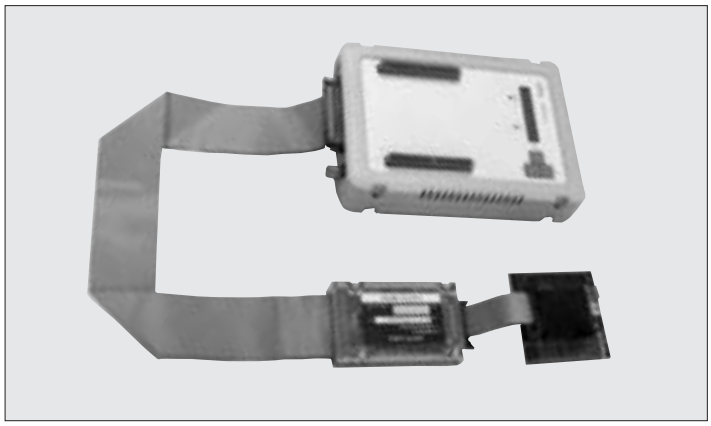
Introducing the SOFTUNE™ Workbench and New ICE with Support for F²MC -16L / 16LX / FR Family

A workspace function for simultaneously developing software for several different projects has been added to the integrated development environment SOFTUNE™ Workbench. The newly developed ICE dramatically improves the real time debugging environment.

Introduction

Software engineers are increasingly expected to develop several models simultaneously, to reduce their development time, and to improve software reliability. To help reduce their work burden, FUJITSU has prepared an environment in which efficient software development and debugging is possible.

Photo 1 External View of MB2198 (1)



Product Overview

SOFTUNE is an integrated development environment in which various software development tools — including C

Photo 2 External View of MB2198 (2)

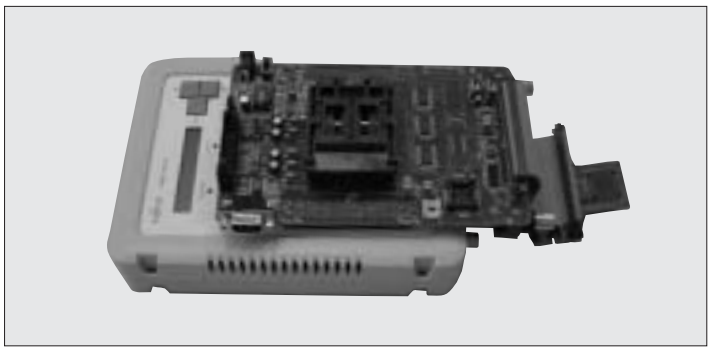
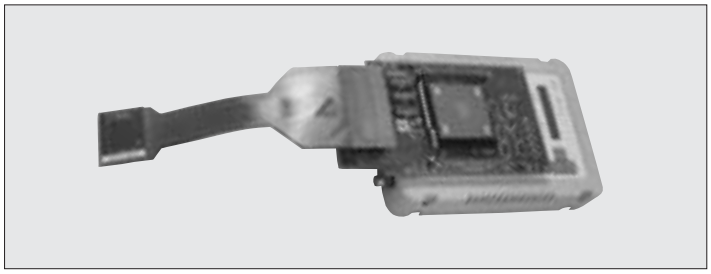


Photo 3 External View of MB2147

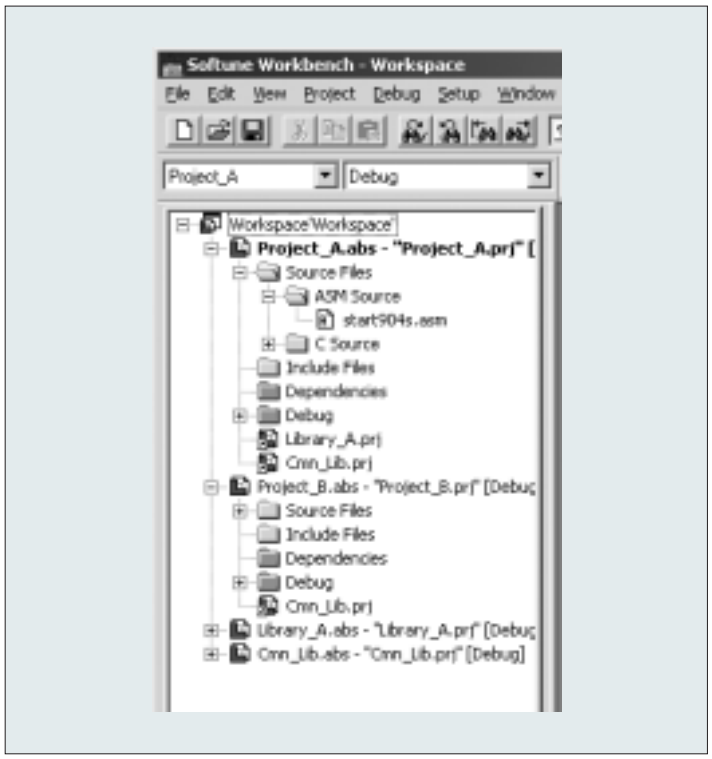


compiler, assembler, linker, and emulator debugger — exist in a single operation environment called Workbench. Anyone can easily perform the overall sequence of tasks, from program coding to compiling and debugging.

The SOFTUNE environment has been newly enhanced with a "workspace function" and an "automatic ROM/RAM checking function". The workspace function enables software development for several different projects. The automatic ROM/RAM checking function compares the size of linked ROM/RAM and domains with the memory map of development target chip and outputs a warning if there is any problem. These functions enable easy switching between projects when programs for different models are developed. Moreover, program misplacement can be detected easily because the checking function automatically checks the section size / placement domain.

In addition, the emulator debugger has been improved with an operability and a command response, making full use of the new ICE function / performance. The new ICE is small and lightweight, and the debugging function during real time execution — e.g., during real time RAM monitoring — has been enhanced.

Figure 1 SOFTUNE Workbench Screen



SOFTUNE™ Workbench

The conventional SOFTUNE Workbench made it easy to develop software at a personal level. But with the continuing expansion of program size, more and more engineers are developing several models simultaneously, or collaborating with other developers on a single model. In response to these changes, the project management function of the SOFTUNE Workbench has been enhanced to enable easy software development by several engineers (**Fig.1**).

Furthermore, the evaluation chip has built-in ROM / RAM with the largest density of the series, which can be debugged using the chip. In conventional development using built-in ROM / RAM, the size and address of the ROM / RAM must always be noted. But in the SOFTUNE evaluation chip, a linker monitors the built-in ROM / RAM domain and address of the target type by specifying the target MCU, and an error is output when there is any problem.

SOFTUNE™ Workbench Characteristics

Project Management Function

Overall Management of Several Projects (Workspace)

In conventional methods, the project file must be managed separately when several products are developed simultaneously on the same MCU. But the SOFTUNE Workbench can register several top-level projects in a single workspace, thereby enabling overall management of simultaneously developed projects.

Hierarchical Management of Projects (Sub-project)

When development is conducted by several groups or by dividing modules, the conventional method requires that the project should be divided for each work group or module. By introducing the concept of sub-projects*, the Workbench enables overall management of several projects, even when one product configuration is divided into several elements.

* *The sub-project is a project dependent on other projects and is treated as an element of a top-level project. The project type is designated as library (LIB) or relative (REL).*

Management of Several Series under a Single Project (Project Configuration)

Conventionally, a copy of the project had to be made for separate management when several products were developed with nearly the same member configuration and an option or a member was changed for purposes of temporary evaluation, etc. In this product, the introduction of several project configurations (option or member configuration required for building output files) allows easy management of a single project even when several load modules with different options or members are developed.

Hierarchical Member Management with Folders

When the number of registered members increases, only flat management is possible by the conventional methods. However, by introducing folders into member management, three-dimensional management becomes possible. Furthermore, members can be registered by the folder configuration managed by the OS when the directory is specified at the time of member registration.

Automatic Built-in ROM/RAM Domain Setup Function

With SOFTUNE tools, the target MCU being used must be specified. Although this information had been used only for optional settings or macros already defined by differences among the MCU series, an automatic setup / domain checking function on the built-in ROM/RAM has been added to this product.

Support with Linker

[Automatic Built-in ROM/RAM Domain Specification (-set_rora)]

The built-in ROM/RAM domain of the target MCU is automatically specified by a ROM domain specification (-ro) option and RAM domain specification (-ra) option.

[Domain Specification Check Other than Built-in ROM/RAM Domain (-check_rora)]

The SOFTUNE Workbench provides domain specification checking for cases in which the domains specified by the ROM domain specification (-ro) option or RAM domain specification (-ra) option are domains other than the target MCU's built-in ROM/RAM domain.

Figure 2 Automatic Setup Screen for Linker Option "Location / Link Option"

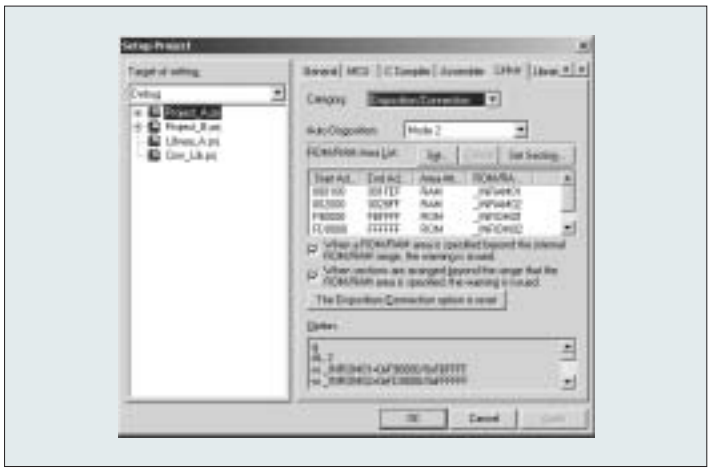


Figure 3 Automatic Map Setup for Simulator / Emulator Debugger

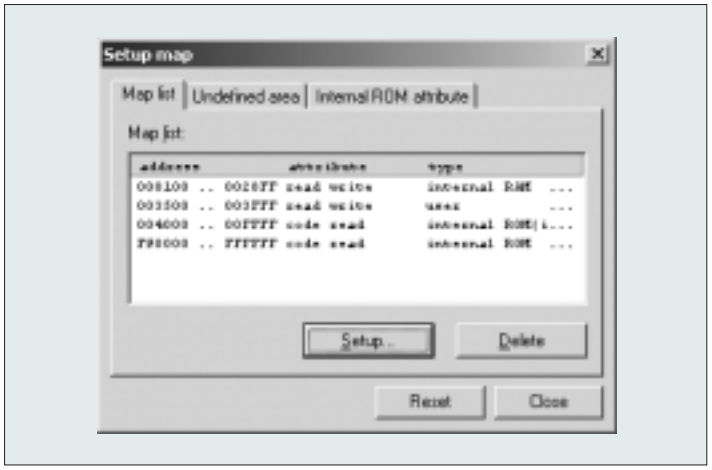


Figure 4 Configuration of ICE MB2147 for F²MC-16L / 16LX

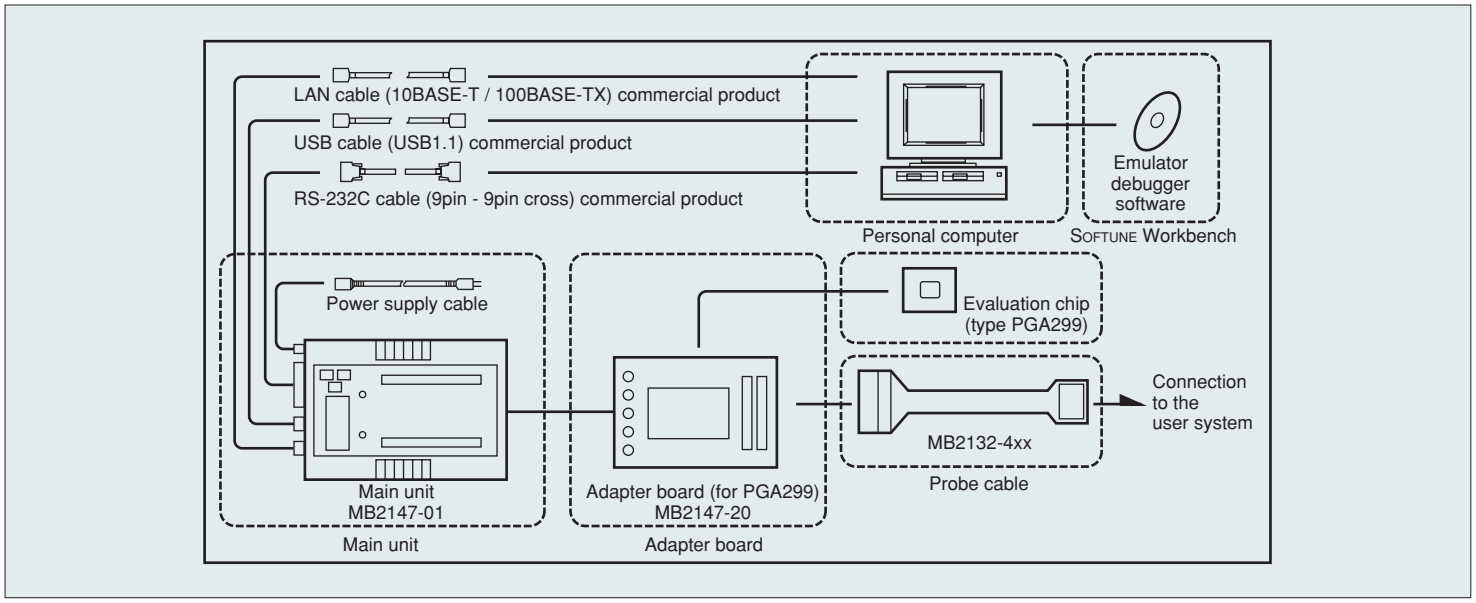


Figure 5 System Configuration of ICE MB2198 for FR

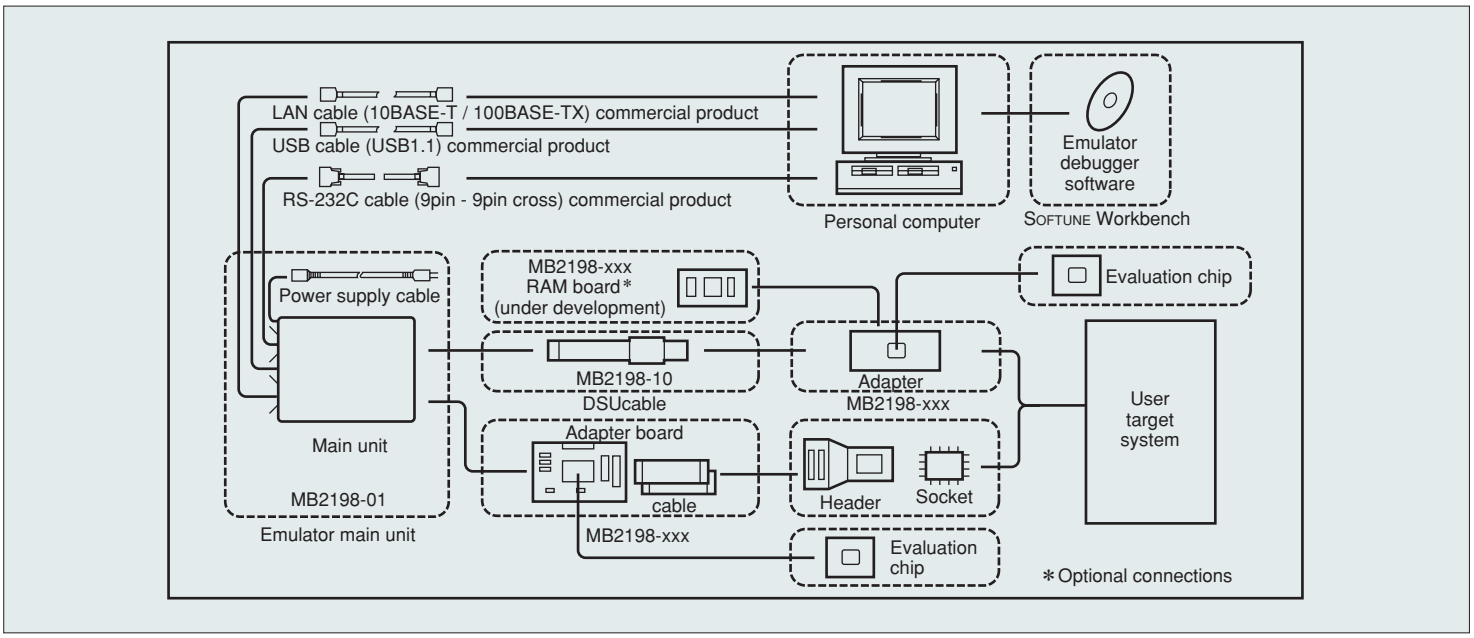
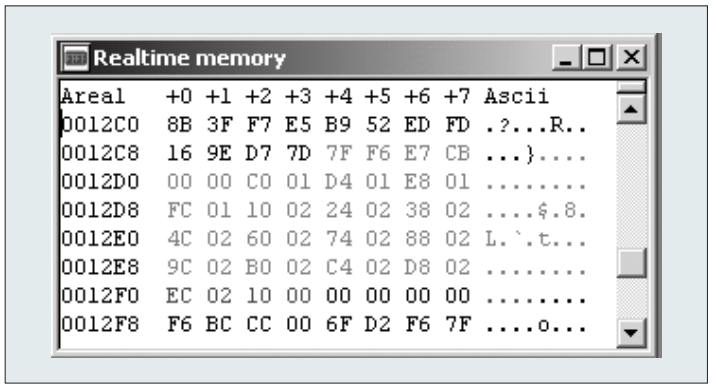


Figure 6 Real Time Memory Window



Support with Workbench

[Automatic Setup on Linker Option "Location / link option" (**Fig.2**)]

The built-in ROM / RAM domain of the target MCU is automatically specified by a ROM domain specification (-ro) option and a RAM domain specification (-ra) option.

[Automatic Map Setup for the Simulator Debugger / Emulator Debugger (**Fig.3**)]

A map is automatically set up for the target MCU's built-in ROM / RAM domain. For the simulator debugger, automatic map setup by load module can also be selected.

Though both FR / F^{2}MC-16 are supported by simulator debuggers, only F^{2}MC-16 is supported by emulator debuggers.

ICE (Emulator Debugger)

A new in-circuit emulator (called ICE hereafter) has been developed for each of F^{2}MC16-L / 16LX and FR. This ICE can be used for purposes from initial hardware debugging to software debugging under control of the SOFTUNE Workbench.

The ICE has been miniaturized to A5 size without any compromise in function. In addition, it has been accelerated and currently allows debugging at 25MHz with F^{2}MC-16L /16LX, or at 68MHz with FR. **Fig. 4** shows the configuration of ICE for F^{2}MC-16L / 16LX, and **Fig. 5** shows the configuration of ICE for FR.

In addition, improvements have been made to the command response and performance during real-time debugging.

ICE Characteristics

Acceleration of Download Speed

As programs grow larger in density, the speed of the cycle of debugging → compiling → downloading → debugging becomes increasingly important for efficient debugging. Therefore this ICE has accelerated the download speed by using USB1.1 for interface with the host computer, achieving 300K bytes/sec. for both F^{2}MC-16L / 16LX and FR.

Improvement in Command Response

If the response from ICE side processing is slow against the

user's command input, such as in step execution or display of real time tracing results, the developer is likely to become stressed and less efficient. To prevent such deterioration, this ICE features improved processing ability and command response in the ICE interface.

Memory Reference Function during MCU Execution (Real Time Monitoring Function)

For FR —The conventional method of accessing memory requires temporarily pausing the MCU, and thus the memory cannot be referenced in real time. But in this product, improvements in the ICE and evaluation chip allow memory referencing without pausing the MCU for 256 bytes×2 areas.

For F^{2}MC-16L/16LX — Since the conventional methods used cycle-stealing (a method in which memory is accessed during the empty bus cycle), loading becomes impossible when the branch commands appear successively. In this product, however, improvements in the ICE and evaluation chip allow memory referencing without pausing the MCU for 256 bytes×2 areas.

Furthermore, the real time memory window has been equipped with a monitoring function at 100 ms intervals and a function that indicates whether access is being made using the three data colors described below (**Fig. 6**).

- No access: The color for the data value is gray.
- Access in progress: The color for the data value is black.
- Data changed: The color for the data value is red.

Memory Rewriting Function during MCU Execution (On-the-Fly Function)

For FR — The combination of DSU4 and this ICE enables memory rewriting with a CPU stop-time of several ten μs.

For F^{2}MC-16L/16LX — Rewriting is possible on emulation memory without stopping the CPU.

Functions

Table 1 shows the list of debugging functions of this ICE. ✱

* SOFTUNE is a tradmark of FUJITSU LIMITED.

Table 1 List of Debugging Functions of this ICE

Function		MB2198 (New ICE for FR)	MB2147 (New ICE for F^{2}MC-16L / 16LX)
Feature	Accelerated downloading speed	○	○
	Accelerated high-level language step execution	○	○
	Response improvement in overall debugging commands	○	○
	Acceleration during 32KHz low-speed sub-clock operation	○	—
Basic Specifications	General register display / change	○	○
	Device status display	○	○
Break	Code break	○ (5 channels)	○ (Limitless)
	Code event	○ (2 channels)	○ (4 channels)
	Data event	○ (2 channels)	○ (4 channels)
	Sequencer	○ (2 levels)	○ (3 levels)
	Path count (for each level)	○	○
	Guarded access break	○ (Domain mask used)	○ (MAP function used)
Simple Trace (DSU4 I/F)	Full trace mode	○ (256K clock)	—
	Real time trace mode	○ (256K clock)	—
	Internal trace mode	○ (128 step)	—
	Display during execution	×	—
External Tracing (Real Time Tracing I/F)	External trace mode	○ (64K step)	○ (64K step)
Common for Tracing	Time stamp	○	○
	DMAC access	×	×
	External bus trace	×	○
Memory Reference	During break	○	○
	Real time memory display (256 bytes×2 areas)	○	○
	During execution	○ (Pseudo on-the-fly)	○ (Emulation memory)
Memory Change	During break	○	○
	During execution	○ (Pseudo on-the-fly)	○ (Emulation memory)
Analysis	Execution start → completion, cycle and time measurement	○	○