APPLICATION		REVISIONS		
NEXT ASSY	USED ON	LTR	DESCRIPTION	DATE
	7506			

REV STATUS OF SHEETS																	
REV																	
SHEET	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—

UPDATE DOCUMENT, DX10 TI Pascal, RELEASE 1.8.0-990

TEXAS INSTRUMENTS INCORPORATED DATA SYSTEMS GROUP	drawing number 2239939-9901		
		REV. **	SHEET 1 OF 14

SECTION 1

INTRODUCTION

This document describes the changes which have been made to the DX10 TI Pascal 1.7.1 package to produce the 1.8.0 release. Section 2 briefly describes the release enhancements and Section 3 lists the problems that have been fixed. Section 4 lists some differences from previous releases that users may need to be aware of.

There are two versions of the 1.8.0 release, one which runs on the DNOS operating system (release 1.1 or 1.2) and one which runs on the DX10 operating system (release 3.5 or 3.6).

SECTION 2

RELEASE ENHANCEMENTS

2.1 COMPILER

LISTOBJ Option. A new compiler option, LISTOBJ, has been added. LISTOBJ is a statement-level option which directs the compiler to display the 990 instructions it has generated in the compiler listing.

The LISTOBJ output lists the assembly language instructions corresponding to particular Pascal source lines. Also included are the values of all literals or constants generated, the hexadecimal value of literals and instructions, and hexadecimal offsets from the beginning of the object module. When used with the WIDELIST option (which lists source line numbers), LISTOBJ output is a very useful debugging tool.

See also the new Assembly Language Extractor utility below.

2.2 UTILITIES

NESTER. The NESTER utility, which formats Pascal source programs, allows NESTER options to be specified in the XNESTER SCI procedure. Previously, all NESTER options had to be specified in the source file itself, inside a NESTER option comment. NESTER option comments are still allowed. The ability to specify options at the command level in addition to inside the source file allows for increased flexibility. (STR 12891)

SPLITPGM. A new SCI command procedure (XSPLITPG) has been added for the Split Program utility SPLITPGM. The SPLITPGM utility, which splits individual modules out of a Pascal source file, can now be operated independently of the configuration processor (CONFIG). SPLITPGM reads a single Pascal source file (possibly created with CONFIG), writes the individual source modules to separate files, and optionally generates a deferred command file for later use with CONFIG.

XALX. A new utility called the Assembly Language Extractor has been added to the TI Pascal package. This utility, invoked by using the SCI command XALX, reads a compiler listing file and extracts any assembly code produced with the LISTOBJ option. The output from this utility is an assembly language source file, which can be submitted as input to the 990 Macro Assembler.

2.3 RUNTIME LIBRARY

The OUTPUT and SYSMSG files may now be opened with EXTEND instead of REWRITE when for use with the minimal run-time (STR 6004). This can be accomplished by including the following commands in the link control file:

```
INCLUDE (EXTOUT)      ; MODULE FOR EXTENDING OUTPUT
INCLUDE (EXTMSG)      ; MODULE FOR EXTENDING SYSMSG
```

The TASK-ID and the RUN-ID of the executing task have been added to the error and termination message that is written to the message file when linking with the MINOBJ library (STR 6004). The format of the message is now:

```
CODE = P000 PC = 0000 STACK: >0114 HEAP: >009C  RUN-ID = >06
TASK-ID = >01
```

SECTION 3

PROBLEMS FIXED

3.1 COMPILER ERROR CHECKING AND REPORTING

A few cases where the internal compiler message "TEMP FROZEN" appeared have been fixed. (STR 4195, 4472, 5153, 7668, 12463)

The cases where error 104 (UNDECLARED IDENTIFIER) is issued incorrectly have been fixed. (STR 5513)

Several instances of the "OUT OF REGS" internal message have been corrected. (STR 6243)

Warning 309 (CONSTANT EXPRESSION OUTSIDE VALID RANGE FOR INTEGER) is now issued if an attempt is made to assign a constant expression less than -32,768 or greater than 32,767 to an integer variable. (STR 8285)

Error 259 (COLUMN AND STATUS PARAMETER MUST BE OF TYPE INTEGER) is now issued if these actual parameters to ENCODE or DECODE are not of type INTEGER. (STR 8302, 14919)

Error 25 (ILLEGAL CHARACTER ENCOUNTERED) is now issued if an invalid character (#01..#19) is detected in the source file. The null character (#00) and delete character (#7F) are ignored. (STR 8467)

Two consecutive single quotes ('') with no characters in between result in error 50 (CONSTANT EXPECTED). (STR 9334, 14055)

If an attempt is made to type-transfer a variable to a larger type, warning 263 (LENGTH OF TYPE TRANSFER IS GREATER THAN ORIGINAL TYPE'S LENGTH) is issued. (STR 9756)

Error 245 (CANNOT TAKE LOCATION OF A PACKED FIELD) is now issued if a packed field is used as the argument to the LOCATION function. (STR 10298)

A record defined to be larger than 32766 bytes results in error 246 (RECORD TOO LARGE). (STR 11861)

The severity level for error 168 (UNDEFINED LABEL) has been changed from non-fatal to fatal. Also, the undefined label number is written to the listing. (STR 12565)

3.2 COMPILER ABORTS

SILT1 will no longer abort if a procedure is declared FORWARD and then defined twice. This case is detected and flagged by issuing warning 164 (ROUTINE NAME MUST BE UNIQUE IN FIRST 6 CHARACTERS). (STR 3542)

Several problems causing CODEGEN to abort with the error "TEMP NOT FOUND IN GENOPERAND" have been fixed. (STR 4527, 6925, 7925, 10796, 11769, 12591, 15683)

Some situations which caused SILT2 to hang in a run loop have been fixed. (STR 5648)

CODEGEN no longer hangs when encountering certain situations involving REAL number conversion. (STR 9093)

Attempting to compile a program with more than 1023 active identifiers results in graceful termination and error 257 instead of SILT1 hanging or SILT2 aborting with a run-time error (STR 9746, 10571, 10972)

Problems where CODEGEN aborts when using the EOF or EOLN functions have been fixed. (STR 7204, 9934)

Two separate problems causing CODEGEN to hang when compiling large programs with complex expressions have been fixed. (STR 10059, 16341)

Situations where SILT2 aborts with an addressing error during error recovery have been corrected. (STR 10492, 11917, 15684)

SILT1 no longer hangs if incorrect syntax appears before the PROGRAM keyword, or if PROGRAM is declared as an identifier. (STR 10801, 11564)

A problem causing CODEGEN to abort with the message "FILE TMPF NOT OPENED FOR WRITING" after assignment of a record has been corrected. (STR 11300)

CODEGEN will no longer abort with an addressing error on certain DECIMAL assignments. (STR 13121)

3.3 COMPILER CODE GENERATION

Occasional incorrect code generated for dynamic array lengths has been fixed. (STR 3174)

The proper length is now passed to the MESSAGE routine when a procedure argument of type packed array [1..?] of char is used as its argument. (STR 3772)

Occasional generation of incorrect code for assigning REAL expressions have been fixed. (STR 9096)

Occasional generation of incorrect code for the set IN operator has been corrected. (STR 9265)

Correct code is now generated for a CASE statement where the case selector is a packed field. (STR 9751)

Instances of incorrect code generation for LONGINT arithmetic operations have been corrected. (STR 10144, 11045, 11628)

Better code is generated for a CASE statement whose case selectors invlove a very large range. (STR 11484)

Occasional incorrect placement of run-time checks in the object file has been fixed. (STR 11597)

Certain situations where incorrect code is generated for an IF statement have been corrected. (STR 12097)

Correct code is now generated when using the WITH statement for more than one variable. (STR 14604)

3.4 OTHER COMPILER PROBLEMS

Code involving FIXED and DECIMAL variables is now more accurate. (STR 5152, 5153, 6788, 6873, 14151)

Occasional incorrect placement of run-time checks in the object file has been fixed. (STR 9246, 11597)

The optimization phase has been fixed to correctly perform sub-expression elimination in certain unusual situations. For example, the code sequence

```
A := X[3] + 1;
X[J] := 0;
B := X[3] + 1;
```

previously assigned the same value to A and B (which is incorrect if J happens to be 3). This case was also exhibited in certain assignment statements involving record variants. (STR 5520, 5732, 9767, 12549, 12754)

Correct code is now generated for a constant set used as an actual parameter to a routine whose corresponding formal argument uses a dynmic upper bound. (STR 9996)

The compiler will now handle the largest positive 32-bit number correctly without issuing an error. (STR 11689)

3.5 RUNTIME

When writing to a sequential file, trailing blanks will no longer be truncated. The amount of data transfered to the file will always be the same as the element size of the file. (STR 3646)

An attempt to ENCODE a real number with a value that is too large will now return an error code of 5 as well as fill the field with asterisks. (STR 4093)

A program using initiate I/O will no longer terminate with "HEAP FULL" because of the get memory SVC failing with initiate I/O in progress. The runtime will wait until the I/O operation is complete and then get the necessary memory. (STR 5149)

The shift routines in the runtime modules SLD$$$ and SRD$$$ have been split into seperate modules to save memory space. (STR 5156)

A TEXT file READ into a REAL variable will now return the error "ILLEGAL INPUT CHARACTER" if the first character in the field is non-numeric. (STR 5315)

A write of a FIXED value using the BIN format will now include a decimal point if the precision of the fixed number is greater than zero. (STR 6360)

Assigning the DUMY device to a Pascal sequential file with element size greater than 80 bytes will now work properly. (STR 7559)

READS and DECODES of values such as 00800 into an integer variable using the HEX format will no longer give the error "DATA VALUE TOO LARGE". Leading zeros will be ignored when evaluating the value of the number. (STR 8175)

When opening a file with RESET, REWRITE or EXTEND, if the open fails after assigning a luno to the file, the luno will be released. (STR 8299)

A set expression of the form [x..y] will now correctly produce the empty set when x > y. (STR 8756)

Procedure KEY$FILE will now use the proper logical record length instead of always using 68 when an insert is done after a read that has failed. (STR 9204)

The KIF_SET_ACCESS operation of the procedure KEY$FILE will work correctly now. Also, an error code of #F5 will be returned if the ACCESS CODE is invalid (greater than 3). (STR 9352)

When linking with the MINOBJ library and using a shared procedure segment, the runtime module GO$MR can now be linked into the procedure segment without having the heap manager (NEW$ or DISPOSE) also linked into the procedure. (STR 9436)

Floating point underflow is now detected, and an error message is given to the user. (STR 9576)

Opening a random file with EXTEND will work correctly now. The file will always be opened with shared access. (STR 13214)

The runtime will no longer allow a program to open a file that is declared to be a random file in the Pascal program but is actually a sequential file. The error "FILE TYPE AND ACCESS METHOD CONFLICT" will be given. (STR 13230)

The batch stream DXPASCAL.TIP.MISC.TIPKIF.BATCH now works correctly and can be used to build the KEY$FILE procedure. (STR 13516, 13517)

When creating a file with a REWRITE or EXTEND call, the open will no longer fail when the logical record length of the file is greater than the default physical record length of the directory and both are less than 864 bytes. Now, the create file operation will use the logical record length for the physical record length. (STR 14035)

The procedure EXTOUT in the LUNOBJ library can now be used to open the output file with a luno other than #3E assigned to it. (STR 14304)

3.6 UTILITIES

The following errors have been fixed in the PREPROCESSOR:

1. An "=" will now be recognized as a delimiter for an IDENTIFIER. Now, statements of the form "?IF NOLISTING=TRUE" will be accepted as proper statements for the preprocessor. (STR 7144)

2. The preprocessor will no longer hang in a loop writing the last record read to the output file when it encounters an I/O error while trying to read from its input file. The error will be reported to the user and the PREPROCESSOR will abort execution. (STR 7441)

3. The output file will be created BLANK SUPRESSED when the PRINT WIDTH option is used to specify a width greater than 80 characters. (STR 10498)

4. The error reporting capabilities of the PREPROCESSOR have been expanded. Now the source line producing the error will be printed along with the line number in the source file where the error occurred. (STR 13581)

The following problems have been fixed in the Configuration Processor:

1. The Configuration Processor will now give the error "BEGIN NOT FOUND IN SOURCE MODULE" when a routine which has other routines nested within it does not have BEGIN starting in column 1. (STR 9758)

2. The Configuration Processor will no longer produce a bad source file when processing routines with static nesting level greater than 10. (STR 10790)

3. The maximum number of modules that the Configuration Processor can handle has been increased from 511 to 767. (STR 12250)

4. The use of the "*SPLIT ALL" command will no longer cause the Configuration Processor to hang in a infinite loop. (STR 13855)

5. The Configuration Processor will no longer terminate processing with the error "ERROR EXPANDING PROCESS" when a large number of "*ADD" commands are being used to expand a process. (STR 15133)

The following problems have been fixed in the source program Nester:

1. The Nester utility will no longer eliminates blank lines in the nested output file. (STR 3395)

2. Nester will now give the proper error when an assignment statement has an "=" instead of a ":=". The error given is error 51 " := EXPECTED ". (STR 8007)

3. The Nester utility will now recognize the use of a caret "^" in addition to "@" for pointers. (STR 10438)

The following problems have been fixed in the CROSS-REFERENCE utility:

1. Comments that begin with "(*" and end with "}" and comments that begin with "{" and end with "*)" will now be handled correctly by the cross-reference utility. (STR 11402)

2. The 72COL option will now be recognized by the cross-reference utility. If this option is true (default), then only the first 72 columns of the source will be cross-referenced. Otherwise, the first 80 columns of the source will be processed. (STR 12391)

The following problems have been fixed in the Split Program utility:

1. The Split Program utility will now correctly handle routines whose names begin with the letter "Z". (STR 13407)

2. The Split Program utility has been improved so that it will now correctly split source files generated by the Configuration Processor. (STR 13530)

SECTION 4

DIFFERENCES

4.1 COMPILER ERRORS

There have been some improvements made in the compiler's error checking such that some problems which previously were not detected are now given a warning or error message. Thus, it is possible that a program which compiled without any errors with release 1.7 may get some of the following messages when compiled with 1.8:

25 E ILLEGAL CHARACTER ENCOUNTERED ON INPUT --
previously, any ASCII characters outside the range '#20'..'7E' did not generate meaningful diagnostics. (STR 8467)

168 F UNDEFINED LABEL --
this error, previously a warning, is now a fatal error, and lists the label which is undefined. (STR 12565)

194 W FILE CONTAINS POINTERS --
previously, this was a non-fatal error; it is now a warning aimed at flagging a condition which may affect program portability. (STR 12946)

245 E CANNOT TAKE LOCATION OF PACKED FIELD --
previously, no error was given if the argument to the standard function LOCATION was a field in a packed record. (STR 10298)

246 F RECORD TOO LONG -- MORE THAN 32766 BYTES --
records defined to be greater than 32,766 bytes long were not previously flagged. (STR 11861)

248 F FILE OF FILES IS NOT ALLOWED --
previously, a file declaration that included another file was reported as non-fatal error 198.

259 F COLUMN AND STATUS PARAMETERS MUST BE OF TYPE INTEGER --
previously, unpredictable results could occur if the column and/or status parameter to the standard

procedures ENCODE and DECODE were not of type INTEGER. (STR 8302, 14919)

263 W LENGTH OF TYPE TRANSFER IS GREATER THAN ORIGINAL TYPE'S LENGTH --
previously, the compiler allowed a type-transfer to be performed for a variable to a type that was larger than that of the variable. (STR 9746)

309 W CONSTANT EXPRESSION VALUE OUTSIDE VALID RANGE FOR INTEGER --
previously, the compiler did not detect the assignment of an integer constant or constant expression that was outside the range -32,768..32,767. (STR 8285)

4.2 RUNTIME ERRORS

P444 FLOATING POINT ERROR - UNDERFLOW --
previously, when performing floating point arithmetic, an underflow condition was not detected. (STR 9756)

P5FF OPEN ERROR - FILE ACCESS METHOD CONFLICT (on a RESET),
P6FF OPEN ERROR - FILE ACCESS METHOD CONFLICT (REWRITE or EXTEND) --
previously, no error was issued if a file declared to RANDOM in the TIP program is actually a sequential file. (STR 13230)

4.3 CONFIGURATION PROCESSOR ERRORS

A new level of severity has been added to errors reported by the configuration processor, so that correctable errors detected while in interactive execution mode can be distinguished from more serious errors. The severity level of a configuration processor error is determined by the condition code synonym $$CC.

VALUE	STATUS
>0000	Normal Termination
>4000	Warning Condition Detected
>6000	Recoverable Error Detected
>8000	Fatal Error Detected
>C000	Abnormal Termination

The new error category, "recoverable errors", sets the

termination status code to >6000. A non-recoverable (or fatal) error (one detected in non-interactive mode) is now indicated by code >8000. Previously, fatal errors set the termination code to >6000. The other status codes are unchanged.

4.4 TX SUPPORT

This release of TIP no longer includes support for the TX990 or TX5 operating systems. In particular, runtime routines which supported these environments no longer exist, and all references have been removed from the Programmer's Guide.

4.5 OTHER DIFFERENCES

Any partial links or procedure segments wich contain old runtime routines should be relinked with the new runtime library to avoid potential problems with mixing runtime versions.

Previously, TIP allowed a maximum of 50 external routines and COMMON identifiers during any one compilation. There is no longer a restriction on the number of EXTERNAL or COMMON names that can appear in a source file. (STR 4961, 6499)

The runtime will no longer terminate the program if the it is unable to open the message file. Execution of the program will continue with the runtime supressing any attempt to write to the message file. (STR 10600)