ELLIOTT 900 SERIES SIMULATOR

SIR DEMONSTRATION PROGRAMS.

DIRECTORY 903SIR

DEMO1.DAT: Assembles and runs a simple test program to demonstrate the simulator's MONITOR command. The program initializes a vector to the values 0 to 7. The MONITOR command intercepts each update to the vector.

DEMO2.DAT: Assembles and runs a simple test program to demonstrate the simulator's TRACEINTERRUPT facility. Note that 0 instructions cannot be interrupted, and the relatively slow execution of the level 4 program when being traced at levels 1, 2 and 3.

DEMO3.DAT: Demonstrates the use of 903 1-pass SIR to produce relocatable tapes, including the use of patching to absolute addresses and to global symbols. Two short programs are assembled, the output of each being directed to DEMO3.RLB and printed, which produces a symbolic representation of the RLB code. (See later section for a full describe of the RLB code).

DEMO4.DAT: a short program to illustrate the instructions used to operate the paper tape station in a polling rather than blocking fashion. The program reads and echoes input characters up until a return character, when it reports "Y" to signal the program worked as expected.

DEMO5.DAT: a demonstration of single length fraction arithmetic using the routines in SGLPAK.

DEMO6.DAT: a demonstration of floating point arithmetic using the routines in FLTPAK (also uses SGLPAK).

DEMO7.DAT: a demonstration of floating point arithmetic double length fraction arithmetic using the routines in DBLPAK (also uses SGLPAK).

DEMO8.DAT: a demonstration of SHELLSORT, sorting the periodic table of elements by chemical symbol, atomic number and atomic weight (also uses SGLPAK).

DEMO9.DAT: illustrates use of the MONITOR subroutine (on which the simulator MONITOR command is based). Monitoring is applied to the Shell Sort demonstration program used in DEMO8. The program is assembled as normal. MONITOR is assembled as a separate program, starting in location 4096, by re-entering SIR at 8 to read an option and patch as a new program, then at 9 to read the MONITOR SIR source. MONITOR is then entered at 0; (i.e., 4096) to read in a parameter tape of locations that are to be monitored.

DEMO10.DAT shows how to build the 903 1-Pass SIR assembler. A "high store" version is assembled using the assembler in low store and vice versa. The same source code is used for both with a different initial patch and option setting. Note that for the high store version, the assembler dictionary is located below the program. The initial block "LOLITS" which references all the literals that appear in the SIR RLB loader is an important contrivance: it ensures these literals are output immediately after the instruction blocks of loader so that when the assembler is requested to punch an RLB loader tape, the code and literals can be copied as a single unit.

The low store assembler is dumped out to paper tape using the T22-23 utility program and the tape checked using the C.4 utility from the 903UTILITIES directory.

For the assembler in high store a version of T22-23 is required that can fit between the low and high store assemblers. T.22 (QSCBOUT) and T.23 (QSCBIN) were reverse engineered from the normal T22-23 that runs in high store. QT2OUT was used to print out T.22 in T.2 format and the simulator DISPLAY command used to print out T.23. (Note: QSCBIN/QSCBOUT are believed to be the names by which SIR versions of T.22 and T.23 were known within Elliotts).

An alternative route would be to use the ACD 2-pass assembler, noting this has a different binary loader than that produced by T22-23 so the output tape will be different, although the loaded image will be identical.

DEMO11.DAT: This shows the use of QCHECK utility to monitor the execution of the SHELLSORT demonstration (DEMO8.DAT). QCHECK is set to print the contents of various data structures at specific monitoring points in the program.

DEMO12A.DAT: the demonstration illustrates the use of QPLOT Issue 1, the graph-plotting library. The program begins by initializing QPLOT: x1, y1, x2, y2 are initialized then the pen raised and moved to beyond the maximum width of a plotter to find the left margin. Various lines are then draw followed by characters. Note that characters drawn centred are drawn twice the size of those drawn on a line – this is a bug. Finally the QPLOT character tables are patched to add the characters (, / and) to those provided by default.

DEMO12B.DAT is essentially the same program as DEMO12A run with a modified QPLOT due to Terry Froggatt: QPLOT(TJF1). This version of is an exact replacement for QPLOT Issue 1, except for taking less store, and therefore the locations at which the additional characters are plugged into the QPLOT character table are different. It is also possible to change the orientation of the axes: if QPLOT-1 is set to +1 it behaves as QPLOT Issue 1 with the Y axis horizontal relative to the plotter and the X axis vertical; if QPLOT-1 is set to +0 the X axis is horizontal and the Y axis vertical.

DEMO12C is yet another version of the same program, using a second improved QPLOT from Terry Froggatt: QPLOT(TJF2). This contains all the space savings and improvements of the TJF1 version and additionally uses a more compact encoding for characters (details are given in the source code). Further it can draw the characters /, \, (,), [,], ‘, ’ and ^. The error handling has also changed – rather than waiting for re-entry at 9 after typing an error message, the user should input any character on the teleprinter to continue.

DEMO13.DAT: this program illustrates the use of QLPOUT, the line printer library routine and QCARDIN the card reader library routine.

It produces a blank line, the line “A1”, fifteen blank lines and then demonstrates how buffer overflow is handled by printing the letter A 512 times.

The program then starts reading cards character-by-character using QCINCH. Note how QCINCH+2 is monitored to detect "end of card" and send a newline character to the printer.

When all the cards are read, a "hopper empty" non-recoverable error is signalled via QERROR.

Note the interrupt handling system. Elliotts would have supplied code similar to this as the SSYS2 package for the particular installation. The code sets up a priority level 2 interrupt handler (QL2ENT) and then terminates to level 4 before commencing printing.

The level 2 interrupt routine first polls the 4100 interface matching unit to determine which devices have raised an interrupt or attention. For each interrupting device control is then transferred to device specific code, e.g., QLPT2 for line printer interrupts and QPLT3 for line printer attentions (these labels are part of QLPOUT). QLPT2 then reads the line printer status word to determine if a line can be output (full line available) or not (line incomplete). Control then passes back to the level 2 interrupt handler QLEV2 to check for further interrupts on the printer or other devices before terminating back to the level 4 main program. Similar entry points QCARD2 and QCARD3 are defined for the card reader.

DEMO14.DAT shows the use of the SIR magnetic tape libraries. Two scratch tapes are created using MTINIT (see 903 Utilities chapter). The first of these is then filled with sequences of records separated by label blocks. The records are written using QMROUT and the labels use QMWRITE. Note how the file table for handler 0 is manipulated to set the different lengths for the record buffer and the label buffer and the use of QMREND to ensure all records are output to tape before label blocks are written. The first tape is then rewound and second tape opened. The first tape is scanned using QMRIN and the records read copied to the second tape by QMROUT. The scanning process skips label blocks (which will include the header and EOF blocks), whereas on the second tape the records are written out as one sequential file. Note how the scanning process resets the record marker when starting each sub-sequence. Also note that the process relies on the fact the first block encountered will be a label block – i.e., the header, and that skipping over the final label block (i.e., EOF) will make the next attempt to read a record fail with a "block not found" error, which is treated as signifying end of tape. To verify the scanning process a simple data check is made to ensure the scanned buffers contain the expected content. When all the records from the first tape have been scanned the second tape is finalized with QMREND and both tapes rewound. QMSTAT is used to poll for completion of the rewinding process. Finally MLIST (see 903 Utilities chapter) is used to print a summary of each tape. The second tape is shorter as it omits the label blocks and the fragmentation of the file into sub-files.

ELLIOTT 900 SERIES SIMULATOR

System Files.

DBLPAK.900: the group of standard routines for double length fraction arithmetic, I/O and mathematical functions. (SIR source code).

Contains:

"903 QDLA, ISS 3 Copy 502"
"903 QDASQRT, (B106A) ISS 1 Copy 502"
"903 QDASIN, (B104A) ISS 1 Copy 259"
"903 QDAATAN, (B105A) ISS 1 Copy 502"

FLTPAK.900: the group of standard routines for floating point arithmetic, I/O and mathematical functions. (SIR source code).

Contains:

"903 QF, ISS 2 Copy 502"
"903 QFIN/OUT, ISS 2 21.2.72 Copy 259"
"903 QFMATH, ISS 2 Copy 502"

MONITOR(ISS1).900: "903 MONITOR ISS 1 Copy 502”:
The MONITOR utility (SIR source code).

MTBLOC(ISS4).900: "MTBLOC (SIR) ISS 4 Copy 502":
Magnetic tape routines (QMOPEN, QMCLOSE, QMREAD, QMFIND, QMWRITE, QMSTAT).

MTREC(ISS3).900: "903 MTREC ISS 3 Copy 502":
Magnetic tape record handling routines (QMRIN, QMROUT, QMREND).

SGLPAK.900: the group of standard routines for integer and single length fraction I/O and single length fraction mathematical functions. (SIR source code).

Contains:

"903 SSYS 1, ISS 1 Copy 259"
"903 QATAN B10 ISSUE 1, COPY 29/5/68 Copy 259"
"903 QEXP, ISS 1", "Copy 502"
"903 QIN 1, ISS 3 Copy 259"
"903 QLN (B1), ISS 2 Copy 259"
"QOUT 1 ISS 4, 10.2.72 Copy 259"
"903 QSIN, (B4) ISS 1 Copy 259"
"903 QSQRT, (B6) ISSUE 1 Copy 259"

SIR(ISS6).900 – "SIR SOURCE, ISSUE 6 – 8 Tapes in One":
source code for the 903 1-Pass SIR assembler, issue 6.

SHELLSORT(ISS3).900 - "900 SHELLSORT, ISS 3 Copy 502"
SIR source of Shell sort routine.

SIR(ISS6).BIN – "900 SIR (SCB), ISS 6 (10.12.71) Copy 502"

SIR(ISS6)(5500).BIN: issue 6 SIR assembled to reside in high store, starting at location 5500.

SIR(ISS6)(512).BIN: issue 6 SIR assembled to occupy low store starting at location 512.

SSYS1(ISS1).900 – "903 SSYS 1, ISS 1 Copy 259"
SIR Systems Tape 1.

QCARDIN(ISS1).900 - "903 CARDIN ISS1 Copy 259":
QCARDIN/QCINCH SIR source.

QLPOUT(ISS1).900 - "903 QLPOUT ISS1 Copy 259":
QLPOUT SIR source.

QPLOT(ISS1).900 – "903 QPLOT ISS 1 Copy 502":
Graph plotting routines.

QPLOT(TJF1).900 - "QPLOT 16/5/11":
Terry Froggatt's first improved QPLOT (see DEMO12B).

QPLOT(TJF2).900 - "QPLOT 4/11/12":
Terry Froggatt's second improved QPLOT (See DEMO12C).

DIRECTORY ACDSIR

SIR1PASS.BIN – "1-PASS SIR 24/3/71":
Binary paper tape for Elliott’s Airborne Computing Division’s 1-Pass SIR Assembler.

SIR2PASS.BIN - "2-PASS SIR 24/3/71":
Binary paper tape for Elliott’s Airborne Computing Division’s 2-Pass SIR Assembler.

There are ten demos:

DEMO1.DAT: Assembles and runs ALGCON, a utility for converting between

903 ALGOL written in 903 telecode
920 ALGOL written in 920 telecode
503 ALGOL written in 920 telecode.

The entry points to ALGCON are:

8. Read and punch first tape, for 903 ALGOL output.
9. Read and punch first tape, for 920 ALGOL output.
10. Read and punch first tape, for 503 ALGOL output.
11. Read and punch subsequent tapes.
12. Punch the final halt code.

If you run it on a 920A it assumes input mode 1. If you run it on anything else it assumes mode 3.

The source file ALGCON_T.903 is derived from a tape labelled "ALGCON 8/9/69" supplied by Terry Froggatt. The program is assembled and then run to translate the ALGOL matrix algebra package (ALMAT.903) from 903 to 920 and 503 ALGOL formats, and, on a 920A converting from 920 to 903 formats.

DEMO2.DAT: Assembles and tests the sources for the 920 ALGOL interpreter. This demonstration illustrates how a program consisting of several tapes can be loaded, and, moreover, the assembler can cope with different sections being in different telecodes (the inline sections will be in 900 telecode).

DEMO3.DAT is a simple program taken from the ACD 903/920 SIR manual to illustrate the use of the 1-pass SIR system. Note that the entry points are specified as octal numbers and the handling of 920 telecode.

DEMO4.DAT is an example program taken from the ACD 903/920 SIR Subroutines manual to illustrate the telecode I/O routines. The program scans input SIR code and prints it out with all comments removed. Note the ability to work with mixed 920 and ACD telecode tapes. Note also the ACD convention of using two global labels for subroutine entry, one for the link address, one for the entry point (e.g., CHIPL, CHIPE respectively). This is to allow the links to be stored in separate data areas from the program.

DEMO5.DAT is the code of DEMO4 linked to an alternative version of the telecode i/o routes which link to the i/o routines of the 1-pass assembler which must therefore be in store at the same time.

DEMO6.DAT is derived from 903 SIR DEMO6 to illustrate the use of the ACD version of the single length fraction i/o and mathematical functions. There are two versions of several of the mathematical functions. Those in QSMATH are a repackaging of those from 903 SIR included in SGLPAK. Those in OPTIMATH are alternative implementations optimised for accuracy and speed. A summary of the entry points and parameters for these routines are included in the file. As with the i/o routines introduced in DEMO5 the L/E convention for link address and entry points is used. Note (1) that compared to 903 SIR QIN1, ACD SIR QIN1 does not support scaling, nor can integers or fractions be distinguished; (2) that QOUT1 produces two leading spaces when printing in the "same line" style; (3) differences in the data: ACD SIR QIN1 only accepts signed integers and fractions, and fractions must be in the form accepted by the assembler (i.e., no leading 0).

DEMO7.DAT is derived from 903SIR DEMO7 to illustrate the use of the ACD version of double length arithmetic. A summary of the entry points, parameters and error codes for these routines are included in the file. As with the i/o routines and single length fraction arithmetic routines the L/E convention for link address and entry point is used.

DEMO8.DAT is derived from 903SIR DEMO5 to illustrate the use of the ACD versions of floating point arithmetic. A summary of entry points, parameters and error codes for these routines are included in the file.

DEMO9.DAT illustrates the use of further ACD input/output routines: INTIP to read integers, ADIP to read addresses and address ranges, IDIP to read SIR identifiers, AMIP to read in SIR-like constants and instructions, SLOP to output words in various formats, ADOP to output addresses and address ranges. A summary of how to use each routine is included in the program source. Note heavy use of the L/E convention for routine link and entry addresses, and a convention on input routines to enable a call to be made either with the first character of the item in the location with label CHIP, or to call the CHIP routine to read the character.

DEMO10.DAT is a trivial program showing the use of the CHIOL routine to punch legible characters to the paper tape punch.

ELLIOTT 900 SERIES SIMULATOR

Other files.

II_A.RAW, II_B.RAW, II_C.RAW, II_D.RAW, II_E.RAW, LOADER.RAW – source for 920 ALGOL interpreter and loader, tapes labelled "ALGOL TAPE II A 19/5/69, Telecode" ACD, "ALGOL TAPE II B 19/5/69, Telecode" ACD, "ALGOL TAPE II C 19/5/69, Telecode" ACD, "ALGOL TAPE II D 19/5/69, Telecode" ACD, "ALGOL TAPE II E 19/5/69, Telecode" ACD, "ALGOL LOADER 19/5/69, Telecode" ACD".

AMIP.900	- AMEND utility input routine (13/3/74)
ADIP.900	- address input routine (13/3/74)
ADOP.900	- address output routine (13/3/74)
AMIP.900	- AMEND input routine (13/3/74)
CHIOL.900	– telecode and legible tape i/o routines (22/05/72)
CHIOSIR1.900	– telecode i/o routines in 1-pass SIR (24/3/71)
IDIP.900	- identifier input routine (8/3/74)
INTIP.900	- integer input routine (8/3/74)
QDLA.900	- double length interpreter (15/12/69).
OPTIMATH.900	- optimised single length mathematical functions (1/12/71)
QDMATH.900	- double length mathematical functions (15/12/69)
QF.900	- floating point interpreter (17/6/71).
QFIO.900	- floating point i/o routines (17/6/71).
QFMATH.900	- floating point mathematical functions (20/12/69)
QSIO.900	– single length input output routines (15/12/69)
QSMATH.900	– single length mathematical functions (15/12/69).