Introduction à l'outil de développement IAR : ARM7TDMI

Par

Kun Mean HOU, Christophe de Vaulx et Alain Tanguy

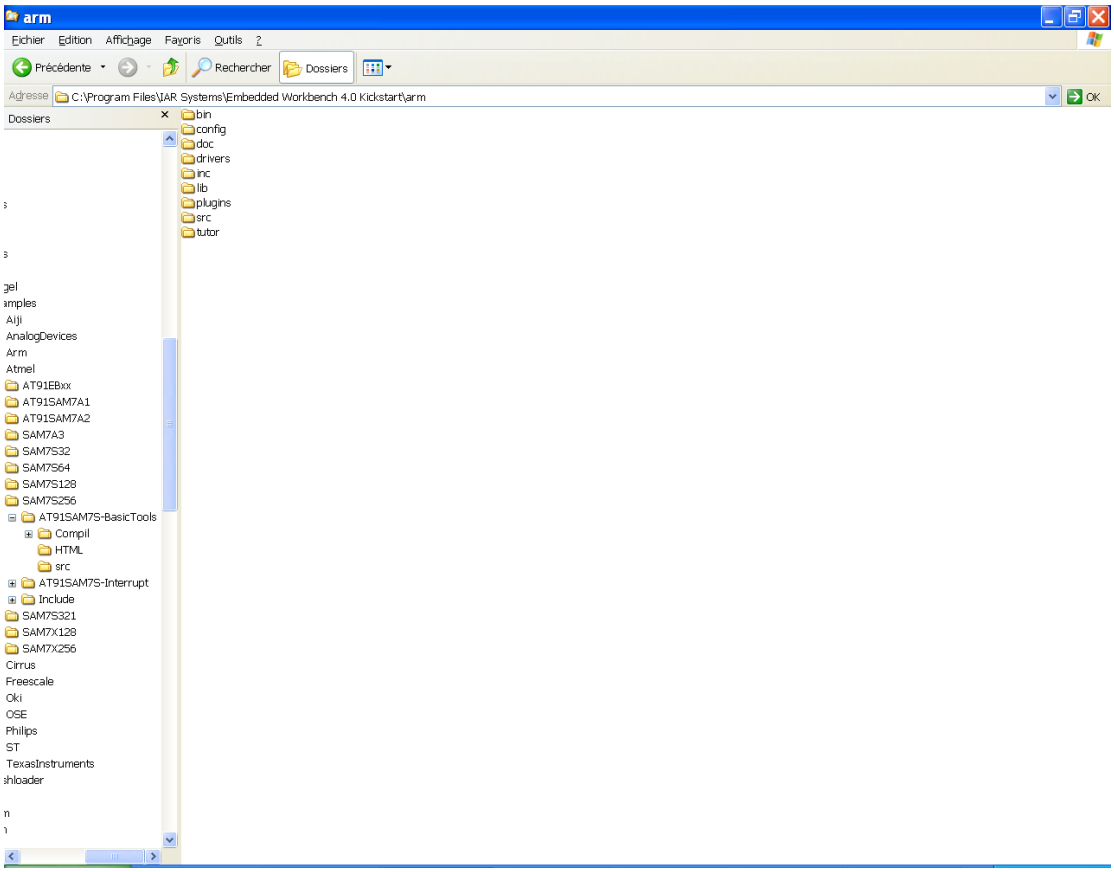
Table des matières

Présentattion de l'outil de développement de l'IAR

L'outil de développement intégré IAR "IAR Embedded Workbench IDE" est un ensemble d'outils suivants :

- Un compilateur C/C++ « ARM IAR C/C++ Compiler »
- Un assembleur « ARM IAR Assembler »
- Un éditeur de lien « IAR XLINK Linker »
- Une bibliothèque « IAR XAR Library Builder and the IAR XLIB Librarian"
- Un éditeur
- Un gestionnaire de projet
- Un debugger « IAR C-SPY® debugger".

Les logiciels d'IAR livrés avec le KIT de développement ACTEL basé sur le microcontôleur ARM7TDMI (AT91SAM7S256) comportent les sous-répertoires suivants (Voir figure ci-dessous).



Répertoire arm\bin

Le répertoire arm\bin contient les fichiers exécutables tells que le compilateur, l'assembleur etc.

Répertoire arm\config

Le répertoire arm\config contient les fichiers utilisés pour configurer l'environnement de développement et de projets. Par exemples :

- Fichier de commande d'éditeur de lien 'Linker command files (*.xcl)'
- Fichier cible 'Chip settings (*.i79)'
- 'The C-SPY device description files (*.ddf)'
- Fichier de configuration 'Syntax coloring configuration files (*.cfg)'
- Fichier d'exemple de projet d'application et de bibliothèque (*.ewp).

Répertoire arm\inc

Le répertoire arm\inc contient les fichiers "include" tels que les fichiers d'entête standard des bibliothèques des langages C/C++, les fichiers spécifiques 'specific header files defining special function registers (SFRs)'.

Répertoire arm\lib

Le répertoire arm\lib contient les fichiers précompilés des bibliothèques utilisées par le compilateur.

Répertoire arm\plugins

Le répertoire arm\plugins contient des fichiers exécutables des composants qui peuvent être téléchargés comme un 'plugin'.

Répertoire arm\src

Le répertoire arm\src contient des fichiers sources de la bibliothèque ainsi les codes d'applications.

Répertoire arm\tutor

Le répertoire arm\tutor contient des fichiers utilisés pour le tutorial.

Répertoire COMMON

Le répertoire 'common directory' contient des composants partagés par tous les produits de l'IAR Embedded Workbench.

Répertoire common\bin

Le répertoire common\bin contient des fichiers communs pour tous tels que 'IAR Embedded Workbench' ' IAR XLINK Linker', 'IAR XLIB', 'IAR XAR Library Builder', 'editor and the graphical user interface'. Le fichier exécutable 'IAR Embedded Workbench IDE' se trouve dans ce répertoire.

Répertoire common\config

Le répertoire common\config contient les fichiers utilisés par le 'IAR Embedded Workbench' pour sauvegarder l'environnement du développement.

Répertoire common\doc

Le répertoire common\doc contient les fichiers 'readme' indiquant les ajouts récents de tous les composants communs à tous les produits de l''IAR Embedded Workbench'. Il est recommandé de lire ces fichiers. Ce répertoire contient les manuels d'utilisation de '*IAR Linker and Library Tools Reference Guide'*.

Répertoire common\plugins

Le répertoire common\plugins contient les fichiers exécutables et de description des composants 'plugin'.

Répertoire common\src

Le répertoire common\src contient les fichiers sources des composants communs à tous les produits 'IAR Embedded Workbench'.

Les différents types de fichiers d'extension sont :

Ext.	Type of file	Output from	Input to
a79	Target application	XLINK	EPROM, C-SPY, etc.
asm	Assembler source code	Text editor	Assembler
c	C source code	Text editor	Compiler
cfg	Syntax coloring configuration	Text editor	IAR Embedded Workbench
cpp	Embedded C++ source code	Text editor	Compiler
d79	Target application with debug information	XLINK	C-SPY and other symbolic debuggers
dbg	Target application with debug information	XLINK	C-SPY and other symbolic debuggers
dbgt	Debugger desktop settings	C-SPY	C-SPY
ddf	Device description file	Text editor	C-SPY
dep	Dependency information	IAR Embedded Workbench	IAR Embedded Workbench
dni	Debugger initialization file	C-SPY	C-SPY
ewd	Project settings for C-SPY	IAR Embedded Workbench	IAR Embedded Workbench
ewp	IAR Embedded Workbench project (current version)	IAR Embedded Workbench	IAR Embedded Workbench
eww	Workspace file	IAR Embedded Workbench	IAR Embedded Workbench
fmt	Formatting information for the Locals and Watch windows	IAR Embedded Workbench	IAR Embedded Workbench
h	C/C++ or assembler header source	Text editor	Compiler or assembler #include
i	Preprocessed source	Compiler	Compiler
i79	ARM chip settings	Text editor	IAR Embedded Workbench
inc	Assembler header source	Text editor	Assembler #include
lst	List output	Compiler and assembler	–

Répertoire *doc* contient les documents liés au systèmes de développement IAR *:*

- Le fichier « template » : décrit les modèles de programmation adoptés par l'IAR
- Le fichier « EWARM_CompilerReference »
- Le fichier « EWARM_AssemblerReference »

- Le fichier « EWARM_UserGuide »

Le debugger "IAR C-SPY" est un debugger de langage haut niveau dédié aux applications embarquées. Il est conçu pour être utilisé avec le compilateur et l'assembleur d'IAR. Il permet d'éditer et de debugger des applications. Le C-SPY debugger peut être utilisé pour faire la mise au point du logiciel ou du matériel (Figure 45).

Figure 45, *IAR C-SPY Debugger and target systems*, shows an overview of C-SPY and possible target systems.

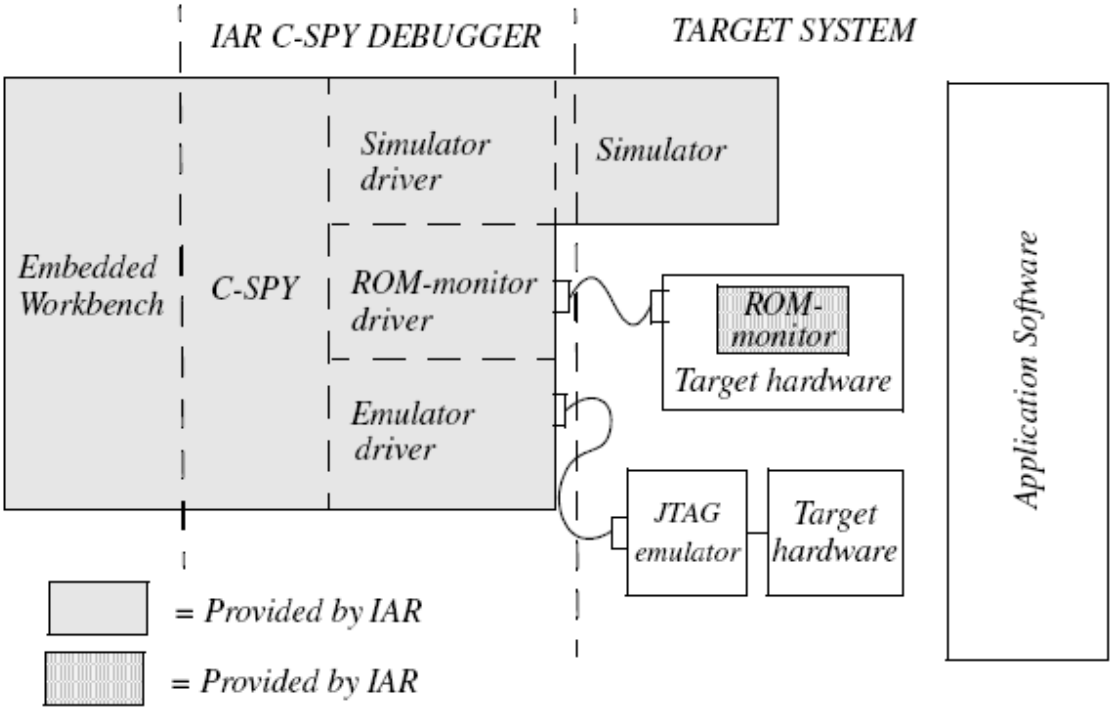


Figure 45: IAR C-SPY Debugger and target systems

Note: For information about which C-SPY drivers are available for the ARM Embedded Workbench IDE, see *IAR C-SPY® Debugger systems*, page 8.

Remappage de la mémoire

Une des caractéristiques des processeurs à base d'ARM est la capacité à remapper (configurer) ses mémoires internes. Après la mise sous tension, le contrôleur de la mémoire mappe l'adresse zéro vers la mémoire non volatile (mémoire flash). Le contrôleur de mémoire peut remapper la mémoire RAM à l'adresse zéro et la mémoire FLASH à l'adresse haute. De ce fait, la table d'exception est placée dans la mémoire RAM. En conséquence, on peut modifier facilement les codes.

On doit configurer le contrôleur de mémoire avant de télécharger les codes en utilisant les MACROS du C-SPY. Exemple de configuration de la mémoire.

```
execUserPreload()
{
  __message "Setup memory controller, do remap command\n";

  // Flash at 0x01000000, 16MB, 2 hold, 16 bits, 3 WS
  __writeMemory32(0x01002529, 0xffe00000, "Memory");

  // RAM   at 0x02000000, 16MB, 0 hold, 16 bits, 1 WS
  __writeMemory32(0x02002121, 0xffe00004, "Memory");

  // unused
  __writeMemory32(0x20000000, 0xffe00008, "Memory");

  // unused
  __writeMemory32(0x30000000, 0xffe0000c, "Memory");

  // unused
  __writeMemory32(0x40000000, 0xffe00010, "Memory");

  // unused
  __writeMemory32(0x50000000, 0xffe00014, "Memory");

  // unused
  __writeMemory32(0x60000000, 0xffe00018, "Memory");

  // unused
  __writeMemory32(0x70000000, 0xffe0001c, "Memory");


  // REMAP command
  __writeMemory32(0x00000001, 0xffe00020, "Memory");

  // standard read
  __writeMemory32(0x00000006, 0xffe00024, "Memory");
}
```

Création d'un projet

Les différentes étapes pour créer un projet d'une application sont :

1- **Créer l'environnement de travail « workspace »**
   File > New > Workspace
2- **Créer un nouveau projet**
   Project>New>Project
3- **Sauvegarder le « workspace »**
4- **Ajouter un fichier au projet**
   Project > Add Files
   Configurer les option du projet
   Project>Options

Créer un nouveau projet et un nouveau workspace

1- Créer un nouveau répertoire (e.g. TestIar) comportant les sous répertoires suivants :
   a. IncAtmel : source des fichiers includes de l'ATMEL
   b. Src : fichiers sources de votre programme d'application
   c. Linkconfig : fichiers pour l'éditeur des liens
   d. SrcIAR : fichiers sources de l'IDE IAR.

Aller dans le répertoire :

C:\ProgramFiles\IARSystems\EmbeddedWorkbench5.0Kickstart\ARM\examples\Atmel\SAM7S256\AT91SAM7S-BasicTools

Les sous répertoires de ...\ AT91SAM7S-BasicTools\Compil sont :

- SrcIar,
- Bin,
- Resource,
- Setting etc.

Copier tous les fichiers du répertoire SrcIar dans le répertoire …/TestIar/SrcIar

Copier tous les fichiers du répertoire Resource dans le répertoire …/TestIar/LinkConfig

Aller au sous répertoire … AT91SAM7S-BasicTools\src

Copier le fichier main.c et le sauvegarder dans le répertoire …/TestIar/Src

Aller au sous répertoire … Atmel\SAM7S256\Include

Copier tous les fichiers et les sauvegarder dans le répertoire …/TestIar/IncAtmel

Copier le fichier Board.h du répertoire …/TestIar/SrcIar dans le répertoire …/TestIar/IncAtmel.

Aller au répertoire ...\ AT91SAM7S-BasicTools\Compil double clicker sur le fichier Basic.eww (Lancer IAR Embedded Workbench). La fenêtre du workbench est ouverte.

Dans le meu bar sélectionne project>rebuild (recompiler le projet).

La carte de développement est connectée à l'ordinateur hôte à travers 2 câble USB.

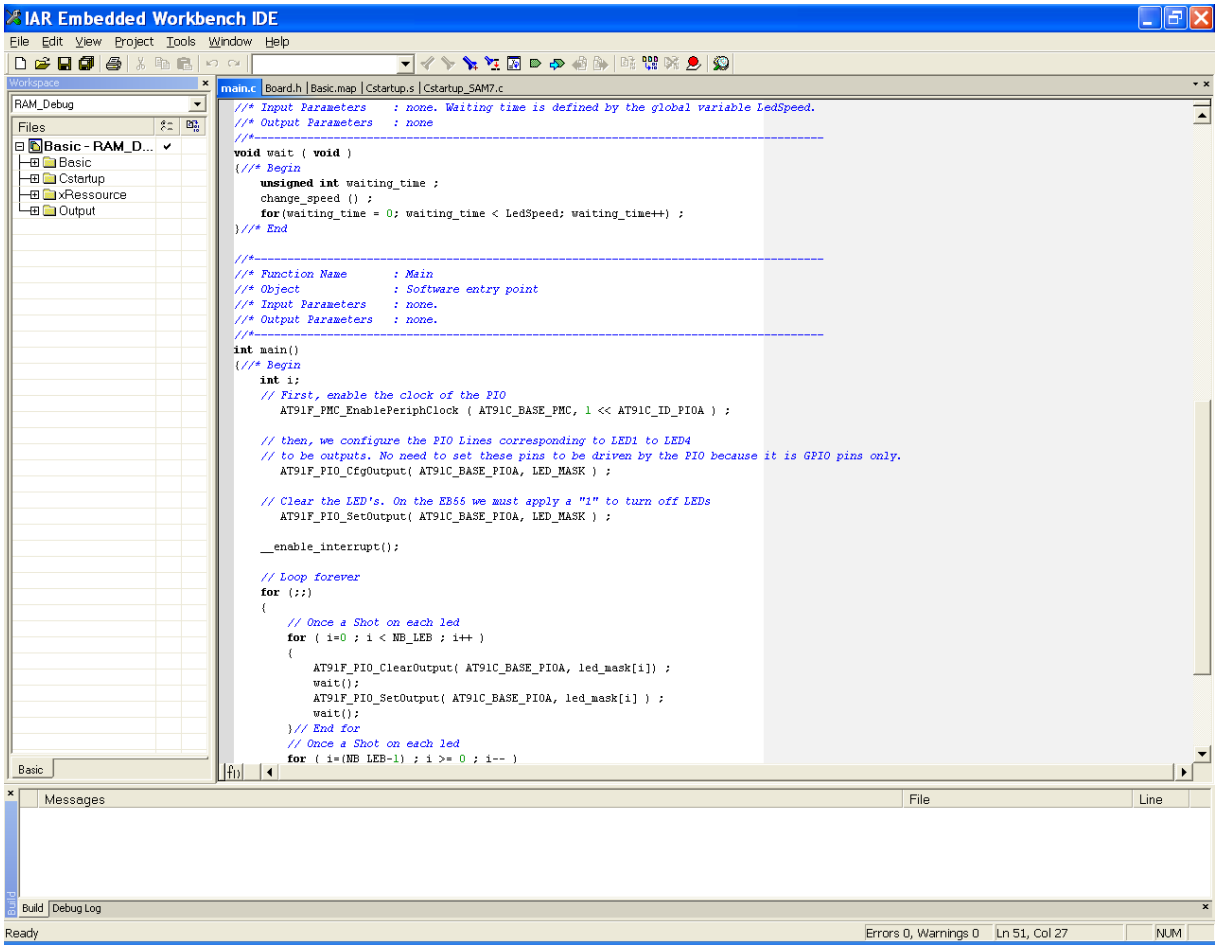
Dans le menu bar sélectionne le project >Debug

Le code exécutable est téléchargé dans le processeur.

Exécuter le programme en clickant sur le symbol →→→ sous le menu bar.

Les diodes de la carte s'allume une à une.

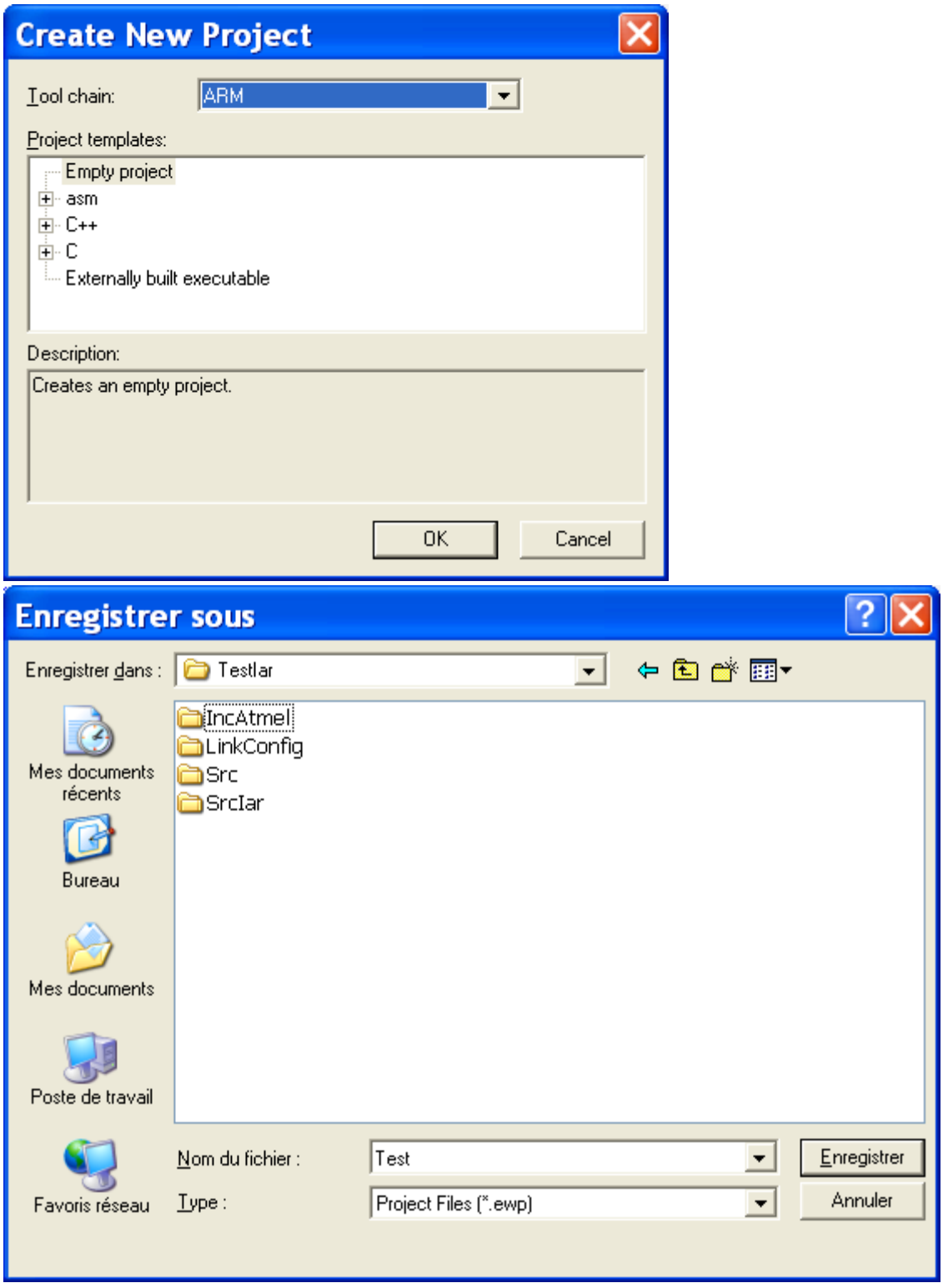
Pour passer en mode compilation clicker sur le symbol (croix rouge sous le menu bar).



NB : L'IDE IAR crée automatiquement deux sous répertoires suivants :

a. Debug : source des fichiers dédiés à la mise au point
b. Setting : fichiers de configuration de l'IDE IAR.

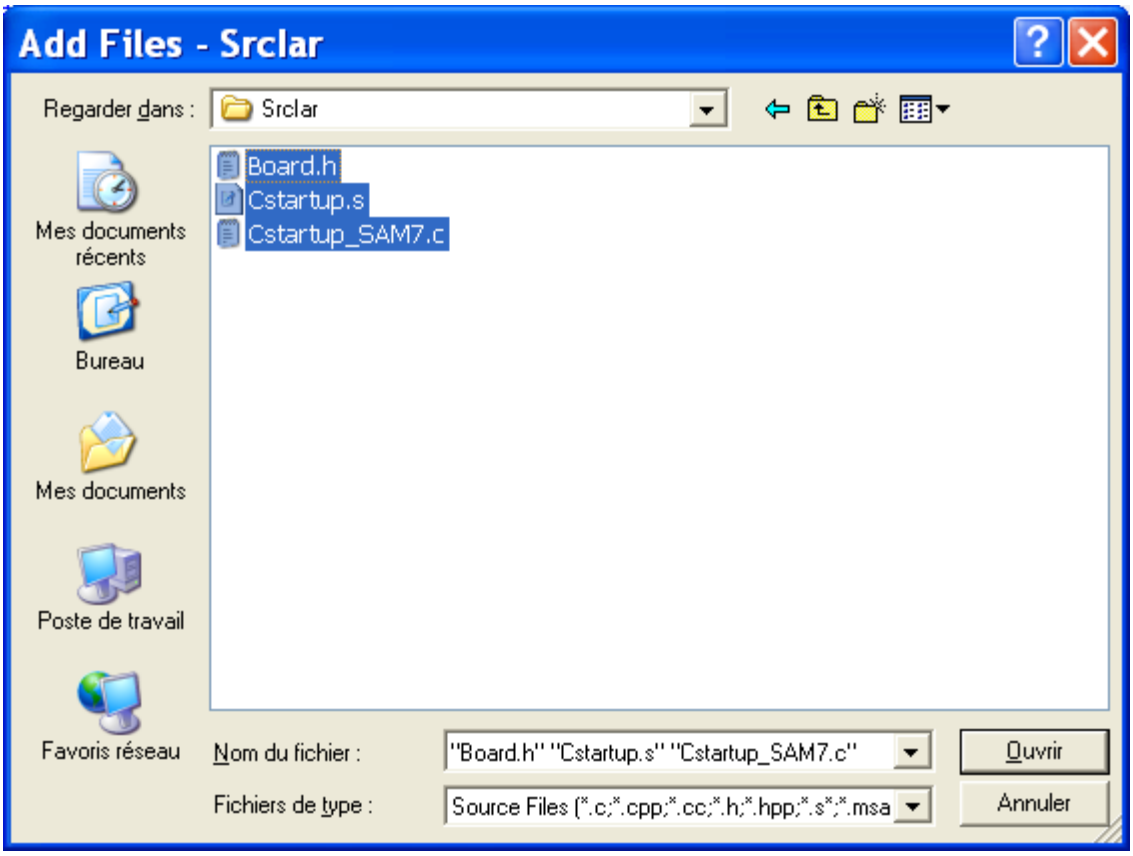
2- Quitter IAR Embedded Workbench
3- Lancer IARWorkBench à partir partir tous les programmes.
4- File > crerate > new workspace
En mode d'édition de programme.
5- PROJECT > CREATE NEW PROJECT



Créer un nouveau projet dans le répertoire Testlar

Clicker sur Test-Debug avec la clé droite

a. ADD group (e.g. Src) Ajouter les sous répertoires du Testlar.
b. Pour ajouter les fichiers dans chaque sous répertoire faire :
c. Sélectionner le sous réperoire correspondant et faire
d. ADD file

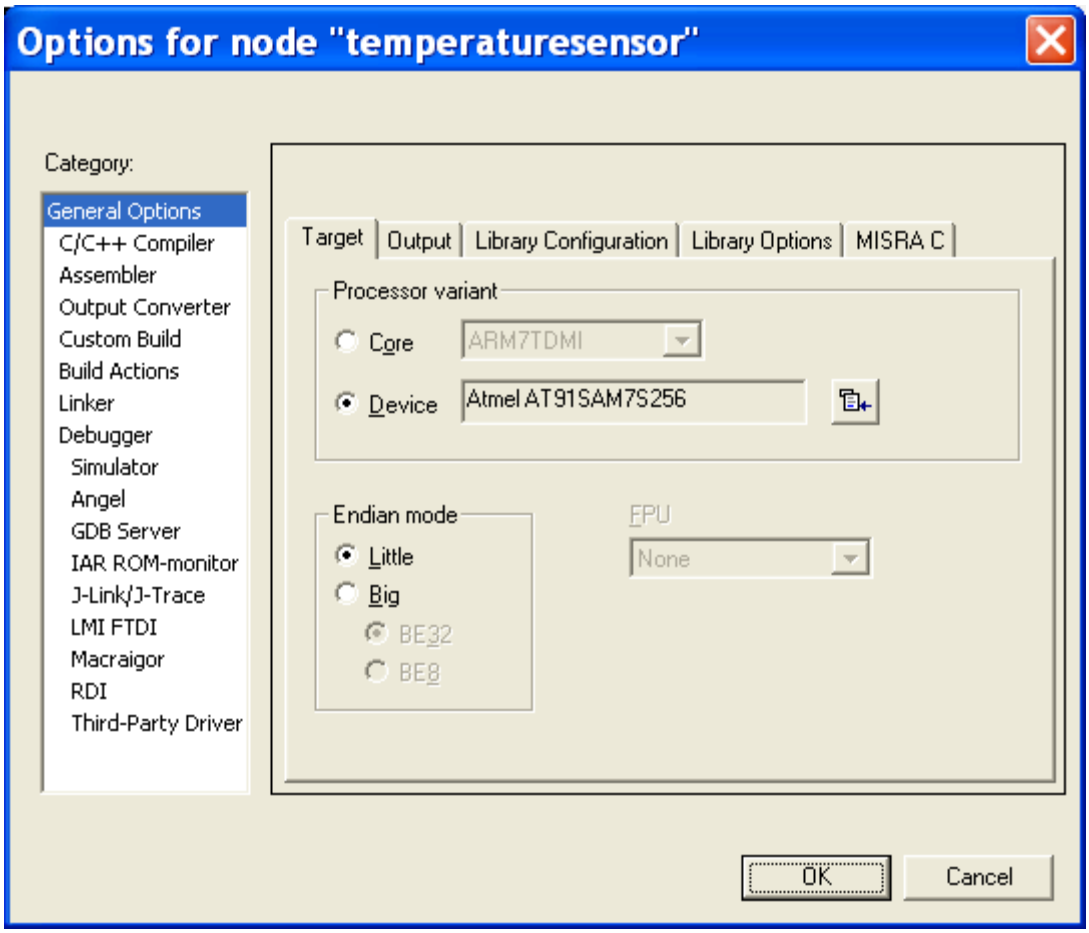


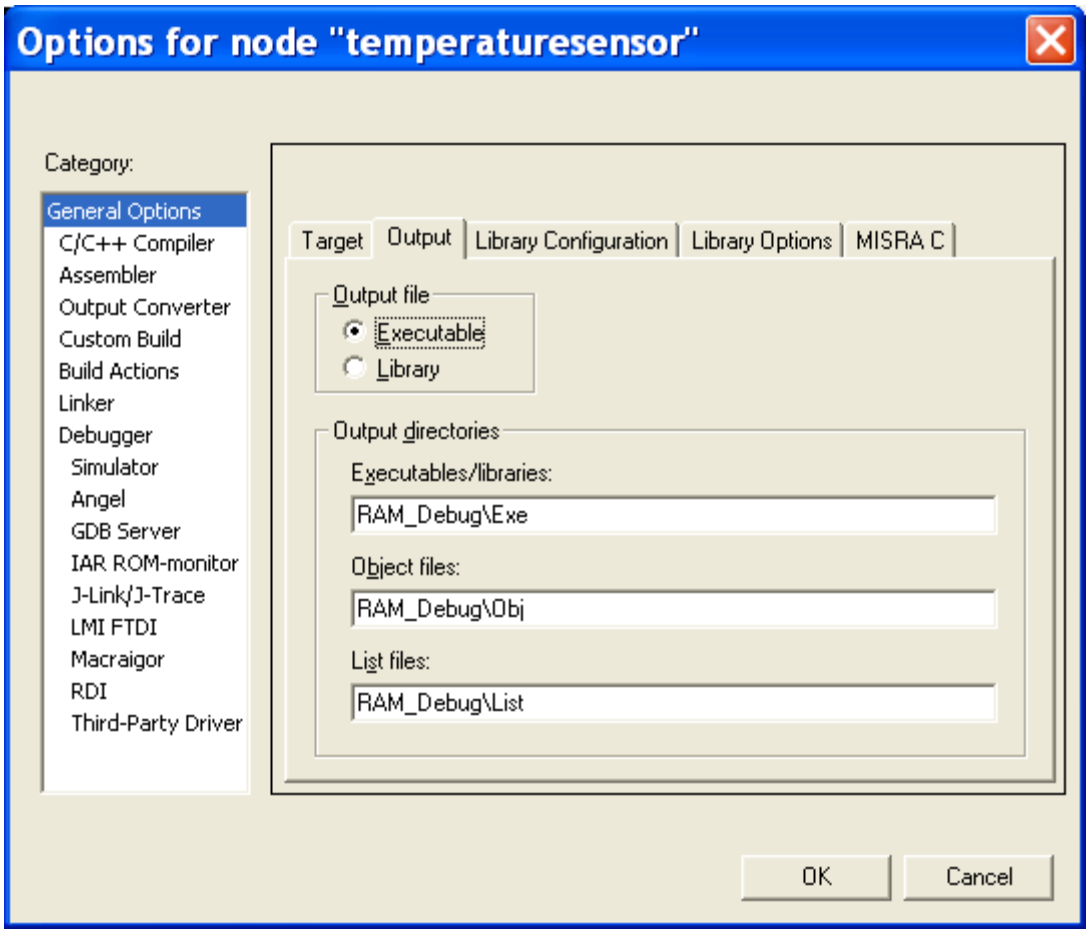
6- PROJECT > OPTION
   a. Initialiser tous les paramètres du projet
   b. Save workspace.

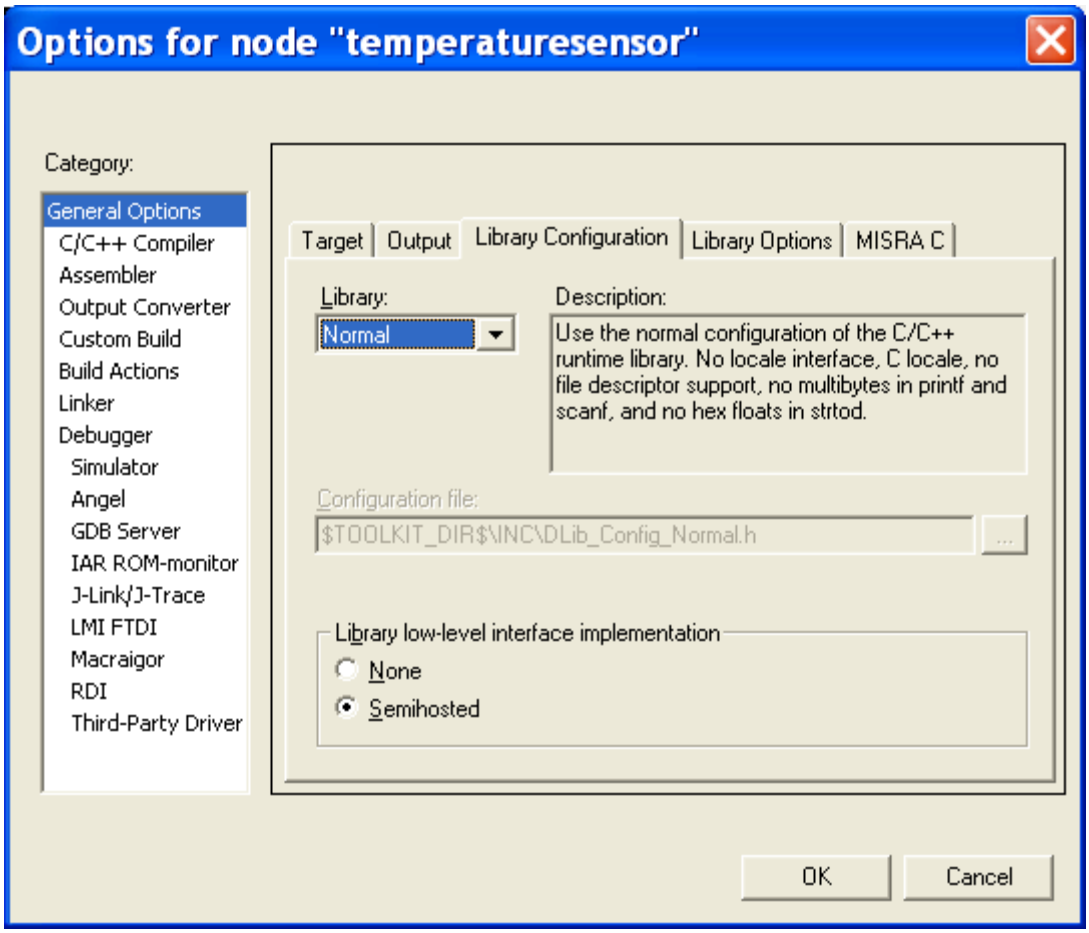
Configuration de l'environnement IDE

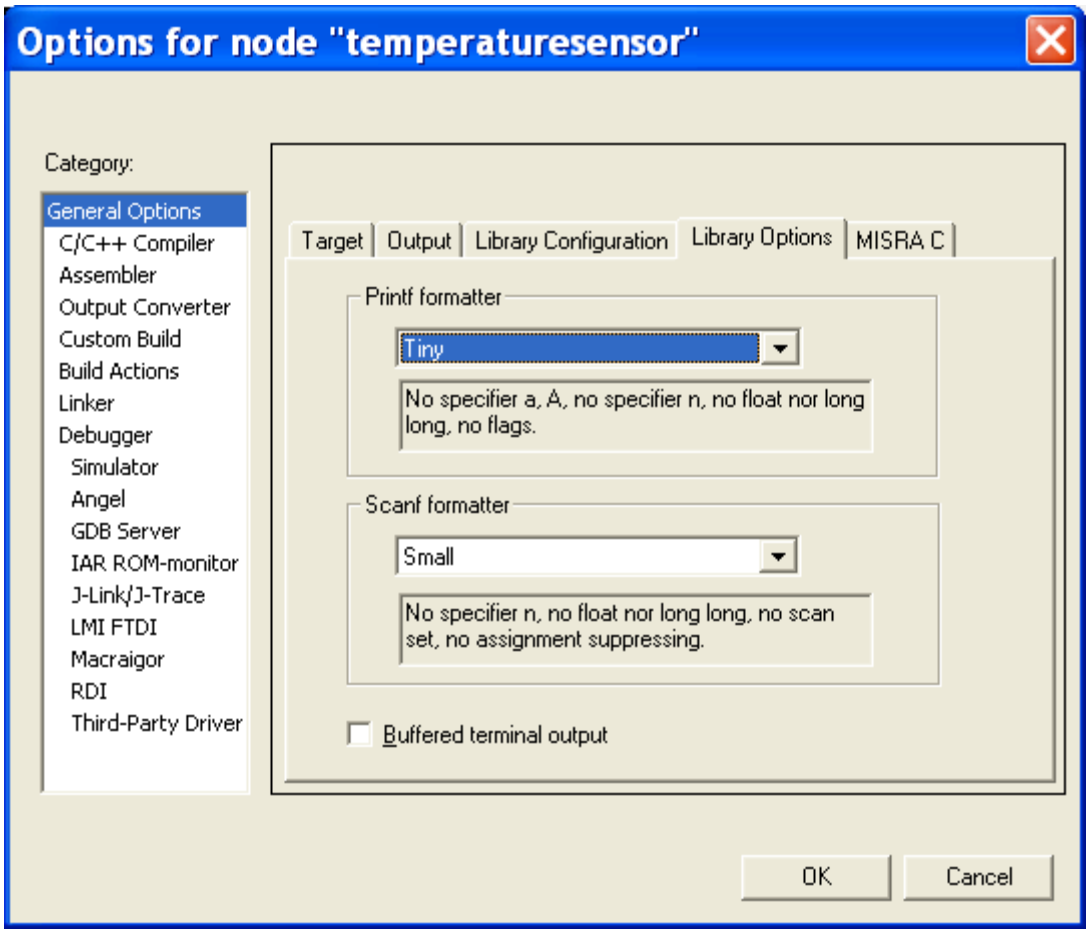
Clicker sur Test-Debug

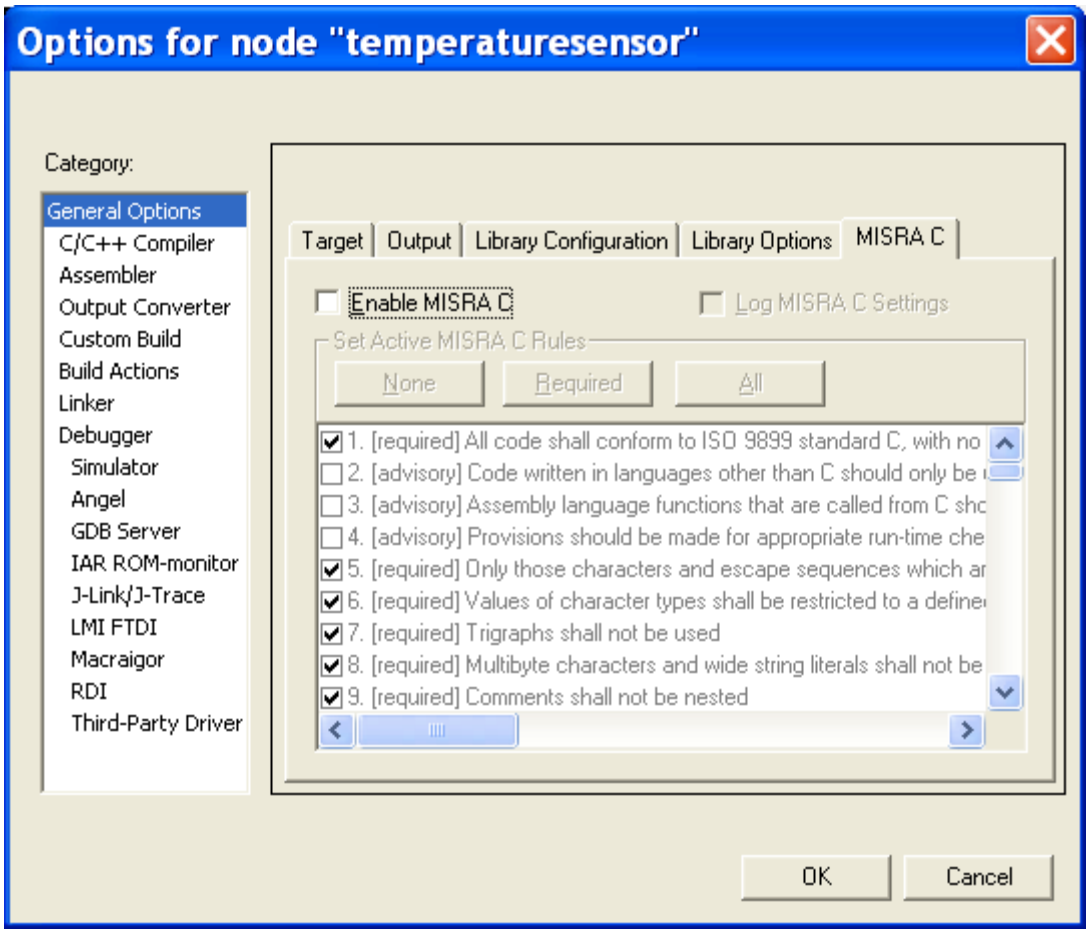
Sélectionner le projet> project > option

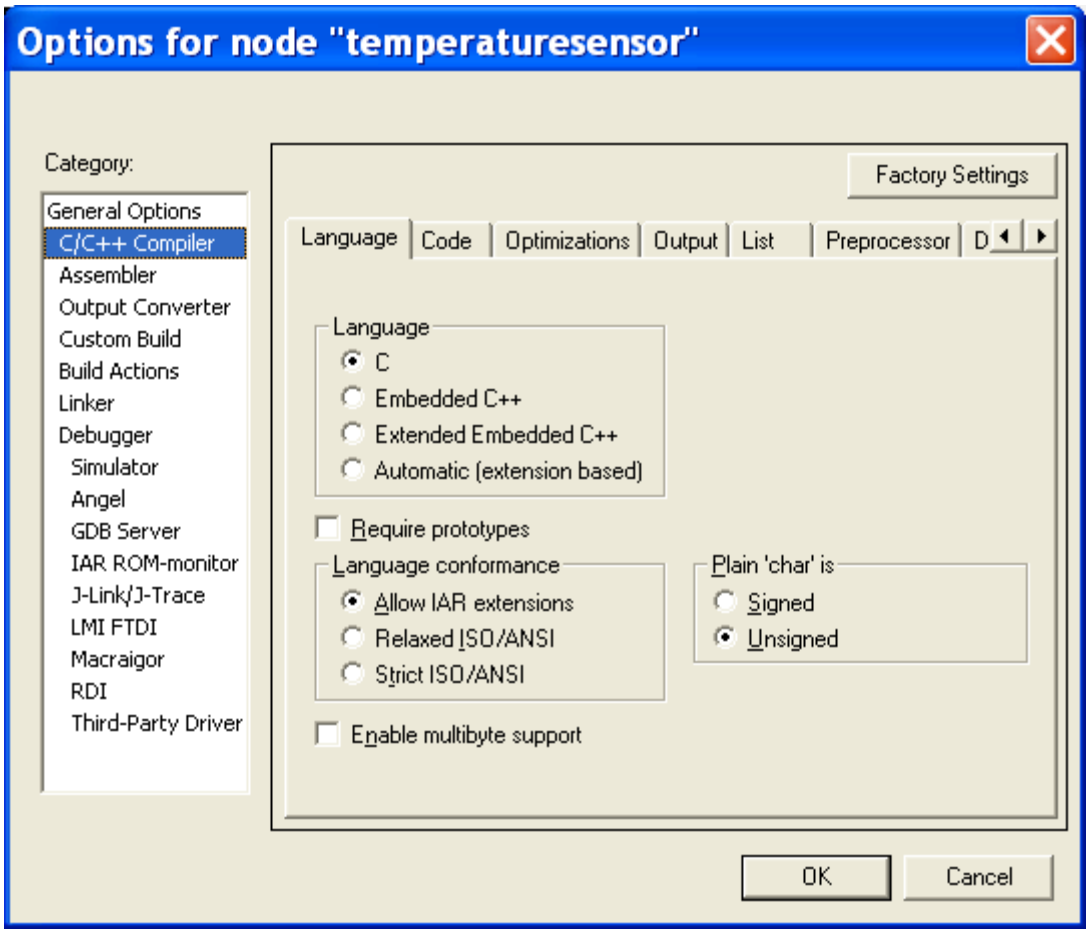

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Target
Output
Library Configuration
Library Options
MISRA C
Processor variant
Core
ARM7TDMI
Device
Atmel AT91SAM7S256
Endian mode
Little
Big
BE32
BE8
FPU
None
OK
Cancel

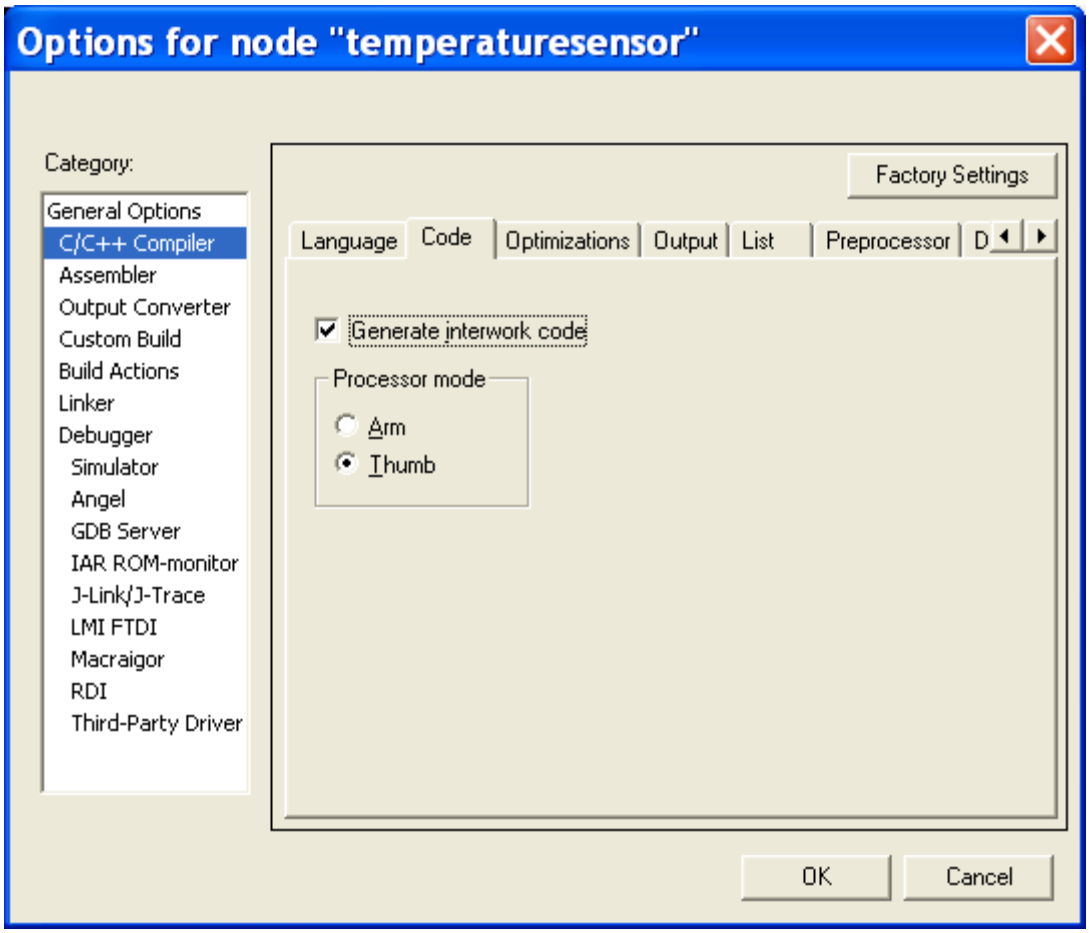

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Target
Output
Library Configuration
Library Options
MISRA C
Output file
Executable
Library
Output directories
Executables/libraries:
RAM_Debug\Exe
Object files:
RAM_Debug\Obj
List files:
RAM_Debug\List
OK
Cancel

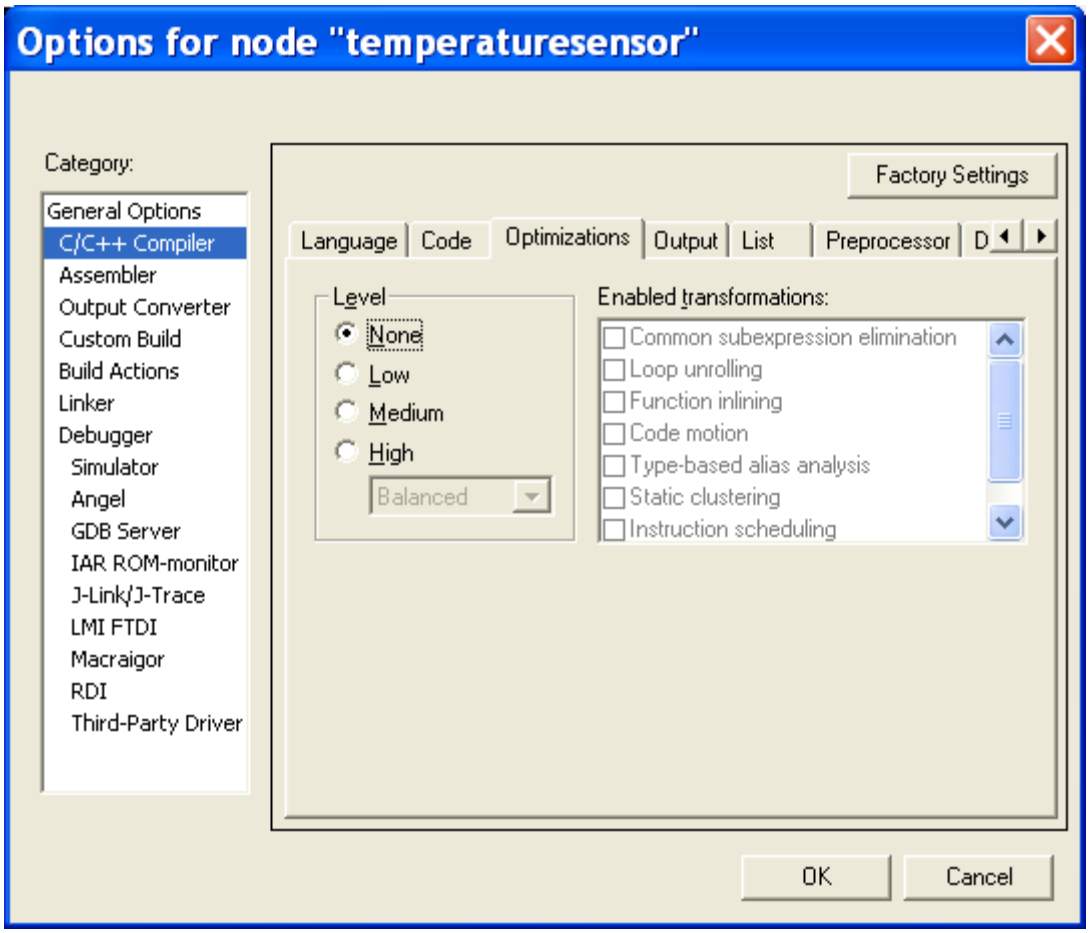

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Target
Output
Library Configuration
Library Options
MISRA C
Library:
Normal
Description:
Use the normal configuration of the C/C++ runtime library. No locale interface, C locale, no file descriptor support, no multibytes in printf and scanf, and no hex floats in strtod.
Configuration file:
$TOOLKIT_DIR$\INC\DLib_Config_Normal.h
...
Library low-level interface implementation
None
Semihosted
OK
Cancel

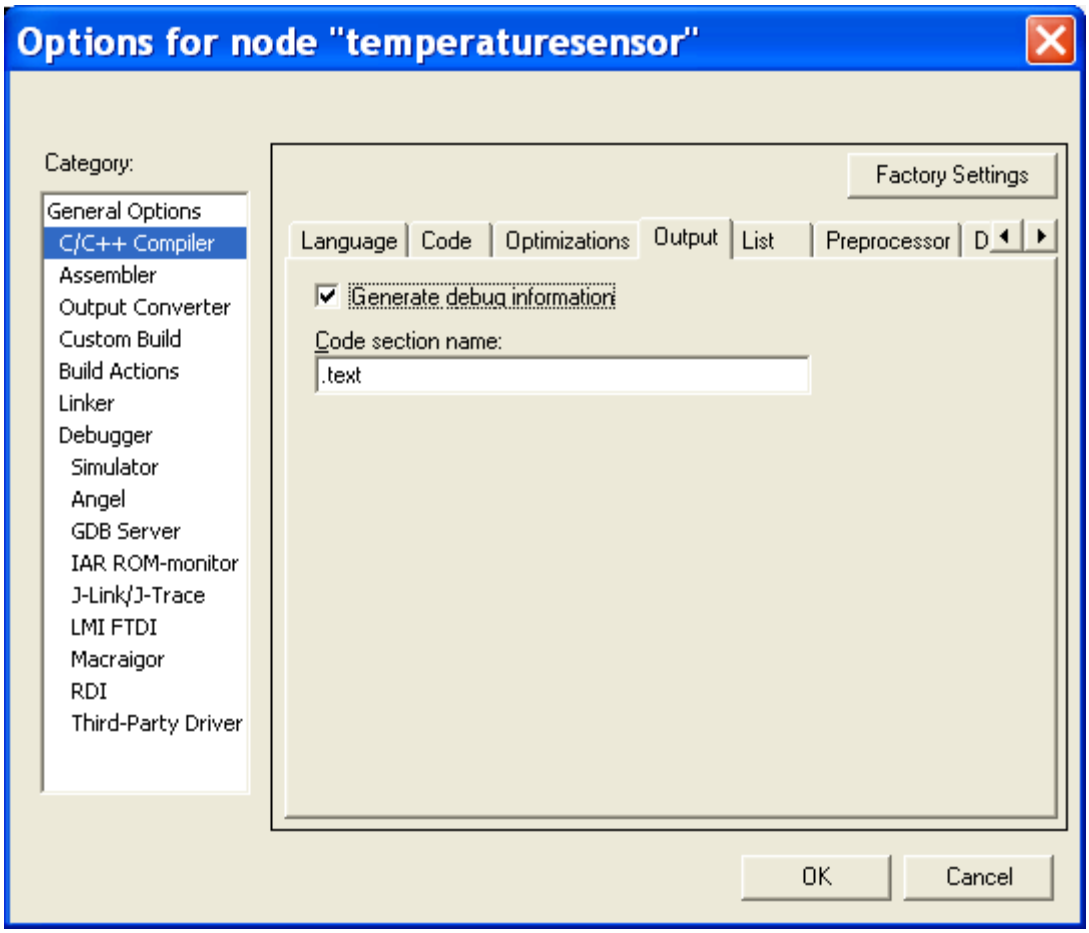

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Target
Output
Library Configuration
Library Options
MISRA C
Printf formatter
Tiny
No specifier a, A, no specifier n, no float nor long long, no flags.
Scanf formatter
Small
No specifier n, no float nor long long, no scan set, no assignment suppressing.
Buffered terminal output
OK
Cancel

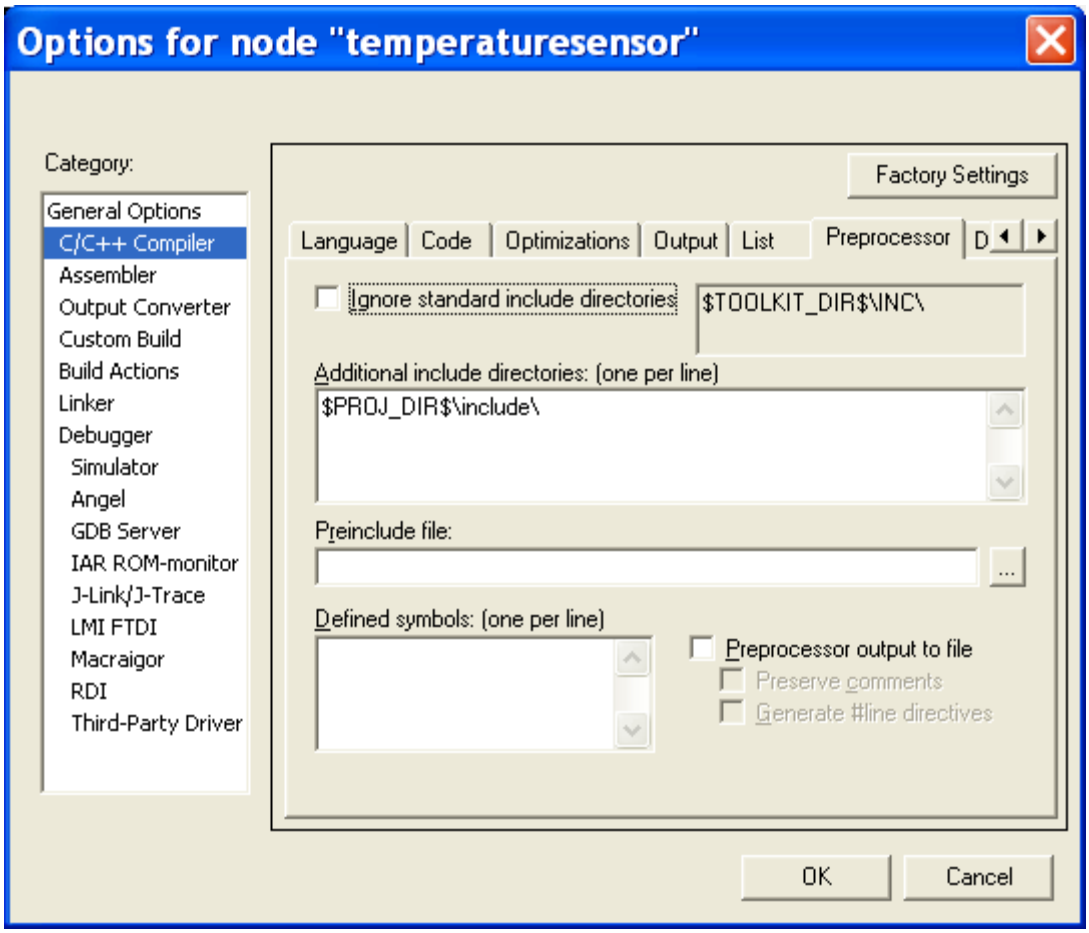

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Target
Output
Library Configuration
Library Options
MISRA C
Enable MISRA C
Log MISRA C Settings
Set Active MISRA C Rules
None
Required
All
1. [required] All code shall conform to ISO 9899 standard C, with no
2. [advisory] Code written in languages other than C should only be
3. [advisory] Assembly language functions that are called from C sho
4. [advisory] Provisions should be made for appropriate run-time che
5. [required] Only those characters and escape sequences which ar
6. [required] Values of character types shall be restricted to a define
7. [required] Trigraphs shall not be used
8. [required] Multibyte characters and wide string literals shall not be
9. [required] Comments shall not be nested
OK
Cancel

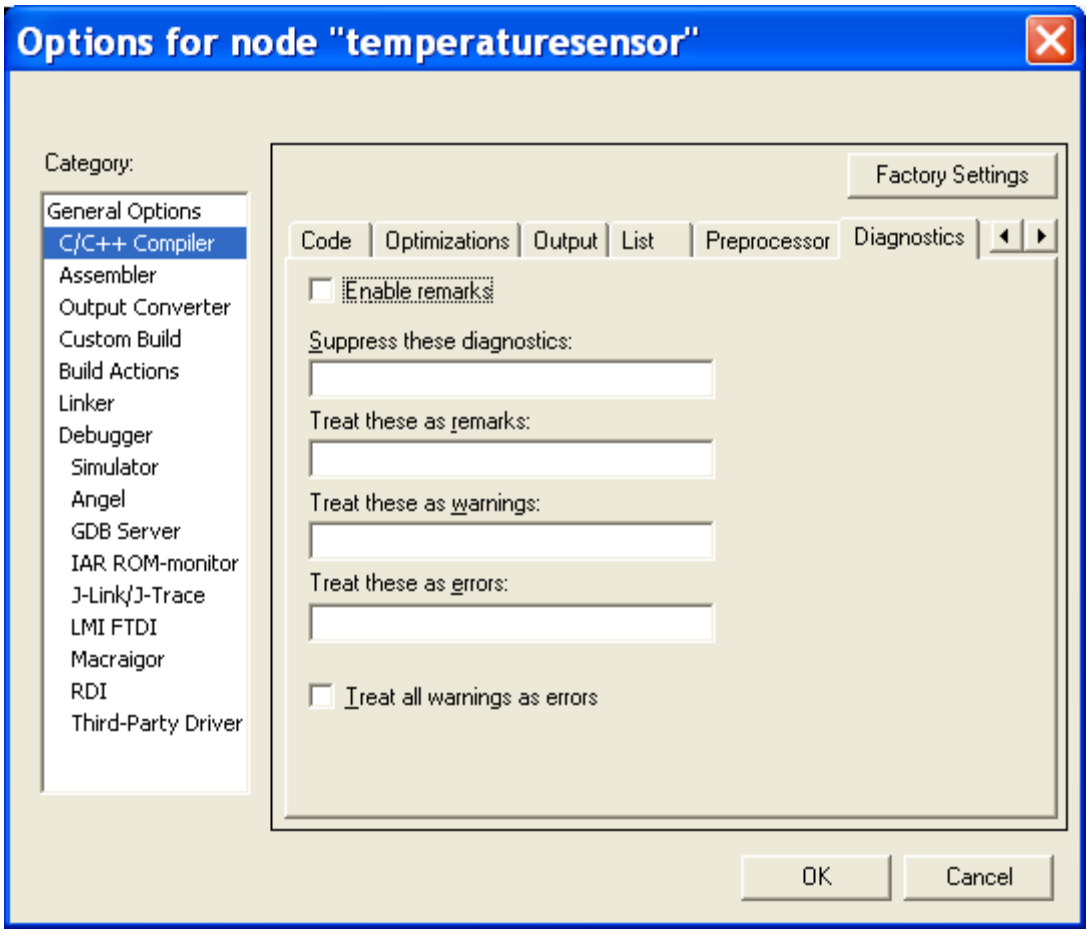

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Language
Code
Optimizations
Output
List
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
OK
Cancel

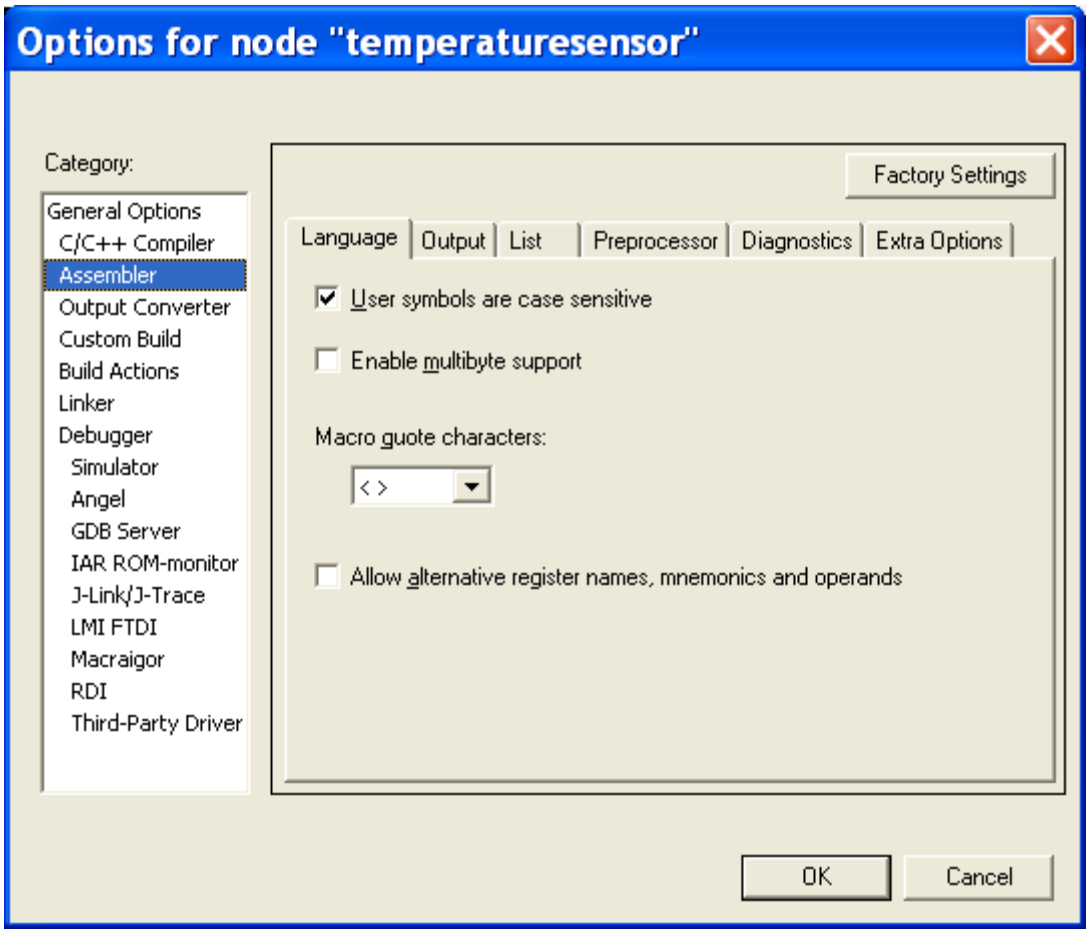

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Language
Code
Optimizations
Output
List
Preprocessor
Generate interwork code
Processor mode
Arm
Thumb
OK
Cancel

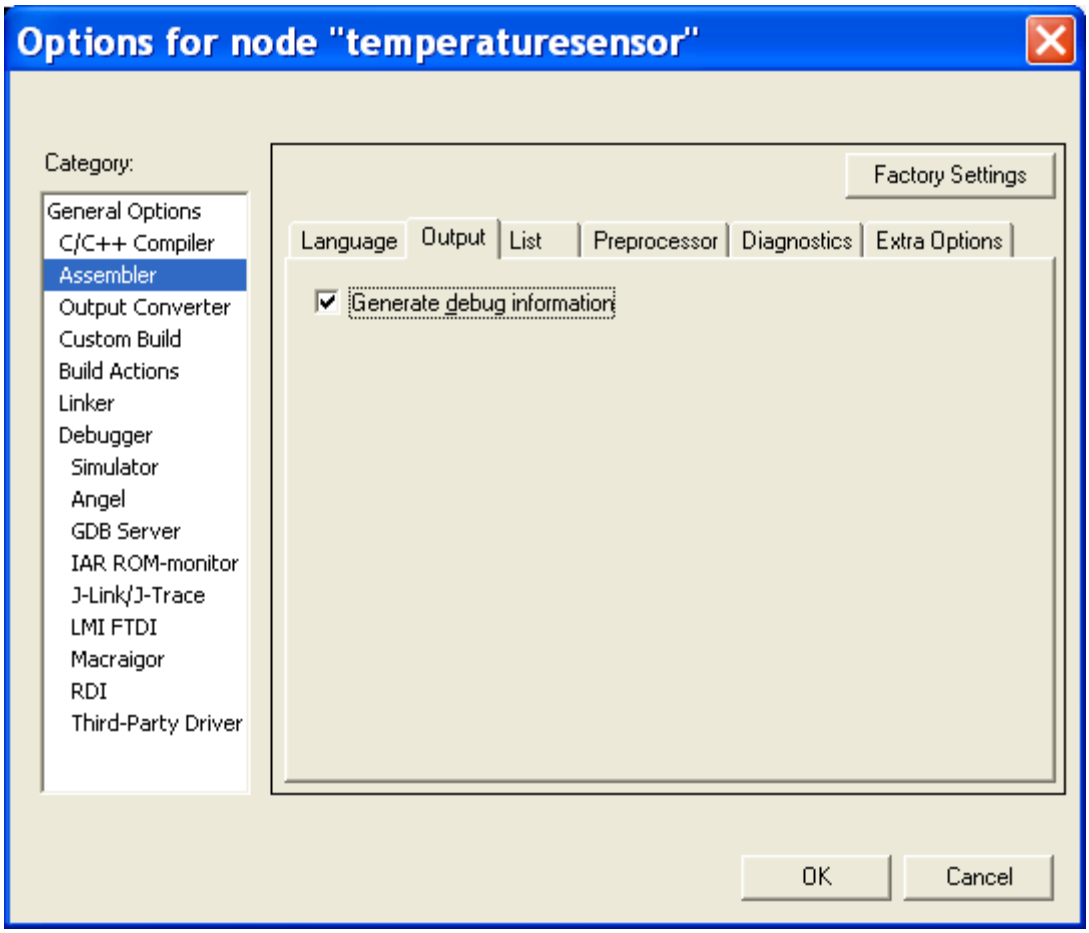

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Language
Code
Optimizations
Output
List
Preprocessor
Level
None
Low
Medium
High
Balanced
Enabled transformations:
Common subexpression elimination
Loop unrolling
Function inlining
Code motion
Type-based alias analysis
Static clustering
Instruction scheduling
OK
Cancel

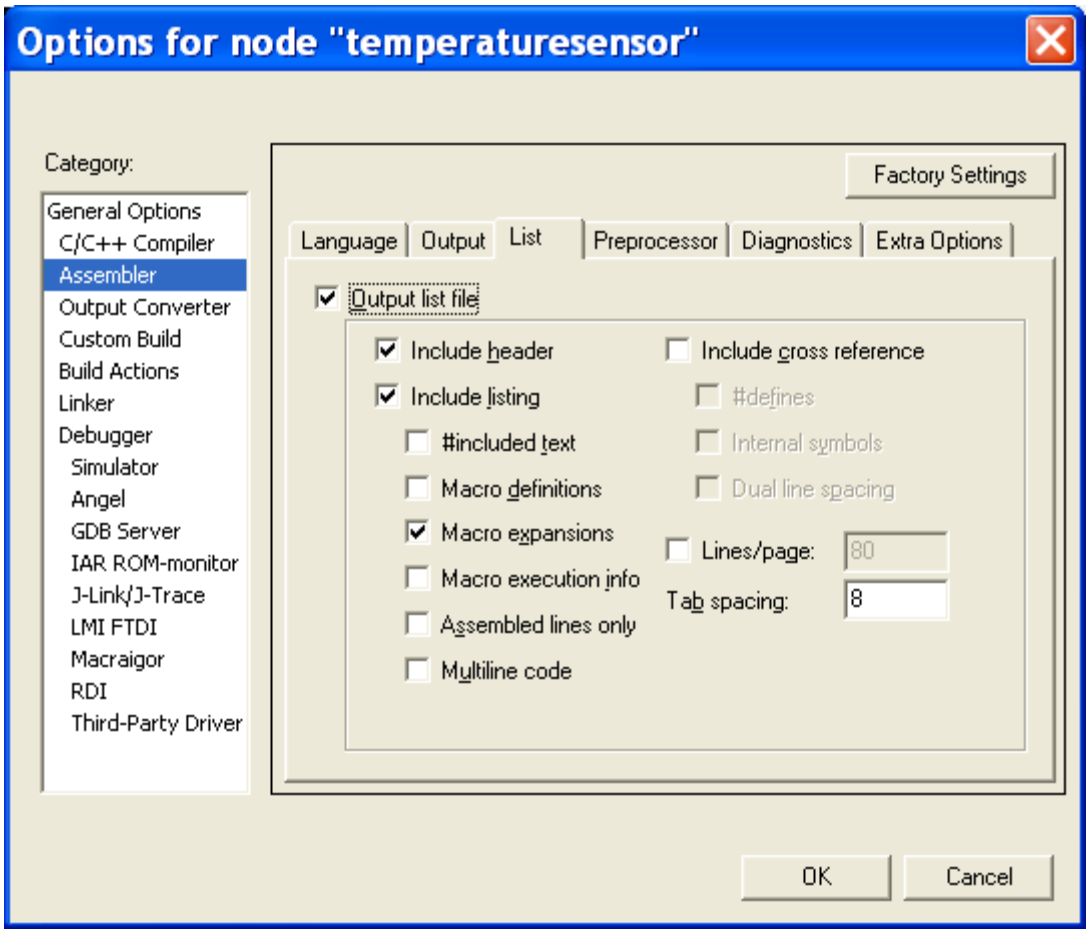

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Language
Code
Optimizations
Output
List
Preprocessor
Generate debug information
Code section name:
.text
OK
Cancel

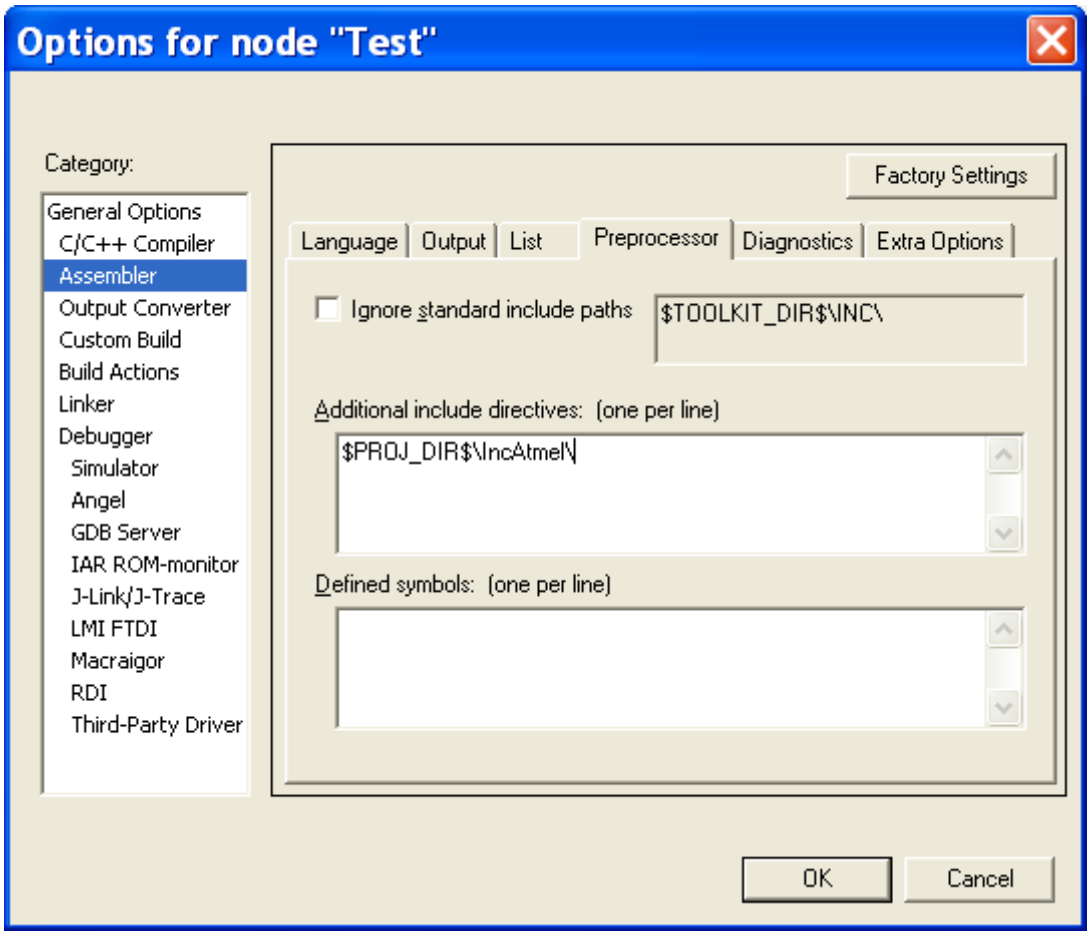

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Language
Code
Optimizations
Output
List
Preprocessor
Ignore standard include directories
$TOOLKIT_DIR$\INC\
Additional include directories: (one per line)
$PROJ_DIR$\include\
Preinclude file:
Defined symbols: (one per line)
Preprocessor output to file
Preserve comments
Generate #line directives
OK
Cancel

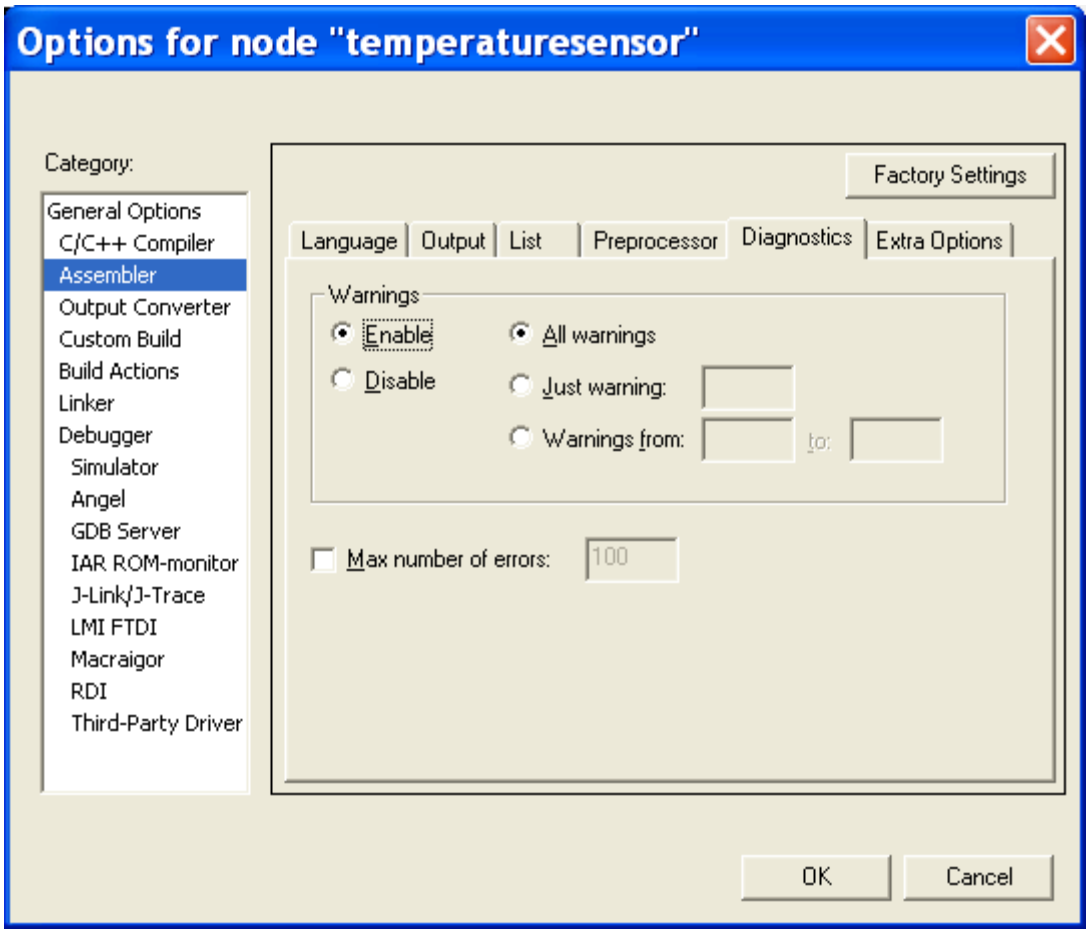

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Code
Optimizations
Output
List
Preprocessor
Diagnostics
Enable remarks
Suppress these diagnostics:
Treat these as remarks:
Treat these as warnings:
Treat these as errors:
Treat all warnings as errors
OK
Cancel

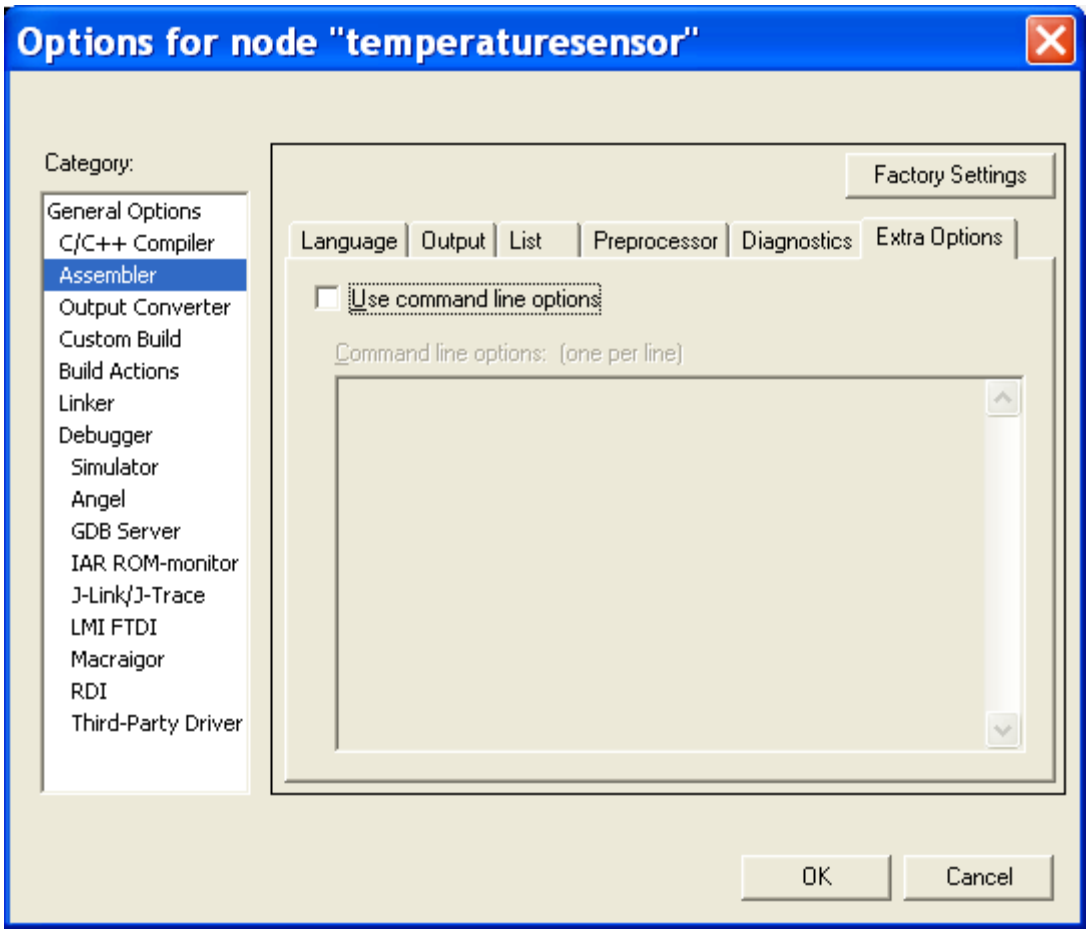

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Language
Output
List
Preprocessor
Diagnostics
Extra Options
User symbols are case sensitive
Enable multibyte support
Macro quote characters:
< >
Allow alternative register names, mnemonics and operands
OK
Cancel

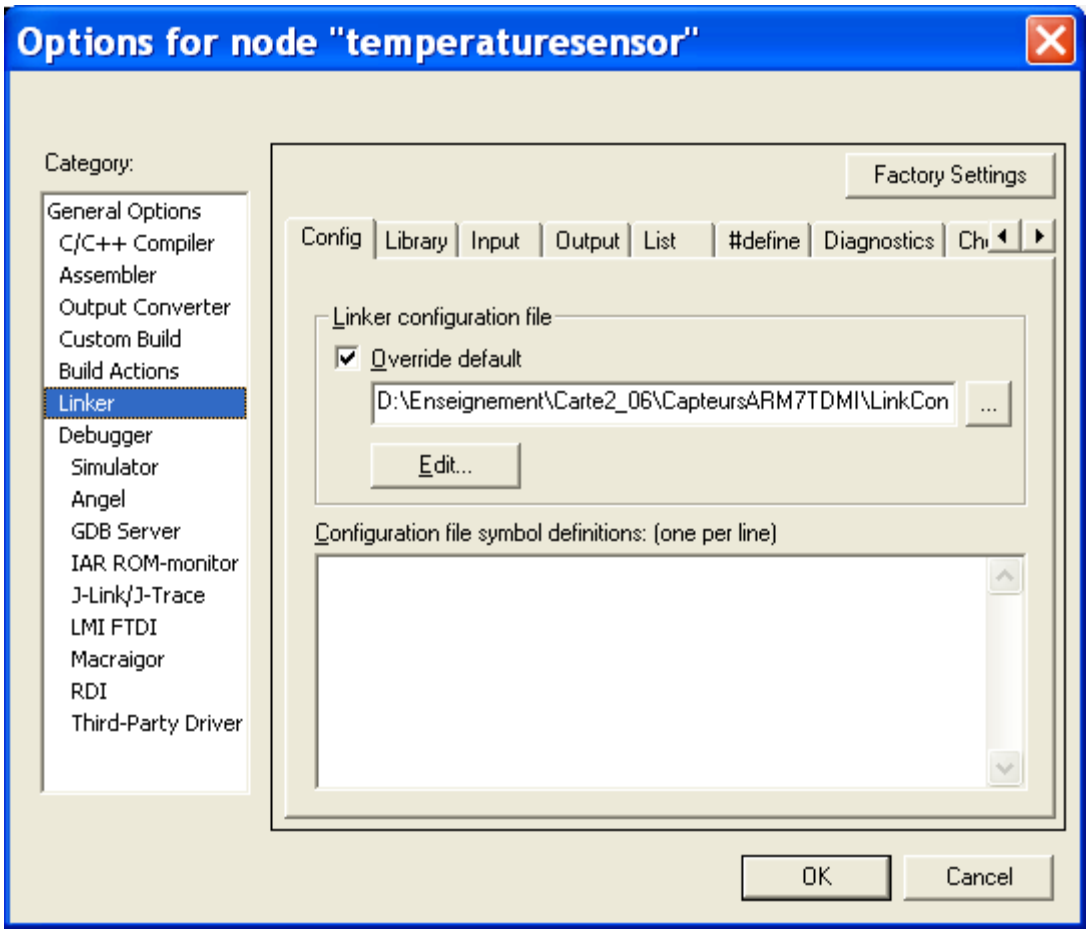

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Language
Output
List
Preprocessor
Diagnostics
Extra Options
Generate debug information
OK
Cancel

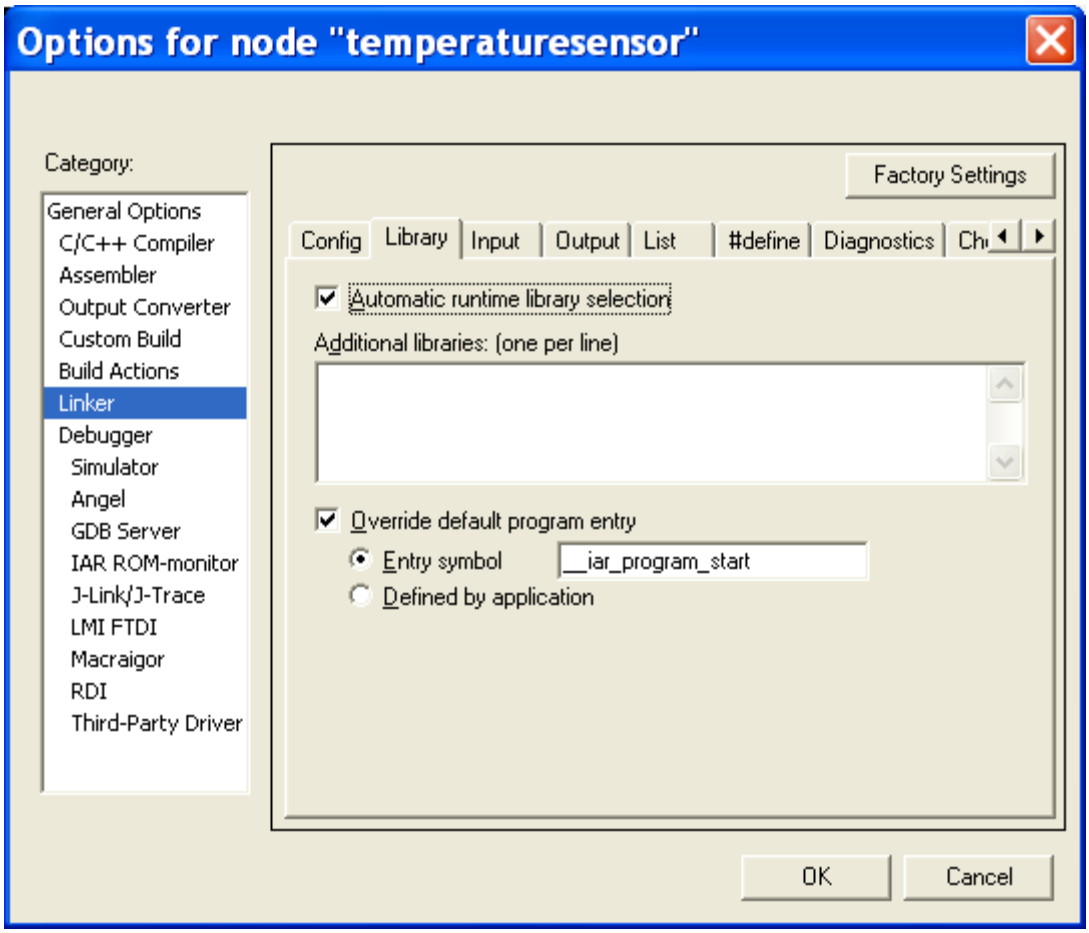

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Language
Output
List
Preprocessor
Diagnostics
Extra Options
Output list file
Include header
Include listing
#included text
Macro definitions
Macro expansions
Macro execution info
Assembled lines only
Multiline code
Include cross reference
#defines
Internal symbols
Dual line spacing
Lines/page:
80
Tab spacing:
8
OK
Cancel

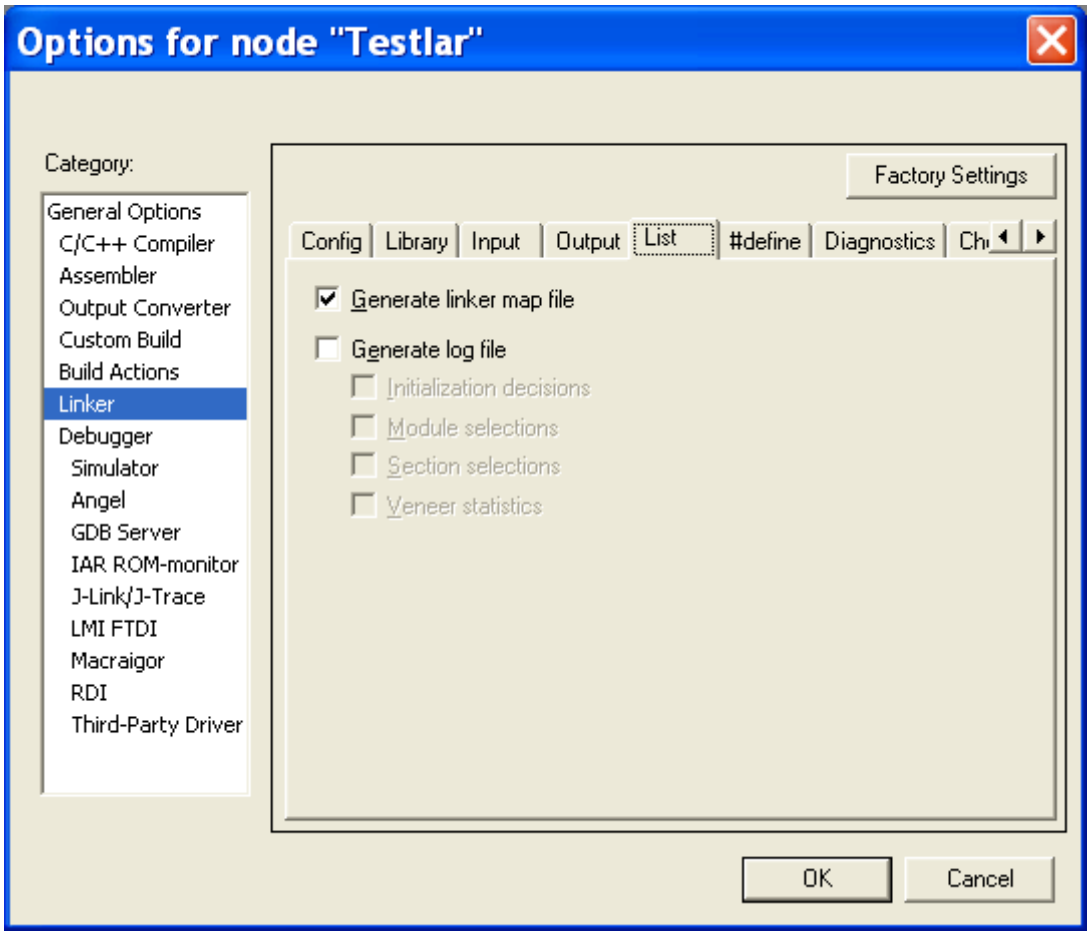

Options for node "Test"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Language
Output
List
Preprocessor
Diagnostics
Extra Options
Ignore standard include paths
$TOOLKIT_DIR$\INC\
Additional include directives: (one per line)
$PROJ_DIR$\IncAtmel\
Defined symbols: (one per line)
OK
Cancel

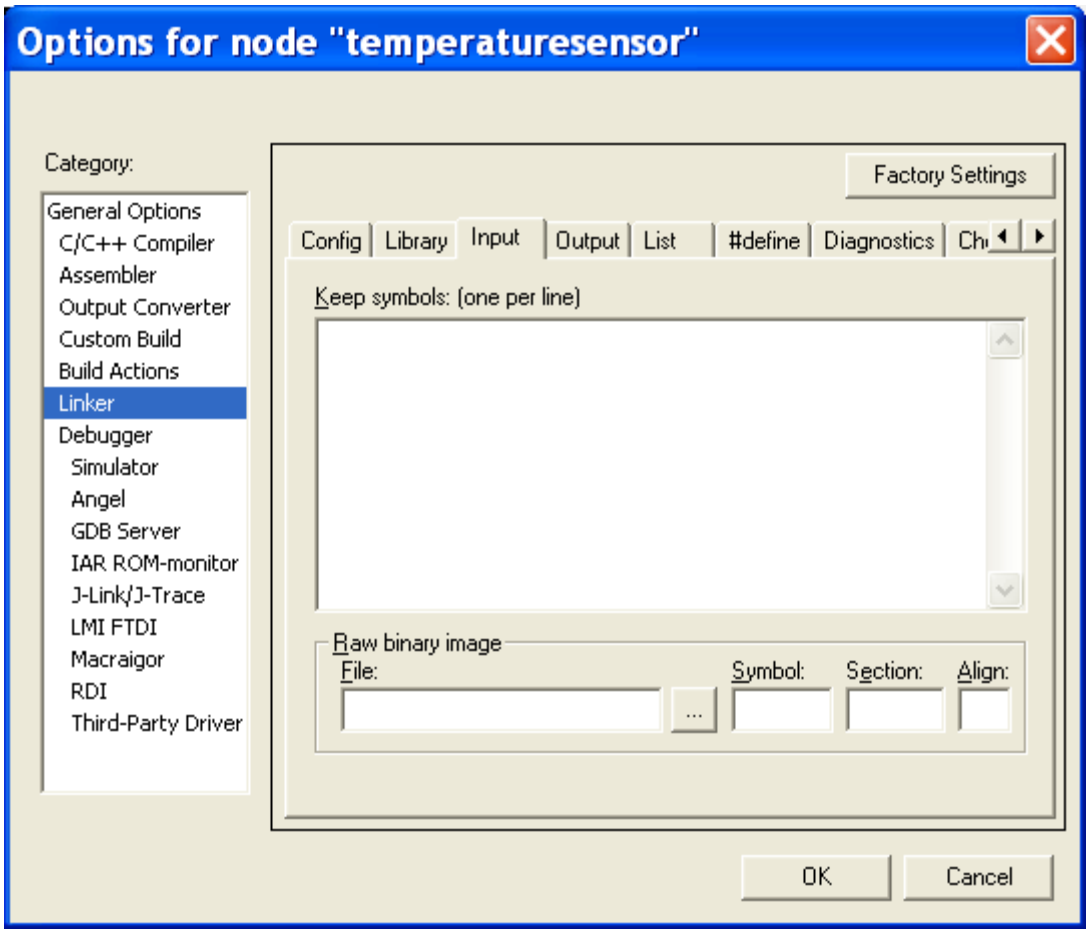

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Language
Output
List
Preprocessor
Diagnostics
Extra Options
Warnings
Enable
Disable
All warnings
Just warning:
Warnings from:
to:
Max number of errors:
100
OK
Cancel

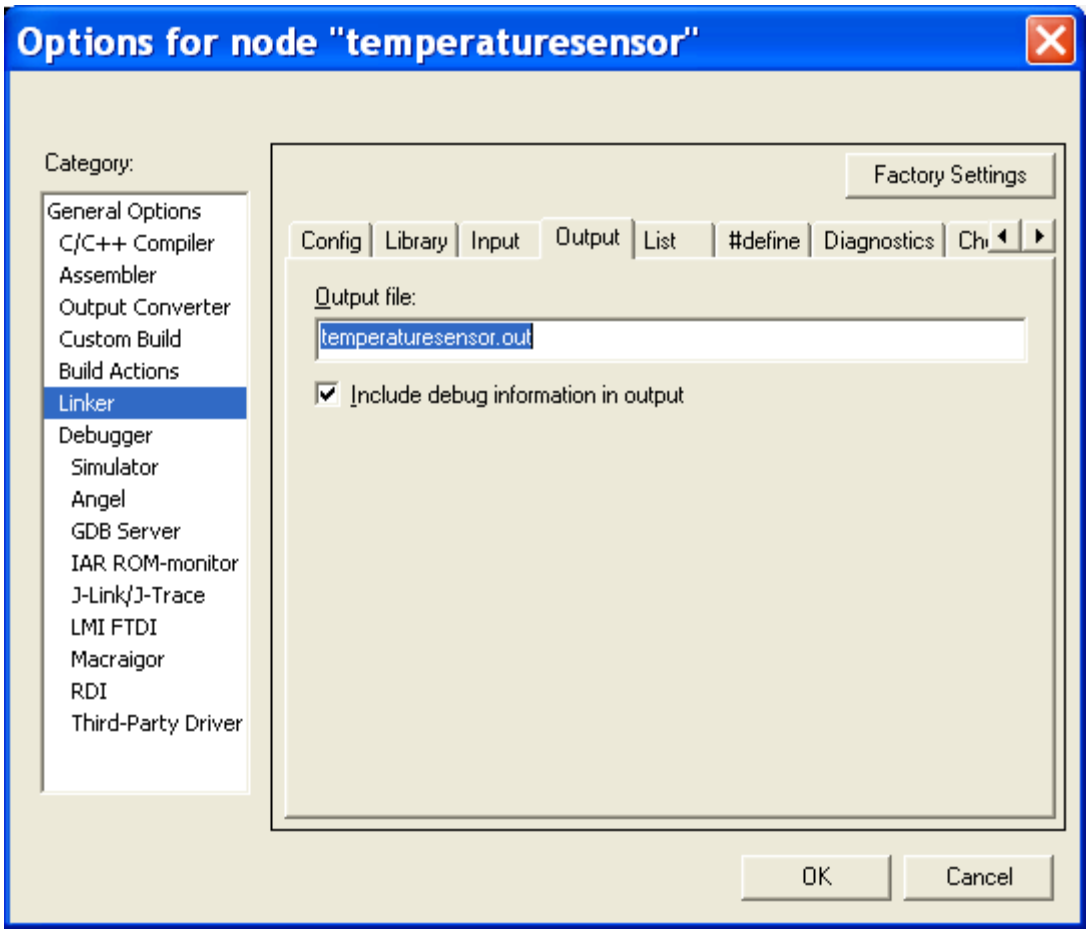

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Language
Output
List
Preprocessor
Diagnostics
Extra Options
Use command line options
Command line options: (one per line)
OK
Cancel

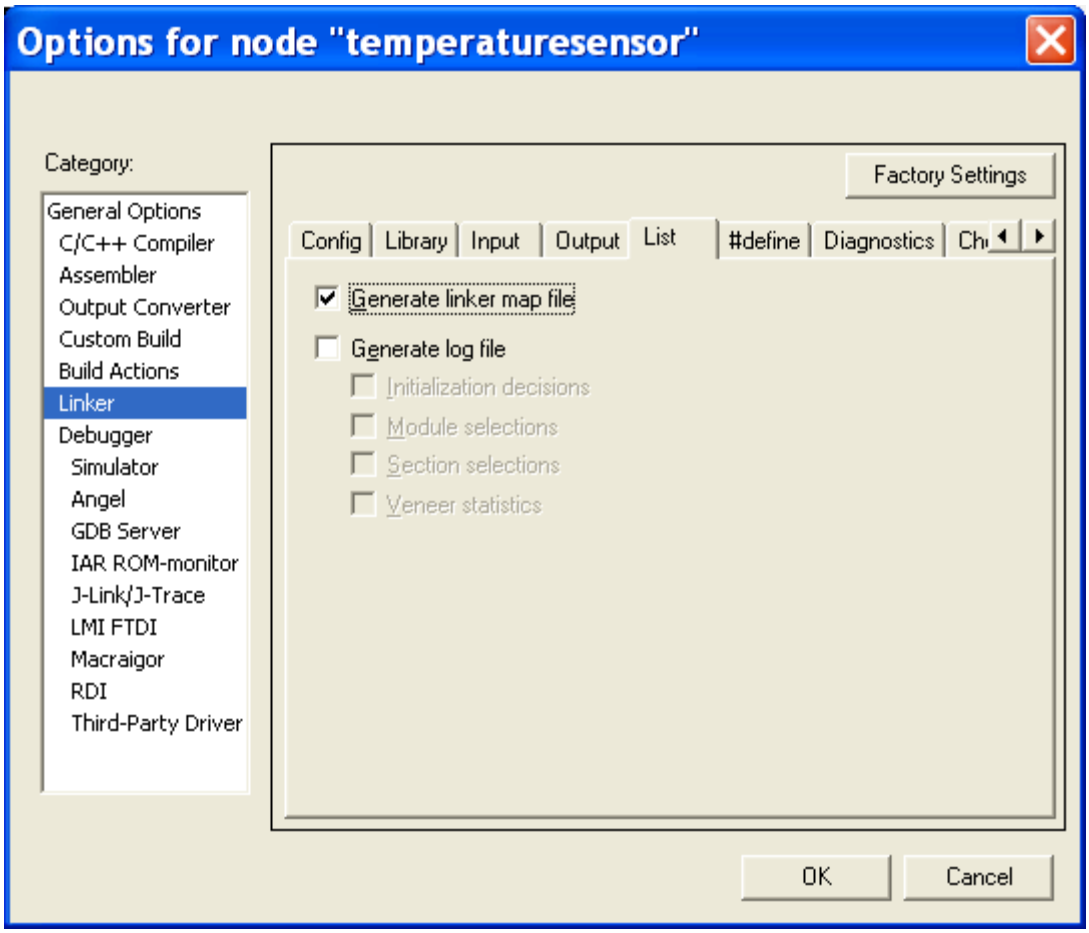

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Config
Library
Input
Output
List
#define
Diagnostics
Linker configuration file
Override default
D:\Enseignement\Carte2_06\CapteursARM7TDMI\LinkCon
...
Edit...
Configuration file symbol definitions: (one per line)
OK
Cancel

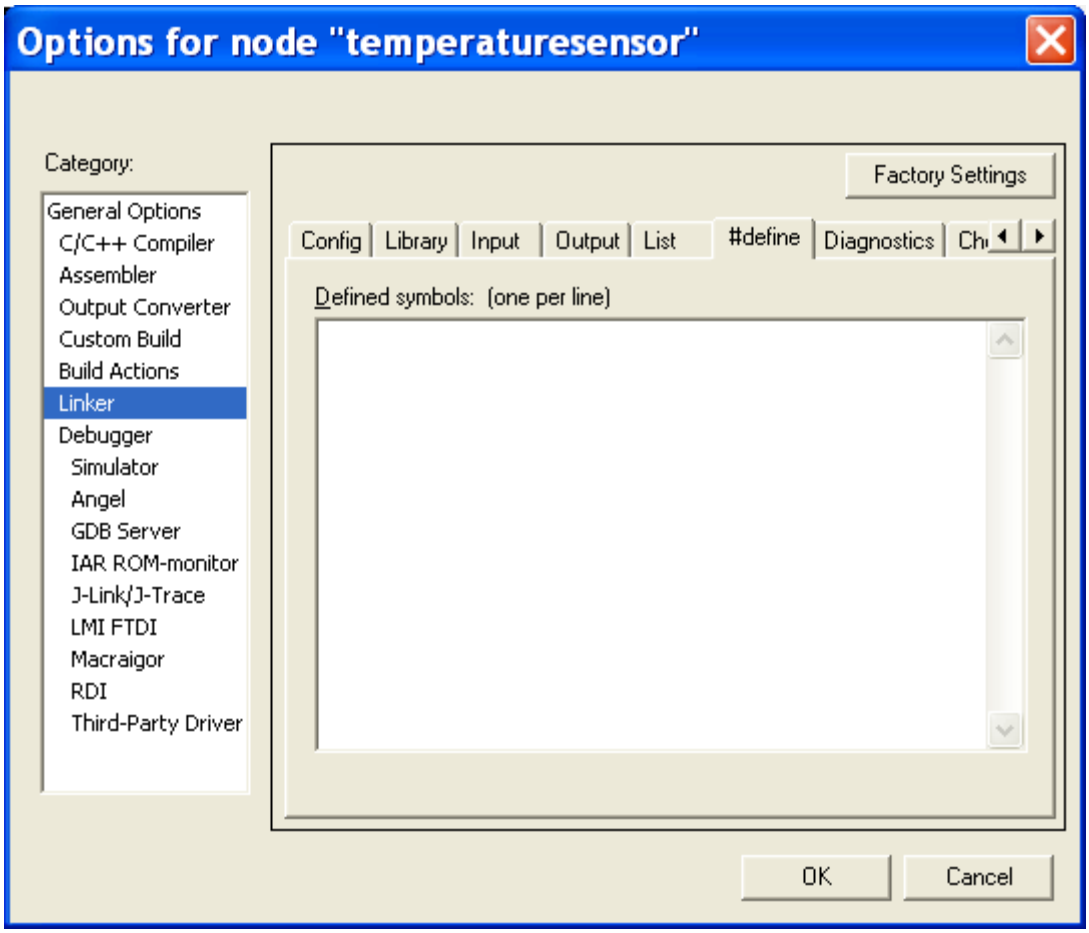

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Config
Library
Input
Output
List
#define
Diagnostics
Automatic runtime library selection
Additional libraries: (one per line)
Override default program entry
Entry symbol
__iar_program_start
Defined by application
OK
Cancel

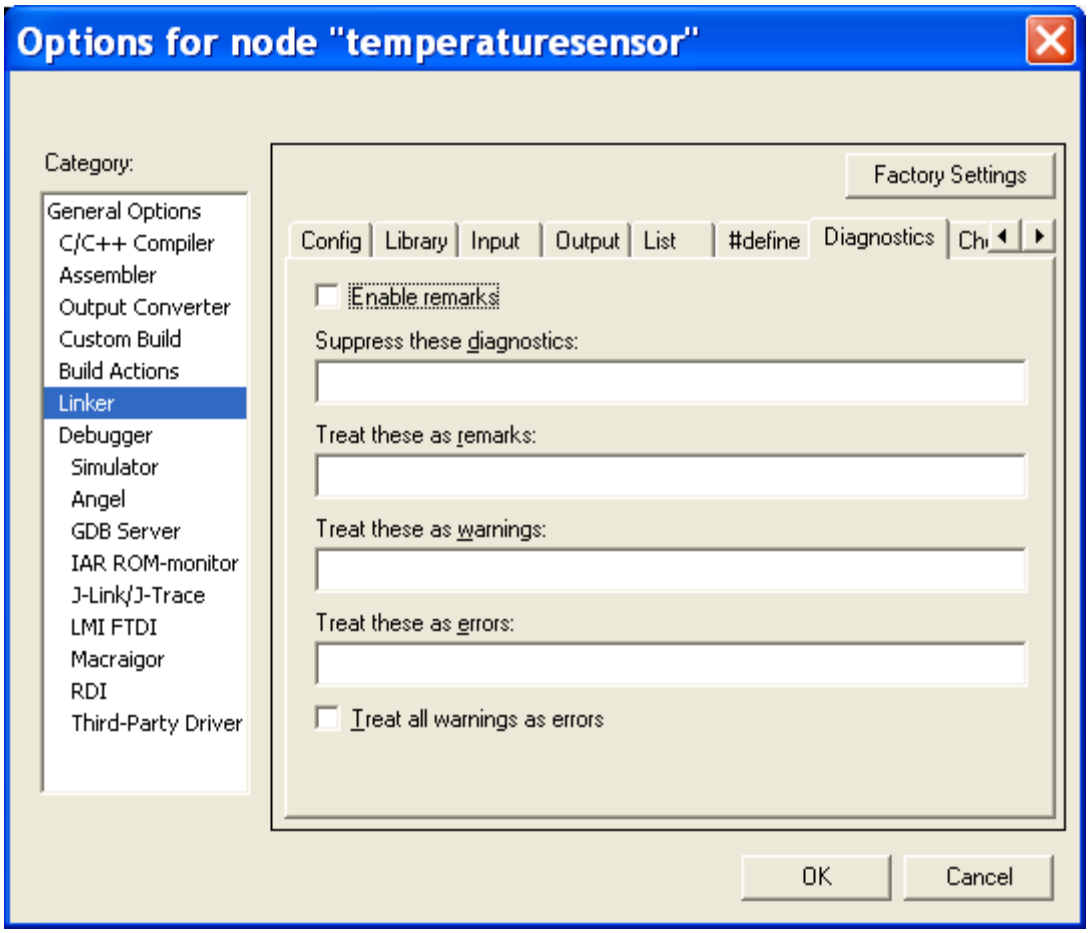

Options for node "Testlar"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Config
Library
Input
Output
List
#define
Diagnostics
Generate linker map file
Generate log file
Initialization decisions
Module selections
Section selections
Veneer statistics
OK
Cancel

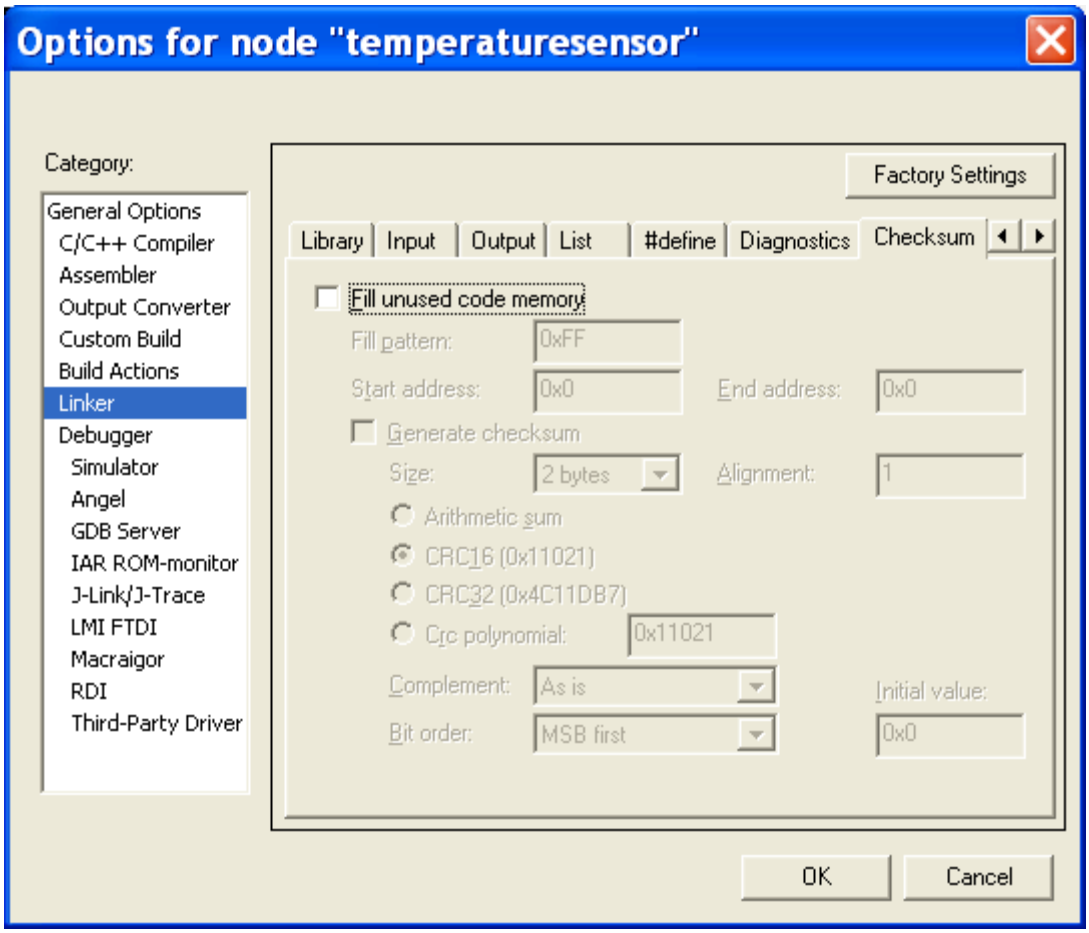

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Config
Library
Input
Output
List
#define
Diagnostics
Keep symbols: (one per line)
Raw binary image
File:
Symbol:
Section:
Align:
...
OK
Cancel

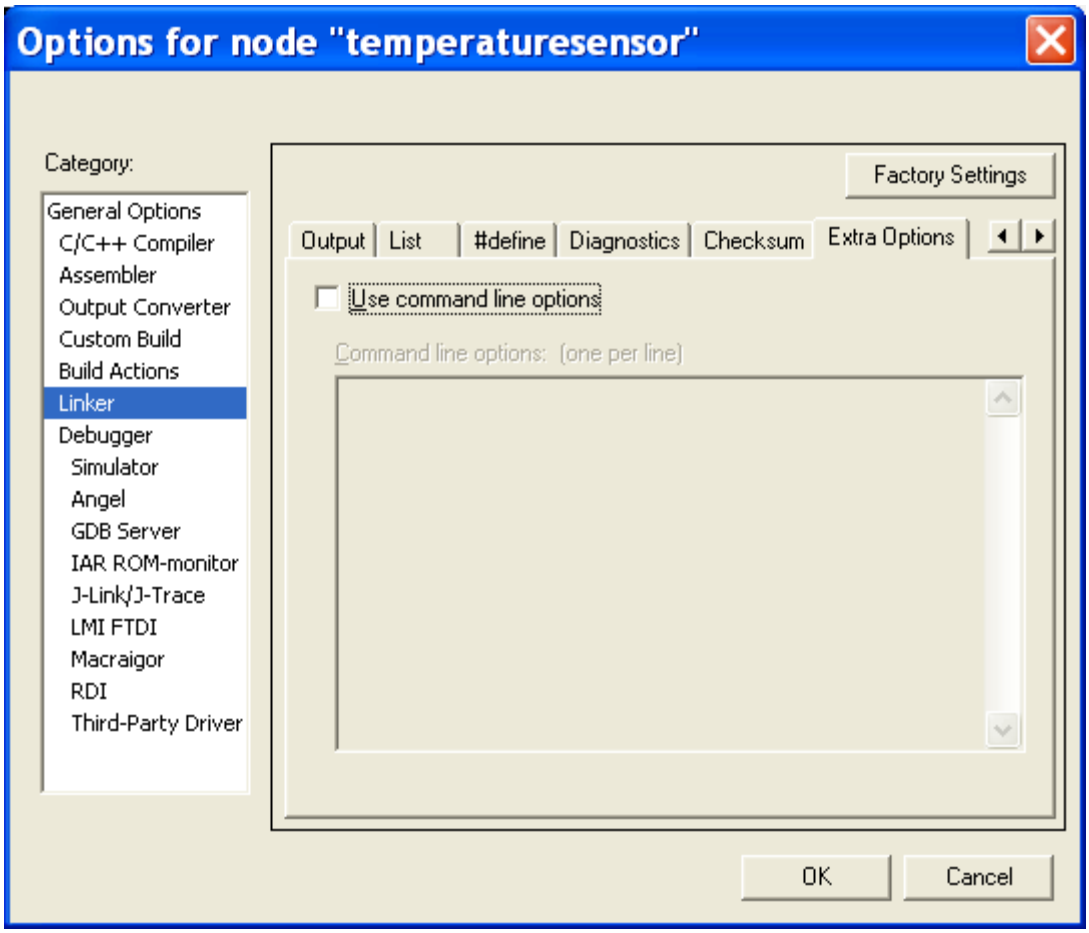

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Config
Library
Input
Output
List
#define
Diagnostics
Output file:
temperaturesensor.out
Include debug information in output
OK
Cancel

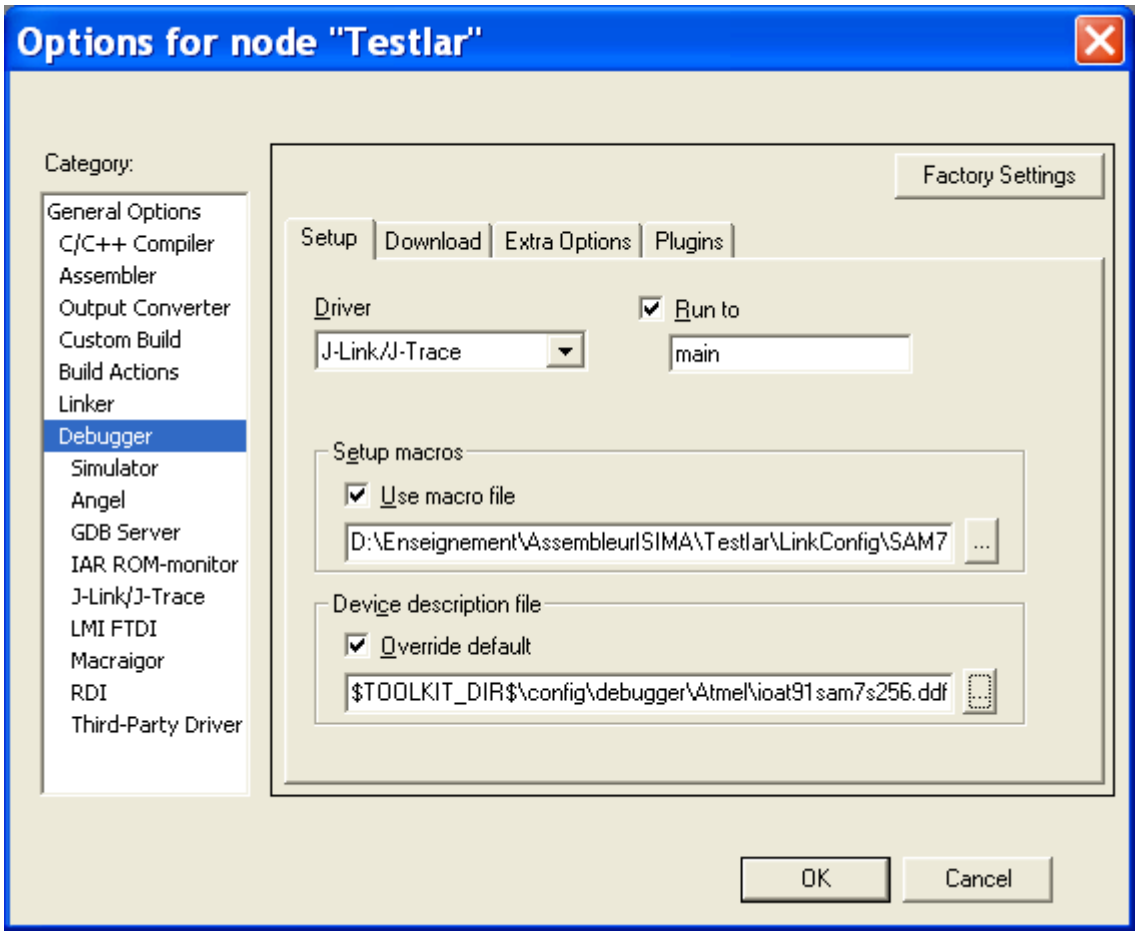

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Config
Library
Input
Output
List
#define
Diagnostics
Generate linker map file
Generate log file
Initialization decisions
Module selections
Section selections
Veneer statistics
OK
Cancel

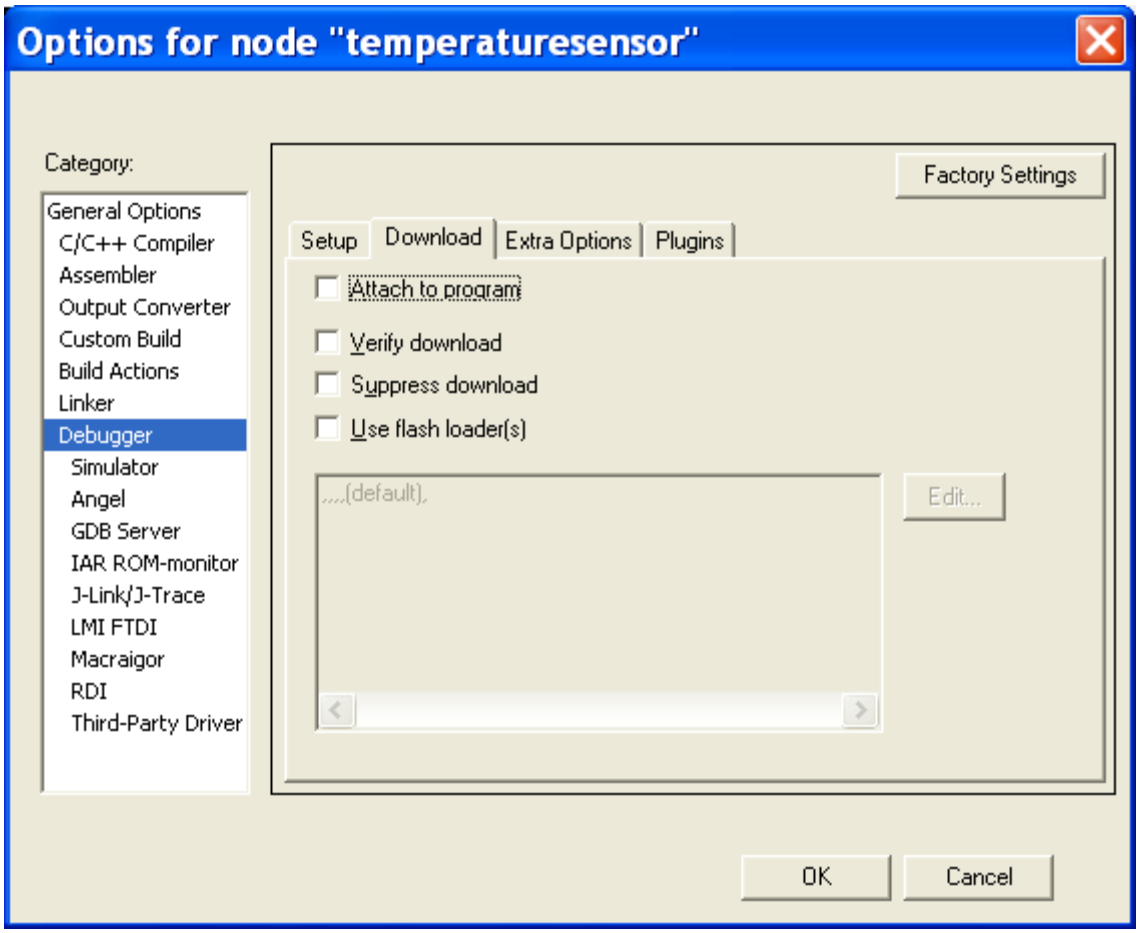

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Config
Library
Input
Output
List
#define
Diagnostics
Defined symbols: (one per line)
OK
Cancel

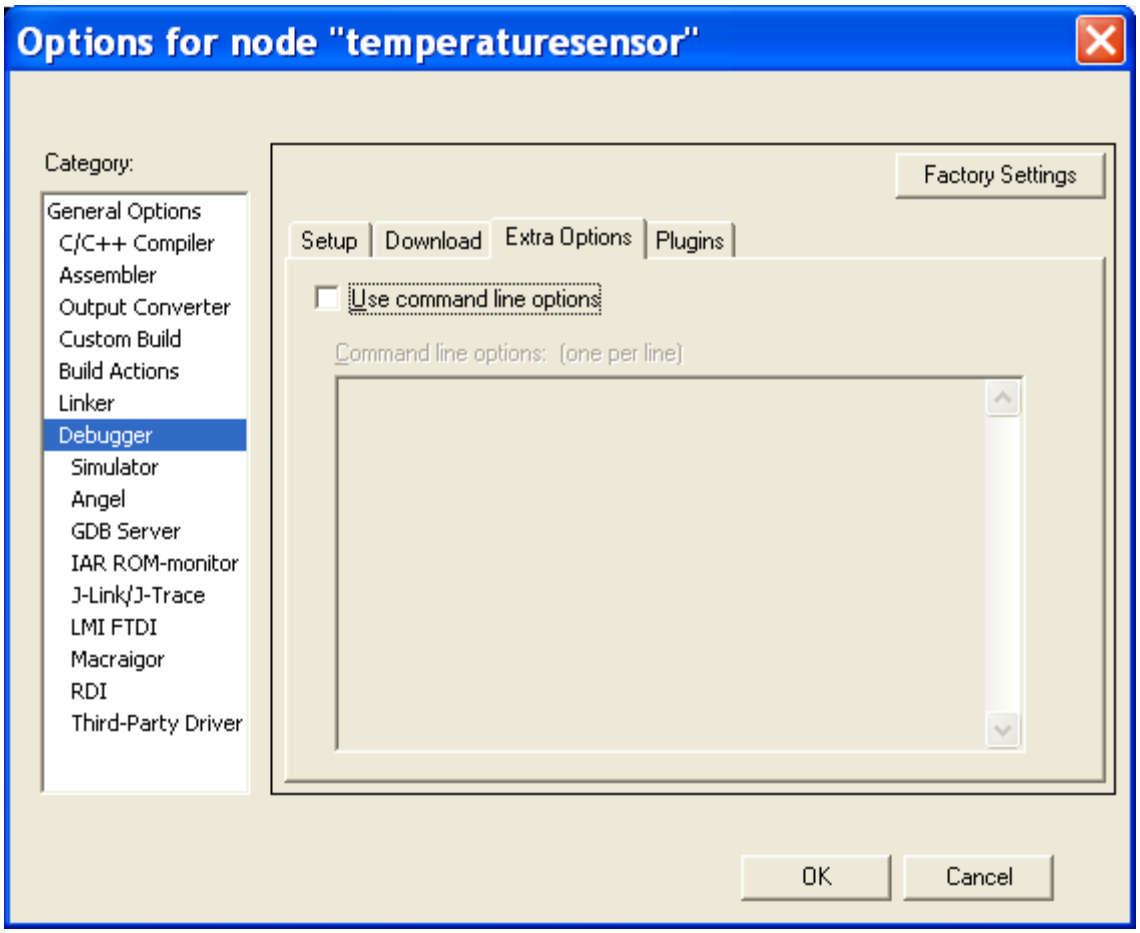

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Config
Library
Input
Output
List
#define
Diagnostics
Enable remarks
Suppress these diagnostics:
Treat these as remarks:
Treat these as warnings:
Treat these as errors:
Treat all warnings as errors
OK
Cancel

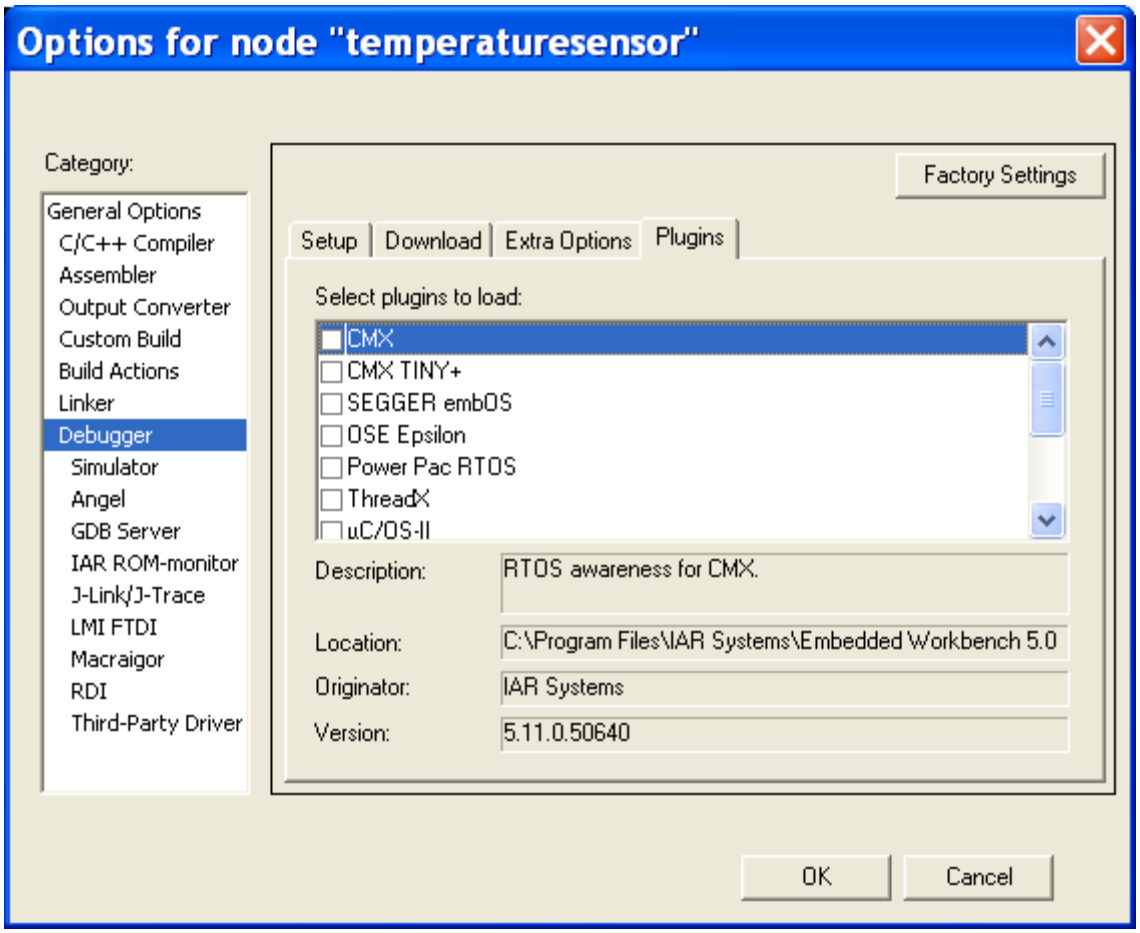

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Library
Input
Output
List
#define
Diagnostics
Checksum
Fill unused code memory
Fill pattern:
0xFF
Start address:
0x0
End address:
0x0
Generate checksum
Size:
2 bytes
Alignment:
1
Arithmetic sum
CRC16 (0x11021)
CRC32 (0x4C11DB7)
Crc polynomial:
0x11021
Complement:
As is
Initial value:
Bit order:
MSB first
0x0
OK
Cancel

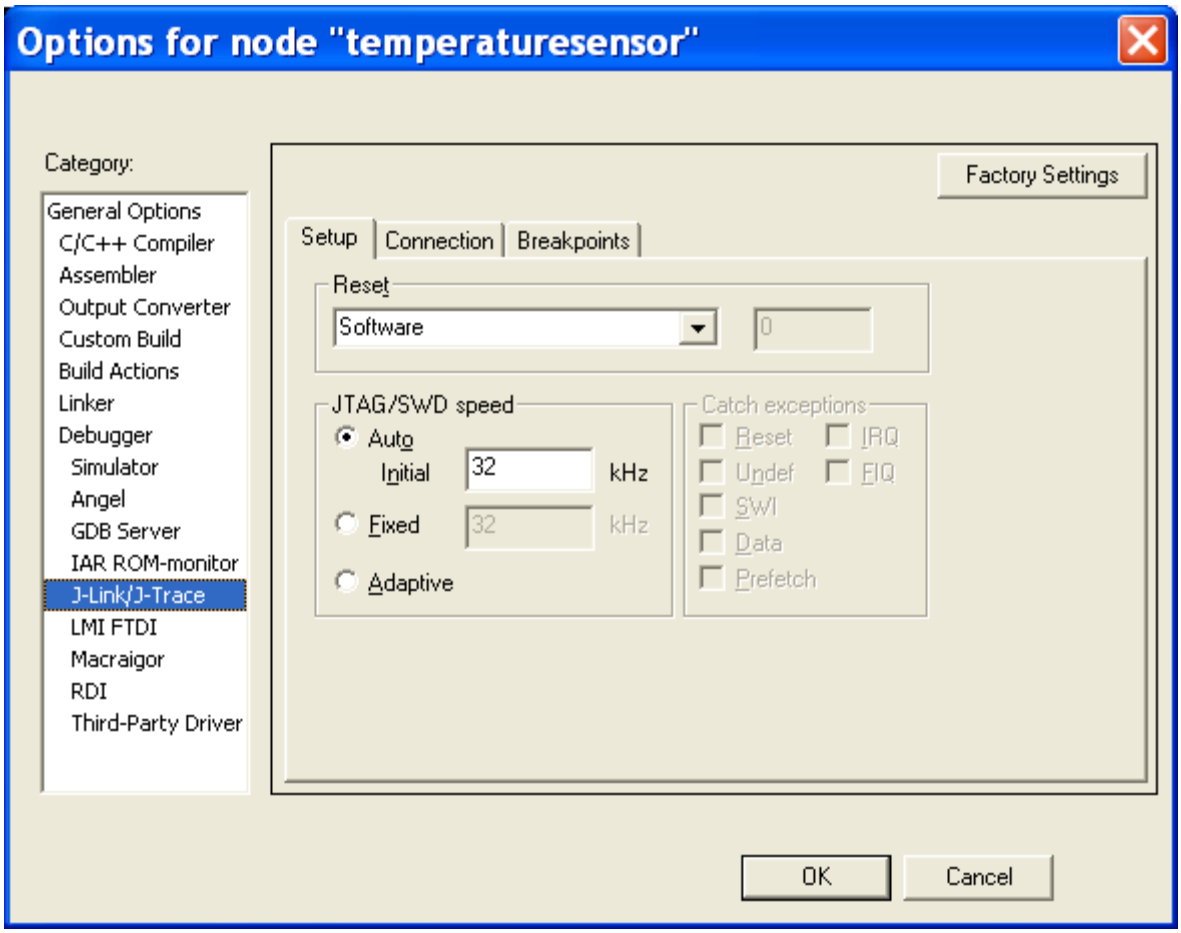

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Output
List
#define
Diagnostics
Checksum
Extra Options
Use command line options
Command line options: (one per line)
OK
Cancel

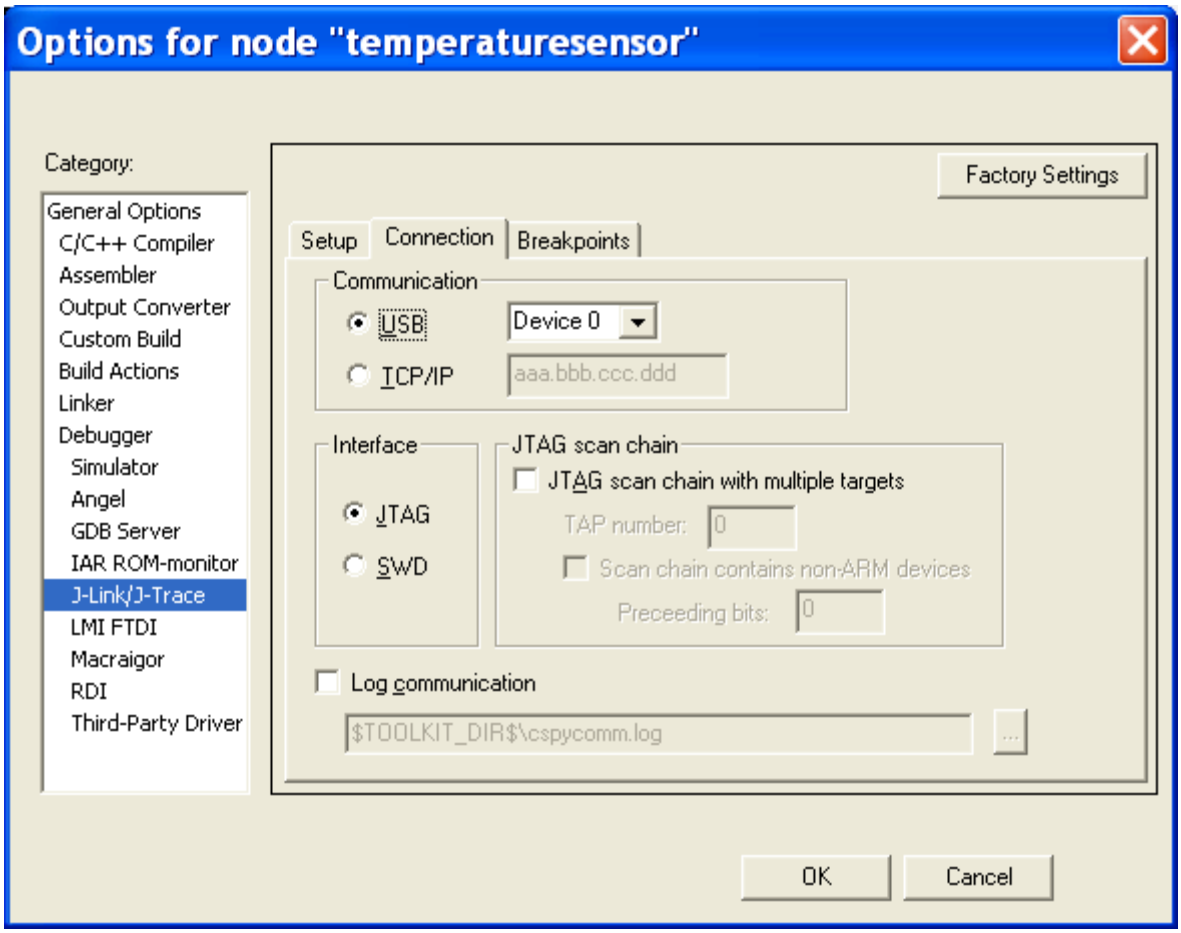

Options for node "Testlar"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Setup
Download
Extra Options
Plugins
Driver
J-Link/J-Trace
Run to
main
Setup macros
Use macro file
D:\Enseignement\AssembleurISIMA\Testlar\LinkConfig\SAM7
Device description file
Override default
$TOOLKIT_DIR$\config\debugger\Atmel\ioat91sam7s256.ddf
OK
Cancel

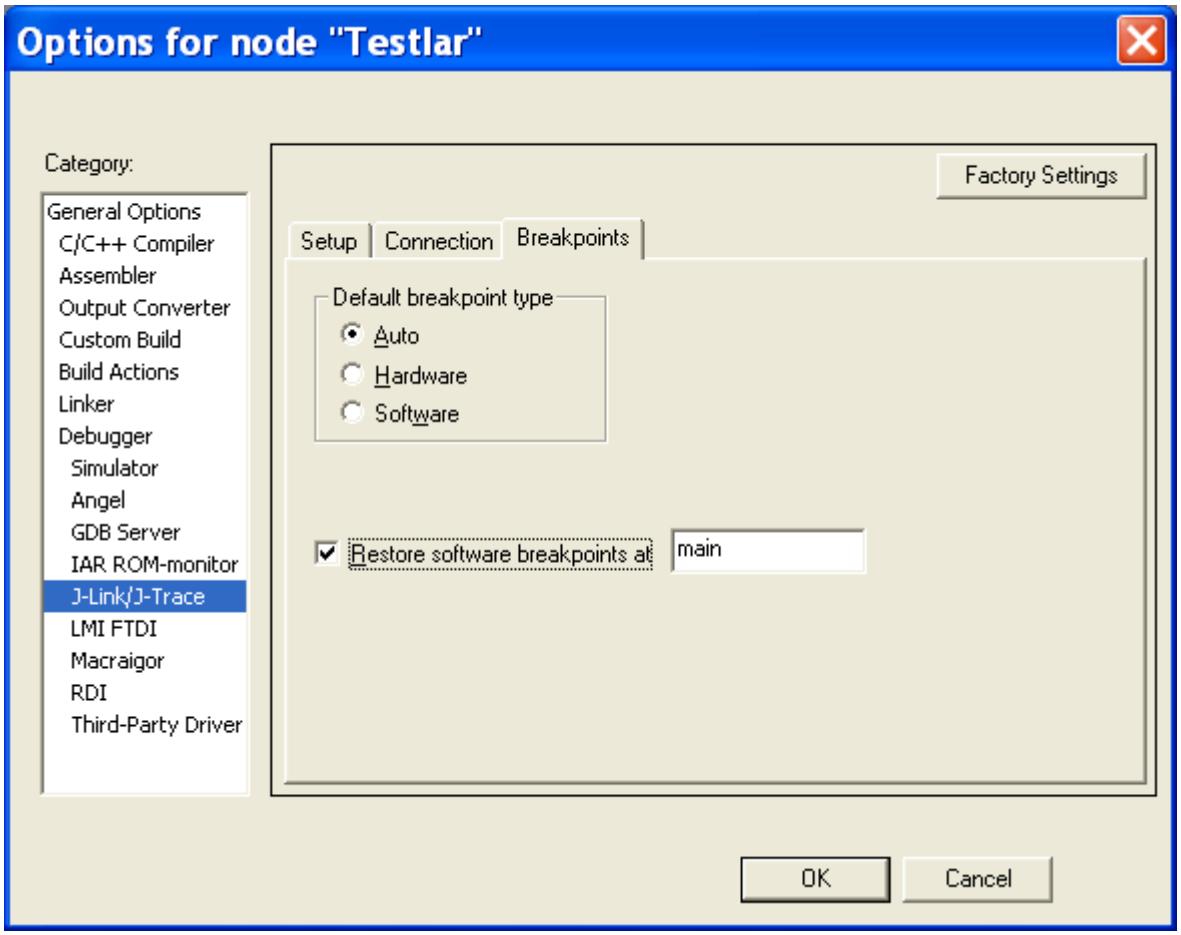

Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Setup
Download
Extra Options
Plugins
Attach to program
Verify download
Suppress download
Use flash loader(s)
,,,,(default),
Edit...
OK
Cancel


Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Setup
Download
Extra Options
Plugins
Use command line options
Command line options: (one per line)
OK
Cancel


Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Setup
Download
Extra Options
Plugins
Select plugins to load:
CMX
CMX TINY+
SEGGER embOS
OSE Epsilon
Power Pac RTOS
ThreadX
uC/OS-II
Description:
RTOS awareness for CMX.
Location:
C:\Program Files\IAR Systems\Embedded Workbench 5.0
Originator:
IAR Systems
Version:
5.11.0.50640
OK
Cancel


Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Setup
Connection
Breakpoints
Reset
Software
0
JTAG/SWD speed
Auto
Initial
32
kHz
Fixed
32
kHz
Adaptive
Catch exceptions
Reset
IRQ
Undef
FIQ
SWI
Data
Prefetch
OK
Cancel


Options for node "temperaturesensor"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Setup
Connection
Breakpoints
Communication
USB
Device 0
TCP/IP
aaa.bbb.ccc.ddd
Interface
JTAG
SWD
JTAG scan chain
JTAG scan chain with multiple targets
TAP number:
0
Scan chain contains non-ARM devices
Preceeding bits:
0
Log communication
$TOOLKIT_DIR$\cspycomm.log
...
OK
Cancel


Options for node "Testlar"
Category:
General Options
C/C++ Compiler
Assembler
Output Converter
Custom Build
Build Actions
Linker
Debugger
Simulator
Angel
GDB Server
IAR ROM-monitor
J-Link/J-Trace
LMI FTDI
Macraigor
RDI
Third-Party Driver
Factory Settings
Setup
Connection
Breakpoints
Default breakpoint type
Auto
Hardware
Software
Restore software breakpoints at
main
OK
Cancel

Annexe

IAR C-SPY DEBUGGER SYSTEMS

The IAR C-SPY Debugger consists of both a general part which provides a basic set of C-SPY features, and a driver. The C-SPY driver is the part that provides communication with and control of the target system. The driver also provides the user interface—menus, windows, and dialog boxes—to the functions provided by the target system, for instance, special breakpoints. There are three main types of C-SPY drivers:

- Simulator driver
- ROM-monitor driver
- Emulator driver

If you have more than one C-SPY driver installed on your computer you can switch between them by choosing the appropriate driver from within the IAR Embedded Workbench IDE.

VIEWING THE LIST FILE

Now examine the compiler list file and notice how it is automatically updated when you, as in this case, will investigate how different optimization levels affect the generated code size.

Open the list file `Utilities.lst` by double-clicking it in the workspace window. Examine the list file, which contains the following information:

- The *header* shows the product version, information about when the file was created, and the command line version of the compiler options that were used
- The *body* of the list file shows the assembler code and binary code generated for each statement. It also shows how the variables are assigned to different segments
- The *end* of the list file shows the amount of stack, code, and data memory required, and contains information about error and warning messages that might have been generated.

DIFFERENCES BETWEEN THE C-SPY DRIVERS

The following table summarizes the key differences between the C-SPY drivers:

Feature	Simulator	RDI	Macraigor	J-Link	Angel	IAR ROM-monitor
Data breakpoints	x		x 2)	x 2)		
Code breakpoints	x	x	x	x	x	x
Execution in real time		x	x	x	x	x
Zero memory footprint	x	x	x	x		
Simulated interrupts	x					
Real interrupts		x	x	x	x	x
Live Watch	x					
Cycle counter	x					
Code coverage	x					
Profiling	x	x 1) 3)	x 1) 3)	x 1) 3)	x 1)	x 1)

Table 26: Differences between available C-SPY drivers

1) Cycle counter statistics are not available.
2) Limited number, implemented using the ARM EmbeddedICE™ macrocell
3) Profiling works provided that enough breakpoints are available. That is, the application is executed in RAM.

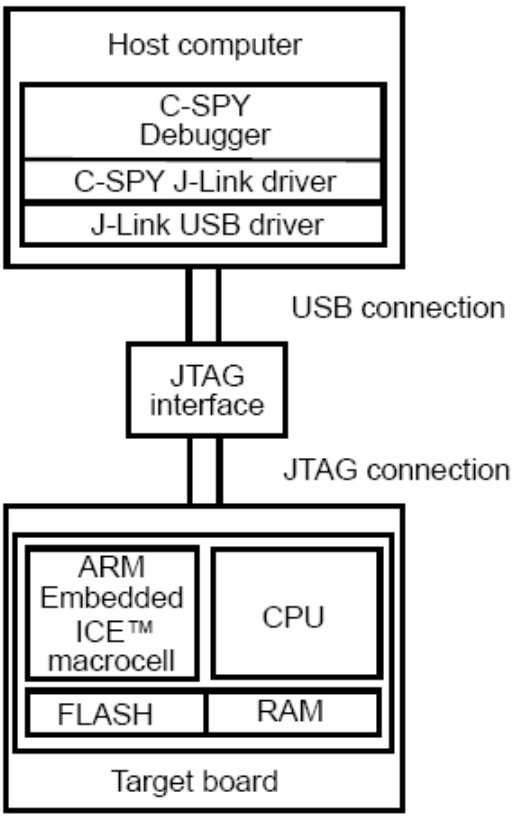


Figure 81: C-SPY J-Link communication overview

INSTALLING THE J-LINK USB DRIVER

Before you can use the J-Link JTAG interface over the USB port, the Segger J-Link USB driver must be installed.

Install ARM IAR Embedded Workbench.

Use the USB cable to connect the computer and J-Link. Do not connect J-Link to the target-board yet. The green LED on the front panel of J-Link will blink for a few seconds while Windows searches for a USB driver.

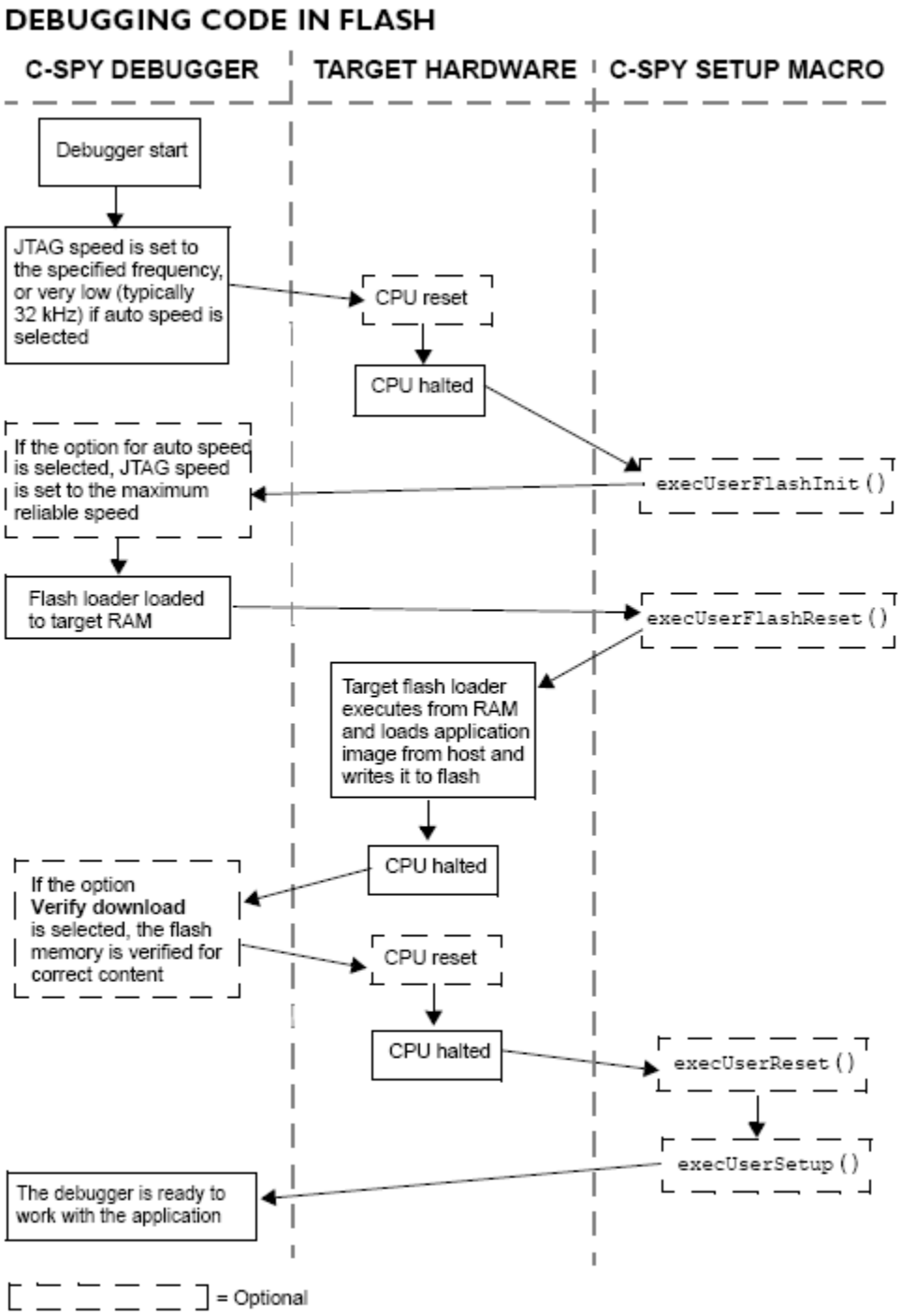


Figure 85: Debugger startup when debugging code in flash

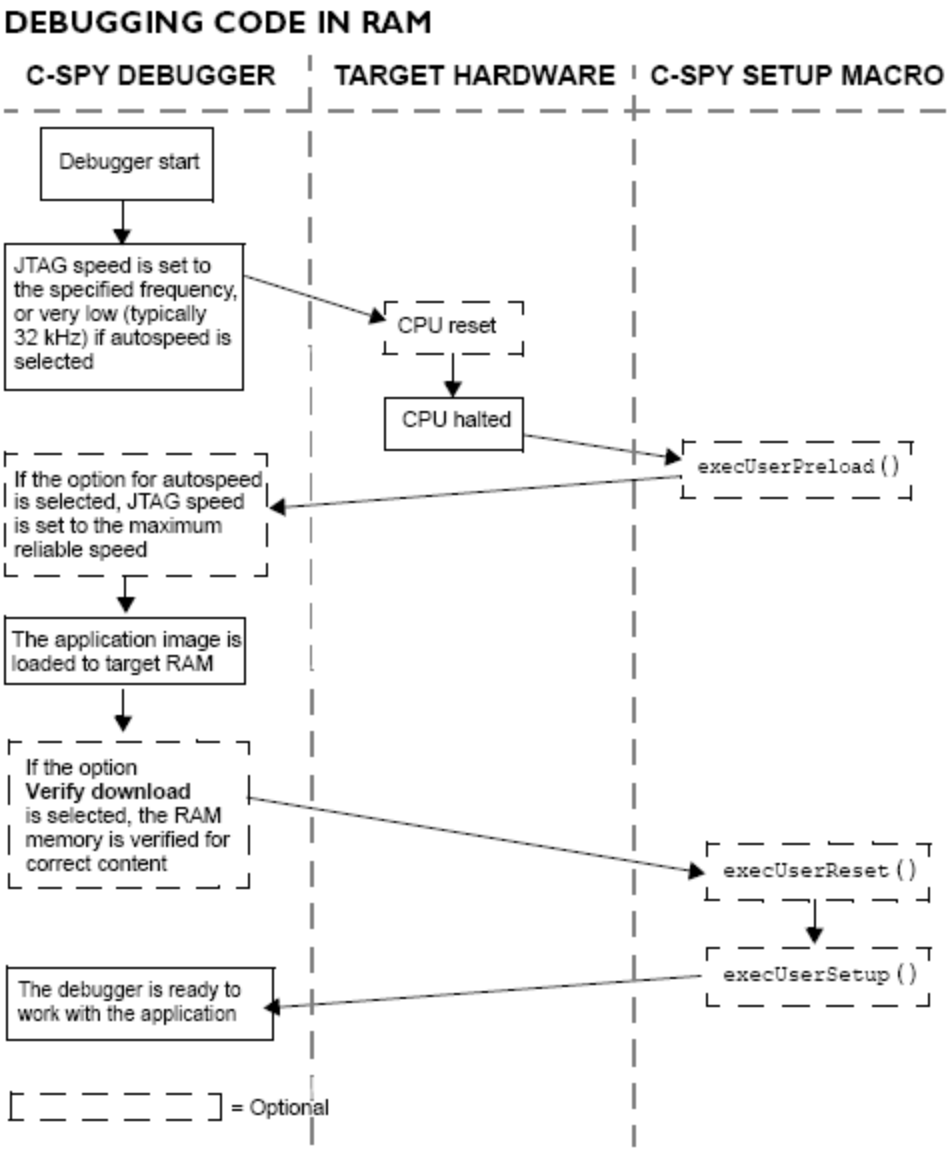
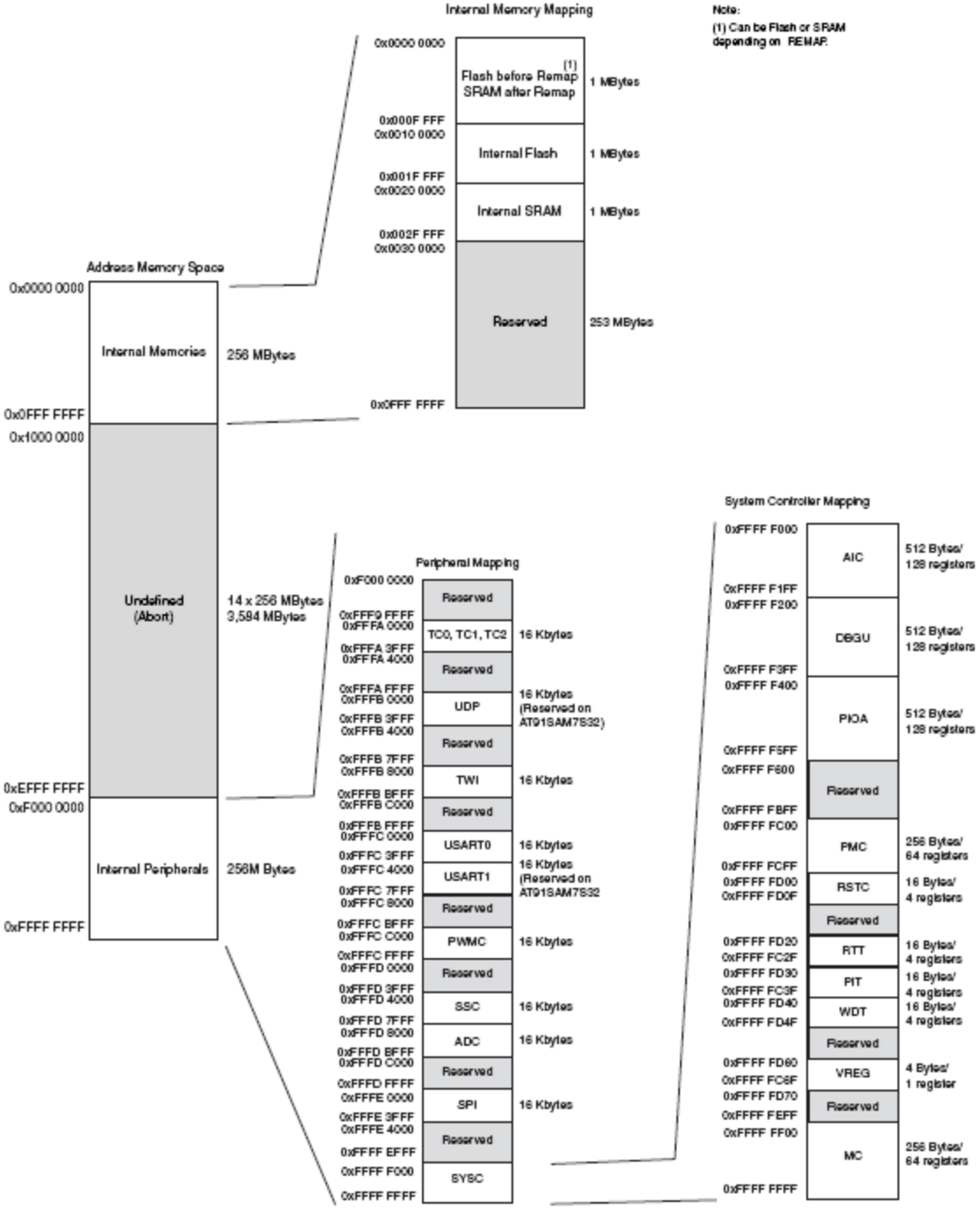


Figure 86: Debugger startup when debugging code in RAM

Figure 8-1. AT91SAM7S512/256/128/64/321/32 Memory Mapping



C:\ProgramFiles\IARSystems\EmbeddedWorkbench5.0Kickstart\arm\src\examples\Atmel\SAM7S256\AT91SAM7S-BasicTools\Compil\resource

Compiler le projet