RL78 Family

RL78 Flash Programmer (RL78 Protocol A)

Introduction

This application note describes how to write the program to the internal flash memory of the RL78 microcontroller that supports the RL78 Protocol A.

Target Device

RL78/G23

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.

Related Documents

Documents related to this application note are listed below, refer to the following documents as well.

- RL78 Microcontrollers RL78 Microcontrollers (RL78 Protocol A) Programmer Edition (R01AN0815)
- RL78 Family Renesas Flash Driver RL78 Type 01 User's Manual (R20UT4830)
- RL78 Family Renesas Flash Driver RL78 Type 01 SC version (Code Flash) (R20AN0653)
- RL78 Family Renesas Flash Driver RL78 Type 01 SC version (Data Flash) (R20AN0654)
- RL78/G23-128p Fast Prototyping Board User's Manual (R20UT4870)

Contents

1. Overview

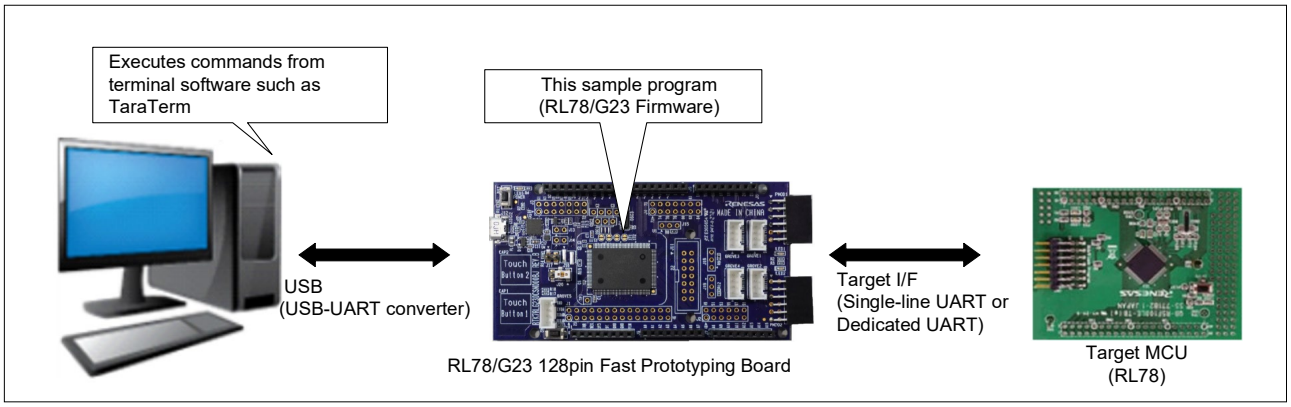
This sample program is the firmware for writing the program to the flash memory in the RL78 microcontroller. The sample program has the following features.

- The writing target RL78 microcontroller (target MCU) conforms to the RL78 Protocol A.
- Serial programming of the RL78 Protocol A is used for writing.
- The program file (write data) conforms to the Motorola S-format.
- The sample program supports “online mode” for writing with a host PC and “offline mode” for writing without a host PC.
- The sample program realizes the flash programmer in combination with the evaluation board (RL78/G23-128p Fast Prototyping Board) containing RL78/G23.

1.1 Online Mode

In online mode, the flash memory in the target MCU is written by commands from the host PC through a USB connection between the host PC and the evaluation board (RL78/G23-128p Fast Prototyping Board).

Figure 1-1 Online mode



1.2 Offline Mode

In offline mode, the flash memory in the target MCU is written by pushing the switch (SW1) on the evaluation board (RL78/G23-128p Fast Prototyping Board).

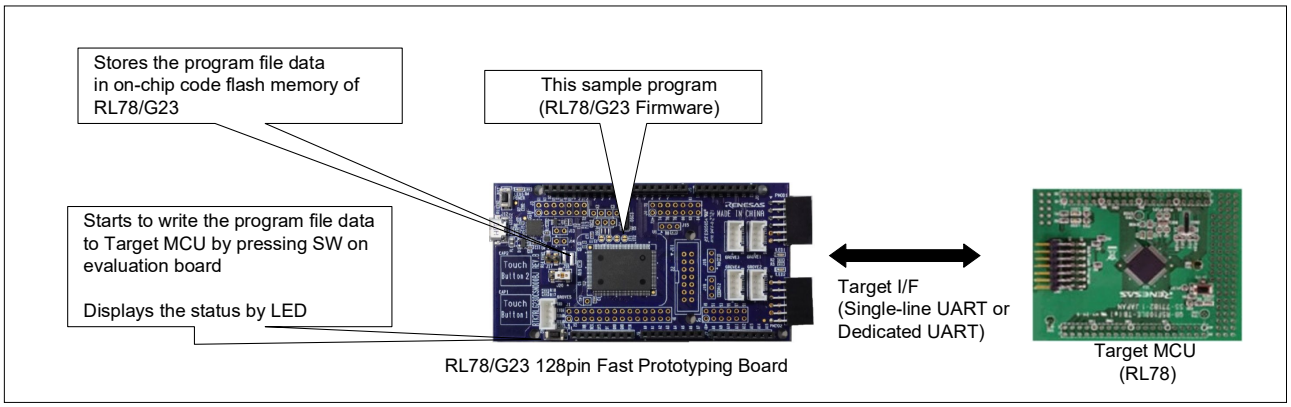
Before writing, make a connection between the host PC and the evaluation board through a USB, and then download the settings and program file by commands.

The program file data is stored in the write data section in the RL78/G23 code flash memory.

Offline mode can accept a code flash memory size of up to 512 KB.

To write more than 512 KB of program file to the code flash memory, use online mode.

Figure 1-2 Offline mode



2. Development Environments

The operation of the sample program provided with this application note has been tested under the following conditions.

Table 2-1 Operation Confirmation Conditions

Item		Description
MCU		Part No.: R7F100GSN2DFB (Internal memory: Code Flash 768KB, RAM 48KB, Data Flash 8KB)
Evaluation board (Programmer board)		RL78/G23-128p Fast Prototyping Board Part Np.: RTK7RLG230CSN000BJ Note: Set VDD to 1.8 V or higher because this sample program runs in HS mode (high-speed on-chip oscillator frequency: 32 MHz).
Emulator		E2 emulator Lite or E2 emulator is required to debug the sample program. The USB on the RL78/G23-128p Fast Prototyping Board is used for communication between the host PC and RL78/G23. Therefore, it cannot be used for debugging (RL78 COM port debug tool).
CS+	Integrated development environment	Renesas Electronics Corp. CS+ for CC V8.09.00
	C compiler	Renesas Electronics Corp. CC-RL V1.12.00
e2studio	Integrated development environment	Renesas Electronics Corp. e2studio V2023-01
	C compiler	Renesas Electronics Corp. CC-RL V1.12.00
IAR	Integrated development environment	IAR Systems Corp. IAR Embedded Workbench for Renesas RL78 V4.21
	C compiler	IAR Systems Corp. IAR C/C++ Compiler for Renesas RL78 V 4.21.1
Flash driver		Renesas Electronics Corp. Renesas Flash Driver RL78 Type01

2.1 Renesas Flash Driver RL78 Type01

Renesas Flash Driver RL78 Type 01 (hereafter called RFD RL78 Type 01) is software for reprogramming the flash memory in the RL78/G2x

For details, refer to Renesas Flash Driver RL78 Type01 Use's manual (R20UT4830).

Renesas Flash Driver RL78 Type01 (RFD) can be downloaded from the following URL.

https://www.renesas.com/document/scd/renesas-flash-driver-rl78-type-01-rl78g23

Remark

There are RFD using Smart Configurator (SC) version and RFD not using SC version (The above URL). RFD using SC version is included in this sample program.

3. External Specification

3.1 Host Communication Specifications

This section describes the specifications for communication between the host PC and the programmer board.

Connect the USB connector (for USB-UART conversion) on the programmer board (RL78/G23-128p Fast Prototyping Board) to the host PC.

This sample program works based on the following settings for communication between the host PC and the programmer board.

Table 3-1 Host communication settings

Item	Setting
Data length (bit)	8
Data transfer direction	LSB first
Parity bit	No parity
Communication speed (bps)	115,200
Stop bit (bit)	1
Flow control	Software (Xon/Xoff)

3.1.1 Specifications Common to All Commands

Command communication is performed according to the following specifications.

- The programmer board sends a command prompt '>' to the host PC after a preparation for reception is completed.
- The programmer board sends a command input echo back.
- Send a linefeed character (CRLF:0x0D,0x0A) from the host PC when executing a command.
- The programmer board sends "PASS" when command execution is successful.
- The programmer board sends "ERROR:XX" when command execution fails.
  XX is expressed as a 2-digit hexadecimal number. For details, see Table 3-7 and Table 3-8.

The following commands of the host PC are available for this sample program.

Table 3-2 List of Host PC Commands

Command	Description
setup	Used for settings for communication mode and communication speed (for communication with the target MCU), VDD voltage, security ID code, and execution of verification when using offline mode.
lod	Used to store the program file (Motorola S-format) used in offline mode in the programmer board.
ep	Used for erasing and writing data in online mode. Execution of verification and checksum can be optionally added.

3.1.2 setup Command

This command makes operation settings of the programmer board.

This command enables settings, including communication mode and communication speed (for communication with the target MCU), VDD voltage, and execution of verification in offline mode.

This command must be executed in advance before the program file is written to the target MCU. The settings made by this command are stored in the data flash memory of RL78/G23 on the programmer board.

Table 3-3 shows the details of the options of the setup command. Figure 3-1 shows an example of usage of the setup command.

Table 3-3 setup Command Options

Option	Setting	Description
-if	uart1	uart1: Performs communication with the target MCU using a single-line UART (TOOL0).
	uart2	uart2: Performs communication with the target MCU using a dedicated UART (TOOL0, TOOLTxD, TOOLRxD). When omitted: It is the same result as when the uart1 is specified.
-speed	115200	Specifies the transmission rate (bps) set by the Baud Rate Set command of the RL78 Protocol A. When omitted: It is the same result as when the 115200 is specified.
	250000	
	500000	
	1000000	
-vdd	x.x (1-digit decimal integer, first decimal place)	Specifies the VDD voltage (V) set by the Baud Rate Set command of the RL78 Protocol A. Set the VDD voltage supplied to the programmer board and the target MCU. When omitted: It is the same result as when the 3.3 is specified.
-v		Verification is additionally executed in offline mode. This option is ignored in online mode. Use the option of the ep command when adding verification in online mode.

Figure 3-1 Terminal Software Screen: Usage example of setup command

```
> setup -if uart1 -speed 115200 -vdd 3.3 -v
PASS
>
```

3.1.3 lod Command

This command stores the program file data in the data write section of the code flash memory in the RL78/G23 on the programmer board for offline mode. For details of section, see 5.4 Section Settings.

The lod command must be executed before writing data to the target MCU in offline mode.

After the lod command is sent, send a Motorola S-format file after the message "Please send a motorola-s file." is received from the programmer board. The data record of Motorola S-format file must be sorted in ascending order of address.

When sending a file using terminal software such as TeraTerm, send it in binary mode.

The lod command can store data in the following range of the code flash memory and data flash memory.

- Code flash memory: Up to 512 kB (0x00000 - 0x7FFFF)
- Data flash memory: Up to 8 kB (0xF1000 - 0xF2FFF)

To write data beyond the above range, use online mode (ep command).

Figure 3-2 shows an example of usage of the lod command.

Figure 3-2 Terminal Software Screen: Usage example of lod command

```
> lod

Please send a motorola-s file.

<Sends a Motorola S-format file>
PASS
>
```

3.1.4 ep Command

This command erases and writes data to the target MCU in online mode while receiving a Motorola S-format file from the host PC.

Verification and checksum can be optionally added, performing erase all, write and verify alternately, and checksum in this order.

The flash memory erasing range covers the entire area of the code flash and data flash memories.

The flash memory write/verify range covers the area including the transmitted Motorola S-format data.

The write/verify operation is made in block units of the internal flash. When data is less than a block size, the area without data is filled with 0xFF.

The flash memory checksum range covers the entire area of the code flash and data flash memories.

After the ep command is sent, send a Motorola S-format file after the message "Please send a motorola-s file." and “Connect” are received from the programmer board. When sending a file using terminal software such as TeraTerm, send it in binary mode.

The RL78/G23 on the programmer board performs communication setting processing with the target MCU while receiving the file data from the host PC. The RL78/G23 on the programmer board sends “PASS” to the host PC when communication setting processing with the target MCU is done. After that, erasing the flash memory of the target MCU is executed. The RL78/G23 on the programmer board sends “PASS” to the host PC when erasure is done. Following that, writing the data to the flash memory of the target MCU is executed. The RL78/G23 on the programmer board sends “PASS” to the host PC when writing is done.

When the option for verification is specified, writing and verification are executed alternately. The RL78/G23 on the programmer board sends “PASS” to the host PC when writing and verification are done.

When the option for checksum is specified, checksum is executed after writing (including verification). The RL78/G23 on the programmer board sends the checksum value of the code flash and data flash received from the target MCU to the host PC.

Table 3-4 describes the options of the ep command. Figure 3-3 shows an example of usage of the ep command.

Table 3-4 ep command options

Option	Description
-v	With this option specified, verification is additionally executed in online mode.
-s	With this option is specified, checksum is additionally executed in online mode.

Figure 3-3 Terminal Software Screen: Usage example of ep command

```
> ep
Please send a motorola-s file.

Connect
<Sends a Motorola S-format file>
PASS
Erase
PASS
Program
PASS
> ep -v -s
Please send a motorola-s file.

Connect
<Sends a Motorola S-format file>
PASS
Erase
PASS
Program, Verify
PASS
Checksum
Code Flash: 0x1234
Data Flash: 0xABCD
PASS
>
```

3.2 Writing in Online Mode

Writing the program file data to the target MCU is started by execution of ep command.

The flow of writing is shown below.

① Execute the setup command.

② Execute the ep command.

3.3 Writing in Offline Mode

Push the switch (SW1) on the programmer board (RL78/G23-128p Fast Prototyping Board) to start erasing, writing, and verifying data without using a host PC.

The flow of writing is shown below.

① Execute the setup command.

② Execute the lod command.

③ Push the “SW1” switch.

Note. ① and ② are executed with the programmer board and the host PC connected.

3.3.1 Executing Writing

Push the switch (SW1) to start erasing, writing, and verifying data. For execution of verification, the -v option is required when the setup command is executed.

Table 3-5 Flash Memory Operating Range in Offline Mode

Operation	Operating Range
Erasure	Entire area of the code flash memory and data flash memory
Writing	Entire area of the code flash memory and data flash memory A value of 0xFF is written to areas where no data is present in the program file data of Motorola S-format.
Verification	Entire area of the code flash memory and data flash memory Areas where no data is present in the program file data of Motorola S-format are regarded as 0xFF.

3.4 LED Display Specifications

Table 3-6 below shows the relationship between LED display and programmer board operating state.

Table 3-6 LED Display and Operating State

LED Display		Operating Status
LED1 (Indicator of Single-line UART / Dedicated UART)	Lighting-off	UART is stopped.
	Lighting-on	UART is operating in Single UART
	Blinking	UART is operating in Dedicated UART
LED2 (Operation indicator)	Lighting-off	Rewriting is stopped.
	Lighting-on	Rewriting is operating.
	Blinking	Error has occurred.

3.5 Error Code Specifications

If command execution fails, the programmer board sends an error message (ERROR:XX) to the host PC. XX means the 2-digit hexadecimal error code.

Table 3-7 and Table 3-8 show the list of error codes.

Table 3-7 List of Error Codes (1/2)

Error Code (Hexadecimal)	Description
04	Command number error This error occurs when a command number error of the RL78 Protocol A status code is received from the target MCU.
05	Parameter error This error occurs when a parameter error of the RL78 Protocol A status code is received from the target MCU.
07	Checksum error This error occurs when a checksum error of the RL78 Protocol A status code is received from the target MCU.
0F	Verification error This error occurs when a verification error of the RL78 Protocol A status code is received from the target MCU.
10	Protection error This error occurs when a protection error of the RL78 Protocol A status code is received from the target MCU.
15	NACK This error occurs when a NACK of the RL78 Protocol A status code is received from the target MCU.
1A	Erasure error This error occurs when an erasure error of the RL78 Protocol A status code is received from the target MCU.
1B	Blank error This error occurs when a blank error of the RL78 Protocol A status code is received from the target MCU.
1C	Write error This error occurs when a write error of the RL78 Protocol A status code is received from the target MCU.

Table 3-8 List of Error Code (2/2)

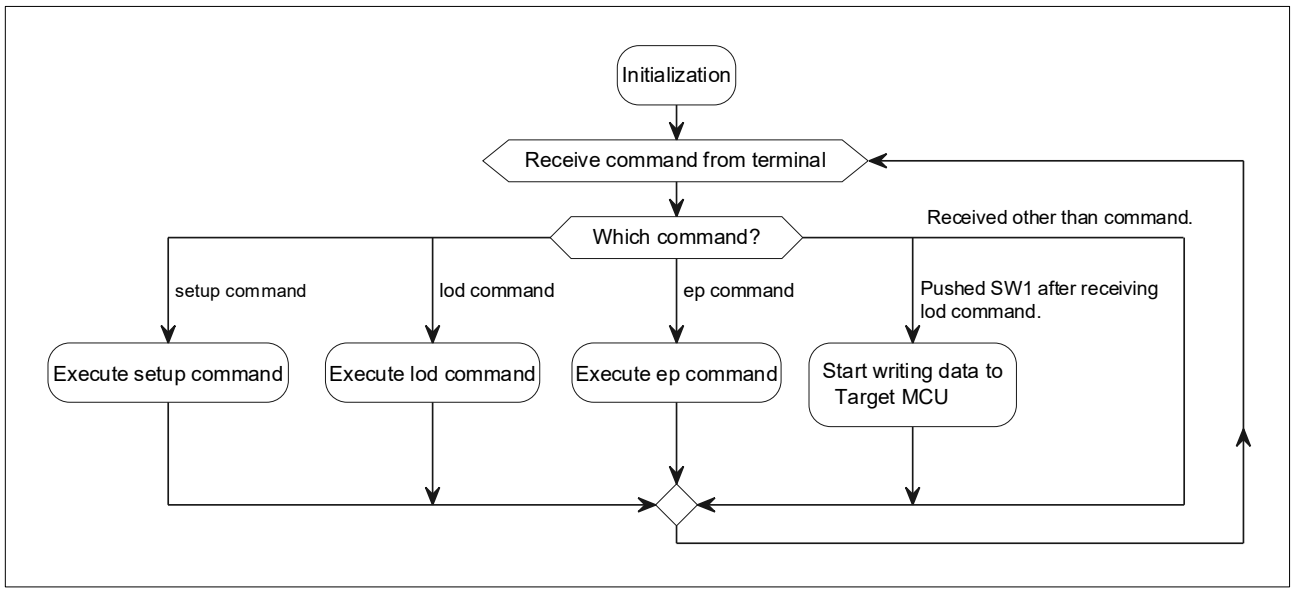
Error Code (Hexadecimal)	Description
F9	Incorrect command or parameter data This error occurs when an incorrect command or parameter data is received from the host PC.
FA	Buffer overrun error This error occurs when the data is received from the host PC and the buffer overruns. Confirm that the flow control with the host PC is set to Software (Xon/Xoff).
FB	Invalid Motorola S-format data This error occurs if Motorola S-format data sent by the lod command or ep command is invalid. This error occurs even when Motorola S-format data is not in ascending order of address.
FC	Target MCU communication timeout This error occurs if a timeout occurs during communication between the programmer board and the target MCU.
FD	Invalid address of Motorola S-format file This error occurs if an out-of-range address is included in the Motorola S-format file received by the lod command.
FE	Command communication data error This error occurs if an invalid packet format is received from the target MCU.
FF	System error This error occurs if the program does not work correctly.

3.6 Flowcharts

3.6.1 Main Loop (main function)

Figure 3-4 shows the operation of main loop.

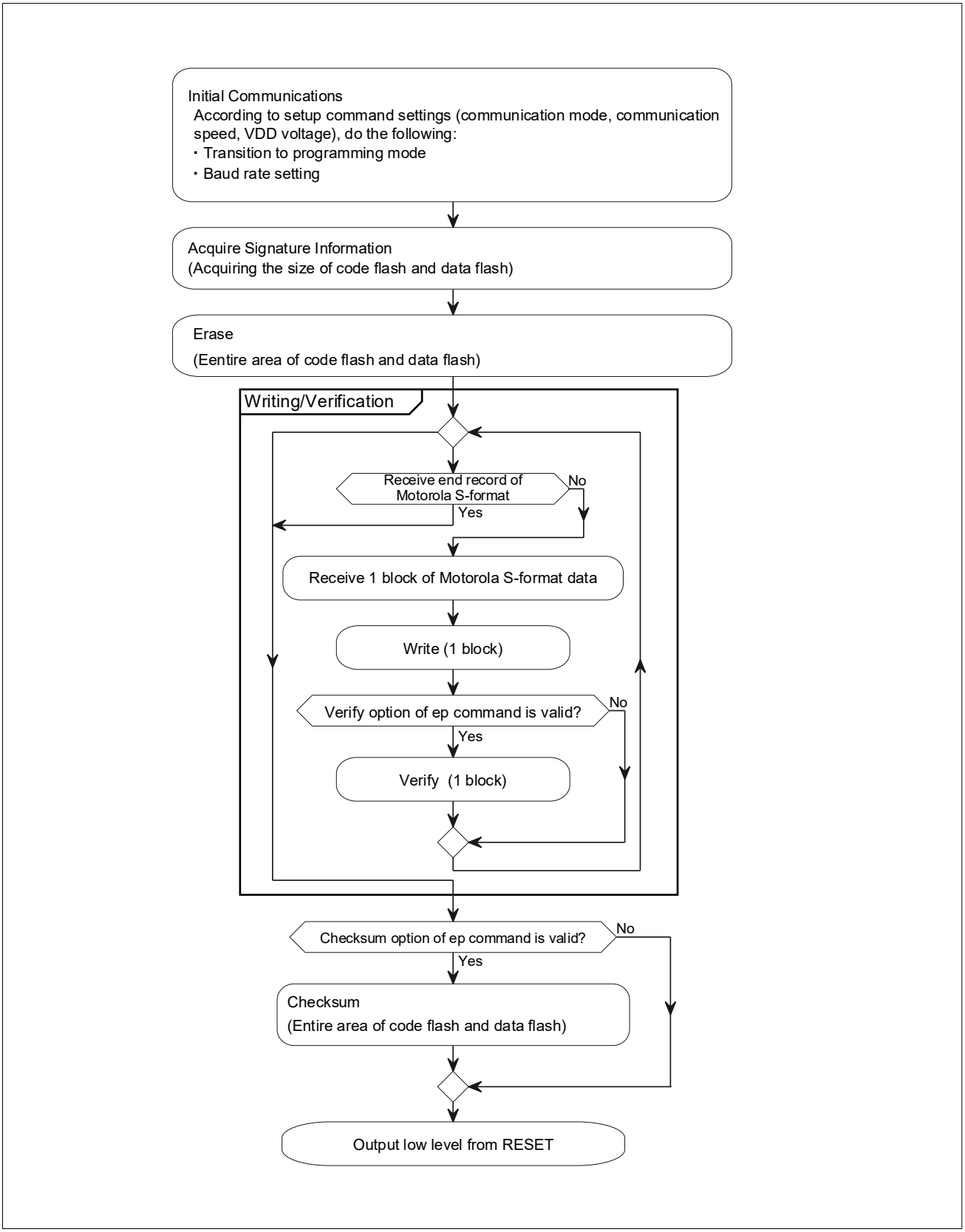
Figure 3-4 Main Loop



3.6.2 Flow of Writing in Online Mode

Figure 3-5 shows the flow of writing processing by executing ep command.

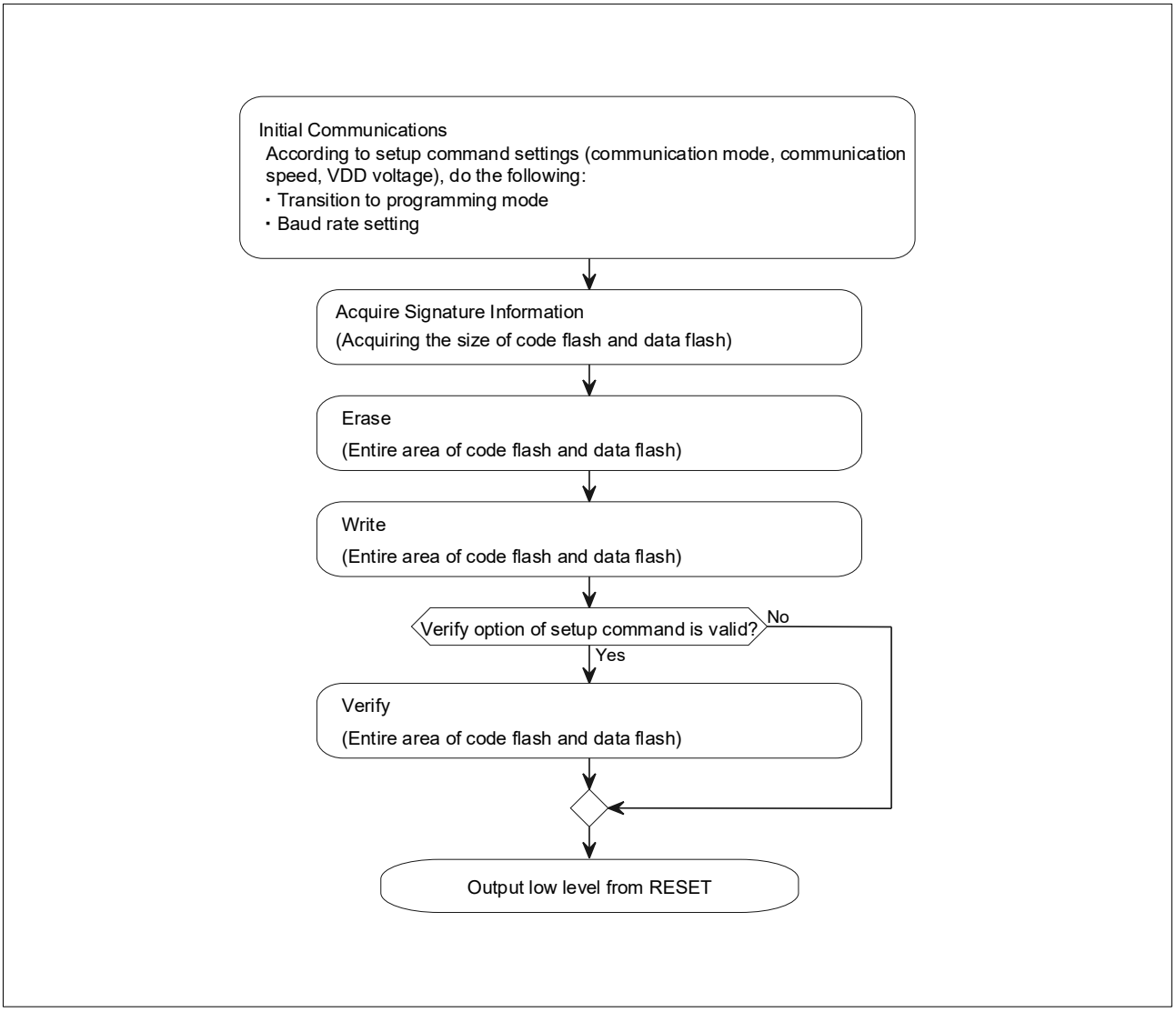
Figure 3-5 Online Mode



3.6.3 Flow of Writing in Offline Mode

Figure 3-6 shows the flow of writing processing by pushing the “SW1” switch.

Figure 3-6 Offline Mode

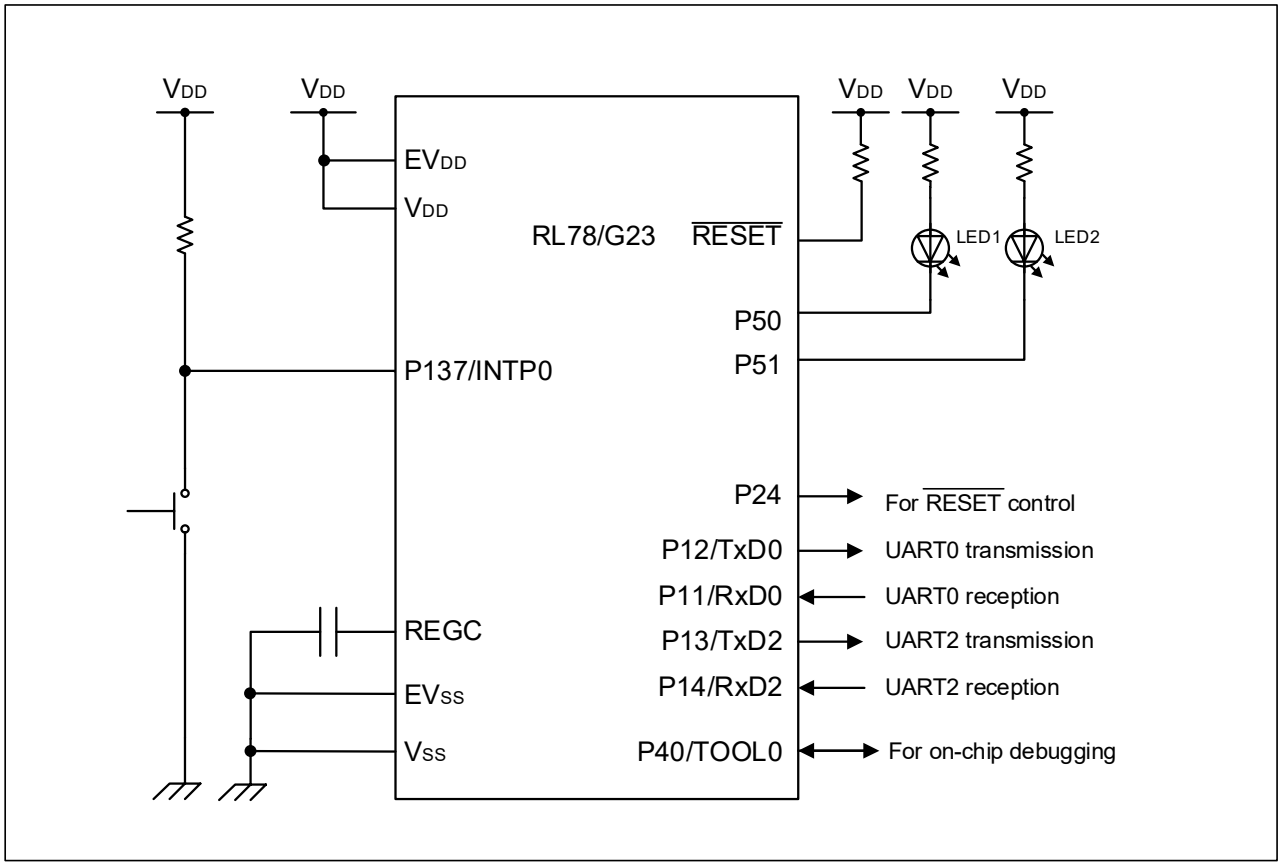


4. Hardware Descriptions

4.1 Example of Hardware Configuration

Figure 4-1 shows an example of the hardware configuration used in the application note.

Figure 4-1 Hardware Configuration



Note 1. This simplified circuit diagram was created to show an overview of connections only. When actually designing your circuit, make sure the design includes appropriate pin handling and meets electrical characteristic requirements (connect each input-only port to V_{DD} or V_{SS} through a resistor.)

Note 2. Connect any pins whose name begins with EV_{SS} to V_{SS}, and any pins whose name begins with EV_{DD} to V_{DD}, respectively.

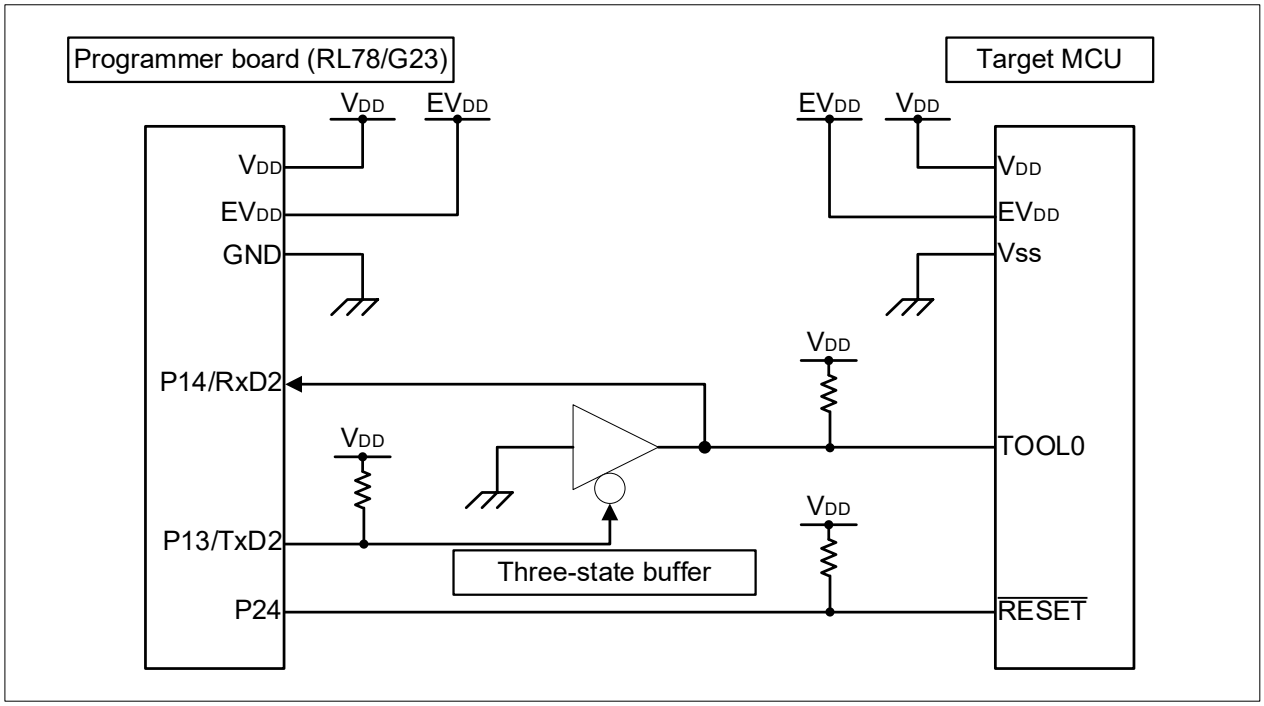
Note 3. V_{DD} must not be lower than the reset release voltage (V_{LVD0}) that is specified for the LVD0.

4.2 Target Interface Specifications

The following figures show how to connect the programmer board (RL78/G23) to the target MCU.

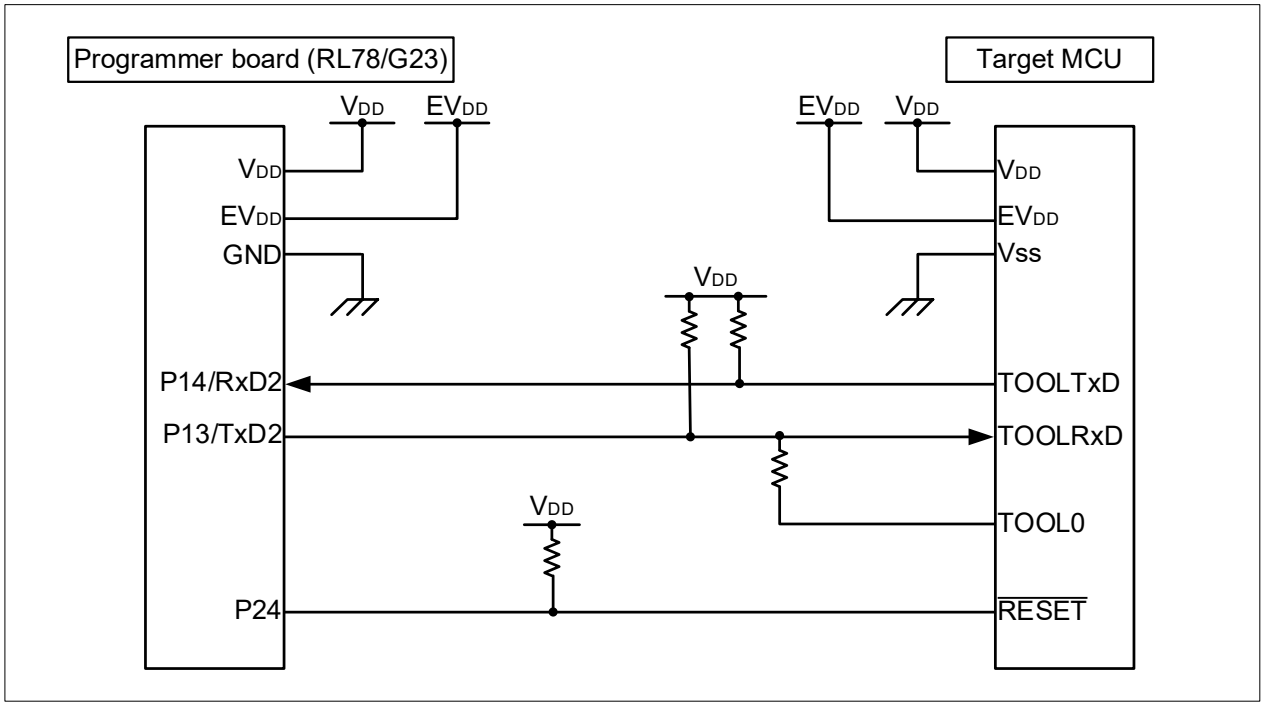
4.2.1 Single Line UART

Figure 4-2 Single Line UART



4.2.2 Dedicated UART

Figure 4-3 Dedicated UART



4.3 List of Pins to be Used

Table 4-1 lists the pins to be used and their functions.

Table 4-1 List of Pins to be Used

Pin Name	I/O	Function
P12/TxD0	Output	Host PC communication transmit pin (UART0)
P11/RxD0	Input	Host PC communication receive pin (UART0)
P13/TxD2	Output	Target interface communication transmit pin (UART2)
P14/RxD2	Input	Target interface communication receive pin (UART2)
P50, P51	Output	Output (to LED1/LED2) pins
P24	Output	Target MCU $\overline{\text{RESET}}$ control pin
P137/INTP0	Input	SW1 interrupt input pin

Caution In this application note, only the used pins are processed. When actually designing your circuit, make sure the design includes sufficient pin processing and meets electrical characteristic requirements.

4.4 Setting of RL78/G23-128p Fast Prototyping Board

When using the E2 emulator Lite or the E2 emulator for debugging, make the following settings for the RL78/G23-128p Fast Prototyping Board.

- Cut the cut-patterns "TOOL0_USB", "RESET", and "T_RESET".
- Short-circuit (between 2 and 3 pins) pin headers "J15", "J16", and "J19".

Set the operating voltages of VDD and EVDD with "J20".

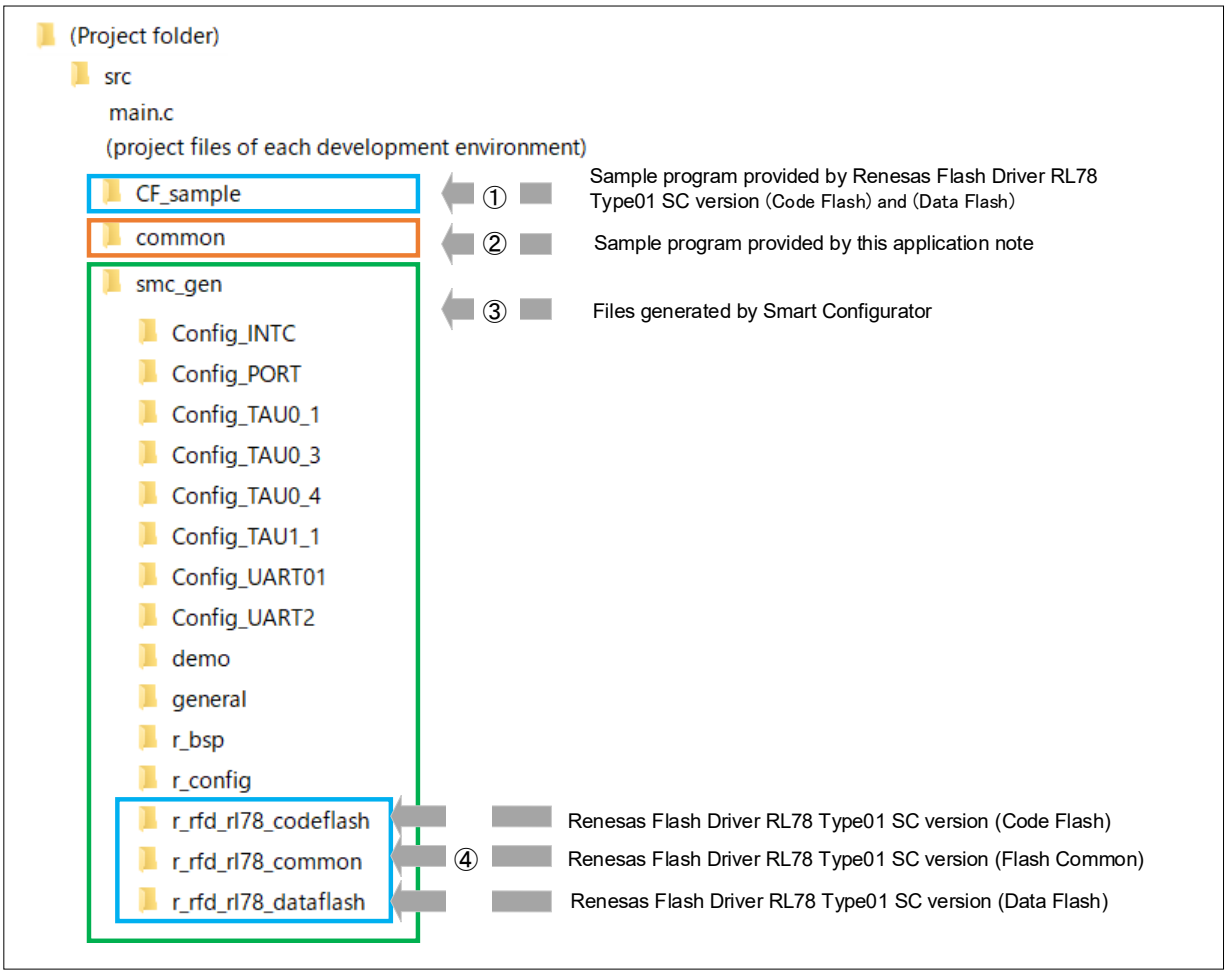
For details, refer to RL78/G23-128p Fast Prototyping Board User's Manual (R20UT4870).

5. Software explanation

5.1 Folder Structure

Figure 5-1 shows the folder structure of the sample program.

Figure 5-1 Folder Structure of Sample Program



Remark

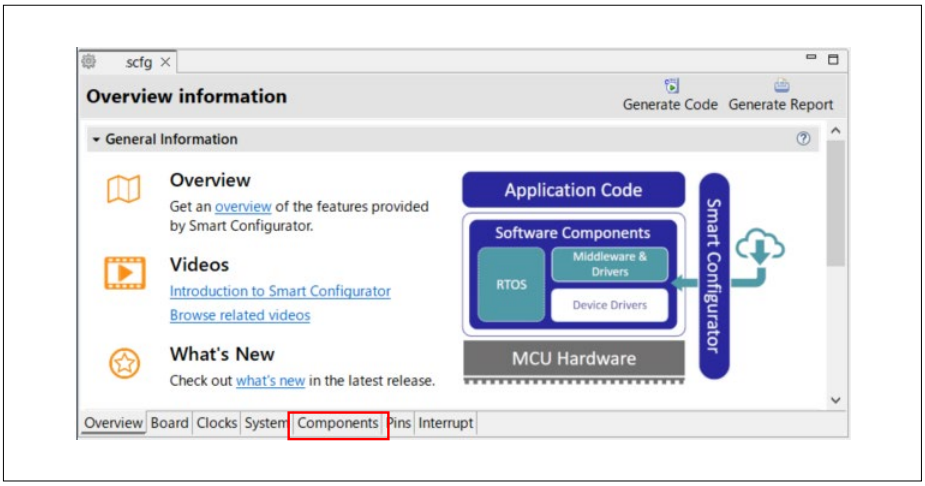
This sample program uses source files generated by Smart Configurator (SC). When source files are regenerated by Smart Configurator, download each Renesas Flash Driver RL78 Type01 SC version of ④ from Renesas Electronics website and copy them to the folder where Smart Configurator refers to. If source files are regenerated without copying each Renesas Flash Driver RL78 Type01 SC version, files under the folders ④ are deleted. If you do not regenerate codes in Smart Configurator, downloading and copying are not required.

[Procedure]

1. Download the followings from Renesas Electronics website and unzip them in any folder.
   - RL78 Family Renesas Flash Driver RL78 Type 01 SC version (Flash Common) - Sample Code
     https://www.renesas.com/us/en/document/scd/rl78-family-renesas-flash-driver-rl78-type-01-sc-version-flash-common-sample-code?language=en&r=488891
   - RL78 Family Renesas Flash Driver RL78 Type 01 SC version (Code Flash) - Sample Code
     https://www.renesas.com/jp/ja/document/scd/rl78-family-renesas-flash-driver-rl78-type-01-sc-version-code-flash-sample-code?r=488891
   - RL78 Family Renesas Flash Driver RL78 Type 01 SC version (Data Flash) - Sample Code
     https://www.renesas.com/us/en/document/scd/rl78-family-renesas-flash-driver-rl78-type-01-sc-version-data-flash-sample-code?language=en&r=488891

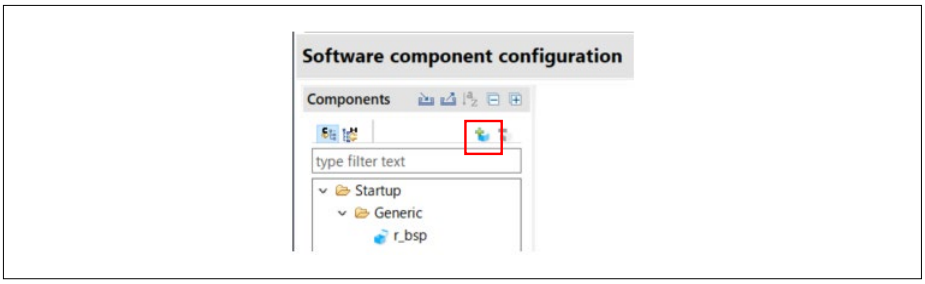
2. Start Smart Configurator and click “Components” tab.

Figure 5-2 Select Smart Configurator “Components” tab



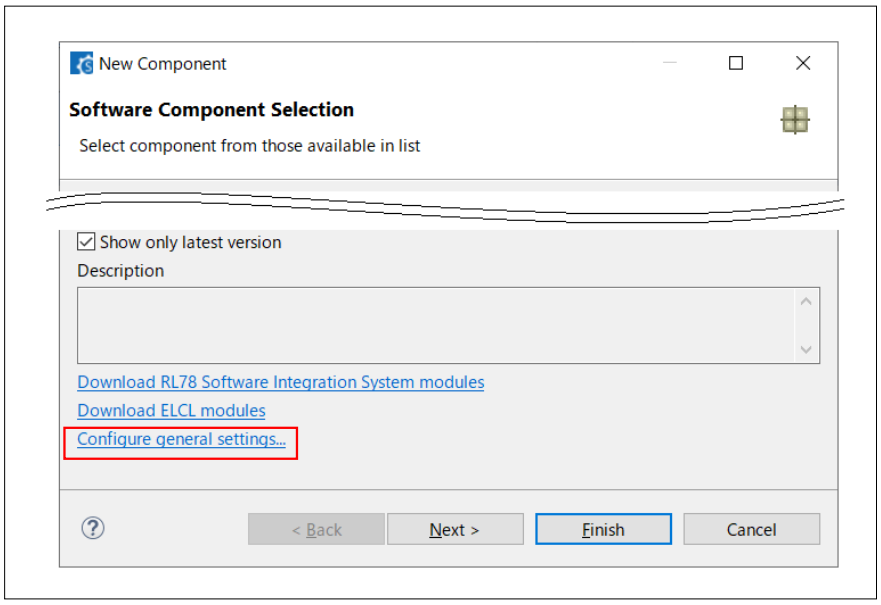
3. Click “addition Add component” button to open “New Component” dialog.

Figure 5-3 Smart Configurator “Add component” button



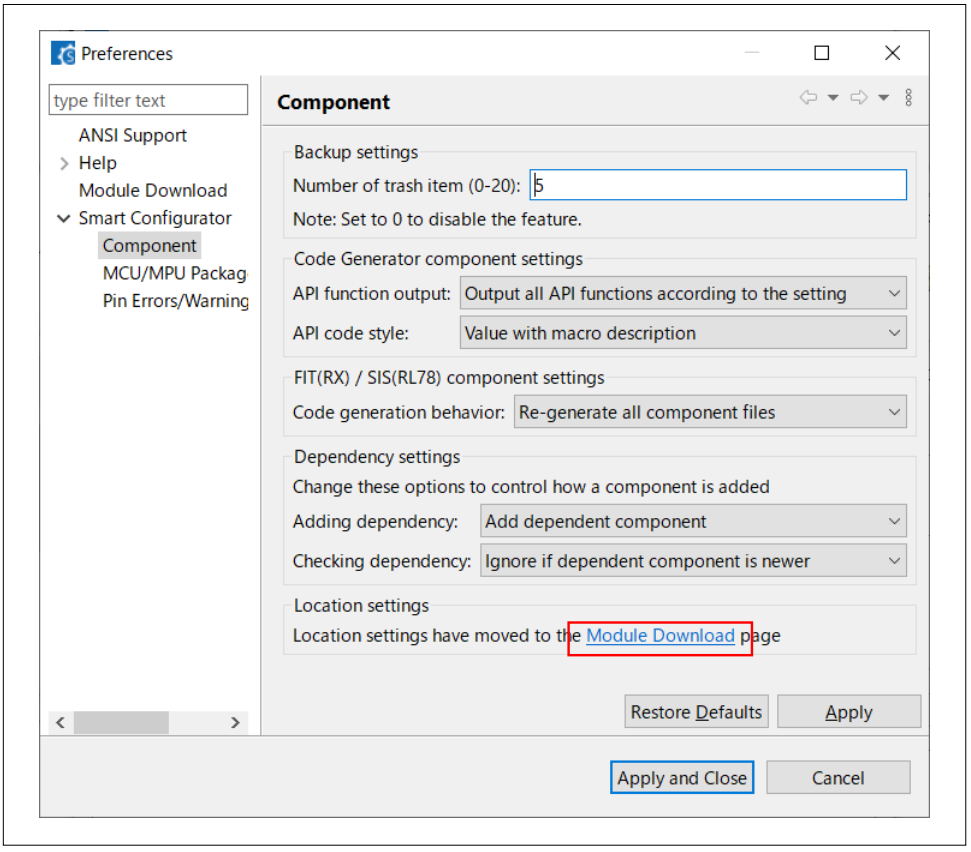
4. Click “Configure general settings” in “New Component” dialog.

Figure 5-4 Smart Configurator “Configure general settings”



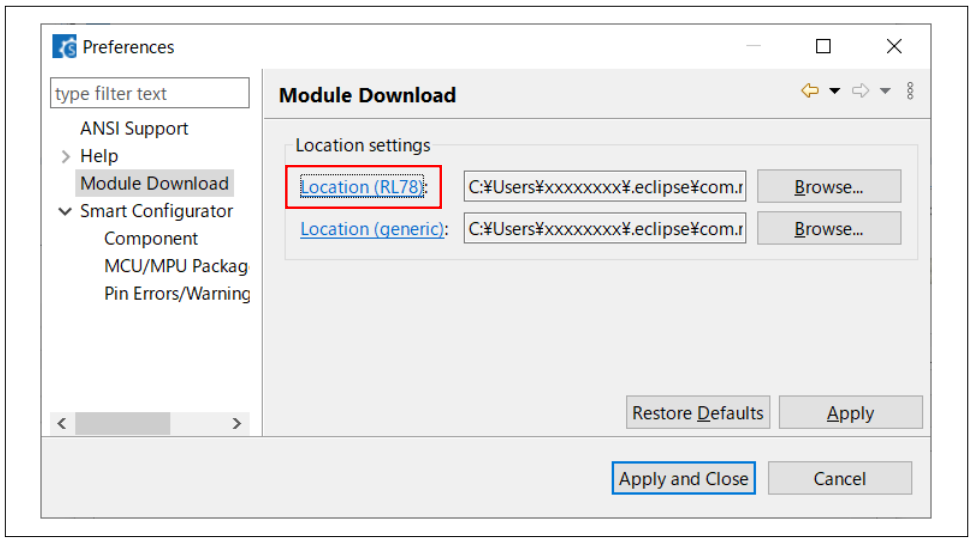
5. Click “Module Download” in “Preferences” dialog.

Figure 5-5 Smart Configurator “Preferences” dialog (1/2)



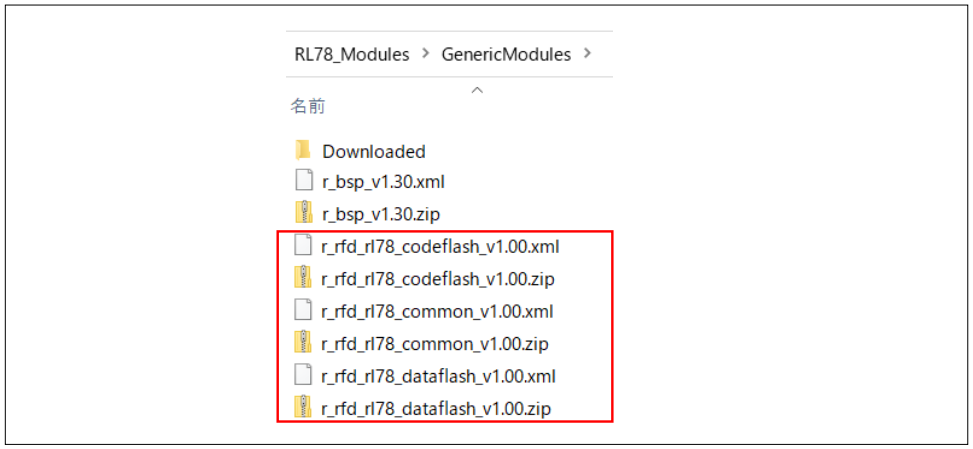
6. Click “Location(RL78)” in “Preferences” dialog.

Figure 5-6 Smart Configurator “Preferences” dialog (2/2)



7. Copy “.xml” files and “.zip” files unzipped in Step 1 to the folder opened in Windows Explorer.

Figure 5-7 Copy Renesas Flash Driver RL78 Type01 (SC version)



8. Click “Apply and Close” in “Preferences” dialog. Then click “Finish” in “New Component” dialog.

9. Restart Smart Configurator before regeneration to make Smart Configurator recognize Renesas Flash Driver RL78 Type01 copied in Step 7.

5.2 Setting of Option Byte

Table 5-1 shows the option byte settings used in this sample program.

Table 5-1 Option Byte Settings

Address	Setting Value	Content
000C0H/040C0H	01101110B	Disables the watchdog timer
000C1H/040C1H	11111111B	LVD0 detection voltage: Reset mode At rising edge TYP. 1.90 V (1.84 V ～ 1.95 V) At falling edge TYP. 1.86 V (1.80 V ～ 1.91 V)
000C2H/040C2H	11101000B	HS mode, High-speed on-chip oscillator clock: 32MHz
000C3H/040C3H	10000101B	Enables on-chip debugging

RL78/G23's option bytes consist of user option bytes (000C0H to 000C2H) and on-chip debug option byte (000C3H).

5.3 On-chip Debug Security ID

To protect against third parties reading the contents of memory, RL78/G23 has an area in the range from 000C4H to 000CDH to hold the security ID code for on-chip debugging.

5.4 Section Settings

Table 5-2 shows the CC-RL section settings.

- Sections beginning with "." are reserved by the compiler.
- Sections beginning with "RFD_" or "SMP_" are defined by the specifications of the Renesas Flash Driver RL78 Type01(RFD) and its sample program.
- Sections "FP_DATA" and "FP_CODE" are prepared for storing data to be written to the target MCU.
- Though reserved section names of IAR vary, IAR has a similar section configuration.

Table 5-2 List of Section

Section Name	Address	Description
.vect	0x00000	Interrupt vector table
.option_byte	0x000C0	Section specific for user option byte and on-chip debugging specification
.security_id	0x000C4	Section specific for security ID specification
.const	0x03000	ROM data (allocated to the near area)
.text	(Automatic allocation)	Section for code (allocated to the near area)
.RLIB	(Automatic allocation)	Section for code of runtime libraries
.SLIB	(Automatic allocation)	Section for code of standard libraries
.textf	(Automatic allocation)	Section for code (allocated to the far area)
.constf	(Automatic allocation)	ROM data (allocated to the far area)
.data	(Automatic allocation)	Section for near initialized data (with initial value)
.sdata	(Automatic allocation)	Section for initialized data (with initial value, variable allocated to saddr)
RFD_DATA_n	(Automatic allocation)	Section for RFD: Data section for initialized global variables
RFD_CMN_f	(Automatic allocation)	Section for RFD: Program section of API functions used in common for flash memory control
RFD_CF_f	(Automatic allocation)	Section for RFD: Program section of API functions for code flash memory control
RFD_DF_f	(Automatic allocation)	Section for RFD: Data section for initialized global variables
SMP_CMN_f	(Automatic allocation)	Section for RFD sample: Program section of sample functions used in common for flash memory control
SMP_CF_f	(Automatic allocation)	Section for RFD sample: Program section of sample functions for code flash memory control
.monitor2	0x1FE00	Debug monitor area
FP_DATA	0x34000	Section for write data to data flash: Save binary data extracted from the Motorola S-format file.
FP_CODE	0x36000	Section for write data to code flash: Save binary data extracted from the Motorola S-format file.
FP_CODE_END	0xB5FFF	End address of Section for write data to code flash
.dataR	0xF3F00	Section for near initialized data (with initial value) (Data copied from ROM area)
.bss	(Automatic allocation)	Section for data area (without initial value, allocated to the near area
RFD_DATA_nR	(Automatic allocation)	Section for RFD: Data section for initialized global variables (Data copied from ROM area)
RFD_CMN_fR	(Automatic allocation)	Section for RFD: Program section of API functions used in common for flash memory control (Program copied from ROM area)
RFD_CF_fR	(Automatic allocation)	Section for RFD: Program section of API functions for code flash memory control (Program copied from ROM area)
SMP_CMN_fR	(Automatic allocation)	Section for RFD sample: Program section of sample functions used in common for flash memory control (Program copied from ROM area)
SMP_CF_fR	(Automatic allocation)	Section for RFD sample: Program section of sample functions for code flash memory control (Program copied from ROM area)
.sdataR	0xFFE20	Section for initialized data (with initial value, variable allocated to saddr) (Data copied from ROM area)
.sbss	(Automatic allocation)	Section for data area (without initial value, variable allocated to saddr)

5.4.1 Data Flash Area (0xF1000 - 0xF2FFF, 8KB)

The information set by the setup command is stored at the start address (0xF1000) of the data flash memory as a st_command_data_t structure.

The programmer board reads the information set by the setup command from the data flash memory during the startup.

Table 5-3 shows the specifications of the st_command_data_t structure.

Table 5-3 st_command_data_t structure

Member	Description
e_terminal_command_t com	Temporary variable (Type of command received from the host PC)
e_uart_if_t uart_if	setup command setting (Communication mode) UART_1LINE: Single line UART (Initial setting) UART_2LINE: Dedicated UART
e_uart_speed_t speed	setup command setting (Transmission rate of the target MCU) UART_SPEED_DEFAULT: 115200 bps UART_SPEED_250000: 250000 bps UART_SPEED_500000: 500000 bps UART_SPEED_1000000: 1000000 bps
uint16_t vdd	setup command setting (VDD voltage)
uint8_t verify	setup command setting (option for execution of verification)
uint8_t checksum	Temporary variable (used by ep command)
uint8_t device_frq	Temporary variable (Signature information)
uint8_t flash_program_mode	Temporary variable (Signature information)
uint32_t device_code	Temporary variable (Signature information)
uint8_t device_name[11]	Temporary variable (Signature information)
uint32_t code_end	Temporary variable (Signature information)
uint32_t data_end	Temporary variable (Signature information)
uint32_t firm_version	Temporary variable (Signature information)

5.5 Smart Configurator Settings

The peripherals of RL78/G23 on the programmer board set by Smart Configurator are shown below.

(1) Initialize the I/O Port.

- Set P24, P50 and P51 to an output port.

(2) Initialize the Serial Array Unit.

- Use UART0. (P12 is used as TXD0, P11 is used as RXD0)
- Set the Operation clock to CK00, set the Clock source to fCLK/2.
- Set the Transfer mode to Single transfer mode.
- Set the Data length to 8 bits.
- Set the Transfer direction to LSB.
- Set the Parity to None.
- Set the Stop bit length to 1 bit.
- Set the Transfer data level to Non-reverse.
- Set the Transfer rate to 115200bps.

- Use UART2. (P13 is used as TXD2, P14 is used as RXD2)
- Set the Operation clock to CK10, set the Clock source to fCLK/2.
- Set the Transfer mode to Single transfer mode.
- Set the Data length to 8 bits.
- Set the Transfer direction to LSB.
- Set the Parity to None.
- Set the Stop bit length to 2 bits (Transmission) and 1 bit (Reception).
- Set the Transfer data level to Non-reverse.
- Set the Transfer rate to 115200bps.
- Set the Transmit end interrupt priority (INTST2) to Level 2.
- Set the Reception end interrupt priority (INTSR2) to Level 2.

(3) Initialize the External Interrupt

- Use INTP0.
- Set the Valid edge to Falling edge.

(4) Initialize the Timer Array Unit

- Use TAU0_1 as Interval Timer by 8 bit counter mode.
- Set the Operation clock to CK02.
- Set the Clock source to fCLK/2.
- Set the Operation mode to Lower 8 bits.
- Set the Interval value (lower 8 bits) to 174us.
- Check the Use End of timer channel 1 count, generate an interrupt (INTTM01) and set the Priority to Level 2.

- Use TAU0_3 as Interval Timer by 8 bit counter mode.
- Set the Operation clock to CK03.
- Set the Clock source to fCLK/2^8.
- Set the Operation mode to Lower 8 bits.
- Set the Interval value (lower 8 bits) to 1ms.
- Check the Use End of timer channel 3 count, generate an interrupt (INTTM03)

- Use TAU0_4 as Interval Timer by 16 bit counter mode.
- Set the Operation clock to CK00.
- Set the Clock source to fCLK/2^8.
- Set the Interval value (16 bits) to 500ms.
- Check the Use End of timer channel 4 count, generate an interrupt (INTTM04)

- Use TAU1_1 as Interval Timer by 8 bit counter mode.
- Set the Operation clock to CK12.
- Set the Clock source to fCLK/2.
- Set the Operation mode to Lower 8 bits.
- Set the Interval value (lower 8 bits) to 1us.
- Check the Use End of timer channel 1 count, generate an interrupt (INTTM11)

- Use TAU1_2 as Interval Timer by 16 bit counter mode.
- Set the Operation clock to CK10.
- Set the Clock source to fCLK/2^5.
- Set the Interval value (16 bits) to 1us.
- Check the Use End of timer channel 2 count, generate an interrupt (INTTM12)

(5) Initialize the Voltage Detector

- Set the Operation mode to Reset mode.
- Set the Reset generation level(VLVD0) to 1.86V.

5.6 List of Functions

Table 5-4 shows the sample program functions.

Table 5-4 List of Functions

Function Name	Outline
r_Terminal_Init_Flash	Initializes Renesas Flash Driver RL78 Type01.
r_Terminal_Command_Init	Initializes the command setting information.
r_Terminal_Command_Init_Dev	Initializes the signature information in the command setting information.
r_Terminal_Command_Read	Acquires the command setting information of the setup command.
r_Terminal_Command_Recieve	Receives the command from the host PC.
r_Terminal_Command_SETUP	setup command processing.
r_Terminal_Command_LOD	lod command processing.
r_Terminal_Command_EP	ep command processing.
r_Terminal_Command_SW	Writing processing in offline mode.
r_Terminal_Send_Error_Message	Sends error message to the host PC.
r_Terminal_Clear_Write_Buffer	Clears the buffer for write data.
r_Decord_SRecord	Analyzes Motorola S-format file.
R_Config_PORT_Dev_Reset	Reset control of the target MCU.
R_Config_UART01_Terminal_Recv_Through	Discards transmit data of the host PC.
R_Config_UART01_Terminal_Set_Echo	Echo back setting for the communication with the host PC.
R_Config_UART01_Terminal_Data_Process_Start	Receives 1 line from the host PC.
R_Config_UART01_Terminal_Data_Process_End	Termination setting of 1 line reception from the host PC.
r_FP_Initial_Communication	Executes communication setting processing with the target MCU.
r_FP_Get_Signature	Acquires the signature information.
r_FP_CMD_Baudrate_A	Executes the Baud Rate Set command.
r_FP_CMD_Reset_A	Executes the Reset command.
r_FP_CMD_Signature_A	Executes the Silicon Signature command.
r_FP_CMD_Erase_A	Executes the Block Erase command.
r_FP_CMD_Program_A	Executes the Programming command.
r_FP_CMD_Verify_A	Executes the Verify command.
r_FP_CMD_Checksum_A	Executes the Checksum command.

5.7 Specification of Functions

The function specifications of the sample code are shown below.

r_Terminal_Init_Flash	
Outline	Initializes Renesas Flash Driver RL78 Type01.
Declaration	int r_Terminal_Init_Flash(void)
Argument	None
Return Value	RFD status code
Description	Initializes Renesas Flash Driver RL78 Type01 (RFD).

r_Terminal_Command_Init	
Outline	Initializes the command setting information.
Declaration	void r_Terminal_Command_Init(st_command_data_t * com_data)
Argument	st_command_data_t * com_data: Command setting information
Return Value	None
Description	Initializes “com_data” specified by the argument.

r_Terminal_Command_Init_Dev	
Outline	Initializes the signature information in the command setting information.
Declaration	void r_Terminal_Command_Init_Dev(st_command_data_t * com_data)
Argument	st_command_data_t * com_data: Command setting information
Return Value	None
Description	Initializes the signature information in the “com_data” specified by the argument.

r_Terminal_Command_Read	
Outline	Acquires the command setting information of the setup command.
Declaration	void r_Terminal_Command_Read(st_command_data_t * com_data)
Argument	st_command_data_t * com_data: Command setting information
Return Value	None
Description	Acquires the command setting information of the setup command from the data flash and set it to the “com_data” specified by the argument.

r_Terminal_Command_Recieve	
Outline	Receives the command from the host PC.
Declaration	void r_Terminal_Command_Recieve(st_command_data_t * com_data)
Argument	st_command_data_t * com_data: Command setting information
Return Value	None
Description	Receives the command line transferred from the host PC and set it to the “com_data” specified by the argument as the command information.

r_Terminal_Command_SETUP	
Outline	setup command processing.
Declaration	void r_Terminal_Command_SETUP(st_command_data_t * com_data)
Argument	st_command_data_t * com_data: Command setting information
Return Value	None
Description	Stores the command setting information specified by the setup command in the data flash.

r_Terminal_Command_LOD	
Outline	lod command processing.
Declaration	int r_Terminal_Command_LOD(st_command_data_t * com_data)
Argument	st_command_data_t * com_data: Command setting information
Return Value	0: Normal end 1: Abnormal end
Description	Erases the FP_CODE and FP_DATA sections in the code flash memory, receives and analyzes the Motorola S-format file from the host PC, and then stores the acquired binary data in the FP_CODE and FP_DATA sections.

r_Terminal_Command_EP	
Outline	ep command processing.
Declaration	int r_Terminal_Command_EP(st_command_data_t * com_data)
Argument	st_command_data_t * com_data: Command setting information
Return Value	0: Normal end 1: Abnormal end
Description	Executes the ep command. For the flow of the processing, refer to 3.6.2 Flow of Writing in Online Mode.

r_Terminal_Command_SW	
Outline	Writing processing in offline mode.
Declaration	int r_Terminal_Command_SW(st_command_data_t * com_data)
Argument	st_command_data_t * com_data: Command setting information
Return Value	0: Normal end 1: Abnormal end
Description	Executes the writing processing in the Offline mode. For the flow of the processing, refer to 3.6.3 Flow of Writing in Offline Mode.

r_Terminal_Send_Error_Message	
Outline	Sends error message to the host PC.
Declaration	void r_Terminal_Send_Error_Message(uint8_t err)
Argument	uint8_t err: Error code
Return Value	None
Description	Sends the error message to the host PC.

r_Terminal_Clear_Write_Buffer	
Outline	Clears the buffer for write data.
Declaration	void r_Terminal_Clear_Write_Buffer(void)
Argument	None
Return Value	None
Description	Initializes the buffer for write data (global variable) used by the lod command and the ep command with 0xFF.

r_Decord_SRecord	
Outline	Analyzes Motorola S-format file.
Declaration	void r_Decord_SRecord(uint8_t * srecord_str, uint16_t len, st_srecord_data_t * srecord_data)
Argument	uint8_t * srecord_str: Record data storage buffer uint16_t len: Length of record data st_srecord_data_t * srecord_data: Record data information
Return Value	None
Description	Analyzes 1 line of the Motorola S-format file and acquires the record type, address, data and data size as the record data information.

R_Config_PORT_Dev_Reset	
Outline	Reset control of the target MCU.
Declaration	void R_Config_PORT_Dev_Reset(RESET_CONTROL reset)
Argument	None
Return Value	RESET_CONTROL reset: Reset control RESET_ENABLE: $\overline{\text{RESET}}$ is active. RESET_DISABLE: $\overline{\text{RESET}}$ is inactive.
Description	Controls the reset of the target MCU.

R_Config_UART01_Terminal_Recv_Through	
Outline	Discards transmit data of the host PC.
Declaration	void R_Config_UART01_Terminal_Recv_Through(const uint16_t timeout)
Argument	const uint16_t timeout: Timeout time (ms)
Return Value	None
Description	Discards data transmitted from the host PC during the specified time-out period. Used to discard Motorola S-format data when the lod command or ep command results in an abnormal termination.

R_Config_UART01_Terminal_Set_Echo	
Outline	Echo back setting for the communication with the host PC.
Declaration	void R_Config_UART01_Terminal_Set_Echo(uint8_t echo)
Argument	uint8_t echo: Echo back setting (Valid = 1, Invalid = 0)
Return Value	Nonw
Description	Specifies whether to enable or disable echo back during communication with the host PC.

R_Config_UART01_Terminal_Data_Process_Start	
Outline	Receives 1 line from the host PC.
Declaration	MD_STATUS R_Config_UART01_Terminal_Data_Process_Start(uint8_t ** rx_buf, uint16_t * rx_num, uint8_t * sw)
Argument	uint8_t ** rx_buf: Receive buffer uint16_t * rx_num: Number of received characters uint8_t * sw: Flag for SW1 (Pushed = 1, Not pushed = 0)
Return Value	MD_OK: Normal end
Description	Receives single-line data from the host PC. Before receiving the next-line data, execute R_Config_UART01_Terminal_Data_Process_End() once.

R_Config_UART01_Terminal_Data_Process_End	
Outline	Termination setting of 1 line reception from the host PC.
Declaration	void R_Config_UART01_Terminal_Data_Process_End(void)
Argument	None
Return Value	None
Description	Always used together with R_Config_UART01_Terminal_Data_Process_Start(). Execute this function after executing R_Config_UART01_Terminal_Data_Process_Start().

r_FP_Initial_Communication	
Outline	Executes communication setting processing with the target MCU.
Declaration	uint8_t r_FP_Initial_Communication(st_command_data_t * command)
Argument	st_command_data_t * com_data: Command setting information
Return Value	0: Normal end Other than 0: Abnormal end (Refer to 3.5 Error Code Specifications)
Description	Performs communication setting processing with the target MCU according to the RL78 Protocol A. Releases the reset state, sends mode information, and executes the Baud Rate Set command, Reset command, and Security ID Authentication command (in the case of a command number error generated by the Reset command).

r_FP_Get_Signature	
Outline	Acquires the signature information.
Declaration	uint8_t r_FP_Get_Signature(st_command_data_t * command)
Argument	st_command_data_t * com_data: Command setting information
Return Value	0: Normal end Other than 0: Abnormal end (Refer to 3.5 Error Code Specifications)
Description	Acquires the signature information of the target MCU and set it to “com_data” specified by the argument.

r_FP_CMD_Baudrate_A	
Outline	Executes the Baud Rate Set command.
Declaration	uint8_t r_FP_CMD_Baudrate_A(const e_uart_speed_t baudrate, const uint16_t vdd, uint8_t * frq, uint8_t * fpm)
Argument	const e_uart_speed_t baudrate : Communication speed UART_SPEED_DEFAULT: 115200 bps UART_SPEED_250000: 250000 bps UART_SPEED_500000: 500000 bps UART_SPEED_1000000: 1000000 bps const uint16_t vdd: VDD voltage (100 mV) uint8_t * frq: CPU operation frequency (MHz) (Acquired from the target MCU) uint8_t * fpm: Flash memory rewriting mode (Acquired from the target MCU)
Return Value	0: Normal end Other than 0: Abnormal end (Refer to 3.5 Error Code Specifications)
Description	Executes the Baud Rate Set command of the RL78 Protocol A.

r_FP_CMD_Reset_A	
Outline	Executes the Reset command.
Declaration	uint8_t r_FP_CMD_Reset_A(void)
Argument	None
Return Value	0: Normal end Other than 0: Abnormal end (Refer to 3.5 Error Code Specifications)
Description	Executes the Reset command of the RL78 Protocol A.

r_FP_CMD_Signature_A	
Outline	Executes the Silicon Signature command.
Declaration	uint8_t r_FP_CMD_Signature_A(uint8_t * dvc, uint8_t * dev, uint32_t * cfe, uint32_t * dfe, uint8_t * fwv)
Argument	All data is acquired from the target MCU. uint8_t * dvc: Device function code uint8_t * dev: Device name uint32_t * cfe: Last address of code flash memory area uint32_t * dfe: Last address of data flash memory area uint8_t * fwv: Boot firmware version
Return Value	0: Normal end Other than 0: Abnormal end (Refer to 3.5 Error Code Specifications)
Description	Executes the Silicon Signature command of the RL78 Protocol A.

r_FP_CMD_Erase_A	
Outline	Executes the Block Erase command.
Declaration	uint8_t r_FP_CMD_Erase_A(const uint32_t addr)
Argument	const uint32_t addr: Address of the block to be erased
Return Value	0: Normal end Other than 0: Abnormal end (Refer to 3.5 Error Code Specifications)
Description	Executes the Block Erase command of the RL78 Protocol A.

r_FP_CMD_Program_A	
Outline	Executes the Programming command.
Declaration	uint8_t r_FP_CMD_Program_A(const uint32_t start, const uint32_t end, const uint8_t __far * data)
Argument	const uint32_t start: Start address of writing const uint32_t end: End address of writing const uint8_t __far * data: Write data
Return Value	0: Normal end Other than 0: Abnormal end (Refer to 3.5 Error Code Specifications
Description	Executes the Programming command of the RL78 Protocol A.

r_FP_CMD_Verify_A	
Outline	Executes the Verify command.
Declaration	uint8_t r_FP_CMD_Verify_A(const uint32_t start, const uint32_t end, const uint8_t __far * data)
Argument	const uint32_t start: Start address of verification const uint32_t end: End address of verification const uint8_t __far * data: Comparison data for verification
Return Value	0: Normal end Other than 0: Abnormal end (Refer to 3.5 Error Code Specifications)
Description	Executes the Verify command of the RL78 Protocol A.

r_FP_CMD_Checksum_A	
Outline	Executes the Checksum command.
Declaration	uint8_t r_FP_CMD_Checksum_A(const uint32_t start, const uint32_t end, uint16_t * checksum)
Argument	const uint32_t start: Start address of checksum const uint32_t end: End address of checksum const uint16_t * checksum: Calculated checksum value
Return Value	0: Normal end Other than 0: Abnormal end (Refer to 3.5 Error Code Specifications)
Description	Executes the Checksum command of the RL78 Protocol A.

6. Reference Documents

RL78/G23 User's Manual: Hardware (R01UH0896)

RL78 Microcontrollers RL78 Microcontrollers (RL78 Protocol A) Programmer Edition (R01AN0815)

RL78 Family Renesas Flash Driver RL78 Type 01 User's Manual (R20UT4830)

RL78 Family Renesas Flash Driver RL78 Type 01 SC version (Code Flash) (R20AN0653)

RL78 Family Renesas Flash Driver RL78 Type 01 SC version (Data Flash) (R20AN0654)

RL78/G23-128p Fast Prototyping Board User's Manual (R20UT4870)

The latest versions can be downloaded from the Renesas Electronics website.

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Feb 16, 2023	-	First edition

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)

   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on

   The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state

   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins

   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.
5. Clock signals

   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.
6. Voltage application waveform at input pin

   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses

   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products

   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

Notice

1. 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 or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving 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, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, 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 and impractical, you are responsible for evaluating the safety of the final products or systems 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. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall 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. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.