# MACRO LANGUAGE

## MACRO (REFERENCE MANUAL SECTION) COMMAND PROMPTS

### MACRO DIRECTORY INITIALIZATION

**IMF** - This command performes the first half of macro directory file initialization.
**PROCEED** - This command performs the second half of macro file initialization if it immediately follows the "IMF" command.

### MACRO CREATION

**MACRO** - The "MACRO" command must be the first command in every macro. It starts macro recording.
**EOM** - The "EOM" command must be the last command in every maco. It stops macro recording.

### MACRO DELETION

**MDEL** - The "MDEL" command deletes the "named macro"

### MACRO EDITING METHODS

**MEDIT (macro name)** - The "MEDIT" command causes the full screen macro editor to open for the creation of or editting of the named macro.
**MREFRESH** - The "MREFRESH" command refreshes the macro contents in the macro directory file while full screen macro editting is in progress.

### LIST ALL MACROS

**MFL** - The "MFL" command causes all macros to be displayed in their entirety.

### LIST ALL MACRO NAMES

**MFLN** - The "MFLN" command causes a condensed listing of the names of all macros, their creation time/date stamp and the first 40 characters of the first comment line in the macro to be displayed

### LIST A SPECIFIC MACRO

**MFL (macro name)** - This version of the "MFL" command causes all lines comprising the named macro to be listed. If no name is included, all the macros are listed. That mode is useful when outputting all macros to a disk file for backup.

### LIST MACRO COMMENTS

**MFLC (macro name)** - The "MFLC" command causes all leading comment lines of the named macro to be listed.

### LIST MACRO COMMENTS

**MSTAT** - The "MSTAT" command generates a message indicating the number of macros currently on file and the remaining number of empty spaces in the macro directory file. The "MFLN" and "MSTAT" commands are combined into one menu item.

### INDEXING COMMANDS

**SET (I, J, K, L, M, N, ITEST, JTEST, KTEST, LTEST, MTEST** or **NTEST) , i** - This version of the "SET" command causes the register named by the qualifier word ("I", "J","K", "L", "M", "N", "ITEST", "JTEST", "KTEST", "LTEST", "MTEST" or "NTEST") to be set to the value of "i". The numeric value can be positive or negative and is stored as a double precision. There is a short cut to setting these registers. Enter the register name with an equal sign appended, a space or comma (necessary) and the numerical value to which the register is to be set.
**MOVE (I, J, K, L, M, N, ITEST, JTEST, KTEST, LTEST, MTEST** or **NTEST)** - The "MOVE" command "moves" the numeric value stored in the register indicated by the qualifier word ("I", "J","K", "L", "M", "N", "ITEST", "JTEST", "KTEST", "LTEST", "MTEST" or "NTEST") into the accumulator register. ("MOVE" generally moves a value to the accumulator which is also called the X-register). "MOVE" may be used with all named registers as well as these "test" registers.
**STORE (I, J, K, L, M, N, ITEST, JTEST, KTEST, LTEST, MTEST** or **NTEST)** - The "STORE" command causes the number in the accumulator to be stored in the register indicated by the qualifier word ("I", "J","K", "L", "M", "N", "ITEST", "JTEST", "KTEST", "LTEST", "MTEST" or "NTEST"). "STORE" may be used with all named registers.
**INCR (I, J, K, L, M** or **N) , i** - The "INCR" command causes the value in the indicated index register "I", "J","K", "L", "M" or "N") to be incremented by "i", a numeric value which may be positive or negative. If a numeric value is not specified, the default numeric value of 1.0 is used. "INCR" can be used with all named registers.

### BRANCHING COMMANDS

**BP (branch point name)** - The "BP" command defines a branch point having the name "branch point name". The branch point name is entered as a qualifier word. A branch point serves only as a marker and may appear anywhere within a macro. Branch commands refer to a branch point by name indicating that if the branch is taken, the next command processed is the command immediately following the named branch point. (This is similar to a LABEL in a programable calculator program.) Branch point names must begin with one of the 26 characters of the alphabet and may not exceed eight alphanumeric characters in length.
**BRQ (branch point name) , (test name)** - The "BRQ" command specifies that if the current qualifier word of the macro (the qualifier word used in the macro invocation line or the qualifier as defined by the "QSUB DV" command) matches the indicated name test name, then the next command to be processed is the command immediately following the branch point named "branch point name". If no match exists, the "BRQ" command results in no action and macro processing continues sequentially.

### ABOUT LINE COUNTS

In the branching commands which follow, a numeric line count, or "lc", may be used. If a non-zero "lc" has been entered, it is used instead of the

branch point name to determine the location of the next command to be processed when branching takes place. If no entry is made for "lc", then the branch point named "branch point name" is used and the macro is searched for "BP branch point name". The line count "lc" may be positive or negative. It specifies the position of the next command to be processed relative to the position of the branch command when branching takes place. A value of "lc" = -9 indicates a branch to a command which is nine commands back from the branch command. The use of "lc" reduces the need for branch points but makes a macro harder to read. Avoid the use of "lc" unless there is no choice. "lc" is intended for use with macros written using HEXAGON, the old Hughes Aircraft Co. program.

**BRERR (branch point name) , lc** - The "BRERR" command, which stands for Branch on Read ERRor, causes branching to "BP branch point name" if the read error flag has been set. The read error flag is set by an unsuccessful execution of the "ATON" CMD level command. The "ATON" command is used to attempt to convert the first 23 characters read by the CMD level "PREAD" command into a numeric value. After the branching occurs, the read error flag is cleared.
**BPOS (branch point name) , lc** or **IF(X>0) (branch point name) , lc** - The "BPOS" or "IF(X>0)"command causes branching to "BP branch point name" if the value in the accumulator ("X"-register) is positive.
**BNEG (branch point name) , lc** or **IF(X<0) (branch point name) , lc** - The "BNEG" or "IF(X<0)"command causes branching to "BP branch point name" if the value in the accumulator is negative.
**BZE (branch point name) , lc** - The "BZE" or "IF(X=0)" command causes branching to "BP branch point name" if the value in the accumulator is zero.

**IF(X=Y) (branch point name) , lc** - The "IF(X=Y)" command causes branching to "BP branch point name" if the value in the X-register is equal to the value in the Y-register.
**IF(X>Y) (branch point name) , lc** - The "IF(X>Y)"command causes branching to "BP branch point name" if the value in the X-register is greater than the value in the Y-register.
**IF(X<Y) (branch point name) , lc** - The "IF(X<Y)"command causes branching to "BP branch point name" if the value in the X-register is less than the value in the Y-register.
**BRI (branch point name) , lc** - The "BRI" command causes branching to "BP branch point name" if index register "I" equals "ITEST".
**BRJ (branch point name) , lc** - The "BRJ" command causes branching to "BP branch point name" if index register "J" equals "JTEST".
**BRK (branch point name) , lc** - The "BRK" command causes branching to "BP branch point name" if index register "K" equals "KTEST".
**BRL (branch point name) , lc** - The "BRL" command causes branching to "BP branch point name" if index register "L" equals "LTEST".
**BRM (branch point name) , lc** - The "BRM" command causes branching to "BP branch point name" if index register "M" equals "MTEST".
**BRN (branch point name) , lc** - The "BRN" command causes branching to "BP branch point name" if index register "N" equals "NTEST".
**BRDQ (branch point name) , lc** - The "BRDQ" command causes branching to "BP branch point name" if the qualifier in the macro invocation line was not explicitly entered or provided with a "QSUB" command.
**BRDF1 (branch point name) , lc** - The "BRDF1" command causes branching to "BP branch point name" if the numeric word #1 in the macro invocation line was not explicitly entered or provided with an "NSUB DV" command.
**BRDF2 (branch point name) , lc** - The "BRDF2" command causes branching to "BP branch point name" if the numeric word #2 in the macro invocation line was not explicitly entered or provided with an "NSUB DV" command.
**BRDF3 (branch point name) , lc** - The "BRDF3" command causes branching to "BP branch point name" if the numeric word #3 in the macro invocation line was not explicitly entered or provided with an "NSUB DV" command.
**BRDF4 (branch point name) , lc** - The "BRDF4" command causes branching to "BP branch point name" if the numeric word #4 in the macro invocation line was not explicitly entered or provided with an "NSUB DV" command.
**BRDF5 (branch point name) , lc** - The "BRDF5" command causes branching to "BP branch point name" if the numeric word #5 in the macro invocation line was not explicitly entered or provided with an "NSUB DV" command.
**BRU (branch point name) , lc** - The "BRU" command causes branching unconditionally to "BP branch point name".
**BRANCH (branch point name) , i , j , lc** - The "BRANCH" command causes branching to "BP branch point name" if "i"th numeric word in the macro invocation line has the value "j". If no branch point name is given and "lc" is non-zero, then a jump of "lc" lines is performed.

## FLAGS IN BRANCHING

**BRT (branch point name), f1 , f2 , f3 , f4 , f5** - The "BRT" command causes branching to "BP branch point name" only if all specified flags test on.
**BRF (branch point name), f1 , f2 , f3 , f4 , f5** - The "BRF" command causes branching to "BP branch point name" if not all specified flags test on.

## TERMINATION OF EXECUTION

**RETURN** - The "RETURN" command terminates a macro's execution and passes control back to the calling macro if there was one. Macro execution also terminates if the bottom of the macro "EOM" is reached and the preceding command is not a branch command.

## EXTERNAL DATA TRANSFER

**CSUB** - The "CSUB" command replaces the command word of the command to be modified with either the qualifier word issued in the macro invocation line or the qualifier word supplied with a "QSUB DV" command.
**QSUB** - The "QSUB" command replaces the qualifier word of the command to be modified with either the qualifier word issued in the macro invocation line or the qualifier word supplied with a "QSUB DV" command.
**QRSUB** - The "QRSUB" command replaces the qualifier word of the command to be modified with the first eight characters of the last string read by a prompted read via the "PREAD" command described in the CMD section of this manual.
**SSUB** - The "SSUB" command replaces the alphanumeric string of the command to be modified with the alphanumeric string issued in the macro invocation line or the alphanumeric string supplied with an "SSUB DV" command.
**CRSUB** - The "CRSUB" command replaces the command word of the command to be modified with the first eight characters of the last string read by a prompted read via the "PREAD" command described in the CMD section of this manual.

## DEFAULT INPUT DATA

**QSUB DV (default qualifier word value)** - The "QSUB DV" command is used to set a non-blank default value for the qualifier word of the macro invocation line. This default value is used only when the macro invocation line contains no qualifier word.
**SSUB DV (default alphanumeric string word value)** - The "SSUB DV" command is used to set a non-blank default value for the alphanumeric string word of the macro invocation line. This default value is used only when the macro invocation line contains no alphanumeric string input word.
**ACCSUB (register name) , i** - The "ACCSUB" command will substitute the named register for the accumulator during the subsequent "i" valid program commands. The accumulator will remain unchanged, and operations which normally operate upon the accumulator will operate upon the named register. All arithmetic processing commands and "STORE", "CSUB", "QSUB", "PUTR" and "WRITE" are valid for "ACCSUB". If a command invalid for accumulator substitution is encountered, it is processed without the substitution. If "i" is not specified, the default value is taken to be 1.0 .
**NSUB , j , k , A , B, C** - The "NSUB" command causes the numeric value of numeric word "j" of the command to be modified to be replaced by:

$$(A \times vk + B)**C$$

where "vk" is the numeric value of numeric word "k" of the macro invocation line. Both j and k are restricted to values 0, 1, 2, 3, 4 or 5. 0 refers to the accumulator or X-register. Default values are:

A = 1.0
B = 0.0
C = 1.0

**NSUB RA , j , k , l, B, C** - The "NSUB RA" command is equivalent to the "NSUB, j , k , A , B , C" command, except that the numeric value of numeric word "l" (lower case L, not 1) of the macro invocation line is used as the multiplicative constant, "A".
**NSUB RB , j, k , A , m, C** - The "NSUB RB" command is equivalent to the "NSUB, j , k , A , B , C" command, except that the numeric value of numeric word "m" of the macro invocation line is used as the additive constant, "B".
**NSUB RC, j , k , A , B , n** - The "NSUB RC" command is equivalent to the "NSUB , j , k , A , B, C" command, except that the numeric value of numeric word "n" of the macro invocation line is used as the power constant, "C".
**NSUB RAB, j , k , l , m , C** and **NSUB RAC , j , k , l , B , n** and **NSUB RBC , j , k , A , m , n** and **NSUB RABC , j , k , l , m , n** - These commands are combinations of the "RA", "RB" and "RC" NSUB commands.

## DEFAULT NUMERIC VALUES

**NSUB DV, NW1, NW2, NW3, NW4, NW5** - The "NSUB DV" command is used to set default values for the five numeric words of the macro invocation command. They are used to replace values of the numeric words left default or blank on the macro invocation line. NOTES:
1. In all numeric data transfer commands, the second numeric value in any "NSUB" command (except "NSUB DV") is the address in the macro invocation line from which the value to be transferred is to be found.
2. In all numeric data transfer commands, the first numeric value in any "NSUB" command (except "NSUB DV") is the address in the next non-NSUB command to which the value to be transferred will be transferred.
3. In any "NSUB" command, a zero for numeric word j or k always means the accumulator. j = 0 causes transfer into the accumulator register; k = 0 causes transfer from the accumulator register. This is the ONLY case where an "NSUB" command does not modify another command.

## OTHER DATA TRANSFERS

**MOVE NW, i** - The "MOVE NW" command is a special case of the "MOVE" command described in the arithmetic processor command section of the CMD section of this manual. The numeric value of the "i" th numeric word of the macro invocation line is moved into the accumulator. If "i" is zero, the value stored in the index register "I" is used in place of "i". If the integer value of the "I" register is equal to 0.0 or is greater than 5.0, then the integer value of the first numeric word is moved into the accumulator.
**PUTR (register name) , i** - The "PUTR" command causes the numeric value of numeric word "i" of the macro invocation line to be replaced by the value of the number in the named register. If "i" is zero, the value of the index register "I" is used in place of "i". If the value of the "I" register is equal to 0 or if it is greater than 5, then the value stored in the named register is moved into the first numeric word of the macro invocation line.

## MACRO NESTING

**SAVE** - The "SAVE" command causes the current contents of the accumulator (register X) and registers A through H, Y, Z, T, IX, IY, IZ , I, J, ITEST, JTEST, LASTX and LASTIX to be saved.
**RELOAD** - The "RELOAD" command causes contents of the registers to be restored to the values which were saved by the last "SAVE" command issued at this nesting level. "SAVE" and "RELOAD" are operational at each of the 20 macro nesting levels.

## TRACING MACRO EXECUTION

**TRACE ON** - The "TRACE ON" command commences tracing the execution of all non-macro processing commands at the current nesting level. If a macro invokes another macro, that macro will only be traced if it contains a "TRACE ON" command as well.
**TRACE OFF** - This command terminates tracing at the current nesting level.

## SINGLE STEP MACRO EXECUTION

**SSTEP ON** - The "SSTEP ON" command commences single step execution of all commands at the current nesting level. If a macro invokes another macro, that macro will only be single stepped if it contains a "SSTEP ON" command as well. During single stepping, pressing of any key on the keyboard causes the next macro command to be processed. There is an option to stop macro execution.

**SSTEP OFF** - This command terminates single step execution at the current nesting level.

## PAUSING MACRO EXECUTION

**PAUSE** - The "PAUSE" command causes macro execution to be temporarily suspended at the location of the "PAUSE" command. Macro execution is resumed by pressing any key on the keyboard.

**AUTOMATIC MACRO TERMINATION** - By default, when a program command is run from within a macro and when that command's execution results in an error message, the macro from which that command was issued and any macros asssociated with that macro which exist in a

macro nest will be automatically terminated. This protects the user from runaway macros. If this automatic macro termination is not desired, the following command is provided in order that the user can turn "on" or "off" this automatic macro termination feature.
**MACFAIL (ON or OFF or YES or NO** - The "MACFAIL" command, issued from the CMD program level, is used to either enable or disable automatic macro termination in the presence of a program error condition. Issued with no input or with the "?", the current state of automatic macro termination will be displayed. "ON" or "YES" is the program default condition

**RENAMING / COPYING MACROS**
**MRENAME (current macro name) (new macro name)** - The command "MRENAME" requires explicit qualifier word and alphanumeric string input. The new macro named "new macro name" is created and the current macro named "current macro name" is extracted into it. Then the old macro is deleted. .
**MCOPY (current macro name) (new macro name)** - The command "MCOPY" requires explicit qualifier word and alphanumeric string input. The new macro named "new macro name" is created and the current macro named "current macro name" is extracted into it. The old macro is left on file.

**MANIPULATING MACRO LIBRARIES**
**MACSAVE** - The command "MACSAVE" causes the contents of the current macro directory to be saved in the ASCII file MACSAV.DAT. This process erases the current contents of the EDITTEXT.DAT file and the previous contents of the MACSAV.DAT file.
**MACREST** - The command "MACREST" causes the contents of the current MACSAV.DAT file to be loaded into the current macro directory. This process erases the current contents of the EDITTEXT.DAT file and the previous contents of the current macro directory. This command may also be issued via the main program GUI interface menu system.
**THE MACRO DIRECTORY** - All user-created macros are stored (by default) in the directory LIBMAC which resides just below the directory containing the PRG.EXE file. The macro directory file (not to be confused with an operating system disk directory) is the file named MAC.DAT. The macro body records comprising each macro and macro function are contained in up to 999 files with names which range from MAC001.DAT to MAC999.DAT. Whenever a macro name is input as a program command from the CMD level, the appropriate macro is loaded into memory from the macro directory file and then its component commands are executed. Macros are stored in a pre-parsed, semi-compiled format which enhances speed of execution.

**ALTERNATE MACRO DIRECTORIES** - **MACDIR or CHGMAC , (qualifier word)** - By default when the program begins execution, the macro library is located in the directory LIBMAC which sits just below the directory into which the main program was installed and from which the main program runs. The "MACDIR" command is used to change the current macro library directory to the directory named by the first six characters of the "qualifier word". If this directory exists, then only the internal program pointer, pointing to the macro library directory, is changed. If the directory does not yet exist, it is created. After creation of a new macro library directory, a new macro library must be initialized in this new directory via the "IMF" and "PROCEED" commands. The "MACDIR" command makes it possible to have access to as many macro libraries as desired, only limited by available disk space. If "MACDIR" is issued followed by a "?", the name of the current macro library directory will be displayed. The new directory name designated by the "qualifier word" must contain exactly six non-blank characters.

**PERMANENT MACROS** - Permanent macros are provided with the program and stored in the PERMAC directory. In order to list, edit, create or delete a permanent macro, change to the PERMAC directory, perform the macro operation and then, return to the default macro library directory. The example below, shows how to load the manufacturers lens librarys after the program is first installed. To run a permanent macro, simply issue the macro name. The program automatically searches the permanent macro library if the macro name issued is not in the current macro directory.
**LODLENS**

# ADVANCED STUFF (FOR EXPERTS ONLY)

**BUILDING A NEW COMMAND**- **The following commands are almost never needed except if you are writing a macro which is intended to create a new macro or which is intended to have advanced "intelligence". Unless you feel fearless, just ignore what follows.** The following commands were designed to be used both from the CMD level and from within macros. They are described here because their real value comes from their use inside macros. The philosophy behind these commands is as follows:
**CWORD (the command word of the command under construction)** - The command "CWORD" takes qualifier word input. If we were building the command "RTG ALL", the input following "CWORD" would be "RTG".
**QWORD (the qualifier word of the command under construction)** - The command "QWORD" takes qualifier word input. If we were building the command "RTG ALL", the input following "QWORD" would be "ALL".
**STWORD (the alphanumeric string of the command under construction)** - The command "STWORD" takes alphanumeric input. If we were building the command "M, Hello world !", the input following "STWORD" would be "Hello world !".
**N1WORD , i**The command "N1WORD" takes numeric word #1 input. If we were building the command "RTG , 2", the input following "N1WORD " would be "2".
**N2WORD , i** and
**N3WORD , i** and
**N4WORD , i** and
**N5WORD , i**- The commands "N2WORD", "N3WORD", "N4WORD" and "N5WORD" are similar to "N1WORD" and are used for setting up numeric words 2, 3, 4 and 5.
**NEWCMD CLEAR**- The "NEWCMD CLEAR" command clears the internal program storage areas set aside for the command under construction. This command should be used prior to the building of a new command.
**NEWCMD** - The "NEWCMD" command causes the newly-built program command to executed. In summary, if the command "RTG ALL" was to be constructed and executed from within a macro, the macro lines which could perform the task would be:

**CWORD RTG**
**QWORD ALL**
**NEWCMD**