SX-Key Version 2.02

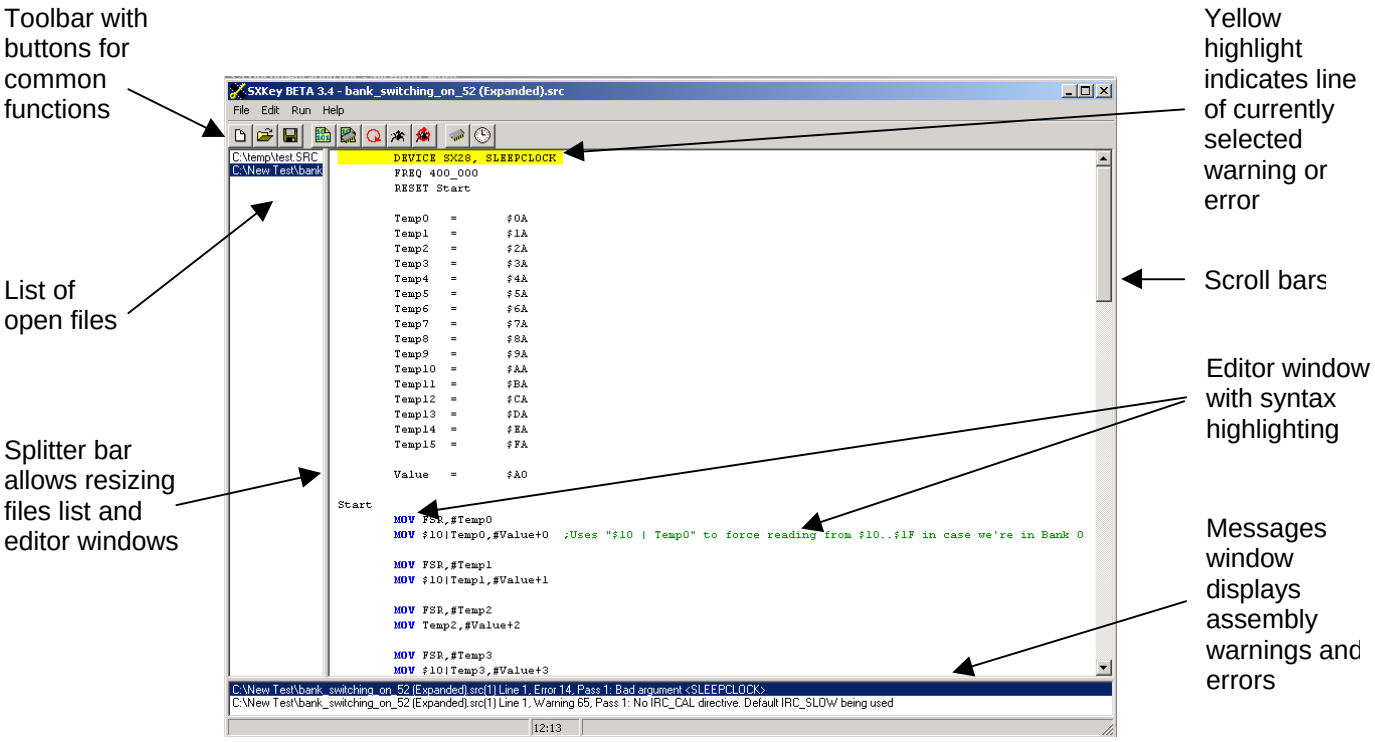
Contents

When the SX-Key editor starts for the first time, it will display the Configure window and the warranty information. Since there are new options in the Configure window, please take a moment to review it and set the options to your liking. See the Configure Window section, for more information.

What's New in Version 2.02

Editor Features

1) In addition to the items in v2.0 (listed below), the following was fixed/added in v2.02:
   a) Fixed bug causing the IDE to error if it tried to load an include file (from a user double-clicking on an error message) that used a partial path. Now both SASM and the IDE treat the path of the filename as the base path for other partial-path files.
   b) Added warning message to SASM when source code is missing an OSCxx device setting.
   c) Fixed bug where re-naming currently opened file outside the IDE caused timestamp error/loop.
   d) Fixed bug that caused incorrect highlighting of errors when using the new editor and the Parallax assembler.
2) The SX-Key Editor v2.0 has the following enhancements and new features:
   a) SCROLLBARS! Yes, you've wondered where they've been all this time and now they're here.
   b) Multiple UN-DO capability. Un-do is available even after a file has been saved.
   c) Colored syntax highlighting. You can choose to color code and/or boldface keywords and color code comments. The Configure Window allows setting the highlighting parameters.
   d) Multi-file capability allows you to open as many files as your machine's memory and resources will allow. The main editor window is divided vertically into a file list and an editor window. The size can be adjusted by dragging the vertical splitter to the desired position.
      i) Clicking on any file name in the open files list will display that file in the editor window.
      ii) To close a file, select the File → Close menu item.
   e) The "Find", "Find Next", and "Find/Replace" functions now use standard, Windows dialog boxes and control keys.
   f) If you are using the new assembler (SASM) as well as the new editor, you can have the IDE automatically jump to the first error after assembly (see the Configure window section).



3) The original editor is still available from the Run → Configure window.

4) All window positions and configuration changes are remembered between sessions.

5) The editor can automatically make backup copies of source files with the .BAK extension whenever they are saved. See the Configure Window section for more information.

6) It is possible to use any combination of the new/old assembler and new/old editor. However, it is HIGHLY recommended that you use the new editor and new assembler (SASM).

7) The editor has a toolbar. All buttons have tooltips that describe what each button does. The buttons are identical to their menu equivalents.

New File
File Open
File Save
Assemble
Program
Run
Debug
Debug Again
Device
Clock

Support for SASM

8) SASM is the new, default assembler. The original Parallax Assembler is also available from the Configure window.

9) When using SASM the following features are available:
   a) Code can have include files by using the new `INCLUDE` directive. Note that include files **only work if you have enabled the new (SASM) assembler**. Important points about using include files:
      i) The **include file may specify either a relative or a full file path**.
      ii) The maximum number of characters in the **include file path and name must *not* exceed 63 characters**. This is an internal limitation in the SASM compiler that will likely be addressed in the future.
   b) If a program is assembled with errors, the bottom of the editor window will split and display a "messages window" with a list of warnings and errors.
      i) This messages window will close itself automatically if there are no errors or warnings after assembly.
      ii) To manually hide the messages window after assembling, select the Edit → Clear Errors menu item.

   c) If you are using SASM in the new editor, double clicking on an error in the list will automatically highlight and jump to that line in the code. If the file is not loaded (which can happen if the error occurs in an include file), it will be automatically loaded as needed.

10) SASM has been enhanced internally to better support the SX-Key. One new feature is the addition of a predefined constant `__SASM` (SASM preceded by two underscores). The Parallax assembler needs to have the `DEVICE` directive as the first non-comment line. However, with `__SASM` predefined internally, you can write source code that will assemble under either the Parallax assembler or SASM.

A sample looks like this:

```
ifdef __SASM
     DEVICE  SASM-specific-settings
     More SASM specific items
else
     DEVICE  Parallax-specific-settings
     More Parallax specific items
endif
```

i) The sample code on the Ubicom website uses the constant `SX_Key` to conditionally compile. Unfortunately, this prevented the code from compiling under the Parallax assembler. To use Ubicom's sample code with the Parallax assembler:
   (1) Comment out the line that defines `SX_Key`
   (2) Do a search and replace, changing `ifdef SX_Key` to `ifndef __SASM`.

11) When using SASM and debugging, the new code window has a toolbar. All buttons have tooltips that describe what each button does.


Jump to Code : Scrolls the window to display the first area that assembles into CPU operations.


Jump to Reset Line : Scrolls the window to display the line of code that will be executed upon reset. The line will be highlighted blue.


Jump to Breakpoint : Scrolls the window to display the line with the breakpoint. The line will be highlighted red.


Jump to "Next Run" Line : Scrolls the window to display the next line of code that will be executed. The line will be highlighted blue.


Jump to Main : Scrolls the window to display the label called "Main" if there is one.

Watch and Break Directives

12) When debugging code generated by the ByteCraft SXC compiler, the WATCH and BREAK directives are now supported. To use them, incorporate the following macros into your C source code:

```
#define WATCH(VAR,SIZE,TYPE) #pragma debug w,(VAR,SIZE,TYPE)
#define BREAK() #pragma debug b,( )
```

a) The WATCH directive arguments follow the same form as the Parallax assembler.

VAR = The name of the variable
SIZE = Number of bits (or bytes if a string) to watch
TYPE = UDEC, SDEC, UHEX, SHEX, UBIN, SBIN, FSTR, ZSTR

b) To watch 8 bits of a variable called ThisVar in binary format, simply type the following into your source code:

```
WATCH(ThisVar, 8, UBIN);
```

c) To use the BREAK macro, simply type BREAK() in your source code.

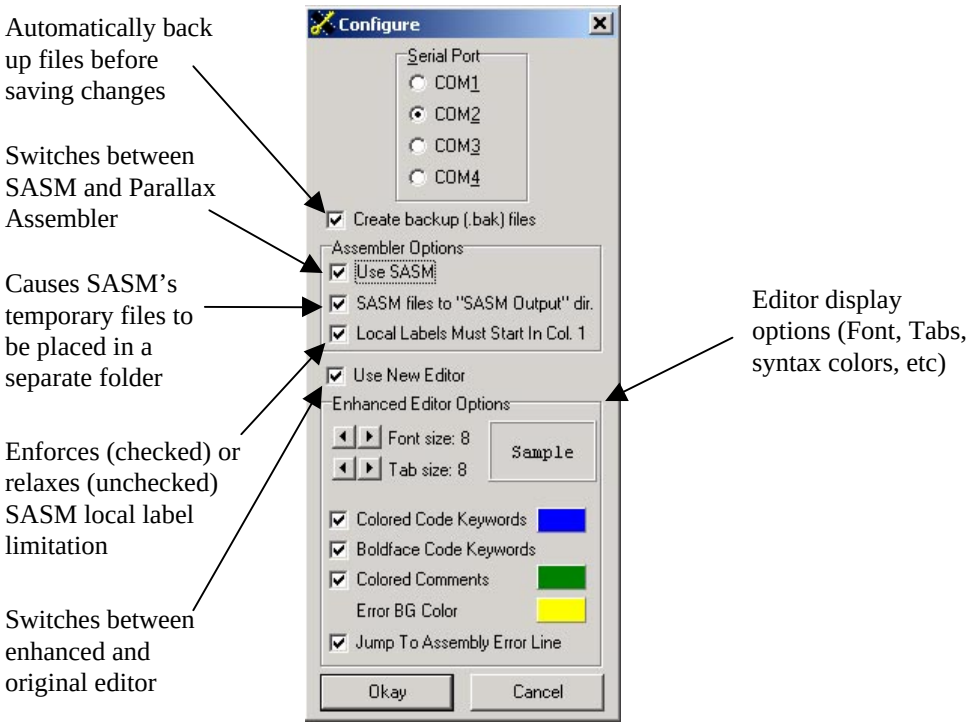
d) NOTE: When the SX-Key is invoked from the ByteCraft IDE, the following will occur:
   i) The main Editor window opens
   ii) If there are errors in the WATCH or BREAK directives, error messages will appear.
   iii) The Device Operation window will open and show the progress of programming the SX chip attached to the SX-Key.
   iv) The main Editor window will disappear
   v) The Debug windows will appear, leaving you ready to debug your code.

   When you are finished debugging, hit the Quit button on the Debug window, and control will return to the ByteCraft IDE.

For more detailed information on the WATCH and BREAK directives, consult the **SX-Key/Blitz! Development System Manual**.

Configure Window

The Configure window includes many more options to support the new features of version 2.0. This section discusses those options that might not be intuitively obvious.



1. The "Create backup (.bak) files" option, when selected, will cause the SX-Key software to create (or overwrite) a backup file (with the same name as your file but with a .bak extension) every time the source code is saved. The .bak file will be a copy of the previously-saved version of code.
2. The "Use SASM" option, when selected, will cause the SX-Key software to use SASM to assemble your source code. If left unchecked, the original Parallax Assembler will be used.
3. The "SASM files…" option, when selected, will cause the temporary files that SASM creates during assembly to be saved in a separate folder called "SASM Output" (within the installation directory). If not selected, those temporary files will be saved to the same directory as the source code.
4. The "Use New Editor" option, when selected, will make the SX-Key software use the new, enhanced editor rather than the original editor. It is recommended to use the new editor.
5. The "Font Size" option allows changing the font from 6 to 32 points.
6. The "Tab Size" option allows setting tabs to 2, 4, 6, or 8 spaces in width.
7. The "Colored Code Keywords" and "Boldface Code Keywords" options control how assembly keywords, such as `MOV`, `JMP`, etc, are displayed. If selected, the keywords are automatically displayed in the color indicated by the color box to the right of the option. Clicking on the color box will display a color-chooser window.
8. The "Colored Comments" options controls how assembly comments, everything to the right of a semicolon (`;`), are displayed. If selected, the comments are automatically displayed in the color

indicated by the color box to the right of the option. Clicking on the color box will display a color-chooser window.

9. The "Error BG Color" option determines the color used to highlight assembly errors within the code. Clicking on the color box will display a color-chooser window.
10. The "Jump to assembly error line" option causes the editor to automatically scroll the editor window so that the line containing the first assembly warning or error is visible. If this option is left unchecked, you will have to manually double-click on the warning or error in the messages window to have the IDE scroll the editor window.

Command Line Switches:

`/1`, `/2`, `/3`, `/4` tells the SX-Key IDE what serial port the key is attached to. While this is provided for backward compatibility, all configuration settings (including the serial port) are stored in the registry between settings. It is recommended that this switch **not** be used. NOTE: If a serial port is provided on the command line, it will override the value stored in the registry until the setting is manually changed in the Configure window.

`/R :` opens the file on the command line as **READ ONLY**. This is indicated in the title bar of the IDE by having the name of the source file followed by : READ ONLY. If the file is to be opened as a binary, EEPROM image file (.SXH), then the IDE will automatically display the device window with all but the "program", "Verify", and "Cancel" buttons disabled.

Important! -- Upgrading to the New IDE

The most critical change between the last released version of the software (v1.33) and this release (v2.0) is the inclusion of Ubicom's SASM assembler. SASM has been enhanced and integrated as the new, default assembler for the SX-Key software. While the original Parallax Assembler is still available as an option via the Configure window, SASM is the recommended assembler.

What impact does this have? On the plus side, features like include files, enhanced macros and user error messages are now available in the SX-Key IDE. On the negative side, you'll likely find that existing source code causes assembly errors they didn't have before. Since SASM accepts most of the Parallax Assembler's mnemonics, many of the errors generated by existing source code are trivial to fix. Of course, if you're in a hurry, you can switch back to the Parallax Assembler via the Configure window.

Upgrading Existing Code for the SASM Assembler

This section describes the most common changes that need to be made to use existing "Parallax-assembled" source code with the SASM assembler. These changes are summarized below and explained in detail in the Understanding SASM Errors and Warnings section.

Summary of Upgrade Procedures:

- Add an `IRC_CAL` directive to all existing projects.
- Add a `FREQ` directive to all existing projects.
- Modify appropriate device settings:
  - Replace `STACKX_OPTIONX` with either `STACKX` or `OPTIONX`, or `STACKX, OPTIONX`.
  - Replace `DRIVEOFF` with `XTLBUFD`.
  - Replace `FEEDBACKOFF` with `IFBD`.
  - Replace `SLEEPCLOCK` with `SLEEPCLK`.
  - Replace `DRT18MS` with `WDRT184`.
  - Replace `DRT960MS` with `WDRT960`.
  - Replace `DRT60MS` with `WDRT60`.
  - Replace `DRT60US` with `WDTR006`.
- Use `LIST Q = 37` to suppress "Literal truncated…" warnings.
- Modify any user-defined symbols that are reserved words in SASM.
- Add equates for Parallax Assembler symbols that are not reserved in SASM.

Understanding SASM Warnings and Errors

SASM reports two types of messages during the assembly process; warnings and errors. Warnings are informative messages about potential problems with the source code, but they do not halt the assembly or download process. Errors are critical messages indicating syntactic problems with the source code. Errors prevent assembly from completing successfully and, if invoked, the download process is prevented as well.

While warnings can sometimes provide useful information, many times they become a nuisance. SASM allows warnings to be suppressed with the LIST Q directive. The format is as follows:

```
LIST Q = {warning_number}
```

where warning_number is the actual number displayed by the warning message. For example, you might see a warning similar to the following:

…Line 15, Warning 37, Pass 2: Literal truncated to 8 bits

If this warning tells you something you already know, and it is just a nuisance to you, use the warning number, "…Warning 37…" in a `LIST Q` directive. For example, near the top of your code, enter:

```
LIST Q = 37
```

To suppress multiple warnings, such as #37 and #64, you can use the `LIST Q` directive below:

```
LIST Q = 37, Q = 64
```

If you desire to suppress a warning only in a specific area of the code, you can surround the code like this:

```
LIST Q = 37
mov  w, #-200
LIST Q = -37
```

The minus sign in front of the warning number will toggle it back to an active warning again.

IRC_CAL Directive
The most common warning with existing code is:

> "…Warning 65… No IRC_CAL directive. Default IRC_SLOW being used."

SASM uses a directive to configure the IRC (internal oscillator) calibration whereas the Parallax Assembler relies on the Configure window.

To configure the IRC and prevent this warning from appearing, enter one of the following directives: *(Note, you can also use the* `LIST Q=65` *directive to prevent this warning)*

```
IRC_CAL   IRC_SLOW
IRC_CAL   IRC_4MHZ
IRC_CAL   IRC_FAST
```

This is usually done below the `DEVICE` directive. For example, at the top of the source code, you may enter:

```
DEVICE    SX28, TURBO
IRC_CAL   IRC_SLOW
```

If you don't intend to use the SX chip's internal oscillator (IRC), use the `IRC_SLOW` or `IRC_FAST` setting rather than the `IRC_4MHZ` setting. The `IRC_4MHZ` setting always increases download time since the SX-Key needs to run special calibration routines.

The IRC_CAL message is just a warning, not an error. If you don't have an `IRC_CAL` directive, it won't prevent the code from assembling or downloading, however, it is recommended to include an `IRC_CAL` directive in all future projects that use SASM in the SX-Key editor.

OSCxxx Directive
Another common warning may be:

> "…Warning 69… No OSCxxx directive – using default OSCRC."

This message appears when there are no DEVICE directives that contain an oscillator setting, such as OSCXT2, OSCHS2, etc. Simply add the proper oscillator setting to the DEVICE directive to ensure proper function of the source code and to remove this warning.

Obsolete Keywords
SASM will issue the message, …Warning 51… Obsolete keyword: <*keyword*> if you use the Parallax Assembler's STACKX_OPTIONX, DRIVEOFF or FEEDBACKOFF device settings; though it will still set the proper fuse bits.

To prevent any of the warnings, shown below, from appearing

"WARNING: OBSOLETE KEYWORD: <STACKX_OPTIONX>…"
"WARNING: OBSOLETE KEYWORD: <DRIVEOFF>…"
"WARNING: OBSOLETE KEYWORD: <FEEDBACKOFF>…"

use the `LIST Q = 51` directive, or use either STACKX or OPTIONX in place of STACKX_OPTIONX, XTLBUFD in place of DRIVEOFF, or IFBD in place of FEEDBACKOFF.

FREQ Directive
The message

…Warning 66… No FREQ directive. Default 50 MHZ being used

simply indicates that, since no FREQ directive was specified in the source code, the SX-Key will use the default of 50 MHz for the on-board clock generator if a RUN or DEBUG option is used for downloading.

To prevent this warning from appearing, either use the FREQ directive (see the SX-Key manual) or the `LIST Q = 66` directive.

Literal Truncated To 8 Bits
The message,

…Warning 37… Literal truncated to 8 bits

indicates that SASM used only the lower 8 bits of a larger value when assembling the specified line of code. Virtually all SX code that uses the RTCC rollover interrupt will generate warning #37 when code such as this is assembled:

```
mov  w, #-200
```

Like all good warnings, SASM is merely trying to protect you from yourself. However, this warning is often unnecessary and erroneous. It is recommended that you include a

```
LIST Q = 37
```

directive in your code to suppress this warning.

Symbol is a Reserved Word
Other warnings that might come up are due to reserved words that are defined in SASM but not in the Parallax Assembler. See the SASM documentation. One example is "ZERO". Code such as:

```
zero equ  $00
```

will generate a Symbol is a reserved word error.

Undefined Symbols
The Parallax Assembler contains some predefined symbols that SASM does not. If any of these symbols exist in current Parallax Assembled code, you will need to add the necessary equates, shown below, for SASM to properly assemble them.

```
CMP                 =   $08   ;Comparator register
COMPARATOR          =   $08
DIRECTION           =   $0F   ;Tristate (direction) register
INDIRECT            =   $00   ;Indirect addressing register
LEVEL               =   $0D   ;Logic Level register
LVL                 =   $0D   ;Logic Level register
PLP                 =   $0E   ;Pull-up resister register
PORT_A              =   $05   ;RA i/o register
PORT_B              =   $06   ;RB i/o register
PORT_C              =   $07   ;RC i/o register
PULL_UP             =   $0E   ;Pull-up resister register
SCHMITT             =   $0C   ;Schmitt-Trigger register
ST                  =   $0C   ;Schmitt-Trigger register
TIMER_CAPTURE_HIGH  =   $01   ;High Timer Capture register   (48/52 only)
TIMER_CAPTURE_LOW   =   $00   ;Low Timer Capture register    (48/52 only)
TIMER_COMPARE1_HIGH =   $05   ;High Timer Compare register 1 (48/52 only)
TIMER_COMPARE1_LOW  =   $04   ;Low Timer Compare register 1  (48/52 only)
TIMER_COMPARE2_HIGH =   $03   ;High Timer Compare register 2 (48/52 only)
TIMER_COMPARE2_LOW  =   $02   ;Low Timer Compare register 2  (48/52 only)
TIMER_CONTROL_A     =   $07   ;Timer Control register A      (48/52 only)
TIMER_CONTROL_B     =   $06   ;Timer Control register B      (48/52 only)
TRIS                =   $0F   ;Tristate (direction) register
WAKE_EDGE           =   $0A   ;Wake-up Edge register
WAKE_ENABLE         =   $0B   ;Wake-up Enable register
WAKE_PENDING        =   $09   ;Wake-up Pending register
WKED                =   $0A   ;Wake-up Edge register
WKEN                =   $0B   ;Wake-up Enable register
WKPEN               =   $09   ;Wake-up Pending register
```

Bad Argument Error
SASM will issue the message, ...Error 14... Bad Argument: <*keyword*> if you use the Parallax Assembler's SLEEPCLOCK, DRT18MS, DRT960MS, DRT60MS or DRT60US device settings. Any of the errors below are "critical errors" that will prevent complete assembly and download.

```
ERROR: BAD ARGUMENT: <SLEEPCLOCK>
ERROR: BAD ARGUMENT: <DRT18MS>
ERROR: BAD ARGUMENT: <DRT960MS>
ERROR: BAD ARGUMENT: <DRT60MS>
ERROR: BAD ARGUMENT: <DRT60US>
```

To correct this error, use SLEEPCLK in place of SLEEPCLOCK, WDRT184 in place of DRT18MS,

`WDRT960` in place of `DRT960MS`, `WDRT60` in place of `DRT60MS` and `WDTR006` in place of `DRT60US`.

SASM Error and Warning Messages

1	Bad instruction statement
2	Redefinition of symbol <symbol_name>
3	Symbol <symbol_name> is not defined
4	Symbol is a reserved word
5	Missing operand(s)
6	Too many operands
7	Missing file register
8	Missing literal
9	Missing Label
10	Missing right parenthesis
11	Missing expression
12	Redefinition of MACRO label <label_name>
13	Bad expression
14	Bad argument <argument>
15	Bad MACRO expression
16	Macro arguments do not match
17	Unmatched MACRO
18	Bad IF-ELSE-ENDIF statement
19	Unmatched ELSE
20	Unmatched ENDIF
21	File nesting error - Exceeded 10 levels
22	If.else.endif nesting error - too deep
23	Invalid digit in numeric constant
24	Value is out of range
25	Bad radix value
26	Unknown microcontroller type
27	Unknown output format
28	Unknown listing parameter
29	Bad string syntax
30	Overwriting same program counter location
31	Expected an '=' sign
32	Unexpected EOF
33	Assume value is in HEXADECIMAL
34	Token length exceeds limit
35	Illegal character – Ignored

36	File register truncated to 5 bits
37	Literal truncated to 8 bits
38	Missing RAM Bank bits
39	No destination bit
40	Destination bit can only be 0 or 1
41	Bit number out of range
42	Destination address not in selected page
43	Address exceeds memory limit
44	Address is not within lower half of memory page
45	Label must begin at column 1
46	Ignoring unknown directive
47	REPT count exceed limit
48	File register not in current bank
49	MODE register value truncated to 4-bits
50	Expected a fr.bit operand
51	Obsolete keyword: <keyword> for this device
52	Reset address not in page 0
53	Applied non bitfield operator to a bitfield value
54	Overriding earlier target device declaration
55	ERROR "text"
56	Source line is too long
57	Local symbol "text" expands to more than 130 characters
58	Division by zero
59	Literal truncated to 12 bits
60	Couldn't open file: “filename”
61	Couldn't open include file: “filename”
62	Include path and file exceeds 64 characters
63	WATCH is missing parameters
64	IRC_CAL has invalid or missing parameters
65	No IRC_CAL directive. Default IRC_SLOW being used
66	No FREQ directive. Default 50 MHz being used
67	Total number of INCLUDE files exceeded 31
68	Tab expanded list file line too long - truncating
69	No OSCxxx directive - using default OSCRC
70	USER WARNING: "warning_message"

Assembling Source Code

When using SASM, every time the code needs to be assembled by selecting Assemble, Program, Run, or Debug options, the current source code will be saved if it has not be saved already. The SX-Key will warn you about this requirement at the moment it is about to assemble the code. You have the option of suppressing that warning dialog box by clicking the “Don’t show this dialog again” checkbox.

Known Issues:

- The SX-Key software does not support any serial ports other than COM1 through COM4.
- When using a WATCH directive within the ByteCraft SXC compiler, the variable being watched must be global in scope and the format argument (UDEC, SDEC, UHEX, SHEX, etc) must be entered in UPPERCASE. We are working with ByteCraft to fix this problem.