31336 VIA COLINAS, WESTLAKE VILLAGE, CA 91361

SMOKE SIGNAL BROADCASTING

MNEMONIC ASSEMBLER

SA-2

COPYRIGHT ©1979
SMOKE SIGNAL BROADCASTING

COPYRIGHT NOTICE

This entire manual and accompanying software have been copyrighted by Smoke Signal Broadcasting. The reproduction of this document or accompanying software for any reason other than archival or backup purposes for or on the computer for which the original copy was aquired is strictly prohibited.

WARRANTEE INFORMATION

The SSB ASSEMBLER is provided AS IS without warrantee. Reasonable care has been taken to insure that the software operates as described in this manual. If you find a situation in which the assembler does not operate as described, please contact Smoke Signal Broadcasting. We will attempt to correct any errors brought to our attention, but we make no gaurantee to do so.

ASSEMBLER VERSION SELECTION

The disk on which the SSB 6800 Assembler is supplied contains 3 versions of the software. Each version is configured for a corresponding DOS68 system base address. Select a version according to the base address of DOS68 under which the assembler will be running. The versions and their corresponding system base addresses are as follows:

ASMB .$ DOS68 at $6000
ASMBA.$ DOS68 at $A000
ASMBC.$ DOS68 at $C000

TABLE OF CONTENTS

INTRODUCTION

The SSB 6800 Assembler for DOS68 is a fast, versatile, and powerful assembler for the M6800 computer system. The assembler will permit very large source files including multiple source files to be assembled, producing either binary disk file, or ASCII tape object code. A number of assembly-time assembler options are provided. The name of the source file(s) to be assembled, unless otherwise specified, default to a filename bearing a type 1 extension. Likewise, binary output filenames will default to bear type Ø extensions. Since DOS68 initially defines a type Ø extension as .BIN and a type 1 extension as .TXT, extension types and their literal equivalents will be used interchangeably throughout the remainder of this manual. This manual reflects information that is compatible with versions 2.Ø, 2.ØA, and 2.ØC of the assembler.

INVOKING THE ASSEMBLER

The assembler is invoked using the ASMB command, which uses the general syntax of:

ASMB,<INPUT FILE SPEC(S)>,<BINARY FILE SPEC>/SWITCHES

The name of the file(s) to be assembled default to the base name bearing a type 1 extension and the drive currently assigned as the work drive. If <BINARY FILE SPEC> is not included in the command line, the binary file will be assigned the same 'name' as the first source file which is being assembled, but will be assigned a type Ø extension. The default filename extensions can be overridden by simply defining the extension explicitly. That is, if FILE.A is specified, then the FILE.A will be operated upon rather than FILE.TXT or FILE.BIN. Refer to the DOS68 documentation for information regarding extension type definitions, work drive assignments, and the use of the SET command for defining such parameters.

COMMAND OPTION SWITCHES

There are 9 option switches which can be included in the option list. The option list is separated from the file specs by a '/'. Each option code is represented by a single letter, with multiple options specified as a list of unseparated option switches. Below is a list of the available options and their meanings.

B Do not create a binary file on the disk. If this option appears in the list, no binary file will be created even if <BINARY FILE SPEC> has been specified. This is useful while running the assembler and checking for errors before the final program is completed.

G Turn 'NOGEN' on. This causes only one line of code to be printed for each FCB, FDB, or FCC in the assembly listing.

L Turn the assembler 'LIST' mode off. The assembler will normally produce an assembly listing. The L option will suppress this listing, and only error lines (if any exist) will be displayed.

S Suppress the output of the Symbol Table. Normally the assembler will output a sorted symbol table at the end of the assembly. Including the S option will suppress the output of this table.

N Omit line numbers in the assembly listing. The assembler will normally include line numbers in the listing. During program debugging, it is often useful to have each statement numbered. These numbers will correspond directly with the line numbers assigned by the editor during the next edit pass. Error lines will always be output with line numbers even if the N option was not specified. regardless of the status of the N option.

T Assemble the file for an 'S1' formatted object tape. This option allows the production of an 'S1' tape directly from the assembler. For additional tape generation information refer to the tape control provisions described in the section on customizing the assembler.

Y The Y switch permits the user to automatically circumvent the "DELETE BINARY FILE?" conversation and accompanying "Y" response. If a binary object file exists bearing the explicit or default binary file name, then that file will be deleted and an new binary object file bearing the same name will be created.

M This option tells the assembler to include a 5 column margin at the beginning of each line in the listing. This feature is helpful if the printer paper has a left side perforated edge to be removed when the listing must be bound, or if the printer being used does not allow wide enough left side margins. The M option defaults to the off state, causing no margin to be added to the listing.

P This option, if selected, routes the assembled listing to the system printer device. The default state of the 'P' option is off, allowing the assembly output to be sent to the system terminal.

Note that if a NOT, SYM, or GEN option is used with the OPT directive it will override the action of a T, S, or G calling switch respectively, but the L switch will override the LIS option.

EXAMPLES

1. ASMB,TEST
2. ASMB,TEST/LS
3. ASMB,Ø.TEST,1:TEST.$/S
4. ASMB,TEST/BNPM
5. ASMB
6. ASMB,PROGØ.TXT+PROG1.TXT+PROG2.TXT
7. ASMB,1:EQU.SYS+PROG3.TXT+2:PROG4.TXT,PROG.$

The first example would assemble the source file TEST.TXT from the working drive and would check to see if TEST.BIN exists on the the same drive. If TEST.BIN was not found, then the assembler will create TEST.BIN. However, if TEST.BIN existed, the assembler would prompt the operator with:

DELETE BINARY FILE ?

A 'Y' response will allow the assembler to delete the current TEST.BIN, create a new TEST.BIN, and proceed with the assembly process. Any response other than 'Y' will abort the assembler and return control to DOS68. Note: The response to DELETE BINARY FILE? requires only a single character, and after the response is typed, it cannot be changed.

The second example would do the same operation as the first, except the listing and symbol table output would be suppressed.

Example 3 would cause the assembly of TEST.TXT found on drive Ø, and would create a transient command file TEST.$ on drive 1. The symbol table listing would be suppressed.

The fourth example causes the file TEST.TXT found on the working drive to be assembled without generating a binary file on the same drive, no line numbers in the assembly listing, and routing of the assembly listing with a left margin to a printer.

The example shown in line 5 is yet another way to invoke the assembler. If ASMB is entered without file specification(s), then the assembler will begin execution, but will prompt the user for the file specifications as shown below.

FILES:

The user can enter the file spec(s) and options. The capability of entering assembler input information after the assembler has been loaded can be advantageous to single drive as well as multi-drive users. The user can remove the diskette that the assembler resides on, and insert another diskette which contains or may yet contain files specified in the file spec. list. The user has in essence gained the advantage of having an "additional" drive on his system!

Example 6 shows the method for assembling a number of source files from the working drive together, forming a single binary output file.

Successful multiple assembly requires only the last input file to contain an 'END' directive. Note also that the same rules for multiple and undefined symbol detection apply between files as well as within files. In addition, if the name of the binary file is not specified, the binary file will take on the name of the first file given in the file specs. This example would place the binary output in a file named 'PROGØ.BIN on the working drive. The number of files that can be assembled together are limited only to the number of file specs that can be typed on the command line.

The command line of example 7 shows again how multiple files can be assembled, with input files being read from different drives, and a transient command file PROG.$ being the binary output file located on the working drive. Note here how an equate file, for example, can be included in the assembly process.

ROUTING LISTINGS TO A PRINTER

To have the listing printed on a printer rather than appearing on the system terminal, simply precede the the ASMB command with the "P," prefix (see 'P' command in the DOS68 description). This is assuming that the appropriate PRINT.SYS file exists in the operating system, and has been run prior to an attempt to use the printer. For example:

```
DOS: P,ASMB,TEST
```

This would cause the assembly listing of the source file TEST.TXT to be printed on the printer. Using the P option with the ASMB command has the same effect:

```
DOS: ASMB,TEST/P
```

ASSEMBLER DESCRIPTION

The SSB 6800 Assembler was written for maximum flexibility for DOS68 disk system users. As always, flexibility adds complexity and therefore the user is advised to read the following application notes before attempting to use the assembler.

It is assummed that the user is familiar with assembly language and, in particular, the mnemonics of the M6800 assembly language. Those who are not should refer to the "M6800 Microprocessor Programming Manual" or the "M6800 Programming Reference Manual", both available from your Motorola distributor.

The source language (input) for the SSB 6800 Assembler consists of a subset of the 7-bit ASCII (American Standard Code for Information Interchange, 1968) character set. Special meaning is attached to many of these characters will be described later. In all cases the parity bit (most significant bit) of each character must be Ø. All source files may contain upper or lower case mnemonics and hex characters.

Each line of source for the assembler consists of the source statement, followed by a carriage return (hex ØD). The source statement consists of up to four "fields" which are free format. From left to right, the four fields are label, operator (mnemonic), operand, and comment. There must be at least one space between each of these fields. Further restrictions and options of these fields are:

LABEL FIELD

1) The label must begin in the 1st column and must be unique.

2) Labels consist of letters (A-Z,a-z) and numerals (Ø-9).

3) Every label must begin with a letter (A-Z,a-z)

4) Only the first 6 characters of any label are significant, the rest are ignored.

5) The label field may be the only field present.

OPERATOR FIELD

1) The operator is 3 alphabetic characters (A-Z,a-z) which must be followed by a space.

2) Mnemonics such as LDA A and AND B may be written as LDAA and ANDB respectively. In this case, the fourth character must be followed by a space.

OPERAND FIELD

1) The operand field may consist of just an addressing mode indicator and expression or just an expression.

2) The addressing mode indicator is either a # (Pound sign) followed by ,X for indexed addression.

3) An operand may or may not be required depending on the addressing mode.

COMMENT FIELD

1) The comment field is optional.

2) Comments may contain any character from SPACE ($20) to DEL ($7F).

EXPRESSIONS

Expressions consist of combinations of numbers separated by one of the four arithmetic operators +,-,*,/. The arithmetic is done with 16 bit integer operands and truncated as necessary. 8 bit results are taken from the least significant 8 bits. Unary (+) and (-) are allowed. Expressions must not contain spaces.

NUMBERS

Numbers are groupings of the numerals 0-9 and possibly letters prefixed or postfixed by a base indicator. Possible base indicators are shown below. The ASCII base allows a single ASCII character ($20-$5F) to be used as an operand when preceded by a single quote.

BASE	PREFIX	POSTFIX	COMMENT
Decimal	none	none	decimal assured
Binary	%	B	0,1 allowed
Octal	@	O or Q	0-7 allowed
Hexadecimal	$	H	0-9, A-F allowed
ASCII	'	not allowed	ASCII equivalence

SYMBOLS

Symbols are groupings of letters and numerals, the first of which are significant and the first of which must be a letter. The single character * is a special symbol whose value is the current value of the program counter (PC). Upper case letters are not equivalent to

lower case, so the label 'START' is different from 'start'.

EVALUATION OF SYMBOLS

Since this is a two pass assembler all symbols must be resolved in two passes. Therefore, only one level of forward referencing is allowed.

AUTO FIELDING

This asembler performs automatic output fielding. No matter what the source file looks like in terms of field spacing, the output will automatically tab each field into a columnar form.

ASSEMBLER DIRECTIVES

In addition to the 72 M6800 mnemonics, this assembler supports 14 assembler directives or pseudo-ops. These pseudo-ops are listed below along with a brief description. More detailed descriptions follow:

FCC	form constant character
FCB	form constant byte
FDB	form double byte
SPC	insert spaces in output listing
OPT	activates or deactivates assenbler options
PAG	skip to next page of output
ORG	define new origin (PC)
EQU	assign value to symbol
END,MON	signal end of source program
NAM,TTL	specify name or title
RMB	reserve memory bytes
LNC	line of characters

FCC

The function of FCC is to create character strings for messages or tables. The character string 'text' is broken down to ASCII, one character per byte. The two allowable formats are shown below.

```
    label FCC  count,text
or
    label FCC  delimiter text same delimiter
```

where count is any decimal number. In the case where a number is used as a delimiter, the first character of text must not be a comma. The character limit of any FCC statement is 255. The use of label is optional.

FCB

The FCB pseudo-op caused an axpression to be evaluated and the resultant 8 bits placed in memory. Usage is shown below:

```
label FCB  expression 1,expression 2,...expression N
```

Each expression is separated by a comma with a maximum of 255 expressions per FCB statement. The label is optional.

FDB

The function of the FDB directive is identical to the FCB except 16 bit quantities are assembled, i.e., two bytes generated for each expression. The required format is shown below:

```
label FDB  expression 1,expression 2,...expression N
```

where the label is opional. The maximum number of expressions is 127.

SPC

The SPC operator causes the specified number of spaces to be inserted in the output listing. The format is shown below:

```
SPC  expression
```

Notice that no label is allowed. If 'expression' evaluates to zero, one space is inserted. The operator SPC itself does not appear in the output listing. If PAGE mode is selected, SPC will not cause spacing past the top of the next page.

OPT

The directive OPT is used to activate or deactivate the assembler options from within the source program. The format is shown below. Note that no label is allowed, and no code is generated.

```
OPT  option 1,option2,...option N
```

The allowable options are:

SYM	print sorted symbol table after listing (default).
NOS	do not print the symbol table.
GEN	print code generated by FCB, FDB, and FCC (default).
NOG	print only one line for each FCB, FDB, or FCC.
LIS	print the assembled source listing (default).
NOL	supress the printing of the assembly listing.
PAG	enable page formatting and numbering.
NOP	disable page mode (default).
TAP	enable the production of MIKBUG object tape.
NOT	disable the production of MIKBUG object tape (default).

If contradicting options appear, the last one appearing takes precedence. All options take effect simultaneously at the beginning of pass 2. The default options specified take effect unless the user specifies a particular option. Only the first 3 characters of an option name are significant and multiple options are separated by a comma.

Note that if a NOT, SYM, or GEN option is used with the OPT directive it will override the action of a T, S, or G calling switch respectively, but the L switch will override the LIS option.

PAG

The PAG operator, if the PAG option is on, causes a page eject and subsequently causes the title (if any) and page number to be printed at the top of the next page. No label is allowed and no code is produced. Notice that the first page of any listing is page Ø and no title is printed on that page. The PAG operator itself will not appear in the listing. The usual procedure is to have all options and the title declaration followed by a PAG be the first statements in a program.

ORG

The ORG operator, whose format is shown below, causes a new origin address (PC) for the code following.

```
ORG  expression
```

No label is allowed and no code is produced. If no ORG appears an origin of ØØØØ is assumed

EQU

EQU is used to equate a symbol to an expression as shown below. A label is required and no code is generated. Only one level of forward referencing is allowed and the equate must not be recursive.

```
label  EQU  expression
```

No code is produced by EQU.

END or MON

These operators signal the assembler that the end of the source input has occured. No label is allowed and no code is generated.

A second use of the END statement allows for the assignment of a transfer address to the binary file created. This can be accomplished by putting a label or value in the 'operand field' of the END statement. As an example, suppose the program you are assembling is to start executing at location $1ØØ, and in the source file you have the label START on the statement which is ORGed at $1ØØ. The END statement should now include this label in its operand field in order to assign $1ØØ as the transfer address, as shown on the next page.

```
        ORG $100
START   LDX $007D

        program here

        END START

           or

        END $0100
```

NAM or TTL

These operators are used to assign a title to be printed at the top of all pages (other than page 0) if the PAG option is on. If the PAG option is off, this operator has no effect. The format, as shown below allows up to 32 characters in the title. No label is allowed.

```
        TTL   text for the title
```

No code is generated. If more than one NAM or TTL operator appears the last one encountered will be printed on the next page.

RMB

This operator causes the assembler to reserve memory for data storage. No code is generated, therefore the contents of the reserved memory locations are undefined at run time. The label is optional as shown below:

```
     label RMB  expression
```

where 'expression' is a 16 bit quantity.

LNC

The LNC operator causes a line of characters to be generated in the assembly listing at list time. The LNC feature is helpful for making a line of characters to set off modules of code, etc. without adding excess character data to the text file. The LNC operator is followed by a decimal character count from 1 to 255, followed by the ASCII character to make up the line. An example of its use is shown below.

```
        LNC 50,*
```

which yields:

```
        **************************************************
```

ASSEMBLER OPERATION

Pass 1

The first pass is used to build the symbol table which is used to resolve forward references.

Pass 2

During the second pass, several things may happen. If the LIST option is on, the assembled source listing is printed with error messages, if any. If the LIS option is off, only offending source lines and their corresponding error messages are printed. If the TAP option is on, a MIKBUG S-format object record is output (through a different control point than the source listing). If the SYM option is on, a sorted symbol table will be printed after the assembled listing (if any).

CUSTOMIZING THE ASSEMBLER

Due to the inherent flexibility of the assembler, it is necessary sometimes for the user to customize the assembler to meet his particular needs. This involves very few changes and can be made by any individual familiar with 6800 assembly language. Use GET,ASMB.$ to load the assembler into memory without executing it. Refer to the partial source listing for the absolute addresses of the parameters to be explained below. After making the desired changes, use the SAVE transient to save the contents of locations $0100 to approx. $1D00 as a Disk resident command file with a transfer address of $0100.

Tape output character routine

The location TAPOUT must be changed to that of your tape punch (or tape record) subroutine. It is through this point that MIKBUG formatted object code is outputted. If you do not have a separate punch or record device, this address may be the same as the DOS Character output routine address, i.e., tape device same as list device.

Tape control characters

There are provisions at locations TAPEON and TAPEOF for four control characters to activate and deactivate, respectively, your punch or record device. Simply place the appropriate control characters for your device in each of the strings. If you desire to send less than four characters, change the byte at TAPCNT to the appropriate value (even 0). This will, of course, affect both turn on and turn off.

Tape control delay

The byte at TAPDEL controls the number of half-seconds (1 MHZ clock) of delay between tape turn on and data, and also between data and tape turn off. The delay is now set to 2 seconds. If you dont need delay at all, set the byte to 0.

Page length control - Page eject

The byte at LINES controls the number of lines to be printed on each page. The number of lines per page is set for 66. EJTIME defines the line number where the page will be advanced to the top of the next form. The difference between LINES and EJTIME define the number of bottom margin lines.

Symbol Table Space

The number of symbols which the assembler will support is dependent upon the amount of memory installed in the computer. As a rough

guide, 12K will allow approximately 200 symbols while 24K will allow 1000 symbols. It is roughly proportionate for values between these extremes. The assembler automatically determines the amount of memory available to it based on whether or not the system parameter MEMAX has been set (if MEMAX=$0000 it is not considered to be set). If MEMAX is not set, the address at location MEMEND will set the upper limit for the symbol table. The user should not set MEMAX or MEMEND to an address close to or within the bounds of the assembler or DOS as the results may be unpredictable. Refer to the DOS68 documentation for details concerning setting MEMAX with SET.$.

The user may want to alter other parameters beyond the scope of the parameters described above. Most modifications of this type will be needed by very few users and therefore will not be elaborated upon here. Such users are encouraged to order a source listing of the assembler before attempting such changes.

ERROR MESSAGES

This assembler supports 12 error messages which are printed after the offending line. The error messages announce violations of any of the restrictions set forth in this manual and are, therefore, self explanatory.

Additionally, the byte 'ERRORS' (cleared by DASMB) will be set if any errors have occurred in any of the passes. Note: The Hex characters $00-$0C, $0E-$1F, and $80-$8F, inclusive are explicitly prohibited from being in any of the source program, as their presence will cause undefined results. The remaining ASCII characters may appear subject to all the foregoing restrictions.

ADDITIONAL FEATURES

This assembler supports 2 extra mnemonics, namely BHS and BLO, which are the logical equivalents to BCC and BCS, respectively. However, Branch if Higher or Same and Branch if Lower are much easier to remember and use.

PARTIAL SOURCE LISTING

```
                    0152 *
0100                0153          ORG    $0100
                    0154 *
0100 7E 14BC        0155 MAIN     JMP    DASMB
0103 5E 00          0156 MEMEND   FDB    MEMLIM       MEMORY LIMIT
0105 42             0157 LINES    FCB    66           LINES PER PAGE
0106 3F             0158 EJTIME   FCB    63           EJECT POSITION
                    0159 *
0107 04             0160 TAPDEL   FCB    4            TAPE DELAY CTR
0108 04             0161 TAPCNT   FCB    4            TAPE CNT OF CTL CHRS
0109 00             0162 TAPEON   FCB    0,0,0,0
010D 00             0163 TAPEOF   FCB    0,0,0,0
                    0164 *
0111 86 20          0165 OUTS     LDA A  #SP
0113 7E 72C1        0166 CHROUT   JMP    ZPUTCH       DOS CHARACTER OUTPUT
0116 7E 0113        0167 TAPOUT   JMP    CHROUT       MIKBUG TAPE DATA
                    0168 *
```