RL78/G13

Self-Programming (Received Data via UART)

R01AN1356EJ0100
Rev. 1.00
Mar.01, 2013

Introduction

This application note gives the outline of flash memory reprogramming using a self-programming technique. In this application note, flash memory is reprogrammed using the flash memory self-programming library Type01.

The sample program described in this application note limits the target of reprogramming to the boot area. For details on the procedures for performing self-programming and for reprogramming the entire area of code flash memory, refer to RL78/G13 Microcontroller Flash Memory Self-Programming Execution (R01AN0718E) Application Note.

Target Device

RL78/G13

When applying the sample program covered in this application note to another microcomputer, modify the program according to the specifications for the target microcomputer and conduct an extensive evaluation of the modified program.

Contents

1. Specifications

This application note explains a sample program that performs flash memory reprogramming using a self-programming library.

The sample program displays the information about the current version of the library on the LCD. Subsequently, the program receives data (reprogramming data) from the sending side and, after turning on the LED indicating that it is accessing flash memory, carries out self-programming to rewrite the code flash memory with the reprogramming data. When the rewrite is completed, the sample program turns off the LED and displays the information about the new version on the LCD.

Table 1.1 lists the peripheral functions to be used and their uses.

Table 1.1 Peripheral Functions to be Used and their Uses

Peripheral Function	Use
Channel 2 of serial array unit 0	Receives data via UART.
Channel 3 of serial array unit 0	Sends data via UART.
Port I/O	Displays text on the LCD. Turns on and off the LED.

1.1 Outline of the Flash Memory Self-Programming Library

The flash memory self-programming library is a software product that is used to reprogram the data in the code flash memory using the firmware installed on the RL78 microcontroller.

The contents of the code flash memory can be reprogrammed by calling the flash memory self-programming library from a user program.

To do flash memory self-programming, it is necessary for the user program to perform initialization for flash memory self -programming and to execute the C or assembler functions that correspond to the library functions to be used.

1.2 Code Flash Memory

The configuration of the RL78/G13 (R5F100LE) code flash memory is shown below.

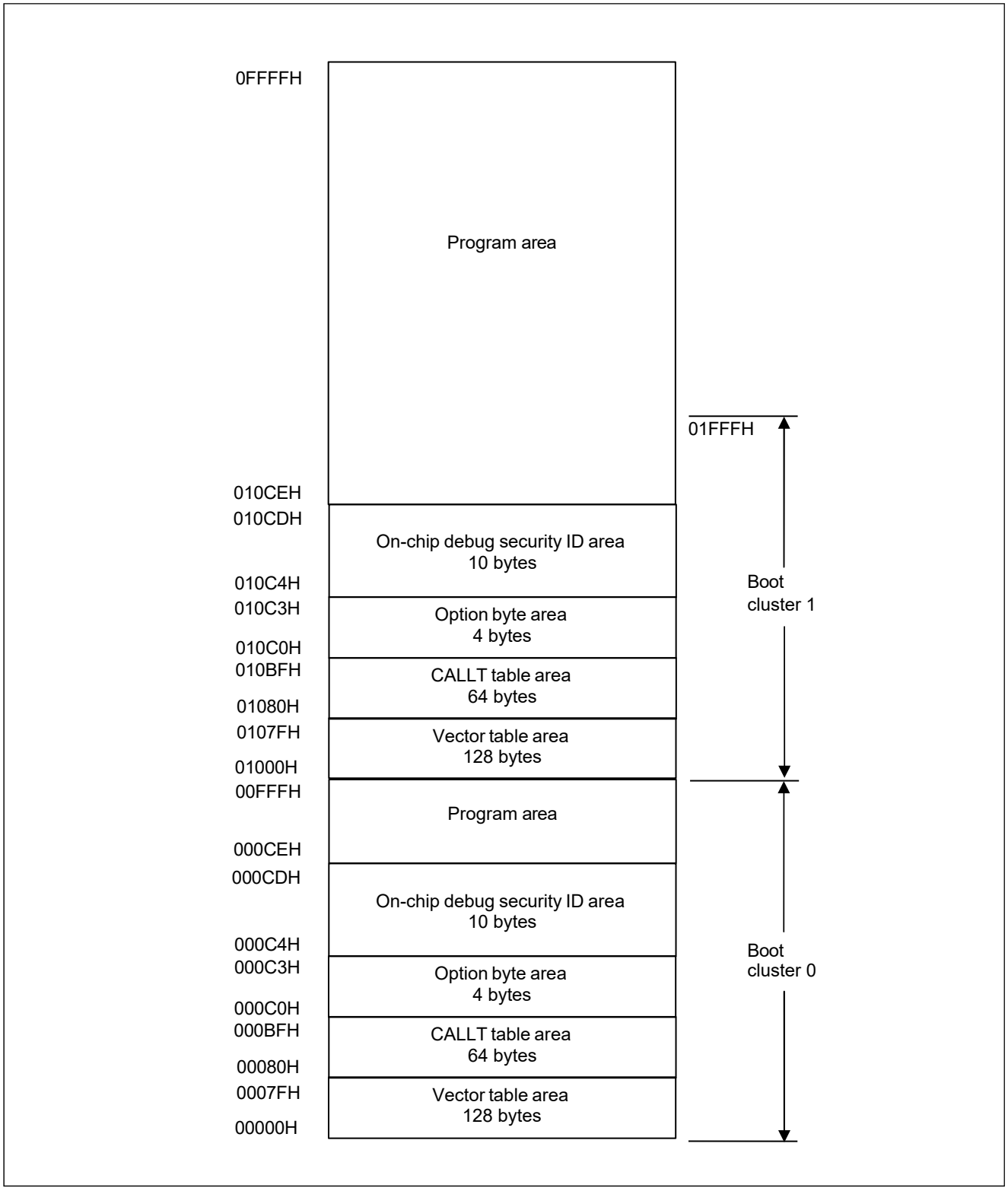


Figure 1.1 Code Flash Memory Configuration

Caution: When the boot swap function is used, the option byte area (000C0H to 000C3H) in boot cluster 0 is swapped with the option byte area (010C0H to 010C3H) in boot cluster 1. Accordingly, place the same values in the area (010C0H to 010C3H) as those in the area (000C0H to 000C3H) when using the boot swap function.

The features of the RL78/G13 code flash memory are summarized below.

Table 1.2 Features of the Code Flash Memory

Item	Description
Minimum unit of erasure and verification	1 block (1024 bytes)
Minimum unit of programming	1 word (4 bytes)
Security functions	Block erasure, programming, and boot area reprogramming protection are supported. (They are enabled at shipment)
	It is possible to disable reprogramming and erasure outside the specified window only at flash memory self-programming time using the flash shield window.
	Security settings programmable using the flash memory self-programming library

Caution: The boot area reprogramming protection setting and the security settings for outside the flash shield window are disabled during flash memory self-programming.

1.3 Flash Memory Self-Programming

The RL78/G13 is provided with a library for flash memory self-programming. Flash memory self-programming is accomplished by calling functions of the flash memory self-programming library from the reprogramming program.

The flash memory self-programming library for the RL78/G13 controls flash memory reprogramming using a sequencer (a dedicated circuit for controlling flash memory). The code flash memory cannot be referenced while control by the sequencer is in progress. When the user program needs to be run while the sequencer control is in progress, therefore, it is necessary to relocate part of the segments for the flash memory self-programming library and the reprogramming program in RAM when erasing or reprogramming the code flash memory or making settings for the security flags. If there is no need to run the user program while the sequencer control is in progress, it is possible to keep the flash memory self-programming library and reprogramming program on ROM (code flash memory) for execution.

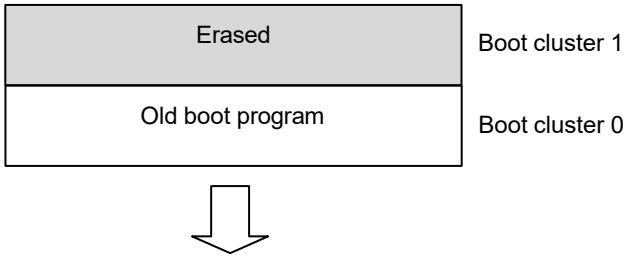
1.3.1 Boot Swap Function

When reprogramming of the area where vector table data, the basic functions of the program, and flash memory self-programming library are allocated fails due to a temporary power blackout or a reset caused by an external factor, the data that is being reprogrammed will be corrupted, as the result of which the restarting of the user program or reprogramming cannot be accomplished when a reset is subsequently performed. This problem is be avoided by the introduction of the boot swap function.

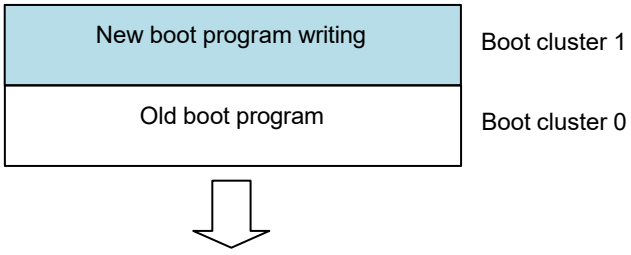
The boot swap function swaps between boot cluster 0 which is the boot program area and boot cluster 1 which is the target of boot swapping. A new program is written into boot cluster 1 before reprogramming is attempted. This boot cluster 1 is swapped with boot cluster 0 and boot cluster 1 is designated as the boot program area. In this configuration, even when a temporary power blackout occurs while the boot program area is being reprogrammed, the system boot will start at boot cluster 1 on the next reset start, thus ensuring the normal execution of the programs.

The outline image of boot swapping is shown in the figure below.

(1) Erasing boot cluster 1
Call the FSL_Erase function to erase boot cluster 1 (blocks 4 to 7).

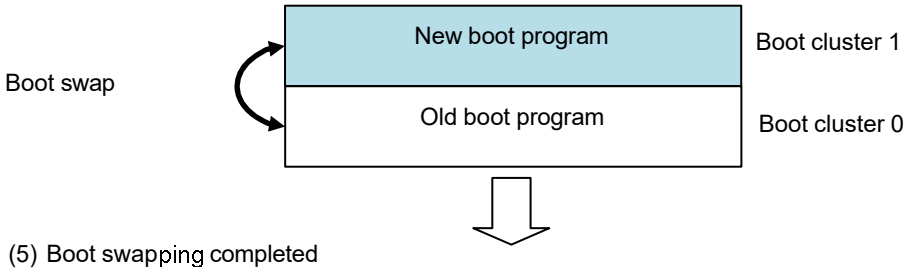


(2) Writing the new boot program into boot cluster 1
Call the FSL_Write function to write the new boot program into boot cluster 1 and call the FSL_IVerify function to verify boot cluster 1.
The steps that have been performed up to here ensure that the programs will run normally even when the programming of the new boot program fails due to a temporary power blackout or reset because the system boot is started by the old boot program.



(3) Setting the boot swap bit
Call the FSL_InvertBootFlag function to invert the state of the boot flag.
When a temporary power blackout or reset occurs after the state of the boot flag is inverted, the programs will run normally because the system boot is started by the new boot program whose reprogramming has been completed.

(4) When a reset occurs
When a reset occurs, boot clusters 0 and 1 are swapped.



(5) Boot swapping completed

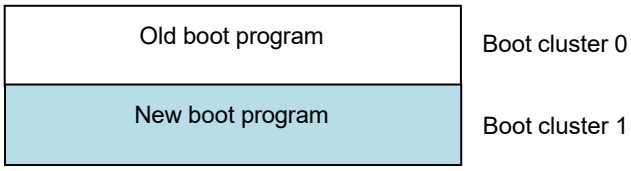


Figure 1.2 Outline of Boot Swapping

1.3.2 Flash Memory Reprogramming

This subsection describes the outline image of reprogramming using the flash memory self-programming technique. The program that performs flash memory self-programming is placed in boot cluster 0.

The sample program covered in this application note limits the target of reprogramming to the boot area. For details on the procedures for perform self-programming and for reprogramming the entire area of code flash memory, refer to RL78/G13 Microcontroller Flash Memory Self-Programming Execution (R01AN0718E) Application Note.

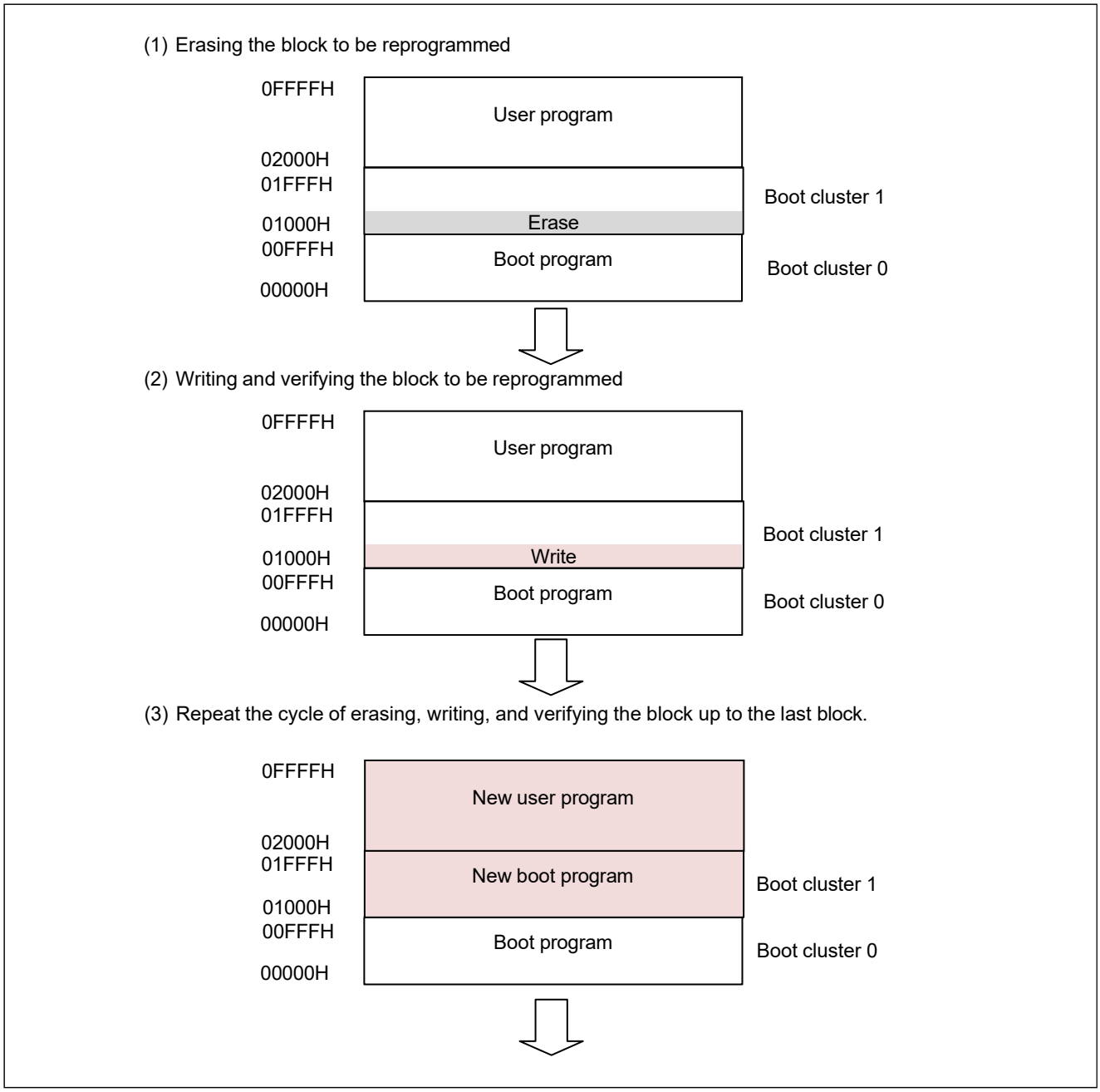


Figure 1.3 Outline of Flash Memory Reprogramming (1/2)

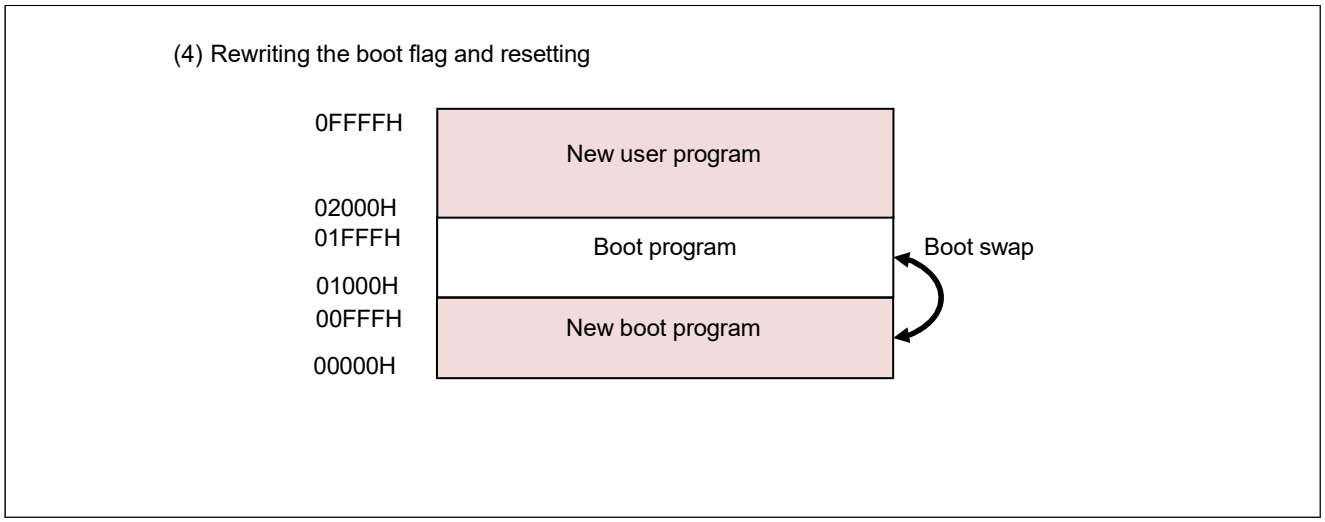


Figure 1.4 Outline of Flash Memory Reprogramming (2/2)

1.3.3 Flash Shield Window

The flash shield window is one of security mechanisms used for flash memory self-programming. It disables the write and erase operations on the areas outside the designated window only during flash memory self-programming.

The figure below shows the outline image of the flash shield window on the area of which the start block is 08H and the end block is 1FH.

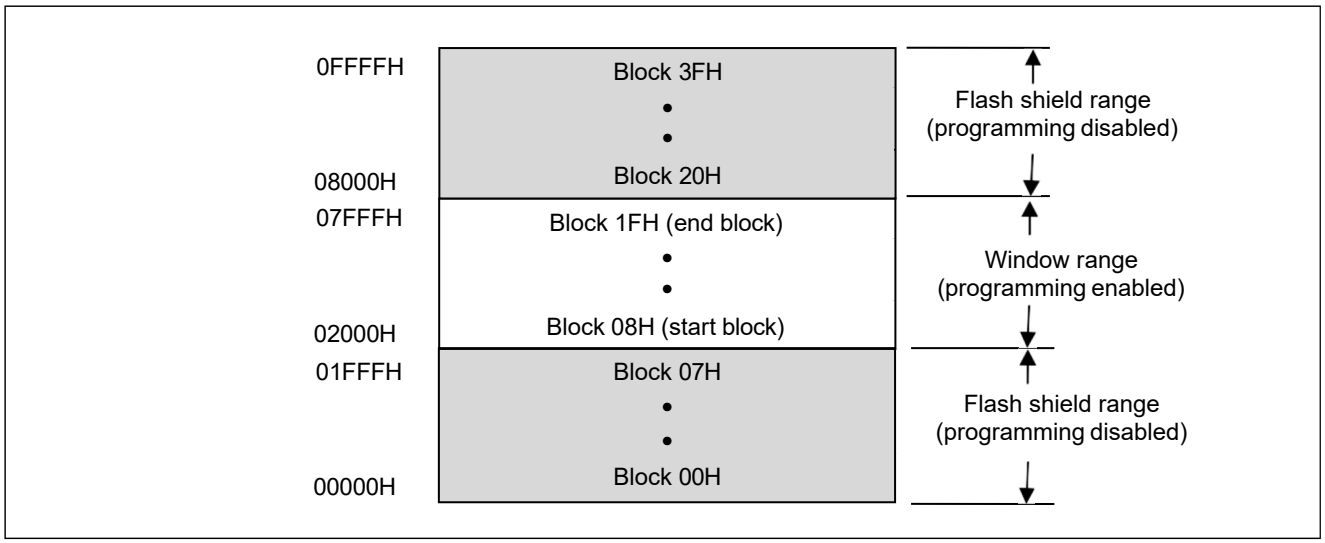


Figure 1.5 Outline of the Flash Shield Window

1.4 How to Get the Flash Memory Self-Programming Library

The flash memory self-programming library can be obtained from the following URL:

htt /products/tools/flash_prom_programming/flash_libraries/index.jsp

2. Operation Check Conditions

The sample code described in this application note has been checked under the conditions listed in the table below.

Table 2.1 Operation Check Conditions

Item	Description
Microcontroller used	RL78/G13 (R5F100LEA)
Operating frequency	• High-speed on-chip oscillator (HOCO) clock: 32 MHz • CPU/peripheral hardware clock: 32 MHz
Operating voltage	5.0 V (Operation is possible over a voltage range of 2.9 V to 5.5 V.) LVD operation (V_{LVI}): Reset mode which uses 2.81 V (2.76 V to 2.87 V)
Integrated development environment	Cube Suite+ V1.02.01 from Renesas Electronics Corp.
C compiler	CA78K0R V1.41 from Renesas Electronics Corp.
Board to be used	Renesas Starter Kit for RL78/G13 (R0K50100LS000BE)
Flash memory self-programming library (Type, Ver)	FSLRL78 Type01, Ver 2.10 Note

Note: Use and evaluate the latest version.

3. Related Application Notes

The application notes that are related to this application note are listed below for reference.

- RL78/G13 Initialization (R01AN0451E) Application Note
- RL78/G13 Serial Array Unit (UART Communication) (R01AN0459E) Application Note
- RL78 Microcontroller Flash Memory Self-Programming Library Type01 (R01AN0350E) Application Note

4. Description of the Hardware

4.1 Hardware Configuration Example

Figure 4.1 shows an example of the hardware configuration used for this application note.

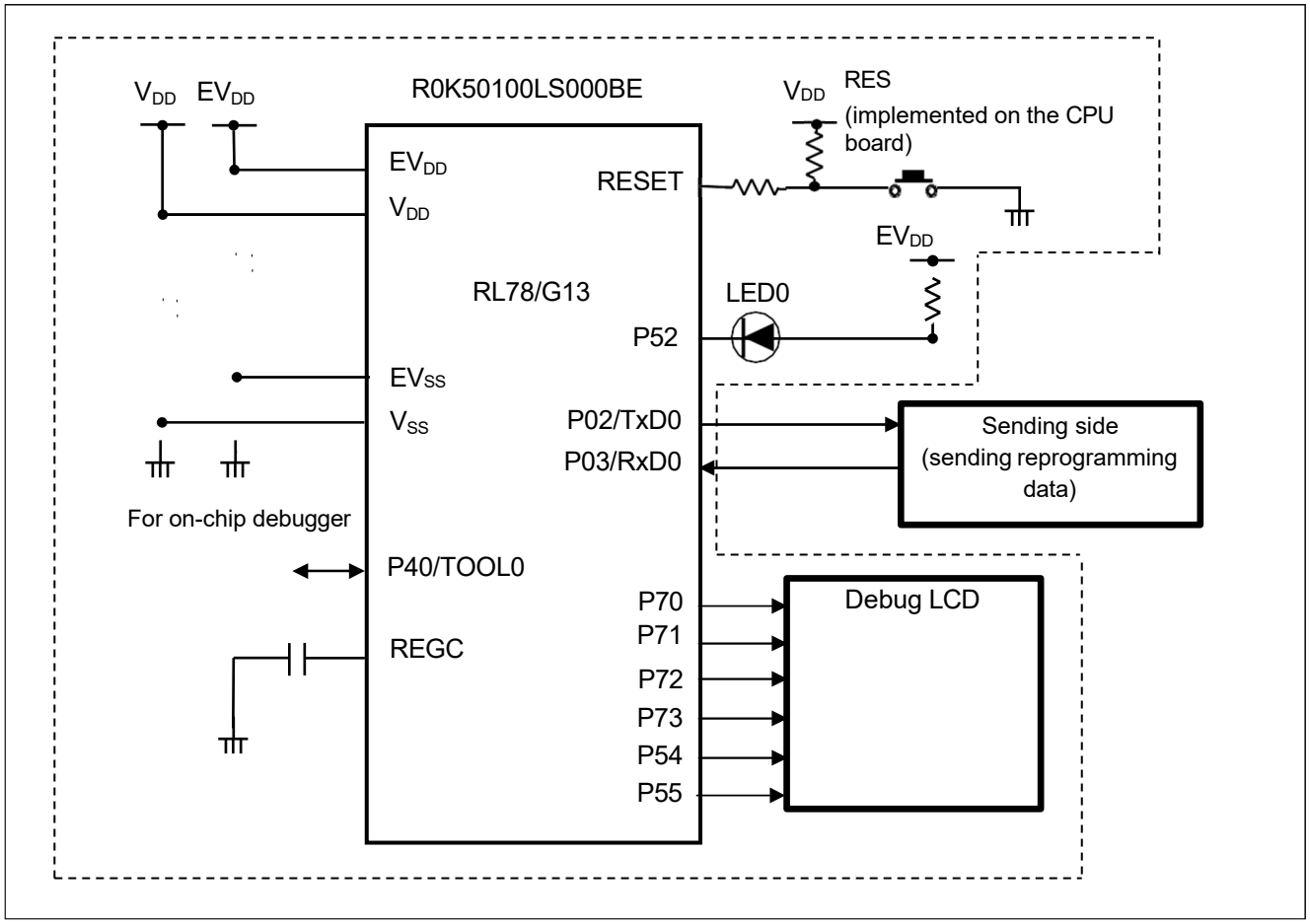


Figure 4.1 Hardware Configuration

Cautions:
1. The purpose of this circuit is only to provide the connection outline and the circuit is simplified accordingly. When designing and implementing an actual circuit, provide proper pin treatment and make sure that the hardware's electrical specifications are met (connect the input-only ports separately to V_{DD} or V_{SS} via a resistor).
2. V_{DD} must be held at not lower than the reset release voltage (V_{LVI}) that is specified as LVD.