

The SPANEX RCM Map feature of SPANEX offers several facilities to increase the usability and flexibility of a SPANEX job networking environment.

The function of the feature is to present, in a clearer form, information obtained from the user-defined data in the RCM for each SPANEX job network and also from the execution-time data maintained by SPANEX as processing progresses. This information is presented in various ways according to options specified by the user when invoking the RCM mapping function, and includes such forms as a full-screen display of real-time data, an equivalent real-time display on a hard-copy device, an analysis of options selected at generation time for the RCM, and a "Wall Chart" showing the relationships between jobs in the network (available in two sizes).

Much of the data displayed by SPANEX RCM Map is not available to user or operator in response to other SPANEX enquiry commands, and this feature will be invaluable to the function of monitoring and controlling the execution of all SPANEX job networks.

8.2 RCM Map Reports

The SPANEX RCM Map feature offers a number of reports based upon both user-defined and SPANEX-maintained data. Reports are network-orientated - separate reports are generated for separate SPANEX job networks. Basic report types available are:

Status Report	a report of the status of each job in a SPANEX job network, with such information as pre- and post-requisite jobs, current execution jobstep status, date and time, and comments indicating any other relevant information. This report is produced by default by the MAP command, and can be explicitly requested in combination with other reports by use of the "STATUS" option of the MAP command PARM field. This report is supplied on both hard-copy (via an optional output DD statement or via spin-off for MVS) and as a full-screen display on time-sharing and dedicated SPANEX 3270 terminals.
Options Report	a report based upon data obtained from the RCM (data originally defined by the user), such as options selected for particular jobs, for the entire network, or for groups of jobs. This is the only way of obtaining this information for a network if the generation-time listing is not available. This report is produced in response to the "OPTS" option of the MAP command PARM field, with additional Calendar data provided if the "CALENDAR" option is also specified. This report is supplied on both hard-copy (via an optional output DD statement or via spin-off for MVS) and as a full-screen display on time-sharing and dedicated SPANEX 3270 terminals.
SPANEX Wallchart	a pictorial representation of the relationships between all jobs in a SPANEX network, with each job represented by a box on the chart. There are two sizes of wall-chart available allowing small or large networks to be accommodated on a chart to fit most office walls. Because of the limitations of line-printer carriages, it is necessary for a small amount of completion work to be performed by hand on these charts, to allow for widths greater than one page, for example, and to connect job boxes with lines to indicate dependencies - full instructions are printed by the RCM Map program to enable this completion work to be performed by staff at any level. A Wallchart is produced in response to the "WALL" option of the MAP command PARM field.
SPANEX Calendars	reports on the contents of SPANEX Calendar modules. Either a list of the day-types in a Calendar module can be obtained, or a calendar display of an individual day-type or of all day-types. This report is produced in response to the "TABLE=" option of the MAP command PARM field.

8.3 RCM Map Implementation

The RCM Map facility is implemented via the "MAP" command of the SPANEX Utility, described in Section 7.10 on page 100 of this manual.

Additional Job Control Language may be required (non-MVS users only) in order to obtain hard-copy output of RCM Map reports. Where required, this takes the form of two additional optional DD statements which should be provided if required in the jobstep which executes the SPANEX Utility (for TSO users these may be allocated by means of the "ALLOC" TSO command). DD statements are:

RCMAPOUT — required for hard-copy output of "Status" and "Options" reports, always required for Wall Chart. For the SPANEX Wall Chart this DD statement is used for manual chart completion instructions. Not required for MVS if the "SPIN=sysoutclass" parameter option is specified.

WALLCHRT — always required for Wall Chart. This DD statement is used for the output of the actual chart - note that output may be printed in 130-byte vertical strips which must be joined together side-by-side to form the complete chart; instructions for this will be printed on the RCMAPOUT DD statement if appropriate. Not required for MVS if the "SPIN=sysoutclass" parameter option is specified.

Both these DD statements refer to print-format datasets, and RCM Map will set the DCB attributes as required for 132-byte output lines and ASA print control characters.

8.4 Explanation of Report Output from RCM Map

8.4.1 Status Report

The RCM Map Status report, whether displayed as hard-copy output or on a time-sharing or SPANEX 3270 terminal, contains a number of columns of data. Page headings identify the RCM (or network) being mapped, and contain a page number which is initialized to 1 for each new display.

At the end of the Status report, a list is given of the jobs in the network for which processing is outstanding before network execution can be considered complete. This will include jobs that are currently executing or which have failed, and jobs which have not yet been reached in the processing of the network. The size of the Status report and output destinations can be modified by the use of the "ABBR" and "NOVDU" parameter options, respectively.

The meanings of the various columns of data are as described below:

JOBNAME	this column contains the name of the job within the network defined by this RCM that is described by this line of output.
PRE-REQ JOBS	this column (which may occupy more than one physical line for a logical output line) contains the names of all jobs defined in this RCM that are pre-requisites of the job in the "JOBNAME" field.
POST-REQ JOBS	this column (which may occupy more than one physical line for a logical output line) contains the names of all jobs defined in this RCM that are post-requisites of the job in the "JOBNAME" field.
CURRENT STEP	this column contains the name of either the SPANEX jobstep that is currently executing or the step that was executing when this job was last active in the system (eg if the job has Abended, this will be the name of the failing step).
STATUS	this column contains the current or most recent SPANEX status of the job (see Appendix A in this manual for valid values of SPANEX Status); this will describe the last event in the execution of this job that was monitored by SPANEX.
CODE	this column contains the current or most recent SPANEX code for this job (eg if the job has Abended as shown by the STATUS value, this field will contain the Abend code). If there is no valid code (eg if the job is performing normally), this field may contain the word "UNKNOWN".
TIME	this column contains the time at which this status for the job was recorded by SPANEX.

DATE — this column contains the date on which this status for the job was recorded by SPANEX.

COMMENTS — this column contains any comments which SPANEX may make to clarify the status of this job based upon other information SPANEX may have available. Note that more than one of these conditions may be true at the same time - in this case SPANEX has a hierarchy of conditions and displays only the most important comment; possible values are:

- COMPLETED — this job has successfully completed processing; post-requisite jobs in the network (if any) have been scheduled.
- EXCLUDED — this job is EXCLUDEd from the network for the current or pending network run.
- EXCL(NEXT) — this job will be EXCLUDEd from the network for the next NETSTART to be issued.
- INCL(NEXT) — this job will be INCLUDEd into the network for the next NETSTART to be issued (cancelling an EXCLUDE command or the PROCESS=EXCLUDE option).
- NOT-TODAY — this Job is EXCLUDEd from the current or pending network run because of SPANEX Calendar definitions.
- I/O ERROR — an input/output error occurred on the SPANEX catalog - notify your SPANEX systems programmer.
- MULTI-X-WT — this job has successfully completed, but is defined with the Multiple-Execution SPANEX job process option and is currently awaiting manual re-scheduling of a further execution.
- H=nnnnnnnn — this job is in HOLD status for the outstanding events numbered as indicated by the “nnnnnnnn” values. The job is awaiting execution of the POST command for each of these events.
- SCHEDULING — this job has successfully completed, and is currently in the process of checking and scheduling post-requisite jobs (if any).
- MUT EXCL — this job would have been submitted for execution, but is held because another job is running that is defined as mutually exclusive with this job.
- J-HALTING — a HALT command with the “EOJ” (end-of-job) option has been issued for this network. This job is executing but will enter HALT status at the end of the job.
- S-HALTING — a HALT command with the “EOS” (end-of-step) option has been issued for this network. This job is executing but will enter HALT status at the end of the current or next SPANEX step.

	HALTED	a HALT command has been issued for this network, and this job has entered HALT status. The job will be re-submitted when a PROCEED SPANEX Utility command is issued.
	NO STATUS	no information at all is currently held about this job (normal if the network is processing but this job has not yet been reached; or the network may have completed and the "ERASEON=YES" option is in effect).

PROCESS COUNT — this column indicates the current Process Count value for the job in the "JOBNAME" field. The Process Count is typically the number of times an execution of the job has been tried for this network execution. This field will be displayed only on "wide-screen" 3270 terminals and on printed MAP command Status Reports.

PROCSTEP NAME — this column contains the procedure stepname of either the SPANEX jobstep that is currently executing or the step that was executing when this job was last active in the system (eg if the job has Abended, this will be the procedure stepname of the failing step). The procedure stepname will be blanks if the step is not within a JCL procedure.

AVG.EXEC (MINS) — this column indicates the average execution time of this job in minutes. This value is the average of all the executions of this job observed by SPANEX, not including the current execution.

FLAG VALUES — this column indicates the hexadecimal values of three bytes of flags that SPANEX maintains for each job. The meanings of these various flags are shown below. This field will be displayed only on "wide-screen" 3270 terminals and on printed MAP command Status Reports.

800000	SPANEX restart AT a step
400000	SPANEX restart AFTER a step
C00000	SPANEX restart BEFORE a step
200000	Job submitted by SCHEDULE command or by automatic function
100000	Job submitted by NETSTART command
080000	Job EXCLUDEd, Network not active
040000	Job EXCLUDEd, NETSTART complete
020000	Job complete, now performing automatic scheduling of dependents
010000	Job submitted because pre-req job was cancelled by means of the #SPXRSTU macro

008000	Job completed but may be re-run, post-reqs are suspended
004000	End-of-Step HALT in progress for this job
002000	End-of-Job HALT in progress for this job
001000	Job HALTed, awaiting PROCEED command
000800	Last Run of Multiple-Execution job
000400	NOT Last Run of Multiple-Execution job
000200	Job not eligible for scheduling
000100	Job eligible for scheduling when HOLD events are completed
000080	Job delayed because it is mutually exclusive with an active job
000040	Job is to be EXCLUDEd from the NEXT network execution
000020	Job is to be INCLUDEd into the NEXT network execution
000010	Job has extended Catalog data (includes job execution statistics)
000008	Job EXCLUDEd by Calendar processing

8.4.2 RCM MAP Options Report

The RCM Map Options report, whether displayed as hard-copy output or on a time-sharing or SPANEX 3270 terminal, has the same data format. One line of output is presented for each RCM definition option (options selected during the RCM generation process), and lists are provided of jobs which conform to certain classification criteria (eg jobs which are defined as "non-excludable", or jobs which are defined with one or more of the SPANEX job process options).

Information provided includes:

- RCM Assembly (generation) time and date.
- SPANEX release level at which RCM generation was performed (and, by implication, SPANEX features that may be available to jobs defined in this RCM).
- "Confirm" option (at RCM level) for operator intervention in the event of restart of jobs defined in this RCM.
- Restart user exit routine name (at RCM level), if any was specified for this RCM.
- Indication if RCM has "USE=TEST" option.
- Indication if RCM has "OPTU=NO" option.
- Indication if RCM describes a job network.
- Name of job submit routine for this job network.
- Indication if UPDATE/DELETE commands are accepted from a time-sharing terminal.
- Indication if network control commands are accepted from a time-sharing terminal.
- End-of-network user exit routine name, if any was specified for this RCM.
- Indication if a network password is required for this RCM.
- Indication if "ERASEON=YES" option is in effect for this RCM.
- DDNAME defined for SPANEX Utility use of Global Log, if any for this RCM.
- DSNAME of the Global Log dataset, if any for this RCM.
- Indication if the Global Log dataset may be shared by other networks.
- NETSTART user exit routine name, if any was specified for this RCM.

- A list of all jobs in the RCM that have user restart exit routines specified that override the specification (if any) of a global exit routine for the whole RCM.
- A list of all jobs in the RCM that are defined with the "EXCLUDE=NO" option, or an indication that no jobs have this option.
- A list of all jobs in the RCM that are defined with the "PROCESS=DELAY" option, if any.
- A list of all jobs in the RCM that are defined with the "PROCESS=WFI" option, if any.
- A list of all jobs in the RCM that are defined with the "PROCESS=MULT" option, if any.
- A list of all jobs in the RCM that are defined with the "PROCESS=IGNERR" option, if any.
- A list of all jobs in the RCM that are defined with the "PROCESS=RECOVERY" option, if any.
- A list of all jobs in the RCM that are defined with the "PROCESS=EXCLUDE" option, if any.
- For each job in the RCM that is defined with mutual exclusion relationships, a list is given of all jobs with which execution of each job is mutually exclusive.
- A list of the Calendar run-days defined for the Job Network as a whole.
- A list of the Calendar non-run days defined for the Job Network as a whole.
- If the "CALENDAR" option is included in the PARM field of the MAP command, a report is produced of all individual jobs with Calendar definitions, with their run-days and non-runs days listed.
- If a User Calendar is defined, its module name is given. This module name can be input to the MAP command TABLE= option to list the User Calendar's contents.

The meanings of all these various options will be familiar to those who have been concerned with the generation of SPANEX RCMs. Explanations will be found in Section 5 of this manual with the descriptions of the RCM generation macro parameters.

8.4.3 RCM MAP Wall Chart

The RCM Map Wall Chart is available only as hardcopy output, and may be requested in two sizes (selected by the "SIZE1" and "SIZE2" options of the MAP command). SIZE2 is the larger of the two, SIZE1 is the default if neither SIZE1 nor SIZE2 is specified.

A SPANEX Wall Chart is a two-dimensional pictorial representation of the dependencies between jobs in a SPANEX job network. Each job in the network is represented by a "box" of asterisks on the chart, with the name of the job printed within the box in each case. The size of each box is fixed according to the "SIZEn" parameter, and the distances between boxes on the chart are dynamically calculated by the RCM Map program so as to fit, evenly spaced, on an integral number of sheets of computer print-out paper.

Each box on the chart is marked with a five-character "address", by which that box is identified within chart completion instructions. These box addresses are formed of two alphabetic and three numeric characters, where the two alphabetic characters are assigned in sequence (AA, AB, AC, etc) vertically down the chart, and the numeric characters are assigned in sequence (001, 002, 003, etc) horizontally across each row of the chart. Thus boxes are marked AA001, AA002, etc, on the top row, AB001, AB002, etc, on the second row, and so on. This permits any given box to be found quickly and easily when the chart is being completed.

8.4.3.1 Small-size Wall Chart

Each box on the smaller size wall chart is twelve character positions wide and five lines high. The minimum horizontal spacing between boxes on the same row is three character spaces.

Jobs that are "non-excludable" are marked with the characters "N/X" in the lower right-hand corner of the job box, but other job attributes are not provided on the small-size chart.

8.4.3.2 Large-size Wall Chart

Each box on the larger size wall chart is twenty character positions wide and twelve lines high. The minimum horizontal spacing between boxes on the same row is five character spaces.

The larger size of the "SIZE2" wall chart permits more information to be contained within each box on the chart. SPANEX RCM Map places in each box a list of job attributes or options, if any are specified for that job, such as non-excludability or any defined SPANEX job process options. Unless a job was defined with all available options there will be space in each box for any desired user comments to be written.

8.4.4 RCM MAP Calendar Displays

The RCM Map Calendar display is invoked by use of the "TABLE=" option of the MAP command, and produces two distinct displays. The TABLE= parameter has two sub-parameters, the Calendar module name, and, optionally, the name of a day-type within that Calendar module.

If only the Calendar module name is entered, SPANEX displays a list of the day-types defined within that Calendar. Information given is the day-type name, any alias assigned to the day-type, and the description of the function of the day-type contained in the Calendar.

If a valid day-type name is added as a second sub-parameter, a display of the calendar is given, indicating which days of the year qualify for this day-type. On a 3270 terminal, high-brightness fields are used to show the qualifying day-types. On a printed report, only the qualifying dates are printed.

Entering "/ALL" as the second sub-parameter causes calendar displays for all of the day-types in the Calendar module to be produced. Note that this may produce a considerable amount of output if a large number of day-types is defined.

8.5 Notes on RCM Generation

Although SPANEX will accept jobs defined in an RCM in any order (provided the jobs, taken together, provide a logically consistent view of the network), there are some implied meanings to the relative positions of jobs in the definition. For example, jobs which become eligible for scheduling according to the defined dependency criteria at the same time, are actually scheduled by SPANEX in a priority order which is the same as the order in which the jobs are defined in the RCM. Similarly, when several jobs which are defined as being mutually exclusive with one another become eligible for scheduling at the same time, as the result of another job's successful completion, jobs that are dependent on the ending job are tried for scheduling first, followed by other jobs in the order that they are defined in the RCM.

Where the drawing of SPANEX Wall Charts is concerned, there is another consideration when planning the order of jobs defined in the RCM. As with all SPANEX functions and features, the Wall Chart processor will function correctly regardless of the order of definition of jobs in the RCM. However, when the wall chart is being assembled ready for printing, the order of definition of jobs is used to define the order of occurrence of jobs within horizontal rows.

Jobs are assigned rows on the printed wall chart according to their levels of dependency. All jobs which have no defined pre-requisites appear in the top row of the chart. Jobs appear on the second row of the chart when all their pre-requisites appear on the top row. In general, each job appears in the row on the chart one below the level of its lowest-level pre-requisite job. And when all the candidate jobs for any given row have been assembled, those jobs are assigned horizontal positions in the row in the order in which they are defined in the RCM.

Thus, the order of jobs in the RCM definition is a factor affecting the readability of the Wall Chart and the ease of drawing connecting lines on the chart to show dependencies. The importance of these factors is a matter for the individual designer of the job network, but the functioning of the SPANEX system will not be affected by any rearrangements of the order of job definition that are made.

8.6 SPANEX RCM Map Messages

Message output from the RCM Map program is minimal, as the main function of the product is to produce reports, which themselves contain the majority of the information produced by the program. Normal SPANEX error messages will detail any errors detected by SPANEX in the running of the RCM Map program - these are the normal SPANEX numbered messages and are documented in the SPANEX Messages and Codes manual. Unnumbered messages explaining the failure to produce the requested output are produced by the RCM Map program on the RCMAPOUT file and via WTO/TPUT where appropriate, together with some instructions for completion of Wall Charts, whenever this is necessary. RCM Map messages are detailed below - note that messages are not assigned message numbers within the normal SPANEX numbering scheme.

WALLCHART COMPLETE, CHART IS IN nnn VERTICAL STRIPS

Explanation: A SPANEX Wall Chart has been successfully written to the "WALLCHRT" DD statement (or spun-off if MVS and the "SPIN=sysoutclass" option was used), and is made up of the number of vertical strips of paper stated. If the number of strips is greater than one then the chart must be manually assembled by attaching strips side-by-side to form a wide sheet of paper large enough to contain all the jobs described in the RCM.

TO COMPLETE WALLCHART, THE FOLLOWING nnn LINES MUST BE DRAWN ON THE CHART:

Explanation: Because of the limitations of computer line-printers, diagonal lines to connect SPANEX-drawn boxes on the Wall Chart must be drawn by hand. Following this message is a series of instructions, in an optimized order for speed of chart completion, which details all the lines needed to complete the chart.

CONNECT BOX aannn TO BOX aannn.

Explanation: This message describes one line to be drawn by hand on the SPANEX Wall Chart in order to show a relationship between dependent jobs.

ATTACH VERTICAL STRIPS TOGETHER ACCORDING TO IDENTIFYING CHARACTERS PRINTED DOWN LEFT AND RIGHT SIDES OF STRIPS.

Explanation: A SPANEX Wall Chart has been drawn which is wider than the 130-character width limit, and so has multiple vertical strips which must be joined together to complete the chart. Each strip has a line of characters printed vertically down each side at which there is a join; ie to make a connection, attach together the strip which has the letter "A" all down the right-hand side to the strip which has the letter "A" all down the left-hand side, removing surplus paper to make an invisible join. All join-marking letters are unique so that the chart can be completed correctly in only one way. All vertical strips consist of a whole number of sheets of paper and all are the same length with blanks added where necessary.

*** WALLCHART CANNOT BE DRAWN - RCM CONTAINS JOB-DEPENDENCY LOOP

Explanation: The RCM definition has been performed in such a way that a loop of dependencies exists (eg JobB has a pre-requisite of JobA and is a post-requisite of JobC, but JobC is also a pre-requisite of JobA). Whereas it can conceivably be valid to define a loop of this type, if certain jobs within the loop are always to be EXCLUDEd (so that the network as executed does not form a loop), this may not be recommended. SPANEX can not successfully execute a network that forms a loop at NETSTART time. A Wall Chart can never be drawn for an RCM that describes a loop, because the chart is not sensitive to jobs that may be EXCLUDEd. SPANEX message SPX908I is also issued.

*** WALLCHART CANNOT BE DRAWN - 'WALLCHRT' DD STATEMENT MISSING

Explanation: The WALLCHRT DD statement is always required for Wall Chart production (unless MVS spin-off is being used) - it may be assigned to any valid device type, but this will typically be a printer or SYSOUT device. This message will be issued if the DD statement is missing - MVS users may avoid it by using the “SPIN=sysoutclass” parameter option.

*** WALLCHART CANNOT BE DRAWN - 'RCMAPOUT' DD STATEMENT MISSING

Explanation: The RCMAPOUT DD statement is always required for Wall Chart production (unless MVS spin-off is being used) - it may be assigned to any valid device type, but this will typically be a printer or SYSOUT device. This message will be issued if the DD statement is missing - MVS users may avoid it by using the “SPIN=sysoutclass” parameter option.

*** END OF SPANEX WALLCHART COMPLETION INSTRUCTIONS ***

Explanation: The various messages appearing before this message represent complete instructions for the manual completion of the SPANEX Wall Chart.

8.7 Examples of MAP Command

All invocations of the RCM Map program are performed by means of the "MAP" command of the SPANEX Utility. Execution of the SPANEX Utility commands is fully explained in Section 7 of this manual.

8.7.1 Example 1

```
MAP NET=RCM1
```

The "MAP" command in this example will produce a Status report on the user terminal (if the SPANEX Utility is executed from a time-sharing or SPANEX 3270 terminal) and on hard-copy (if the RCMAPOUT DD statement is supplied - MVS spin-off could also have been used).

8.7.2 Example 2

```
MAP NET=RCM2,PARM='OPTS,WALLCHRT,SIZE2,STATUS'
```

The "MAP" command in this example will produce an Options report on the user terminal (if the SPANEX Utility is executed from a time-sharing or SPANEX 3270 terminal) and on hard-copy (if the RCMAPOUT DD statement is supplied - MVS spin-off could also have been used), a Status report (under the same conditions), and a large-size Wall Chart (if the WALLCHRT DD statement is supplied).

8.7.3 Example 3

```
MAP NET=RCM3,PARM='WALLCHRT,SPIN=A'
```

The "MAP" command in this example will produce a Wall Chart only, and will Dynamically Allocate two SYSOUT datasets, one for the chart itself and one for the completion instructions.

8.7.4 Example 4

```
MAP NET=RCM4,PARM=ABBR
```

The "MAP" command in this example will produce an abbreviated Status report on the user terminal (if the SPANEX Utility is executed from a time-sharing or SPANEX 3270 terminal) and on hard-copy (if the RCMAPOUT DD statement is supplied - MVS spin-off could also have been used). All jobs that are EXCLUDEd from the network, jobs which have not yet been reached in the current execution of the network, and all jobs which have successfully completed already for this execution of the network, are omitted from the report so as to reduce the number of lines of output.

8.7.5 Example 5

```
MAP NET=RCM4,PARM='TABLE=SPXCAL95'
```

The “MAP” command in this example will produce a list of all the Calendar day-types defined in the SPANEX System Calendar for the year 1995. The output will appear on the user terminal (if the SPANEX Utility is executed from a time-sharing or SPANEX 3270 terminal) and on hard-copy (if the RCMAPOUT DD statement is supplied - MVS spin-off could also have been used).

8.7.6 Example 6

```
MAP NET=RCM4,PARM='TABLE=(SPXCAL95,MONDAY)'
```

The “MAP” command in this example will produce a display of the days in the year that are included in the day-type “MONDAY” in the SPANEX System Calendar for the year 1995. The output will appear on the user terminal (if the SPANEX Utility is executed from a time-sharing or SPANEX 3270 terminal) and on hard-copy (if the RCMAPOUT DD statement is supplied - MVS spin-off could also have been used).

9 Defining and Using SPANEX Calendars

9.1 The SPANEX Calendar Feature

The SPANEX Calendar feature allows all processing that is date-dependent to be handled automatically. Each SPANEX Job Network, and each Job within each Network, may optionally have calendar definitions associated with it.

In the case of a Job Network, a NETSTART command issued on a day on which the Network (or application system) does not run will be ignored. On a day on which the Network or application system does run, the NETSTART will be executed as normal, and calendar processing will be applied to any jobs within the Network that have calendar definitions.

Thus it is possible to define on which days of the week (or days of the year, or types of day) a SPANEX Job Network will be run. When the Network is run, the combination or sequence of jobs to be run may also be varied according to calendar specifications. Naturally, as with all SPANEX features, the use of calendars is always optional, at the job or network level: calendar processing need be defined only when it is required. Processing of existing SPANEX Job Networks, or of new networks for which calendars are not required, is not affected at all.

When calendar definitions are created, each Job or Job Network may have two lists of symbolic calendar names associated with it. The first list consists of days or day-types on which the Job or Job Network is to be run. The second list is usually used to define a subset of the days in the first list on which the Job or Job Network does not run. Thus a job could be defined to run every Monday, but not if Monday is a Public Holiday. Since SPANEX allows an unlimited number of calendars to be defined, any combination of the days of the year can be represented.

The use of the calendar definitions occurs during the NETSTART process for the Job Network. The date for which the NETSTART is performed governs the selection of jobs for that run of the Network, so that this combination of jobs is retained until the Network run completes, even if this takes more than one day. There are comprehensive facilities for determining the run date, and for overriding the run date in exceptional circumstances, so that the correct combination of jobs should be selected in all cases.

9.2 System and User Calendars

SPANEX has a System Calendar, which is global to the installation and defines standard days. This includes some pre-defined day-types, such as all the Mondays of the year, and may also have a large number of user-defined day-types. The days represented by the pre-defined day-types may also be modified if desired (see the discussion of Calendar definitions below). A total of 1000 day-types may be defined in the SPANEX System Calendar. The SPANEX System Calendar is represented in a load module in the SPANEX Load Library with a name of the form SPXCALnn, where “nn” is the last two digits of the year.

SPANEX allows calendars to be defined for the years 1989 to 2087 (module names SPXCAL89 to SPXCAL87).

In addition to the System Calendar, SPANEX supports an unlimited number of User Calendars, which are local to a Job Network. A maximum of one User Calendar may be specified for each Job Network, and User Calendars may be shared between many Job Networks, if desired. SPANEX User Calendars are also load modules, residing in either the SPANEX Load Library or in a user library accessed via the TASKLIB DD statement (or via any of the standard user load module search facilities used by SPANEX). A User Calendar module has a name of the form SPXxxxnn, where "nn" is the last two digits of the year, and "xxx" is a unique identifier for each User Calendar. If a User Calendar is required, its identifier is specified during the RCM generation process for the Job Network. A total of 1000 day-types may be defined in each SPANEX User Calendar.

The standard "starter" System Calendar is automatically generated by the SPANEX software installation process. The following section describes how to modify this, how to generate the System Calendar for subsequent years, and how to generate User Calendars.

SPANEX Calendars can be displayed at any time by using the MAP command with the TABLE= option (see section 7.10 on page 100 of this manual).

9.3 Defining a SPANEX Calendar

Defining a SPANEX Calendar is similar to defining a SPANEX RCM. The process consists of assembling, by means of the System Assembler, a series of macro statements that define the characteristics of the calendar, together with the day-types and days that make up the various calendar definitions. Each Calendar definition requires a CALSTART macro, one or more CALNAME macros (each defining the name of a Calendar day-type), each one followed by one or more CALDAY macros, each of these describing a day that belongs to the day-type described by the preceding CALNAME macro. When all the days and day-types have been defined, the Assembler input is terminated by means of a CALEND macro, which delimits the input and causes the Calendar to be built. These SPANEX macros are described in this section of this manual. No knowledge of Assembler language is required in order to perform SPANEX Calendar definition.

Output from the Assembler consists of an object module for the Calendar (if no severe errors were found in the generation) and an Assembler listing which contains messages describing any errors or warnings encountered during the generation, together with a series of printed calendars, which may be retained as documentation of the calendars available in the system. Error and warning messages are documented in the SPANEX Messages and Codes manual.

Notation for Macros in this Section

Square brackets, [ ], denote (1) that a macro parameter is optional: if an entire parameter with its options is enclosed in square brackets, then that parameter is optional; (2) that a range of values is permissible for a given parameter: if a series of possible values for a parameter is shown in a vertical manner, all surrounded by additional square brackets, then choose one from the values shown.

Normal parentheses, ( ), signify that parentheses should appear when the macro is coded, denoting, for example, a list of sub-parameters.

Underlining, ___, denotes default values for parameters.

Coding Conventions

Standard Assembler language coding conventions are used for SPANEX macros:

- Macro names and operands may be placed anywhere on the statement but are conventionally in columns 10 and 20 respectively for SPANEX;
- Continuations are indicated by a non-blank character in column 72 of the continued statement;
- Continuation statements must begin in column 16.

9.4 Sample JCL for User Calendar Definition

```
//CALGEN    PROC  USERCAL=,YEAR=
//CALASM    EXEC  PGM=SPANEX,
//          PARM='ASMA90,4/DECK,NOOBJ,LIST,XREF'
//SYSPRINT  DD SYSOUT=A
//SYSUT1    DD UNIT=SYSDA,SPACE=(CYL,(5,1))
//SYSUT2    DD UNIT=SYSDA,SPACE=(CYL,(5,1))
//SYSUT3    DD UNIT=SYSDA,SPACE=(CYL,(5,1))
//SYSLIB    DD DSN=SPANEX.SOURCE,DISP=SHR
//SYSPUNCH  DD DSN=&&LOAD,DISP=(,PASS,DELETE),
//          SPACE=(TRK,(5,1)),
//          UNIT=SYSDA,
//          DCB=(BLKSIZE=400,LRECL=80,RECFM=FB)
//CALLKED   EXEC  PGM=SPANEX,
//          PARM='IEWL/LIST,XREF,OL'
//SYSPRINT  DD SYSOUT=A
//SYSUT1    DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSLIN    DD DSN=&&LOAD,DISP=(OLD,DELETE)
//SYSLMOD   DD DSN=user.load(SPX&USERCAL&YEAR),DISP=SHR
//          PEND

//USERCAL   EXEC  CALGEN,USERCAL=ABC,YEAR=89
//CALASM.SYSIN DD  *
       .             .     .
       .             .     .
     enter user input (CALSTART, CALNAME, CALDAY, CALEND
          macros) here
       .             .     .
       .             .     .
/*
//
```


9.5 Calendar Definition Macro Statements

9.5.1 The CALSTART Macro

The CALSTART macro must be the first statement in the input to a SPANEX Calendar definition, and it specifies general information about the calendar tables being defined.

format:

```
CALSTART YEAR=nnnn   [,NEWDAY=hhmm]

                     [,DEFAULT=NO]  [,DEBUG=YES]

                     [,USERCAL=xxx]
```

where:

YEAR=
- 4 decimal digits, a valid calendar year
- specifies the year for which this calendar is being generated. SPANEX supports years from 1989 to 2087. Each calendar must be separately generated for each year in which it is to be used.

NEWDAY=
- 4 decimal digits, a valid time of day (hhmm)
- specifies, in local time, the notional start-of-day time to be used. SPANEX does not necessarily process a change of date at midnight. It is common for processing of batch work for a day to continue on into the following day, and it is unusual for processing of a day's work to begin in the early hours of the day. This value specifies the time-of-day at which, by default, a new day is considered to begin. If this parameter is omitted, a new day is considered to begin at midnight. The value for this parameter in the SPANEX System Calendar will be overridden by any value in a User Calendar, which will in turn be overridden by a NEWDAY value specified in the RCM. The processing date for a SPANEX Job Network can also be forced, if necessary, by the use of the DATE= parameter of the NETSTART command.

DEFAULT=
fixed keyword value
- "DEFAULT=NO" specifies that the SPANEX default calendar tables are not to be included in the calendar being generated. This applies only to definitions of the SPANEX System Calendar. See Section 9.6 below for a list of the default calendar tables generated.

DEBUG=
- fixed keyword value
- "DEBUG=YES" specifies that assembler debugging information is to be printed for this calendar. This option should be used only if you believe the calendar definition process is producing an invalid calendar.

USERCAL= 3 characters, user calendar identifier

- specifies that this calendar definition is for a user calendar table, and specifies the identifier for the user calendar being generated. This may be any unique 3-character string, excluding some reserved SPANEX strings (such as “CAL”, “M0x” and “NJS”). An error message will be issued if an attempt is made to use one of the reserved strings.

9.5.2 The CALNAME Macro

The CALNAME macro defines the name of a calendar day-type, and also delimits any previous day-type definition. The day-type name may or may not be unique within this calendar definition; if it is not unique, then it is assumed that the following CALDAY statements represent modifications to an existing day-type definition - this is the way in which SPANEX default day-types can be edited.

format:

```
CALNAME  day-type  [,TITLE='title text']

                   [,ALIAS=aliasname]
```

where:

day-type
- symbolic day-type name, max=8 characters
- specifies the name for this day-type. This is the name referred to by the RUNDAYS= and NONDAYS= parameters in the RCM generation, and allows this specific combination of calendar days to be identified. If this is the first occurrence of this day-type name in this calendar definition, a new day-type table is generated. If this day-type name has been previously defined, then this statement begins editing of the previous definition; a warning message is issued to highlight this case.

TITLE=
- day-type description enclosed in single quotes, max=64 characters
- specifies descriptive text to help with recognition of the purpose of this calendar day-type definition. If this CALNAME macro begins an edit of this day-type, then the value of the TITLE= parameter replaces any existing title description for this day-type. If a quotation mark appears in the title it should be entered as two consecutive quotation marks; if an ampersand (&) appears in the title it should be entered as two consecutive ampersands.

ALIAS=
- alternative symbolic day-type name, max=8 characters
- specifies an alternative name for this day-type. This feature can be used, when editing SPANEX default calendar definitions, to assign a preferred name to a standard day-type. For example, for French speakers, the day-type “MONDAY” could be given an alias of “LUNDI”. If this CALNAME macro begins an edit of this day-type, then the value of the ALIAS= parameter replaces any existing alias name for this day-type.

9.5.3 The CALDAY Macro

The CALDAY macro is used to define, modify or delete a day within a SPANEX calendar day-type. If a new day-type is being defined (see the CALNAME macro description above) then the CALDAY macro adds a calendar day to the list of days constituting this day-type. If an existing day-type is being edited, the CALDAY macro allows a day to be added to the day-type, modified or deleted from the day-type. The day to be defined may be specified in either "mmdd" (month-number and day-number within month) format or "julian" (day-number within year) format.

format:

```
                    [[ADD,    ]]
          CALDAY    [[REMOVE,]]
                    [[DELETE,]]
                    [[CANCEL,]]

                    [MMDD=mmdd ]
                    [JULIAN=nnn]
```

where:

ADD - fixed keyword value
- specifies that this day is to be added to the day-type identified by the preceding CALNAME macro. ADD is the default option.

REMOVE - fixed keyword option
DELETE - specifies that this day is to be deleted from the
CANCEL day-type identified by the preceding CALNAME macro. The three options, REMOVE, DELETE and CANCEL, all have the same effect. These options are accepted only if an existing day-type is being edited.

MMDD= - day number in "mmdd" format, 4 decimal digits
- specifies a day within the calendar year identified by the CALSTART macro, to be added to, or deleted from, the day-type identified by the preceding CALNAME macro. The first two digits must specify a valid month (ie 01 to 12), and the third and fourth digits must specify a valid day within the month (eg "0101" represents the 1st of January, "1225" represents Christmas Day). This parameter will be validated, including taking leap-years into consideration.

JULIAN= - day number within year, 3 decimal digits
- specifies a day within the calendar year identified by the CALSTART macro, to be added to, or deleted from, the day-type identified by the preceding CALNAME macro. The three-digit number must be the julian day number within the year (eg "001" represents the 1st of January). This parameter will be validated, including taking leap-years into consideration. If both the MMDD=

and JULIAN= parameters are specified, the JULIAN= parameter will be ignored.

9.5.4 The CALEND Macro

The CALEND macro is used to terminate the input to the SPANEX Calendar definition process. If there were no errors in the preceding input, the Calendar table will be generated. The CALEND macro must be the last statement in the input.

format:

```
CALEND
```

where:

there are no parameters

9.6 SPANEX Default System Calendar Definitions

If a SPANEX Calendar definition for the System Calendar is performed, without the "DEFAULT=NO" option specified, then a series of standard day-types is generated. These are customized to the calendar year being defined. The list of these day-types is given below. Note that the names of day-types are limited to 8 characters.

Day-type	Title	Description
MONDAY	Every Monday	All Mondays in the year
TUESDAY	Every Tuesday	All Tuesdays in the year
WEDNESDY	Every Wednesday	All Wednesdays in the year
THURSDAY	Every Thursday	All Thursdays in the year
FRIDAY	Every Friday	All Fridays in the year
SATURDAY	Every Saturday	All Saturdays in the year
SUNDAY	Every Sunday	All Sundays in the year
EVERYDAY	Every day of the Year	All days
1STMON	First Monday of each month	First Monday of each calendar month
1STTUE	First Tuesday of each month	First Tuesday of each calendar month
1STWED	First Wednesday of each month	First Wednesday of each calendar month
1STTHU	First Thursday of each month	First Thursday of each calendar month
1STFRI	First Friday of each month	First Friday of each calendar month
1STSAT	First Saturday of each month	First Saturday of each calendar month
1STSUN	First Sunday of each month	First Sunday of each calendar month
2NDMON	Second Monday of each month	Second Monday of each calendar month
2NDTUE	Second Tuesday of each month	Second Tuesday of each calendar month
2NDWED	Second Wednesday of each month	Second Wednesday of each calendar month
2NDTHU	Second Thursday of each month	Second Thursday of each calendar month
2NDFRI	Second Friday of each month	Second Friday of each calendar month
2NDSAT	Second Saturday of each month	Second Saturday of each calendar month
2NDSUN	Second Sunday of each month	Second Sunday of each calendar month
3RDMON	Third Monday of each month	Third Monday of each calendar month
3RDTUE	Third Tuesday of each month	Third Tuesday of each calendar month
3RDWED	Third Wednesday of each month	Third Wednesday of each calendar month

Day-type	Title	Description
3RDTHU	Third Thursday of each month	Third Thursday of each calendar month
3RDFRI	Third Friday of each month	Third Friday of each calendar month
3RDSAT	Third Saturday of each month	Third Saturday of each calendar month
3RDSUN	Third Sunday of each month	Third Sunday of each calendar month
4THMON	Fourth Monday of each month	Fourth Monday of each calendar month
4THTUE	Fourth Tuesday of each month	Fourth Tuesday of each calendar month
4THWED	Fourth Wednesday of each month	Fourth Wednesday of each calendar month
4THTHU	Fourth Thursday of each month	Fourth Thursday of each calendar month
4THFRI	Fourth Friday of each month	Fourth Friday of each calendar month
4THSAT	Fourth Saturday of each month	Fourth Saturday of each calendar month
4THSUN	Fourth Sunday of each month	Fourth Sunday of each calendar month
MONTOFRI	Weekdays (Monday to Friday inclusive)	All the working days of the year (Monday to Friday inclusive)
MONTOSAT	Weekdays (Monday to Saturday inclusive)	All the working days of the year (Monday to Saturday inclusive)
1STDAY	First day of each month	Day number 1 of each month
2NDDAY	Second day of each month	Day number 2 of each month
3RDDAY	Third day of each month	Day number 3 of each month
4THDAY	Fourth day of each month	Day number 4 of each month
5THDAY	Fifth day of each month	Day number 5 of each month
6THDAY	Sixth day of each month	Day number 6 of each month
7THDAY	Seventh day of each month	Day number 7 of each month
8THDAY	Eighth day of each month	Day number 8 of each month
9THDAY	Ninth day of each month	Day number 9 of each month
10THDAY	Tenth day of each month	Day number 10 of each month
11THDAY	Eleventh day of each month	Day number 11 of each month
12THDAY	Twelfth day of each month	Day number 12 of each month
13THDAY	Thirteenth day of each month	Day number 13 of each month
14THDAY	Fourteenth day of each month	Day number 14 of each month
15THDAY	Fifteenth day of each month	Day number 15 of each month
16THDAY	Sixteenth day of each month	Day number 16 of each month

Day-type	Title	Description
17THDAY	Seventeenth day of each month	Day number 17 of each month
18THDAY	Eighteenth day of each month	Day number 18 of each month
19THDAY	Nineteenth day of each month	Day number 19 of each month
20THDAY	Twentieth day of each month	Day number 20 of each month
21STDAY	Twenty-first day of each month	Day number 21 of each month
22NDDAY	Twenty-second day of each month	Day number 22 of each month
23RDDAY	Twenty-third day of each month	Day number 23 of each month
24THDAY	Twenty-fourth day of each month	Day number 24 of each month
25THDAY	Twenty-fifth day of each month	Day number 25 of each month
26THDAY	Twenty-sixth day of each month	Day number 26 of each month
27THDAY	Twenty-seventh day of each month	Day number 27 of each month
28THDAY	Twenty-eighth day of each month	Day number 28 of each month

9.7 SPANEX Calendar Definition Examples

Example 1

This example shows the input statements for a standard SPANEX System Calendar generation for the year 1999.

```
CALSTART YEAR=1999
CALEND
```

Example 2

This example shows the input statements for a standard SPANEX System Calendar generation for the year 1995, with the addition of a new day-type to include public holidays (these are the public holidays for the UK, and are shown purely for illustrative purposes). This day-type might be used in the NONDAYS= parameter of the SPXRCM statement for the Job Network, to ensure that an application system is not run on Public Holidays.

```
CALSTART YEAR=1995
CALNAME  PUBLHOLS,TITLE='PUBLIC HOLIDAYS'
CALDAY   MMDD=0102
CALDAY   MMDD=0324
CALDAY   MMDD=0327
CALDAY   MMDD=0501
CALDAY   MMDD=0529
CALDAY   MMDD=0828
CALDAY   MMDD=1225
CALDAY   MMDD=1226
CALEND
```


Example 3

This example shows the input statements for a SPANEX User Calendar generation for the year 1995, to define the month-end dates for a particular application. For this example, the month-end date is shown as the last working day (Monday-Friday) of each month, and a pre-month-end day is defined as the previous working day.

```
CALSTART YEAR=1995,USERCAL=ABC
CALNAME  MONTHEND,TITLE='MONTH-END PROCESSING DATES'
CALDAY   MMDD=0131
CALDAY   MMDD=0228
CALDAY   MMDD=0330
CALDAY   MMDD=0430
CALDAY   MMDD=0531
CALDAY   MMDD=0629
CALDAY   MMDD=0731
CALDAY   MMDD=0831
CALDAY   MMDD=0929
CALDAY   MMDD=1031
CALDAY   MMDD=1130
CALDAY   MMDD=1231
CALNAME  PREMEND,TITLE='PRE-MONTH-END PROCESSING
DATES'
CALDAY   MMDD=0130
CALDAY   MMDD=0227
CALDAY   MMDD=0329
CALDAY   MMDD=0427
CALDAY   MMDD=0530
CALDAY   MMDD=0628
CALDAY   MMDD=0730
CALDAY   MMDD=0830
CALDAY   MMDD=0928
CALDAY   MMDD=1030
CALDAY   MMDD=1129
CALDAY   MMDD=1228
CALEND
```


This page intentionally left blank.

10 The SPANEX Quicknet Feature

10.1 Quicknet Introduction

SPANEX Quicknet is an Installation Aid to enable SPANEX Job Networking to be implemented in as short a time as possible. It is an additional feature of SPANEX and in no way alters or removes any previously existing SPANEX facilities.

Quicknet simplifies the definition of job dependencies, and removes the need for any special preparation of user JCL. Quicknet job networks are started and controlled via SPANEX Utility commands in the same way as normal SPANEX networks.

10.1.1 Benefits

- No JCL changes to existing user Jobs.
- Full support for all SPANEX Job Networking facilities, including unlimited job inter-dependency, commands for network tailoring, SPANEX Wall Charts, etc.
- New high-level macros for simplified Network definition.
- Optional scan facility to check for good condition codes even in non-SPANEX jobsteps.
- Support for Automatic Step Restart, even without SPANEX executions in user JCL.
- Can co-exist with full-function SPANEX, even in the same Network.

10.1.2 Restrictions

- Quicknet supports only the OS/390, MVS, MVS/XA and MVS/ESA operating systems.
- User JCL must be held either on a Partitioned Dataset or on a CA-PANVALET or CA-LIBRARIAN library, one job per member, member name is the jobname.
- SPANEX Restart Facility is supported by Quicknet, but without all of the validation of restart points provided by full SPANEX Restart implementations.
- Jobs in a Quicknet network must be submitted only via SPANEX commands (or automatically by SPANEX); SPANEX will not recognize jobs it does not submit.

10.2 SPANEX Quicknet Features and Facilities

SPANEX Quicknet provides rapid and simple implementation of SPANEX Job Networking in an existing application or job suite. It is not intended to replace the standard SPANEX Networking implementation, which can provide more sophisticated facilities.

Despite a few minor limitations when compared with the standard SPANEX Networking implementation (described in the earlier sections of this manual), Quicknet is a viable and very satisfactory means of quickly automating the scheduling and tracking of jobs within an installation. SPANEX Quicknet offers the same tailoring facilities as with standard SPANEX (for instance, customizing an application suite to daily, weekly, special runs).

New Applications

In general, new applications designed around the use of SPANEX Restart and Job Networking control features will have greater scope if the standard SPANEX Networking implementation is used. If the use of Quicknet is required, the procedure is as below.

Existing Applications - How to Convert to SPANEX Using Quicknet

There are two requirements before converting an existing application to SPANEX Quicknet.

1. The execution JCL (not necessarily the JCL Procedures) must reside on a library (PDS, CA-PANVALET or CA-LIBRARIAN) that can be made accessible to SPANEX.

2. The dependencies between the jobs of the application must be known.

How Quicknet Works

Generating a network using the Quicknet macros produces a customized RCM in standard SPANEX format, with one of the supplied Quicknet submit routines being defined as the SPANEX submit routine. This routine obtains the user JCL from the library where it resides, calls an exit routine (SPXQJCL0, which may be user modified) to add SPANEX execution to the job, and passes the job to the operating system (JES2 or JES3) via an Internal Reader. The execution of SPANEX within the user's job is completely transparent to user programs and is of negligible overhead.

For more details, see the Technical Description in Section 10.4 below.

10.3 SPANEX Quicknet Implementation

10.3.1 General Description

This section does not assume that the reader has a full knowledge of the concepts of SPANEX or of how the various functions are implemented.

Simply, SPANEX Job Networking uses a control module known as the RCM to specify job dependencies. If the SPANEX Restart Facility is being used, the restart logic for each job in the suite is also defined here. The RCM is a load module and is created by the user by assembling a set of simple SPANEX macros which define job names, dependencies, and so on.

SPANEX is then executed within the user jobs. SPANEX makes job scheduling decisions by monitoring user program completion codes while examining the definition of the RCM, and acts upon these decisions.

SPANEX Quicknet dynamically adds JCL to user jobs when they are built and submitted by SPANEX. The user therefore does not need to add SPANEX executions to his jobs.

SPANEX utility commands initiate and control SPANEX job networks. These commands will usually be entered from a SPANEX-supported 3270-type terminal. The commands are fully documented in Section 7 of this manual, and are summarized on the SPANEX Reference Card.

The most frequently-used SPANEX utility commands are:

NETSTART, to begin the execution of an application suite;

SCHEDULE, to submit an individual job;

EXCLUDE, to remove a job from one run of the network (eg a weekly job omitted from a daily run);

MAP, to display the status of jobs or to produce a SPANEX Wall Chart of job dependencies.

10.3.2 Network RCM Generation

The generation process for a SPANEX RCM consists of compiling, by means of the system Assembler, a series of SPANEX macro statements which define to SPANEX the jobs and jobsteps which make up the application system. SPANEX Quicknet supplies some additional Quicknet macros to simplify this process, but these may be used in combination with the Standard SPANEX Networking macros SPXJOB, SPXSTEP and SPXRCM.

SPANEX Quicknet macros are:

QUICKJOB, which describes a user job within the network (or job suite);

QUICKNET, which terminates the definition process and specifies general options for the network;

QUICKSTP, which defines a jobstep within a job, and which is used only if SPANEX retrospective condition code checking is to be used.

This section describes only the additional parameters provided by the SPANEX Quicknet macros. The parameters of the standard SPANEX RCM generation macros, a full description of the RCM generation process, and a complete explanation of all SPANEX Networking options are Section 5 of this manual.

10.3.2.1 Notation for Macros in this Section

Square brackets, [ ], denote (1) that a macro parameter is optional: if an entire parameter with its options is enclosed in square brackets, then that parameter is optional; (2) that a range of values is permissible for a given parameter: if a series of possible values for a parameter is shown in a vertical manner, all surrounded by additional square brackets, then choose one from the values shown.

Normal parentheses, ( ), signify that parentheses should appear when the macro is coded, denoting, for example, a list of sub-parameters.

Underlining, ___, denotes default values for parameters.

10.3.2.2 Coding Conventions

Standard Assembler language coding conventions are used for SPANEX macros:

- Labels (ie Jobnames, Stepnames, RCMname) must begin in column 1;
- Macro names and operands may be placed anywhere on the statement but are conventionally in columns 10 and 20 respectively for SPANEX;
- Continuations are indicated by a non-blank character in column 72 of the continued statement;
- Continuation statements must begin in column 16.

10.3.3 QUICKJOB Macro - Define a Job to SPANEX RCM

The QUICKJOB macro should be used to define each job in a suite of one or more jobs that is described by this Quicknet RCM generation. For each job there should be one QUICKJOB macro. The SPXJOB macro may also be used to describe a job within the same RCM as the QUICKJOB macro, if normal SPANEX restart is to be used for that job. There may be no duplicate job names within an RCM generation.

Note that, in addition to the macro parameters described here, all the other options of the SPXJOB macro may also be specified on the QUICKJOB macro.

format:

```
                              [         [0]]
jobname   QUICKJOB            [SCANOPT=[ ]]
                              [         [n]]

                              [,PREREQ=(jjj,jjj,...)]

                              [,MUTEXCL=(jjj,jjj,...)]
```

where:

jobname - JCL jobname, max=8 bytes, required
- specifies the jobname for the job described by this QUICKJOB macro. This jobname must be the jobname that appears in the JCL "JOB" statement for this job, and must be unique in this RCM.

SCANOPT= - single numeric character, default is SCANOPT=0
- specifies an optional scan function to be performed for this job before dependent jobs are scheduled. For the simplest implementation of a SPANEX network, this parameter may be omitted. Specification of a SCANOPT value greater than zero will cause a step to be added to the end of the job at execution time; this step will contain an execution of the SPANEX support routine SPXM0310 (also known as SPXRCCC0), which will examine Operating System control blocks to determine the condition codes returned by user steps. Meanings of the currently supported SCANOPT values are:
SCANOPT=0 - no condition code scan is to be performed. The generated extra step at the end of the user job JCL will execute program IEFBR14 (under the control of SPANEX).
SCANOPT=1 - basic condition code scan is to be performed. The generated extra step at the end of the user job JCL will execute SPANEX routine SPXM0310, and will scan condition codes for all prior steps of the job. If any jobstep (not including the generated dummy SPANEX jobsteps) abended or ended with a condition code greater than zero,

an error will be recognized and dependent jobs will not be scheduled by SPANEX.
SCANOPT=2 - extended condition code scan is to be performed. The generated extra step at the end of the user job JCL will execute SPANEX routine SPXM0310, and will scan condition codes only for those prior steps of the job for which a QUICKSTP macro was specified in the RCM generation. Steps for which there is no QUICKSTP macro, and generated dummy SPANEX jobsteps, will be ignored for checking purposes. If any defined jobstep abended or ended with a condition code greater than the value specified in the "ACCRC=" parameter of the QUICKSTP macro (default zero), an error will be recognized and dependent jobs will not be scheduled by SPANEX.
SCANOPT=3 - extended condition code scan is to be performed. The generated extra step at the end of the user job JCL will execute SPANEX routine SPXM0310, and will scan condition codes for all prior steps of the job (not including the generated dummy SPANEX jobsteps). If any step for which a QUICKSTP macro is supplied abended or ended with a condition code greater than the value specified in the "ACCRC=" parameter (default zero), an error will be recognized and dependent jobs will not be scheduled by SPANEX. All steps for which there is no QUICKSTP macro are checked for a condition code greater than zero, and an error is recognized if one is found.
SCANOPT=4 - extended condition code scan is to be performed. The generated extra step at the end of the user job JCL will execute SPANEX routine SPXM0310, and will scan condition codes for all prior steps of the job (not including the generated dummy SPANEX jobsteps). If any step for which a QUICKSTP macro is supplied ended with a condition code greater than the value specified in the "ACCRC=" parameter (default zero), an error will be recognized and dependent jobs will not be scheduled by SPANEX. When the job is resubmitted, SPANEX Automatic Step Restart will be provided, using restart condition codes as found on QUICKSTP macros for this job. All steps for which there is no QUICKSTP macro are checked for a condition code greater than zero, and an error is recognized if one is found. Automatic Step Restart will not be supplied for steps for which there is no QUICKSTP macro.

PREREQ= - one or more jobnames, each max 8 bytes
- specifies a list of one or more jobs which must have completed successfully before this job is to begin execution. Since multiple jobs may be specified, and the same pre-requisite job may be

specified by multiple QUICKJOB macros, a Job Network of any complexity may be defined in the RCM. If more than one job is specified, enclose the list of jobs, separated by commas, in parentheses. All pre-requisite jobs specified must be defined by QUICKJOB or SPXJOB macros in this RCM generation. If the PREREQ= parameter is omitted, the job defined by this QUICKJOB macro has no pre-requisite jobs, and so will be submitted by the SPANEX NETSTART command for this Network.

MUTEXCL= - one or more jobname symbols, each max 8 bytes
- specifies a list of one or more jobs which may not execute at the same time as this job. Multiple jobs may be specified, and the same mutually-exclusive job may be specified by multiple QUICKJOB macros. When several mutually-exclusive jobs become eligible for scheduling at the same time, the job that is defined first in the RCM will be the one to be scheduled. It is not necessary to define mutual-exclusion relationships in both directions, ie if JobA is mutually exclusive with JobB, it is not necessary also to specify that JobB is mutually exclusive with JobA, although it is not an error to do so. If more than one job is specified, enclose the list of jobs, separated by commas, in parentheses. All jobs specified must be defined by QUICKJOB or SPXJOB macros in this RCM generation.

10.3.4 QUICKNET Macro - Generate a SPANEX Quicknet RCM

The QUICKNET macro should be used to terminate the input (QUICKJOB, and, optionally QUICKSTP, SPXJOB and SPXSTEP macros) to the SPANEX RCM generation process. It should be the last statement in the input to the Assembler.

Note that, in addition to the macro parameters described here, all the other options of the SPXRCM macro may also be specified on the QUICKNET macro. In particular, it may be necessary to define a Global Log dataset, if your installation of SPANEX specifies that Global Logs are mandatory for all networks.

format:

```
                          [         [LIBR]]
 netname  QUICKNET        [LIBRARY=[PDS ]]
                          [         [PANV]]
```

where:

netname - RCM load module name, max=8 bytes, required
- specifies the load module name to be used for this RCM. This is also the name by which this network is referred to in SPANEX commands and report headings, etc, and so is often the name by which the application system is commonly known. This name is included in the generated RCM for checking and debugging purposes.

LIBRARY= - fixed keyword value
- specifies the library type (PDS, CA-PANVALET or CA-LIBRARIAN) that is used to hold the execution JCL for the jobs in this network. The library is assumed to contain one member for each job in the network. The member name of each member is, by default, the same as the jobname in each case, but an alternative member name for each job can optionally be specified on the QUICKJOB macro statement. The default if this parameter is omitted is to assume that the JCL is on a PDS (partitioned dataset).
Note that a DD statement for the user JCL library must be provided for the execution of the SPANEX Utility that issues INCLUDE, NETSTART, SCHEDULE, POST or PROCEED commands for this network.
If "LIBRARY=PDS" is specified or defaulted, the DD statement must have a DDNAME of JOBPDS and must be allocated to a Partitioned Dataset. An optional second DD statement, with a DDNAME of TEMPPDS, may be supplied, and must be allocated to a second Partitioned Dataset, which is to be used for temporary changes or corrections to job JCL. If supplied, the TEMPPDS

dataset is always searched before the JOBPDS dataset for the JCL to be submitted. This allows production JCL to be protected from unauthorized updates whilst still permitting emergency changes.
If "LIBRARY=PANV" is specified, the DD statement must have a DDNAME of PANVALET and must be allocated to a CA-PANVALET library. An optional second DD statement, with a DDNAME of TEMPPANV, may be supplied, and must be allocated to a second CA-PANVALET library, which is to be used for temporary changes or corrections to job JCL. If supplied, the TEMPPANV dataset is always searched before the PANVALET dataset for the JCL to be submitted. This allows production JCL to be protected from unauthorized updates whilst still permitting emergency changes.
If "LIBRARY=LIBR" is specified, the DD statement must have a DDNAME of LIBRMAST and must be allocated to a CA-LIBRARIAN Master. An optional second DD statement, with a DDNAME of LIBRTEMP, may be supplied, and must be allocated to a second CA-LIBRARIAN Master, which is to be used for temporary changes or corrections to job JCL. If supplied, the LIBRTEMP dataset is always searched before the LIBRMAST dataset for the JCL to be submitted. This allows production JCL to be protected from unauthorized updates whilst still permitting emergency changes.

USE= - fixed keyword value
- "USE=TEST" specifies that this RCM is to be used for testing the SPANEX Quicknet facilities, or for the testing of the job or suite of jobs that is defined by this RCM. If the "ACK" SPANEX EXEC statement parameter option was made an installation default when SPANEX was installed this will be negated for jobs run under a "USE=TEST" RCM. If the ACK parameter option is required, it should be specified in the JCL EXEC PARM field. JCL steps generated by SPANEX for this RCM will include a SYSUDUMP DD statement for aid in debugging.

10.3.5 QUICKSTP Macro - Define a Jobstep to Quicknet RCM

The QUICKSTP macro should be used to define each user jobstep in the job defined by the previous QUICKJOB macro in this RCM generation for which condition code checking is required. Steps that are not to be checked need not be defined by QUICKSTP macros. Note that this checking, and the use of the QUICKSTP macro, is valid only if SCANOPT=2, SCANOPT=3 or SCANOPT=4 is specified on the QUICKJOB macro for this job. QUICKSTP macros must appear in the RCM generation input in the order in which the steps occur in the corresponding job JCL. Duplicate step names within a job are supported if there are differing procedure step names, which must be specified via the PROCSTP= parameter of the QUICKSTP macro. There may be no duplicate stepname/procstepname combinations within any one job.

Note that, in addition to the macro parameters described here, all the other options of the SPXSTEP macro may also be specified on the QUICKSTP macro. In particular, the PROCESS=IGNERR option may be used to ignore error condition codes from this step.

format:

```
stepname QUICKSTP  [ACCRC=nnnn]

                   [,CHKEXIT=modname]
```

where:

stepname - JCL stepname, max=8 bytes, required
- specifies the stepname for the jobstep described by this QUICKSTP macro. This stepname must be the stepname that appears in the JCL "EXEC" statement for this step.

ACCRC= - numeric condition code value, max=4095
- specifies an acceptable condition code value for the user program in this jobstep. This value is the highest condition code from this program that does not constitute an error that should cause suspension of dependent jobs. If the ACCRC parameter is omitted, any condition code from this jobstep that is greater than zero will be recognized as an error. If the ACCRC value is specified as 4095, the highest legal condition code from a user program, then all condition code values will be treated as a good end of the user program. Note that the "Acceptable Abend Code" feature, described with the SPXSTEP macro, is not supported by Quicknet. Multiple conditions for acceptable errors can be implemented by means of a user exit (see the description of the CHKEXIT parameter).

CHKEXIT=	-	Load module name, max=8 bytes
	-	specifies the name of a user module that is to be called retrospectively to check the execution of the program that runs in this jobstep. See the discussion of the use of the standard SPANEX User Check Exit, SPXUCHEK, in Section 10.3.6 below. The definition of the interface to user check exit modules is contained in the SPANEX General Usage manual.

10.3.6 QUICKNET JCL Considerations

JCL for SPANEX QUICKNET jobs is exactly the same as for jobs that do not use SPANEX at all. Each job must be stored as a separate member of the JCL Library, and must contain a complete JOB statement, and one or more EXEC statements.

A null JCL statement (a statement containing "//" in columns 1-2 and no other data) is optional as the last statement in a job's JCL member. However, an exception to this is if the standard SPANEX User Check Exit facility (SPXUCHEK) is to be used. This facility is requested by means of the "CHKEXIT=SPXUCHEK" parameter of the QUICKSTP macro, and uses the SPXUCHEK routine to perform Retrospective Condition Code checking.

The SPXUCHEK routine is driven by control statements which are input via the SPXUCHKI DD statement, and, for QUICKNET, this DD statement must be supplied to the last step of the job, which is automatically generated by SPANEX and added to the end of each job as it is submitted. In order to input these control statements to the generated last step of the job, a particular JCL convention is used. A null JCL statement must be included at the end of the job, and the appropriate SPXUCHKI DD statement must follow this in the JCL member. An example of this is shown below. The SPXUCHEK routine is fully documented in the SPANEX General Usage Manual.

```
//jobname JOB (account),'SPANEX QUICKNET',CLASS= ..etc
//*  SAMPLE JCL FOR QUICKNET AND SPXUCHEK
//STEP1    EXEC PGM=userpgm1
//SYSPRINT DD SYSOUT=A
//STEP2    EXEC PGM=userpgm2
//
//SPXUCHKI DD *
CONDITION CODE=4,ACCEPT,STEP=STEP1
CONDITION CODE=8,REJECT,STEP=STEP2
```


10.4 SPANEX Quicknet Technical Description

This section provides a more technical description of the modules that constitute the SPANEX Quicknet feature. It may be of use in problem diagnosis, and for those familiar with SPANEX it enables them to use some of the facilities of Quicknet in their standard SPANEX Networking implementation.

Apart from the additional RCM generation macros, SPANEX Quicknet consists of some SPANEX Job Submit Routines (see Section 6.2 of this manual) and an extra SPANEX module, SPXM0310 or SPXRCCC0, which performs the optional scanning of jobs to examine step condition codes.

The Quicknet submit routines call a common subroutine, SPXQJCL0, to examine each user JCL statement as read from the JCL library, and it is this subroutine which inserts the additional JCL for SPANEX into the jobstream. Although the JCL generated by this routine should be suitable for most users, it may be necessary to customize it to a specific installation. In this case, the SPXQJCL0 source code may be modified to generate differing JCL, or, for example, to add a TASKLIB DD statement to specify the load library where the RCM resides (see discussion on TASKLIB below).

JCL as produced by SPXQJCL0 is defined in three tables within this module, and the source code fully describes where user modifications may be made in order to generate differing JCL. Note that the JCL stepnames used by SPANEX Quicknet are significant, and specific stepnames are used to trigger various options. Reserved stepnames are "@SPXFRSx" and "@SPXLAST", and these are interrogated by SPXM0310 and matched with the Quicknet-generated RCM stepnames throughout SPANEX.

The standard SPXQJCL0 routine handles TASKLIB DD statements in the following way:

(1) if there is a TASKLIB DD statement in the JCL defined within SPXQJCL0 then this is included in the generated JCL and no further action is taken;

(2) if there is no TASKLIB DD statement in the JCL defined within SPXQJCL0, and there is no TASKLIB DD statement in the jobstep or TSO session that is executing SPXQJCL0, then the generated JCL has no TASKLIB DD statement, and the assumption is that the RCM is obtainable either from the Operating System Link List or from a JOBLIB library;

(3) if there is no TASKLIB in the JCL defined within SPXQJCL0, but there is a TASKLIB allocated in the jobstep or TSO session that is executing SPXQJCL0, then the same TASKLIB allocation is included on an additional DD statement that is added to the generated JCL.

Note that if TASKLIB is a concatenation all datasets in the concatenation are propagated to the generated JCL. It is a requirement that all task libraries are catalogued datasets.

The standard SPXQJCL0 routine provides special support for STEPCAT DD statements. These are processed in the following way:

(1) if there is a STEPCAT DD statement in the JCL defined within SPXQJCL0 then this is included in the generated JCL and no further action is taken;

(2) if there is no STEPCAT DD statement in the JCL defined within SPXQJCL0, and there is no STEPCAT DD statement in the jobstep or TSO session that is executing SPXQJCL0, then the generated JCL has no STEPCAT DD statement;

(3) if there is no STEPCAT in the JCL defined within SPXQJCL0, but there is a STEPCAT allocated in the jobstep or TSO session that is executing SPXQJCL0, then the same STEPCAT allocation is included on an additional DD statement that is added to the generated JCL.

Note that if STEPCAT is a concatenation, only the first dataset in the concatenation is propagated to the generated JCL.

The standard SPXQJCL0 routine also provides special support for SPXCAT1 and SPXCAT2 DD statements. These statements allow local SPANEX KSDS Catalogs to be implemented. These are processed in the following way:

(1) if there is a SPXCAT1 and/or SPXCAT2 DD statement in the JCL defined within SPXQJCL0 then these are included in the generated JCL and no further action is taken;

(2) if there is no SPXCAT1 or SPXCAT2 DD statement in the JCL defined within SPXQJCL0, and there is no SPXCAT1 or SPXCAT2 DD statement in the jobstep or TSO session that is executing SPXQJCL0, then the generated JCL has no SPXCATn DD statements;

(3) if there is no SPXCAT1 or SPXCAT2 in the JCL defined within SPXQJCL0, but there is a SPXCAT1 and/or SPXCAT2 allocated in the jobstep or TSO session that is executing SPXQJCL0, then the same SPXCATn allocation is included on additional DD statements that are added to the generated JCL.

The supplied submit routines require an input library to contain the user JCL for all jobs in each network. Consequently, a DD statement must be provided for this library in the JCL (or TSO CLIST) for executing the SPANEX Utility INCLUDE, NETSTART, SCHEDULE, POST and PROCEED commands. If a second JCL input library is to be set up, for temporary JCL overrides, a DD statement must be provided for this. Concatenated JCL libraries are supported. When the Quicknet submit routines create SPANEX executions in submitted user jobs, the DD statement(s) for the JCL dataset(s) will be propagated to the generated JCL. It is a requirement that all Quicknet JCL libraries are catalogued datasets.

Currently supplied Quicknet submit routines are:

SPXQCK01	Reads JCL from Partitioned Dataset to Internal Reader. DDnames: JOBPDS (required), TEMPPDS (optional for temporary JCL overrides).

SPXQCK02	Reads JCL from CA-PANVALET library to Internal Reader. DDnames: PANVALET (required), TEMPPANV (optional for temporary JCL overrides).
SPXQCK03	Reads JCL from CA-LIBRARIAN Master to Internal Reader. DDnames: LIBRMAST (required), LIBRTEMP (optional for temporary JCL overrides).

These module names are automatically generated in Quicknet RCMs according to the value specified for the "LIBRARY=" parameter of the QUICKNET macro. The DDnames must be included in all SPANEX Utility execution jobsteps, and in all TSO CLISTs used to invoke SPANEX.

10.5 Examples of SPANEX Quicknet RCM Generation Input

This section contains examples of the use of SPANEX Quicknet RCM Generation Macros.

10.5.1 Example 1

10.5.1.1 RCM Generation Input

```
JOB001    QUICKJOB
JOB002    QUICKJOB PREREQ=JOB001
JOB003    QUICKJOB PREREQ=JOB002
NET001    QUICKNET
```

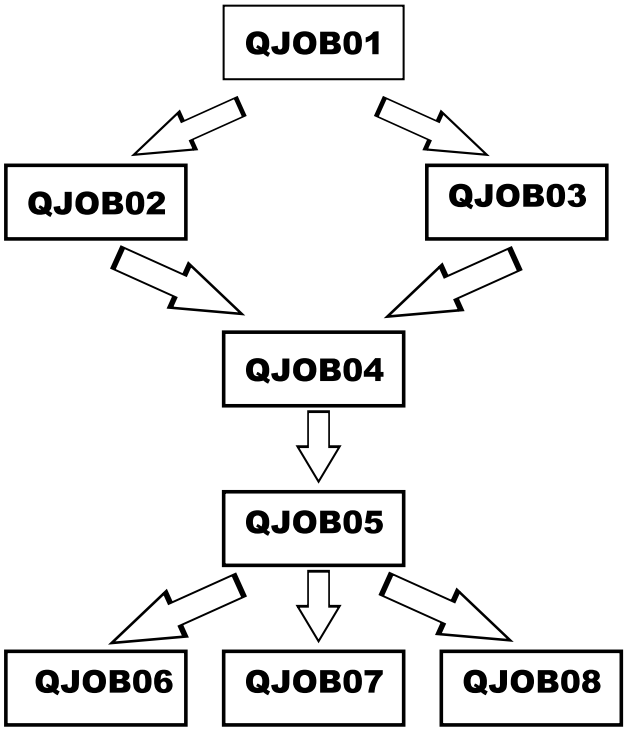
10.5.1.2 Explanatory Notes

This example shows a simple network of three jobs with linear dependency. In other words, when JOB001 ends successfully, JOB002 is to be run; when JOB002 ends, JOB003 is to be run. The QUICKJOB and QUICKNET macros define this dependency, and the default of keeping the JCL on a partitioned dataset is assumed. Note that a JOBPDS DD statement, allocated to the PDS containing the Job JCL, is required in the SPANEX Utility execution so that the SPANEX command processors can submit the Jobs.

10.5.2 Example 2

The job network depicted in the diagram below is a simple example of job dependencies. Following the network diagram is the Quicknet macro sequence to describe this job relationship.

10.5.2.1 Network Diagram



Description continued on next page

10.5.2.2 RCM Generation Input

```
QJOB01    QUICKJOB
QJOB02    QUICKJOB PREREQ=QJOB01
QJOB03    QUICKJOB PREREQ=QJOB01
QJOB04    QUICKJOB PREREQ=(QJOB02,QJOB03)
QJOB05    QUICKJOB PREREQ=QJOB04
QJOB06    QUICKJOB PREREQ=QJOB05
QJOB07    QUICKJOB PREREQ=QJOB05
QJOB08    QUICKJOB PREREQ=QJOB05
NETQ01    QUICKNET GLOGDD=GLOGQ01,
          GLOGDSN=Q01.GLOBAL.LOG
```

10.5.2.3 Explanatory Notes

The QUICKJOB and QUICKNET macros define this dependency in the same way as the SPXJOB, SPXSTEP and SPXRCM macros shown in the example in Section 11 of this manual for full-function SPANEX examples, and the default of keeping the JCL on a partitioned dataset is assumed. Additionally, a Global Log is defined for this network, and SPANEX Dynamic Allocation is to be used for the Global Log dataset. Note that a JOBPDS DD statement, allocated to the PDS containing the Job JCL, is required in the SPANEX Utility execution so that the SPANEX command processors can submit the Jobs.

10.6 How to Get Started (SPANEX Networking from Scratch)

This section gives a basic list of all the activities that are required to get SPANEX Quicknet job networking implemented on an existing application.

References are included to other SPANEX publications when extra information may be required.

1. Install SPANEX using the SPANEX Installation and Maintenance manual, which includes a Check List of activities.

2. Compile a list of all the jobs in your application, and determine the job dependencies (this may be a “straight-line” dependency, or any complexity of inter-dependency and multi-programming is supported).

3. Define your jobs to SPANEX using the Quicknet macro statements described in Section 10.3 on page 153 of this manual.

4. Assemble your Quicknet macros to form a SPANEX RCM, using JCL similar to that shown on the SPANEX Reference Card.

5. Ensure that your application job JCL is held on the library type (Partitioned Dataset, CA-PANVALET library or CA-LIBRARIAN Master) that you have chosen.

6. Execute the SPANEX Utility (With 3270-type terminal: consult SPANEX Terminal User's Guide. Without 3270-type terminal: consult SPANEX Restart and Job Networking Guide).

7. Enter a NETSTART command for your network.

This page intentionally left blank.

11 Examples of SPANEX Restart Facility and Job Networking

This section contains examples of the use of the SPANEX Restart Facility and of SPANEX Job Networking. Each example consists of a sample RCM generation, skeleton JCL for the job(s) described in the RCM, and notes explaining the function of the various options shown. Some examples are also given of the use of the SPANEX Utility.

11.1 Example 1 -- Simple Job Restart

11.1.1 RCM Generation Input

```
JOB101    SPXJOB
STEP1101  SPXSTEP  CODE=(0,0)
STEP1102  SPXSTEP  CODE=(4,8)
STEP1103  SPXSTEP  CODE=(8,0)
RCM001    SPXRCM   CONFIRM=YES
```

11.1.2 Job Control Language

```
//JOB101   JOB  (xxx,xxx,...),CLASS=x,...
//STEP1101 EXEC PGM=SPANEX,
//              PARM='pgmname,OPT=IM,RCM=RCM001/parm'
          .       .
      user DD statements here
          .       .
//STEP1102 EXEC  PGM=SPANEX,PARM='pgmname,OPT=M/parm',
//         COND=(4,LT,STEP1101)
          .       .
      user DD statements here
          .       .
//STEP1103 EXEC  PGM=SPANEX,PARM='pgmname,OPT=M/parm'
          .       .
      user DD statements here
          .       .
```

11.1.3 Explanatory Notes

This straightforward example shows a job with three processing steps. If a failure occurs in any step, restart execution will begin with that step; if a failure occurs after any step and before the next step, restart execution will begin at the point of failure (beginning with the next step). The operator will have the opportunity to confirm or override every restart.

11.2 Example 2 -- Intelligent Job Restart

11.2.1 RCM Generation Input

```
JOB201    SPXJOB   CONFIRM=YES
STEP2101  SPXSTEP
STEP2102  SPXSTEP  CODE=(8,4)
STEP2103  SPXSTEP  CODE=(0,4)
STEP2104  SPXSTEP  CODE=(8,12)
STEP2105  SPXSTEP  CODE=(12,0)
RCM002    SPXRCM   OPTU=NO
```

11.2.2 Job Control Language

```
//JOB201   JOB  (xxx,xxx,...),CLASS=x,...
//STEP2101 EXEC  PGM=SPANEX,PARM=',OPT=I,RCM=RCM002'
//STEP2102 EXEC  PGM=SPANEX,PARM='restore,OPT=M/parm',
//         COND=(8,NE,STEP2101)
                .      .
      user DD statements here
                .      .
//STEP2103 EXEC  PGM=SPANEX,PARM='pgmname,OPT=M/parm',
//         COND=(0,NE,STEP2101)
                .      .
      user DD statements here
                .      .
//STEP2104 EXEC  PGM=SPANEX,PARM='updpgm,OPT=M/parm',
//         COND=(8,LT,STEP2101)
                .      .
      user DD statements here
                .      .
//STEP2105 EXEC  PGM=SPANEX,PARM='pgmname,OPT=M/parm'
                .      .
      user DD statements here
                .      .
```

11.2.3 Explanatory Notes

This example shows a job with three processing steps, one of which, STEP2104, performs a file update. STEP2102 executes a utility to recover the file that STEP2104 updates, in the event of a restart after a failure during STEP2104. STEP2101 executes only the SPANEX Restart Initialization processor. Other steps are executed in the order STEP2103-STEP2104-STEP2105 for a clean-start or for a restart after a failure during STEP2103; in the order STEP2102-STEP2104-STEP2105 for a restart after a failure during STEP2104; and STEP2105 only for a restart after a failure during STEP2105. If, during a restart run, a second failure occurs during the file restore in STEP2102, the

second restart will again attempt STEP2102 followed by STEP2104 and STEP2105.

11.3 Example 3 -- Dependent Jobs

11.3.1 RCM Generation Input

```
JOB301    SPXJOB
STEP3101  SPXSTEP  CODE=(0,0)
STEP3102  SPXSTEP  CODE=(4,8)
STEP3103  SPXSTEP  CODE=(8,0)
JOB302    SPXJOB   PREREQ=JOB301
STEP3201  SPXSTEP
STEP3202  SPXSTEP  CODE=(8,4)
STEP3203  SPXSTEP  CODE=(0,4)
STEP3204  SPXSTEP  CODE=(8,12)
STEP3205  SPXSTEP  CODE=(12,0)
RCM003    SPXRCM   OPTU=NO,CONFIRM=YES
```

11.3.2 Job Control Language

```
//JOB301   JOB  (xxx,xxx,...),CLASS=x,...
//STEP3101 EXEC PGM=SPANEX,
//              PARM='pgmname,OPT=IM,RCM=RCM003/parm'
              .       .
      user DD statements here
              .       .
//STEP3102 EXEC  PGM=SPANEX,PARM='pgmname,OPT=M/parm',
//         COND=(4,LT,STEP3101)
              .       .
      user DD statements here
              .       .
//STEP3103 EXEC  PGM=SPANEX,PARM='pgmname,OPT=M/parm'
              .       .
      user DD statements here
              .       .
//

//JOB302   JOB  (xxx,xxx,...),CLASS=x,...
//STEP3201 EXEC  PGM=SPANEX,PARM=',OPT=I,RCM=RCM003'
//STEP3202 EXEC  PGM=SPANEX,PARM='restore,OPT=M/parm',
//         COND=(8,NE,STEP3201)
              .       .
      user DD statements here
              .       .
//STEP3203 EXEC  PGM=SPANEX,PARM='pgmname,OPT=M/parm',
//         COND=(0,NE,STEP3201)
```



```
             .       .
      user DD statements here
             .       .

//STEP3204 EXEC  PGM=SPANEX,PARM='updpgm,OPT=M/parm',
//         COND=(8,LT,STEP3201)
             .       .
      user DD statements here
             .       .
//STEP3205 EXEC  PGM=SPANEX,PARM='pgmname,OPT=M/parm'
             .       .
      user DD statements here
             .       .
```

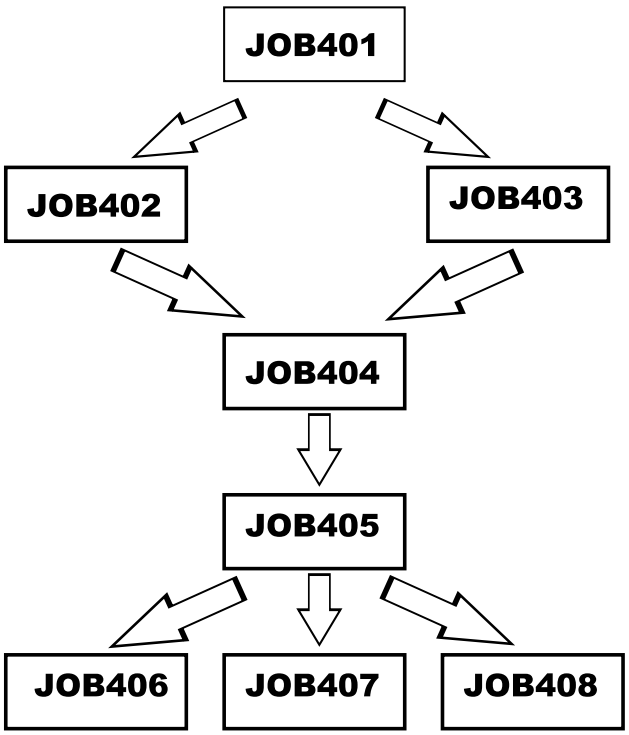
11.3.3 Explanatory Notes

This example shows the jobs from the previous two examples combined into a job suite, where the first job, JOB301, is a pre-requisite to the second job, JOB302. Note that this does not constitute a SPANEX job network because the "JOBNET=YES" parameter is not specified on the SPXRCM macro. SPANEX will in this case not perform any automatic job scheduling, but will perform dependency checking when JOB302 begins execution.

11.4 Example 4 -- Job Networking

The job network depicted in the diagram below is an example of a simple suite of jobs. Following the diagram is a partial RCM generation input which describes the job interrelationships. A similar job network is the subject of some sample modules supplied with SPANEX (see Section 12 of this manual).

11.4.1 Network Diagram



Description continued on next page

11.4.2 RCM Generation Input

```
JOB401    SPXJOB
STEP401   SPXSTEP
JOB402    SPXJOB   PREREQ=JOB401
STEP402   SPXSTEP
JOB403    SPXJOB   PREREQ=JOB401
STEP403   SPXSTEP
JOB404    SPXJOB   PREREQ=(JOB402,JOB403)
STEP404   SPXSTEP
JOB405    SPXJOB   PREREQ=JOB404
STEP405   SPXSTEP
JOB406    SPXJOB   PREREQ=JOB405
STEP406   SPXSTEP
JOB407    SPXJOB   PREREQ=JOB405
STEP407   SPXSTEP
JOB408    SPXJOB   PREREQ=JOB405
STEP408   SPXSTEP
RCM401    SPXRCM   JOBNET=YES,NETRTN=SPXNJS02
```

11.4.3 Explanatory Notes

SPANEX Job Networking will process the above network as follows:

- the NETSTART command will schedule JOB401 only.
- successful completion of JOB401 will cause automatic scheduling of both JOB402 and JOB403.
- the second of JOB402 and JOB403 to complete successfully will cause automatic scheduling of JOB404.
- successful completion of JOB404 will cause automatic scheduling of JOB405.
- successful completion of JOB405 will cause automatic scheduling of JOB406, JOB407 and JOB408.

- if JOB401 is EXCLUDEd, NETSTART will schedule JOB402 and JOB403.
- if JOB404 is EXCLUDEd, the second of JOB402 and JOB403 to complete successfully will cause automatic scheduling of JOB405.
- if JOB405 is EXCLUDEd, successful completion of JOB404 will cause automatic scheduling of JOB406, JOB407 and JOB408.
- if both JOB404 and JOB405 are EXCLUDEd, the second of JOB402 and JOB403 to complete successfully will cause automatic scheduling of JOB406, JOB407 and JOB408.

11.5 Example 5 -- Use of SPANEX Utility - 1

The SPANEX Utility may be executed as a batch job, as a started task, via a dedicated SPANEX 3270-type terminal, or under the control of a time-sharing system such as TSO. Typical Job Control Language for a batch job or started task is given below.

```
//stepname EXEC  PGM=SPANEX,PARM=',OPT=U'
//SPXPRINT DD  SYSOUT=A
//SPXRSCTL DD  *
  PRINT  RCM=USERRCM1
  DELETE RCM=USERRCM2,JOB=RCM2JOB1
/*
```

This execution of the SPANEX utility will print the status of all jobs defined by RCM “USERRCM1”, and will then delete the existing restart status for job “RCM2JOB1” which job is defined by the RCM “USERRCM2”. Note that if the SPXRSCTL DD statement were omitted, an outstanding reply would be sent to the MCS console requesting SPANEX command input.

11.6 Example 6 -- Use of SPANEX Utility - 2

The SPANEX RCM in this example has been defined as a SPANEX Job Network. This example shows the SPANEX Utility commands that could be used to start the execution of the job network.

```
//stepname EXEC  PGM=SPANEX,PARM=',OPT=U,RCM=USERRCM3'
//SPXPRINT DD  SYSOUT=A
//TASKLIB  DD  DSN=user.load.library,DISP=SHR
//SPXRSCTL DD  *
  EXCLUDE  JOB=RCM3JOB4
  EXCLUDE  JOB=RCM3JOB5
  NETSTART
/*
```

This execution of the SPANEX Utility will operate only on the RCM named “USERRCM3” because this is specified in the EXEC statement PARM field. The Utility will prevent the execution of jobs “RCM3JOB4” and “RCM3JOB5” during this run of the network, and will initiate the execution of the network. Note that any additional JCL DD statements required by the Job Submit routine defined in RCM USERRCM3 must be included in this jobstep.

This page intentionally left blank.

12 The SPANEX Sample Network

The SPANEX installation process (the #SPXGEN macro) places in the SPANEX source library a number of modules that may be used as an illustration of a simple SPANEX job network. These are generated to conform with user installation standards, and may be used in order to obtain a demonstration SPANEX network with the minimum of delay. The functions of these source modules are given below. Note that many of the member names are derived from the parameters of the #SPXGEN macro and so cannot be listed here.

Source Module	Function
SAMPBTAM	Sample JCL to execute the SPANEX Utility Local 3270 BTAM support routine. JCL must be altered to specify the correct dataset names for user installation standards.
SAMPEONR	Sample end-of-network routine that may be used by MVS installations to perform the standard function of printing the Global Log dataset at the end of network execution.
sample JCL 1	First of eight sample jobs - each job consists of skeleton JCL and the execution, via SPANEX, of null utility program IEFBR14. The member name is the same as the job name, and will conform to user installation standards.
sample JCL 2 sample JCL 3 sample JCL 4 sample JCL 5 sample JCL 6 sample JCL 7 sample JCL 8	Remaining sample jobs
SAMPNET	Sample RCM generation input defining the eight sample jobs. The RCM generation input is constructed by the #SPXGEN macro so as to contain the correct jobnames for the sample network. The relationship between the jobs is equivalent to the example given in section 11.4 of this manual. Some use is made of SPANEX job Hold Events and Process Options for illustrative purposes.
SAMPTSOP	Sample TSO Command Procedure (CLIST) to execute the SPANEX Extended TSO 3270 support routine for full-screen SPANEX Utility. Dataset allocations will need to be altered to conform to user installation standards.
SAMPUTIL	Sample JCL to execute the SPANEX Utility as a batch job or as a Started Task. This JCL will need to be changed to conform to user installation JCL standards.

This page intentionally left blank.

Appendix A - Valid Values for SPANEX Job Status

This Appendix contains a list of the possible values for the SPANEX "STATUS" field, as listed by the RCM Map facility, or by SPANEX itself in normal SPANEX messages, or in SPANEX Restart and Networking facility control blocks, together with a brief description of each. Note that all SPANEX status values are four-character printable words.

ABNS — Job has Abended in a SPANEX-controlled step with a System Abend (the code will be contained in the SPANEX "CODE" field).

ABNU — Job has Abended in a SPANEX-controlled step with a User Abend (the code will be contained in the SPANEX "CODE" field).

ABNX — Job has Abended in a non-SPANEX-controlled step, and the Abend has been detected by the Retrospective Condition Code checking function. The Abend code is unknown.

ABPF — Job was terminated by the system operator by means of a SPANEX STOP or MODIFY command (SPANEX option "P" or "F" was in effect for a SPANEX restart-controlled step).

ABRC — Job has abnormally terminated because an error condition code was set by the user program in a SPANEX restart-controlled step (the code will be contained in the SPANEX "CODE" field).

BLNK or blank — The SPANEX Restart Initialization process has decided upon a clean start of the job but the first SPANEX-controlled processing step has not yet begun execution.

INIT — Job is in the process of Restart Initialization (the SPANEX OPT=I function is active) and SPANEX is analyzing whether a Clean Start or a Restart of the job is required.

PREX — Job has been scheduled by SPANEX (automatically or as a result of a NETSTART, INCLUDE, POST, PROCEED or SCHEDULE SPANEX Utility command) but has not yet begun execution.

RSTR — SPANEX has attempted a restart of the job after an earlier failure - the step which SPANEX has nominated for the restart has not yet begun execution (the code used by SPANEX for the restart will be contained in the SPANEX "CODE" field).

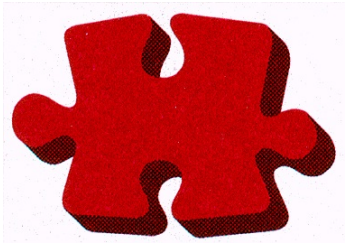
STEN — Job is processing normally and a SPANEX-controlled step has successfully completed execution.

STST — Job is processing normally and a SPANEX-controlled step has begun execution.

SUCC — Job has successfully completed execution.

UCHK	Job has been set abnormally-terminated as a result of a request by a User Check Exit routine.
URST	SPANEX has attempted a restart of the job after an earlier failure upon the request of the SPANEX restart user exit routine defined for this job - the step which the user exit routine has nominated for the restart has not yet begun execution (the code specified by the user exit routine for the restart will be contained in the SPANEX "CODE" field).
USRR	The job has been treated by SPANEX as failed because of an internal call by the SPANEX user program in a SPANEX-controlled step to the SPANEX Utility, specifying that a processing error has occurred.

This manual is published by



Span Software Consultants Limited

Little Moss, Peacock Lane

High Legh

Knutsford

Cheshire

WA16 6PL

England

Tel: +44/0 1565 832999

Fax: +44/0 1565 830653

Email: spanex@spansoftware.com

www.spansoftware.com

to whom all comments and suggestions should be sent.

