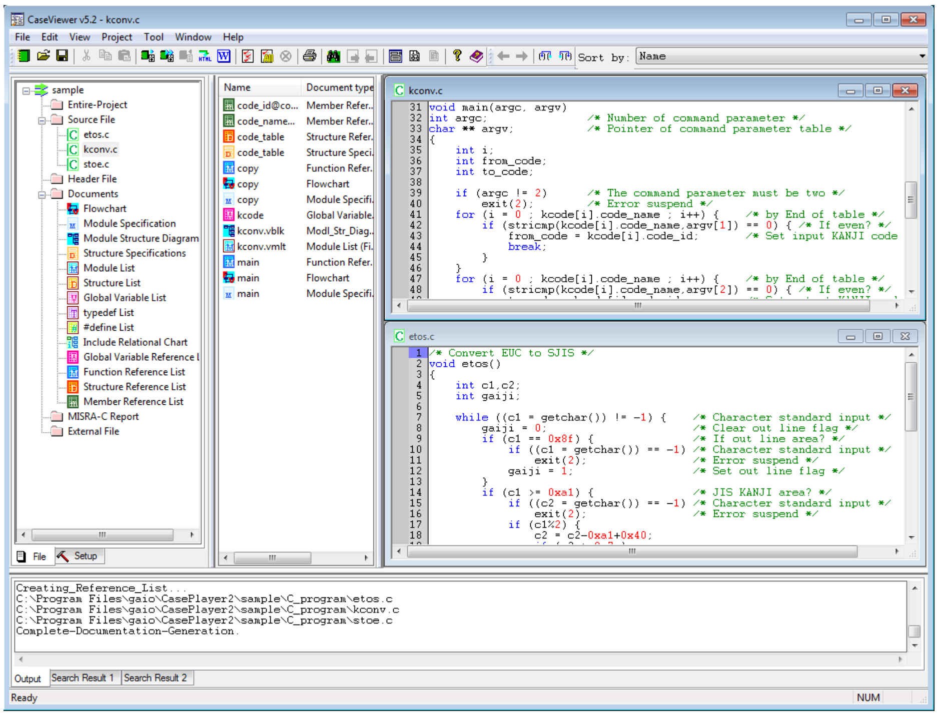
Solution House GAIO TECHNOLOGY



Name	Document type
code_id@co...	Member Refer..
code_name...	Member Refer..
code_table	Structure Refer.
code_table	Structure Speci.
copy	Function Refer.
copy	Flowchart
copy	Module Specifi.
kcode	Global Variable.
kconv.vblk	Modl_Str_Diag..
kconv.vmlt	Module List (Fi.
main	Function Refer.
main	Flowchart
main	Module Specifi.

```
void main(argc, argv)
int argc;                    /* Number of command parameter */
char ** argv;                /* Pointer of command parameter table */
{
    int i;
    int from_code;
    int to_code;

    if (argc != 2)           /* The command parameter must be two */
        exit(2);             /* Error suspend */
    for (i = 0 ; kcode[i].code_name ; i++) {     /* by End of table */
        if (stricmp(kcode[i].code_name,argv[1]) == 0) { /* If even? */
            from_code = kcode[i].code_id;        /* Set input KANJI code
            break;
        }
    }
    for (i = 0 ; kcode[i].code_name ; i++) {     /* by End of table */
        if (stricmp(kcode[i].code_name,argv[2]) == 0) { /* If even? */
```

```
/* Convert EUC to SJIS */
void etos()
{
    int c1,c2;
    int gaiji;

    while ((c1 = getchar()) != -1) {    /* Character standard input */
        gaiji = 0;                      /* Clear out line flag */
        if (c1 == 0x8f) {               /* If out line area? */
            if ((c1 = getchar()) == -1) /* Character standard input */
                exit(2);                /* Error suspend */
            gaiji = 1;                  /* Set out line flag */
        }
        if (c1 >= 0xa1) {               /* JIS KANJI area? */
            if ((c2 = getchar()) == -1) /* Character standard input */
                exit(2);                /* Error suspend */
            if (c1%2) {
                c2 = c2-0xa1+0x40;
```

```
Creating_Reference_List...
C:\Program Files\gaio\CasePlayer2\sample\C_program\etos.c
C:\Program Files\gaio\CasePlayer2\sample\C_program\kconv.c
C:\Program Files\gaio\CasePlayer2\sample\C_program\stoe.c
Complete-Documentation-Generation.
```

CasePlayer2 Tutorial

Ver. 1.1
Jan. 2014

Table of Contents

CasePlayer2 Tutorial

Introduction

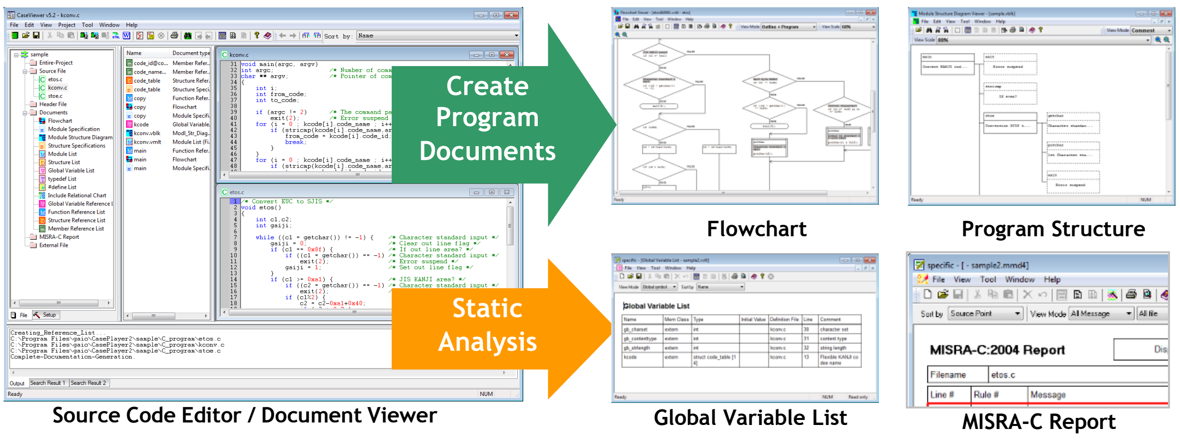
At this time we wish to thank you for your interest in GAIO TECHNOLOGY's static analysis and program document creation tool, CasePlayer2.

This tutorial includes practice exercises designed for first time users. By following this tutorial and completing the exercises, the user will gain an understanding of CasePlayer2's basic usage and features.

Before using this tutorial, CasePlayer2 should be installed. The default installation folder is C:¥Program Files¥gaio¥Caseplayer2.

CasePlayer2 Overview

CasePlayer2 is an integrated reverse CASE tool that can create program documents such as flowcharts and MISRA-C reports by analyzing the source code. It includes a Document Browser for easy access to program documents and source code for review.

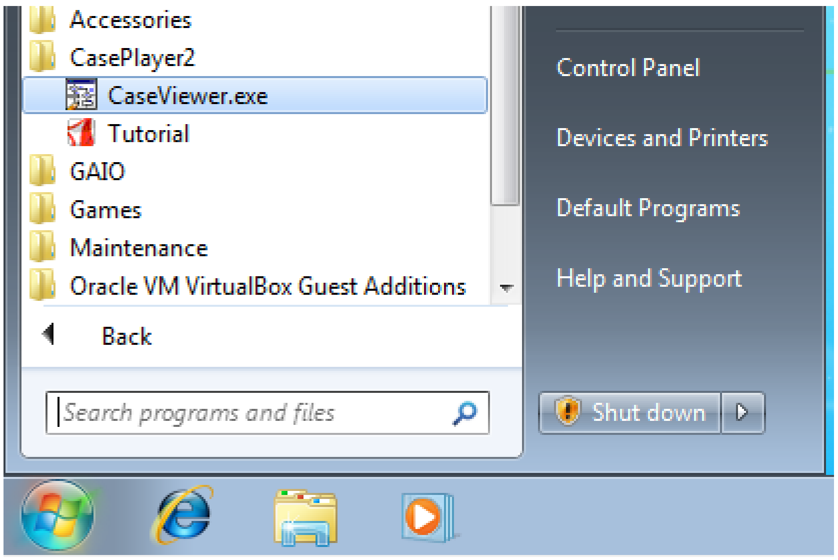


System Requirements

* IBM Compatible
* USB port (for USB license key)
* Windows XP, Windows Vista (32-bit), Windows 7 (32/64-bit)
* Hard Drive: 50 MB free space
* Microsoft Internet Explorer 5.5 or newer to view HTML format documents
* Microsoft Word 2000 or newer to view MS-Word format documents

1. Start CasePlayer2

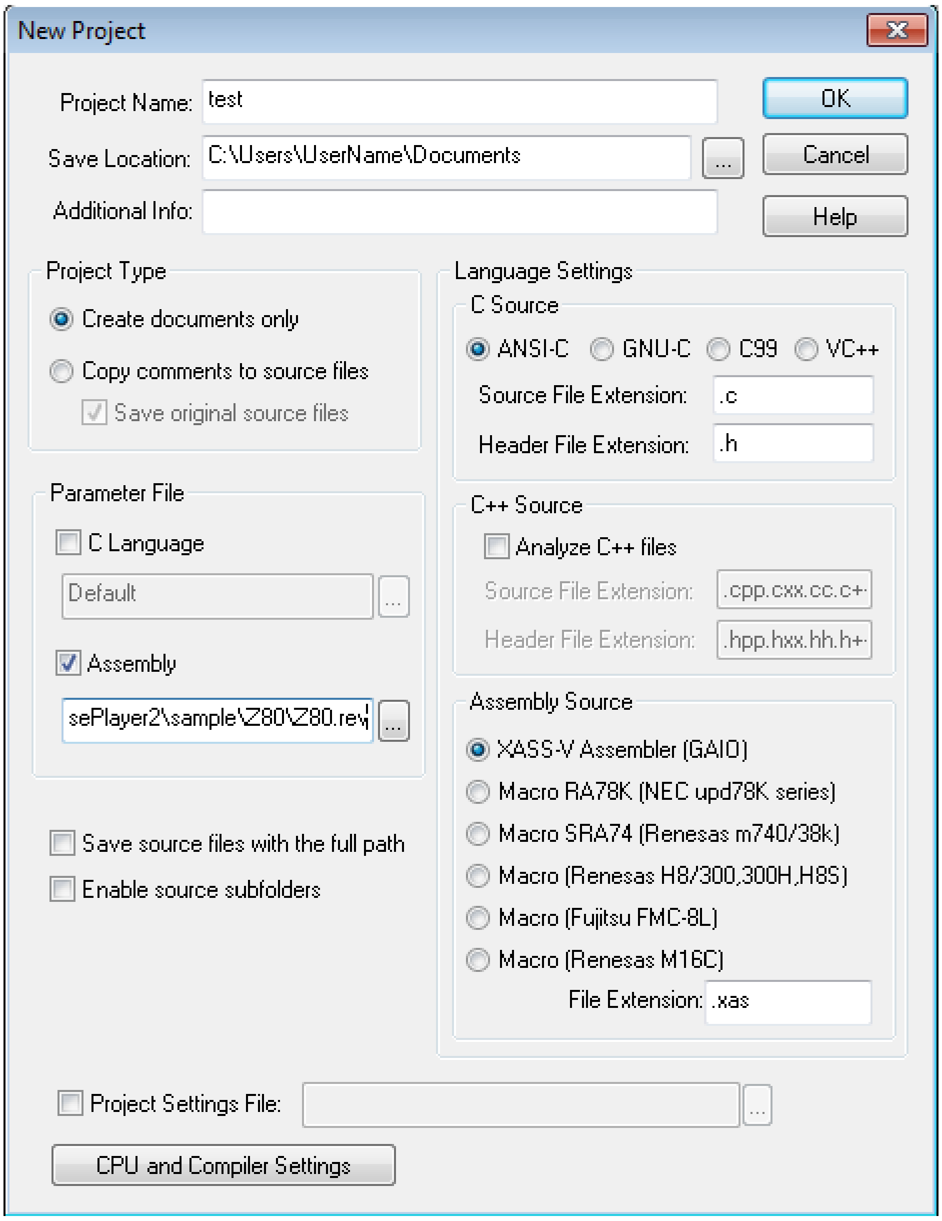
Start CasePlayer2 from the Windows Start Menu: **All Programs** -> **CasePlayer2** -> **CaseViewer.exe**.



2. Create a New Project

To begin this tutorial first we will create a new project to manage the source code analysis and output documents.

1. From the application menu: **File** -> **New** -> **Project**
2. Project Name: **test**
3. Save Location: (user selected)
4. Project Type: **Create documents only**
5. Language Settings: **ANSI-C**
6. Parameter File: Check the **Assembly** box,
   rev file: **C:¥Program Files¥gaio¥CasePlayer2¥sample¥Z80¥Z80.rev**
7. Leave other options at default settings
8. Click **OK**



New Project

Project Name: test

Save Location: C:\Users\UserName\Documents

Additional Info:

Project Type
- [✓ Create documents only]
- Copy comments to source files
- [✓ Save original source files]

Parameter File
- C Language: Default
- [✓ Assembly]: sePlayer2\sample\Z80\Z80.re

- Save source files with the full path
- Enable source subfolders

Language Settings

C Source
- [✓ ANSI-C] GNU-C C99 VC++
- Source File Extension: .c
- Header File Extension: .h

C++ Source
- Analyze C++ files
- Source File Extension: .cpp.cxx.cc.c+
- Header File Extension: .hpp.hxx.hh.h+

Assembly Source
- [✓ XASS-V Assembler (GAIO)]
- Macro RA78K (NEC upd78K series)
- Macro SRA74 (Renesas m740/38k)
- Macro (Renesas H8/300,300H,H8S)
- Macro (Fujitsu FMC-8L)
- Macro (Renesas M16C)
- File Extension: .xas

Project Settings File:

The test project folder and related project files will be created in the location selected. The **test.vproj** file is the CasePlayer2 project file which may be double-clicked to start CasePlayer2 and open the project.

About Other Settings (New Project Dialog)

Project Type

Create documents only: This mode is used to create program documents from the source code only. The source code files will not be modified in this mode.
(This mode will be used in this tutorial).

Copy comments to source files: This mode is used to create program documents from the source code, and allows for copying comments added to created program documents to the original source code files. If the **Save original source files** box is checked, comments will be copied to duplicate versions of the source files.

Parameter File

C Language: The parameter files used to analyze compiler extension instructions or ANSI-C non-compliant code. A parameter file does not need to be set if the source code is compliant with the ANSI-C standard.

Assembly: The parameter files used to analyze the assembly language source files. The parameter file contains the instruction grouping information for the microprocessor. GAIO provides sample parameter files for over 40 microprocessor types. Assembly sample code is used for the Z80 microprocessor in this tutorial.

Language Settings

C Source

Select the C language source code type. **ANSI-C** mode is used in this tutorial.

C++ Source

Enable the **Analyze C++ files** setting if the source code includes C++ code. This option is not used in this tutorial.

Assembly Source

Select the assembly macro type for analyzing assembly macro descriptions. **XASS-V Assembler (GAIO)** is used in this tutorial.

Enable source subfolders

Subfolders will appear in the file tree view if this option is checked. This option is not used in this tutorial.

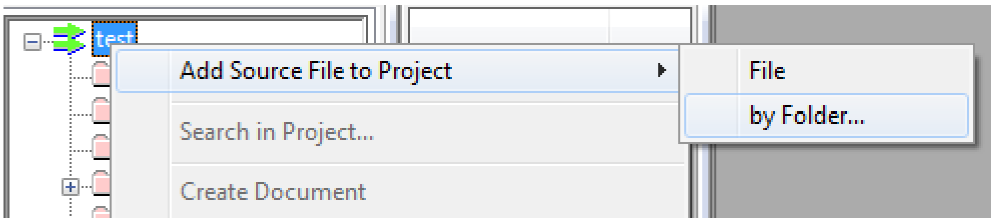
Project Settings File

The project settings may be saved to a project setting file. Use this option if you wish to save and reuse the selected project settings. This option is not used in this tutorial.

3. Add Source Files to the Project

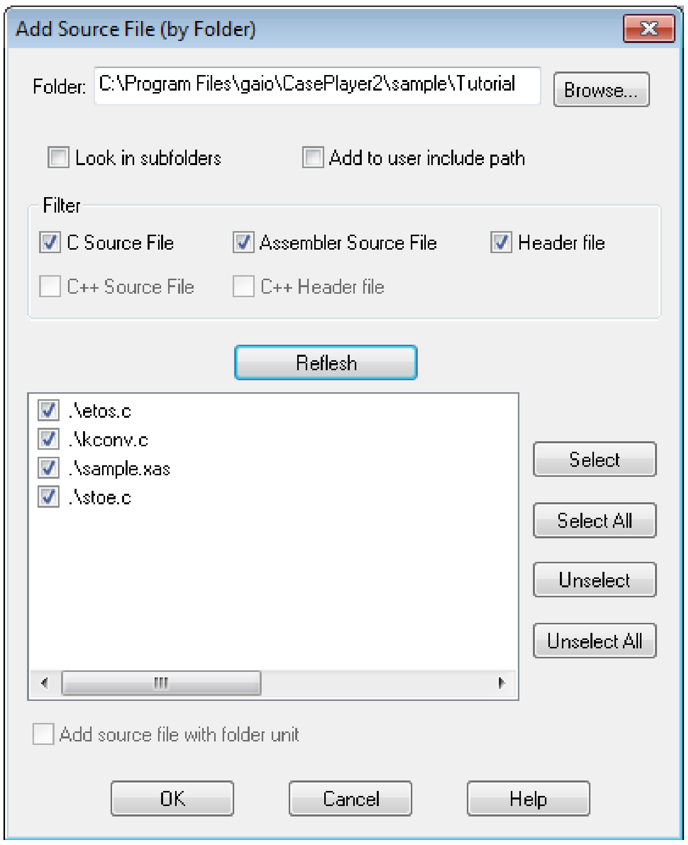
After creating a project, next add source files to the project to be analyzed.

1. **Right-click** on the **test** (project icon) in the file tree view, then select **Add Source File to Project** -> **by Folder**.



2. Select the tutorial sample folder:
   C:¥Program Files¥gaio¥CasePlayer2¥sample¥Tutorial

The file with .xas extension is an assembly source file. In this tutorial the Z80 assembly source file **sample.xas** is used. Assembly file extension file settings may be changed from the project properties dialog accessible from the menu: **Project** -> **Properties**.



Add Source File (by Folder)

Folder: C:\Program Files\gaio\CasePlayer2\sample\Tutorial

Look in subfolders: ☐
Add to user include path: ☐

Filter:
- C Source File: ☑
- Assembler Source File: ☑
- Header file: ☑
- C++ Source File: ☐
- C++ Header file: ☐

- ☑ .\etos.c
- ☑ .\kconv.c
- ☑ .\sample.xas
- ☑ .\stoe.c

Add source file with folder unit: ☐

3. Click **OK**.

About Other Settings (Add Source File Dialog)

Look in subfolders

Check this option to display files located in subfolders.

Add to user include path

Check this option to add the folder for the selected header files to the **User Include Path** list in the **Pre-processor** settings.

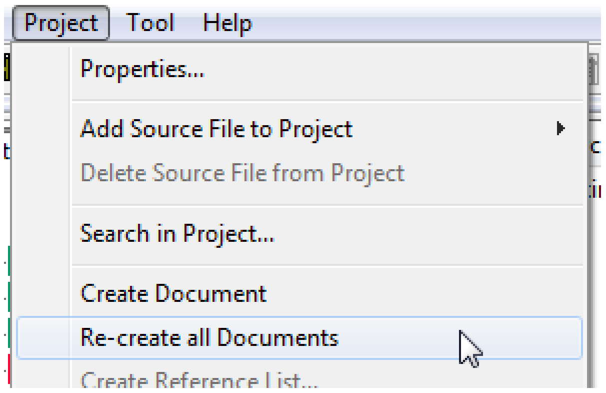
Add source file with folder unit

Check this option to retain subfolder structure when adding source files to the project. The **Enable source subfolders** option in the New Project dialog must be enabled in order to use this setting.

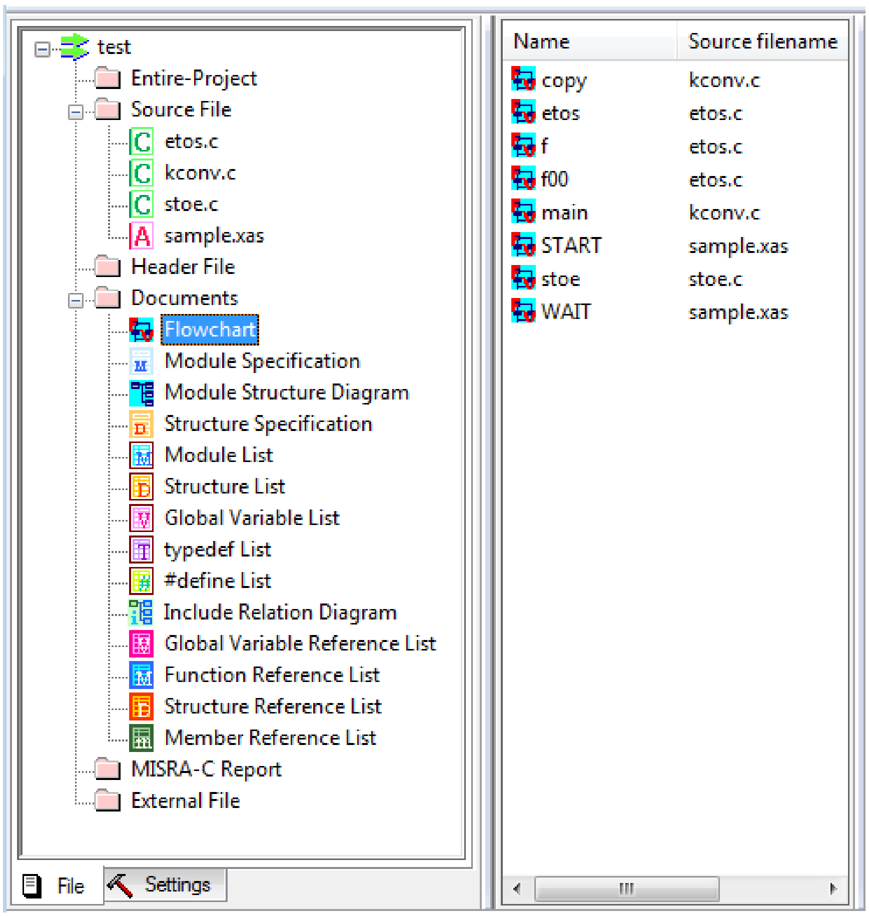
4. Create Program Documents (Simple Mode)

Now let’s create program documents using the Simple (default) Mode. In this mode program documents such as flowcharts, module specifications, etc. may be created without setting any of the advanced settings.

1. From the application menu select **Project** -> **Re-create all documents**. (this will analyze the source files and then create program documents)



2. Expand the Documents folder in the file tree view to display a list of created program document types.



- test
  - Entire-Project
  - Source File
    - etos.c
    - kconv.c
    - stoe.c
    - sample.xas
  - Header File
  - Documents
    - Flowchart
    - Module Specification
    - Module Structure Diagram
    - Structure Specification
    - Module List
    - Structure List
    - Global Variable List
    - typedef List
    - #define List
    - Include Relation Diagram
    - Global Variable Reference List
    - Function Reference List
    - Structure Reference List
    - Member Reference List
  - MISRA-C Report
  - External File

Name	Source filename
copy	kconv.c
etos	etos.c
f	etos.c
f00	etos.c
main	kconv.c
START	sample.xas
stoe	stoe.c
WAIT	sample.xas

The following document types may be created in Simple Mode:

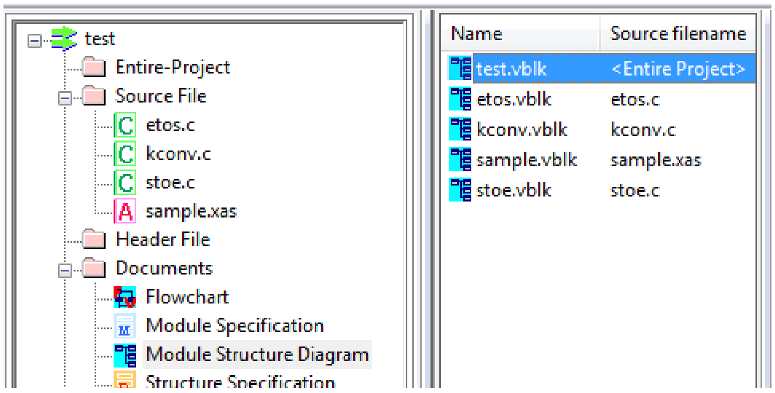
Flowchart: Shows program flow for the function
Module Specification: Includes module (function) information such as return type, function parameters, etc.
Module Structure Diagram: Shows function call structure
Structure Specification: Includes structure information such as a list of member variables
Module List: Includes a list of modules (functions) found in the source files
Structure List: Includes a list of structures found in the project

Additional document types require usage of the Advanced Mode (described later in this tutorial).

View Created Documents

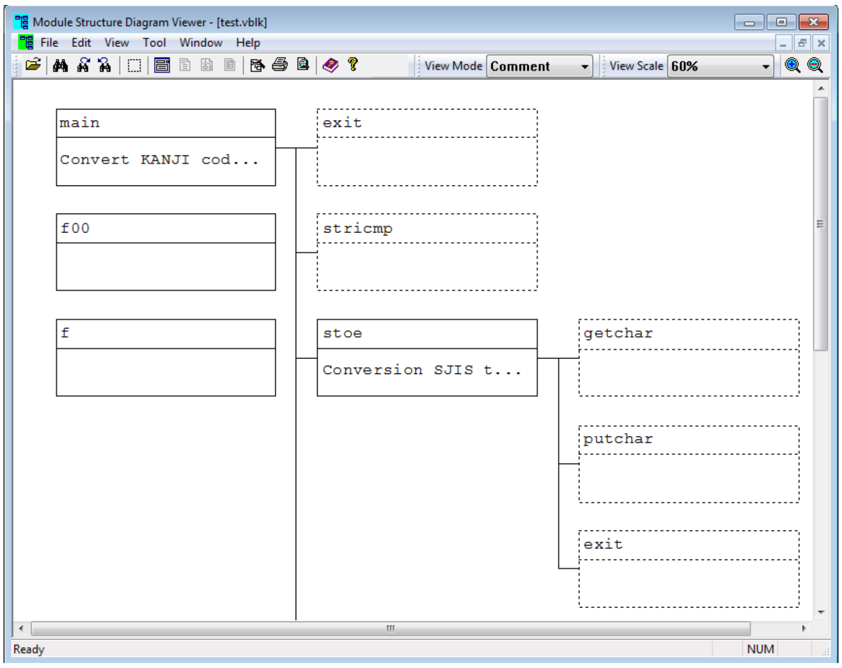
Next let's take a look at the created Module Structure Diagram.

1. Select **Module Structure Diagram** located within the Documents folder of the file tree view.
2. Double-click **test.vblk <Entire Project>** that appears in the list to the right of the file tree view.



Name	Source filename
test.vblk	<Entire Project>
etos.vblk	etos.c
kconv.vblk	kconv.c
sample.vblk	sample.xas
stoe.vblk	stoe.c

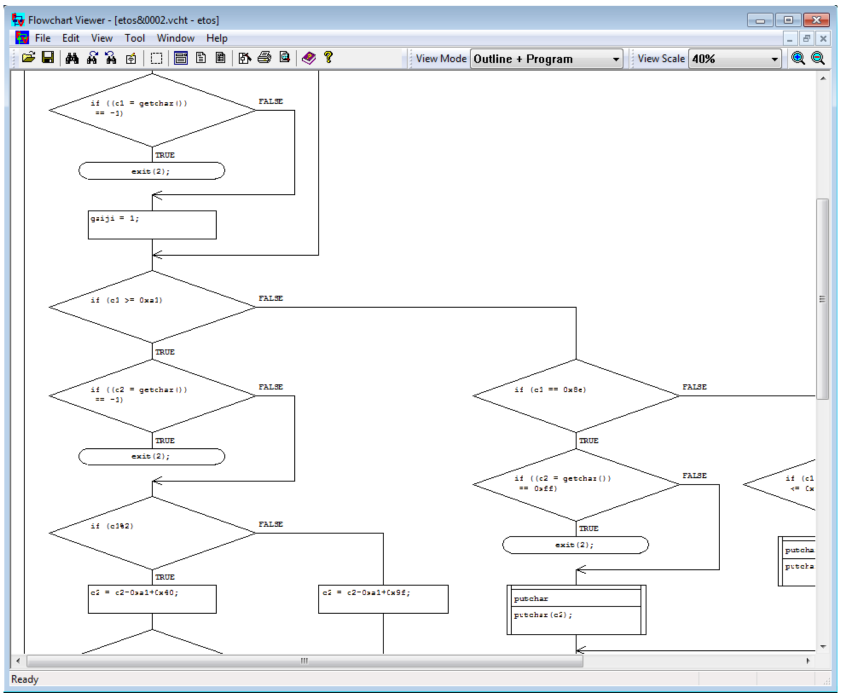
Scroll to view the document or change the View Scale (zoom in/out) to change the display.



Easy Access to and from Program Documents and Source Code

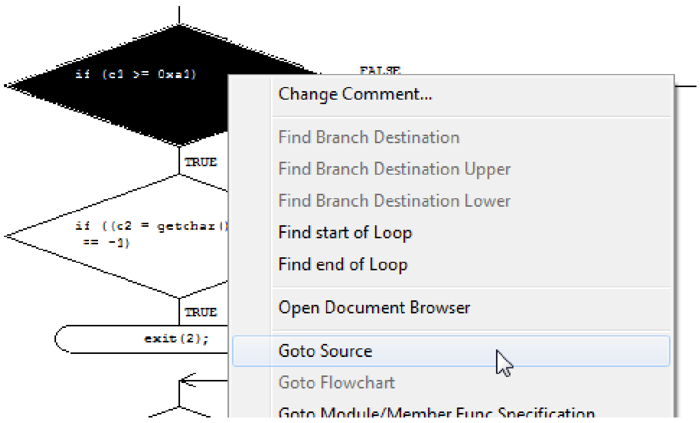
Next let's view a created flowchart and use the right-click menu feature to goto the source code from the flowchart.

1. Select **Flowchart** located within the Documents folder of the file tree view.
2. Double-click **etos** that appears in the list to the right of the file tree view.

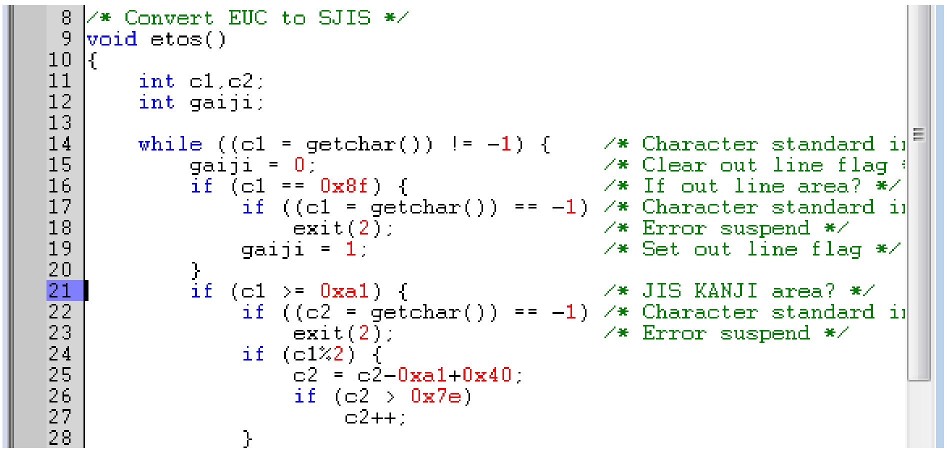

Flowchart Viewer - [etos&0002.vcht - etos]
File Edit View Tool Window Help
View Mode Outline + Program
View Scale 40%
if ((c1 = getchar()) == -1)
FALSE
TRUE
exit(2);
if (c1 >= 0xa1)
if ((c2 = getchar()) == -1)
if (c1&2)
c2 = c2-0xa1+0x40;
c2 = c2-0xa1+0x9f;
if (c1 == 0x8e)
if ((c2 = getchar()) == 0xff)
putchar
putchar(c2);
Ready

3. The flowchart of the function etos() will be displayed. The view scale may be changed from the pull-down menu.

4. Right-click on a procedure box in the flowchart, then select **Goto Source** from the right-click menu.

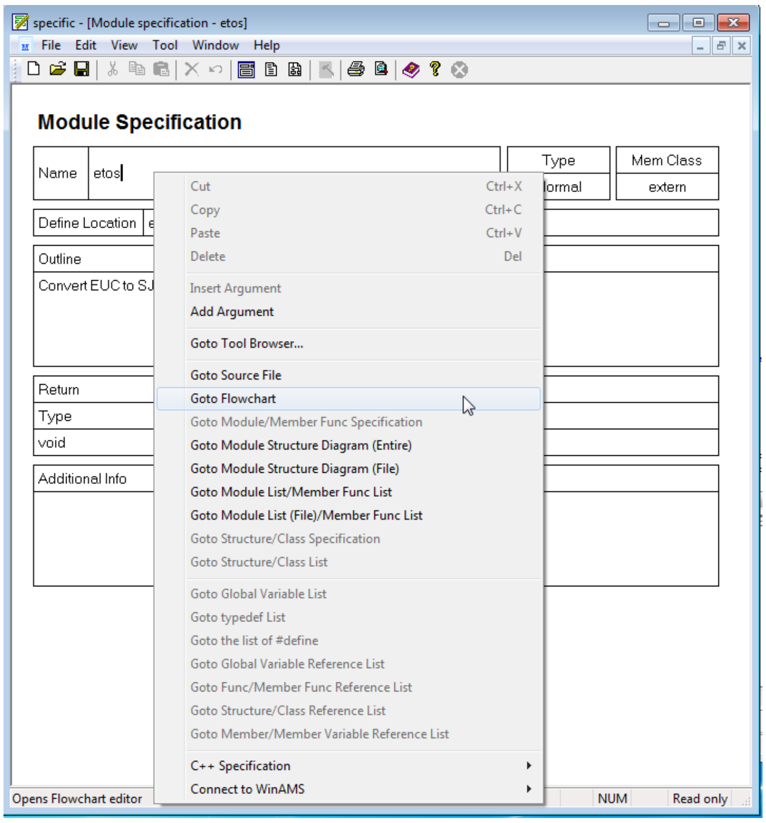


The source code line corresponded to the selected procedure box will be displayed.

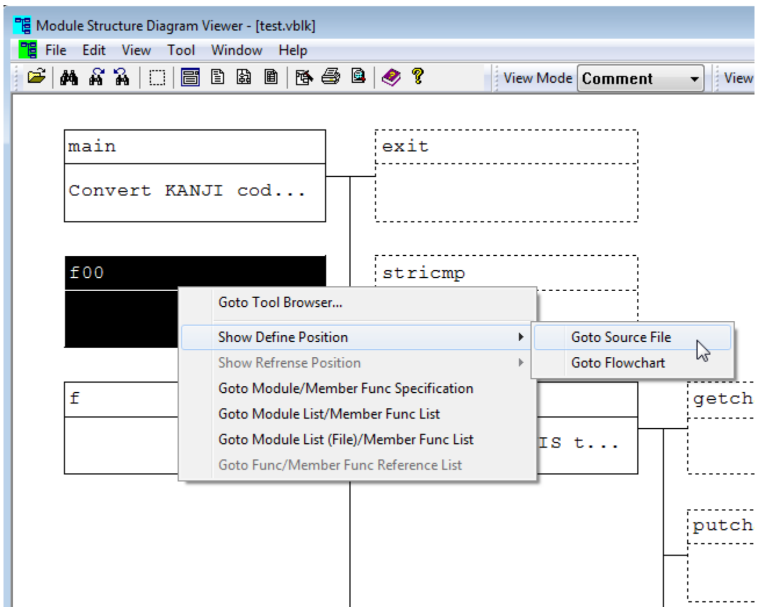


```
/* Convert EUC to SJIS */
void etos()
{
    int c1,c2;
    int gaiji;

    while ((c1 = getchar()) != -1) {     /* Character standard i
        gaiji = 0;                        /* Clear out line flag
        if (c1 == 0x8f) {                 /* If out line area? */
            if ((c1 = getchar()) == -1)  /* Character standard i
                exit(2);                  /* Error suspend */
            gaiji = 1;                    /* Set out line flag */
        }
        if (c1 >= 0xa1) {                 /* JIS KANJI area? */
            if ((c2 = getchar()) == -1)  /* Character standard i
                exit(2);                  /* Error suspend */
            if (c1%2) {
                c2 = c2-0xa1+0x40;
                if (c2 > 0x7e)
                    c2++;
            }
```

Since program documents and the source code are linked each other, the user may easily open program documents and source code from the right-click menu.


specific - [Module specification - etos]
File Edit View Tool Window Help
Module Specification
Name
etos
Type
Mem Class
lormal
extern
Define Location
Outline
Convert EUC to SJ
Return
Type
void
Additional Info
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Delete Del
Insert Argument
Add Argument
Goto Tool Browser...
Goto Source File
Goto Flowchart
Goto Module/Member Func Specification
Goto Module Structure Diagram (Entire)
Goto Module Structure Diagram (File)
Goto Module List/Member Func List
Goto Module List (File)/Member Func List
Goto Structure/Class Specification
Goto Structure/Class List
Goto Global Variable List
Goto typedef List
Goto the list of #define
Goto Global Variable Reference List
Goto Func/Member Func Reference List
Goto Structure/Class Reference List
Goto Member/Member Variable Reference List
C++ Specification
Connect to WinAMS
Opens Flowchart editor
NUM
Read only

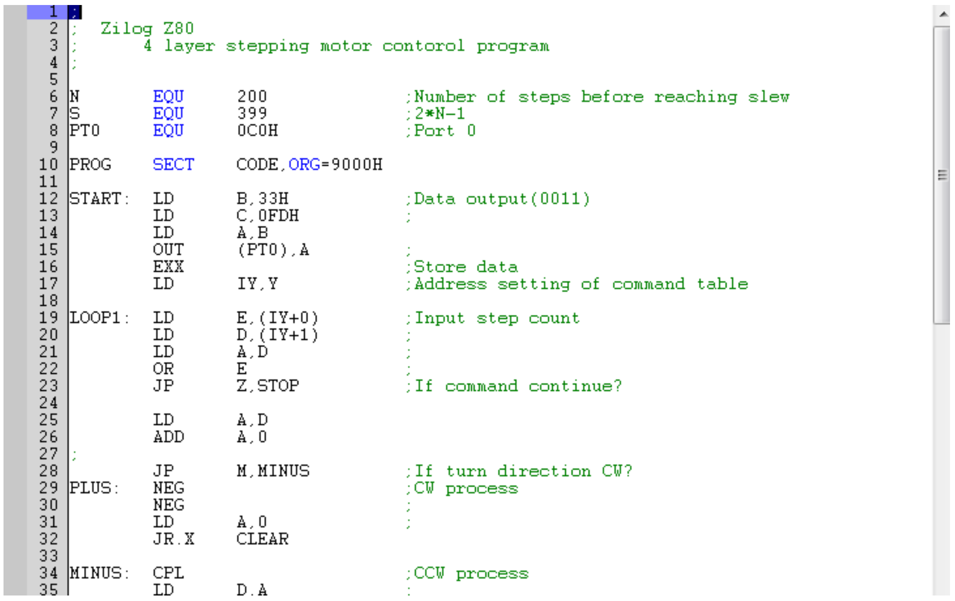
Example: Move from Module Specification to other documents.


Module Structure Diagram Viewer - [test.vblk]
File Edit View Tool Window Help
View Mode Comment
View
main
Convert KANJI cod...
exit
f00
stricmp
f
IS t...
getch
putch
Goto Tool Browser...
Show Define Position
Show Refrense Position
Goto Module/Member Func Specification
Goto Module List/Member Func List
Goto Module List (File)/Member Func List
Goto Func/Member Func Reference List
Goto Source File
Goto Flowchart

Example: Move from Module Structure Diagram to other documents.

Assembly Source Code Flowchart

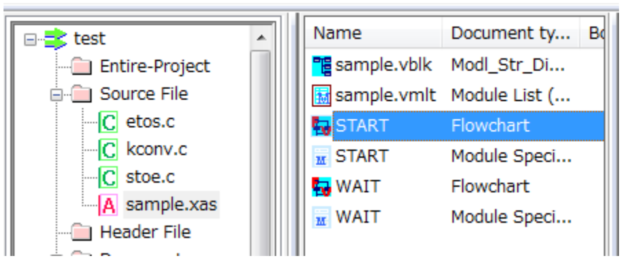
CasePlayer2 can also create program documents from assembly source code. Included in this tutorial project is assembly source code (**sample.xas**) for a Z80 device:



```
;
;   Zilog Z80
;       4 layer stepping motor contorol program
;

N       EQU     200             ;Number of steps before reaching slew
S       EQU     399             ;2*N-1
PT0     EQU     0C0H            ;Port 0

PROG    SECT    CODE,ORG=9000H

START:  LD      B,33H           ;Data output(0011)
        LD      C,0FDH          ;
        LD      A,B
        OUT     (PT0),A         ;
        EXX                     ;Store data
        LD      IY,Y            ;Address setting of command table

LOOP1:  LD      E,(IY+0)        ;Input step count
        LD      D,(IY+1)        ;
        LD      A,D             ;
        OR      E               ;
        JP      Z,STOP          ;If command continue?

        LD      A,D
        ADD     A,0
;
        JP      M,MINUS         ;If turn direction CW?
PLUS:   NEG                     ;CW process
        NEG                     ;
        LD      A,0             ;
        JR.X    CLEAR

MINUS:  CPL                     ;CCW process
        LD      D,A             ;
```

In order to view the flowchart for **sample.xas**:

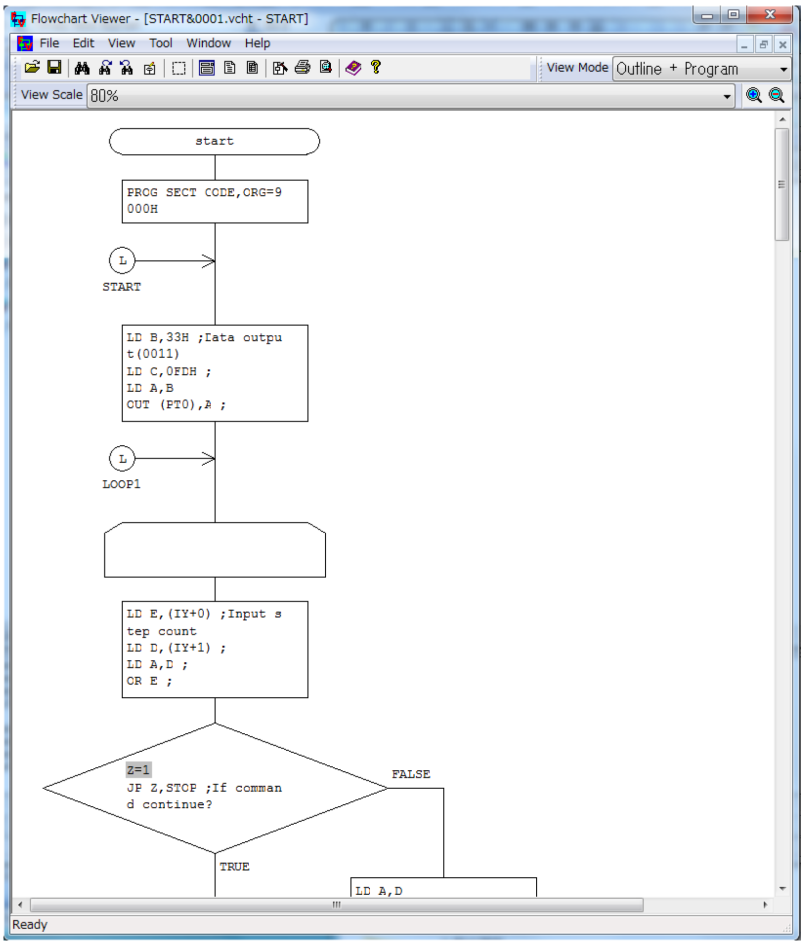
1. Click on **sample.xas** located within the **Source File** folder of the file tree view (**File** tab selected).
2. Double-click **START Flowchart** that appears in the list to the right of the file tree view.



- test
  - Entire-Project
  - Source File
    - etos.c
    - kconv.c
    - stoe.c
    - sample.xas
  - Header File

Name	Document ty...	Bo
sample.vblk	Modl_Str_Di...	
sample.vmlt	Module List (...	
START	Flowchart	
START	Module Speci...	
WAIT	Flowchart	
WAIT	Module Speci...	

The flowchart of the assembly routine START will be displayed.
Note: Since assembly instructions may differ depending on the MPU, the appropriate assembly parameter file (*.rev) should be specified when creating the project (see page 5).



```
start

PROG SECT CODE,ORG=9
000H

L
START

LD B,33H ;Data outpu
t(0011)
LD C,0FDH ;
LD A,B
OUT (PTO),A ;

L
LOOP1

LD E,(IY+0) ;Input s
tep count
LD D,(IY+1) ;
LD A,D ;
OR E ;

Z=1
JP Z,STOP ;If comman
d continue?

FALSE
TRUE
LD A,D
```

Additional sample rev files are located in **C:¥Program Files¥gaio¥CasePlayer2¥sample** for a variety of typical MPUs. If created program documents do not appear correctly for the selected assembly MPU, the .rev parameter file may require modification to match with your assembly code

5. Advanced Mode

Additional program documents, such as global variable lists and structure reference/assignment lists, may be created using the advanced mode. These documents provide a quick reference, and may be useful for locating unexpected read/write global variable and structure related errors.

In the advanced mode all compilable source and header files for the project should be added for analysis. In depth settings are not required in this tutorial project due to the simplicity of the sample code used. However, for actual embedded development projects with a large number of source and header files, detail settings such as pre-processor settings and C-option parameters may need to be set in order to avoid analysis errors caused by language description conflicts. These settings are discussed toward the end of the tutorial.

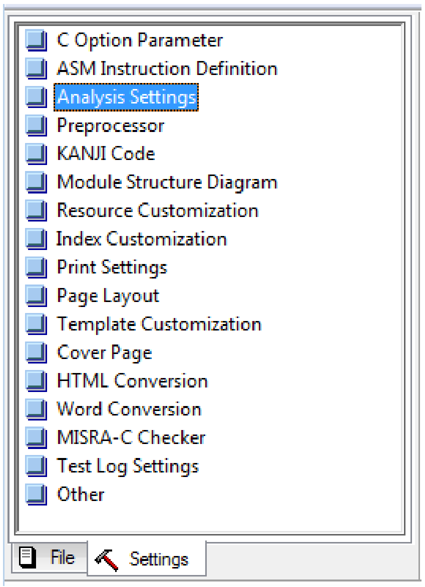
Using the Advanced Mode, the following documents may be created in addition to the documents creatable in Simple Mode.

Global Variable List
typedef list
#define list
Include Relation Diagram
Global Variable Reference List
Function Reference List
Structure Reference List
Member Reference List

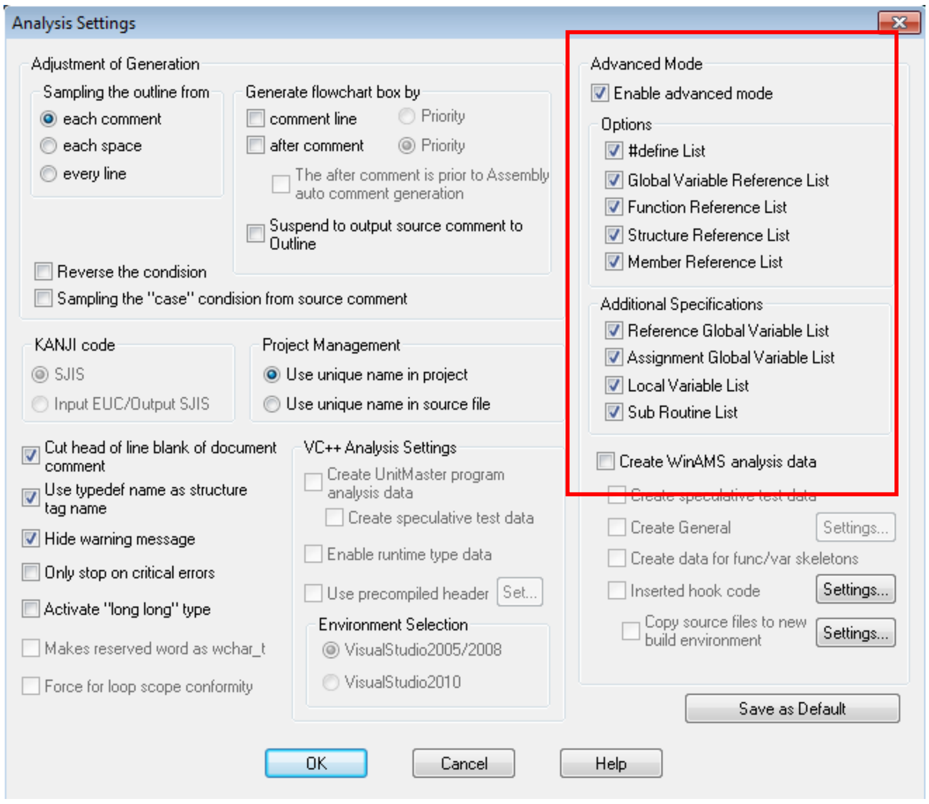
Analysis Settings

In order to enable the Advanced Mode:

3. Click on the **Settings** tab located in the lower portion of the file tree view.
4. Double-click on **Analysis Settings** in the settings list.



5. Check all **Advanced Mode** options.
6. Click **OK**.



Analysis Settings

Adjustment of Generation

Sampling the outline from
- [✓ ●] each comment
- ○ each space
- ○ every line

Generate flowchart box by
- ☐ comment line ○ Priority
- ☐ after comment ◉ Priority
- ☐ The after comment is prior to Assembly auto comment generation
- ☐ Suspend to output source comment to Outline

- ☐ Reverse the condision
- ☐ Sampling the "case" condision from source comment

KANJI code
- ◉ SJIS
- ○ Input EUC/Output SJIS

Project Management
- [✓ ●] Use unique name in project
- ○ Use unique name in source file

- [✓] Cut head of line blank of document comment
- [✓] Use typedef name as structure tag name
- [✓] Hide warning message
- ☐ Only stop on critical errors
- ☐ Activate "long long" type
- ☐ Makes reserved word as wchar_t
- ☐ Force for loop scope conformity

VC++ Analysis Settings
- ☐ Create UnitMaster program analysis data
- ☐ Create speculative test data
- ☐ Enable runtime type data
- ☐ Use precompiled header Set...

Environment Selection
- ◉ VisualStudio2005/2008
- ○ VisualStudio2010

Advanced Mode
- [✓] Enable advanced mode

Options
- [✓] #define List
- [✓] Global Variable Reference List
- [✓] Function Reference List
- [✓] Structure Reference List
- [✓] Member Reference List

Additional Specifications
- [✓] Reference Global Variable List
- [✓] Assignment Global Variable List
- [✓] Local Variable List
- [✓] Sub Routine List

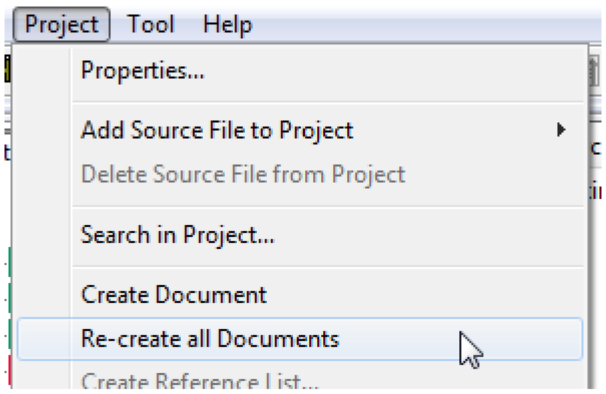
- ☐ Create WinAMS analysis data
- ☐ Create speculative test data
- ☐ Create General Settings...
- ☐ Create data for func/var skeletons
- ☐ Inserted hook code Settings...
- ☐ Copy source files to new build environment Settings...

Save as Default

OK Cancel Help

Create Documents

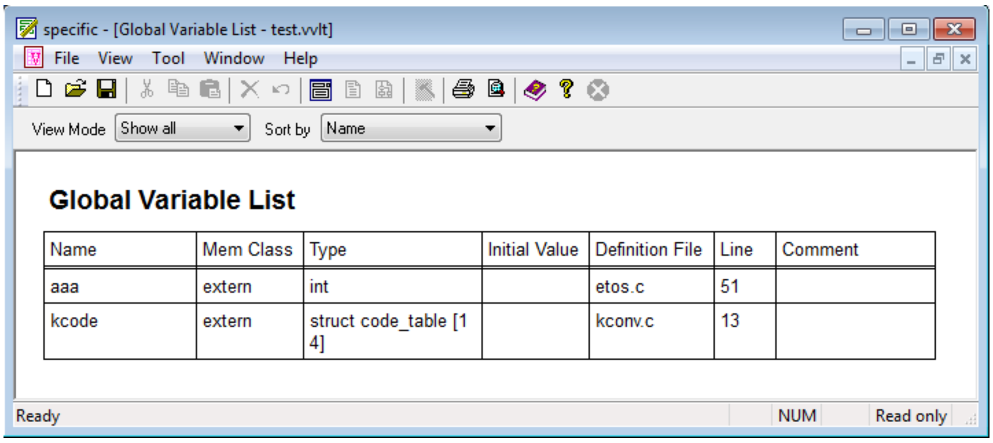
Now analyze the source code and re-create documents by selecting **Project** -> **Re-create all documents** in the application menu.



View the Global Variable List and Global Variable Reference List

The operation to view created program documents is the same as in Simple Mode mentioned previously.

1. Return to the **File** tab view.
2. Select **Global Variable List** located within the **Documents** folder of the file tree view.
3. Double-click **test.vvlt <Entire Project>** that appears in the list to the right of the file tree view.

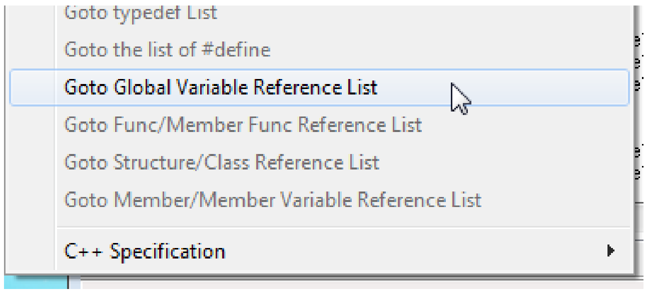


specific - [Global Variable List - test.vvlt]
File View Tool Window Help
View Mode Show all
Sort by Name
Ready
NUM
Read only

Global Variable List

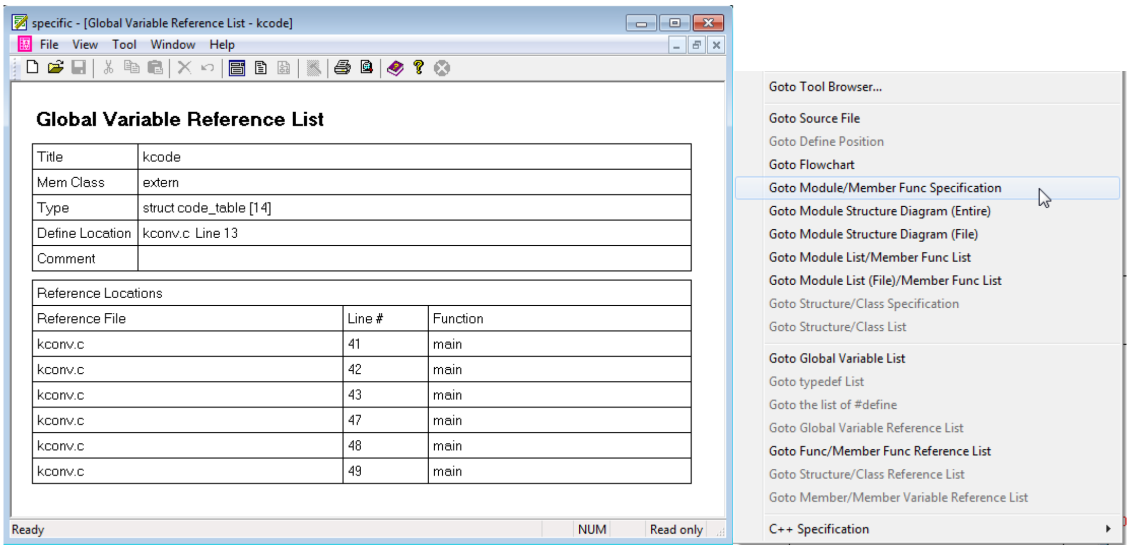
Name	Mem Class	Type	Initial Value	Definition File	Line	Comment
aaa	extern	int		etos.c	51	
kcode	extern	struct code_table [1 4]		kconv.c	13	

4. Right-click on **kcode** in the Global Variable List.
5. Select **Goto Global Variable Reference List** from the right-click menu.



Goto typedef List
Goto the list of #define
Goto Global Variable Reference List
Goto Func/Member Func Reference List
Goto Structure/Class Reference List
Goto Member/Member Variable Reference List
C++ Specification

The Global Variable Reference List shows a list of references to the global variable **kcode** including the source file, line number and function name. Right-click on an item in the list in order to goto the location where **kcode** is referred to within each document.



specific - [Global Variable Reference List - kcode]

File View Tool Window Help

Global Variable Reference List

Title	kcode
Mem Class	extern
Type	struct code_table [14]
Define Location	kconv.c Line 13
Comment	

Reference Locations

Reference File	Line #	Function
kconv.c	41	main
kconv.c	42	main
kconv.c	43	main
kconv.c	47	main
kconv.c	48	main
kconv.c	49	main

Ready NUM Read only

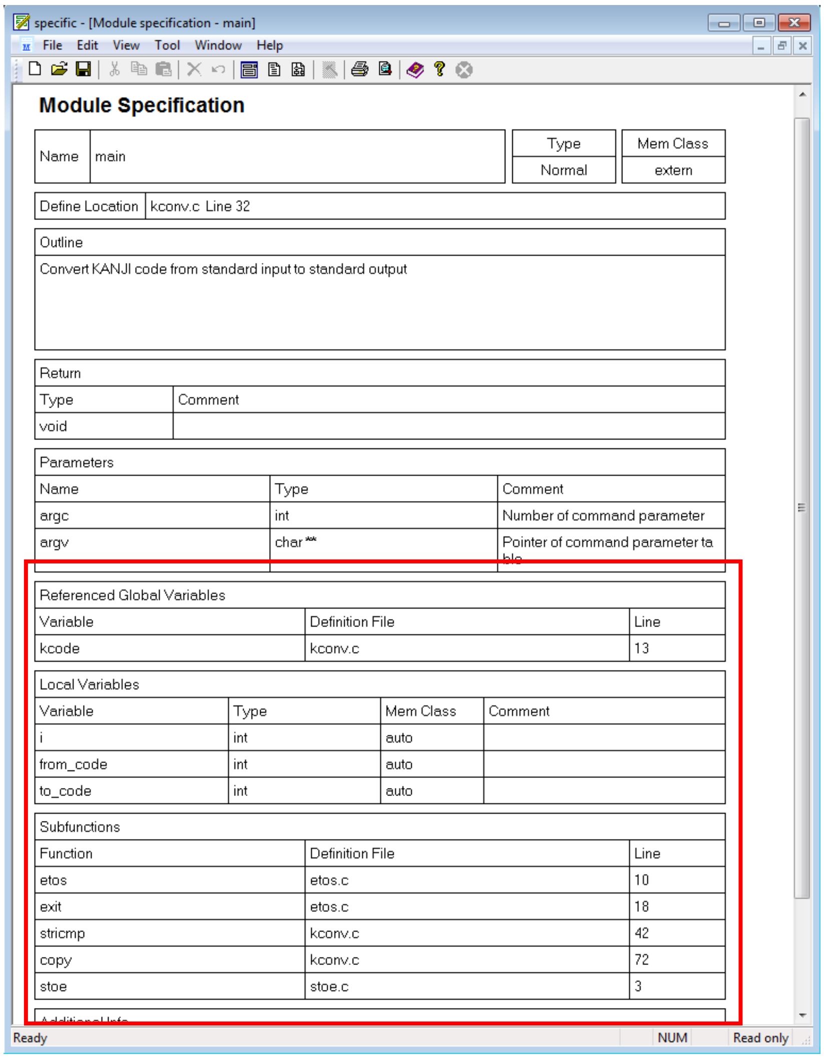
Goto Tool Browser...
Goto Source File
Goto Define Position
Goto Flowchart
Goto Module/Member Func Specification
Goto Module Structure Diagram (Entire)
Goto Module Structure Diagram (File)
Goto Module List/Member Func List
Goto Module List (File)/Member Func List
Goto Structure/Class Specification
Goto Structure/Class List
Goto Global Variable List
Goto typedef List
Goto the list of #define
Goto Global Variable Reference List
Goto Func/Member Func Reference List
Goto Structure/Class Reference List
Goto Member/Member Variable Reference List
C++ Specification

View Additional Variable Reference Information

Additional variable reference information will be added to the Module Specification document when using the Advanced Mode.

1. Select **Module Specification** located within the **Documents** folder of the file tree view.
2. Double-click **main** that appears in the list to the right of the file tree view.

The following additional sections are included when documents are created using the Advanced Mode: Referenced Global Variables, Local Variable, and Subfunctions.



Module Specification

Name	main	Type	Mem Class
		Normal	extern

Define Location	kconv.c Line 32

Outline
Convert KANJI code from standard input to standard output

Return	
Type	Comment
void	

Parameters		
Name	Type	Comment
argc	int	Number of command parameter
argv	char **	Pointer of command parameter ta ble

Referenced Global Variables		
Variable	Definition File	Line
kcode	kconv.c	13

Local Variables			
Variable	Type	Mem Class	Comment
i	int	auto	
from_code	int	auto	
to_code	int	auto	

Subfunctions		
Function	Definition File	Line
etos	etos.c	10
exit	etos.c	18
stricmp	kconv.c	42
copy	kconv.c	72
stoe	stoe.c	3

Additional Info

This completes the program document creation portion of the tutorial. Topics covered included the two analysis modes, Simple Mode and Advanced Mode, and moving to and from program documents and source code files using the right-click menu.

6. MISRA-C Rule Checker

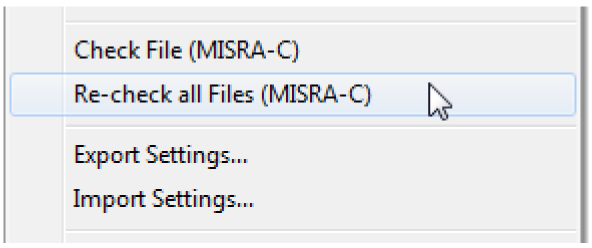
MISRA-C is a software development standard for the C programming language developed by MISRA (Motor Industry Software Reliability Association) to facilitate code safety, portability and reliability in the embedded software.

This portion of the tutorial will discuss the MISRA-C rule check features in CasePlayer2. MISRA-C rule sets supported include MISRA-C 1998 and MISRA-C 2004.

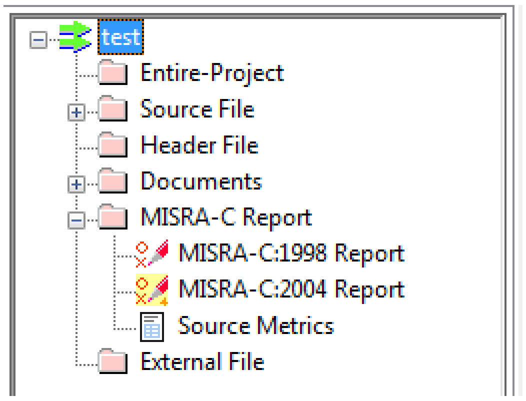
Perform the MISRA-C Rule Check

The MISRA-C rule check feature requires that the CasePlayer2 analysis setting to be set to Advanced Mode which was performed previously in this tutorial. We will continue to use the same sample project (test.vproj) setup in the steps prior to this section of the tutorial.

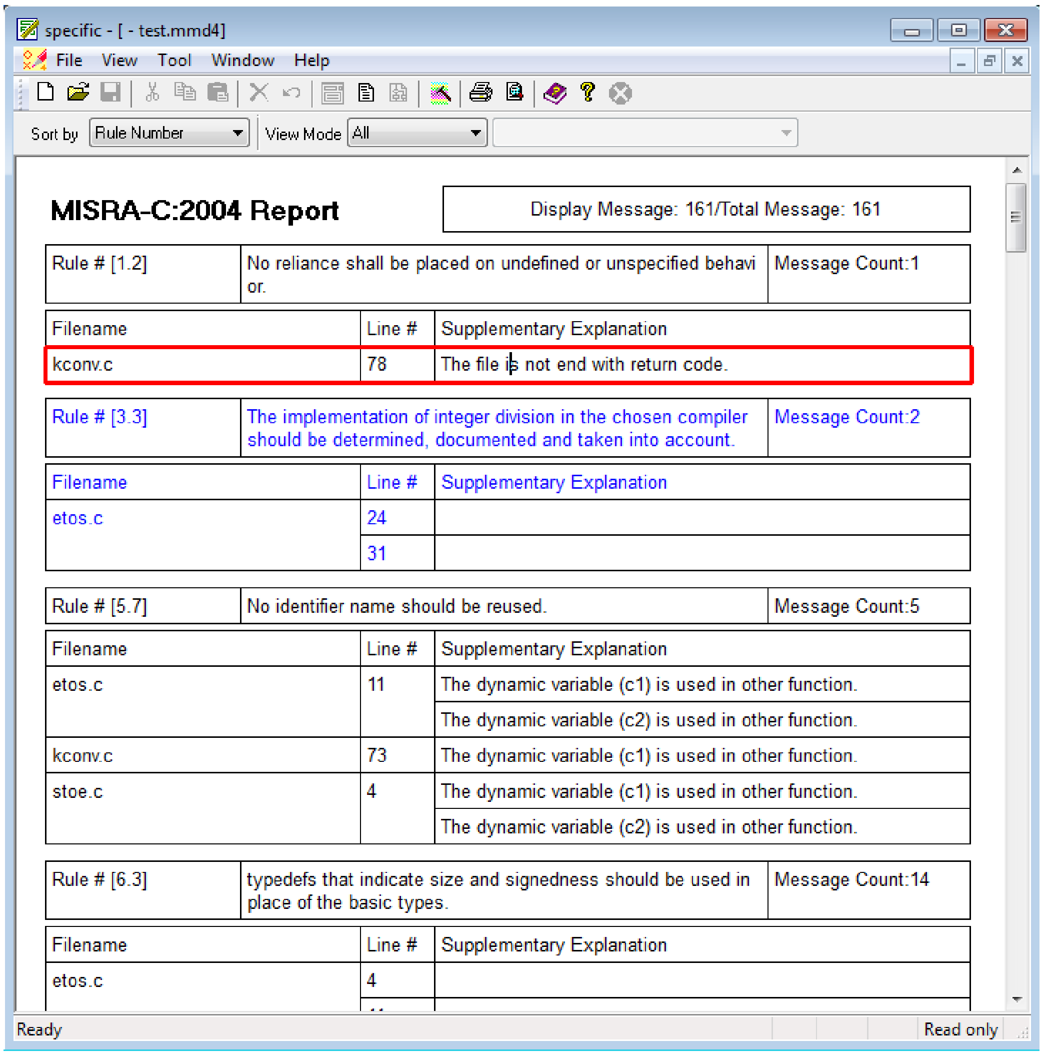
1. From the application menu select **Project** -> **Re-check all files (MISRA-C)**.



2. The rule check results will be created in the **MISRA-C Report** folder in the file tree view.



3. Double-click **MISRA-C:2004 Report** to open the report for viewing.



MISRA-C:2004 Report

Display Message: 161/Total Message: 161

Rule # [1.2]	No reliance shall be placed on undefined or unspecified behavior.	Message Count:1

Filename	Line #	Supplementary Explanation
kconv.c	78	The file is not end with return code.

Rule # [3.3]	The implementation of integer division in the chosen compiler should be determined, documented and taken into account.	Message Count:2

Filename	Line #	Supplementary Explanation
etos.c	24	
	31	

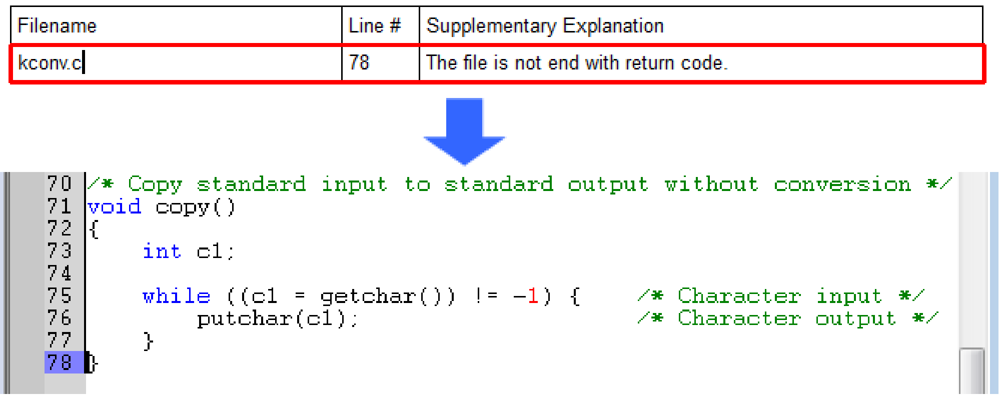
Rule # [5.7]	No identifier name should be reused.	Message Count:5

Filename	Line #	Supplementary Explanation
etos.c	11	The dynamic variable (c1) is used in other function.
		The dynamic variable (c2) is used in other function.
kconv.c	73	The dynamic variable (c1) is used in other function.
stoe.c	4	The dynamic variable (c1) is used in other function.
		The dynamic variable (c2) is used in other function.

Rule # [6.3]	typedefs that indicate size and signedness should be used in place of the basic types.	Message Count:14

Filename	Line #	Supplementary Explanation
etos.c	4	

The report shows a list of the MISRA-C incompatibilities including rule #, source filename, source line number, and a description. Double-click on an item in the list in order to goto the source code line.



Filename	Line #	Supplementary Explanation
kconv.c	78	The file is not end with return code.

```
70 /* Copy standard input to standard output without conversion */
71 void copy()
72 {
73     int c1;
74 
75     while ((c1 = getchar()) != -1) {     /* Character input */
76         putchar(c1);                     /* Character output */
77     }
78 }
```

Display Options

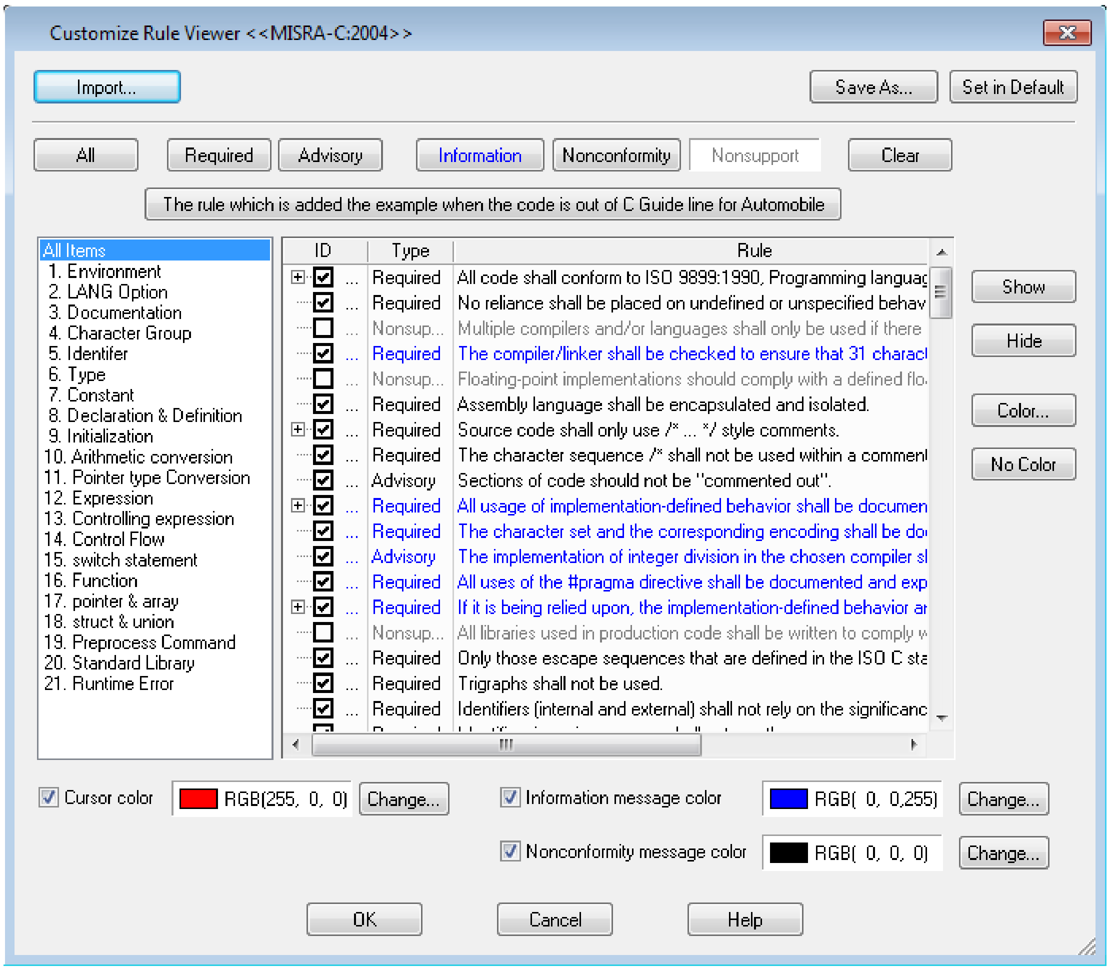
Sort by - Rule Number: The report will be sorted by rule number.
Sort by - Source Point: The report will be sorted by source line number for each source file.

View Mode – All: All rule information will be displayed.
View Mode – Simple: Displays a simplified version of the rule information.
View Mode - Part Message: Displays check information per selected rule number. Note: this option is only available when using the Sort by – Rule Number option.

Rule Selection

The rules to be checked for may be selected using the following dialog.

1. Double-click on the **MISRA-C:2004 Report** (skip this step if already open).
2. From the application menu select **Tool** -> **Set Display Rule**.



All Items
1. Environment
2. LANG Option
3. Documentation
4. Character Group
5. Identifer
6. Type
7. Constant
8. Declaration & Definition
9. Initialization
10. Arithmetic conversion
11. Pointer type Conversion
12. Expression
13. Controlling expression
14. Control Flow
15. switch statement
16. Function
17. pointer & array
18. struct & union
19. Preprocess Command
20. Standard Library
21. Runtime Error

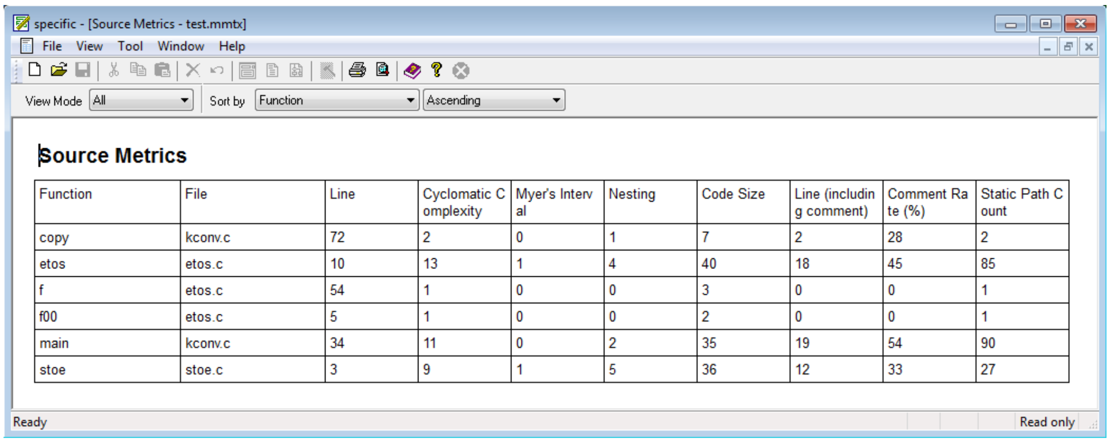
ID	Type	Rule
✓ ...	Required	All code shall conform to ISO 9899:1990, Programming languag
✓ ...	Required	No reliance shall be placed on undefined or unspecified behav
☐ ...	Nonsup...	Multiple compilers and/or languages shall only be used if there
✓ ...	Required	The compiler/linker shall be checked to ensure that 31 charact
☐ ...	Nonsup...	Floating-point implementations should comply with a defined flo
✓ ...	Required	Assembly language shall be encapsulated and isolated.
✓ ...	Required	Source code shall only use /* ... */ style comments.
✓ ...	Required	The character sequence /* shall not be used within a comment
✓ ...	Advisory	Sections of code should not be "commented out".
✓ ...	Required	All usage of implementation-defined behavior shall be documen
✓ ...	Required	The character set and the corresponding encoding shall be do
✓ ...	Advisory	The implementation of integer division in the chosen compiler sl
✓ ...	Required	All uses of the #pragma directive shall be documented and exp
✓ ...	Required	If it is being relied upon, the implementation-defined behavior ar
☐ ...	Nonsup...	All libraries used in production code shall be written to comply w
✓ ...	Required	Only those escape sequences that are defined in the ISO C sta
✓ ...	Required	Trigraphs shall not be used.
✓ ...	Required	Identifiers (internal and external) shall not rely on the significanc

☑ Cursor color 🟥 RGB(255, 0, 0) Change...
☑ Information message color 🟦 RGB(0, 0,255) Change...
☑ Nonconformity message color ⬛ RGB(0, 0, 0) Change...

Select a category on the left side to only display rules related to that category.
Rules checked in the dialog will be checked for in the MISRA-C rule check report. Rules unchecked will not be displayed in the MISRA-C rule check report.

View Source Code Metrics

MISRA source code metrics may be viewed by double-clicking on **Source Metrics** located within the **MISRA-C Report** folder of the file tree view.


specific - [Source Metrics - test.mmtx]
File View Tool Window Help
Ready
Read only

View Mode: All
Sort by: Function, Ascending

Source Metrics

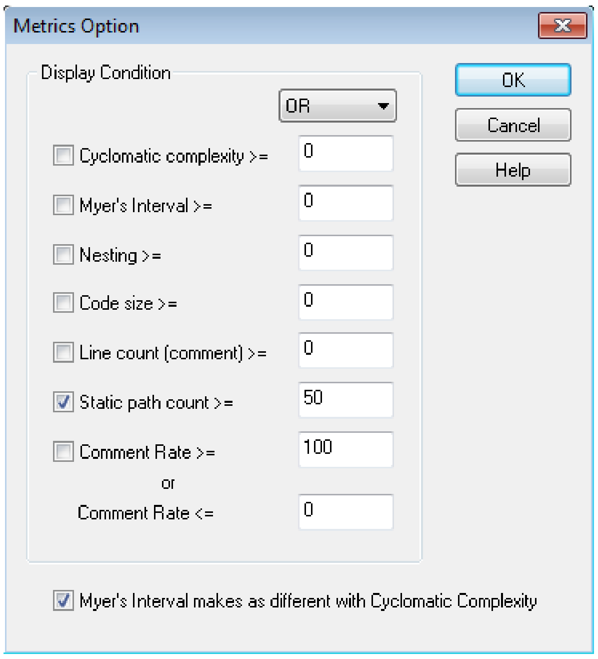
Function	File	Line	Cyclomatic Complexity	Myer's Interval	Nesting	Code Size	Line (including comment)	Comment Rate (%)	Static Path Count
copy	kconv.c	72	2	0	1	7	2	28	2
etos	etos.c	10	13	1	4	40	18	45	85
f	etos.c	54	1	0	0	3	0	0	1
f00	etos.c	5	1	0	0	2	0	0	1
main	kconv.c	34	11	0	2	35	19	54	90
stoe	stoe.c	3	9	1	5	36	12	33	27

The following information is displayed for each function in the source metrics report:

Function: Function name.
File: Source file name.
Line: Source file line number the function is defined at.
Cyclomatic Complexity: Number of linearly independent paths through the function.
Myer's Interval: Cyclomatic Complexity plus ‘?;’ , ‘&&’ and ‘||’ operators.
Nesting: Maximum number of nests.
Code Size: Number of lines from ‘{‘ to ‘}’ of the function.
Line (including comment): Number of comment lines.
Comment Rate (%): The rate (percentage) of lines that are comment lines.
Static Path Count: The maximum number of possible paths in the function.

The results may be filtered by setting a minimum threshold for the various metrics by clicking **Tool** -> **Display Option** from the application menu.

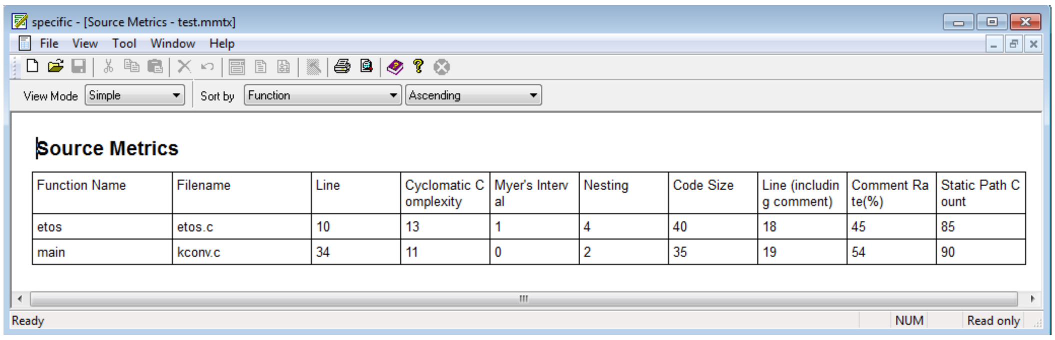
For example, only check the **Static path count** option, and set the value to **50** to only show items that have a static path count of 50 or greater in the source metrics list.



☐ Cyclomatic complexity >=	0
☐ Myer's Interval >=	0
☐ Nesting >=	0
☐ Code size >=	0
☐ Line count (comment) >=	0
☑ Static path count >=	50
☐ Comment Rate >=	100
or	
Comment Rate <=	0

☑ Myer's Interval makes as different with Cyclomatic Complexity

The filtered result will be displayed as:


specific - [Source Metrics - test.mmtx]
File View Tool Window Help
View Mode Simple
Sort by Function
Ascending
Ready
NUM
Read only

Source Metrics

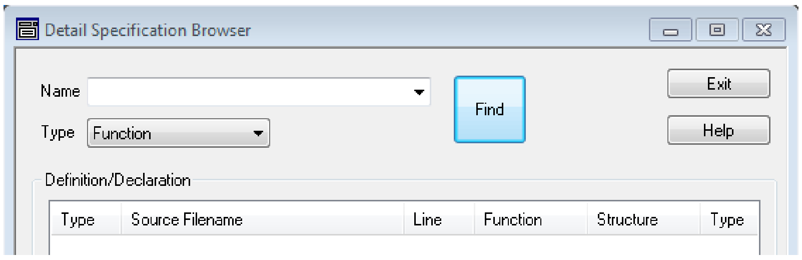
Function Name	Filename	Line	Cyclomatic Complexity	Myer's Interval	Nesting	Code Size	Line (including comment)	Comment Rate(%)	Static Path Count
etos	etos.c	10	13	1	4	40	18	45	85
main	kconv.c	34	11	0	2	35	19	54	90

This completes the MISRA-C rule check portion of the tutorial.

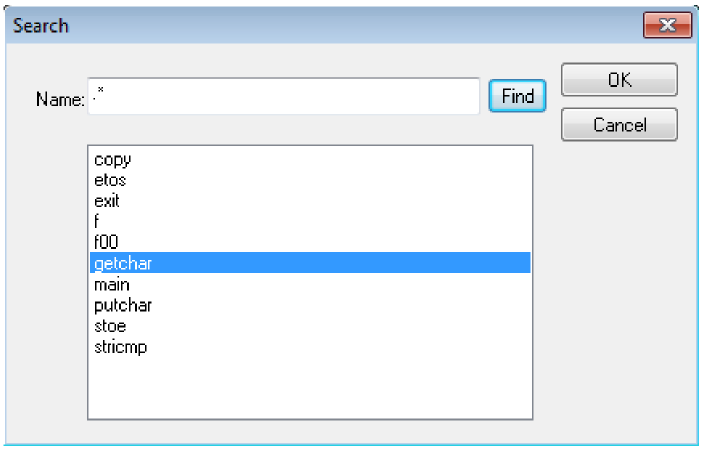
7. Search Feature

The **Detail Specification Browser** makes it possible to easily search for a variety of items.

1. Open the tutorial project **test.vproj** used throughout this tutorial.
2. From the application menu select **Tool** -> **Goto Tool Browser...**
3. Select **Function** as the Type.
4. Leave the Name field blank.
5. Click the **Find** button.



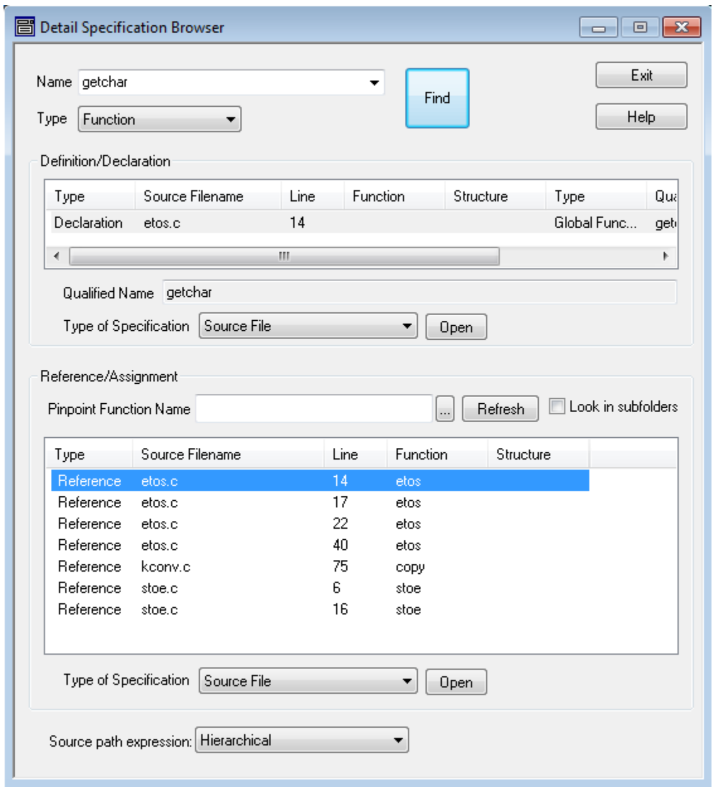
The functions located in the project will be displayed.



Name: .*

copy
etos
exit
f
f00
getchar
main
putchar
stoe
stricmp

6. Select **getchar** from the function list and click **OK**



Name: getchar
Type: Function

Definition/Declaration

Type	Source Filename	Line	Function	Structure	Type	Qu
Declaration	etos.c	14			Global Func...	get

Qualified Name: getchar
Type of Specification: Source File

Reference/Assignment

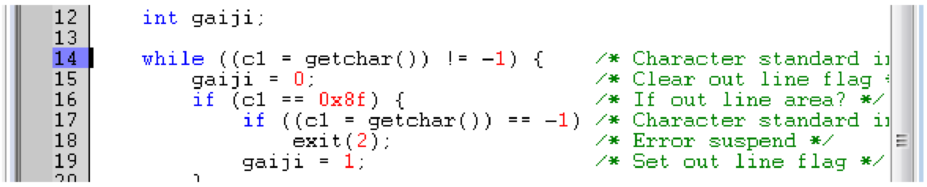
Pinpoint Function Name:

Type	Source Filename	Line	Function	Structure
Reference	etos.c	14	etos	
Reference	etos.c	17	etos	
Reference	etos.c	22	etos	
Reference	etos.c	40	etos	
Reference	kconv.c	75	copy	
Reference	stoe.c	6	stoe	
Reference	stoe.c	16	stoe	

Type of Specification: Source File

Source path expression: Hierarchical

Information regarding the getchar function will be displayed such as the definition/declaration location and reference/assignments. Double-click on a result to display the relevant source file line.



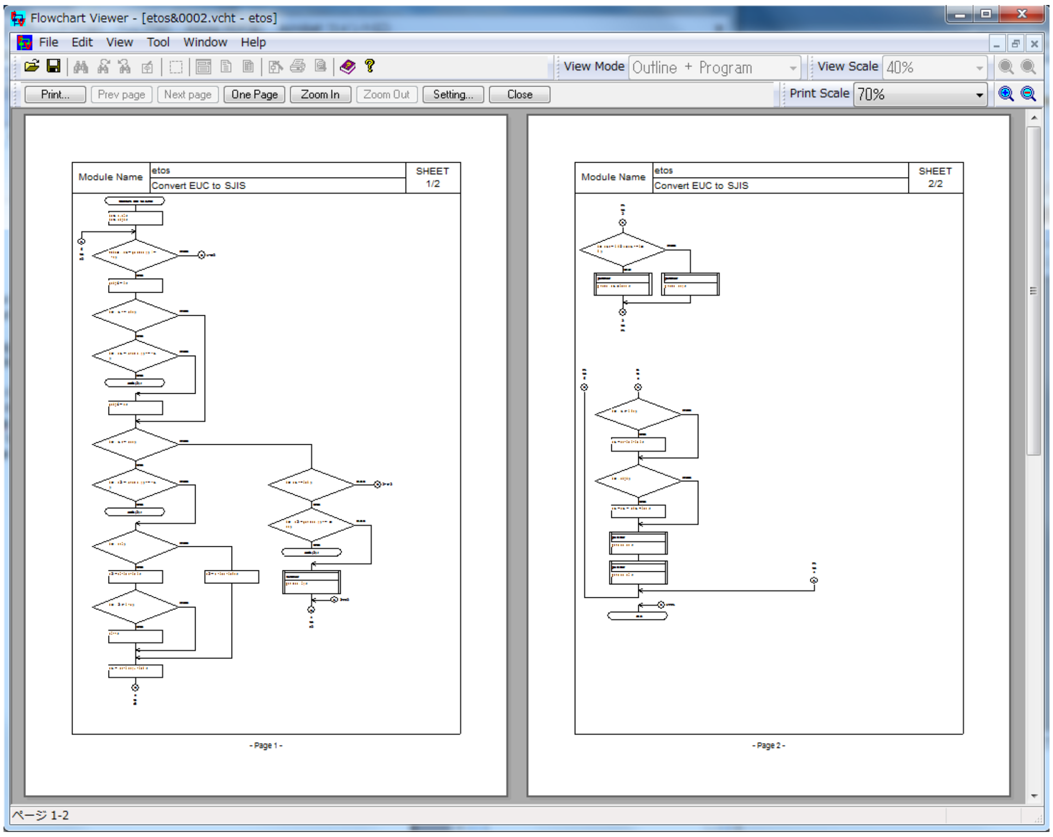
```
    int gaiji;

    while ((c1 = getchar()) != -1) {     /* Character standard i
        gaiji = 0;                       /* Clear out line flag
        if (c1 == 0x8f) {                /* If out line area? */
            if ((c1 = getchar()) == -1)  /* Character standard i
                exit(2);                 /* Error suspend */
            gaiji = 1;                   /* Set out line flag */
        }
```

A variety of other items may be searched for by selecting the appropriate type in the Detail Specification Browser.

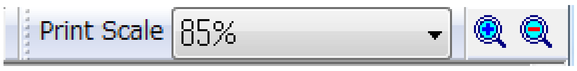
8. Printing Features

CasePlyer2 includes a **Print Preview** feature for customizing and verifying the print layout before actually printing. In particular this can be useful for minimizing wasted white space and preventing documents from using multiple pages.

1. Click on the **File** tab located in the lower portion of the file tree view.
2. Click on **Flowchart** within the **Documents** folder.
3. Double-click on **etos** in the right selection box.
4. From the Flowchart Viewer's menu select **File** -> **Print Preview**.
5. Click the **Two Page** button located under the menu bar.


Flowchart Viewer - [etos&0002.vcht - etos]
File Edit View Tool Window Help
View Mode Outline + Program
View Scale 40%
Print... Prev page Next page One Page Zoom In Zoom Out Setting... Close
Print Scale 70%
Module Name etos Convert EUC to SJIS SHEET 1/2
Module Name etos Convert EUC to SJIS SHEET 2/2
- Page 1 -
- Page 2 -
ページ 1-2

6. To the right of the **Print Scale** selection, click the **(+)** or **(-)** magnifier buttons in order to increase or decrease the scale.

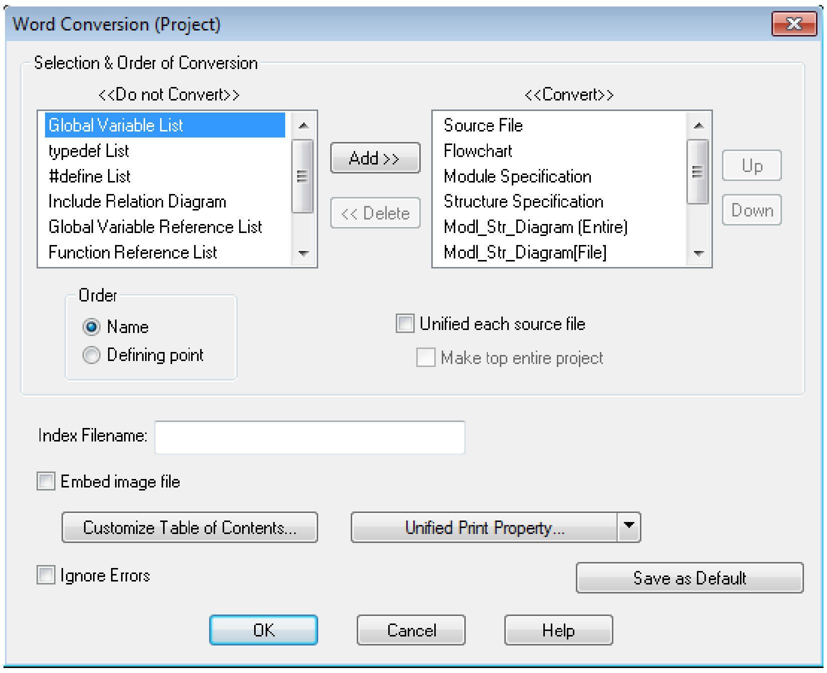


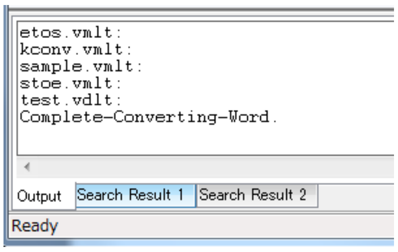
The print preview will be automatically updated to reflect the print scale setting as it is modified. This way the user may verify the print view prior to printing.

9. Create Microsoft Word Format Documents

Using a built in feature, CasePlayer2 created program documents may be converted to MS-Word .doc files using the following method:

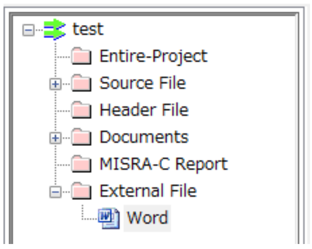
1. From the application menu select **Project** -> **Convert to Word…**
2. Add the type of documents to convert to the **<<Convert>>** list to the right.
3. Click **OK**.



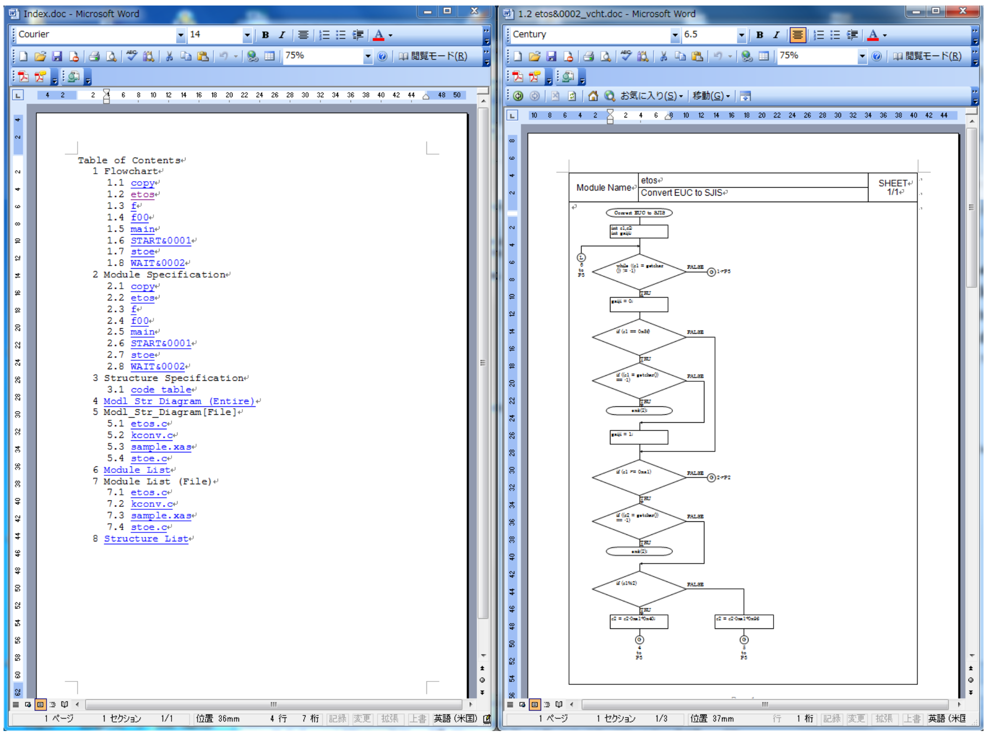


```
etos.vmlt:
kconv.vmlt:
sample.vmlt:
stoe.vmlt:
test.vdlt:
Complete-Converting-Word.
```

4. Expand the **External File** folder in the file tree view.
5. Double-click on **Word**.



A word document with links to the word format program documents will be opened. Hold down the CTRL key and click on a link to open the linked document.



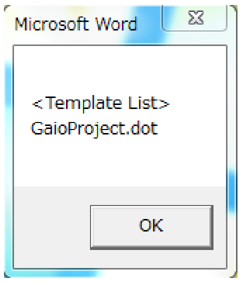
Index.doc - Microsoft Word

```
Table of Contents
  1 Flowchart
    1.1 copy
    1.2 etos
    1.3 f
    1.4 f00
    1.5 main
    1.6 START&0001
    1.7 stoe
    1.8 WAIT&0002
  2 Module Specification
    2.1 copy
    2.2 etos
    2.3 f
    2.4 f00
    2.5 main
    2.6 START&0001
    2.7 stoe
    2.8 WAIT&0002
  3 Structure Specification
    3.1 code table
  4 Modl Str Diagram (Entire)
  5 Modl_Str_Diagram[File]
    5.1 etos.c
    5.2 kconv.c
    5.3 sample.xas
    5.4 stoe.c
  6 Module List
  7 Module List (File)
    7.1 etos.c
    7.2 kconv.c
    7.3 sample.xas
    7.4 stoe.c
  8 Structure List
```

1.2 etos&0002_vcht.doc - Microsoft Word

Module Name	etos Convert EUC to SJIS	SHEET 1/1

Notes:
Microsoft Word must be installed in order to use this feature.
Close all Microsoft word documents before performing the word conversion operation.
If a <Template List> error occurs during conversion, change the Microsoft Word security settings as described below.



Microsoft Word

<Template List>
GaioProject.dot

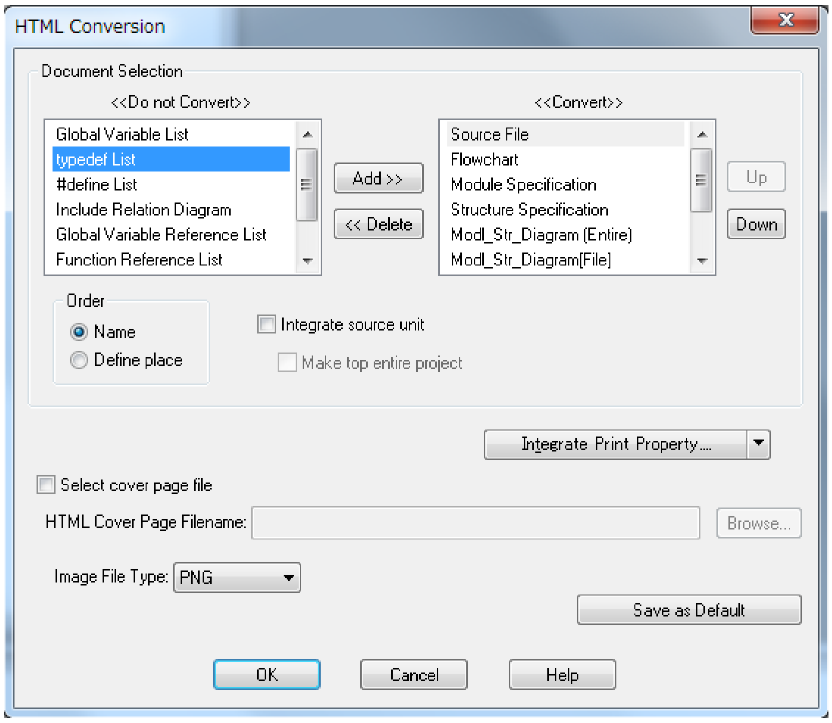
OK

Enable the Trust Access to Visual Basic Project option from MS Word menu: **Tools** -> **Macro** -> **Security** -> **Trusted Publishers**.

10. Create HTML Format Documents

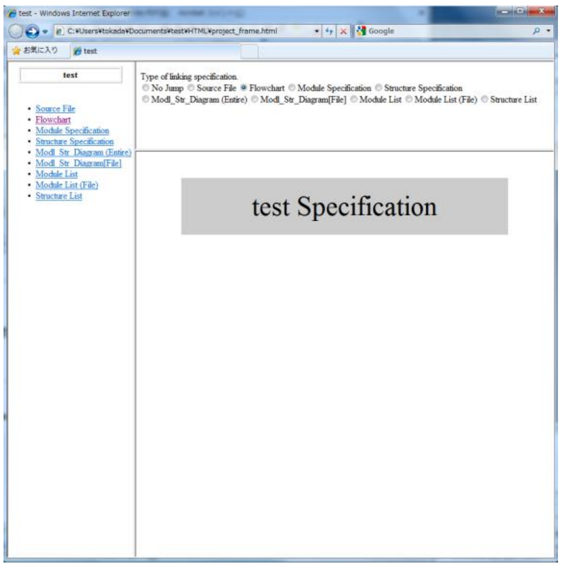
Using a built in feature, CasePlayer2 created program documents may be converted to.html files using the following method:

1. From the application menu select **Project** -> **Convert to HTML…**
2. Add the type of documents to convert to the **<<Convert>>** list to the right.
3. Click **OK**.



4. Expand the **External File** folder in the file tree view.
5. Double-click on **HTML** in the file tree view.

An HTML file with links to the HTML format program documents will be opened. Click on the links in order to open the linked HTML files.



test - Windows Internet Explorer

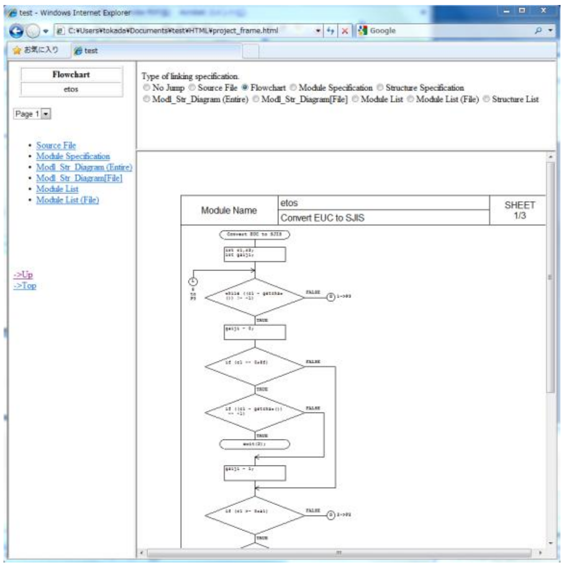
C:¥Users¥tokada¥Documents¥test¥HTML¥project_frame.html

test

- Source File
- Flowchart
- Module Specification
- Structure Specification
- Modl_Str_Diagram (Entire)
- Modl_Str_Diagram[File]
- Module List
- Module List (File)
- Structure List

Type of linking specification.
No Jump / Source File / [✓ Flowchart] / Module Specification / Structure Specification / Modl_Str_Diagram (Entire) / Modl_Str_Diagram[File] / Module List / Module List (File) / Structure List

test Specification



test - Windows Internet Explorer

C:¥Users¥tokada¥Documents¥test¥HTML¥project_frame.html

Flowchart

etos

Page 1

- Source File
- Module Specification
- Modl_Str_Diagram (Entire)
- Modl_Str_Diagram[File]
- Module List
- Module List (File)

->Up
->Top

Type of linking specification.
No Jump / Source File / [✓ Flowchart] / Module Specification / Structure Specification / Modl_Str_Diagram (Entire) / Modl_Str_Diagram[File] / Module List / Module List (File) / Structure List

Module Name	etos Convert EUC to SJIS	SHEET 1/3

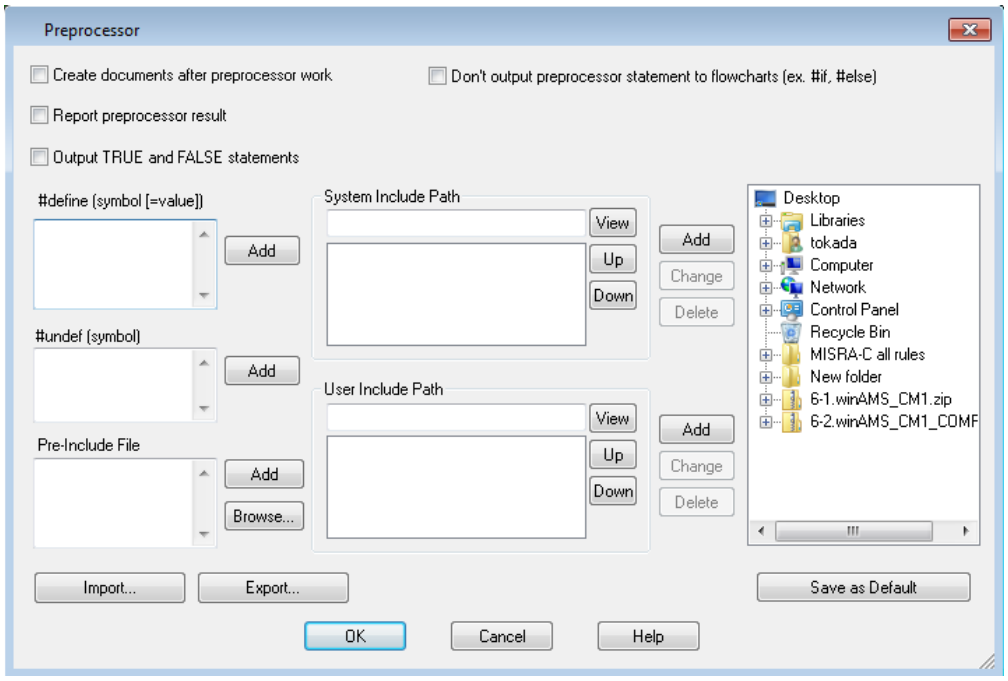
11. Detailed Settings for Advanced Mode

As additional reference this section will explain some of the detailed settings for use with the Advanced (document creation) Mode. These settings are not required for the tutorial project due to the simplicity of the sample code used. However, for actual embedded development projects these detailed settings may be required.

Preprocessor Settings

In order to analyze source code that includes header files, preprocessor settings configured in the IDE will be required for CasePlayer2 as well. These includes the #define definition and the include path information.

1. Click on the **Settings** tab located in the lower portion of the file tree view.
2. Double-click on **Preprocessor**.


Preprocessor

☐ Create documents after preprocessor work
☐ Don't output preprocessor statement to flowcharts (ex. #if, #else)
☐ Report preprocessor result
☐ Output TRUE and FALSE statements

#define (symbol [=value])

Define macros should be specified here. After entering an item click the **Add** button or press CTRL + ENTER in order to go to a new line to add additional items.

#undef (Symbol)

Existing #define macros may be undefined by entering them here.

Pre-Include File

Enter a header file here in order to include it before other source files. A pre-include file that describes implicit definitions may be entered in order to avoid CasePlayer2 analysis errors when the cross compiler has implicit key words or definitions.

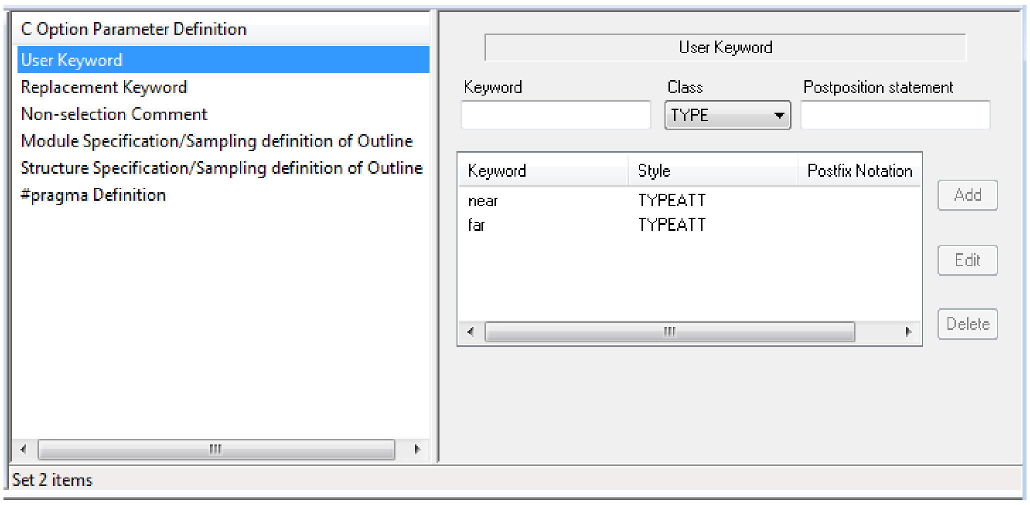
System Include Path / User Include Path

Enter system include paths or user include paths in these fields. The paths may be selected using the tree view to the right then clicking the Add button. Since Subfolders are not included each folder must be specified individually.

C Option Parameter

C-option parameters may need to be set in order to avoid the analysis errors caused by incompatibilities in the code with the ANSI-C standard. To access the C Option Parameter settings:

1. Click on the **Settings** tab located in the lower portion of the file tree view.
2. Double-click on **C Option Parameter**.



C Option Parameter Definition
- User Keyword
- Replacement Keyword
- Non-selection Comment
- Module Specification/Sampling definition of Outline
- Structure Specification/Sampling definition of Outline
- #pragma Definition

User Keyword

Keyword: (blank) Class: TYPE Postposition statement: (blank)

Keyword	Style	Postfix Notation
near	TYPEATT	
far	TYPEATT	

Set 2 items

User Keyword

ANSI-C incompatibilities or compiler specific keywords may be fixed here. The following examples are typical settings to avoid the analysis errors:

Example #1: int near p1;

Using the **TYPEATT** style for the **near** keyword will cause near keywords to be analyzed as a type instruction modifier.
Keyword: near, Style: TYPEATT, Postfix Notation: [**leave blank**]

Example #2: direct int array[100];

Using the **STRAGE** style for the **direct** keyword will cause the direct keyword to be analyzed as a storage class instruction modifier.
Keyword: direct, Style: STRAGE, Postfix Notation: [**leave blank**]

Example #3: __asm (“ .... “)

Using the **ASM** style for the **__asm** keyword will cause the __asm keyword to be analyzed as an inline assembler description.
Keyword: __asm, Style: ASM, Postfix Notation: (EXPRESSION)

Example #4: __except (.......)

Using the **IGNORE** style for the **__except** keyword will cause the __except keyword to be ignored during analysis.
Keyword: __except, Style: IGNORE, Postfix Notation: (EXPRESSION)

Example #5: INT32 val;
Using the **TYPE** style for the **INT32** keyword will cause the INT32 keyword to be analyzed as a type definition.
Keyword: INT32, Style: TYPE, Postfix Notation: [**leave blank**]

Replacement Keyword

Use to replace keywords in the code.

Example #1: typedef __WCHAR_T_TYPE__ _Wchart;
__**WCHAR_T_TYPE**__ can be replaced by **int** in order to avoid errors.
New Keyword: int, Existing Keyword: __WCHAR_T_TYPE__

Example #2: typedef __SIZE_T_TYPE__ _Sizet;
__**SIZE_T_TYPE**__ can be replaced by **int** in order to avoid errors.
New Keyword: int, Existing Keyword: __SIZE_T_TYPE__

Other Settings in C Option Parameter

Most errors can be fixed using the above described User Keywords and Replacement Keywords. Additional settings found in C Option Parameter include:

Non-selection Comment
Module Specification/Sampling definition of Outline
Structure Specification/Sampling definition of Outline
#pragma Definition

12. Conclusion

This concludes the CasePlayer2 tutorial. We hope that the information provided in this tutorial has helped aid you in better understanding how to use GAIO's CasePlayer2 static analysis and program document creation tool.

Copyright Notice and Disclaimer