Software Development Tools

Chapter 4

Renesas Electronics America Inc.
Embedded Systems using the RX63N

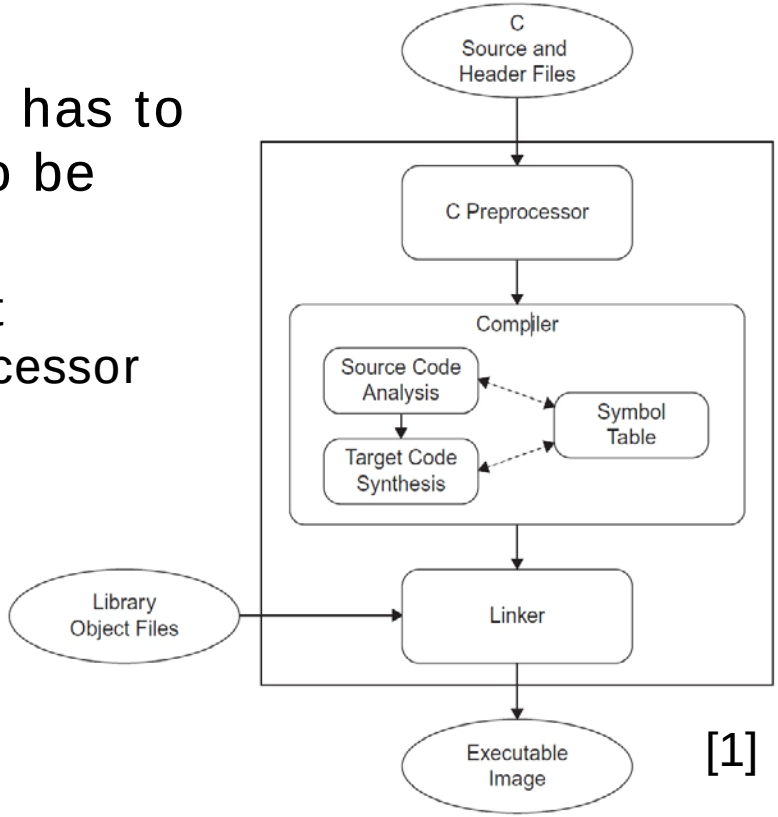
Rev. 1.0

In this chapter we will learn

- The process of compilation
- Features of the RX Family compiler
- Debugging tools
- Features of the High-performance Embedded Workshop
- Various header files associated with a C/C Compiler and the RX Family

Compilation Mechanism

- The three tools that a C program has to pass through before it is ready to be executed are:
  - The **preprocessor** performs text replacement according to preprocessor directives.
  - The **compiler** transforms source code (in this case C) into object code which is in binary form.
  - The **linker** combines all objects generated by the compiler into a single executable program.



[1]

PROCESSOR	INPUT	OUTPUT
Preprocessor	C source code file	Source code file with processed preprocessor commands
Compiler	Source code file with processed preprocessor commands	Re-locatable object code
Linker	Re-locatable object code and the standard C library functions	Executable code in machine language

[1]

Compilers for Embedded Systems

- Compilers for embedded systems differ from compilers for general-purpose processors (GPPs).
- Minimizing code and data size is often critical for embedded systems since memory sizes are limited due by price pressures.
- Optimizing code for speed is also often critical.
- Unlike GPPs, embedded processors usually run at lower clock rates, meaning that optimizations in how the processor accesses registers and memory have to be made.

RX Compiler Package

- The RX compiler package includes the following embedded system development tools:
  - **C/C++ Compiler**
    - Translates source code into assembly code
    - Supports C and C++
    - Generates compact and high-speed object code
  - **Assembler**
    - Translates assembly code into object (machine) code
  - **Optimizing Linkage Editor**
    - Creates load modules and library files
  - **Standard Library Generator**
    - Creates customized versions of the standard library files based on user-specified options
  - **Simulator/Debugger**
  - **Utilities**
    - Various utilities for monitoring the stack, map file, etc.

RX Compiler Package (cont.)

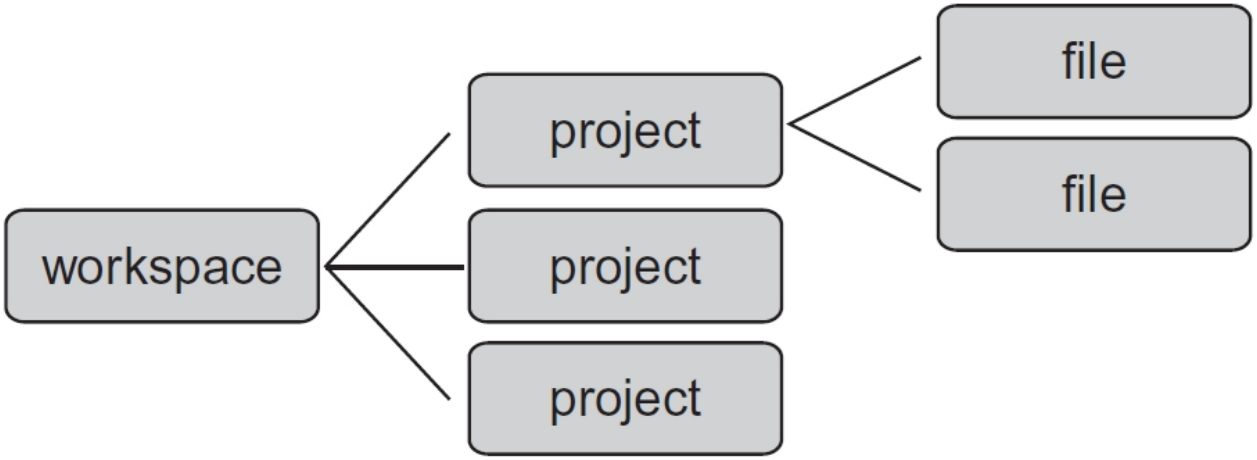
- **IDE (Integrated Development Environment)**
  - We will be using **HEW (High-performance Embedded Workshop)**
  - GUI based IDE for debugging embedded applications

Debugging Tools

- Two primary tools are used for debugging embedded real-time systems:
- Debugger
  - Allows the developer to control a program's execution and data
  - Execution can be started, paused, or stopped
  - Breakpoints can be inserted, pausing the execution of the program once that line of code has been reached
- Monitor
  - Allows the programmer to examine the program's temporal behavior
  - For example, oscilloscopes and logic analyzers
  - Monitoring tools can be used to find the max/min frequency and giving you insights about the timing of operations in the system

Introduction to HEW

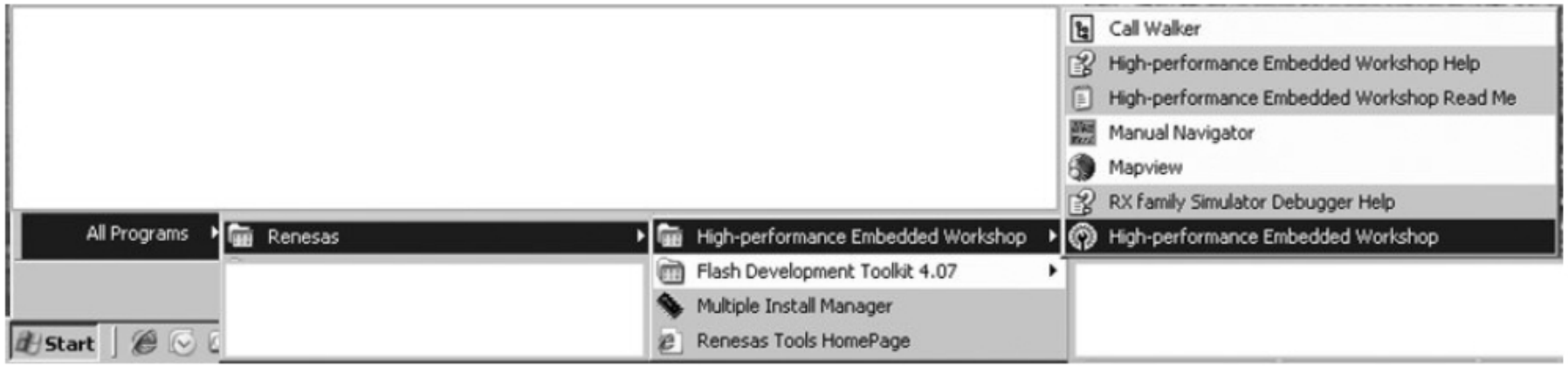
- HEW organizes your work with the concepts of workspaces and projects
- HEW organizes your work with **Workspaces** and **Projects**
- The workspace contains programs written in HEW
- In a project is where you write your program
  - Creating a new project is creating a new program
  - Each project consists of one or many files



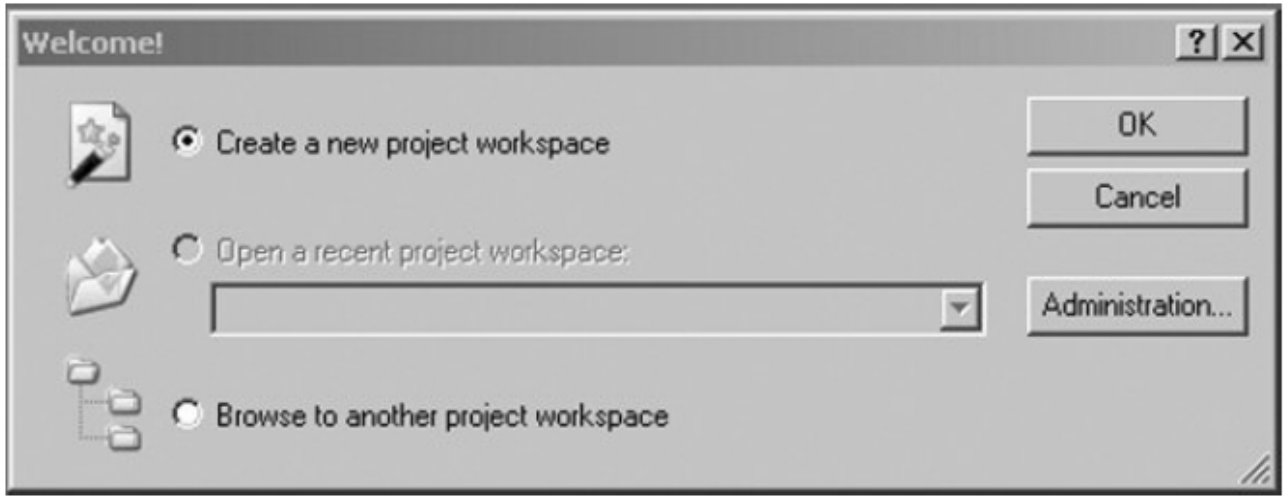
[1]

Getting Started with HEW

- The HEW IDE can be launched from the start menu



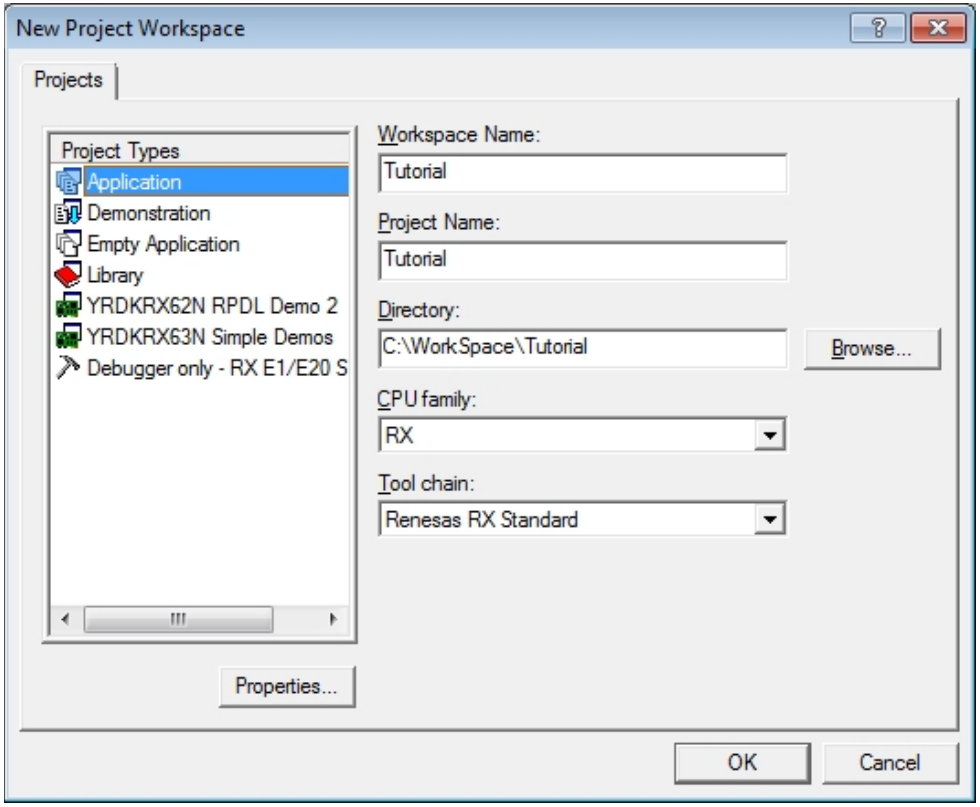
- And this is what you will see once you have launched HEW [1]



[1]

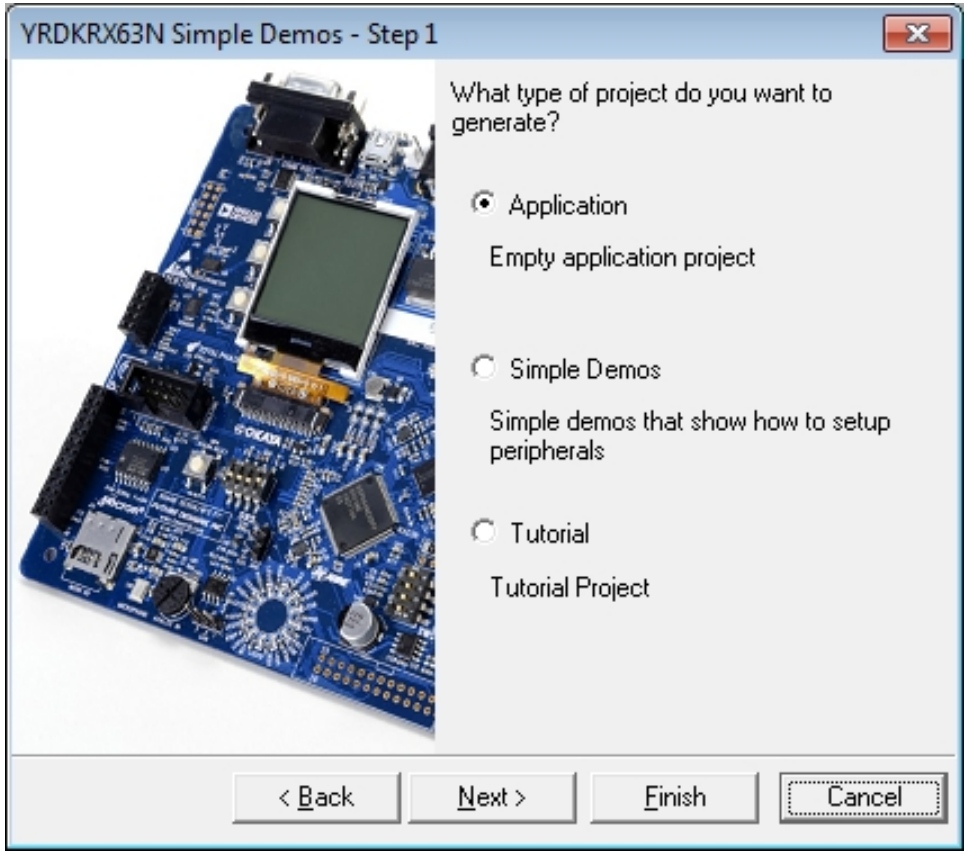
Getting Started with HEW (cont.)

- click the **OK** button to create a new project
- Enter a new Workspace Name and Project Name
- Select a directory for your Workspace and make sure the **RX** is selected in CPU family, and **Renesas RX Standard** is selected for the tool chain
- Select **YRDKRX63N Simple Demos** under Project Types
- Press **OK** to continue



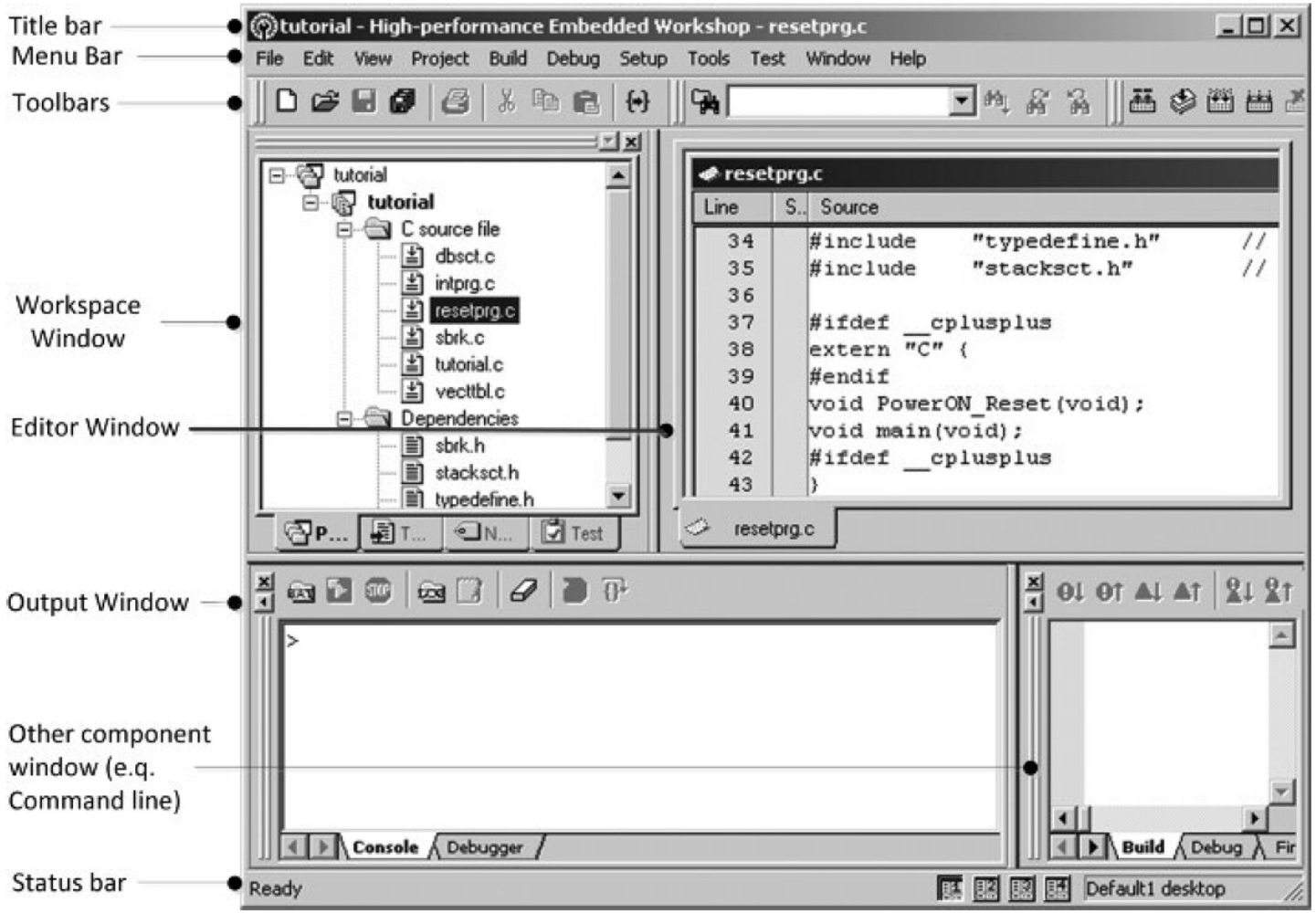
Getting Started with HEW (cont.)

- In this window you may either create an empty application, choose a demo, or select a tutorial project
- An empty application creates a framework upon which you may write a new program
- Demos are used to demonstrate various peripherals on the RX63N development board
- The tutorial project will flash the LEDs on the development board once the program has been executed



Getting Started with HEW (cont.)

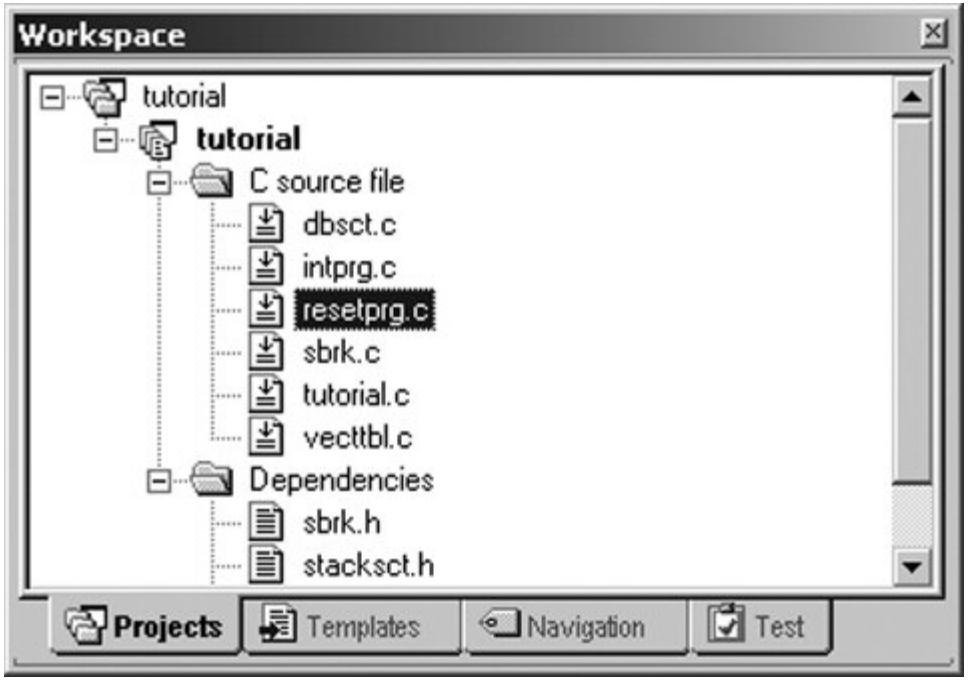
- This is what the IDE will look like once a project has been created



[1]

Windows in HEW

- There are various windows in HEW used to keep your programs organized
- The **Workspace Window** shows the projects currently in the workspace, and all files associated with it



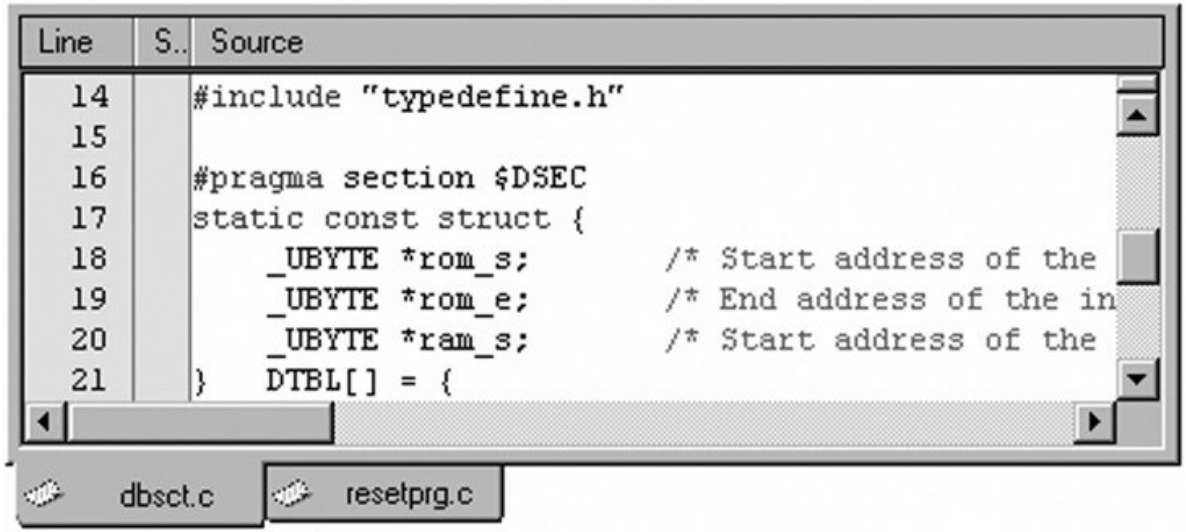
[1]

Windows in HEW (cont.)

- The **Output Window** shows the result of various processes (e.g., compiling)

[1]

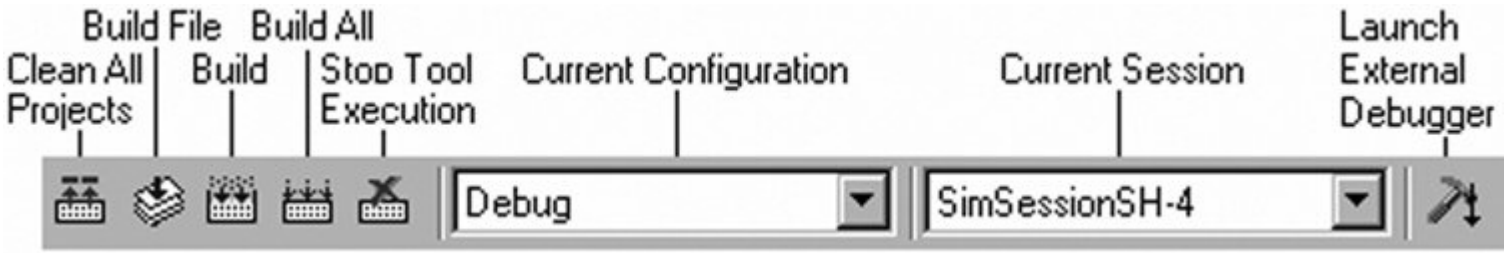
- The **Editor Window** is where you will write your code
  - In the window below, both dbsct.c and resetprg.c are open



[1]

Toolbars in HEW

- There are several different toolbars available in HEW; available are functions such as connecting the development board, stopping execution of the code, resetting the CPU etc.
- The toolbar below contains options for building your project, and selecting the current configuration and current session



[1]

- When you create your project, if you have selected a target debugger HEW will automatically select the configuration best suited for that debugger
- Sessions are used to manage various settings related to debugging options

Header Files

- Most programming languages use header files
- Header files contains elements which are reusable; for example, a function which performs string comparisons
- The following are header files used in the demo program for the RX63N development board

STANDARD INCLUDE FILE	DESCRIPTION
<config.h>	Defines macros relating to the font size.
<glyph.h>	Defines external constants, typedef enumerations and structures, prototype for minimum, and full access for glyph API library.
<preamble.h>	Defines basic definitions of all simple constants and types.
<st7579_lcd.h>	Defines prototypes that are required by the LCD driver in the glyph API.
<r_glyph.h>	Defines prototypes for the glyph communication API.
<typedefine.h>	Defines aliases of Integer Type.
<sbrk.h>	Defines macro relating to size of area managed by sbrk.
<vect.h>	Defines vector table.
<stacksct.h>	Defines macros that refer to setting of stack area.
<iodefine.h>	Define various input and output registers.

[1]

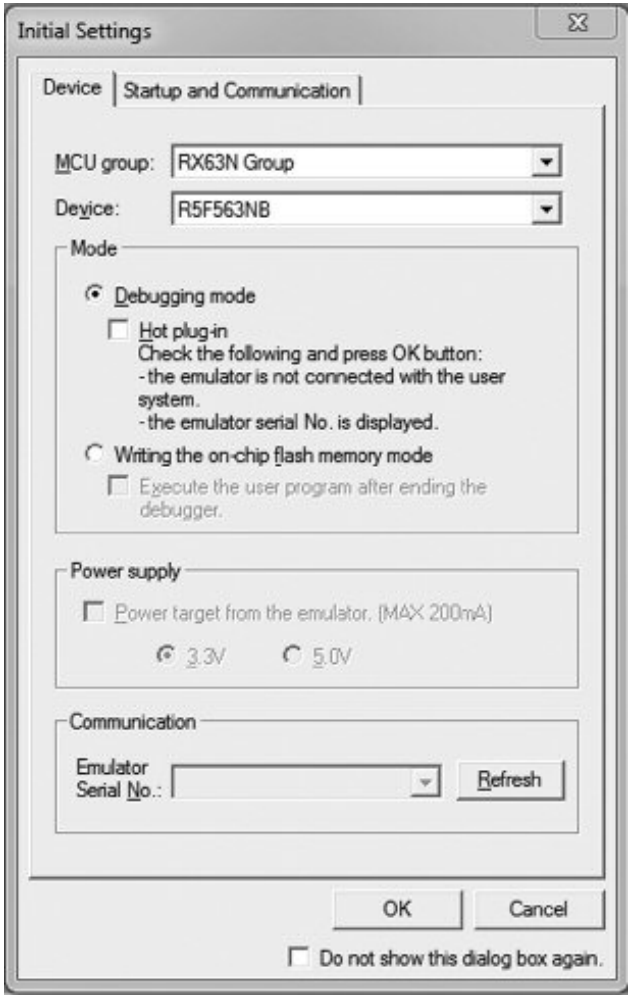
Running Your First Program

- Once a project has been created, select JLink in the standard toolbar as the current session

[1]

- Press connect on the debug toolbar

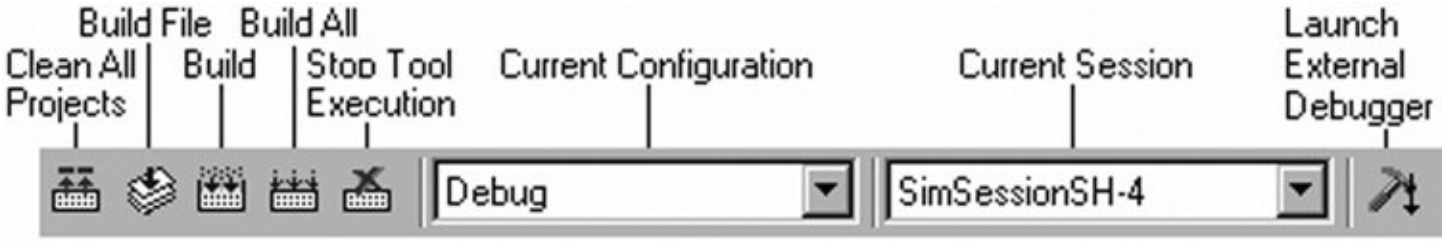
- Ensure that RX63N Group is selected under MCU group, and that the decvice is set to R5F563NB



[1]

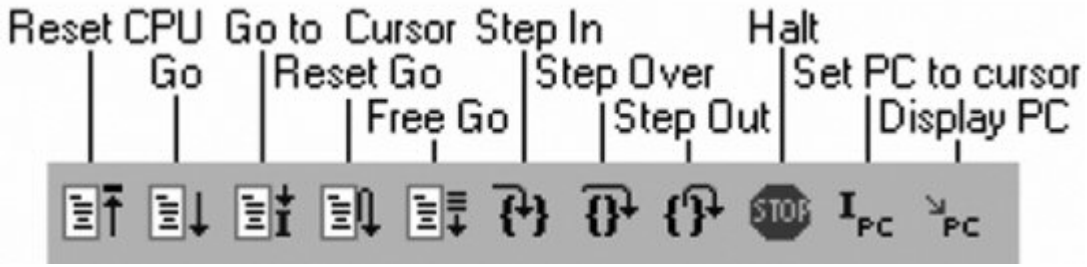
Running Your First Program (cont.)

- After connecting the board, the next step is to compile and run the program
- Click on **Build All** in the standard toolbar pictured below



[1]

- Finally click on **CPU Go** to run the program on the board



[1]

- If you, for example, ran the tutorial project, the LEDs will flash on the board
- Once you are done, click on **STOP** to stop the program

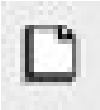
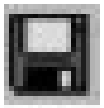
Useful Keyboard Shortcuts

- The following keyboard shortcuts are useful in writing programs

OPERATION	EFFECT	ACTION
Undo	Reverses the last editing operation.	Select [Edit → Undo]
		Press CTRL + Z
Redo	Repeats the last undone editing operation.	Select [Edit → Undo]
		Press CTRL + Z
Cut	Removes highlighted text and places it on the Windows clipboard.	Click the Cut toolbar button ( )
		Press CTRL + X
		Select [Edit → Cut]
		Select Cut from the pop-up menu
Copy	Places a copy of the highlighted text into the Windows clipboard.	Click the Copy toolbar button ( )
		Press CTRL + C
		Select [Edit → Copy]
		Select Copy from the pop-up menu
Paste	Copies the contents of the Windows® clipboard into the active window at the position of the insertion cursor.	Click the Paste toolbar button ( )
		Press CTRL + V
		Select [Edit → Paste]
		Select Paste from the pop-up menu
Clear	Removes highlighted text (it is not copied to the Windows clipboard).	Select [Edit → Clear]
		Press Delete
Select All	Selects (i.e. highlights) the entire contents of the active window.	Select [Edit → Select All]
		Press CTRL + A

[1]

Creating a New File

- To create a new file either:
  - Click the **file** tab and then **new file**
  - Press CTRL + N
  - Or click on the new file toolbar icon 
- Make sure to write a description in each of the files you create, comments in C are written in two ways:
  - /* comment goes here */
  - //comment goes here
- The descriptions
- To save your file either:
  - Click the **file** tab and then **Save**
  - Press CTRL + S
  - Or click on the new file toolbar icon 
- If you want to save a header file, for example, make sure that the file extension is .h
- If you want to save a c file, use .c as the file extension

Adding and Removing Files in the Current Project

- To add a file:
  - Click on the **project** tab in the manu bar and select **Add Files...**
  - Navigate to your file, select it, and press **Add**
- To remove a file:
  - Click on the **project** tab in the manu bar and select **Remove Files..**
  - Select the file you want to remove and click **Remove**

What we have covered

- How programs are compiled
- Debugging tools available with the RX63N Development Kit
- How to launch HEW and create a new project
- How to connect the development board and run a program
- What header files are used for
- Keyboard shortcuts useful in programming
- How to add and remove files

References

[1] Embedded Systems, An Introduction Using the Renesas RX63N Microcontroller

Renesas Electronics America Inc.