RX210 and RX21A Groups
MTU2 Complementary PWM Output Port Switching Using the MPC

Abstract
This application note describes a sample code that changes the output ports for a three-phase complementary PWM (pulse-width modulation) output with a non-overlapping relationship between the positive and negative phases generated by multifunction timer unit 2 (MTU2) using the multifunction pin controller (MPC) in the RX210 and RX21A Groups.

Products
RX210 and RX21A Groups

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.
Contents

1. Specifications .......................................................................................................................... 3
2. Confirmed Operating Condition ............................................................................................. 4
3. Reference Application Notes .................................................................................................. 4
4. Hardware .................................................................................................................................. 5
   4.1 Pins Used ........................................................................................................................... 5
5. Software ................................................................................................................................... 6
   5.1 Operational Overview .......................................................................................................... 6
   5.2 File Composition .................................................................................................................. 8
   5.3 Option Settings Memory ..................................................................................................... 8
   5.4 Constants .............................................................................................................................. 8
   5.5 Variables ............................................................................................................................. 9
   5.6 Functions ............................................................................................................................. 9
   5.7 Function Specifications ....................................................................................................... 10
   5.8 Flowcharts .......................................................................................................................... 12
      5.8.1 Main Processing ............................................................................................................. 12
      5.8.2 Peripheral Function Initialization ............................................................................... 12
      5.8.3 MTU2a Initialization ...................................................................................................... 13
      5.8.4 TGI3 Interrupt Handler ................................................................................................ 17
      5.8.5 Read Switch Input ........................................................................................................ 18
      5.8.6 Complementary PWM Output Port Switching ............................................................... 19
6. Applying This Application Note to the RX21A Group ............................................................... 21
7. Sample Code ............................................................................................................................. 22
8. Reference Documents ............................................................................................................... 22
1. Specifications

This sample program outputs a three-phase complementary PWM waveform using the MTU2a’s complementary PWM mode functions and switches the ports that output the complementary PWM waveform according to a switch input.

An active-low complementary PWM waveform is output from MTU2a channels 3 and 4. An interrupt is generated every output period, the duty is increased or decreased in the interrupt handler, and that adjustment is reflected in the output.

The sample program also reads a switch input signal from pin P34 and changes the complementary PWM output port when the input has been confirmed.

Table 1.1 lists the peripheral functions used and their applications and figure 1.1 presents an overview of this operation.

**Table 1.1 Peripheral Functions Used and Their Applications**

<table>
<thead>
<tr>
<th>Peripheral Function</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>P34</td>
<td>Reads the switch input signal</td>
</tr>
<tr>
<td>MTU2a</td>
<td>Generates pulse outputs</td>
</tr>
<tr>
<td>MPC</td>
<td>Changes a port for complementary PWM output from the MTU2a.</td>
</tr>
</tbody>
</table>

**Figure 1.1 Operational Overview**
2. Confirmed Operating Condition

Operation of the sample code in this application note has been verified under the following conditions.

Table 2.1 Confirmed Operating Condition

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microcontroller used</td>
<td>R5F52108ADF (RX210 Group)</td>
</tr>
<tr>
<td>Operating frequency</td>
<td>Main clock: 20.0 MHz&lt;br&gt;Sub-clock: 32.768 kHz&lt;br&gt;PLL: Main clock divided by 2 and multiplied by 10&lt;br&gt;System clock (ICLK): 50 MHz (PLL divided by 2)&lt;br&gt;Peripheral module clock B (PCLKB): 25 MHz (PLL divided by 4)</td>
</tr>
<tr>
<td>Operating voltage</td>
<td>5.0 V is supplied from E1.</td>
</tr>
<tr>
<td>Integrated development environment</td>
<td>Renesas Electronics Corporation&lt;br&gt;High-performance Embedded Workshop Version 4.09.00.007</td>
</tr>
<tr>
<td>C compiler</td>
<td>RX Family C/C++ Compiler V.1.02&lt;br&gt;-cpu=rx200 -output=obj=&quot;${CONFIGDIR}${FILELEAF}.obj&quot; -debug -nologo (The integrated development environment default settings are used.)</td>
</tr>
<tr>
<td>iodefine.h version</td>
<td>Version 1.0B</td>
</tr>
<tr>
<td>Endian order</td>
<td>Little endian</td>
</tr>
<tr>
<td>Operating mode</td>
<td>Single-chip mode</td>
</tr>
<tr>
<td>Processor mode</td>
<td>Supervisor mode</td>
</tr>
<tr>
<td>Sample code version</td>
<td>Version 1.00</td>
</tr>
<tr>
<td>Board used</td>
<td>Renesas Starter Kit for RX210 (Product number: R0K505210C000BE)</td>
</tr>
</tbody>
</table>

3. Reference Application Notes

For additional information associated with this document, refer to the following application notes.
- RX210 Group Initial Setting Rev. 1.00 (R01AN1002EJ)
- RX21A Group Initial Setting Rev. 1.10 (R01AN1486EJ)

The initial setting functions in the reference application notes are used in the sample code in this application note. The revision numbers of the reference application notes are current as of when this application note was made. However the latest version is always recommended. Visit the Renesas Electronics Corporation website to check and download the latest version.
4. Hardware

4.1 Pins Used

Table 4.1 lists the pins used and their functions.

Table 4.1 Pin Functions

<table>
<thead>
<tr>
<th>Pin</th>
<th>I/O</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>PB7/MTIOC3B</td>
<td>Output</td>
<td>Complementary PWM output</td>
</tr>
<tr>
<td>PB6/MTIOC3D</td>
<td>Output</td>
<td>Complementary PWM output</td>
</tr>
<tr>
<td>PE2/MTIOC4A</td>
<td>Output</td>
<td>Complementary PWM output</td>
</tr>
<tr>
<td>PE1/MTIOC4C</td>
<td>Output</td>
<td>Complementary PWM output</td>
</tr>
<tr>
<td>PE3/MTIOC4B</td>
<td>Output</td>
<td>Complementary PWM output</td>
</tr>
<tr>
<td>PE4/MTIOC4D</td>
<td>Output</td>
<td>Complementary PWM output</td>
</tr>
<tr>
<td>PC5/MTIOC3B</td>
<td>Output</td>
<td>Complementary PWM output</td>
</tr>
<tr>
<td>PC4/MTIOC3D</td>
<td>Output</td>
<td>Complementary PWM output</td>
</tr>
<tr>
<td>PB3/MTIOC4A</td>
<td>Output</td>
<td>Complementary PWM output</td>
</tr>
<tr>
<td>PB1/MTIOC4C</td>
<td>Output</td>
<td>Complementary PWM output</td>
</tr>
<tr>
<td>PC2/MTIOC4B</td>
<td>Output</td>
<td>Complementary PWM output</td>
</tr>
<tr>
<td>PC3/MTIOC4D</td>
<td>Output</td>
<td>Complementary PWM output</td>
</tr>
<tr>
<td>P34</td>
<td>Input</td>
<td>Complementary PWM output pin switching input (SW3 on RSK board)</td>
</tr>
</tbody>
</table>
5. Software

5.1 Operational Overview

During initialization, the sample program performs complementary PWM output from the MTIOC3B, MTIOC3D, MTIOC4A, MTIOC4C, MTIOC4B, and MTIOC4D pins using the MTU2a output. The output ports for these pins are set to PB7, PB6, PE2, PE1, PE3, and PE4, respectively.

The complementary PWM output is disabled at the time the SW3 input is confirmed on the RSK.

After the output is disabled, the complementary PWM output ports (MTIOC3B, MTIOC3D, MTIOC4A, MTIOC4C, MTIOC4B, and MTIOC4D) are switched to PC5, PC4, PB3, PB1, PC2, and PC3, respectively, and then complementary PWM output is enabled again.

After that, each time a new value of the SW3 input is confirmed, the output ports are switched alternately between one set (PC5, PC4, PB3, PB1, PC2, and PC3) and the other set (PB7, PB6, PE2, PE1, PE3, and PE4).

Switching ports is performed using the following procedure.

- MTU output is disabled.
- The port modes for the current MTU output pins are set to “not used as peripheral function pins”.
- The port functions for the current MTU output pins are set to the general-purpose I/O pin function.
- The port functions for the new MTU output pins are set to the peripheral function pins.
- The port modes for the new MTU output pins are set to MTU output pins.
- MTU output is enabled.
Figure 5.1 shows the timing chart for this operation.
5.2 File Composition

Table 5.1 lists the files used for the sample code. Note that the files generated automatically by the integrated development environment are not shown.

<table>
<thead>
<tr>
<th>File</th>
<th>Overview</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>main.c</td>
<td>Main processing</td>
<td></td>
</tr>
<tr>
<td>non_existent_port_init.c</td>
<td>Nonexistent port initialization</td>
<td></td>
</tr>
<tr>
<td>non_existent_port_init.h</td>
<td>External reference include header for nonexistent port initialization</td>
<td></td>
</tr>
<tr>
<td>clock_init.c</td>
<td>Clock initialization</td>
<td></td>
</tr>
<tr>
<td>clock_init.h</td>
<td>External reference include header for clock initialization</td>
<td></td>
</tr>
<tr>
<td>intprg.c</td>
<td>Interrupt handling</td>
<td></td>
</tr>
<tr>
<td>pwm.c</td>
<td>Sets up PWM output</td>
<td></td>
</tr>
</tbody>
</table>

5.3 Option Settings Memory

Table 5.2 lists the states of the option settings memory used by the sample code. Set these locations to appropriate values for your user system as required.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Address</th>
<th>Set value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFS0</td>
<td>FFFF FF8Fh – FFFF FF8Ch</td>
<td>FFFF FFFFh</td>
<td>Stops IWDT after a reset</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Stops WDT after a reset</td>
</tr>
<tr>
<td>OFS1</td>
<td>FFFF FF8Bh – FFFF FF88h</td>
<td>FFFF FFFFh</td>
<td>Disables voltage monitoring resets after a reset</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Disables HOCOC oscillation after a reset</td>
</tr>
<tr>
<td>MDES</td>
<td>FFFF FF83h – FFFF FF80h</td>
<td>FFFF FFFFh</td>
<td>Little endian (in single-chip mode)</td>
</tr>
</tbody>
</table>

5.4 Constants

Table 5.3 lists the constants used in the sample code.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Set Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOW</td>
<td>0</td>
<td>Output value for low</td>
</tr>
<tr>
<td>HIGH</td>
<td>1</td>
<td>Output value for high</td>
</tr>
<tr>
<td>OUT1</td>
<td>0</td>
<td>Complementary PWM output port setting: output 1</td>
</tr>
<tr>
<td>OUT2</td>
<td>1</td>
<td>Complementary PWM output port setting: output 2</td>
</tr>
<tr>
<td>DEAD_TIME</td>
<td>25</td>
<td>Dead time</td>
</tr>
<tr>
<td>CYCLE</td>
<td>1250</td>
<td>Carrier period</td>
</tr>
<tr>
<td>C_CYCLE</td>
<td>625</td>
<td>Carrier half period</td>
</tr>
<tr>
<td>PUL_CYCLE</td>
<td>650</td>
<td>Carrier half period + dead time</td>
</tr>
</tbody>
</table>
5.5 Variables

Table 5.4 lists the global variables.

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>unsigned char</td>
<td>sw3_fix</td>
<td>Confirmed value of the complementary PWM output port switching input</td>
<td>input_read, chg_out_port</td>
</tr>
<tr>
<td>unsigned char</td>
<td>c_loop</td>
<td>Time measurement counter used in main loop</td>
<td>main</td>
</tr>
<tr>
<td>int</td>
<td>Duty_select</td>
<td>Output duty increment/decrement direction</td>
<td>int_mtu2_tgia3</td>
</tr>
<tr>
<td>unsigned short</td>
<td>Pul_pwm_duty1</td>
<td>Set value for the complementary PWM output duty (MTIOC3B and MTIOC3D)</td>
<td>int_mtu2_tgia3</td>
</tr>
<tr>
<td>unsigned short</td>
<td>Pul_pwm_duty2</td>
<td>Set value for the complementary PWM output duty (MTIOC4A and MTIOC4C)</td>
<td>int_mtu2_tgia3</td>
</tr>
<tr>
<td>unsigned short</td>
<td>Pul_pwm_duty3</td>
<td>Set value for the complementary PWM output duty (MTIOC4B and MTIOC4D)</td>
<td>int_mtu2_tgia3</td>
</tr>
</tbody>
</table>

5.6 Functions

Table 5.5 lists the functions.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>Main processing</td>
</tr>
<tr>
<td>port_init</td>
<td>Port initialization</td>
</tr>
<tr>
<td>non_existent_port_init</td>
<td>Nonexistent port initialization</td>
</tr>
<tr>
<td>clock_init</td>
<td>Clock initialization</td>
</tr>
<tr>
<td>peripheral_init</td>
<td>Peripheral function initialization</td>
</tr>
<tr>
<td>mtu2_init</td>
<td>MTU2a initialization</td>
</tr>
<tr>
<td>int_mtu2_tgia3</td>
<td>TGIA3 interrupt handler</td>
</tr>
<tr>
<td>input_read</td>
<td>Reads the switch input</td>
</tr>
<tr>
<td>chg_out_port</td>
<td>Switches the complementary PWM output ports.</td>
</tr>
</tbody>
</table>
## 5.7 Function Specifications

This section lists the specifications of the functions in the sample code.

### main

<table>
<thead>
<tr>
<th>Overview</th>
<th>Main processing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header</td>
<td>None</td>
</tr>
<tr>
<td>Declaration</td>
<td>void main(void)</td>
</tr>
<tr>
<td>Description</td>
<td>Initializes the SFRs and peripheral functions. Reads the switch input and switches the complementary PWM output ports every 5 ms.</td>
</tr>
<tr>
<td>Arguments</td>
<td>None</td>
</tr>
<tr>
<td>Return values</td>
<td>None</td>
</tr>
</tbody>
</table>

### non_existen_port_init

<table>
<thead>
<tr>
<th>Overview</th>
<th>Nonexistent port initialization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header</td>
<td>non_existen_port_init.h</td>
</tr>
<tr>
<td>Declaration</td>
<td>void non_existen_port_init(void)</td>
</tr>
<tr>
<td>Description</td>
<td>Initializes the nonexistent ports.</td>
</tr>
<tr>
<td>Arguments</td>
<td>None</td>
</tr>
<tr>
<td>Return values</td>
<td>None</td>
</tr>
<tr>
<td>Remarks</td>
<td>For details on this function, refer to the Initial Setting application note for the product used. There are cases where ports that do not exist must be initialized due to the number of pins in the particular microcontroller product used. This processing is not required for products that include the RSK used by this system.</td>
</tr>
</tbody>
</table>

### clock_init

<table>
<thead>
<tr>
<th>Overview</th>
<th>Clock initialization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header</td>
<td>clock_init.h</td>
</tr>
<tr>
<td>Declaration</td>
<td>void clock_init(void)</td>
</tr>
<tr>
<td>Description</td>
<td>Initializes the clocks.</td>
</tr>
<tr>
<td>Arguments</td>
<td>None</td>
</tr>
<tr>
<td>Return values</td>
<td>None</td>
</tr>
<tr>
<td>Remarks</td>
<td>For details on this function, refer to the Initial Setting application note for the product used.</td>
</tr>
</tbody>
</table>

### peripheral_init

<table>
<thead>
<tr>
<th>Overview</th>
<th>Peripheral function initialization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header</td>
<td>None</td>
</tr>
<tr>
<td>Declaration</td>
<td>void peripheral_init(void)</td>
</tr>
<tr>
<td>Description</td>
<td>Initializes the used peripheral functions.</td>
</tr>
<tr>
<td>Arguments</td>
<td>None</td>
</tr>
<tr>
<td>Return values</td>
<td>None</td>
</tr>
</tbody>
</table>
### mtu2_init

**Overview**
MTU2a initialization

**Header**
None

**Declaration**
void mtu2_init(void)

**Description**
Sets the complementary PWM output ports to high-level output.
Sets channels 3 and 4 to complementary PWM mode 3.
Sets the output period to 200 µs and the dead time to 4 µs.
Sets MTU3.TGRD, MTU4.TGRC, and MTU4.TGRD to be the buffer registers for MTU3.TGRB, MTU4.TGRA, and MTU4.TGRB, respectively.

**Arguments**
None

**Return values**
None

### int_mtu2_tgia3

**Overview**
TGIA3 interrupt handler

**Header**
None

**Declaration**
void int_mtu2_tgia3(void)

**Description**
Sets the output duty increase/decrease direction and increments or decrements the duty by 1.

**Arguments**
None

**Return values**
None

### input_read

**Overview**
Read switch input

**Header**
None

**Declaration**
void input_read(void)

**Description**
Reads the input state of the SW3 on the RSK and confirms the input value if the input level matches four times consecutively.

**Arguments**
None

**Return values**
None

### chg_out_port

**Overview**
Change complementary PWM output ports

**Header**
None

**Declaration**
void chg_out_port(void)

**Description**
Switches the complementary PWM output ports.

**Arguments**
None

**Return values**
None
5.8 Flowcharts

5.8.1 Main Processing

Figure 5.2 shows the flowchart for the main processing.

- **main**
- Disable maskable interrupts
- Initialize non-existent ports `non_existent_port_init`
- Initialize clocks `clock_init`
- Initialize peripheral functions `peripheral_init`
- Enable maskable interrupts

- **Has 5 ms elapsed?**
  - **No**
  - Clear main loop time measurement counter
  - Read switch input `input_read`
  - Switch complementary PWM output ports `chg_out_port`
  - Sets up the clocks.
  - Reads SW3 input on the RSK and confirms the input level.
  - Changes the complementary PWM output ports.
  - **Yes**

Figure 5.2 Main Processing

5.8.2 Peripheral Function Initialization

Figure 5.3 shows the flowchart for peripheral function initialization.

- **peripheral_init**
- Initializes MTU2 `mtu_init`
- return

Figure 5.3 Peripheral Function Initialization
5.8.3 MTU2a Initialization

Figure 5.4 to 5.7 show the flowchart for MTU2a initialization.

```
mtu_init

Set register protection

----- PRCR register ← 0xA500
  PRC0 bit = 0: Disables writes to the clock generator circuit related registers
  PRC1 bit = 1: Enables writes to the operating mode, power saving functions,
  and software reset related registers.
  PRC2 bit = 0: Disables writes to location 0008 0200h.
  PRC3 bit = 0: Disables writes to the LVD related registers.
  Bits 7 to 4: These bits must be set to 0.
  PRKEY bits = A5h: Enables writes to bits PRC0 to PRC3.

Clear MTU2 module stop state

----- MSTPA register
  MSTPA9 = 0: Clears the MTU module stop state.

Set register protection

----- PRCR register ← 0xA500
  PRC0 bit = 0: Disables writes to the clock generator circuit related registers
  PRC1 bit = 1: Enables writes to the operating mode, power saving
  functions, and software reset related registers.
  PRC2 bit = 0: Disables writes to location 0008 0200h.
  PRC3 bit = 0: Disables writes to the LVD related registers.
  Bits 7 to 4: These bits must be set to 0.
  PRKEY bits = A5h: Enables writes to bits PRC0 to PRC3.

Stop counter

----- MTU.TSTR register ← 0x00
  CST3 bit = 0: Stops the MTU3.TCNT count operation.
  CST4 bit = 0: Stops the MTU4.TCNT count operation.
```

Figure 5.4 MTU2a Initialization (1/4)
Set up output ports

---

PORTB.PODR register ← 0xCA
B0 bit = 0: Sets PB0 to 0 output.
B1 bit = 1: Sets PB1 to 1 output.
B2 bit = 0: Sets PB2 to 0 output.
B3 bit = 1: Sets PB3 to 1 output.
B4 bit = 0: Sets PB4 to 0 output.
B5 bit = 0: Sets PB5 to 0 output.
B6 bit = 1: Sets PB6 to 1 output.
B7 bit = 1: Sets PB7 to 1 output.

PORTE.PODR register ← 0x1E
B0 bit = 0: Sets PE0 to 0 output.
B1 bit = 1: Sets PE1 to 1 output.
B2 bit = 1: Sets PE2 to 1 output.
B3 bit = 1: Sets PE3 to 1 output.
B4 bit = 1: Sets PE4 to 1 output.
B5 bit = 0: Sets PE5 to 0 output.
B6 bit = 0: Sets PE6 to 0 output.
B7 bit = 0: Sets PE7 to 0 output.

PORTC.PODR register ← 0x3C
B0 bit = 0: Sets PC0 to 0 output.
B1 bit = 0: Sets PC1 to 0 output.
B2 bit = 1: Sets PC2 to 1 output.
B3 bit = 1: Sets PC3 to 1 output.
B4 bit = 1: Sets PC4 to 1 output.
B5 bit = 1: Sets PC5 to 1 output.
B6 bit = 0: Sets PC6 to 0 output.
B7 bit = 0: Sets PC7 to 0 output.

PORTB.PDR register ← 0xCA
B0 bit = 0: Sets PB0 to input (functions as an input port)
B1 bit = 1: Sets PB1 to output (functions as an output port)
B2 bit = 0: Sets PB2 to input (functions as an input port)
B3 bit = 1: Sets PB3 to output (functions as an output port)
B4 bit = 0: Sets PB4 to input (functions as an input port)
B5 bit = 0: Sets PB5 to input (functions as an input port)
B6 bit = 1: Sets PB6 to output (functions as an output port)
B7 bit = 1: Sets PB7 to output (functions as an output port)

PORTE.PDR register ← 0x1E
B0 bit = 0: Sets PE0 to input (functions as an input port)
B1 bit = 1: Sets PE1 to output (functions as an output port)
B2 bit = 1: Sets PE2 to output (functions as an output port)
B3 bit = 1: Sets PE3 to output (functions as an output port)
B4 bit = 1: Sets PE4 to output (functions as an output port)
B5 bit = 1: Sets PE5 to input (functions as an input port)
B6 bit = 0: Sets PE6 to input (functions as an input port)
B7 bit = 0: Sets PE7 to input (functions as an input port)

PORTC.PDR register ← 0x3C
B0 bit = 0: Sets PC0 to input (functions as an input port)
B1 bit = 0: Sets PC1 to input (functions as an input port)
B2 bit = 1: Sets PC2 to output (functions as an output port)
B3 bit = 1: Sets PC3 to output (functions as an output port)
B4 bit = 1: Sets PC4 to output (functions as an output port)
B5 bit = 1: Sets PC5 to output (functions as an output port)
B6 bit = 0: Sets PC6 to input (functions as an input port)
B7 bit = 0: Sets PC7 to input (functions as an input port)
Figure 5.6 MTU2a Initialization (3/4)
Set up output control

- TOCR1 register ← 0x40
  - OLSP bit = 0: Initial output = high, active level = low,
    compare match output: when incrementing = high,
    when decrementing = low.
  - OLSN bit = 0: Initial output = high, active level = low,
    compare match output: when incrementing = low,
    when decrementing = high.
  - TOCS bit = 0: Enables the TOCR1 settings.
  - TOCL bit = 0: Enables writing to the TOCS, OLSN, and OLSP bits.
  - Bits 5 to 4: These bits must be set to 0.
  - PSYE bit = 1: Enables toggle output
  - Bit 7: This bit must be set to 0.

Set timer mode

- MTU3.TMDR register ← 0x3F
  - MD bit = 1111b: Complementary PWM mode 3
    (transfer at both crest and trough)
  - BFA bit = 1: The TGRA and TGRC registers operate as buffers.
  - BFB bit = 1: The TGRB and TGRD registers operate as buffers.
  - BFE bit = 0: MTU0.TGRE and MTU0.TGRF operate in normal mode.
  - Bit 7: This bit must be set to 0.

Enable outputs

- TOER register ← 0xFF
  - OE3B bit = 1: Enables MTU output.
  - OE4A bit = 1: Enables MTU output.
  - OE4B bit = 1: Enables MTU output.
  - OE3D bit = 1: Enables MTU output.
  - OE4C bit = 1: Enables MTU output.
  - OE4D bit = 1: Enables MTU output.
  - Bits 7 to 6: These bits must be set to 1.

Enable interrupts

- IPR129 register
  - IPR bit ← 0001b: Sets the MTU3 (TGIA3, TGIB3, TGIC3, and TGID3) interrupt level to 1.
- IER10 register
- IEN1 bit ← 1: Enables the TGIA3 interrupt request.
- MTU3.TIER register
- TGIEA bit ← 1: Enables interrupt requests (TGIA).

Start timer

- MTU.TSTR register ← 0xC0
  - CST0 bit = 0: Stops the MTU0.TCNT count.
  - CST1 bit = 0: Stops the MTU1.TCNT count.
  - CST2 bit = 0: Stops the MTU2.TCNT count.
  - Bits 5 to 3: These bits must be set to 0.
  - CST3 bit = 1: Sets the MTU3.TCNT count to operate.
  - CST4 bit = 1: Sets the MTU4.TCNT count to operate.

return
5.8.4 TGIA3 Interrupt Handler

Figure 5.8 shows the flowchart for the TGIA3 interrupt handler.

![Flowchart for TGIA3 Interrupt Handler]

- **int_mtu2_tgia3**
- Increment 5 ms counter by 1
- Is output duty value 0 or larger?
  - Yes
  - Output duty value ≥ PWM period?
    - Yes
    - Increment duty
    - No
    - Decrement duty
  - No
  - Increment output duty value
  - Decrement output duty value
- Is output duty increment/decrement direction increment?
  - Yes
  - Increment output duty value
  - No
  - Decrement output duty value
- Set buffer register to output duty value
- return

Measurement of the 5 ms period in the main loop.

Sets the increment/decrement state for the PWM duty setting. If the duty is decremented to 0, set to increment, and if it is incremented to the PWM period, set to decrement.

Increment by 1 if the output duty increment/decrement direction is increment. Decrement by 1 if the output duty increment/decrement direction is decrement.

Figure 5.8 TGIA3 Interrupt Handler
5.8.5 Read Switch Input

Figure 5.9 shows the flowchart for reading switch input.

```
input_read

Read input level

Do current and previous input levels match?
  No
  Yes

Did input levels match 3 times consecutively?
  No
  Yes

Is consecutive input level matches plus 1 ≥ 3 times?
  No
  Yes

Input level = 0?
  No
  Yes

Was previous confirmed input level = 1?
  No
  Yes

Set previous confirmed input level = 1

Set input confirmed flag

Update previous input level

return
```

Stores the P34 input level as the current input level in RAM.

Stores the input level for confirming a match at the next period.

Increments the consecutive input level match counter each time the current input level matches the previous input level.

Clears the count if the input levels do not match.

The input level is confirmed if the input levels match three times.

If the confirmed value is 0, a switch input is only confirmed if the previous confirmed input level was 1.

If the confirmed value is 1, the previous confirmed input level is updated to be 1.

Figure 5.9 Reading Switch Input
5.8.6 Complementary PWM Output Port Switching

Figure 5.10 and 5.11 shows the flowchart for complementary PWM output port switching.

The input confirmation flag is only used for one period, so it is cleared immediately after determination.

- **PORTB.PMR register**
  - B6 bit ← 0: Sets PB6 to be used as a general-purpose I/O port
  - B7 bit ← 0: Sets PB7 to be used as a general-purpose I/O port
- **PORTE.PMR register**
  - B1 bit ← 0: Sets PE1 to be used as a general-purpose I/O port
  - B2 bit ← 0: Sets PE2 to be used as a general-purpose I/O port
  - B3 bit ← 0: Sets PE3 to be used as a general-purpose I/O port
  - B4 bit ← 0: Sets PE4 to be used as a general-purpose I/O port
- **PWPR register**
  - B0WI ← 0: Enables writing to the PFSWE bit.
  - PWPR register
    - PFSWE ← 1: Enables writing to the PFS register.
- **PBF6FS register**
  - PSEL bit ← 0000: Sets PB6 not to be used as a general-purpose I/O port
  - PB7PFS register
  - PSEL bit ← 0000: Sets PB7 not to be used as a general-purpose I/O port
- **PE1PFS register**
  - PSEL bit ← 0001: Sets PE1 not to be used as a general-purpose I/O port
  
- **PE2PFS register**
  - PSEL bit ← 0000: Sets PE2 not to be used as a general-purpose I/O port
- **PE3PFS register**
  - PSEL bit ← 0000: Sets PE3 not to be used as a general-purpose I/O port
  
- **PE4PFS register**
  - PSEL bit ← 0000: Sets PE4 not to be used as a general-purpose I/O port
- **PC5PFS register**
  - PSEL bit ← 0000: Sets PC5 not to be used as a general-purpose I/O port
  
- **PC4PFS register**
  - PSEL bit ← 0001: Uses PC5 as the MTIOC3B pin
  - PC4PFS register
  - PSEL bit ← 0001: Uses PC4 as the MTIOC3D pin
- **PB3PFS register**
  - PSEL bit ← 0010: Uses PB3 as the MTIOC4A pin
- **PB1PFS register**
  - PSEL bit ← 0010: Uses PB1 as the MTIOC4C pin
  
- **PC2PFS register**
  - PSEL bit ← 0001: Uses PC2 as the MTIOC4B pin
- **PC3PFS register**
  - PSEL bit ← 0001: Uses PC3 as the MTIOC4D pin
- **PWPR register**
  - PFSWE ← 0: Enables writing to the PFS register.
  - PWPR register
  - B0WI ← 1: Enables writing to the PFSWE bit.
- **PORTB.PMR register**
  - B1 bit = 1: Sets PB1 to be used as a peripheral function pin.
  - B3 bit = 1: Sets PB3 to be used as a peripheral function pin.
- **PORTC.PMR register**
  - B2 bit = 1: Sets PC2 to be used as a peripheral function pin.
  - B4 bit = 1: Sets PC4 to be used as a peripheral function pin.
  - B5 bit = 1: Sets PC5 to be used as a peripheral function pin.
Set complementary PWM output port setting to output 1

Set complementary PWM output ports to general-purpose I/O

Clear PFS register protection

Change port functions of the MTU2 output pins

Set PFS register protection

Set complementary PWM output ports as peripheral function pins

--- PORTB.PMR register
  B1 bit ← 0: Sets PB1 to be used as a general-purpose I/O port
  B3 bit ← 0: Sets PB3 to be used as a general-purpose I/O port

PORTC.PMR register
  B2 bit ← 0: Sets PC2 to be used as a general-purpose I/O port
  B3 bit ← 0: Sets PC3 to be used as a general-purpose I/O port
  B4 bit ← 0: Sets PC4 to be used as a general-purpose I/O port
  B5 bit ← 0: Sets PC5 to be used as a general-purpose I/O port

--- PWPR register
  B0WI ← 0: Enables writing to the PFSWE bit.
  PWPR register
  PFSEWE ← 1: Enables writing to the PFS register.

--- PC5PFS register
  PSEL bit ← 0000: Sets PC5 not to be used as a general-purpose I/O port
  PC4PFS register
  PSEL bit ← 0000: Sets PC4 not to be used as a general-purpose I/O port
  PB3PFS register
  PSEL bit ← 0000: Sets PB3 not to be used as a general-purpose I/O port
  PB1PFS register
  PSEL bit ← 0000: Sets PB1 not to be used as a general-purpose I/O port
  PC2PFS register
  PSEL bit ← 0000: Sets PC2 not to be used as a general-purpose I/O port
  PC3PFS register
  PSEL bit ← 0000: Sets PC3 not to be used as a general-purpose I/O port
  PB7PFS register
  PSEL bit ← 0001: Uses PB7 as the MTIOC3B
  PB6PFS register
  PSEL bit ← 0001: Uses PB6 as the MTIOC3D
  PE2PFS register
  PSEL bit ← 0001: Uses PE2 as the MTIOC4A
  PE1PFS register
  PSEL bit ← 0001: Uses PE1 as the MTIOC4C
  PE3PFS register
  PSEL bit ← 0001: Uses PE3 as the MTIOC4B
  PE4PFS register
  PSEL bit ← 0001: Uses PE4 as the MTIOC4D

--- PWPR register
  PFSWE ← 0: Enables writing to the PFS register.
  PWPR register
  B0WI ← 1: Enables writing to the PFSWE bit.

--- PORTB.PMR register
  B6 bit ← 1: Sets PB6 to be used as a peripheral function pin
  B7 bit ← 1: Sets PB7 to be used as a peripheral function pin

PORTE.PMR register
  B1 bit ← 1: Sets PE1 to be used as a peripheral function pin
  B2 bit ← 1: Sets PE2 to be used as a peripheral function pin
  B3 bit ← 1: Sets PE3 to be used as a peripheral function pin
  B4 bit ← 1: Sets PE4 to be used as a peripheral function pin

--- PWPR register
  B0WI ← 0: Enables writing to the PFSWE bit.

--- PWPR register
  PWPR register
  B0WI ← 1: Enables writing to the PFSWE bit.

--- PORTB.PMR register
  B6 bit ← 1: Sets PB6 to be used as a peripheral function pin
  B7 bit ← 1: Sets PB7 to be used as a peripheral function pin

Figure 5.11  Complementary PWM Output Port Switching (2/2)
6. Applying This Application Note to the RX21A Group

The sample code accompanying this application note has been confirmed to operate with the RX210 Group. To make the sample code operate with the RX21A Group, use this application note in conjunction with the RX21A Initial Setting application note.

To use this application note with the RX21A Group, modify the main.c and pwm.c files accompanying this application note as shown steps (1) to (5) below, and then refer to “5. Applying the RX200 Series Application Note to the RX21A Group” in the RX21A Group Initial Setting application note.

1) Change the #include for “iodefine.h” to “../iodefine.h” in main.c and pwm.c.
2) Add a #include for “r_init_stop_module.h” in main.c.
3) In main.c, change the #includes for “clock_init.h” and “non_existent_port_init.h” to “r_init_clock.h” and “r_init_non_existent_port.h”, respectively.

```c
11 /* **************************************************
12 #include <machine.h>
13 #include “../iodefine.h”
14 #include “r_init_clock.h”
15 #include “r_init_stop_module.h”
16 #include “r_init_non_existent_port.h”
17 */
```

4) In main.c, add a call for the R_INIT_StopModule() function in the main function.

5) In main.c, change the calls for “non_existent_port_init()” and “clock_init()” in the main function to calls for “R_INIT_NonExistentPort()” and “R_INIT_Clock()”, respectively.

```c
40 * Return Value : none
41 * "FUNC COMMENT END"x *********************************************/
42 void main(void)
43 {
44  /* ---- Disable maskable interrupts ---- */
45  clrpsw_();
46  
47  R_INIT_StopModule();  
48  /* ---- Initialize non-existent ports ---- */
49  R_INIT_NonExistentPort();
50  /* ---- Initialize the clock ---- */
51  R_INIT_Clock();
52  /* peripheral initialize */
53  peripheral_init();
54  /* ---- Disable maskable interrupts ---- */
55  setpsw_();
56  
57  while(1){
58    while(c_loop <= 25);  /* 5ms wait */
59    c_loop = 0;
60    input_read();  /* read input information */
61    check_out_port();  /* change output port */
62  }
63 }
```
7. Sample Code
Sample code can be downloaded from the Renesas Electronics website.

8. Reference Documents
User’s Manual: Hardware
   RX210 Group User's Manual: Hardware Rev.1.50 (R01UH0037EJ)
   RX21A Group User's Manual: Hardware Rev.1.00 (R01UH0251EJ)
   The latest versions can be downloaded from the Renesas Electronics website.

Technical updates and technical news
   (The latest technical information can be downloaded from the Renesas Electronics Corporation website.)

C compiler manual
   RX210 C Compiler Package, Version 1.02
   C Compiler User’s Manual, Revision 1.00
   (Download the latest version of this manual from the Renesas Electronics Corporation website.)

Website and Support
Renesas Electronics website
   http://www.renesas.com

Inquiries
   http://www.renesas.com/contact/
<table>
<thead>
<tr>
<th>Rev.</th>
<th>Date</th>
<th>Page</th>
<th>Description</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00</td>
<td>Sep. 25, 2012</td>
<td>—</td>
<td>First edition issued</td>
<td></td>
</tr>
<tr>
<td>1.01</td>
<td>July 1, 2014</td>
<td>1</td>
<td>Products: Added the RX21A and RX220 Groups.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>3. Reference Application Notes: Added the Initial Setting application notes for the RX21A and RX220 Groups.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>Modified the description of reference application note in the following functions: non_existent_port_init and clock_init</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>21</td>
<td>6. Applying This Application Note to the RX21A or RX220 Group: Added.</td>
<td></td>
</tr>
<tr>
<td>1.02</td>
<td>Oct. 1, 2014</td>
<td>—</td>
<td>Deleted the RX220 Group from the target products. (Refer to the &quot;RX220 Group Application Example of Exclusive Operation of Two Motors by One Set of Complementary PWM Outputs Rev.1.00&quot; application note (R01AN2234JJ) for the RX220 Group. It also provides the sample code.)</td>
<td></td>
</tr>
</tbody>
</table>

All trademarks and registered trademarks are the property of their respective owners.
General Precautions in the Handling of MPU/MCU Products

The following usage notes are applicable to all MPU/MCU 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. 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 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. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on
   The state of the product is undefined at the moment 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 moment 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 moment 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 moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses
   Access to reserved addresses is prohibited.
   - The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals
   After applying a reset, only release the reset line after the operating clock signal has become stable.
   When switching the clock signal during program execution, wait until the target clock signal has 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.
     Moreover, 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.

5. Differences between Products
   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.
   - The characteristics of an MPU or MCU in the same group but having a different part number may differ in terms of the 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.