EmbeddedCraft
crafting of intelligent systems

IAR WORKBENCH FOR 8051 PART1

CONTENTS

Embedded Development Using IAR EW8051

- IDE: For complete project management
- C/C++ Compiler : Compiler with MISRA C Support
  (MISRA: Motor Industry Software Reliability Association)
- Assembler : Assembler for 8051
- XLINK : Linker and Locator
- XAR : Library Builder for making libraries
- Simulator : CPU simulator and macro

File Structure

• .c	C Language program file
• .cpp	C++ Language file
• .s51	Assembly source file
• .ewp	Embedded work bench project file
• .d51	Output file with debug information (use for debugging purpose)
• .a51	Output file without debug information (cab be loaded in flash)
• .r51	Library module file

Steps for creating a project

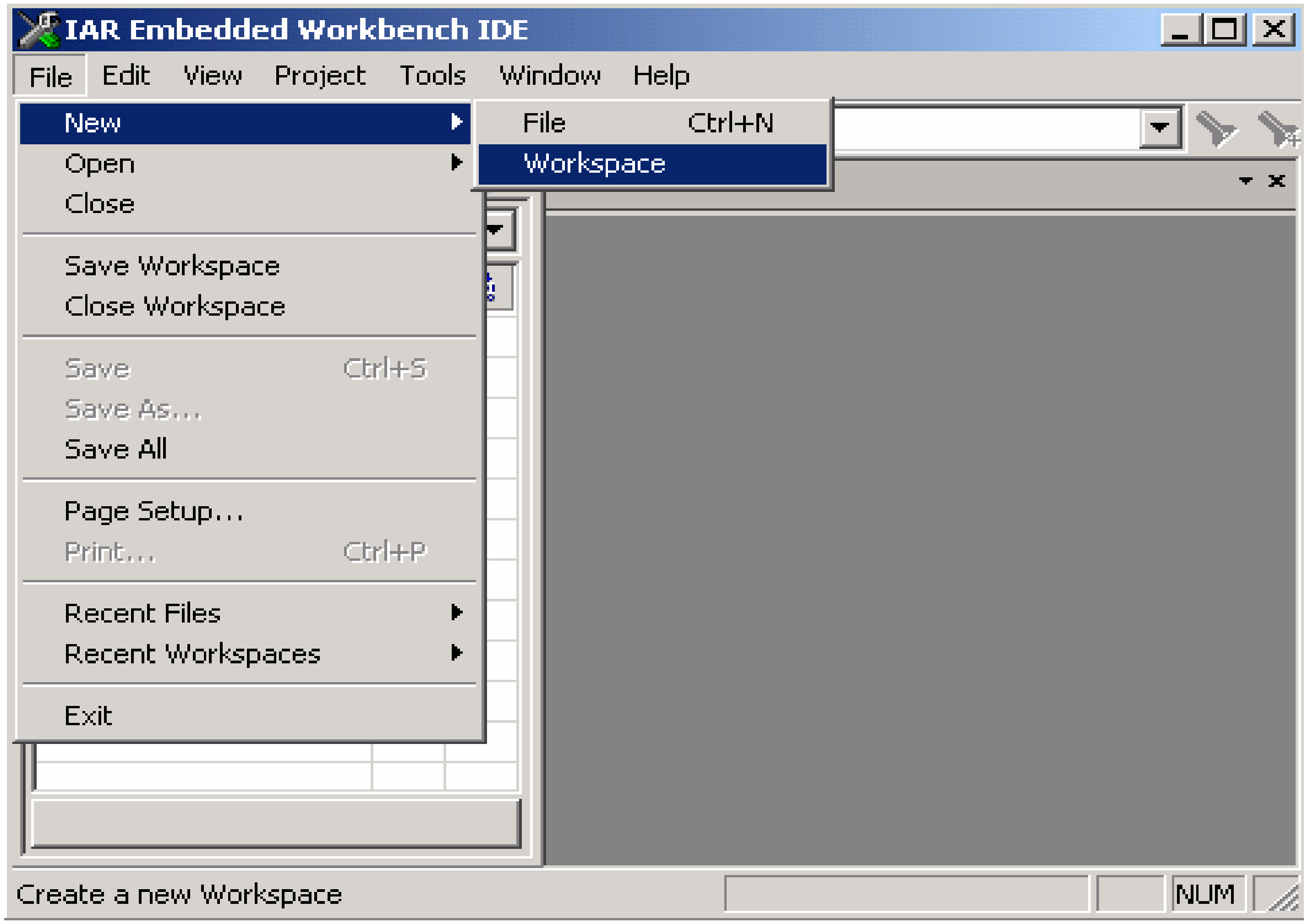
- Create a workspace file
- Create a project file
- Add source file
- Setting options for CPU and target
- Compiling the files
- Linking files by make

Making First Project

- Source code is to display fibonacci series
- Location of source file
  Installation directory > 8051 > tutor

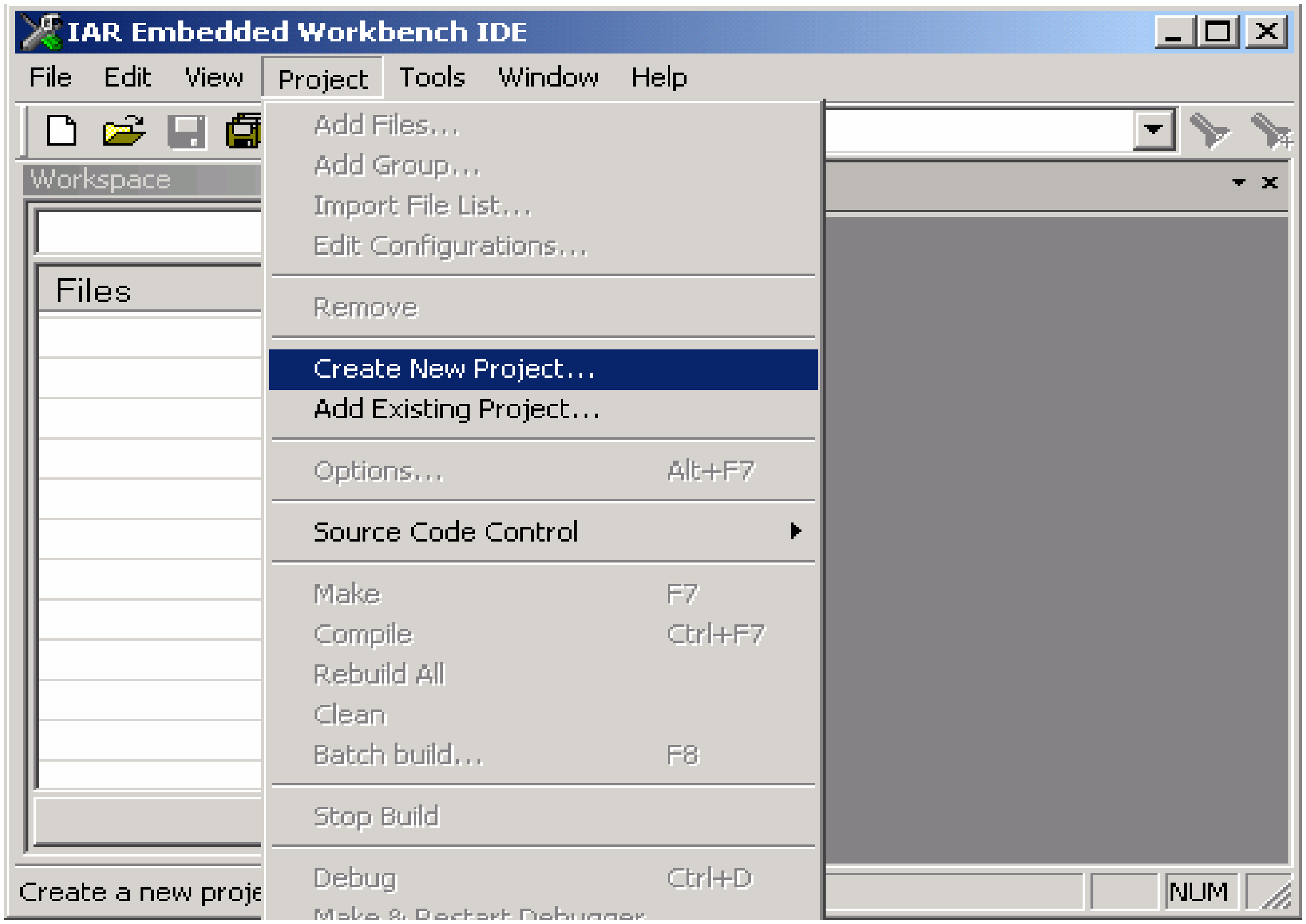
Creating a workspace

EmbeddedCraft
crafting of intelligent systems



Create a new Workspace

Create a new project


IAR Embedded Workbench IDE

File Edit View Project Tools Window Help

Workspace

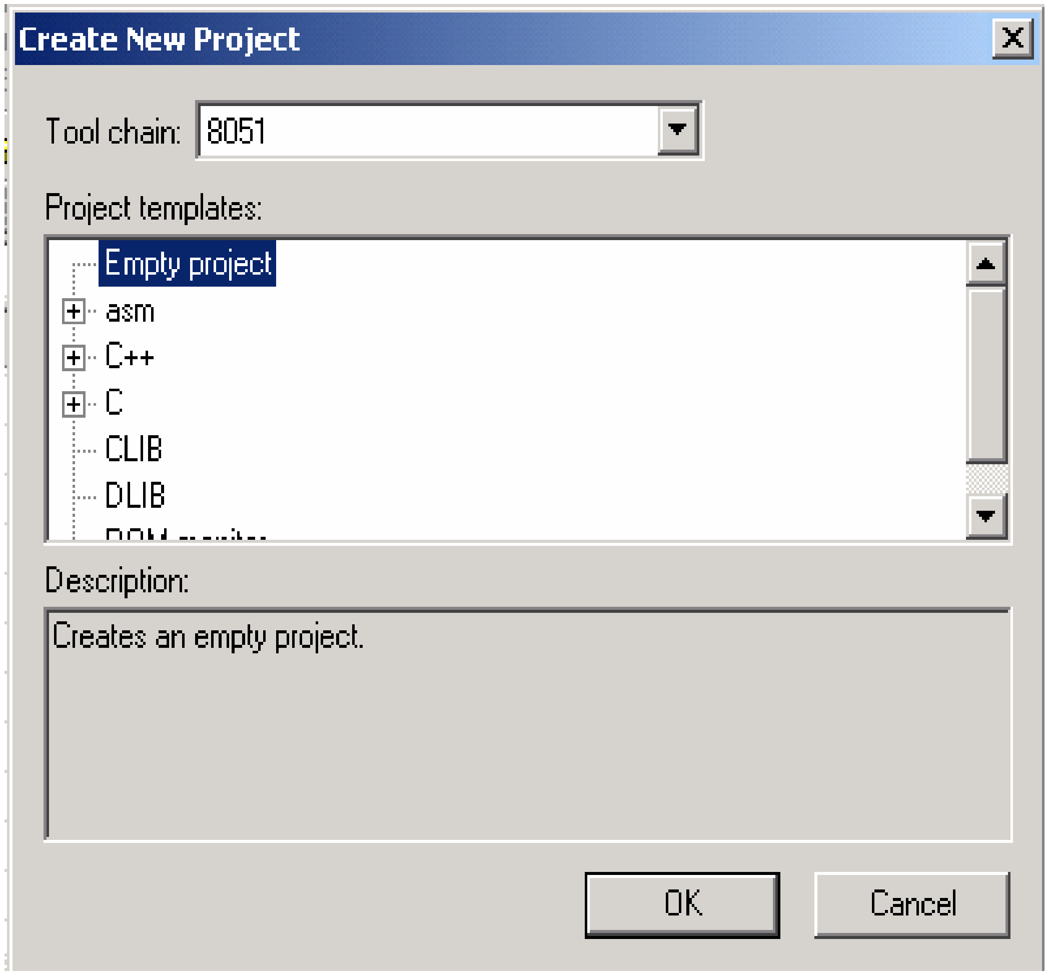
Files

Create a new proje

NUM

Select project template

EmbeddedCraft
crafting of intelligent systems


Create New Project

Tool chain: 8051

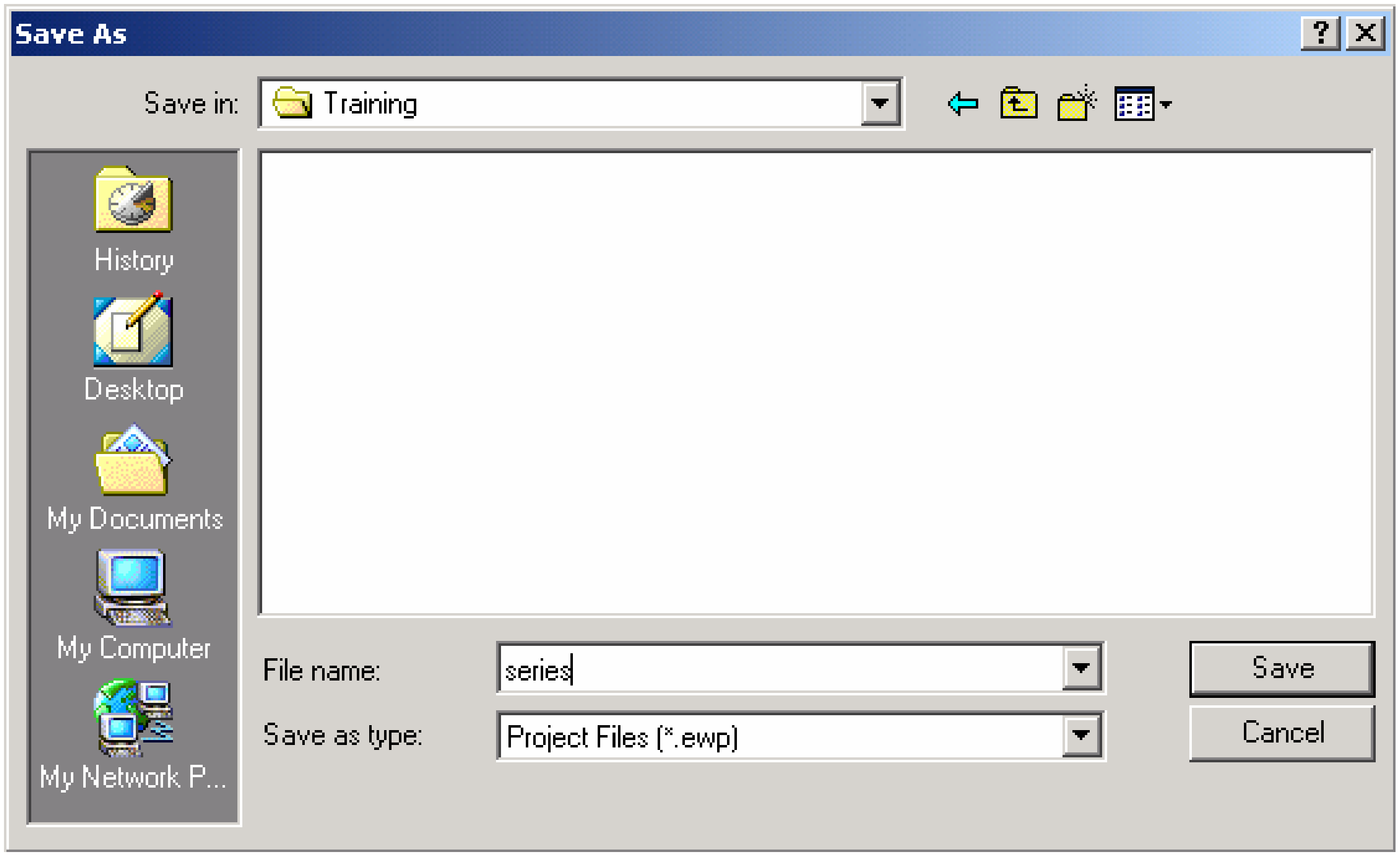
Project templates:

- Empty project
- asm
- C++
- C
- CLIB
- DLIB

Description:

Creates an empty project.

Save project file

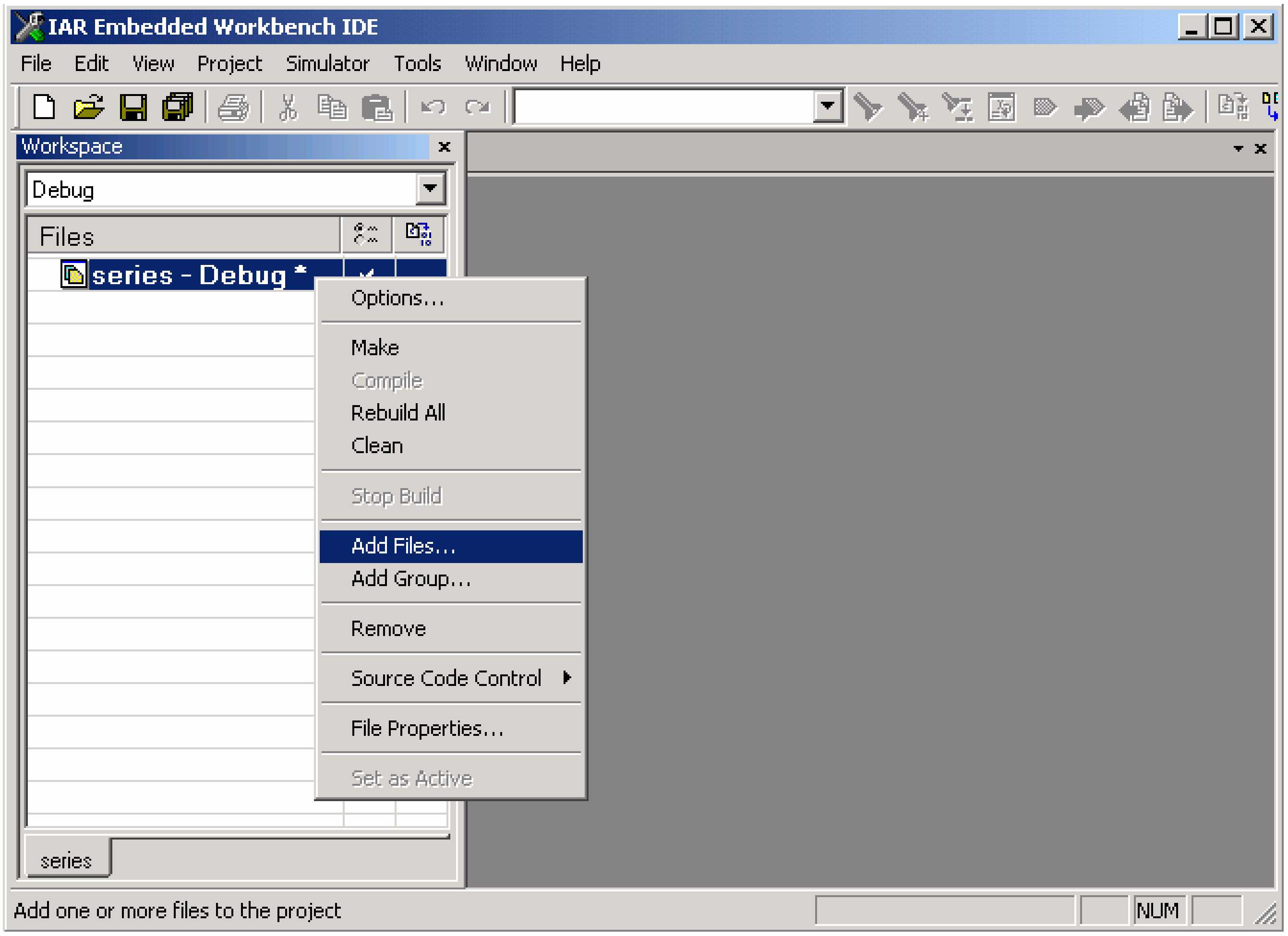


Save in: Training

File name: series

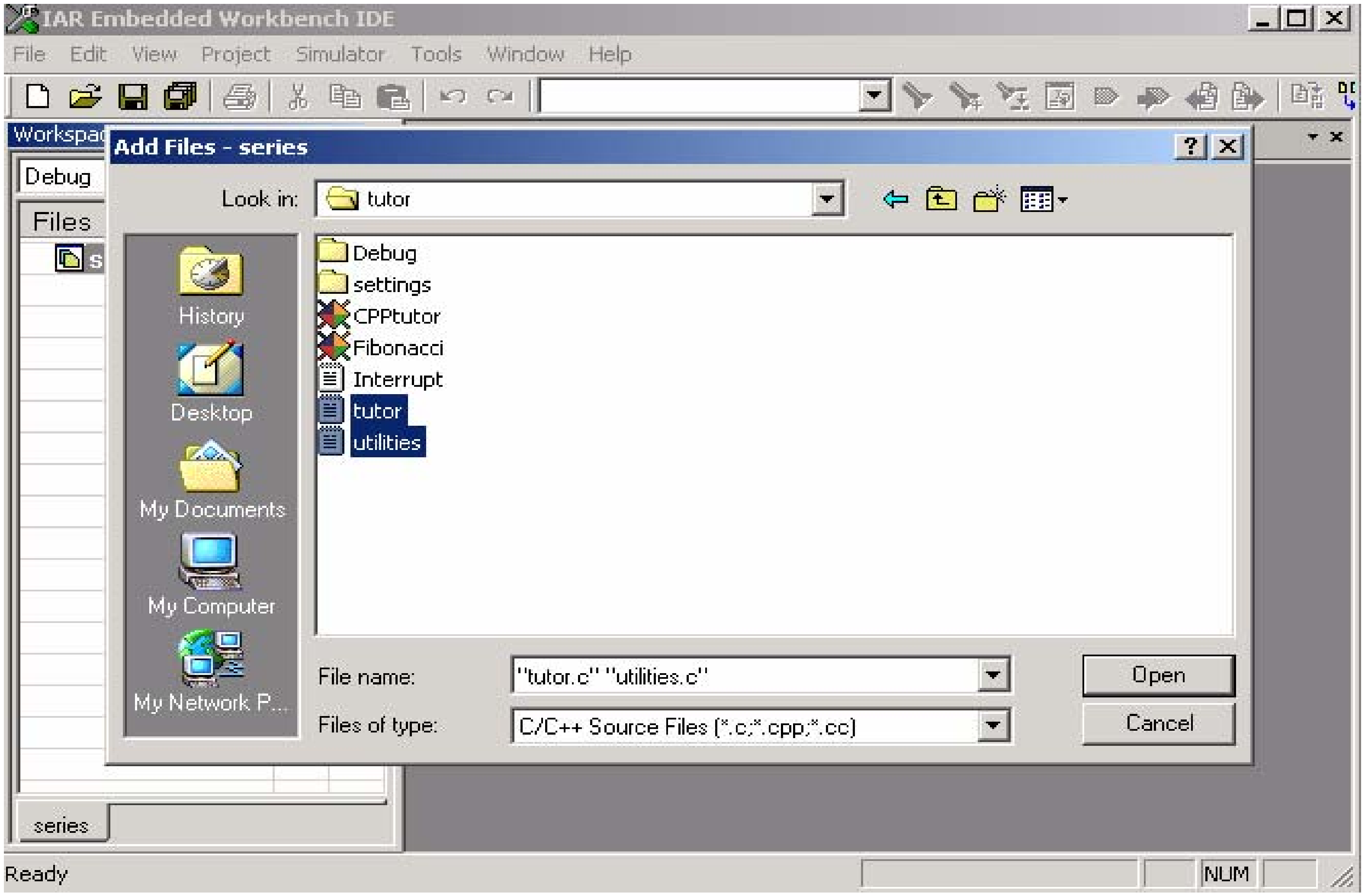
Save as type: Project Files (*.ewp)

Adding source files



Add one or more files to the project

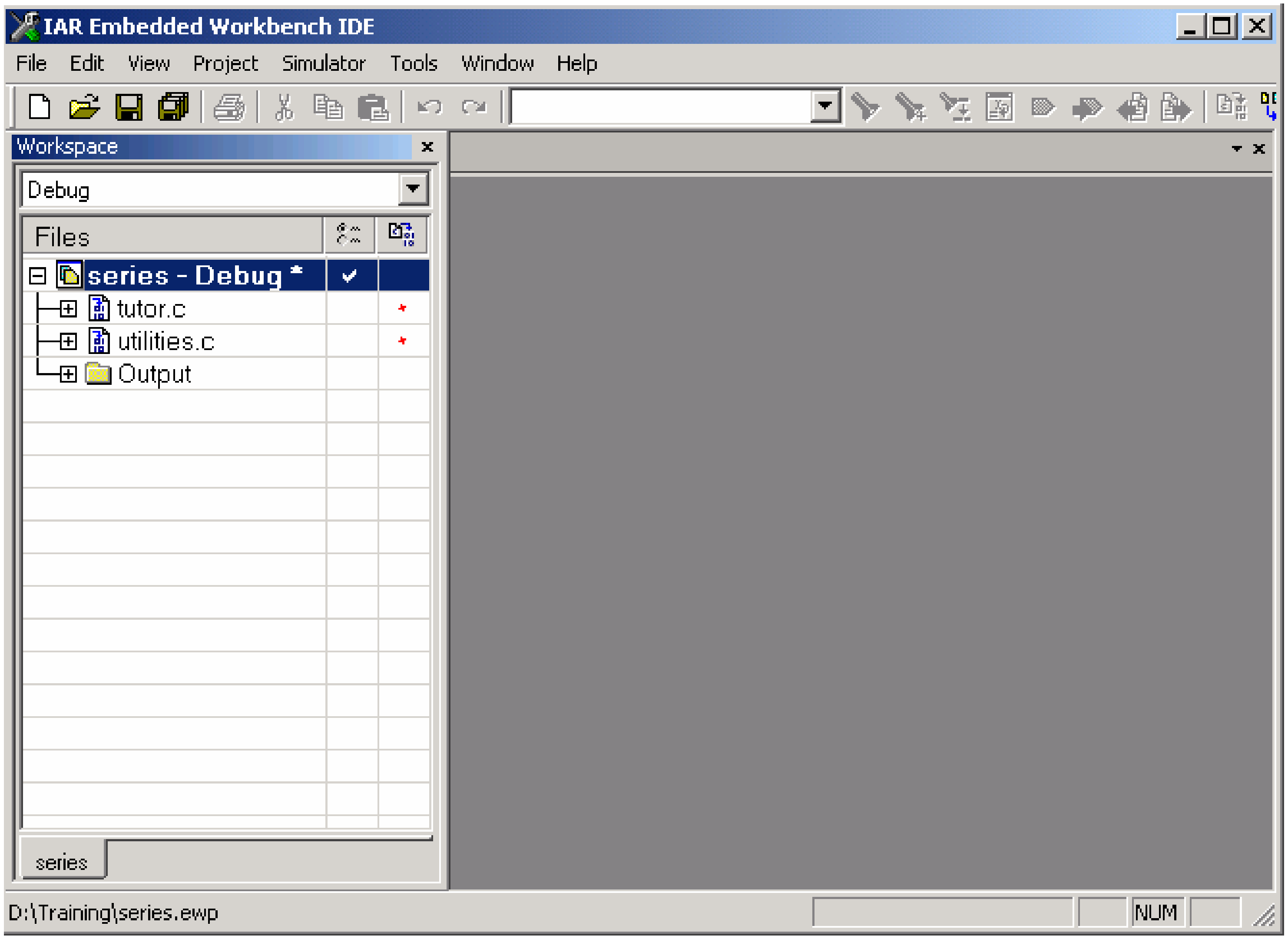
Adding source files



File name: "tutor.c" "utilities.c"

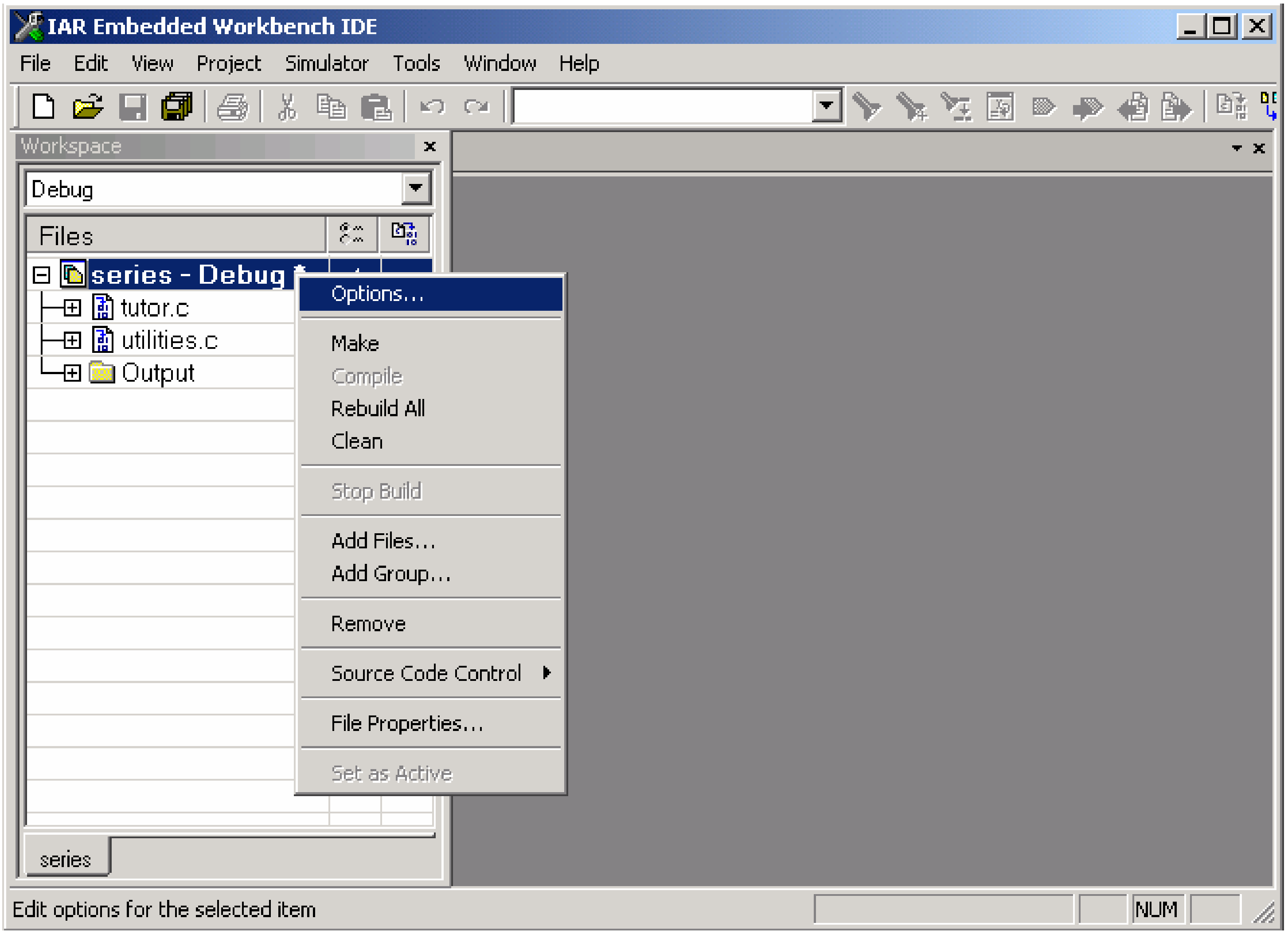
Files of type: C/C++ Source Files (*.c;*.cpp;*.cc)

Tutor.c and utilities.c added to project



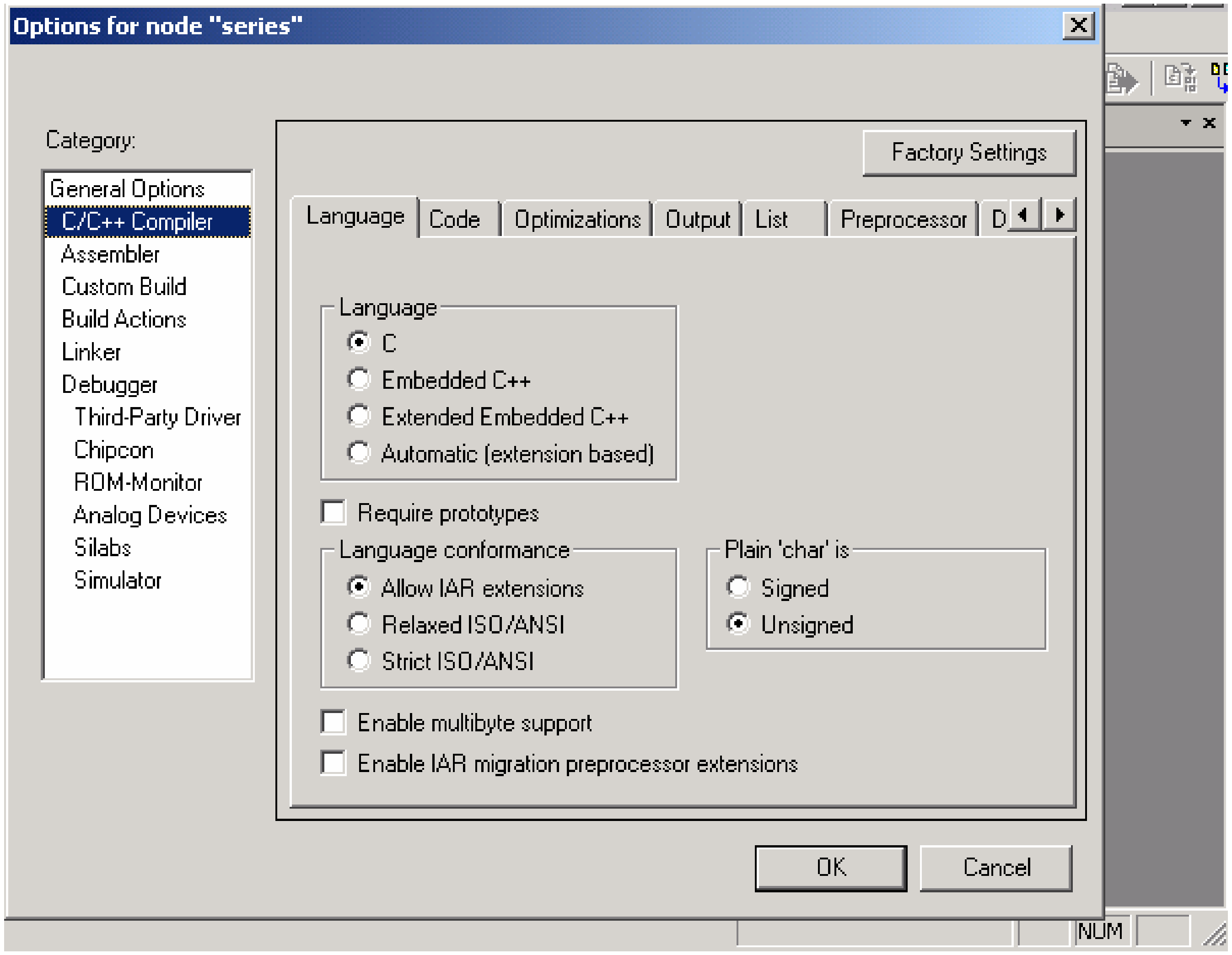
Project options 1/3

EmbeddedCraft
crafting of intelligent systems



Project options 2/3

EmbeddedCraft
crafting of intelligent systems



Options for node "series"

Category:

- General Options
- C/C++ Compiler
- Assembler
- Custom Build
- Build Actions
- Linker
- Debugger
  - Third-Party Driver
  - Chipcon
  - ROM-Monitor
  - Analog Devices
  - Silabs
  - Simulator

Factory Settings

Language | Code | Optimizations | Output | List | Preprocessor | D

Language
- [✓] C
- [ ] Embedded C++
- [ ] Extended Embedded C++
- [ ] Automatic (extension based)

[ ] Require prototypes

Language conformance
- [✓] Allow IAR extensions
- [ ] Relaxed ISO/ANSI
- [ ] Strict ISO/ANSI

Plain 'char' is
- [ ] Signed
- [✓] Unsigned

[ ] Enable multibyte support

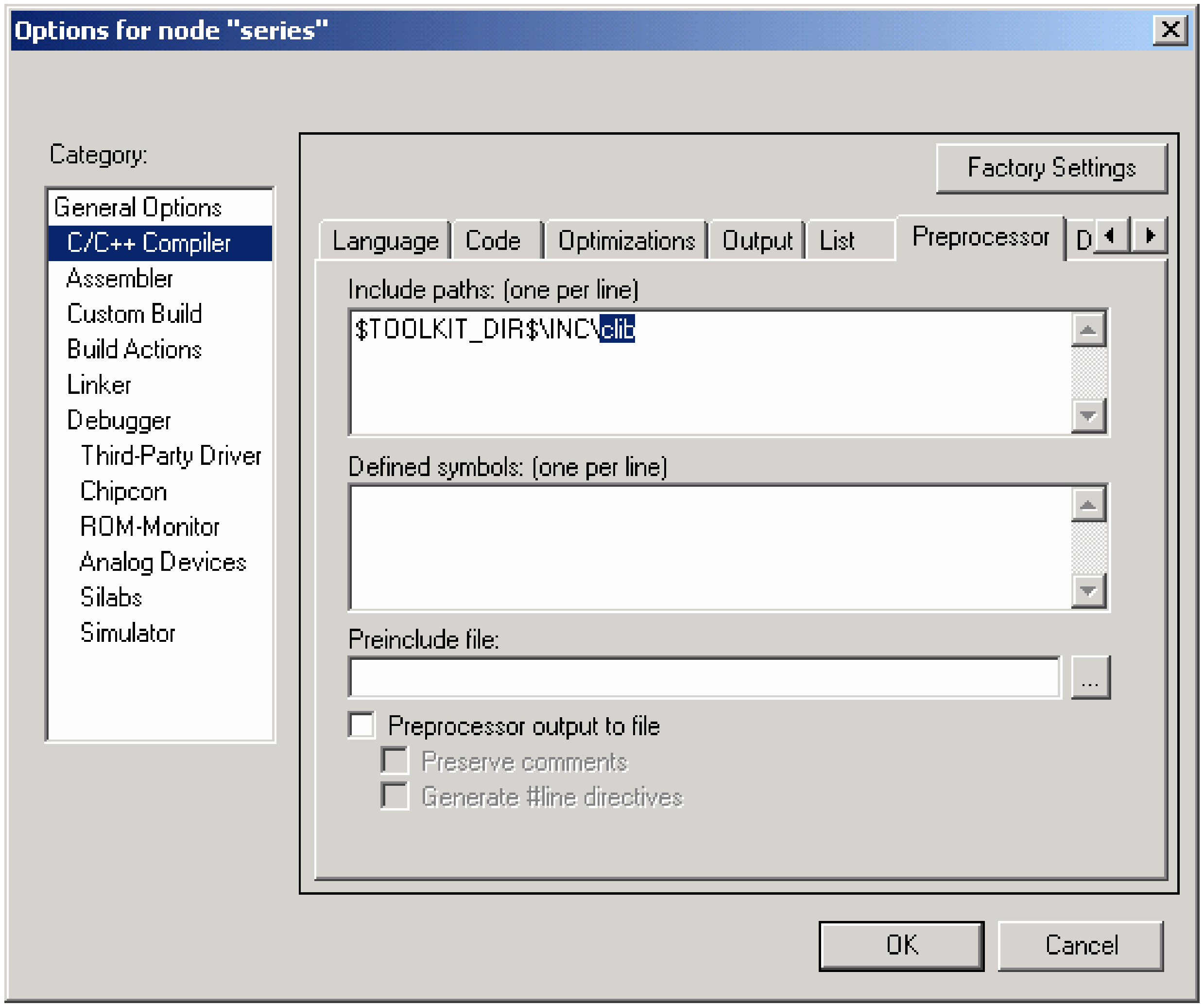
[ ] Enable IAR migration preprocessor extensions

OK Cancel

NUM

EmbeddedCraft
crafting of intelligent systems

Project options 2/3: Set path for header file



Options for node "series"

Category:
- General Options
- C/C++ Compiler
- Assembler
- Custom Build
- Build Actions
- Linker
- Debugger
  - Third-Party Driver
  - Chipcon
  - ROM-Monitor
  - Analog Devices
  - Silabs
  - Simulator

Factory Settings

Language | Code | Optimizations | Output | List | Preprocessor | D

Include paths: (one per line)

```
$TOOLKIT_DIR$\INC\clib
```

Defined symbols: (one per line)

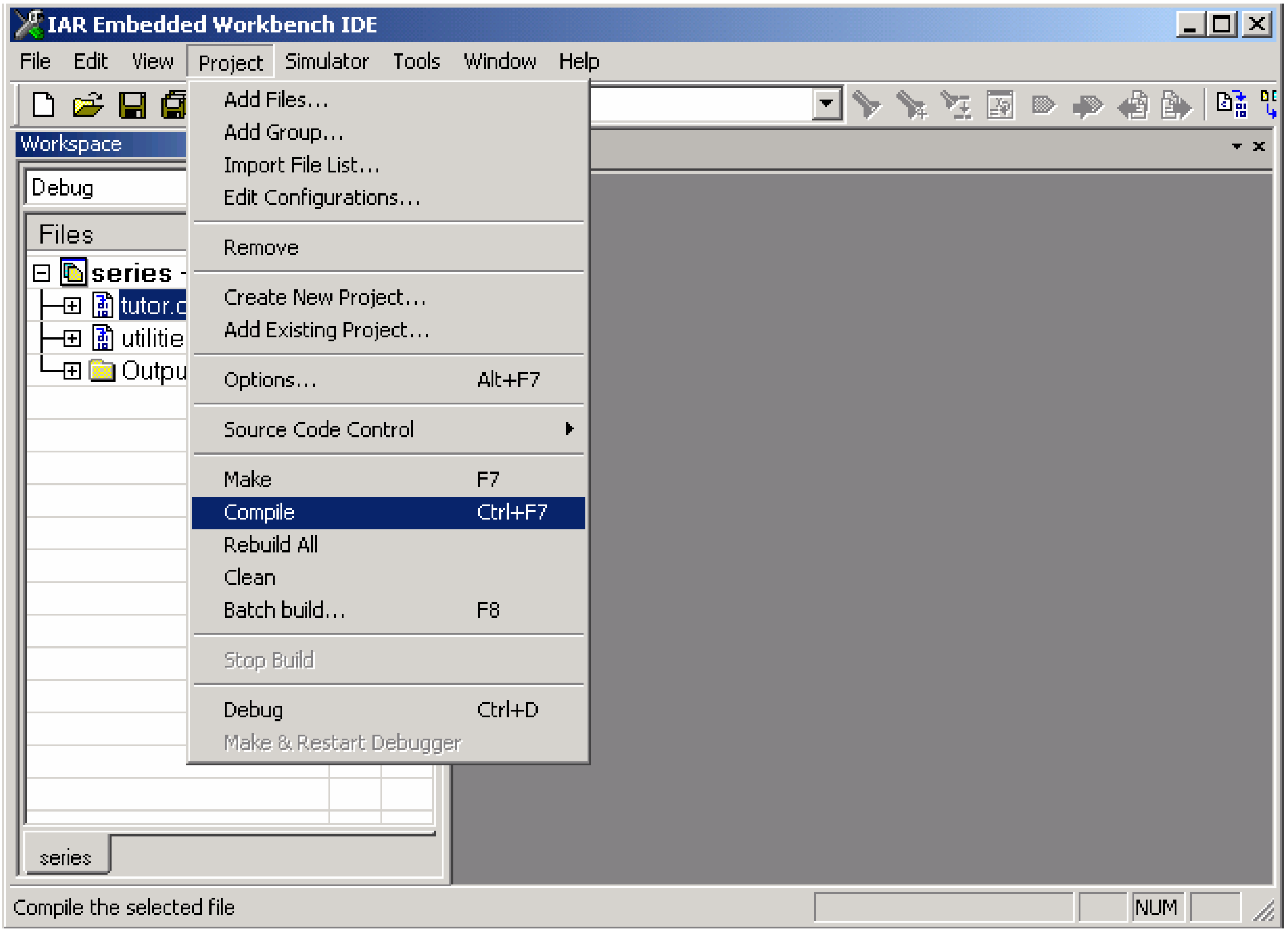
Preinclude file:

- [ ] Preprocessor output to file
  - [ ] Preserve comments
  - [ ] Generate #line directives

OK | Cancel

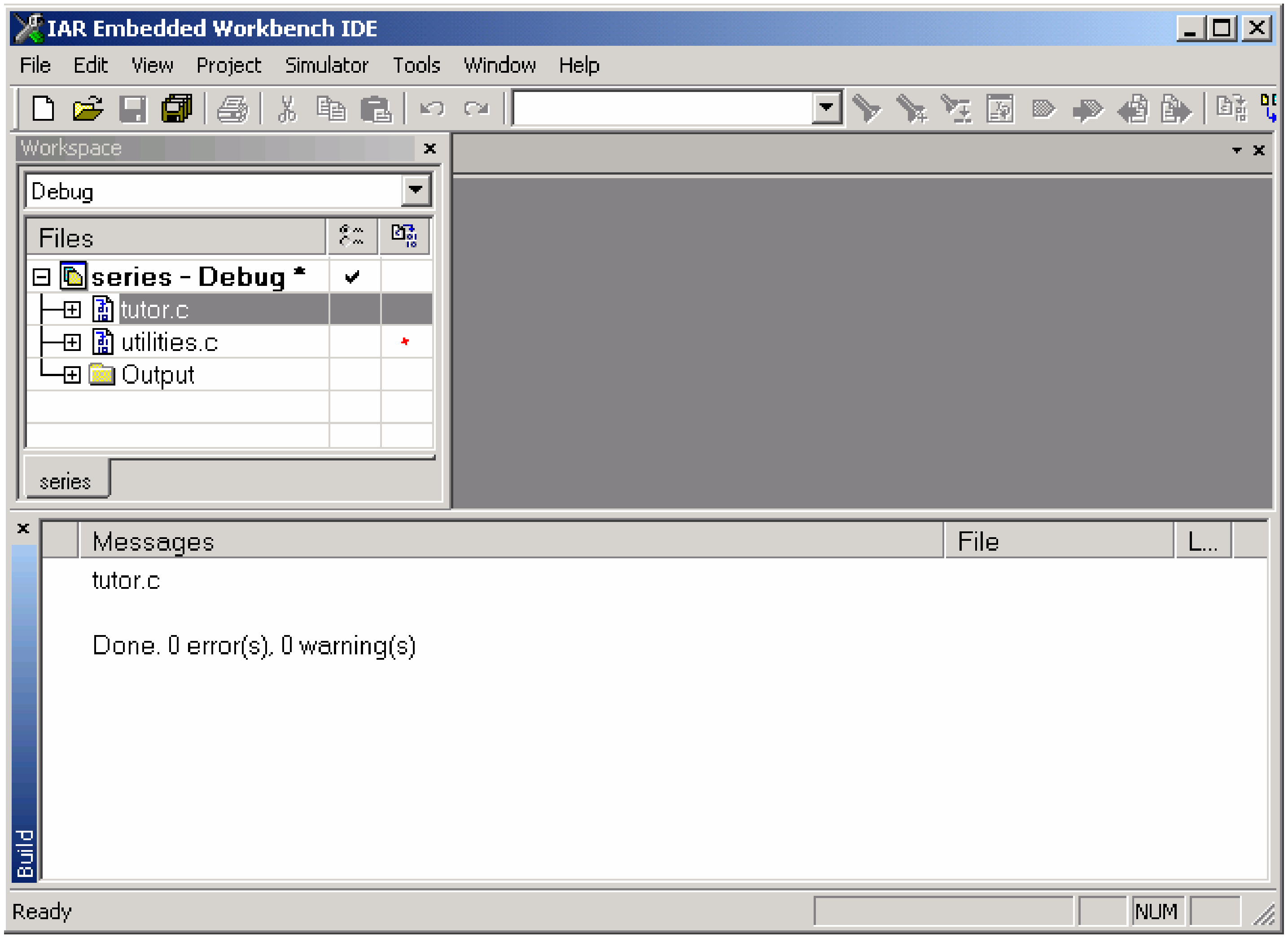
Compiling files

EmbeddedCraft
crafting of intelligent systems



Compile the selected file

Successful compilation

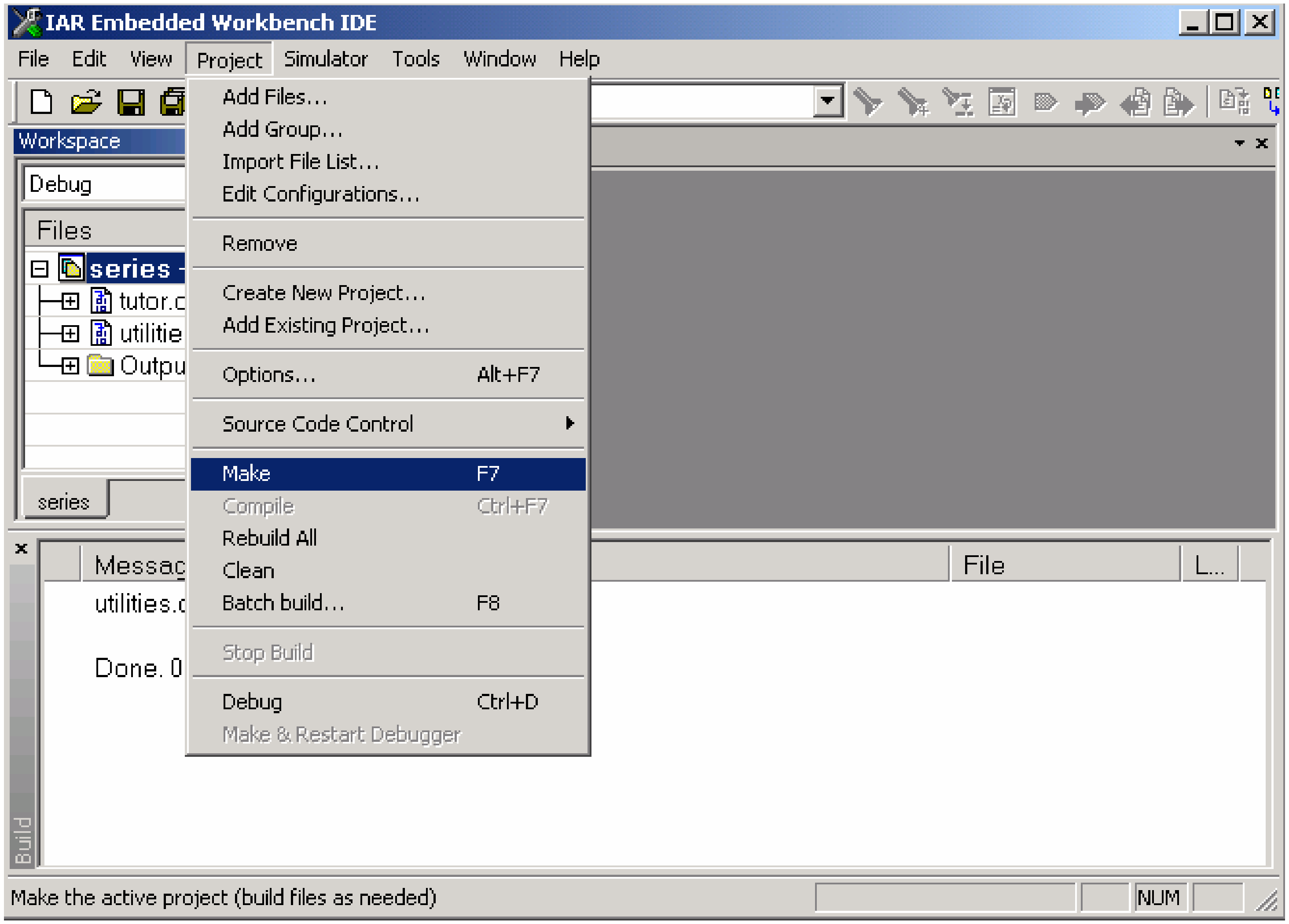


Messages | File | L...

```
tutor.c

Done. 0 error(s), 0 warning(s)
```

Linking project (make)

EmbeddedCraft
crafting of intelligent systems

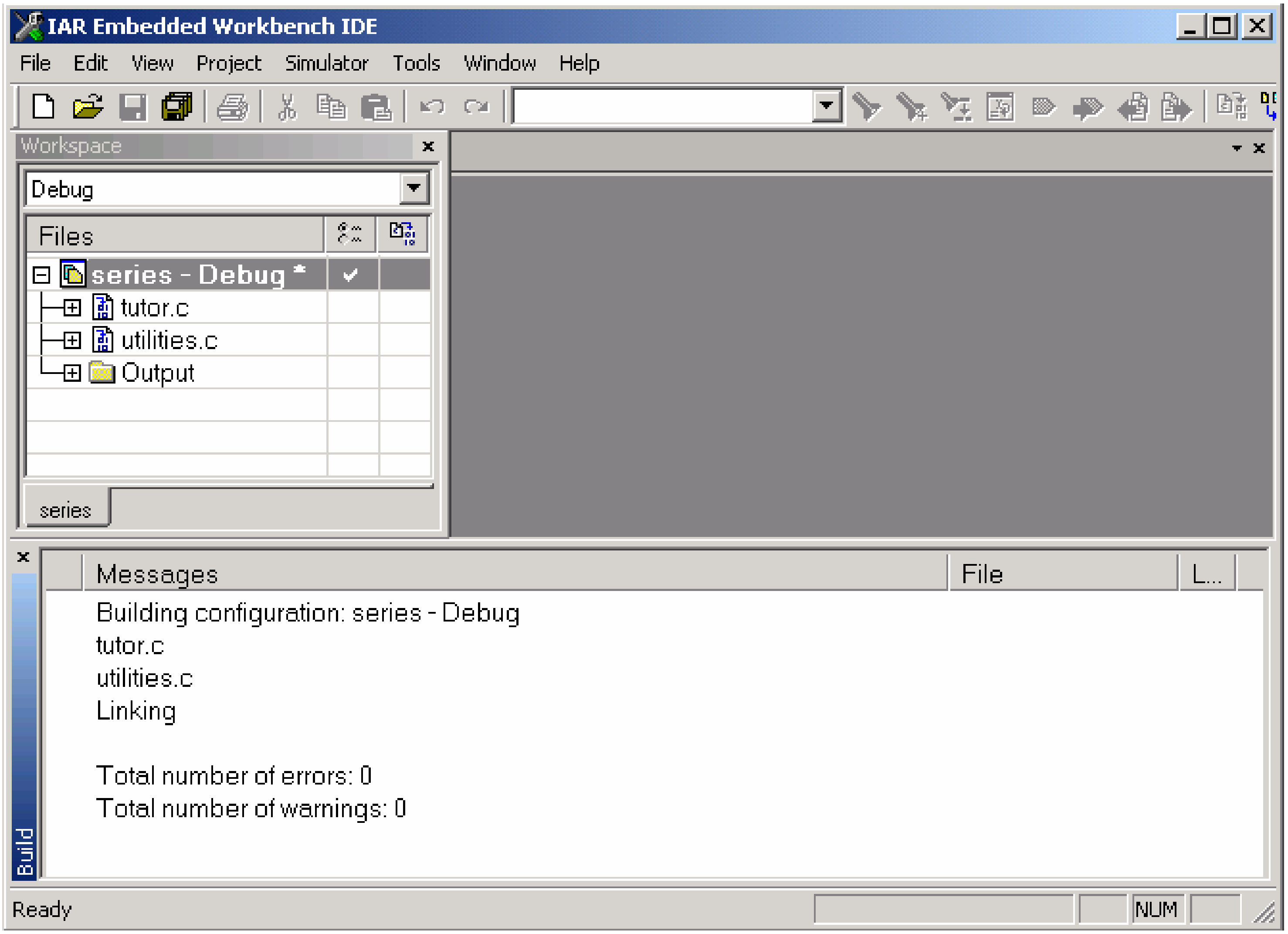


utilities.c

Done. 0

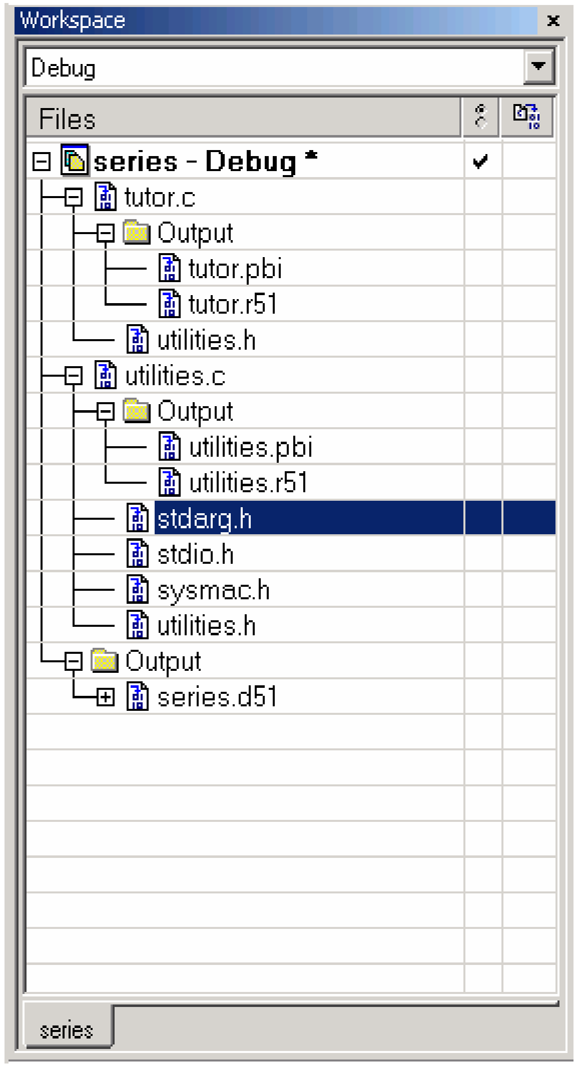
Make the active project (build files as needed)

Linking successful



```
Messages                                   File        L...
Building configuration: series - Debug
tutor.c
utilities.c
Linking

Total number of errors: 0
Total number of warnings: 0
```

File dependencies


Workspace

Debug

Files

```
series - Debug *
  tutor.c
    Output
      tutor.pbi
      tutor.r51
    utilities.h
  utilities.c
    Output
      utilities.pbi
      utilities.r51
    stdarg.h
    stdio.h
    sysmac.h
    utilities.h
  Output
    series.d51
```

series

D51= main o/p file for debug (default)

Xcl= linker files

R51= library file (CLIB)

H = header files

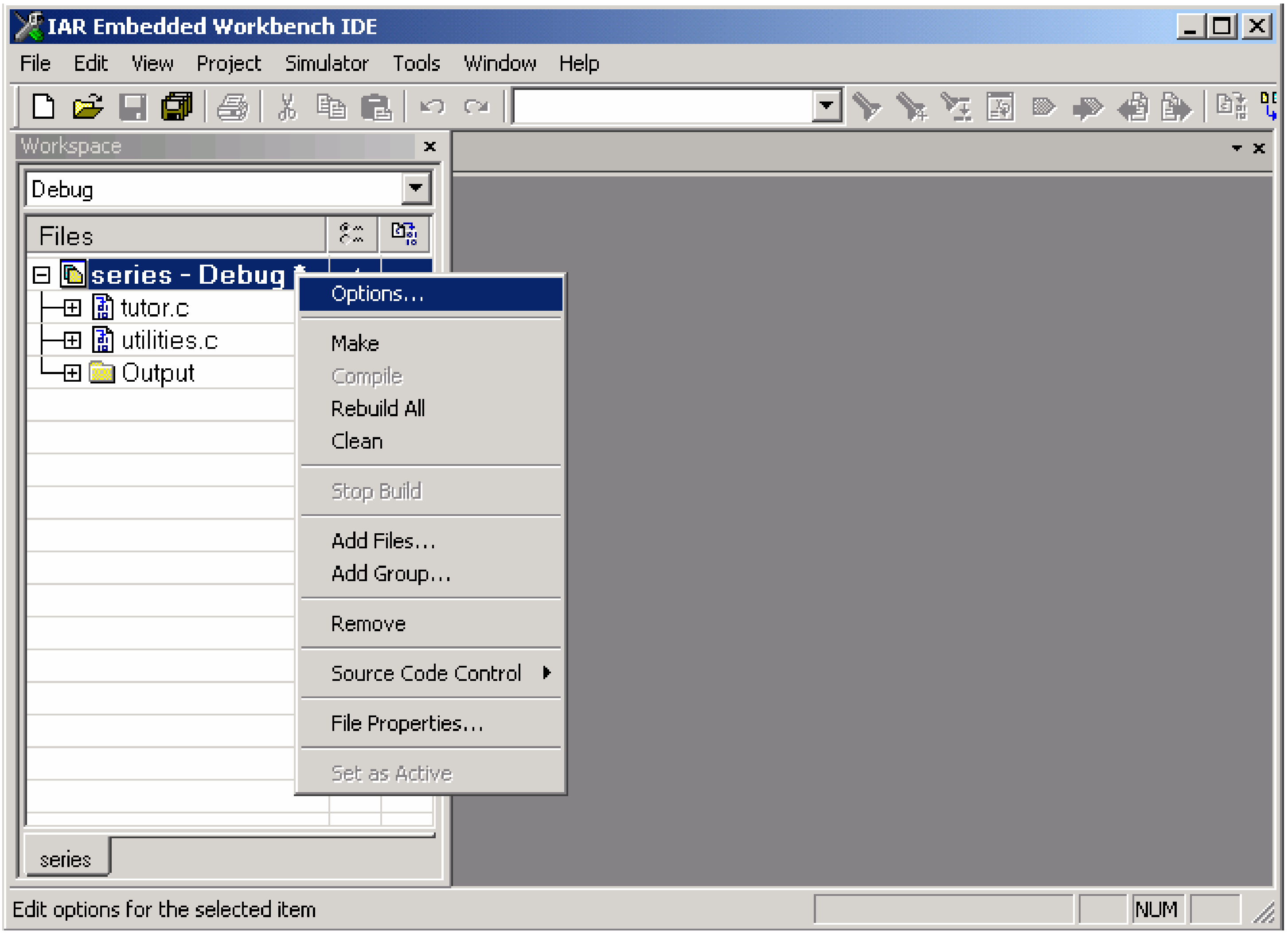
pbi=Source browser information file

Stdarg.h = standard argument header file

Sysmac.h = system macro header file

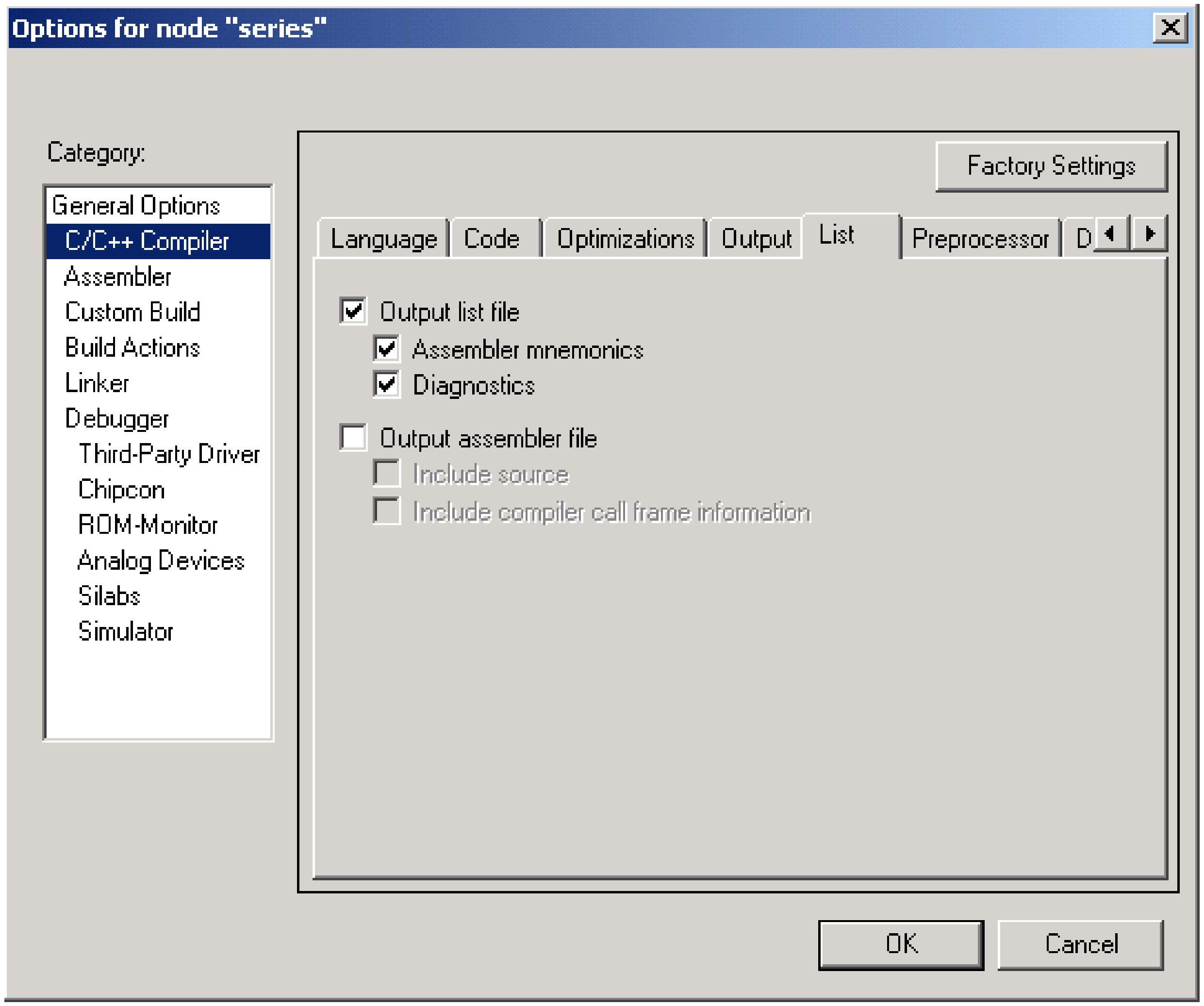
Generation of list and map files 1/3

EmbeddedCraft
crafting of intelligent systems



Edit options for the selected item

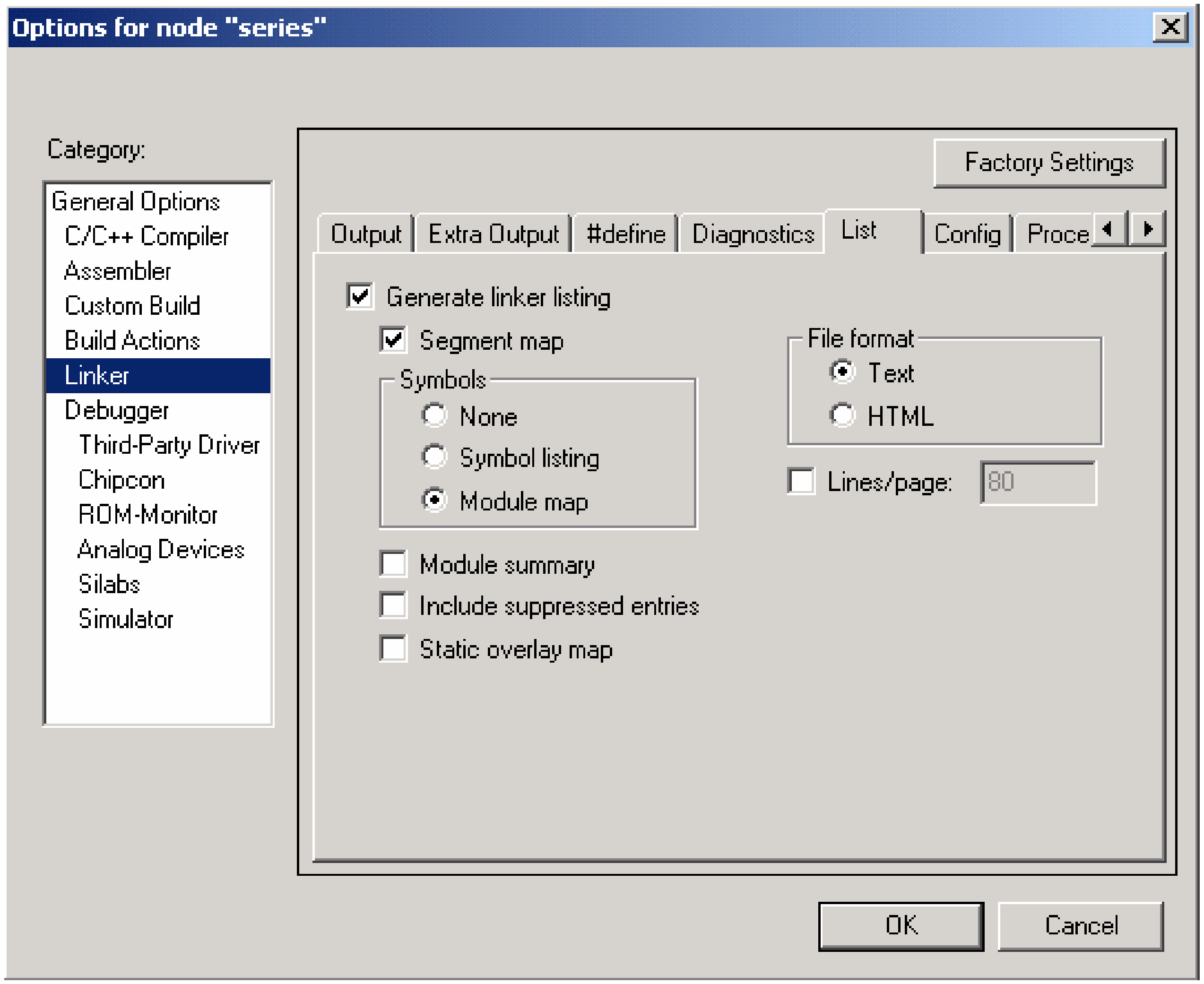
Generation of list and map files 2/3
Enable o/p list file option


Options for node "series"

Category:
- General Options
- C/C++ Compiler
- Assembler
- Custom Build
- Build Actions
- Linker
- Debugger
  - Third-Party Driver
  - Chipcon
  - ROM-Monitor
  - Analog Devices
  - Silabs
  - Simulator

- [✓] Output list file
  - [✓] Assembler mnemonics
  - [✓] Diagnostics
- [ ] Output assembler file
  - [ ] Include source
  - [ ] Include compiler call frame information

Generation of list and map files 3/3
Map file enable



Options for node "series"

Category:

- General Options
- C/C++ Compiler
- Assembler
- Custom Build
- Build Actions
- Linker
- Debugger
- Third-Party Driver
- Chipcon
- ROM-Monitor
- Analog Devices
- Silabs
- Simulator

Factory Settings

Output | Extra Output | #define | Diagnostics | List | Config | Proce

[✓] Generate linker listing

[✓] Segment map

Symbols: None / Symbol listing / [✓ Module map]

File format: [✓ Text] / HTML

[ ] Lines/page: 80

[ ] Module summary

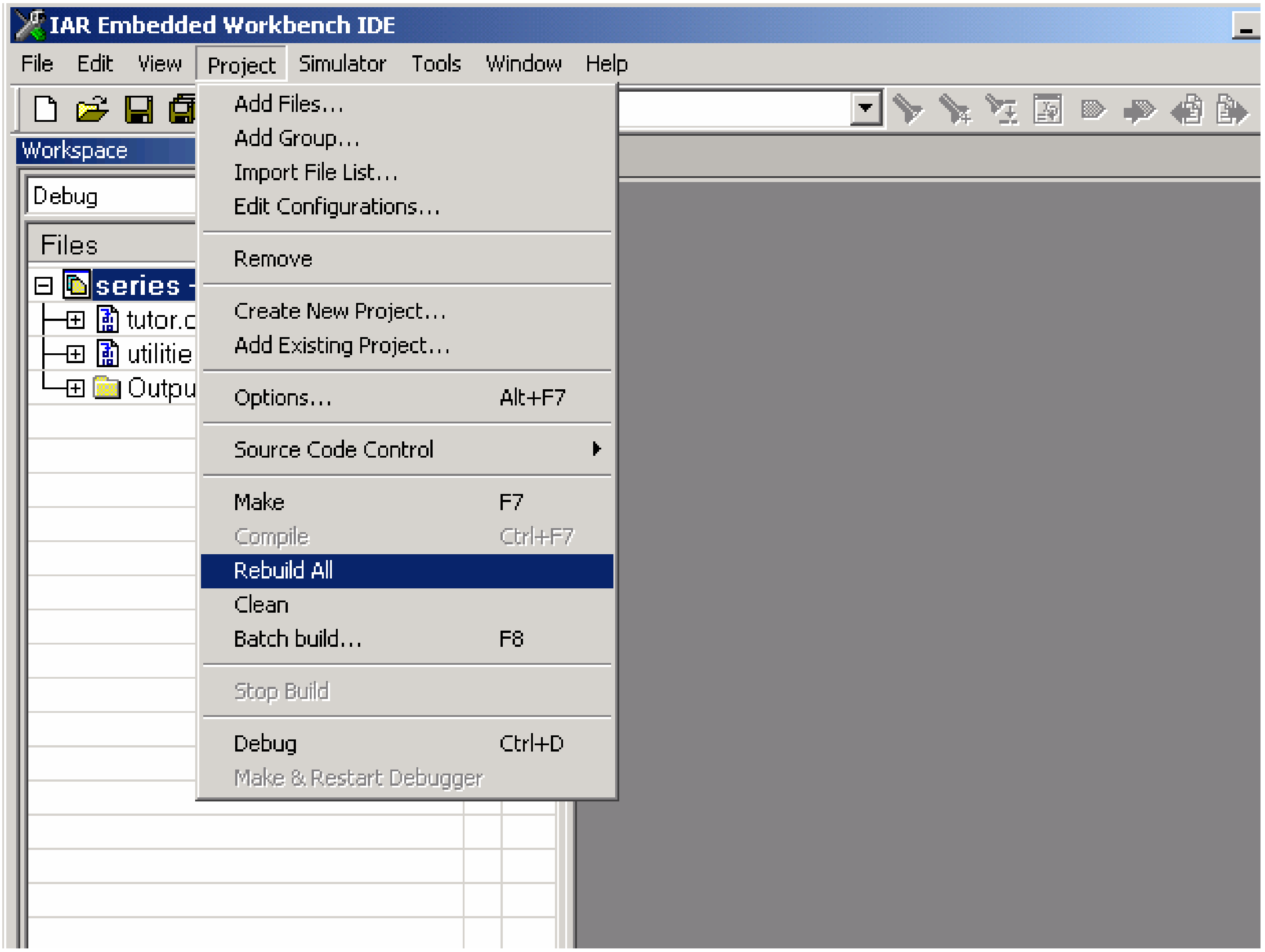
[ ] Include suppressed entries

[ ] Static overlay map

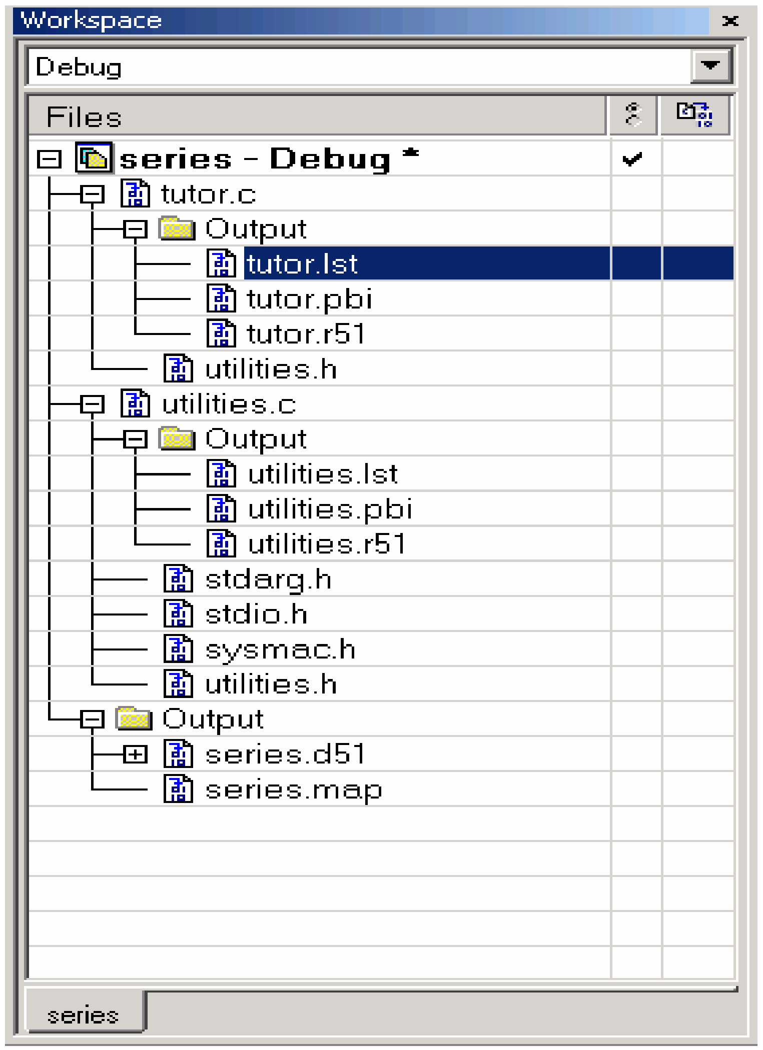
OK | Cancel

Generation of list and map files 4
Rebuild the project

EmbeddedCraft
crafting of intelligent systems



List and Map File



```
series - Debug *
├─ tutor.c
│  ├─ Output
│  │  ├── tutor.lst
│  │  ├── tutor.pbi
│  │  └── tutor.r51
│  └── utilities.h
├─ utilities.c
│  ├─ Output
│  │  ├── utilities.lst
│  │  ├── utilities.pbi
│  │  └── utilities.r51
│  ├── stdarg.h
│  ├── stdio.h
│  ├── sysmac.h
│  └── utilities.h
└─ Output
   ├─ series.d51
   └── series.map
```

List File: (lst)

-Files generated by compiler after compilation

-Display assembly and hex code for statements

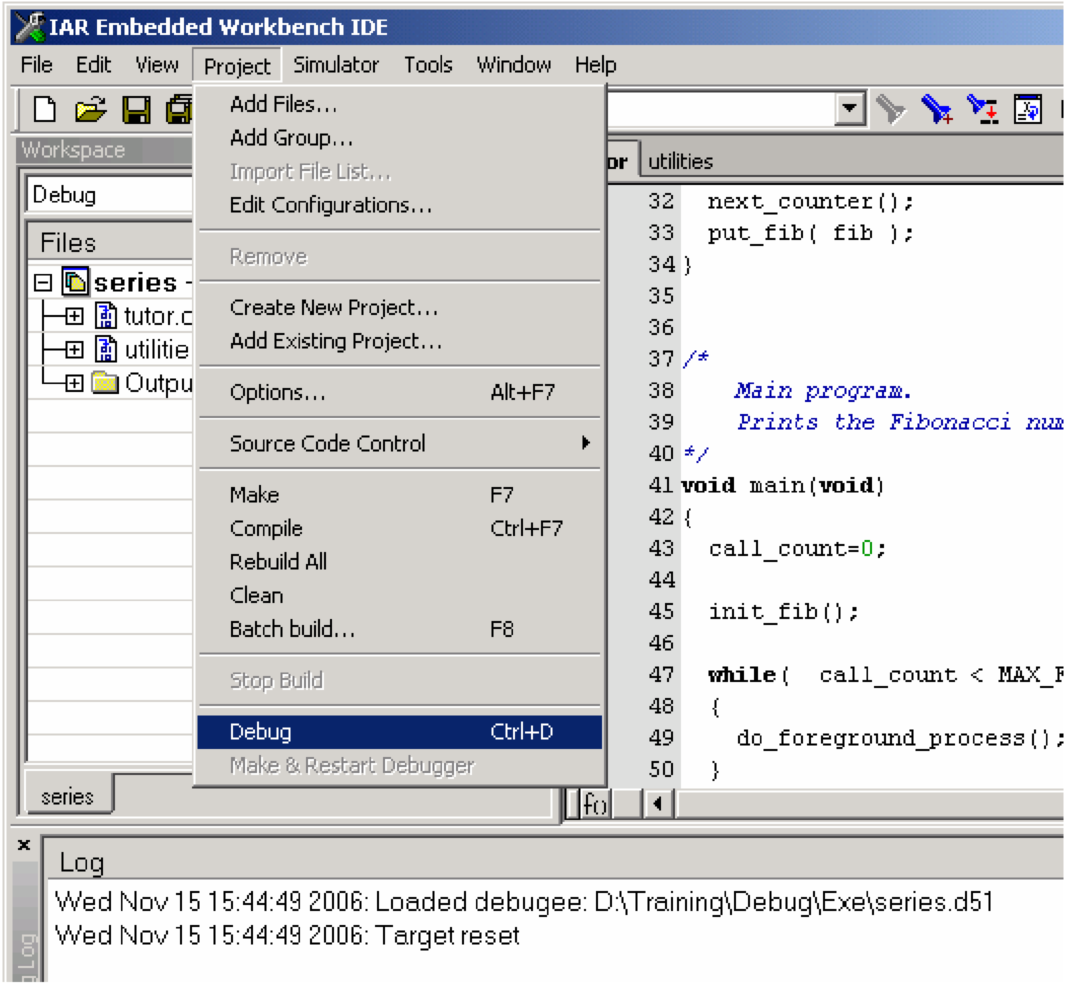
-**Map File:**

-File generated by linker

-Displays code and symbols placements in memory

Debugging the project

Open Debugger Project>Debug [CTL+D]

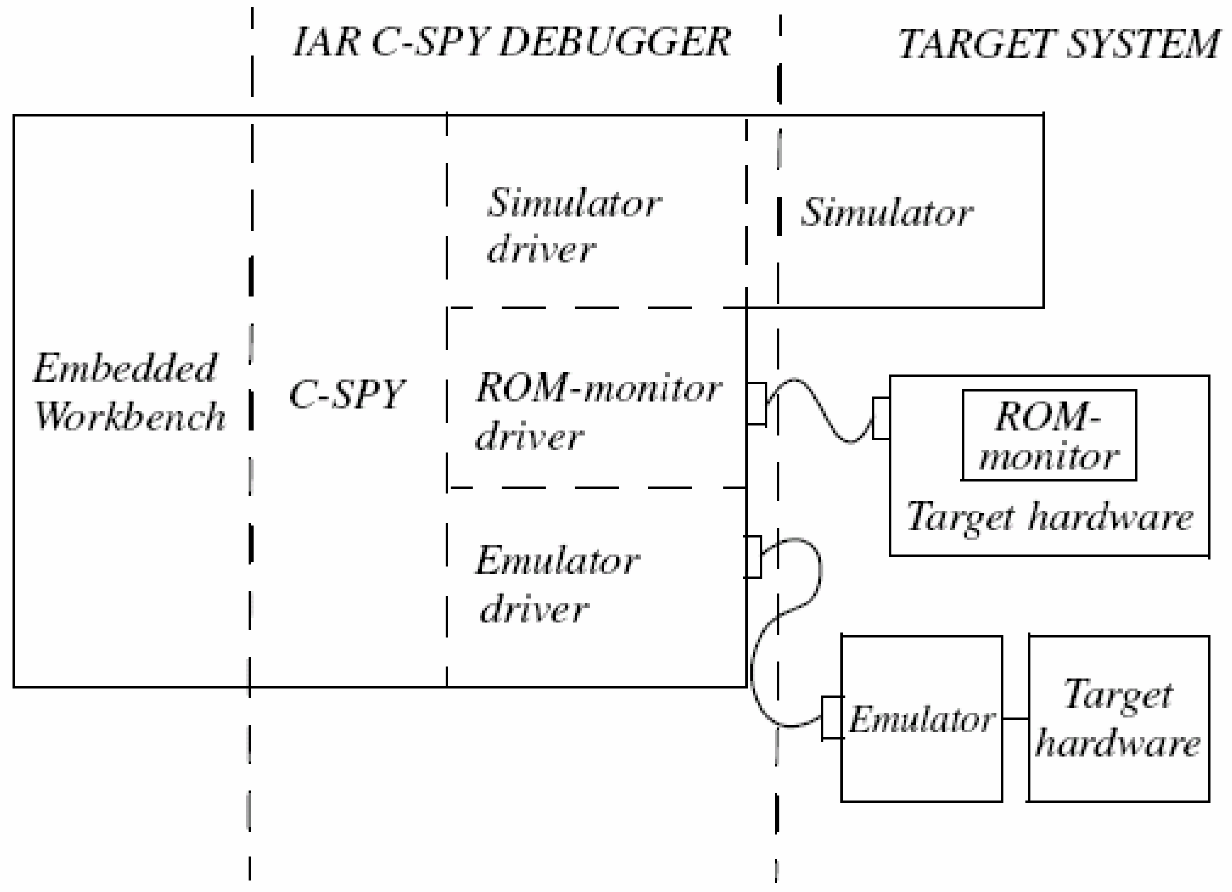


```
  next_counter();
  put_fib( fib );
}


/*
   Main program.
   Prints the Fibonacci num
*/
void main(void)
{
  call_count=0;

  init_fib();

  while(  call_count < MAX_F
  {
    do_foreground_process();
  }
```

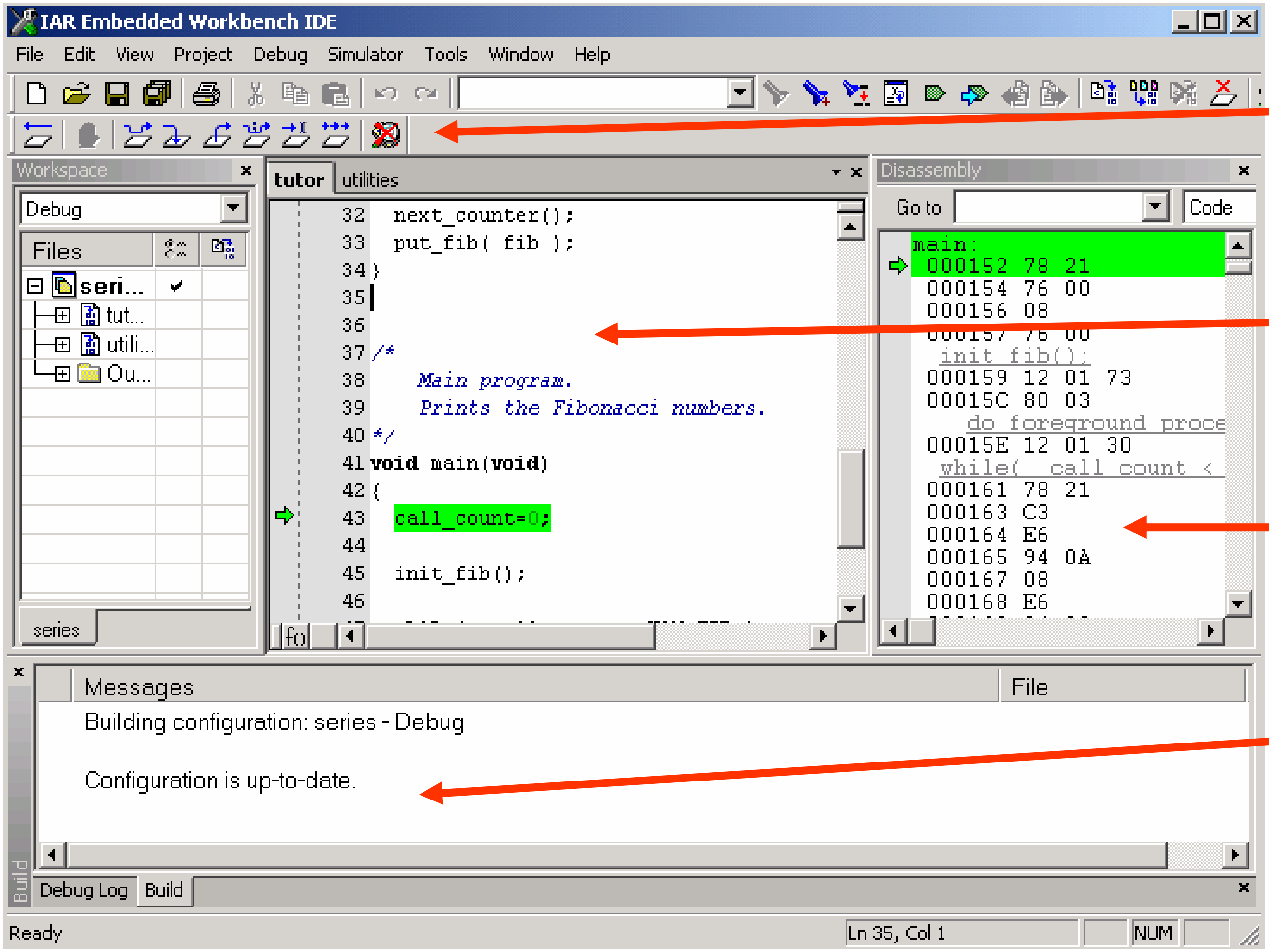
```
Wed Nov 15 15:44:49 2006: Loaded debugee: D:\Training\Debug\Exe\series.d51
Wed Nov 15 15:44:49 2006: Target reset
```

Debugger: C-Spy Debugger

EmbeddedCraft
crafting of intelligent systems



Debug Window



```
32   next_counter();
33   put_fib( fib );
34 }
35
36
37 /*
38    Main program.
39    Prints the Fibonacci numbers.
40 */
41 void main(void)
42 {
43   call_count=0;
44
45   init_fib();
46
```

```
main:
000152 78 21
000154 76 00
000156 08
000157 76 00
  init_fib();
000159 12 01 73
00015C 80 03
    do_foreground_proce
00015E 12 01 30
  while(  call_count <
000161 78 21
000163 C3
000164 E6
000165 94 0A
000167 08
000168 E6
```

```
Building configuration: series - Debug

Configuration is up-to-date.
```

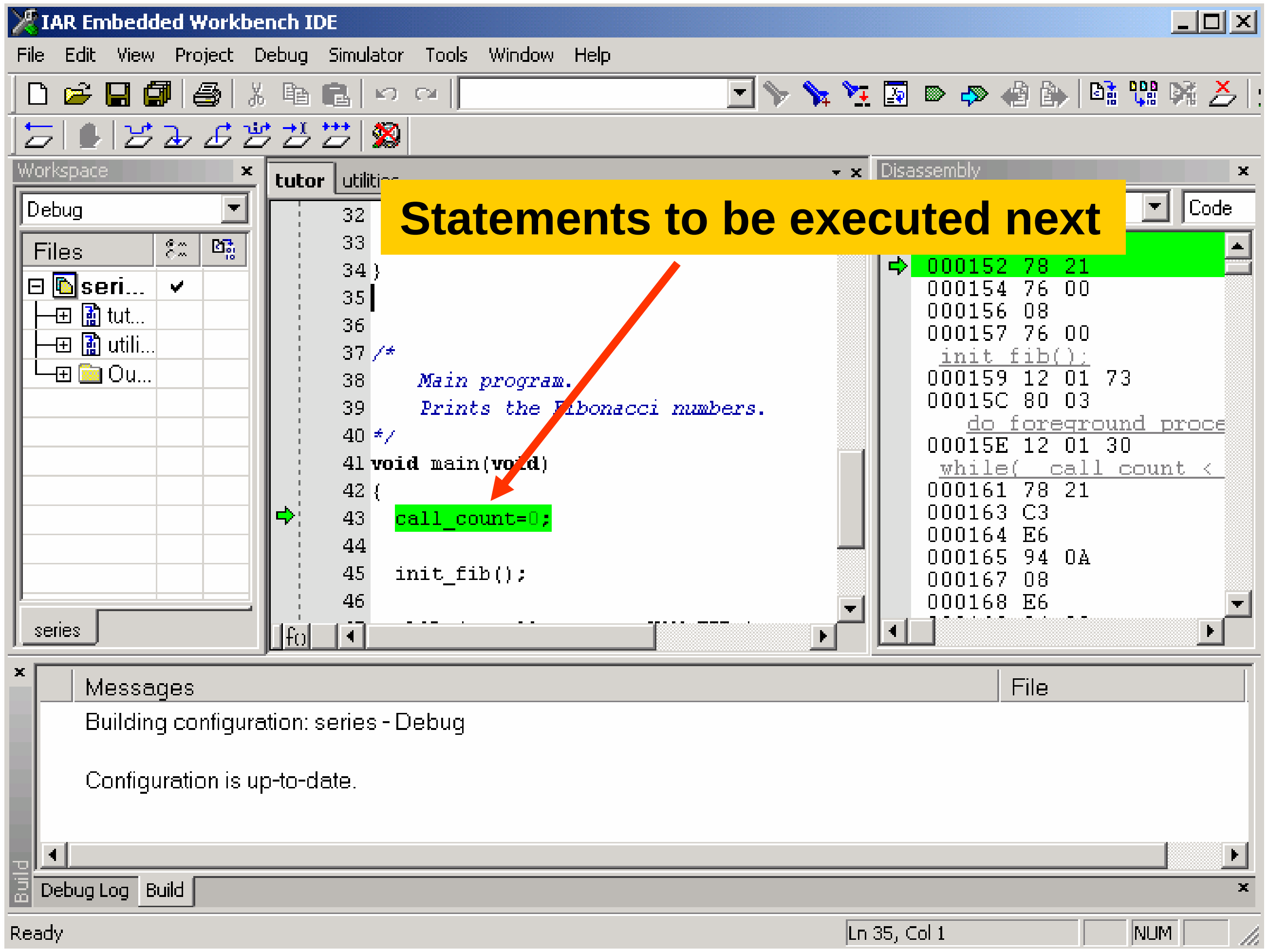
Debug Toolbar

Source Code

Disassembly view

Message Window

Debug Window



```
}

/*
   Main program.
   Prints the Fibonacci numbers.
*/
void main(void)
{
  call_count=0;

  init_fib();
```

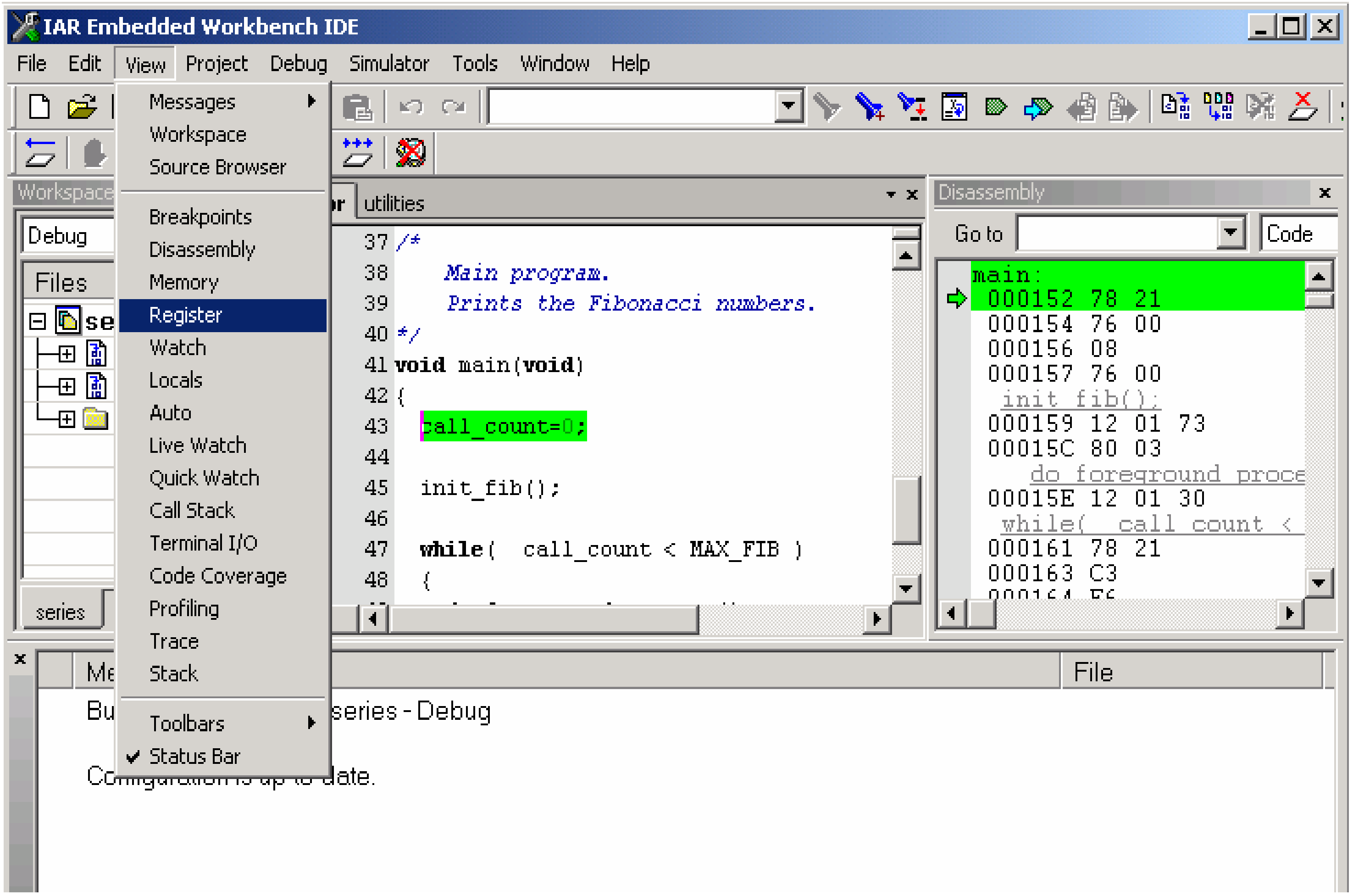
```
000152 78 21
000154 76 00
000156 08
000157 76 00
  init_fib();
000159 12 01 73
00015C 80 03
    do_foreground_proce
00015E 12 01 30
  while( call_count <
000161 78 21
000163 C3
000164 E6
000165 94 0A
000167 08
000168 E6
```

```
Building configuration: series - Debug

Configuration is up-to-date.
```

Debugging Steps

- Steps
  - Step into [F11] step into a function
  - Step over [F10] do not step into a function
  - Step out [Shift + F10] step out from function
  - Next Statement directly go to next statement
- Go [F5]: To run the program from current position
- Auto Stepping: do stepping automatically with defined time

Debugging: watching register

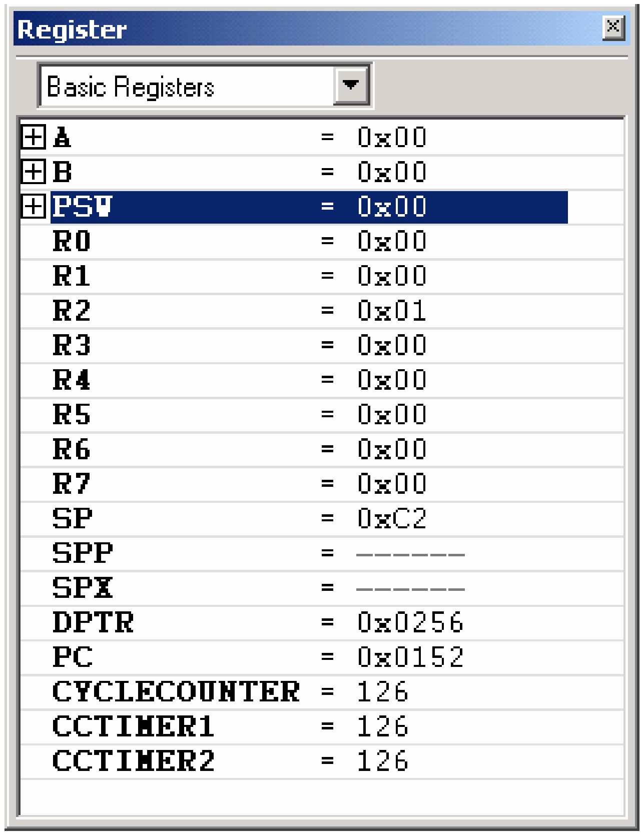


```
/*
    Main program.
    Prints the Fibonacci numbers.
*/
void main(void)
{
  call_count=0;

  init_fib();

  while(   call_count < MAX_FIB  )
  {
```

```
main:
 000152 78 21
 000154 76 00
 000156 08
 000157 76 00
   init_fib();
 000159 12 01 73
 00015C 80 03
     do_foreground_proce
 00015E 12 01 30
   while(   call_count <
 000161 78 21
 000163 C3
```

```
series - Debug
Configuration is up-to-date.
```

Debugging: Register Window

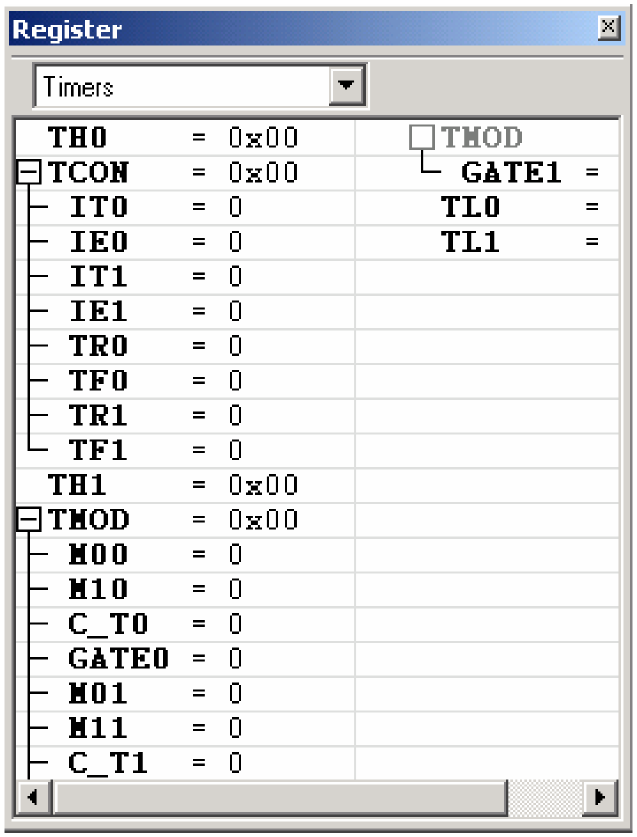


Register

Basic Registers

```
⊞A            = 0x00
⊞B            = 0x00
⊞PSW          = 0x00
 R0           = 0x00
 R1           = 0x00
 R2           = 0x01
 R3           = 0x00
 R4           = 0x00
 R5           = 0x00
 R6           = 0x00
 R7           = 0x00
 SP           = 0xC2
 SPP          = -------
 SPX          = -------
 DPTR         = 0x0256
 PC           = 0x0152
 CYCLECOUNTER = 126
 CCTIMER1     = 126
 CCTIMER2     = 126
```

Basic Register



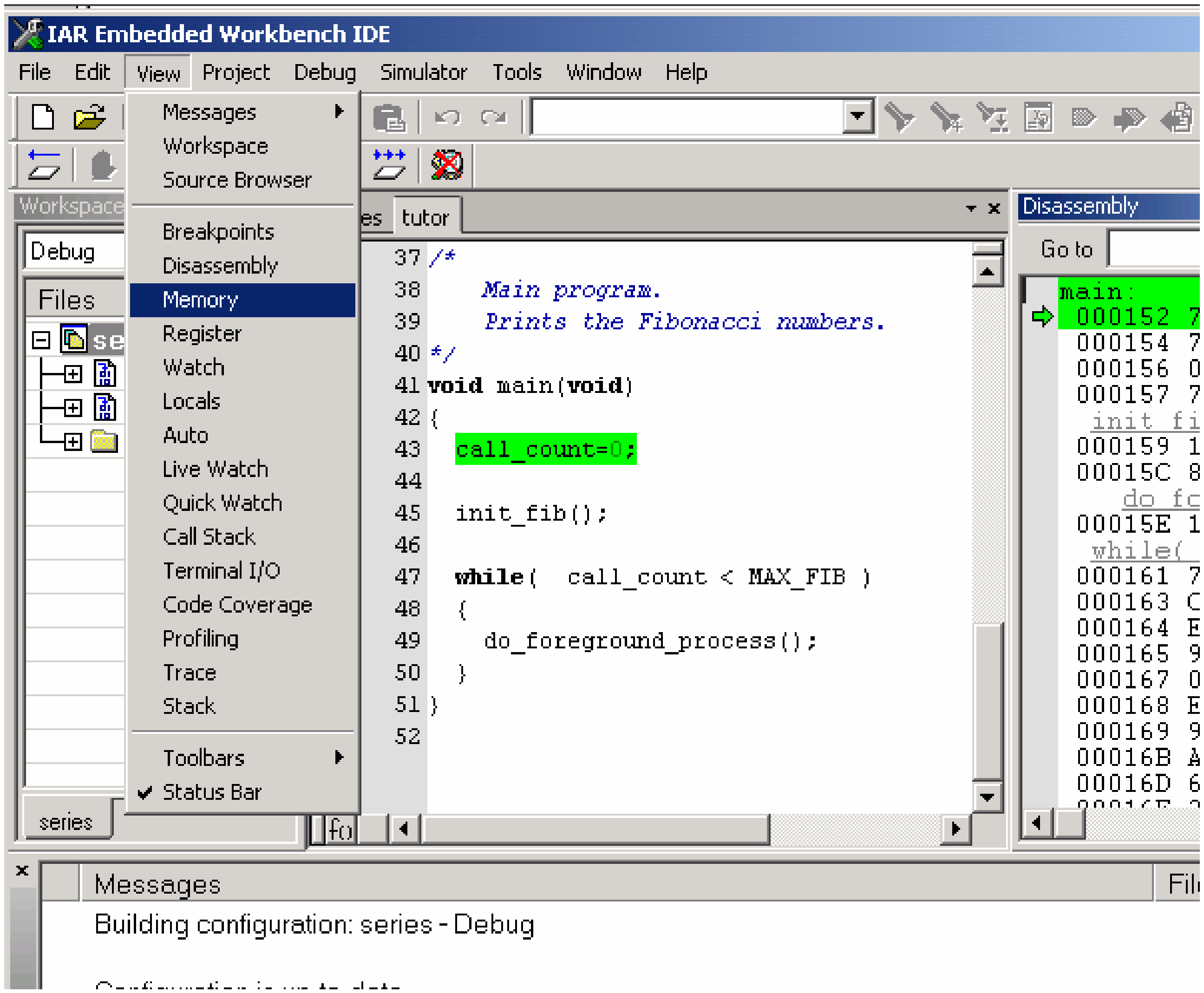
Register

Timers

```
 TH0     = 0x00     TMOD
⊟TCON    = 0x00       GATE1 =
 ├ IT0   = 0          TL0   =
 ├ IE0   = 0          TL1   =
 ├ IT1   = 0
 ├ IE1   = 0
 ├ TR0   = 0
 ├ TF0   = 0
 ├ TR1   = 0
 └ TF1   = 0
 TH1     = 0x00
⊟TMOD    = 0x00
 ├ M00   = 0
 ├ M10   = 0
 ├ C_T0  = 0
 ├ GATE0 = 0
 ├ M01   = 0
 ├ M11   = 0
 ├ C_T1  = 0
```

Timer Register

Debugging: watching memory



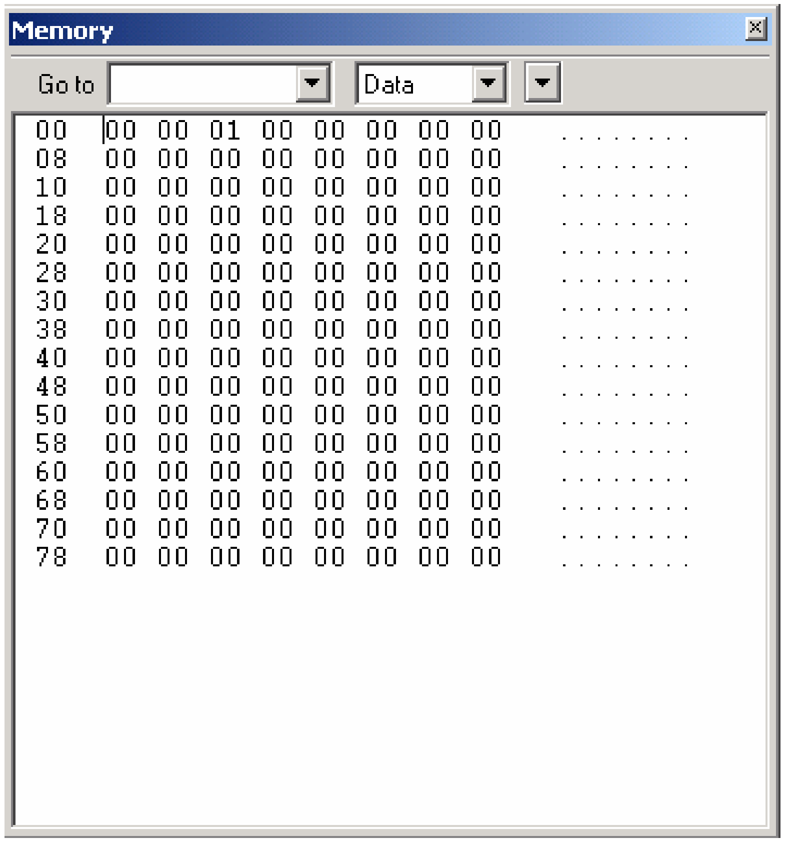
```
/*
   Main program.
   Prints the Fibonacci numbers.
*/
void main(void)
{
  call_count=0;

  init_fib();

  while(  call_count < MAX_FIB )
  {
    do_foreground_process();
  }
}
```

```
main:
 000152 7
 000154 7
 000156 0
 000157 7
  init_fi
 000159 1
 00015C 8
    do_fc
 00015E 1
  while(
 000161 7
 000163 C
 000164 E
 000165 9
 000167 0
 000168 E
 000169 9
 00016B A
 00016D 6
```

Messages

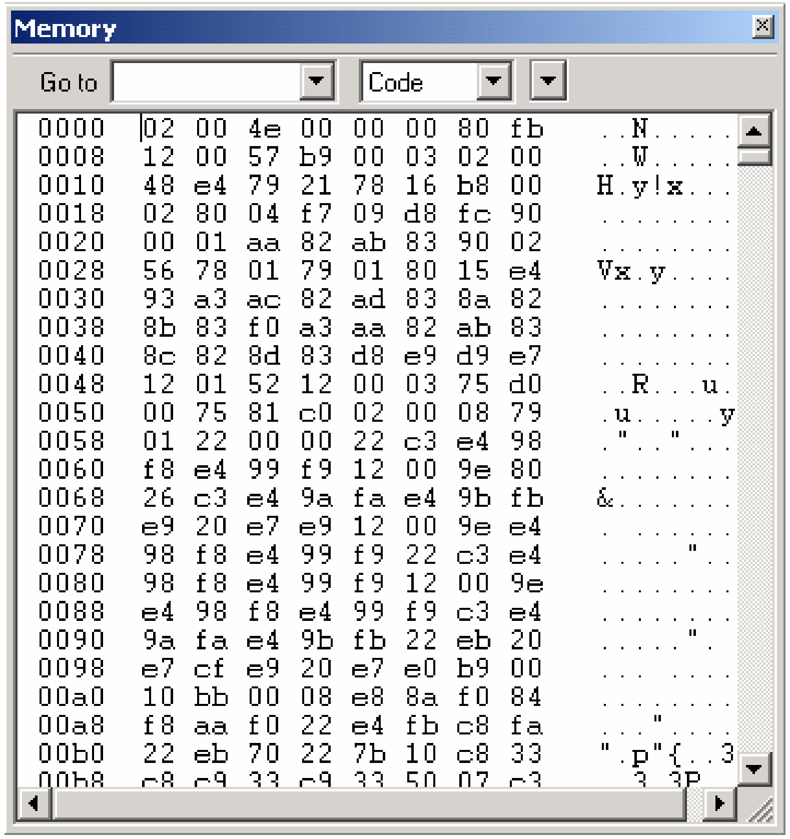
Building configuration: series - Debug

Debugger: Memory Window



```
00   00 00 01 00 00 00 00 00   ........
08   00 00 00 00 00 00 00 00   ........
10   00 00 00 00 00 00 00 00   ........
18   00 00 00 00 00 00 00 00   ........
20   00 00 00 00 00 00 00 00   ........
28   00 00 00 00 00 00 00 00   ........
30   00 00 00 00 00 00 00 00   ........
38   00 00 00 00 00 00 00 00   ........
40   00 00 00 00 00 00 00 00   ........
48   00 00 00 00 00 00 00 00   ........
50   00 00 00 00 00 00 00 00   ........
58   00 00 00 00 00 00 00 00   ........
60   00 00 00 00 00 00 00 00   ........
68   00 00 00 00 00 00 00 00   ........
70   00 00 00 00 00 00 00 00   ........
78   00 00 00 00 00 00 00 00   ........
```

Data Memory

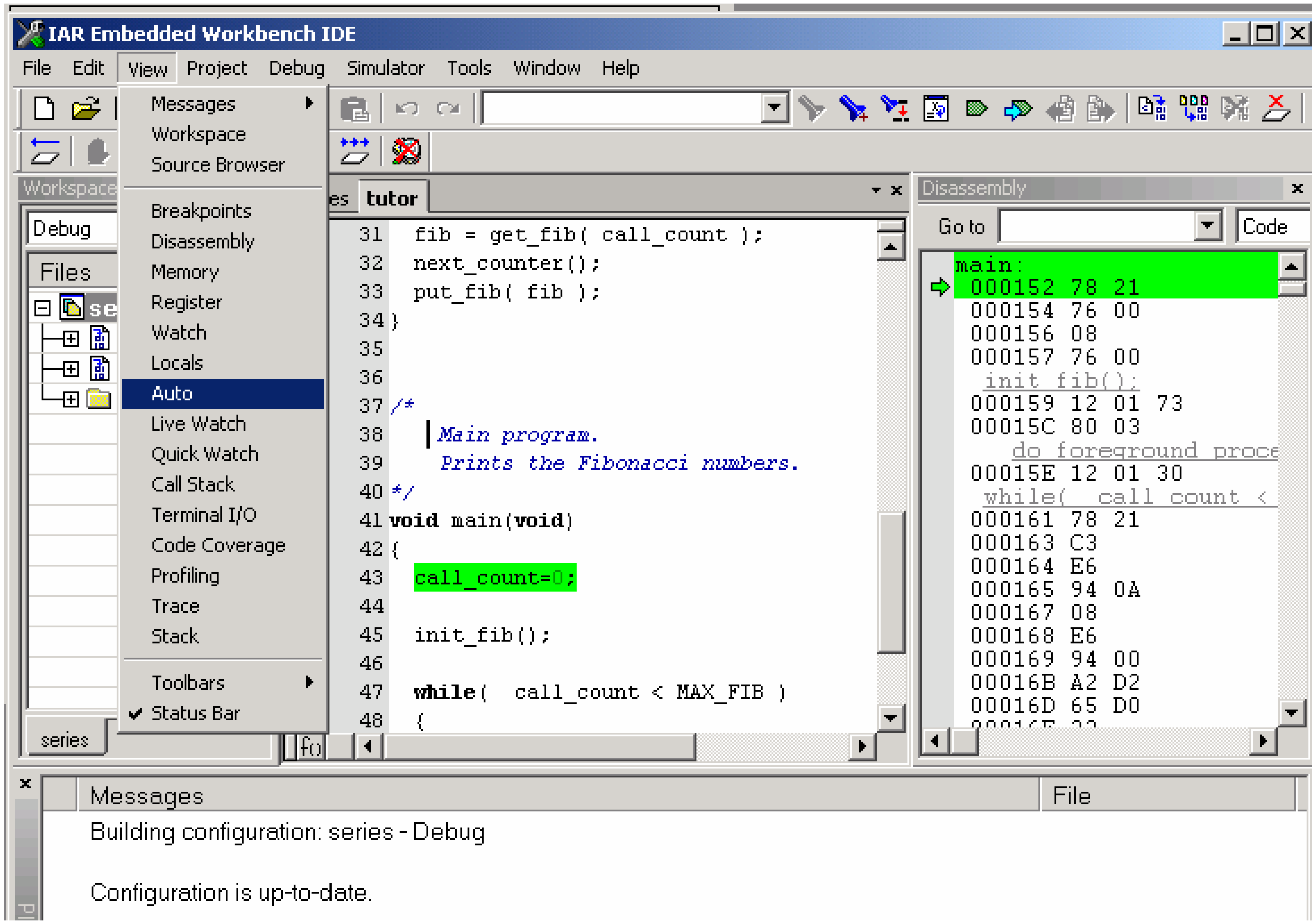


```
0000   02 00 4e 00 00 00 80 fb   ..N.....
0008   12 00 57 b9 00 03 02 00   ..W.....
0010   48 e4 79 21 78 16 b8 00   H.y!x...
0018   02 80 04 f7 09 d8 fc 90   ........
0020   00 01 aa 82 ab 83 90 02   ........
0028   56 78 01 79 01 80 15 e4   Vx.y....
0030   93 a3 ac 82 ad 83 8a 82   ........
0038   8b 83 f0 a3 aa 82 ab 83   ........
0040   8c 82 8d 83 d8 e9 d9 e7   ........
0048   12 01 52 12 00 03 75 d0   ..R...u.
0050   00 75 81 c0 02 00 08 79   .u.....y
0058   01 22 00 00 22 c3 e4 98   ."..."...
0060   f8 e4 99 f9 12 00 9e 80   ........
0068   26 c3 e4 9a fa e4 9b fb   &.......
0070   e9 20 e7 e9 12 00 9e e4   . ......
0078   98 f8 e4 99 f9 22 c3 e4   ....."..
0080   98 f8 e4 99 f9 12 00 9e   ........
0088   e4 98 f8 e4 99 f9 c3 e4   ........
0090   9a fa e4 9b fb 22 eb 20   ....."..
0098   e7 cf e9 20 e7 e0 b9 00   ... ....
00a0   10 bb 00 08 e8 8a f0 84   ........
00a8   f8 aa f0 22 e4 fb c8 fa   ..."....
00b0   22 eb 70 22 7b 10 c8 33   ".p"{..3
00b8   c8 c9 33 c9 33 50 07 c3    3 3P
```

Code Memory

Debugging: watching variables

EmbeddedCraft
crafting of intelligent systems



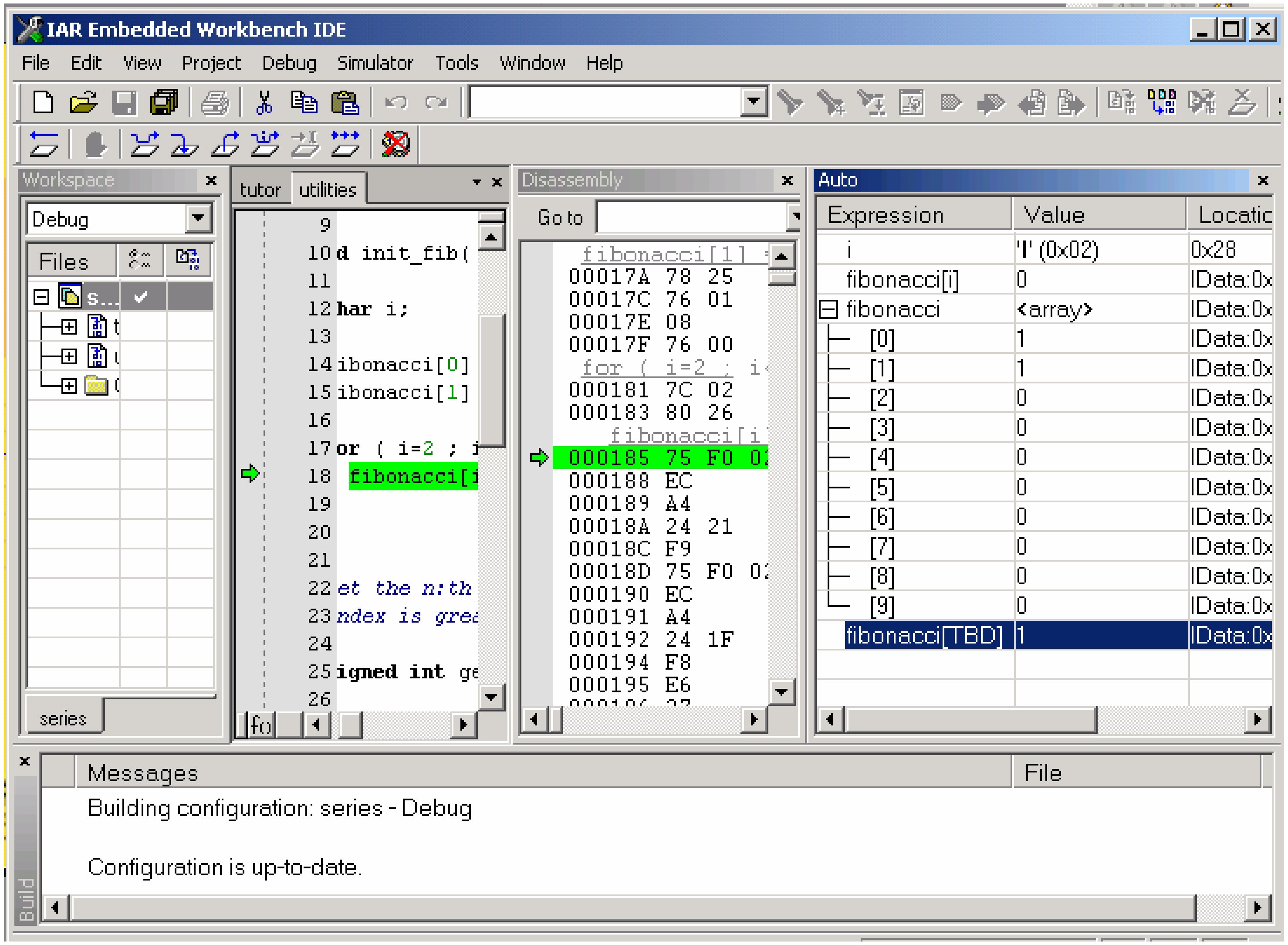
```
    fib = get_fib( call_count );
    next_counter();
    put_fib( fib );
}


/*
   Main program.
   Prints the Fibonacci numbers.
*/
void main(void)
{
    call_count=0;

    init_fib();

    while(  call_count < MAX_FIB )
    {
```

```
main:
000152 78 21
000154 76 00
000156 08
000157 76 00
 init_fib();
000159 12 01 73
00015C 80 03
   do_foreground_proce
00015E 12 01 30
 while(  call_count <
000161 78 21
000163 C3
000164 E6
000165 94 0A
000167 08
000168 E6
000169 94 00
00016B A2 D2
00016D 65 D0
```

Building configuration: series - Debug

Configuration is up-to-date.

Debugger: auto watch window


IAR Embedded Workbench IDE
File Edit View Project Debug Simulator Tools Window Help
Workspace
Debug
Files
series
tutor utilities
Disassembly
Go to
Auto
Build

```
d init_fib(

har i;

ibonacci[0]
ibonacci[1]

or ( i=2 ; i
 fibonacci[i



et the n:th
ndex is grea

igned int ge
```

```
  fibonacci[1] =
00017A 78 25
00017C 76 01
00017E 08
00017F 76 00
  for ( i=2 ; i<
000181 7C 02
000183 80 26
    fibonacci[i]
000185 75 F0 0
000188 EC
000189 A4
00018A 24 21
00018C F9
00018D 75 F0 0
000190 EC
000191 A4
000192 24 1F
000194 F8
000195 E6
```

Expression	Value	Locatic
i	'' (0x02)	0x28
fibonacci[i]	0	IData:0x
⊟ fibonacci	<array>	IData:0x
[0]	1	IData:0x
[1]	1	IData:0x
[2]	0	IData:0x
[3]	0	IData:0x
[4]	0	IData:0x
[5]	0	IData:0x
[6]	0	IData:0x
[7]	0	IData:0x
[8]	0	IData:0x
[9]	0	IData:0x
fibonacci[TBD]	1	IData:0x

Messages	File
Building configuration: series - Debug	
Configuration is up-to-date.	

Various watch windows

EmbeddedCraft
crafting of intelligent systems

Watch
Variables of current scope and global are visible.
Variables have to be defined by user.

Live Watch
Display only global variables , LIVE

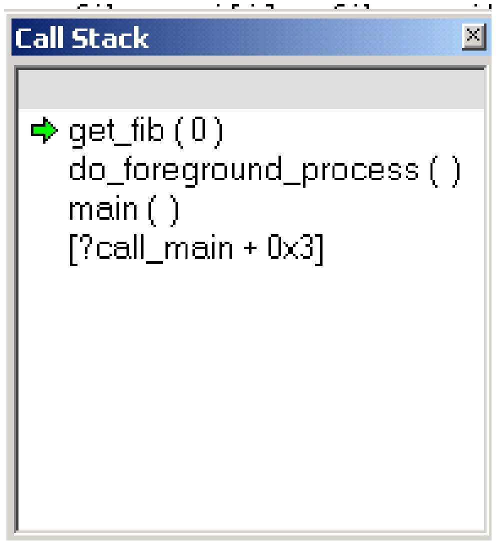
Auto
Show all variables at current line or near to the current line.

Local
It only shows local variable.

Debugging: Call Stack & Stack

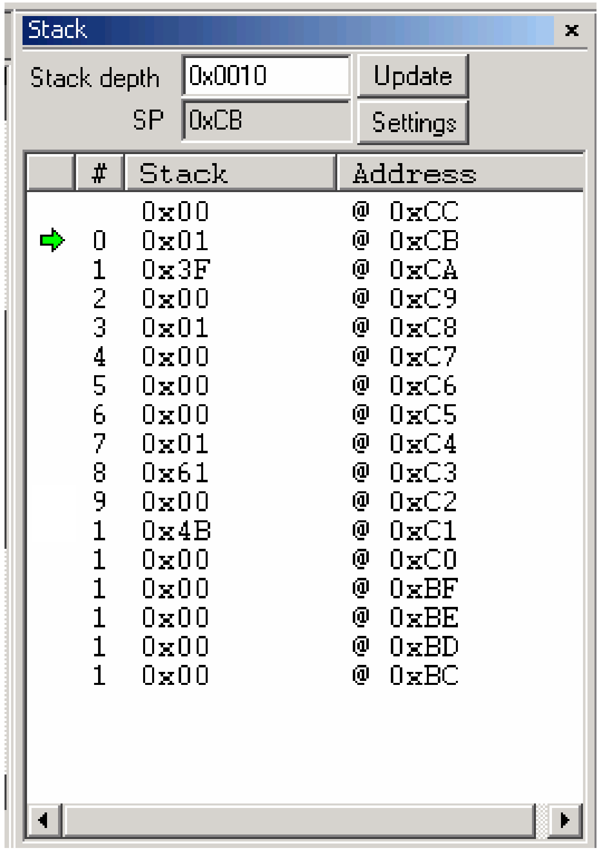
EmbeddedCraft
crafting of intelligent systems

View > Call Stack
C function call stack



```
➔ get_fib ( 0 )
  do_foreground_process ( )
  main ( )
  [?call_main + 0x3]
```

View > Stack
Shows stack usages



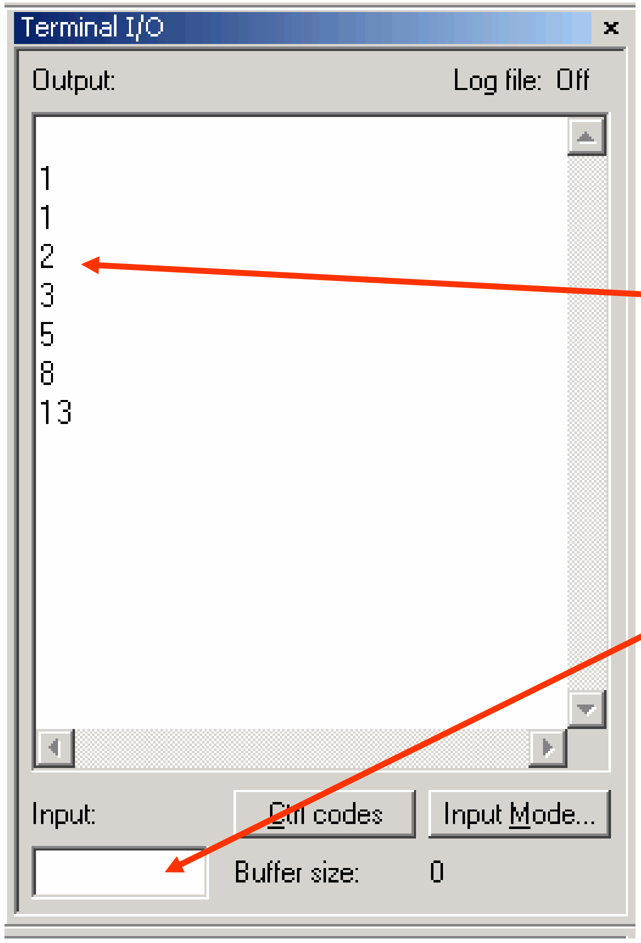
Stack depth 0x0010
SP 0xCB

	#	Stack	Address
		0x00	@ 0xCC
➔	0	0x01	@ 0xCB
	1	0x3F	@ 0xCA
	2	0x00	@ 0xC9
	3	0x01	@ 0xC8
	4	0x00	@ 0xC7
	5	0x00	@ 0xC6
	6	0x00	@ 0xC5
	7	0x01	@ 0xC4
	8	0x61	@ 0xC3
	9	0x00	@ 0xC2
	1	0x4B	@ 0xC1
	1	0x00	@ 0xC0
	1	0x00	@ 0xBF
	1	0x00	@ 0xBE
	1	0x00	@ 0xBD
	1	0x00	@ 0xBC

Debugging: Terminal I/O

EmbeddedCraft crafting of intelligent systems

View > Terminal I/O



```
1
1
2
3
5
8
13
```

-For input and output purpose

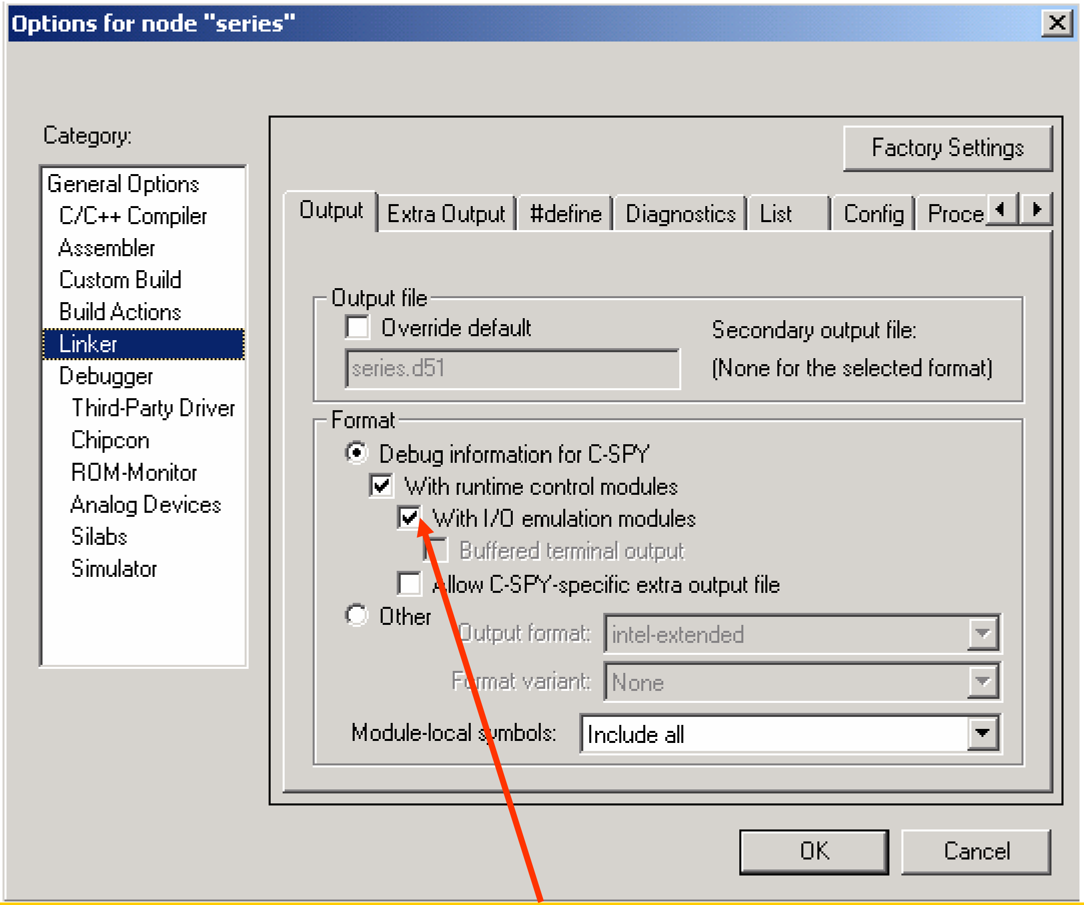
-For this following option in general option window must be checked

-Output

-Input

Debugging: Terminal I/O

EmbeddedCraft
crafting of intelligent systems


Options for node "series"

Category:
- General Options
- C/C++ Compiler
- Assembler
- Custom Build
- Build Actions
- Linker
- Debugger
- Third-Party Driver
- Chipcon
- ROM-Monitor
- Analog Devices
- Silabs
- Simulator

Factory Settings

Output | Extra Output | #define | Diagnostics | List | Config | Proce

Output file
- [ ] Override default
- series.d51
- Secondary output file:
- (None for the selected format)

Format
- [✓ Debug information for C-SPY]
  - [x] With runtime control modules
  - [x] With I/O emulation modules
    - [ ] Buffered terminal output
  - [ ] Allow C-SPY-specific extra output file
- Other
  - Output format: intel-extended
  - Format variant: None
- Module-local symbols: Include all

OK | Cancel

-For this following option in general option window must be checked http://www.embeddedcraft.org

EmbeddedCraft

crafting of intelligent systems