Appendix A - VCL Support Programs

A.1 ALGDWN - Algorithm Download

ALGDWN downloads a VCL algorithm to an execution platform. The execution platform can be a VMS host or an RTU. A request to download an algorithm file is sent to the algorithm downloader (ADPROC) and the program ALGDWN waits for a response to that request. ADPROC causes the algorithm compiler to compile the algorithm and then passes the ALG p-code to the execution platform. If any errors happen during the download or compile, an error code is returned to ALGDWN. See the example below:
The form of the command is:

```
$ ALGDWN
Enter algorithm name: filename
```

Note that the algorithm filename does not have the ".ALG" extension specified.

```
$ ! Define the ALGDWN as a VMS/MISER command
$ algdwn:==$mnet$exe:algdwn
$ algdwn
Enter algorithm name: victor
Download succeeded
$ ! Specify a bad algorithm (does not exist ...)
$ algdwn
Enter algorithm name: jerabek
Download failed
$
```

A.2 ALGDEL - Algorithm Delete

ALGDEL deletes the requested VCL algorithm from inside the specified execution platform. The execution platform can be a VMS host or an RTU. A request to delete an algorithm file is sent to the algorithms execution platform (AXPROC or an RTU) and the program ALGDEL waits for a response to that request. AXPROC stops and deletes all references to the specified algorithm. If any errors happen during the delete, an error code is returned to ALGDEL. The algorithm file, which is stored in MNET$ALG:*.alg, is not deleted from disk when it is deleted from the execution platform memory. See the example below:
The form of the command is:

```
$ ALGDEL
Enter algorithm name: filename
```

Note that the algorithm filename does not have the ".ALG" extension specified

```
$ ! Define the ALGDEL as a VMS/MISER command
$ algdel:==$mnet$exe:algdel
$ algdel
Enter algorithm name to delete: victor
Delete in execution unit processor succeeded
$ dir mnet$alg:victor.alg

Directory $SITE:[ALG]

VICTOR.ALG;1              10/18       12-APR-2000 08:32:50.49

$ ! Specify a bad algorithm (does not exist ...)
$ algdel
Enter algorithm name to delete: jerabek
Delete in execution unit processor failed
$
```

A.3 ALGCMP - Algorithm Compiler

ALGCMP compiles a VCL algorithm so it can be loaded into an execution platform. The execution platform can be a VMS host or an RTU. ALGCMP takes the results of the VCL draw program and converts it into an ALG p-code (execution platform machine code) which can be executed by any execution platform. ALGCMP is used by ADPROC, ALGDWN, and VCL to compile ALG files. See the example below:
The form of the command is:

```
$ ALGCMP filename
```

or

```
$ ALGCMP [-q|-v|-d|-s] -i filename -o outfilename -e errorfilename
```

The ALGCMP switches are:

-q or -Q

ALGCMP is quiet when compiling

-v or -V

ALGCMP is noisy (verbose) when compiling

-d or -D

ALGCMP compiles the ALG file with debug (RPT) instructions

-s or -S

ALGCMP does a syntax check only

-o filename or -O filename

ALGCMP uses "filename" as output filename for p-code output

-l filename or -L filename

ALGCMP uses "filename" as output filename for assembly code listing

-e filename or -E filename

ALGCMP uses "filename" as output filename for compiler errors

An example of compiling an ALG file is shown below:

```
$ ! Define the ALGCMP as a VMS/MISER command
$ algcmp:==$mnet$exe:algcmp
$ algcmp mnet$alg:victor.alg
Compiling algorithm MNET$ALG:VICTOR.ALG
Algorithm MNET$ALG:VICTOR.ALG passed syntax check
Algorithm MNET$ALG:VICTOR.ALG successfully compiled

$ ! Specify a bad algorithm (does not exist ...)
$ algcmp mnet$alg:jerabek.alg
Error opening or reading ALG file MNET$ALG:JERABEK.ALG
Unable to open input ALG file: MNET$ALG:JERABEK.ALG
```

A.4 ALGDIAG - Algorithm Diagnostic

ALGDIAG is used to do limited performance measurements on algorithms running in the VMS host AXPROC execution platform. ALGDIAG and AXPROC measure the following items inside the execution platform:

Number of Siginificant Events executed
DDS message Counters
DDS dtimer table entry count and size
Algorithm Reference Counters
Algorithm Calculation Reference Counters
Algorithm Variable Reference Counters
Algorithm Point Reference Counters
The Number of Point records visited and the Number of Points total referenced

The DDS message Counters are:

DDS_COS_EVENT	COS Packets sent by MEMIO to AXPROC.
DDS_TIMER_EVENT	Timer expiration events sent to AXPROC by the MISER Timer task.
DDS_VARIABLE_EVENT	Variable COS packets sent by AXPROC to itself.
DDS_MSG_EVENT	Request packets from ADPROC or VCL that cause AXPROC to perform various functions such as Download ALG, Run ALG with DEBUG, etc.
Unknown DDS EVENT	Unknown or bad DDS packets.

ALGDIAG has the following command parameters:

CLEAR	Clear all AXPROC counters
START	START all AXPROC counters counting
STOP	STOP all AXPROC counters
DUMP	DUMP all AXPROC counters
BRIEF	DUMP all non-zero AXPROC counters
LOGGING	Toggle the logging flag. When logging enabled, AXPROC puts verbose messaging in the MISER error logfile file diagnostics purposes. By default, when MISER is started the logging flag is disabled.

An example of usage of ALGDIAG follows:

```
$ ! Define the ALGDIAG as a VMS/MISER command
$ ALGDIAG:==$mnet$exe:algdiag
$ algdiag
Enter diag command (CLEAR,START,STOP,DUMP,BRIEF,LOGGING): CLEAR
Enter execution unit (ie. hsqvsa::2:1 ): SIPVSA::
DIAG command CLEAR succeeded
$ algdiag
```

```
Enter diag command (CLEAR,START,STOP,DUMP,BRIEF,LOGGING): START
Enter execution unit (ie. hsqvsa::2:1 ): SIPVSA::
DIAG command START succeeded
$ algdiag
Enter diag command (CLEAR,START,STOP,DUMP,BRIEF,LOGGING): STOP
Enter execution unit (ie. hsqvsa::2:1 ): SIPVSA::
DIAG command STOP succeeded
$ algdiag
Enter diag command (CLEAR,START,STOP,DUMP,BRIEF,LOGGING): DUMP
Enter execution unit (ie. hsqvsa::2:1 ): SIPVSA::
DIAG command DUMP succeeded
```

The result of a dump (or BRIEF dump) are written in a file mnet$alg:axproc.dump. Each time you do a DUMP you get a new file version in the MNET$ALG directory.

```
PERFMON Counters

 start time: 26-Jul-2000 16:47:45.759 (DST), stop time: 26-Jul-2000
16:52:17.275 (DST)

 Siginificant Events: 2180

PERFMON DDS message Counters

 DDS_COS_EVENT: 35
 DDS_TIMER_EVENT: 181
 DDS_VARIABLE_EVENT: 1963
 DDS_MSG_EVENT: 1
 Unknown DDS EVENT: 0

PERFMON dtimer table entry count and size

 PERFMON dtimer entry count = 110
 PERFMON dtimer entry size  = 68

PERFMON Algorithm Reference Counters

Algorithm: A_RW_FLW, count: 867
Algorithm: D_MIMIC_BOARD, count: 2
Algorithm: D_PEER, count: 3
Algorithm: E_BWDLY, count: 272
Algorithm: FILT_RUNTIM_REQ, count: 30
Algorithm: G_FILT_RUNT_RES, count: 30
Algorithm: K_LOAD, count: 1297
Algorithm: RW_DETSP, count: 270
Algorithm: TEST_PEER_GNT, count: 18
Algorithm: WWLVL_AVG, count: 5

PERFMON Algorithm Calculation Reference Counters

Calculation A_RW_FLW(274): run: 379, hlt: 379
Calculation A_RW_FLW(304): run: 163, hlt: 163
```

```
Calculation A_RW_FLW(322): run: 163, hlt: 54
Calculation A_RW_FLW(450): run: 108, hlt: 108
Calculation A_RW_FLW(622): run: 54, hlt: 54
Calculation D_MIMIC_BOARD(36): run: 1, hlt: 1
Calculation D_MIMIC_BOARD(51): run: 1, hlt: 1
Calculation D_PEER(45): run: 1, hlt: 1
Calculation D_PEER(225): run: 2, hlt: 2
Calculation E_BWDLY(0): run: 16, hlt: 16
Calculation E_BWDLY(14): run: 16, hlt: 12
Calculation E_BWDLY(63): run: 16, hlt: 16
Calculation E_BWDLY(83): run: 12, hlt: 0
Calculation E_BWDLY(124): run: 16, hlt: 16
Calculation E_BWDLY(138): run: 16, hlt: 12
Calculation E_BWDLY(187): run: 16, hlt: 16
Calculation E_BWDLY(207): run: 12, hlt: 0
Calculation E_BWDLY(248): run: 16, hlt: 16
Calculation E_BWDLY(262): run: 16, hlt: 12
Calculation E_BWDLY(311): run: 16, hlt: 16
Calculation E_BWDLY(331): run: 12, hlt: 0
Calculation E_BWDLY(372): run: 16, hlt: 16
Calculation E_BWDLY(386): run: 16, hlt: 12
Calculation E_BWDLY(435): run: 16, hlt: 16
Calculation E_BWDLY(455): run: 12, hlt: 0
Calculation E_BWDLY(496): run: 32, hlt: 32
Calculation FILT_RUNTIM_REQ(42): run: 5, hlt: 5
Calculation FILT_RUNTIM_REQ(56): run: 5, hlt: 5
Calculation FILT_RUNTIM_REQ(70): run: 5, hlt: 5
Calculation FILT_RUNTIM_REQ(84): run: 5, hlt: 5
Calculation FILT_RUNTIM_REQ(98): run: 5, hlt: 5
Calculation FILT_RUNTIM_REQ(112): run: 5, hlt: 5
Calculation G_FILT_RUNT_RES(13): run: 5, hlt: 0
Calculation G_FILT_RUNT_RES(58): run: 5, hlt: 0
Calculation G_FILT_RUNT_RES(119): run: 5, hlt: 0
Calculation G_FILT_RUNT_RES(148): run: 5, hlt: 0
Calculation G_FILT_RUNT_RES(193): run: 5, hlt: 0
Calculation G_FILT_RUNT_RES(254): run: 5, hlt: 0
Calculation K_LOAD(90): run: 205, hlt: 205
Calculation K_LOAD(108): run: 205, hlt: 68
Calculation K_LOAD(542): run: 477, hlt: 477
Calculation K_LOAD(602): run: 205, hlt: 205
Calculation K_LOAD(620): run: 205, hlt: 68
Calculation RW_DETSP(139): run: 54, hlt: 54
Calculation RW_DETSP(171): run: 108, hlt: 108
Calculation RW_DETSP(189): run: 108, hlt: 108
Calculation TEST_PEER_GNT(0): run: 6, hlt: 6
Calculation TEST_PEER_GNT(37): run: 4, hlt: 1
Calculation TEST_PEER_GNT(54): run: 5, hlt: 5
Calculation TEST_PEER_GNT(92): run: 3, hlt: 0
Calculation WWLVL_AVG(0): run: 2, hlt: 2
Calculation WWLVL_AVG(34): run: 1, hlt: 1
Calculation WWLVL_AVG(56): run: 1, hlt: 1
Calculation WWLVL_AVG(78): run: 1, hlt: 1
```

```
PERFMON Algorithm Variable Reference Counters

Algorithm A_RW_FLW variable A_RW_FLW(12): 163
Algorithm A_RW_FLW variable A_RW_FLW(14): 216
Algorithm A_RW_FLW variable A_RW_FLW(15): 163
Algorithm A_RW_FLW variable A_RW_FLW(20): 54
Algorithm A_RW_FLW variable A_RW_FLW(22): 54
Algorithm A_RW_FLW variable A_RW_FLW(24): 27
Algorithm A_RW_FLW variable A_RW_FLW(27): 27
Algorithm E_BWDLY variable E_BWDLY(0): 16
Algorithm E_BWDLY variable E_BWDLY(1): 16
Algorithm E_BWDLY variable E_BWDLY(5): 12
Algorithm E_BWDLY variable E_BWDLY(6): 16
Algorithm E_BWDLY variable E_BWDLY(10): 12
Algorithm E_BWDLY variable E_BWDLY(11): 16
Algorithm E_BWDLY variable E_BWDLY(15): 12
Algorithm E_BWDLY variable E_BWDLY(16): 16
Algorithm E_BWDLY variable E_BWDLY(20): 12
Algorithm E_BWDLY variable E_BWDLY(22): 16
Algorithm K_LOAD variable K_LOAD(5): 205
Algorithm K_LOAD variable K_LOAD(6): 205
Algorithm K_LOAD variable K_LOAD(25): 272
Algorithm K_LOAD variable K_LOAD(28): 205
Algorithm RW_DETSP variable RW_DETSP(8): 54
Algorithm RW_DETSP variable RW_DETSP(9): 54
Algorithm RW_DETSP variable RW_DETSP(11): 54
Algorithm RW_DETSP variable RW_DETSP(13): 27
Algorithm RW_DETSP variable RW_DETSP(16): 27
Algorithm TEST_PEER_GNT variable TEST_PEER_GNT(0): 4
Algorithm TEST_PEER_GNT variable TEST_PEER_GNT(2): 2
Algorithm TEST_PEER_GNT variable TEST_PEER_GNT(3): 3
Algorithm TEST_PEER_GNT variable TEST_PEER_GNT(5): 1
Algorithm WWLVL_AVG variable WWLVL_AVG(2): 1
Algorithm WWLVL_AVG variable WWLVL_AVG(3): 1

PERFMON Point Reference Counters

Point AV-RW-VPOS-SP, input: 0, output: 54
Point C3-WW-T2HI-LIT, input: 0, output: 1
Point C3-WW-T2LO-LIT, input: 0, output: 1
Point AV-RW-DET-SP, input: 0, output: 108
Point FD-FL09-BW-GNT, input: 0, output: 2
Point DV-FL09-BW-GNT, input: 0, output: 1
Point AV-RW-FLW-SP, input: 0, output: 54


7 Point records visited of 14435 points total in system

PERFMON Counters end
```