



Keil C51

Version 9.60a

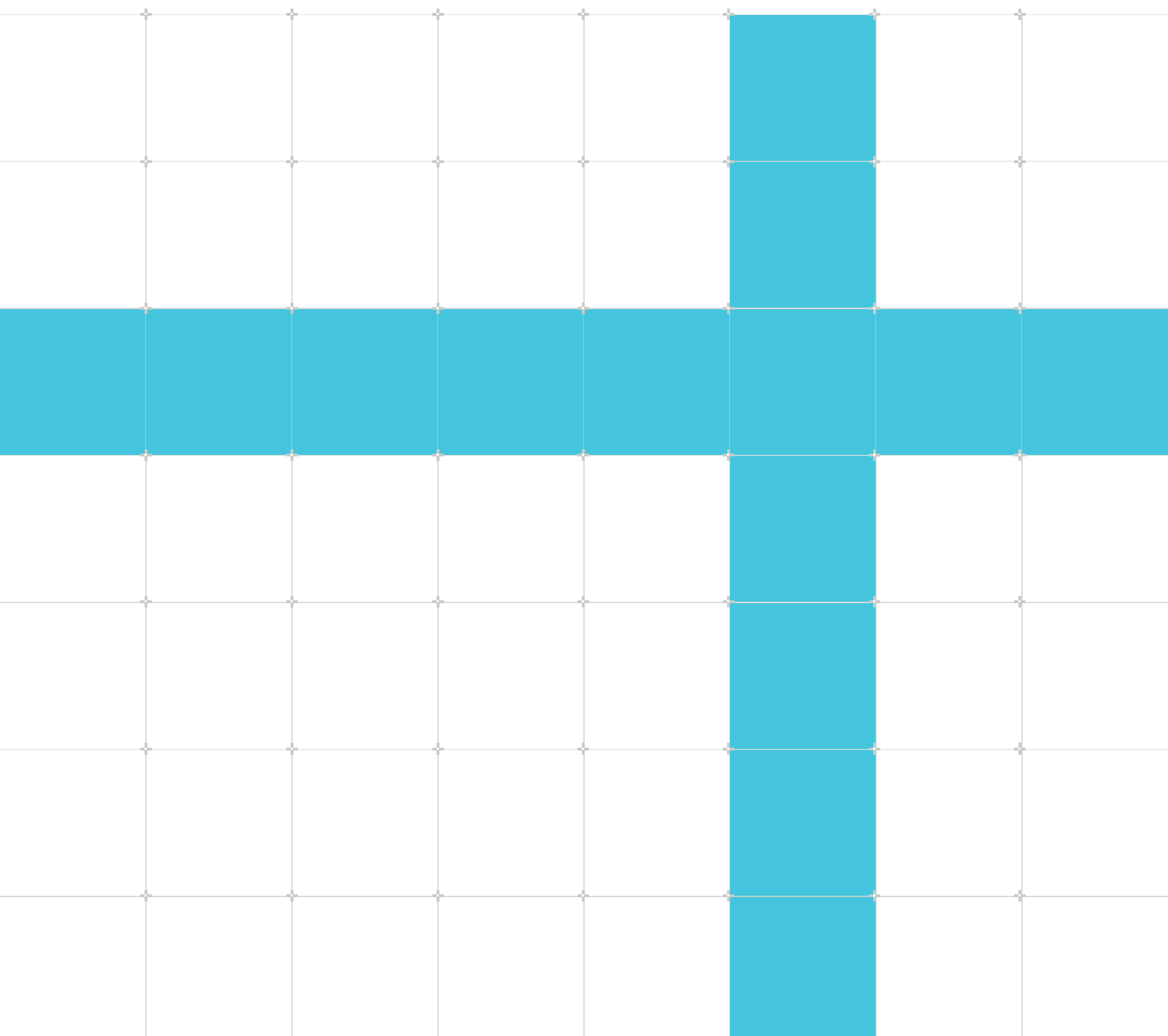
Release Notes

Non-Confidential

Copyright © 2019 Arm Limited (or its affiliates).
All rights reserved.

Issue

107780_9.60a_en



Keil C51

Release Notes

Copyright © 2019 Arm Limited (or its affiliates). All rights reserved.

Release information

Document history

Issue	Date	Confidentiality	Change
1.0	28 May 2019	Non-Confidential	Initial release

Proprietary Notice

This document is protected by copyright and other related rights and the practice or implementation of the information contained in this document may be protected by one or more patents or pending patent applications. No part of this document may be reproduced in any form by any means without the express prior written permission of Arm. No license, express or implied, by estoppel or otherwise to any intellectual property rights is granted by this document unless specifically stated.

Your access to the information in this document is conditional upon your acceptance that you will not use or permit others to use the information for the purposes of determining whether implementations infringe any third party patents.

THIS DOCUMENT IS PROVIDED "AS IS". ARM PROVIDES NO REPRESENTATIONS AND NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, SATISFACTORY QUALITY, NON-INFRINGEMENT OR FITNESS FOR A PARTICULAR PURPOSE WITH RESPECT TO THE DOCUMENT. For the avoidance of doubt, Arm makes no representation with respect to, and has undertaken no analysis to identify or understand the scope and content of, patents, copyrights, trade secrets, or other rights.

This document may include technical inaccuracies or typographical errors.

TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL ARM BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF ARM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

This document consists solely of commercial items. You shall be responsible for ensuring that any use, duplication or disclosure of this document complies fully with any relevant export laws

and regulations to assure that this document or any portion thereof is not exported, directly or indirectly, in violation of such export laws. Use of the word “partner” in reference to Arm’s customers is not intended to create or refer to any partnership relationship with any other company. Arm may make changes to this document at any time and without notice.

This document may be translated into other languages for convenience, and you agree that if there is any conflict between the English version of this document and any translation, the terms of the English version of the Agreement shall prevail.

The Arm corporate logo and words marked with ® or ™ are registered trademarks or trademarks of Arm Limited (or its affiliates) in the US and/or elsewhere. All rights reserved. Other brands and names mentioned in this document may be the trademarks of their respective owners. Please follow Arm’s trademark usage guidelines at <https://www.arm.com/company/policies/trademarks>.

Copyright © 2019 Arm Limited (or its affiliates). All rights reserved.

Arm Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

(LES-PRE-20349|version 21.0)

Confidentiality Status

This document is Non-Confidential. The right to use, copy and disclose this document may be subject to license restrictions in accordance with the terms of the agreement entered into by Arm and the party that Arm delivered this document to.

Unrestricted Access is an Arm internal classification.

Product Status

The information in this document is Final, that is for a developed product.

Feedback

Arm® welcomes feedback on this product and its documentation. To provide feedback on the product, create a ticket on <https://support.developer.arm.com>

To provide feedback on the document, fill the following survey: <https://developer.arm.com/documentation-feedback-survey>.

Inclusive language commitment

Arm values inclusive communities. Arm recognizes that we and our industry have used language that can be offensive. Arm strives to lead the industry and create change.

We believe that this document contains no offensive language. To report offensive language in this document, email terms@arm.com.

Contents

1. Introduction.....	8
1.1 Conventions.....	8
1.2 Useful resources.....	9
1.3 Other information.....	9
2. Preface.....	10
3. C51 Version 9.60a.....	11
4. C51 Version 9.60.....	12
5. C51 Version 9.59.....	14
6. C51 Version 9.57.....	18
7. C51 Version 9.56.....	20
8. C51 Version 9.55.....	24
9. C51 Version 9.54.....	29
10. C51 Version 9.53.....	31
11. C51 Version 9.52.....	34
12. C51 Version 9.51a.....	36
13. C51 Version 9.51.....	37
14. C51 Version 9.50a.....	38
15. C51 Version 9.06.....	40
16. C51 Version 9.05.....	41
17. C51 Version 9.03.....	43

18. C51 Version 9.02a.....	45
19. C51 Version 9.01.....	47
20. C51 Version 9.00.....	48
21. C51 Version 8.18.....	50
22. C51 Version 8.17a.....	52
23. C51 Version 8.16a.....	54
24. C51 Version 8.15.....	55
25. C51 Version 8.12.....	56
26. C51 Version 8.11a.....	57
27. C51 Version 8.10.....	58
28. C51 Version 8.09a.....	60
29. C51 Version 8.08a.....	61
30. C51 Version 8.06.....	62
31. C51 Version 8.05.....	63
32. C51 Version 8.04.....	65
33. C51 Version 8.02a.....	66
34. C51 Version 8.01.....	68
35. C51 Version 7.50a.....	70
36. C51 Version 7.20.....	73
37. C51 Version 7.10.....	75
38. C51 Version 7.09.....	76
39. C51 Version 7.08.....	77

40. C51 Version 7.07a.....	79
41. C51 Version 7.06a.....	81
42. C51 Version 7.05.....	82
43. C51 Version 7.04.....	83
44. C51 Version 7.03a.....	84
45. C51 Version 7.02b.....	85
46. C51 Version 7.01.....	86
47. C51 Version 7.00.....	88
48. C51 Version 6.23.....	89
49. C51 Version 6.22.....	90
50. C51 Version 6.21.....	91
51. C51 Version 6.20.....	92
52. C51 Version 6.14.....	93
53. C51 Version 6.12.....	94
54. C51 Version 6.10a.....	95
55. C51 Version 6.02.....	96
56. C51 Version 6.01.....	97
57. C51 Version 6.00.....	98

1. Introduction

1.1 Conventions

The following subsections describe conventions used in Arm documents.

Glossary

The Arm Glossary is a list of terms used in Arm documentation, together with definitions for those terms. The Arm Glossary does not contain terms that are industry standard unless the Arm meaning differs from the generally accepted meaning.

See the Arm® Glossary for more information: developer.arm.com/glossary.

Typographic conventions

Arm documentation uses typographical conventions to convey specific meaning.

Convention	Use
<i>italic</i>	Citations.
bold	Interface elements, such as menu names. Terms in descriptive lists, where appropriate.
monospace	Text that you can enter at the keyboard, such as commands, file and program names, and source code.
monospace <u>underline</u>	A permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.
<and>	Encloses replaceable terms for assembler syntax where they appear in code or code fragments. For example: <pre>MRC p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2></pre>
SMALL CAPITALS	Terms that have specific technical meanings as defined in the <i>Arm® Glossary</i> . For example, IMPLEMENTATION DEFINED , IMPLEMENTATION SPECIFIC , UNKNOWN , and UNPREDICTABLE .
 Caution	Recommendations. Not following these recommendations might lead to system failure or damage.
 Warning	Requirements for the system. Not following these requirements might result in system failure or damage.
 Danger	Requirements for the system. Not following these requirements will result in system failure or damage.
 Note	An important piece of information that needs your attention.

Convention	Use
 Tip	A useful tip that might make it easier, better or faster to perform a task.
 Remember	A reminder of something important that relates to the information you are reading.

1.2 Useful resources

This document contains information that is specific to this product. See the following resources for other useful information.

Access to Arm documents depends on their confidentiality:

- Non-Confidential documents are available at developer.arm.com/documentation. Each document link in the following tables goes to the online version of the document.
- Confidential documents are available to licensees only through the product package.



Arm tests its PDFs only in Adobe Acrobat and Acrobat Reader. Arm cannot guarantee the quality of its documents when used with any other PDF reader.

Adobe PDF reader products can be downloaded at <http://www.adobe.com>

1.3 Other information

See the Arm website for other relevant information.

- [Arm® Developer](#).
- [Arm® Documentation](#).
- [Technical Support](#).
- [Arm® Glossary](#).

2. Preface

The industry-standard Keil C compilers, macro assemblers, debuggers, and real-time kernels support all 8051 derivatives and help you get your projects completed on schedule.

Supported Operating Systems

Refer to [System Requirements](#) for hardware and operating system requirements.

Example Programs

Example programs included in the \c51\EXAMPLES folder demonstrate how to use the μ Vision project manager and debugger.

Device Database

A unique feature of the Keil μ Vision IDE is the Device Database which contains information about more than 3500 supported microcontrollers. When you create a new μ Vision project and select the target chip from the database, μ Vision sets all assembler, compiler, linker, and debugger options for you. The only option you must configure is the memory map.

As new devices become available, they are added to the database along with data sheets and header files. For information about adding your own chips to the database or about creating your own personal databases refer to the following knowledge base articles.

- [Adding Custom Parts to the Device Database](#)
- [Creating Custom Databases](#)

Peripheral Simulation

The μ Vision debugger provides complete simulation for the CPU and on-chip peripherals of most embedded devices. To discover which peripherals of a device are supported, in μ Vision select the simulated peripherals item from the help menu. You may also use the web-based [Device Database](#).

Technical Support

Open a [support case](#) for technical problems or inquiries.

You can also search the [documentation](#) for application notes, knowledge base articles, user guides, and product information.

In the [Keil Forum](#) you may post questions and comments about Keil products.

Contact Details

You may contact us directly at one of the offices listed on the [Keil Support](#) page. You may also receive sales and support through your [local distributor](#).

3. C51 Version 9.60a

Release Date: 28th May, 2019

Target Debugging/Device Programmer

Updated: Segger JLink IS2083 debug driver to version 6.44.4.


```

0007 4C          ORL      A,R4
0008 70F6        JNZ      ?C0001
...

```

New Supported Devices

ABOV Semiconductor MC94F1202A, MC97FG216, MC96F8104, MC96F8216S, MC96F7416S, MC96F6632S, MC96F6432Q, MC96F6432S, MC96F6332S, MC96F6409, A94B114, A97C450, A96R717, MC97F2664A, MC96F8316A, MC96P6608, MC96F8316S, MC96F8208S, MC96F6509, MC96F6432A, and MC94F1102A.

ISSI IS31CS8973, - Megawin MG86Fx508, MG86Fx104, MG84FG516, MG84FL54BD, MG82G5E32, MG82FG5D16, MG82FG5C64, MG82FG5C32, MG82FG5B32, MG82FG5B16, MG82FG5A64, MG82FG5A32, MG82Fx316, MG82Fx308, MG82Fx564, MG82Fx532, MG87Fx04, MG87Fx6051, MG87Fx4051, MG87Fx2051, and M87Fx52.

Microchip IS2083B.

Device Simulation

Corrected: for Atmel AT89C51CC03 the simulation crashes during the startup phase when the project was initialized with the -pATXC3 setting.

Target Debugging/Device Programmer

Added: Segger JLink IS2083 debug driver to support the [IS2083B](#) from Microchip.

Updated: NULink driver to version 2.06.6875.

uVision

This C51 release comes with μ Vision V5.27.1.

The [Customize Tools Menu...](#) option is extended with Export/Import for sharing the tools menu setup across PCs.

5. C51 Version 9.59

Release Date: 9th May, 2018

New Supported Devices

Silabs EFM8UB30F40G_QFN20, EFM8UB31F40G_QFN24, EFM8UB31F40G_QSOP24

Weltrend WT51F116, WT51F108, and WT51F104.

AX51 Macro Assembler

Corrected: allow slash / in addition to backslash \ as a directory separator in all places.

Cx51 Compiler

Corrected: sometimes the compiler shows warning C290: missing return value or warning C291: not every path returns a value although a value is returned.

Corrected: in some cases post increment was not applied. Example:

```
unsigned char reader;
unsigned char count;
unsigned int index;

index = 0;

write_byte(offset);

for (count = 0; count < 10; count++) {
    reader = read_byte();
    pData[index++] = reader;
}
...
```

Wrong code was generated:

```
...
000D 120000      R      LCALL    read_byte
;---- Variable 'reader' assigned to Register 'R5' ----
0010 AD07              MOV      R5,AR7
0012 EB              MOV      A,R3
0013 2500      R      ADD      A,index+01H
0015 F582              MOV      DPL,A
0017 EA              MOV      A,R2
0018 3500      R      ADDC     A,index
001A F583              MOV      DPH,A
001C ED              MOV      A,R5
001D F0              MOVX     @DPTR,A
001E 0C              INC      R4
...
```

Now, the correct code is generated:

```
...
;---- Variable 'reader' assigned to Register 'R5' ----
001A AD07              MOV      R5,AR7
```

```

001C 8B82      MOV     DPL,R3
001E 8A83      MOV     DPH,R2
0020 0500      R       INC     index+01H
0022 E500      R       MOV     A,index+01H
0024 AE00      R       MOV     R6,index
0026 7002      JNZ     ?C0007
0028 0500      R       INC     index
002A          ?C0007:
002A 14        DEC     A
002B 2582      ADD     A,DPL
002D F582      MOV     DPL,A
002F E583      MOV     A,DPH
0031 3E        ADDC    A,R6
0032 F583      MOV     DPH,A
0034 ED        MOV     A,R5
0035 F0        MOVX    @DPTR,A
0036 0C        INC     R4
...

```

Corrected: allow slash / in addition to backslash \ as a directory separator in all places.

Corrected: in rare cases a signed compare with unsigned char treated the unsigned char as signed. This happens if the unsigned char value results from a calculated assignment. Example:

```

char i;
unsigned char uc0 = 254;
char c1 = 0;

void main() {
    if (c1 <= (uc0 += c1))    // uc0 is wrongly treated signed => -2 therefore the
        condition (0 <= -2) is false
        i = 1;
    else
        i = 0;
}

```

Wrong code was generated:

```

0000 E500      R       MOV     A,uc0
0002 2500      R       ADD     A,c1
0004 F500      R       MOV     uc0,A
0006 D3        SETB    C
0007 6480      XRL     A,#080H      \
0009 F8        MOV     R0,A        |
000A E500      R       MOV     A,c1        |
000C 6480      XRL     A,#080H      |
000E 98        SUBB    A,R0        |
000F 5004      JNC     ?C0001      -> signed char compare
0011 750001    R       MOV     i,#01H
0014 22        RET
0015          ?C0001:
0015 E4        CLR     A
0016 F500      R       MOV     i,A
0018          ?C0003:
0018 22        RET

```

The correct code is much longer because according to the C standard a cast to int is necessary:

```

0000 E500      R       MOV     A,uc0
0002 2500      R       ADD     A,c1
0004 FF        MOV     R7,A
0005 F500      R       MOV     uc0,A

```

```

0007 AD00      R      MOV      R5,c1      \
0009 ED        MOV      A,R5             |
000A 33        RLC      A                 |
000B 95E0      SUBB     A,ACC             |
000D FC        MOV      R4,A             -> cast of signed char to int
000E D3        SETB     C                 \
000F ED        MOV      A,R5             |
0010 9F        SUBB     A,R7             |
0011 7480      MOV      A,#080H          +> implicit cast of unsigned char to
int
0013 F8        MOV      R0,A             |
0014 6C        XRL      A,R4             |
0015 98        SUBB     A,R0             |
0016 5004      JNC      ?C0001           -> signed int compare
0018 750001     R      MOV      i,#01H
001B 22        RET
001C          ?C0001:
001C E4        CLR      A
001D F500      R      MOV      i,A
001F          ?C0003:
001F 22        RET

```

To generate smaller code the signedness of the values to be compared should be the same:

```

...
if ((unsigned char)c1 <= (uc0 += c1))
...
0000 E500      R      MOV      A,uc0
0002 2500      R      ADD      A,c1
0004 FF        MOV      R7,A
0005 F500      R      MOV      uc0,A
0007 E500      R      MOV      A,c1
0009 D3        SETB     C                 \
000A 9F        SUBB     A,R7             |
000B 5004      JNC      ?C0001           -> unsigned char compare
000D 750001     R      MOV      i,#01H
0010 22        RET
0011          ?C0001:
0011 E4        CLR      A
0012 F500      R      MOV      i,A
0014          ?C0003:
0014 22        RET

```

LX51 Linker/Locater

Corrected: in rare cases the linker crashes if using long function names and [Global Register Coloring](#)

Corrected: somehow not only function names were listed in the [call tree](#) and warning L48: IGNORED RECURSION, CALL REMOVED appeared

Corrected: sometimes automatic rebuilds are not executed when [Global Register Coloring](#) is used.

Corrected: in case of banked applications and under some circumstances the CONST- as well as the CODE-Segments located to the same bank address.

Corrected: under some circumstances the [MERGEPUBLICS](#) does not work for sbit variables.

Debug Commands

Added: New debugger command **COVTOFILE filespec [\app] [DETAILS] [ASM]** which works like the **COVERAGE** command but sends the output into a file. This is much faster than into the output window.

uVision

This C51 release comes with μ Vision V5.25.3.

6. C51 Version 9.57

Release Date: 9th November, 2017

A51 Assembler and AX51 Assembler

Corrected: two line feeds in [Macro Processor Language \(MPL\)](#) error message.

LX51 Linker/Locater

Corrected: in banking mode 8 there were sometimes misplaced segments and warning L30: MEMORY SPACE OVERLAP appeared.

Added: the warning L59: REENTRANT CALLS NON REENTRANT

FUNCTION, COULD LEAD TO WRONG OVERLAY CALCULATION will be generated when a reentrant function calls a non-reentrant one.

Corrected: automatic rebuilds are not executed when 'Global Register Coloring' is used. This problem was introduced with C51 version 9.56.

Corrected: in case of banked applications and under some circumstances the CONST- as well as the CODE-Segments located to the same bank address. This problem was introduced with C51 version 9.56.

Corrected: under some circumstances the [MERGEPUBLICS](#) does not work for sbit variables.

Corrected: the LX51 may locate the Stack-Segment to wrong address. As described in the [knowledge base article 3842](#) this problem was introduced with C51 version 9.55.

Corrected: under some circumstances the LX51 erroneously throws the warning L48: IGNORED RECURSIVE CALL. This regression was introduced with C51 version 9.56.

BL51 Linker/Locater

Corrected: erroneously the BL51 throws the ERROR L121:

```
IMPROPER FIXUP for bit variables located in the bdata memory space.
```

OHX51 Object to Hex Converter

Corrected: under some circumstances the OHX51 terminates the OMF51 to Intel HEX conversion by throwing the following message:

```
ERROR: BANKED APPLICATION CANNOT HAVE AN OFFSET VALUE.
```

New Supported Devices

[Nuvoton N76E885](#), [N76E616](#), and [N76E003](#).

uVision

This C51 release comes with μ Vision V5.24.2.86.

Enhanced: New PC-Lint configuration option to add project target and compiler specific preprocessor symbols.

μ Vision now offers Japanese localization on Windows PCs with the 'primary language' Japanese. To select the language use the uVision menu item [Edit - Configuration - Other - Startup - Language](#).

A [Japanese Getting Started user's guide](#) is available in the uVision Books Window.

Added: new option to limit the [Find in Files](#) utility to the "Current Document".

Enhanced: editor now supports Arabic, Baltic, Eastern European, Greek, Hebrew, Russian, Thai, Turkish, and Vietnamese character sets.

Corrected: [Bookmark](#) navigation is now working only on the "Current Document".

Corrected: opening struct elements in the [Watch Window](#) did not always show up-to-date values.

7. C51 Version 9.56

Release Date: 10th August, 2016

Cx51 Compiler

Improved C90 conformity for ASSERT.H, FLOAT.H, MATH.H, STDARG.H, STDDEF.H, STDLIB.H, and STRING.H header files.

Corrected: redundant code generation. Under some circumstances the C51 compiler generates code which loads the accumulator register A twice. Example:

```

.....
10:
11:     union X {
12:         struct {
13:             unsigned char l8 :8;
14:         } v;
15:     };
16:
17:     union X xdata x _at_ 0x5800;
18:     unsigned char t;
19:
20:     void MyFunc(void) {
21:         x.v.l8 = t;
22:     }
23:
.....

          21:         x.v.l8 = t;
code generation C:0x088C 90581D MOV DPTR,#0x581D -----+-----o Suboptimal
register A load C:0x088F E0 MOVX A,@DPTR | <---- Superfluous
C:0x0890 ED MOV A,R5 |
C:0x0891 F0 MOVX @DPTR,A -----+

          21:         x.v.l8 = t;
code generation C:0x13BA 90581D MOV DPTR,#0x581D -----+-----o Corrected
C:0x13BD ED MOV A,R5 |
C:0x13BE F0 MOVX @DPTR,A -----+

```

Corrected: wrong code generation in case of bit-field computation inside a switch-case statement.
Example:

```

#include <REG51F.H>

typedef struct sTest {
    unsigned char b1 : 1;
    unsigned char b2 : 1;
} t_test;

xdata t_test tSt;

void main (void) {
    switch (tSt.b2) {

```

```

case 1:
    P1 = 1;
    break;
default:
    P1 = 0;
    break;
}

+-----+
Wrong code generation
+-----+
    ; FUNCTION main (BEGIN)
                                ; SOURCE LINE # 10
                                ; SOURCE LINE # 12
0000 900000      R      MOV     DPTR,#tSt
0003 E0          MOVX     A,@DPTR

0004 5402          ANL      A,#02H      -----\
0006 14          DEC      A              +---- The compiler
erroneously generates code which evaluates
0007 7004          JNZ      ?C0003      -----/      the bit position 0
instead the bit position 1.

+-----+
Corrected code generation
+-----+
    ; FUNCTION main (BEGIN)
                                ; SOURCE LINE # 10
                                ; SOURCE LINE # 12
0000 900000      R      MOV     DPTR,#tSt
0003 E0          MOVX     A,@DPTR

0004 C3          CLR      C      -----\
0005 13          RRC      A              \
0006 5401          ANL      A,#01H      +---- The generated
code evaluates now the bit position 1.
0008 14          DEC      A              /
0009 7004          JNZ      ?C0003      -----/

```

LX51 Linker/Locater

Corrected: Under some circumstances the LX51 locates the stack segment to a wrong address inside the idata address space. This problem was introduced with C51 version 9.55. Example:

```

+-----+
Wrong stack location inside the data/idata memory space
+-----+

MEMORY MAP OF MODULE:  Repro (?C_STARTUP)

START      STOP      LENGTH      ALIGN      RELOC      MEMORY CLASS      SEGMENT NAME
=====
* * * * * D A T A   M E M O R Y * * * * *
000000H    000007H    000008H    ---      AT..      DATA      "REG BANK 0"
000008H    000015H    00000EH    BYTE     UNIT      DATA      ?DT?MAIN
000016H    000017H    000002H    BYTE     UNIT      DATA      ?DT?USB
000018H    000018H    000001H    BYTE     UNIT      DATA      ?DT?WATCHDOG
000019H    000019H    000001H    BYTE     UNIT      DATA      ?C?LIB_DATA
00001AH    00001AH    000001H    BYTE     UNIT      IDATA      ?STACK      <-----
+--- Wrong stack location
00001BH.0  00001FH.7  000005H.0  ---      ---      **GAP**
|
000020H.0  000020H.3  000000H.4  BIT      UNIT      BIT      _BIT_GROUP_
|

```

```

000020H.4 000020H.6 000000H.3 BIT UNIT BIT ?BI?MAIN
|
000020H.7 000020H 000000H.1 --- --- **GAP**
|
000021H 00004FH 00002FH BYTE UNIT DATA _DATA_GROUP_
|
000050H 000077H 000028H BYTE UNIT DATA ?DT?GLOBALS
|
|
|
|
|
+-----+
|
| Corrected stack location inside the data/idata memory space
|
+-----+
|
|
|
MEMORY MAP OF MODULE:  Repro (?C_STARTUP)
|
|
| START STOP LENGTH ALIGN RELOC MEMORY CLASS SEGMENT NAME
|=====|
|
|
| * * * * * D A T A M E M O R Y * * * * *
|
| 000000H 000007H 000008H --- AT.. DATA "REG BANK 0"
|
| 000008H 000015H 00000EH BYTE UNIT DATA ?DT?MAIN
|
| 000016H 000017H 000002H BYTE UNIT DATA ?DT?USB
|
| 000018H 000018H 000001H BYTE UNIT DATA ?DT?WATCHDOG
|
| 000019H 000019H 000001H BYTE UNIT DATA ?C?LIB_DATA
|
| 00001AH.0 00001FH.7 000006H.0 --- --- **GAP**
|
| 000020H.0 000020H.3 000000H.4 BIT UNIT BIT _BIT_GROUP_
|
| 000020H.4 000020H.6 000000H.3 BIT UNIT BIT ?BI?MAIN
|
| 000020H.7 000020H 000000H.1 --- --- **GAP**
|
| 000021H 000050H 000030H BYTE UNIT DATA _DATA_GROUP_
|
| 000051H 000078H 000028H BYTE UNIT DATA ?DT?GLOBALS
|
| 000079H 000079H 000001H BYTE UNIT IDATA ?STACK <-----
+--- Correct stack location

```

New: the warning L48: Ignored Recursive Call Callee: function-name Caller: *function-name* will be issued when a reentrant function calls a non-reentrant function.

A51 Assembler and AX51 Assembler

Removed: a path length limitation of 127 characters.

New Supported Devices

ABOV MC95FG0128, MC95FG6128, MC95FG8128, MC95FG208, MC95FG308, MC95FR332, MC95FR364, MC95FR432, MC95FR464, MC96F1206, MC96F4548, MC96F6332, MC96F6432, MC96F6408A, MC96F6508A, MC96F6632, MC96F6832, MC96F7064, MC96F7416A, MC96F7616A, MC96F7616T, MC96F7816, MC96F7664, MC96F7864, MC96F7848C, MC96F8204, MC96F8208, MC96F8216, MC96F8316, MC96FC664A, MC96FC864A, MC96FD316, MC96FM204, MC96FM214, MC96FM408, MC96FR116C, MC96FR3128, MC96FR4128, MC96FR332A, MC96FR364B, MC96FR364C, MC96FT1616, MC96FT1704, MC96FT241, MC96FT242, MC97F2464, MC97F2664, MC97F60128, MC97F66128, MC97F68128, MC97F68128A, and MC97FG316.

uVision

This C51 release comes with μ Vision V5.20.0.39.

Supported Operating Systems

μ Vision and its dynamically loaded libraries (DLL) have been ported to MSVC 2015. MSVC 2015 does not officially support Windows XP any longer.

8. C51 Version 9.55

Release Date: 14th March, 2016

Cx51 Compiler

Corrected: [WARNING C182: pointer to different objects](#) sometimes incorrect. Example:

```
int xxx[6];
int (*ptr)[6];

void main(void) {
    ptr = &xxx; /* The compiler erroneously throws the WARNING C182: pointer to
different objects for this assignment. */
}
```

Corrected: constant strings may be stored in the [memory class](#) CODE instead of CONST.. For example:

```
code char str1[] = "aaaaa";
code char str2[] = "bbbbb";
char * arC;
```

```
void main(void) {
    arC = str1;
    arC = "xxxxx";
}
```

Wrong storage location for "aaaaa", "bbbbb", and "xxxxx".
Correct storage location for "aaaaa", "bbbbb", and "xxxxx".

?PR?main?MAIN	SEGMENT CODE		?
PR?main?MAIN	SEGMENT CODE		
?CO?MAIN	SEGMENT CODE	-----+	?
CO?MAIN	SEGMENT CONST	-----+	
?DT?MAIN	SEGMENT DATA		?
DT?MAIN	SEGMENT DATA		
EXTRN CODE (?C_STARTUP)			
EXTRN CODE (?C_STARTUP)			
PUBLIC arC			
PUBLIC arC			
PUBLIC str2			
PUBLIC str2			
PUBLIC str1			
PUBLIC str1			
PUBLIC main			
PUBLIC main			
RSEG ?DT?MAIN			
RSEG ?DT?MAIN			
arC: DS 3			
arC: DS 3			
RSEG ?CO?MAIN	<-----+ Program Code		
RSEG ?CO?MAIN	<-----+ Constant		
?SC_0:			?
SC_0:			
DB 'x','x','x','x','x',000H			
DB 'x','x','x','x','x',000H			
str1:			
str1:			
DB 'a','a','a','a','a',000H			
DB 'a','a','a','a','a',000H			

```

str2:
str2:
    DB  'b' , 'b' , 'b' , 'b' , 'b' , 000H
    DB  'b' , 'b' , 'b' , 'b' , 'b' , 000H

```

LX51 Linker/Locater

Added: in the LX51 map file a new section LINKER CODE PACKING CROSS-REFERENCE shows the usage of common code blocks from the various program segments. For example:

```

.....
21:  if (*line == '+' || *line == '-')  sign = (*line++ == '+');
002593 EF          MOV      A,R7
002594 642B        XRL      A,#02BH
002596 6005        JZ       ?C0009?CSAMPLE2
002598 D102        ACALL    ?L?COM0001 <-----+
00259A B42D0D      CJNE     A,#02DH,?C0008?CSAMPLE2
00259D           ?C0009?CSAMPLE2:
00259D D131        ACALL    ?L?COM0002 <-----+
00259F 7165        ACALL    ?C?CLDPTR
0025A1 B42B03      CJNE     A,#02BH,?C0010?CSAMPLE2
.....

----- FUNCTION ?L?COM0001 (BEGIN) ----- <-----+
002602 90102C      MOV      DPTR,#line
002605 E0          MOVX     A,@DPTR
002606 FB          MOV      R3,A
002607 A3          INC      DPTR
002608 E0          MOVX     A,@DPTR
002609 FA          MOV      R2,A
00260A A3          INC      DPTR
00260B E0          MOVX     A,@DPTR
00260C F9          MOV      R1,A
00260D 6165        AJMP     ?C?CLDPTR
----- FUNCTION ?L?COM0001 (END) -----

----- FUNCTION ?L?COM0002 (BEGIN) ----- <-----+
002631 90102C      MOV      DPTR,#line
002634 E0          MOVX     A,@DPTR
002635 FB          MOV      R3,A
002636 A3          INC      DPTR
002637 E4          CLR      A
002638 75F001      MOV      B,#01H
00263B 71B8        ACALL    ?C?ILDIX
00263D A9F0        MOV      R1,B
00263F FA          MOV      R2,A
002640 22          RET
----- FUNCTION ?L?COM0002 (END) -----

LINKER CODE PACKING CROSS-REFERENCE LISTING

NAME . . . . CLASS SIZE  TYPE   SEGMENT NAMES (ADDR)
=====
?L?COM0001 . CODE  000DH PART   ?L?COM0001  ?PR?_ATOI?CSAMPLE2 (10025B1H)  ?PR?_ATOI?
CSAMPLE2 (1002598H)
|
?L?COM0002 . CODE  0010H PART   ?L?COM0002  ?PR?_ATOI?CSAMPLE2 (100259DH)  ?PR?
_GETLINE?CSAMPLE2 (100261CH)
...
...

```

Corrected: When using the SEGMENTS directive to order segments, the order was violated (incorrect size optimization). For example:

Assembler source:

```
C2 SEGMENT ECODE
  RSEG C2
  T2: DB  0x7F

C1 SEGMENT ECODE SEG
  RSEG C1
  T1: DB  0x7F

C3 SEGMENT ECODE SEG
  RSEG C3
  T3: DB  0x7F
```

defined segment order

```
|-----|
LX51.EXE T1.obj TO T1.abs PRINT CLASSES (CODE (C:0X4000-C:0X8000), ECODE (C:0X4000-
C:0X28000), XDATA (X:0X0000-X:0X09FF)) SEGMENTS (C3, C1, C2)
```

START	STOP	LENGTH	ALIGN	RELOC	MEMORY CLASS	SEGMENT NAME	
010000H	010007H	000008H	SEG	UNIT	ECODE	C3	-----+---
Wrong segment order.							
010008H	010008H	000001H	BYTE	UNIT	ECODE	C2	
010009H	01FFFFH	00FFF7H	---	---	**GAP**		
020000H	020000H	000001H	SEG	UNIT	ECODE	C1	-----+

START	STOP	LENGTH	ALIGN	RELOC	MEMORY CLASS	SEGMENT NAME	
010000H	010007H	000008H	SEG	UNIT	ECODE	C3	-----+---
Correct segment order.							
010008H	01FFFFH	00FFF8H	---	---	**GAP**		
020000H	020000H	000001H	SEG	UNIT	ECODE	C1	
020001H	020001H	000001H	BYTE	UNIT	ECODE	C2	-----+

Corrected: Using the LAST in the SEGMENTS directive was ignored for linker code packing segments. For example:

```
LX51 TC 0.obj, TC 1.obj TO T PRINT CLASSES (EDATA (0X0000-0X03FF), XDATA (X:0X0000-
X:0X113F)) SEGMENTS (?PR?TEST2?TC_1, ?PR?TEST?TC_1 (LAST))
```

wrong:

START	STOP	LENGTH	ALIGN	RELOC	MEMORY CLASS	SEGMENT NAME
.....						
000087H	00008EH	000008H	BYTE	UNIT	CODE	?L?COM0007
00008FH	000096H	000008H	BYTE	UNIT	CODE	?PR?MAIN?TC_0

000097H	---	000000H	BYTE	UNIT	CODE		?PR?TEST2?TC_1
000097H	0000AFH	000019H	BYTE	UNIT	CODE		?PR?TEST?TC_1
<-----+-----							
0000B0H	00011CH	00006DH	BYTE	UNIT	CODE		?PR?_TESTDUMMYCODE?TC_1
00011DH	00012CH	000010H	BYTE	UNIT	CODE		?L?COM0002
00012DH	000147H	00001BH	BYTE	UNIT	CODE		?L?COM0001
000148H	000149H	000002H	BYTE	UNIT	CODE		?PR?FAIL?TC_1
correct:							
START	STOP	LENGTH	ALIGN	RELOC	MEMORY	CLASS	SEGMENT NAME
=====							
....							
000081H	000088H	000008H	BYTE	UNIT	CODE		?L?COM0006
000089H	000090H	000008H	BYTE	UNIT	CODE		?L?COM0007
000091H	000098H	000008H	BYTE	UNIT	CODE		?PR?MAIN?TC_0
000099H	00009AH	000002H	BYTE	UNIT	CODE		?PR?FAIL?TC_1
00009BH	000107H	00006DH	BYTE	UNIT	CODE		?PR?_TESTDUMMYCODE?TC_1
000108H	000117H	000010H	BYTE	UNIT	CODE		?L?COM0002
000118H	000132H	00001BH	BYTE	UNIT	CODE		?L?COM0001
000133H	00014BH	000019H	BYTE	UNIT	CODE		?PR?TEST?TC_1
<-----+-----							

Corrected: CONST When using the SEGMENTS directive to specify an address, the segment address was moved during linker code packing. For example:

C-Code snippet:

```
#pragma userclass(code = XXX)
const char code xxx[30] _at_ 0x1234;
```

```
LX51 CSAMPLE2.obj TO CSample CLASSES (CODE(C:0X2000-C:0X2FFF), CONST(C:0X2000-C:0X2FFF), ECODE(C:0X2000-C:0X2FFF), HCONST(C:0X2000-C:0X2FFF))
```

wrong:

START	STOP	LENGTH	ALIGN	RELOC	MEMORY	CLASS	SEGMENT NAME
=====							
000000H	000002H	000003H	---	OFFS..	CODE		?CO??C_STARTUP?0
000003H	000019H	000017H	BYTE	UNIT	CODE_XXX		?PR?GETNUMBER?CSAMPLE2
00001AH	0000A3H	00008AH	BYTE	UNIT	CODE_XXX		?PR?_atoi?CSAMPLE2
0000A4H	0000B4H	000011H	BYTE	UNIT	CODE_XXX		?PR?_getline?CSAMPLE2
0000B5H	0000B6H	000002H	---	---	**GAP**		
0000B7H	0000D4H	00001EH	BYTE	OFFS..	CODE_XXX		?CO?CSAMPLE2?0
0000D5H	001FFFH	001F2BH	---	---	**GAP**		
002000H	002364H	000365H	BYTE	UNIT	CODE		?PR?PRINTF?PRINTF

correct:

START	STOP	LENGTH	ALIGN	RELOC	MEMORY	CLASS	SEGMENT NAME
=====							
000000H	000002H	000003H	---	OFFS..	CODE		?CO??C_STARTUP?0
000003H	000019H	000017H	BYTE	UNIT	CODE_XXX		?PR?GETNUMBER?CSAMPLE2
00001AH	0000A3H	00008AH	BYTE	UNIT	CODE_XXX		?PR?_atoi?CSAMPLE2
0000A4H	0000B4H	000011H	BYTE	UNIT	CODE_XXX		?PR?_getline?CSAMPLE2

```

0000B5H  001233H  00117FH  ---  ---  **GAP**
001234H  001251H  00001EH  BYTE  OFFS..  CODE XXX  ?CO?CSAMPLE2?0
001252H  001FFFH  000DAEH  ---  ---  **GAP**
002000H  002364H  000365H  BYTE  UNIT  CODE  ?PR?PRINTF?PRINTF

```

Corrected: Do not show empty segments as overlapping. For example:

```

START      STOP      LENGTH    ALIGN  RELOC    MEMORY CLASS  SEGMENT NAME
=====
010000H    010007H    000008H    SEG    UNIT    ECODE         C3
010008H    010008H    000001H    BYTE   UNIT    ECODE         C2
*** OVERLAP ***                                     <--- There is no
real overlap because the next segment is empty
010008H    ---          000000H    BYTE   UNIT    ECODE         C1

```

uVision

This C51 release comes with μ Vision V5.14.2.1.

9. C51 Version 9.54

Release Date: 24th April, 2015

LX51 Linker/Locater

Corrected: under some circumstances the LX51 ignores the [NOJMPTAB](#) directive and and throws erroneously the following error:

```
* ERROR L210: I/O ERROR ON INPUT FILE:
EXCEPTION 0021H: PATH OR FILE NOT FOUND
FILE: C:\KEIL\C51\LIB\L51_BANK.OBJ
```

A51 Macro Assembler and AX51 Macro Assembler

Corrected: under some circumstances the assembler shows wrong line numbers if an \$include file cannot be found. The root cause for this behavior is the erroneous line number generation by the preprocessor. Example:

```
NAME xxx
S      SEGMENT CODE
      RSEG S
#line          ; The preprocessor generates wrong line numbers if an $include
file cannot be found.
PUT:  MOV  A,#x  ; The assembler shows these wrong line numbers inside the
build output window as part of a warning or error message.
      RET
      END
```

OHX51 Object to Hex Converter

The OHX51 creates [hex records](#) that crosses 64Kbyte boundaries which is not allowed. For example: defining a HCONST segment ?HC?UCL with a size of 128Kbyte will cross two times the 64Kbyte boundaries at 0x810000 and 0x820000. Mapfile entry for this segment.

```
.....
800020H  82000FH  01FFF0H  BYTE  UNIT HCONST ?HC?UCL
.....
```

The resulting hex file is wrong at these 64 KBytes boundaries. One HEX record (equates to one line) must not cross a 64 KBytes boundary:

```
Line  HEX record

.....
00021 :0200000400807A ; switch to segment 0x80
.....
04119 :10FFFC00215E5F5E48656C6C6F204169726F686151 ; starts at 0xFFFFC and crosses
the 64K boundary
      |||||
      ||AAAA -> Address
      LL-----> Record length

04120 :02000004008179 ; switch to segment 0x81
04121 :10000C00215E5F5E48656C6C6F204169726F686140 ; starts at 0x000C
```

```

.....
.....
.....
08218  :10FFF800726F6861215E5F5E48656C6C6F20416955 ; starts at 0xFFF8 and crosses
the 64K boundary
      |||||
      ||AAAA -> Address
      LL-----> Record length

08219  :02000004008278                                ; switch to segment 0x82
08220  :08000800726F6861215E5F5E0A                    ; starts at 0x0008
.....

```

Device Simulation

Corrected: for Silabs C8051F33x based devices the simulation of the watchdog timer triggers unexpectedly a reset.

New Supported Devices

[CAST T8051XC3](#), [L8051XC1-515](#), [L8051XC1-320](#), [S8051XC3-C\(517\)](#), and [S8051XC3-C\(430\)](#).

[Silabs EFM8BB10F2G_A_QFN20](#), [EFM8BB10F4G_A_QFN20](#), [EFM8BB10F8G_A_QFN20](#), [EFM8BB10F8G_A_QSOP24](#), [EFM8BB10F8G_A_SOIC16](#), [EFM8BB21F16G_A_QFN20](#), [EFM8BB21F16G_A_QSOP24](#), [EFM8BB22F16G_A_QFN28](#), [EFM8SB10F2G_A_QFN20](#), [EFM8SB10F4G_A_QFN20](#), [EFM8SB10F8G_A_QFN20](#), [EFM8SB10F8G_A_QFN24](#), [EFM8SB10F8G_A_QSOP24](#), [EFM8SB20F16G_A_QFN24](#), [EFM8SB20F32G_A_LQFP32](#), [EFM8SB20F32G_A_QFN24](#), [EFM8SB20F32G_A_QFN32](#), [EFM8SB20F64G_A_LQFP32](#), [EFM8SB20F64G_A_QFN24](#), [EFM8SB20F64G_A_QFN32](#), [EFM8UB10F16G_A_QFN20](#), [EFM8UB10F16G_A_QFN28](#), [EFM8UB10F8G_A_QFN20](#), [EFM8UB11F16G_A_QSOP24](#), [EFM8UB20F32G_A_LQFP32](#), [EFM8UB20F32G_A_QFN32](#), [EFM8UB20F32G_A_TQFP48](#), [EFM8UB20F64G_A_LQFP32](#), [EFM8UB20F64G_A_QFN32](#), and [EFM8UB20F64G_A_TQFP48](#).

uVision

This C51 release comes with μ Vision V5.14.1.

10. C51 Version 9.53

Release Date: 5th August, 2014

LX51 Linker/Locator

Implemented: the **PUBLICONLY** Linker directive. The generated objectfile contains only public symbol information from the input file.

Enhanced: LX51 code optimization now removes common blocks for unused functions. Example:

```

unsigned char a, b, c;
unsigned char darr [0x10];

void FuncA (void) {
    darr[c] = darr[b];
}

void FuncB (void) {
    darr[c] = darr[b];
}

void FuncC (void) {
    darr[c] = darr[a] + darr[b];
}

void FuncD (void) {
    darr[c] = darr[a] + darr[b];
}

void main(void) {
    FuncA ();
    FuncB ();
    while(1);
}

; FUNCTION FuncA (BEGIN)      ; FUNCTION FuncB (BEGIN)      ; FUNCTION FuncC (BEGIN);
      FUNCTION FuncD (BEGIN)
      R MOV A,#LOW darr      R MOV A,#LOW darr      R MOV A,#LOW darr
----- + ----- R MOV A,#LOW darr
      R ADD A,b              R ADD A,b              R ADD A,b
      |      R ADD A,b              R ADD A,b              R ADD A,b
      |      MOV R0,A      ----- + ----- MOV R0,A              MOV R0,A
      |      MOV A,@R0      |      MOV A,@R0              MOV A,@R0
      |      MOV R7,A      |      MOV R7,A              MOV R7,A
      |      MOV R7,A              MOV R7,A              MOV R7,A
      R MOV A,#LOW darr      |      R MOV A,#LOW darr      R MOV A,#LOW darr
----- + ----- R MOV A,#LOW d      arr
      R ADD A,c              |      R ADD A,c              R ADD A,a
      |      R ADD A,a              |      R ADD A,c              MOV R0,A
      |      MOV R0,A              |      MOV R0,A              MOV R0,A
      |      MOV R0,A              |      MOV R0,A              MOV A,@R0
      MOV @R0,AR7      ----- + ----- MOV @R0,AR7              ADD A,R7
      |      MOV A,@R0              RET              MOV R7,A
      |      RET              |      RET              R MOV A,#LOW darr
      |      ADD A,R7              |      RET              R ADD A,c
      |      MOV R7,A              o-> Common code for FuncA and FuncB
      |      R MOV A,#LOW darr
      |      R ADD A,c

```

```

|          MOV R0,A
|          MOV @R0,AR7
|          RET
o-> Common code for FuncC and FuncD

```

In the example above the functions FuncC() and FuncD() are removed when the [REMOVEUNUSED](#) linker directive is specified. Now, the first part of the common code block (FUNCTION ?L?COM0001) is no longer necessary. In previous versions this was still part of the image. With the new linker enhancement even this code block is removed.

```

----- FUNCTION ?L?COM0001 (BEGIN) -----
000021 7408 MOV A,#LOW darr ---+
000023 2519 ADD A,b          | \
000025 F8MOV R0,A           | \
000026 E6MOV A,@R0          |  o-> Common code for FuncC and FuncD
000027 FFMov R7,A           |
000028 7408 MOV A,#LOW darr ---+
00002A ?L?COM0002:
00002A F8MOV R0,A           ---+
00002B E6MOV A,@R0          | \
00002C FFMov R7,A           | \
00002D 7408 MOV A,#LOW darr |  o-> Common code for FuncA and FuncB
00002F 251A ADD A,c          |
000031 F8MOV R0,A           |
000032 A607 MOV @R0,AR7      ---+
000034 22RET
----- FUNCTION ?L?COM0001 (END) -----

```

AX51 Macro Assembler

Enhanced: [SEGMENT](#) assembler statement now also supports ALIGN (1) as minimum *alignment* value.

New Supported Devices

[CAST R8051XC3](#).

[Maxim 78M6613](#).

[SigmaDesigns ZM3102](#), [ZM4102](#), [ZM4101](#), and [SD3402](#).

[Silabs C8051F970-A-GM](#), [C8051F971-A-GM](#), [C8051F972-A-GM](#), [C8051F973-A-GM](#), [C8051F974-A-GM](#), [C8051F975-A-GM](#), [C8051F370](#), [C8051F371](#), [C8051F374](#), [C8051F375](#), [C8051F388](#), [C8051F389](#), [C8051F38A](#), [C8051F38B](#), [C8051F38C](#), [C8051F390](#), [C8051F391](#), [C8051F392](#), [C8051F393](#), [C8051F394](#), [C8051F395](#), [C8051F396](#), [C8051F397](#), [C8051F398](#), [C8051F399](#), [C8051F750](#), [C8051F750B](#), [C8051F751](#), [C8051F751B](#), [C8051F752](#), [C8051F752B](#), [C8051F755B](#), [C8051F756B](#), [C8051F757B](#), [C8051F760](#), [C8051F761](#), [C8051F762](#), [C8051F765](#), [C8051F766](#), [C8051F767](#), [C8051T626](#), [C8051T627](#), [C8051T670](#), and [C8051T671](#).

[Texas Instruments CC2541F128](#), [CC2541F256](#), [CC2543](#), [CC2544](#), and [CC2545](#).

[Vitesse VSC7388](#), [VSC7389](#), [VSC7390](#), [VSC7391](#), [VSC7395](#), [VSC7420](#), and [VSC7422](#).

uVision

This C51 release comes with μ Vision V5.11.2.0.

11. C51 Version 9.52

Release Date: 4th July, 2013

Cx51 Compiler

Modified: the warning C294: unreachable code will be issued instead of a compiler error for unreachable code statements. Example:

```
int foo = 1 ;
int bar ;

int main( void )
{
    switch( foo )
    {
        bar = 0 ;    // warning C294: unreachable code
    case 1:
        bar = 1 ;
        break ;
    case 2:
        bar = 2 ;
        break ;
    }
    return( 0 ) ;
}
```

Corrected: wrong xDATA address calculation which may occurs with combined pointer and int arithmetic. Example:

```
unsigned char xdata b[256]; // Problem does not exist when array size > 256
void xdata *p;
unsigned int i = 256;

void main (void) {
    p = &b[256-i];    // Correct result.
    p = b + 256 - i;  // Incorrect result. Only the LOW BYTE of i has been used for
    the calculation.
}
```

Corrected: an ignored pointer cast which occurs under some circumstances with far and generic pointers. Example:

```
unsigned short foo (char far *farPtr) {
    return (unsigned char) farPtr; // Explicit cast is ignored.
}
```

LX51 Linker/Locator

Corrected: a wrong address calculation which occurs when const in code banks combined with linker code packing.

Added: Error 144: OVERLAY GROUP SEGMENT CANNOT HAVE 'LAST' ADDRESS ASSIGNMENT message. The LAST attribute cannot be used to locate segments that collect overlayable segments.

AX51 Assembler

Corrected: an erroneously issued `Error A57 'REGISTER USAGE' REQUIRES A PUBLIC CODE SYMBOL` which occurs when `REGUSE` directive is used by mixed-case (composed by upper and lower case characters) symbols.

New Supported Devices

Texas Instruments [CC2541F128](#) and [CC2541F256](#).

ULINK2

With this release the firmware of the ULINK2 target debugger will be updated to version 2 which will not work with older C51 installations. The `.\c51\ULINK\Utilities\UL2_Configure.exe` tool allows to switch back to an older firmware version when backward compatibility is needed.

uVision

This C51 release comes with μ Vision V4.72.9.0.

12. C51 Version 9.51a

Release Date: 25th February, 2013

uVision

This C51 release comes with μ Vision V4.60.6.10.

Corrected: synchronization of settings between tabs in "Options for Target" dialog.

13. C51 Version 9.51

Release Date: 21st January, 2013

New Supported Devices

Silicon Laboratories Inc. C8051F370, C8051F371, C8051F374, C8051F375, C8051F390, C8051F391, C8051F392, C8051F393, C8051F394, C8051F395, C8051F396, C8051F397, C8051F398, C8051F399, C8051T620, C8051T621, C8051T622, C8051T623, C8051T626, and C8051T627 devices.

Cx51 Compiler

Corrected: calculation of negative constants within nested calls may create incorrect results (this problem was introduced in C51 V9.50a). For example:

```
#define TDO 5
#define GET_TDO() (Arr[TDO])
unsigned char xdata Arr[10];

unsigned char TestTDO() {
    unsigned char ret;
    ret = (unsigned char)((GET_TDO()-1)*2)-1; // Incorrect result. For the
    negative constant a subtraction has been used instead of an addition.
    return ret;
}
```

Corrected: incorrect pointer arithmetic with subtract of unsigned int variables for XDATA arrays with sizeof < 256 bytes. For example:

```
unsigned char xdata b[256]; /* Problem does not exist when array size > 256 */
void xdata *p;
unsigned int i = 256;      /* Problem only appears for unsigned int variables */

void main (void) {
    p = &b[256-i]; /* Works no problem when array index is used */
    p = b + 256 - i; /* Failed on pointer arithmetic when uint variable is
    subtracted */
}
```

LX51 Linker/Locator

Corrected: a potential DPTR corruption which may occurs in code-banking applications when Global Register Coloring is enabled.

uVision

This C51 release comes with µVision V4.60.6.8.

Enhanced: the Logic Analyzer allows rearranging signals through a simple drag&drop of the signal name. Signals can be scaled in width and height.

Corrected: under certain circumstances the Source Browser incorrectly reported several definitions of an enum.

14. C51 Version 9.50a

Release Date: 15th June, 2012

New Supported Devices

Cypress CY8C3244AXI-153, CY8C3244LTI-123, CY8C3244LTI-130, CY8C3244PVI-133, CY8C3245AXI-158, CY8C3245AXI-166, CY8C3245LTI-129, CY8C3245LTI-139, CY8C3245LTI-144, CY8C3245LTI-163, CY8C3245PVI-134, CY8C3245PVI-150, CY8C3246AXI-131, CY8C3246AXI-138, CY8C3246LTI-125, CY8C3246LTI-128, CY8C3246LTI-149, CY8C3246LTI-162, CY8C3246PVI-122, CY8C3246PVI-147, CY8C3444AXI-116, CY8C3444LTI-110, CY8C3444LTI-119, CY8C3444PVI-100, CY8C3445AXI-104, CY8C3445AXI-108, CY8C3445LTI-078, CY8C3445LTI-079, CY8C3445LTI-081, CY8C3445LTI-089, CY8C3445PVI-090, CY8C3445PVI-094, CY8C3446AXI-099, CY8C3446AXI-105, CY8C3446LTI-073, CY8C3446LTI-074, CY8C3446LTI-083, CY8C3446LTI-085, CY8C3446PVI-076, CY8C3446PVI-091, CY8C3446PVI-102, CY8C3665AXI-010, CY8C3665AXI-016, CY8C3665LTI-006, CY8C3665LTI-044, CY8C3665PVI-007, CY8C3665PVI-008, CY8C3665PVI-080, CY8C3666AXI-036, CY8C3666AXI-037, CY8C3666AXI-052, CY8C3666LTI-027, CY8C3666LTI-050, CY8C3865AXI-019, CY8C3865LTI-014, CY8C3865LTI-062, CY8C3865PVI-060, CY8C3865PVI-063, CY8C3866AXI-035, CY8C3866AXI-039, CY8C3866AXI-040, CY8C3866LTI-030, CY8C3866LTI-067, CY8C3866LTI-068, CY8C3866PVI-021, and CY8C3866PVI-070.

Zilog Z51F0410, Z51F3220, Z51F3221, Z51F6412, and Z51F811.

uVision

This C51 release comes with µVision V4.53.0.6. which includes the new Scintilla based editor.

The new editor includes the following enhancements:

- Encoding for UTF-8 Unicode, DBCS Korean, DBCS Japanese, and DBCS Chinese languages. Unicode and Asian ANSI encoding is recognized automatically when a file is opened.
- Monospaced fonts and proportional fonts are supported.
- Syntax coloring has been extended.
- Unprintable characters, such as End-Of-Line, can be visualized in the editor.
- The [Outlining](#) menu has been simplified. Outlining information is saved and restored for each file.
- [Search and replace](#) utilities (Incremental Find, Find-in-Files, and Replace) have been reworked.
- Text can be zoomed with Ctrl+mouse wheel. The information is saved and restored for each file.
- In case device-specific books are not found in the local installation, then www.keil.com is scanned for a matching document.

Corrections: Scrolling quickly through large files with Page Up or Page Down works smoothly. The editor's context menu can be closed by pressing ESC. Breakpoints can be set now with a simple

click into the editor margin. Under some circumstances the Debugger showed wrong values of arrays or structures in the Watch window.

Refer to [Revision History](#) for a complete list.

15. C51 Version 9.06

Release Date: 15th February, 2012

New Supported Devices

Atmel [AT89LP51RB2](#), [AT89LP51RC2](#), [AT89LP51IC2](#), [AT89LP51RD2](#), [AT89LP51ED2](#), and [AT89LP51ID2](#) devices.

Infineon [TLE9831](#), [TLE9832](#), [TLE9833](#), [TLE9834](#), and [TLE9835](#) devices.

Maxim [71M6531D](#), [71M6531F](#), [71M6532D](#), [71M6532F](#), [71M6534H](#), [71M6541D](#), [71M6541F](#), [71M6542F](#), [71M6543F](#), and [71M6543G](#) devices.

Silabs [C8051F969](#), [C8051F968](#), [C8051F967](#), [C8051F966](#), [C8051F965](#), [C8051F964](#), [C8051F963](#), [C8051F962](#), [C8051F961](#), [C8051F960](#), [Si1037](#), [Si1036](#), [Si1035](#), [Si1034](#), [Si1033](#), [Si1032](#), [Si1031](#), [Si1030](#), [Si1027](#), [Si1026](#), [Si1025](#), [Si1024](#), [Si1023](#), [Si1022](#), [Si1021](#), and [Si1020](#) devices.

Device Support

Added: Quick_Test example for Infineon TLE983x based devices in folder `..\C51\Examples\Infineon TLE983x\`.

Added: banking example for Maxim 71M6543G device in folder `..\C51\Examples\Maxim\BankEx1\`.

Added: banking example for Maxim 71M6534H device in folder `..\C51\Examples\Maxim\BankEx2\`.

uVision

This C51 release comes with μ Vision V4.24.00.

16. C51 Version 9.05

Release Date: 8th August, 2011

C51 Compiler

Improved: access to bit-field members with size 1 bit. The compiler uses bit instructions to access such bit-field members objects that are defined with the `bdata` memory type. This is now extended also to structs that are defined with the `extern` attribute.

Corrected: Common sub-expression elimination can deliver incorrect values when array pointers are used. Example:

```
int foo (unsigned char dat[]) {
    int len1, len2, ofs;

    ofs = 5;
    len1 = dat[ofs];
    if(len1 > 0x10) return -1;
    ofs += len1 + 1; // modify 'ofs'
    len2 = dat[ofs]; // 'dat[ofs]' not reloaded, instead value of 'len1' is used
    return len2;
}
```

Corrected: Wrong code with pointer arithmetic and conversions to long. Example:

```
unsigned char *p;
unsigned int code a1[10];
unsigned char xdata a2[500];

void foo (void) {
    unsigned long r1 = (unsigned long)(p - (a2 + a1[0])); // wrong
    unsigned long r2 = (unsigned long)(p - (unsigned long)(a2 + a1[0])); // work
    around
}
```

Corrected: Pointer arithmetic with conversion to 'unsigned long' rejected. Example:

```
unsigned char xdata *ptr1;
unsigned char xdata *ptr2;
unsigned long i4;

void foo (void) {
    i4 = (ptr1 - ptr2); // pointer conversion rejected, instead an error was issued
}
```

Lx51 Linker/LOcater

Corrected: When Linker Code Packing is used with a banking application, segments or the content of segments may get lost. This problem was introduced with C51 Version 9.03.

New Supported Devices

CoreRiver [ADCore200](#), [ADCore210](#), [ADCore220](#), [AmpCore100](#), [ChargerCore2.0](#), [GC230](#), [GC400](#), [GC410](#), [GC81L541A0](#), [GC81L581A0](#), [GC81L591A0](#), [GC89L541A0](#), [GC89L581A0](#), [GC89L591A0](#), and [HallCore110](#) devices.

Nordic [nRF8200](#) device.

Nuvoton [N78E055A](#), [N78E059A](#), [N78E366A](#), [N78E517A](#), [N79E234](#), [N79E235](#), [N79E342](#), [N79E352](#), [N79E822](#), [N79E823](#), [N79E824](#), [N79E825](#), [N79E843](#), [N79E844](#), [N79E845](#), [N79E853](#), [N79E854](#), [N79E855](#), [N79E875](#), [W78E051D](#), [W78E052D](#), [W78E054D](#), [W78E058D](#), and [W78E516D](#) devices.

Silabs [C8051T620](#), [C8051T621](#), [C8051T622](#), [C8051T623](#), [C8051T320](#), [C8051T321](#), [C8051T322](#), [C8051T323](#), [C8051T326](#), [C8051T327](#), [C8051F380](#), [C8051F381](#), [C8051F382](#), [C8051F383](#), [C8051F384](#), [C8051F385](#), [C8051F386](#), and [C8051F387](#) devices.

uVision

This C51 release comes with μ Vision V4.22.00.

17. C51 Version 9.03

Release Date: 14th February, 2011

Cx51 Compiler

Improved: Evatronix R8051XC2 core now uses the MDU for signed long divisions.

Corrected: With OPTIMIZE(8, SIZE) or higher optimization levels there is a potential problem with common code generation of long/float store operations that store constants with different memory types. Example:

```
int flag;

long xdata l;

long *pl;

void main (void) {

    if(flag) {    l = 0; flag = 1; }    // store with identical constant value but
different space
    else        { *pl = 0; flag = 1; }    // may create incorrect common blocks }
```

LX51 Compiler

Improved: When using Linker Code Packing the gaps in the BANKAREA are not optimized.

Corrected: Potential SEGMENT OVERLAPS when using Linker Code Packing on a banking application.

New Device Support

Atmel [AT80C51RD2](#), [AT89LP213](#), [AT89LP216](#), [AT89LP52](#), and [AT89LP6440](#) devices.

Infineon [XC878LM-13F](#), [XC878CLM-13F](#), [XC878LM-16F](#), [XC878CLM-16F](#), [XC874-13F](#), [XC874LM-13F](#), [XC874CM-13F](#), [XC874CLM-13F](#), [XC874-16F](#), [XC874LM-16F](#), [XC874CM-16F](#), [XC874CLM-16F](#), and [XC836MT-1F](#) devices.

NXP [P87C51FA](#), [P87C51FB](#), [P89CV51RB2](#), [P89CV51RC2](#), [P89CV51RD2](#), [P89LPC779](#), and [P87V660X2](#) devices.

Silabs [C8051F348](#), [C8051F349](#), [C8051T606](#), [C8051T630](#), [C8051T631](#), [C8051T632](#), [C8051T633](#), [C8051T634](#), [C8051T635](#), [C8051F540](#), [C8051F541](#), [C8051F542](#), [C8051F543](#), [C8051F544](#), [C8051F545](#), [C8051F546](#), [C8051F547](#), [C8051F550](#), [C8051F551](#), [C8051F552](#), [C8051F553](#), [C8051F554](#), [C8051F555](#), [C8051F556](#), [C8051F557](#), [C8051F560](#), [C8051F561](#), [C8051F562](#), [C8051F563](#), [C8051F564](#), [C8051F565](#), [C8051F566](#), [C8051F567](#), [C8051F568](#), [C8051F569](#), [C8051F570](#), [C8051F571](#), [C8051F572](#), [C8051F573](#), [C8051F574](#), [C8051F575](#), [C8051F34A](#), [C8051F34B](#), [C8051F34C](#), [C8051F34D](#), [C8051F716](#), [C8051F717](#), [C8051F980](#), [C8051F981](#), [C8051F982](#), [C8051F983](#), [C8051F985](#), [C8051F986](#), [C8051F987](#), [C8051F988](#), [C8051F989](#), [C8051F990](#), [C8051F991](#), [C8051F996](#), [C8051F997](#), [C8051F800](#), [C8051F801](#), [C8051F802](#), [C8051F803](#), [C8051F804](#), [C8051F805](#), [C8051F806](#), [C8051F807](#), [C8051F808](#),

C8051F809, C8051F810, C8051F811, C8051F812, C8051F813, C8051F814, C8051F815, C8051F816, C8051F817, C8051F818, C8051F819, C8051F820, C8051F821, C8051F822, C8051F823, C8051F824, C8051F825, C8051F826, C8051F827, C8051F828, C8051F829, C8051F830, C8051F831, C8051F832, C8051F833, C8051F834, C8051F835, C8051F901, C8051F902, C8051F911, C8051F912, Si1000, Si1001, Si1002, Si1003, Si1004, Si1005, Si1010, Si1011, Si1012, Si1013, Si1014, Si1015, C8051F520A, C8051F521A, C8051F523A, C8051F524A, C8051F526A, C8051F527A, C8051F530A, C8051F531A, C8051F533A, C8051F534A, C8051F536A, and C8051F537A devices.

Teridian Semiconductors 78M6612, 78M6618, and 71M6543 devices.

Texas Instruments CC2530F32, CC2530F64, CC2530F128, CC2530F256, CC2531F128, CC2531F256, CC2533F32, CC2533F64, CC2533F96, CC2540F128, and CC2540F256 devices.

uVision IDE

This C51 release comes with μ Vision V4.14.16.

18. C51 Version 9.02a

Release Date: 12th July, 2010

Cx51 Compiler

Improved: access to bit-field members with size 1 bit. The compiler uses bit instructions to access such bit-field members. When objects are defined with the `bdata` memory type, direct bit addressing is used. Example:

```
struct bf { unsigned char b0:1; unsigned char b1:1; };
struct bf a;
struct bf bdata b;
:
if (a.b0 && b.b1) b.b1 = 0;
```

Corrected: multiplication `long = int * int` is potentially incorrect in Dallas 390 mode.

Corrected: explicit cast to unsigned char was ignored with complex address arithmetic. Example:

```
unsigned char far table[256];
unsigned char i, v;
:
v = table[(unsigned char)(16+i+20)]; // index now truncated to 8-bit
```

Corrected: when using conditional operators (`? :`) in complex parameter lists, there is a potential for unbalanced PUSH/POP instructions. This typically creates a application program crash at the function call.

C Run-Time Library

Corrected: the function `toint` did not detect the value range `0x59 - 0x40` as invalid. Now the function returns -1 for these values.

Corrected: timing of Multiplication Division Unit (MDU) in Evatronix R8051XC2 device is faster and now reflected in the C Library. The MDU timing for int/long multiplication and long division is adjusted.

New Supported Devices

Infineon [XC835MT-2F](#), [XC836-2F](#), [XC836M-1F](#), [XC836M-2F](#), [XC836MT-2F](#), and [XC836T-2F](#) devices.

Device Simulation

Corrected: SiLabs C8051F41x devices: SMBus simulation when using I²C generator.

Corrected: SiLabs C8051F12x devices: automatic page switch for interrupts and timing of timer 2/3/4.

Corrected: SiLabs C8051F12x devices: on I²C the receive of more than 256 bytes now generates a stop.

Corrected: SiLabs C8051F36x devices: crossbar did not connect the right I/O signals under some circumstances.

Corrected: Evatronix T8051: CPU instruction timing.

19. C51 Version 9.01

Release Date: 24th February, 2010

uVision IDE

C51 now includes the new [µVision4 IDE](#).

New Supported Devices

Infineon [XC822T-0F](#), [XC822M-1F](#), [XC822MT-1F](#), [XC824MT-1F](#), and [XC824M-1F](#) devices.

Device Support

Added: debug support for Infineon XC82X devices.

Cx51 Compiler

Corrected: when MODDA is used and *int* numbers are multiplied and assigned to *long*, the result was potentially incorrect.

20. C51 Version 9.00

Release Date: 26th October, 2009

uVision IDE

C51 now includes the new [µVision4 IDE](#).

New Supported Devices

Evatronix [R8051XC\(1DPTR\)](#), [R8051XC2\(1 DPTR\)](#), [R8051XC2\(2 DPTR\)](#), [R8051XC2\(8 DPTR\)](#), [R8051XC2-A\(1 DPTR\)](#), [R8051XC2-A\(2 DPTR\)](#), [R8051XC2-A\(8 DPTR\)](#), [R8051XC2-AF](#), [R8051XC2-B\(1 DPTR\)](#), [R8051XC2-B\(2 DPTR\)](#), [R8051XC2-B\(8 DPTR\)](#), and [R8051XC2-BF](#) devices.

Nordic [nRFLU1P-F16](#) and [nRFLU1P-F32](#) devices.

NXP [P89LPC9361](#) device.

Silabs [C8051F580](#), [C8051F581](#), [C8051F582](#), [C8051F583](#), [C8051F584](#), [C8051F585](#), [C8051F586](#), [C8051F587](#), [C8051F588](#), [C8051F589](#), [C8051F590](#), and [C8051F591](#) devices.

Device Support

Corrected device settings for Infineon [XC888-6FF](#), [XC888CM-8FF](#), [XC888LM-6FF](#), [XC886-6FF](#), [XC866L-1FR](#), [XC866L-2FR](#), [XC866L-4FR](#), and [XC864-1FRI](#) devices.

Corrected: device settings for Nordic Semiconductor [nRF24E1](#), [nRF24E2](#), [nRF9E5](#), [nRF24LU1](#), and [nRF24LE1](#) devices.

Cx51 Compiler

Corrected: constant folding of two negative array index values. For example:

```
unsigned char arr[512];
unsigned int i;

i = arr[i-1-5]; // incorrect in C51 V8: arr[i-4] instead of arr[i-6]
```

Corrected: when using the NOAREGS directive, complex arithmetic with nested calls may create incorrect results. For example:

```
#pragma NOAREGS

int result;
extern char f(unsigned char idx);

result = (f(1)*0x100+f(0)) - (f(3)*0x100+f(2)); // incorrect result. POP destroys
value in ACC
```

Lx51 Linker

Corrected: when using OPTIMIZE(10) or above, there was a potential that common code blocks are called incorrectly. Therefore programs may have operated incorrectly.

BL51 Linker/Locater

Corrected: when using RTX51 user interrupt functions were overlapping with RTX ISR vectors which resulted in a linker warning.

Corrected: data overlaying may not work when the last input module contains an interrupt function; the linker incorrectly issues WARNING 16: main uncalled.

21. C51 Version 8.18

Release Date: 30th March, 2009

Device Support

Added debug support for the NXP [P89LPC9408](#) in the [LPC900 EPM Emulator/Programmer](#).

New Supported Device

Nuvoton [W681308](#) device.

NXP:

- [C8051F500](#)
- [C8051F501](#)
- [C8051F504](#)
- [C8051F505](#)
- [C8051F506](#)
- [C8051F507](#)
- [C8051F508](#)
- [C8051F509](#)
- [C8051F510](#)
- [C8051F511](#)
- [C8051F700](#)
- [C8051F701](#)
- [C8051F702](#)
- [C8051F703](#)
- [C8051F704](#)
- [C8051F705](#)
- [C8051F706](#)
- [C8051F707](#)
- [C8051F708](#)
- [C8051F709](#)
- [C8051F710](#)
- [C8051F711](#)
- [C8051F712](#)
- [C8051F713](#)
- [C8051F714](#)

- [C8051F715](#)

ULINK2 Support

Corrected potential deadlock on ST uPSD targets.

Device Simulation

Corrected simulation of Infineon XC800 MDU.

Corrected behaviour of EXFn and TOGn on SiLabs C8051F12x/F13x devices.

Added simulation for Atmel [AT89C51RE2](#), including simulation of second UART.

Cx51 Compiler

Corrected failed initialization on far addresses when the object is located with `_at_`.

22. C51 Version 8.17a

Release Date: 22nd December, 2008

Device Support

Added debug support in the ADI Monitor Driver for the following devices from Analog Devices:

- [ADE5166](#)
- [ADE5169](#)
- [ADE5566](#)
- [ADE5569](#)
- [ADE7166F16](#)
- [ADE7166F8](#)
- [ADE169F16](#)
- [ADE7566F16](#)
- [ADE7566F8](#)

Added debug support for NXPP89LPC9321 and P89LPC9351 devices in the LPC900 EPM Emulator/Programmer.

Added Nuvoton devices in the Device Database.

Corrected NXP [P89LPC917](#) peripheral dialog to show pin P2.2 is available (instead of P2.5).

Corrected UART0 baudrate display when Timer 2/3/4 is used as baudrate generator on SiLabs [C8051F13x](#).

Device Simulation

Added support for V: user-defined memory area for NXP 80C51MX devices.

Corrected simulation of Reset Source Register (RSTSRC) and SFR Page Control Register (SFRPGCN) for SiLabs [C8051Fxxx](#) devices.

Corrected handling of Automatic Page Control Enable (SFRPGCN) and Reset Source Register (RSTSRC) for SiLabs [C8051Fxxx](#) devices.

Corrected simulation of PLLLCK (PLL Lock Flag) for SiLabs [C8051F12x/13x](#) devices. PLLLCK is now set when PLL is configured correctly and frequency is locked.

Corrected simulation issues with the Evatronix [R8051XC](#) peripherals DMA and interrupt.

Cx51 Compiler

Corrected issue with operations where two long operands are loaded from complex arrays. There was a potential R0 register overwrite and the result of the long operation was in such cases incorrect.

Ax51 Macro Assembler

Corrected issue with NXP 80C51MX mode. DATA, IDATA, and EDATA can now be placed to absolute addresses 0x7F0000 and above.

Added [ECRM directive](#) that allows the expansion of generic `CALL` instructions to `ECALL` for NXP 80C51MX devices.

23. C51 Version 8.16a

Release Date: 26th August, 2008

Cx51 Compiler

Corrected a problem introduced in V8.15. When int numbers are multiplied and assigned to long, the result may be incorrect.

Corrected `_at_` problem with linker code packing fixed.

AX51 Macro Assembler

Added enhancement for the NXP i.MX devices, `CALL/JMP` instructions are encoded to `ECALL/JMP` when needed.

Device Support

Added ULINK and Infineon DAS (Device Access Server) support for the [XC864](#) device.

Optimization for Evatronix [R8051XC](#) XDATA Banking example.

Enhanced Infineon XC800 startup code.

Added support for Infineon [XC864](#) including a Blinky example.

Added Syntek Semiconductors [STK6031](#) and [STK6032](#) devices to device database.

Device Simulation

Added support for SiLABS [C8051F360/1/2/3/4/5/6/7/8/9](#) and [C8051F410/1/2/3](#).

24. C51 Version 8.15

Release Date: 30th May, 2008

Cx51 Compiler

Corrected a problem where interrupt functions combined with [NOINTVECTOR](#) were not detected by the linker as a new root, this reported an incorrect linker warning.

Corrected, when using Dallas 390 mode with ROM(D512K) or ROM(D16M), pdata arrays could not be located anywhere in memory.

Corrected, when using the XCROM directive in combination with function pointers, constant initializations were omitted.

Long multiplication performance with two unsigned int/char arguments has been improved.

Device Support

Support for the Infineon USCALE XC800 hardware via the Infineon DAS Client for XC800 has been added.

Device Simulation

Access to MACACC for SiLAB [C8051F12x](#) and [C8051F13x](#) devices has been corrected.

Device support and simulation for Infineon [XC878](#) has been added.

Lx51 Linker/LOcater

Corrected a Linker Code Packing issue which may have incorrectly combined blocks from several code banks into common areas.

ULINK2 Support

Added support for Debug and Flash-Programming support of NXP [P89LPC952](#) and [P89LPC954](#).

25. C51 Version 8.12

Release Date: 31st January, 2008

Device Support

Added: device support and simulation for SiLABS [C8051T600/1/2/3/4/5](#) and [C8051T610/1/2/3/4/5/6/7](#).

Cx51 Compiler

Corrected a problem where nested calls with struct pointer arguments were incorrectly processed.

Lx51 Linker/Locater

Corrected: sfr16 definitions in assembly code and C source file may generate Warning L46: SFR SYMBOL HAS DIFFERENT VALUE.

uVision IDE

Corrected PORTx and PCA output pins on SiLABS C8051F12x did not correctly update in simulation.

26. C51 Version 8.11a

Release Date: 15th January, 2008

Device Support

Added support for Ramtron VRS51L3072 and VRS51L3174 devices.

Added support for Nordic Semiconductor nRF24LU1.

Added support and simulation for SiLABS C8051F336, C8051F337, C8051F338, and C8051F339.

uVision IDE

Corrected PORTx and PCA output pins on SiLABS C8051F12x did not correctly update in simulation.

27. C51 Version 8.10

Release Date: 19th November, 2007

Device Support

Added support for new Megawin MPC82G516A and MPC82L54A devices.

Corrected startup code for Infineon XC88x AC step devices, this requires the device to be set to VCO bypass mode before PLL switching.

NXP P89V52X2 device support added.

uVision IDE

Corrected: simulation of MULRDY and OSCICL for SiLABS C8051F3xx series.

Enhanced simulation support for Evatronix R8051XC watchdog timer with optional prescaler. For details refer to [Application Note 191: Toolchain Extensions for R8051XC Core](#).

Corrected the DPTR simulation of Evatronix R8051XC; when 2 DPTR were selected the auto-increment feature (DPC register) did not work. Two R8051XC devices are now in the device database: R8051XC (8 DPTR) with simulation for 8 DPTR's, R8051XC (2 DPTR) with simulation for 2 DPTR's.

SiLABS simulation for UART #1 had a problem with the transmit interrupt bit (TI) when SFR page was set to 1. This is now corrected.

Cx51 Compiler

Corrected the `MODC2` directive which did not correctly save and restore multiple DPTR registers on interrupt entry/exit.

Assembler instructions inserted with `#pragma ASM` trigger now register usage of all CPU registers and therefore avoids register clashes.

Improved the detection of conflicting memory types when used in combination with typedef's, for example:

```
typedef char    code CCHAR;  
typedef CCHAR xdata XCHAR; // generates now WARNING C185: different memory space  
CCHAR    idata var2;       // generates now WARNING C185: different memory space
```

Lx51 Linker/LOcater

Corrected a problem where REMOVEDUNUSED did not correctly work with SROM symbols and linker code packing.

Corrected a problem where debug symbols of absolute bits generated in AX51 had the wrong offset.

Segment locating with the `LAST` keyword generated unnecessary memory gaps when used with code banking. This is now corrected.

BL51 Linker/Locater

Corrected a problem where segments with a `JMP` instruction as last instruction were located at the end of a 2KB block which generated a linker error.

ULINK and ULINK2

Corrected a problem where verify failed on μ PSD devices when common segments were located to code banks, but no bank 0 exists.

28. C51 Version 8.09a

Release Date: 30th July, 2007

uVision IDE

Added support for the Megawin [MPC82G516A](#) and [MPC82L54A](#).

Corrected a problem with the simulated timing of Timer 1 on Infineon XC800 devices.

Corrected a problem with the clock calculation for Infineon XC88x devices.

Corrected a problem with Dallas D80C400 simulation that prevented correctly switching to contiguous mode.

Added simulation for the [Atmel AT89C51AC3](#).

Corrected a problem with displaying data/idata variable values that occurred when debugging Infineon XC800 Devices with the DAS interface.

Enhanced the ADI Monitor Driver to support the 1-Pin Pod interface and new ADE7xxx devices. This driver may be selected under "Options - Project - Debug - Use: ADI Monitor Driver" to provide target driver support for Analog Devices ADuC834, ADuC84x, and ADE7xxx devices.

Cx51 Compiler

Corrected a code generation problem with bit-field arrays when the array index was the return value of a function.

MON51 Monitor

Corrected a potential communications problem with low-cost USB to COM-Port adapters.

MON390 Monitor

Corrected a potential communications problem with low-cost USB to COM-Port adapters.

29. C51 Version 8.08a

Release Date: 25th March, 2007

uVision IDE

Corrected a problem that caused a delay when starting signal functions. This delay has been removed and the startup behavior is identical to releases prior to version 8.06.

Corrected a problem with JBC instructions with regards to I/O ports. JBC instructions now read the SFR register (Px) instead of the I/O port value (PORTx).

Corrected a problem that could cause the IDE to crash when the mouse was right-clicked in the project window when no item was selected.

Added peripheral display dialogs for Port 4 and Port 5 of the NXP 89LPC952.

Added a new debugging options for Infineon XC800 devices. Under "Project - Options - Debug - ULINK Settings", "Disable interrupts during steps" has been implemented. This option disables interrupts during single-stepping which has the effect of executing instructions only from the current function.

Added support for Infineon TLE78xx devices.

Corrected problems with simulating the external interrupt inputs EINT0 and EINT1 on Infineon XC800 devices.

Corrected debugger startup problems with the Infineon DAS server.

Cx51 Compiler

Corrected a problem that neglected to perform integer promotion on complex arithmetic with char/unsigned char and multiplication or division.

AX51 Macro Assembler

Added definition for the `_DATE2_` macro for the A51 and AX51 Macro Assemblers.

30. C51 Version 8.06

Release Date: 15th January, 2007

uVision IDE

Added debugging and Flash programming support for new Infineon XC800 Devices (XC866-1FR and XC856). Support was added for ULINK and the Infineon DAS Server.

Enhanced XC800 startup code (`START_XC.A51`) to support pdata addressing ([C51: USING PDATA VARIABLES ON INFINEON XC800](#)).

Added core feature simulation for SST SmartCards.

Corrected a problem with the interrupt vector for the second UART on the NXP (Philips) P89LPC952. This had been incorrectly configured in the simulator and target dialog.

Initialized the PPAGE VTREG to 0 for all NXP (Philips) LPC900 devices. This allows you to simulate `movx @Ri` instructions without any configuration changes.

Corrected a problem with AT89S8252 EEPROM simulation.

Corrected a problem with the baudrate generator simulation on Atmel devices that have the X2 feature. Previously, the baudrate displayed incorrectly.

Corrected a problem with code banking on Mentor M8051EW cores.

Cx51 Compiler

Corrected a problem that removed unused code (i.e. with macros) even when it created side-effects.

Corrected a problem that generated a syntax error for a syntactically correct statement. For example:

```
struct st2 { unsigned char uc1; unsigned char uc2; };
struct st1 { struct st2 st2; unsigned char u1; unsigned char u2; };
struct st1 st;

void main (void) {
    (&st.st2)->uc1 = 0; // incorrectly generated an error
}
```

Corrected a problem with the `toint` library routine that flagged values `0x3A - 0x40` incorrectly.

ULINK/ULINK2

Added ULINK2 support for Infineon XC800 and STMicroelectronics uPSD Devices.

Added device support for Infineon XC886, XC888, and XC856 devices.

31. C51 Version 8.05

Release Date: 26th July, 2006

Cx51 Compiler

Corrected a semantic interpretation problem that prevented the compiler from issuing an error when accessing struct members with `s->member` rather than `s.member`.

Corrected a problem that may cause registers to be overwritten when using multiple dummy assignments to avoid unused variable warnings.

Corrected a potential code problem with incrementing far pointers using long constants. For example:

```
void func (void) {
    unsigned int i;
    long far* entry;

    while(i) {
        i--;          // increment missing
        entry += 1L; // due to 'far' pointer increment with 1L
    }
}
```

Corrected a potential problem on SmartMX devices when using Optimize level 8 or 9 without OBJECTADVANCED. For example:

```
#pragma MXP

int  xdata x;
char xdata * xdata p;

void f1 (char *);
void f2 (char *);

void main (void) {
    // uses CMPW instruction that affects 'Z' flag
    if (x == 0xFFFF) f1 (p); // MOVX A,Rx,@DPTR destroys 'Z' flag
    else f2 (p);           // and cannot be combined
}
```

uVision IDE

Added device simulation support for Infineon XC886 and Infineon XC888 devices.

Corrected simulation problems with two serial windows for Philips P89LPC952/954 devices.

Added several excluded simulation features for the Mentor M8051EW and corrected a problem with debugging non-banking applications.

Added several excluded simulation features for the Cast R8051XC and added simulation for peripheral timing and write operations into code memory.

Lx51 Linker/Locator

Corrected a linker optimization problem when using interbank call tables (?B_RST_BANK != 0xFF).

ULINK

Corrected a flash programming problem with the driver for STMicroelectronics uPSD3422 devices.

32. C51 Version 8.04

Release Date: 24th May, 2006

Cx51 Compiler

Enhanced *Warning C259: Pointer: Different MSpace*. This warning message is now used to indicate problems in situations where an address value is assigned.

Added *Warning C289: Converting Non-Pointer to Pointer* which is issued when an integer value is assigned to a pointer.

Corrected a problem with SRC file output having to do with optimized address values.

Corrected a problem with a missing dummy read to MD3 for `int*int` multiplication when using the Infineon/Evatronix MDU.

Added support for the Silicon Labs Arithmetic Accelerator (available in C8051F12x and F13x devices) in the far banking library.

uVision IDE

Added complete simulation and compiler support for the features of the Evatronix/Cast R8051XC core. Detailed information is provided in [Application Note 191: Toolchain Extensions for the R8051XC Core](#).

Enhanced simulation support for the Mentor M8051EW memory extension features. It is now possible to simulate code banking and far memory applications and the core features of the R8051XC. Detailed information is provided in [Application Note 171: Using M8051EW Memory Extension](#).

Added ULINK debugging support and DAS driver for new Infineon XC800 devices (XC886, XC888).

BL51 & LX51 Linker/Locator

Corrected a problem that may cause incorrect *WARNING L15: MULTIPLE CALL TO FUNCTION* messages when using `OVERLAY (* ! (func1, func2, func3, ...))` to group more than two functions.

33. C51 Version 8.02a

Release Date: 17th February, 2006

uVision IDE

Added peripheral simulation support and target debugging dialogs for the following devices:

- [Atmel AT89C51CC03](#),
- [Atmel AT89C51IE2](#),
- [Atmel AT89C51RE2](#),
- [Atmel AT89LP2052](#),
- [Atmel AT89LP4052](#),
- [Atmel AT89S8253](#),
- [Atmel AT8xC51SND1](#),
- [Philips P89LPC9102](#),
- [Philips P89LPC9103](#),
- [Philips P89LPC9107](#),
- [Philips P89LPC9221](#),
- [Philips P89LPC9311](#),
- [Philips P89LPC932A1](#),
- [Philips P89LPC938](#),
- [Philips P89LPC9401](#),
- [P89LPC9408](#) (without LCD Driver),
- [Philips P89LPC952](#),
- [Philips P89LPC964](#),
- [Philips P89LPC966](#),
- [Philips P89V660](#),
- [Philips P90V662](#),
- [Philips P89V664](#).

Revised peripheral simulation support and target debugging dialogs for the following devices:

- [AT89C5131/AT89C5131A](#) (added TWI),
- [AT8xC5132](#) (added TWI, ADC).

Added ULINK debugging support and DAS driver for new Infineon XC800 devices (XC886, XC888).

Lx51 Linker/Locator

Corrected a problem with code banking that caused the linker to incorrectly report Error L124: BANK SWITCH MODULE INCORRECT.

Cx51 Compiler

Corrected macro expansion problems with array index calculations with negative offset and far access to absolute memory locations.

ULINK

Added new JTAG device ID's for ST uPSD3212, uPSD3312, and uPSD3422.

34. C51 Version 8.01

Release date: 16th November, 2005.

uVision IDE

Added peripheral simulation support and target debugging dialogs for the following devices:

- [Silicon Labs C8051F120](#),
- [Silicon Labs C8051F121](#),
- [Silicon Labs C8051F122](#),
- [Silicon Labs C8051F123](#),
- [Silicon Labs C8051F124](#),
- [Silicon Labs C8051F125](#),
- [Silicon Labs C8051F126](#),
- [Silicon Labs C8051F127](#),
- [Silicon Labs C8051F130](#),
- [Silicon Labs C8051F131](#),
- [Silicon Labs C8051F132](#),
- [Silicon Labs C8051F133](#).

Support for Philips P89LPC952 Devices

Added support for the Philips P89LPC95x device series and the MCB950 Evaluation Board.

AX51 Macro Assembler

Corrected a problem that generated an error for forward references in symbols.

C51 Version 8 now includes the μ Vision3 IDE.

Corrected device simulation timing for Dallas DS89C420, DS89C430, DS89C440, and DS89C450 devices. Previously, the timer simulation was based on an older data book which was incorrect.

Support for Analog Devices ADuC83x and ADuC84x

Added support for a new debug driver for parts from Analog Devices. This driver connects the PC's COM port to the serial interface of the ADuC83x and ADuC84x. The on-chip Download/Debug Kernel may be used for program download and debugging. No additional monitor or firmware is required. Options – Project – Debug – Use: ADI Monitor Driver selects the target driver for the ADuC834 and ADuC84x devices. Detailed documentation is available in the Analog Devices ADuC83x/84x Download/Debug Driver User's Guide.

Support for Infineon XC866

Added support for the new Infineon XC800 Device series including simulation, ULINK driver, and MCBXC866 Evaluation Board support.

- Example projects are provided in the \KEIL\C51\EXAMPLES\INFINEON_XC866\ folder.
- Complete documentation (including details about the ULINK driver) is available in the MCBXC866 User's Guide.

Support for ST uPSD34xx Series

Added ULINK support for the new ST uPSD34xx device series.

Lx51 Linker/Locator

Corrected a potential problem with code packing when modules are translated with different optimization levels (less than 8) and the OBJECTADVANCED directive is used.

Cx51 Compiler

Changed the behavior of the L51_BANK.A51 configuration file to avoid glitches when using the RTX51 or RTX51 Tiny operating system together with code banking. This change has to do with the ?B_RESTORE_BANK entry when ?B_MODE=0.

Version number changed for logistic reasons. No other changes compared to V8.00.

C51 Compiler

Corrected a problem with the tan library routine (with INF values) that caused incorrect results for Dallas 390, 400, 5240, and 5250 devices.

Corrected a problem that caused the printf routine to output incorrect results for INF and NaN.

Corrected a problem with the scan and sscanf routines that caused an invalid return value of 0xFF instead of -1 when no arguments were processed.

Corrected library routines to properly handle updated behavior of the Memory Accelerator on the Dallas DS80C390 Rev C Devices.

Corrected a problem that caused the memmove routine to fail when copying overlapping xdata memory areas on Dallas 390, 400, 5240, and 5250 devices.

35. C51 Version 7.50a

Release date: 8th November, 2004.

Lx51 Linker/Locator

Corrected a potential problem with Linker Code Packing that might cause inefficient operation or a MEMORY SPACE OVERLAY warning.

Corrected a program that caused fixup error messages when using the new REMOVEUNUSED directive.

BL51 Linker/Locator

Incorporated a new BL51 Linker/Locator.

uVision IDE

Added extended memory simulation for the Mentor M8051EW. Refer to [Application Note 171: Using M8051EW Memory Extension](#) for more information.

Added peripheral simulation support and target debugging dialogs for the following devices:

- [Dallas Semiconductor DS89C420](#),
- [Dallas Semiconductor DS89C430](#),
- [Dallas Semiconductor DS89C440](#),
- [Dallas Semiconductor DS89C450](#),
- [Silicon Labs C8051F000](#),
- [Silicon Labs C8051F001](#),
- [Silicon Labs C8051F002](#),
- [Silicon Labs C8051F005](#),
- [Silicon Labs C8051F006](#),
- [Silicon Labs C8051F007](#),
- [Silicon Labs C8051F010](#),
- [Silicon Labs C8051F011](#),
- [Silicon Labs C8051F012](#),
- [Silicon Labs C8051F015](#),
- [Silicon Labs C8051F016](#),
- [Silicon Labs C8051F017](#),
- [Silicon Labs C8051F350](#),
- [Silicon Labs C8051F351](#),
- [Silicon Labs C8051F352](#),

- [Silicon Labs C8051F353](#),
- [SST SST89E554RC](#),
- [SST SST89E564RD](#),
- [SST SST89V554RC](#),
- [SST SST89V564RD](#).

ISD51 In-System Debugger

Added example configuration for Dallas DS89C420, 430, 440, and 450 devices.

C51 Compiler

Corrected a code generation issue for negative array index values. For example:

```
signed int i = v1 - v2;
value = table [i + 4]; // code works for 'signed char' index
                      // but fails with 'signed int' index
```

Added a new compiler directive ([MODC2](#)) that enables dual data pointer support on Cast and Evatronix R80515 cores.

Added a new compiler directive ([MODH2](#)) that enables dual data pointer support on Hynix, ST uPSD 33xx, and ST uPSD 34xx devices.

Cx51 Compiler

Corrected the following problems for the SmartMX instruction set:

- Switch/case with long types and ROM(HUGE).
- Over optimization with CMPW instruction.
- Stack adjustment failure with setjmp/longjmp library routines.

C Library

Corrected several library problems including:

- Corrected a problem on Dallas 390, 400, 5240, and 5250 devices with asin, acos, and atan when const data is not in stored in segment 0 (C:0x0000-C:0xFFFF).
- The labs function has been optimized and is now fully reentrant.
- Added a configuration symbol ?C?DPSEL that defines the DPSEL SFR address for MOD517(NOAU) multiple DPTR support. This may be used for Mentor M8051EW-based devices with multiple DPTRs similar to the Infineon 517 but with a different DPSEL SFR address. The following ?C?DPSEL definition may be included in a chip-specific STARTUP.A51 file.

```
PUBLIC ?C?DPSEL
?C?DPSEL DATA 0A2H ; DPSEL address for Mentor M8051EW
```



If this definition is not included, the DPSEL SFR is defined at the default address 0x92.

Lx51 Linker

Added the REMOVEUNUSED (abbreviation RU) directive which removes unused program and data segments provided that Data Overlaying is enabled.

ULINK

Added instruction trace support to the STMicroelectronics uPSD ULINK Driver. Refer to [Application Note 177: Using ULINK with STMicroelectronics Turbo \$\mu\$ PSD 3300/3400 Devices](#) and the example projects in the \KEIL\C51\EXAMPLES\ST_uPSD folder.

BL51/LX51 Linker

Improved the OVERLAY directive. Now, using `OVERLAY (* !(func1, func2))`, you may combine the segments of several function call trees. This is useful for interrupt functions that have overlayable data but use the same interrupt level. Such interrupt functions cannot interrupt each other. Therefore, data overlaying of both call trees is possible. For example:

```
void irq0 (void) interrupt 0 {
    unsigned char arr[10];
    arr[0] = 0;
}

void irq1 (void) interrupt 1 {
    unsigned char arr[10];
    arr[0] = 0;
}
```

If irq0 and irq1 are set to the same priority level their data areas may be overlaid. The OVERLAY directive may be specified to do that as follows:

```
BL51 ... OVERLAY (* ! (irq0, irq1))
```

The linker map file shows the following OVERLAY MAP.

```
SEGMENT  DATA_GROUP
+--> CALLED_SEGMENT  STARTLENGTH
-----
?PR?IRQ1?I 0008H000AH
* NEW ROOT *
?PR?IRQ0?I 0008H000AH
```

Note that both functions' call trees are overlaid.

36. C51 Version 7.20

Release date: 5th July, 2004.

uVision IDE

Added peripheral simulation support for the following devices:

- [Analog Devices ADuC841](#),
- [Analog Devices ADuC842](#),
- [Analog Devices ADuC843](#),
- [Analog Devices ADuC845](#),
- [Analog Devices ADuC847](#),
- [Analog Devices ADuC848](#),
- [Altium Nexar TSK51x](#) MCU Core,
- [Cast/Evatronix R8051](#) MCU Core,
- [Cast/Evatronix R80515](#) MCU core,
- [DCD DR8051](#) MCU core,
- [DCD DR80390](#) MCU core,
- [DCD DR8051XP](#) MCU core,
- [DCD DR80390XP](#) MCU core,
- [Mentor M8051EW](#) MCU core,
- [TI MSC1200Y2](#),
- [TI MSC1200Y3](#).

CPU core simulation may be expanded with user peripherals using the [AGSI Interface](#).

ISD51 Configuration Examples

Added ISD51 configuration for the TI MSC1200. Refer to [Application Note 193: Use ISD51 on TI MSC1200](#) for more information.

Revised the configuration for Philips LPC900 to match the current Philips LPC900 device versions.

FlashMonitor

Added a new configuration for the [Atmel AT8xC5122](#). Current configurations support almost all new Atmel device variants and are summarized in the `\KEIL\C51\F\FlashMon\ReadMe.txt` file.

C51 Compiler

Added intrinsic functions `_push_` and `_pop_` that may be used to save and restore sfr registers in interrupt functions.

Added support for the [Silicon Labs C8051F12x](#) multiply and accumulate unit (MAC0). The MDU_F120 directive enables use of MAC0 for int and long multiplication and long shift operations.

Cx51 Compiler

Corrected a problem with far pointers and long arithmetic.

ULINK Driver for ST uPSD series

Corrected a problem while programming Flash memory that caused the microcontroller to execute random instructions. The MCU is now forced into RESET while programming the Flash.

37. C51 Version 7.10

Release date: 18th March, 2004.

uVision IDE

Added simulation support for the following devices:

- Philips P89LPC935 and other downgraded LPC900 devices
- Philips P89C669
- ST μ PSD33xx

Corrected A/D converter simulation of the ADuC831.

Lx51 Linker Code Packing

Corrected a problem with code packing and JMP optimizations.

ULINK Driver for ST μ PSD series

Merged the Flash and Debug Setup Dialog. The ST Merge Utility (UTLADRM.EXE) may be called automatically. Additionally, a problem with lock-ups was corrected.

C51 Compiler

Corrected a problem with unbalanced PUSH/POP sequences in complex indirect function calls.

38. C51 Version 7.09

Release date: 2nd February, 2004.

uVision IDE - ULINK Support for STMicroelectronics uPSD3300/3400

Added final release of the ULINK debugging and Flash programming support for the new STMicroelectronics μ PSD3300/3400 devices. Program examples and an [Application Note](#) are provided in the \KEIL\C51\EXAMPLES\ST μ PSD folder.

Lx51 Linker Code Packing

Corrected a potential problem with linker code packing which may cause the linker/locater to hang.

39. C51 Version 7.08

Release date: 20th November, 2003.

C51 Compiler

Corrected a potential problem with the CSTXPTR function of the EEPROM program examples for the Atmel parts (\KEIL\C51\EXAMPLES\FarMemory\E2PROM on T89C51RD2 and \KEIL\C51\EXAMPLES\FarMemory\3 XData Areas on T89C51RD2).

Cx51 Compiler

When the setjmp and longjmp library routines are used in a code banking application, you must include the source file \KEIL\C51\LIB\SETJMP.A51 in your project. This file contains versions of these routines that support code banking. The routines in the standard library do not support code banking applications.

Corrected a problem passing complex function parameters when far variables are used as function arguments.

A51/AX51 Macro Assembler

Modified the assembler so that core SFR register symbols (ACC, B, DPL, DPH, PSW, SP) are automatically defined even when the NOMOD51 directive is used. This avoids error messages when generating assembler source (SRC) files from C modules that do not include a register definition file.

AX51 Macro Assembler

Added the EVEN directive which is described in the Ax51 User's Guide. Previously, this directive was available only in the A251 Assembler.

Lx51 Linker

Corrected a problem with the memory allocation strategy that was introduced in Version 7.07. This problem caused incorrect address calculations for constant segments that were located after packed code segments.

Corrected a problem with wildcards in the SEGMENTS directive. For example, SEGMENTS (?PR?*? module (C:0x4000)) did not locate all segments above 0x4000. Instead only the first segment was located at C:0x4000 and other segments were located within the CLASS definition.

ISD51 In-System Debugger

Corrected a problem with flash breakpoints when the flash block size (CBLK_SZ) was configured for 1 byte.

Cx51 Run-Time Library

Improved the rand library routine to deliver better distributed pseudo-random numbers. The new algorithm is based on a galois LFSR generator.

uVision IDE/Simulator

Added simulation for the following IP Cores:

- [Actel Core8051](#)
- [Cast C8051 Core](#)
- [Cast D80530 Core](#)
- [Cast R8051 Core,](#)
- [Cast R80515 Core](#)
- [Dolphin Flip8051 Breeze](#)
- [Dolphin Flip8051 Cyclone](#)
- [Dolphin Flip8051 Thunder](#)
- [Dolphin Flip8051 Wind.](#)

Added ULINK debugging and Flash programming support for the new STMicroelectronics μ PSD3300/3400 series of devices. Also added Flash programming support for the μ PSD3200 series of devices. Program examples and an [Application Note](#) are provided in the \KEIL\C51\EXAMPLES\ST μ PSD folder.

Corrected a problem with menu and shortcut configuration on Windows NT machines.

Corrected simulation and target display problems with the Philips LPC900 series (i.e. the DIVM factor).

40. C51 Version 7.07a

Release date: 1st August, 2003.

uVision IDE

Added a new dialog for project component management under Project - Components, Environment and Books. This dialog allows you to change the order of project targets and file groups.

Added simulation for the following devices:

- Atmel AT89C5131 (except USB)
- Atmel AT89C5132 (except A/D Converter, USB, Audio Interface, MMC Controller, and IDE/ATAPI Interface)
- Cygnal C8051F300
- Cygnal C8051F301
- Cygnal C8051F302
- Cygnal C8051F303
- Cygnal C8051F304
- Cygnal C8051F305
- Cygnal C8051F310
- Cygnal C8051F311
- Cygnal C8051F320
- Cygnal C8051F321
- Cygnal C8051F330
- Cygnal C8051F331
- TI MSC1210 (I#C Simulation)
- TI MSC1211
- TI MSC1212

Corrected the following simulation problems:

- Analog Devices ADuC831: The internal RC clock (32768 Hz) is used for the watchdog timer. This was incorrectly documented in the first datasheets.
- Dallas Semiconductor Devices: The Watchdog EWT reset is now performed only on power-up reset and not on every reset.
- Winbond Devices: The Watchdog EWT reset is now performed only on power-up reset and not on every reset.

Corrected problems viewing local variables in projects linked with the LX51 Extended Linker/Locator.

Cx51 Compiler

Corrected problems for Philips MX when far/generic pointer assignments are reused in the subsequent statements. For example:

```
unsigned char far * far pbMemory;  
unsigned long volatile ulAddress;  
  
pbMemory += 1L;          // uses ?C?PSTPTR to save the result  
ulAddress = pbMemory;    // uses wrong result of the previous  
                          // ?C?PSTPTR library function
```

C51 Compiler

Corrected problems in Dallas Contiguous Mode with `*(ptr++)` type constructs and far pointer initialization at file level.

Corrected problems with SRC output on extended 8051 platforms like Dallas Contiguous Mode and Philips 51MX.

Added the ability to locate far memory variables in HDATA memory using the `_at_` keyword.

Lx51 Linker

Improved memory allocation in linker code packing to reduce the size of segment gaps when the SEGMENTS directive is used with assembler segments.

Ax51 Macro Assembler

Added the DEFINE command line directive. This directive allows you to supply C preprocessor on the command line. The syntax is identical to that of the Cx51 Compiler.

Monitor-390

Corrected problems with xdata memory updates and added support for Dallas 400 and Dallas 5240 devices.

ISD51 In-System Debugger - Version 2

Corrected a problem that caused serial break to fail when configured for non-Flash breakpoints. Added example for TI MSC121x devices in the `\KEIL\C51\EXAMPLES\TI MSC121X` folder.

41. C51 Version 7.06a

Release date: 17th May, 2003.

ISD51 In-System Debugger - Version 2

Corrected a problem that caused serial break to fail on ISD51 when configured for non-Flash breakpoints.

Flash Monitor-51 Version 4

Added configurations for Atmel AT89C51RD2 and AT89C51SND1.

Corrected a problem that caused Break on Serial Interrupt to fail when the monitor was generated using older versions of the tools.

uVision IDE/Simulator

Added simulation for the following devices:

- Atmel AT89C51RD2
- Atmel AT89C51ED2
- Atmel AT89C51ID2
- Dallas Semiconductor 80C530
- Philips 8xC652
- Philips 8xC654

Corrected simulation problems with the CCU Timer on the Philips P89LPC932.

Added context menu commands in the Source Window and Disassembly Window for Set Program Counter, Show Disassembly, and Show Source Code.

Cx51 Compiler

Corrected a problem that caused far pointer comparisons to NULL to fail when the statement immediately following re-used the same pointer.

Corrected a problem with the NOINTPROMOTE directive that caused the compiler to generate incorrect code. This problem was introduced in C51 V7.04 in an effort to correct another integer promotion problem.

42. C51 Version 7.05

Release date: 31st March, 2003.

ISD51 In-System Debugger - Version 2

Added several new features including:

- Real-Time Flash Breakpoints using In-System Application Programming (IAP),
- User I/O via serial debugging interface,
- Address range support in the memory verify function.

Flash Monitor-51 - Version 4

A new variant of Monitor-51 is included in the PK51 Professional Developers Kit. The new Monitor runs on unmodified Flash Devices that provide IAP programming. It requires no von-Neumann memory and can run from the on-chip resources of standard 8051 Flash Devices. The Flash Monitor includes Flash download and real-time breakpoint support. Currently the Monitor is pre-configured for the Atmel T89C51CC01, T89C51RC2, and T89C51RD2 but it can easily be configured for other devices.

uVision IDE/Simulator

Added simulation support for the Atmel T89C51CC02, T8xC5115, AT89C1051, AT89C1051U, AT89C2051, and AT89C4051 devices.

Added simulation support for the Philips P8xC51Rx2, P8xC51RB2H/RC2H/RD2H, and P8xC3xX2 devices.

Corrected a simulation problem with the AT89S8252 EEPROM and Dual DPTR access.

Corrected a simulation problem with the ADuC832 ADC DMA Stop.

Corrected a problem with local variables not displaying in the watch window - locals tab.

LX51 Extended Linker

Corrected a potential problem with linker code packing on optimize level 10 and 11.

C51 Compiler

Corrected erroneous combining of identical end sequences of `while (1)` loops.

43. C51 Version 7.04

Release date: 7th February, 2003.

Philips MX Device Support

Changed INIT_MX.A51. Now, initialization is enabled for far variables by default. far variables may now be absolutely located with `_at_`.

MON390 Monitor for Dallas Contiguous Mode

Corrected problems with breakpoints above 64KB code. Corrected a problem with single-stepping in switch/case statements.

uVision IDE

Added call stack display and step out command for classic 8051 devices. Added [EPM900](#) Emulator/Programmer support for Philips 89LPC9xx Devices.

C51 Compiler

Corrected integer promotion problems with combined pointer and char arithmetic. For example:

```
int xdata *Test (int xdata * adr, unsigned char a, unsigned char b) {  
    return adr+(a+b);    // did not promote 'a+b' to int  
}
```

A51 Assembler, AX51 Assembler

Added support for C-style bitwise operators (`|` (OR), `&` (AND), `~` (NOT)) to the A51 Assembler and AX51 assembler. These operators are useful for common C and assembler header files that use `#define` statements.

44. C51 Version 7.03a

Release date: 9th December, 2002.

RTX51 Tiny2

Corrected os_wait problems on K_IVL, K_TMO+K_SIG events. Refer to \C51\RTX51TINY2\README.TXT for details.

uVision IDE

Added Cygnal 80C51F02x device simulation.

C51 Compiler

Corrected incorrect warnings on enum mismatches.

Lx51 Linker

Corrected fixup error messages on Dallas 390 target.

45. C51 Version 7.02b

Release date: 7th November, 2002.

Lx51 Linker

Corrected problems with linker code packing and code banking and a potential problem with the bank switch table location in banking mode 4.

Added the SPEEDOVL directive to makes LX51 and BL51 compatible for data overlaying. Detailed information on SPEEDOVL is available in the Assembler/Utilities User's Guide, Chapter 9, Control Summary.

BL51 Linker, LX51 Linker, Libraries

Added support for Mentor M8051EW Memory Extension that provides access to 1MB ROM and 1MB RAM. The IBANKING directive supports the on-chip banking hardware on M8051EW-based devices and is available in both the BL51 Linker and the LX51 Linker. Additionally, LX51 may be configured with the L51IBANK.A51 file that supports a 64KB bank for constants and 16 x 64KB banks for far variables. Refer to the \C51\EXAMPLES\M8051EW folder for example code and additional information.

C51 Compiler, CX51 Compiler

Corrected a potential problem with generic and far pointer comparisons to a NULL pointer constant.

Enhanced warning messages for enum and memory-typed pointer assignments.

Monitor for Dallas Contiguous Mode

Released MON390 which provides a target monitor for the Dallas Contiguous Mode. Detailed information, pre-configured Monitor versions, and example programs may be found in the \C51\MON390 folder.

uVision IDE

Added simulation for several new devices (Atmel 89C51lx and the Cygnal 80C51F02x). The Cygnal 80C51F02x devices are currently in beta status.

Added Flash menu to μ Vision2. This menu provides a direct interface to external Flash programming tools like Philips FlashMagic. Flash programming commands are configured under Options for Target - Utilities.

RTX51 Tiny Version 2

Released RTX51 Tiny Version 2. This release contains several new enhancements like code banking support and cooperative task switching.

46. C51 Version 7.01

Release date: 1st July, 2002.

C51 Compiler

Added support for Extended Call Return Mode (ECRM) available in the new Philips 51MX devices. This mode is configured in START_MX.A51. It enhances the code density of the ROM(HUGE) memory model. This optimization requires that [Linker Code Packing](#) is enabled. Once enabled, ACALL, LCALL, and ECALL instructions are optimized.

Corrected minor problems in the ROM(HUGE) memory model.

Corrected problems in L51_BANK.A51 with regards to variable code banking on classic 8051 devices that used standard banking hardware.

Corrected a syntax problem (that was introduced in Version 7.00) in the setjmp.h header file.

uVision IDE

Added several new devices to the µVision2 device database.

Added peripheral simulation for the new Philips 89LPC932.

Added peripheral simulation for the second UART in Winbond devices.

Added peripheral simulation for the four priority levels in the new version of the Philips 8xC552 device.

Lx51 Linker

Validated LX51 [Linker Code Packing](#) for Philips 51MX and Dallas 390/400 devices.

BETA RELEASE

Released RTX51 Tiny Version 2 BETA with the following new features and enhancements:

- Code Banking Support
- Explicit Task Switch Function
- RUN Status Flag
- CPU IDLE Mode Support
- Hooks for Adding User Code to the RTX51 Tiny Hardware Timer Interrupt
- Improved Handling for Interval Events
- Reduced Code and Data Size
- Improved Performance

Released MON390 BETA which provides a target monitor for the Dallas Contiguous Mode. Detailed information, pre-configured Monitor versions, and example programs may be found in the \c51\mon390 folder.

47. C51 Version 7.00

Release date: 22nd April, 2002.

C51 Compiler

Added the ROM(HUGE) directive which provides support for the Philips 51MX Linear Programming Model. Select this option in μ Vision2 using Options for Target-Code Rom Size: Huge: 8MB program. More information is available in [Application Note 160](#): Programming the Philips 51MX Architecture with the Keil PK51.

Added the ability to perform 24-bit arithmetic calculations using far pointers. For more information, refer to [Application Note 160](#): Programming the Philips 51MX Architecture with the Keil PK51.

ISD51

Released [ISD51 \(In-System Debugger\)](#): a new target debugger that may be linked to user applications. Refer to `\c51\ISD51` for more information.

Added two new optimizations to the C51 Compiler that reduce program code size. In μ Vision2, enable these optimization in Options for Target - C51: Code Optimization: Level.

Lx51 Linker

Added a new LX51 Linker-Level Optimization called [Linker Code Packing](#). This optimization analyzes and reduces total program size. In μ Vision2, enable this optimization in Options for Target - C51: Code Optimization: Linker Code Packing. This optimization is available for all projects even those that use code banking. Note that this optimization is still a BETA RELEASE for the Philips 51MX and Dallas 390/400 devices.

Added Linker [Disassembly Output File](#). This output file contains the complete disassembly of your application complete with intermixed high-level source and all addresses. In μ Vision2, enable this option in Options for Target - Listing - Linker Code Listing.

48. C51 Version 6.23

Release date: 15th February, 2002.

C51 Compiler

Corrected problems with register optimizations in while loops.

Corrected problems implicitly casting types in ternary statements.

Enhanced performance of the run-time library and pointer operations for the Dallas Semiconductor 80C390 Contiguous Mode.

Added ability to locate XDATA and CODE in regions other than 00:xxxx for the Dallas Semiconductor 80C390.

Corrected various problems with initializations with Lx51 and bit objects.

A51 Assembler

Corrected problems [synchronizing MPL Macros](#) while debugging.

OHx51 Object File Converter

Added the MERGE32K directive which generates merged HEX files for hardware with 32K common areas. Refer to the Ax51 User's Guide, Chapter 9, Bank Switch Configuration for more information. In μ Vision2, select this option in Options for Target - Output - Merge32K HEX File.

Corrected problems with Lx51 HEX file generation.

μ Vision IDE

Added simulation support for Analog Devices MicroConverters.

BETA RELEASE

ISD51 In-System Debugger is a new target debugger that may be linked to user applications. Refer to \c51\ISD51 for more information.

Added a new LX51 Linker-Level Optimization called [Linker Code Packing](#). This optimization analyzes and reduces total program size. In μ Vision2, enable this optimization in Options for Target - C51: Code Optimization: Linker Code Packing.

Added two new optimizations to the C51 Compiler that reduce program code size. In μ Vision2, enable these optimization in Options for Target - C51: Code Optimization: Level.

Added Linker [Disassembly Output File](#). This output file contains the complete disassembly of your application complete with intermixed high-level source and all addresses. In μ Vision2, enable this option in Options for Target - Listing - Linker Code Listing.

49. C51 Version 6.22

Release date: 15th December, 2001.

C51 Compiler

Corrected several problems that were introduced with Dynamic Register Allocation in Version 6.21.

Added new examples for the const far memory type.

Added support for the extended stack in the Analog Devices MicroConverters.

Lx51 Linker

Added examples for far const memory with classic 8051 devices in `\C51\EXAMPLES\FARMEMORY`
`\1MB CONSTANTS ON CLASSIC 8051`. These examples show how to use memory banking with text constants.

50. C51 Version 6.21

Release date: 15th November, 2001.

C51 Compiler

Corrected several minor problems.

Added the MODDA directive and library support for the Dallas 390 Math Accelerator.

Added [dynamic register allocation](#) optimization which reduces program size and data usage.

Added [switch/if path combination](#) optimization.

Added optimization for [long 0 comparisons](#).

Corrected several optimizer problems that were introduced in C51 V6.20.

BL51 Linker

Corrected several minor problems.

uVision IDE

Added simulation support for the Dallas Semiconductor 80C390 peripherals.

51. C51 Version 6.20

Release date: 9th January, 2001.

C51 Compiler

Enhanced the L51_BANK.A51 file to support even larger code banking programs (up to 4MB).

Added enhanced optimization for register variables.

Added variable banking support for classic 8051 devices.

uVision IDE

Added debug dialogs for classic 8051 devices.

Added four 64KB user memory areas (S:, T:, U:, and V:) that may be used when debugging.

Added functions or for special simulation capabilities (EEPROM, I²C communication, and so on).

Improved the Version Control (SVCS) Connection and corrected several problems with environment variables.

Added several new items to the Help Menu.

Added several project management enhancements.

Added numerous chips to the Device Database.

Added simulation support for the on-chip peripherals of the following devices (complete information is available in the μ Vision Help Menu - Simulated Peripherals item.):

- Infineon C504.
- Infineon C505C.
- Infineon C508.
- Infineon C515A.

Added new debugging dialogs for MON51.

52. C51 Version 6.14

Release date: 5th January, 2001.

C51 Compiler

Added examples to demonstrate the Dallas 390 contiguous mode (`\c51\EXAMPLES\DALLAS 390`).

Added examples to demonstrate the Philips 51MX architecture 16MB support (`\c51\EXAMPLES\PHILIPS 80C51MX`).

Added far memory (above 64K) support using the `\c51\LIB\XBANKING.A51` configuration file.

Added far memory examples for the 16MB memory space of the Analog Devices ADuC812 and for the Atmel 89C51RD E2PROM area (`\c51\EXAMPLES\FARMEMORY`).

Added macros for absolute far memory access in `ABSACC.H`.

Added the `INCDIR` directive where you may specify include paths.

Ax51 Assembler

Added the `INCDIR` directive where you may specify include paths.

uVision IDE

Added simulation support for the on-chip peripherals of the following devices:

- Infineon C509.
- Infineon C517A.
- Infineon C515C.

53. C51 Version 6.12

Release date: 3rd January, 2001.

C51 Compiler

Removed the 256-symbol limit from OMF51 object files.

Extended the length of C variable names to 256 characters.

uVision IDE

Added simulation support for the on-chip peripherals of the following devices:

- Atmel WM T87C5111.
- Atmel WM T87C5112.

54. C51 Version 6.10a

Release date: 1th January, 2001.

C51 Compiler

Finalized support for the Philips 80C51MX.

Finalized support for the Dallas Semiconductor 80C390.

Added banking mode 4 to L51_BANK.A51 for user-provided bank switching macros.

55. C51 Version 6.02

Release date: 8th January, 1999.

C51 Compiler

Corrected several minor problems with Optimizer Level 9.

uVision IDE

Added simulation support for the on-chip peripherals of the following devices:

- Analog Devices ADuC812.
- Philips LPC Devices.

Added simulation support for multiple DPTR registers.

56. C51 Version 6.01

Release date: 7th January, 1999.

C51 Compiler

Added the RET_ISTK, RET_PSTK, and RET_XSTK directives which unload the on-chip stack and use the reentrant stack for storing return addresses. Refer to [The RET_ISTK Directive](#), [The RET_PSTK Directive](#), or [The RET_XSTK Directive](#) for more information.

Added ANSI library routines modf, strtod, strtol, and strtoul. Refer to the on-line compiler manual for more information.

BL51 Linker

Modified the CODE and XDATA directives to allow address ranges. For example:

```
BL51 my_prog.obj CODE(0x0000-0x3FFF, 0x8000-0xFFFF)
```

Modified segment control directives to allow wildcards in segment names. For example:

```
BL51 my_file.obj (CODE (?PR?*my_file (0x100))
```

Locates all program code segments in the module my_file to address 0x100 and up.

uVision2 IDE

Added debugger support for MON51.

Added simulation support for almost all 40-pin DIP devices (8051FC, RD, RD+, 8052, and so on).

Added support for syntax coloring in assembler code.

Added item in the context menu to insert the CPU register definition file.

Added context-sensitive help for library routines and error messages. To get help for a library routine, position the cursor on the function and press the F1 key. To get help for an error or warning message, select the message and press the F1 key.

57. C51 Version 6.00

Release date: 1st January, 1999.

C51 Compiler

Added 3 new optimizer levels which shrink program size up to 15%:

- Optimize Level 7 (Extended Access Optimization) uses DPTR for register variables. Pointer and array accesses have been optimized for both speed and code size.
- Optimize Level 8 (Reuse of Common Entry Code) moves common function entry code to the beginning of a function which saves code memory. Optimize (8) is the new default optimization level for C51 Version 6.xx.
- Optimize Level 9 (Common Block Subroutines) detects and packs multiple-instruction sequences into subroutines. This optimization is most efficient on large source files.

Added specific header file support for almost all devices.

Added configuration file (\C51\LIB\CONF151.A51) for the Intel 151.

Updated the enum type to automatically adjust to 8 or 16 bits.

Added dual data pointer support for Atmel devices (AT89S8252). Use the MODA2 directive to enable dual data pointer support. Use the NOMODA2 directive to disable support.

Added dual data pointer support for Philips devices. Use the MODP2 directive to enable dual data pointer support. Use the NOMODP2 directive to disable support.

Added dual data pointer support for Temic devices. Use the MODP2 directive to enable dual data pointer support. Use the NOMODP2 directive to disable support.

C51 now ensures that register bank 0 is selected for interrupts declared without the using attribute. The instruction MOV PSW, #0 is added to these routines. Previously, you were required to add the using 0 attribute to high-priority interrupts when low-priority interrupts used a different register bank. This was the case for RTX51 Full and RTX51 Tiny applications. If your application uses only register bank 0, you may use the ONEREBANK directive to specify that the C51 compiler does not generate the additional MOV PSW, #0 instruction.

A51 Assembler

Added C preprocessor support that expands text before the source file is assembled. Directives like #if, #else, #endif, and #include are supported in assembly source code (refer to the C51 User's Guide, Chapter 4). The #include file path is obtained from the C51INC environment variable.

Added the following predefined Macros:

- __FILE__: Name of the file being assembled.
- __LINE__: Current line number in the file being assembled.
- __TIME__: Time when the assembly was started.

- `__DATE__`: Date when the assembly was started.
- `__STDC__`: Defined to 1.
- `__A51__`: Version number of the A51 Assembler (for example 600 for V6.00).
- `__KEIL__`: Defined to 1.

Added support for C-style `sfr` and `sbit` declarations. The A51 Assembler now accepts standard C-style register definition files. This allows you to use the same header files for your C and assembler source files. The following `sfr` and `sbit` declarations were added:

```
sfr P0 = 0x80;
sbit P0_1 = P0^1;
```

Added error output using the `__ERROR__` directive. For example:

```
IF CVAR1LEN > 10
  __ERROR__ "CVAR1 LEN EXCEEDS 10 BYTES"
ENDIF
```

Or using the C-style preprocessor.

```
#ifdef TESTVERS && RELEASE
#error TESTVERS GENERATED IN RELEASE MODE
#endif
```

Added the `INCDIR` (abbreviation ID) directive where you may specify the paths to assembler include files. You may specify one or more paths to search when a `$INCLUDE` directive is processed. The search order for `$INCLUDE` is:

- Current directory (typically, the folder of the μ Vision2 project file).
- Paths specified with `$INCDIR`.
- Path derived from the bin folder using `..\ASM (\C51\ASM)`. For example:

```
$INCDIR (C:\C51\ASM) A51 STARTUP.A51
INCDIR (C:\C51\INC,C:\MYDIR)
```

The search order for `#include` is identical to that used by the C51 Compiler.

BL51 Linker

Added the `DISABLEWARNING` directive (abbreviation DW) which allows you to selectively disable linker warning messages. For example (disables warnings 1 and 5):

```
BL51 myfile.obj DISABLEWARNING (1,5)
```

BL51 now sorts and locates segments according to their length. This ensures fewer gaps in memory. Use the `NOSORTSIZE` directive (abbreviation NOSO) to disable this feature.

Added the SPEEDOVL directive (abbreviation SP) which causes the linker to ignore references to constant segments that start with ?CO?. This speeds up the overlay process but may result in a lack of warnings with regard to constant segments. This could lead to problems if you use pointers to functions and do not manually specify the call tree references to the linker. Refer to [SPEEDOVL Directive](#) and [Application Note 129: Function Pointers in C51](#) for more details. This directive may be useful for applications with complex pointer to function tables.

Added the RECURSIONS directive (abbreviation RC) which allows you to specify the maximum number of recursive calls allowed before the linker aborts. The default number of recursions allowed is 10. A recursive call generates the following linker warning:

```
* WARNING L13: RECURSIVE CALL TO SEGMENT
```

When the maximum number of recursions is exceeded, the linker responds with the following error:

```
FATAL ERROR 232: APPLICATION CONTAINS TOO MANY RECURSIONS.
```

To use this directive, enter the following on the linker command line or in the Misc box in μ Vision2:

```
BL51 test.obj RECURSIONS (100)
```



The linker may run a long time to detect all recursions and remove the references. Unless you have specific reasons to change this setting, you should leave it at the default level of 10.

Added the NOAJMP directive (abbreviation NOAJ) which disables use of the AJMP instruction in the inter-bank jump table in code banking programs. This option is required for 8051 derivatives which do not support the AJMP instruction.

BL51 Linker - Code Banking

Added the NOINDIRECTCALL directive (abbreviation NOIC) which specifies that function pointers do not access functions outside the current code bank (in code banking programs). By default, in code banking programs the BL51 Linker inserts inter-bank calls into the call table for functions called through a function pointer. If your application uses function pointers and if you can ensure that indirect calls never cross a code bank you may use the NOINDIRECTCALL directive to save space in the call table. Refer to [The Code Banking Mechanism](#) for more information about the scheme used by the BL51 Linker for code banking.

Added the NOJMPTAB directive (abbreviation NOJT) which specifies that the call tree is not created for code banking applications. This feature is provided for developers who will create their own code banking scheme. This directive modifies the behavior of the BL51 Linker as follows: - The L51_BANK.A51 code banking logic file is not required. - The jump and call instructions are not modified for banked functions. - No warnings are generated if a jump or call is made to another code bank. If you use this directive you must ensure that the proper bank is selected before a call is performed. The BL51 Linker no longer selects the code bank.

