

# Contents

## A51 Assembler

[A51 Assembler Command \(Options Menu\)](#)

[A51 Assembler Options: Listing](#)

[A51 Assembler Options: Object](#)

[A51 Assembler Options: Misc](#)

## C51 Compiler

[C51 Compiler Command \(Options Menu\)](#)

[C51 Compiler Options: Listing](#)

[C51 Compiler Options: Object](#)

[C51 Compiler Options: Optimization](#)

[C51 Compiler Options: Memory Model](#)

[C51 Compiler Options: Misc](#)

## PL/M-51 Compiler

[PL/M-51 Compiler Command \(Options Menu\)](#)

[PL/M-51 Compiler Options: Listing](#)

[PL/M-51 Compiler Options: Object](#)

## BL51 Linker

[BL51 Code Banking LinkerCommand \(Options Menu\)](#)

[BL51 Code Banking Linker Options: Listing](#)

[BL51 Code Banking Linker Options: Linking](#)

[BL51 Code Banking Linker Options: Size/Location](#)

[BL51 Code Banking Linker Options: Additional](#)

[BL51 Code Banking Linker Options: Segments](#)

## dScope Debugger

[dScope Debugger Command \(Options Menu\)](#)

[dScope Debugger Command \(Run Menu\)](#)

[dScope Debugger Options Dialog Box](#)

## Environment Pathsspecs

[Environment Pathsspecs Command \(Options Menu\)](#)

[Environment Pathsspecs Dialog Box](#)

## **Make**

**Make Command (Options Menu)**

**Make Options: After Compile**

**Make Options: After Make**

**Make Options: Misc**

**Make Options: Extensions**

## **Project Files**

**Editing a Project**

## **ProROM EPROM Emulator**

**ProROM EPROM Emulator Command (Options Menu)**

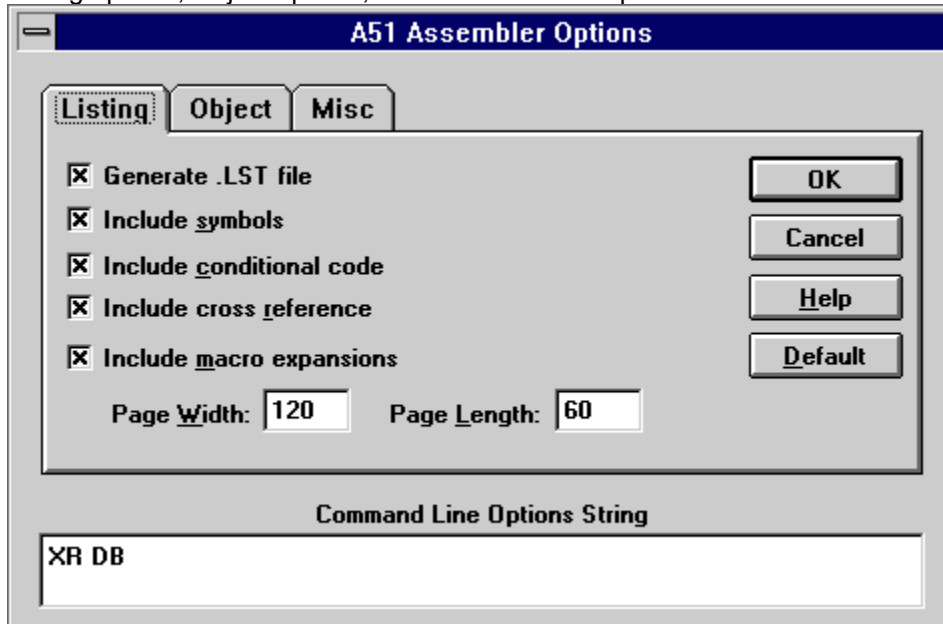
**ProROM Options**

## **A51 Assembler Command (Options Menu)**

Opens the A51 Assembler Options dialog box where you may set the options used during assembly. Options are provided for controlling the listing file and the object file generated by the assembler.

## A51 Assembler Options: Listing

The A51 Assembler Options dialog box lets you select options the assembler uses when it creates listing files and object files from your source files. Three groups of options are available in this dialog box: Listing options, Object options, and Miscellaneous options.



The Listing thumb tab shows the options that control how the assembler generates the listing file. Click on a control for more information.

### **Command Line Options String**

Shows the current command line options for the assembler. The text displayed in this box is updated to reflect the options you select.

**OK**

Saves the changes and exits the dialog box.

**Cancel**

Abandons any changes and exits the dialog box.

**Help**

Displays context sensitive help for the dialog box.



**Default**

Restores all settings to their default state.

**Generate .LST file**

Creates a listing file. This is equivalent to the **PRINT** assembler directive.

## **Include symbols**

Includes symbols in the listing file. This is equivalent to the **SYMBOLS** assembler directive.

**Include conditional code**

Includes conditional assembly code in the listing file. This is equivalent to the **COND** assembler directive.

**Include cross reference**

Includes a cross reference chart in the listing file. This is equivalent to the **XREF** assembler directive.

### **Include macro expansions**

Includes macro source in the listing file. This is equivalent to the **GEN** assembler directive.

## **Page Width**

Sets the number of characters on a line in the listing file. The number specified must be between 78 and 132. This is equivalent to the **PAGEWIDTH** assembler directive.

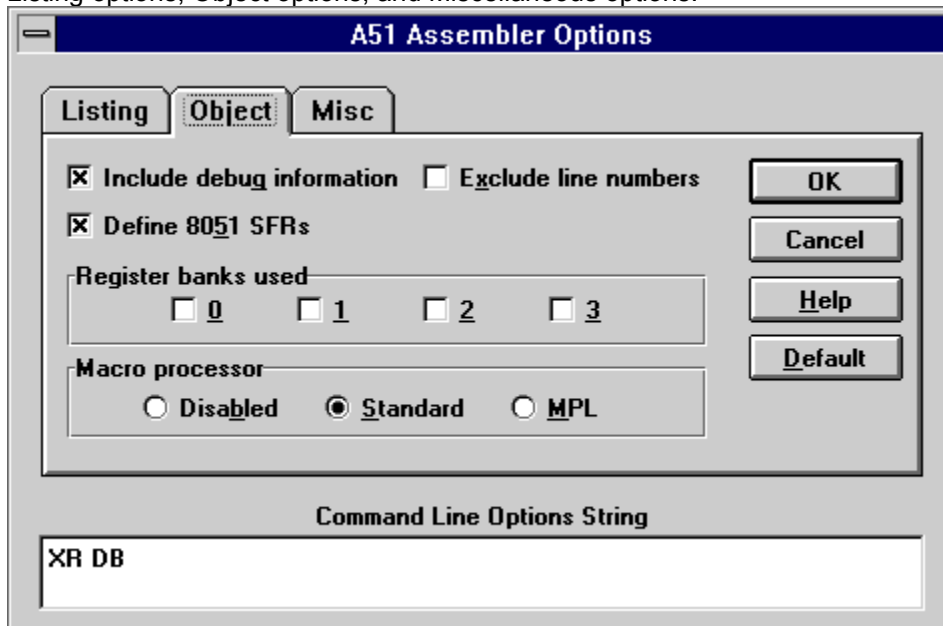
## **Page Length**

Sets the number of lines on a page in the listing file. The number specified must be between 10 and 65535. This is equivalent to the **PAGELength** assembler directive.



## A51 Assembler Options: Object

The A51 Assembler Options dialog box lets you select options the assembler uses when it creates listing files and object files from your source files. Three groups of options are available in this dialog box: Listing options, Object options, and Miscellaneous options.



The Object thumb tab shows the options that control how the assembler generates the object file. Click on a control for more information.

### **Include debug information**

Includes debugging information in the object file. This is equivalent to the **DEBUG** assembler directive.

**Exclude line numbers**

Excludes line number information from the object file. This is equivalent to the **NOLINES** assembler directive.

## **Define 8051 SFRs**

Defines the special function registers of the 8051. This is equivalent to the **MOD51** assembler directive.

## **Register banks used**

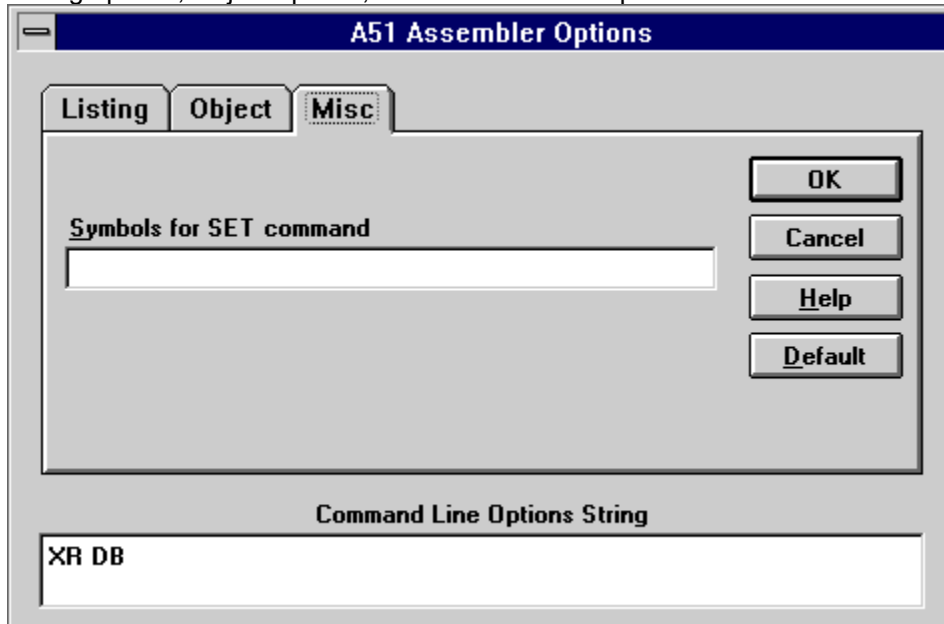
Specifies the register banks used. This is equivalent to the **REGISTERBANK** assembler directive.

**Macro processor**

Specifies the type of macro processor used to interpret macros. Standard interprets the default assembler macros. MPL interprets Intel ASM-51 macros. Disabled ignores macro definitions.

## A51 Assembler Options: Misc

The A51 Assembler Options dialog box lets you select options the assembler uses when it creates listing files and object files from your source files. Three groups of options are available in this dialog box: Listing options, Object options, and Miscellaneous options.



The Misc thumb tab shows miscellaneous assembler options. Click on a control for more information.

## **Symbols for SET command**

Lets you define symbols on the command line. This is equivalent to the **SET** assembler directive.

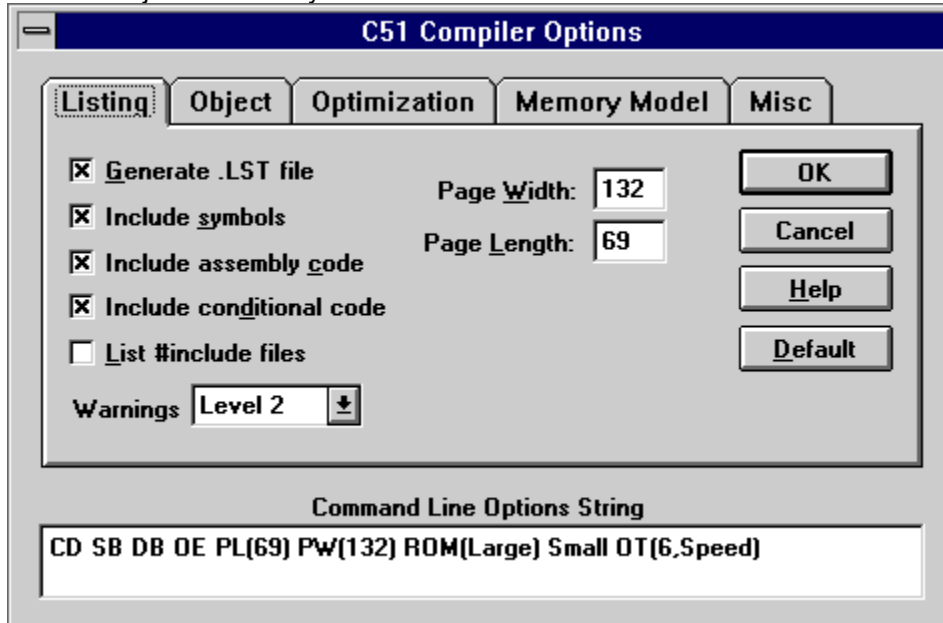


## **C51 Compiler Command (Options Menu)**

Opens the C51 Compiler Options dialog box where you may set the options used during compilation of your C source files. Options are provided for controlling how the listing file and object file are generated, the memory model used, and the level of optimization performed.

## C51 Compiler Options: Listing

The C51 Compiler Options dialog box lets you select options the C compiler uses when it creates listing files and object files from your source modules.



The Listing thumb tab shows the options that control how the compiler generates the listing file. Click on a control for more information.

### **Command Line Options String**

Shows the current command line for the compiler. The text displayed in this box is updated to reflect the options you select.

**OK**

Saves the changes and exits the dialog box.

**Cancel**

Abandons any changes and exits the dialog box.

**Help**

Displays context sensitive help for the dialog box.

**Default**

Restores all settings to their default state.

**Generate .LST file**

Creates a listing file. This is equivalent to the **PRINT** compiler directive.



**Include symbols**

Includes symbols in the listing file. This is equivalent to the **SYMBOLS** compiler directive.

### **Include assembly code**

Includes the generated assembly code in the listing file. This is equivalent to the **CODE** compiler directive.

**Include conditional code**

Includes conditional code in the listing file. This is equivalent to the **COND** compiler directive.

**List #include files**

Includes the contents of header files in the listing file. This is equivalent to the **LISTINCLUDE** compiler directive.

## Warnings

Specifies the level of warnings reported by the compiler. Higher warning levels indicate more severe warnings. This is equivalent to the **WARNINGLEVEL** compiler directive.

## **Page Width**

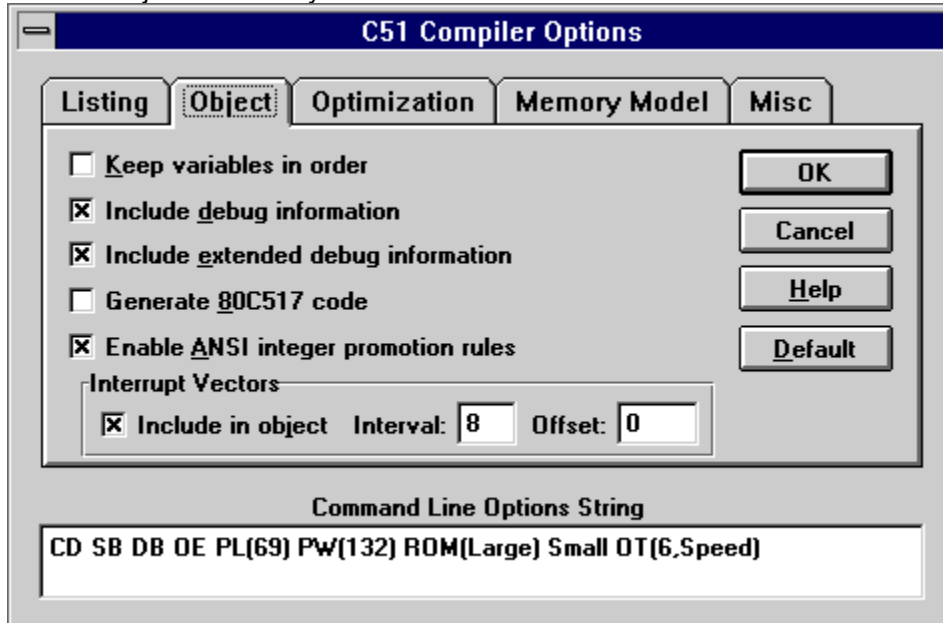
Sets the number of characters on a line in the listing file. The number specified must be between 78 and 132. This is equivalent to the **PAGEWIDTH** compiler directive.

## **Page Length**

Sets the number of lines on a page in the listing file. The number specified must be a number up to 65535. This is equivalent to the **PAGELength** compiler directive.

## C51 Compiler Options: Object

The C51 Compiler Options dialog box lets you select options the C compiler uses when it creates listing files and object files from your source modules.



The Object thumb tab shows the options that control how the compiler generates the object file. Click on a control for more information.



### **Keep variables in order**

Stored variables in memory in the order in which they are defined. This is equivalent to the **ORDER** compiler directive.

### **Include debug information**

Includes debugging information in the object file. This is equivalent to the **DEBUG** compiler directive.

**Include extended debug information**

Includes additional variable type information in the object file. This is equivalent to the **OBJECTTEXTEND** compiler directive.

### **Generate 80C517 code**

Generates code for the additional hardware components of the Siemens 80C517. This is equivalent to the **MOD517** compiler directive.

### **Enable ANSI integer promotion rules**

Promotes smaller types to integer expressions before comparison. This is equivalent fo the **INTPROMOTE** compiler directive.

## **Interrupt Vectors**

Sets parameters for interrupt vectors in the object file.

**Include in object**

Includes interrupt vectors in the object file. This is equivalent to the **INTVECTOR** compiler directive.

### **Interrupt Vector Interval**

Specifies the number of bytes between interrupt vectors stored in the object file. The default is 8. This is equivalent to the **INTERVAL** compiler directive.

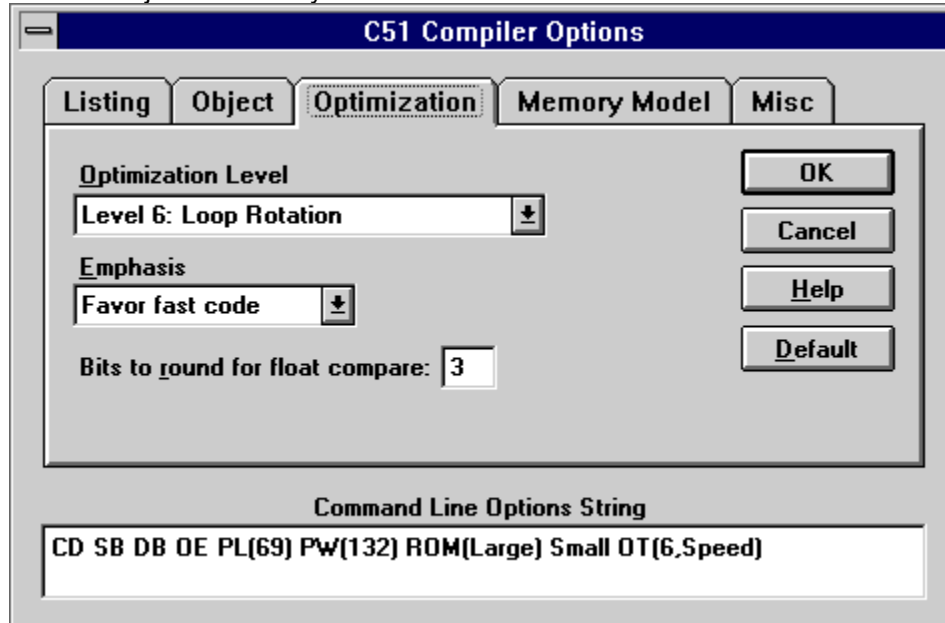


### **Interrupt Vector Offset**

Specifies the starting offset for the interrupt vector table in the object file. The default is 0. You may be required to change this offset when using the 8051 monitor. This is equivalent to the **INTVECTOR** compiler directive.

## C51 Compiler Options: Optimization

The C51 Compiler Options dialog box lets you select options the C compiler uses when it creates listing files and object files from your source modules.



The Optimization thumb tab shows the options that control how the compiler optimizes the generated code. Click on a control for more information.

## **Optimization Level**

Selects the level of optimization (0-6) performed by the compiler. Higher numbers indicate more optimization. This is equivalent to the **OPTIMIZE** compiler directive.

## **Emphasis**

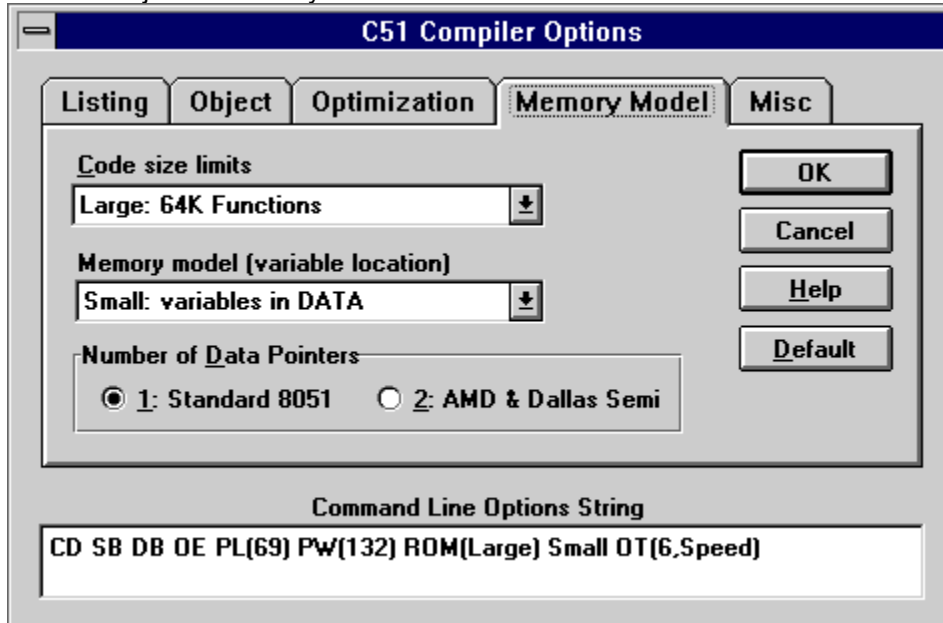
Selects the emphasis for optimization. The compiler can either generate faster code or smaller code. This is equivalent to the **OPTIMIZE** compiler directive.

**Bits to round for float compare**

Specifies the number of bit to use in floating-point comparisons. The higher the number, the more time floating-point comparisons require. The number specified must be between 0 and 7. This is equivalent to the **FLOATFUZZY** compiler directive.

## C51 Compiler Options: Memory Model

The C51 Compiler Options dialog box lets you select options the C compiler uses when it creates listing files and object files from your source modules.



The Memory Model thumb tab shows the options that control how the generated code uses code space and variable memory. Click on a control for more information.

**Code size limits**

Specifies which **CALL** and **JMP** instructions are used in the code generated by the compiler. This is equivalent to the **ROM** compiler directive.

## **Memory model**

Specifies the default location of all automatic and global variables. This is equivalent to the **SMALL**, **COMPACT**, and **LARGE** compiler directives.



**Number of Data Pointers**

Defines the number of data pointers the code generated by the compiler can use.

**1: Standard 8051**

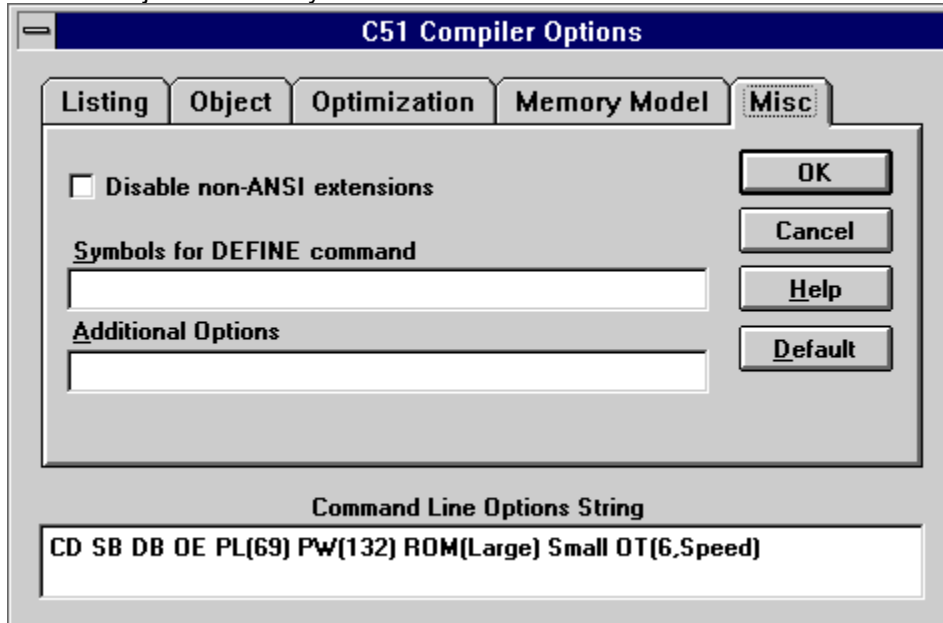
Specifies that only 1 data pointer is available.

## **2: AMD & Dallas Semi**

Specifies that 2 data pointers are available. This is equivalent to the **MODDP2** compiler directive.

## C51 Compiler Options: Misc

The C51 Compiler Options dialog box lets you select options the C compiler uses when it creates listing files and object files from your source modules.



The Misc thumb tab shows miscellaneous compiler options. Click on a control for more information.

### **Disable non-ANSI extensions**

Disables 8051-specific compiler extensions. This is equivalent to the **NOEXTEND** compiler directive.

## **Symbols for DEFINE command**

Defines names which may be queried by the preprocessor. This is equivalent to the **DEFINE** compiler directive.

## **Additional Options**

Specifies additional arguments on the command line passed to the compiler.

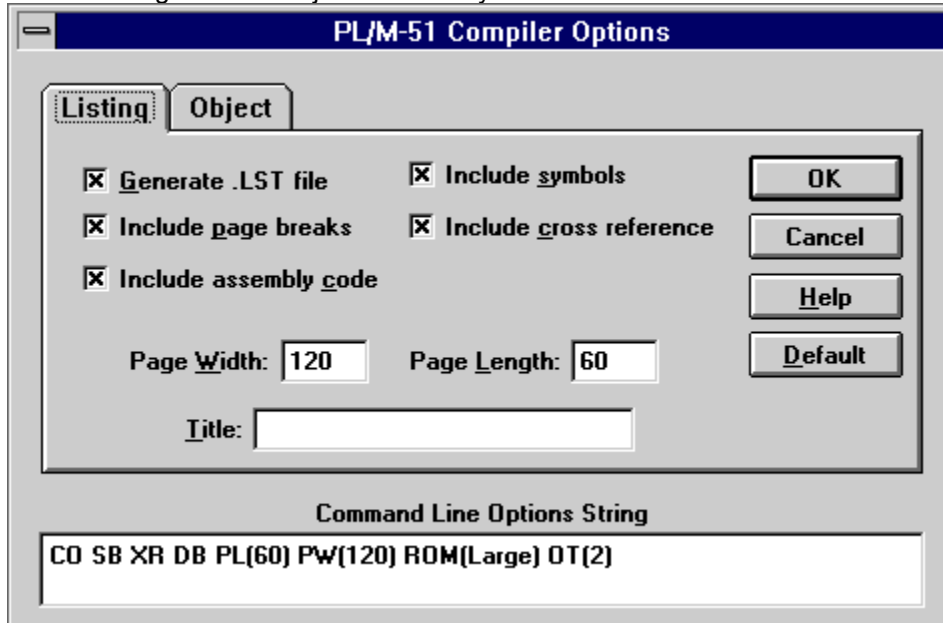
## **PL/M-51 Compiler Command (Options Menu)**

Opens the PL/M-51 Compiler Options dialog box where you may set the options used during compilation of your PL/M source files. Options are provided for controlling how the listing file and object file are generated.



## PL/M-51 Compiler Options: Listing

The PL/M-51 Compiler Options dialog box lets you select options the PL/M-51 compiler uses when it creates listing files and object files from your source modules.



The Listing thumb tab shows the options that control how the compiler generates the listing file. Click on a control for more information.

### **Command Line Options String**

Shows the current command line for the compiler. The text displayed in this box is updated to reflect the options you select.

**OK**

Saves the changes and exits the dialog box.

**Cancel**

Abandons any changes and exits the dialog box.

**Help**

Displays context sensitive help for the dialog box.

**Default**

Restores all settings to their default state.

**Generate .LST file**

Creates a listing file.

**Include page breaks**

Includes page breaks in the listing file.



**Include assembly code**

Includes the generated assembly code in the listing file.

**Include symbols**

Includes symbols in the listing file.

**Include cross reference**

Includes a cross reference table in the listing file.

**Page Width**

Sets the number of characters on a line in the listing file. The number specified must be between 78 and 132.

**Page Length**

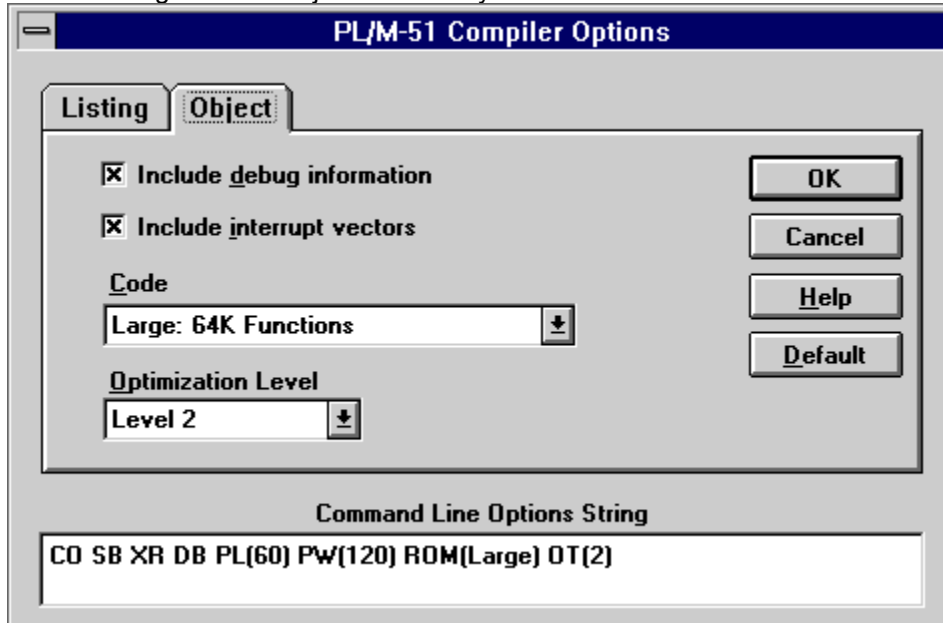
Sets the number of lines on a page in the listing file. The number specified must be a number from 4 to 65535.

**Title**

Includes a title in the header of the listing file.

## PL/M-51 Compiler Options: Object

The PL/M-51 Compiler Options dialog box lets you select options the PL/M-51 compiler uses when it creates listing files and object files from your source modules.



The Object thumb tab shows the options that control how the compiler generates the object file. Click on a control for more information.

**Include debug information**

Includes debugging information in the object file.



**Include interrupt vectors**

Includes interrupt vectors in the object file.

**Code**

Specifies which **CALL** and **JMP** instructions are used in the code generated by the compiler.

**Optimization level**

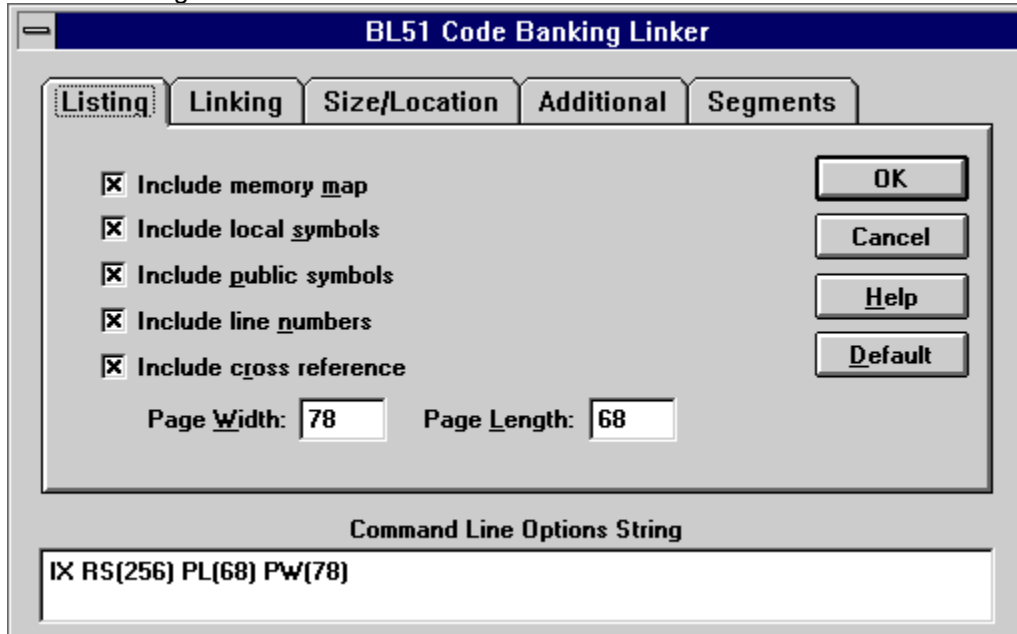
Selects the level of optimization (0-3) performed by the compiler. Higher numbers indicate more optimization.

## **BL51 Code Banking LinkerCommand (Options Menu)**

Opens the BL51 Code Banking Linker Options dialog box where you may set the options used when linking your project. Options are provided for controlling how the listing file and absolute object file are generated. You may also use the controls in this dialog box to specify segment size and location.

## BL51 Code Banking Linker Options: Listing

The [BL51](#) Code Banking Linker dialog box lets you select options the linker uses when it creates your executable target file.



The Listing thumb tab shows controls the linker uses when creating a map or listing file. Click on a control for more information.

### **Command Line Options String**

Shows the current command line for the linker. The text displayed in this box is updated to reflect the options you select.

**OK**

Saves the changes and exits the dialog box.

**Cancel**

Abandons any changes and exits the dialog box.



**Help**

Displays context sensitive help for the dialog box.

**Default**

Restores all settings to their default state.

**Include memory map**

Includes a map of the memory used by the application in the listing file.

**Include local symbols**

Includes a table of the local symbols in the listing file.

**Include public symbols**

Includes a table of the public symbols in the listing file.

**Include line numbers**

Includes a table of line number information in the listing file.

**Include cross reference**

Includes a corss reference in the listing file.

**Page Width**

Sets the number of characters on a line in the map file. The number specified must be between 78 and 132.

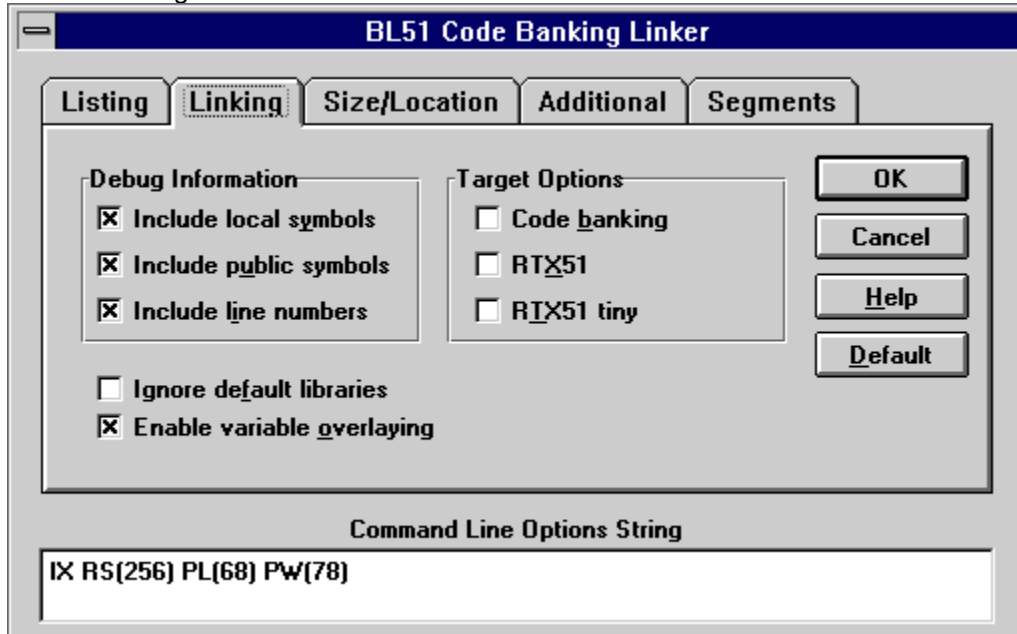


**Page Length**

Sets the number of lines on a page in the map file. The number specified must be a number from 10 to 65535.

## BL51 Code Banking Linker Options: Linking

The BL51 Code Banking Linker dialog box lets you select options the linker uses when it creates your executable target file.



The Linking thumb tab shows controls the linker uses when creating an absolute object module or a banked object module. Click on a control for more information.

**Include local symbols**

Includes local symbol definitions in the absolute object module.

**Include public symbols**

Includes public symbol definitions in the absolute object module.

**Include line numbers**

Includes line number information in the absolute object module.

**Code banking**

Creates a banked object module that includes a separate program memory area for each code bank.

**RTX51**

Creates a real-time application using the RTX-51 operating system.

**RTX51 tiny**

Creates a real-time application using the RTX-51 tiny operating system.



**Ignore default libraries**

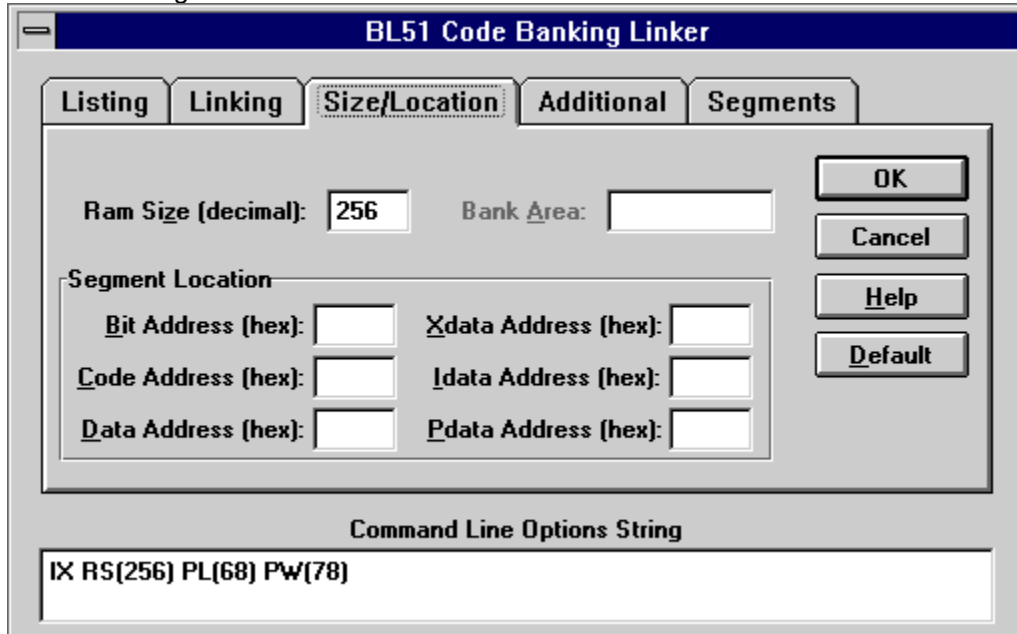
Excludes default library modules from the absolute object module.

**Enable variable overlaying**

Overlays variables in the internal data memory.

## BL51 Code Banking Linker Options: Size/Location

The [BL51](#) Code Banking Linker dialog box lets you select options the linker uses when it creates your executable target file.



The screenshot shows the "BL51 Code Banking Linker" dialog box with the "Size/Location" tab selected. The dialog has a title bar and five tabs: "Listing", "Linking", "Size/Location", "Additional", and "Segments". The "Size/Location" tab contains the following controls:

- "Ram Size (decimal):" with a text box containing "256".
- "Bank Area:" with an empty text box.
- A "Segment Location" section containing six text boxes for addresses:
  - "Bit Address (hex):" (empty)
  - "Xdata Address (hex):" (empty)
  - "Code Address (hex):" (empty)
  - "Idata Address (hex):" (empty)
  - "Data Address (hex):" (empty)
  - "Pdata Address (hex):" (empty)
- Four buttons on the right: "OK", "Cancel", "Help", and "Default".
- A "Command Line Options String" section at the bottom with a text box containing "IX RS(256) PL(68) PW(78)".

The Size/Location thumb tab shows controls that specify the size or location of different memory areas. Click on a control for more information.

**Ram Size**

Specifies the size of the internal data area. For 8051-compatible parts, this value should be 128. For 8052-compatible parts, this value is 256. Other derivatives may have as little as 64 bytes of internal data memory.

**Bank Area**

Specifies the starting and ending address for the code banking area. Specify these two numbers separated by a comma. For example:

0x8000,0xFFFF

**Bit Address**

Specifies the starting memory address for segments placed in the bit-addressable internal data memory. The address must be from 00 to 7F.

**Code Address**

Specifies the starting memory address for segments placed in program memory. The address must be from 0000 to FFFF.

**Data Address**

Specifies the starting memory address for segments placed in the internal data memory. The address must be from 00 to 7F.



**Xdata Address**

Specifies the starting memory address for segments placed in the external data memory. The address must be from 0000 to FFFF.

**Idata Address**

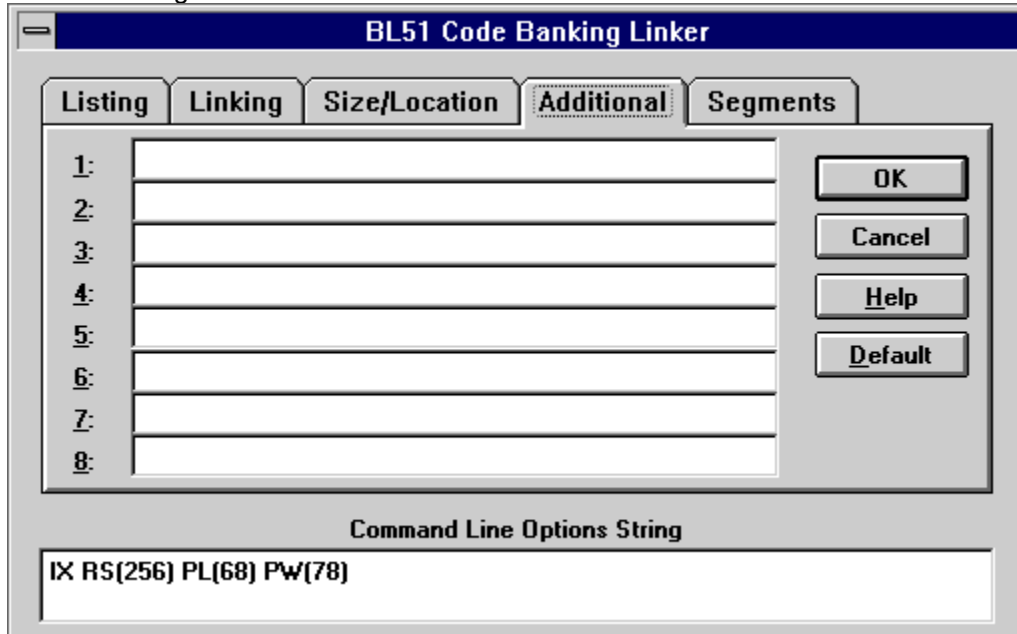
Specifies the starting memory address for segments placed in the indirectly addressable internal data memory. The address must be from 00 to FF.

**Pdata Address**

Specifies the starting memory address for **PDATA** segments placed in the external data memory. The address must be from 0000 to FFFF.

## BL51 Code Banking Linker Options: Additional

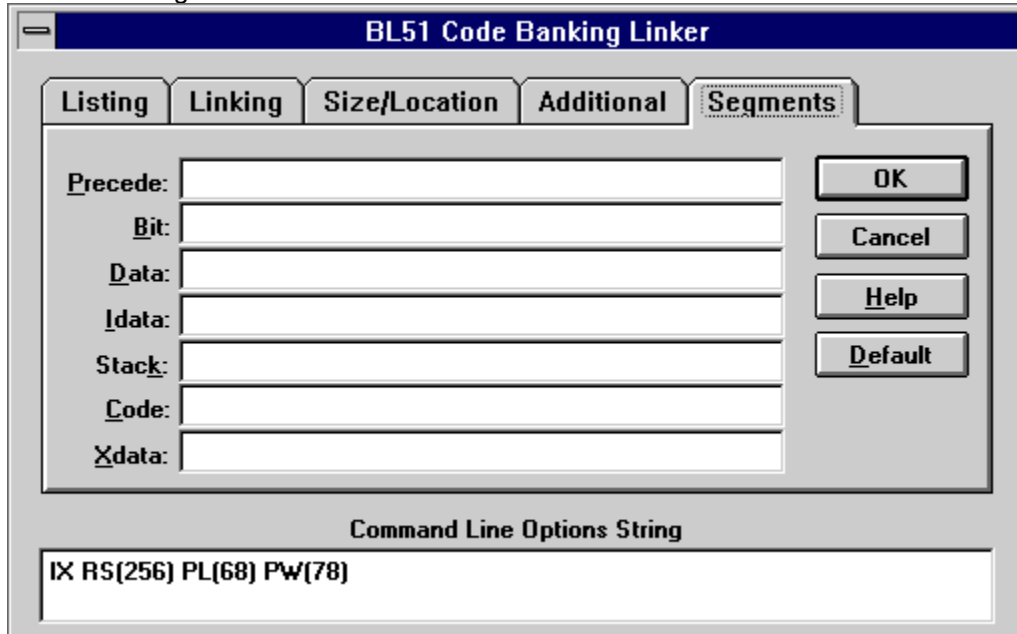
The BL51 Code Banking Linker dialog box lets you select options the linker uses when it creates your executable target file.



The Additional thumb tab shows 8 input lines you may use for additional linker directives.

## BL51 Code Banking Linker Options: Segments

The BL51 Code Banking Linker dialog box lets you select options the linker uses when it creates your executable target file.



The screenshot shows the 'BL51 Code Banking Linker' dialog box with the 'Segments' tab selected. The dialog has five tabs: 'Listing', 'Linking', 'Size/Location', 'Additional', and 'Segments'. The 'Segments' tab contains several input fields for segment names and their locations, and a 'Command Line Options String' field at the bottom. The 'Command Line Options String' field contains the text: 'IX RS(256) PL(68) PW(78)'. On the right side of the dialog, there are four buttons: 'OK', 'Cancel', 'Help', and 'Default'.

Field Label	Value
Precede:	
Bit:	
Data:	
Idata:	
Stack:	
Code:	
Xdata:	

Command Line Options String

IX RS(256) PL(68) PW(78)

The Segments thumb tab shows the controls for locating and ordering segments. Enter multiple segments separated by commas. Enter locations in parentheses following segment names. For example:

```
?PR?_ST_SCROLL_TEXT?SETUP(1000h),?PR?_ST_DONE?SETUP
```

Click on a control for more information.

**Precede**

Specifies the order and location of segments that precede all others in internal data memory.

**Bit**

Specifies the order and location of **BIT** segments.

**Data**

Specifies the order and location of **DATA** segments.



**Idata**

Specifies the order and location of **IDATA** segments.

**Stack**

Specifies the order and location of segments that follow all others in internal data memory.

**Code**

Specifies the order and location of **CODE** segments.

**Xdata**

Specifies the order and location of **XDATA** segments.

## **dScope Debugger Command (Options Menu)**

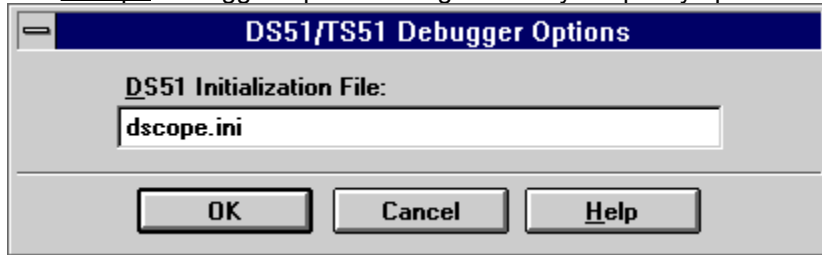
Opens the dScope Debugger Options dialog box where you may set the initialization file used by the debugger.

## **dScope Debugger Command (Run Menu)**

Runs the dScope Debugger. Use this command to invoke the dScope Debugger/Simulator from within  $\mu$ Vision. The initialization file specified in the dScope Debugger Options dialog box is used to setup the dScope options for this project.

## dScope Debugger Options Dialog Box

The dScope Debugger Options dialog box lets you specify options for the dScope debugger.



Click on a control for more information.

**dScope Initialization File**

Specifies the name of the initialization file for the dScope debugger.

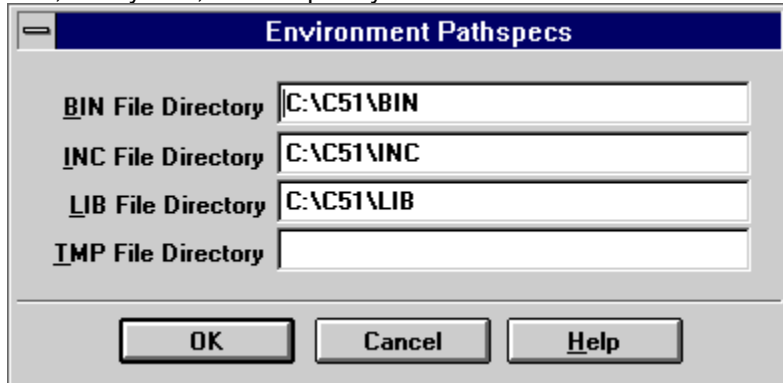


## **Environment Pathspecs Command (Options Menu)**

Opens the Environment Pathspecs dialog box where you may set the path to the development tools executable programs (compiler, assembler, and linker), the path to the standard C include files, and the path the run-time library files.

## Environment Pathspecs Dialog Box

The Environment Pathspecs dialog box lets you specify the path where the development tools, include files, library files, and temporary files are stored.



Click on a control for more information.

**OK**

Saves the changes and exits the dialog box.

**Cancel**

Abandons any changes and exits the dialog box.

**Help**

Displays context sensitive help for the dialog box.

**BIN File Directory**

Specifies the directory where the C51 compiler, A51 assembler, BL51 linker, and dScope debugger executable programs are located.

**INC File Directory**

Specifies where the standard C header files are located.

**LIB File Directory**

Specifies where the standard C library files are located.



**TMP File Directory**

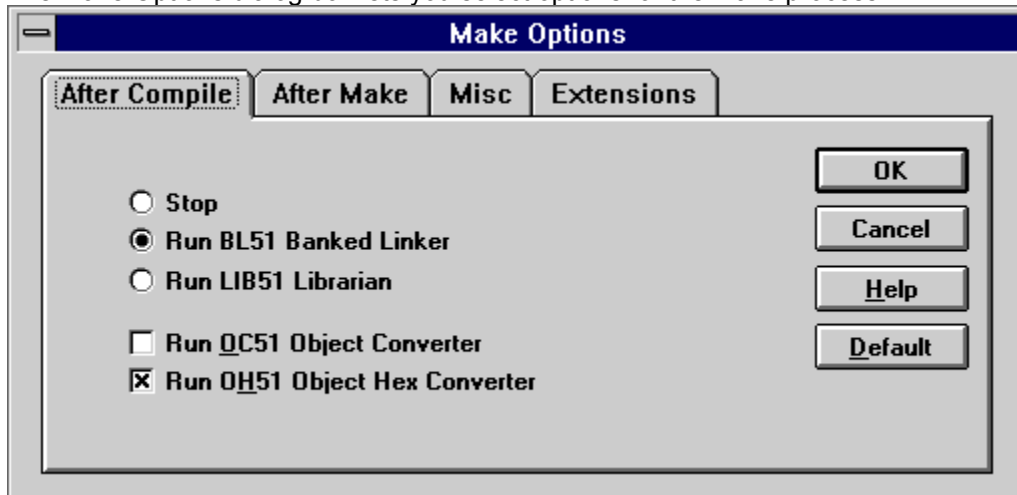
Specifies where temporary files created by the compiler, assembler, and linker are located.

## **Make Command (Options Menu)**

Opens the Make Options dialog box where you specify how  $\mu$ Vision builds the files in your project.

## Make Options: After Compile

The Make Options dialog box lets you select options for the make process.



The After Compile thumb tab shows what  $\mu$ Vision does after all files in the project are assembled and compiled. Click on a control for more information.

**OK**

Saves the changes and exits the dialog box.

**Cancel**

Abandons any changes and exits the dialog box.

**Help**

Displays context sensitive help for the dialog box.

**Default**

Restores all settings to their default state.

**Stop**

Specifies that no further operations are started.



**Run BL51 Banked Linker**

Specifies that the project's object files are linked by the banked linker into an absolute object module.

**Run LIB51 Librarian**

Specifies that the project's object files are stored in a library by the library manager.

### **Run OC51 Object Converter**

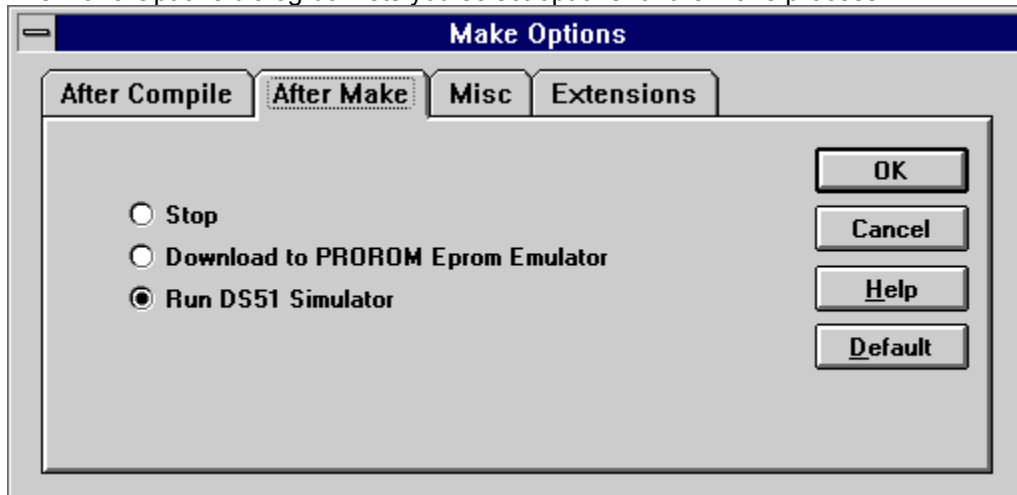
Specifies that the banked object module, created by the linker, is converted into absolute object modules for each code bank.

### **Run OH51 Object Hex Converter**

Specifies that the absolute object modules created by the OC51 object convert or the BL51 code banking linker are converted into Intel HEX files.

## Make Options: After Make

The Make Options dialog box lets you select options for the make process.



The After Make thumb tab shows what  $\mu$ Vision does after all files in the project are processed and the linker, library manager, and object converters have been run. Click on a control for more information.

**Stop**

Specifies that no further operations are started.

**Download to ProROM**

Specifies that the target program is downloaded to the ProROM EPROM Emulator.

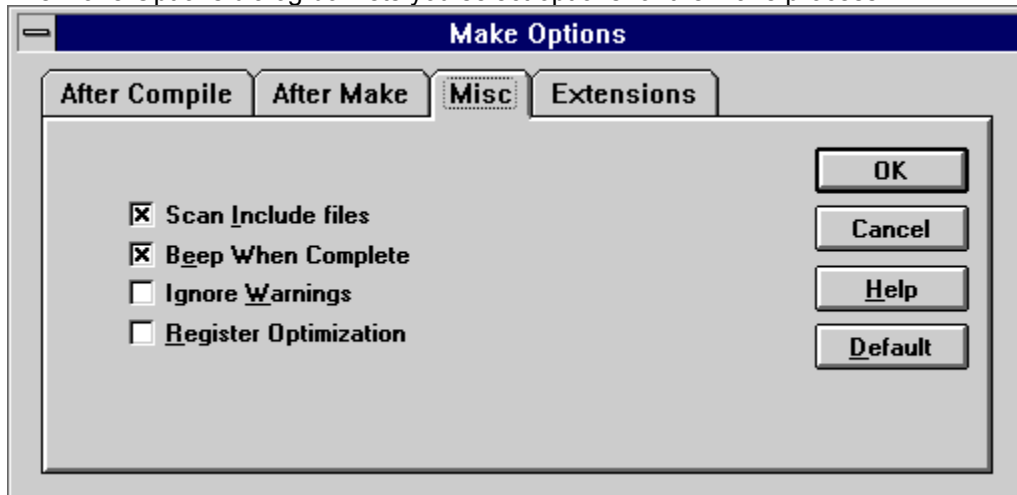
**Run dScope**

Specifies that the dScope Debugger is started.



## Make Options: Misc

The Make Options dialog box lets you select options for the make process.



The Misc thumb tab shows miscellaneous options used during the make process. Click on a control for more information.

## **Scan Include Files**

Scans your source files to determine if an included header file has been updated more recently than the object file. If one has, the object file is rebuilt.

## **Beep When Complete**

Beeps the speaker when the make process is complete.

**Ignore Warnings**

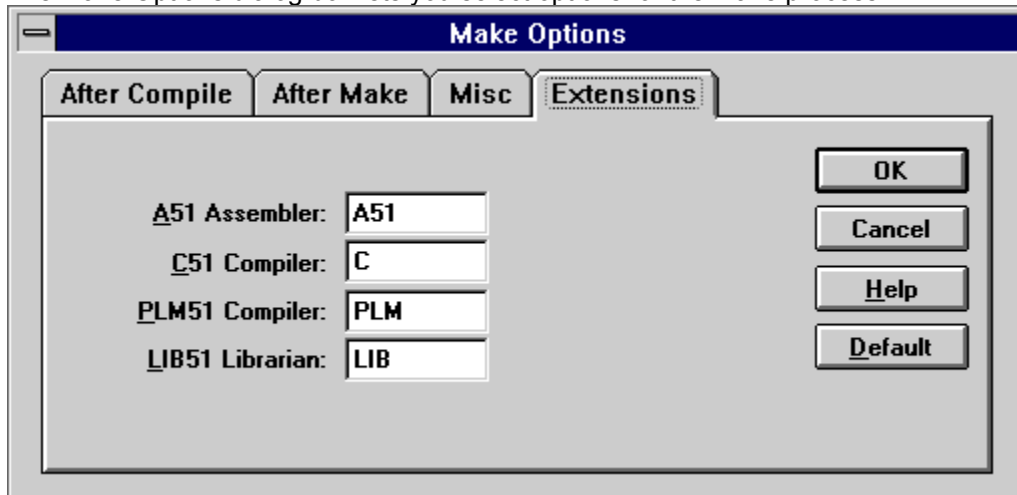
Ignores all warnings emitted by the tools during the make process. Use this control with caution.

## **Register Optimization**

Optimizes inter-module register usage by recompiling components of a project. Using this control increases the time required to build a project.

## Make Options: Extensions

The Make Options dialog box lets you select options for the make process.



The Extensions thumb tab shows the file extensions used by your source and library files. Click on a control for more information.

**A51 Assembler**

Specifies the file extension used for 8051 assembly source files.

**C51 Compiler**

Specifies the file extension used for 8051 C source files.



**PL/M-51 Compiler**

Specifies the file extension used for 8051 PL/M-51 source files.

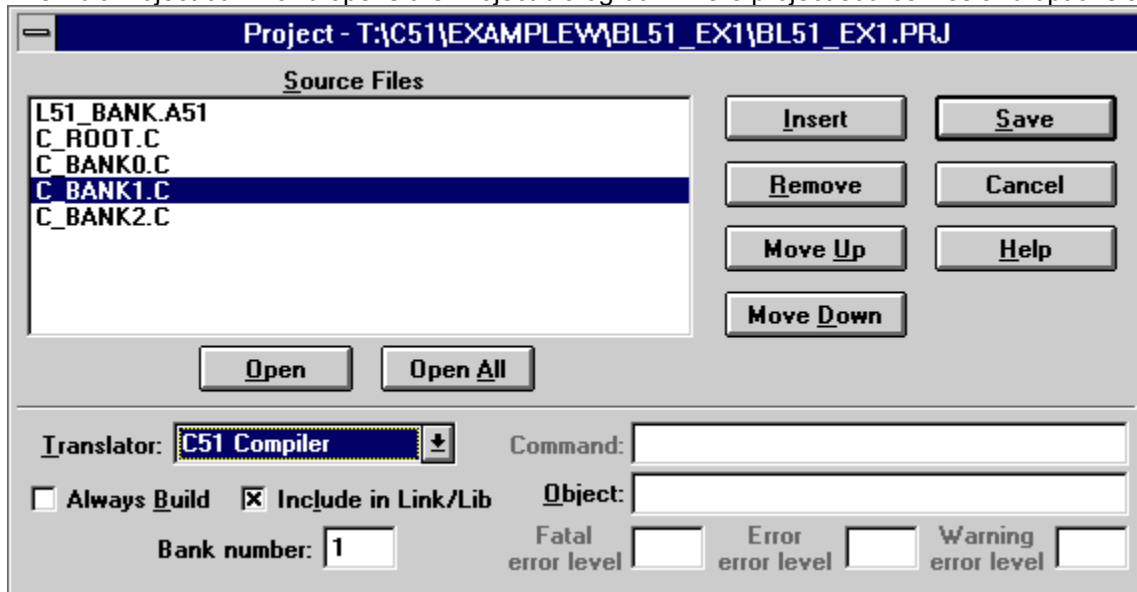
**LIB51 Librarian**

Specifies the file extension used for 8051 library files.

## Editing a Project

Use the Edit Project command from the Project menu to edit the properties for the current project.

The Edit Project command opens the Project dialog box where project source files and options are listed.



### Related Topics:

[Adding a Source File](#)

[Changing Properties for a Source File](#)

[Removing a Source File](#)

[Moving a Source File](#)

## **Adding a Source File**

When you create a project file, you must add source files using the Project dialog box.

### **To add a file to the project list...**

1. Select the Insert pushbutton.
2. Use the Add File to Project dialog box to select the file to add.

### **HINT**

You may type **Alt+I** to rapidly select the Insert pushbutton.

---

µVision automatically fills in the Translator type and other properties for file types that it recognizes. You must manually enter properties for any other file types.

Make sure you select the Save pushbutton when you are through making changes.

## **Changing Properties for a Source File**

After a source file has been added to the project, you may decide to change some of its properties.

### **To change the properties for a file in the source file list...**

1. Select the file whose properties you wish to change.
2. Enter the correct information for the source file at the bottom of the dialog box.

Make sure you select the Save pushbutton when you are through making changes.

## **Removing a Source File**

You may remove unnecessary files from the source file list.

### **To remove a file from the source file list...**

1. Select the file you want to remove.
2. Select the Remove pushbutton.

### **NOTE**

The remove operation cannot be undone. Make sure you select the correct source file before you proceed.

---

Make sure you select the Save pushbutton when you are through making changes.

## Moving a Source File

The project manager lets you move the files in the source file list.

### To move a program in the source file list...

1. Select the file you want to move.
2. Select the Move Up pushbutton to move the file up one position in the list.
3. Select the Move Down pushbutton to move the file down one position in the list.

### HINT

You may type **Alt+U** to move the file up or **Alt+D** to move the file down.

---

Make sure you select the Save pushbutton when you are through making changes.

**Source Files**

Lists the files included in the project. Properties for each file are listed in the bottom portion of the Project dialog box.



**Open**

Opens the selected source file.

**Open All**

Opens all files in the project.

**Insert**

Inserts a file into the project. This pushbutton opens the Add File to Project dialog box where you may select the drive, directory, and file to insert.

**Remove**

Removes the selected file from the project.

**Move Up**

Moves the selected file up in the project list.

**Move Down**

Moves the selected file down in the project list.

**Translator**

Specifies which translator (compiler or assembler) to use to build the selected file. Also lets you specify special files like text files, library files, and user programs.

**Object**

Specifies the object file name created by the translator for the selected file.



## **Always Build**

Specifies that make always build the selected file.

**Bank number**

Specifies the bank number for each file in a bank switching application.

**Include in Link/Lib**

Specifies that the selected file is processed by the linker or library manager when build the target.

**Command**

Displays the command line for the custom translator or the modules to include from a library file.

**Fatal error level**

Displays the errorlevel returned that indicates a fatal error occurred. Leave this text box empty if there is no fatal errorlevel.

**Error error level**

Displays the errorlevel returned that indicates an error occurred. Leave this text box empty if there is no error errorlevel.

**Warning error level**

Displays the errorlevel returned that indicates a warning was issued. Leave this text box empty if there is no warning errorlevel.

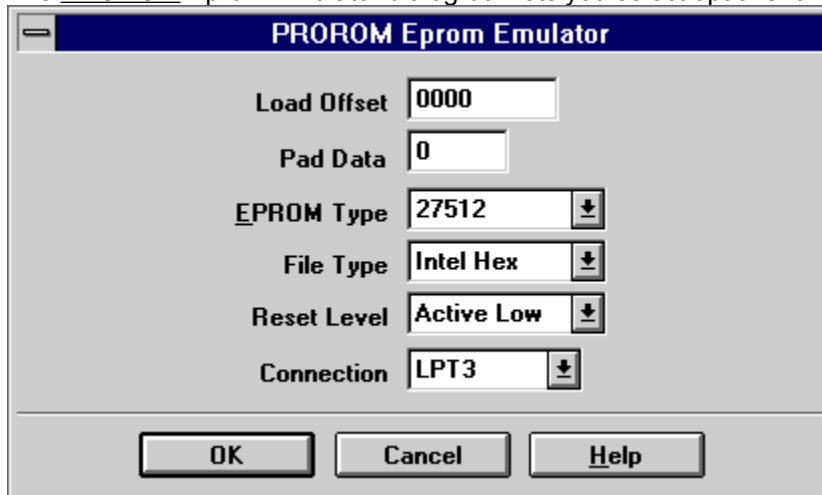
## **ProROM EPROM Emulator Command (Options Menu)**

Opens the ProROM EPROM Emulator Options dialog box where you set the options used by ProROM to download code from  $\mu$ Vision to your target board.



# ProROM Options

The PROROM Eprom Emulator dialog box lets you select options for the ProROM EPROM emulator.



The screenshot shows a dialog box titled "PROROM Eprom Emulator". It contains the following controls:

- Load Offset:
- Pad Data:
- EPROM Type:  (dropdown arrow)
- File Type:  (dropdown arrow)
- Reset Level:  (dropdown arrow)
- Connection:  (dropdown arrow)

At the bottom of the dialog are three buttons: **OK**, **Cancel**, and **Help**.

Click on a control for more information.

**OK**

Saves the changes and exits the dialog box.

**Cancel**

Abandons any changes and exits the dialog box.

**Help**

Displays context sensitive help for the dialog box.

**Load Offset**

Specifies the offset value to add to addresses in the file downloaded.

**Pad Data**

Specifies a value that is preloaded into the entire address range.

## **EPROM Type**

Specifies the type of EPROM to emulate. The following devices may be emulated: 2764 (8K), 27128 (16K), 27256 (32K), and 27512 (64K).

**File Type**

Specifies the type of file to download. Motorola S-Record files, Intel HEX files, and flat Binary files may be used.



**Reset Level**

Specifies the reset level to assert on the reset pin after a successful download.

**Connection**

Specifies the parallel port to which ProROM is connected.

# Glossary of Terms

A51

C51

dScope

BL51

OH51

PL/M-51

ProROM

## **A51**

8051 Assembler program that creates object modules from your 8051 assembly source files. Object modules may be linked into a program using the BL51 code banking linker.

## **C51**

8051 C Compiler that creates object modules from your 8051 C source files. Object modules may be linked into a program using the BL51 code banking linker.

## **dScope**

8051 debugger that you may use to simulate the microcontroller, on-chip peripherals, and external hardware in your target. You may use the debugger to help locate problems and errors in your target programs.

## **BL51**

8051 Code Banking Linker/Locator program that combines object modules created by the A51 Assembler and the C51 Compiler and creates an absolute object module that you may use with an incircuit emulator, device programmer, or simulator.

## **OH51**

Object to HEX converter program that creates Intel HEX files from absolute object files created by the BL51 code banking linker.



## **PL/M-51**

8051 PL/M Compiler that creates object modules from your 51 C source files. Object modules may be linked into a program using the L251 linker.

## **ProROM**

EPROM emulator that you may use to emulate the operation of a 2764, 27128, 27256, or 27512 compatible EPROM.

