RL78/F13, F14, F15

SENT (Single Edge Nibble Transmission) Communication

Introduction

This application note describes setup procedures and example implementations of Single Edge Nibble Transmission (hereinafter, “SENT”) Communication for the RL78/F13, F14, F15 (hereinafter, “RL78/F1x”). The explanations are for both SENT transmission and SENT reception.

Under certain use conditions, the operations of the microcontroller might be different from examples that this document provides. Customers are required to sufficiently evaluate the use of the SENT in their environment. Customers are also required to refer to the user's manual corresponding to their products for detailed functions of the SENT, clock generator, and interrupts.

Target Devices

- RL78/F13, F14, F15 products

Contents

1. Summary of SENT Implementations for RL78/F1x

This section summarizes specifications about the SENT example implementations for RL78/F1x.

1.1 Used RL78/F1x Resources for this SENT Communication

Table 1-1 summarizes RL78/F1x hardware functions and the settings for SENT transmission. Timer array unit (TAU00 and 01) is used to generate PWM waveforms, and Data Transfer Controller (DTC) transfers pulse signal length values according to each transmission data. The waveform signal is output to P30 / TO01 port.

Table 1-1 Used RL78/F1x Resources for SENT Transmission

Item		Description
CPU/peripheral hardware clock frequency (f_{CLK})		32 MHz
Used hardware resources		TAU00/01+DTC: PWM generation - TAU00: PWM mode (Master: Interval timer mode) - TAU01: PWM mode (Slave: One-count mode) - DTC: Source: RAM, Destination: TDR00, Trigger: INTTM01 CRC: “SENT” mode, J2716 standard.
Output port		P30 / TO01
TAU0 operation clock frequency	TAU00/01	f_{CLK}/32 = 1 MHz

Table 1-2 summarizes RL78/F1x hardware functions and the settings for SENT reception. Timer array unit (TAU02) is used to measure wave pulse signal length input from P16 / TI02 port.

Table 1-2 Used RL78/F1x Resources for SENT Reception

Item		Description
CPU/peripheral hardware clock frequency (f_{CLK})		32 MHz
Used hardware resources		TAU02: Input pulse interval measurement - TAU02: Input pulse capture mode, falling edge to next falling edge. CRC: “SENT” mode, J2716 standard.
Input port		P16 / TI02
TAU0 count clock frequency	TAU02	f_{CLK} = 32 MHz

1.2 Specification of SENT Communication Example

Table 1-3 shows specifications about this SENT communication example. This example uses fixed 6 data nibbles with automatic nibble calibration. A SENT message is 32 bit (8 nibbles) and consists of the following components: Sync period, Status nibble, 6 data nibbles, CRC nibble, and Pause pulse. This example also supports CRC errors detection.

Table 1-3 Specification of SENT Communication Example

Item	Setting
1 Tick	3 μs (typically)
Low level width of each pulse	5 ticks (15 μs typically)
Width of Sync period	56 ticks (168 μs typically) *Allowed within 160-216 μs (-5% - +29%). Calibrate tick width with the detected Sync width.
Width of each nibble	12-27 ticks (36-81 μs typically)
Status nibble	Open for users.
Number of data nibbles	6 data nibbles (fixed).
CRC nibble	J2716 standard.
Width of a frame	284-920 ticks (852-2760 μs typically)
Pause pulse	Supported: 12-768 ticks (36-2304 μs typically) *Calculate Pause pulse width for each frame transmission based on “Def_SENT_FrameWidth” which is defined at r_sent_tx_user.h.
Error detections during execution	- CRC error

1.3 SENT Waveform

Typical SENT waveform is shown in Figure 1-1, and the real waveform generated by this SENT transmission example is in Figure 1-2. The Pause pulse adjusts the constant frame width for each transmission units.

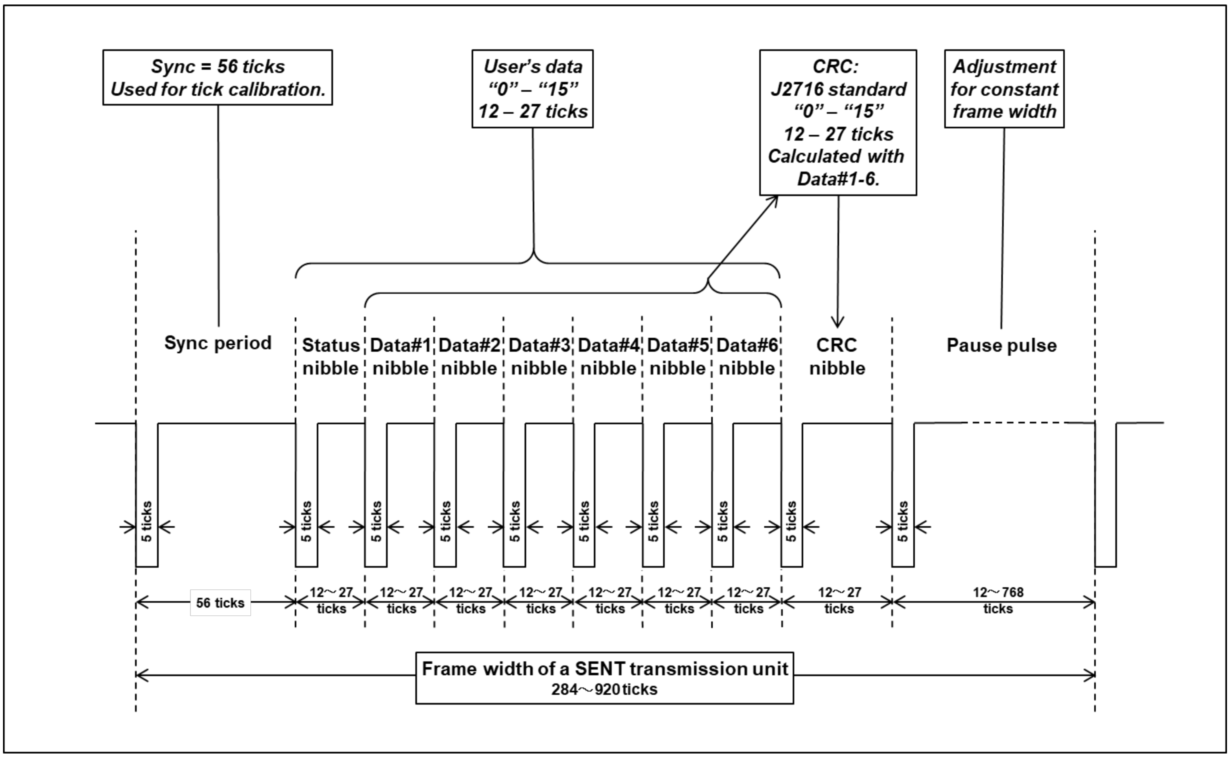

Figure 1-1 SENT Waveform

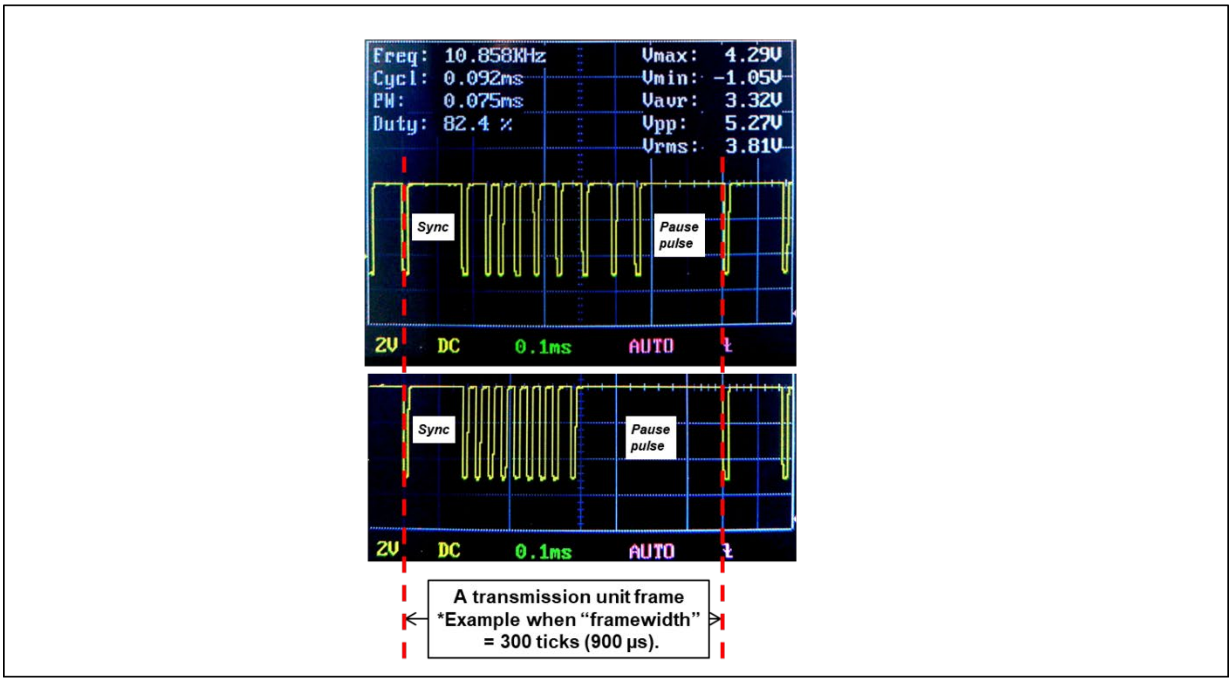

Figure 1-2 Real SENT Waveforms Generated by this Example

1.4 CRC Calculation for SENT Communication

RL78/F1x hardware supports CRC operation function for SENT communication. This example uses the RL78/F1x CRC operation function for SENT CRC calculation.

Figure 1-3 is example source list of CRC calculation for SENT communication. The calculation procedure is according to the J2716 standard. The CRC is calculated as 4 bit data and the inputs are the values of Data[#1 to #6] nibbles, as shown in Figure 1-1.

```
static const uint8_t __near SENT_CRC4_tbl[16] = {
//  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
// -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
    0,13, 7,10,14, 3, 9, 4, 1,12, 6,11,15, 2, 8, 5
};

// S/W: Basic model
uint8_t sent_crc4(uint8_t* pdata, uint16_t ndata)
{
    uint8_t    crc;
    uint16_t  i;

    crc = 5;                                    // Seed.
    for(i=0; i<ndata; i++){
     crc = *pdata++ ^ SENT_CRC4_tbl[crc];       // Data[#1 to #6]
    }
    //crc = 0 ^ crc4sent_tbl[crc];              // Post-process.

    //return crc;                               // Return the CRC result.
    return SENT_CRC4_tbl[crc];
}
```

Figure 1-3 Basic Procedure for CRC Calculation (J2716 standard)

Figure 1-4 shows a listing using RL78/F1x CRC operation function. Set the RL78/F1x general-purpose CRC calculator to SENT calculation mode, enter Data[#1 to #6] values into the CRC calculator and get the result.

Figure 1-5 shows another more optimized listing, which is using loop-unrolling.

```
// H/W#0: Normal.
uint8_t sent_crc4(uint8_t* pdata, uint16_t ndata)
{
   uint16_t  i;

   CRCMD = 0x01;              // CRC mode: For SENT (X4+X3+X2+1)
   CRCD  = 5;                 // Seed.
   for(i=0; i<ndata; i++){
       CRCIN = *pdata++;      // Data[#1 to #6]
   }
   CRCIN = 0x0000;            // Post-process.
   NOP();                     // Wait for 1 clock.

   return CRCD;               // Return the CRC result.
}
```

Figure 1-4 CRC Calculation Procedure using RL78/F1x CRC Operation Function

```
// H/W#1: Optimized: loop-unrolling.
uint8_t sent_crc4_data6(uint8_t* pdata)
{
   CRCMD = 0x01;              // CRC mode: For SENT (X4+X3+X2+1)
   CRCD  = 5;                 // Seed.
   CRCIN = *pdata++;          // Data#1
   CRCIN = *pdata++;          // Data#2
   CRCIN = *pdata++;          // Data#3
   CRCIN = *pdata++;          // Data#4
   CRCIN = *pdata++;          // Data#5
   CRCIN = *pdata++;          // Data#6
   CRCIN = 0x0000;            // Post-process.
   NOP();                     // Wait for 1 clock.

   return CRCD;               // Return the CRC result.
}
```

Figure 1-5 CRC Calculation Procedure using RL78/F1x CRC Operation Function (Optimized)

2. Implementation of SENT Transmission

This section explains about implementation of SENT transmission for RL78/F1x.

2.1 Process Overview of SENT Transmission

Figure 2-1 shows the process overview of the SENT transmission example.

Timer array unit TAU00 and TAU01 generate PWM waveforms and output the waveform signal to P30 / TO01 port. The low level period is fixed as 5 ticks, which is generated by TAU01. The high level period is flexible, and the width of the combined pulse waveform represents the data value. So, the flexible pulse wave, i.e. TAU00 setting is adjusted for each transmission data.

The DTC feature of RL78/F1x products simplifies the software process a lot. Only completion timing for the last pulse (Pause pulse) invokes TAU01 interrupt, and software has to take care only for this interrupt.

User's hook function is also invoked by the completion interrupt. The user describes the hook function process to prepare transmission data for the next transmission.

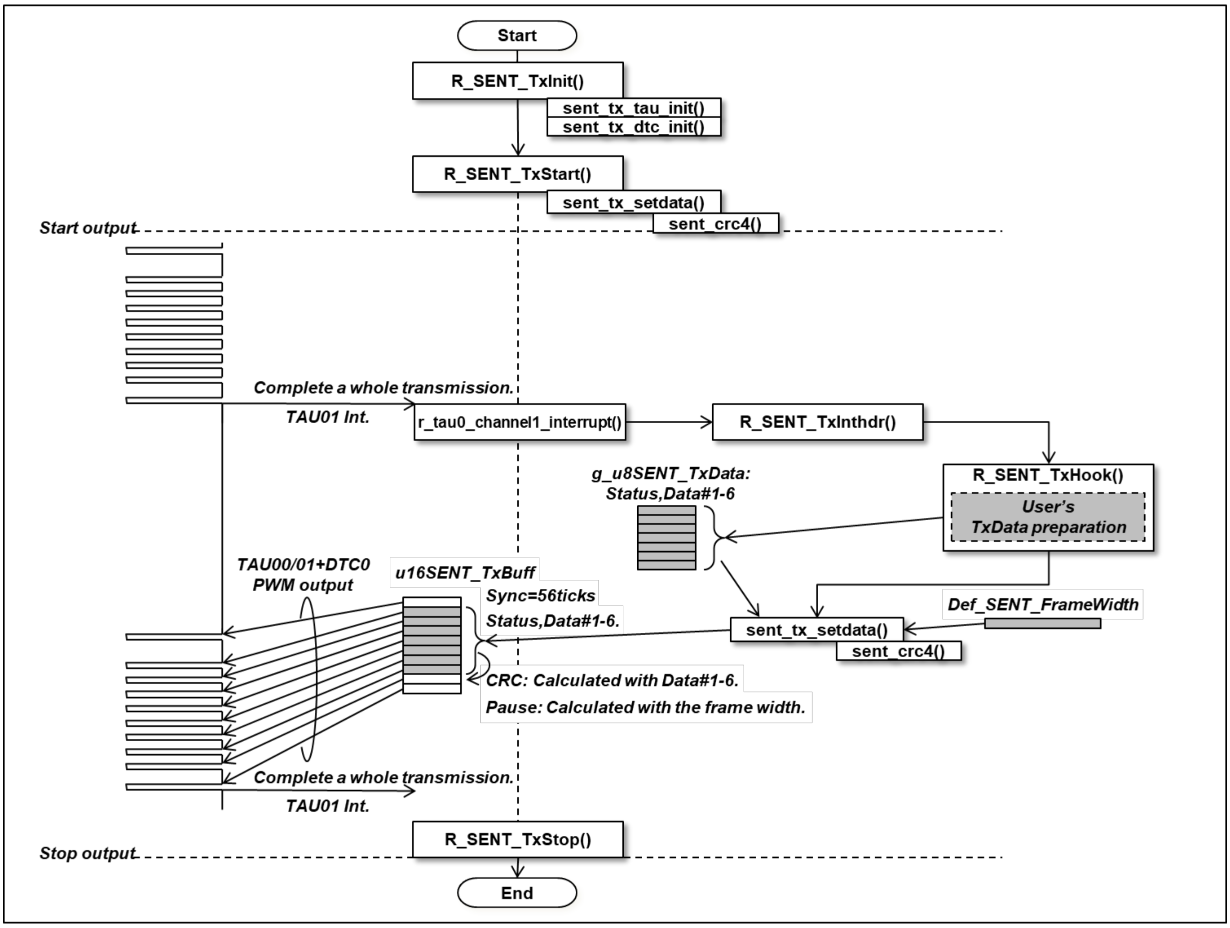

Figure 2-1 Process Overview of SENT Transmission

2.2 SFR Settings for SENT Transmission

Table 2-1 and Table 2-2 show the RL78/F1x products SFR settings for SENT transmission example.

Table 2-1 RL78/F1x SFR Settings for SENT Transmission Example: TAU00/01 Settings

Register name	Setting
TDR00	TO01 PWM frequency cycle for SENT transmission Set by DTC.
TDR01	TO01 PWM on duty cycle for SENT transmission 0x000FU (15 μs)
TPS0	0x5320U (CK00=32MHz, CK01=8MHz, CK02=4MHz, CK03=1MHz)
TMR00	0xC001U (CKS00[1:0]=11B: CK03 (1MHz) selected. MD00[3:1]=000B: Interval timer mode selected. MD000=1: Interrupt occurred when stating the timer count.)
TMR01	0xC409U (CKS01[1:0]=11B: CK03 (1MHz) selected. STS01[2:0]=100B: Slave channel selected. MD01[3:1]=100B: One count mode selected. MD010=1: Interrupt occurred when stating the timer count.)
TOE0	TOE0 |= 0x0002U; (TOE01=1)
TO0	TO0 |= 0x0002U; (TO01=1)
TOL0	TOL0 |= 0x0002U; (TOL01=1)
TOM0	TOM0 |= 0x0002U; (TOM01=1)
PWMDLY1	0x0000U (TO01[1:0]=00B)

Table 2-2 RL78/F1x SFR Settings for SENT Transmission Example: DTC Settings

Register name	Setting
DTCEN2	DTCEN2 |= 0x10U; (DTCEN24=1)
DTCBAR	0xFDU
DTCVECT_ADDR[19]	0x40U
DTCCR0	0x04U (SZ=0, SAMOD=1, DAMOD=0, MODE=0)
DTBLS0	0x01U
DTCCT0	0x0AU
DTSAR0	u16SENT_TxBuff (Source Address)
DTDAR0	0xFF18U (Destination Address: TDR00 register)

2.3 Variables for SENT Transmission Example

This section describes variables for SENT transmission example, as shown in Table 2-3.

g_u8SENT_TxData[8] is API variable array, and user sets transmission data to the array in the notification function of SENT transmission completion R_SENT_TxHook(). The transmission data array is each nibble value of Status and Data[#1 to #6], and is in the range of “0” to “15”. After the user sets the transmission data, the data setup function sent_tx_setdata() prepares the pulse length table u16SENT_TxBuff[10] for DTC transfer based on the transmission data.

Def_SENT_FrameWidth is a configuration constant to define frame width of a SENT transmission waveform. User should define the constant value within the range of 284 to 920 ([ticks]). Please refer Table 1-3.

Table 2-3 Variables for SENT Transmission Example

Variable Name	Definition	Specification
g_u8SENT_TxData[8]	uint8_t (unsigned char)	Data to be sent by SENT transmission (public): g_u8SENT_TxData[0]: RESERVED g_u8SENT_TxData[1]: Status nibble data g_u8SENT_TxData[2:7]: Data[#1 to #6] nibble data
u16SENT_TxBuff[10]	uint16_t (unsigned short)	Data buffer for SENT transmission (local): u16SENT_TxBuff[0]: Sync time length [μs] u16SENT_TxBuff[1]: Status nibble time length [μs] u16SENT_TxBuff[2:7]: Data[#1 to #6] nibble time length [μs] u16SENT_TxBuff[8]: CRC nibble time length [μs] u16SENT_TxBuff[9]: Pause pulse time length [μs]
Def_SENT_FrameWidth	Macro definition	Frame width for a SENT transmission waveform (configuration constant by macro definition): The constant value should be set within the range of 284 to 920.

2.4 Process Flow for SENT Transmission

Figure 2-2 shows process flow (interrupt processing function) for SENT transmission example. Procedure of the interrupt processing is very simple with DTC support (see also section 2.1).

The procedure invokes the hook function for user to notify a transmission completion, and the user sets next transmission data. The pulse length table is prepared based on the transmission data, and DTC will be restarted. The timer restart is not necessary, because TAU00/01 is not stopped by each transmission unit.

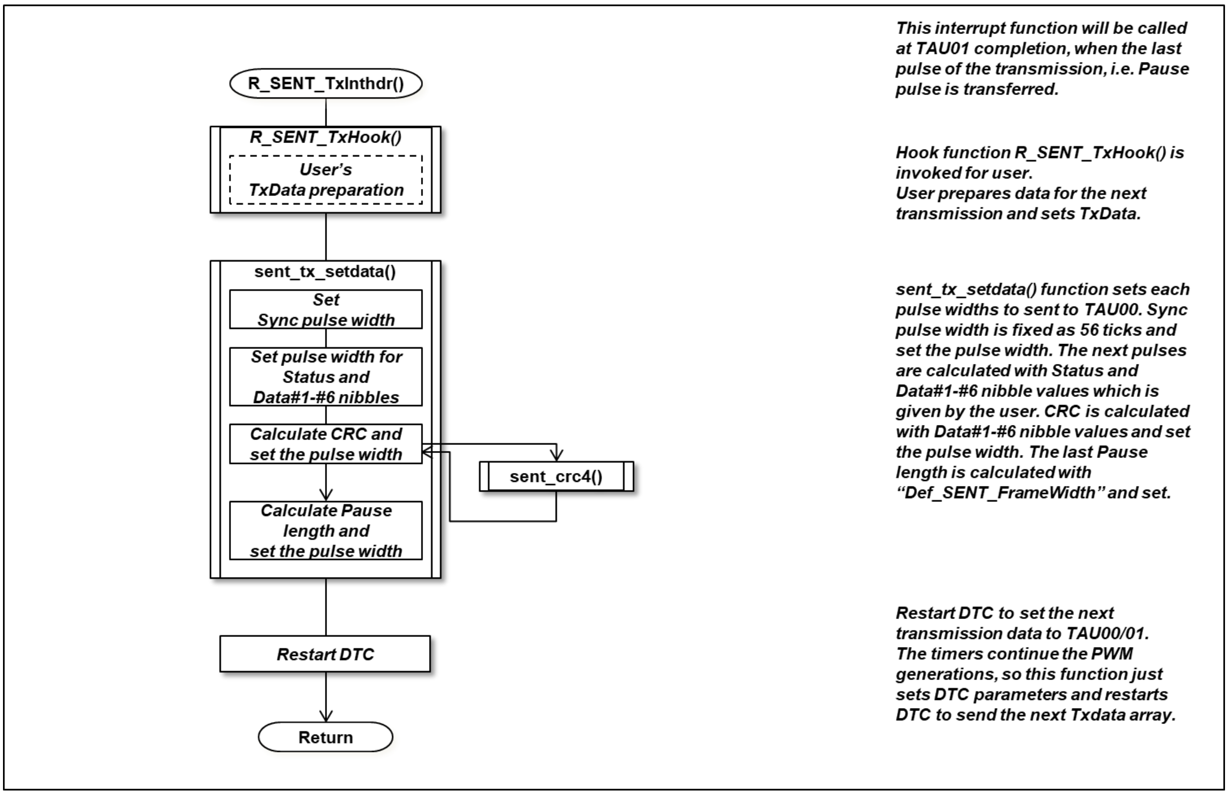

Figure 2-2 Process Flow for SENT Transmission Example (Interrupt Processing Function: R_SENT_TxInthdr())

2.5 Functions for SENT Transmission Example

This section describes functions for SENT transmission example, as shown in Table 2-4.

Table 2-4 Functions for SENT Transmission Example

Function	Prototype
SENT transmission initialization (public)	void R_SENT_TxInit(void);
Timer array unit (TAU00, TAU01) initialization for SENT transmission (local)	void sent_tx_tau_init(void);
DTC initialization for SENT transmission (local)	void sent_tx_dtc_init(void);
SENT transmission start (public)	void R_SENT_TxStart(void);
SENT transmission stop (public)	void R_SENT_TxStop(void);
Interrupt processing function for SENT transmission completion (public)	void R_SENT_TxInthdr(void);
Notification function of SENT transmission completion (public)	void R_SENT_TxHook(void);
Data setup for SENT transmission (local)	void sent_tx_setdata(void);
SENT CRC calculation (local)	uint8_t sent_crc4(uint8_t* pdata);

2.5.1 SENT Transmission Initialization

Table 2-5 SENT Transmission Initialization

Syntax		void R_SENT_TxInit(void);
Parameters	In	None
	Out	None
Return value		None
Description		Initialize SENT transmission, with setting up timer array unit (TAU00, TAU01) and DTC which are used by SENT transmission. This function also initializes transmission data uint8_t g_u8SENT_TxData[1:7] as 0, which will be used for the first waveform output.

2.5.2 Timer Array Unit (TAU00, TAU01) Initialization for SENT Transmission

Table 2-6 Timer Array Unit (TAU00, TAU01) Initialization for SENT Transmission

Syntax		void sent_tx_tau_init(void);
Parameters	In	None
	Out	None
Return value		None
Description		Initialize timer array unit (TAU00, TAU01) for SENT transmission. This function is called by R_SENT_TxInit(). Please refer Table 2-1 about the settings.

2.5.3 DTC Initialization for SENT Transmission

Table 2-7 DTC Initialization for SENT Transmission

Syntax		void sent_tx_dtc_init(void);
Parameters	In	None
	Out	None
Return value		None
Description		Initialize DTC for SENT transmission. This function is called by R_SENT_TxInit(). Please refer Table 2-2 about the settings.

2.5.4 SENT Transmission Start

Table 2-8 SENT Transmission Start

Syntax		void R_SENT_TxStart(void);
Parameters	In	None
	Out	None
Return value		None
Description		Start SENT transmission, with starting timer array unit (TAU00, TAU01) and DTC. This function also setup pulse length table uint16_t u16SENT_TxBuff[] using data setup function sent_tx_setdata() for the first waveform output. User can set transmission data uint8_t g_u8SENT_TxData[1:7] which will be used for the first waveform output before this function.

2.5.5 SENT Transmission Stop

Table 2-9 SENT Transmission Stop

Syntax		void R_SENT_TxStop(void);
Parameters	In	None
	Out	None
Return value		None
Description		Stop SENT transmission, with stopping timer array unit (TAU00, TAU01) and DTC. P30/TO01 port is initialized as high level.

2.5.6 Interrupt Processing for SENT Transmission Completion

Table 2-10 Interrupt Processing Function for SENT Transmission Completion

Syntax		void R_SENT_TxInthdr(void);
Parameters	In	None
	Out	None
Return value		None
Description		Interrupt processing function for DTC transfer completion when SENT transmission is completed. The timing is just after the low level output of the last Pause pulse of the transmission frame is completed. This function is called by INTTM01 (TAU01 completion interrupt), which calls notification function of SENT transmission completion for user, prepares pulse length table for the next transmission based on the user's requested data, and restarts DTC. Please refer Figure 2-2.

2.5.7 Notification Function of SENT Transmission Completion

Table 2-11 Hook Function for SENT Transmission Completion

Syntax		void R_SENT_TxHook(void);
Parameters	In	None
	Out	None
Return value		None
Description		Called by R_SENT_TxInthdr() to notify user of SENT transmission completion. The user describes the function process to prepare transmission data for the next frame. The transmission data are 7-elements of 4-bit data stored in 8-bit data array, i.e. uint8_t g_u8SENT_TxData[1:7]. g_u8SENT_TxData[0] is reserved. The data are consisting of Status nibble and Data[#1 to #6] nibbles. Please refer Table 2-3.

2.5.8 Data Setup for SENT Transmission

Table 2-12 Data Setup for SENT Transmission

Syntax		void sent_tx_setdata(void);
Parameters	In	None
	Out	None
Return value		None
Description		Setup transmission data for the next frame. This function is called after transmission completion notification function R_SENT_TxHook() by R_SENT_TxInthdr(), which prepares pulse length table uint16_t u16SENT_TxBuff[] for the next transmission based on the user's requested data uint8_t g_u8SENT_TxData[].

2.5.9 SENT CRC Calculation

Table 2-13 SENT CRC Calculation Function

Syntax		uint8_t sent_crc4(uint8_t* pdata);		
Parameters	In	uint8_t	data	Input data array.
	Out	None		
Return value		uint8_t	crc	Calculation result for CRC.
Description		Calculate 4-bit CRC for SENT, J2716 standard. The input data are 6-elements of 4-bit data stored in 8-bit data array, i.e. uint8_t data[6]. The data are consisting of Data[#1 to #6] nibbles. This function uses RL78/F1x operation function for SENT CRC calculation to optimize the speed.		

3. Implementation of SENT Reception

This section explains about implementation of SENT reception for RL78/F1x.

3.1 Process Overview of SENT Reception

Figure 3-1 shows the process overview of the SENT reception example.

Timer array unit TAU02 measures wave pulse signal length input from P16 / TI02 port.

INTTM02 (TAU02 completion interrupt) occurs for measurements of each pulse signals (Pause, Sync, Status, Data[#1 to #6] and CRC).

User's hook function is invoked after the last pulse (CRC) of the frame was detected and checked. The user processes the reception data in the hook function.

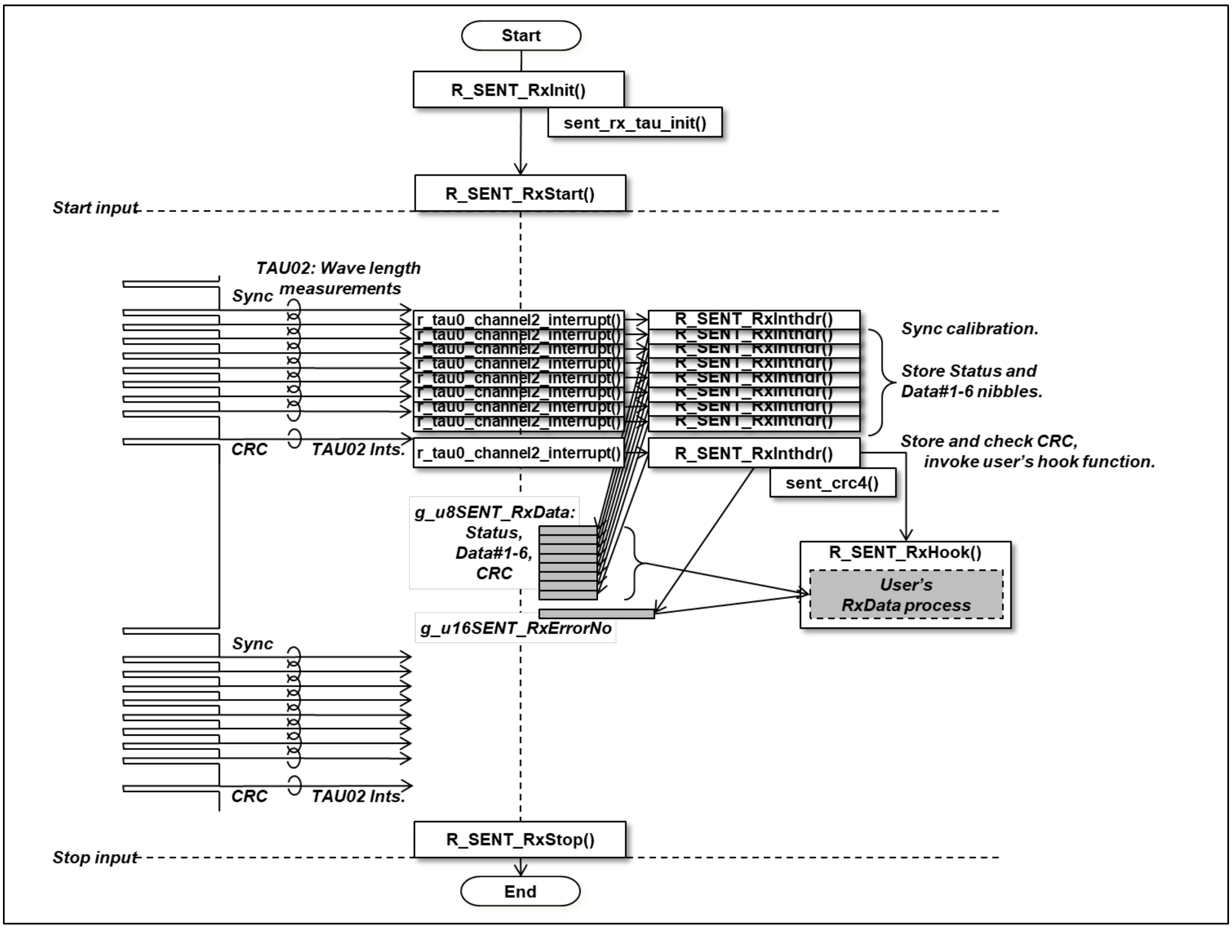

Figure 3-1 Process Overview of SENT Reception

3.2 SFR Settings for SENT Reception

Table 3-1 shows the RL78/F1x products SFR settings for SENT reception example.

Table 3-1 RL78/F1x SFR Settings for SENT Reception Example: TAU02 Settings

Register name	Setting
TDR02	TI02 pulse width capture time for SENT reception
TPS0	0x5320U (CK00=32MHz, CK01=8MHz, CK02=4MHz, CK03=1MHz)
TMR02	0x0104U (CKS02[1:0]=00B; CK00 selected. STS02[2:0]=001B; TI02 input enabled. CIS02[1:0]=00B; Falling edge selected. MD02[3:1]=010B; Input capture mode selected.)
TIS0	TIS0 &= ~ 0x40; (TIS06=0)
TOE0	TOE0 &= ~0x0004U; (TOE02=0)
TOL0	TOL0 &= ~0x0004U; (TOL02=0)
TO	TO0 &= ~0x0004U; (TO02=0)
NFEN1	NFEN1 |= 0x04U; (TNFEN02=1)

3.3 Variables for SENT Reception Example

This section describes variables for SENT reception example, as shown in Table 3-2.

g_u8SENT_RxData[10] is API variable array which holds reception data, and user can read the reception data from the array in the notification function of SENT reception completion R_SENT_RxHook(). The reception data is each nibble value of Status, Data[#1 to #6] and CRC, and is in the range of “0” to “15”.

g_u16SENT_RxErrorNo is also API variable to hold error status of SENT reception processes.

Table 3-2 Variables for SENT Reception Example

Variable Name	Definition	Specification
g_u8SENT_RxData[10]	uint8_t (unsigned char)	Data received by SENT reception (public): g_u8SENT_RxData[0]: RESERVED g_u8SENT_RxData[1]: Status nibble data g_u8SENT_RxData[2:7]: Data[#1 to #6] nibble data g_u8SENT_RxData[8]: CRC nibble data g_u8SENT_RxData[9]: RESERVED
g_u16SENT_RxErrorNo	uint16_t (unsigned short)	Error status for SENT reception (public): 0000H: No error 000AH: CRC error

3.4 Process Flow for SENT Reception

Figure 3-2 shows process flow (interrupt processing function) for SENT reception example. The function recognizes every pulse signal input and detects and stores certain reception data. (See also section 3.1)

The procedure invokes the hook function for SENT reception completion for user to read and process the reception data when CRC nibble is captured and checked for the value.

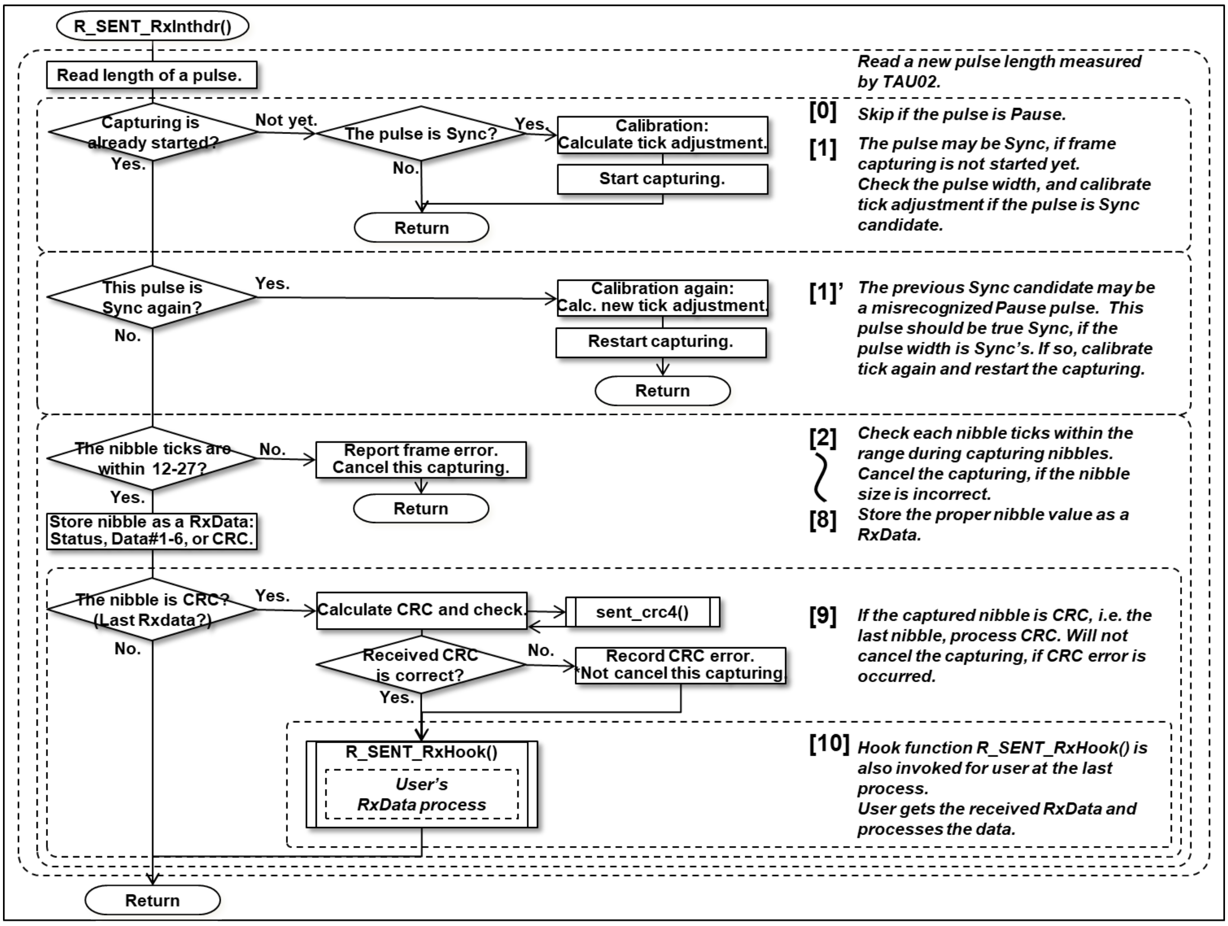

Figure 3-2 Process Flow for SENT Reception Example (Interrupt Processing Function: R_SENT_RxInthdr())

Figure 3-3 shows the timing chart of the SENT reception process. The each numbering in the figure ("[0]" – "[10]") is corresponding to Figure 3-2's.

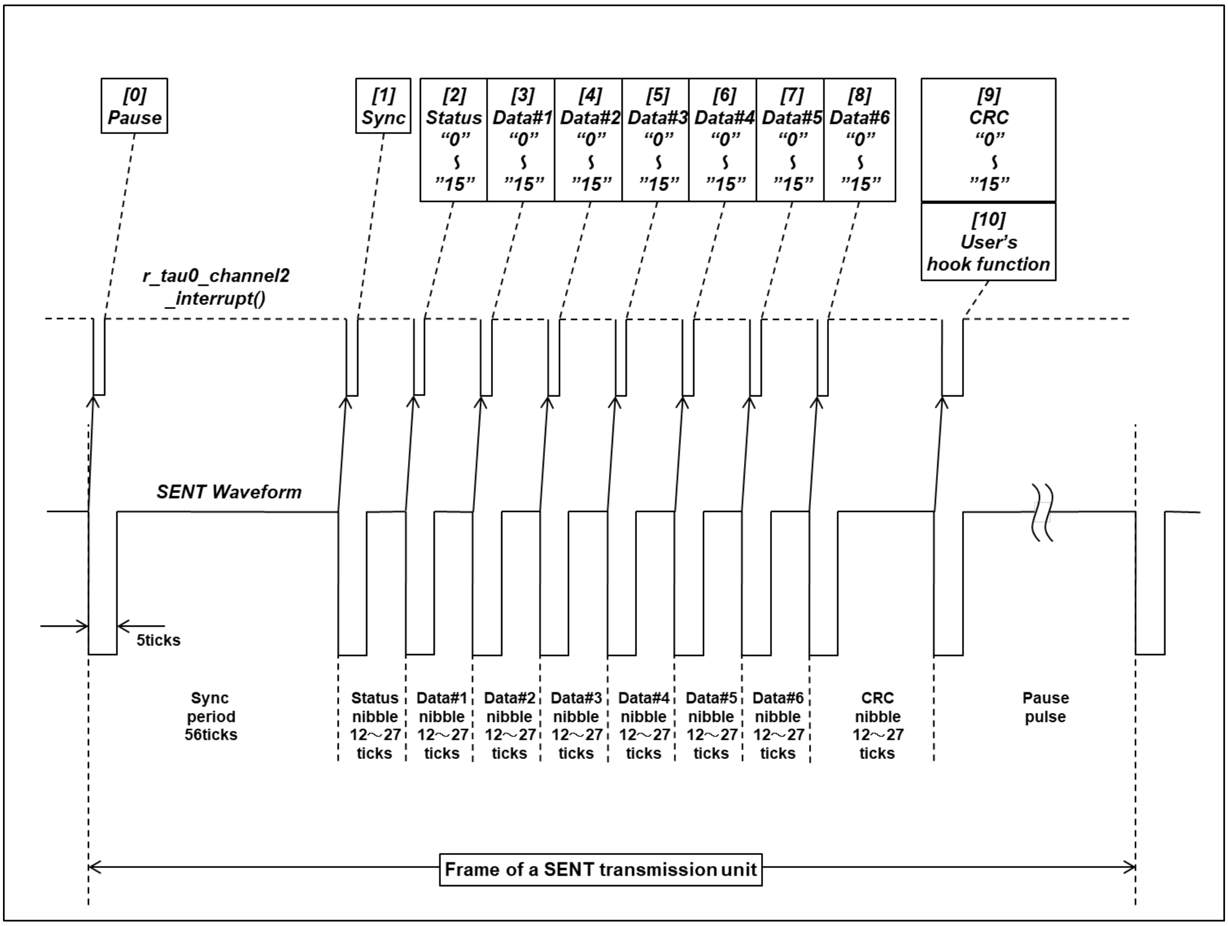

Figure 3-3 Timing Chart of SENT Reception Example Process

3.5 Functions for SENT Reception Example

This section describes functions for SENT reception example, as shown in Table 3-3.

Table 3-3 Functions for SENT Reception Example

Function	Prototype
SENT reception initialization (public)	void R_SENT_RxInit(void);
Timer array unit (TAU02) initialization for SENT reception (local)	void sent_rx_tau_init(void);
SENT reception start (public)	void R_SENT_RxStart(void);
SENT reception stop (public)	void R_SENT_RxStop(void);
Interrupt processing function for SENT reception (public)	void R_SENT_RxInthdr(void);
Notification function of SENT reception completion (public)	void R_SENT_RxHook(void);
SENT CRC calculation (local)	uint8_t sent_crc4(uint8_t* pdata);

3.5.1 SENT Reception Initialization

Table 3-4 SENT Reception Initialization

Syntax		void R_SENT_RxInit(void);
Parameters	In	None
	Out	None
Return value		None
Description		Initialize SENT reception, with setting up timer array unit (TAU02) which is used by SENT reception.

3.5.2 Timer Array Unit (TAU02) Initialization for SENT Reception

Table 3-5 Timer Array Unit (TAU02) Initialization for SENT Reception

Syntax		void sent_rx_tau_init(void);
Parameters	In	None
	Out	None
Return value		None
Description		Initialize timer array unit (TAU02) for SENT reception. This function is called by R_SENT_RxInit(). Please refer Table 3-1 about the settings.

3.5.3 SENT Reception Start

Table 3-6 SENT Reception Start

Syntax		void R_SENT_RxStart(void);
Parameters	In	None
	Out	None
Return value		None
Description		Start SENT reception, with starting timer array unit (TAU02).

3.5.4 SENT Reception Stop

Table 3-7 SENT Reception Stop

Syntax		void R_SENT_RxStop(void);
Parameters	In	None
	Out	None
Return value		None
Description		Stop SENT reception, with stopping timer array unit (TAU02).

3.5.5 Interrupt Processing for SENT Reception

Table 3-8 Interrupt Processing Function for SENT Reception

Syntax		void R_SENT_RxInthdr(void);
Parameters	In	None
	Out	None
Return value		None
Description		Interrupt processing function when SENT reception is completed. This function is called by INTTM02 (TAU02 completion interrupt) which is invoked when measurement of the each input pulse length is completed. When Sync/Calibration pulse is received, the 1-tick pulse length is calculated from the received pulse. After receiving Sync pulse, Status and Data[#1 to #6] and CRC pulse are received in sequence. The reception of Sync/Calibration pulse is always judged. The function calls notification function of SENT reception completion for user, when CRC nibble is captured and checked for the value. Please refer Figure 3-2.

3.5.6 Notification Function of SENT Reception Completion

Table 3-9 Hook Function for SENT Reception Completion

Syntax		void R_SENT_RxHook(void);
Parameters	In	None
	Out	None
Return value		None
Description		Called by R_SENT_RxInthdr() to notify user of SENT reception completion. The user describes the function process to get and process reception data which are read from uint8_t g_u8SENT_RxData[]. The reception data are 8-elements of 4-bit data stored in 8-bit data array, i.e. uint8_t g_u8SENT_RxData[1:8]. g_u8SENT_RxData[0] and g_u8SENT_RxData[9] are reserved. The data are consisting of Status nibble, Data[#1 to #6] nibbles and CRC nibble. Please refer Table 3-2. If CRC error is detected during reception, the error code will be set in variable uint16_t g_u16SENT_RxErrorNo. User can confirm the error status by reading the variable. 0000H: No error 000AH: CRC error The reception will not be canceled even if the CRC error is occurred, and the CRC nibble value user will read from the variable is captured one.

3.5.7 SENT CRC Calculation

Same as sent_crc4() function for SENT transmission. Please refer 2.5.9

4. References

The documents referenced in this application note are shown below. When referring to these documents, make sure to obtain the latest version of each document from Renesas Electronics website.

- RL78/F13, F14 User's Manual: Hardware
- RL78/F15 User's Manual: Hardware

Also recommend to refer the following document for the SENT specification.

- SAE International, SENT - Single Edge Nibble Transmission for Automotive Applications J2716 APR2016, SAE International, 2016.

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Aug. 30, 2020	-	First edition issued

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 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.
5. 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.
6. 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.
7. 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.
8. 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.
9. 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.
10. 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.
11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. 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.4.0-1 November 2017)

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/.