To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

H8S/2456

GUI Application to Control QVGA LCD and Touch Screen Using EXDMAC and A/D Converter Based on H8S/2456

Introduction

This application note describes a GUI Application for controlling a QVGA LCD and touch screen by using the EXDMAC and A/D converter of a H8S/2454, H8S/2456, H8S/2426 or H8S/2424 product.

Target Device

H8S/2456, H8S/2426, H8S/2454, H8S/2424

Contents

1. Overview

It is able to design H8S/2456 software for Display and Touch screen by using GUI Tools running on PC site.

The GUI tool has the following features:

(1) By using the form designer function on Visual Studio, WYSIWYG design as shown in Figure 1 is capable on the GUI screen.
(2) It is able to generate C source program from display form made by Visual Studio.
(3) It can be confirmed the status by building the generated program on PC as shown in Figure 2.
(4) Once operation has been confirmed on the PC, porting the program simply requires rebuilding it for the target processor (see Figure 3).

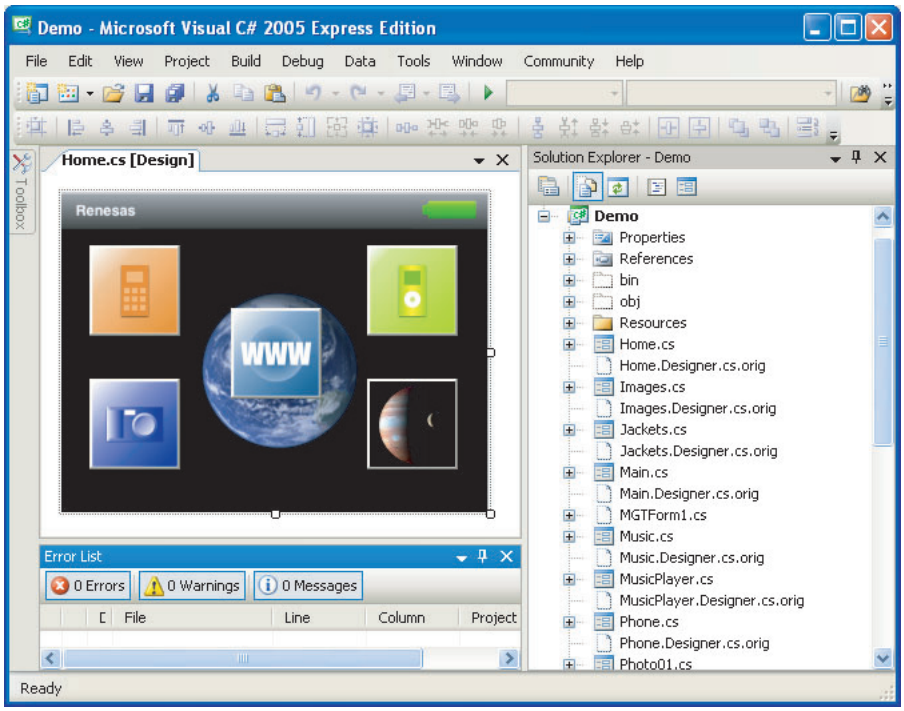


Figure 1 Design Example of GUI Screen

Figure 2 Status of Confirming Designed Form on PC

Figure 3 Confirmation of Software Running on Targeting Board

2. Installation of PC Tools

WYSIWYG on GUI screen and GUI program generation tool (guigen) automatically generate C source programs. To confirm the operation after building those programs, the following tool software on PC must be installed.

- Install "Visual C# 2005 Express Edition" and "Visual C++ 2005 Express Edition".
- After getting Visual Studio 2005 Service Pack 1, and update installed "Visual C++ 2005 Express Edition" and "Visual C# 2005 Express Edition" to the latest version.
- Start registration and installation for custom control for display form by MGTControls.
- Install "MGTForm.zip " for custom form template for display form.
- Refer to chapter 3 of enclosed GUI Tools user's manual.

3. WYSIWYG Design for GUI Screen

Do WYSIWYG design for GUI screen by using Visual C# 2005 Express Edition.

Then, it is impossible to build on C# because the C sources are built by "Visual C++ 2005 Express Edition" after C# codes automatically generated from display design are converted into C sources by GUI program generation tool (guigen).

3.1 Usable GUI Parts

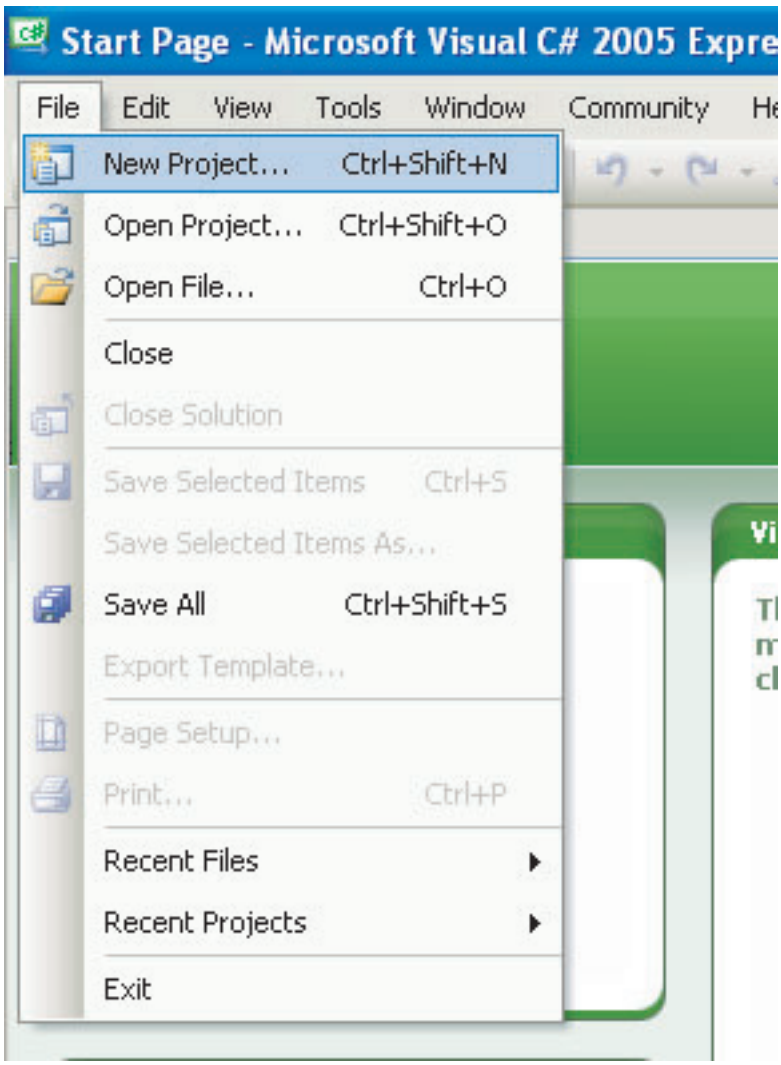
GUI parts used on WYSIWYG of GUI Screen are limited. It is able to use only added parts by custom control for display design, named "MGTControls".

Refer to chapter 4.3 and chapter 5 on enclosed user's manual for details.

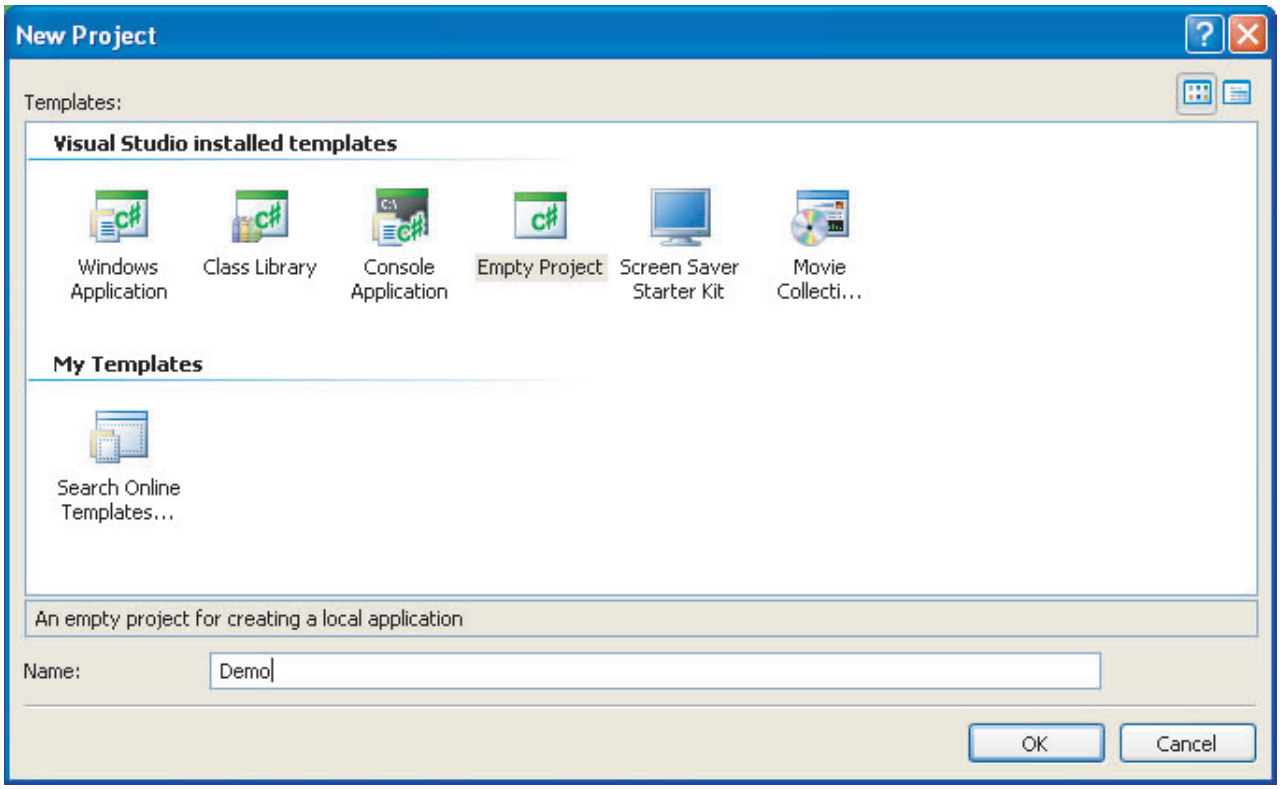
3.2 Generation of New Project

It can be realized to make new project as follows.

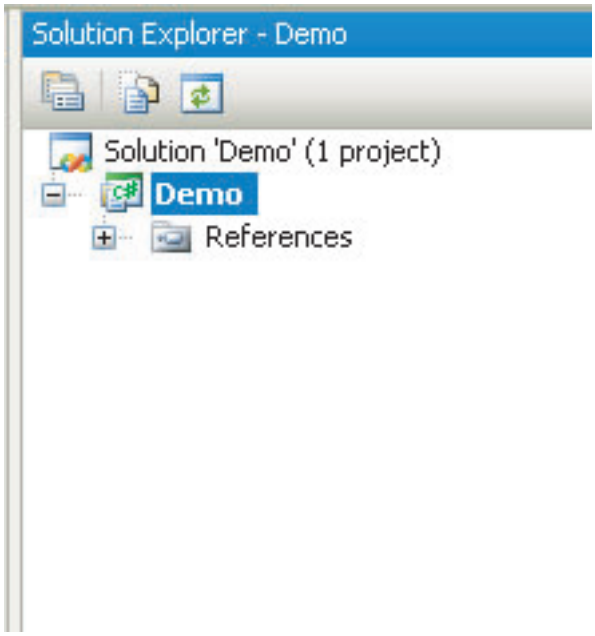
(1) Start "Microsoft Visual C# 2005 Express Edition".
(2) Select "new project" from menu of "File".



(3) Open "New Project" dialog and select "Empty Project". And also input project name in "projectname" column and click "OK".



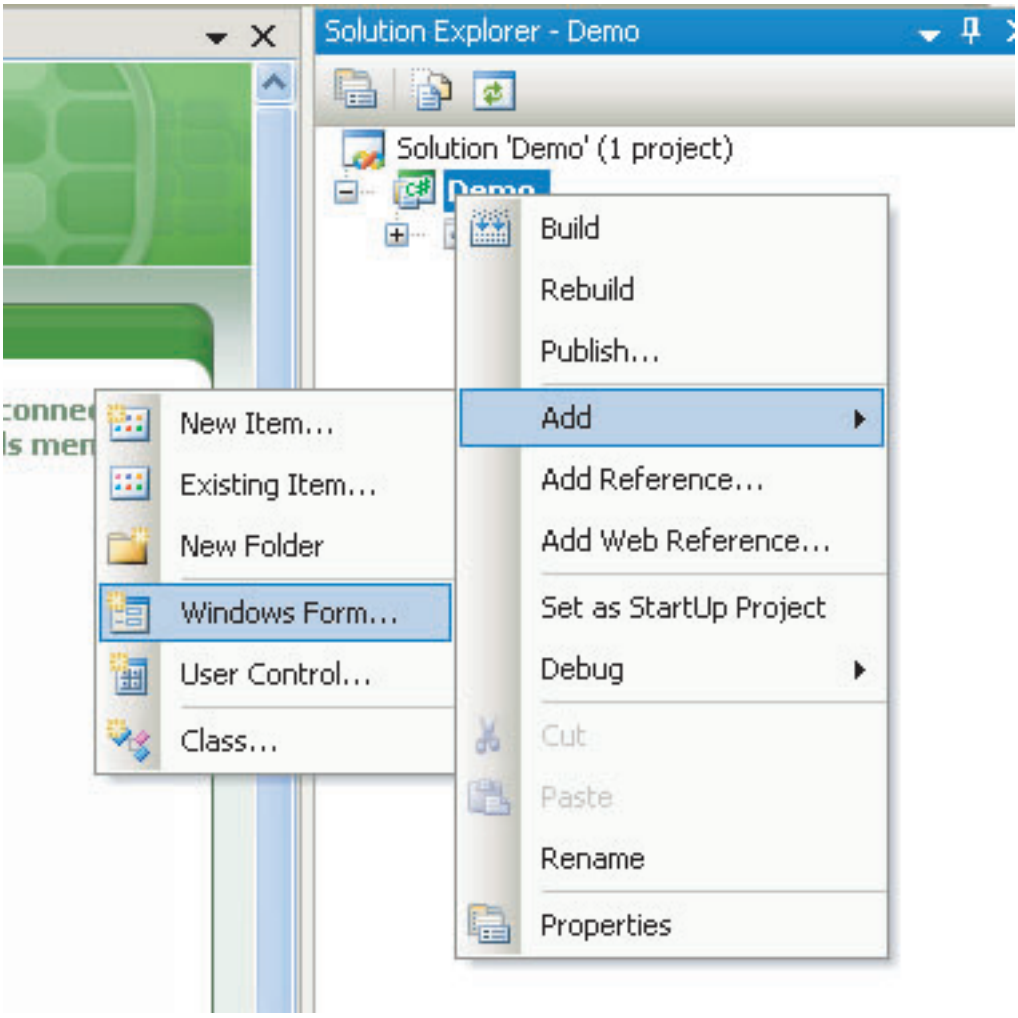
(4) New project is generated and displayed on Solution Explorer Window.



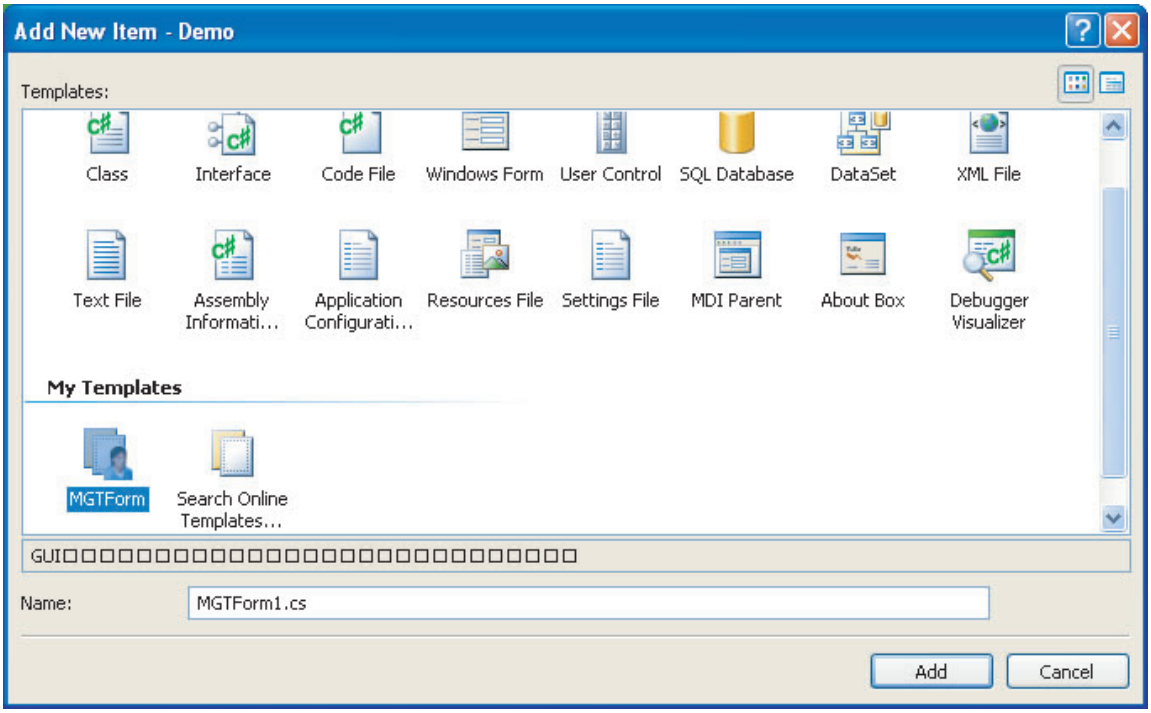
3.3 Addition of the Display Form

By using installed the custom form template named "MGTForm", add the new form in project as following steps.

(1) Select the project name you would like to add in tree display on solution explorer window
(2) Select "Windows Form" from project menu and "Add".

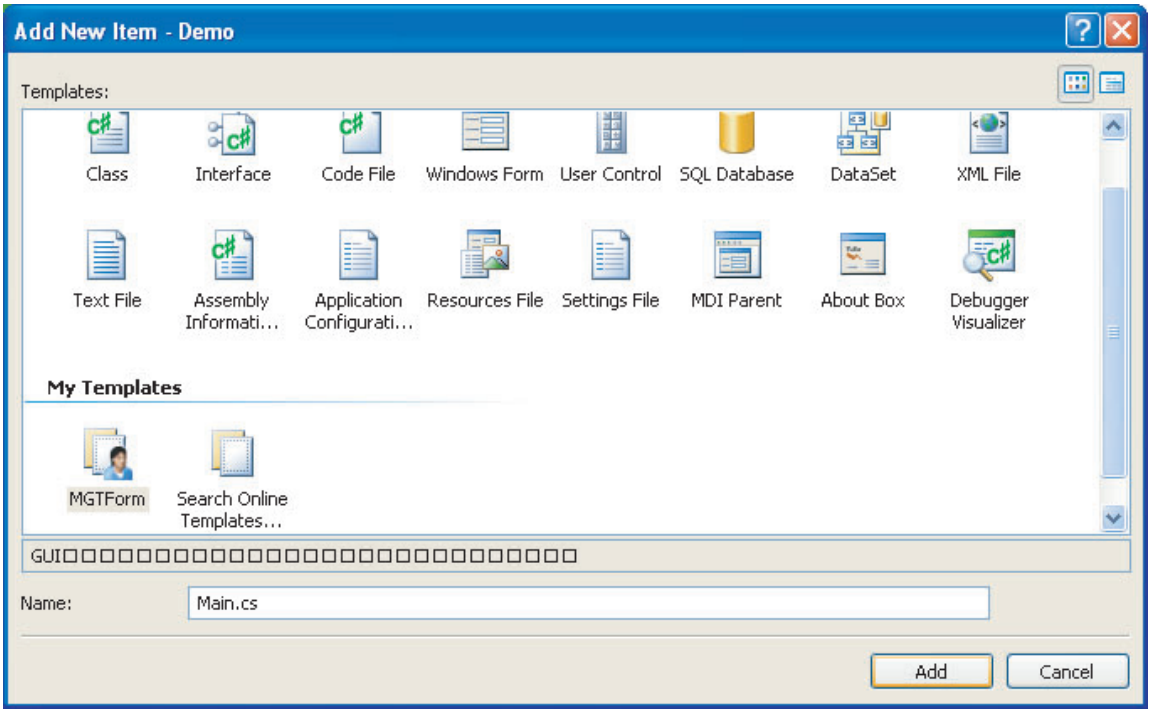


(3) "Add New Item" dialog opens, and click MGTForm icon in my template. Note that it is impossible to use Windows form on Visual Studio as it is. For display design, use the custom form template named "MGTForm" dedicated for GUI Tool, which has been installed in advance.

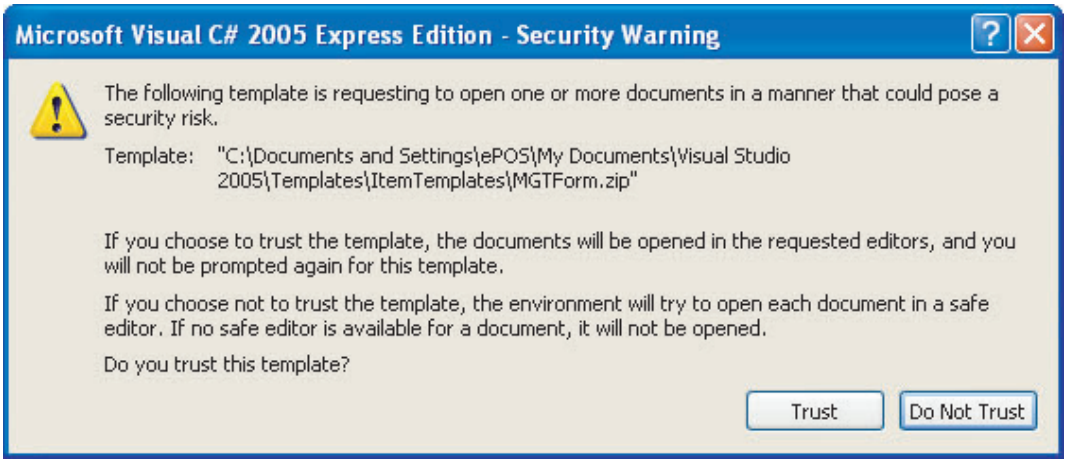


(4) Input a display form in Name Column and click "Add". Note that is will be need to make an initial form having the following name.

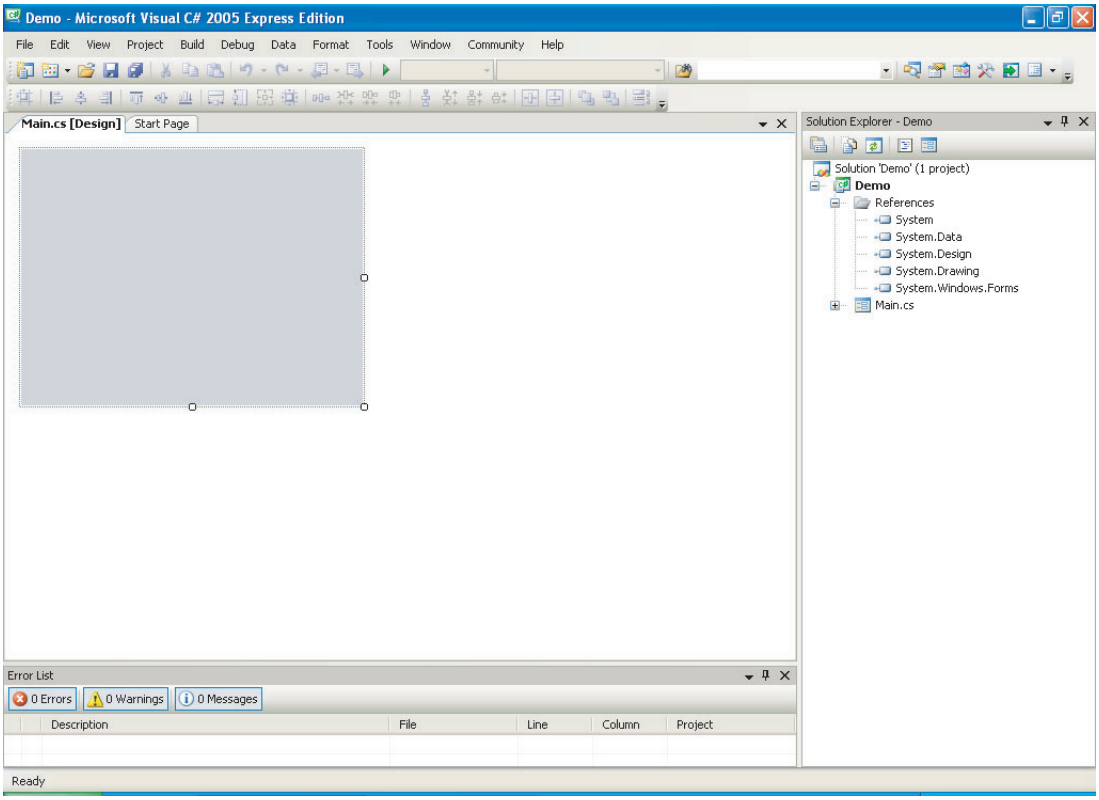
MAIN Main main



After seeing the following Window, Click "Trust".



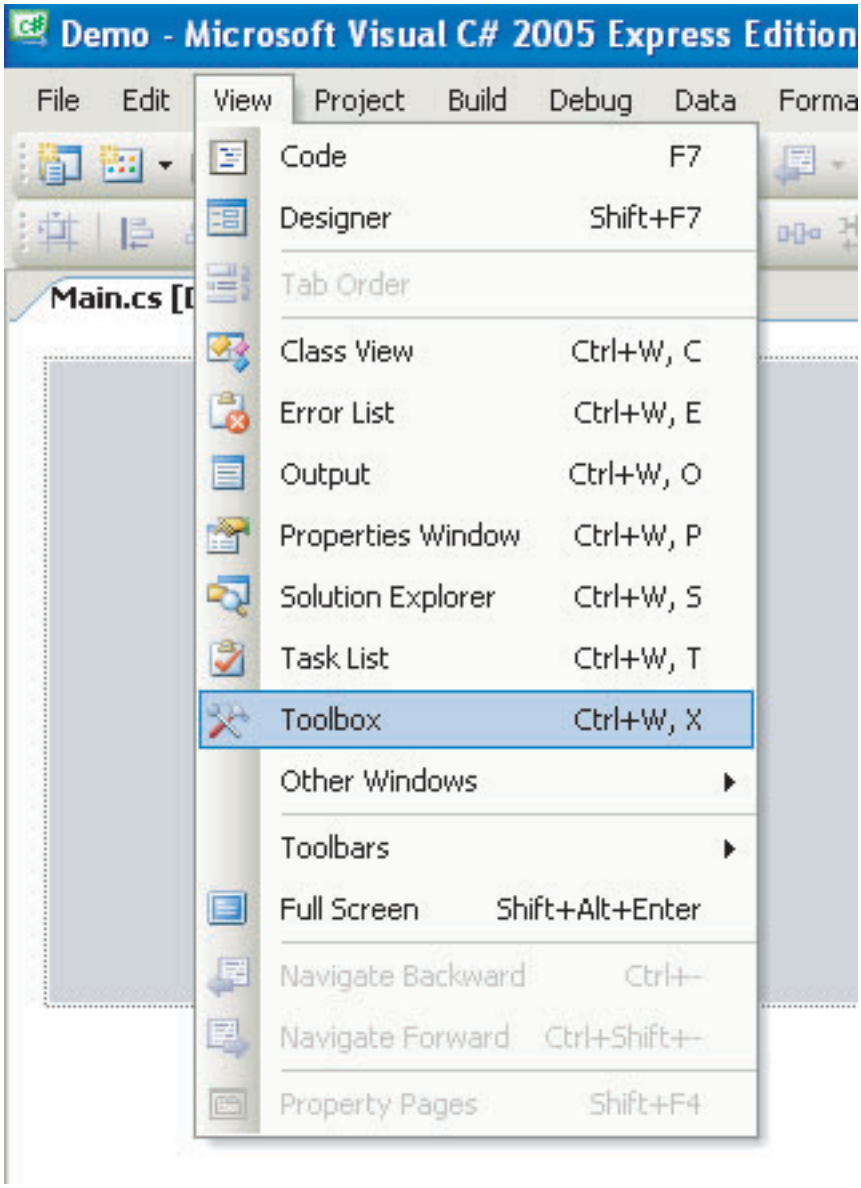
(5) A new display is generated and it is displayed in the project selected by Solution Explorer Windows. At the same time, designer form will be opened.



3.4 Addition of the GUI Parts

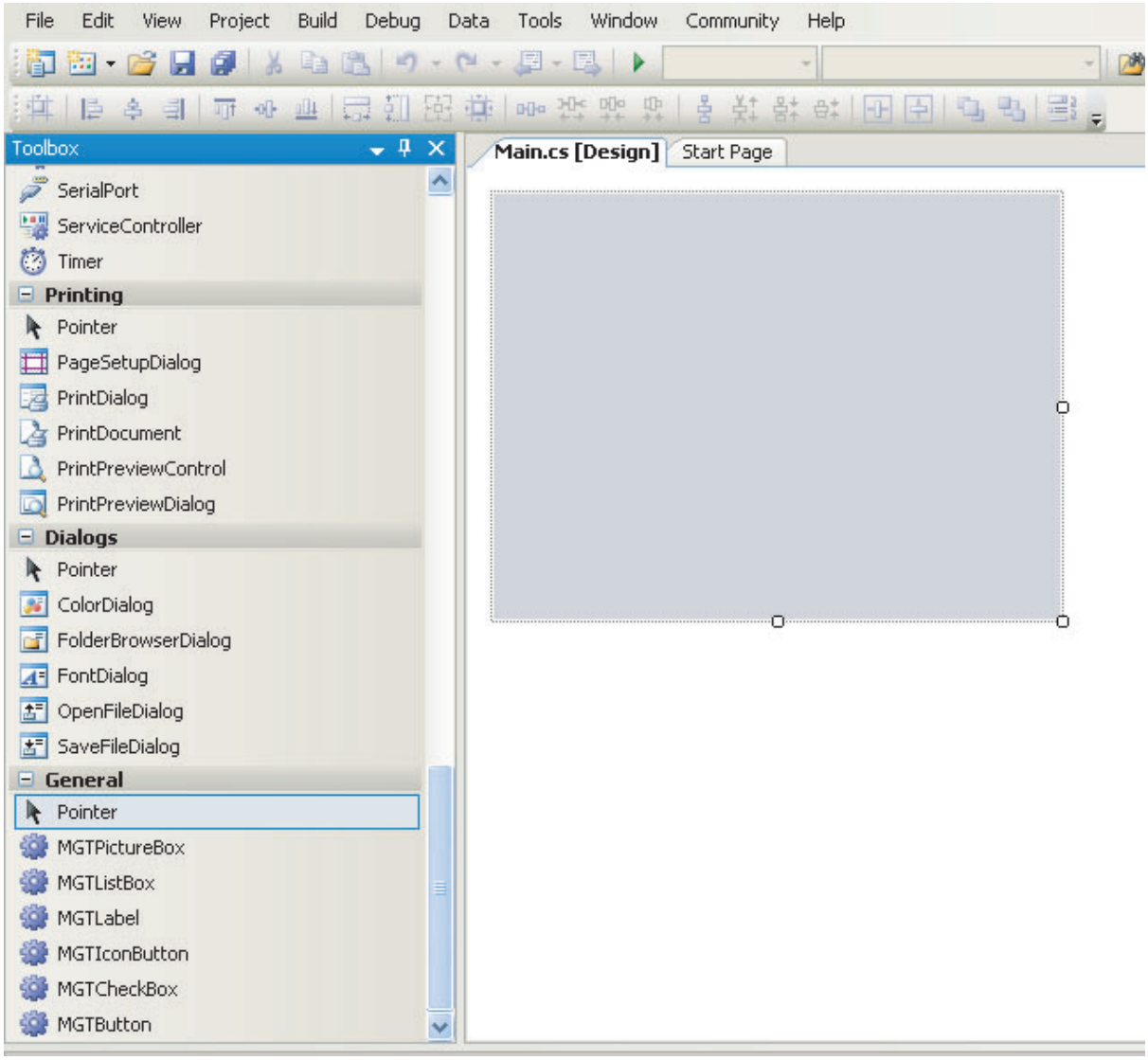
Add GUI parts to the form (screen) according to the following procedure.

(1) Open "Toolbox" screen, select target GUI parts from custom control (MGTControls) that exists under "Whole" tab, and then click them.



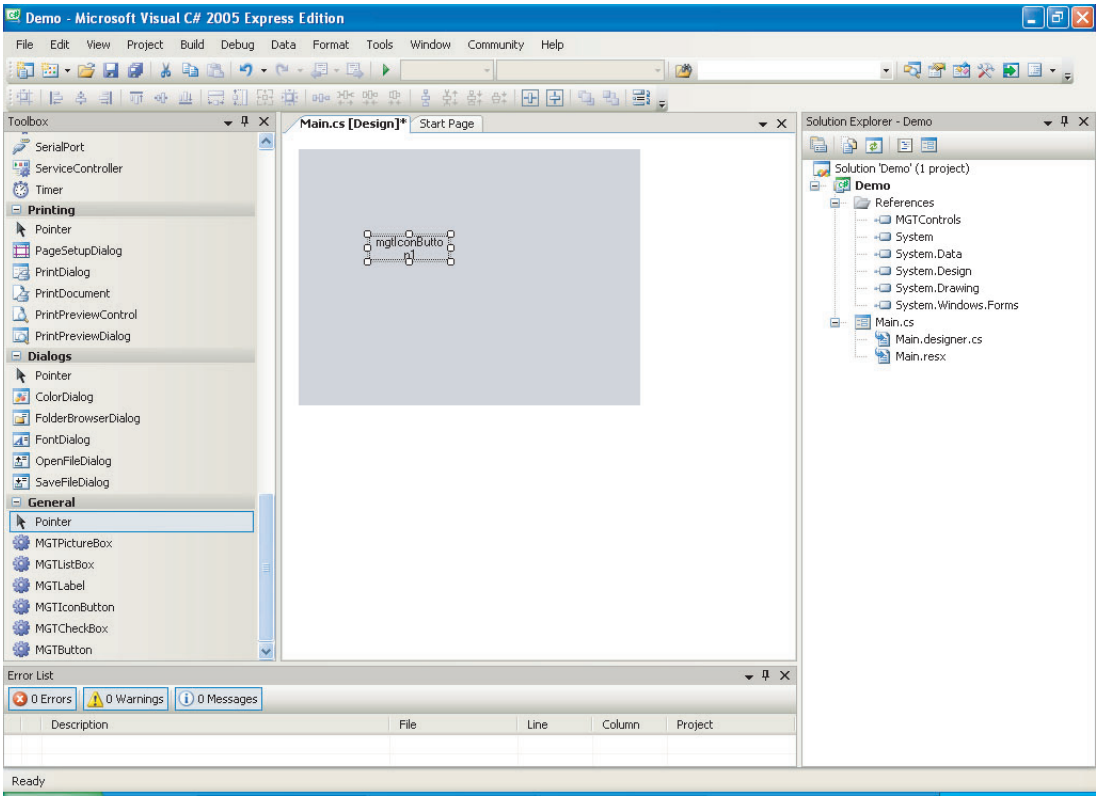
If the installation has been successfully done, the following GUI parts are registered in the toolbox.

- MGTLabel
- MGTButton
- MGTIconButton
- MGTCheckBox
- MGTListBox
- MGTPicutreBox

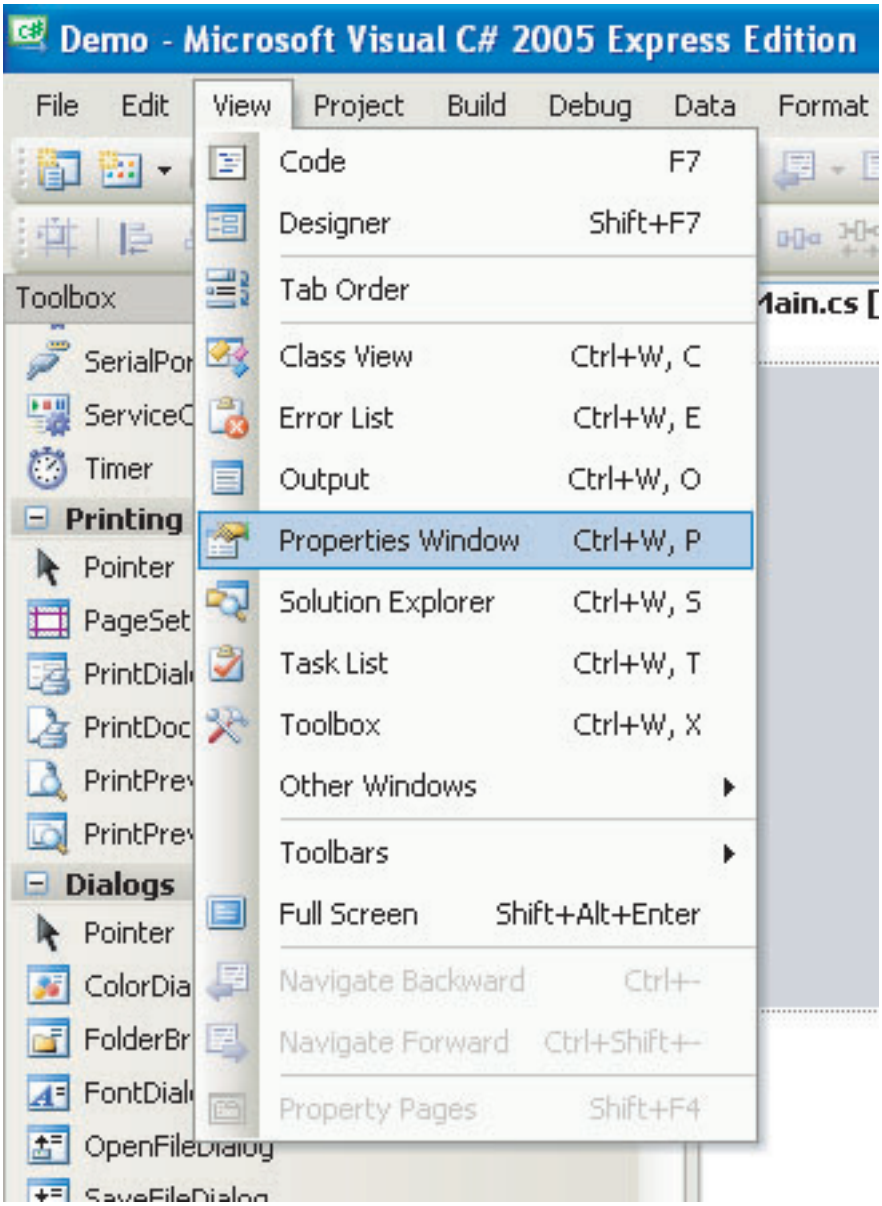


(2) Drag and drop the selected GUI parts on the designer screen
(3) The selected GUI parts are added to the screen.

This is an example of adding IconButton on the screen as follows.



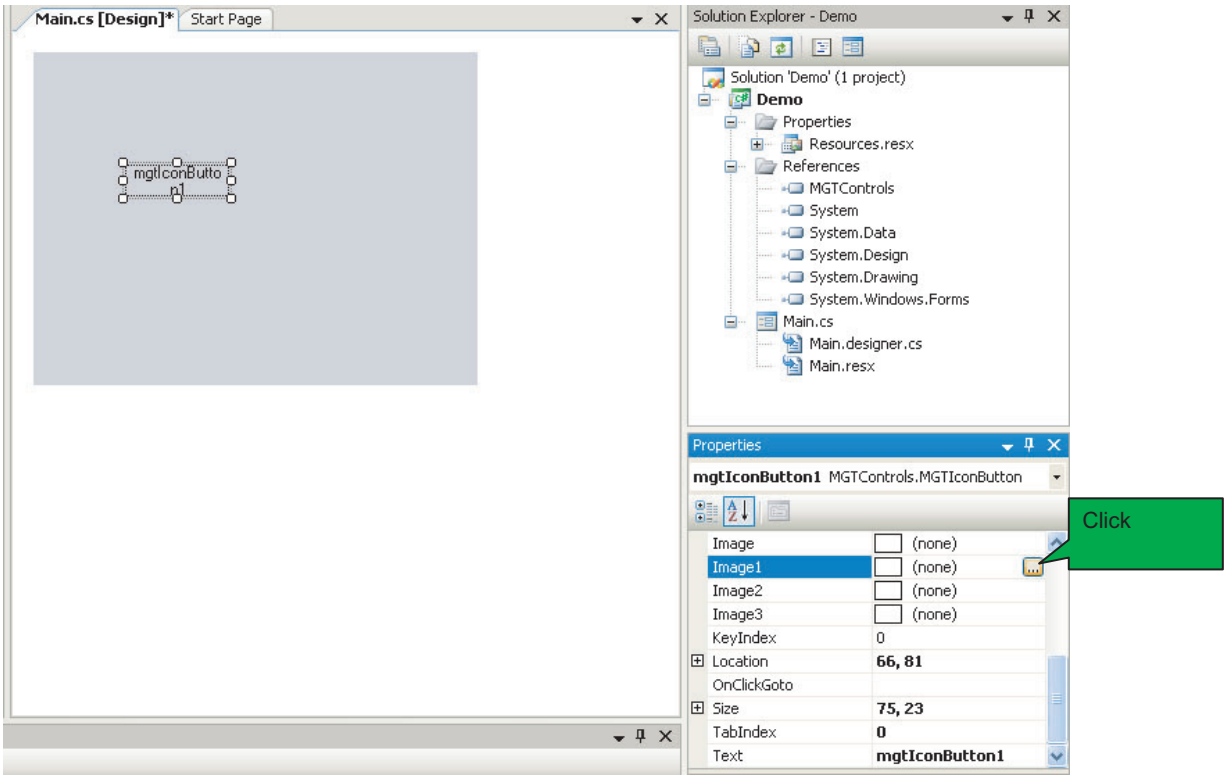
The property can be set to the form (screen) or GUI parts. The confirmation and the change of the property can be done by selecting the form (screen) or GUI parts with the property window opened. Refer to Chapter 6 the properties of GUI parts of the GUI development tool user's manual of the attached paper for details of the property.



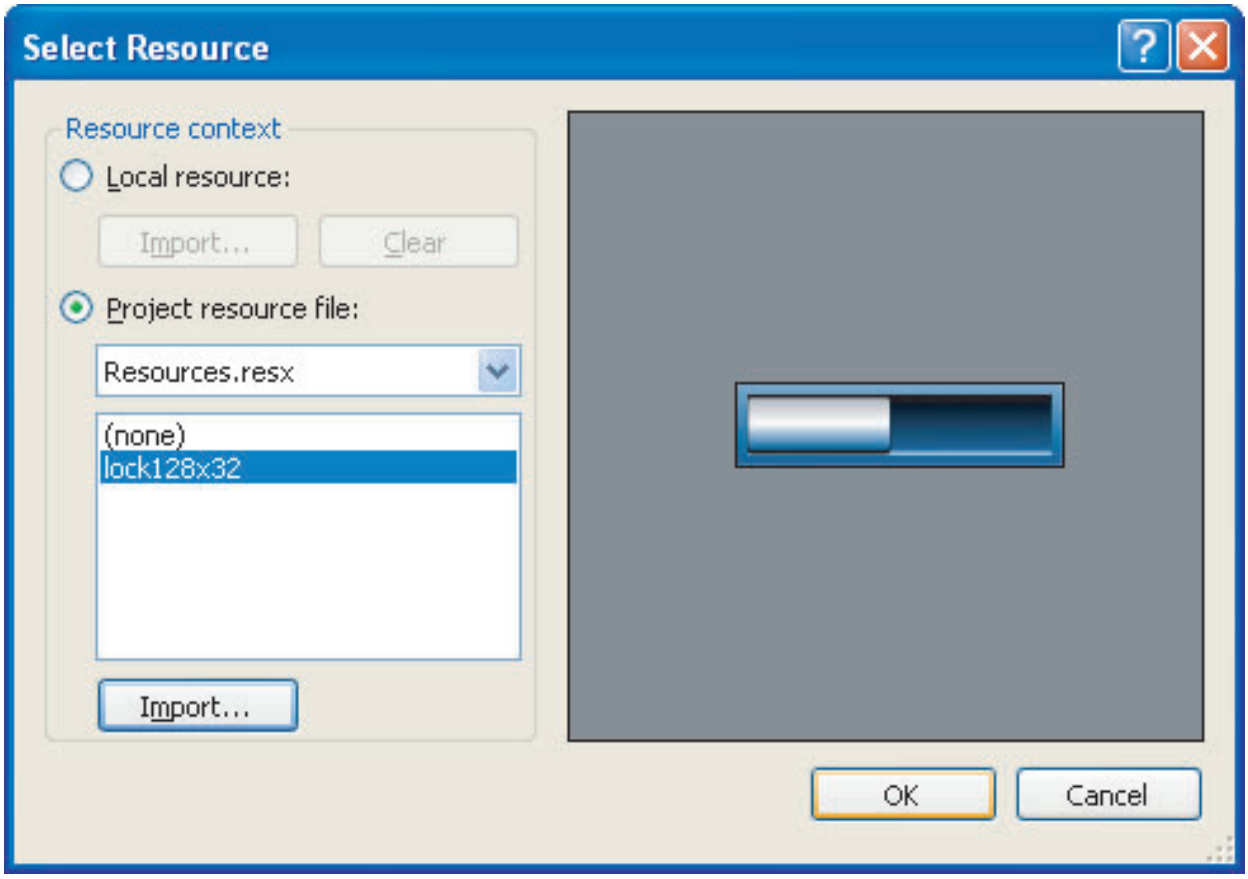
These are examples of the properties of IconButton. The following examples specify some of the properties.

- Specification of the image displayed when the button is pushed and when the button is not pushed (Image, Image1, Image2)
- Specification of size of button (Size)
- Specification of position of button (Location)
- Deletion of text (Text)

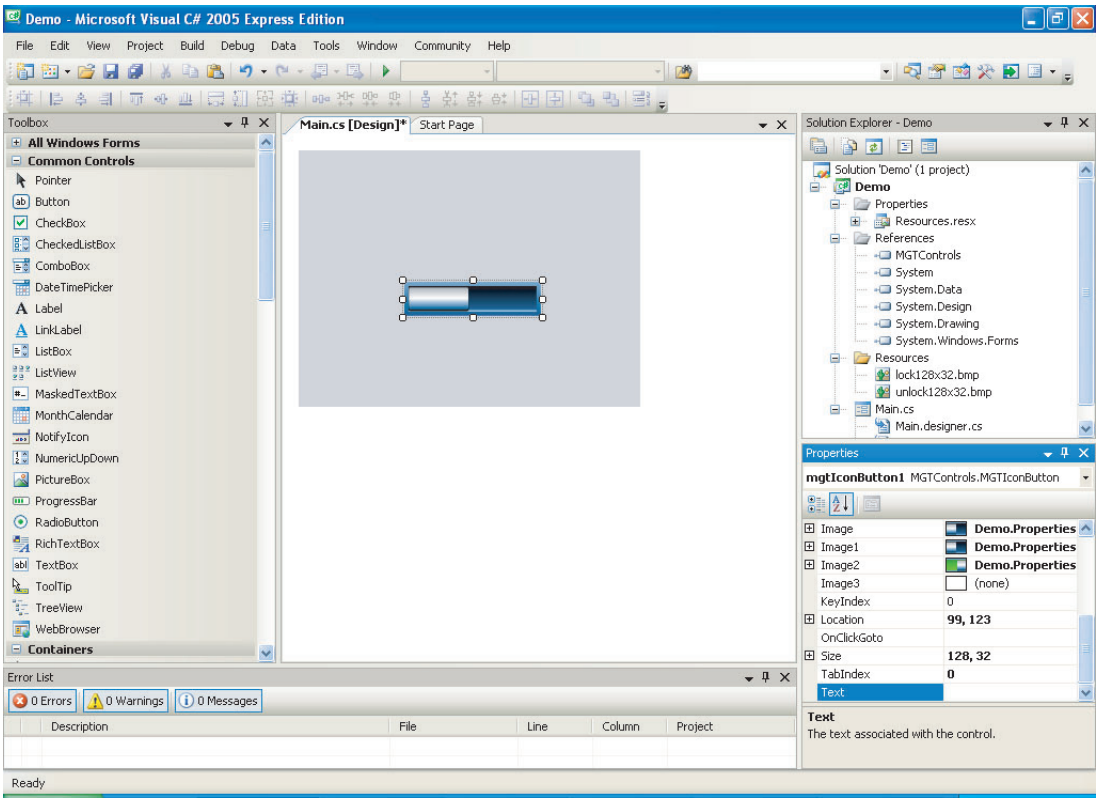
This is an example when the image of Image1 is specified.. The image of Image and Image2 can be specified by a same method.



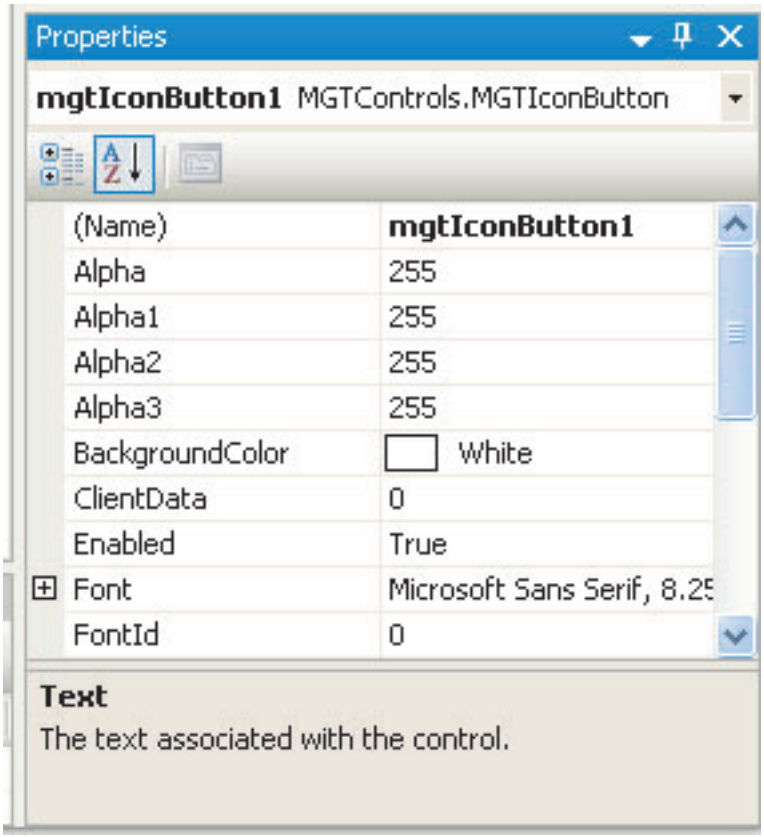
Select the image to be put from the Resources folder and click the "OK" button to decide the image. To select the image from places other than the Resources folder, click the "Import" button, select the image, and then click the "Open" button to decide the image.



This is an example of specifying the property of Image, Image1, Image2, Size, Location, and Text.



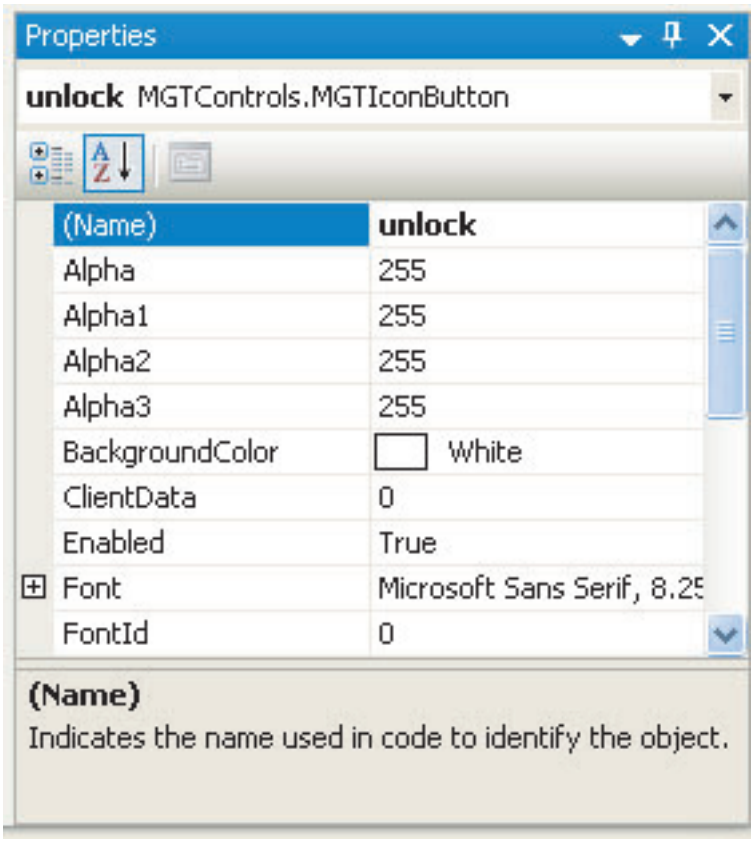
When GUI parts are dropped and dragged on the form (screen), the parts are automatically named. It is possible to change the names in the Name property.



The Name property has been changed to "unlock" in the following examples. This name is defined in "gui.out.h" as follows when the C# design is converted into C source programs with GUI program generation tool (guigen).

```
#define MGT_O_Main__unlock  20
```

Refer to chapter 4.1 of object ID of the GUI development tool user's manual of the attached paper, for the rule of the definition. This definition is used in C source programs because of the judgment of the event (button detection).

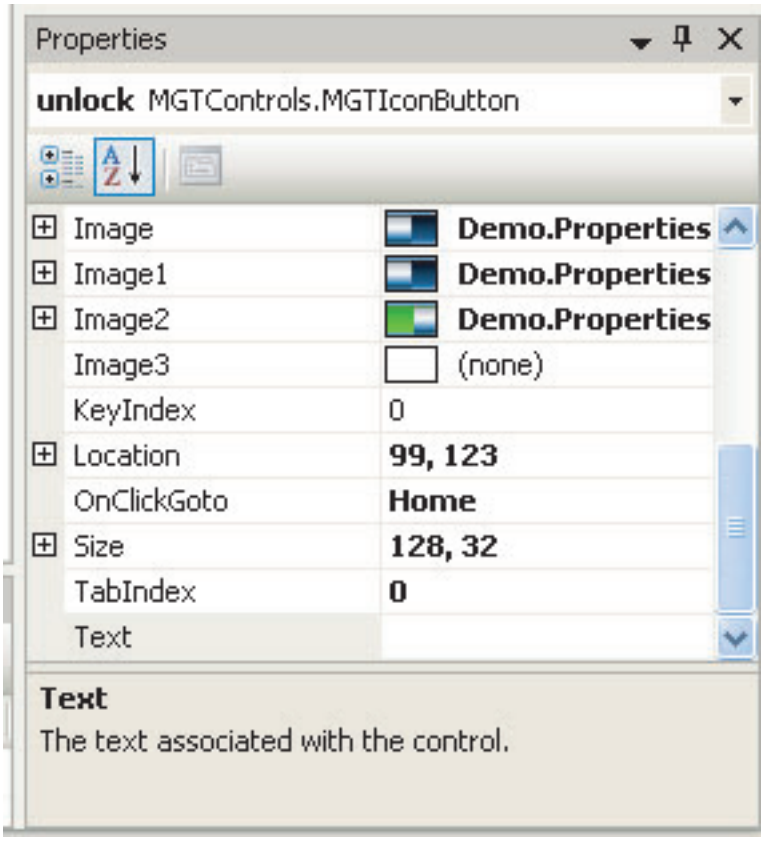


3.4.1 Switch of Form (Screen) by OnClickGoto Event

When a form (screen) name is specified as a jump destination in the OnClickGoto property, the display switches to the specified form (screen) when a mouse button is pressed. The user does not need to write the code for this event processing because the GUI program generation tool (guigen) automatically generates the C source code from the automatically generated C# code. Simply follow the method in section 3.3 to add the jump destination form (screen) as the specified OnClickGoto property.

The example when the Home form is specified for the OnClickGoto property of the unlock button of the

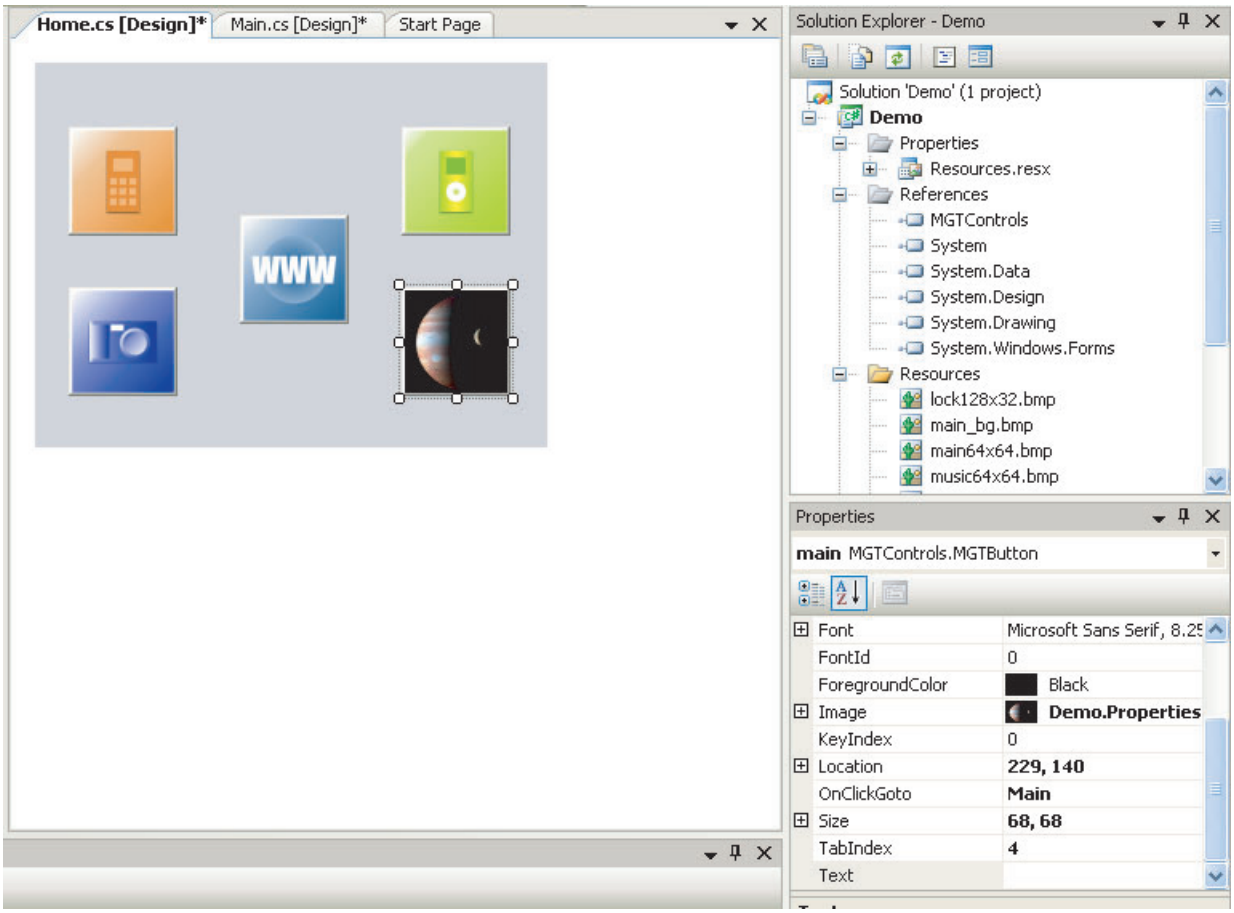
Main form is shown as follows.



5 MGTButtons are arranged on the Home form of a jump destination. MGTButton has the same function as that of IconButton.

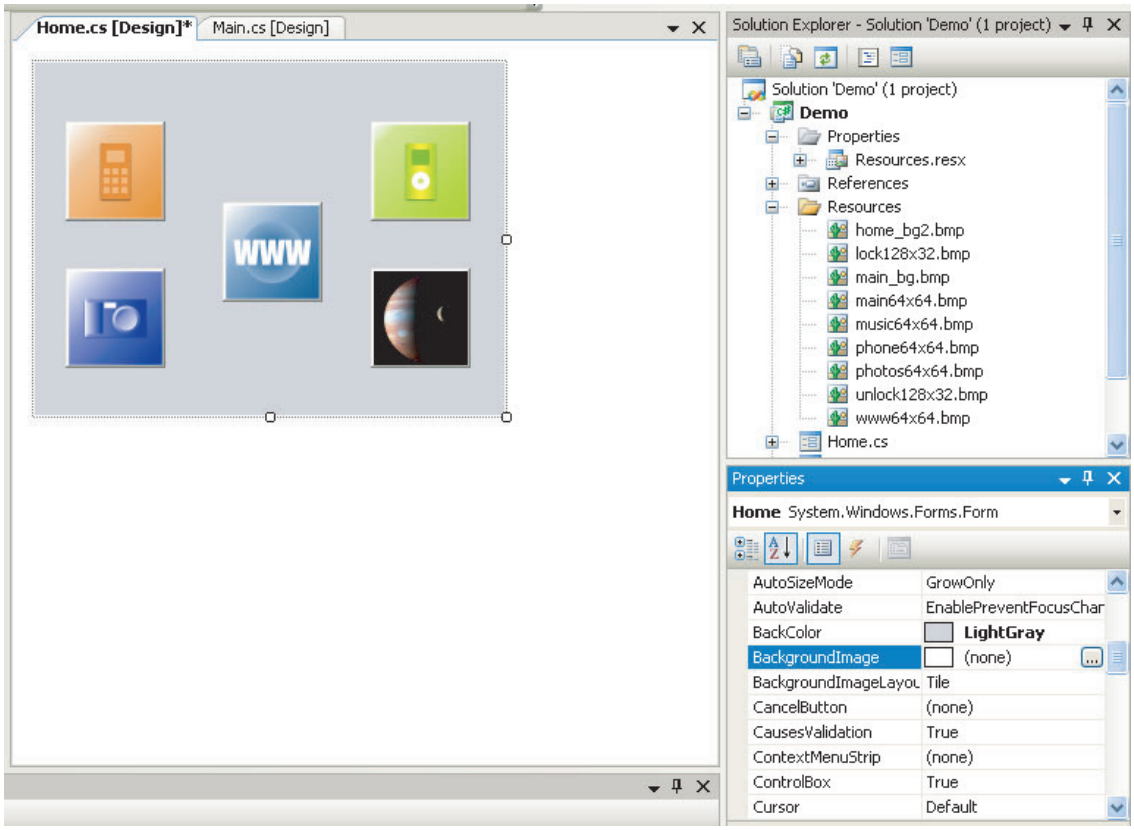
The properties of Name, Image, Location, OnClickGoto, Size, and Text for each button are set in the example of the Home form. Explanation of these operations is omitted due to the similar operation to that of IconButton of the Main form.

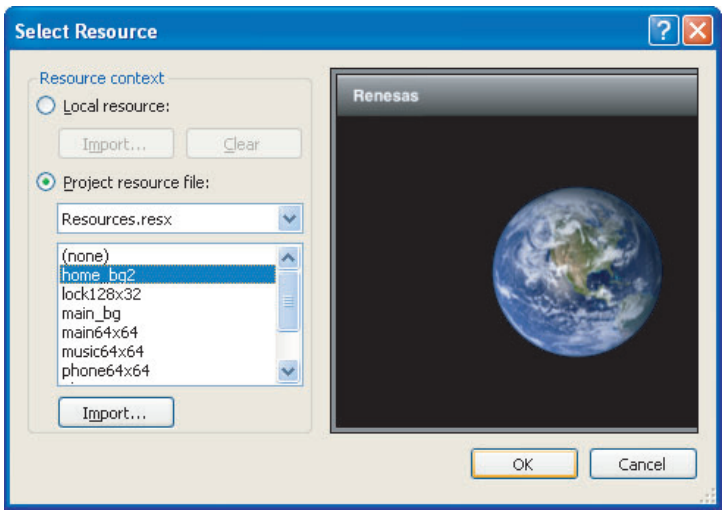
Select Home form (Home.cs) on Solution Explorer, and refer to the properties of each GUI part for details.



3.5 Specification of BackgroundImage for the Form (Screen)

The same way as that of image specifications for buttons can be used to specify the background in the form (screen) BackgroundImage property.

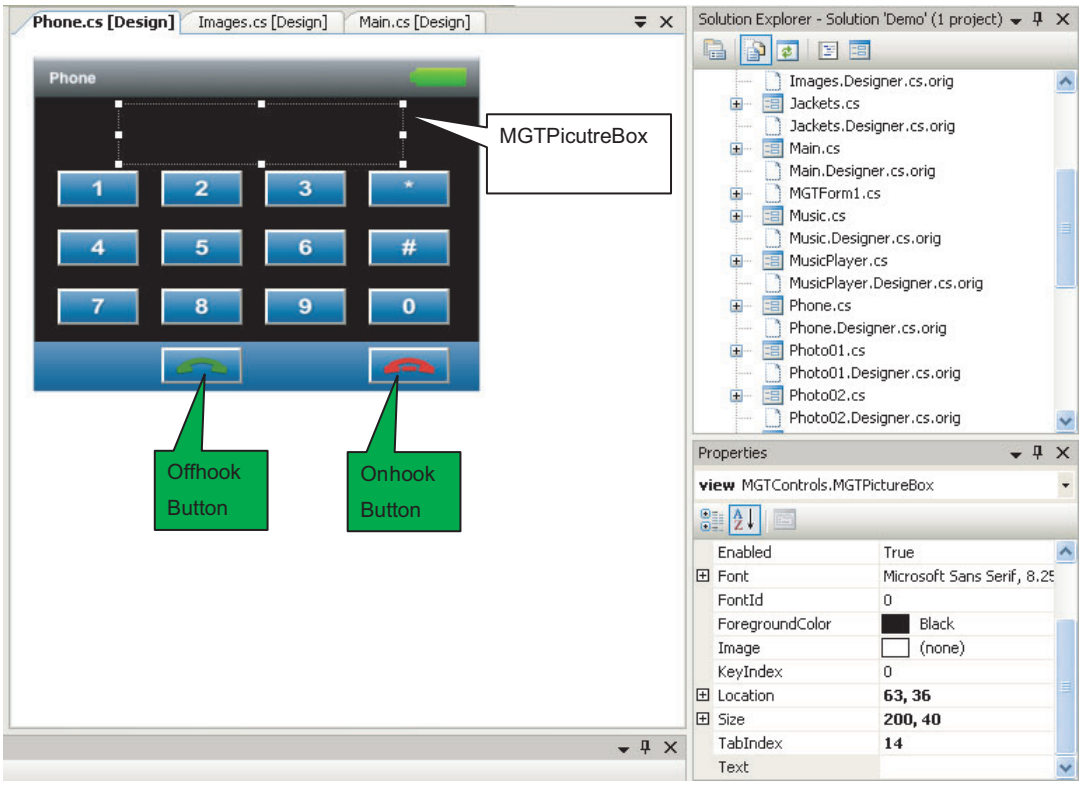




3.6 Mounting Example for PictureBox and Button Event

The example of mounting on the Phone form is shown as follows.

(1) For all the buttons, MGTButton is used. The jump destination in the OnClickGoto property is not registered.
(2) MGTPicutreBox is used to display animation.
(3) The picture of MGTPicutreBox changes at a constant cycle if the Offhook button is pushed when the animation display has stopped. This processing has been achieved by combining cycle processing of the screen (MGT_HOOK_POLLING) and detection event (MGT_HOOK_ONCLICK) of the Offhook button and the screen in the main.c program that the user makes.
(4) The animation display stops when the Onhook button is pushed in the animation display. It changes to the Home form (screen) when the Onhook button is additionally pushed with the animation halted. The C# OnClickGoto property cannot be used to make transition of forms because two functions are allocated in the Onhook button. After judgement of the state within detection event (MGT_HOOK_ONCLICK) of the Offhook button of the main.c program that the user makes, transition to the Home form (screen) is made in the mgt_transit() function.

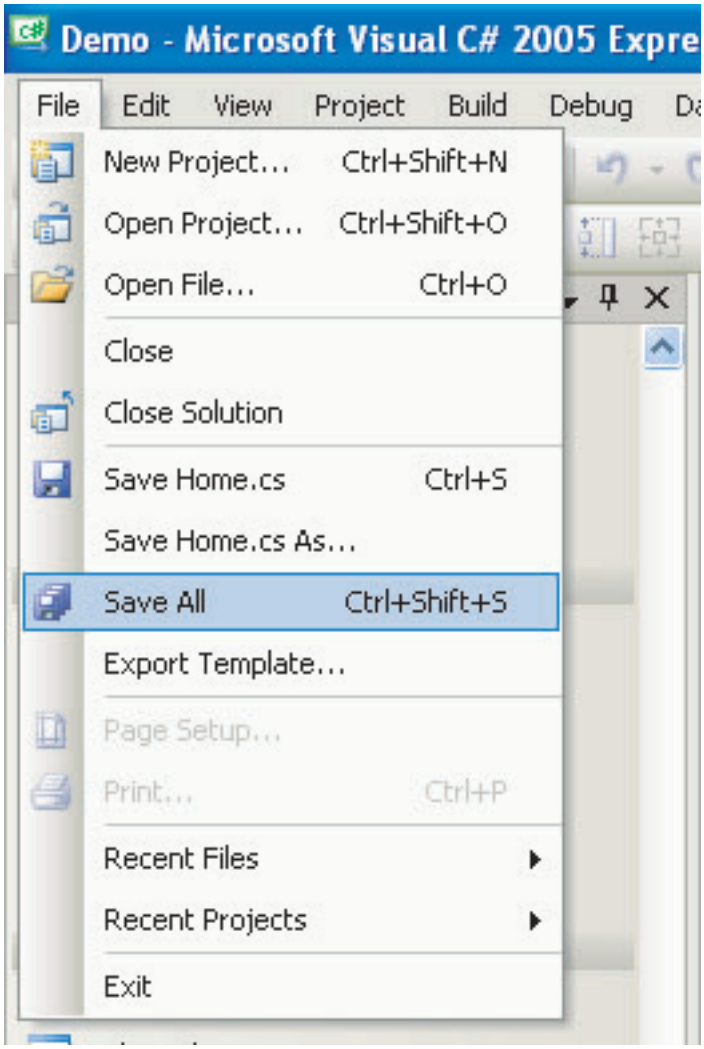


Because Photo to Photo 5 screens and Web01 to Web03 screens are mounted by the same mechanism as that of Home Form, the explanations are omitted. Refer to the sample code for details.

3.7 Preservation of Design

When the layout of the form is completed, save the file by the "SaveAll" command.

Select the "Exit" command when you finish VisualC #.



Building on VisualC# is not reqired. The generated C# source programs and the image data are converted into C source programs and the header files with GUI program generation tool (guigen.exe).

Building is done on the command line of VisualC++.

3.8 Correction of Design

To correct the design again after end of VisualC#, click ***.sln (Microsoft Visual Studio Solution) under the folder to which you saved the data. And the made design file is started.

3.9 More Information on How to Use VisualC#s

For more information on how to use VisualC#s, refer to Help menu on Visual C#, the Microsoft homepage, store-bought books, etc.

4. Generation of EXE File Executable under Windows

To generate EXE file executable under Windows, the user should prepare the following two contents.

(1) WYSIWYG of the GUI screen is designed by Visual C # and the C# source programs and the image data are automatically generated.
(2) The event processing is added to userevent.c for C source program*.
When this two are copied onto the project folder of the sample and MAKE.BAT is executed from the command line of VisualC++, the EXE file that operates by Windows is generated.

Note: * Although the separate user's manual for the GUI development tool covers adding user-specified events to main.c, simply copying main.c for Windows to the High-performance Embedded Workshop environment for the H8S/2456 leads to various errors. We have thus prepared samples for both Windows and the H8S/2456 of main.c files that have equivalent functions. Add event code created by the user to the userevent.c source code without writing anything in main.c.

Table 1 Sample Project Folder File List

Directory	File Name	Classification	Explanation
bin	guigen.exe		GUI Program generation tool
bin	w9xpopen.exe		
bin	MSVCR71.dll		
include	lcd.h	Header file	No need to modify
include	mgt.h	Header file	No need to modify
lib	LCD.lib	Library	It specifies at the build process.
lib	MGT.lib	Library	It specifies at the build process.
lib	SDL.lib	Library	It specifies at the build process.
lib	SDLmain.lib	Library	It specifies at the build process.
lib	WGP16.lib	Library	It specifies at the build process.
lib	LCD.dll	DLL	It needs when generated EXE is executed
lib	SDL.dll	DLL	It needs when generated EXE is executed
lib	WGP16.dll	DLL	It needs when generated EXE is executed
SampleProject	userevent.c	C source file	The user must add the event processing if necessary.
SampleProject	MAKE.BAT	Batch file	The EXE file that operates by Windows is generated when executing it from the VisualC++ command line.
SampleProject	Makefile	Makefile	NMAKE file of VisualC++
SampleProject	main.c	C source file	Main program. No need to modify.
SampleProject	gui.out.h	Header file	Automatic generation by the GUI program generation tool
SampleProject	gui.out.c	C source file	Automatic generation by the GUI program generation tool
SampleProject	gui.out.img????	Image data (Put numbers into ????.)	Automatic generation by the GUI program generation tool
SampleProject \GUIbuilder_demo_ C#_design		C# Project	The content of the directory of the form design generated with VisualC# is copied here.

4.1 Correction of Makefile

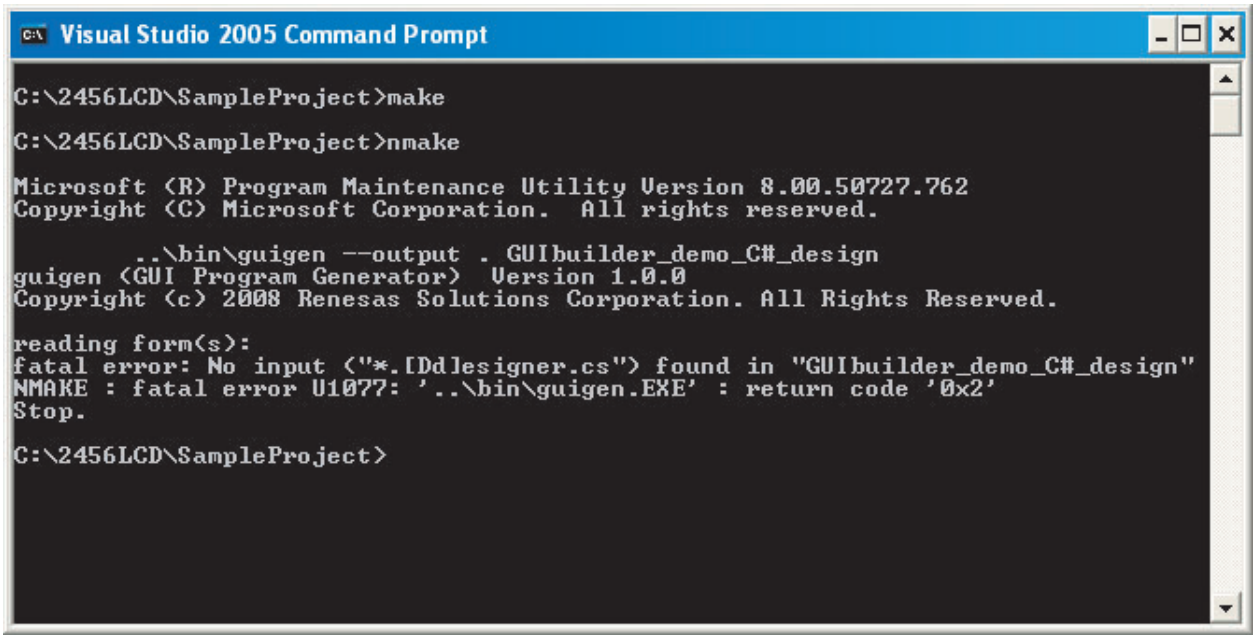
An example of Makefile when C# source file (***.cs) exists in SampleProject\GUIbuilder_demo_C#_design is shown below. When the path is different,

```
$(GUIGEN) --output . GUIbuilder_demo_C#_design
```

It is necessary to correct the above path.

```
# -*- mode: makefile; coding: sjis-dos -*-
# Sample NMAKE Makefile

LD = link
RM = -DEL
CP = -COPY
GUIGEN   = ..\bin\guigen

CFLAGS = /c /O2 /FD /EHsc /MD /nologo \
   /D "__WIN32__" /D "NDEBUG" /D "_WINDOWS" /D "_UNICODE" /D "UNICODE" \
   /I ..\include

#LDFLAGS = /SUBSYSTEM:WINDOWS /MACHINE:X86 /NOLOGO
LDFLAGS = /SUBSYSTEM:console /MACHINE:X86 /NOLOGO

LIBDIR   = ..\lib
LIBS  = MGT.lib WGP16.lib LCD.lib SDL.lib SDLmain.lib


TARGET   = GUIbuilder_demo.exe

all:  $(TARGET)

$(TARGET):  main.obj userevent.obj gui.out.obj WGP16.dll LCD.dll SDL.dll
  $(LD) $(LDFLAGS) /OUT:$(TARGET) main.obj userevent.obj gui.out.obj \
  /LIBPATH:$(LIBDIR) $(LIBS)

WGP16.dll LCD.dll SDL.dll:
  $(CP) ..\lib\*.dll .

main.obj:main.c
userevent.obj: userevent.c
gui.out.obj:gui.out.c

gui.out.c:  guigen


guigen:
  $(GUIGEN) --output . GUIbuilder_demo_C#_design

clean:
  $(RM) *.obj *.manifest vc80.idb

distclean:  clean
  $(RM) $(TARGET) *.dll gui.out.*
```

The following message is displayed at the build process when the C# source cannot be found due to the wrong .



4.2 Correction of userevent.c

Userevent.c is a file to add the user's event processing. There is only mgt_hook() function in default in this file. It is necessary to add the event processing such as buttons added by C# design. However, because the event processing of form (screen) switching in the OnClickGoto property is generated in gui.out.c automatically, the addition of processing to the mgt_hook() functions is unnecessary.

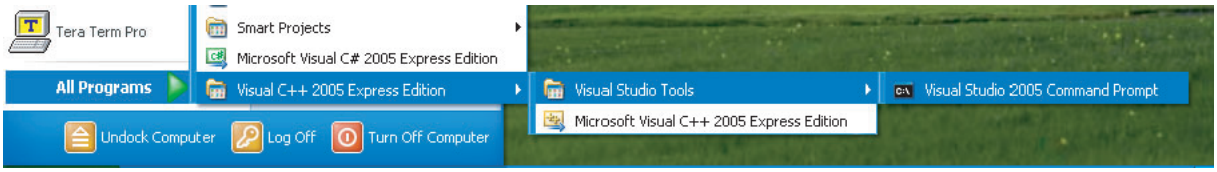
The example of describing the mgt_hook() function is shown as follows.

```
void mgt_hook(int type, int id)
{
  switch (type) {
  case MGT_HOOK_ENTER:
       The processing at the beginning of the form (screen) is described.
    break;

  case MGT_HOOK_LEAVE:
       The processing at the end of the form (screen) is described.
    break;

  case MGT_HOOK_POLLING:
       The cycle processing is described.
    break;

  case MGT_HOOK_ONFOCUS:
       The event when focus is acquired is described.
    break;

  case MGT_HOOK_ONCLICK:
       The click event of the button is described.
    break;
  }
}
```

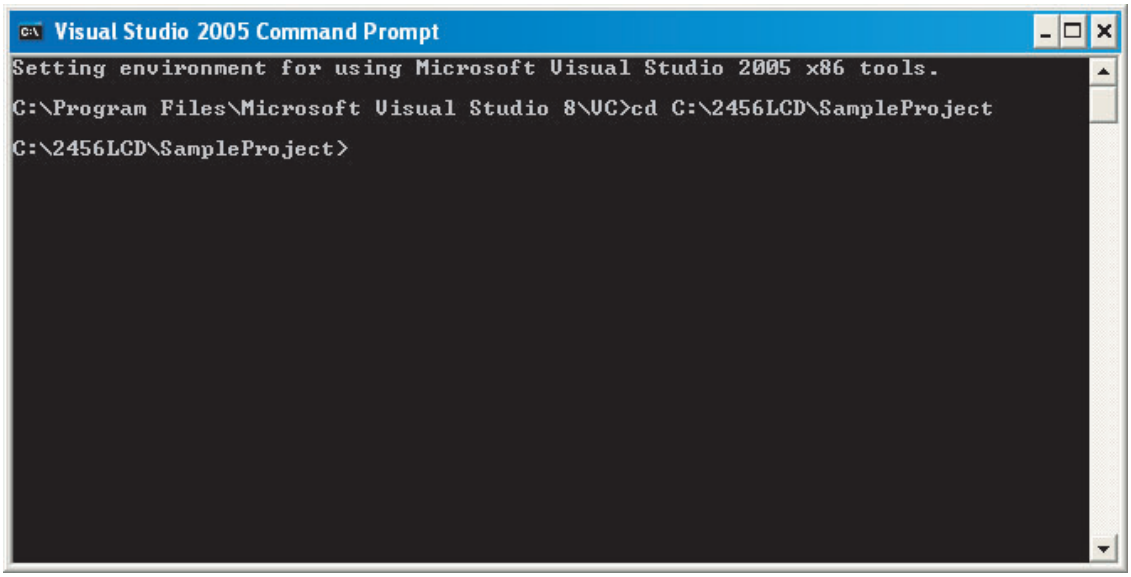
Also, add the user-created function to be called from the event branch of the mgt_hook() function. For a more concrete example for reference, see the code in SampleProject\userevent.c.

4.3 Building under Windows

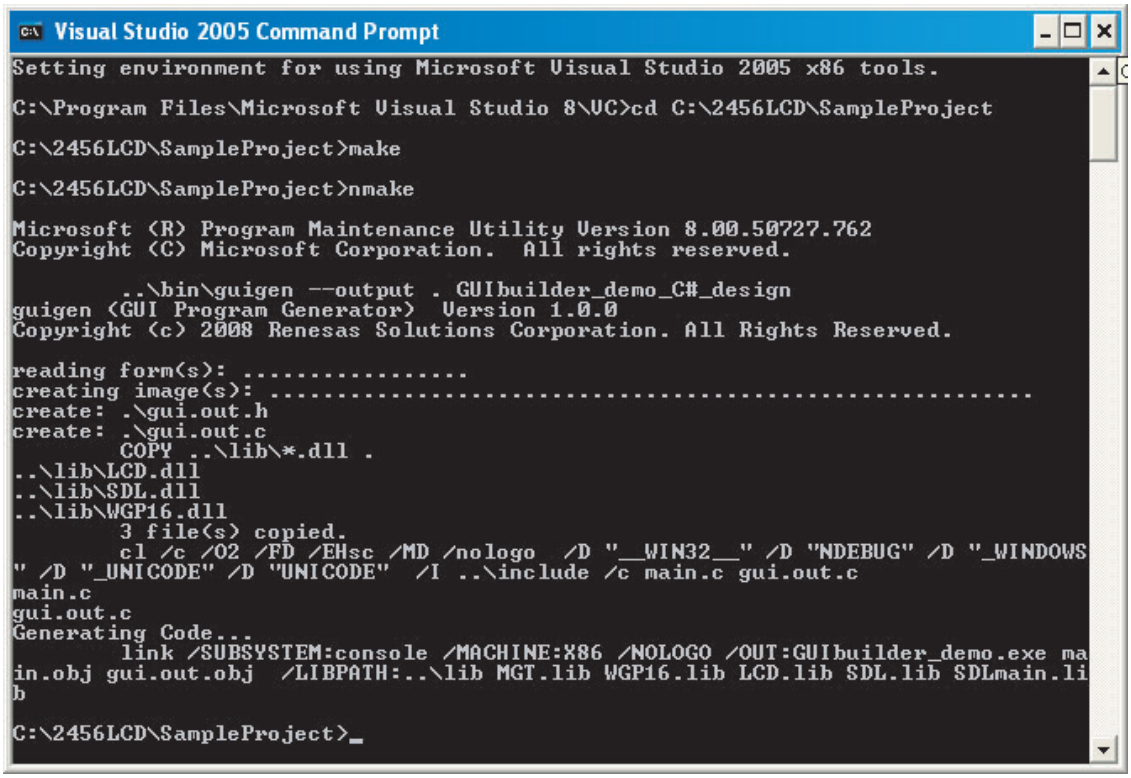
The command prompt of VisualC++ is started.



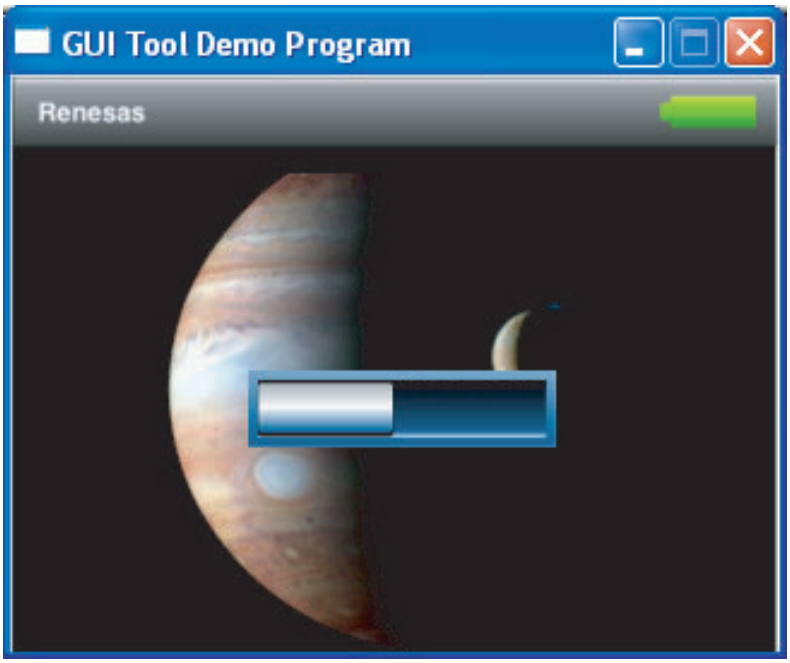
Specify the SampleProject folder in Table 1 of Chapter 4 as the directory.



Write "make" on the command line and start NMAKE.



GUIbuilder_demo.exe is generated to the SampleProject folder. The demonstration operates by click of GUIbuilder_demo.exe because LCD.dll, SDL.dll, and WGP16.dll are copied in the same folder.



Refer to Chapter 9.5 of the GUI development tool user's manual for correction, when the GUI program generation tool outputs the error message and it ends.

5. Operation in H8S/2456 Environment

The system configuration of H8S/2456 is shown in Figure 4 and the list of the components is shown in Table 2.

For development, the content of Table 3 is separately needed.

Figure 4 System Configuration of H8S/2456

Table 2 System Configuration List of H8S/2456

Part Number	Content	Note
R0K402456D000BR (Renesas)	H8S/2456 CPU Board	For a fee
R0K402472D010BR (Renesas)	• LCD I/F Board • Back Light Connection Cable • Flat Cable made by Sumitomo (SML2CD-33×200-AD×10-P0.5-S4-FUL2896) • Evaluation Board Connection Cable	For a fee
TCG057QV1AD-G10 (Kyocera)	LCD Panel	For a fee
CXA-L0605A-VJL (TDK Lambda)	Back Light board	For a fee
SPU30-102 (SINPRO)	5V- AC Adapter	For a fee

Table 3 Development Tool List

Name	Content	Note
High-performance Embedded Workshop (Renesas)	Including Complier for H8S	For a fee
E10A Emulator (Renesas)	—	For a fee
Execution library for H8S	GUI Execution library	For a fee
H8S/2456 High-performance Embedded Workshop reference project	GUI Execution Environment	Free (Excluding Execution library for H8S)
File for Programming into external Flash Memory	File for Programming into External Flash Memory mounted on R0K402456D000BR via E10A	Free

5.1 High-performance Embedded Workshop Reference Project for H8S/2456

Table 4 shows the file list of High-performance Embedded Workshop reference project (2456lcd_demo) for H8S/2456. The GUI execution environment for the LCD panel with the touch panel made by Kyocera is included. Therefore, only by modifying a part of files generated in Chapter 4.3, copying them to this project, and building them, the GUI application that operates on H8S/2456 can be made. Refer to Chapter 5.2 for details.

The folder of the reference project has files which have been changed for the configuration in Figure 4 and Table 2. It is possible to write them as they are to H8SS/2456 and confirm the operation.

Purchase is separately needed to file (*1) in Table 4. Confirm it to Renesas Technology or the special agent. Copy the relevant file for the purchased product onto the directory in Table 4 before building with High-performance Embedded Workshop.

Table 4 File List of High-performance Embedded Workshop Reference Work Space (2456lcd_demo) for H8S/2456

Directory Name	File Name	Content	Note
2456lcd_demo	2456lcd_demo.Hbp	High-performance Embedded Workshop administrative file	—
2456lcd_demo	2456lcd_demo.hws	High-performance Embedded Workshop administrative file	High-performance Embedded Workshop starts after click
2456lcd_demo	2456lcd_demo.tws	High-performance Embedded Workshop administrative file	—
2456lcd_demo\2456lcd_demo	2456lcd_demo.hwp	High-performance Embedded Workshop administrative file	—
2456lcd_demo\2456lcd_demo	2456lcd_demo.nav	High-performance Embedded Workshop administrative file	—
2456lcd_demo\2456lcd_demo	2456lcd_demo.pgs	High-performance Embedded Workshop administrative file	—
2456lcd_demo\2456lcd_demo	2456lcd_demo.tps	High-performance Embedded Workshop administrative file	—

Directory Name	File Name	Content	Note
2456lcd_demo\2456lcd_demo	commondef.h	High-performance Embedded Workshop administrative file	—
2456lcd_demo\2456lcd_demo	dbsct.c	C Source file generated by High-performance Embedded Workshop	—
2456lcd_demo\2456lcd_demo	DefaultSession.hsf	High-performance Embedded Workshop administrative file	—
2456lcd_demo\2456lcd_demo	iodefine_h8s2456.h	H8S/2456 Register Definition	—
2456lcd_demo\2456lcd_demo	resetprg.c	It executes after RESET	Modefy if necessary
2456lcd_demo\2456lcd_demo	sbrk.c	C Source file generated by High-performance Embedded Workshop	—
2456lcd_demo\2456lcd_demo	sbrk.h	C Source file generated by High-performance Embedded Workshop	—
2456lcd_demo\2456lcd_demo	stacksct.h	Definition for stack size	Modefy if necessary
2456lcd_demo\2456lcd_demo	typedefine.h	C Source file generated by High-performance Embedded Workshop	—
2456lcd_demo\2456lcd_demo\ Debug	2456lcd_demo.abs	File with debugging information	It specifies in case of writing by High-performance Embedded Workshop.
2456lcd_demo\2456lcd_demo\ Debug	2456lcd_demo.mot	Motorola format file	—
2456lcd_demo\2456lcd_demo\ gui_usrapp	gui.out.h	Header file generated with guigen.exe	The file generated with guigen.exe is copied as it is.
2456lcd_demo\2456lcd_demo\ gui_usrapp	gui.out.img****	Picture file generated with guigen.exe	The file generated with guigen.exe is copied as it is.
2456lcd_demo\2456lcd_demo\ gui_usrapp	guiout.c	C source file generated with guigen.exe	Modification is necessary for this file.
2456lcd_demo\2456lcd_demo\ gui_usrapp	main.c	The main processing	It is not possible to share with main.c on the Windows side. Use main.c included in this folder.
2456lcd_demo\2456lcd_demo\ gui_usrapp	userevent.c	The event processing	The user must add the event processing if necessary.
2456lcd_demo\2456lcd_demo\ H8S_LIB	libmgtH8S.lib	Execution library for H8S	(*1) Purchase is separately needed.
2456lcd_demo\2456lcd_demo\ H8S_LIB	libwgp16r.lib	Execution library for H8S	(*1) Purchase is separately needed.
2456lcd_demo\2456lcd_demo\ include	lcd.h	Execution library for H8S	(*1) Purchase is separately needed.
2456lcd_demo\2456lcd_demo\ include	mgt.h	Execution library for H8S	(*1) Purchase is separately needed.

Directory Name	File Name	Content	Note
2456lcd_demo\2456lcd_demo\include	wgp.h	Execution library for H8S	(*1) Purchase is separately needed.
2456lcd_demo\2456lcd_demo\include	wgp_chr.h	Execution library for H8S	(*1) Purchase is separately needed.
2456lcd_demo\2456lcd_demo\include	wgp_depth.h	Execution library for H8S	(*1) Purchase is separately needed.
2456lcd_demo\2456lcd_demo\LCD	DirectLCD_CNF_Kyocera.h	Set value related to LCD	For LCD made by Kyocera
2456lcd_demo\2456lcd_demo\LCD	gui_drv.c	Low level driver for GUI	—
2456lcd_demo\2456lcd_demo\LCD	hwsetup.c	H8S/2456 register setting	—
2456lcd_demo\2456lcd_demo\LCD	lcd_common.h	Set value related to LCD	—
2456lcd_demo\2456lcd_demo\LCD	LCD_driver.h	LCD display driver	For LCD made by Kyocera
\LCD	LCD_Kyocera.c	LCD display driver	For LCD made by Kyocera
2456lcd_demo\2456lcd_demo\TouchScreen	Calib.c	Touch panel driver	For LCD made by Kyocera
2456lcd_demo\2456lcd_demo\TouchScreen	Calib.h	Touch panel driver	For LCD made by Kyocera
2456lcd_demo\2456lcd_demo\TouchScreen	TouchScreen2.c	Touch panel driver	For LCD made by Kyocera
2456lcd_demo\2456lcd_demo\TouchScreen	TouchScreen2.h	Touch panel driver	For LCD made by Kyocera
2456lcd_demo\2456lcd_demo\TouchScreen	TouchScreen.c	Touch panel driver	For LCD made by Kyocera

5.2 Correction of File

Copy the file under the SampleProject folder generated in Chapter 4.3 shown in Table 5 into 2456lcd_demo\2456lcd_demo\gui_usrapp folder on the H8 side shown in Table 4 before building under High-performance Embedded Workshop. As for the file excluding this, the correction is basically unnecessary. Correct it if there is a necessity.

Gui.out.h and gui.out.img**** are copied onto the folder as they are.

Gui.out.c and userevent.c need adjusting, and refer to Chapter 5.2.1 and Chapter 5.2.2, respectively.

Using main.c for Windows on the H8S/2456 side causes various errors. Therefore, for main.c, use SampleProject\main..

Table 5 File under SampleProject Folder to be Copied

File Name	Content	Processing
gui.out.h	Header file generated from C#	Copy the file for Windows as it is.
gui.out.img****	Picture file	Copy the file for Windows as it is.
gui.out.c	C source program generated from C#	(1) Change the name of gui.out.c to guiout.c because a compile error occurs under High-performance Embedded Workshop when two "." are included. (2) Allocate the picture file to external flash memory.
userevent.c	Event processing of user making	(1) Copy the file for Windows as it is. (2) Comment out Printf() for the debugging output in mgt_hook() because the H8S side doesn't support it.

5.2.1 Correction of gui.out.c File

Gui.out.c is C source automatically generated from C# by the guigen.exe tool. To build the C source in the High-performance Embedded Workshop environment of H8S/2456, the following two corrections are needed.

(1) Because High-performance Embedded Workshop cannot treat the file name including two "."or more, it is necessary to change the file name of gui.out.c to guiout.c. It is registered in High-performance Embedded Workshop reference project (2456lcd_demo) for H8S/2456 as guiout.c.

(2) Map the picture files (gui.out.img****) included in gui.out.c to external flash memory. H8S/2456 has the products of 128 Kbytes or 256 Kbytes in the flash capacity. It is not possible to put the data in on-chip flash memory when a lot of image data is used. In this case, allocate it to external flash memory. The R0K402456D000BR board used in this application note has 4-Mbyte NOR flash memory (S29JL032H70TFI410) made by Spansion in 0x00400000 and subsequent addresses.

Section name (EXT_FLASH) is set to external flash memory space under High-performance Embedded Workshop, and place the part of picture files (gui.out.img****) included in gui.out.c between # pragma section EXT_FLASH and # pragma section // EXT_FLASH.

When the image data is not specified in the design of VisualC #, picture files (gui.out.img****) are not made.

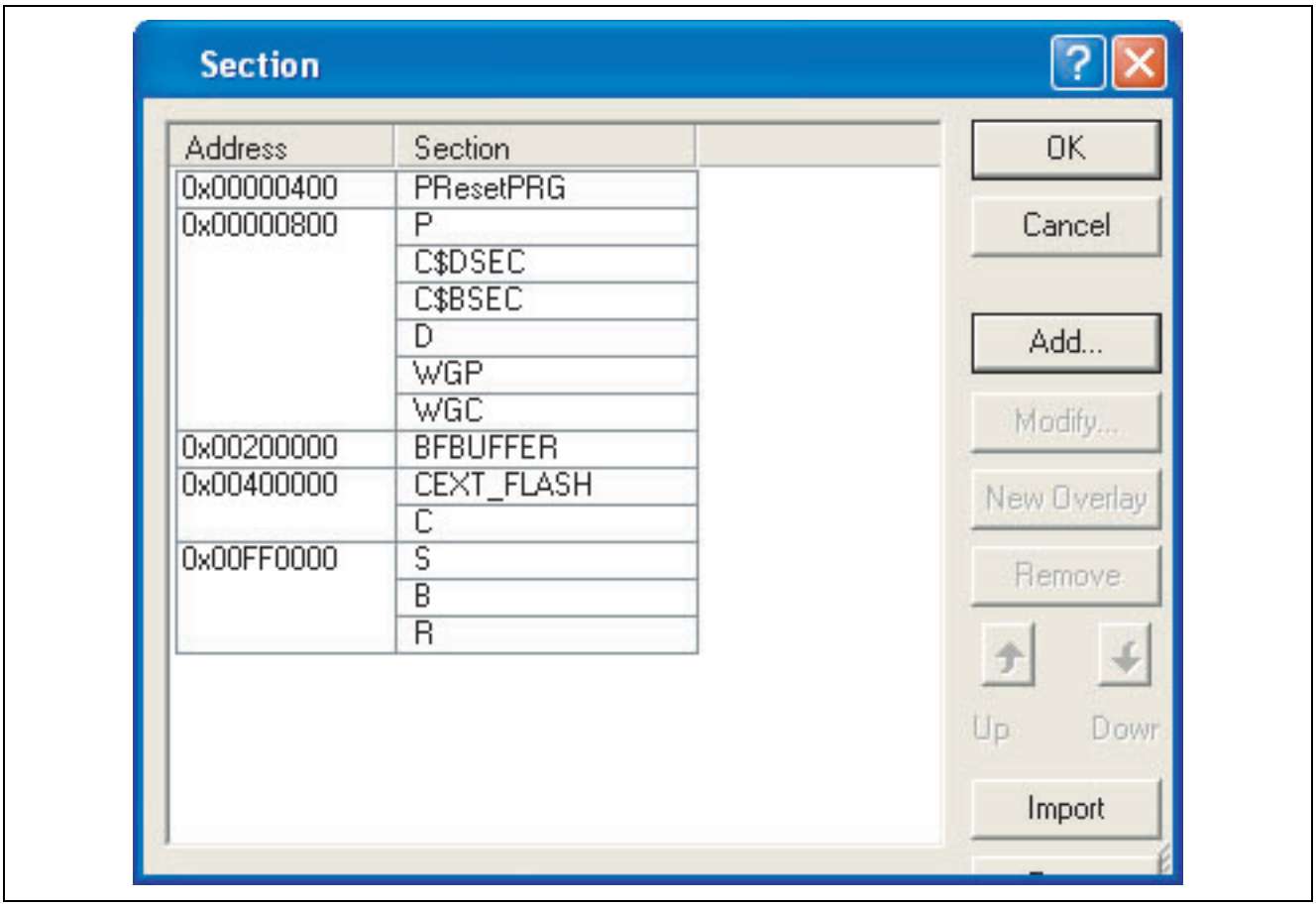


Figure 5 Section Name in External Flash Memory Space

[Before changing gui.out.c]

```
/* -*- encoding: sjis -*- */
/* generated by guigen (GUI Program Generator)  Version 1.0.0 */

#include <stddef.h>
#include "mgt.h"
#include "gui.out.h"

static const unsigned short mgt_img0001[] = {
#include "gui.out.img0001"
};
```

The number of statements with function equivalent to the above is the same as the number of image files.

```
static const unsigned short mgt_img0057[] = {
#include "gui.out.img0057"
};
```

[After changing gui.out.c]

```
/* -*- encoding: sjis -*- */
/* generated by guigen (GUI Program Generator)  Version 1.0.0 */

#include <stddef.h>
#include "mgt.h"
#include "gui.out.h"

#pragma section EXT_FLASH
static const unsigned short mgt_img0001[] = {
#include "gui.out.img0001"
};
```

The number of statements with function equivalent to the above is the same as the number of image files.

```
static const unsigned short mgt_img0057[] = {
#include "gui.out.img0057"
};
#pragma section    // EXT_FLASH
```

5.2.2 Correction of userevent.c File

(1) Because the H8S side doesn't support printf() function, comment out printf() function used for the debugging output on the Windows side. When the printf() function is used, the following error message is displayed under High-performance Embedded Workshop.

```
Phase OptLinker starting
L2310 (E) Undefined external symbol "_write" referenced in "_flshbuf"
Optimizing Linkage Editor Abort
Phase OptLinker finished
```

(2) The default constant for H8 is 16 bits while the default constant of C++ for Windows is 32 bits. H8S doesn't operate predictably when the integration result of the constant value exceeds the range of 16 bits. In this case, it is necessary to specify longword (32 bits) intentionally by adding "L" to the numerical value.

5.3 main.c File

Use SampleProject\main.c because using main.c of Windows on the H8S/2456 side causes the inconvenience problems. Do a correction equivalent to correction for the H8S/2456 side when you change main.c on the Windows side.

(1) Two frame buffers are allocated in external RAM

The example of the reference of main.c in the case of one frame buffer is described in "Sample of the main program for appendix B. PC" of the GUI development tool user's manual of the attached paper. However, because screen rewriting is displayed every time an event occurs, in the case of one frame buffer for the R0K402456D000BR board, the screen flickers. Therefore, we recommend two frame buffers for the flicker prevention.

0x4B000 is required as the capacity of RAM because two frame buffers of 320×240 are used in the RGB565 format (16 bits/dot). These buffers are assigned in external RAM that exists on 0x0020000 and subsequent addresses. FBUFFER is defined in the section and specified under main.c as follows.

```
#pragma section FBUFFER
static unsigned short fb1[WIDTH * HEIGHT];
static unsigned short fb2[WIDTH * HEIGHT];
#pragma section   //FBUFFER
```

(2) The default constant for H8 is 16 bits while the default constant of C++ for Windows is 32 bits. H8S doesn't operate predictably when the integration result of the constant value exceeds the range of 16 bits. In this case, it is necessary to specify longword (32 bits) intentionally by adding "L" to the numerical value.

L is added to the constant value of WIDTH and HEIGHT because the size of the frame buffer exceeds the range of 16 bits.

(3) The mgt_fontset16 font is used in the Main() function in the GUI development tool user's manual of the attached paper. However, the capacity of ROM after building is very large. Therefore, we recommend the mgt_fontset8 for H8S/2456 because H8S/2456 ROM capacity is small.

When the mgt_fontset16 font is used, approximate 247-Kbyte ROM capacity is additionally needed, compared with the case of the mgt_fontset8 font. However, this shall not apply in the case where mass flash memory is installed outside. Mgt_fontset8 is used in 2456lcd_demo folder as a reference.

(4) The GUI development tool user's manual of the attached paper shows an example of calling lcd_delay() function for adjusting timing and adjusting the timing within while (1) loop of main() function lcd_delay() function. However, we recommend that lcd_delay() function should be used only in main.c on the Windows. While the R0K402456D000BR board has the advantage in which the QVGA LCD display can be achieved at a low price becomes because the board has achieved the LCD display in the steel mode of the cycle of EXDMAC of H8S/2456, external bus access is slow accordingly. The processing of the main function becomes heavy if _ lcd_delay() function is executed on the H8S/2456 side, and the reaction of the touch panel worsens.

5.4 High-performance Embedded Workshop Section Definition

The section is allocated to external RAM that exists in 0x0020000 and subsequent addresses and to 4-Mbyte NOR flash memory that exists in 0x00400000 and subsequent addresses besides H8S/2456 on-chip memory.

(1) Picture files (gui.out.img****) included in gui.out.c are allocated to external flash memory area (Refer to Chapter 5.2.1).
(2) The constant (section C) is allocated to external flash memory area.
(3) Two frame buffers are allocated to external RAM (Refer to Chapter 5.3).

Assign the areas of section C and section EXT_FLASH to on-chip ROM when the capacity of ROM above-mentioned (1) (2) is smaller than that of on-chip ROM area. Because the image data are transmitted between external RAM and LCD by using EXDMAC, and the size of the frame buffer is large, the allocation of external RAM of (3) is indispensable.

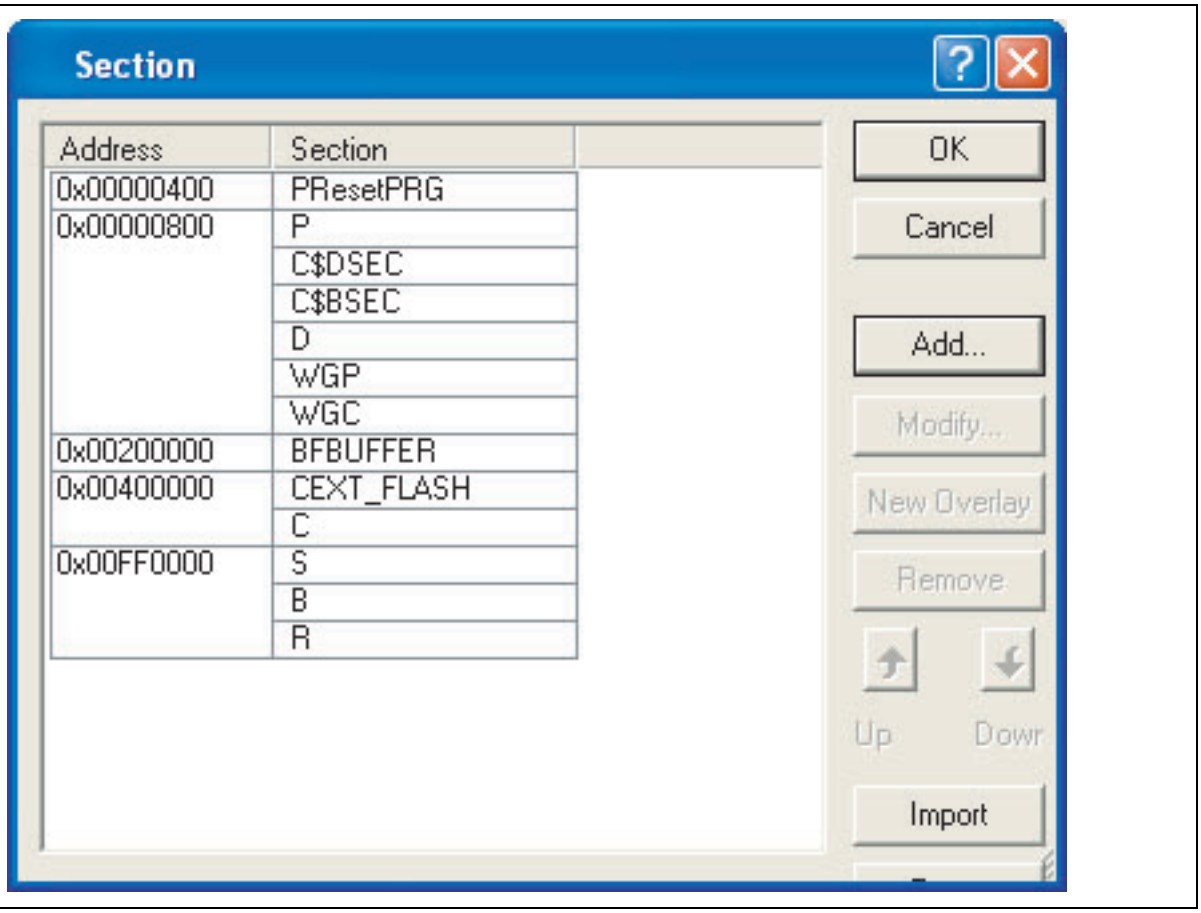


Figure 6 High-performance Embedded Workshop Section Definition

5.5 Building under High-performance Embedded Workshop

Select Build All or Build of the Build menu and build all files. Refer to the manual of High-performance Embedded Workshop for the usage of High-performance Embedded Workshop.

5.6 Copy of Files for Writing to External Flash Memory

Copy the configuration file and driver in advance, for erasure and programming from High-performance Embedded Workshop to 4-Mbyte NOR flash memory (S29JL032H70TFI410) made by Spansion, mounted on the R0K402456D000BR board . The destination of the copy of the file is not cared about anywhere.

Correct pass of 1MB.EFF and 3MB.EFF files according to the copy destination of FMTOOL.mot.

```
[FLASH_MODE]
USE_FLASH=1
ERASE_FLASH=1
[FLASH_MODULE]
FLIB_NAME=C:\Program Files\Renesas\ExtFlashDriver\FMTOOL.mot
FLIBOF_FLAG=0
FLIB_OFFSET=0
FLIB_TOP=ff2000
```

Table 6 File List for Programming to External Flash Memory

File Name	Feature	Note
FMTOOL.mot	Programming and erasing driver to S29JL032H70TFI410 for R0K402456D000BR board	Indirectly specified from 1MB.EFF or 3MB.EFF configuration file.
1MB.EFF	Configuration file corresponding to erasure and programming to external flash memory area from 0x400000 to 0x4fffff	External flash setting file Either 1MB.EFF or 3MB.EFF is specified from HIGH-PERFORMANCE EMBEDDED WORKSHOP.
3MB.EFF	Configuration file corresponding to erasure and programming to external flash memory area from 0x400000 to 0x6fffff and writing	Same as above

5.7 Connection of R0K402456D000BR Board and E10A

To set the operational mode of H8S/2456 to on-chip ROM effective enhancing and the emulator use mode, turn on the switches 1 and 2 of DIPSW(DSW1) next to the E10A connector and off the switches 3 and 4 (Refer to Figure 8.)

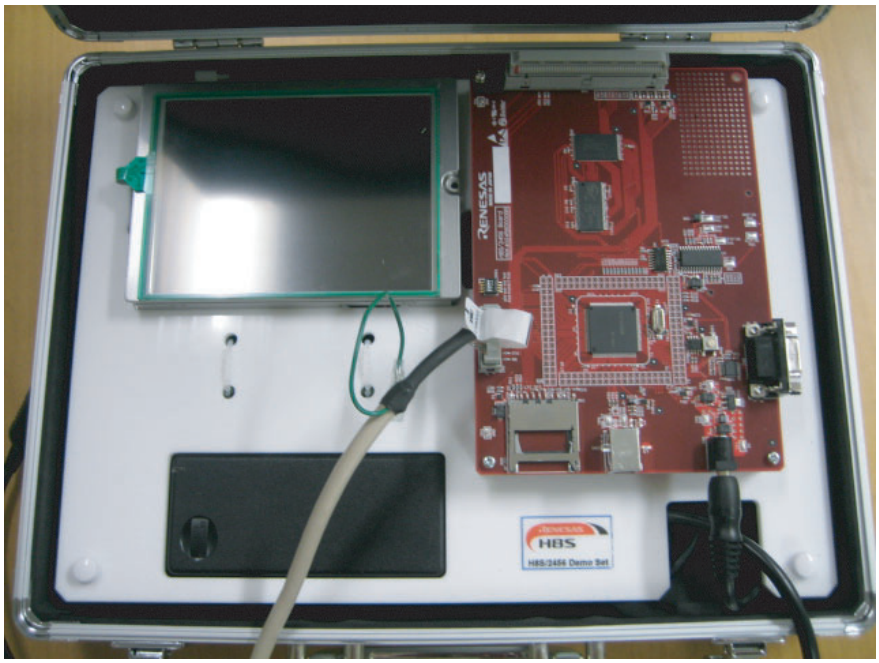


Figure 7 Connection of R0K402456D000BR Board and E10A

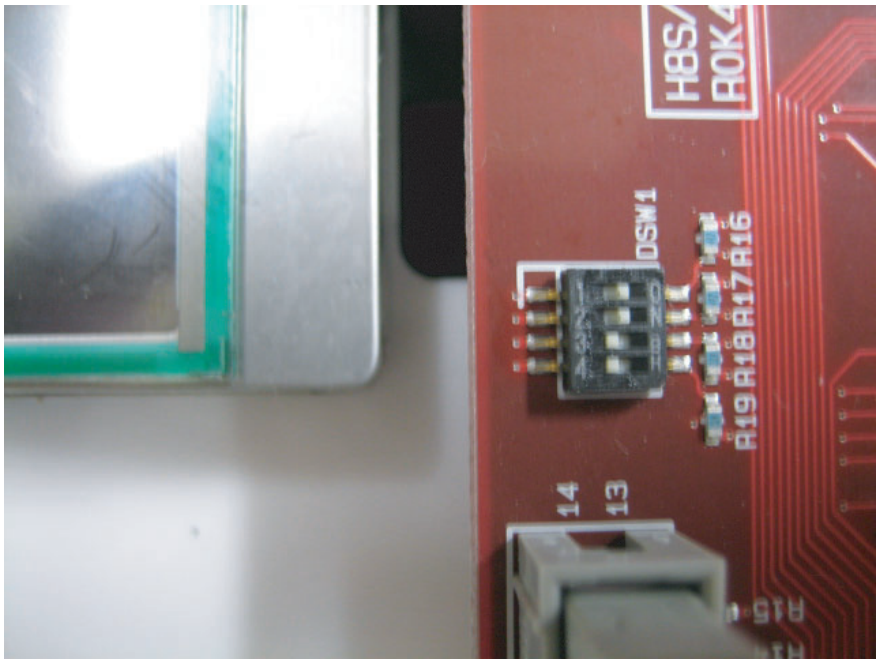


Figure 8 SW Setting when E10A Operates

5.8 Writing of Program to Memory Including External Flash Memory

Connect the USB connector of E10A with Windows beforehand while power is supplied to the R0K402456D000BR board. Note that E10A cannot be recognized even if the USB connector of E10A is connected with Windows with the power supply not supplied to the R0K402456D000BR board.

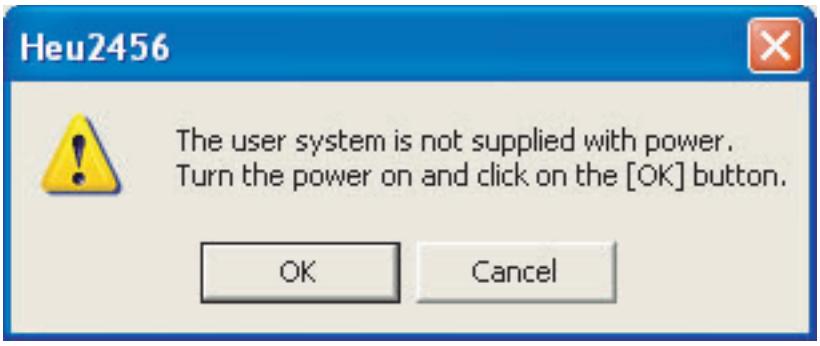


Figure 9 When you Connect the USB Connector of E10A with Windows with the Power Supply not Supplied

Click 2456lcd_demo\2456lcd_demo.hws and start HIGH-PERFORMANCE EMBEDDED WORKSHOP. In Figure 10,

(1) Foe the device, select H8S/24568.
(2) For the mode, select E10A-USB Emulator.
(3) To write external flash memory, put check mark put in Use External Flash memory setting.

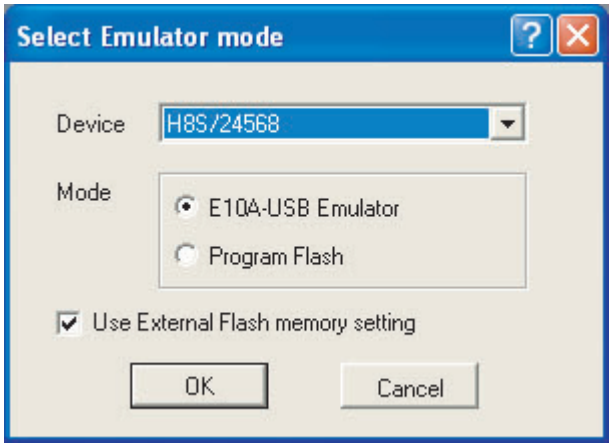


Figure 10 Select Emulator Mode

Specify External Flash setting file (3MB.EFF) copied under Windows in Chapter 5.6. The settings to write external flash memory and the path of the driver for erasure and programming to external flash memory have been written to this file. There is External Flash setting file (1MB.EFF), supporting erasure and programming up to 1 Mbyte. However, because the example in this application note uses approximately 2.5-Mbyte capacity in external flash memory area, select 3MB.EFF.

Push the Browse button of Figure 11 and specify the destination in Select External Flash setting.

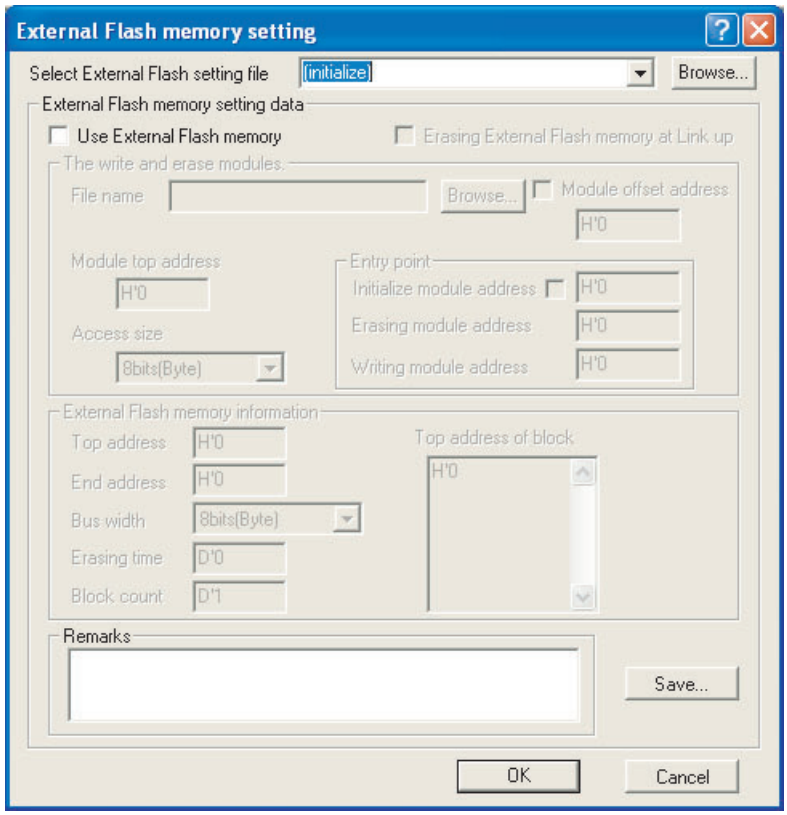


Figure 11 Selection of External Flash Setting File

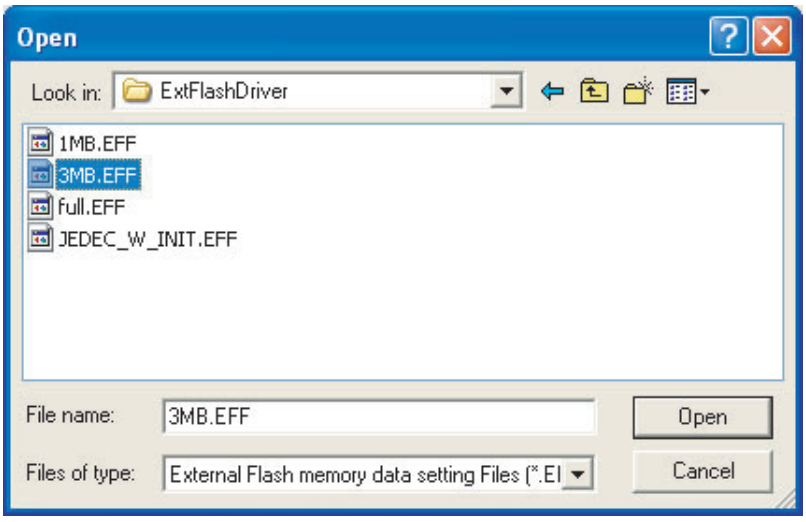


Figure 12 Selection of External Flash Setting File (2)

When reading External Flash setting file is completed as shown in Figure 13, push the OK button.

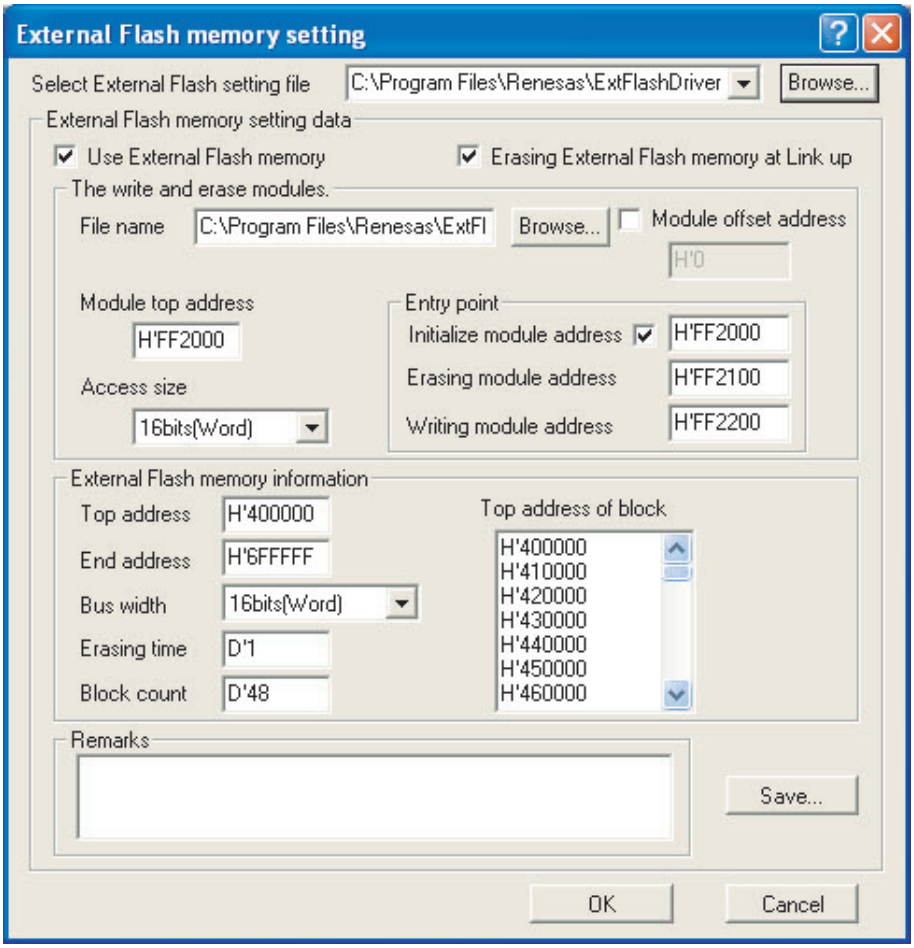


Figure 13 After the External Flash Setting File Reading

Value (16MHz) of the frequency of EXTAL is set.

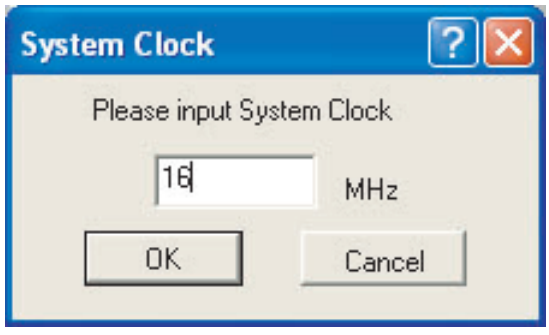


Figure 14 Setting of Frequency for EXTAL

Push the OK button without changing default of ID Code.

Figure 15 Input of ID Code

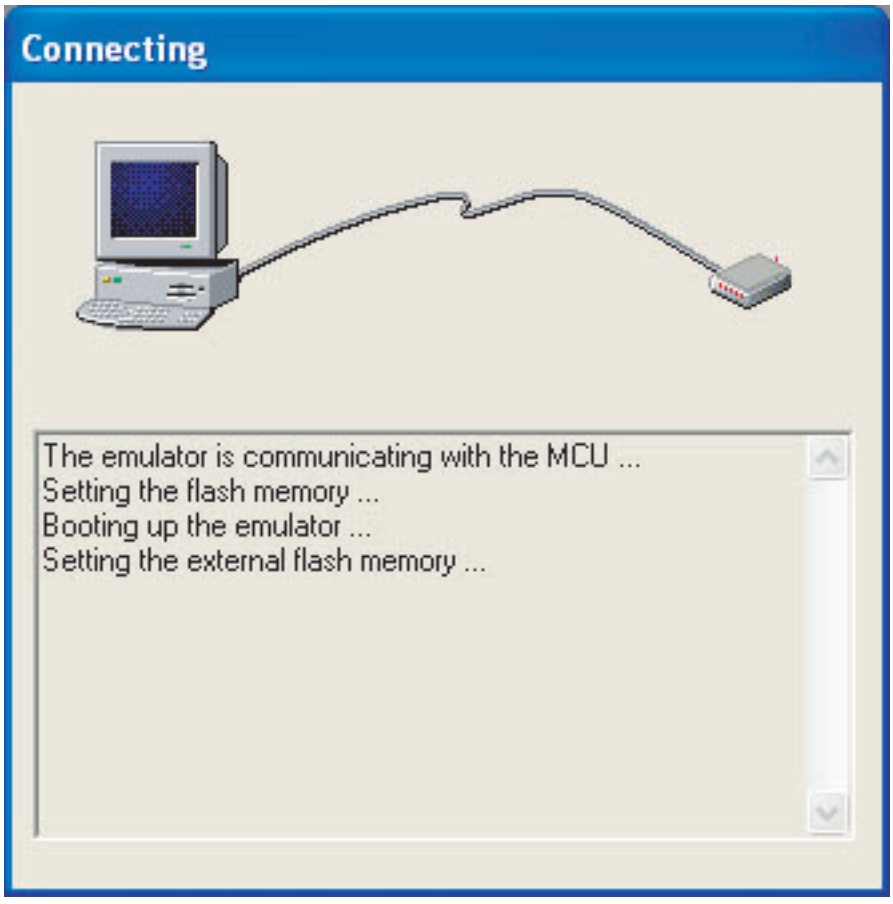
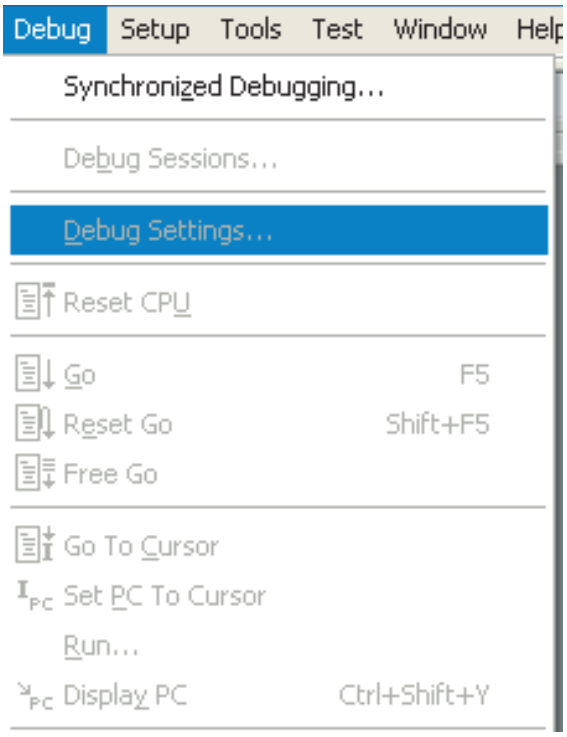


Figure 16 Picture of Connecting

When High-performance Embedded Workshop starts, select Debug Settings (D) from the Debug menu.



For Target, select H8S/2456 E10A-USB SYSTEM (CPU2600).

For Debug format, select Elf/Dward2.

Set $(CONFIGDIR) \$(PROJECTNAME).abs as a filename of Download modules.

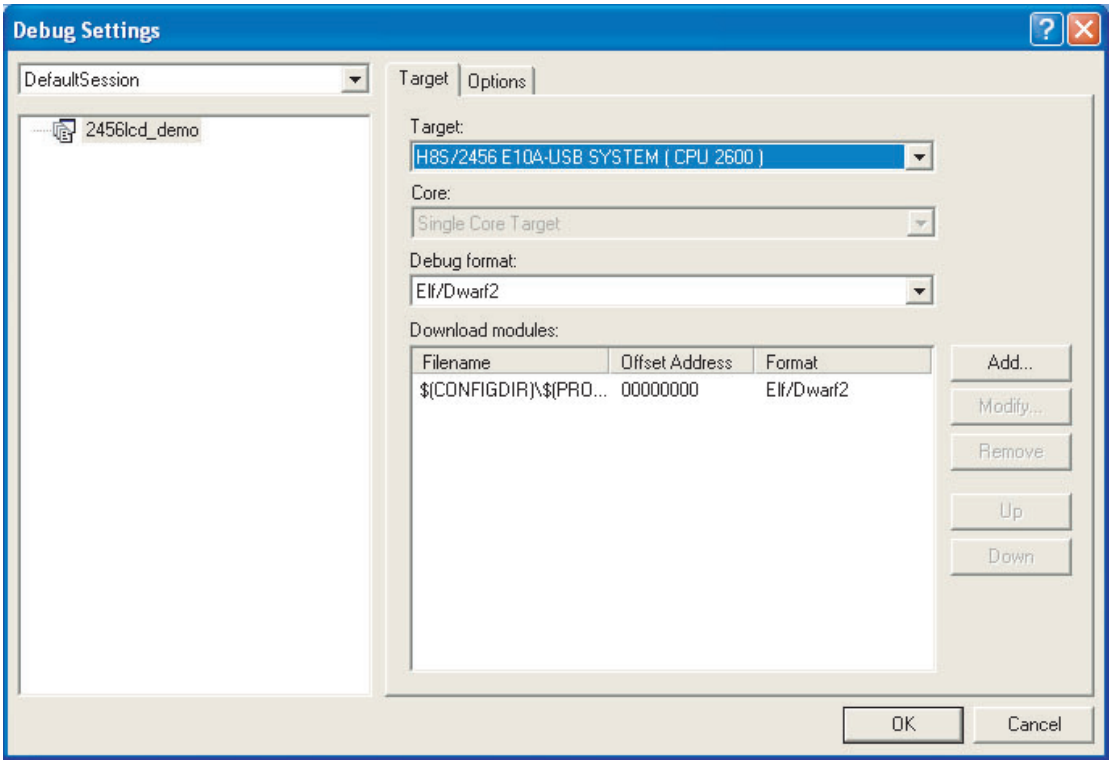


Figure 17 Picture of Debug Settings

Select "2456lcd_demo.abs" built in Chapter 5.5 from Download modules command in Debug.

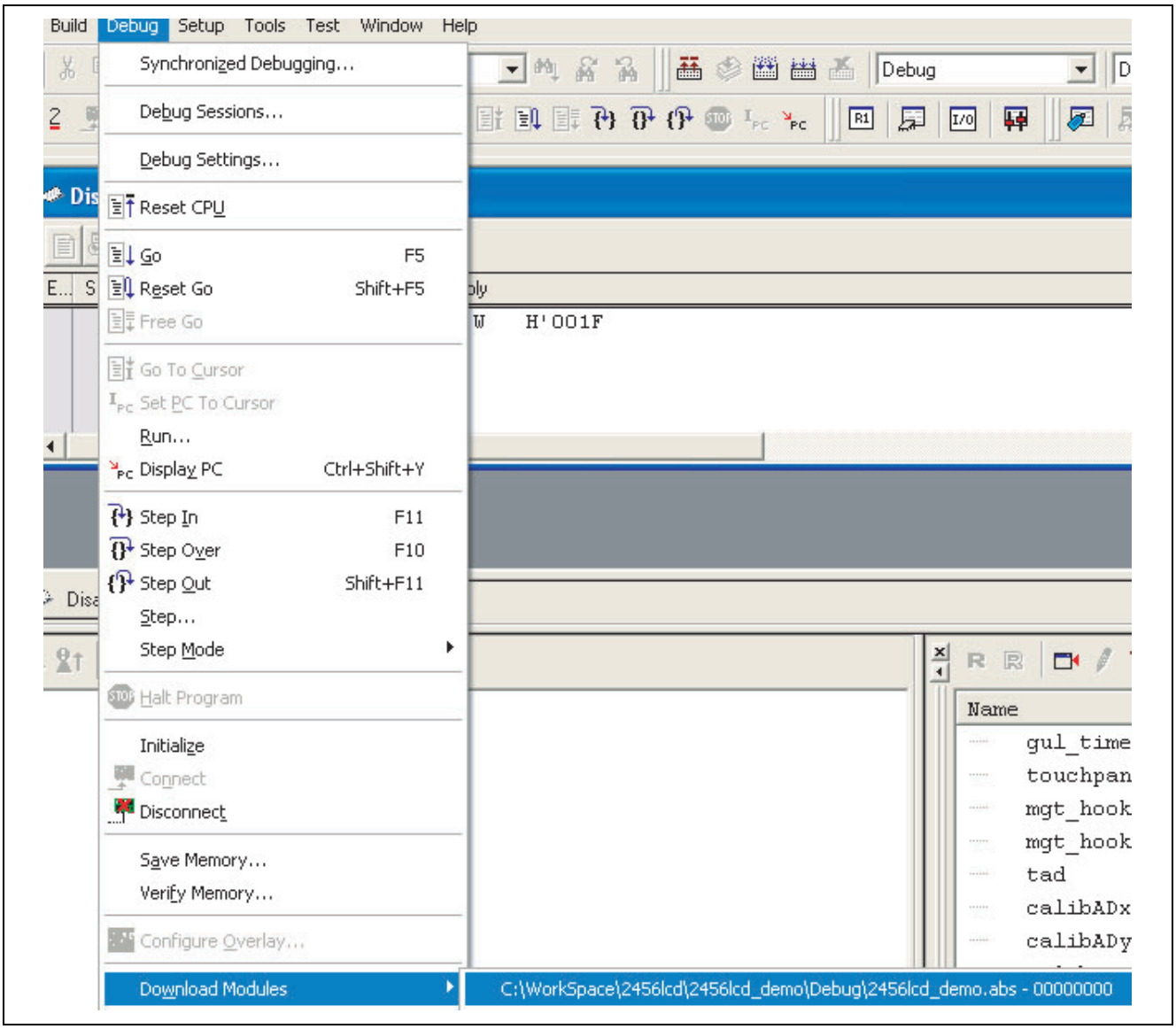


Figure 18 Selection of 2456lcd_demo.abs File

Push the reset button and the GO button, and write programs to on-chip flash memory and external flash memory.

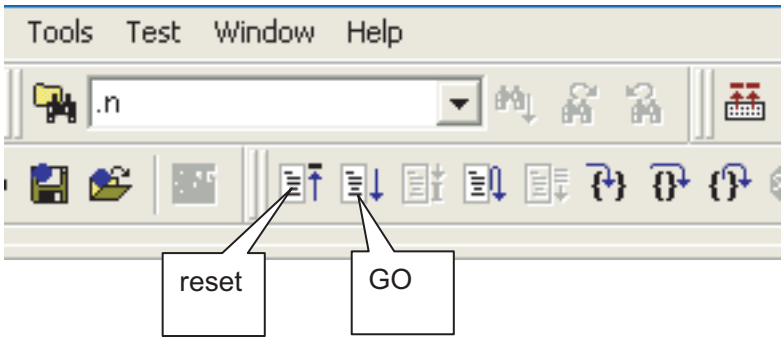


Figure 19 Reset and GO

If the following message is displayed, writing is completed.

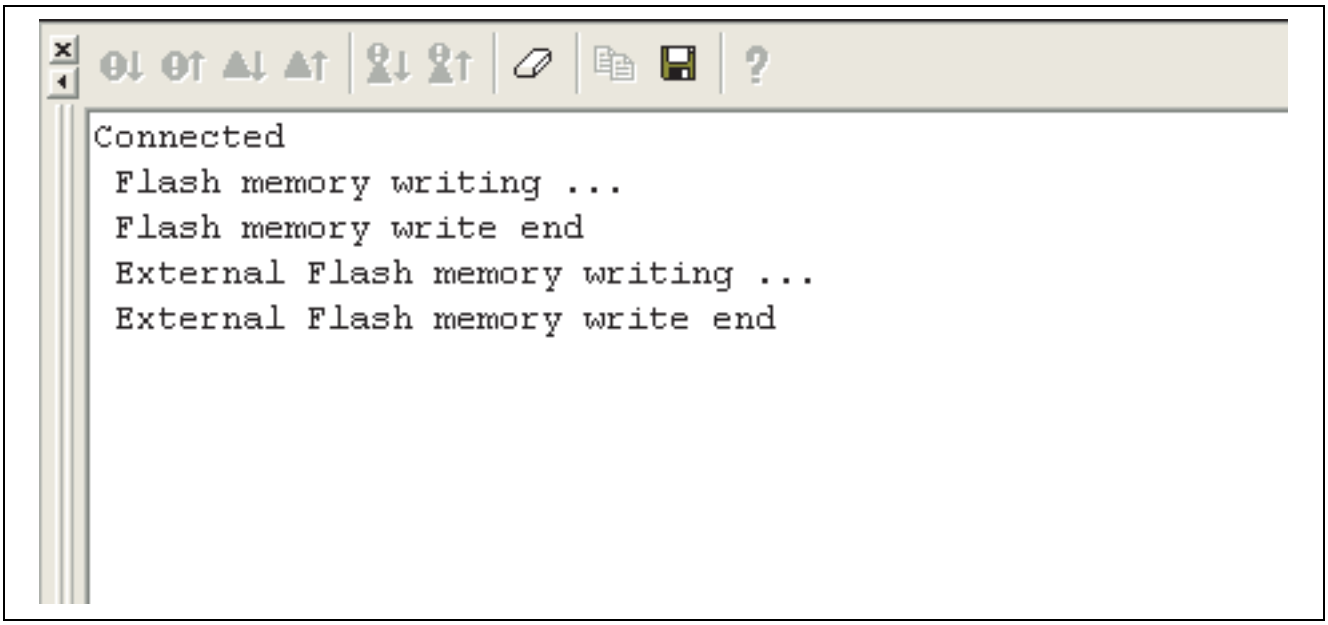


Figure 20 Completion of Writing to External Flash Memory

5.9 Execution of Program

When the Go button is pushed under High-performance Embedded Workshop, the program on H8S side is executed on completion of writing programs.

After the program is executed, it is necessary to adjust the touch panel first. Click "Plus Mark" on the upper left with a fine-tipped thing. Subsequently click "Plus Mark" displayed on the lower right as well. The position where the button is detected shifts when the click position shifts, and click the mark accurately.

Figure 21 Adjustment of Touch Panel (1)

Figure 22 Adjustment of Touch Panel (2)

The main screen starts after completing the adjustment. Moreover, transition to this screen is made when the main button is pushed from the Home screen. There is no corresponding event code in the userevent.c file because transition to the Home screen is made by VisualC# OnClickGoto event when the button at the center is pushed.

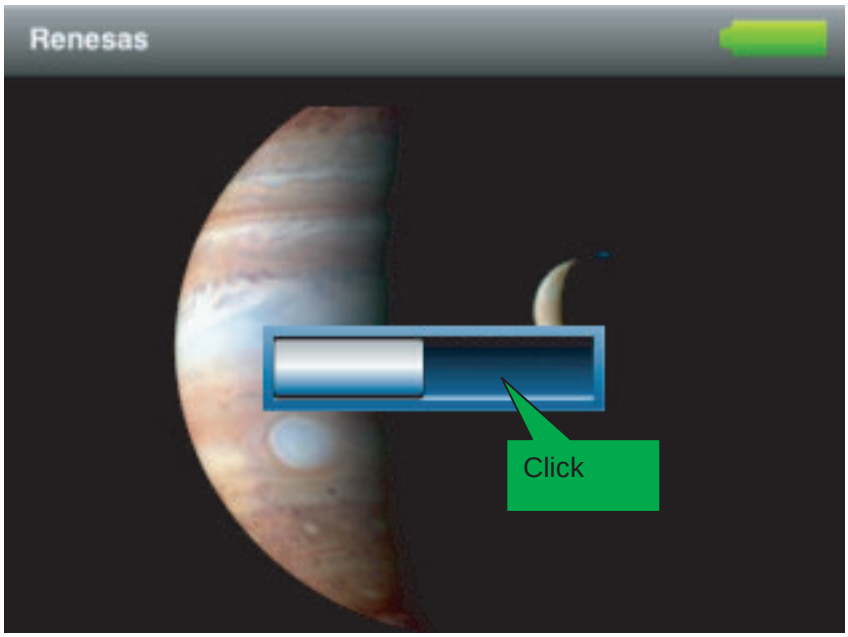


Figure 23 Main Display

The Home screen starts when the button at the center is clicked. There are five buttons in the Home screen, and it changes into the screen corresponding to the pushed button. There is no corresponding event code in the userevent.c file because this processing is included in VisualC# OnClickGoto event.

Figure 24 Home Display

Neither the OnClickGoto event of Visual C# nor the event code corresponding to the userevent.c file are described in the button for 1 to 9, *, and #. Therefore, pushing these buttons causes no change.

When the Offhook button is pushed, animation is displayed on MGTPicutreBox. Under such a condition, animation on MGTPicutreBox stops when the onhook button is pushed. Transition to the Home screen is made when the onhook button is pushed with animation stopped. This has been achieved with MGT_HOOK_POLLING (processing at the cycle) and MGT_HOOK_ONCLICK (button click event) in the mgt_hook() function of userevent.c file.

Figure 25 Phone Display

There are six buttons in the Photos screen. When a small image is put on the button, and the button is pushed, the corresponding screen (big image) is displayed. It returns to the Home screen when X button is pushed. There is no corresponding event code in the userevent.c file because this processing is included in VisualC# OnClickGoto event.

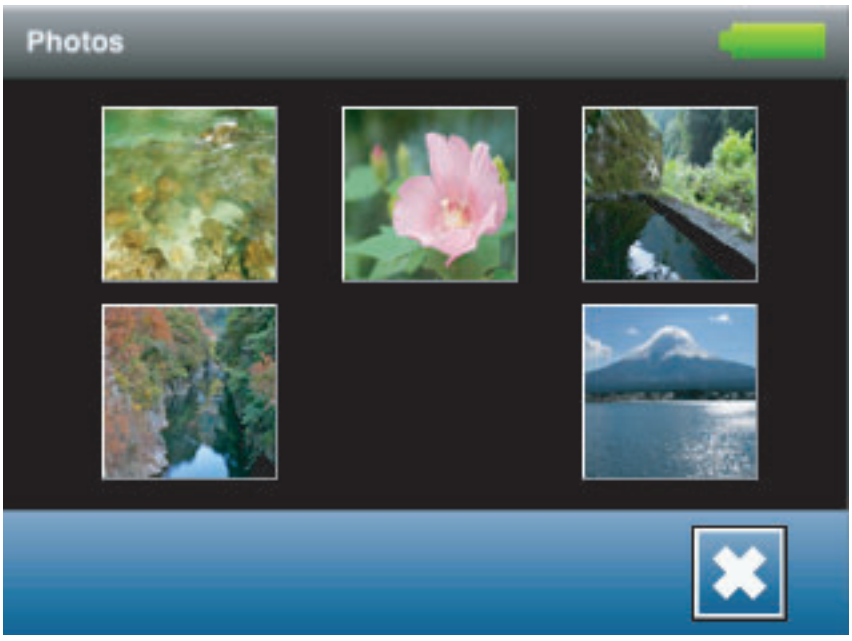


Figure 26 Photo Display

This display is mounted with the same mechanism as shown in Figure 26. It returns to the previous screen with a left button. Moreover, it goes forward to the following screen with a right button. Because other Photos screens are mounted by the same mechanism, explanation is omitted.

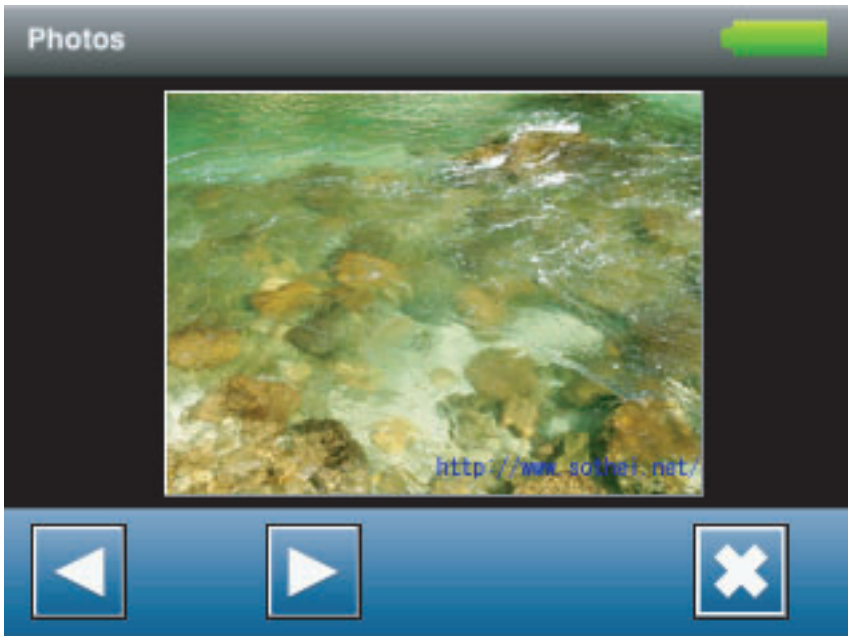


Figure 27 Photos Display 2

The WWW screen is composed of two or three buttons and the BackgroundImage image. The image of the WWW site is put on BackgroundImage property of the form (screen) under VisualC#. WWW is composed of three screens. All are mounted by the same mechanism as Figure 26.

It returns to the previous screen with a left button. Moreover, it goes forward to the following screen with a right button and it returns to the Home screen when X button is pushed.

Figure 28 WWW Display

The Music selection screen is composed of MGTPictureBox, MGTListBox, and four buttons.

The text of MGTListBox and the image of MGTPictureBox are switched with a left button and a right button. The screen is switched to the music play screen with the play button. Moreover, it returns to the music selection screen when X button is pushed on the music play screen. It returns to the Home screen when X button is pushed on the music selection screen.

These have been achieved by the OnClickGoto event of VisualC#, and MGT_HOOK_ENTER (initialization of the screen), MGT_HOOK_POLLING (processing at the cycle), and MGT_HOOK_ONCLICK (button click event) in the mgt_hook() function of the userevent.c file.

Figure 29 MUSIC Selection Display

Website and Support

Renesas Technology Website
http://www.renesas.com/

Inquiries
http://www.renesas.com/inquiry
csc@renesas.com

Revision Record

		Description	
Rev.	**Date**	**Page**	**Summary**
0.90	Jan.26.09	—	First edition issued
1.00	Sep.28.09	31-33	Configuration files in Table 4 File List of High-performance Embedded Workshop Reference Work Space (2456lcd_demo) for H8S/2456 changed

Notes regarding these materials

1. This document is provided for reference purposes only so that Renesas customers may select the appropriate Renesas products for their use. Renesas neither makes warranties or representations with respect to the accuracy or completeness of the information contained in this document nor grants any license to any intellectual property rights or any other rights of Renesas or any third party with respect to the information in this document.
2. Renesas shall have no liability for damages or infringement of any intellectual property or other rights arising out of the use of any information in this document, including, but not limited to, product data, diagrams, charts, programs, algorithms, and application circuit examples.
3. You should not use the products or the technology described in this document for the purpose of military applications such as the development of weapons of mass destruction or for the purpose of any other military use. When exporting the products or technology described herein, you should follow the applicable export control laws and regulations, and procedures required by such laws and regulations.
4. All information included in this document such as product data, diagrams, charts, programs, algorithms, and application circuit examples, is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas products listed in this document, please confirm the latest product information with a Renesas sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas such as that disclosed through our website. (http://www.renesas.com)
5. Renesas has used reasonable care in compiling the information included in this document, but Renesas assumes no liability whatsoever for any damages incurred as a result of errors or omissions in the information included in this document.
6. When using or otherwise relying on the information in this document, you should evaluate the information in light of the total system before deciding about the applicability of such information to the intended application. Renesas makes no representations, warranties or guaranties regarding the suitability of its products for any particular application and specifically disclaims any liability arising out of the application and use of the information in this document or Renesas products.
7. With the exception of products specified by Renesas as suitable for automobile applications, Renesas products are not designed, manufactured or tested for applications or otherwise in systems the failure or malfunction of which may cause a direct threat to human life or create a risk of human injury or which require especially high quality and reliability such as safety systems, or equipment or systems for transportation and traffic, healthcare, combustion control, aerospace and aeronautics, nuclear power, or undersea communication transmission. If you are considering the use of our products for such purposes, please contact a Renesas sales office beforehand. Renesas shall have no liability for damages arising out of the uses set forth above.
8. Notwithstanding the preceding paragraph, you should not use Renesas products for the purposes listed below:
   (1) artificial life support devices or systems
   (2) surgical implantations
   (3) healthcare intervention (e.g., excision, administration of medication, etc.)
   (4) any other purposes that pose a direct threat to human life

   Renesas shall have no liability for damages arising out of the uses set forth in the above and purchasers who elect to use Renesas products in any of the foregoing applications shall indemnify and hold harmless Renesas Technology Corp., its affiliated companies and their officers, directors, and employees against any and all damages arising out of such applications.
9. You should use the products described herein within the range specified by Renesas, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas shall have no liability for malfunctions or damages arising out of the use of Renesas products beyond such specified ranges.
10. Although Renesas endeavors to improve the quality and reliability of its products, IC products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Please be sure to implement safety measures to guard against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other applicable measures. Among others, since the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
11. In case Renesas products listed in this document are detached from the products to which the Renesas products are attached or affixed, the risk of accident such as swallowing by infants and small children is very high. You should implement safety measures so that Renesas products may not be easily detached from your products. Renesas shall have no liability for damages arising out of such detachment.
12. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written approval from Renesas.
13. Please contact a Renesas sales office if you have any questions regarding the information contained in this document, Renesas semiconductor products, or if you have any other inquiries.