RL78/G13

Clock Generator (Clock Switching)

R01AN0811EJ0100
Rev. 1.00
Feb. 08, 2012

Introduction

This application note explains how to use the clock generator of the RL78/G13.

The clock from the clock generator is switched when a switch is pressed. The clock generator uses the high-speed on-chip oscillator clock, (32 MHz), X1 oscillation clock (20 MHz), or XT1 oscillation clock (32.768 kHz) as the CPU/peripheral hardware clock (f_{CLK}).

Target Device

RL78/G13 (40 pins or more)

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

The sample code covered in this application note switches its operating clock when the switch on the target board is pressed according to the sequence below.

(1) High-speed on-chip oscillator clock (32 MHz) → X1 oscillation clock (20 MHz)
(2) X1 oscillation clock (20 MHz) → XT1 oscillation clock (32.768 kHz)
(3) XT1 oscillation clock (32.768 kHz) → High-speed on-chip oscillator clock (32 MHz)
Subsequently, steps (1) to (3) are repeated.

The sample code takes the following actions according to its operating state:

- When the high-speed on-chip oscillator clock (HOCO clock) operates: Stops the X1 oscillation clock.
- When the X1 oscillation clock operates: Stops the HOCO clock.
- When the XT1 oscillation clock operates: Stops the X1 oscillation clock and the HOCO clock.

The XT1 oscillation clock is always generated.

The sample code also changes the LED blinking period on the target board as shown below according to the operating clock. This allows the operating clock to be visually checked.

LED blinking period of the HOCO clock (32 MHz): 0.5 seconds

LED blinking period of the X1 oscillation clock (20 MHz): 1 seconds

LED blinking period of the XT1 oscillation clock (32.768 kHz): 2 seconds

Table 1.1 summarizes the peripheral functions to be used and their uses. Figure 1.1 shows the outline of the clock switching.

Table 1.1 Peripheral Functions to be Used and their Uses

Peripheral Function	Use
Clock generator	Generates oscillation clocks and switches the operating clocks.
External interrupt input (INTP0)	Detects the press of the switch.
Timer array unit 0 channel 0	Generates the timing signal to determine the LED blinking period.
12-bit interval timer	Generates the wait time to deal with chattering.
P62	Generates the output signal to the LED.

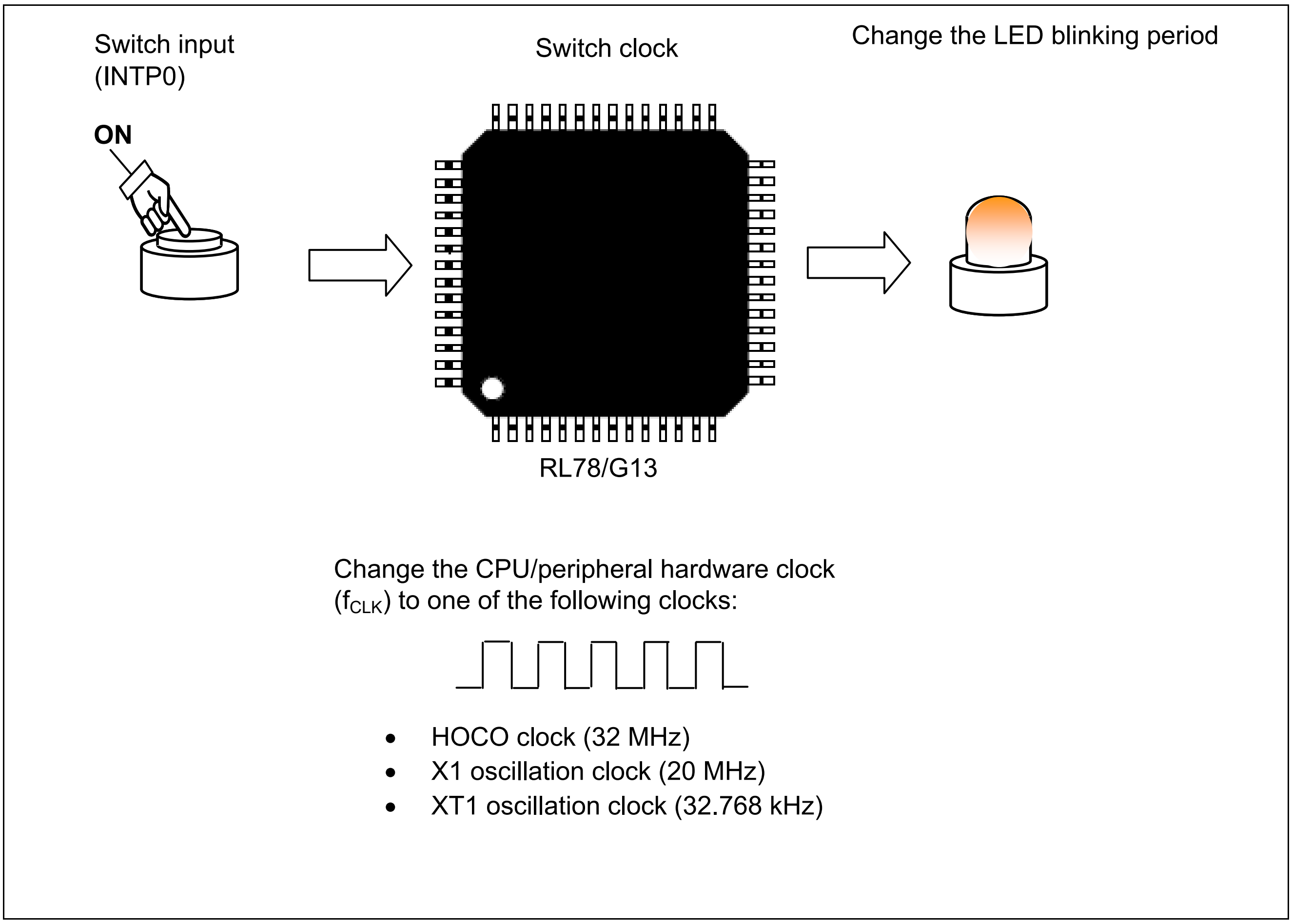

Figure 1.1 Outline of Clock Switching

2. Operation Check Conditions

The sample code contained 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	• CPU/peripheral hardware clock: Switches the operating clock when the switch on the target board is pressed. When the HOCO clock is selected: 32 MHz When the X1 oscillation clock is selected: 20 MHz When the XT1 oscillation clock is selected: 32.768 kHz
Operating voltage	5.0 V (can run on a voltage range of 2.9 V to 5.5 V.) LVD operation (V_{LVI}): Reset mode 2.81 V (2.76 V to 2.87 V)
Integrated development environment	CubeSuite + V1.00.01 from Renesas Electronics Corp.
C compiler	CA78K0R V1.20 from Renesas Electronics Corp.
Board used	RL78/G13 target board (QB-R5F100LE-TB)

3. Related Application Note

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

RL78/G13 Initialization (R01AN0451E) Application Note

4. Description of the Hardware

4.1 Hardware Configuration Example

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

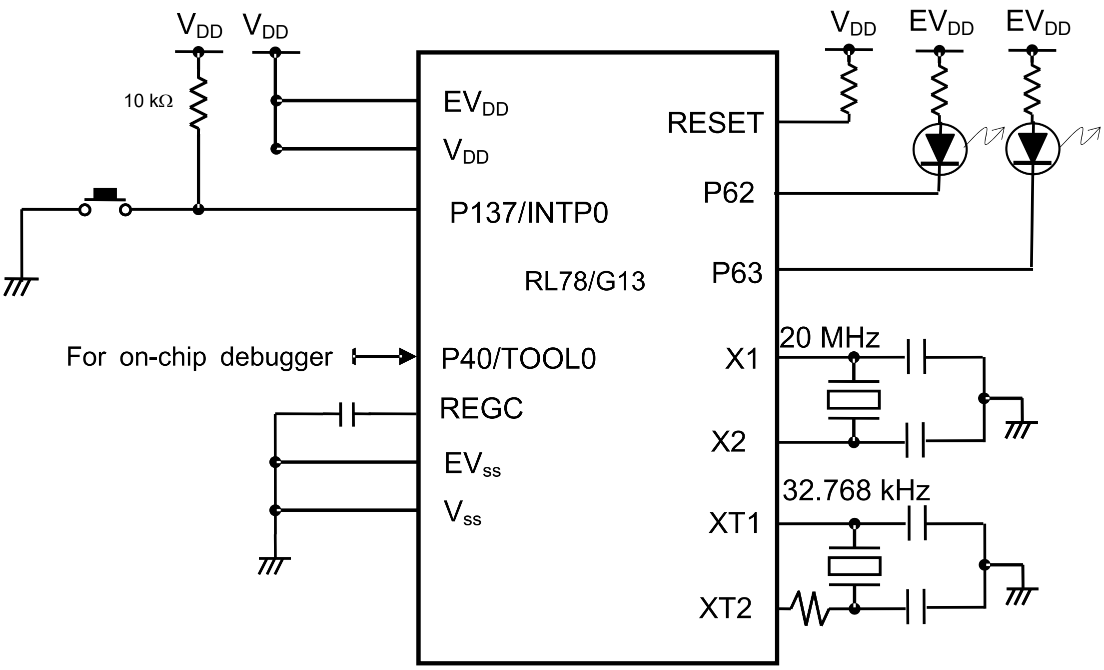

Figure 4.1 Hardware Configuration

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. 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.
3. V_{DD} must be held at not lower than the reset release voltage (V_{LVI}) that is specified as LVD.
4. The LED connected to P63 is always off.

4.2 List of Pins to be Used

Table 4.1 lists the pins to be used and their functions.

Table 4.1 Pins to be Used and their Functions

Pin Name	I/O	Description
P137/INTP0	Input	Switch input
P62	Output	LED output

5. Software Description

5.1 Operation Overview

The sample code covered in this application note switches its operating clock when the switch on the target board is pressed.

(1) Initialize the clock generator.
The sample code initializes the I/O ports, clock generator, timer array unit 0 (TAU0), 12-bit interval timer, and external interrupt input hardware. It enables interrupt processing after the initialization.

The LED blinks at the interval of the TAU0 interval interrupts corresponding to the selected operating clock.

(2) Get the switch state.
The sample code gets the switch state. It switches the operating clock when the press of the switch is detected. The switch state is tested when an INTP0 external interrupt occurs. If the press of the switch is not detected, the sample code places the CPU into the HALT mode.

(3) Switch the clock.
The CPU/peripheral hardware clock (f_{CLK}) is switched when the switch is pressed.

The CPU/peripheral hardware clock (f_{CLK}) is switched according to the sequence below.

1) HOCO clock (32 MHz) → X1 oscillation clock (20 MHz)

2) X1 oscillation clock (20 MHz) → XT1 oscillation clock (32.768 kHz)

3) XT1 oscillation clock (32.768 kHz) → HOCO clock (32 MHz)

Subsequently, steps (1) to (3) are repeated.

(4) Get the clock status.
The sample code gets the clock status. If the clock status is found to have been changed, the sample code takes one of the following actions according to the clock operating state:

- When the high-speed on-chip oscillator clock (HOCO clock) operates: Stops the X1 oscillation clock.
- When the X1 oscillation clock operates: Stops the HOCO clock.
- When the XT1 oscillation clock operates: Stops the X1 oscillation clock and the HOCO clock.

The XT1 oscillation clock is always generated.

(5) Change the LED blinking period.
The sample code changes the TAU0 interrupt interval as follows according to the operating CPU/peripheral hardware clock (f_{CLK}):

LED blinking period of the HOCO clock (32 MHz):	0.5 seconds
LED blinking period of the X1 oscillation clock (20 MHz):	1 seconds
LED blinking period of the XT1 oscillation clock (32.768 kHz):	2 seconds

(6) Transition to the HALT mode.
The sample code transitions to the HALT mode. It returns from the HALT mode by TAU0 interval interrupt or external interrupt generated by the switch. After returning from the HALT mode, the sample code performs step (2). Subsequently, it repeats a cycle of steps (2) to (6).

5.2 File Configuration

Table 5.1 lists the files that are used in this sample code. This table excludes files which are automatically generated by the integrated development environment.

Table 5.1 File Configuration

File Name	Description	Remarks
r_cgc.c	Clock generator module	CPU clock initialization
r_cg_cgc.h	External reference header file for the clock generator module	
r_cgc_user.c	Processing specific to the clock generator sample code	Additional functions: R_CGC_ChangeClock, R_CGC_HOCOToX1, R_CGC_X1ToXT1, R_CGC_XT1ToHOCO, R_CGC_GetClockStatus, R_CGC_Get_X1_Status R_CGC_Get_XT1_Status, R_CGC_Get_HOCO_Status, R_CGC_StopClock
r_intc.c	External interrupt input module	
r_cg_intc.h	External reference header file for the external interrupt input module	
r_intc_user.c	External interrupt input module INTP0 external interrupt	
r_it.c	12-bit interval timer module	
r_cg_it.h	External reference header file for the 12-bit interval timer module	
r_it_user.c	Processing specific to the 12-bit interval timer module sample code	Additional function: R_IT_Wait_ms
r_main.c	Main processing	
r_cg_macrodriver.h	Common header file	Type definitions, Macro definitions about error status
r_cg_userdefine.h	Macro definitions specific to the sample code	
r_port.c	Port function module	I/O port setting
r_cg_port.h	External reference header file for the port function module	
r_systeminit.c	System module	Initialization and system functions
r_timer.c	Timer module	
r_cg_timer.h	External reference header file for the timer module	
r_timer_user.c	Processing specific to the timer module sample code TAU0 channel 0 interrupt	Additional functions: R_TAU0_Channel0_GetParameter, R_TAU0_Channel0_Restart, R_TAU0_Channel0_ChangeInterval, R_TAU0_Channel0_Interrupt

5.3 List of Option Byte Settings

Table 5.2 summarizes the settings of the option bytes.

Table 5.2 Option Byte Settings

Address	Value	Description
000C0H/010C0H	11101111B	Disables the watchdog timer. (Stops counting after the release from the reset status.)
000C1H/010C1H	01111111B	LVD reset mode, 2.81 V (2.76 V to 2.87 V)
000C2H/010C2H	11101000B	HS mode HOCO: 32 MHz
000C3H/010C3H	10000100B	Enables the on-chip debugger. Erases the data in the flash memory when on-chip debugging security ID authentication fails.

5.4 List of Constants

Table 5.3 lists the constants that are used in this sample program.

Table 5.3 Constants for Sample Program

Constant	Setting	Description
HOCO_NEXT_STATUS_X1	1	Clock status : Operating on the HOCO clock and the next clock is the X1 oscillation clock.
X1_NEXT_STATUS_XT1	2	Clock status : Operating on the X1 oscillation clock and the next clock is the XT1 oscillation clock.
XT1_NEXT_STATUS_HOCO	3	Clock status : Operating on the XT1 oscillation clock and the next clock is the HOCO clock.
X1_STATUS	1	The current status is operation on X1 oscillation clock.
XT1_STATUS	2	The current status is operation on XT1 oscillation clock.
HOCO_STATUS	3	The current status is operation on HOCO clock.
CHATTERING_WAIT	10	Wait time to deal with chattering is 10 ms.
HOCO_LED_SETTING_CHANNEL_PRESCALER	9	Frequency division ratio of channel 0 of TAU0 when HOCO clock is selected
HOCO_LED_SETTING_CHANNEL_COUNT	15625	Count value of channel 0 of TAU0 when HOCO clock is selected
X1_LED_SETTING_CHANNEL_PRESCALER	9	Frequency division ratio of channel 0 of TAU0 when X1 oscillation clock is selected
X1_LED_SETTING_CHANNEL_COUNT	19531	Count value of channel 0 of TAU0 when X1 oscillation clock is selected
XT1_LED_SETTING_CHANNEL_PRESCALER	9	Frequency division ratio of channel 0 of TAU0 when XT1 oscillation clock is selected
XT1_LED_SETTING_CHANNEL_COUNT	64	Count value of channel 0 of TAU0 when XT1 oscillation clock is selected
SWITCH_OFF	0	Switch is not pressed
SWITCH_ON	1	Switch is pressed
SWITCH_ON_PORT_LEVEL	0	Input port level when switch is on
CLOCK_NOT_OSCILLATING	0	Clock is not oscillating.
CLOCK_OSCILLATING	1	Clock is oscillating.

5.5 List of Variables

Table 5.4 lists the global variables that are used in this sample program.

Table 5.4 Global Variable

Type	Variable Name	Contents	Function Used
uint8_t	g_ClockStatus	Clock status	main() R_INTC0_Interrupt
uint8_t	g_TAU0_Channel0_Clkdiv	Frequency division ratio of channel 0 of TAU0	main() R_TAU0_Channel0_GetParameter()
uint16_t	g_TAU0_Channel0_Count	Counter value of channel 0 of TAU0	main() R_TAU0_Channel0_GetParameter()
uint8_t	g_SwitchStatus	Switch status	main() R_INTC0_Interrupt

5.6 List of Functions

Table 5.5 lists the functions that are used in this sample program.

Table 5.5 Functions

Function Name	Outline
R_INTC0_Start	Sets start of INTP0 external interrupt processing.
R_TAU0_Channel0_Start	Sets start of channel 0 of TAU0.
R_CGC_ChangeClock	Switches clocks.
R_CGC_HOCOToX1	Switches from HOCO clock to X1 oscillation clock.
R_CGC_X1ToXT1	Switches from X1 oscillation clock to XT1 oscillation clock.
R_CGC_XT1ToHOCO	Switches from XT1 oscillation clock to HOCO clock.
R_CGC_GetClockStatus	Gets clock status.
R_CGC_Get_X1_Status	Gets X1 oscillation clock status.
R_CGC_Get_XT1_Status	Gets XT1 oscillation clock status.
R_CGC_Get_HOCO_Status	Gets HOCO clock status.
R_CGC_StopClock	Stops clock.
R_TAU0_Channel0_GetParameter	Gets parameters of channel 0 of TAU0.
R_TAU0_Channel0_Restart	Restarts channel 0 of TAU0.
R_TAU0_Channel0_ChangeInterval	Changes interval of channel 0 of TAU0.
R_TAU0_Channel0_Stop	Sets stop of channel 0 of TAU0.
R_TAU0_Channel0_Interrupt	Processes an interval timer interrupt of channel 0 of TAU0.
R_INTC0_Interrupt	Processes an INTP0 external interrupt.
R_IT_Wait_ms	Waits in units of 1 ms.

5.7 Function Specifications

This section describes the specifications for the functions that are used in the sample code.

[Function Name R_INTC0_Start

Synopsis	Sets start of INTP0 external interrupt processing.
Header	#include "r_cg_macrodriver.h" #include "r_cg_intc.h" #include "r_cg_userdefine.h"
Declaration	void R_INTC0_Start(void)
Explanation	Releases mask of INTP0 interrupts to enables them.
Arguments	None
Return value	None
Remarks	None

[Function Name] R_TAU0_Channel0_Start

Synopsis	Sets start of channel 0 of TAU0.
Header	#include "r_cg_macrodriver.h" #include "r_cg_timer.h" #include "r_cg_userdefine.h"
Declaration	void R_TAU0_Channel0_Start(void)
Explanation	Releases mask of INTP0 interrupts to start counting.
Arguments	None
Return value	None
Remarks	None

[Function Name] R_CGC_ChangeClock

Synopsis	Switches clocks.	
Header	#include "r_cg_macrodriver.h" #include "r_cg_cgc.h" #include "r_cg_userdefine.h"	
Declaration	void R_CGC_ChangeClock(uint8_t status)	
Explanation	Switches clocks.	
Arguments	• First argument: status	: Clock status (1 to 3) Set one of the following constants: HOCO_NEXT_STATUS_X1 → Switches to X1 oscillation clock. X1_NEXT_STATUS_XT1 → Switches to XT1 oscillation clock. XT1_NEXT_STATUS_HOCO → Switch to HOCO clock.
Return value	None	
Remarks	None	

[Function Name] R_CGC_HOCOToX1

Synopsis	Switches from HOCO clock to X1 oscillation clock.
Header	#include "r_cg_macrodriver.h" #include "r_cg_cgc.h" #include "r_cg_userdefine.h"
Declaration	void R_CGC_HOCOToX1 (void)
Explanation	Switches the CPU/peripheral hardware clock (f_{CLK}) from the HOCO clock to the X1 oscillation clock.
Arguments	None
Return value	None
Remarks	None

[Function Name] R_CGC_X1ToXT1

Synopsis	Switches from X1 oscillation clock to XT1 oscillation clock.
Header	#include "r_cg_macrodriver.h" #include "r_cg_cgc.h" #include "r_cg_userdefine.h"
Declaration	void R_CGC_X1ToXT1(void)
Explanation	Switches the CPU/peripheral hardware clock (f_{CLK}) from X1 oscillation clock to XT1 oscillation clock.
Arguments	None
Return value	None
Remarks	None

[Function Name] R_CGC_XT1ToHOCO

Synopsis	Switches from XT1 oscillation clock to HOCO clock.
Header	#include "r_cg_macrodriver.h" #include "r_cg_cgc.h" #include "r_cg_userdefine.h"
Declaration	void R_CGC_XT1ToHOCO(void)
Explanation	Switches the CPU/peripheral hardware clock (f_{CLK}) from XT1 oscillation clock to HOCO clock.
Arguments	None
Return value	None
Remarks	None

[Function Name] R_CGC_GetClockStatus

Synopsis	Gets clock status.	
Header	#include "r_cg_macrodriver.h" #include "r_cg_cgc.h" #include "r_cg_userdefine.h"	
Declaration	uint8_t R_CGC_GetClockStatus(uint8_t status)	
Explanation	Gets the clock status. Gets the status of the clock specified in the argument so that the user can check if the clock oscillates.	
Arguments	First argument: status	: Clock status (1 to 3) Set one of the following constants: X1_STATUS → Gets the status of X1 oscillation clock. XT1_STATUS → Gets the status of XT1 oscillation clock. HOCO_STATUS → Gets the status of HOCO clock.
Return value	• If the clock has not been switched: CLOCK_NOT_OSCILLATING (0x00) • If the clock has been switched: CLOCK_OSCILLATING (0x01)	
Remarks	None	