2200 ECL
Script Processor

eclexec

Inglenet Business Solutions
www.inglenet.com

Revision Date: August 2014

Table of Contents

ECL Processor 4
Data files location 5
Job log 5
Daily log 6
ECLEXEC – Run Control files 8
BLOCKJOB definition 13
PRINTCOMMAND definition 14
SYMQUEUE Options 15
Debugging ECLEXEC 18
@start 19
st 19
@start test date 20
@xqt 20
@@CONS 20
ECLEXEC command line options 21
ECL Commands supported 22
Extra ECL Commands 23
SETENV Variables 23
TCOPY – Tape/file copy 24
ADD extra options 24
CYCLE extra options 24
SYM extra options 25
CAT extra options 25
ASG – Test if file exists 25
TEST condition 25
MSG with Question 26
COPY & DATA use of Unix path names 26
Environment Variables 27
Running other Unix commands from ECL 27
ECL Job Return Status 27
Interactive use of ECLEXEC 28
@JOBS 28
@jobs and ‘at’ from Unix shell 28
@jobs from in eclexec 29
@STOP jobname 30
@SPL command for held reports 30
@VIEW command for data files & reports 31

tipview..... 33
Inline ECL..... 35

ECL Processor

This document describes the ECL processor which is part of the TIP/ix system for migrating Unisys 2200 applications to Unix/Linux. This utility processes the ECL statements much like they would be handled on the 2200 and in addition, if an application is constructed using embedded ECL statements, those can be processed as well.

For extensive details on ECL you should refer to the Unisys documentation. This utility implements a subset of the ECL statements and options and this document will only summarize what is implemented and any special details as they relate to the Unix environment. At conversion time the 2200 ECL is processed by the '`scansrc`' utility and will be copied to the '`ecl`' directory and each file will have `#!/bin/eclexec` inserted. An example file called n032.ecl follows:

```
#!/bin/eclexec
@RUN          N032,SAF0000000/VSPSYS,INSP
@SYM,D        PRINT$.
@DELETE,C     N032BP.
@CAT,P        N032BP.,F///9999
@ASG,A        N032BP.
@BRKPT        PRINT$/N032BP
@MSG          INSPECTION STATISTICS REPORT
@ASG,A        PROD*N032DT.
@DELETE,C     PROD*N032P.
@FURPUR
@ASG,UP       PROD*N032P.,F///9999
@ASG,A        PROD*PROG$N.
@XQT          PROD*PROG$N.N032
@FREE         PROD*N032P.
@SYM,U        PROD*N032P.,,HOLD1
@FREE         PROD*N032DT.
@FREE         PROD*PROG$N.
@BRKPT        PRINT$
@SYM,D        PRINT$.
@FREE         N032BP.
 . @SYM,U        N032BP.
@FIN
```

When the file is marked with execute permission and the first line starts with #!, then Unix will run the named script processor and pass it the name of the script file.

To run this example, just run n032.ecl and the sample output follows:

```
/home/rjn:16 >n032.ecl
n032 Job started
n032 : INSPECTION STATISTICS REPORT
|**** INVALID INPUT YEAR 1980|
n032 Job completed successfully
```

The line enclosed in vertical bars '|' is output that the program had generated using DISPLAY UPON PRINTER.

To be similar to the 2200 you will likely want to start the jobs using the @start command. '@start' is a link to 'eclexec' and you can type from the standard Unix/Linux shell:

@start n032

Eclexec takes the name given and searches the PATH environment variable for 'name.ecl' and then executes the script.

Data files location

The base location of the data files is defined by the TIPHOMEDATA environment variable and the 2200 qualifier names map to a subdirectory of that location. For example if TIPHOMEDATA is `/home/vsp/data` then under the '`prod`' subdirectory you may fine files such as the following:

```
/home/vsp/data/prod:22 >ls -la
total 104
drwxrwxr-x    2 rjn      rjn          4096 Nov 25 17:40 .
drwxrwxr-x   10 rjn      rjn          4096 Nov 24 19:10 ..
-rw-rw-r--    1 rjn      rjn          4050 Nov 24 21:33 n012d1
-rw-rw-r--    1 rjn      rjn            29 Nov 21 14:17 n012dt
-rw-rw-r--    1 rjn      rjn             7 Nov 17 13:08 n023p
-rw-rw-r--    1 rjn      rjn          4154 Nov 24 10:23 n024p
-rw-rw-r--    1 rjn      rjn         39526 Nov 24 15:40 n025p.001
-rw-rw-r--    1 rjn      rjn            46 Nov 24 15:40 n025p.stt
-rw-rw-r--    1 rjn      rjn           560 Nov 25 17:40 n027d1
-rw-rw-r--    1 rjn      rjn            92 Nov 25 09:28 n027dt
-rw-rw-r--    1 rjn      rjn           393 Nov 25 17:16 n032dt
```

In the above example, the file n025p was referenced as a cycled file. For cycled files there is an extra file created (n025p.stt) which is used to manage the file cycles. The '.stt' file is just a text file with a few parameters in it to define the current cycle of the file.

Job log

Under the TIPHOMEDATA location are a couple of extra directories. One is called 'joblogs' and it holds the log file from the execution of the various ECL streams.

For example:

```
home/vsp/data/joblogs:36 >ls -la
total 48
drwxrwxr-x    2 rjn      rjn          4096 Nov 28 11:07 .
drwxrwxr-x   10 rjn      rjn          4096 Nov 24 19:10 ..
-rw-rw-r--    1 rjn      rjn          1180 Nov 25 17:26 n032.1.log
-rw-rw-r--    1 rjn      rjn          1180 Nov 25 17:21 n032.2.log
-rw-rw-r--    1 rjn      rjn          1143 Nov 25 17:17 n032.3.log
-rw-rw-r--    1 rjn      rjn          1186 Nov 25 17:16 n032.4.log
-rw-rw-r--    1 rjn      rjn          1100 Nov 25 17:15 n032.5.log
-rw-rw-r--    1 rjn      rjn           866 Nov 25 17:14 n032.6.log
-rw-rw-r--    1 rjn      rjn          1100 Nov 25 17:14 n032.7.log
-rw-rw-r--    1 rjn      rjn          1100 Nov 25 17:12 n032.8.log
-rw-rw-r--    1 rjn      rjn          1100 Nov 25 17:11 n032.9.log
-rw-rw-r--    1 rjn      rjn          1179 Nov 28 10:55 n032.log
```

The Unix file system keeps the date/time of each file so you can tell when the job was run. Eclexec will keep the last 10 executions of the ECL. In the above example the most recent is just called n032.log, the next recent is n032.1.log, then n032.2.log etc. The oldest joblog kept is n032.9.log.

A job log file looks like the following:

```
10:55:43.44 n032 Job Log generated by eclexec.c,v 4.2 on Fri Nov 28, 2003
10:55:43.44 Log file is /home/vsp/data/joblogs/n032.log
10:55:43.44 Run by User: rjn
10:55:43.44 Temp directory is /tmp/ecl.9103
10:55:43.44 @RUN          N032,SAF0000000/VSPSYS,INSP
10:55:43.44 @SYM,D        PRINT$.
10:55:43.44 @DELETE,C     N032BP.
10:55:43.44 @CAT,P        N032BP.,F///9999
10:55:43.44 @ASG,A        N032BP.
10:55:43.44 @BRKPT        PRINT$/N032BP
10:55:43.44 @MSG          INSPECTION STATISTICS REPORT
10:55:43.44 @ASG,A        PROD*N032DT.
10:55:43.44 @DELETE,C     PROD*N032P.
10:55:43.44 @FURPUR
10:55:43.44 @ASG,UP       PROD*N032P.,F///9999
10:55:43.44 @ASG,A        PROD*PROG$N.
10:55:43.44 @XQT          PROD*PROG$N.N032
10:55:44.07 @FREE         PROD*N032P.
10:55:44.07 @SYM,U        PROD*N032P.,,HOLD1
10:55:44.07 @FREE         PROD*N032DT.
10:55:44.07 @FREE         PROD*PROG$N.
10:55:44.07 @BRKPT        PRINT$
10:55:44.07   **** INVALID INPUT YEAR 1980
10:55:44.07 @SYM,D        PRINT$.
10:55:44.07 @FREE         N032BP.
10:55:44.07  . @SYM,U        N032BP.
10:55:44.07 @FIN
10:55:44.07 n032 Job completed successfully
```

Daily log

By default the daily log file is not created, but it is easy to turn it on if wanted. Just add CONSOLELOG=YES to the eclrc (ECL Run Control file) or $HOME/.eclrc.

The daily logs will be kept in `$TIPHOMEDATA/log` directory and the file name is `yyyymmdd.log`. For example:

```
/home/vsp/data/log:46 >ls -la
total 12
drwxrwxr-x    2 rjn      rjn          4096 Nov 28 11:13 .
drwxrwxr-x   10 rjn      rjn          4096 Nov 24 19:10 ..
-rw-rw-r--    1 rjn      rjn          2331 Nov 28 11:13 20031128.log
```

The content of the daily log file is a collection of all job logs run on that day. For example:

```
11:13:50.16 n032 Job Log generated by eclexec.c,v 4.2 on Fri Nov 28, 2003
11:13:50.16 Log file is /home/vsp/data/joblogs/n032.log
```

```
11:13:50.16 Run by User: rjn
11:13:50.16 Temp directory is /tmp/ecl.9147
11:13:50.16 @RUN          N032,SAF0000000/VSPSYS,INSP
11:13:50.16 @SYM,D        PRINT$.
11:13:50.16 @DELETE,C     N032BP.
11:13:50.16 @CAT,P        N032BP.,F///9999
11:13:50.16 @ASG,A        N032BP.
11:13:50.16 @BRKPT        PRINT$/N032BP
11:13:50.16 @MSG          INSPECTION STATISTICS REPORT
11:13:50.16 @ASG,A        PROD*N032DT.
11:13:50.16 @DELETE,C     PROD*N032P.
11:13:50.16 @FURPUR
11:13:50.16 @ASG,UP       PROD*N032P.,F///9999
11:13:50.16 @ASG,A        PROD*PROG$N.
11:13:50.16 @XQT          PROD*PROG$N.N032
11:13:50.71 @FREE         PROD*N032P.
11:13:50.71 @SYM,U        PROD*N032P.,,HOLD1
11:13:50.71 @FREE         PROD*N032DT.
11:13:50.71 @FREE         PROD*PROG$N.
11:13:50.71 @BRKPT        PRINT$
11:13:50.71   **** INVALID INPUT YEAR 1980
11:13:50.71 @SYM,D        PRINT$.
11:13:50.71 @FREE         N032BP.
11:13:50.71  . @SYM,U        N032BP.
11:13:50.71 @FIN
11:13:50.71 n032 Job completed successfully
11:13:53.66 n012 Job Log generated by eclexec.c,v 4.2 on Fri Nov 28, 2003
11:13:53.66 Log file is /home/vsp/data/joblogs/n012.log
11:13:53.66 Run by User: rjn
11:13:53.66 Temp directory is /tmp/ecl.9154
11:13:53.66 @RUN          N012,SAF0000000/VSPSYS,INSP
11:13:53.66 @SYM,D        PRINT$.
11:13:53.66 @DELETE,C     N012BP.
11:13:53.66 @CAT,P        N012BP.,F///9999
11:13:53.66 @ASG,A        N012BP.
11:13:53.66 @BRKPT        PRINT$/N012BP
11:13:53.66 @HDG          **** MOTOR VEHICLE INSPECTION SYSTEM ****
11:13:53.66 @DELETE,C     PROD*N012D1.
11:13:53.66 @FURPUR
11:13:53.66 @ASG,A        PROD*N012DT.
11:13:53.66 @ASG,UP       PROD*N012D1.,F///999
11:13:53.66 @ASG,A        PROD*PROG$N.
11:13:53.66 @XQT          PROD*PROG$N.N012
11:13:54.68 @FREE         PROD*N012DT.
11:13:54.68 @FREE         PROD*N012D1.
11:13:54.68 @FREE         PROD*PROG$N.
11:13:54.68 @SYM,U        PROD*N012D1.,,HOLD1
11:13:54.69 @BRKPT        PRINT$
11:13:54.69 @SYM,D        PRINT$.
11:13:54.69 @FREE         N012BP.
11:13:54.69  . @SYM,U        N012BP.
11:13:54.69 @FIN
11:13:54.69 n012 Job completed successfully
```

ECLEXEC – Run Control files

When `eclexec` is used, it checks for two control files. First it looks for and processes `/etc/default/eclrc` and then it looks for and processes `$HOME/.eclrc`. These files names are chosen to be consistent with other Unix based script processors. The content of the control files is the same. If `/etc/default/eclrc` is not present it then looks for `$TIPROOT/conf/eclrc`.

The `/etc/default/eclrc` would hold system wide parameters and the `$HOME/.eclrc` would hold parameters that you want to override for some special reason. (By `$HOME/.eclrc`, it means the user's home directory and the file called `.eclrc` in that location.) It is ok for either or both files to not exist.

The run control file may contain ECL commands which start with the '@' and are then processed for everything ECL script that is run. This may be needed for cataloguing files which do not actually need to exist, but which must be defined somehow. For example an ECL script may XQT a program from the program library, but on UNIX there is nothing like the program library, since executables are found doing a path search. In this case you may need to add a line to the `eclrc` file such as the following:

```
@CAT,P PROD*PROG$N.
```

And then ECL scripts may reference this on the XQT as follows:

```
@XQT   PROD*PROG$N.N012
```

Comment lines in the run control file start with the '#' character. For example:

```
#
# eclexec: Startup commands
#
```

The run control file may also contain 'setenv' commands to set environment variables:

```
setenv FOOBAR ${HOME}/foo
```

The `${name}` is replaced by the current value of the environment variable 'name'.

Some control commands for eclexec may also be placed in the run control file as a `keyword = value` parameter.

Command	Value	Description
ABENDROLLBACK	NO	If the job terminated abnormally then any cycle files advanced conditionally for the step of the job which aborted will be rolled back to the previous version. If you do not want this to happen give the value 'NO'.
BACKLOG	ht	Define logging options for jobs started in background. If this is defined, then a log file is created in the /tmp directory called ecl*nnn*.log where *nnn* is the process id. You must remember to remove these log files.
BLOCKJOB		Defines a job which should be blocked from running at defined times. See examples following.
CATREMOVE	YES	Indicates that when @CAT advances the file cycle and an old file if the new cycle value exists, that file should be removed.
CONSOLE	Device	Device or file to which job console messages are written.
CONSOLELOG	YES	To turn on the creation of the daily job accumulation log file. The default is NO.
CONSOLELOG	Directory	To define the exact location of the daily job log. The default location is $TIPHOMEDATA/log
CONSOLEUPPER	YES	To force all terminal input to Upper case. Default is the data is passed to the application as it was keyed in.
DATARCSZ	N	Define the size of records to be copied via @DATA from the ECL run and also following each XQT as re-directed input to the application. The default is 80.
DEFERADD	NO	YES means delay processing of @ADD until the last second. The file referenced is pulled in when the @ADD is executed. NO which means the file referenced by @ADD is pulled inline when the ECL stream is first loaded. Default YES
DEFERFREE	NO	By default @FREE is deferred until the next XQT or FIN. If the want the FREE to be completed immediately then add this option to eclrc.

Command	Value	Description
DEFERSTART	NO	By default @START is deferred until the job reaches normal termination. If the job aborts then the @START is not executed. If you want the @START to be executed as it occurs then add this option to the eclrc control file.
DEFINECMD	Cmd str	Used to define a simple macro. For example: `DEFINECMD=LF PRT,LF &1` Defines a command called 'LF' which is the PRT,LF command passing parameter 1 from LF to PRT.
DIRECTORY	Dir	Defines the default 2200 'directory' id to be used for file names.
DISPLAYBRKPT	NO	Default is YES which means that BRKPT files will be displayed when processed. If NO, then the BRKPT file is not copied to the print log file.
ECLTOBRKPT	YES	Will log ECL statements executed to the BRKPT file if other than PRINT$. Default NO
EMPTYFILE	YES	Will create a zero length file on @ASG,P and @CAT,P. Default is NO.
FILEGID	Num	Defines the numeric Unix group id value to be assigned to files created by eclexec
FILEGROUP	Name	Defines the Unix group name to be used to assign as a file group name.
FILEUID	Num	Defines the numeric Unix user id value to be assigned to files created by eclexec If FILEUID is not defined, the default is to place the user-id of the user which is running the ECL job on any files created by eclexec.
FILEUSER	Name	Defines the Unix user name to be used to assign as a file user name.
HOME	Directory	Define the directory which is to be made current for job started in background. The default is $TIPHOMEDATA.
IGNORECTRLC	YES	If the user is running a job and types Ctrl-C, this is the standard Unix command to "Cancel" the job. The default is to cancel the job. If you want Ctrl-C to be ignored then add IGNORECTRLC=YES to the eclrc or you local $HOME/.eclrc file.

Command	Value	Description
IGNOREHANGUP	YES	If the user is running a job and the terminal session is closed for any reason then the job is cancelled. The default is to cancel the job. If you want HANGUP to be ignored and the job to go into background then add IGNOREHANGUP=YES to the eclrc or you local $HOME/.eclrc file.
IGNORESTOPRUN	YES	Ignore non-zero status from application program. Default is to treat non-zero status as Abnormal termination.
JOBLOG	NO	To turn off the creation of the individual job logs. The default is YES.
JOBLOG	Directory	To define the exact location of the job logs. The default location is $TIPHOMEDATA/joblogs
JOBLOGEXPIRE	*n*D	Daily job log files created by the JOBLOG parameter will be removed after *n* days. *n*M indicates *n* months, *n*W indicates *n* weeks, nY indicates n years
KEEPCYCLE	YES	Will retain filename.stt even if there is no data file. Default is NO.
KEEPOVERPRINT	YES	For files of type LINE ADVANCING, keep all over print data.
KEEPOVERPRINT	NO	Do not keep any over print data at all. Maybe be overridden by the KEEPOVER on the SYMQUEUE or PRINTCOMMAND definitions.
KEEPSYMFILE	YES	Will keep the file being @SYMed even if ,U is not given. Default is to remove the file after @SYM if there is no ,U option.
LOGEXPIRE	*n*D	Daily log files created by the CONSOLELOG parameter will be removed after *n* days. *n*M indicates *n* months, *n*W indicates *n* weeks, nY indicates n years
MAXCARDSIZE	N	Defines the maximum length of data lines following @XQT. Default is 2048
MAXCYCLE	N	Defines the maximum number of files in a cycle. The default is 32.
MAXFILESIZE	N	Used to set ‘ulimit’ value. Can be ‘unlimited’ or nK or nM (mega) or nG (giga).

Command	Value	Description
MAXJOBLOG	N	Define how many past versions of the job log should be retained. The default is 10.
MAXSPOOLCYCLE	N	Defines max cycle of held spool files to keep. Default is 24
PRINTCOMMAND	Name, `lpp`, 'lpcmd', options	To define a printer name, its lines per page and the print disposition command. If `lpp` is defined, then trailing blank lines are removed and a Form Feed character is inserted. If `lpp` is not defined, then the file being printed is assumed to have already been created for a printing.
QUALIFER	Qual	Defines the default file qualifier name. Default is nothing.
STARTSTATUS	Sn	Defines where in the condition word the status of the last @START command is placed. The default is S6.
SKIPLINK	NO	Do not follow file system ljnks. The Default (YES) is to follow symbolic links.
SYMLINK	YES	If YES, then a Unix file is created as a link to the current file.nnn cycle. The default is to not create such links.
TIPROOT		Defines value to use for TIPROOT. The default value is taken from the environment.
TPFTOTEMP	NO	Do not map TPF$ to /tmp directory. The Default (YES) is to map TPF$ to a temporary directory under /tmp.
UMASK	num	Value to be used to set 'umask' value. Default is 2.
XQTDATA	YES	Any data following the @XQT command is sent into a Unix pipe as 'stdin' to the application program. This data would then be read by ACCEPT FROM CONSOLE inside the program. The data is also copied to a tempory file pointed to by the environement variable DD_CARDXREADER so that if the program has an FD referencing CARD-READER it can read the data. The default action is to pipe data following XQT to the program's stdin.
XQTDATA	NO	The data following XQT is not sent to 'stdin' and is only copied to the temporary CARD-READER file.

You may also use an include directive such as the following example:

```
setenv TIPROOT /home/tipsrc
INCLUDE ${TIPROOT}/conf/eclrc
```

BLOCKJOB definition

The BLOCKJOB statement has a few additional keywords

FROM=hhmm	Defines starting time when job should be blocked
UNTIL=hhmm	Defines ending time when job should be blocked
ONDAY=	Defines which days of the week the job should be blocked Listed in single quotes using Mo Tu We Th Fr Sa Su, A dash (‘-‘) indicates a day range Default is to block the job on all days.
DELAYTO=hhmm	Delay running job until defined time. If job is started during blocked time, then it is delayed until the defined HHMM, time of day. Default is just block the job and not schedule for a later time.
BACKGROUND=NO	Do not allow this job to be run in background mode.

Example:

```
BLOCKJOB=TSTPR  from=0700 until=1300
BLOCKJOB=BR5401 from=0730 until=1800 onday='Mo,Wed-Fri' Delayto=2300
BLOCKJOB=EV006T1 BACKGROUND=NO
```

PRINTCOMMAND definition

The run control file may have many PRINTCOMMAND definitions. The print command name of DEFAULT is used to define attributes for any printer not specifically defined.

PRINTCOMMAND	name, `lpp`, 'lpcmd', options	To define a printer name, its lines per page and the print disposition command. If `lpp` is defined, then trailing blank lines on each page are removed and a Form Feed character is inserted. If `lpp` is not defined, then the file being printed is assumed to have already been created for a printing. 'lpcmd' is the command to which the data is piped. If 'lpcmd' is omitted, the default is to use the Unix 'lp' program.
SPOOLDIR	path	Directory where reports will be held. Default is ${TIPHOMEDATA}/spool.
SYMQUEUE	name options	'name' is the @SYM queue name and the 'options' are described below

Following the PRINTER 'lpcmd' may be some options as follows:

Option	**Meaning**
ASIS	Pass file to 'lpcmd' with no changes
DEL1CR	If data file starts with a CR then remove it and pass the rest of the file.
DEL1CRLF	If data file starts with CR LF then remove that initial CR LF and pass the rest of the file.
DEL1LF	If data file starts with a LF then remove it and pass the rest of the file.
FFLF	When a new page is encountered a Form Feed & Line Feed will be emitted. The default is to only emit a Form Feed.
IGNORE	Just drop the output
KEEPOVER	Keep print lines which are over printing. The default is to remove and/or overlay lines which are overprinting for @SYM of regular data files. If the file being @SYMed was a LINE ADVANCING file, then the default is to keep print lines which are over printing.
NOCR	When formatting the file for printing each line will be ended with a single LF (line feed) rather than CR LF (carriage return, line feed)
SKIPEMPTY	If file does not exist or is zero length or (small and only has blank lines) then do not print the file.

SYMQUEUE Options

The SYMQUEUE name of DEFAULT defines the action for any @SYM queue name not otherwise defined.

Option	Meaning
IFBANNER=pattern	If the 'banner' parameter of the @SYM command matches 'pattern' then select this SYMQUEUE definition
IFFILE=pattern	If the first parameter of the @SYM matches 'pattern' then select this SYMQUEUE definition
IFPAGES=value	If the number of pages is more than 'value' then select this SYMQUEUE definition
IFQUAL=pattern	If the job qualifier mathes 'pattern' then select this SYMQUEUE definition.
IFRUN=pattern	If the job runid matches 'pattern' then select this SYMQUEUE definition
IFUSER=pattern	If the user name matches 'pattern' then select this SYMQUEUE definition.
DROP=YES	Drop report data
FORM=name	Use this 'form' name for printing
HOLD=YES	Hold report in SPOOLDIR
KEEPOVER=YES	Keep print lines which are over printing. The default is to remove and/or overlay lines which are overprinting for @SYM of regular data files. If the file being @SYMed was a LINE ADVANCING file, then the default is to keep print lines which are over printing.
MAXCYCLE=value	Max number of cycles of 'held' spool file to keep. Default is 24
PRINTER=name	Send the report data to the defined printer command
RETAIN=YES	Print to 'printer' and then HOLD report in SPOOLDIR
SETP1=value	Set $P1 to value for use by PRINTER disposition command
SETP2=value	Set $P2 to value for use by PRINTER disposition command
SETP3=value	Set $P3 to value for use by PRINTER disposition command
SKIPEMPTY=YES	If file does not exist or is zero length or (small and only has blank lines) then do not print the file.

For IFBANNER, IFUSER, IFQUAL, IFRUN the 'pattern' may be a simple character value, or a value enclosed in quotes or a pattern enclosed in quotes. If the pattern is '*name' that means to test for ends with 'name', 'name*' means to test for begins with 'name' and '*name*' means to test for contains 'name'. To use a simple pattern the value needs to enclosed in quotes.

For all of the IFxxxx tests, the SYMQUEUE name must match that used on the @SYM command and any specified IFxxxx test must also match. The possibilities are searched in the order in which they are defined in the eclrc (or .eclrc) file.

The print disposition command may contain some parameter substitution variables.

$$	Is replaced by the current Unix process id
${envname}	Is replaced by the value of the named environment variable
$1	Replaced by the value of the 1st positional parameter on the @SYM command.
$2 ... $9	2nd thru 9th parameters of the @SYM command
$B	is replaced by the @SYM banner parameter and prefixed by `-t,` if there was no banner parameter then it uses the 1st parameter on the @SYM
$C	is replaced by the @SYM number of copies value and prefixed by `-n`.
$D	is replaced by the @SYM destination name in lower case and prefixed by `-d`.
$E	Is replaced by the 1st parameter on the @SYM
$F	The filename being @SYMed
$G	Is replaced by the file name. If @SYM qual*file then just 'file' is used. If it was @SYM *usename* then the file name that the *usename* references is used.
$P1	Replaced by SETP1 value
$P2 … $P3	Replaced by SETP2, SETP3 values
$R	Is replaced by the run-id
$R	The job Run-Id from the @RUN statement
$U	Is replaced by the user-id who started the job
$W	Replaced by FORM= value

The default '`lpcmd`' is '`lp -s $D $B $C`'.

For example if the printer disposition command is "`lp $D $C $B`" and the @SYM command is "`@SYM,U  PROD*N024P.,1,P685`", then the resulting command to process the print file is "`lp -s -d p685`".

For example the `/etc/default/eclrc` file may be as follows:

```
#
# eclexec: Startup commands
#
PRINTCOMMAND=DEFAULT,"lp -s $D $B $C"
PRINTCOMMAND=HOLD1,66,"cat >${HOME}/prt.txt"
PRINTCOMMAND=P685,88,"lp -s $D $B $C "
PRINTCOMMAND=LAB12,12,"lp -s $D $B $C ",NOCR
PRINTCOMMAND=PLIC,66,"lp -s $D $B $C -W $W ",FFLF
PRINTCOMMAND=PLIX,"lp -s $D $B $C -W $W ",DEL1CR
PRINTCOMMAND=MAILRJN,"mailx $P1",ASIS
SPOOLDIR=${HOME}/spool
PRINTCOMMAND=RJNPC,,"smbclient -U uid%pwd -W dom.ca //mpc/prntr -c 'print -'"
SYMQUEUE=DEPC01 IFBANNER='*RJN*' PRINTCOMMAND=MAILRJN SETP1=rjn@inglenet.com
SYMQUEUE=DEPC01 IFBANNER=SANDY PRINTCOMMAND=MAILRJN SETP1=sandya@inglenet.com
SYMQUEUE=DEPC01 IFBANNER=RJNPC RETAIN=YES LPP=50 USEPRINTCOMMAND=RJNPC
SYMQUEUE=DEPC01 IFBANNER='*PR' PRINTCOMMAND=HOLD1
SYMQUEUE=DEPC01 PRINTCOMMAND=DEFAULT
DEFINECMD=LF PRT,LF &1
```

And in your home directory you might have an `.eclrc` as follows:

```
# Local eclexec control parameters
PRINTCOMMAND=HOLD1,"cat >${HOME}/hold1.txt"
```

With the above examples, any @SYM would send data to the Unix `lp` and then onto a printer except for @SYM referencing HOLD1 as the output. In that case the print file would just be written to the file called "hold1.txt" in my home directory.

Debugging ECLEXEC

If you run an ECL script with the option `-l` then a log file called `ecl.log` is created in your home directory. This file will have many detailed lines of information from the running of the job.

```
@start myjob -l
```

There is also the BACKLOG= parameter in $TIPROOT/conf/eclrc which will turn on logging for all jobs run in background.

Another option is to add to the `#/bin/eclexec` which is the first line of each job the –l option. So the first line of the job for which logging is enabled would become:

```
#/bin/eclexec  -l
```

This will create a log file in the user's home directory called ecl*jobname*.*nnnn*.log where *jobname* is the @RUN name and *nnnn* is the process id.

If the job terminated abnormally then any cycle file advanced conditionally will be reset back to the previous version. If you do not want this to happen you can run the job with the –k option

```
@start myjob -k
```

@start

@start is set up as a link to `eclexec`. If `eclexec` is started with the name @start then it takes the parameter as the name of an ECL to be run.

For example (from the Unix/Linux shell prompt):

```
@start i104
@start i104,100
```

In the 2nd example above, the initial switches are set to 100 octal. The initial switch value is placed into T2 of the condition word.

When invoked as `@start, eclexec` knows to append `.ecl` when looking for the job to run.

If you want the ecl.log file created then append the `-l` option (lower case L). Example:

```
@start i104,100 -l
```

If you want the job to run in background then append –b option. For example:

```
@start i104,100 -b
```

You may also supply parameter values which can be picked up in the ECL via ${n} where 'n' is the number of the positional parameter. A 'positional parameter' is anything after the job name that does not start with a dash ('-'). So if the job is:

```
#!/bin/eclexec
@RUN,/W HR3000,PROD-SUPPORT,PROD
@ADD PROD*ECL-HR3000.A/SSG
YEAR ${1:99}
@EOF
@FIN
```

And you run:

```
@start hr3000 12
```

Then ${1:99} is replaced by '12'. The default is '99' so if you run

```
@start hr3000
```

Then ${1:99} is replaced by 99.

st

'st' is set up as a link to `eclexec`. If `eclexec` is started with the name '`st`' then it takes the parameter as the name of an ECL to be run. For example:

```
st i104
st i104,100
```

In the 2nd example above, the initial switches are set to 100 octal. When invoked as `st`, `eclexec` knows to append `.ecl` when looking for the job to run.

@start test date

You may follow the job name with DATE= to define an alternate date to be used by the program for execution. `eclexec` creates a Micro Focus config file with the alternate date defined and sets the environment variable (COBCONFIG) so that the Micro Focus run-time picks this up. The date given can be in the form CCYYMMDD (example: DATE=20130416) or just MMDD (example DATE=0416) and it uses the current year. Example:

```
@start myjob date=20121231
```

@xqt

@xqt is set up as a link to `eclexec`. If `eclexec` is started with the name @xqt then it takes the parameter as the name of an binary to be run. For example:

```
@xqt tip295
```

This may be useful to run batch programs which do not need any other ECL statement to control their execution.

@@CONS

ECL jobs may be started by any of the following methods at the Unix prompt:

```
i499.ecl

@start i499

@start i499,100
```

While the job is running, your terminal session is waiting for it to complete. If a program in the job does DISPLAY UPON CONSOLE and then ACCEPT FROM CONSOLE. You are effectively being prompted to enter some input to the program. Just key in the correct response and press 'Enter'.

If the job/program is a long running one and you want to check that it is still active you can key in:

```
@@CONS RC
```

Also, if you want to cancel a long running job, you could key in:

```
@@CONS X
```

The @@CONS is only valid for the job which is currently running on your terminal. Any jobs which have been started in background can not be processed with this @@CONS method. @@STOP is the same as @@CONS X and kill the current job.

ECLEXEC command line options

You may follow `eclexec` with some command line options. If using the `@start` (alias for `eclexec`) command then following the job name may be some Unix/Linux style options as follows:

`-l`	Turns on logging of `eclexec` commands to a file called ecl.log in the user's home directory. The 'l' can be followed by logging options such as 5M to create up to a 5 MB log file.
`-c`	Tells `eclexec` that the job is being run by a job scheduler such as 'cron'.
`-b`	Tells `eclexec` that it should switch to running as a background daemon and execute the job. The user will not be able to interact with the running job at all.
`-k`	If the job terminates abnormally then any cycle file advanced conditionally will be reset back to the previous version. If you do not want this to happen you can run the job with the –k option.

The following job would run and create an ecl.log log file

```
@start myjob -l
```

The following job will be run as a background process:

```
@start myjob –b
st myjob -b
```

The following job will be run in background with logging to `$HOME/ecl`***`pidnum`***`.log` where ***`pidnum`*** is the process number.

```
@start myjob –b -l
```

ECL Commands supported

This table summarizes the ECL commands and the options which are implemented.

Command	Options	Comments
ADD		
ADD	I	Process @ADD on initial load of the job. Ignore DEFERADD parameter in eclrc.
ADD	X	Process @ADD later, just before it is used. Ignore DEFERADD parameter in eclrc.
ASG	ACDIKTPU	
BRKPT		
CAT	IUP	
CPFTP		Simple FTP command
CYCLE	C	
DATA		
DELETE	C	
DMU		INITILIZE AREA is implemented. This invokes the `dbidmu` utility.
END		
EOF		
FILE		
FIN		
FREE	D	
HELP		Provide summary list of commands possible
JUMP		
MSG		
QUAL		
RUN		
SETC		
SORT		Invokes `armsort`
SPL	L	For listing and view held print spool files
START	F	Start job inline in foreground. Normally and @start in an ECL job results in the @started job being run in background while the job containing the @start continues to run. @START,F starts the job inline in foreground and waits for it to complete before proceeding to the next statement in the current ECL job.

Command	Options	Comments
SYM	DU	
SYM	C	just copy report to the file named in the 3rd parameter
TCOPY		Tape copy. see description below
TEST		
TEST	EQ	Compare parameter 1 & 2 and set condition word
VIEW	L	For listing and viewing data files
	E	Open file and if small do an inline edit of the contents. This is handy for updated those small control card type of files.
USE		
XQT		
XQT	C	The C option will send data following the XQT to stdin for the program to read with ACCEPT from CONSOLE regardless of the XQTDATA option defined.
XQT	N	The N option will NOT send data following the XQT to stdin regardless of the XQTDATA option defined.
XQT	X	Do not send any data thru stdin and do not intercept stdout form the running program.

Extra ECL Commands

SETENV name value	To set an environment variable
ECHOENV name	To display the value of an environment variable
UNSETENV name	To remove and environment variable
SETRC n	Set the return code value. A negative value causes job abnormal termination.

SETENV Variables

TMPDIR	used on Unix to define the location for temporary work files. The default is normally /tmp. Some programs (such as SORT) use temporary work files to do their work. `eclexec` also creates a temp file for each job which is running in the TMPDIR location. You can change the location of these work files by redefining TMPDIR in the global `eclrc` control file or on you private `$HOME/.eclrc` control file.
SORTTEMP	defines a directory where `armsort` which is invoked by @SORT will write temporary work files. If this is defined it overrides the value of TMPDIR. `armsort` also calls the Unix `sort` program to do the actual data sorting.
SORKWORK	defines a directory where the Unix `sort` program should create its temporary work files.

TCOPY – Tape/file copy

Copy Data To/From tape

@TCOPY,options INPUT,OUTPUT,rcsz,bksz,vsn,lbl

Options:

```
A  - Convert data from Ascii to Ebcdic
E  - Convert data from Ebcdic to Ascii
F  - Tape is fixed length blocked format
     Default is Variable length
L  - Log the records in ecl.log
I  - Input is disk image of tape
O  - Output is to be disk image of tape
R  - Disk file is Record Sequential
D  - Disk file is Relative
S  - Disk file is Line Sequential
```

If copy to tape, then:

```
@TCOPY,E  MYQL*MYFILE,/dev/rmt/ctape1,512,12280
```

If copy from tape, then:

```
@TCOPY,A  /dev/rmt/ctape1,MYQL*MYFILE,512,12280
```

ADD extra options

@ADD,X file

Will defer including the contents of the file until the @ADD statement is reached. The default action is to include the file as the job statements are being read into memory before any statement gets executed.

Optionally you can add the option DEFERADD=YES to `.eclrc` or `/etc/default/eclrc` to force all @ADD processing to be deferred as long as possible. An @ADD following @XQT is always processed immediately as the file being added may contain data for the program executed by @XQT

@ADD,I file

Will include the contents of the file when the job is initially loaded and before anything has been executed in the job.

CYCLE extra options

@CYCLE,C file

The ‘C’ option will check that a file exists for the current cycle as defined on the file.stt control file. If the file does not exist, then file.stt is corrected to match what file does actually exist.

SYM extra options

The SYM command is used to print files on the 2200. But not all files where defined in the COBOL program as printer files. If you want to print a file which was not originally defined as a printer file in the COBOL code, then you have to give SYM some extra options so that `eclexec` knows how to format the file for printing.

On the 2200 the SYM command only uses the first 5 parameters. With `eclexec`, the 6th parameter may be the record size of the file and the 7th parameter may be the number of lines per page for processing the file. For example:

```
@SYM,U PROD*N023P.,1,L6,,N023P,80
```

or

```
@SYM,U PROD*N023P.,1,L6,,N023P,80,66
```

or

```
@SYM,U PROD*N023P.,1,L6,,N023P,,66
```

In addition the 'C' option to SYM indicates that the file is to be formatted for printing but just copied to the out destination as another file. For example

```
@SYM,UC PROD*N023P.,,PROD*N023RJN,,,80,66
```

Would copy the file N023P as 80 byte records and format it as print file using 66 lines per page and write out to N023RJN.

The 'X' option to SYM indicates that all cycles of the file being printed should be deleted after the print is sent to the printer. Without the X option and no U option only the specific cycle of the file is deleted. The U option indicates that nothing should be deleted.

CAT extra options

@CAT,I will remove any pre-existing file when cycle is advanced. (Just the one file is removed so that no stall data is left.) Normally, a batch program would then do an OPEN OUTPUT to write to the new file cycle, but if that does not happen there could have been an old file lying in the directory. @CAT,I will remove the old file.

ASG – Test if file exists

Use the Q option to assign a file and set the condition code if the file does not exist.

```
For example
@ASG,Q GC0800-C1
@TEST TE/0/S6
@JUMP LSTDTA
```

TEST condition

The @TEST statement can have the option string to be a condition to test for when comparing the string values of parameter 1 to parameter 2. The condition may be one of EQ, NE, LE, GE, LT, GT. For example:

```
@TEST,NE "${D}"," "
@JUMP RUNJOB
```

MSG with Question

The @MSG statement may be used to display a message, display a Yes/No type of question and set the condition code, or display a question and save the answer in the variable ${D}.

Example of displaying a message:

```
@MSG   "Enter nothing to terminate data entry for ${JOB}"
```

Example of display a question and save the answer:

```
@MSG,D "Enter next Data Card for ${JOB}"
@TEST,NE "${D}"," "
@JUMP RUNJOB
```

Example of display a Yes/No question (default answer is Yes) and set condition code:

```
@MSG,YN 'Do you want to change this'
@TEST TE/1
@JUMP RunJob
```

Example of display a No/Yes question (default answer is No) and set condition code:

```
@MSG,NY 'Do you want to change this'
@TEST TNE/1
@JUMP RunJob
```

COPY & DATA use of Unix path names

The COPY and DATA commands normally use 2200 style file names. As an option, you may also use a full Unix path name. This allows you to place data into files other that those being used to simulate the 2200 environment. If the 1st character of the file name is a slash (‘/’) or a dollar sign (‘$’) then it will be treated like a Unix/Linux path name. You may also have `${envname}` which will be replaced by the environment variables value. Some examples:

```
@DATA,IL      PROD*N069DK.
 99990   100   99
@END
@DATA      /tmp/foobat.txt
 99990   100   99X
@END
@COPY,A       PROD*PROG$N.N011B,TPF$.
@COPY  PROD*N069DK.,${TIPHOMEDATA}/prod/n069x
@ASG,UP SCRIBE*ERRORHISTSAV.
@COPY SCRIBE*ERRORHISTSAV.,/tmp/errhstbck
@FREE SCRIBE*ERRORHISTSAV.
@ASG,UP SCRIBE*ERRORHISTSAV(+1).
@COPY /tmp/errhstbck,SCRIBE*ERRORHISTSAV.
```

When you use a Unix path name, there is no support for file cycling.

To initialize a file before writing the data to it use `@DATA,I` *`filename`*

To append data to a file use `@DATA,A` *`filename`*

To list the contents of a file use `@DATA,L` *`filename`*

Environment Variables

The ECL processor has commands @SETENV & @UNSETENV to define and remove environment variables. On any ECL statement you may code ${envname} and have the string replaced by environment variable name. You may also code ${envname:defaultvalue} and then if 'evnname' is undefined the 'defaultvalue' will be used. Without a 'defaultvalue' the string ${envname} would just be left in the ECL statement as is. Whenever a file is assigned using the @ASG statement, the exact Unix path name to the assigned cycle of the file is defined in an environment variable called DD_filename. Eclexec will search for DD_envname if envname is not defined. So you can do things like the following:

```
@ASG,P GP1070-D1
@MSG "Processing file ${GP1070-D1}"
```

Running other Unix commands from ECL

If you want to run any other Unix utility from ECL, just code up @command where command is an executable found in the search PATH defined in your environment.

You can even pass Unix style command line parameters. For example:

```
@dbiexpimp -g -C 50000 -i ${GP1070-H1} -s pawnschema -t -r GP-HIST
```

If the name following @ is not an ECL command, then eclexec will check for it in /bin, /sbin or /usr/bin and if found, it is considered to be a Unix utility and is invoked in a special way.

You can also follow the name with some options such as:

X Do not send any data to command via stdin and do not interfere with anything coming thru stdin.

P Send data folloing the `@command` to the *command* via Unix pipe using popen.

D Send data folloing the `@command` to the *command* via Unix pipe using popen but end each line with CR LF instead of just LF.

Example:

@sftp,P

ECL Job Return Status

An ECL script may be executed by other processes such as `cron`. When the ECL script is terminated by the ECL processor a job status is retuned as well which the process which started the ECL script may test for.

Normal terminal status is 0 (zero).

Abnormal terminal status is -1 (minus one).

In addition the @SETRC may be used to set a different job termination status.

Interactive use of ECLEXEC

In addition to using `eclexec` as a script processor you may also run it interactively. Just type `eclexec` at the Unix prompt and hit return. You will then be prompted to enter ECL commands one at a time. To exit, just enter `@FIN`.

You can enter most ECL commands into the interactive session. Usually there is no default file qualifier so remember to enter file names in full such as

@ASG MYQL*MYFILE

Doing a @XQT and passing data to the program is not supported for the interactive use of `eclexec`. If you need to do this, create a regular ECL job and run it.

Other useful commands for interactive use are @SPL and @VIEW.

Hint: You can use the scan-up, scan-down keys to recall previously executed commands and then reuse the command by pressing enter or scan left & right to edit the command and then press enter to execute it.

@JOBS

The interactive command @JOBS will list other jobs which are currently running.

@jobs is also an alias for eclexec so that you can run @jobs form the standard Unix?Linc command shell

Example:

```
@jobs
Active ECL Jobs on   2010/07/22   at  11:42:30
Job:tstpr   XQT SLEEP P8958 at 2010/07/22 11:42:28
```

Jobs which are started and have a delayed start time on the @START or on the @RUN will be placed in the Unix/Linux ‘at’ job queue. Check your system man pages for details on ‘at’, ‘atq’ and ‘atm’. To list jobs which are in the ‘at’ queue you may also use @jobs with some options.

@jobs and ‘at’ from Unix shell

`@jobs -l`	Will list jobs currently in the (at) job queue
`@jobs -d` *`jobn`*	Will remove the job (by number) from the queue
`@jobs -r` *`jobn hhmm`*	Will remove ‘jobn’ & add it back at the new time HHMM
`@jobs -h`	Will check for and recover from a ‘futex’ hang on Linux
`@jobs` *`jobname`* `-T`	Will collect trace information about the running application program into a file called /tmp/ptrc*nnnn*.log using strace and gdb on Linux

Examples:

```
>@jobs -l
Jobnum User         Jobname                  Run at
 ===== =========    =========                ========
    93 rjn          tstibx1.ecl              2013-07-28 14:51 a
>@jobs -r 93 1650
Reschedule for 16:50
  '/bin/eclexec /home/rjn/bin/tstibx1.ecl -B -D'
>@jobs -l
Jobnum User         Jobname                  Run at
 ===== =========    =========                ========
    94 rjn          tstibx1.ecl              2013-07-28 16:50 a
>@jobs -d 94
Job 94 removed from 'atq'
/home/rjn:32 bit:55 >@jobs -l
No jobs in backlog queue (atq)
```

@jobs from in eclexec

`@jobs,L`	Will list jobs currently in the job queue
`@jobs,D` *`jobn`*	Will remove the job (by number) from the queue
`@jobs,R` *`jobn`* `hhmm`	Removes 'jobn' & adds it back at the new time HHMM
`@jobs,T` *`jobname`*	Collects trace information for given job name.

Examples

```
>eclexec
ECL Script processor (4.404 2013/07/27) © 1991-2013 Inglenet Business Solutions
    Enter @command (or @fin to exit)

ECL>@jobs
Active ECL Jobs on   2013/07/28   at  12:36:18
No jobs currently running
ECL>@jobs,l
Jobnum User         Jobname                  Run at
 ===== =========    =========                ========
    95 rjn          tstibx1.ecl              2013-07-28 14:51 a
ECL>@jobs,r 95,1705
Reschedule for 17:05
  '/bin/eclexec /home/rjn/bin/tstibx1.ecl -B -D'
ECL>@jobs,l
Jobnum User         Jobname                  Run at
 ===== =========    =========                ========
    96 rjn          tstibx1.ecl              2013-07-28 17:05 a
ECL>@jobs,d 96
Job 96 removed from 'atq'
```

@STOP jobname

If you need to kill another actively running job you can use the interactive @STOP command. @STOP is also an alias for eclexec so that you can run @stop from the standard Unix/Linux command shell.

Example:

```
@stop tstpr
Active ECL Jobs on   2010/07/22   at  11:42:40
Job:tstpr   XQT SLEEP P8958 at 2010/07/22 11:42:28
Killed Process 8958 was SLEEP of tstpr
```

@SPL command for held reports

The @SPL command has parameters with mostly the same meaning as the @SYM command except that parameter 1 refers to a held report. The command is most likely used interactively by running `eclexec`.

@SPL with no parameters will list how many reports are currently held under each SYMQUEUE. For example:

```
ECL>@spl
  depc01                  has 3 reports
  depc02                  has 4 reports
```

@SPL with 1 parameter will list reports held within the named SYMQUEUE. For example:

```
ECL>@spl depc01
  depc01*tspprnty(1)              Sun Oct  4 11:34:15 2009  Size 3K
  depc01*tspprnty(3)              Sun Oct  4 12:03:12 2009  Size 3K
  depc01*tspprnty(2)              Sun Oct  4 12:01:46 2009  Size 3K
  depc01 has 3 reports
```

@SPL can also list just information for a specific report as follows:

```
ECL>@spl depc01*tspprnty(3)
  depc01*tspprnty(3)              Sun Oct  4 12:03:12 2009  Size 3K
```

@SPL,V can be used to view the report contents. The default viewer program is ‘view’ unless and environment variable of SPOOLVIEW is defined with an alternate command name. For example:

```
ECL>@spl,v depc01*tspprnty(3)
```

@SPL,D can be used to delete a report as follows:

```
ECL>@spl,d depc01*tspprnty(3)
  depc01*tspprnty(3) removed
```

@SPL,R can be used to release a report for printing as follows:

```
ECL>@spl,r depc02*tspprnty(3),,RJNPC,,MyTitlePage
```

@VIEW command for data files & reports

The @view command can be used to get a summary list of files and to view individual data files using the Unix program defined by the environment variable SPOOLVIEW (default is the 'view' Unix program).

To get a count of files under each qualifier enter @view,L. For example

```
ECL>@view,l
  ccc has 2 files
  gcic has no files present
  2000 has 38 files
  rjn has 2 files
  prod has 16 files
  ifp has 2 files
```

If you want all files older than 90 days:

```
ECL>@view,l ,90
  ccc has no files present older than 90 days
  gcic has no files present older than 90 days
  2000 has 30 files older than 90 days
  rjn has 2 files older than 90 days
  prod has 15 files older than 90 days
  ifp has no files present older than 90 days
```

If you want all files larger than 90K enter:

```
ECL>@view,l ,,90
  ccc has no files present larger than 90K
  gcic has no files present larger than 90K
  2000 has 3 files larger than 90K
  rjn has no files present larger than 90K
  prod has no files present larger than 90K
  ifp has no files present larger than 90K
```

Files older than 90 days and larger than 90K:

```
ECL>@view,l ,90,90
  ccc has no files present older than 90 days & larger than 90K
  gcic has no files present older than 90 days & larger than 90K
  2000 has 1 file older than 90 days & larger than 90K
ECL>@view,l 2000,90,90
  2000*boatdata(1)                Wed Nov 11 21:41:32 2009    Size 92.3K
  2000 has 1 file older than 90 days & larger than 90K
```

To list all files in the RJN qualifier:

```
ECL>@view,l rjn
  rjn*foobar                        Tue May  8 16:34:20 2012    Size 0.1K
  rjn*tspbck.dat(1)                 Fri May  4 11:09:17 2012 ** Size 12.0K
  rjn*tspbck.idx(1)       VB-ISAM   Fri May  4 11:09:17 2012 ** Size 20.0K
  rjn*tspfile(87)                   Tue May  8 16:35:03 2012 ** Size 10.5K
  rjn*tspfile.idx(87)      C-ISAM   Tue May  8 16:35:03 2012 ** Size 5.0K
  rjn has 7 files
```

The ** indicates the file which is current in the cycle. ISAM file type may also be shown.

If you want to list all files with the prefix *gc172* enter:

```
ECL>@view,l 2000*gc172*
  2000*gc1720-d1(1)                 Wed Nov 11 02:09:42 2009    Size 84.9K
  2000*gc1720-d1(2)                 Wed Nov 11 02:15:28 2009    Size 84.9K
  2000*gc1720-d1(3)                 Wed Nov 11 02:18:40 2009    Size 84.9K
  2000*gc1720-d1(4)                 Wed Nov 11 21:41:32 2009 ** Size 84.9K
  2000*gc1720-r1(1)                 Wed Nov 11 02:09:42 2009    Size 0.5K
  2000*gc1720-r1(2)                 Wed Nov 11 02:15:28 2009    Size 0.5K
  2000*gc1720-r1(3)                 Wed Nov 11 02:18:40 2009    Size 0.5K
  2000*gc1720-r1(4)                 Wed Nov 11 21:41:32 2009 ** Size 0.7K
  2000*gc172a-d1(1)                 Wed Nov 11 02:09:52 2009    Size 84.9K
  2000*gc172a-d1(2)                 Wed Nov 11 02:15:38 2009    Size 84.9K
  2000*gc172a-d1(3)                 Wed Nov 11 02:18:50 2009    Size 84.9K
  2000*gc172a-d1(4)                 Wed Nov 11 21:41:33 2009 ** Size 84.9K
  2000*gc172* has 12 files
```

To view the cycle 13 version:

```
ECL>@view prod*ftp11(13)
```

@VIEW,L will list the files by name and then file modification time. If you want to list the files by the time last accessed use **@VIEW,A**

If you wanted to delete files based on some @VIEW,L criteria shown above, then add the X option.

@VIEW,X qual*file,daysold

Example:

@VIEW,X co-posts,240

Will ask to delete all files in the co-posts qualifier that are older than 240 days

@VIEW,XY qual*file,240

will automatically delete all files in the co-posts qualifier that are older than 240 days without asking permission to proceed. So the ,X option means to delete and ,Y option means to assume Yes answer.

The @VIEW command reads a bit of the data file and if the file contains CR or FF codes it is assumed to be a report file and will use 'tipview' to view the report.

Otherwise 'vim –R' or 'view' is used to view the data file.

`@VIEW,E` will invoke vim or vi to allow you to edit the file (providing the file is all valid ASCII data.) This also checks the environment for EDITOR and it defined uses that as the editing command.

If the file is just a single line file, then eclexec will display the line and allow you to edit it using simple cursor movement keys and when you press Enter the file is updated with the new contents. The Backspace key does back space and delete, the Delete key deletes the current character, then Insert key toggles between insert and overtype modes.

`@VIEW,R` will invoke 'tipview' to display the file one record at a time. If the file has binary data then it is displayed in hex and character mode. This will only work if the file has LF to separate the records. (I.e. it must be LINE SEQUENTIAL format).

@VIEW,R will invoke 'tipview' to display the file one record at a time in hex and character mode. This will only work if the file has LF to separate the records. (I.e. it must be LINE SEQUENTIAL format). Tipview removes NULs inserted by Micro Focus LINE SEQUENTIAL writing before displaying the data to you.

Hint: You can use the scan-up, scan-down keys to recall previously executed commands and then reuse the command by pressing enter or scan left & right to edit the command and then press enter to execute it.

tipview

This utility takes a page oriented view of the report file and displays the report a page at a time on the user's terminal. It will figure out the screen size and crop the report to the screen size so if you want to view the full width of the report, drag your terminal session (TIP/fe or putty) to be wide enough and as long as you want to view.

The screen will end with a line such as the following:

```
Line 31 on Page 1 of 3 (Quit, page# or Enter)>
```

At this point if you press enter it will display the rest of the current page and move thru the report accordingly with each 'Enter'. You may also type in a specific page number and the report will jump to that page and display from there. If you type Q and Enter it will quit.

You may also type `/string` to have `tipview` search form the current position in the report file for the string you typed. When found it will display the page that had that string.

Tipview usage information

```
TIP/ix ver 2013/08/21 2.5 R0 - 0254 © 1991-2013 Inglenet Business Solutions
 View a report file
tipview [-l lpp -r width] filename
  -l n   Defines that n lines are on a page; Default: 66
  -r n   Defines that each line is n characters long; Default: 132
         Use -r n option if the file has no CR/LF separaters
  -x     Display the file one record at a time (hex and/or character)
  -X     Display the file one record at a time (hex and/or character)
         And also remove MF inserted NULs
  Will display the report file a screen and page at a time
  When prompted enter:
    page#     to skip to that specific page (or record)
    Q         to quit viewing
    /string   to search forward for a page with the given string
    N         to search forward for next page with the last search string
    <Enter>   to move to next screen of data
```

Inline ECL

An API function call is available for processing ECL statements which a COBOL application may construct dynamically. The API function in C is called:

```
int parseExecEcl(char *eclstring);
```

Many 2200 customers have some MASM routine that does an `ER CSF$`. For each project we would write a small C routine with the same entry point name as the customers MASM routine, but then we would call `parseExecEcl` instead.