ELLIOTT 900 SERIES SIMULATOR

SIR ASSEMBLERS.

SIR (Symbolic Input Routine) is the machine assembler for the Elliott 900 series. It was derived from an earlier assembler called "T2" and a derivative version called "MASIR" (Macro SIR) was provided for the later machines in the range.

SIR is a simple assembler with support for labels, relative addresses and literal operands. There are no macro facilities.

There were several SIR assemblers, some working in a single pass and permitting "load and go" style operations, others operated in two passes and producing pure binary paper tapes suitable for loading under initial instructions. The single pass assemblers also provide the option to translate SIR code to relocatable binary format, for use as code libraries.

Three SIR assemblers are included with the simulator: 903 1-pass SIR Issue 6, ACD 1-pass SIR and ACD 2-pass SIR, the last two supplied by Terry Froggatt.

The information provided below is derived from "903/920 SIR ACD Library Book No. 23, Copy 23 Amendment No. 3" and the Elliott 900 Technical Manual, Volume 2.

Summary of SIR statements.

The general form of a SIR program consists of optional assembly options introduced by a *, followed by one of more blocks, terminated by a %. Blocks consist of a global identifier list in square brackets followed by statements representing successive words of memory.

If there is only one block in a program and no global labels are to be declared, the global identifier list can be omitted, or be empty (as in "[]"). Unless declared to be "global" labels can only be referenced in the block in which they are introduced.

Statements can be either an instruction or a constant (data word).

ELLIOTT 900 SERIES SIMULATOR

Instructions take the form of an operand field comprising optional / to indicate B-modification then a numeric function code, in the range 0-15, followed by an address field, as in "/4 BASE". Statements can be preceded and followed by comments in round brackets.

Statements can be labelled with an identifier, as in "START 8 MAIN (ENTRY POINT)". In addition to identifiers introduced as labels, identifiers can be given a value by assignment as in "END=99".

i. IDENTIFIER Group of up to six letters or numbers starting with a letter.

ii. GLOBAL IDENTIFIER LIST [ ] e.g. [CAT DOG "RAT]
CAT and DOG are Global, RAT is Sub-Global

iii. CONSTANTS

a. <u>Integer or fraction</u>	+ or - e.g. -71032, +0.475, and in 2-pass SIR only +.012/2 (=.012x2^2=.048), -12.63/-4(= -12.63x2^-4)
b. <u>Octal groups</u>	& e.g. &77123
c. <u>Alphanumeric groups</u>	\ e.g. \A23

iv. ADDRESSES

a. Absolute — an unsigned integer N < 8192

b. Block relative — N;

(N locations from the start of the current block.)

c. SIR relative — ;+N or ;-N

(N locations before or after the current instruction.)

d. Identified address — Identifier+N or Identifier-N

(N locations before or after the identified location.)

e. Literal address   Constant or quasi instruction

(Any constant as in (iii) above or = followed by an instruction(absolute address only), e.g. =/1 0.)

v. SKIP   >+N

(Skip over N words.)

vi. COMMENT   (THIS IS A COMMENT)

vii. TITLE   ((THIS IS A TITLE)

viii. PATCH   ↑A or ↑A+N or ↑A-N
↑"A or ↑"A+N or ↑"A-N

(Continue assembling at address A, A+N, A-N respectively. " form sets patch address in second store module, i.e., adds 8192)

ix. RESTORE $

(Revert to assembling at next location to be used before first patch encountered, if any.)

x. TRIGGER   <

(When program loaded under initial instructions commence execution at current placing address.)

xi. OBEYED INSTRUCTION   ‘8 A’

(Execute instruction immediately.)

xii. END OF TAPE   Halt code.

xiii. END OF PROGRAM   %

(Locate literals after program, cancel sub-global identifiers, list undeclared Global Identifiers and display a FIRST NEXT message.)

xiv. OPTION *N or *+N

(Where N is the sum of the following for 1-pass SIR:
1 display labels
2 load-and-go operation (otherwise produce relocatable binary tape)
4 Clear store up to the assembler
8 Punch a binary loader tape
16 Continue assembly at location 32
32 Set dictionary below program
64 Check without assembly

and for 2-pass SIR:
1 display labels - decimal
2 list labels - octal
4 punch zeros for skips
8 set store pointer to 8192
16 set store pointer to 8
32 tie off present binary tape with a sumcheck, punch 360 blanks and start new binary tape
Any multiple of 8192 punch a clear-store for that much memory.)

N is an unsigned integer
A is any currently located (i.e., non-literal) address.

Character set.

903 SIR uses the 903 or 900 telecodes only.

ACD SIR can use the 920, 903 or 900 character sets. To detect the code in use, every tape input to ACD SIR must commence with a newline sequence appropriate to telecode.

All these character sets includes upper and lower case letters, but they are regarded as the same. On output all letters are in upper case.

Only characters appearing in the SIR internal code can be used in SIR programs (including in comments).

Miscellaneous.

Integers must be in the range -131071 to +131071, fractions in the range -0.999999 to +0.999999.

-131072 and -1.0 can be input using the octal constant &400000.

Octal groups are stored right justified.

In an alphanumeric group the symbol ↑ (or ^) is translated to newline (&01). Space and tab both translate to space (&00). The symbol % is not allowed. If constants containing these symbols are required an octal group should be used with the SIR internal code values for the symbols represented (↑, ^ = &76/+62, % = &5/+5). The group is terminated once three characters (including any embedded spaces or tabs) are read in. If terminated by a newline, the group is stored left justified, padded with spaces (&00).

Space must be allowed for the loader read in under initial instructions. User programs can use locations: 2-8163 and 8192-16383.

The 2-pass SIR assembler provided requires the first non-blank character to be a carriage return or newline symbol so that it can determine which telecode (920 or 900) is being used.

In 2-pass SIR, comments that start with two brackets are treated as "titles" and printed out as programs are assembled.)

Obeyed instructions are only supported by 1-pass SIR assemblers.

Triggers are only supported by ACD SIR systems.

The ACD 2-pass SIR assembler produces binary in an ACD defined format different to the T22/23 binary format used by Elliotts.