To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

GRADE A

MESC TECHNICAL NEWS

No.M7200-06-9804

Note on Using Multi-master I²C-BUS Interface of 7200 Series (REV.B)

When using multi-master I²C-BUS interface function of TV microcomputer 7200 series, be sure to note the following.

■Precautions when using multi-master I²C-BUS interface

(1) Read-modify-write instruction

The precautions when the read-modify-write instruction such as SEB, CLB etc. is executed for each register of the multi-master I²C-BUS interface are described below.

- I²C data shift register (S0)
  When executing the read-modify-write instruction for this register during transfer, data may become a value not intended.
- I²C address register (S0D)
  When the read-modify-write instruction is executed for this register at detecting the STOP condition, data may become a value not intended. It is because hardware changes the read/write bit (RBW) at the above timing.
- I²C status register (S1)
  Do not execute the read-modify-write instruction for this register because all bits of this register are changed by hardware.
- I²C control register (S1D)
  When the read-modify-write instruction is executed for this register at detecting the START condition or at completing the byte transfer, data may become a value not intended. Because hardware changes the bit counter (BC0–BC2) at the above timing.
- I²C clock control register (S2)
  The read-modify-write instruction can be executed for this register.

(2) START condition generating procedure using multi-master

①Procedure example (The necessary conditions of the generating procedure are described as the following ② to ⑤).

```
          :
          :
      LDA   —              (Taking out of slave address value)
      SEI                  (Interrupt disabled)
      BBS   5,S1,BUSBUSY   (BB flag confirming and branch process)
BUSFREE:
      STA   S0             (Writing of slave address value)
      LDM   #$F0, S1       (Trigger of START condition generating)
      CLI                  (Interrupt enabled)
          :
          :
BUSBUSY:
      CLI                  (Interrupt enabled)
          :
          :
```

②Use “STA,” “STX” or “STY” of the zero page addressing instruction for writing the slave address value to the I²C data shift register.

③Use “LDM” instruction for setting trigger of START condition generating.

④Write the slave address value of above ② and set trigger of START condition generating of above ③ continuously shown the above procedure example.

⑤Disable interrupts during the following three process steps:

- BB flag confirming
- Writing of slave address value
- Trigger of START condition generating

When the condition of the BB flag is bus busy, enable interrupts immediately.

(3) RESTART condition generating procedure

①Procedure example (The necessary conditions of the generating procedure are described as the following ② to ⑥.)
Execute the following procedure when the PIN bit is “0.”

```
          :
          :
LDM   #$00, S1       (Select slave receive mode)
LDA   —              (Taking out of slave address value)
SEI                  (Interrupt disabled)
STA   S0             (Writing of slave address value)
LDM   #$F0, S1       (Trigger of RESTART condition generating)
CLI                  (Interrupt enabled)
          :
          :
```

②Select the slave receive mode when the PIN bit is “0.” Do not write “1” to the PIN bit. Neither “0” nor “1” is specified for the writing to the BB bit. The TRX bit becomes “0” and the SDA pin is released.

③The SCL pin is released by writing the slave address value to the I²C data shift register. Use “STA,” “STX” or “STY” of the zero page addressing instruction for writing.

④Use “LDM” instruction for setting trigger of RESTART condition generating.

⑤Write the slave address value of above ③ and set trigger of RESTART condition generating of above ④ continuously shown the above procedure example.

⑥Disable interrupts during the following two process steps:

- Writing of slave address value
- Trigger of RESTART condition generating

(4) STOP condition generating procedure

①Procedure example (The necessary conditions of the generating procedure are described as the following ② to ④.)

```
          :
          :
SEI                  (Interrupt disabled)
LDM   #$C0, S1       (Select master transmit mode)
NOP                  (Set NOP)
LDM   #$D0, S1       (Trigger of STOP condition generating)
CLI                  (Interrupt enabled)
          :
          :
```

②Write “0” to the PIN bit when master transmit mode is select.

③Execute “NOP” instruction after setting of master transmit mode. Also, set trigger of STOP condition generating within 10 cycles after selecting of master trasmit mode.

④Disable interrupts during the following two process steps:

- Select of master transmit mode
- Trigger of STOP condition generating

(5) Writing to I²C status register

Do not execute an instruction to set the PIN bit to “1” from “0” and an instruction to set the MST and TRX bits to “0” from “1” simultaneously. It is because it may enter the state that the SCL pin is released and the SDA pin is released after about one machine cycle. Do not execute an instruction to set the MST and TRX bits to “0” from “1” simultaneously when the PIN bit is “1.” It is because it may become the same as above.

(6) Process of after STOP condition generating

Do not write data in the I²C data shift register S0 and the I²C status register S1 until the bus busy flag BB becomes “0” after generating the STOP condition in the master mode. It is because the STOP condition waveform might not be normally generated. Reading to the above registers do not have the problem.