3.00.00 CONFIGURING PL/9 TO YOUR SYSTEM HARDWARE ENVIRONMENT

This section will provide details of how to install PL/9 in your system.

(1) The first thing you should do upon receipt of this software is to complete and return the registration form that accompanies it. This is very important as it is through registration forms (not sales records) that we keep you informed of any upgrades that are available for the product. We will only support one user per copy sold. If you fail to register your copy with us and then phone for technical support we have no way of Knowing whether you are a legitimate user or someone who has come by this product by dishonest means. We will not answer any questions until we have a signed registration form in our files. DO IT NOW!

(2) Remember that this product is licensed for use by a single user on a single computer system. If any of your friends or associates within the same organization wants a copy contact the factory for details of our low cost 'KLONING' service whereby we create klone copies of your disk and manual. Each copy may be registered to a different individual and EACH user will be eligible for the same upgrades, support service, etc. as the original purchaser. THIS IS A LOW COST SERVICE DESIGNED TO KEEP YOU HONEST!

(3) The next thing you should do is to format a fresh disk in the system you will be using PL/9 with. You should then copy ALL of the files from the disk we have supplied you to this new disk. The original disk we supplied should then be put in a safe place as we will require that you return it to us if you want to upgrade your copy of PL/9 at a later date.

(4) The next thing you should do is to copy the following files, which are the files associated with the compiler itself, from the working copy of the PL/9 disk onto your SYSTEM disk, i.e. the disk that normally resides in drive #0. These files MUST be on the drive defined as the SYSTEM drive (as defined by the FLEX 'ASN' command) for the compiler to operate properly

a. PL9 .CMD the compiler
b. PL9_TD .CMD the tracer
c. PL9 .ERR the compiler error message file
d. SETPL9 .CMD the PL/9 configuration program

(5) If you intend to use any of the library files we supply and your SYSTEM disk has adequate spare capacity we suggest that you also copy the following files on to your system disk. These files are not required for the compiler to operate unless one of your PL/9 source files 'INCLUDEs' them. A selection of these files can also be part of your work disk if you desire.

a. TRUFALSE.DEF
b. HEXGLOBL.DEF
c. IOSUBS .LIB
d. TERMSUBS.LIB
e. HEXIO .LIB
f. BITIO .LIB
g. HARDIO .LIB
h. STRSUBS .LIB
i. BASTRING.LIB
j. FLEX .LIB
k. SCIPACK .LIB
l. REALCON .LIB
m. REALIO .LIB
n. NUMCON .LIB
o. SORT .LIB

(6) There will also be several files on the disk with the extension '.PL9'. These files are sample programs and need not be copied to your system disk. All files are present on the disk and their purpose will be described in file called 'READ-ME.TXT'. List this file for further information.

(7) The last thing you must do is to configure your copy of PL/9 to match the system you will be using it in. A special program has been provided to greatly simplify the task of configuring PL/9 to your FLEX system and it's terminal. The program is called 'SETPL9.CMD' and it runs like this:

```
+++SETPL9<CR>

*****   PL/9 Configuration Program    *****
        ==========================

      For use with PL/9 version 4.XX

This program allows you to configure PL/9
to your own particular requirements and
those of your computer system.

Some of the questions do not need answers
unless you wish to alter the current
settings. In these cases the existing data
will be displayed. To leave the data <---------- note!
alone just hit <CR>.


        PUT YOUR PL/9 DISK IN DRIVE
        ZERO THEN HIT ANY KEY......         <---------- hit <CR>


            INPUT WITHOUT ECHO
            ==================


PL/9 requires your keyboard input routine
to return the ASCII value of the key
pressed, in the A-accumulator, WITHOUT the
character being echoed. Early versions of
FLEX9 did not support this feature, so you
should first check whether you have an
"INCHNE" vector (i.e. address of an input-
without-echo routine) at $D3E5. If you need
to exit SETPL9 to examine your system, just
type <CR>.


Does your system have INCHNE at $D3E5? Y    <--------- 'Y'for most systems.
```

If you answer this question 'Y' the prompt just after '***' on the following page will appear.

If you have an early FLEX 9 system produced by SWTP (et al) the INCHNE vector at $D3E5 will not be

present. If this is the case answer this question 'N' and the following options will be available.

There are three ways to implement input without echo:

1) You can give me the address of a routine in your system that performs input without echo;

2) You can give me the address of a vector that points to your input routine (NOT the address of the routine itself);

3) You can give me the address of your input device (6850 ACIA, 6821 PIA etc.) at which the ASCII key code can be found. It is essential that the act of reading the character clears the "character waiting" flag;

4) You can exit and think about it.

Which do you want to do (1-4)? 2 <------- if you have an 'SBUG-E' compatible system monitor.

Address of your input vector: $F804 <--------------- 'INCHNE' vector

NOTE: If you are not sure of the address you are supplying OPEN THE DISK DRIVE DOORS as an invalid address will cause the system to CRASH!

***

Just to check, type the number "1": 1 <---------------------- type '1'

I got a "1"!! Did it echo on the screen? N <-------------- it should be 'N'

NOTE: If any part of this check fails SETPL9 will return to FLEX.

That's the difficult part over. Now for the easy bits!

KEYBOARD

Next let's set up PL/9 for your keyboard. Each question should be answered with a single keypress or control key combination, e.g. (control H) for backspace:

First press your backspace key..... <----------- CTRL H if in doubt
Now your forward tab key..... <----------- CTRL I if in doubt
Next your line cancel key..... <----------- CTRL X if in doubt
Now your escape key..... <----------- CTRL [if in doubt
And lastly Control C or equivalent... <----------- CTRL C if in doubt

PRINTER

Now to set up PL/9 for your printer.

How many listing lines are to be printed on each page (leave some room for top and bottom margins)?
..... (currently 55)? 55 <---------- number then <CR>

Total length (in lines) of each sheet?
..... (currently 66)? 66 <---------- number then <CR>

Does your printer support form feed? Y <---------- Y or N (<CR> = Y)
What HEX character is it?.....$0C <---------- $0C<CR> for most

NOTE: PL/9 obeys the TTYSET 'WD' parameter when it outputs to the system terminal AND the system printer. If you have a 132 column printer and wish to prevent PL/9 from truncating the source lines simply set WD=132 before using PL/9.

STRINGS

In PL/9 as supplied, strings are terminated with nulls, and the library routines are set up to recognise this character. You can change the null to anything else you like (e.g. $04) but many of the library routines will then not work unless modified.

.....Is the null terminator OK? Y <---------- 'Y' for most users.

What HEX character do you want?.....$04 <------- You get this question if you answer the one above 'N'. $04 is the end of transmission character recognized by most system monitors and FLEX

DISK FILES

Now I want to know the default filename extensions and default drive numbers: <--------- hit <CR> to accept existing defaults.

LOAD and SAVE file extension.(currently PL9): <--------- new extension <CR>
and its default drive number..(currently #1): <--------- new drive number

OBJECT (A:O) file extension..(currently BIN): <--------- new extension <CR>
and its default drive number..(currently #1): <--------- new drive number

INCLUDE file extension.....(currently LIB): <--------- new extension <CR>

```
and its default drive number..(currently #0): <--------- new drive number

LISTING (A:L) file extension.(currently OUT): <--------- new extension <CR>
and its default drive number..(currently #1): <--------- new drive number

READ and WRITE use a file called: "1.SCRATCH.SCR".
.................................is this O.K.? <--------- Y or N (<CR> = Y)

                                                    If you answer the above
                                                    prompt 'N' you will  be
                                                    asked the following:

R/W scratch file name?...(currently SCRATCH): <--------- new name <CR>
R/W scratch file extension?..(currently SCR): <--------- new extension <CR>
and its default drive number..(currently #1): <--------- new drive number
```

SYSTEM INTERRUPT VECTORS

Lastly, I need to know where your interrupt vectors are. These are the RAM vectors used by your system monitor.

```
RESET (ROM=$FFFE) vector (fixed  at: $FFFE)

NMI   (ROM=$FFFC) vector (currently: $E740):  <--------- 4 HEX digits <CR>
SWI   (ROM=$FFFA) vector (currently: $E746):  <--------- 4 HEX digits <CR>
IRQ   (ROM=$FFF8) vector (currently: $E744):  <--------- 4 HEX digits <CR>
FIRQ  (ROM=$FFF6) vector (currently: $E742):  <--------- 4 HEX digits <CR>
SWI2  (ROM=$FFF4) vector (currently: $E748):  <--------- 4 HEX digits <CR>
SWI3  (ROM=$FFF2) vector (currently: $E74A):  <--------- 4 HEX digits <CR>
```

Your copy of PL/9 is now configured!

+++

The hex addresses supplied as the default RAM vectors are compatible with all versions of the windrush MC6809 system monitor 'GT-BUG9'.

If your system uses a GIMIX GMXBUG, Version 2, or an SSB MON-69D, or an SWTP SBUG-E the following interrupt vectors should be correct. As we have no control over what alterations these manufacturers make to their products this information is supplied for GUIDANCE only. We do not assume any responsibility for its accuracy. Consult the documentation supplied with your system to be on the safe side.

VECTOR	GIMIX	SSB	SWTP
NMI	$E408	$F3C0	?????
SWI	$DFCA	$F3CA	$DFCA
IRQ	$DFC8	$F3C8	$DFC8
FIRQ	$DFC6	$F3C6	$DFC6

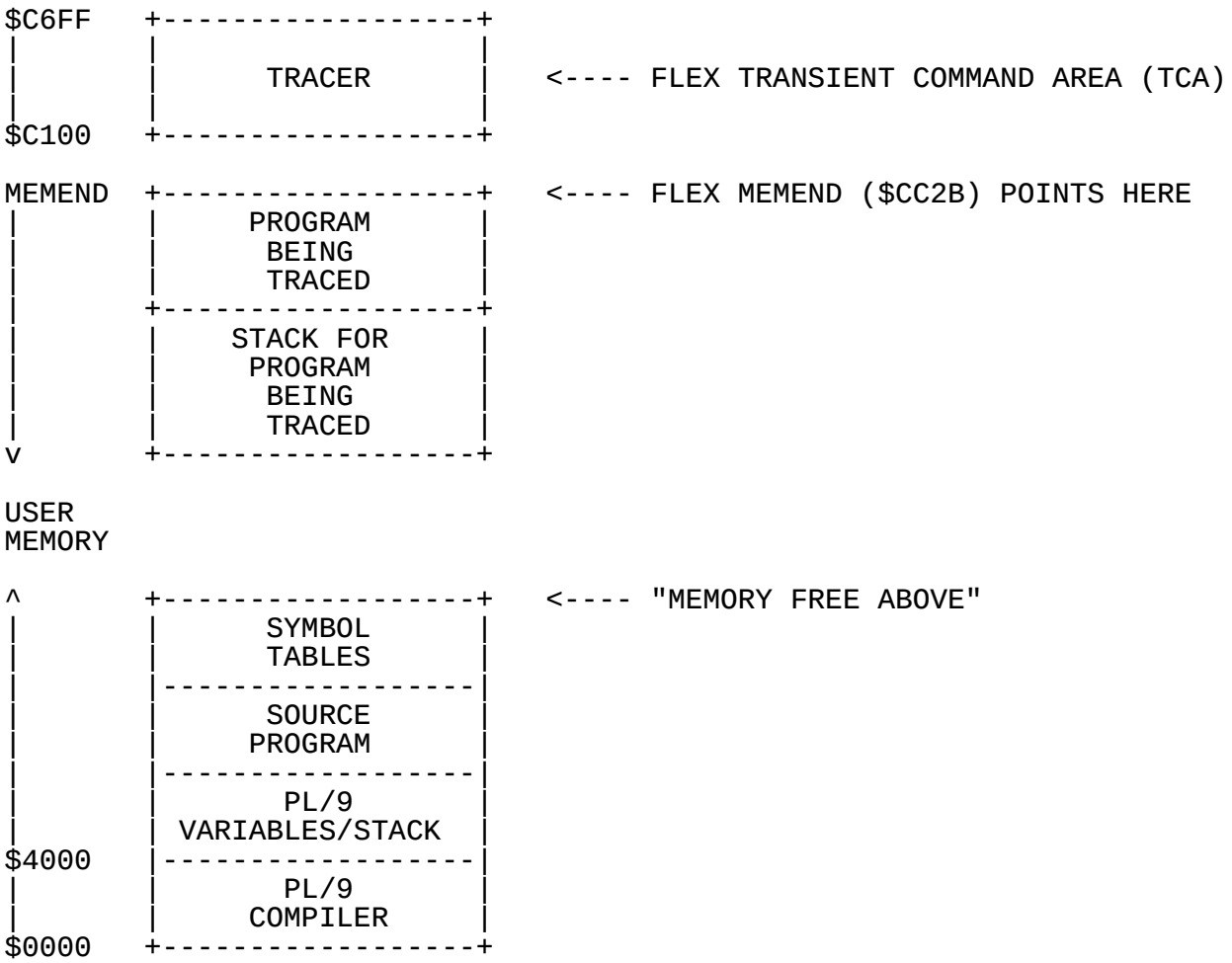
SWI2	$DFC4	$F3C4	$DFC4
SWI3	$DFC2	$F3C2	$DFC2

N O T E

If you compile a file using the 'R' option (see Compiler Reference) the MC6809 hardware interrupt vectors at $FFF2 through $FFFF will be substituted for the RAM vectors specified.

3.01.00 MEMORY MAP

PL/9 uses memory as follows:



PLEASE READ THIS

From this memory map it should not be difficult to ascertain that a program being run under the tracer cannot have a massive amount of variable storage ALLOCATED on the stack NOR can the program USE a massive amount of stack. For example the statement 'GLOBAL BYTE BUFFER(32000);' would cause the program to relocate the stack right into the middle of of your source program and clobber it the moment the program was run.

It is up to the programmer to ensure that there is adequate space on the stack before invoking the tracer. You should remember also that although the tracer runs in the FLEX TCA it also makes use of some of the facilities in the compiler which resides between $0000 and $3FFF. If your program writes into this area (or relocates the stack into this area) the result will undoubtedly be a system crash.

3.02.00 PL/9 AND THE FLEX ENVIRONMENT

PL/9 makes certain assumptions about the FLEX environment it is running in, and assumes that you have configured it for your system hardware. Generally PL/9 and its associated library modules should be placed on your SYSTEM disk (drive 0), duplicate (or modified) copies can be maintained on your software development work disk if desired as this will reduce the number of alternate disk accesses during compilation and the attendant 'head banging'.

The TTYSET 'PS' (pause) should be turned off (+++TTYSET,PS=NO<CR>). Leaving the TTYSET pause function enabled can cause annoying stops when listing lines in the PL/9 editor and strange stoppages during program execution while in the tracer.

The TTYSET 'WD' (width) should be set to the width of your printer or terminal which ever is the greater. If, for example, you leave the WD value set to 80 columns PL/9 will truncate all terminal (A:T) printer (A:P) and listing (A:L) output at 80 columns. If you have a 132 column printer we suggest that you set WD=132 and leave it there. The TTYSET WD parameter is not normally used by FLEX unless some program exceeds the width of the system console at which time FLEX will generate a CR-LF.

3.02.01 PL/9 AND FLEX PRINTER DRIVERS

A set of TSC standard printer drivers should be loaded into memory at $CCC0 - $CCFF (e.g. +++GET,PRINT.SYS<CR>). If you don't know what we mean by 'standard' see your FLEX User's Manual for details of 'standard' serial and parallel printer drivers. If you have ANY problems with your printer and PL/9 it is almost certainly going to be due to non-standard printer drivers.

The printer drivers will be accessed via the standard entry points (INIT=$CCC0 and POUT=$CCE4), if your printer drivers reside elsewhere these locations should point to your routines via an extended jump (7E XXXX). The printer driver routines must preserve the registers specified by TSC.

A problem arises for users of SouthWest Technical Products' FLEX, which has a different way of handling printer drivers to that used by most other versions of FLEX. SWTP FLEX requires the programmer to use the "P" command to cause output to be sent to the printer. This method will not work satisfactorily with PL/9. The recommended action is either to generate a printer driver that PL/9 can use (see above) or to direct output to a disc file for later printing.

3.02.02 PL/9 AND FLEX ITSELF

It is important to note that PL/9 expects, no demands, that your system have FLEX implemented properly. This means that the INCHNE routine (pointed to by the address held in $D3E5) must strip the parity bit (the most significant bit) of the incoming character as specified by TSC and that all routines within FLEX preserve the registers as specified by TSC.

We have tested this program on FLEX from SSB for their DCB-4A controller, FLEX from GIMIX for their DD-5 disk controller and DMA disk controller. It has also been tested on all Windrush FLEX systems. Therefore we are satisfied that the program will work 'as advertised' if your system has had FLEX implemented in accordance with TSC's standards.

3.03.00 JUST TO PROVE THAT IT WORKS

This section is a quick exercise just to prove to you that the product works. Before starting this exercise you must have the following files on the disk in drive #0.

a. PL9 .CMD
b. PL9_TD .CMD
c. PL9 .ERR
d. IOSUBS .LIB

First we will call PL9 off the system disk so that we can use it in the interactive mode:

```
+++PL9 <CR>
```

PL9 will greet you with its startup banner:

```
PL/9 Compiler, Version X.XX
Serial #XXXX
    #
```

The hash symbol '#' signifies that the EDITOR is in command mode and waiting for a command. First we must instruct the editor to enter the 'INSERT LINE' mode of the editor so we can type in our test program. We do this with the 'I' command thus:

```
#I<CR>       ..... type 'I' followed by <CR>
```

PL9 will then respond with a '+' to tell you that you are in the 'INSERT' mode. We will now enter the text of the program. You can use 'BACK SPACE' to correct any errors on the line before you type <CR>. If you hit <CR> before you spot the error you will have to fix it later using the 'O' (OVERLAY) command or delete the line and re-enter it. To use these latter facilities of the EDITOR you will have to read section four.

```
        +ORIGIN=$C100; /* PUT IN FLEX TCA */<CR>
        +<CR>
        +INCLUDE 0.IOSUBS.LIB;<CR>
        +<CR>
        +PROCEDURE CONFIDENCE;<CR>
        +   PRINT "\n\nHELLO WORLD\n";<CR>
        +<ESCAPE>
```

Hitting <ESCAPE> when the cursor is adjacent to the '+' symbol means you wish to exit the INSERT LINE mode and return to editor command mode. PL9 signifies this mode by the appearance of the hash symbol '#' once again. Now just to make sure that we have typed the program in correctly lets list it. We do this with the command '1P22'. This command means 'go to line number 1 and print 22 lines on the system console':

```
    #1P22<CR>
0001 ORIGIN=$C100; /* PUT IN FLEX TCA */
0002
0003 INCLUDE 0.IOSUBS.LIB;
0004
0005 PROCEDURE CONFIDENCE;
0006    PRINT "\n\nHELLO WORLD\n";
0007 /EOF
    #
```

NOTE: The '\n' sequence used three times in the above string means NEW LINE. The 'PRINT' routine (in IOSUBS.LIB) will interpret this sequence and generate a carriage-return followed by a line feed.

Now we should save a copy of the source file on disk for later use. We do this by typing:

```
#S=0.HELLO.PL9<CR>
```

This command will cause a disk access to drive #0 as the compiler saves the source file to a disk file called 'HELLO.PL9'.

Next we should compile the file so that we can use it. We do this by typing:

```
#A:O=0.HELLO.CMD<CR>

Memory Free Above $XXXX
Last PC Value was $XXXX
```

This command will cause a disk access to drive #0 as the compiler first reads and compiles the library file 'IOSUBS.LIB' and then compiles the source file resident in memory to a disk file called 'HELLO.CMD'. PL/9 will automatically generate the return to FLEX (JMP $CD03) at the end of the program as we have left off the last 'ENDPROC'. PL/9 will also automatically generate the transfer address of the program. The mechanisms that control these actions of the compiler will be discussed in subsequent sections of this manual.

If any errors are detected by the compiler they will be reported and the compiler will stop at the line containing the error. If you hit <CR> the compiler will abort the compilation and return to command mode POINTING to the line where the error occurred. If you hit <SPACE> the compiler will continue until either it completes the compilation or encounters another error.

If the program compiled without errors we are now ready to return to FLEX. We do this by typing:

```
    #X<CR>
IS TEXT SECURE? Y <------- type 'Y'

+++
```

Now we can invoke the program we just compiled by typing:

```
+++HELLO<CR>
```

The program will load and then print:

```
HELLO WORLD

+++
```

Well done! You have just written and compiled your first program with PL/9.

Now you might want to try a couple of the examples in the next section which demonstrate some of the other features of PL/9.

3.03.01 OTHER MODES OF OPERATION

PL/9 has two distinct modes of operation. The first is calling it from the FLEX command line and giving it the name of the input file, the output file(s), and the compile option(s). The second is the interactive mode where PL/9 is called as a FLEX command, i.e. (+++PL9<CR>) and used in much the same manner as BASIC. This latter mode was the mode we used in the previous section.

Whatever mode of operation you use PL/9 will load into memory and start up. A banner will appear, announcing the name of the compiler, the version number and the serial number of your copy. Any questions to your supplier concerning the product should quote both of these numbers.

When using PL/9 in the inter-active mode a hash (#) symbol will appear on the next line, indented by four spaces. This is the prompt that signifies the editors readiness to accept commands. The editor is where you will always be if you are not actually testing a program using the tracer. Commands are available to allow you to create or alter a source file, to load from or save to floppy disc, to pass commands to FLEX and to tell the compiler part of the program to do its job. The reference sections contain a full description of each of the editor commands, compiler commands, and tracer commands. It is to these sections that you should go to gain familiarity with this part of the system.

PL/9 can alternatively be invoked from FLEX and instructed to load a file, compile it and return to FLEX, much in the same way as other compilers. Suppose that you have a program called HELLO.PL9 on your working drive, that you wish to compile to generate a command file HELLO.CMD on the system drive, and that you also wish to have a listing, complete with machine code, sent to your printer.
The command line that would achieve all this is as follows:

```
+++PL9,0.HELLO.PL9,O=0.HELLO.CMD,P,C<CR>
```

Note that in order for this particular example to work, a suitable printer driver must have been installed in FLEX before the command is given.

You can also instruct PL/9 to do the same job but instead of directing the listing output to a printer the listing may be directed to a disk file for later use.

```
+++PL9,0.HELLO.PL9,O=0.HELLO.CMD,L=0.HELLO.OUT,C<CR>
```

PL/9 will generate a listing file on drive zero called HELLO.OUT, which can then be spooled to the printer or "P,LIST"ed.

You can also instruct PL/9 to do the same job but instead of directing the listing output to a printer the listing may be directed to the system console.

```
+++PL9,0.HELLO.PL9,O=0.HELLO.CMD,C,T<CR>
```

More information on this subject is contained in section five.

THIS PAGE INTENTIONALLY LEFT BLANK