GENERAL PURPOSE MACROGENERATOR

The General Purpose Macrogenerator (GPM) was invented by Christopher Strachey and written up in The Computer Journal (1965) 8 (3): 225-241. doi: 10.1093/comjnl/8.3.225.

There are four versions of GPM available for the Elliott 900 series:

1. GPM10: A version based on the CPL program included in Strachey's paper, found in Terry Froggatt's paper tape collection, author unknown.
2. FRED: Another version based on the CPL code, by an unknown author, probably from Elliott Automation's Airborne Computing Division.
3. An version of GPM10 with improved I/O and error handling written by the author
4. An implementation based on Martin Richard's version of GPM.
5. A version written by Harold Thimbleby when at Queen Elizabeth College as a means to port the ML/1 macrogenerator to the Elliott 905.

GPM10, FRED, GPM

GPM10, FRED and GPM are almost identical, being transcriptions of Strachey's CPL code. The most significant difference is in the "warning characters" used to trigger macroprocessing actions.

A GPM/GPM10 macro call is enclosed in square brackets [...] and contains arguments separated by colon : characters. The arguments are macro expanded as they are read in. To avoid macro expansion text can be enclosed within nested angle brackets <...>. On reaching the closing square bracket at the end of a macro call, the zeroth argument is looked up in the environment of defined macros and macrogeneration continues from the beginning of its value. When the end of this value is reached the expansion of the call is complete and macrogeneration continues from just after the closing square bracket. While a macro call is being expanded, a parameter of the form ?n is replaced by a copy of the nth argument of the current call. The number n is given as a single digit. (Note: it is possible to use characters that follow the digit 9 in the 900 telecode collating sequence to access higher numbered

parameters). Macro definitions are added to the environment as they are encountered and a later definition will therefore hide an earlier one. Macros defined during the evaluation of macro call parameters are removed from the environment once the corresponding macro call is completed, i.e., they are temporary definitions that only apply for the duration of the call.

The following macros are predefined.

[DEF:name:value]
: This causes a macro with the given name and value to be declared.

[UPDATE:name:value]
: This updates a named macro with a new value that will be truncated if necessary.

[BIN:number]
: Convert a decimal number to binary form.

[DEC:value]
: The inverse of BIN - convert a number from binary to decimal form.

[BAR:op:a:b]
: Perform binary arithmetic. Op can be one of _, -, *, Quot (quotient – i.e., divide) or Rem (remainder).

[VAL:name]
: The macro produces as output the value associated with name.

Derived from GPM10, GPM uses the ACD CHIOL library for input/output (buffered input is kinder to the tape reader on a real 903) and has more informative diagnostics, copied from the original CPL in Strachey's Computer Journal paper. The BGPM comment feature has been added (see below). In addition DEF can have a second parameter to provide "padding" allowing UPDATE to bind a value in length up to that of the two arguments combined.

GPM10 ignores runout and erase. All even parity characters are significant, odd parity characters result in a MONITOR report.

GPM uses CHIOL so input must start with a newline sequence: carriage return, erase and runout are ignored. Odd parity and codes outside the ACD character set give rise to a CHIOL error message. A halt code terminates input.

GPM and GPM10 both have 4 entry points:

8: to read and macrogenerate an input tape
9: to continue after stopping on a halt code

10: to punch a halt code sequence on the output tape
11: to read a short tape redefining the GPM special characters.

FRED has the same built in macros as GPM/GPM10 but uses different warning characters as in the following examples:

```
Macro call     *NAME,ARG1,ARG2;
Quoted text:   *DEF,NEW,<*OLD,STUFF;>;
Parameters:    *DEF,+,<*BAR,+,"1,"2;>;
```

FRED does not have a comment facility, nor does its version of DEF accept a second argument as padding. The symbol '.' is used in place of '*' in BAR to denote multiplication.

FRED uses CHIOL and has the same input characteristics as GPM10.

FRED contains a number of errors. A patch "FREDFIX" has been provided, which if loaded immediately after FRED removes the most severe errors.

FRED has three entry points:
8: to read in and process an input paper tape, producing an expanded output tape in 920 telecode
10: to read in and process an input paper tape, producing an expanded output tape in 900 telecode
14: to continue processing a subsequent tape.

If an error is detected an error number is punched in legible form after 18" of blank tape following the immediately preceding output.

GPM10, GPM and FRED give numerical error reports as follows:
1: Unmatched] in definition string
2: Unquoted ? in argument list
3: Impossible character as argument number
4: Not enough arguments supplied in a call
5: Terminator in an impossible place, either internal error or due to a missing]
6: Parity error (GPM10)
7: Undefined macro name
8: Wrong exit. Internal error
9: UPDATE string too long
10: Non-digit in argument for BIN

BGPM

BGPM is a transliteration of a BCPL program of the same name by Martin Richards that can be found on his BCPL web site: http://www.cl.cam.ac.uk/~mr10/BCPL.html.

BGPM has a number of differences in detail from the implementation described in Strachey's paper. Most importantly it uses two stacks in place of the single stack used by GPM and this removes an inelegant feature that arises when a parameter contains a call of the def macro.

A BGPM macro call is enclosed in square brackets [...] and contains arguments separated by backslash \ characters. The arguments are macro expanded as they are read in. To avoid macro expansion text can be enclosed within nested curly brackets {...}. On reaching the closing square bracket at the end of a macro call, the zeroth argument is looked up in the environment of defined macros and macrogeneration continues from the beginning of its value. When the end of this value is reached the expansion of the call is complete and macrogeneration continues from just after the closing square bracket. While a macro call is being expanded, a parameter of the form ^n is replaced by a copy of the n-th argument of the current call (n should be an integer). Macro definitions are added to the environment as they are encountered and a later definition will therefore hide an earlier one. Macros defined during the evaluation of macro call parameters are removed from the environment once the corresponding macro call is completed, i.e., they are temporary definitions that only apply for the duration of the call.

Layout is significant in BGPM unless following a ` (grave). On encountering ` it and all following text up to and including the next newline character are ignored. Then, all following white space (tab, space, newline) characters up to but not including the next non-white space characters are ignored. This feature provides a convenient way to use layout to improve readability and add comments.

The following macros are built in:

[def\name\value\padding]
: This causes a macro with the given name and value to be declared (i.e., as GPM DEF). The padding is optional.

[set\name\value]
: This updates a named macro with a new value which may be

truncated if necessary (i.e., as GPM UPDATE).

[eval\expression]
This evaluate the given integer expression consisting of numbers and the numeric operators *, /, <percent>, + and -. Parentheses may be used for grouping and spaces may appear anywhere except within numbers (replacing BIN, DEC, BAR).

[lquote\]
[rquote\]
These macros expand to the curly left and right brackets Respectively (not in GPM).

[eof]
This macro generates the end of file symbol and can be used to terminate input (not in GPM).

[val\name]
The macro produces as output the value associated with name (i.e., GPM VAL).

The entry points for BGPM are:

8	Normal macrogeneration.
9	Continue after halt code on input tape.
10	As for 8, but with tracing of macro calls to the teleprinter.

BGPM uses CHIOL so has the same input data characteristics as GPM and FRED, in particular the requirement for an initial newline sequence.

TGPM

HGPM is also a transcription of Strachey’s CPL code. It was written by Harold Thimbleby, then a student at Queen Elizabeth College, London as a means to port the ML/1 macroprocessor.

TGPM provides the following built-in macros:

DEF: as in GPM
UPDATE: as in GPM
BAR: takes string arguments for numbers
LEG: tests a number for less than, equal to or greater than zero.
NOTE: outputs a monitor message to the teleprinter.
VAL: as in GPM.

There is no DEC or BIN.

The default warning characters are as in the following:

```
[DEF,TEST,<[>?1<]>].
```

[BAR,+,HEAD1TAIL,--2STUFF] – note that leading text is stripped until a sign or digit is encountered, the an integer is read and any trailing text ignored. Signs (+ or -) are treated as monadic operators, thus – is the same as +.

[LEG,[NUM],<[LT]>,<[EQ]>,<[GT]>] will call the macro LT if the macro NUM evaluates to a number less than 0, EQ if it equals 0, otherwise GT.
[NOTE,helpful message to the operator].

The warning character ! absorbs any following newlines to allow macros to be split over several lines.

All input characters except runout, carriage return and erase are significant, including wrong parity characters.

The entry points are:
- 32: (Re)start macroprocessing
- 33: Output a traceback
- 34: Change warning characters.

Entry at 34 expects a sequence of characters to replace <,]?!>[as warning characters. All characters except runout, return and erase are significant.

DEMO1.DAT: This script builds GPM10 and runs a series of examples taken from Strachey's Computer Journal paper and additional tests devised by Martin Richards. The "Sum" example is subtly different to the version in the paper, correcting an error that prevented it working. The input is in GPM10DATA.900.

DEMO2.DAT: This script builds GPM and runs the same examples as in GPM10.DAT. The input data is in GPMDATA.900.

DEMO3.DAT: This script build BGPM and runs same examples as in GPM10.DAT and further, more demanding, examples taken from Martin Richard's BCPL version of BGPM. The BGPM input is in the file BGPMDATA.900.

DEMO4.DAT: This script builds BGPM and runs a further set of GPM examples, written by Eiiti Wada, an eminent Japanese computer scientist. These illustrate the use of GPM to evaluate a number of recursive functions, including factorials, Ackermann, and to print the words of the song "Twelve Days of Christmas" amongst others. The BGPM input is taken from BGPMWADA.900.

DEMO5.DAT: This is a further set of macros by Eiiti Wada, which solve the Eight Queens Problem. The first set of macros generate the macros needed to generate the solution. Note the patch to allow BGPM to run in a 24K store. The data in in BGPMQUEENS.900.

DEMO6.DAT: This demo shows off TGPM. It starts off with a symbol change then demonstrates several macros taken from Thimbleby's project report, followed by a pass over the functional tests applied to the other macro-generators. Since TGPM has a small stack, some examples have been simplified compared to their use with the other generators. The input is taken from TGPMDATA.900. Note that the original TGPM is not a complete program and has some bugs, so the first step is to edit the source with BOWDLER to turn it into a complete and correct program.

DEMO7.DAT: This demo illustrates FRED. It starts off with a short input tape to show FRED working on simple input and the effect of some of the errors it contains. The FREDFIX patch is applied and then FRED shown to run the same input as GPM10.

SYSTEM FILES

BGPM.900: The source for BGPM, written by the author.

BGPM.BIN: The result of assembling BGPM.900.

BGPMDATA.900: BGPM input tape for DEMO3.DAT

BGPMWADA.900: BGPM input tape for DEMO4.DAT

BGPMQUEENS.900: BGPM input tape for DEMO5.DAT

FRED.900: "FRED 9/2/71" a binary tape from the Terry Froggatt collection.

GPM10.900: "GPM10 9/5/77, 900-Series Telecode", the source for GPM10 from Terry Froggatt's tape.

GPM.900: The source of GPM, written by the author by amending GPM10.900.

TGPM.900: The original source of TGPM, tape THIMBLEBY_11 in the authors collection of tapes copied from the collection of Howard Thimbleby.

TGPMDATA: input tape for DEMO6.