HI-TECH PICC Release Notes for Version 9.50PL2

14th July 2006

THIS FILE CONTAINS IMPORTANT INFORMATION RELATING TO THIS COMPILER. PLEASE READ IT BEFORE RUNNING THIS SOFTWARE.

Description

This is a minor update from the previous version and adds some new features and corrects reported issues.

Notes

Please note that this version of the compiler has a new command line driver which has a different format for options compared with previous versions. Please refer to the user manual for details on using the command line options. During the transitional period, the new command line driver will accept the old-style options, but this should not be relied upon for future versions.

Previous Versions

The previous release version was 9.50, released October 2005.

New Features

General

HI-TIDE (9.50) The compiler now comes with HI-TIDE, an integrated development environent based on Eclipse. Please refer to the HI-TIDE readme file and accompanying documentation for details.

Mac OSX Version (9.50) In addition to the Windows, Linux and SUN versions, there is now a Mac OS X version of the compiler and HI-TIDE.

New Processors (9.50) Support for the latest processors has been added (10F220, 10F222, 12F510, 16F506, 16F639, 16F685, 16F687, 16F689, 16F690, 16F785, 16F913, 16F914, 16F916, 16F917, 16F946).

New Qualifier (9.50) New in v9.50 is the `eeprom` qualifier. An object using this qualifier will be stored in eeprom and can be read/written in the same manner as a regular RAM variable would. Due to efficiency concerns, only simple expressions involving eeprom qualified object are supported.

New Message System (9.50) All error, warning and advisory messages are now stored in language specific text files in the compiler's DAT directory. Each message is assigned a unique number which is displayed with the printed message.

In addition to error and warning messages issued by the compiler, there is now a third "advisory" type. Advisory messages are messages which offer some additional information about an operation the compiler is performing or a situation it has encountered. These are simply provided as helpful messages which don't fall into the error or warning category.

Language Support (9.50) The compiler now supports error, warning and advisory messages being displayed in languages other than English. Currently French and German are the other languages available.

Multiple Version Installation (9.50) The compiler now support multiple versions being installed. This means that in future you could install different versions of the compiler and have them all work correctly without interferring with each other.

Debug Information (9.50) There has been some improvements to the debugging information the compiler generates when dealing with assembler files. This improvement should improve the accuracy of stepping through assembler files for all debugging file formats.

Version Information (9.50) All components of the compiler (cpp, aspic, cgpic, hlink, etc...) now have a `--ver` option which will report what product and version they are apart of, and the date at which they were built. This has been provided to reduce confusion when interim patches are applied.

User Manual (9.50) The user manual has been updated to reflect changes in the compiler. Its format and layout have also been improved.

Make Utility (9.50) The GNU Make utility (and rm utility) is now included with the compiler distribution. Please refer to gpl.txt in the compiler's DOCS directory for license information. GNU Make sources and documentation can be obtained from the GNU web site. HI-TECH Software provides no warranty for this utility.

Driver

Time option (9.50PL2) The driver has a new option `--time`, which will display a summary of the amount of time the compiler took at each stage of compilation.

Hexmate Controls (9.50PL2) Additional options `--fill` and `--serial` have been added to the driver which allows easier control over hexmate. Among other things, these options allow unused memory ranges to be filled with a specific value and serial numbers to be inserted. Refer to the user manual for full details.

New driver (9.50) The compiler has a completely new command line driver. The new driver's options are different from the previous version, but is also compatible with the older style options. Running the compiler with the `--help` option will list them. The new options provide a more consistent interface and flexability in controlling things like memory configuration. Please refer to the user manual for details about the command line driver and new features it has to offer.

Start-up Code (9.50) The driver now has the ability to write custom start-up code for the application being build. In previous versions of the compiler there was a generic runtime startup routine which always got linked. Instead, the driver examines the application being build and "writes" a specifically-tailored routine. This is a completely automatic process and allows the application to have smaller, faster start-up code. In order for the compiler to do this, there is now two link stages when compiling a program.

Library Generation (9.50) The driver will now accept .lib as an output file type and generate a library for the files it is building. Previously is was necessary to manually run the libr utility to create a library file.

Hexmate Support (9.50) The Hexmate utility is now incorporated into the regular build process for applications. This means that HEX files are now accepted by the driver and when used, will merge the named hex file with the application which is being built. Hexmate also can provide additional summary information about addresses used within a hex file. This summary information can be turned on via the –summary option. **New in 9.50PL2**: The driver now has –serial and –fill options which drive hexmate to insert a serial number or fill unused memory ranges.

Specific debugger Support (9.50) Microchip's ICD1 and ICD2 require certain RAM and ROM locations to be reserved. The compiler has always supported this, however some addresses needed to be reserved by the ICD1 and ICD2 differ. In the case where the ranges required to be reserved on the ICD1 differed to the ranges required for the ICD2, previous versions of the compiler would reserve all of those locations just to be safe. The driver now has a –debugger option in which ICD1 or ICD2 can be directly selected and thus only the exact locations will be reserved. This selection is automatic when used inconjunction with MPLAB v6.50 and higher with the HI-TECH tool suite plug-ins.

Code Generator

Speed/Space Optimizations (9.50) The code generator now has the ability to generate code which is optimized for speed or space. This is controlled via the –opt command line option. By default, space optimziations will be used.

Assembler

New Optimizations (9.50) The assembler optimizer can now more accurately track and remove redundant bank swapping and ROM page selection instructions. This improvement also allows other optimizations to have more of an effect resulting in a code size reduction for many processors.

W Register Usage (9.50) The assembler optimizer will now utilize additional information about W regsiter usage passed to it from the code generator in order to perform extra optimizations.

Header Files

HTC.H (9.50) To reduce the work load required to migrate a project from PICC to PICC-18 or dsPICC, we have introduced a new top-level header file, `htc.h`, which can be used in place of where `PIC.H` would regularly be used. When HTC.H is included, it will include PIC.H, PIC18.H or dsPIC.h as appropriate.

Changes

General

HTC_PIC Environment Variable (9.50) It is no longer necessary to set the HTC_PIC environment variable. For the Windows version, the compiler obtains its path information from the Windows registry. For the Linux, SUN and Mac versions the path is obtained from the configuration XML file. The configuration XML file is shared by all HI-TECH compilers and its location is determined by the HTC_XML environment variable.

PATCH.LVL (9.50) The patch.lvl file previously located in the compiler's BIN directory is no longer used and is not included with the compiler. The compiler version can be obtain by running any component of the compiler with the –ver option.

HPDPIC (9.50) The text-based IDE HPDPIC is no longer included with the compiler. HI-TIDE or Microchip's MPLAB IDE are the recommended alternatives.

PICINFO.INI (9.50) The picinfo.ini file is now called picc.ini and has a different format. The new picc.ini file is located in the compiler's DAT directory.

COD File Generation (9.50) The default output file formats produced by previous versions of the compiler included HEX, COF and COD files. This has changed and now just HEX and COF files are generated by default. If you need a COD file generated, this can still be achived by using the –output option.

Start-up Bank (9.50) On power-up, the processor ensures that bank zero is selected. The runtime start-up code now assumes this to be the case whereas previously it did not. The effect of this is that if you have code which directly jumps to

address zero, you should first ensure that bank zero is selected. An alternative way to reset the processor is enter an “endless” loop and wait for the watch dog timer to expire and thus reset the processor, negating the need to manually select bank 0.

Installer

Activation changes (9.50) There have been changes to the way in which the compiler is activated. To simplify the process, only the compiler’s serial number is needed and it is now a requirement to have access to the internet. Activation is done on-line and is immediate. No personal information is transmitted in the process, only the serial number is sent.

Code Generator

String Packing (9.50) The string packing optimization found in previous versions of the compiler is now incorporated as part of the speed/space optimization selection. When space optimization is selected string packing will be used, when speed is selected it will be disabled.

Assembler

Psect Merging (9.50) A technique used in the assembler optimizer is to merge code psects. In previous versions of the assembler, any code psect may be merged together. This meant that in some instances where a custom psect is being used, the custom psect may get merged with another psect which results in the inability to specifically link the custom psect. This behaviour has changed so that only psects which have the string “text” in their name may be merged. If you have a custom psect and you don’t want it to be optimized with other psects, be sure its name does not include the substring “text”.

Limitations

Preprocessor

Incorrect result for sizeof The result of the `sizeof` preprocessor operator when applied to a pointer will always return 1. Use of the C sizeof operator will return the correct value.

Parser

Missing parenthesis not detected A missing end parenthesis is not detected in variable declarations when the variable is type cast. For example,

```
static char C @ ((unsigned)&PORTA;
```

This problem does not appear to affect any output code, but may cause problems when using third-party software tools.

Qualified arrays generate error The compiler generates an error when compiling complicated qualified arrays, e.g.:

```
const char * const * const listnames[] = {menu0, menu1};
```

The error is not issued if the array is not qualified, or for arrays of more basic types.

Structure initialization Locally defined structures cannot be initialized.

Code Generator

Functions returning ROM structure For Highend devices (17Cxxx), a function cannot return a copy of a structure which resides in ROM.

Indirect function calls For Highend devices (17Cxxx), certain indirect function calls with more than one byte of argument may produce incorrect code.

Functions as arguments to printf The argument list to printf or sprintf should not include function calls. For example,

```
printf("x = %f, sin(x) = %f\n", x, sin(x*3.141592/180.0));
```

This code demonstrates a workaround:

```
y = sin(x*3.141592/180.0);
printf("x = %f, sin(x) = %f\n", x, y);
```

Error on initialization of complex structures Any structure using structures within a structure or union cannot be directly initialized. For example, using the following types where a structure is a member of a union, initialization of the structure's bitfields will generate an error.

```
typedef struct {
    unsigned b0:1, b1:1, b2:1, b3:1, b4:1, b5:1, b6:1, b7:1
} byte_bits;
typedef union {
    byte_bits    bits;
    char         byte;
} byte_or_bits;
```

This will generate a compiler error:

```
byte_or_bits example = { {1,0,1,1,0,1,0,0 } };
```

Instead, use a single value:

```
byte_or_bits example = { 0b10110100 };
```

Multiple block variable declaration Functions which have variables declared within multiple blocks where code within the block requires the use of temporary memory, may get corrupted when compiled with global optimizations. A workaround is to move the inner variable declaration outsize of the block.

Void pointer size Void pointers are one byte in size and may not be large enough to point to all objects.

Libraries

Minimum floating point value The floating point math libraries cannot perform operations predictably on numbers below the order of `1e-35`. The result is either correct or zero.

Bug Fixes

The following are descriptions of bugs that were present in the previous version(s) of the compiler and have been fixed in this release.

Driver

Fixup errors in main (9.50PL2) In cases where the main() function was very large, fixup errors could be displayed. This was due to the way main was being linked. It is now linked differently, and in the case where main is too large, a can't find space error will instead be issued.

SFR definitions (9.50PL2) When compiling for a 17 series processor (17Cxxx), multiple or missing definitions of some SFRs (TBLPTR/PCL) occured. Changes to the runtime startup code generation has fixed this.

String Packing (9.50PL2) In some instances the driver enabled the string packing optimization on some mid-range processors that didn't support this feature. This has been corrected.

Custom Powerup (9.50PL2) When a custom powerup routine was used, incorrect linker options were issued causing build problems. Custom powerup routines are now handled correctly.

Timeout/Powerdown bits (9.50PL2) The driver has the ability to save the timeout and powerdown status bits at startup. This operation, however, was occuring too late and in some instances the bits got corrupted. The saving of these bits is now done earlier to avoid this problem.

OSCCAL Value (9.50PL2) On mid-range processors that have an oscillator value pre-programmed (eg: 12F675) the runtime startup code, in some instances, caluclated the location of the oscillator value and/or the corresponding SFR incorrectly. This has been fixed.

Spurious Warning (9.50PL1) For processors which have SFRs in banks where this is no general purpose RAM (for example 16F73), a warning "`psect intsave_X not specified in -P option`" may be displayed. This has been corrected.

Oscillator Constant (9.50PL1) For processors that have a pre-programmed oscillator constant (for example 12F675), the message "`undefined symbol osccal_val`" may be displayed. This has been corrected.

Windows XP SP2 Crash (9.50PL1) The driver could crash when the MPLAB ICD/ICD2 option is used in conjunction on computers that have Windows XP service pack 2. This has been fixed and is now compatible with Windows XP SP2.

Multiply Defined Symbol (9.50PL1) For some applications that are being compiled for base-line processors (12-bit chips), an error message may be given for a multiply defined symbol "`count`" and/or "`ptr`". This has been fixed.

Variable Initialization (9.50PL1) Variable initialization and clearing performed by the runtime startup code may cause the program counter to go astray, or, result in variables not correctly being initialized. This problem tended to occur when initializing or clearing variables on mid-range processors with multiple banks of general purpose RAM.

Code Generator

Byte Right Shift (9.50) Incorrect code could be generated for the right shift operator when shifting by four on the result of a function which returns a signed byte. This has been fixed.

Byte Left Shift (9.50) Fixed a problem when casting an unsigned char to be an unsigned long, left shifting by 24 bits and assigning to an unsigned long variable. One byte of the four byte assignment was being corrupted.

IRP bit corruption (9.50) The IRP bank select bit may get corrupted over a function call and not get set again prior to incrementing a variable which needs to be accessed indirectly (via a pointer). This has been fixed.

Assembler

Bank select bits (9.50PL2) In an unusual case of two if statements inside of a loop (for/while) where one if statement tests an object in bank 0 and the other an object in bank 1, and there is very little other code, a bank select instruction may have been incorrectly removed. This occurred when optimizations were turned on and the problem has now been fixed.

Page select bits (9.50) In some instances, as a result of assembler optimizations, a rare sequence of code could be generated where the ROM page is incorrectly set prior to a jump and after accessing a constant object. This has been corrected and new tests have been added to prevent this type of thing reoccurring.

Inlined Functions (9.50) A fault was detected with the function inlining optimization. The following things needed to occur for this problem to arise: (1) There are three functions (A, B, C). (2) Function A calls function B. (3) The only code in function B is a call to function C. (4) Assembler optimizations are turned on. (5) All three functions A, B, C are in the same C file. In this circumstance the assembler may turn the call from B to C into a jump and inline it into function A. The problem occurs because function C still has a return instruction, however no call was required to get to function C. The inlining optimization has now been corrected to detect such a situation.

Loop Detection (9.50) A speed optimization the assembler will attempt to do is to move some instructions outside of loops. In code which has a loop containing heavily nested if statements or many if, while, switch, for statements which access several objects located in different banks, some bank select instruction may get shifted outside of the loop. A problem was found with this where an incorrect bank was selected prior to accessing an object. The problem tended to occur only after several iterations where different paths through the if/while/switch/for statements were taken. It would be very rare that this problem would be encountered and changing almost anything in the code avoided the problem. This optimization has been fixed to correct this issue.

Linker

Missing Module Name (9.50PL1) When displaying error/warning messages, the name of the module for which the message refers was not being mentioned. This has been corrected and the module name is now shown.

Libraries

Wrong Bank (9.50PL2) A bank select instruction used in a common macro was missing which caused some issues with some of the low-level library routines. This problem occurred on mid-range processors that have four banks of RAM and with C code that calls these routines after just having accessed an object that required setting RP1 in the status register. This missing instruction has been added.

Watchdog Timeout (9.50) In applications which used large amounts of RAM that needed to be initialized, the watchdog timer may timeout before getting to main(). A CLRWDT instruction in now incorporated into the startup code.

Cromwell

Crash (9.50PL2) When debugging information was turned on, the -o option was used to output files into a directory other than where the source code resided and the current directory when the compiler was run is set to something other than the directory of the source file, cromwell may crash. This has been fixed.

Hexmate

Corruption on OSX (9.50PL2) When running on a Mac OSX, hexmate was generated corrupt HEX files. This has been fixed.

Processor configuration INI file

Memory Ranges (9.50PL2) The RAM ranges used for the 16F914 and 16F916 processor were incorrect. The correct values have been added.

Configuration Word (9.50) For 10F2XX parts, the location of the configuration word was incorrect. The correct location for these parts has been added.

ICD Ranges (9.50) The ICD reservation ranges for the 16F688 and 12F683 processors were incorrect. The correct ranges have been added.

Updates

Monitor the announcements forum on HI-TECH Software's web site for information relating to new versions or patches, and/or subscribe to our announcement mailing list - send email to HI-TECH Software's mailing list for more information.