RL78/F23, F24

タイマ・アレイ・ユニットと DTC を用いた SENT 通信の実現

R01AN6623JJ0100
Rev.1.00
2022.09.30

要旨

本アプリケーションノートでは、RL78/F23、RL78/F24（以下、「RL78/F2x」）の DTC と PWM 機能を用いた Single Edge Nibble Transmission 通信（以下、「SENT」）機能の実装例について説明します。本アプリケーションノートでは、SENT 送信機能と SENT 受信機能、それぞれについて説明しています。

使用条件によっては、本アプリケーションノートで説明する例と動作が異なる場合があります。実装後は十分な評価を実施してください。また、SENT 通信の機能やクロック、割り込み機能の詳細は必ずご使用する製品のユーザーズマニュアルで確認してください。

動作確認デバイス

RL78/F23、RL78/F24

目次

1. RL78/F2x 用 SENT 通信実装の概要 3
1.1 SENT 通信で使用する RL78/F2x ハードウェアリソース 3
1.2 SENT 通信実装例の仕様 4
1.3 SENT 信号の波形 5
1.4 SENT 通信の CRC 計算 6

2. SENT 送信の実装 8
2.1 SENT 送信処理の概要 8
2.2 SENT 送信時の SFR 設定 9
2.3 SENT 送信実装例の使用変数 10
2.4 SENT 送信の処理フロー 11
2.5 SENT 送信実装例の処理関数 12
2.5.1 SENT 送信初期化 12
2.5.2 SENT 送信用タイマ・アレイ・ユニット（TAU00、TAU01）初期設定 12
2.5.3 SENT 送信用 DTC 初期設定 12
2.5.4 SENT 送信開始処理 13
2.5.5 SENT 送信停止処理 13
2.5.6 SENT 送信完了割り込み処理 13
2.5.7 SENT 送信完了通知関数 14
2.5.8 SENT 送信用データ設定処理 14
2.5.9 SENT 用 CRC 計算処理 14

3. SENT 受信の実装 15
3.1 SENT 受信処理の概要 15
3.2 SENT 受信時の SFR 設定 16
3.3 SENT 受信実装例の使用変数 16
3.4 SENT 受信の処理フロー 17
3.5 SENT 受信実装例の処理関数 19
3.5.1 SENT 受信初期化 19
3.5.2 SENT 受信用タイマ・アレイ・ユニット（TAU02）初期設定 19
3.5.3 SENT 受信開始処理 19
3.5.4 SENT 受信停止処理 20
3.5.5 SENT 受信割り込み処理 20
3.5.6 SENT 受信完了通知関数 20
3.5.7 SENT 用 CRC 計算処理 21

4. 参考資料 22

改訂記録 23

1.　RL78/F2x 用 SENT 通信実装の概要

本章では、RL78/F2x 用の SENT 通信実装例の仕様概要を説明します。

1.1　SENT 通信で使用する RL78/F2x ハードウェアリソース

表 1-1 に SENT 送信のために使用する RL78/F2x ハードウェア機能を示します。タイマ・アレイ・ユニット TAU00/01 を使用して PWM 信号を生成します。タイマへ各送信データに対応したパルス幅値を転送するために、データトランスファコントローラ（DTC）を使用します。この実装例では、波形信号は P30 / TO01 ポートから出力します。

表 1-1 Used RL78/F2x Resources for SENT Transmission

Item		Description
CPU/peripheral hardware clock frequency (f_{CLK})		40 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}/8 = 5 MHz

注意　SENT 送信に使用する基準クロックとして高速オンチップ・オシレータを使用しています。周波数設定はユーザ・オプション・バイト(000C2H)で行います。設定の詳細は、ユーザーズマニュアルを参照して下さい。

表 1-2 に SENT 受信のために使用する RL78/F2x ハードウェア機能を示します。タイマ・アレイ・ユニット TAU02 で SENT 受信信号の入力パルス幅を測定し受信したデータ値を読み取ります。この実装例では、P16 / TI02 ポートに波形信号を入力します。

表 1-2 Used RL78/F2x Resources for SENT Reception

Item		Description
CPU/peripheral hardware clock frequency (f_{CLK})		40 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} = 40 MHz

注意　SENT 受信に使用する基準クロックとして高速オンチップ・オシレータを使用しています。周波数設定はユーザ・オプション・バイト(000C2H)で行います。設定の詳細は、ユーザーズマニュアルを参照して下さい。

1.2　SENT 通信実装例の仕様

表 1-3 に SENT 通信実装例の仕様を示します。本実装例では、データ nibble 数は 6 個に固定しています。また、SENT 受信時の nibble キャリブレーションもサポートします。一つの SENT メッセージフレームは 32 ビット（8 個の nibble）データを含み、Sync 期間、Status nibble、6 個のデータ nibble、CRC nibble、および Pause パルスから成ります。また、CRC エラー検知をサポートします。

表 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 to 216 µs (-5 to +29%). Calibrate tick width with the detected Sync width.
Width of each nibble	12 to 27 ticks (36 to 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 to 920 ticks (852 to 2760 µs typically)
Pause pulse	Supported: 12 to 768 ticks (36 to 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 信号の波形

図 1-1 に基本的な SENT 信号の波形と、図 1-2 に本実装例で出力した実際の波形例を示します。この波形例では、Pause パルスによってフレーム幅を一定に調整しています。

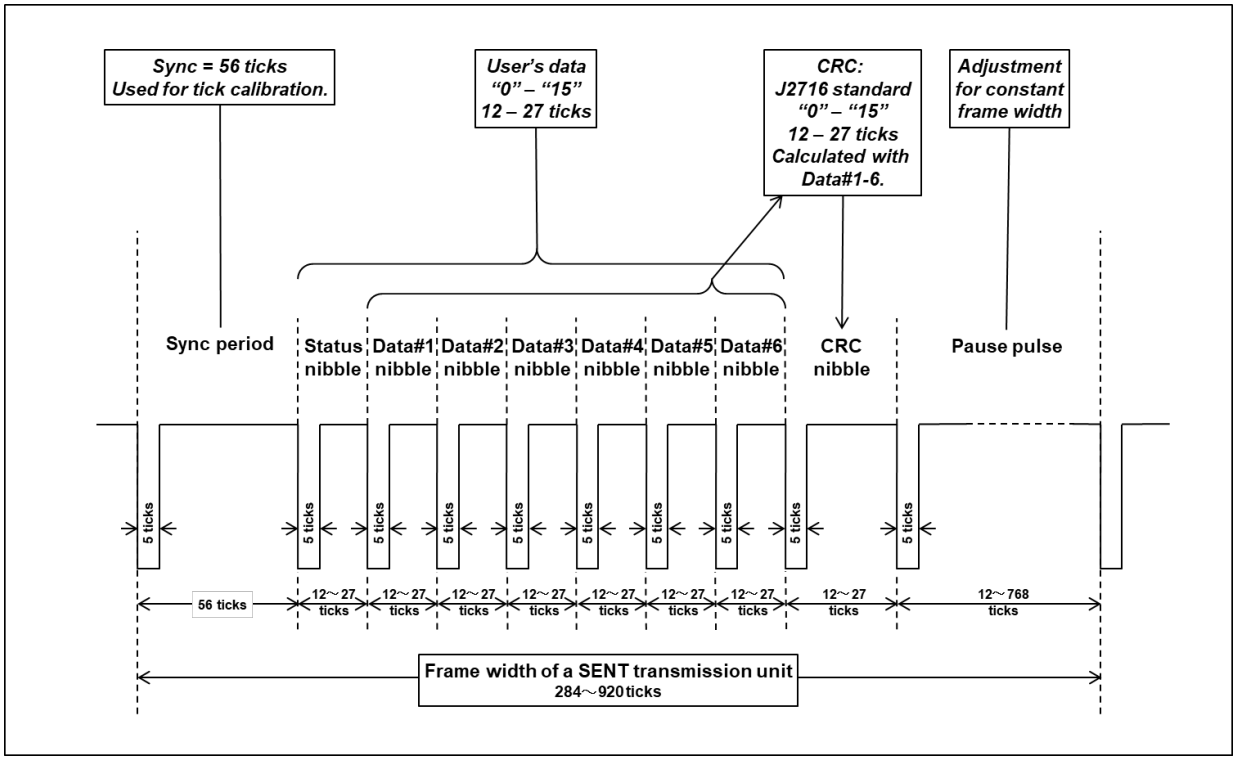

図 1-1 SENT Waveform

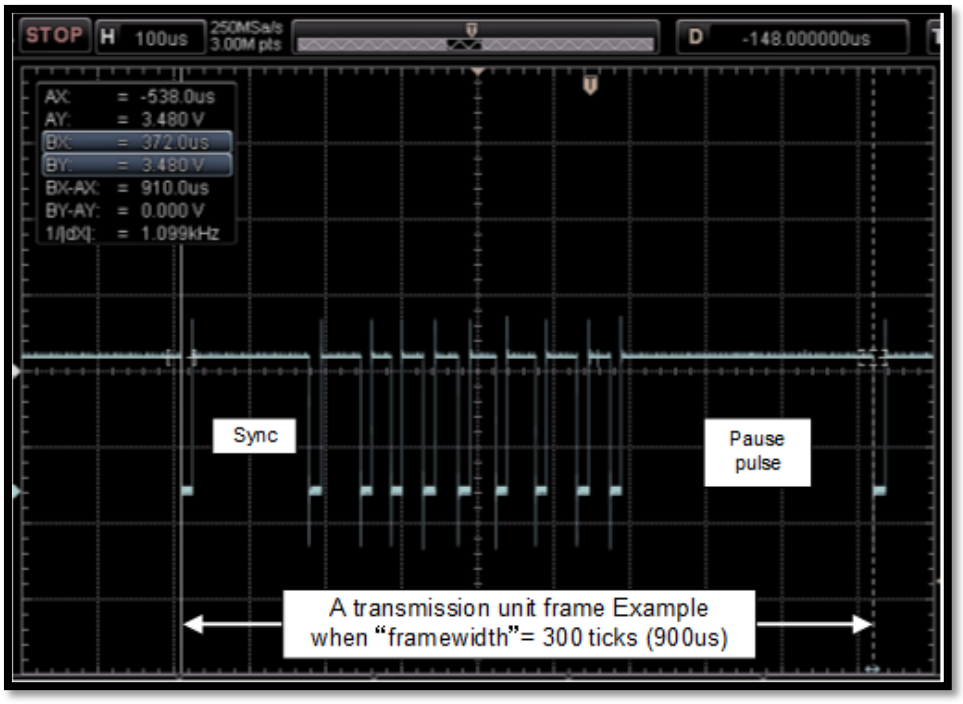

図 1-2 Real SENT Waveforms Generated by this Example

1.4　SENT 通信の CRC 計算

RL78/F2x は SENT の CRC 計算をハードウェアでサポートしています。本実装例では、CRC 演算回路を用いてハードウェアによる CRC 計算を行います。

図 1-3 は SENT 通信用 CRC 計算のソースリスト例です。計算手順は J2716 方式に準拠します。CRC 値は、データ#1～データ#6 の nibble 値を入力として、4 ビット長で計算します。図 1-1 も参照ください。

```
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];
}
```

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

図 1-4 は RL78/F2x の CRC 演算回路を用いた CRC 計算手順です。CRC 演算モードを SENT 用に設定して、データ#1～データ#6 の nibble 値を入力して計算結果を得ます。

図 1-5 は、さらに高速化した計算手順リストです。loop-unrolling 手法で高速化しています。

```
// 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.
}
```

図 1-4 CRC Calculation Procedure using RL78/F2x 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.
}
```

図 1-5 CRC Calculation Procedure using RL78/F2x CRC Operation Function (Optimized)

2.　SENT 送信の実装

本章では、RL78/F2x の SENT 送信機能実装について説明します。

2.1　SENT 送信処理の概要

図 2-1 に、SENT 送信実装例の処理概要を示します。

タイマ・アレイ・ユニット TAU00 と TAU01 を用いて、P30 / TO01 端子から信号を出力します。本 SENT 通信では各パルス波形の low レベル期間を 5 ticks 固定としており、この low レベル期間を TAU01 で生成します。一方、high レベル期間は可変であり、これらの low レベルと high レベルを組み合わせたパルス波形でデータ値を表します。各送信データ値を基にしたそれぞれのパルス幅を TAU00 に毎回設定することで SENT 通信に準拠した可変長のパルス波形を生成しています。

各送信データに対応したパルス幅は RL78/F2x の DTC 機能を用いて TDR00 (TAU00 の high レベル時間設定レジスタ)に転送することで、ソフトウェア処理を簡素化しています。

SENT メッセージフレームの最後のパルス (Pause パルス) の low レベル出力完了後 DTC の転送完了割り込みが発生します。ソフトウェア処理はこの DTC 転送完了割り込みで実施します。

SENT 送信完了通知関数もこの完了割り込みで呼び出され、ユーザは次の送信のためのデータ準備を行います。

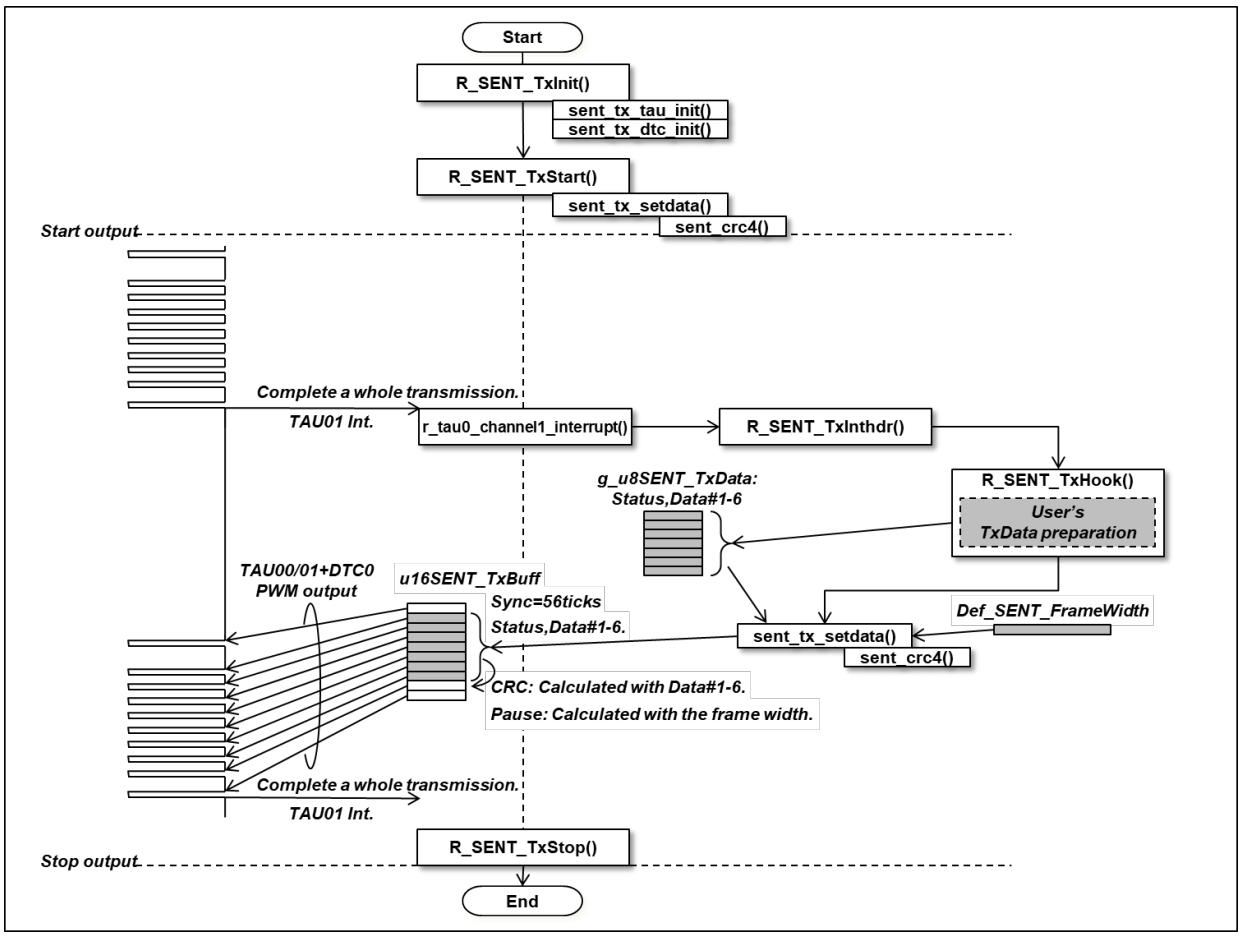

図 2-1 Process Overview of SENT Transmission

2.2　SENT 送信時の SFR 設定

表 2-1 と表 2-2 に SENT 送信実装例の RL78/F2x SFR 設定（TAU00/01 関連と DTC 関連それぞれ）を示します。

表 2-1 RL78/F2x 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	0x3210U (CK00=40MHz, CK01=20MHz, CK02=10MHz, CK03=5MHz)
TMR00	0xC001U (CKS00[1:0]=11B: CK03 (5MHz) 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 (5MHz) 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)

表 2-2 RL78/F2x 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　SENT 送信実装例の使用変数

本章では SENT 送信実装例の使用する変数について説明します。表 2-3 に SENT 送信の変数一覧を示します。

g_u8SENT_TxData[8]は API 用変数配列です。ユーザが次の送信で送りたいデータを SENT 送信完了通知関数 R_SENT_TxHook()の中でセットします。送信用データは Status 及びデータ#1～データ#6 の各 nibble 値で、”0”～”15”の範囲です。ユーザが送信用データをセット後、SENT 送信データ設定処理 sent_tx_setdata()が送信用データを元に DTC 転送用のパルス長テーブル u16SENT_TxBuff[10]を準備します。

Def_SENT_FrameWidth は、一回の SENT 送信波形の長さを定義するコンフィギュレーション定数です。284～920（[ticks]）の範囲内で定義します。表 1-3 も参照ください。

表 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　SENT 送信の処理フロー

図 2-2 に SENT 送信実装例の処理（割り込み処理関数）を示します。この割り込み処理は RL78/F2x の DTC 機能を使用することでシンプルな構造となっています（2.1 章も参照ください）。

この処理ではユーザ通知用にフック関数を呼び出し、ユーザは次の送信データの要求を行います。そして、要求された送信データに対応したパルス長テーブルを準備して、DTC を再起動します。TAU00/01 は動作し続けていますので、次のフレーム送信のためのタイマ再始動は必要ありません。

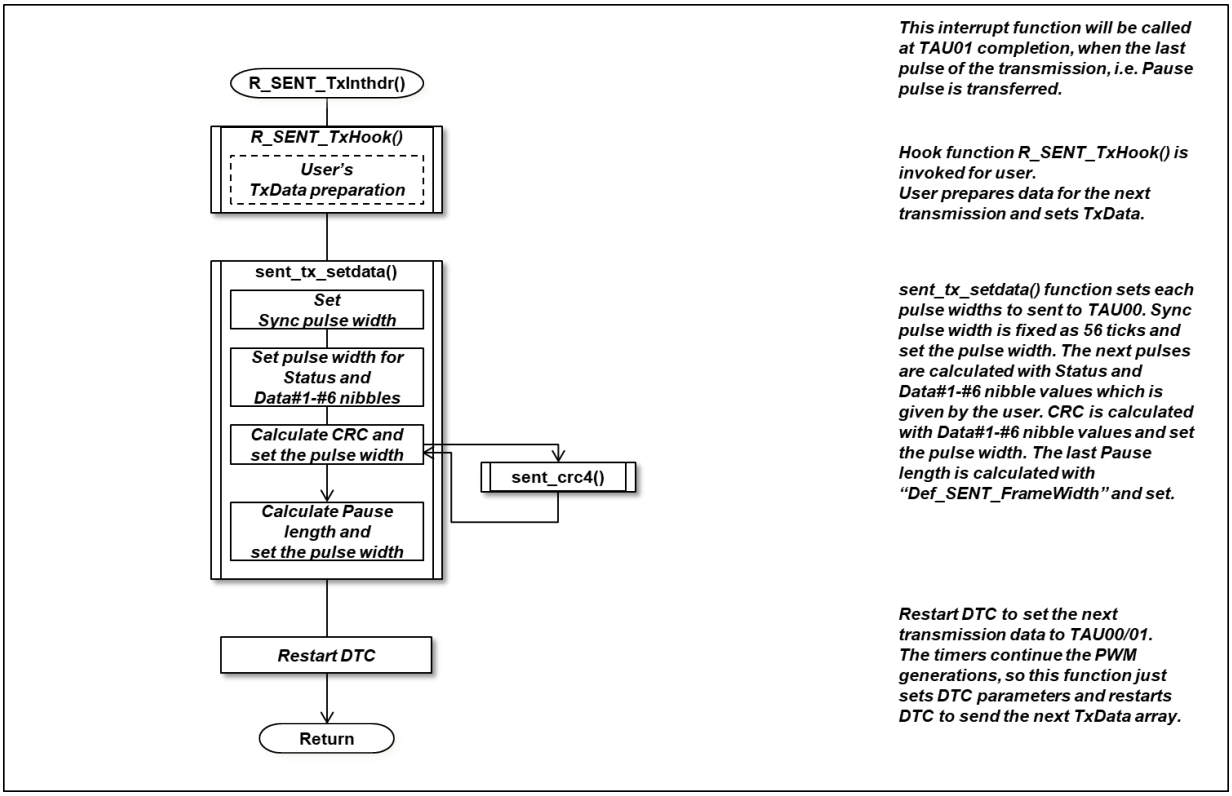

図 2-2 Process Flow for SENT Transmission Example
(Interrupt Processing Function: R_SENT_TxInthdr())

2.5　SENT 送信実装例の処理関数

本章では SENT 送信実装例の各処理関数について説明します。表 2-4 に SENT 送信の処理関数一覧を示します。

表 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 送信初期化

表 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　SENT 送信用タイマ・アレイ・ユニット（TAU00、TAU01）初期設定

表 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 表 2-1 about the settings.

2.5.3　SENT 送信用 DTC 初期設定

表 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 表 2-2 about the settings.

2.5.4　SENT 送信開始処理

表 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 送信停止処理

表 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　SENT 送信完了割り込み処理

表 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 図 2-2.

2.5.7　SENT 送信完了通知関数

表 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 表 2-3.

2.5.8　SENT 送信用データ設定処理

表 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 用 CRC 計算処理

表 2-13 SENT CRC Calculation Function

Syntax		uint8_t sent_crc4(uint8_t* pdata)	
Parameters	In	uint8_t* pdata	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/F2x operation function for SENT CRC calculation to optimize the speed.	

3.　SENT 受信の実装

本章では、RL78/F2x の SENT 受信機能実装について説明します。

3.1　SENT 受信処理の概要

図 3-1 に、SENT 受信実装例の処理概要を示します。タイマ・アレイ・ユニット TAU02 を用いて、P16 / TI02 端子に入力されるパルス信号の長さを測定します。INTTM02（TAU02 完了割り込み）は各パルス信号（Pause, Sync, Status, Data[#1 to #6], CRC）の長さ測定完了毎に発生します。CRC データの受信終了後、SENT 受信完了通知関数を呼び出してユーザに通知します。

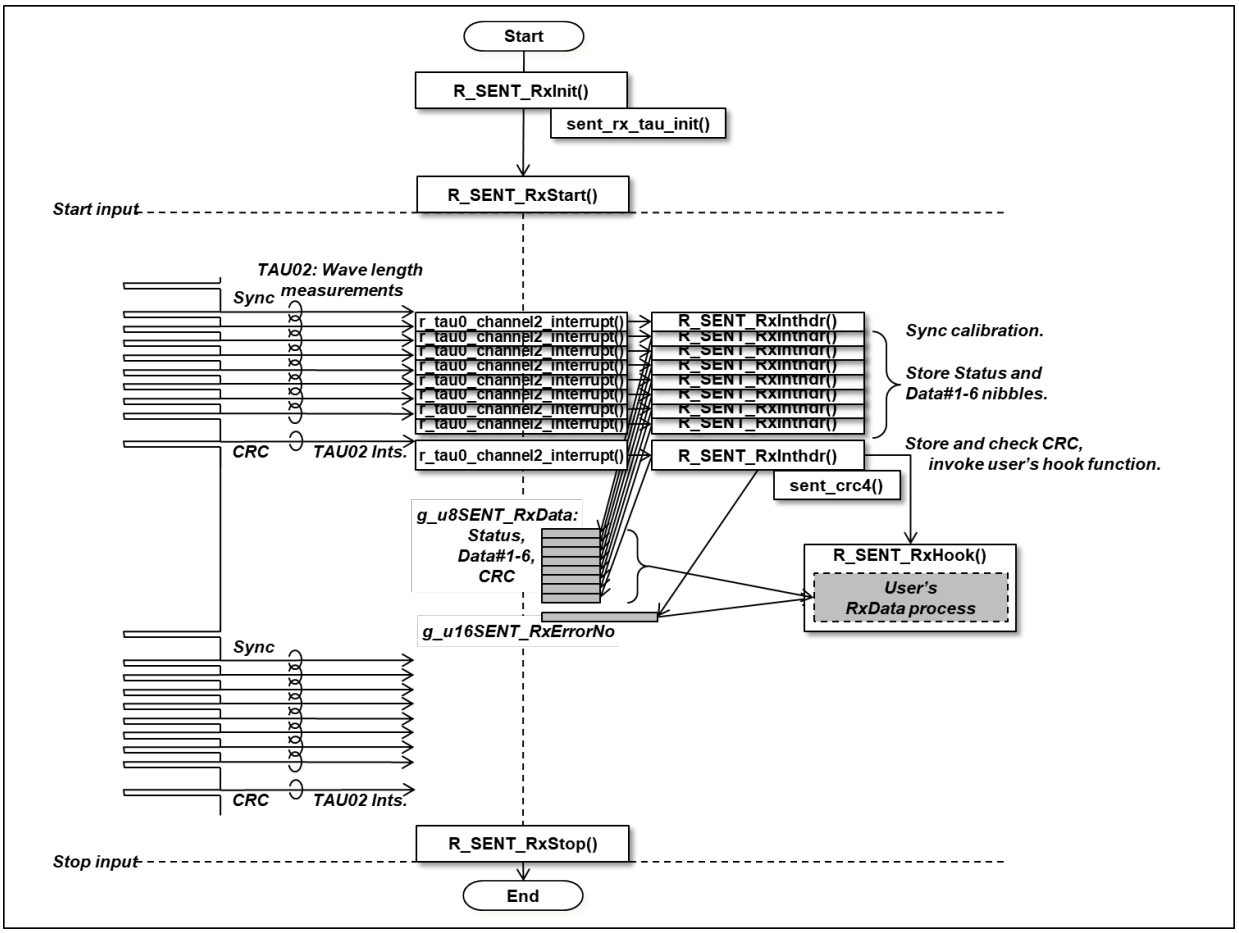

図 3-1 Process Overview of SENT Reception

3.2　SENT 受信時の SFR 設定

表 3-1 に SENT 受信実装例の RL78/F2x SFR 設定（TAU02 関連）を示します。

表 3-1 RL78/F2x SFR Settings for SENT Reception Example: TAU02 Settings

Register Name	Setting
TDR02	TI02 pulse width capture time for SENT reception
TPS0	0x3210U (CK00=40MHz, CK01=20MHz, CK02=10MHz, CK03=5MHz)
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　SENT 受信実装例の使用変数

本章では SENT 受信実装例の使用する変数について説明します。表 3-2 に SENT 受信の変数一覧を示します。

g_u8SENT_RxData[10]は API 用変数配列です。受信データを保持しており、ユーザは SENT 受信完了通知関数 R_SENT_RxHook()の中で読み出します。受信データは Status、Data[#1 to #6]及び CRC の各 nibble 値で、”0”～”15”の範囲です。g_u16SENT_RxErrorNo も API 用変数であり、SENT 受信処理のエラー状態を保持します。

表 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　SENT 受信の処理フロー

図 3-2 に SENT 受信実装例の処理（割り込み処理関数）を示します。この割り込み処理は各入力パルスを判断してそれぞれの対応を行い、正しく受信できたデータを保持します（3.1 章も参照ください）。

ユーザ用フック関数は、各フレームの最後の受信データである CRC を処理した後に呼び出されます。

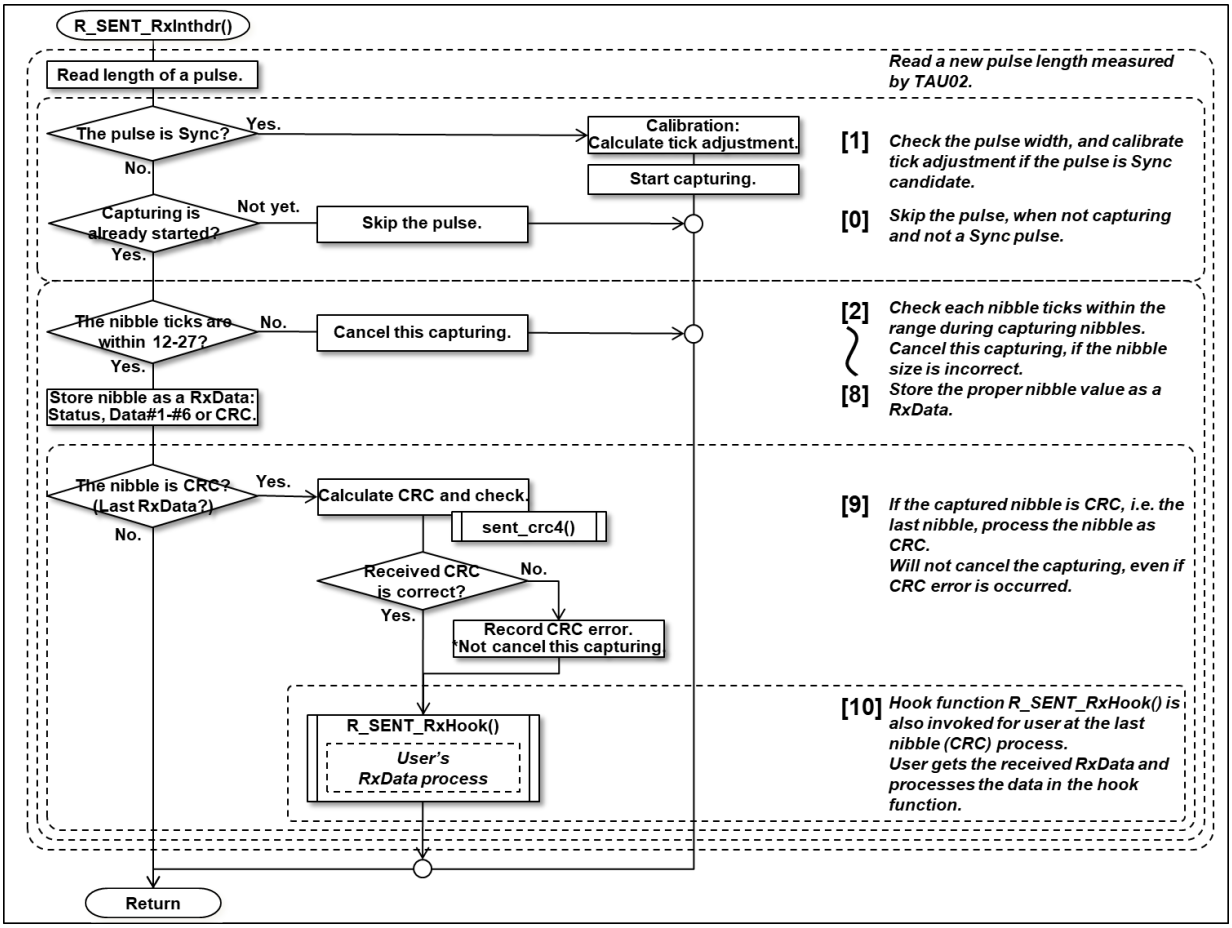

図 3-2 Process Flow for SENT Reception Example
(Interrupt Processing Function: R_SENT_RxInthdr())

図 3-3 に SENT 受信処理のタイミングチャートを示します。図中の各番号（“[0]” – “[10]”）は図 3-2 の処理フローの番号に対応しています。

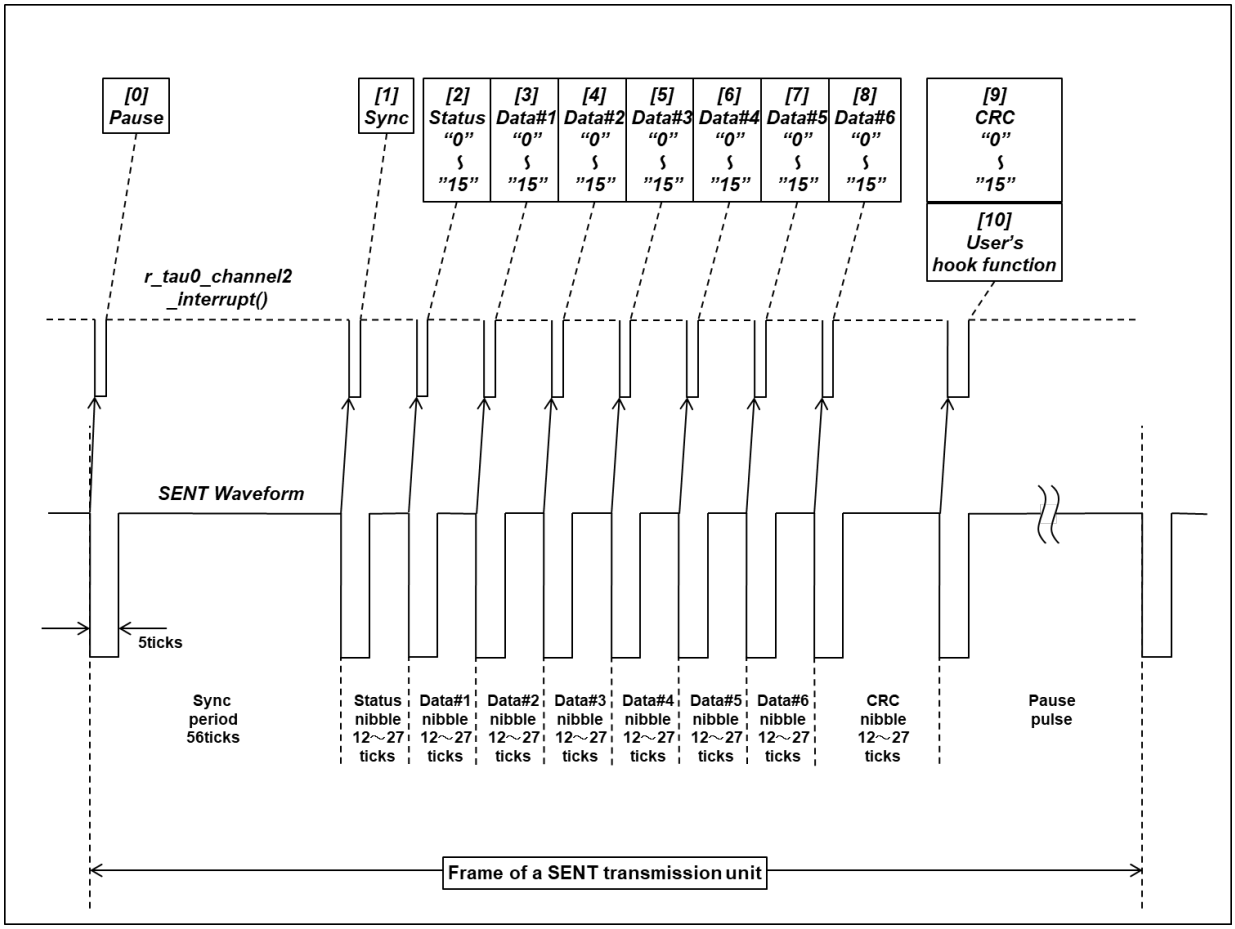

図 3-3 Timing Chart of SENT Reception Example Process

3.5　SENT 受信実装例の処理関数

本章では SENT 受信実装例の各処理関数について説明します。表 3-3 に SENT 受信の処理関数一覧を示します。

表 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 受信初期化

表 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　SENT 受信用タイマ・アレイ・ユニット（TAU02）初期設定

表 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 表 3-1 about the settings.

3.5.3　SENT 受信開始処理

表 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 受信停止処理

表 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　SENT 受信割り込み処理

表 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 図 3-2.

3.5.6　SENT 受信完了通知関数

表 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 表 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 用 CRC 計算処理

SENT 送信用の sent_crc4()関数と同じ処理内容です。2.5.9 章を参照ください。

4.　参考資料

本アプリケーションノートの参考資料を以下に示します。参照の際は、ルネサスエレクトロニクスホームページから最新版を入手してください。

- RL78/F23, F24 ユーザーズマニュアル ハードウェア編 Rev. 1.00
- RL78 ファミリユーザーズマニュアル ソフトウェア編 Rev. 2.30

また、SENT 通信仕様に関しまして、以下の資料の参照を推奨します。

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

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2022.09.30	ー	初版発行

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本ドキュメントおよびテクニカルアップデートを参照してください。

1. 静電気対策
   CMOS 製品の取り扱いの際は静電気防止を心がけてください。CMOS 製品は強い静電気によってゲート絶縁破壊を生じることがあります。運搬や保存の際には、当社が出荷梱包に使用している導電性のトレーやマガジンケース、導電性の緩衝材、金属ケースなどを利用し、組み立て工程にはアースを施してください。プラスチック板上に放置したり、端子を触ったりしないでください。また、CMOS 製品を実装したボードについても同様の扱いをしてください。
2. 電源投入時の処置
   電源投入時は、製品の状態は不定です。電源投入時には、LSI の内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。
3. 電源オフ時における入力信号
   当該製品の電源がオフ状態のときに、入力信号や入出力プルアップ電源を入れないでください。入力信号や入出力プルアップ電源からの電流注入により、誤動作を引き起こしたり、異常電流が流れ内部素子を劣化させたりする場合があります。資料中に「電源オフ時における入力信号」についての記載のある製品は、その内容を守ってください。
4. 未使用端子の処理
   未使用端子は、「未使用端子の処理」に従って処理してください。CMOS 製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI 周辺のノイズが印加され、LSI 内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。
5. クロックについて
   リセット時は、クロックが安定した後、リセットを解除してください。プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。
6. 入力端子の印加波形
   入力ノイズや反射波による波形歪みは誤動作の原因になりますので注意してください。CMOS 製品の入力がノイズなどに起因して、V_{IL}（Max.）から V_{IH}（Min.）までの領域にとどまるような場合は、誤動作を引き起こす恐れがあります。入力レベルが固定の場合はもちろん、V_{IL}（Max.）から V_{IH}（Min.）までの領域を通過する遷移期間中にチャタリングノイズなどが入らないように使用してください。
7. リザーブアドレス（予約領域）のアクセス禁止
   リザーブアドレス（予約領域）のアクセスを禁止します。アドレス領域には、将来の拡張機能用に割り付けられている リザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。
8. 製品間の相違について
   型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。同じグループのマイコンでも型名が違うと、フラッシュメモリ、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ幅射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。回路、ソフトウェアおよびこれらに関連する情報を使用する場合、お客様の責任において、お客様の機器・システムを設計ください。これらの使用に起因して生じた損害（お客様または第三者いずれに生じた損害も含みます。以下同じです。）に関し、当社は、一切その責任を負いません。
2. 当社製品または本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。
3. 当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を組み込んだ製品の輸出入、製造、販売、利用、配布その他の行為を行うにあたり、第三者保有の技術の利用に関するライセンスが必要となる場合、当該ライセンス取得の判断および取得はお客様の責任において行ってください。
5. 当社製品を、全部または一部を問わず、改造、改変、複製、リバースエンジニアリング、その他、不適切に使用しないでください。かかる改造、改変、複製、リバースエンジニアリング等により生じた損害に関し、当社は、一切その責任を負いません。
6. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。
   標準水準：　コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット等
   高品質水準：輸送機器（自動車、電車、船舶等）、交通制御（信号）、大規模通信機器、金融端末基幹システム、各種安全制御装置等
   当社製品は、データシート等により高信頼性、Harsh environment 向け製品と定義しているものを除き、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（宇宙機器と、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することは想定していません。たとえ、当社が想定していない用途に当社製品を使用したことにより損害が生じても、当社は一切その責任を負いません。
7. あらゆる半導体製品は、外部攻撃からの安全性を 100%保証されているわけではありません。当社ハードウェア／ソフトウェア製品にはセキュリティ対策が組み込まれているものもありますが、これによって、当社は、セキュリティ脆弱性または侵害（当社製品または当社製品が使用されているシステムに対する不正アクセス・不正使用を含みますが、これに限りません。）から生じる責任を負うものではありません。当社は、当社製品または当社製品が使用されたあらゆるシステムが、不正な改変、攻撃、ウイルス、干渉、ハッキング、データの破壊または窃盗その他の不正な侵入行為（「脆弱性問題」といいます。）によって影響を受けないことを保証しません。当社は、脆弱性問題に起因しまたはこれに関連して生じた損害について、一切責任を負いません。また、法令において認められる限りにおいて、本資料および当社ハードウェア／ソフトウェア製品について、商品性および特定目的との合致に関する保証ならびに第三者の権利を侵害しないことの保証を含め、明示または黙示のいかなる保証も行いません。
8. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合および事故につきましては、当社は、一切その責任を負いません。
9. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は、データシート等において高信頼性、Harsh environment 向け製品と定義しているものを除き、耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
10. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。かかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
11. 当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。当社製品および技術を輸出、販売または移転等する場合は、「外国為替及び外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。
12. お客様が当社製品を第三者に転売等される場合には、事前に当該第三者に対して、本ご注意書き記載の諸条件を通知する責任を負うものといたします。
13. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。
14. 本資料に記載されている内容または当社製品についてご不明な点がございましたら、当社の営業担当者までお問合せください。

注 1. 本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社が直接的、間接的に支配する会社をいいます。

注 2. 本資料において使用されている「当社製品」とは、注 1 において定義された当社の開発、製造製品をいいます。

(Rev.5.0-1　2020.10)

本社所在地

〒135-0061 東京都江東区豊洲 3-2-24（豊洲フォレシア）

www.renesas.com

お問合せ窓口

弊社の製品や技術、ドキュメントの最新情報、最寄の営業お問合せ窓口に関する情報などは、弊社ウェブサイトをご覧ください。

www.renesas.com/contact/

商標について

ルネサスおよびルネサスロゴはルネサス エレクトロニクス株式会社の商標です。すべての商標および登録商標は、それぞれの所有者に帰属します。

© 2022 Renesas Electronics Corporation. All rights reserved.