RZ/A2M グループ

RS-CANFD サンプルプログラムアプリケーションノート

要旨

本書は RZ/A2M グループ MCU 用の RS-CANFD Driver を使用し、CAN FD 通信を行うサンプルプログラムについて説明します。

動作確認デバイス

RZ/A2M

本書で説明する RS-CANFD サンプルプログラムは、既製のルネサス製 RZ/A2M SUB ボード（RTK79210XXB00000BE）では動作しません。別途、CAN トランシーバと CAN クロック用の発振器の実装が必要になります。

目次

1. 概要

RS-CANFD サンプルプログラム アプリケーションノート(以下サンプルプログラム)は、RZ/A2M の CANFD インタフェース(RS-CANFD)用 Driver を使用した CAN FD 通信の動作例です。動作例として下記の 3 項目を記載しています。

表 1-1　動作例一覧

#	内容	RS-CANFD の使用リソース
1	送受信動作	送信側：送受信 FIFO 受信側：受信 FIFO
2	ゲートウェイ動作	送受信 FIFO
3	外部ループバックテスト	送信バッファ 受信バッファ

API の詳細については、RZ/A2M グループ RS-CANFD ドライバ アプリケーションノート(R01AN4868)を参照してください。なお、本サンプルプログラムでは以下の API のみ使用しています。

表 1-2　使用 API 一覧

API 名	機能	動作例 1	動作例 2	動作例 3
R_CAN_Create	初期設定	○	○	○
R_CAN_Control	動作モード遷移	○	○	○
R_CAN_PortSet	ポート機能/ポートテストの設定	○	○	○
R_CAN_SetBitrate（注）	ビットレート設定	○	○	○
R_CAN_TxSet	送信データ設定	○		○
R_CAN_Tx	送信要求設定	○		○
R_CAN_TxCheck	送信完了チェック			○
R_CAN_RxSet	受信許可設定	○		○
R_CAN_RxPoll	受信完了チェック	○		○
R_CAN_RxRead	受信データリード	○		○
R_CAN_Gateway	ゲートウェイ動作許可		○	

【注】R_CAN_SetBitrate は R_CAN_Create 内部で使用。

表 1-3　使用する周辺機能と用途

周辺機能	用途
CANFD インタフェース(RS-CANFD)	CAN FD 通信用モジュール
割り込みコントローラ(INTC)	RS-CANFD の割り込みの設定
汎用入出力ポート(GPIO)	RS-CANFD 端子機能の設定

2. 動作確認条件

本アプリケーションノートのサンプルコードは、下記の条件で動作を確認しています。

表 2-1　動作確認条件

項目	内容
使用 MCU	RZ/A2M
動作周波数（注 1）	CPU クロック（Iφ）：528MHz 画像処理クロック（Gφ）：264MHz 内部バスクロック（Bφ）：132MHz 周辺クロック 1（P1φ）：66MHz 周辺クロック 0（P0φ）：33MHz QSPI0_SPCLK：66MHz CKIO：132MHz CAN_CLK：32MHz（注 2）
動作電圧	電源電圧（I/O）：3.3V 電源電圧（1.8/3.3V 切替 I/O（PVcc_SPI））：3.3V 電源電圧（内部）：1.2V
統合開発環境	e2 studio V7.4.0
C コンパイラ	GNU Arm Embedded Toolchain Version8-2018-q4-major
	コンパイラオプション（ディレクトリパスの追加は除く） Hardware Debug: -mcpu=cortex-a9 -march=armv7-a –marm -mlittle-endian -mfloat-abi=hard -mfpu=neon -Os -ffunction-sections -fdata-sections -Wnull-dereference -g3 -Wstack-usage=100 -std=gnu99 -fdiagnostics-parseable-fixits
エミュレータ	SEGGER Microcontroller 製 J-Link LITE または J-Link BASE
動作モード	ブートモード 3 （シリアルフラッシュブート 3.3V 品）
使用ボード（注 3）	RZ/A2M CPU ボード　RTK7921053C00000BE RZ/A2M SUB ボード　RTK79210XXB00000BE
使用デバイス （ボード上で 使用する機能）	• シリアルフラッシュメモリ（SPI マルチ I/O バス空間に接続） メーカ名：Macronix 社、型名：MX25L51245GXD • CAN トランシーバ メーカ名：NXP 社、型名：TJF1051T/3

【注 1】クロックモード 1（EXTAL 端子からの 24MHz のクロック入力）で使用時の動作周波数です。

【注 2】CAN クロックソースは外部クロックを使用します。

【注 3】CAN トランシーバと CAN クロック用の発振器は未実装のため、CAN 使用時は実装する必要があります。

2.1　システム構成

2.1.1　ハードウェア

本アプリケーションノートにおける、ハードウェアの構成図を図 2-1 に示します。送受信動作は各ボードのチャネル 0 のみを使用します。ゲートウェイ動作は各ボードのチャネル 0、1 を両方使用します。外部ループバックテストは 1 台のボードのチャネル 0 のみを使用します。詳細は、各動作例の動作概要を参照してください。

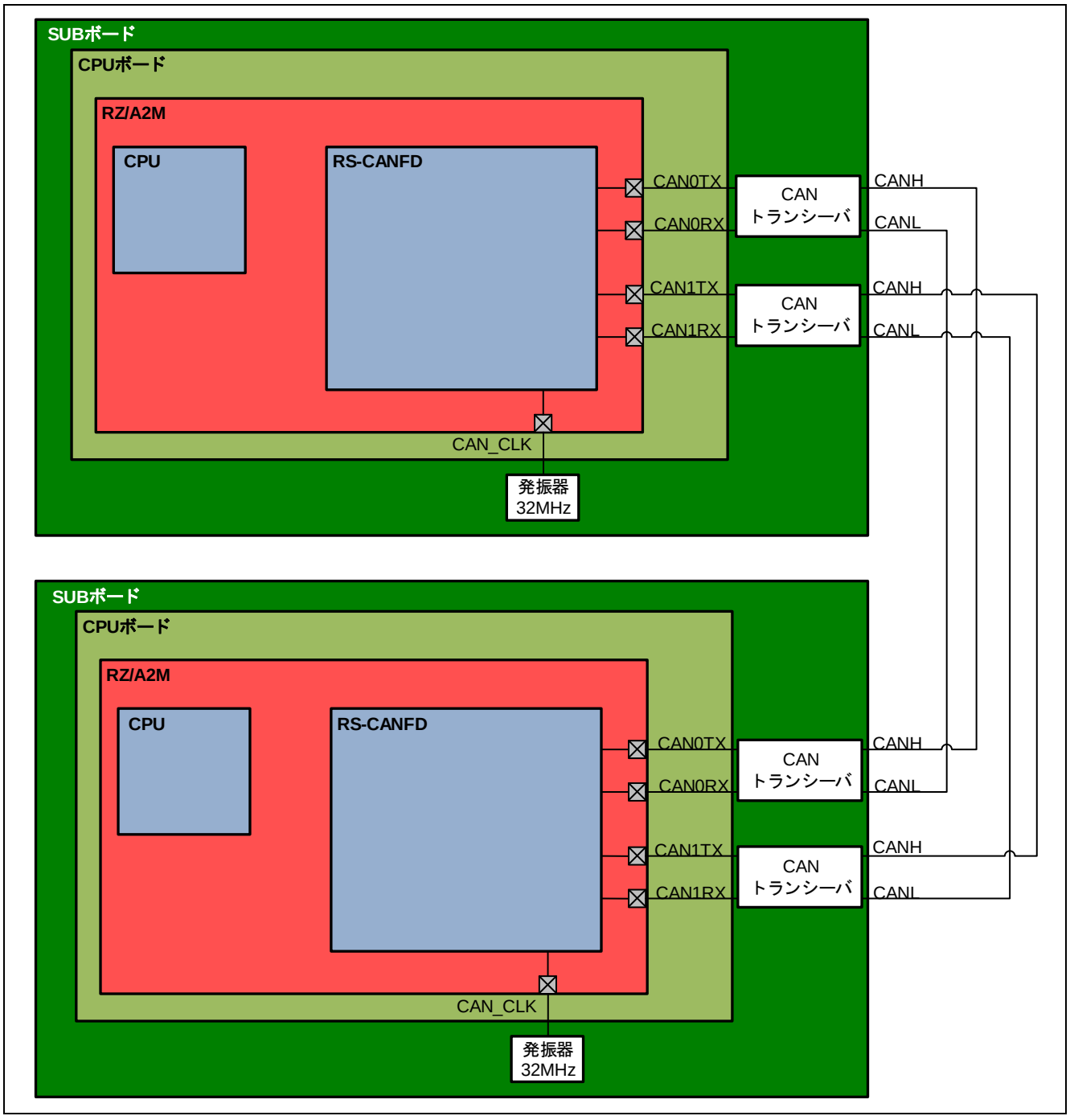


図 2-1 ハードウェア構成

RZ/A2M SUB ボードには、CAN トランシーバと CAN クロック用の発振器が未実装のため、CAN 使用時は実装する必要があります。図 2-2 に RZ/A2M SUB ボードの回路図の一部を抜粋します。点線部が未実装部分となります。

詳細は、RTK79210XXB00000BE（RZ/A2M SUB ボード）ユーザーズマニュアルを参照してください。

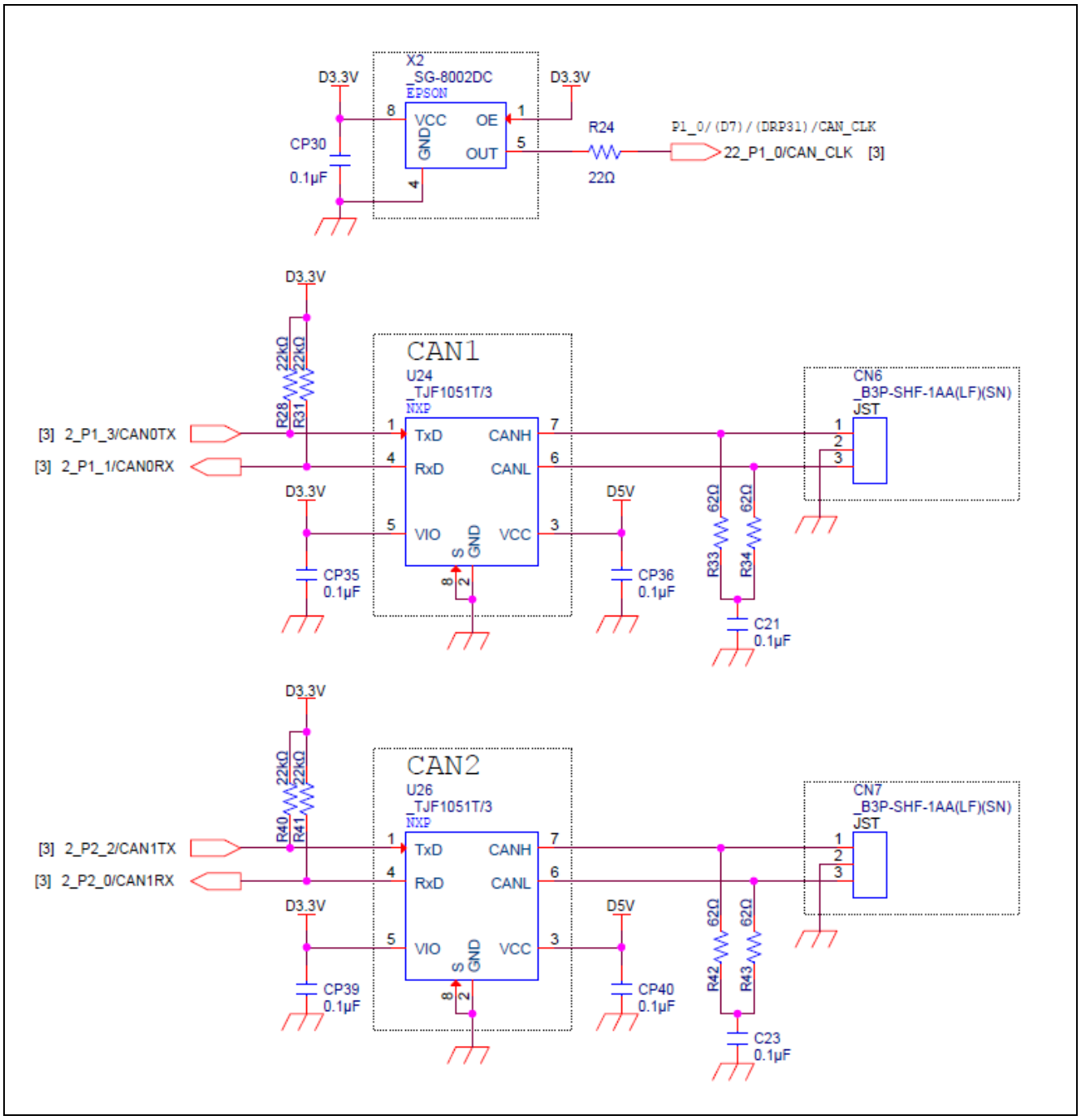


図 2-2 RZ/A2M SUB ボードの回路図(抜粋)

2.1.2　ソフトウェア

本サンプルプログラムのシステムブロックを図 2-3 に記載します。本サンプルプログラムは、CAN の送受信を制御するタスクと、割り込み要求により実行されるタスクが存在します。

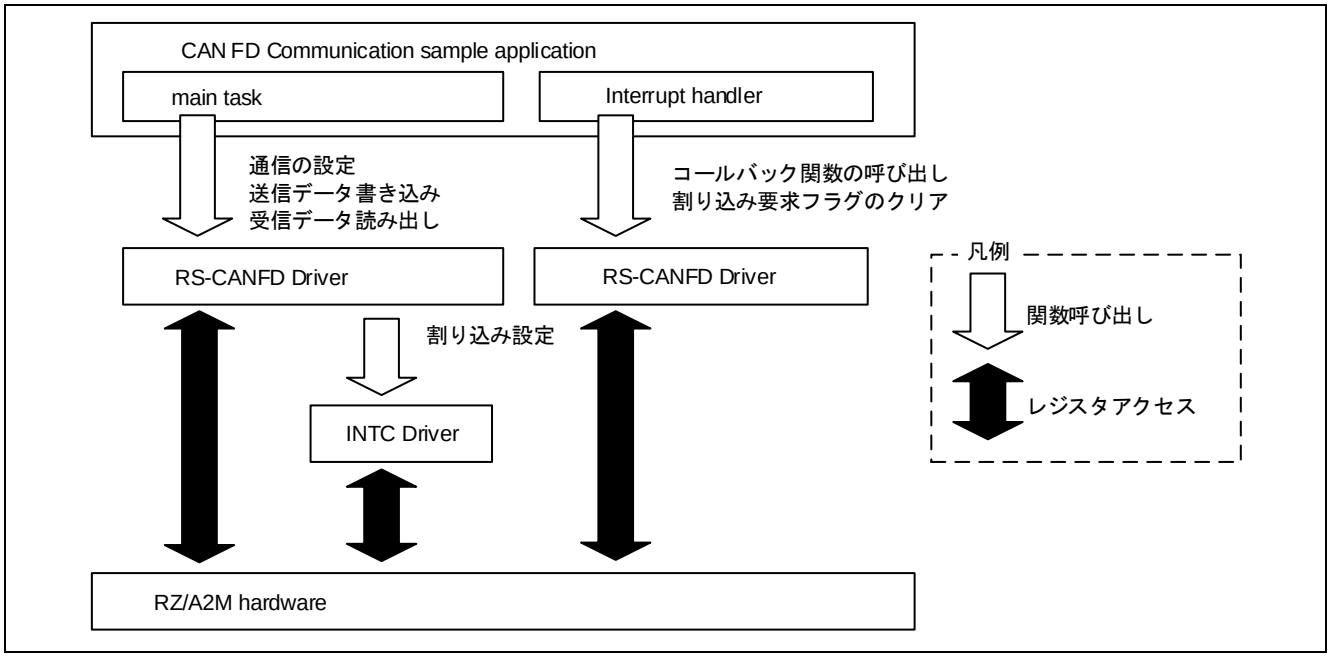


図 2-3 サンプルプログラムのシステムブロック

本サンプルプログラムのフォルダ構成を、表 2-2 に記載します。

表 2-2 フォルダ構成

フォルダ名	プロジェクト名	実行ファイル名	デバッグ構成名	説明
Transmit_64byte	Transmit_64byte	Transmit_64byte.elf	Transmit_64byte Hardware Debug	動作例 1 送信側サンプルプロジェクト
Receive_64byte	Receive_64byte	Receive_64byte.elf	Receive_64byte Hardware Debug	動作例 1 受信側サンプルプロジェクト
Gateway	Gateway	Gateway.elf	Gateway Hardware Debug	動作例 2 ゲートウェイ側サンプルプロジェクト
Gateway_Opposite	Gateway_Opposite	Gateway_Opposite.elf	Gateway_Opposite Hardware Debug	動作例 2 対向ボード側サンプルプロジェクト
Loopback_Test	Loopback_Test	Loopback_Test.elf	Loopback_Test Hardware Debug	動作例 3 サンプルプロジェクト

3. 動作説明

3.1 共通設定

本章では、RS-CANFD ドライバを使用した動作例 1～3 で、共通で行っている設定内容について記載します。

3.1.1 CAN 状態（モード）

RS-CANFD モジュールには、RS-CANFD モジュール全体の状態を制御するグローバルモードが 4 種類と、個々のチャネル状態を制御するチャネルモードが 4 種類あります。

- グローバルストップモード：モジュール全体のクロックを停止させ、低消費電力を実現します。
- グローバルリセットモード：モジュール全体の初期設定を行います。
- グローバルテストモード：テスト設定を行います。また、RAM テストを実施します。
- グローバル動作モード：モジュール全体を動作可能にします。

- チャネルストップモード：チャネルのクロックが停止します。
- チャネルリセットモード：チャネルの初期設定を行います。
- チャネル待機モード：CAN 通信を停止、チャネルのテストを許可します。
- チャネル通信モード：CAN 通信を行います。

図 3-1 に、本サンプルプログラムにおけるグローバルモードの状態遷移図を記載します。MCU リセット後、グローバルモードはグローバルストップモードになっています。初期設定(R_CAN_Create 関数)を実行すると、中間状態としてグローバルリセットモードを経由し、グローバルテストモードに遷移します。その後、実際に送受信を実行するため、グローバル動作モードに遷移させます。グローバル動作モードへの遷移には、通常 R_CAN_Control 関数を使用します(動作例 1、2 の場合)が、CAN ポートテストの設定(R_CAN_PortSet 関数の引数にポートテストを指定の場合)でも、グローバル動作モードに遷移します(動作例 3 の場合)。

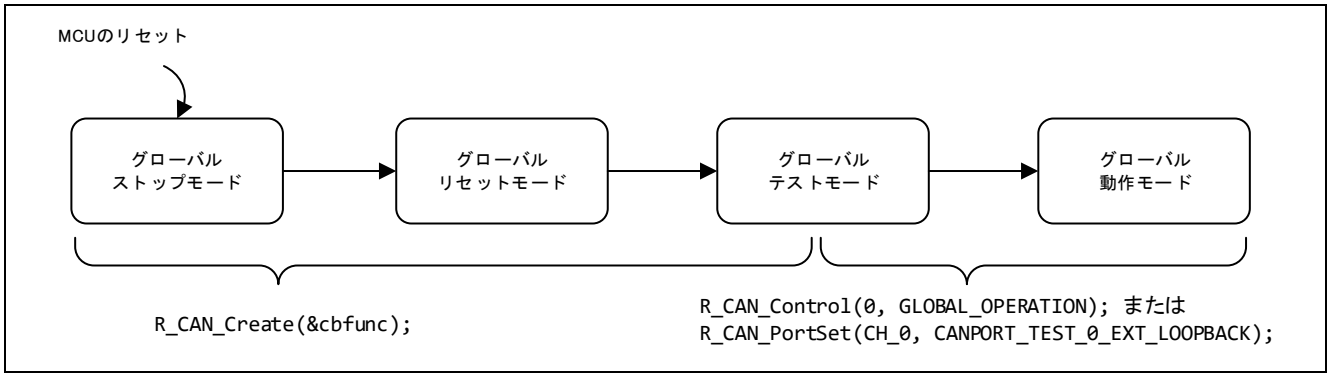


図 3-1 グローバルモードの状態遷移図

図 3-2 に、本サンプルプログラムにおけるチャネルモードの状態遷移図を記載します。MCU リセット後、チャネルモードはチャネルストップモードになっています。初期設定(R_CAN_Create 関数)を実行すると、中間状態としてチャネルリセットモードを経由し、チャネル待機モードに遷移します。その後、実際に送受信を実行するため、チャネル通信モードに遷移させます。チャネル通信モードへの遷移には、通常 R_CAN_Control 関数を使用します(動作例 1、2 の場合)が、CAN ポートテストの設定(R_CAN_PortSet 関数の引数にポートテストを指定の場合)でも、チャネル通信モードに遷移します(動作例 3 の場合)。

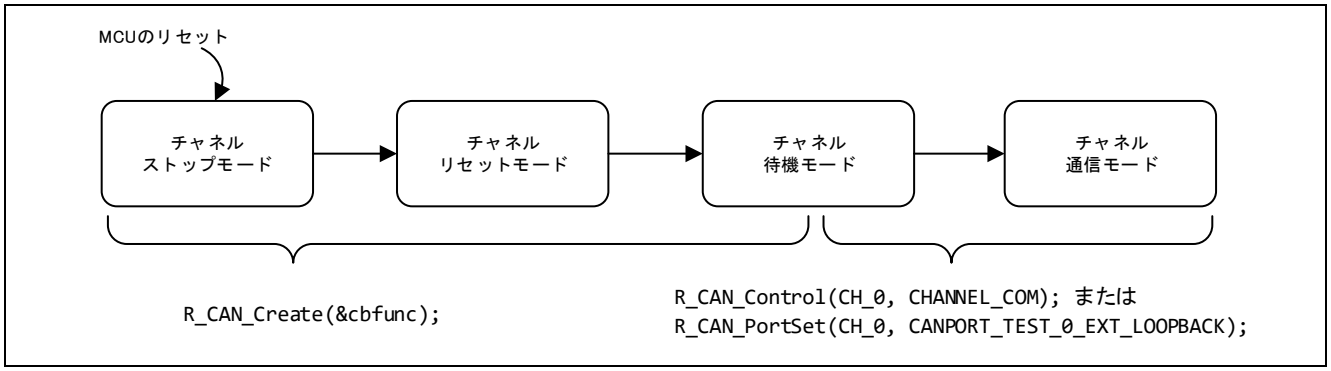


図 3-2 チャネルモードの状態遷移図

3.1.2　通信速度

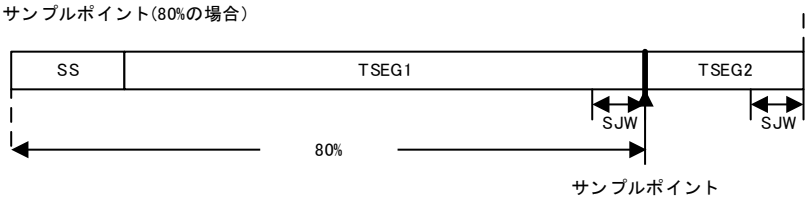
3.1.2.1　CAN ビットタイミングの設定

本 RS-CANFD モジュールの CAN ビットタイミング設定では、通信フレームの 1 ビットを 3 つのセグメントで構成しています。図 3-3 にビットのセグメントとサンプルポイントを示します。

これらのセグメントのうち、Time Segment 1（以下、TSEG1 という）、Time Segment 2（以下、TSEG2 という）は、サンプルポイントを指定するもので、これらの値を変えることでサンプリングするタイミングを変更できます。CAN FD モードは、2 種類のビットレート（通常ビットレートとデータビットレート）がありそれぞれを設定します。

このタイミング設定の最小単位を 1 Time Quanta（以下、Tq という）といい、RS-CANFD モジュールに入力されたクロック周波数とボーレートプリスケーラ分周値で決められます。

表 3-1 にビットタイミングの設定例を示します。ビットタイミングは『r_can_rz_config.h』で定義されたマクロで設定します。本サンプルプログラムでは通常ビットレートのビットタイミングを 1bit=32Tq、サンプルポイント 59.38%、データビットレートのビットタイミングを 1bit=16Tq、サンプルポイント 62.5%に設定しています。



クラシカルCANモード:
SS = 1Tq固定
TSEG1 = 4～16Tqの範囲で設定
TSEG2 = 2～8Tqの範囲で設定
SJW = 1～4Tqの範囲で設定
SS + TSEG1 + TSEG2 = 8～25Tqの範囲で設定
TSEG1 ＞ TSEG2 ≧ SJW

CAN FDモード:
通常ビットレート
SS = 1Tq固定
TSEG1 = 4～128Tqの範囲で設定
TSEG2 = 2～32Tqの範囲で設定
SJW = 1～32Tqの範囲で設定
SS + TSEG1 + TSEG2 = 8～161Tqの範囲で設定
TSEG1 ＞ TSEG2 ≧ SJW

データビットレート
SS = 1Tq固定
TSEG1 = 2～16Tqの範囲で設定
TSEG2 = 2～8Tqの範囲で設定
SJW = 1～8Tqの範囲で設定
SS + TSEG1 + TSEG2 = 8～25Tqの範囲で設定
TSEG1 ≧ TSEG2 ≧ SJW

・SS(同期セグメント):
インターフレームスペース中に、レセシブからドミナントへのエッジをモニタして同期をとるセグメントです。
インターフレームスペース(Interframe Space)は、インターミッション(Intermission)、サスペンドトランスミッション(Suspend Transmission)、バスアイドル(Bus Idle)で構成されます。バスアイドル中では、全ノードが送信を開始することができます。
・TSEG1（タイムセグメント1）:
CANネットワーク上の物理的な遅延を吸収するセグメントです。ネットワーク上の物理的な遅延は、CANバス上の遅延、入力コンパレータ遅延および出力ドライバ遅延の総和の2倍になります。
・TSEG2（タイムセグメント2）:
周波数の誤差によるフェーズエラーを補償するセグメントです。
・SJW（再同期ジャンプ幅）:
フェーズエラーによっておこる位相誤差を補償するために、タイムセグメントを延長または短縮する長さです。

図 3-3 ビットのセグメント構成とサンプルポイント

表 3-1　ビットタイミング設定例

1 ビット	設定値（注）			サンプル ポイント（%）
	CmNCFG_NTSEG1 CmDCFG_DTSEG1	CmNCFG_NTSEG2 CmDCFG_DTSEG2	CmNCFG_NSJW CmNCFG_DSJW	
8Tq	3	2	1	62.50
	4	1	1	75.00
16Tq	8	5	2	62.50
	9	4	2	68.75
	10	3	2	75.00
32Tq	20	9	2	68.75
	22	7	2	75.00
	17	12	2	59.38

【注】m：チャネル番号(0, 1)
通常ビットレートは RSCFDnCFDCmNCFG レジスタに、データビットレートは
RSCFDnCFDCmDCFG レジスタの各ビットに対応しています。

3.1.2.2 通信速度の設定

通信速度は、RS-CANFD モジュールのクロック源である CAN クロック(fCAN)、ボーレートプリスケーラ分周値、および 1 ビットの Tq 数で決まります。fCAN は clkc(内部クロック P1φ/2)と clk_xincan(外部クロック CAN_CLK)のいずれかを選択可能です。

クロック選択は『r_can_rz_config.h』に定義のマクロ GCFG_DCS(0: clkc, 1: clk_xincan)で行います。通常ビットレートプリスケーラ分周値はマクロ CmNCFG_NBRP(m=0,1)、データビットレートプリスケーラ分周値はマクロ CmDCFG_DBRP(m=0,1)で設定します。各マクロの設定値に 1 を足したものが分周値になります。CmNCFG_NBRP と CmDCFG_DBRP には必ず同じ値を設定してください。

表 3-2、表 3-3 に主な通信速度の算出式と実現例を示します。本サンプルプログラムは、fCAN に clk_xincan 32MHz を使用し、通常ビットレート 1 Mbps、データビットレート 2Mbps で動作しています。

表 3-2　通信速度の設定例（通常ビットレート）

通信速度の算出式	fCAN 通常ビットレートプリスケーラ分周比(注 1)×1 通常ビットタイムの Tq 数		
fCAN 通信速度	32MHz	16MHz	8MHz
1Mbps	16Tq(2) 32Tq(1) (注 2)	8Tq(2) 16Tq(1)	8Tq(1)
500Kbps	8Tq(8) 16Tq(4)	8Tq(4) 16Tq(2)	8Tq(2) 16Tq(1)
250Kbps	8Tq(16) 16Tq(8)	8Tq(8) 16Tq(4)	8Tq(4) 16Tq(2)
125Kbps	8Tq(32) 16Tq(16)	8Tq(16) 16Tq(8)	8Tq(8) 16Tq(4)

【注 1】 ( )内の数字はボーレートプリスケーラ分周値、各マクロの設定値にはボーレートプリスケーラ分周値から 1 を引いて設定してください。

【注 2】 本サンプルプログラムでの設定

表 3-3　通信速度の設定例（データビットレート）

通信速度の算出式	fCAN データビットプリスケーラ分周比(注 1)×1 データビットタイムの Tq 数	
fCAN 通信速度	32MHz	16MHz
4Mbps	8Tq(1)	なし
2Mbps	16Tq(1) (注 2)	8Tq(1)

【注 1】 ( )内の数字はボーレートプリスケーラ分周値、各マクロの設定値にはボーレートプリスケーラ分周値から 1 を引いて設定してください。

【注 2】 本サンプルプログラムでの設定

3.2　動作例 1 送受信動作

3.2.1　動作概要

本動作例では RZ/A2M ボード 2 台を使用した送信と受信を行います。送信側は送受信 FIFO バッファを使用して 64Byte データを連続して 2 回送信、受信側は受信 FIFO バッファを使用して 2 回受信します。

送信データは、1 データずつ設定する必要があります。そのため、2 回目の送信データをセットする前に、メッセージの送信を実行しています。

受信データの読み出しは、受信 FIFO 割り込みの、割り込み処理内で行います。受信 FIFO のバッファ段数を 8 メッセージ分に設定し、FIFO バッファに 2/8 までメッセージ格納時、割り込み要求が発生するように設定しています。

図 3-4 に本動作例のシステム構成を示します。

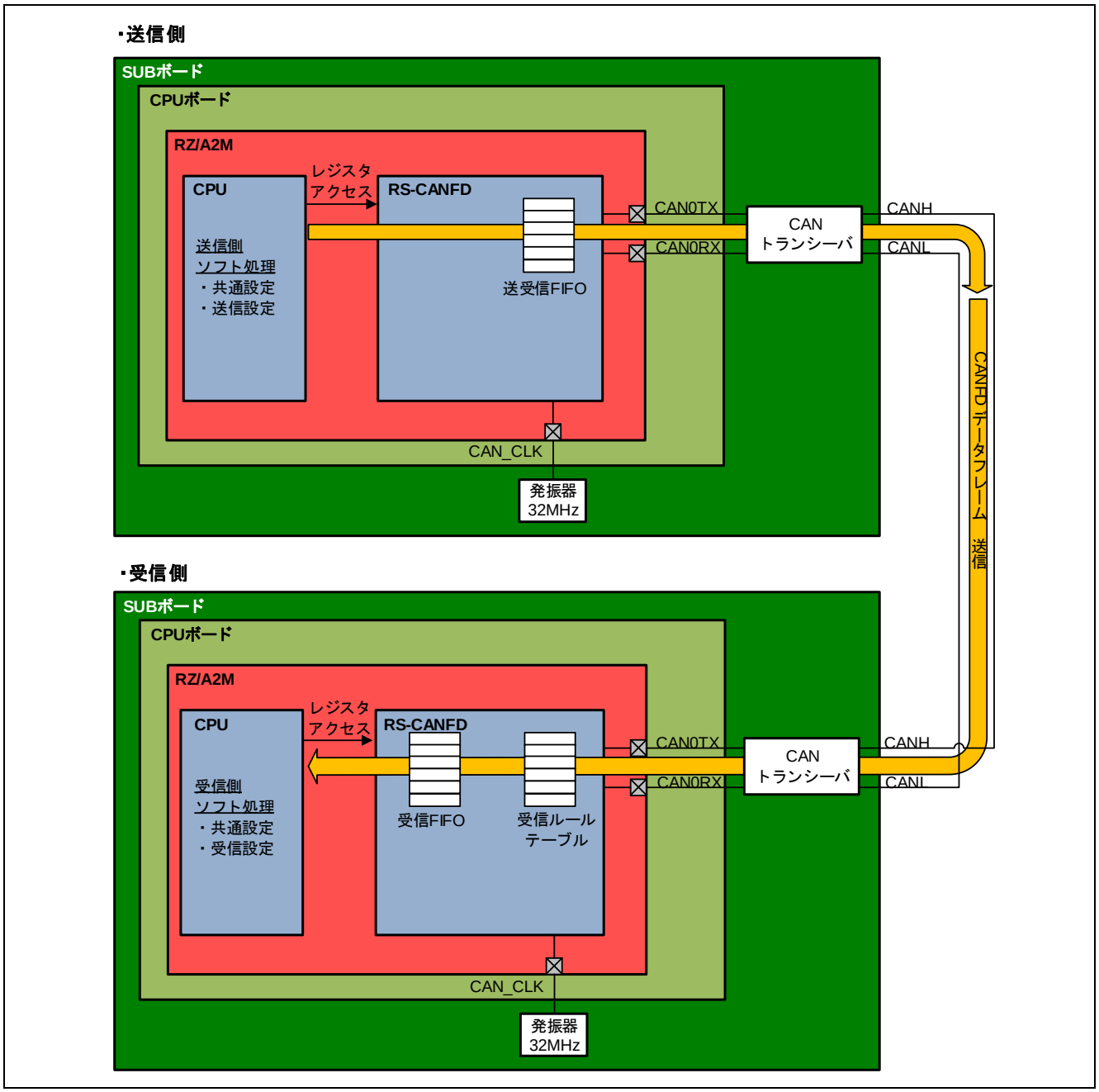


図 3-4 システム構成

送受信動作の、動作確認手順を以下に示します。送受信動作の動作確認は、受信側のプログラムで行います。

- 操作方法
  1. お互いのボードの、CH0（CAN1 のコネクタ）の CANH(1 番ピン)同士、CANL(3 番ピン)同士を接続します。
  2. 受信側のプログラムの、メイン関数の最後の while 文にブレークポイントを設定します。また、受信側のプログラムを実行し、受信待ち状態にします。
  3. 送信側のプログラムを実行し、送信を行います。

- 確認内容
  1. 受信側のプログラムが、ブレークポイントに到達し、終了していることを確認します。
  2. また、受信側で、受信データ格納用の配列（rx_buf_table）の内容を確認し、送信データと一致することを確認します。

- 確認結果

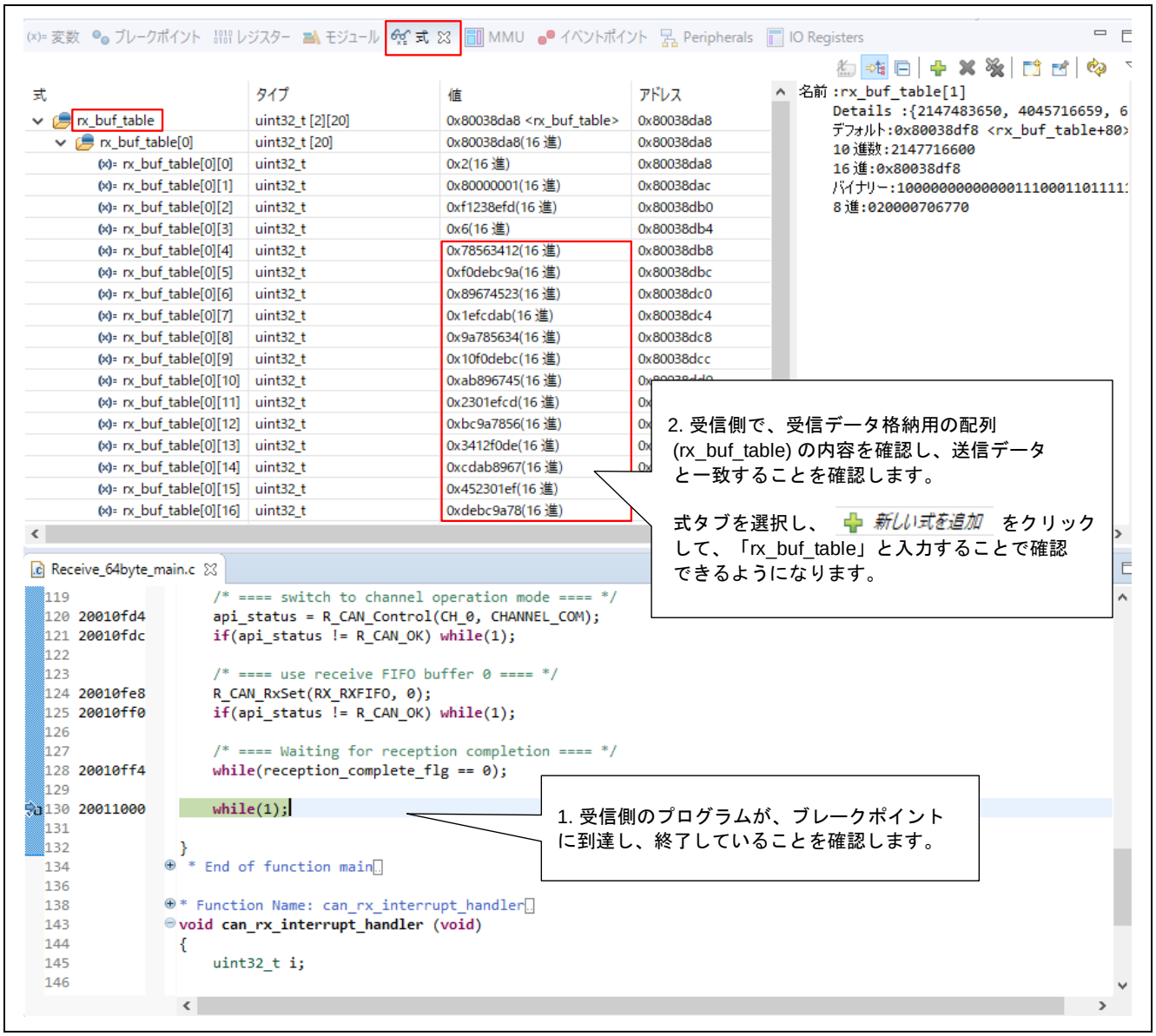


図 3-5 確認結果

式	タイプ	値	アドレス
rx_buf_table	uint32_t [2][20]	0x80038e48 <rx_buf_table>	0x80038e48
rx_buf_table[0]	uint32_t [20]	0x80038e48(16 進)	0x80038e48
(x)= rx_buf_table[0][0]	uint32_t	2	0x80038e48
(x)= rx_buf_table[0][1]	uint32_t	2147483649	0x80038e4c
(x)= rx_buf_table[0][2]	uint32_t	4045608611	0x80038e50
(x)= rx_buf_table[0][3]	uint32_t	6	0x80038e54
(x)= rx_buf_table[0][4]	uint32_t	0x78563412(16 進)	0x80038e58
(x)= rx_buf_table[0][5]	uint32_t	0xf0debc9a(16 進)	0x80038e5c
(x)= rx_buf_table[0][6]	uint32_t	0x89674523(16 進)	0x80038e60
(x)= rx_buf_table[0][7]	uint32_t	0x1efcdab(16 進)	0x80038e64
(x)= rx_buf_table[0][8]	uint32_t	0x9a785634(16 進)	0x80038e68
(x)= rx_buf_table[0][9]	uint32_t	0x10f0debc(16 進)	0x80038e6c
(x)= rx_buf_table[0][10]	uint32_t	0xab896745(16 進)	0x80038e70
(x)= rx_buf_table[0][11]	uint32_t	0x2301efcd(16 進)	0x80038e74
(x)= rx_buf_table[0][12]	uint32_t	0xbc9a7856(16 進)	0x80038e78
(x)= rx_buf_table[0][13]	uint32_t	0x3412f0de(16 進)	0x80038e7c
(x)= rx_buf_table[0][14]	uint32_t	0xcdab8967(16 進)	0x80038e80
(x)= rx_buf_table[0][15]	uint32_t	0x452301ef(16 進)	0x80038e84
(x)= rx_buf_table[0][16]	uint32_t	0xdebc9a78(16 進)	0x80038e88
(x)= rx_buf_table[0][17]	uint32_t	0x563412f0(16 進)	0x80038e8c
(x)= rx_buf_table[0][18]	uint32_t	0xefcdab89(16 進)	0x80038e90
(x)= rx_buf_table[0][19]	uint32_t	0x67452301(16 進)	0x80038e94
rx_buf_table[1]	uint32_t [20]	0x80038e98(16 進)	0x80038e98
(x)= rx_buf_table[1][0]	uint32_t	2147483650	0x80038e98
(x)= rx_buf_table[1][1]	uint32_t	4045685882	0x80038e9c
(x)= rx_buf_table[1][2]	uint32_t	6	0x80038ea0
(x)= rx_buf_table[1][3]	uint32_t	0xefcdab89(16 進)	0x80038ea4
(x)= rx_buf_table[1][4]	uint32_t	0x67452301(16 進)	0x80038ea8
(x)= rx_buf_table[1][5]	uint32_t	0xdebc9a78(16 進)	0x80038eac
(x)= rx_buf_table[1][6]	uint32_t	0x563412f0(16 進)	0x80038eb0
(x)= rx_buf_table[1][7]	uint32_t	0xcdab8967(16 進)	0x80038eb4
(x)= rx_buf_table[1][8]	uint32_t	0x452301ef(16 進)	0x80038eb8
(x)= rx_buf_table[1][9]	uint32_t	0xbc9a7856(16 進)	0x80038ebc
(x)= rx_buf_table[1][10]	uint32_t	0x3412f0de(16 進)	0x80038ec0
(x)= rx_buf_table[1][11]	uint32_t	0xab896745(16 進)	0x80038ec4
(x)= rx_buf_table[1][12]	uint32_t	0x2301efcd(16 進)	0x80038ec8
(x)= rx_buf_table[1][13]	uint32_t	0x9a785634(16 進)	0x80038ecc
(x)= rx_buf_table[1][14]	uint32_t	0x10f0debc(16 進)	0x80038ed0
(x)= rx_buf_table[1][15]	uint32_t	0x89674523(16 進)	0x80038ed4
(x)= rx_buf_table[1][16]	uint32_t	0x1efcdab(16 進)	0x80038ed8
(x)= rx_buf_table[1][17]	uint32_t	0x78563412(16 進)	0x80038edc
(x)= rx_buf_table[1][18]	uint32_t	0xf0debc9a(16 進)	0x80038ee0
(x)= rx_buf_table[1][19]	uint32_t	0x0(16 進)	0x80038ee4

データ1

データ2

図 3-6 受信データ

表 3-4 送信側 仕様一覧

項目			仕様
使用チャネル			チャネル 0
使用端子			P1_0: CAN_CLK P1_1: CAN0RX P1_3: CAN0TX
送信用バッファ			送受信 FIFO バッファ 0
通信速度			通常ビットレート：1Mbps、データビットレート：2Mbps
送信履歴			OFF
送信データ設定			
	データ 1	ID ビット	ID：0x00000001
		RTR ビット	データフレーム
		IDE ビット	拡張 ID
		DLC ビット	64Byte
		FDSTS ビット	CANFD フレーム
		BRS ビット	ビットレート可変
		送信データ	0x12,0x34,0x56,0x78,0x9a,0xbc,0xde,0xf0, 0x23,0x45,0x67,0x89,0xab,0xcd,0xef,0x01, 0x34,0x56,0x78,0x9a,0xbc,0xde,0xf0,0x10, 0x45,0x67,0x89,0xab,0xcd,0xef,0x01,0x23, 0x56,0x78,0x9a,0xbc,0xde,0xf0,0x12,0x34, 0x67,0x89,0xab,0xcd,0xef,0x01,0x23,0x45, 0x78,0x9a,0xbc,0xde,0xf0,0x12,0x34,0x56, 0x89,0xab,0xcd,0xef,0x01,0x23,0x45,0x67
	データ 2	ID ビット	ID：0x00000002
		RTR ビット	データフレーム
		IDE ビット	拡張 ID
		DLC ビット	64Byte
		FDSTS ビット	CANFD フレーム
		BRS ビット	ビットレート可変
		送信データ	0x89,0xab,0xcd,0xef,0x01,0x23,0x45,0x67, 0x78,0x9a,0xbc,0xde,0xf0,0x12,0x34,0x56, 0x67,0x89,0xab,0xcd,0xef,0x01,0x23,0x45, 0x56,0x78,0x9a,0xbc,0xde,0xf0,0x12,0x34, 0x45,0x67,0x89,0xab,0xcd,0xef,0x01,0x23, 0x34,0x56,0x78,0x9a,0xbc,0xde,0xf0,0x10, 0x23,0x45,0x67,0x89,0xab,0xcd,0xef,0x01, 0x12,0x34,0x56,0x78,0x9a,0xbc,0xde,0xf0
使用する割り込み			なし

表 3-5 受信側 仕様一覧

項目			仕様
使用チャネル			チャネル 0
使用端子			P1_0: CAN_CLK P1_1: CAN0RX P1_3: CAN0TX
受信用バッファ			受信 FIFO バッファ 0
通信速度			通常ビットレート：1Mbps、データビットレート：2Mbps
受信ルール数			2
受信ルール			
	ルール 0	IDE ビット	拡張 ID
		RTR ビット	データフレーム
		LB ビット	受信ルール対象：他のノードが送信したメッセージを受信時
		ID ビット	受信するメッセージの ID：0x00000001
		IDE マスクビット	IDE ビットを比較する
		RTR マスクビット	RTR ビットを比較する
		ID マスクビット	ID の全ビット比較対象
		DLC ビット	64Byte
		PTR ビット	受信ルールのラベル：0x0123
		RMV ビット	受信バッファを使用しない
		RMDP ビット	受信バッファ選択なし
		FDP_TRFIFO ビット	送受信 FIFO バッファ選択なし
		FDP_RXFIFO ビット	受信 FIFO バッファ 0 選択
	ルール 1	IDE ビット	拡張 ID
		RTR ビット	データフレーム
		LB ビット	受信ルール対象：他のノードが送信したメッセージを受信時
		ID ビット	受信するメッセージの ID：0x00000002
		IDE マスクビット	IDE ビットを比較する
		RTR マスクビット	RTR ビットを比較する
		ID マスクビット	ID の全ビットを比較する
		DLC ビット	64Byte
		PTR ビット	受信ルールのラベル：0x0124
		RMV ビット	受信バッファを使用しない
		RMDP ビット	受信バッファ選択なし
		FDP_TRFIFO ビット	送受信 FIFO バッファ選択なし
		FDP_RXFIFO ビット	受信 FIFO バッファ 0 選択
使用する割り込み			受信 FIFO 割り込み

3.2.2　CAN フレームヘッダ情報の設定

送信を行う際には、送信データごとに ID、データ長などのヘッダ情報を設定します。ヘッダ情報は、R_CAN_TxSet 関数の引数で設定します。ヘッダ情報の設定例として、本動作例におけるヘッダ情報の設定値を以下に記載します。

表 3-6 ヘッダ情報の設定値(データ 1)

引数	対応するレジスタ名	設定値	説明
p_frame-> ID	RSCFDnCFDCFIDk	0x00000001	ID を設定
p_frame-> THDSE		0x00	送信履歴を格納しない
p_frame-> RTR		0x00	データフレーム
p_frame-> IDE		0x01	拡張 ID
p_frame-> LBL	RSCFDnCFDCFPTRk	0x0000	ラベルデータを設定 送信履歴を格納しないため本ビットの設定値は使用されない
p_frame-> DLC		0x0F	64 データバイト
p_frame-> FDSTS	RSCFDnCFDCFFDCSTSk	0x01	CAN FD フレーム
p_frame-> BRS		0x01	データ領域のビットレートは変わる
p_frame-> ESI		0x00	チャネルのエラーステートを ESI ビットとして送信するモードのため本ビットの設定値は使用されない

表 3-7 ヘッダ情報の設定値(データ 2)

引数	対応するレジスタ名	設定値	説明
p_frame-> ID	RSCFDnCFDCFIDk	0x00000002	ID を設定
p_frame-> THDSE		0x00	送信履歴を格納しない
p_frame-> RTR		0x00	データフレーム
p_frame-> IDE		0x01	拡張 ID
p_frame-> LBL	RSCFDnCFDCFPTRk	0x0000	ラベルデータを設定 送信履歴を格納しないため本ビットの設定値は使用されない
p_frame-> DLC		0x0F	64 データバイト
p_frame-> FDSTS	RSCFDnCFDCFFDCSTSk	0x01	CAN FD フレーム
p_frame-> BRS		0x01	データ領域のビットレートは変わる
p_frame-> ESI		0x00	チャネルのエラーステートを ESI ビットとして送信するモードのため本ビットの設定値は使用されない

3.2.3　送受信 FIFO バッファの設定

本動作例では、送信用のバッファとして送受信 FIFO バッファを使用します。送受信 FIFO バッファの設定は、『r_can_rz_config.h』で行います。送受信 FIFO バッファの設定例として、本動作例での設定値を以下に記載します。

表 3-8 送受信 FIFO バッファ 0 設定マクロの設定値

定義	対応するレジスタ名	設定値	説明
CFCC0_CFITT	RSCFDnCFDCFCCk	0x00	メッセージ送信間隔を設定 インターバル送信機能未使用のため 0 を設定
CFCC0_CFTML		0x00	送信バッファ 0 にリンクさせる
CFCC0_CFITR		0x00	送受信 FIFO インターバルタイマ分解能 インターバル送信機能未使用のため初期値を設定
CFCC0_CFITSS		0x00	送受信 FIFO インターバルタイマクロックソース インターバル送信機能未使用のため初期値を設定
CFCC0_CFM		0x01	送受信 FIFO モード：送信モード
CFCC0_CFIGCV		0x00	受信割り込み禁止のため初期値を設定
CFCC0_CFIM		0x00	送信割り込み禁止のため初期値を設定
CFCC0_CFDC		0x01	送受信 FIFO バッファ段数：4 メッセージ
CFCC0_CFPLS		0x07	送受信 FIFO バッファペイロード格納サイズ：64 バイト
CFCC0_CFTXIE		0x00	送受信 FIFO 送信割り込み禁止
CFCC0_CFRXIE		0x00	送受信 FIFO 受信割り込み禁止

3.2.4　受信ルールの設定

受信ルールの設定は、『r_can_rz_config.h』で行います。受信ルールの設定例として、本動作例における受信ルールの設定値を以下に記載します。

表 3-9 受信ルール数設定マクロの設定値

定義	対応するレジスタ名	設定値	説明
CAN0_RX_RULE_NUM	RSCFDnCFDGAFLCFG0	0x02	チャネル 0 受信ルール数：2
CAN1_RX_RULE_NUM		0x00	チャネル 1 受信ルール数：0

表 3-10 受信ルール関連マクロの設定値(受信ルール 0)

定義	対応するレジスタ名	設定値	説明
GAFLID_GAFLIDE_000	RSCFDnCFDGAFLIDj	0x01	拡張 ID
GAFLID_GAFLRTR_000		0x00	データフレーム
GAFLID_GAFLLB_000		0x00	受信ルール対象：他のノードが送信したメッセージを受信時
GAFLID_GAFLID_000		0x00000001	受信するメッセージの ID：0x00000001
GAFLM_GAFLIDEM_000	RSCFDnCFDGAFLMj	0x01	IDE ビットを比較する
GAFLM_GAFLRTRM_000		0x01	RTR ビットを比較する
GAFLM_GAFLIDM_000		0x1FFFFFFF	ID の全ビットを比較する
GAFLP0_GAFLDLC_000	RSCFDnCFDGAFLP0_j	0x0F	64 データバイト
GAFLP0_GAFLPTR_000		0x0123	ラベルデータを設定
GAFLP0_GAFLRMV_000		0x00	受信バッファを使用しない
GAFLP0_GAFLRMDP_000		0x00	受信バッファ選択なし
GAFLP1_GAFLFDP_TRFIFO_000	RSCFDnCFDGAFLP1_j	0x00	送受信 FIFO バッファ選択なし
GAFLP1_GAFLFDP_RXFIFO_000		0x01	受信 FIFO バッファ 0 選択

表 3-11 受信ルール関連マクロの設定値(受信ルール 1)

定義	対応するレジスタ名	設定値	説明
GAFLID_GAFLIDE_001	RSCFDnCFDGAFLIDj	0x01	拡張 ID
GAFLID_GAFLRTR_001		0x00	データフレーム
GAFLID_GAFLLB_001		0x00	受信ルール対象：他のノードが送信したメッセージを受信時
GAFLID_GAFLID_001		0x00000002	受信するメッセージの ID：0x00000002
GAFLM_GAFLIDEM_001	RSCFDnCFDGAFLMj	0x01	IDE ビットを比較する
GAFLM_GAFLRTRM_001		0x01	RTR ビットを比較する
GAFLM_GAFLIDM_001		0x1FFFFFFF	ID の全ビットを比較する
GAFLP0_GAFLDLC_001	RSCFDnCFDGAFLP0_j	0x0F	64 データバイト
GAFLP0_GAFLPTR_001		0x0124	ラベルデータを設定
GAFLP0_GAFLRMV_001		0x00	受信バッファを使用しない
GAFLP0_GAFLRMDP_001		0x00	受信バッファ選択なし
GAFLP1_GAFLFDP_TRFIFO_001	RSCFDnCFDGAFLP1_j	0x00	送受信 FIFO バッファ選択なし
GAFLP1_GAFLFDP_RXFIFO_001		0x01	受信 FIFO バッファ 0 選択

3.2.5　受信 FIFO バッファの設定

本動作例は、受信用のバッファとして受信 FIFO バッファを使用します。受信 FIFO バッファの設定は、『r_can_rz_config.h』で行います。受信 FIFO バッファの設定例として、本動作例での設定値を以下に記載します。

表 3-12 受信 FIFO バッファ 0 設定マクロの設定値

定義	対応するレジスタ名	設定値	説明
RFCC0_RFIGCV	RSCFDnCFDRFCCx	0x01	FIFO バッファに 2/8 までメッセージ格納時割り込み要求発生
RFCC0_RFIM		0x00	RFCC0_RFIGCV で設定した条件に達したときに割り込み要求発生
RFCC0_RFDC		0x02	受信 FIFO バッファ段数：8 メッセージ
RFCC0_RFPLS		0x07	受信 FIFO バッファペイロード格納サイズ：64 バイト
RFCC0_RFIE		0x01	受信 FIFO 割り込み許可

3.3　動作例 2 ゲートウェイ動作

3.3.1　動作概要

送受信 FIFO バッファをゲートウェイモードに設定すると、CPU を介さずに受信したメッセージを任意のチャネルから送信することができます。また、CAN FD モードでゲートウェイ機能を使用するときは、送信するフレームをクラシカル CAN フレームまたは CAN FD フレームに置換することができます。本動作例では、チャネル 0 で受信したクラシカル CAN フレームを、CAN FD フレームに置換し、チャネル 1 から送信します。

上記の処理は、全てハードウェアで行うため、ソフトウェアで実装する必要はありません。ソフトウェアの処理は、共通設定およびゲートウェイモードの設定のみです。

動作確認用の対向ボード用プログラムは、チャネル 0 からクラシカル CAN フレームのデータを送信し、変換された CANFD フレームのデータをチャネル 1 で受信します。

図 3-7 に本動作例のシステム構成を示します。

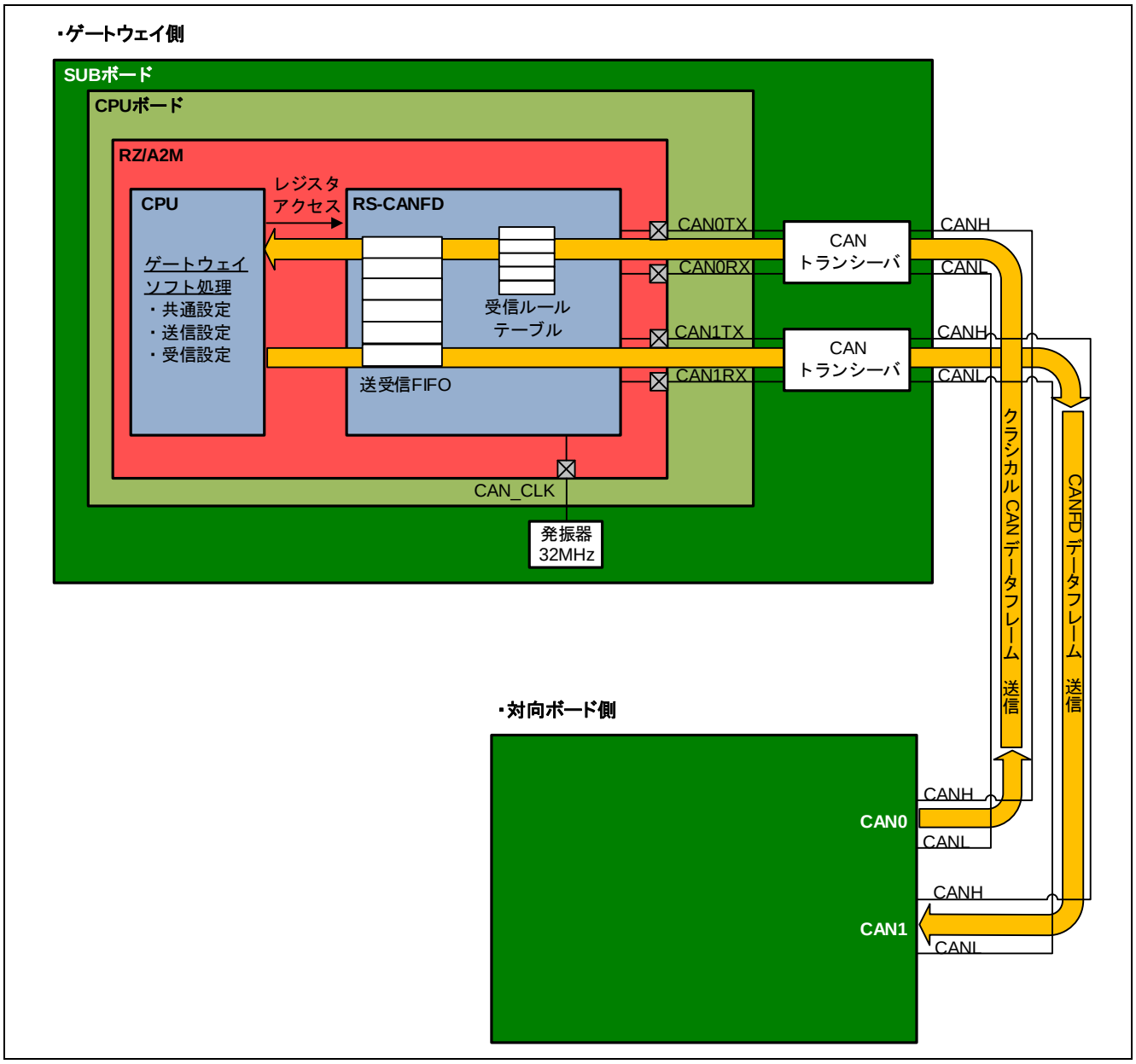


図 3-7 システム構成

ゲートウェイ動作の、動作確認の手順を下記に示します。ゲートウェイ動作の動作確認は、対向ボード側のプログラムで行います。

- 操作方法
  1. お互いのボードの、CH0（CAN1 のコネクタ）の CANH(1 番ピン)同士、CANL(3 番ピン)同士を接続、CH1（CAN2 のコネクタ）の CANH 同士、CANL 同士を接続します。
  2. ゲートウェイ側のプログラムを実行し、受信待ち状態にします。
  3. 対向ボード側のプログラムの、メイン関数の最後の /*NG*/ および /*OK*/ のコメントが添えられた while 文にブレークポイントを設定します。また、対向ボード側のプログラムを実行し、送受信を行います。

- 確認内容
  1. 送信データと受信データが一致して、/*OK*/のコメントが添えられているループに入り、プログラムが終了していることを確認します。

- 確認結果

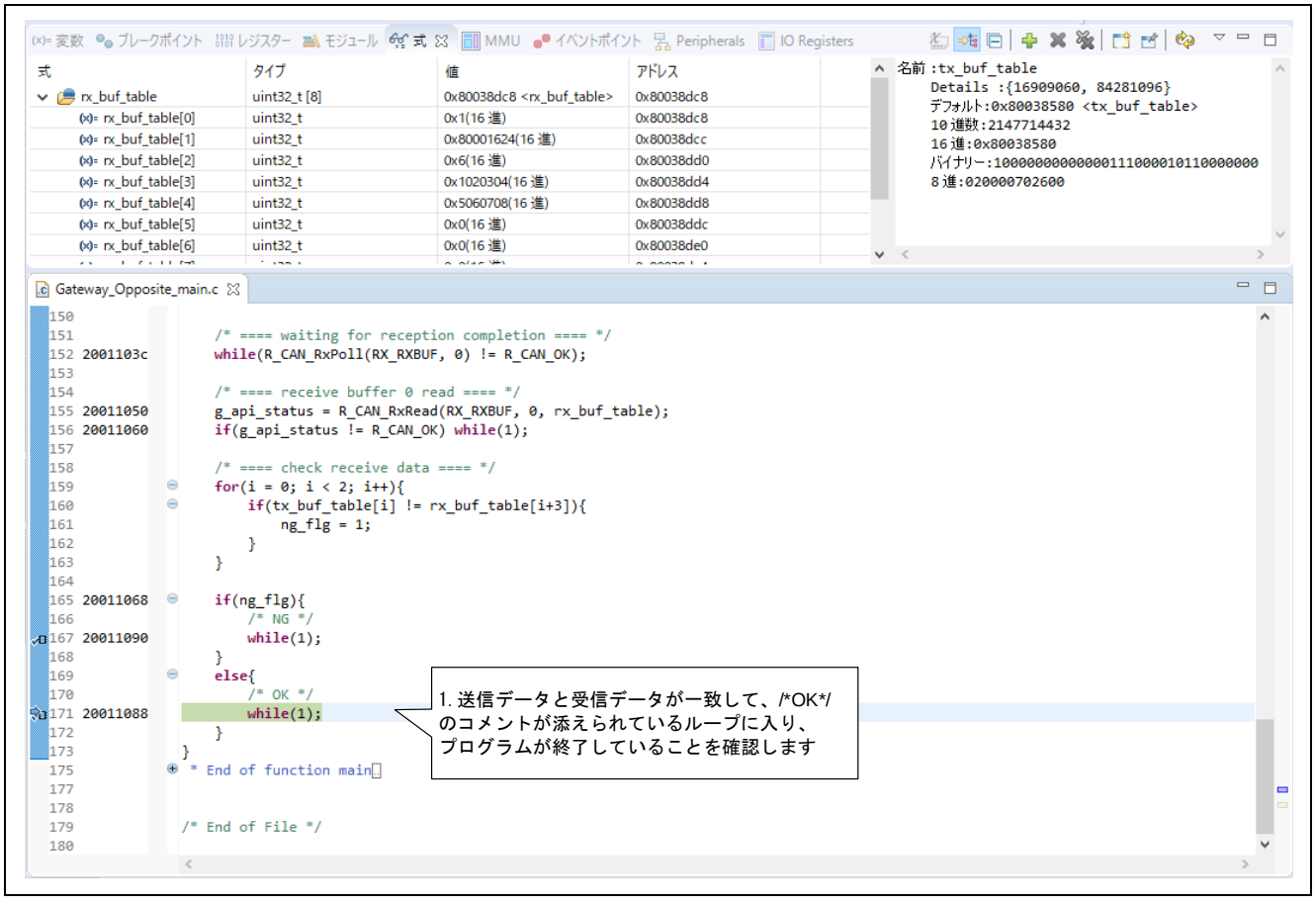


図 3-8 確認結果

表 3-13 ゲートウェイ仕様一覧

項目		仕様
使用チャネル		チャネル 0、チャネル 1
使用端子		P1_0: CAN_CLK P1_1: CAN0RX P1_3: CAN0TX P2_0: CAN1RX P2_2: CAN1TX
ゲートウェイ用バッファ		送受信 FIFO バッファ 3(ゲートウェイモード)
通信速度		◆クラシカル CAN：1Mbps ◆CAN FD： 通常ビットレート：1Mbps、データビットレート：2Mbps
受信ルール		
	IDE ビット	標準 ID
	RTR ビット	データフレーム
	LB ビット	受信ルール対象：他のノードが送信したメッセージを受信時
	ID ビット	受信するメッセージの ID：0x00000000
	IDE マスクビット	IDE ビットを比較しない
	RTR マスクビット	RTR ビットを比較しない
	ID マスクビット	ID を比較しない
	DLC ビット	DLC ビットを比較しない
	LBL ビット	受信ルールのラベル：0x0000
	RMV ビット	受信バッファを使用しない
	RMDP ビット	受信バッファ選択なし
	FDP_TRFIFO ビット	送受信 FIFO バッファ 3 選択
	FDP_RXFIFO ビット	受信 FIFO バッファ選択なし
使用する割り込み		なし

表 3-14 対向ボード側仕様一覧

項目		仕様
使用チャネル		チャネル 0、チャネル 1
使用端子		P1_0: CAN_CLK P1_1: CAN0RX P1_3: CAN0TX P2_0: CAN1RX P2_2: CAN1TX
送信用バッファ		送信バッファ 0
受信用バッファ		受信バッファ 0
通信速度		◆クラシカル CAN：1Mbps ◆CAN FD： 通常ビットレート：1Mbps、データビットレート：2Mbps
送信履歴		OFF
送信データ設定		
	ID ビット	ID：0x00000001
	RTR ビット	データフレーム
	IDE ビット	標準 ID
	DLC ビット	8Byte
	FDSTS ビット	クラシカル CAN フレーム
	BRS ビット	ビットレート不変
	送信データ	0x01020304,0x05060708
受信ルール		
	IDE ビット	標準 ID
	RTR ビット	データフレーム
	LB ビット	受信ルール対象：他のノードが送信したメッセージを受信時
	ID ビット	受信するメッセージの ID：0x00000000
	IDE マスクビット	IDE ビットを比較しない
	RTR マスクビット	RTR ビットを比較しない
	ID マスクビット	ID を比較しない
	DLC ビット	DLC ビットを比較しない
	LBL ビット	受信ルールのラベル：0x0000
	RMV ビット	受信バッファを使用する
	RMDP ビット	受信バッファ 0 使用
	FDP_TRFIFO ビット	送受信 FIFO バッファ選択なし
	FDP_RXFIFO ビット	受信 FIFO バッファ選択なし
使用する割り込み		なし

3.3.2　ゲートウェイの設定

本動作例では、送受信 FIFO バッファをゲートウェイモードで使用しています。ゲートウェイの設定は、『r_can_rz_config.h』で行います。ゲートウェイ動作の設定例として、本動作例での設定値を以下に記載します。

表 3-15 送受信 FIFO バッファ 3 コンフィグマクロの設定値

定義	対応するレジスタ名	設定値	説明
CFCC3_CFITT	RSCFDnCFDCFCCk	0x00	メッセージ送信間隔を設定 インターバル送信機能未使用のため 0 を設定
CFCC3_CFTML		0x00	送信バッファ 0 にリンクさせる
CFCC3_CFITR		0x00	送受信 FIFO インターバルタイマ分解能 インターバル送信機能未使用のため初期値を設定
CFCC3_CFITSS		0x00	送受信 FIFO インターバルタイマクロックソース インターバル送信機能未使用のため初期値を設定
CFCC3_CFM		0x02	送受信 FIFO モード：ゲートウェイモード
CFCC3_CFIGCV		0x00	受信割り込み禁止のため初期値を設定
CFCC3_CFIM		0x00	送信割り込み禁止のため初期値を設定
CFCC3_CFDC		0x01	送受信 FIFO バッファ段数：4 メッセージ
CFCC3_CFPLS		0x07	送受信 FIFO バッファペイロード格納サイズ：64 バイト
CFCC3_CFTXIE		0x00	送受信 FIFO 送信割り込み禁止
CFCC3_CFRXIE		0x00	送受信 FIFO 受信割り込み禁止

表 3-16 チャネル 1 CAN FD 設定マクロの設定値

定義	対応するレジスタ名	設定値	説明
C1FDCFG_REFE	RSCFDnCFDCmFDCFG	0x00	受信データエッジフィルタ禁止
C1FDCFG_FDOE		0x00	FD オンリーモード禁止
C1FDCFG_TMME		0x00	送信バッファマージモード禁止
C1FDCFG_GWBRS		0x01	受信フレームの BRS ビットを“1”にして送信
C1FDCFG_GWFDF		0x01	受信フレームを CAN FD フレームとして送信
C1FDCFG_GWEN		0x01	CAN-CAN FD ゲートウェイ許可
C1FDCFG_ESIC		0x00	常にチャネルのエラーステートをフレームの ESI ビットとして送信
C1FDCFG_EOCCFG		0x00	エラー発生回数カウント方式：全ての送信メッセージと受信メッセージ

3.3.3　受信ルールの設定

受信ルールは、『r_can_rz_config.h』で設定を行います。受信ルールの設定例として、本動作例における受信ルールの設定値を以下に記載します。本動作例では、受信ルールのマスクを使用して、全てのメッセージを受信する設定にしています。

表 3-17 受信ルール数設定マクロの設定値

定義	対応するレジスタ名	設定値	説明
CAN0_RX_RULE_NUM	RSCFDnCFDGAFLCFG0	0x01	チャネル 0 受信ルール数：1
CAN1_RX_RULE_NUM		0x00	チャネル 1 受信ルール数：0

表 3-18 受信ルール関連マクロの設定値

定義	対応するレジスタ名	設定値	説明
GAFLID_GAFLIDE_000	RSCFDnCFDGAFLIDj	0x00	標準 ID
GAFLID_GAFLRTR_000		0x00	データフレーム
GAFLID_GAFLLB_000		0x00	受信ルール対象：他のノードが送信したメッセージを受信時
GAFLID_GAFLID_000		0x00000000	受信するメッセージの ID：0x00000000
GAFLM_GAFLIDEM_000	RSCFDnCFDGAFLMj	0x00	IDE ビットを比較しない
GAFLM_GAFLRTRM_000		0x00	RTR ビットを比較しない
GAFLM_GAFLIDM_000		0x00000000	ID を比較しない
GAFLP0_GAFLDLC_000	RSCFDnCFDGAFLP0_j	0x00	DLC ビットを比較しない
GAFLP0_GAFLPTR_000		0x0000	受信ルールの ID：0x0000
GAFLP0_GAFLRMV_000		0x00	受信バッファを使用しない
GAFLP0_GAFLRMDP_000		0x00	受信バッファ選択なし
GAFLP1_GAFLFDP_TRFIFO_000	RSCFDnCFDGAFLP1_j	0x08	送受信 FIFO バッファ 3 選択
GAFLP1_GAFLFDP_RXFIFO_000		0x00	受信 FIFO バッファ選択なし

3.4　動作例 3 外部ループバックテスト

3.4.1　動作概要

本動作例では、外部ループバックモードを使用し、チャネル 0 の送信端子から送信したデータを、チャネル 0 の受信端子で受信します。また、送信データと受信データを比較して、送受信に異常がないことを確認します。送信は送信バッファ、受信は受信バッファを使用します。受信完了の監視は、ポーリングで行います。

図 3-9 に本動作例のシステム構成を示します。

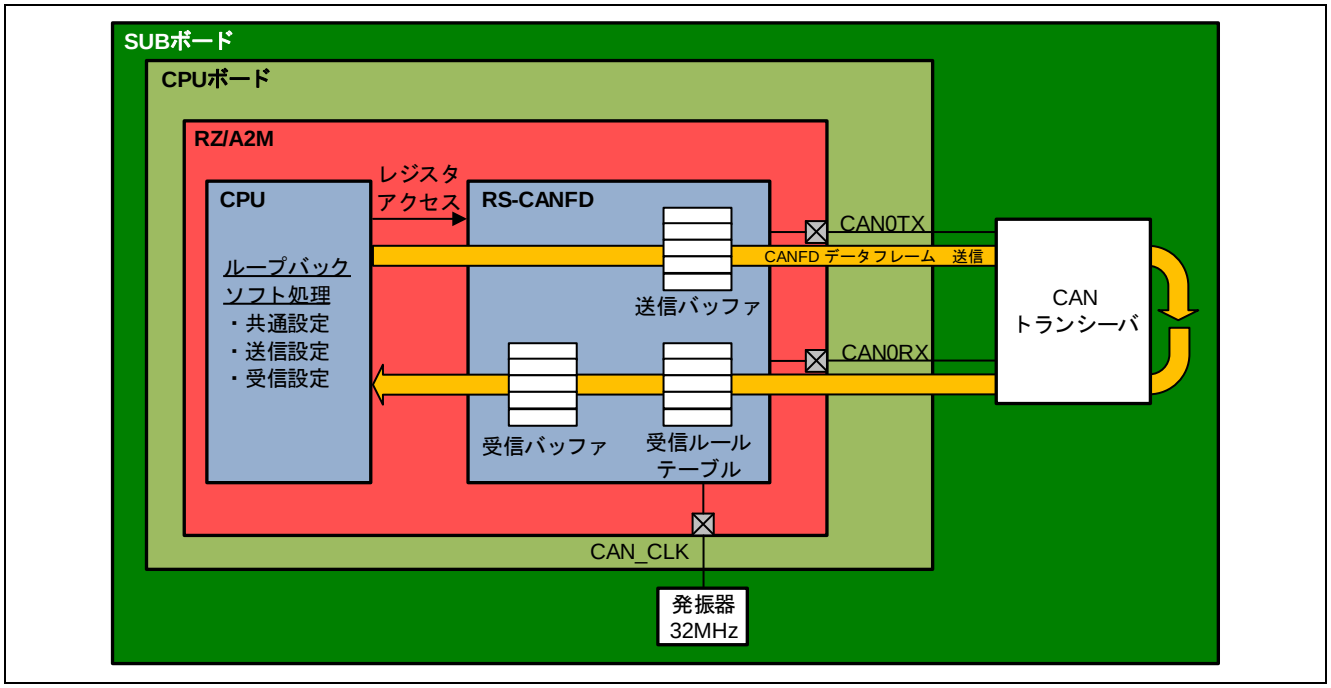


図 3-9 システム構成

送受信動作の、動作確認の手順を下記に示します。

- 操作方法
  1. 端子の接続等は不要です。
  2. 外部ループバックテストプログラムの、メイン関数の最後の /*NG*/ および /*OK*/ のコメントが添えられた while 文にブレークポイントを設定し、プログラムを実行します。

- 確認内容
  1. 送信データと受信データが一致して、/*OK*/のコメントが添えられているループに入り、プログラムが終了していることを確認します。

- 確認結果

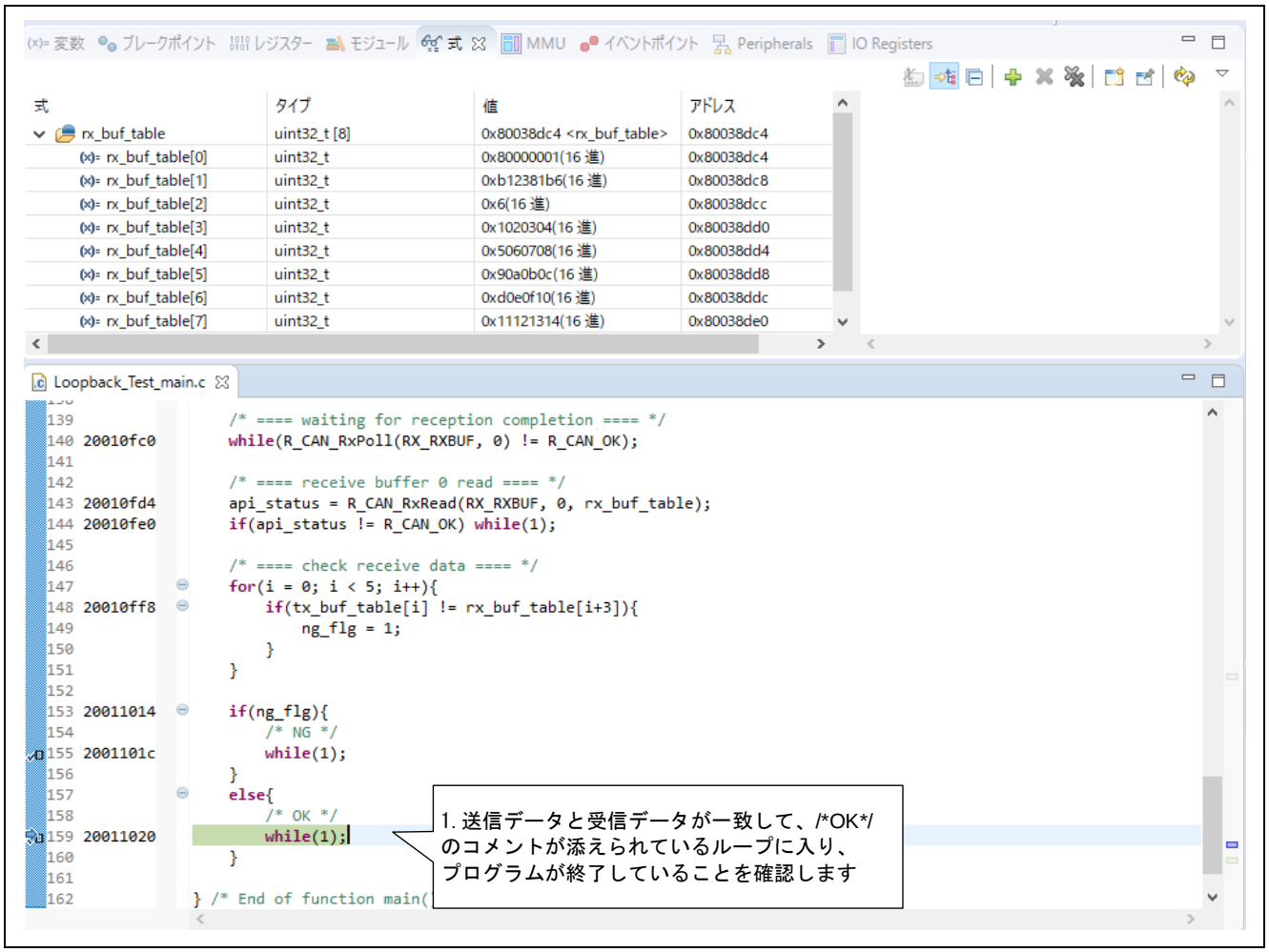


図 3-10 確認結果

表 3-19 ループバック仕様一覧

項目		仕様
使用チャネル		チャネル 0（セルフテストモード 0（外部ループバックモード））
使用端子		P1_0: CAN_CLK P1_1: CAN0RX P1_3: CAN0TX
送信用バッファ		送信バッファ 0
受信用バッファ		受信バッファ 0
通信速度		通常ビットレート：1Mbps、データビットレート：2Mbps
送信履歴		OFF
送信データ設定		
	ID ビット	ID：0x00000001
	RTR ビット	データフレーム
	IDE ビット	拡張 ID
	DLC ビット	20Byte
	FDSTS ビット	CANFD フレーム
	BRS ビット	ビットレート可変
	送信データ	0x01020304,0x05060708, 0x090A0B0C,0x0D0E0F10, 0x11121314
受信ルール		
	IDE ビット	拡張 ID
	RTR ビット	データフレーム
	LB ビット	受信ルール対象：他のノードが送信したメッセージを受信時
	ID ビット	受信するメッセージの ID：0x00000001
	IDE マスクビット	IDE ビットを比較する
	RTR マスクビット	RTR ビットを比較する
	ID マスクビット	ID の全ビットを比較する
	DLC ビット	DLC ビットを比較しない
	LBL ビット	受信ルールのラベル：0x0123
	RMV ビット	受信バッファを使用する
	RMDP ビット	受信バッファ 0 使用
	FDP_TRFIFO ビット	送受信 FIFO バッファ選択なし
	FDP_RXFIFO ビット	受信 FIFO バッファ選択なし
使用する割り込み		なし

3.4.2　CAN フレームヘッダ情報の設定

送信を行う際には、送信データごとに ID、データ長などのヘッダ情報を設定します。ヘッダ情報は、R_CAN_TxSet 関数の引数で設定します。ヘッダ情報の設定例として、本動作例におけるヘッダ情報の設定値を以下に記載します。

表 3-20 ヘッダ情報の設定値

引数	対応するレジスタ名	設定値	説明
p_frame->ID	RSCFDnCFDTMIDp	0x00000001	ID を設定
p_frame->THDSE		0x00	送信履歴を格納しない
p_frame->RTR		0x00	データフレーム
p_frame->IDE		0x01	拡張 ID
p_frame->LBL	RSCFDnCFDTMPTRp	0x0000	ラベルデータを設定 送信履歴を格納しないため本ビットの設定値は使用されない
p_frame->DLC		0x0B	20 データバイト
p_frame->FDSTS	RSCFDnCFDTMFDCTRp	0x01	CAN FD フレーム
p_frame->BRS		0x01	データ領域のビットレートは変わる
p_frame->ESI		0x00	チャネルのエラーステートを ESI ビットとして送信するモードのため本ビットの設定値は使用されない

3.4.3　送信バッファの設定

本サンプルプログラムでは、送信用のバッファとして送信バッファを使用します。送信バッファの設定は、『r_can_rz_config.h』で行います。送信バッファの設定例として、本サンプルプログラムでの設定値を以下に記載します。

表 3-21 送信バッファ設定マクロの設定値

定義	対応するレジスタ名	設定値	説明
TMIEC0_TMIE0	RSCFDnCFDTMIECy	0x00	送信バッファ割り込み禁止

3.4.4　受信ルールの設定

受信ルールの設定は、『r_can_rz_config.h』で行います。受信ルールの設定例として、本動作例における受信ルールの設定値を以下に記載します。本動作例では、DLC ビットを比較しないため、受信メッセージのすべてのデータバイトがバッファに格納されます。

表 3-22 受信ルール数設定マクロの設定値

定義	対応するレジスタ名	設定値	説明
CAN0_RX_RULE_NUM	RSCFDnCFDGAFLCFG0	0x01	チャネル 0 受信ルール数：1
CAN1_RX_RULE_NUM		0x00	チャネル 1 受信ルール数：0

表 3-23 受信ルール関連マクロの設定値

定義	対応するレジスタ名	設定値	説明
GAFLID_GAFLIDE_000	RSCFDnCFDGAFLIDj	0x01	拡張 ID
GAFLID_GAFLRTR_000		0x00	データフレーム
GAFLID_GAFLLB_000		0x00	受信ルール対象：他のノードが送信したメッセージを受信時
GAFLID_GAFLID_000		0x00000001	受信するメッセージの ID：0x00000001
GAFLM_GAFLIDEM_000	RSCFDnCFDGAFLMj	0x01	IDE ビットを比較する
GAFLM_GAFLRTRM_000		0x01	RTR ビットを比較する
GAFLM_GAFLIDM_000		0x1FFFFFFF	ID の全ビットを比較する
GAFLP0_GAFLDLC_000	RSCFDnCFDGAFLP0_j	0x00	DLC ビットを比較しない
GAFLP0_GAFLPTR_000		0x0123	ラベルデータを設定
GAFLP0_GAFLRMV_000		0x01	受信バッファを使用する
GAFLP0_GAFLRMDP_000		0x00	受信バッファ 0 選択
GAFLP1_GAFLFDP_TRFIFO_000	RSCFDnCFDGAFLP1_j	0x00	送受信 FIFO バッファ選択なし
GAFLP1_GAFLFDP_RXFIFO_000		0x00	受信 FIFO バッファ選択なし

3.4.5　受信バッファの設定

本サンプルプログラムでは、受信用のバッファとして受信バッファを使用します。受信バッファの設定は、『r_can_rz_config.h』で行います。受信バッファの設定例として、本サンプルプログラムでの設定値を以下に記載します。

表 3-24 受信バッファ 0 設定マクロの設定値

定義	対応するレジスタ名	設定値	説明
RMNB_RMPLS	RSCFDnCFDRMNB	0x03	受信バッファペイロード格納サイズ：20 バイト
RMNB_NRXMB		0x01	受信バッファ数：1

4. 制限事項

本アプリケーションノートにおける、制限事項はありません。

5. 注意事項

本アプリケーションノートの注意事項を以下に示します。

表 5-1 注意事項

No	種別	内容
1	環境	本アプリケーションノートのプロジェクトをそのままビルドして、ビルドエラーが発生する場合、環境が正しく設定されていない可能性があります。 以下の対処をお願い致します。 - RZ/A2M Software Package クイックスタートガイド(R01QS0027)の「サンプルプロジェクトの立ち上げ」を参照してください。 - それでも改善しない場合、e2studio バージョン 7.4 以降を再インストールしてください。
2	環境	プロジェクトは、日本語を含まないフォルダに展開してください。 （ビルドが通らない場合があるため。）
3	環境	プロジェクトは、出来るだけフルパスの短いフォルダに展開してください。 （ビルドが通らない場合があるため。）
4	環境	本アプリケーションノートは、ブートローダはバイナリでのみ同梱しており、プロジェクトは含まれておりません。ブートローダのプロジェクトを入手したい場合は、以下のアプリケーションノートを Renesas サイトからダウンロードしてください。 RZ/A2M グループ シリアルフラッシュメモリからのブート例(R01AN4333)
5	環境	RZ/A2M SUB ボードには、CAN トランシーバと CAN クロック用の発振器が未実装のため、CAN 使用時は実装する必要があります。 詳細は、2.1 システム構成を参照してください。

6. 使用オープンソースソフトウェアとライセンス

本アプリケーションノートで使用しているオープンソースソフトウェアとそのライセンスについて以下に示します。

- newlib は以下に示されるライセンスの元で使用されています。
  https://www.sourceware.org/newlib/COPYING.NEWLIB

7. 参考ドキュメント

ユーザーズマニュアル：ハードウェア

RZ/A2M グループ ユーザーズマニュアル ハードウェア編
（最新版をルネサス エレクトロニクスホームページから入手してください。）

RTK7921053C00000BE（RZ/A2M CPU ボード）ユーザーズマニュアル
（最新版をルネサス エレクトロニクスホームページから入手してください。）

RTK79210XXB00000BE（RZ/A2M SUB ボード）ユーザーズマニュアル
（最新版をルネサス エレクトロニクスホームページから入手してください。）

Arm Architecture Reference Manual ARMv7-A and ARMv7-R edition Issue C
（最新版を Arm ホームページから入手してください。）

Arm Cortex™-A9 Technical Reference Manual Revision: r4p1
（最新版を Arm ホームページから入手してください。）

Arm Generic Interrupt Controller Architecture Specification - Architecture version2.0
（最新版を Arm ホームページから入手してください。）

Arm CoreLink™ Level 2 Cache Controller L2C-310 Technical Reference Manual Revision: r3p3
（最新版を Arm ホームページから入手してください。）

テクニカルアップデート／テクニカルニュース
（最新の情報をルネサス エレクトロニクスホームページから入手してください。）

ユーザーズマニュアル：統合開発

統合開発環境 e2 studio のユーザーズマニュアルは、ルネサス エレクトロニクスホームページから入手してください。
（最新版をルネサス エレクトロニクスホームページから入手してください。）

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2020.01.10	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。お客様の機器・システムの設計において、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には、お客様の責任において行ってください。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、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、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。
5. 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。
6. 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。
7. 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。
8. 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。
9. 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。
10. 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。
11. 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。
12. 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.4.0-1　2017.11)

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。