

RL78/G23

NOR FLASH の ADPCM データを I²S で音声再生

要旨

本アプリケーションノートでは、「[RL78/G23 ELCL、SPI による I²S 通信 アプリケーションノート \(R01AN6420\)](#)」に記載されている I²S マスタ機能を使用し、NOR FLASH に格納された ADPCM データを再生する方法を説明します。NOR FLASH 制御には RL78 Family Serial NOR Flash Memory 制御 モジュール Software Integration System(SIS)、ADPCM encoder/decoder には M3S-S2-Tiny ミドルウェアを使用します。

動作確認デバイス

RL78/G23

本アプリケーションノートを他のマイコンへ適用する場合、そのマイコンの仕様に合わせて変更し、十分評価してください。

目次

1. 仕様	5
2. 動作確認条件	6
3. 構成	7
3.1 使用端子一覧	7
3.2 ハードウェア構成例	8
3.3 ソフトウェア構成	10
3.4 APN 構成（ファイル構成）	11
4. NOR FLASH への音声データ（ADPCM）書き込み方法	12
4.1 ソフトウェア説明	12
4.2 ハードウェア接続	12
4.3 ターミナルソフト	12
4.4 デバッグの使用	13
4.5 コマンド一覧	13
4.6 操作方法	13
4.6.1 NOR FLASH への書き込み方法	13
4.6.2 NOR FLASH 内容表示方法	16
4.6.3 NOR FLASH 消去方法	16
4.7 エラー表示	17
4.8 ソフトウェア説明詳細	18
4.8.1 動作概要	18
4.8.2 フォルダ構成	19
4.8.3 オプション・バイトの設定一覧	20
4.8.4 ROM/RAM サイズ	20
4.8.5 定数一覧	21
4.8.6 変数一覧	21
4.8.7 関数一覧	22
4.8.8 関数仕様	23
4.8.9 フローチャート	29
4.8.9.1 main ()	29
4.8.9.2 r_adpcm_writer_main()	29
4.8.9.3 init ()	30
4.8.9.4 state_ctrl()	30
4.8.9.5 check_command ()	31
4.8.9.6 get_flow_read_next ()	31
4.8.9.7 get_flow_enc_adpcm ()	32
4.8.9.8 wait_send_end ()	32
4.8.9.9 status_init ()	33
4.8.9.10 wait_and_print ()	33
4.8.9.11 wait_and_print_len()	34
4.8.9.12 adpcm_encode_8byte()	34
4.8.9.13 write_nor_flash()	35

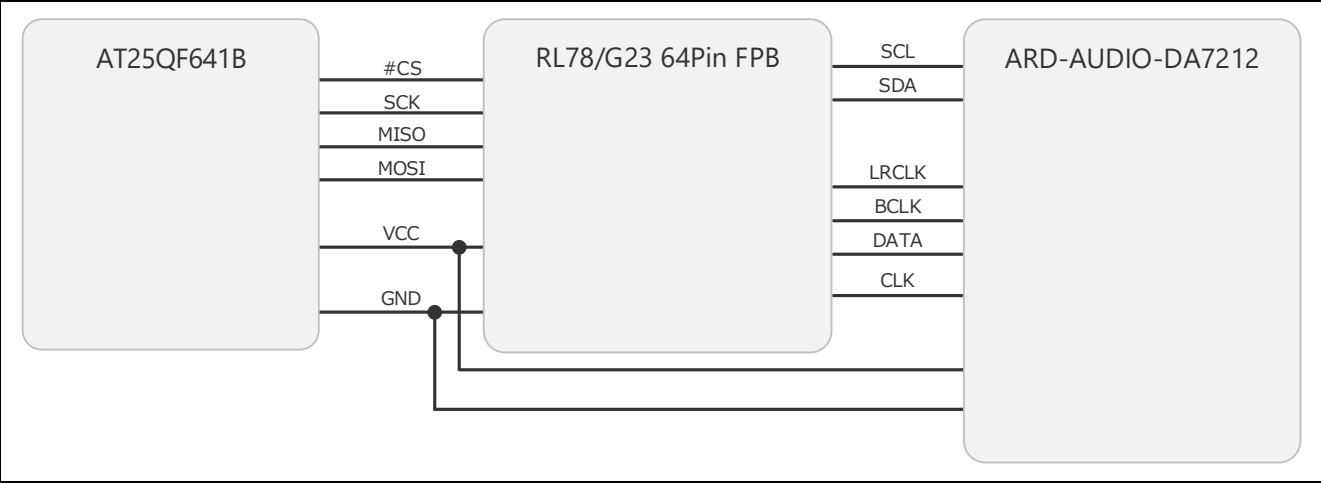
4.8.9.14	parse_command()	35
4.8.9.15	check_last()	36
4.8.9.16	write_header_nor_flash()	37
4.8.9.17	check_success_and_busy()	37
4.8.9.18	init()	38
4.8.9.19	r_nor_flash_data_read()	38
4.8.9.20	r_nor_flash_data_write()	39
4.8.9.21	r_wave_decoder_parse()	40
4.8.10	r01an7416_adpcm_writer.scfg	41
4.8.10.1	クロック	42
4.8.10.2	システム	42
4.8.10.3	r_bsp	42
4.8.10.4	Config_LVD0	43
4.8.10.5	Config_CSI20	43
4.8.10.6	Config_UART0	43
4.8.10.7	r_nor_flash	43
5.	音声再生方法	44
5.1	ソフトウェア説明	44
5.2	ハードウェア接続	44
5.3	ターミナルソフト	44
5.4	デバッグの使用	44
5.5	コマンド一覧	44
5.6	操作方法	44
5.7	エラー表示	44
5.8	ソフトウェア説明詳細	45
5.8.1	動作概要	45
5.8.2	フォルダ構成	46
5.8.3	オプション・バイトの設定一覧	48
5.8.4	ROM/RAM サイズ	48
5.8.5	定数一覧	48
5.8.6	変数一覧	49
5.8.7	関数一覧	50
5.8.8	関数仕様	51
5.8.9	フローチャート	56
5.8.9.1	main()	56
5.8.9.2	r_adpcm_player_operation()	56
5.8.9.3	init()	57
5.8.9.4	play_start()	58
5.8.9.5	wait_sw()	59
5.8.9.6	check_rcv_sb()	59
5.8.9.7	check_dec_sb()	60
5.8.9.8	status_init()	61
5.8.9.9	rcv_sb_status_clear()	62
5.8.9.10	dec_sb_status_clear()	62
5.8.9.11	i2s_sb_status_clear()	63
5.8.9.12	get_i2s_data()	63

5.8.9.13 adpcm2pcm ()	64
5.8.9.14 r_codec_init ()	64
5.8.9.15 change_i2s_buf ()	65
5.8.9.16 send_i2c ()	66
5.8.9.17 r_codec_change_sampling_rate ()	66
5.8.9.18 r_codec_start ()	67
5.8.9.19 r_codec_stop ()	68
5.8.10 r01an7416_adpcm_player.scfg	69
5.8.10.1 クロック	71
5.8.10.2 システム	71
5.8.10.3 r_bsp	71
5.8.10.4 Config_LVD0	71
5.8.10.5 Config_INTC	72
5.8.10.6 Config_TAU00	72
5.8.10.7 Config_TAU01	72
5.8.10.8 Config_TAU07	72
5.8.10.9 Config_Through	72
5.8.10.10 Config_PORT	72
5.8.10.11 Config_CSI20	72
5.8.10.12 r_nor_flash	72
5.8.10.13 Config_PCLBUSZ0	72
6. サンプルコード	73
7. 参考ドキュメント	73
改訂記録	74

1. 仕様

本アプリケーションノートでサポートする I²S 通信の構成を図 1-1、仕様を表 1-1 に示します。

図 1-1 I²S 通信の構成



補足：端子の詳細は、表 3-1 を参照してください。

表 1-1 I²S 通信フォーマット

項目	内容
音声データフォーマット	PCM
機能	マスタ
	BCLK 周波数：64fs（fs：サンプリング周波数）
	サンプリング周波数：8 kHz, 11.025 kHz, 16 kHz, 22.05 kHz
	PCM データサイズ：16 ビット

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
使用マイコン	RL78/G23 (R7F100GLG)
動作周波数	- 高速オンチップ・オシレータ・クロック : 32MHz - CPU/周辺ハードウェア・クロック : 32MHz
動作電圧	3.3V - LVD0 動作 (V_{LVD0}) : リセット・モード 立ち上がり時 TYP. 1.90V 立ち下がり時 TYP. 1.86V
統合開発環境 (CS+)	ルネサスエレクトロニクス製 CS+ for CC V8.12.00
C コンパイラ (CS+)	ルネサスエレクトロニクス製 CC-RL V1.14
統合開発環境 (e ² studio)	ルネサスエレクトロニクス製 e ² studio 2024-07 (24.07.0)
C コンパイラ (e ² studio)	ルネサスエレクトロニクス製 CC-RL V1.14
統合開発環境 (IAR)	IAR システム製
C コンパイラ (IAR)	IAR Embedded Workbench for Renesas RL78 V5.10.3
スマート・コンフィグレータ	V.1.11.0
ボードサポートパッケージ (r_bsp)	V.1.62
エミュレータ	COM ポート
使用ボード	RL78/G23-64p Fast Prototyping Board (RL78/G23-64p FPB) (RTK7RLG230CLG000BJ)
	ARD-AUDIO-DA7212 (190-03-B1-001)
	NOR FLASH AT25QF641B 搭載ブレッドボード

3. 構成

3.1 使用端子一覧

表 3-1 に使用端子と機能を示します。

表 3-1 使用端子と機能

端子名	入出力	内容
P60/SCLA0	出力	シリアル・クロック
P61/SDAA0	出力	シリアル・データ
P50/EO50	出力	LRCLK
P51/EO51	出力	BCLK
P73/SO01	出力	SDIN
P31/PCLBUZ0	出力	DA7212 用クロック出力
P16/GPIO	出力	NOR FLASH 用 SPI CS
P15/SCK20	出力	NOR FLASH 用 SPI クロック
P14/SI20	入力	NOR FLASH 用 SPI MISO
P13/SO20	出力	NOR FLASH 用 SPI MOSI
P12/TXD	出力	動作状況表示、フロー制御 (Xon/Xoff)
P11/RXD	入力	コマンド、WAVE データ入力

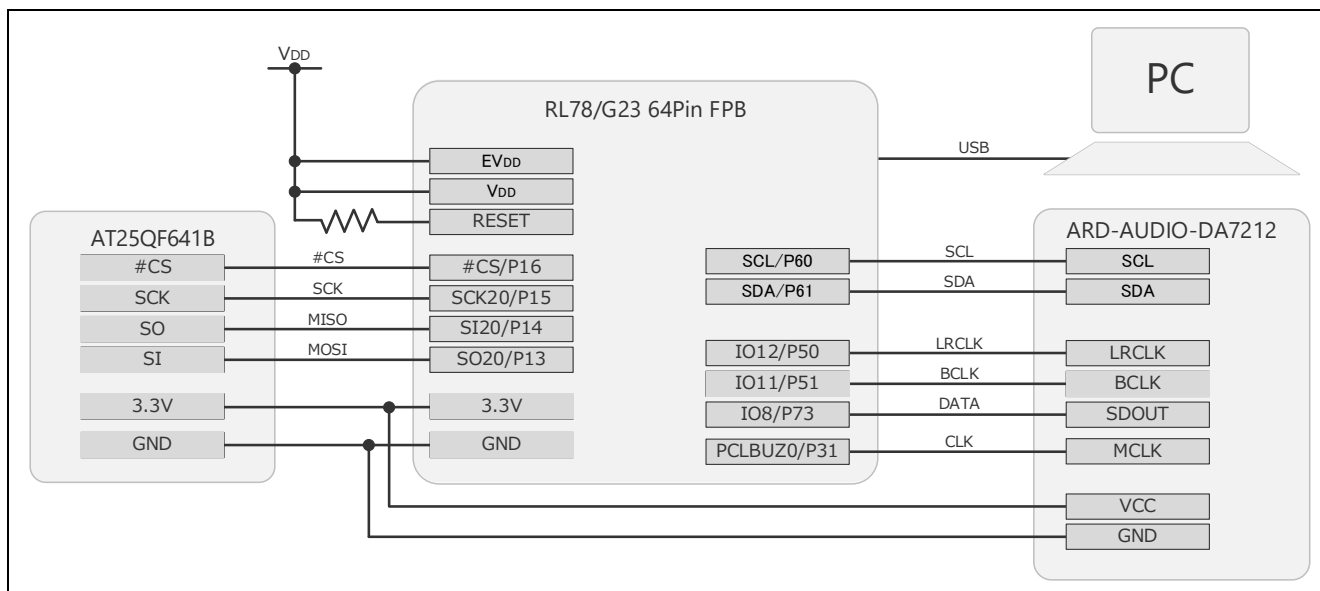
注意 本アプリケーションノートは、使用端子のみを端子処理しています。実際に回路を作成される場合は、端子処理などを適切に行い、電気的特性を満たすように設計してください。

3.2 ハードウェア構成例

図 3-1 に本アプリケーションのサンプルコードで使用するハードウェア構成例を示します。

また、図 3-2 に実際の接続例、図 3-3 と図 3-4 に接続部を拡大したものを示します。

図 3-1 ハードウェア構成例



補足 I²C 通信については 5.8.1 動作概要を参照してください。

- 注意 1. この回路イメージは接続の概要を示す為に簡略化しています。実際に回路を作成される場合は、端子処理などを適切に行い、電気的特性を満たすように設計してください（入力専用ポートは個別に抵抗を介して V_{DD} 又は V_{SS} に接続して下さい）。
- 注意 2. EV_{SS} で始まる名前の端子がある場合には V_{SS} に、EV_{DD} で始まる名前の端子がある場合には V_{DD} にそれぞれ接続してください。
- 注意 3. V_{DD} は LVD0 にて設定したリセット解除電圧（V_{LVD0}）以上にしてください。

図 3-2 実際の接続例

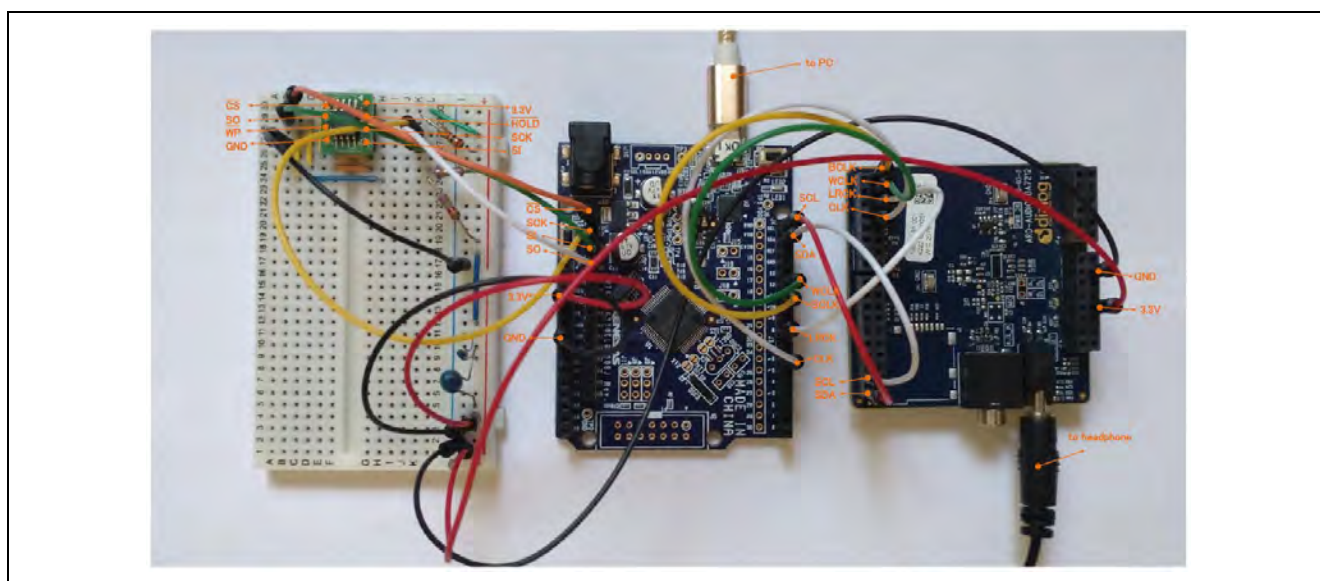


図 3-3 RL78/G23 と ARD-AUDIO-DA7212 の接続部拡大

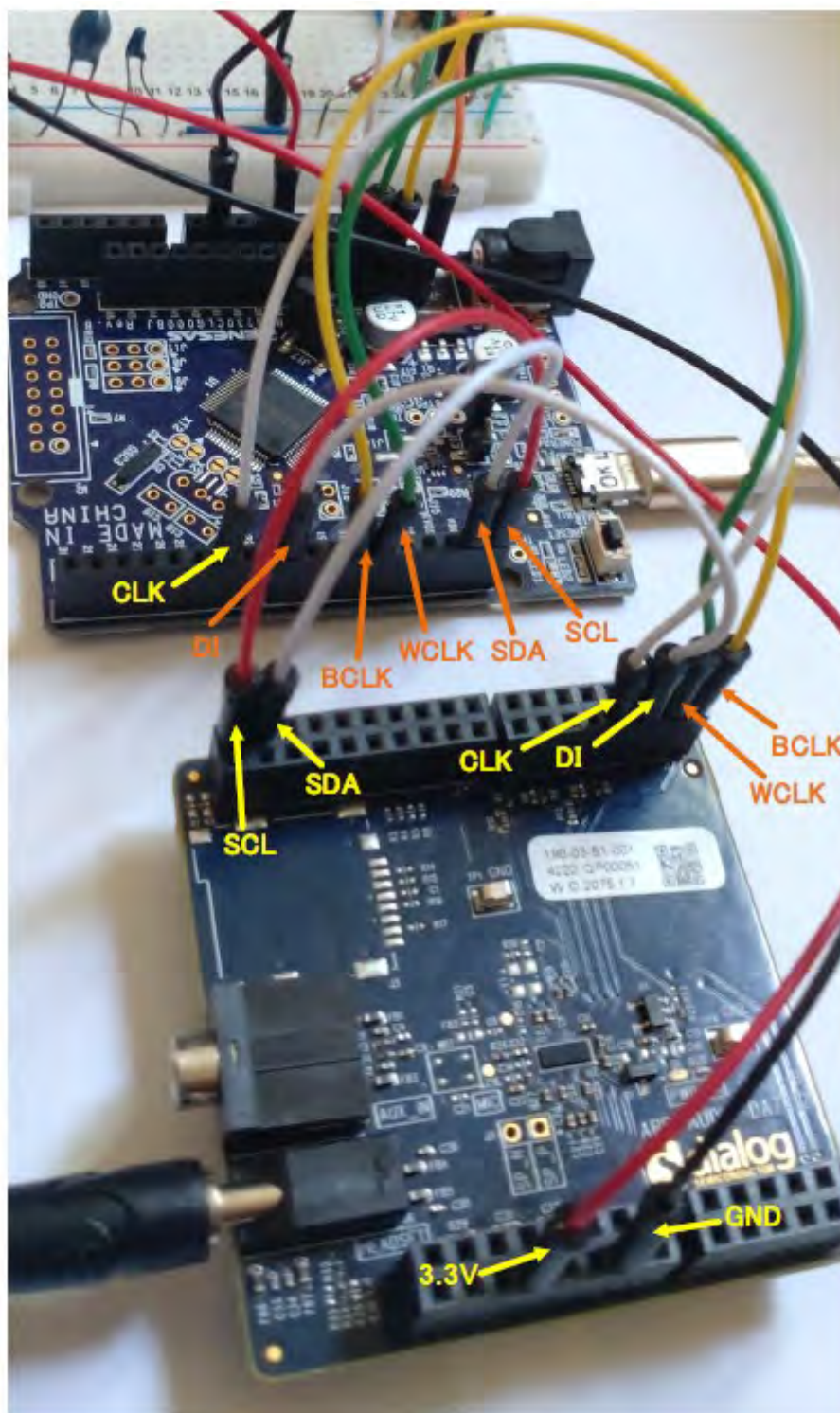
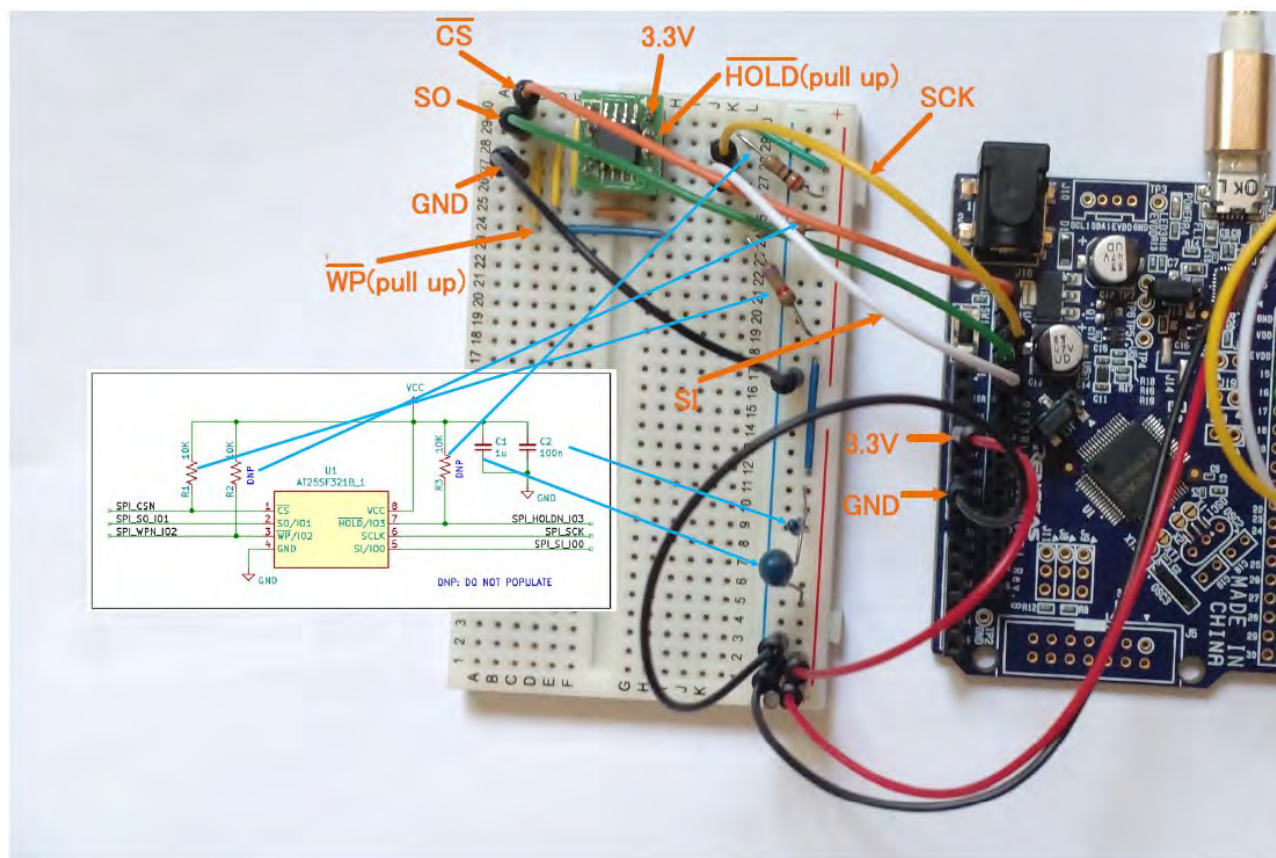


図 3-4 RL78/G23 と NOR FLASH 接続部拡大



補足 R2 と R3 は上記ブレッドボードでは実装しています。

3.3 ソフトウェア構成

ソフトウェア構成を示します。

本 APN では、ADPCM 書き込みプロジェクト (ra01an7416_adpcm_writer) と、ADPCM 再生プロジェクト (ra01an7416_adpcm_player) の 2 本のプロジェクトで構成します。ADPCM 書き込みプロジェクトは、Windows オペレーティングシステムで標準的に使用されるオーディオファイルフォーマットである WAVE (WAV) ファイルを読み込み、ADPCM にエンコード後、NOR FLASH にデータを書き込みます。ADPCM 再生プロジェクトは、NOR FLASH から ADPCM データを読み出し、ADPCM デコードを行い I²S で再生します。

表 3-2 プロジェクトの名称と機能

名称	内容
ra01an7416_adpcm_writer	ADPCM 書き込みプロジェクトです。
ra01an7416_adpcm_player	ADPCM 再生プロジェクトです。

3.4 APN 構成（ファイル構成）

APN のフォルダ構成を示します。

表 3-3 フォルダ構成

フォルダ、ファイル名	説明
ra01an7416xx0100-rl78g23-Communication Interface<DIR>	APN のフォルダ
└ workspace<DIR>	プロジェクト格納用ワークスペース
└ └ player<DIR>	音声再生用プロジェクト
└ └ └ CS+<DIR>	CS+用
└ └ └ └ ra01an7416_adpcm_player_cs<DIR>	CS+用ワークスペース
└ └ └ e2studio<DIR>	e ² studio 用
└ └ └ └ ra01an7416_adpcm_player<DIR>	e ² studio 用ワークスペース
└ └ └ IAR<DIR>	IAR 用
└ └ └ └ ra01an7416_adpcm_player_iar<DIR>	IAR 用ワークスペース
└ └ └ ra01an7416_adpcm_player.mot	ビルド済みバイナリ (Motorola S-record)
└ └ writer<DIR>	音声書き込み用プロジェクト
└ └ └ CS+<DIR>	CS+用
└ └ └ └ ra01an7416_adpcm_writer_cs<DIR>	CS+用ワークスペース
└ └ └ e2studio<DIR>	e ² studio 用
└ └ └ └ ra01an7416_adpcm_writer<DIR>	e ² studio 用ワークスペース
└ └ └ IAR<DIR>	IAR 用
└ └ └ └ ra01an7416_adpcm_writerr_iar<DIR>	IAR 用ワークスペース
└ └ └ ra01an7416_adpcm_writer.mot	ビルド済みバイナリ (Motorola S-record)
└ wave_files<DIR>	音声サンプル用
└ └ wave_8KHz<DIR>	8kHz 音声サンプル
└ └ wave_11.025KHz<DIR>	11.025kHz 音声サンプル
└ └ wave_16KHz<DIR>	16kHz 音声サンプル
└ └ wave_22.05KHz<DIR>	22.05kHz 音声サンプル
└ RA01AN7416JJ0100.pdf	本 APN

4. NOR FLASH への音声データ（ADPCM）書き込み方法

4.1 ソフトウェア説明

まず、ADPCM データを NOR FLASH に書き込む必要があります。ADPCM データを NOR FLASH に書き込むために、本アプリケーションノートに同梱の RL78/G23-64p FPB 用プロジェクト (ra01an7416_adpcm_writer) を使用します。このプロジェクトは、WAVE ファイルを入力とし、RL78/G23-64p FPB で ADPCM エンコードを行い、NOR FLASH にデータを書き込みます。ADPCM データは最大 10 個まで書き込むことができます。

WAVE ファイルは、本アプリケーションノートに同梱のサンプルデータまたは任意の.wav ファイルを使用できますが、表 4-1 の制約があります。

表 4-1 WAVE ファイルの制約

名称	制約
サンプリングレート	8 kHz, 11.025 kHz, 16 kHz, 22.05 kHz のいずれか
ビット深度	16bit のみ
チャンネル数	モノラルのみ
オーディオフォーマット	PCM のみ

補足 対応可能な音声データの長さは 10 秒までです。

4.2 ハードウェア接続

ハードウェア接続については「図 3-1 ハードウェア構成例」を参照してください。

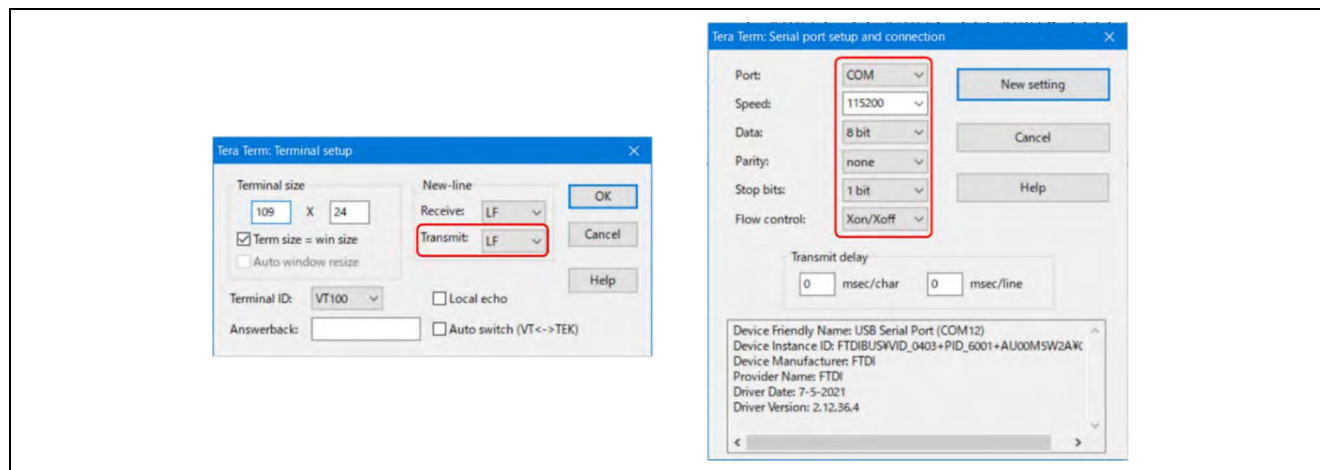
補足 ra01an7416_adpcm_writer の RL78/G23-64p FPB と NOR FLASH の SPI 通信速度は 1MHz としています。
これは、ブレッドボードやジャンパワイヤを使用しての接続における波形訛りを考慮してのことです。

4.3 ターミナルソフト

ra01an7416_adpcm_writer は、TeraTerm などのターミナルソフトを使用して、コマンドを使用して操作します。本 APN では、TeraTerm を使用した場合の例を示します。

Terminal ソフトの設定は以下を参照してください（Flow control には Xon/Xoff を指定してください）。

図 4-1 TeraTerm の設定



4.4 デバッガの使用

ra01an7416_adpcm_writer は、ターミナルソフトとの通信に USB を使用するためデバッガを使用することはできません。Renesas Flash Programmer 等で RL78/G23-64p FPB に書き込み後、スタンドアロンで起動してください。

4.5 コマンド一覧

コマンド一覧を表 4-2 に示します。wr コマンドは、NOR FLASH の任意の番号に ADPCM データを書き込みます。番号は、0-9 の 10 個の中から任意の番号を指定できます。また、wr コマンドは、最大 20 文字のコメントを指定することができます。de コマンドは、NOR FLASH の任意の番号の ADPCM データを削除します。番号は、0-9 の 10 個の中から任意の番号を指定できます。li コマンドは、NOR FLASH に書き込まれている情報を表示します。

表 4-2 コマンド一覧

コマンド	説明
wr_<num>_<comment>	NOR FLASH の番号 num に ADPCM を書き込みます (write)。
de_<num>	NOR FLASH の番号 num のデータを消去します (del)。
li	NOR FLASH に書き込まれている情報を表示します (list)。

補足 _はスペースを表します。

4.6 操作方法

まず、Renesas Flash Programmer を使用して、本 APN に同梱の ra01an7416_adpcm_writer.mot を RL78/G23-64p FPB に書き込みます。

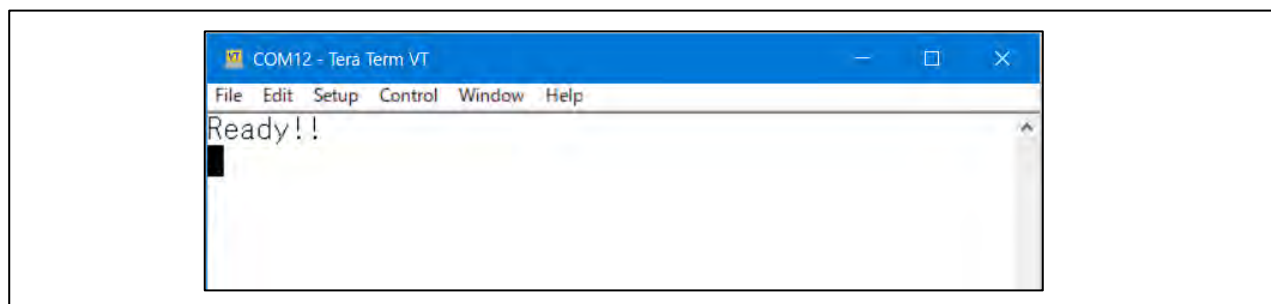
4.6.1 NOR FLASH への書き込み方法

wr コマンドで書き込みを行います。wr コマンドには、引数として 0-9 の書き込み番号および最大 20 文字のコメントを指定します。

(1) 起動

ターミナルソフトを起動し、RL78/G23-64p FPB のリセット SW を押下すると、ターミナル画面に “Ready” と表示されます。

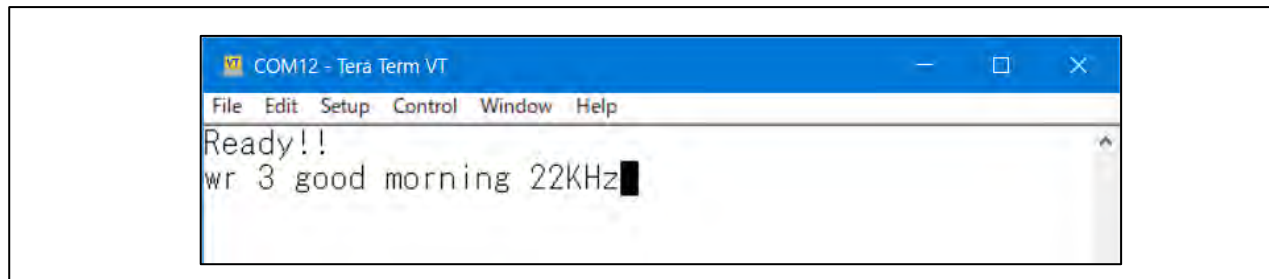
図 4-2 ターミナルソフト画面：起動画面



(2) 書き込みコマンド

書き込みコマンドは“wr”です。wr の後に、0-9 のいずれかの書き込み番号と、20 文字以内のコメントを指定します。

図 4-3 書き込みコマンド入力例



(3) WAVE データ入力

書き込みコマンドを入力後、エンターを押下すると、図 4-4 の通り、“Please drag&drop wave data”と表示されるので、WAVE ファイルをターミナルに Drag&Drop します。Drag&Drop すると、図 4-5 の通り、確認画面が表示されるので、Binary にチェックをつけ、OK をクリックします。

図 4-4 WAVE 入力要求画面

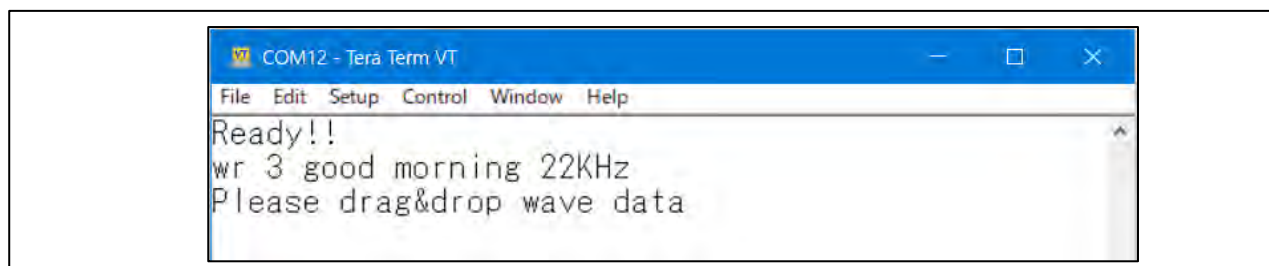
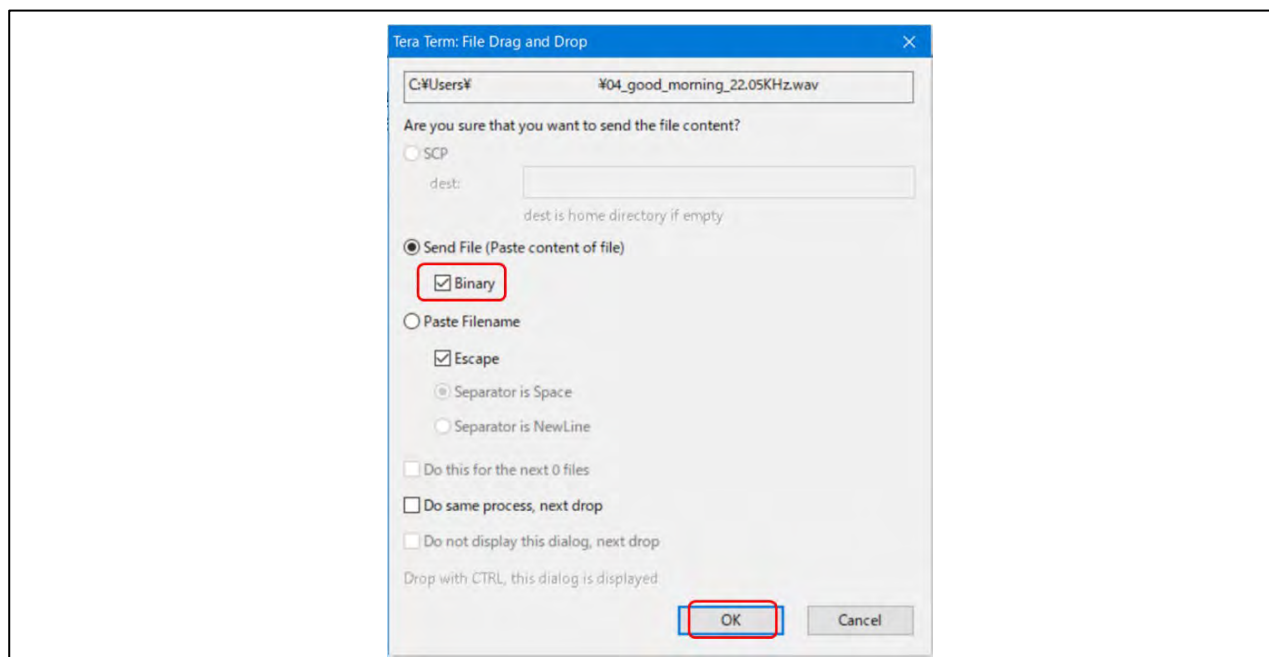


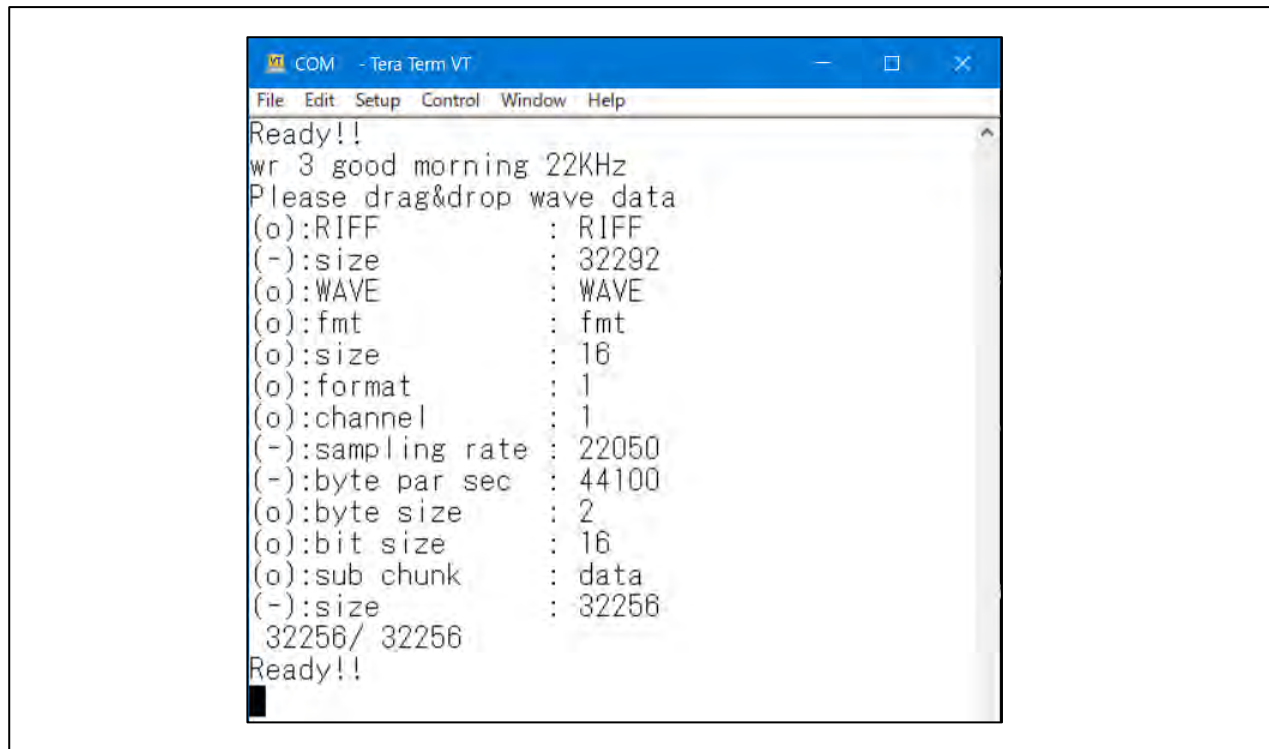
図 4-5 WAVE 設定画面



(4) 書き込み完了

書き込みが完了すると、図 4-6 のように、WAVE の情報を表示後、再び “Ready!!” が表示されます。

図 4-6 書き込み完了画面



- 注
1. WAVE の情報に(x)が表示された場合、WAVE ファイルにフォーマットエラーがあることを示します。その場合、正常に書き込めません（エラー処理などは行っていない）。
 2. 通信品質の影響により TeraTerm などのターミナルソフトから RL78/G23 にファイルサイズ以上のデータが送信されると、最後の Ready が表示されずにデッドロックします。この場合は、RL78/G23 をリセットして、再度書き込みを行ってください。

4.6.2 NOR FLASH 内容表示方法

li コマンドで NOR FLASH の内容を表示することができます。li コマンドの使用例を図 4-7 に示します。

図 4-7 li コマンド使用例

Num	size	sampling rate	name
0: (x)			
1: (o)	0x00004800	22050	we speak
2: (o)	0x0000b1c0	22050	renesas 22.05KHz
3: (o)	0x00001f80	22050	good morning 22KHz
4: (o)	0x000058e0	11025	renesas 11.025KHz
5: (o)	0x00004260	8000	renesas 8KHz
6: (x)			
7: (o)	0x00001f80	22050	good by 22KHz
8: (x)			
9: (x)			

4.6.3 NOR FLASH 消去方法

de コマンドで NOR FLASH の内容を消去することができます。de コマンドの使用例を図 4-8 に示します。de コマンド後に、li コマンドを実行すると、4 番が削除されていることが確認できます。

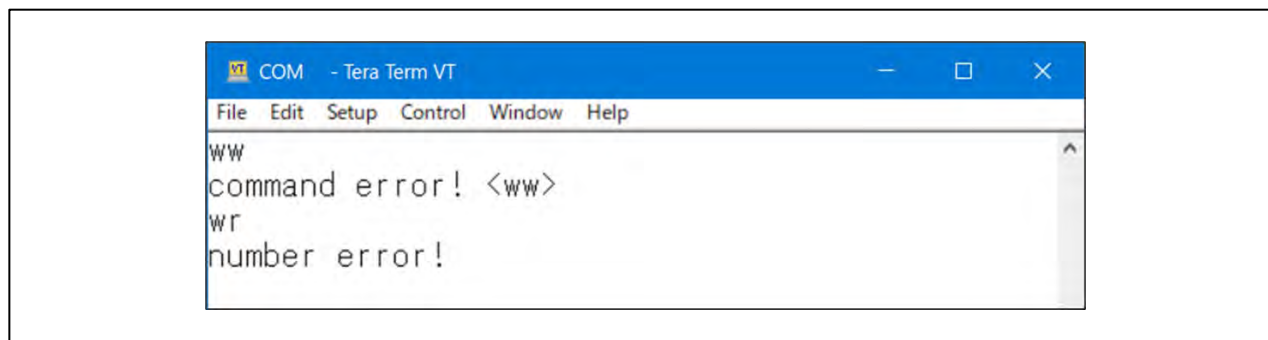
図 4-8 de コマンド使用例

Num	size	sampling rate	name
0: (x)			
1: (o)	0x00004800	22050	we speak
2: (o)	0x0000b1c0	22050	renesas 22.05KHz
3: (o)	0x00001f80	22050	good morning 22KHz
4: (x)			
5: (o)	0x00004260	8000	renesas 8KHz
6: (x)			
7: (o)	0x00001f80	22050	good by 22KHz
8: (x)			
9: (x)			

4.7 エラー表示

実行できないコマンドや、番号を指定しなかった場合などエラーを表示します。図 4-9 にエラー表示例を示します。

図 4-9 エラー表示例



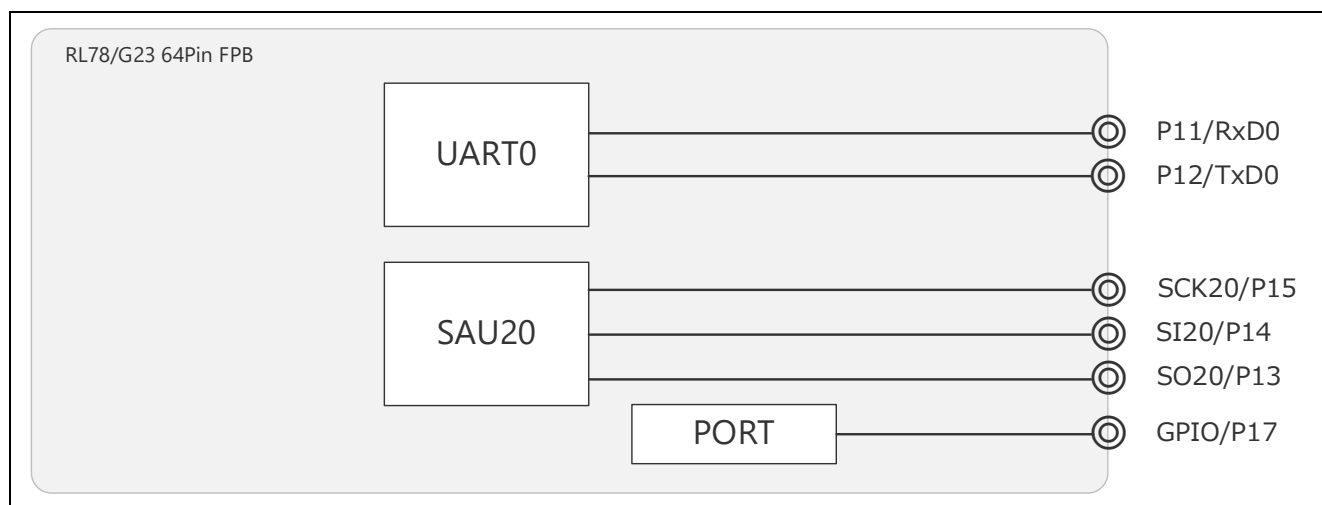
4.8 ソフトウェア説明詳細

4.8.1 動作概要

本サンプルコードは、WAVE ファイルを ADPCM 変換し、SPI 通信(SAU20 と PORT)によって、Renesas 製 AT25QF641B (NOR FLASH) に書き込むプログラムです。本サンプルコードは、「wr/li/de」の 3 つのコマンドを使用し、TeraTerm などのターミナルソフト経由で操作します。ターミナルソフトとの通信に UART0 を使用しているため、デバッガは使用できません (スタンドアロンで動作します)。本サンプルコードは、図 4-10 のみ必要としていますが、図 3-1 と全く同じ構成で使うことができます。

図 4-10 にサンプルコードのシステム構成を示します。

図 4-10 サンプルコードのシステム構成



4.8.2 フォルダ構成

表 4-3 に本サンプルコードで使用するソースファイル／ヘッダファイルの構成を示します。なお、統合開発環境で自動生成されるファイル、bsp 環境のファイルは除きます。

表 4-3 フォルダ構成

フォルダ、ファイル名	説明	スマート・コンフィグレータを使用
r01an7416_adpcm_writer<DIR> ^{注1}	サンプルコードのフォルダ	
└ src<DIR>	プログラム格納用フォルダ	
└ s2<DIR>	M3S-S2-Tiny (ADPCM encoder/decoder)	
└ smc_gen<DIR>	スマート・コンフィグレータ生成フォルダ	√
└ └ Config_CSI20	CSI20 用プログラム格納フォルダ	√
└ └ └ Config_CSI20.c	CSI20 用ソースファイル	√
└ └ └ Config_CSI20.h	CSI20 用ヘッダファイル	√
└ └ └ Config_CSI20_user.c	CSI20 用割り込みソースファイル	√
└ └ Config_UART0	UART0 用プログラム格納フォルダ	√
└ └ └ Config_UART0.c	UART0 用ソースファイル	√
└ └ └ Config_UART0.h	UART0 用ヘッダファイル	√
└ └ └ Config_UART0_user.c	UART0 用割り込みソースファイル	√
└ └ general<DIR>	初期化、共通プログラム格納フォルダ	
└ └ r_bsp<DIR>	BSP 用プログラム格納フォルダ	
└ └ r_config<DIR>	プログラム格納フォルダ	
└ └ r_nor_flash_rl78<DIR>	NOR FLASH ライブラリ格納フォルダ ^{注2}	
└ └ r_pincfg<DIR>	pincfg 格納フォルダ	
└ main.c	メインプログラム	
└ main.h	ヘッダファイル	
└ r_adpcm_writer_main.c	NOR FLASH 書き込みメインプログラム	
└ r_adpcm_writer_main.h	ヘッダファイル	
└ r_flash_write.c	NOR FLASH 書き込み処理	
└ r_flash_write.h	ヘッダファイル	
└ r_wave_decoder.c	WAVE デコード処理	
└ r_wave_decoder.h	ヘッダファイル	

補足 ” <DIR> ” は、ディレクトリを意味します。

注1. IAR 版のサンプルコードは、ra01an7416_adpcm_writer_iar.ipcf を格納しています。ipcf ファイルについては、「RL78 スマート・コンフィグレータ ユーザーガイド：IAR 編（R20AN0581）」を確認してください。

注2. Smart Configurator のコンポーネント追加で、RL78 Family Serial NOR Flash Memory 制御 モジュール Software Integration System モジュールをダウンロードしてください。

4.8.3 オプション・バイトの設定一覧

表 4-4 に本サンプルコードで使用するオプション・バイト設定を示します。

表 4-4 オプション・バイト設定

アドレス	設定値	内容
000C0H/040C0H	1110 1111B (EFH)	ウォッチドッグ・タイマ動作停止 (リセット解除後、カウント停止)
000C1H/040C1H	1111 1110B (FEH)	LVD0 リセット・モード 検出電圧：立ち上がり 1.90V／立下り 1.86V
000C2H/040C2H	1110 1000B (E8H)	フラッシュ動作モード：高速メインモード 高速オンチップ・オシレータの周波数：32MHz
000C3H/040C3H	1111 1011B (EBH)	オンチップ・デバッグ使用不可

4.8.4 ROM/RAM サイズ

表 4-5 に本サンプルコードで使用する ROM/RAM サイズを記載します（CC-RL 最適化レベル=一部の最適化(-O-lite)でのサイズです。最適化レベルで変動します）。

表 4-5 ROM/RAM サイズ

サンプルプロジェクト名	ROM サイズ	RAM サイズ
ra01an7416_adpcm_writer	20631Byte	1031Byte

4.8.5 定数一覧

表 4-6 にサンプルコードで使用する定数を示します。

表 4-6 サンプルコードで使用する定数

定数名	設定値	内容	ファイル
WAVE_DATA_BUF_SIZE	256	PCM バッファサイズ	r_flash_writer_main.h

4.8.6 変数一覧

表 4-7 に本サンプルコードで使用するグローバル変数を示します。

表 4-7 サンプルコードで使用するグローバル変数

型	変数名	内容	使用関数
st_flow_ctrl_t	g_flow_ctrl	フロー制御構造体	Config_UART0_user.c r_flash_writer_main.c r_wave_decoder.c
st_wave_data_t	g_wave_data	ADPCM 用構造体	Config_UART0_user.c r_flash_writer_main.c r_wave_decoder.c
e_flow_state_t	g_flow_state	ステート	Config_UART0_user.c r_flash_writer_main.c
st_nor_flash_info_t	g_nor_flash_info_p	Flash 書き込み構造体	r_flash_writer_main.c r_wave_decoder.c
st_nor_flash_erase_info_t	g_nor_flash_info_e	Flash 消去構造体	r_flash_writer_main.c r_wave_decoder.c

4.8.7 関数一覧

表 4-8 にサンプルコードで使用する関数を示します。ただし、スマート・コンフィグレータで生成された関数の内、変更を行っていないものは除きます。

また、「[RL78/G23 ELCL、SPI による I²S 通信 アプリケーションノート](#)」から変更していない関数については、本アプリケーションノートには記載していません。「[RL78/G23 ELCL、SPI による I²S 通信 アプリケーションノート](#)」を参照してください。

表 4-8 関数一覧

	関数名	概要	ソースファイル
g	main	メイン処理	main.c
g	r_adpcm_writer_main	adpcm_writer のメイン処理	r_adpcm_writer_main.c
s	init	adpcm_writer 初期化	r_adpcm_writer_main.c
s	state_ctrl	adpcm_writer のステート制御	r_adpcm_writer_main.c
s	check_command	PC から受け取ったコマンドの処理	r_adpcm_writer_main.c
s	get_flow_read_next	flow_read の次のステートを返す	r_adpcm_writer_main.c
s	get_flow_enc_adpcm	flow_enc の次のステートを返す	r_adpcm_writer_main.c
s	status_init	ステータス初期化	r_adpcm_writer_main.c
g	wait_send_end	PC への送信完了待ち合わせ	r_adpcm_writer_main.c
g	wait_and_print	PC への送信完了待ち合わせ&print	r_adpcm_writer_main.c
g	wait_and_print_len	送信文字長つき wait_and_print	r_adpcm_writer_main.c
s	adpcm_encode_8byte	ADPCM エンコード処理	r_adpcm_writer_main.c
s	write_nor_flash	NOR FLASH への書き込み処理	r_adpcm_writer_main.c
s	check_last	最終 ADPCM 変換と書き込み	r_adpcm_writer_main.c
s	parse_command	PC から受け取ったコマンドの解析	r_adpcm_writer_main.c
s	write_header_nor_flash	NOR FLASH へのヘッダ情報書き込み	r_adpcm_writer_main.c
s	check_success_and_busy	NOR FLASH への書き込み完了待ち合わせ	r_nor_flash_write.c
s	init	r_nor_flash_write の初期化	r_nor_flash_write.c
g	r_nor_flash_data_write	NOR FLASH へのデータ書き込み	r_nor_flash_write.c
g	r_nor_flash_data_read	NOR FLASH からのデータ読み込み	r_nor_flash_write.c
g	r_wave_decoder_parse	WAVE フォーマットチェック	r_wave_decoder.c

注 1. 先頭の「g」はグローバル関数、「s」はスタティック関数を示します。

4.8.8 関数仕様

サンプルコードの関数仕様を示します。

なお、「[RL78/G23 ELCL、SPI による I²S 通信 アプリケーションノート](#)」から変更していない関数については、本アプリケーションノートには記載していません。「[RL78/G23 ELCL、SPI による I²S 通信 アプリケーションノート](#)」を参照してください。

[関数名] main

概要	メイン処理
ヘッダ	stdint.h, stdbool.h, main.h, r_adpcm_writer_main.h,
宣言	void main (void);
説明	main 関数です。r_adpcm_writer_main をコールします。
引数	なし
リターン値	なし
備考	なし

[関数名] r_adpcm_writer_main

概要	adpcm_writer のメイン処理
ヘッダ	stdint.h, stdbool.h, string.h, r_adpcm_writer_main.h, r_nor_flash_write.h, r_wave_decoder.h
宣言	void r_adpcm_writer_main (void);
説明	adpcm_writer の main 関数です。
引数	なし
リターン値	なし
備考	なし

[関数名] init

概要	adpcm_writer 初期化
ヘッダ	stdint.h, stdbool.h, string.h, r_adpcm_writer_main.h, r_nor_flash_write.h, r_wave_decoder.h
宣言	static void init (void);
説明	adpcm_writer の初期化を行います。
引数	なし
リターン値	なし
備考	なし

[関数名] state_ctrl

概要	adpcm_writer のステート制御
ヘッダ	stdint.h, stdbool.h, string.h, r_adpcm_writer_main.h, r_nor_flash_write.h, r_wave_decoder.h
宣言	static void state_ctrl (void);
説明	adpcm_writer のステート制御を行います。
引数	なし
リターン値	なし
備考	なし

[関数名] check_command

概要	PC から受け取ったコマンドの処理
ヘッダ	stdint.h, stdbool.h, string.h, r_adpcm_writer_main.h, r_nor_flash_write.h, r_wave_decoder.h
宣言	static void check_command (void);
説明	PC から受け取ったコマンドの処理をおこないます。
引数	なし
リターン値	なし
備考	なし

[関数名] get_flow_read_next

概要	flow_read の次のステートを返す
ヘッダ	stdint.h, stdbool.h, string.h, r_adpcm_writer_main.h, r_nor_flash_write.h, r_wave_decoder.h
宣言	static void get_flow_read_next (void);
説明	flow_read の次のステートを返します。
引数	なし
リターン値	なし
備考	なし

[関数名] get_flow_enc_adpcm

概要	flow_enc の次のステートを返す
ヘッダ	stdint.h, stdbool.h, string.h, r_adpcm_writer_main.h, r_nor_flash_write.h, r_wave_decoder.h
宣言	static void get_flow_enc_adpcm (void);
説明	flow_enc の次のステートを返します
引数	なし
リターン値	なし
備考	なし

[関数名] status_init

概要	ステータス初期化
ヘッダ	stdint.h, stdbool.h, string.h, r_adpcm_writer_main.h, r_nor_flash_write.h, r_wave_decoder.h
宣言	static void status_init (void);
説明	ステータスを初期化します。
引数	なし
リターン値	なし
備考	なし

[関数名] wait_send_end

概要	PC への送信完了待ち合わせ
ヘッダ	stdint.h, stdbool.h, string.h, r_adpcm_writer_main.h, r_nor_flash_write.h, r_wave_decoder.h
宣言	void wait_send_end (void);
説明	PC への送信完了待ち合わせを行います。
引数	なし
リターン値	なし
備考	なし

[関数名] wait_and_print

概要	PC への送信完了待ち合わせ&print
ヘッダ	stdint.h, stdbool.h, string.h, r_adpcm_writer_main.h, r_nor_flash_write.h, r_wave_decoder.h
宣言	void wait_and_print (void);
説明	PC への送信完了待ち合わせ後、続けて PC への送信を行います。
引数	なし
リターン値	なし
備考	なし

[関数名] wait_and_print_len

概要	送信文字長つき wait_and_print
ヘッダ	stdint.h, stdbool.h, string.h, r_adpcm_writer_main.h, r_nor_flash_write.h, r_wave_decoder.h
宣言	void wait_and_print_len (void);
説明	送信文字長つき wait_and_print です。
引数	なし
リターン値	なし
備考	なし

[関数名] adpcm_encode_8byte

概要	ADPCM エンコード処理
ヘッダ	stdint.h, stdbool.h, string.h, r_adpcm_writer_main.h, r_nor_flash_write.h, r_wave_decoder.h
宣言	static void adpcm_encode_8byte (void);
説明	ADPCM エンコード処理 (8Byte->2Byte) を行います。
引数	なし
リターン値	なし
備考	なし

[関数名] write_nor_flash

概要	NOR FLASH への書き込み処理
ヘッダ	stdint.h, stdbool.h, string.h, r_adpcm_writer_main.h, r_nor_flash_write.h, r_wave_decoder.h
宣言	static void write_nor_flash (void);
説明	NOR FLASH への書き込み処理を行います。
引数	なし
リターン値	なし
備考	なし

[関数名] check_last

概要	最終 ADPCM 変換と書き込み
ヘッダ	stdint.h, stdbool.h, string.h, r_adpcm_writer_main.h, r_nor_flash_write.h, r_wave_decoder.h
宣言	static void check_last (void);
説明	256Byte に満たない PCM の ADPCM 変換と NOR FLASH への書き込みを行います。
引数	なし
リターン値	なし
備考	なし

[関数名] parse_command

概要	PC から受け取ったコマンドの解析
ヘッダ	stdint.h, stdbool.h, string.h, r_adpcm_writer_main.h, r_nor_flash_write.h, r_wave_decoder.h
宣言	static void parse_command (void);
説明	PC から受け取ったコマンドの解析を行います。
引数	なし
リターン値	なし
備考	なし

[関数名] write_header_nor_flash

概要	NOR FLASH へのヘッダ情報書き込み
ヘッダ	stdint.h, stdbool.h, string.h, r_adpcm_writer_main.h, r_nor_flash_write.h, r_wave_decoder.h
宣言	static void write_header_nor_flash (void);
説明	NOR FLASH に、ADPCM の有無およびサイズを書き込みます。
引数	なし
リターン値	なし
備考	なし

[関数名] check_success_and_busy

概要	NOR FLASH への書き込み完了待ち合わせ
ヘッダ	stdint.h, stdbool.h, string.h, r_adpcm_writer_main.h, r_nor_flash_write.h, r_wave_decoder.h
宣言	static void check_success_and_busy (void);
説明	NOR FLASH への書き込み完了待ち合わせを行います。
引数	なし
リターン値	なし
備考	なし

[関数名] init(r_nor_flash_write.c)

概要	r_nor_flash_write の初期化
ヘッダ	stdint.h, stdbool.h, string.h, r_smc_entry.h, r_adpcm_writer_main.h, r_nor_flash_write.h
宣言	static void init (void);
説明	r_nor_flash_write の初期化を行います。r_nor_flash_write の初期化
引数	なし
リターン値	なし
備考	なし

[関数名] r_nor_flash_data_write

概要	NOR FLASH へのデータ書き込み
ヘッダ	stdint.h, stdbool.h, string.h, r_smc_entry.h, r_adpcm_writer_main.h, r_nor_flash_write.h
宣言	void r_nor_flash_data_write (void);
説明	NOR FLASH へのデータ書き込みを行います。
引数	なし
リターン値	なし
備考	なし

[関数名] r_nor_flash_data_read

概要	
ヘッダ	stdint.h, stdbool.h, string.h, r_smc_entry.h, r_adpcm_writer_main.h, r_nor_flash_write.h
宣言	void r_nor_flash_data_read (void);
説明	
引数	なし
リターン値	なし
備考	なし

[関数名] r_wave_decoder_parse

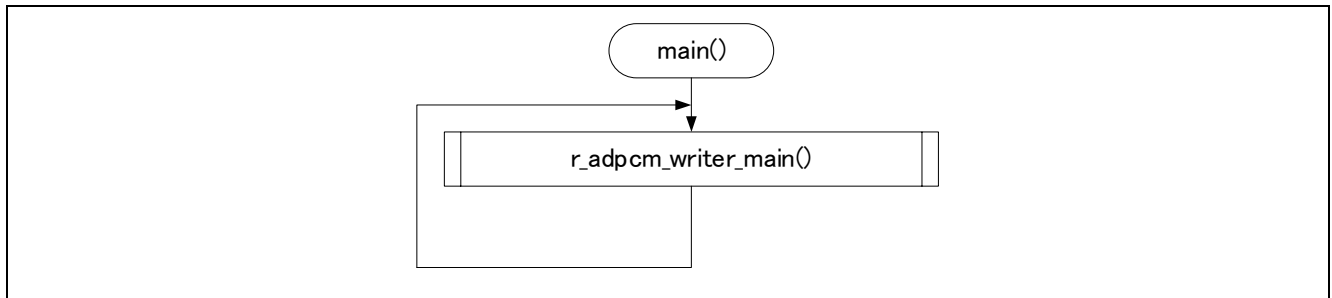
概要	WAVE フォーマットチェック
ヘッダ	stdint.h, stdbool.h, r_adpcm_writer_main.h
宣言	void r_wave_decoder_parse (void);
説明	WAVE フォーマットチェック（解析）を行います。
引数	なし
リターン値	なし
備考	なし

4.8.9 フローチャート

4.8.9.1 main ()

図 4-11 に main () のフローチャートを示します。

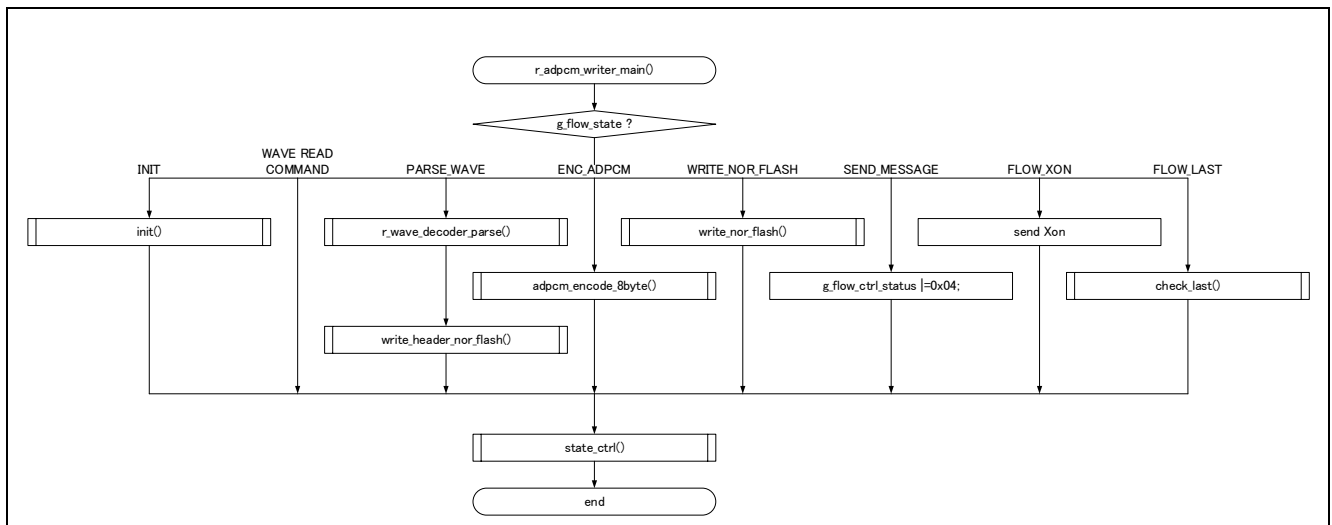
図 4-11 main ()



4.8.9.2 r_adpcm_writer_main()

図 4-12 に `r_adpcm_writer_main()` のフローチャートを示します。

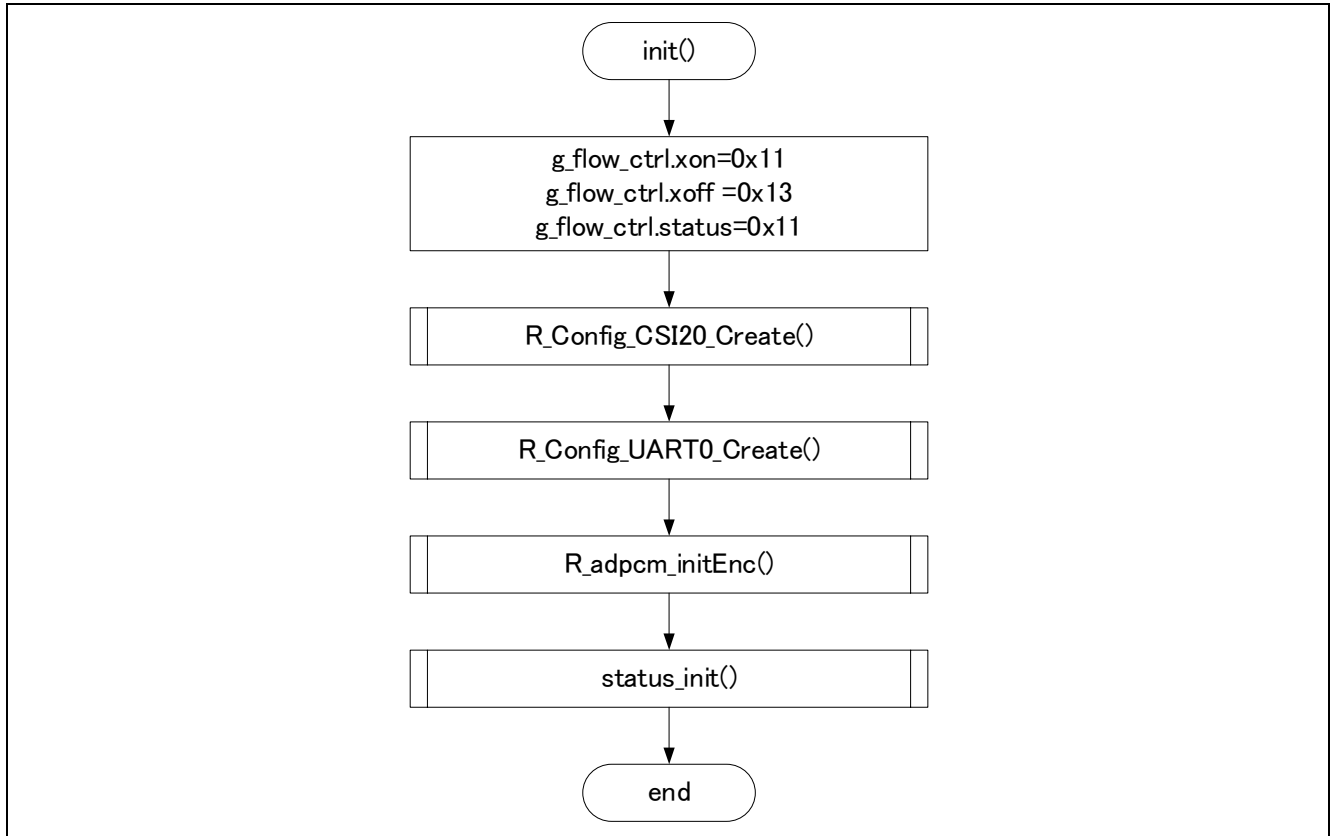
図 4-12 `r_adpcm_writer_main()`



4.8.9.3 init ()

図 4-13 に `init()` のフローチャートを示します。

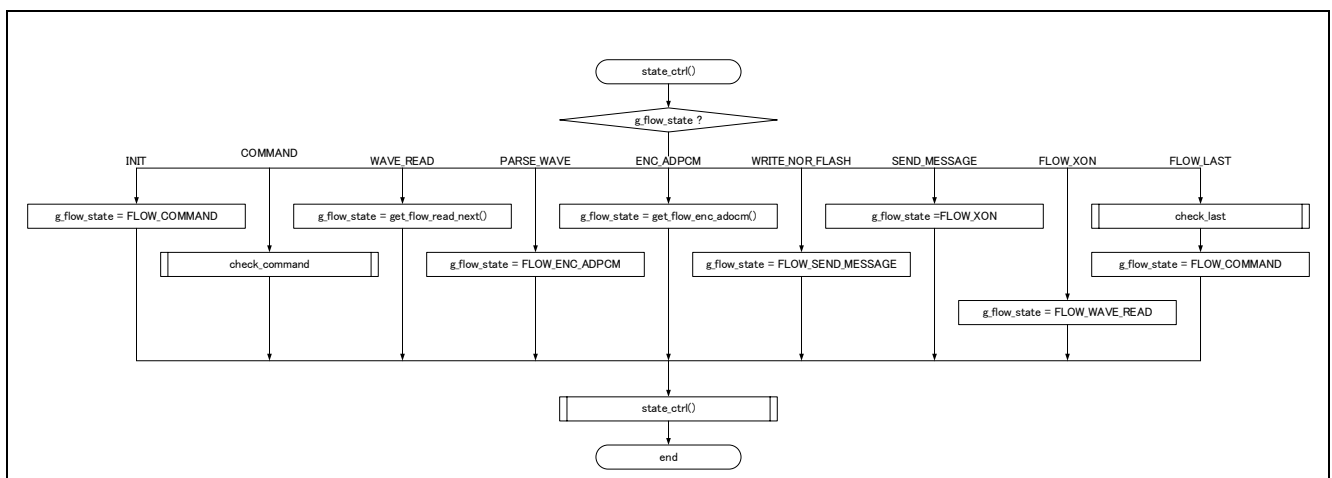
图 4-13 init()



4.8.9.4 state_ctrl()

図 4-14 に `state_ctrl()` のフローチャートを示します。

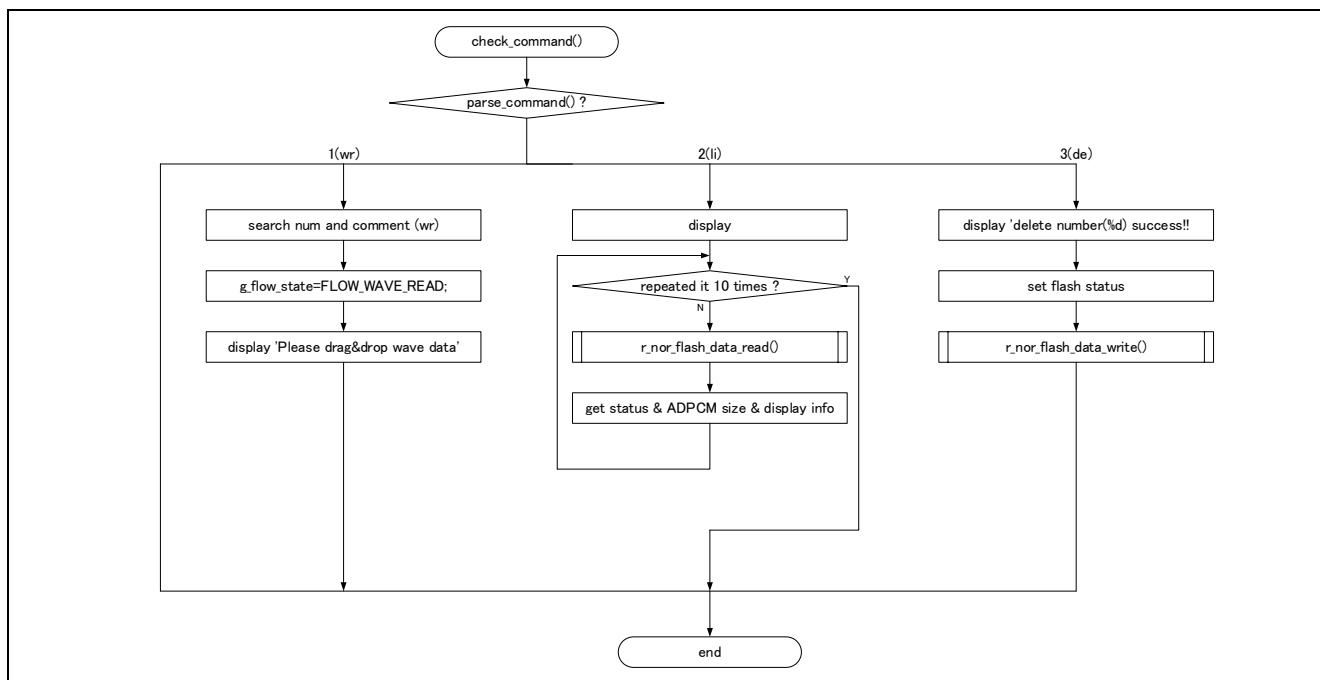
图 4-14 state_ctrl()



4.8.9.5 check_command ()

図 4-15 に check_command() のフローチャートを示します。

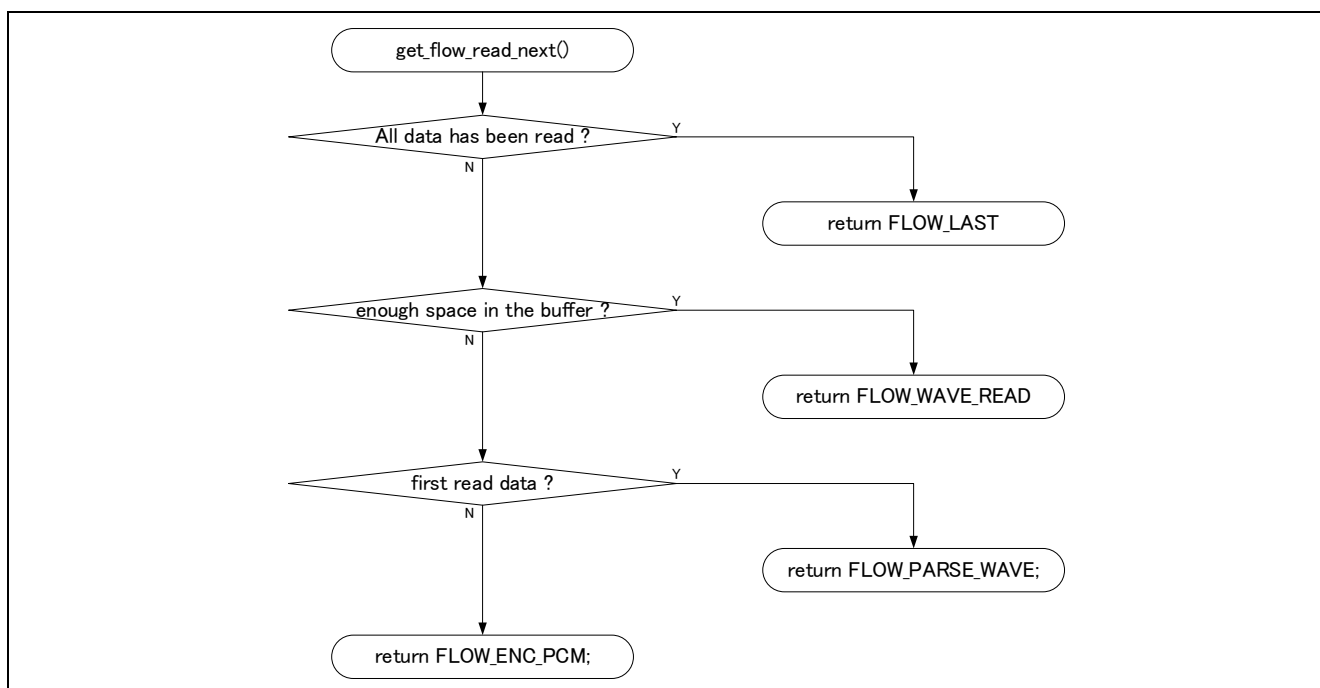
図 4-15 check_command()



4.8.9.6 get_flow_read_next ()

図 4-16 に get_flow_read_next() のフローチャートを示します。

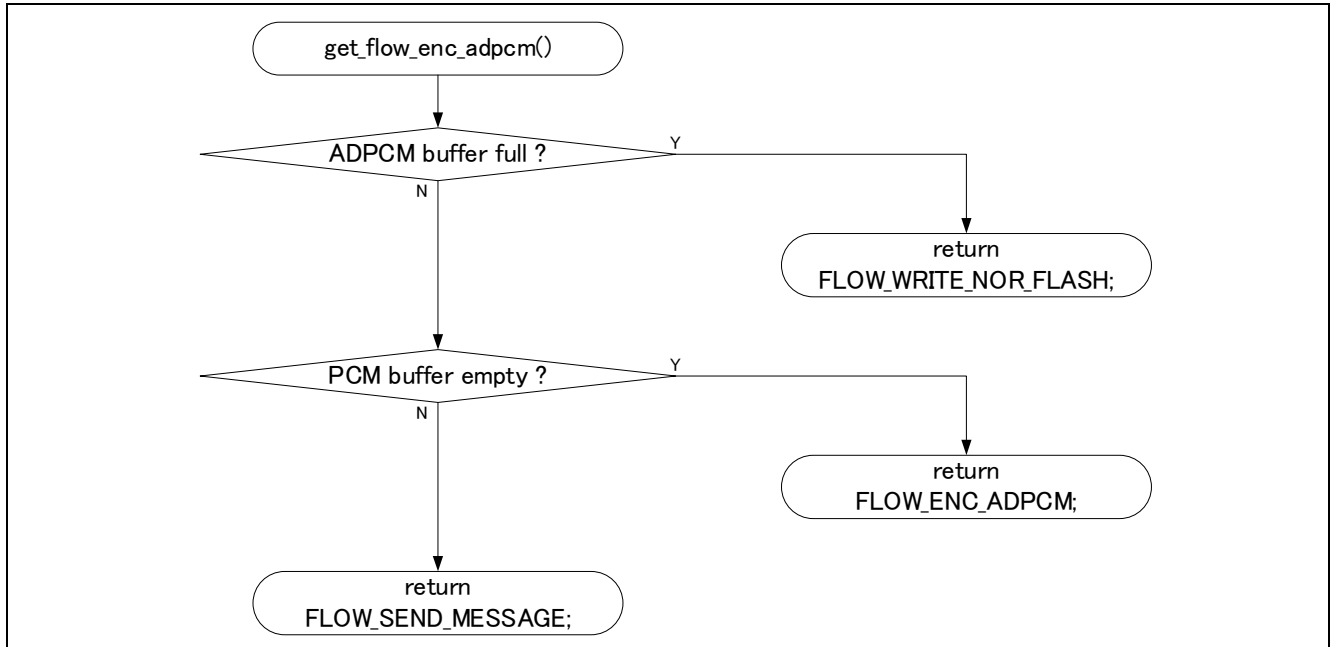
図 4-16 get_flow_read_next()



4.8.9.7 get_flow_enc_adpcm ()

図 4-17 に get_flow_enc_adpcm()のフローチャートを示します。

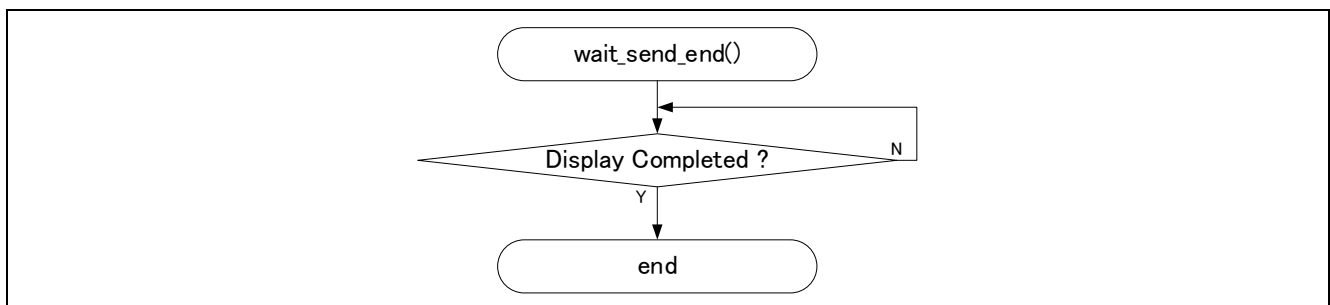
図 4-17 get_flow_enc_adpcm()



4.8.9.8 wait_send_end ()

図 4-18 に wait_send_end ()のフローチャートを示します。

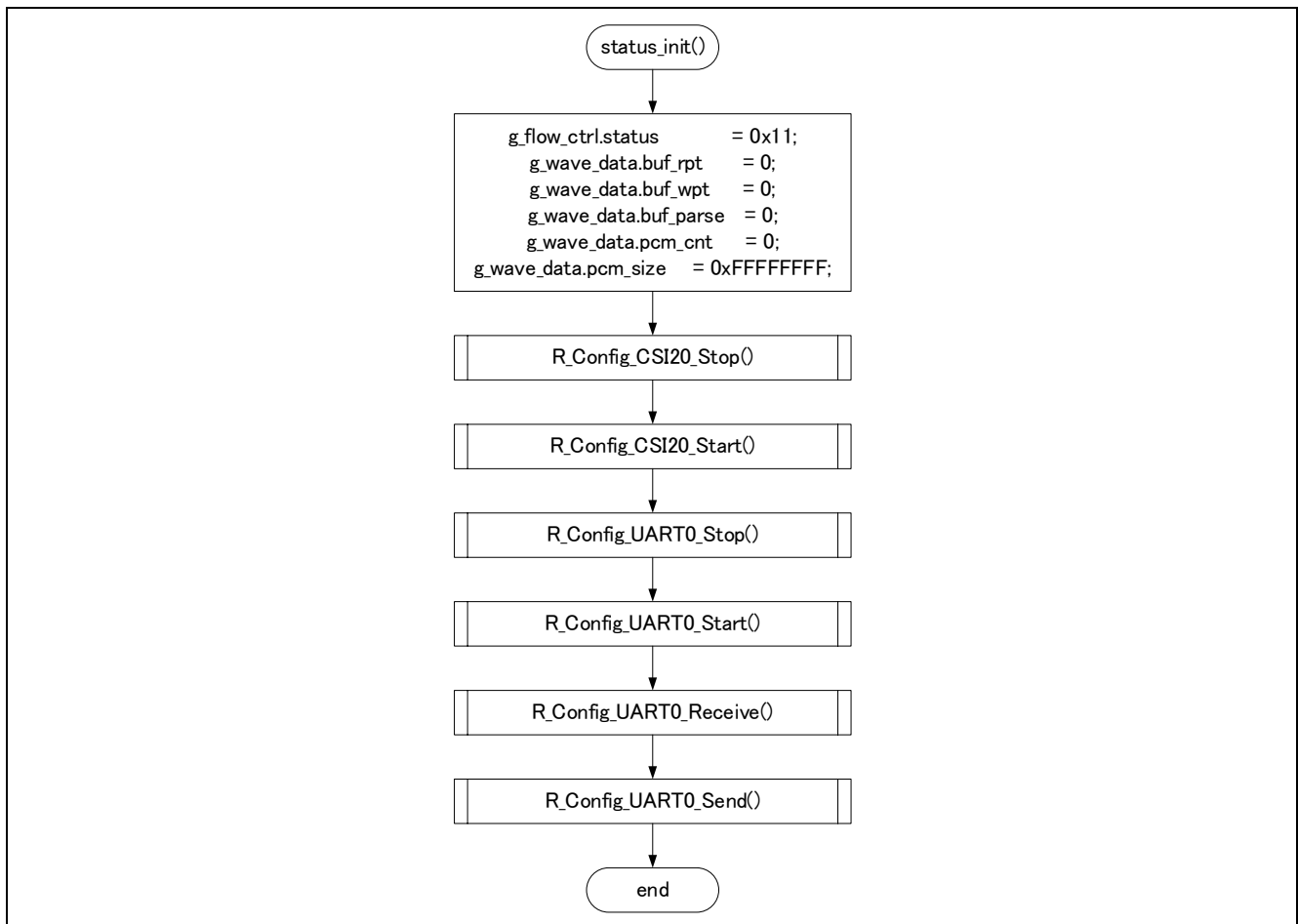
図 4-18 wait_send_end ()



4.8.9.9 status_init ()

図 4-19 に status_init()のフローチャートを示します。

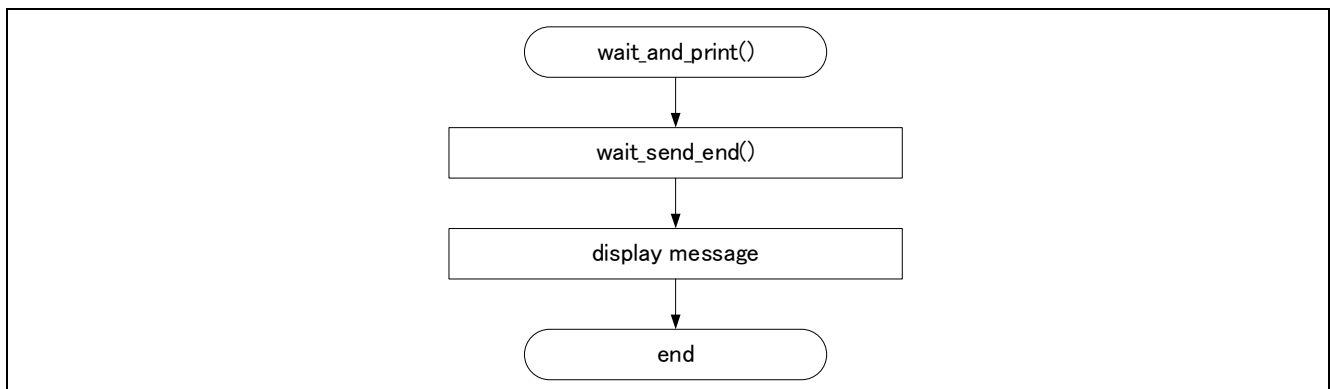
図 4-19 status_init()



4.8.9.10 wait_and_print ()

図 4-20 に wait_and_print ()のフローチャートを示します。

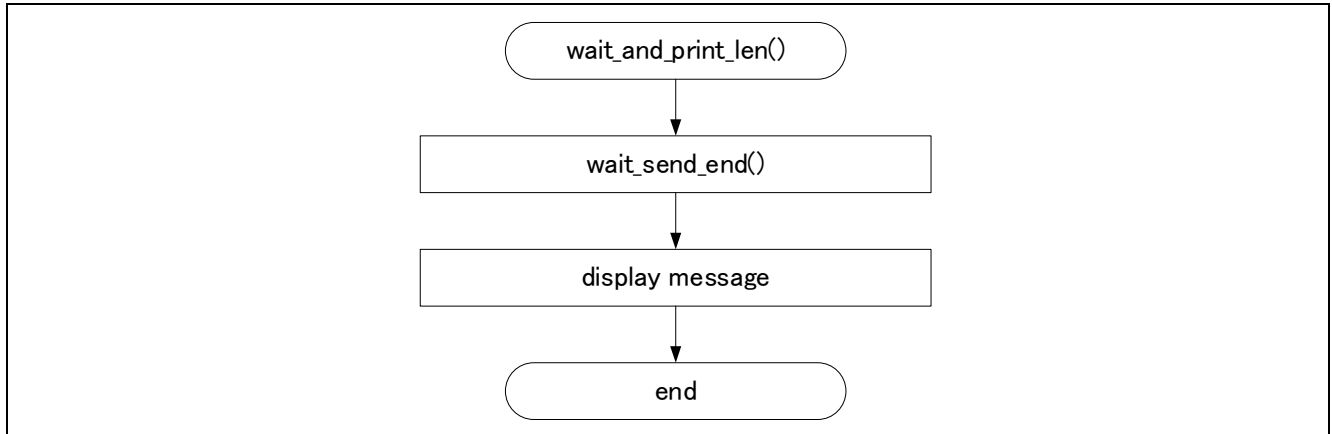
図 4-20 wait_and_print ()



4.8.9.11 wait_and_print_len()

図 4-21 に wait_and_print_len()のフローチャートを示します。

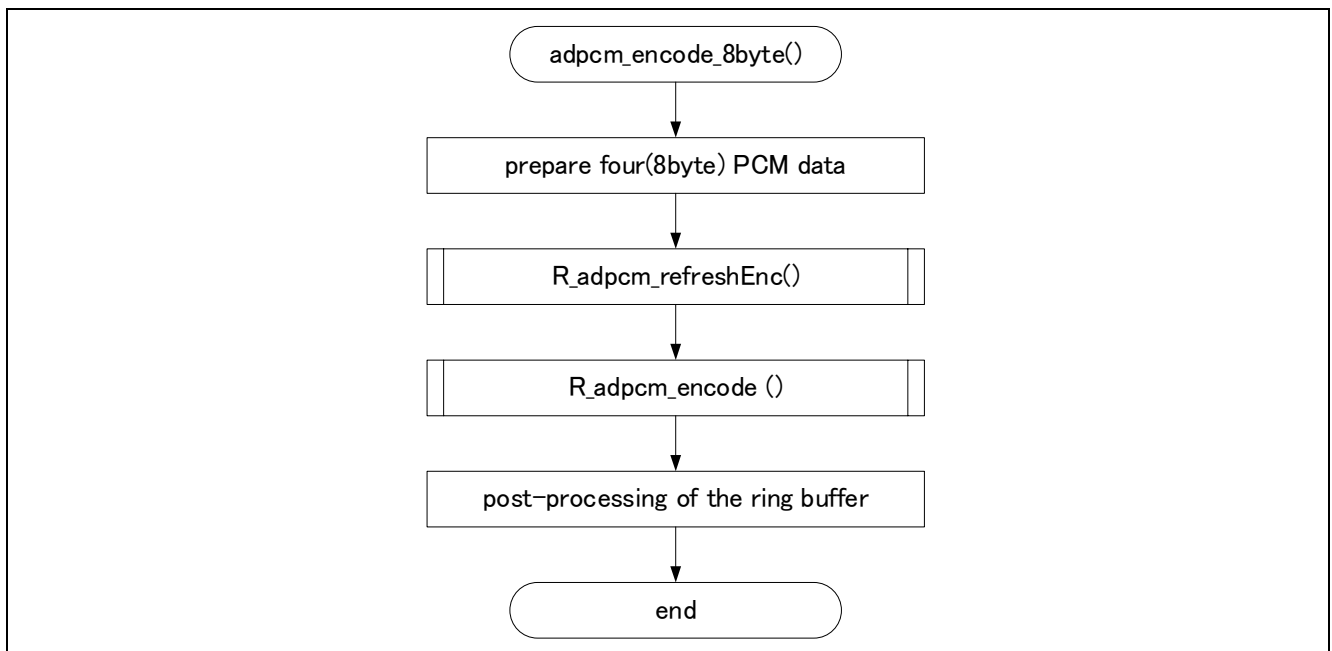
図 4-21 wait_and_print_len()



4.8.9.12 adpcm_encode_8byte()

図 4-22 に adpcm_encode_8byte()のフローチャートを示します。

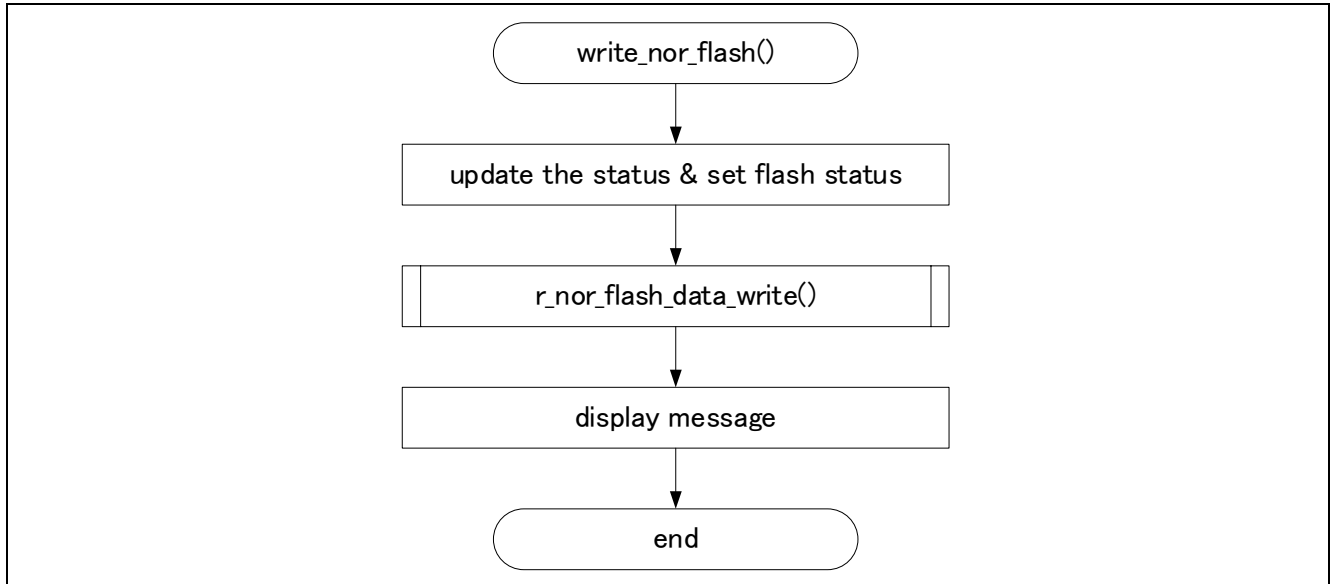
図 4-22 adpcm_encode_8byte()



4.8.9.13 write_nor_flash()

図 4-23 に write_nor_flash() のフローチャートを示します。

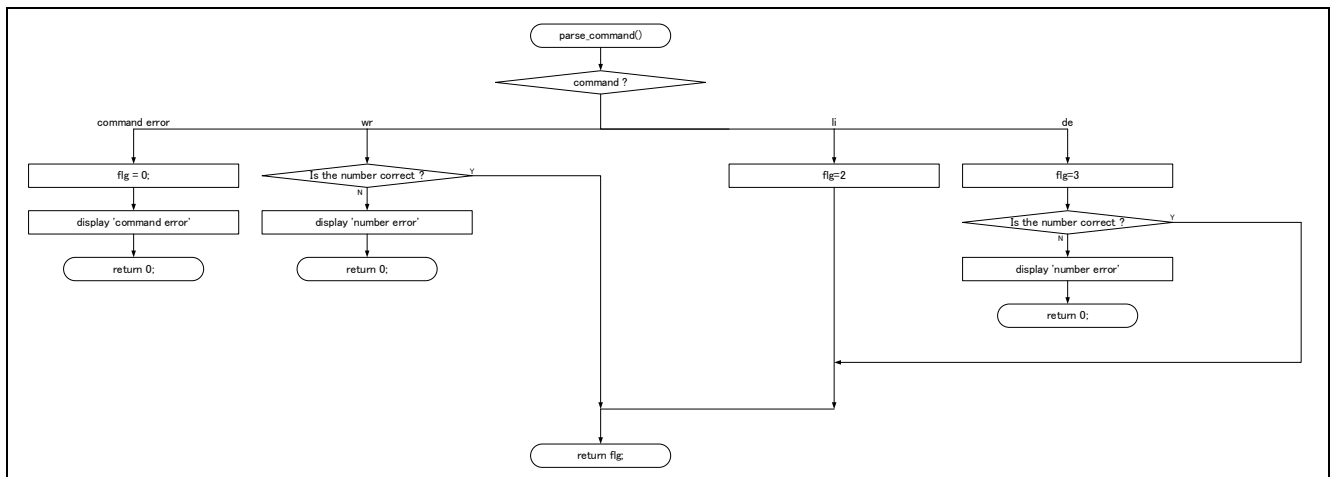
図 4-23 write_nor_flash()



4.8.9.14 parse_command()

図 4-24 に parse_command() のフローチャートを示します。

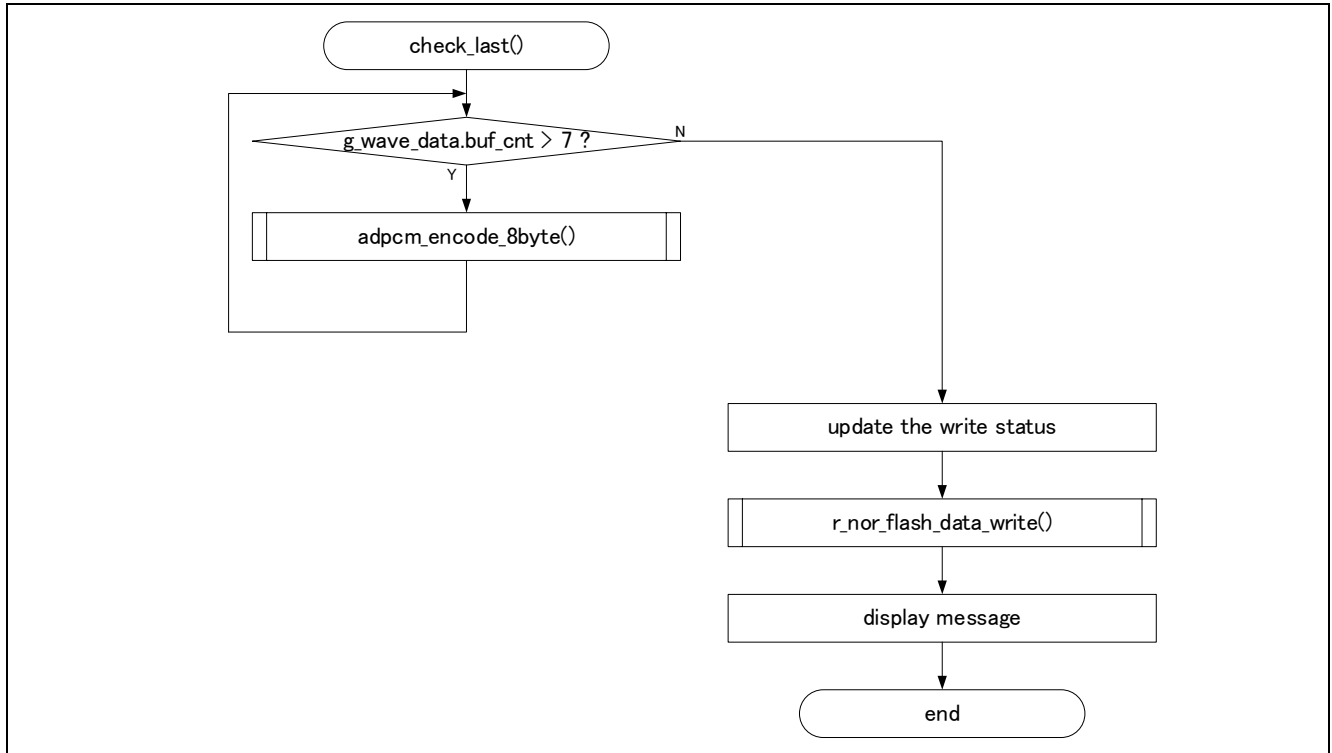
図 4-24 parse_command()



4.8.9.15 check_last()

図 4-25 に check_last()のフローチャートを示します。

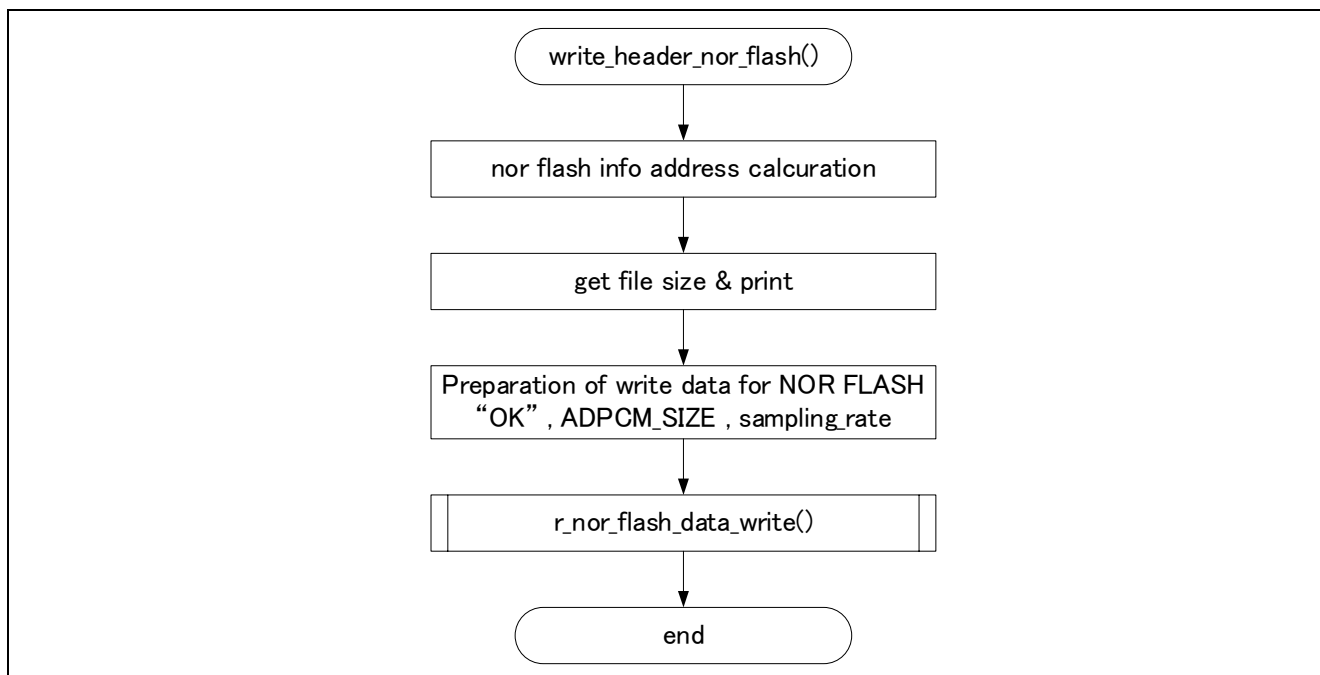
図 4-25 check_last()



4.8.9.16 write_header_nor_flash ()

図 4-26 に write_header_nor_flash () のフローチャートを示します。

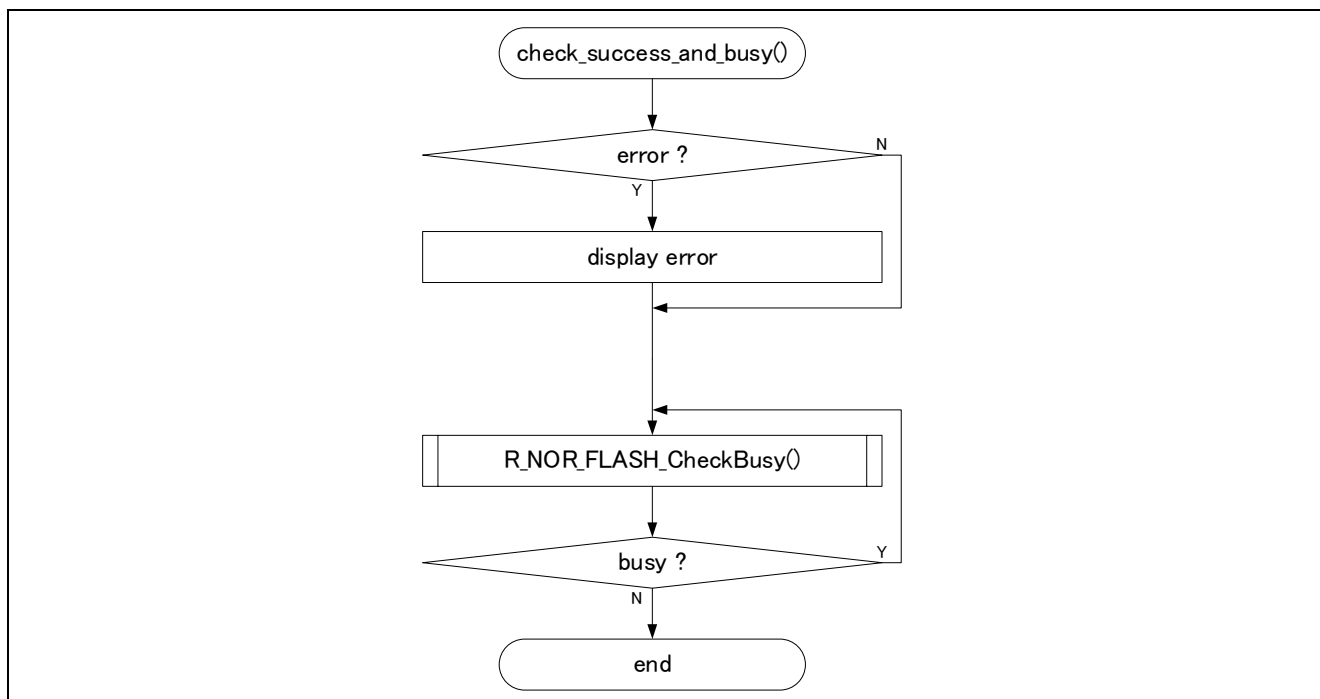
図 4-26 write_header_nor_flash ()



4.8.9.17 check_success_and_busy()

図 4-27 に check_success_and_busy() のフローチャートを示します。

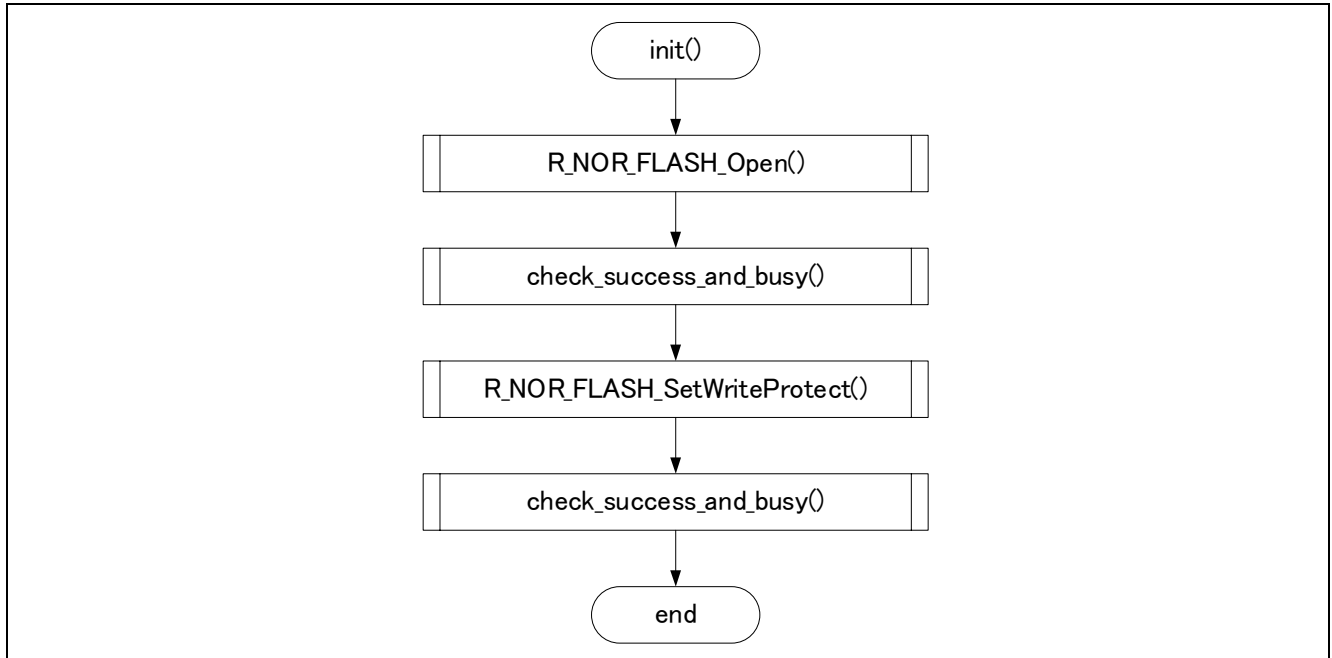
図 4-27 check_success_and_busy()



4.8.9.18 init ()

図 4-28 に init () のフローチャートを示します。

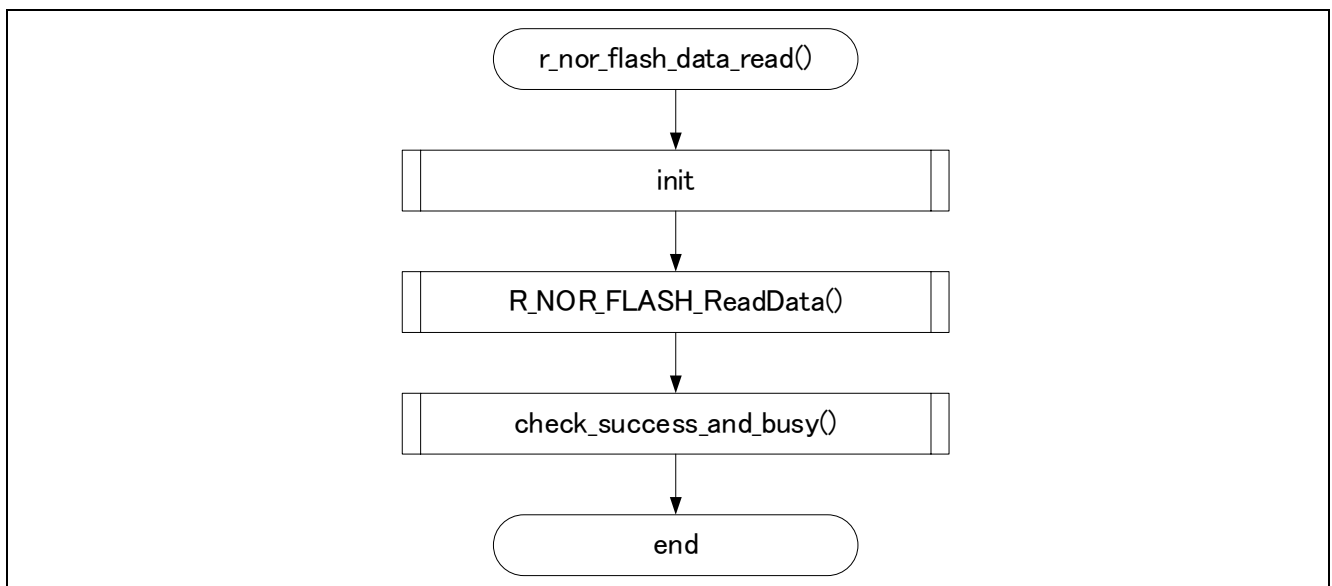
図 4-28 init ()



4.8.9.19 r_nor_flash_data_read()

図 4-29 に `r_nor_flash_data_read()` のフローチャートを示します。

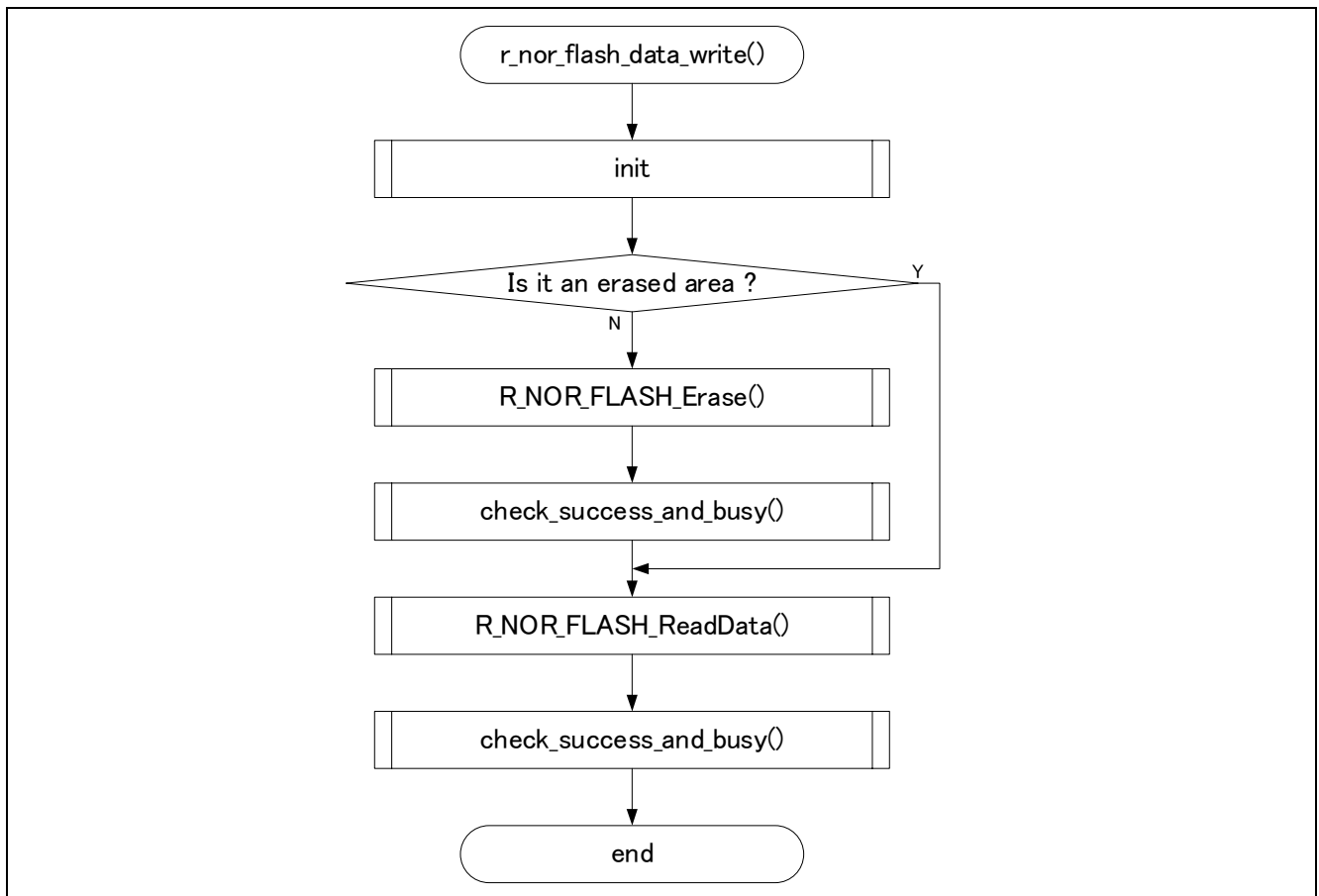
図 4-29 `r_nor_flash_data_read()`



4.8.9.20 r_nor_flash_data_write()

図 4-30 に r_nor_flash_data_write() のフローチャートを示します。

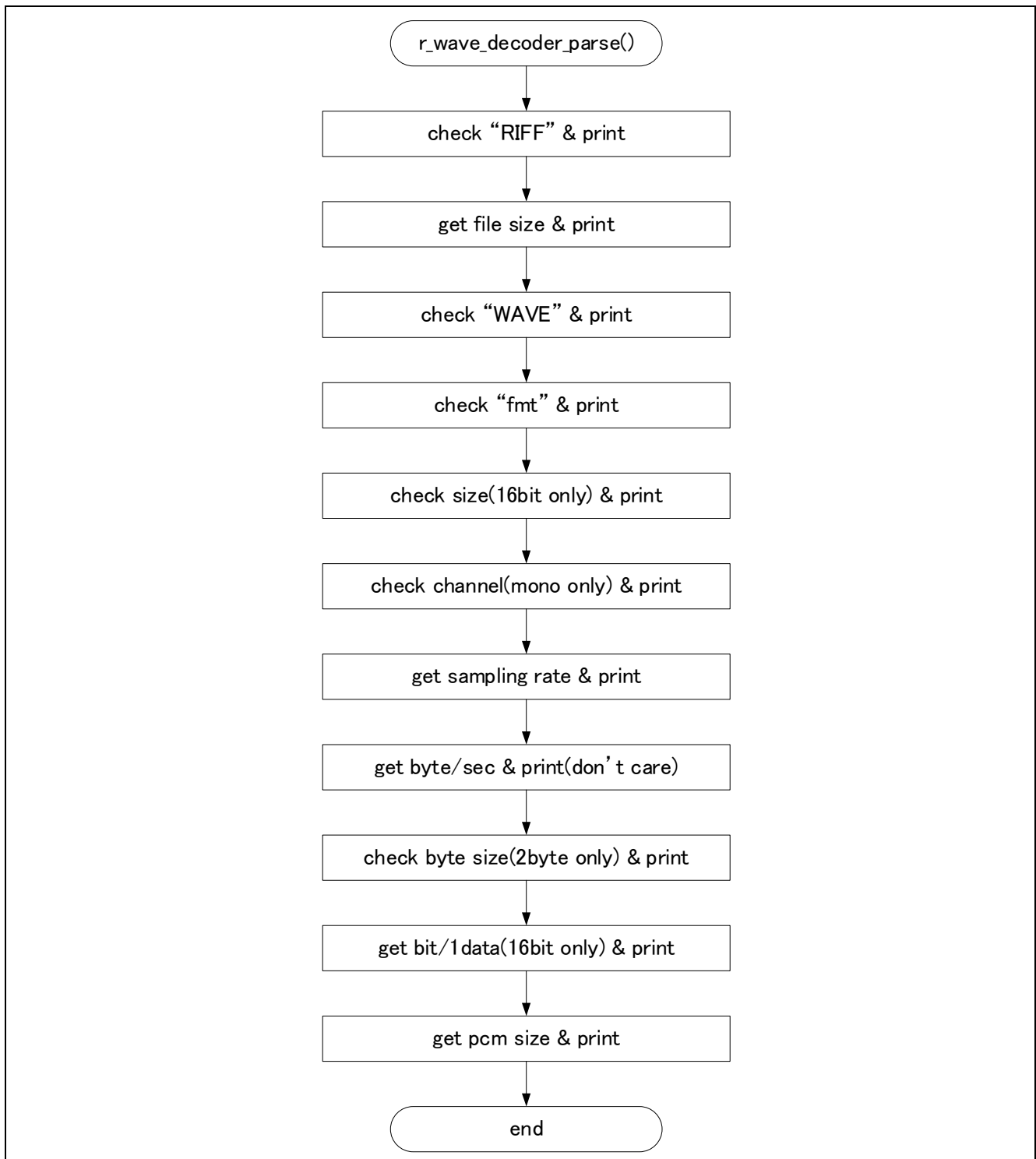
図 4-30 r_nor_flash_data_write()



4.8.9.21 r_wave_decoder_parse()

図 4-31 に r_wave_decoder_parse() のフローチャートを示します。

図 4-31 r_wave_decoder_parse()



4.8.10 r01an7416_adpcm_writer.scfg

サンプルコードで使用しているスマート・コンフィグレータの設定ファイルです。スマート・コンフィグレータで設定されている全ての機能が含まれています。サンプルコードの設定は以下の通りです。

表 4-9 スマート・コンフィグレータの設定値

タグ名	コンポーネント	内容
クロック	-	動作モード：高速メインモード 2.4 (V) ~ 5.5 (V) EV _{DD} 設定：1.8V ≤ EV _{DD0} < 5.5V 高速オンチップ・オシレータ：32MHz f _{IHP} ：32MHz f _{CLK} ：32000kHz（高速オンチップ・オシレータ） f _{SXP} ：32.768kHz（低速オンチップ・オシレータ）
システム	-	オンチップ・デバッグ動作設定：使用しない 疑似 RRM/DMM 機能設定：－ Start/Stop 関数機能設定：－ トレース機能設定：－ セキュリティ ID 設定：セキュリティ ID を設定する セキュリティ ID：0x00000000000000000000 セキュリティ ID 認証失敗時の設定：－
コンポーネント	r_bsp	Start up select：Enable (use BSP startup) Control of invalid memory access detection：Disable RAM guard space (GRAM0-1)：Disabled Guard of control registers of port function (GPORT)：Disabled Guard of registers of interrupt function (GINT)：Disabled Guard of control registers of clock control function, voltage detector, and RAM parity error detection function (GCSC)：Disabled Data flash access control (DFLEN)：Disables Initialization of peripheral functions by Code Generator/Smart Configurator：Enable API functions disable：Enable Parameter check enable：Enable Setting for starting the high-speed on-chip oscillator at the times of release from STOP mode and of transitions to SNOOZE mode：High-speed Enable user warm start callback (PRE)：Unused Enable user warm start callback (POST)：Unused Watchdog Timer refresh enable：Unused
	Config_LVD0	動作モード設定：リセット・モード 電圧検出設定：リセット発生電圧 (V _{LVD0})：1.86 (V)

表 4-10 スマート・コンフィグレータの設定値

タグ名	コンポーネント	内容
コンポーネント	Config_CSI20	転送クロックモード：内部クロック（マスタ） 動作クロック：CK10 クロックソース：f _{CLK} 転送モード：シングル転送モード データビット長設定：8 ビット データ転送方向設定：MSB データ送受信タイミング設定：タイプ 1 ボーレート：1000000bps 通信完了割り込み優先順位：レベル 3（低優先順位） コールバック機能設定：送信完了、受信完了、オーバーランエラー
	Config_UART0	動作クロック：CK00 クロックソース：f _{CLK} /2 動作モード設定：シングル転送モード データビット長設定：8 ビット データ転送方向設定：LSB パリティ設定：パリティ・ビットなし ストップビット長設定：1 ビット 送信データ・レベル設定：非反転（通常） 転送ボーレート設定：115200bps 送信完了割り込み設定（INTST0）：レベル 3（低優先順位） コールバック機能設定：送信完了
	r_nor_flash	Parameter check：use system default ENABLE CHECKING OF THE WEL BIT.：Check WEL bit Select serial flash memory device：Renesas Electronics AT25QF Select serial flash memory capacity：64M bit CS Port Number：PORT1 CS Bit Number：BIT7 Data transfer mode：CPU transfer (Software transfer) SPI(CSI) channel number：Channel 4 DTC Control Data Number：0

4.8.10.1 クロック

サンプルコードで使用するクロックの設定を行います（設定値は表 4-9 を参照してください）。

4.8.10.2 システム

サンプルコードのオンチップ・デバッグ設定を行います（オンチップデバッグは使用しません）。

「オンチップ・デバッグ動作設定」、「セキュリティ ID 認証失敗時の設定」は、「表 5-3 オプション・バイト設定」の「オンチップ・デバッグ動作許可」に影響を与えます。設定を変更する際は注意してください。

4.8.10.3 r_bsp

サンプルコードのスタートアップの設定を行います。

4.8.10.4 Config_LVD0

サンプルコードの電源管理の設定を行います。

「表 5-3 オプション・バイト設定」の「LVD0 の設定」に影響を与えます。設定を変更する際は注意してください。

4.8.10.5 Config_CSI20

NOR FLASH 用 SPI を制御するために使用します。

4.8.10.6 Config_UART0

PC との通信を行うために使用します。

4.8.10.7 r_nor_flash

NOR FLASH 用ミドルウェアです。

5. 音声再生方法

5.1 ソフトウェア説明

本アプリケーションノートに同梱の RL78/G23-64p FPB 用プロジェクト (ra01an7416_adpcm_player) を使用し、音声を再生します。このプロジェクトは、NOR FLASH に書き込まれた ADPCM データを読み出し、デコードした PCM データを、Renesas 製 ARD-AUDIO-DA7212 に送信し、音声を再生します。

ユーザスイッチ (SW1) を押下するたびに、0-9 の音声を順次再生します。

5.2 ハードウェア接続

ハードウェア接続については「図 3-1 ハードウェア構成例」を参照してください。

補足 ra01an7416_adpcm_player の RL78/G23-64p FPB と NOR FLASH の SPI 通信速度は 1MHz としています。

これは、ブレッドボードやジャンパワイヤを使用しての接続における波形訛りを考慮してのことです。

5.3 ターミナルソフト

ra01an7416_adpcm_player では、ターミナルソフトは使用しません。

5.4 デバッガの使用

ra01an7416_adpcm_player では、ターミナルソフトを使用しないためデバッガを使用可能です。

ra01an7416_adpcm_writer と同様、スタンドアロンでも動作します。

5.5 コマンド一覧

コマンドは使用しません。

5.6 操作方法

まず、Renesas Flash Programmer を使用して、本 APN に同梱の ra01an7416_adpcm_player.mot を RL78/G23-64p FPB に書き込みます。

RL78/G23-64p FPB のユーザスイッチ (SW1) を押下するたびに、0-9 の音声を再生します。

5.7 エラー表示

エラー表示機能はありません。

5.8 ソフトウェア説明詳細

5.8.1 動作概要

本サンプルコードでは、Renesas 製 ARD-AUDIO-DA7212 を使用します。また、オーディオ CODEC 内のレジスタ設定に I²C 通信を、サウンド・データ送信処理に I²S 通信を使用します。

BCLK は ELCL の Through コンテンツを使用し、出力端子を P51 に変更しています。

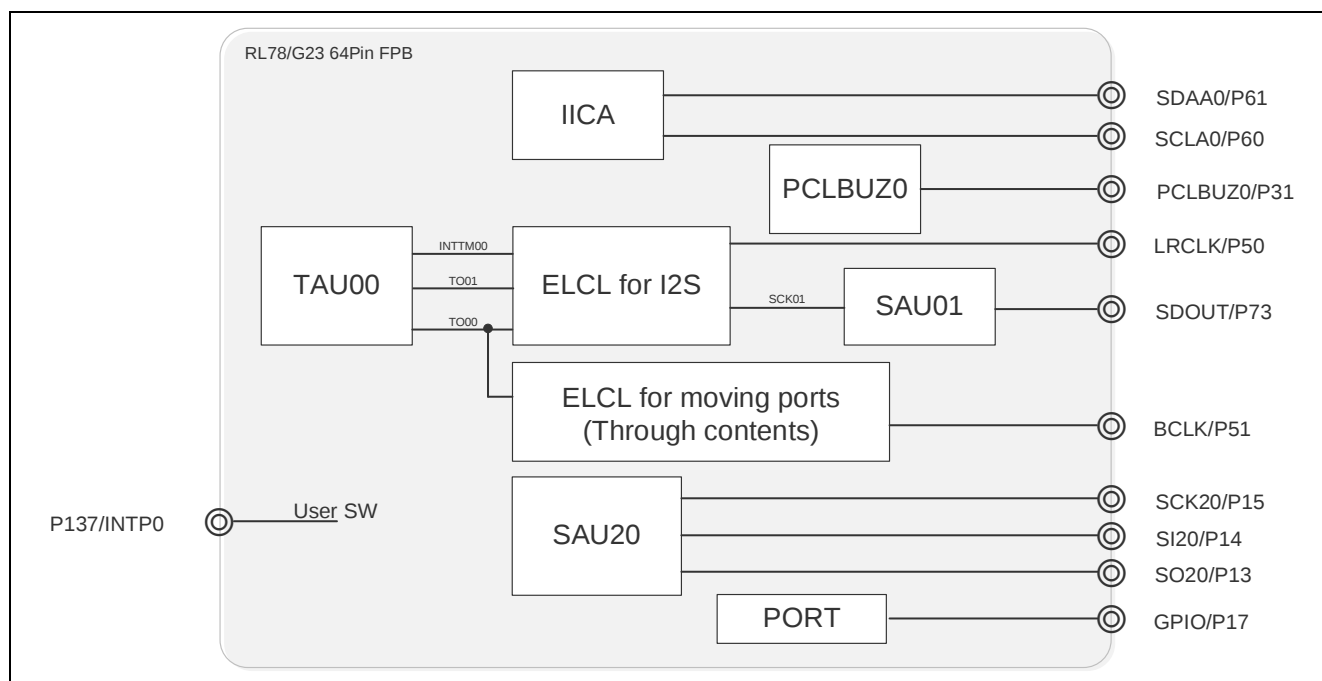
また、Renesas 製 AT25QF641B (NOR FLASH) の読み出し処理に、SPI 通信を使用します。

サンプルコードは以下のような動作をします。

- (1) ユーザスイッチ (SW1) が押下されると I²C 通信用の IICA が動作を開始し ARD-AUDIO-DA7212 のレジスタ設定を行います
 - (2) SPI 通信により、NOR FLASH から ADPCM データを読み出します。
 - (3) I²S 通信用の CSI01 が動作を開始します (SCK01 待ち状態)
 - (4) BCLK、LRCLK 用の TAU0 が動作を開始し、データ送信が開始します
 - (5) CSI01、TAU0 が停止し、(1)に戻ります
- ※(2)は音声再生が完了するまで繰り返し実行されます。

図 5-1 にサンプルコードのシステム構成を示します。

図 5-1 サンプルコードのシステム構成



5.8.2 フォルダ構成

表 5-1、表 5-2 にサンプルコードの使用しているソースファイル／ヘッダファイルの構成を示します。なお、統合開発環境で自動生成されるファイル、bsp 環境のファイルは除きます。

表 5-1 フォルダ構成 (1/2)

フォルダ、ファイル名	説明	スマート・コンフィグレータを使用
r01an7416_adpcm_player<DIR> ^{注3}	サンプルコードのフォルダ	
└ src<DIR>	プログラム格納用フォルダ	
└ └ csi01<DIR>	CSI01 用プログラム格納フォルダ	
└ └ └ csi01.c	CSI01 用ソースファイル	
└ └ └ csi01.h	CSI01 用ヘッダファイル	
└ elcl<DIR>	ELCL 用プログラム格納フォルダ	
└ └ elcl.c	ELCL 用ソースファイル	
└ └ elcl.h	ELCL 用ヘッダファイル	
└ iica0<DIR>	IICA0 用プログラム格納フォルダ	
└ └ iica0.c	IICA0 用ソースファイル	
└ └ iica0.h	IICA0 用ヘッダファイル	
└ s2<DIR>	M3S-S2-Tiny (ADPCM encoder/decoder)	
└ smc_gen<DIR>	スマート・コンフィグレータ生成フォルダ	√
└ └ Config_CSI20	CSI20 用プログラム格納フォルダ	√
└ └ └ Config_CSI20.c	CSI20 用ソースファイル	√
└ └ └ Config_CSI20.h	CSI20 用ヘッダファイル	√
└ └ └ Config_CSI20_user.c	CSI20 用割り込みソースファイル	√
└ └ Config_INTC<DIR>	INTC 用プログラム格納フォルダ	√
└ └ └ Config_INTC.c	INTC 用ソースファイル	√
└ └ └ Config_INTC.h	INTC 用ヘッダファイル	√
└ └ └ Config_INTC_user.c	INTC 用割り込みソースファイル	√
└ └ Config_PCLBUS0<DIR>	PCLBUZ0 用プログラム格納フォルダ	√
└ └ └ Config_PCLBUZ0.c	PCLBUZ0 用ソースファイル	√
└ └ └ Config_PCLBUS0.h	PCLBUZ0 用ヘッダファイル	√
└ └ └ Config_PCLBUS0_user.c	PCLBUZ0 用割り込みソースファイル	√ ^{注2}
└ └ Config_PORT<DIR>	ポート用プログラム格納フォルダ	√
└ └ └ Config_PORT.c	ポート用ソースファイル	√
└ └ └ Config_PORT.h	ポート用ヘッダファイル	√
└ └ └ Config_PORT_user.c	ポート用割り込みソースファイル	√ ^{注1}

補足 ” <DIR> ” は、ディレクトリを意味します。

注1. 本サンプルコードでは使用しません。

注2. スマート・コンフィグレータで生成したファイルに初期設定を追加しています。

注3. IAR 版のサンプルコードは、buildinfo.ipcf を格納しています。

ipcf ファイルについては、「RL78 スマート・コンフィグレータ ユーザーガイド : IAR 編 (R20AN0581)」を確認してください。

表 5-2 フォルダ構成 (2/2)

フォルダ、ファイル名	説明	スマート・コンフィグレータを使用
└└ Config_TAU0_1<DIR>	TAU00 用プログラム格納フォルダ	√
└└ └ Config_TAU0_0.c	TAU00 用ソースファイル	√
└└ └ Config_TAU0_0.h	TAU00 用ヘッダファイル	√
└└ └└ Config_TAU0_0_user.c	TAU01 用割り込みソースファイル	√ ^{注1}
└└ Config_TAU0_1<DIR>	TAU01 用プログラム格納フォルダ	√
└└ └ Config_TAU0_1.c	TAU01 用ソースファイル	√
└└ └ Config_TAU0_1.h	TAU01 用ヘッダファイル	√
└└ └└ Config_TAU0_1_user.c	TAU01 用割り込みソースファイル	√ ^{注1}
└└ Config_TAU0_7<DIR>	TAU07 用プログラム格納フォルダ	√
└└ └ Config_TAU0_7.c	TAU07 用ソースファイル	√
└└ └ Config_TAU0_7.h	TAU07 用ヘッダファイル	√
└└ └└ Config_TAU0_7_user.c	TAU07 用割り込みソースファイル	√ ^{注4}
└└ Config_Through<DIR>	Through 用プログラム格納フォルダ ^{注5}	√
└└ └ Config_Through.c	Through 用ソースファイル	√
└└ └ Config_Through.h	Through 用ヘッダファイル	√
└└ └└ Config_Through_user.c	Through 用割り込みソースファイル	√
└└ general<DIR>	初期化、共通プログラム格納フォルダ	
└└ r_bsp<DIR>	BSP 用プログラム格納フォルダ	
└└ r_config<DIR>	プログラム格納フォルダ	
└└ r_nor_flash_rl78<DIR>	NOR FLASH ライブラリ格納フォルダ ^{注6}	
└└ r_pincfg<DIR>	pincfg 格納フォルダ	
└ main.c	メインプログラム	
└ main.h	ヘッダファイル	
└ r_flash_adpcm_player.c	ADPCM 再生メインプログラム	
└ r_flash_adpcm_player.h	ヘッダファイル	
└ r_codec.c	DA7212 設定	
└ r_codec.h	ヘッダファイル	

補足 ”<DIR>” は、ディレクトリを意味します。

注 4. スマート・コンフィグレータで生成したファイルに割り込み処理ルーチンを追加しています。

注 5. Smart Configurator のコンポーネント追加で、「ELCL モジュールをダウンロードする」から ELCL Through をダウンロードしてください。

注 6. Smart Configurator のコンポーネント追加で、RL78 Family Serial NOR Flash Memory 制御モジュール Software Integartion System モジュールをダウンロードしてください。

I²S 通信は CSI を使用しますが、本来の CSI の規格に沿わず、オーバーランエラーが発生します。スマート・コンフィグレータで生成したコードからエラータイプの判定(*1)と、未使用関数を削除しています。

また、IICA は、スマート・コンフィグレータで生成されたコードから不要な関数を削除しています(*2)。

*1 : err_type = (uint8_t)(SSR01 & _0001_SAU_OVERRUN_ERROR);
if (1U != err_type) を削除
Config_CSI01_User.c に関連する関数

*2 : Config_IICA0_User.c に関連する関数を削除しています。

5.8.3 オプション・バイトの設定一覧

表 5-3 にオプション・バイト設定を示します。

表 5-3 オプション・バイト設定

アドレス	設定値	内容
000C0H/040C0H	1110 1111B (EFH)	ウォッチドッグ・タイマ動作停止 (リセット解除後、カウント停止)
000C1H/040C1H	1111 1110B (FEH)	LVD0 リセット・モード 検出電圧：立ち上がり 1.90V／立下り 1.86V
000C2H/040C2H	1110 1000B (E8H)	フラッシュ動作モード：高速メインモード 高速オンチップ・オシレータの周波数：32MHz
000C3H/040C3H	1000 0101B (85H)	オンチップ・デバッグ動作許可

5.8.4 ROM/RAM サイズ

表 4-5 に本サンプルコードで使用する ROM/RAM サイズを記載します（CC-RL 最適化レベル＝一部の最適化(-O-lite)でのサイズです。最適化レベルで変動します）。

表 5-4 ROM/RAM サイズ

サンプルプロジェクト名	ROM サイズ	RAM サイズ
ra01an7416_adpcm_player	9019Byte	919Byte

5.8.5 定数一覧

表 5-5 にサンプルコードで使用する定数を示します。

表 5-5 サンプルコードで使用する定数

定数名	設定値	内容	ファイル
WAIT_TIME	100	IICA0 通信待ち時間	r_codec.h
SENSOR_ADD	0x34	オーディオ CODEC IC のアドレス	r_codec.h

5.8.6 変数一覧

表 5-6 に本サンプルコードで使用するグローバル変数を示します。

表 5-6 サンプルコードで使用するグローバル変数

型	変数名	内容	使用関数
st_sound_buf_t	gp_rcv_sb	ADPCM 受信構造体へのポインタ	rcv_sb_status_clear status_init check_rcv_sb adpcm2pcm check_dec_sb play_start
st_intp0_t	g_intp0	ユーザスイッチ (SW1) 割り込み関連構造体	r_Config_INTC_intp0_interrupt wait_sw play_start r_nor_flash_operation
volatile uint16_t	g_ms_timer	ウェイト処理のカウント値	r_ms_delay r_Config_TAU0_7_interrupt
volatile uint8_t	g_tx_done_flag	送信完了フラグ	r_nor_flash_operation r_csi01_callback_sendend
volatile uint8_t	g_sample_mode	I ² C 通信ステータス	r_codec_init send_i2c r_codec_stop r_iica0_callback_master_sendend

5.8.7 関数一覧

表 5-7 にサンプルコードで使用する関数を示します。ただし、スマート・コンフィグレータで生成された関数の内、変更を行っていないものは除きます。

また、「[RL78/G23 ELCL、SPI による I²S 通信 アプリケーションノート](#)」から変更していない関数については、本アプリケーションノートには記載していません。「[RL78/G23 ELCL、SPI による I²S 通信 アプリケーションノート](#)」を参照してください。

表 5-7 関数一覧

	関数名	概要	ソースファイル
g	main	メイン処理	main.c
g	r_adpcm_player_operation	adpcm_player のメイン処理	r_adpcm_player.c
s	init	adpcm_player の初期化处理	r_adpcm_player.c
s	wait_sw	ユーザスイッチ (SW1) 押下判定	r_adpcm_player.c
s	play_start	adpcm 再生開始	r_adpcm_player.c
s	check_recv_sb	受信バッファ判定	r_adpcm_player.c
s	check_dec_sb	adpcm デコードバッファ判定	r_adpcm_player.c
s	status_init	adpcm_player のステータス初期化	r_adpcm_player.c
s	recv_sb_status_clear	受信関連構造体状態クリア	r_adpcm_player.c
s	dec_sb_status_clear	デコード関連構造体状態クリア	r_adpcm_player.c
s	i2s_sb_status_clear	再生関連構造体状態クリア	r_adpcm_player.c
s	get_i2s_data	I ² S 転送データ取得	r_adpcm_player.c
s	adpcm2pcm	ADPCM デコード	r_adpcm_player.c
s	change_i2s_buf	再生構造体変更	r_adpcm_player.c
g	r_codec_init	I ² C のオープン処理	r_codec.c
s	send_i2c	I ² C 送信	r_codec.c
g	r_codec_start	DA7212 初期設定	r_codec.c
g	r_codec_change_sampling_rate	DA7212 再生レート変更	r_codec.c
g	r_codec_stop	I ² C 停止確認	r_codec.c

注 1. 先頭の「g」はグローバル関数、「s」はスタティック関数を示します。

5.8.8 関数仕様

サンプルコードの関数仕様を示します。

なお、「[RL78/G23 ELCL、SPI による I²S 通信 アプリケーションノート](#)」から変更していない関数については、本アプリケーションノートには記載していません。「[RL78/G23 ELCL、SPI による I²S 通信 アプリケーションノート](#)」を参照してください。

[関数名] main

概要	メイン処理
ヘッダ	stdint.h, stdbool.h, main.h, r_adpcm_player.h
宣言	void main (void);
説明	main 関数です。r_adpcm_player_operation をコールします。
引数	なし
リターン値	なし
備考	なし

[関数名] r_adpcm_player_operation(r_adpcm_player.c)

概要	adpcm_player のメイン処理
ヘッダ	stdint.h, stdbool.h, r_smc_entry.h, csi01.h, elcl.h, r_adpcm.h, r_adpcm_player.h, r_nor_flash_rl78_if, r_codec.h
宣言	void r_adpcm_player_operation(void);
説明	adpcm_player のメイン処理を行います
引数	なし
リターン値	なし
備考	なし

[関数名] init(r_adpcm_player.c)

概要	adpcm_player の初期化処理
ヘッダ	stdint.h, stdbool.h, r_smc_entry.h, csi01.h, elcl.h, r_adpcm.h, r_adpcm_player.h, r_nor_flash_rl78_if, r_codec.h
宣言	static void init(void);
説明	adpcm_player の初期化処理を行います
引数	なし
リターン値	なし
備考	なし

[関数名] wait_sw(r_adpcm_player.c)

概要	ユーザスイッチ (SW1) 判定処理
ヘッダ	stdint.h, stdbool.h, r_smc_entry.h, csi01.h, elcl.h, r_adpcm.h, r_adpcm_player.h, r_nor_flash_rl78_if, r_codec.h
宣言	static void wait_sw(void);
説明	ユーザスイッチ (SW1) 判定処理を行います
引数	なし
リターン値	なし
備考	なし

[関数名] play_start(r_adpcm_player.c)

概要	再生開始処理
ヘッダ	stdint.h, stdbool.h, r_smc_entry.h, csi01.h, elcl.h, r_adpcm.h, r_adpcm_player.h, r_nor_flash_rl78_if, r_codec.h
宣言	static void play_start(void);
説明	ユーザスイッチ（SW1）押した回数から NOR FLASH に格納されたデータを識別し、再生開始処理を行います。データサイズ識別や、DA7212 へのサンプリングレート設定などもこの関数で行います。
引数	なし
リターン値	なし
備考	なし

[関数名] check_rcv_sb (r_adpcm_player.c)

概要	ADPCM 受信処理
ヘッダ	stdint.h, stdbool.h, r_smc_entry.h, csi01.h, elcl.h, r_adpcm.h, r_adpcm_player.h, r_nor_flash_rl78_if, r_codec.h
宣言	static void check_rcv_sb (void);
説明	ADPCM 受信バッファが空いていれば NOR FLASH からの読み出し処理を行います。
引数	なし
リターン値	なし
備考	なし

[関数名] check_dec_sb (r_adpcm_player.c)

概要	ADPCM デコード処理
ヘッダ	stdint.h, stdbool.h, r_smc_entry.h, csi01.h, elcl.h, r_adpcm.h, r_adpcm_player.h, r_nor_flash_rl78_if, r_codec.h
宣言	static void check_dec_sb (void);
説明	ADPCM 受信完了してれば、ADPCM デコード処理を行います。
引数	なし
リターン値	なし
備考	なし

[関数名] status_init (r_adpcm_player.c)

概要	r_adpcm_player 内のステータス初期化
ヘッダ	stdint.h, stdbool.h, r_smc_entry.h, csi01.h, elcl.h, r_adpcm.h, r_adpcm_player.h, r_nor_flash_rl78_if, r_codec.h
宣言	static void status_init (void);
説明	r_adpcm_player 内のステータス初期化処理を行います。
引数	なし
リターン値	なし
備考	なし

[関数名] `recv_sb_status_clear (r_adpcm_player.c)`

概要	受信関連構造体状態クリア
ヘッダ	<code>stdint.h</code> , <code>stdbool.h</code> , <code>r_smc_entry.h</code> , <code>csi01.h</code> , <code>elcl.h</code> , <code>r_adpcm.h</code> , <code>r_adpcm_player.h</code> , <code>r_nor_flash_rl78_if</code> , <code>r_codec.h</code>
宣言	<code>static void recv_sb_status_clear (void);</code>
説明	受信関連構造体の状態を初期化します。
引数	なし
リターン値	なし
備考	なし

[関数名] `dec_sb_status_clear (r_adpcm_player.c)`

概要	デコード関連構造体状態クリア
ヘッダ	<code>stdint.h</code> , <code>stdbool.h</code> , <code>r_smc_entry.h</code> , <code>csi01.h</code> , <code>elcl.h</code> , <code>r_adpcm.h</code> , <code>r_adpcm_player.h</code> , <code>r_nor_flash_rl78_if</code> , <code>r_codec.h</code>
宣言	<code>static void dec_sb_status_clear (void);</code>
説明	ADPCM デコード関連構造体の状態を初期化します。
引数	なし
リターン値	なし
備考	なし

[関数名] `i2s_sb_status_clear (r_adpcm_player.c)`

概要	I ² S 関連構造体状態クリア
ヘッダ	<code>stdint.h</code> , <code>stdbool.h</code> , <code>r_smc_entry.h</code> , <code>csi01.h</code> , <code>elcl.h</code> , <code>r_adpcm.h</code> , <code>r_adpcm_player.h</code> , <code>r_nor_flash_rl78_if</code> , <code>r_codec.h</code>
宣言	<code>static void i2s_sb_status_clear (void);</code>
説明	I ² S 関連構造体の状態を初期化します。
引数	なし
リターン値	なし
備考	なし

[関数名] `get_i2s_data (r_adpcm_player.c)`

概要	I ² S 転送データ取得
ヘッダ	<code>stdint.h</code> , <code>stdbool.h</code> , <code>r_smc_entry.h</code> , <code>csi01.h</code> , <code>elcl.h</code> , <code>r_adpcm.h</code> , <code>r_adpcm_player.h</code> , <code>r_nor_flash_rl78_if</code> , <code>r_codec.h</code>
宣言	<code>static void get_i2s_data (void);</code>
説明	I ² S に送信するデータを取得する関数です。
引数	なし
リターン値	なし
備考	なし

[関数名] adpcm2pcm (r_adpcm_player.c)

概要	ADPCM デコード
ヘッダ	stdint.h, stdbool.h, r_smc_entry.h, csi01.h, elcl.h, r_adpcm.h, r_adpcm_player.h, r_nor_flash_rl78_if, r_codec.h
宣言	static void adpcm2pcm (void);
説明	ADPCM デコードを行います。
引数	なし
リターン値	なし
備考	なし

[関数名] change_i2s_buf (r_adpcm_player.c)

概要	再生構造体変更
ヘッダ	stdint.h, stdbool.h, r_smc_entry.h, csi01.h, elcl.h, r_adpcm.h, r_adpcm_player.h, r_nor_flash_rl78_if, r_codec.h
宣言	static void change_i2s_buf (void);
説明	I ² S 再生バッファを ADPCM デコード済みバッファと差し替えます。
引数	なし
リターン値	なし
備考	なし

[関数名] r_codec_init (r_codec.c)

概要	I ² C のオープン処理
ヘッダ	stdint.h, stdbool.h, iica0.h, r_codec.h, Config_TAU0_7.h
宣言	void change_i2s_buf (void);
説明	I ² C のオープン処理を行います。
引数	なし
リターン値	なし
備考	なし

[関数名] send_i2c (r_codec.c)

概要	I ² C 送信
ヘッダ	stdint.h, stdbool.h, iica0.h, r_codec.h, Config_TAU0_7.h
宣言	static void send_i2c (void);
説明	I ² C の送信関数です。
引数	なし
リターン値	なし
備考	なし

[関数名] r_codec_start (r_codec.c)

概要	DA7212 の初期設定
ヘッダ	stdint.h, stdbool.h, iica0.h, r_codec.h, Config_TAU0_7.h
宣言	static void r_codec_start (void);
説明	DA7212 の初期設定を行います。
引数	なし
リターン値	なし
備考	なし

[関数名] r_codec_change_sampling_rate (r_codec.c)

概要	DA7212 再生レート変更
ヘッダ	stdint.h, stdbool.h, iica0.h, r_codec.h, Config_TAU0_7.h
宣言	static void r_codec_change_sampling_rate (void);
説明	DA7212 のサンプリングレートを変更します。
引数	なし
リターン値	なし
備考	なし

[関数名] r_codec_stop (r_codec.c)

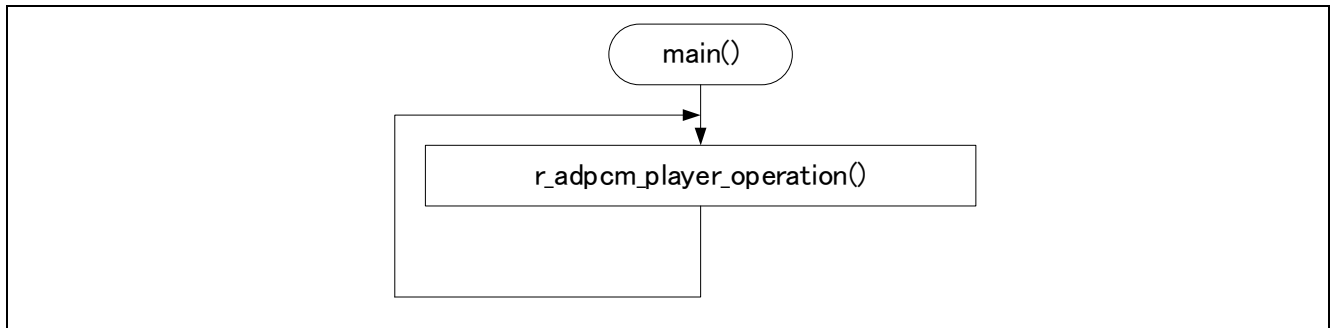
概要	I ² C 停止確認
ヘッダ	stdint.h, stdbool.h, iica0.h, r_codec.h, Config_TAU0_7.h
宣言	static void r_codec_stop (void);
説明	I ² C の停止確認と、100ms の終了待ちを行います。
引数	なし
リターン値	なし
備考	なし

5.8.9 フローチャート

5.8.9.1 main()

図 5-2 にメイン処理のフローチャートを示します。

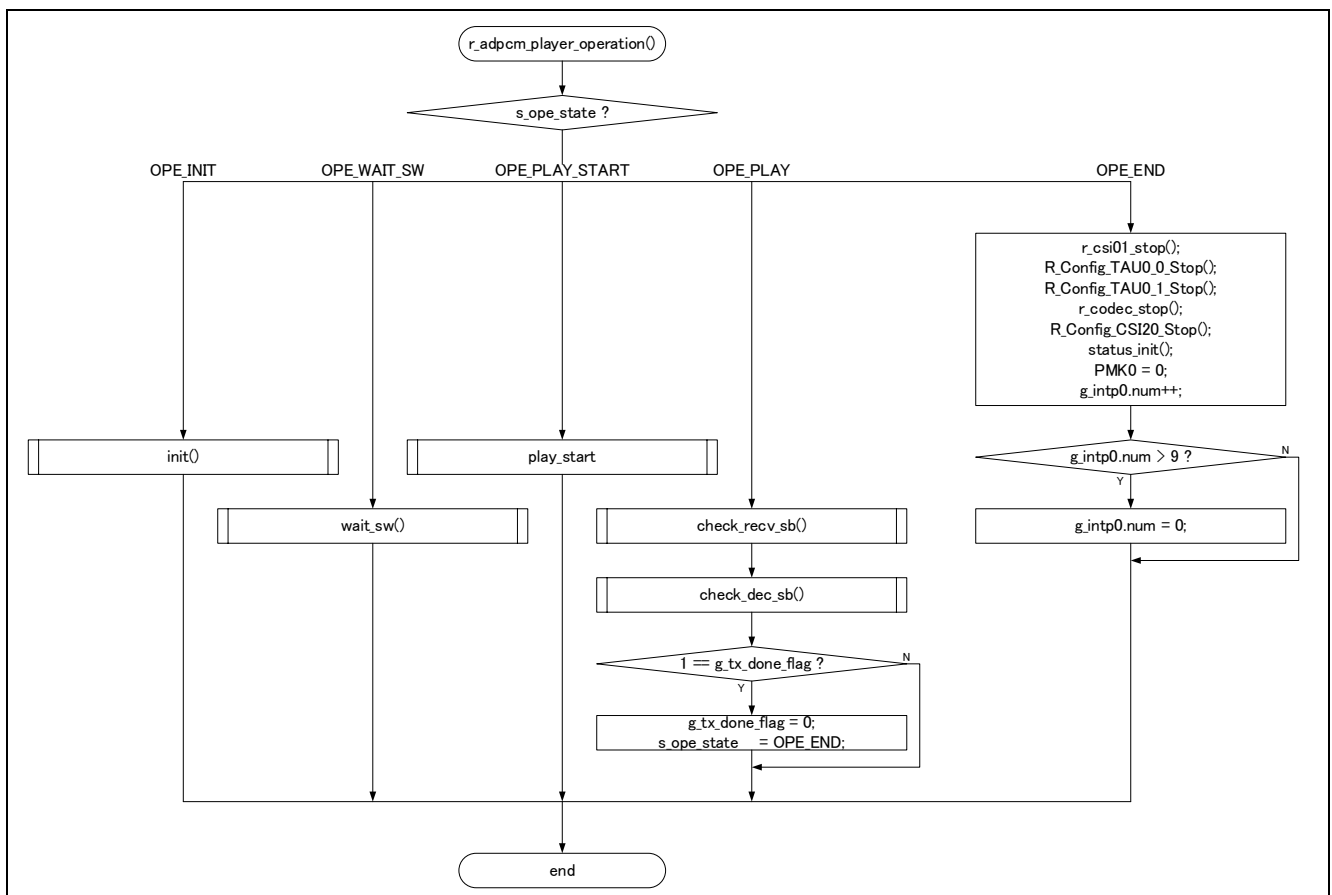
図 5-2 メイン処理



5.8.9.2 r_adpcm_player_operation ()

図 5-3 に `r_adpcm_player_operation()` のフローチャートを示します。

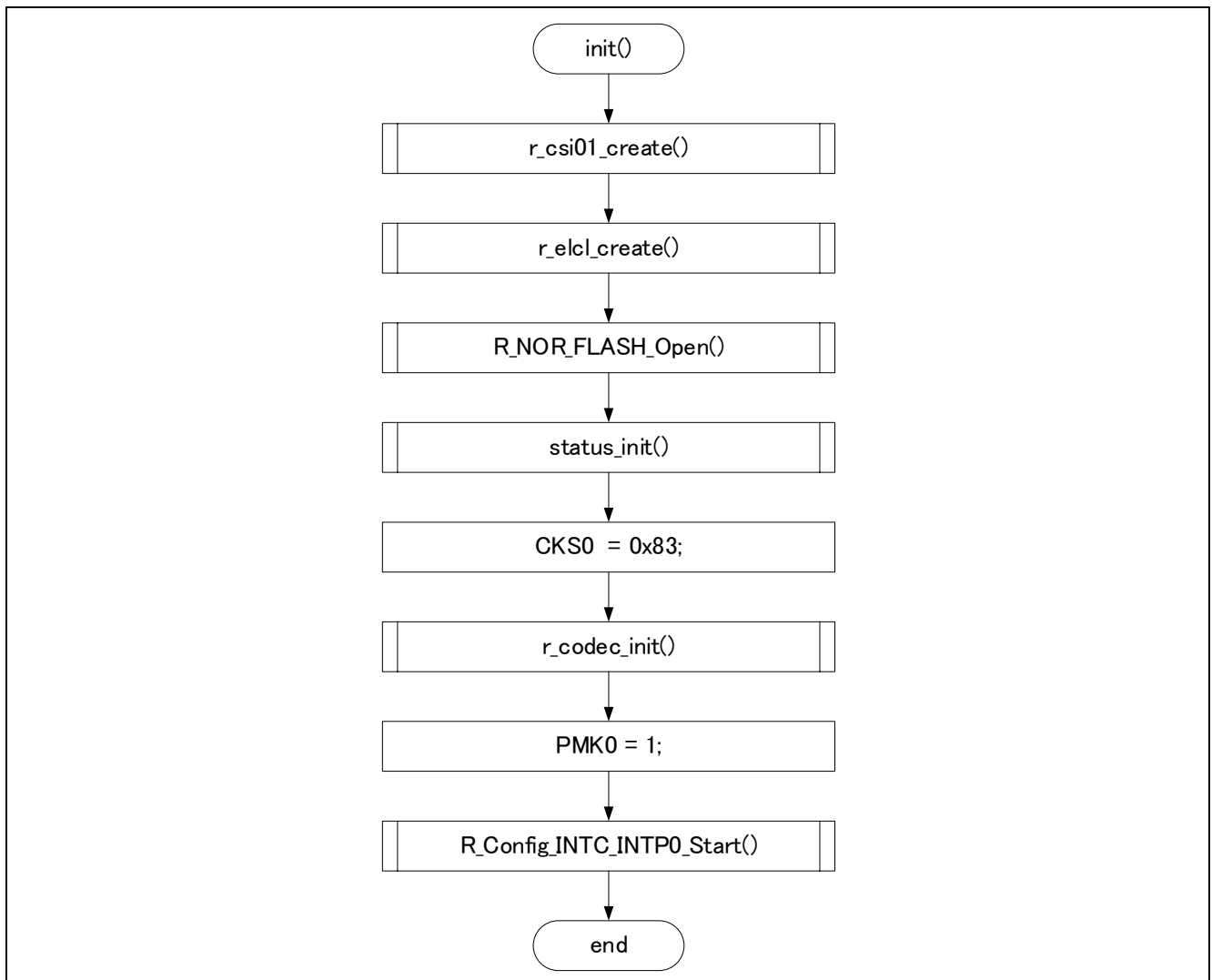
図 5-3 r_adpcm_player_operation ()



5.8.9.3 init ()

図 5-4 に init () のフローチャートを示します。

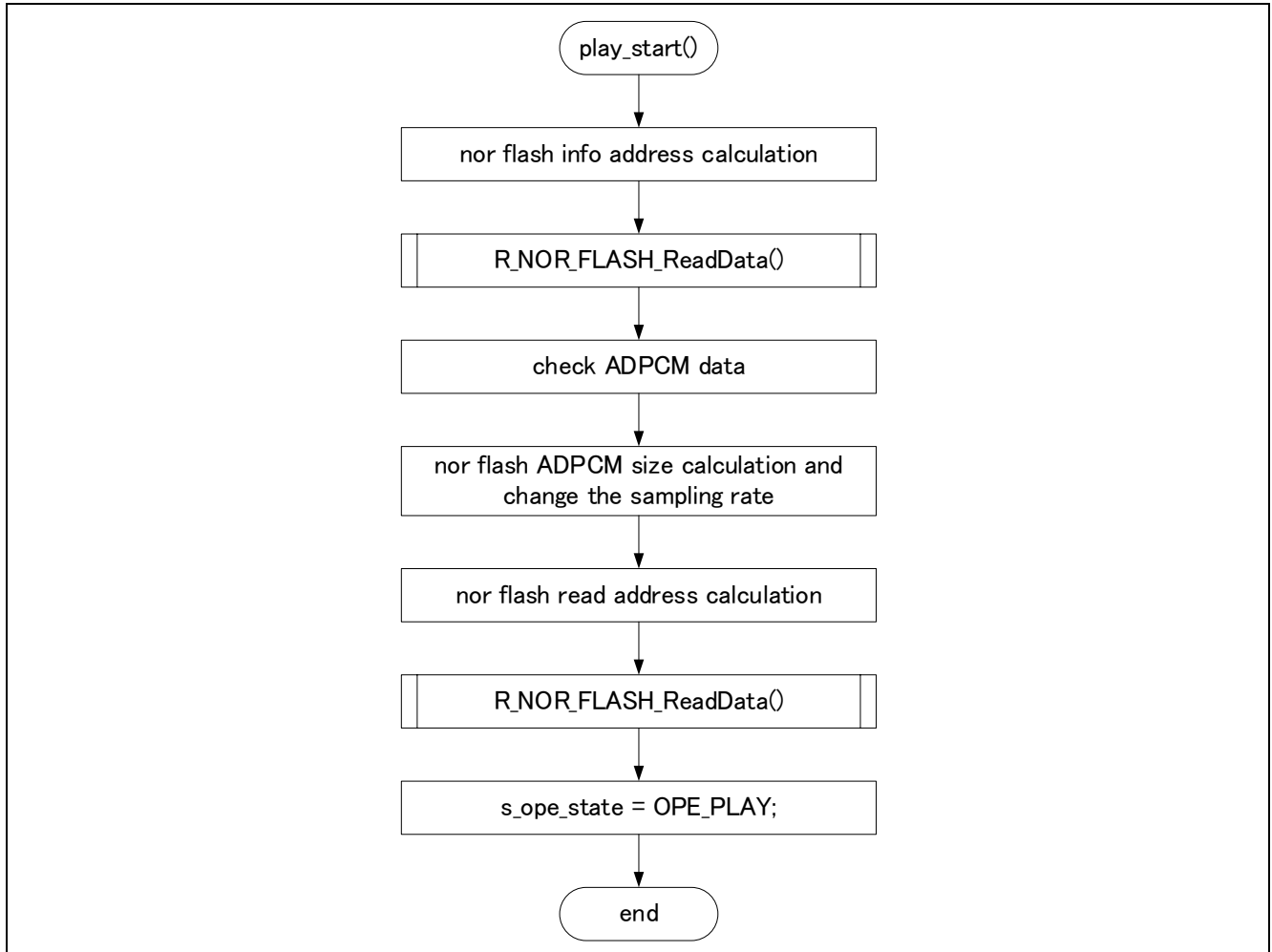
図 5-4 init ()



5.8.9.4 play_start ()

図 5-5 に play_start () のフローチャートを示します。

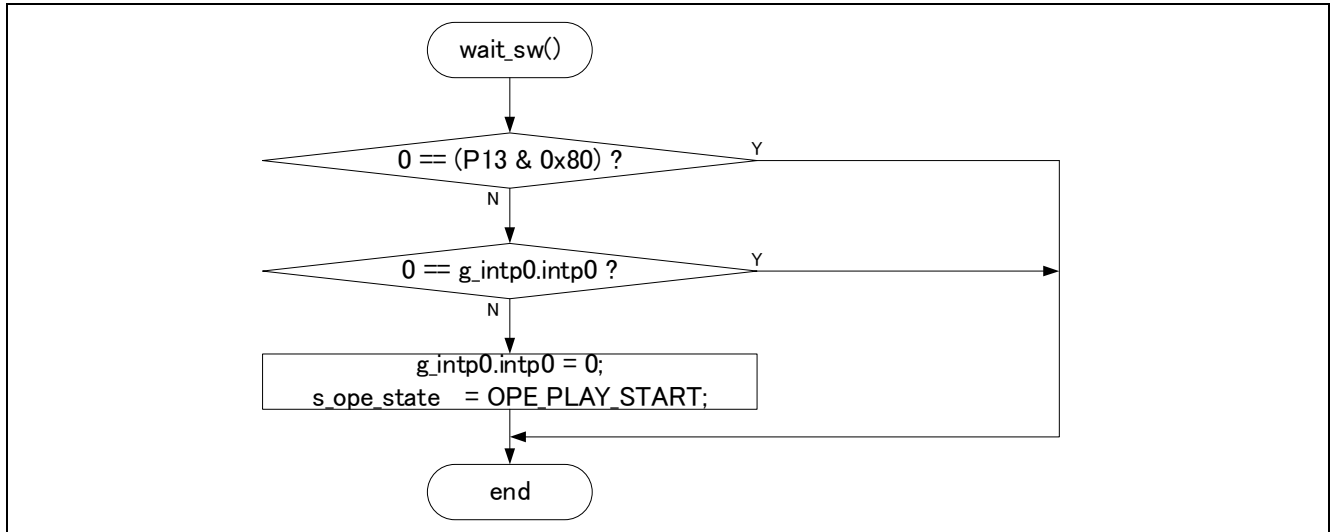
図 5-5 play_start ()



5.8.9.5 wait_sw ()

図 5-6 に wait_sw () のフローチャートを示します。

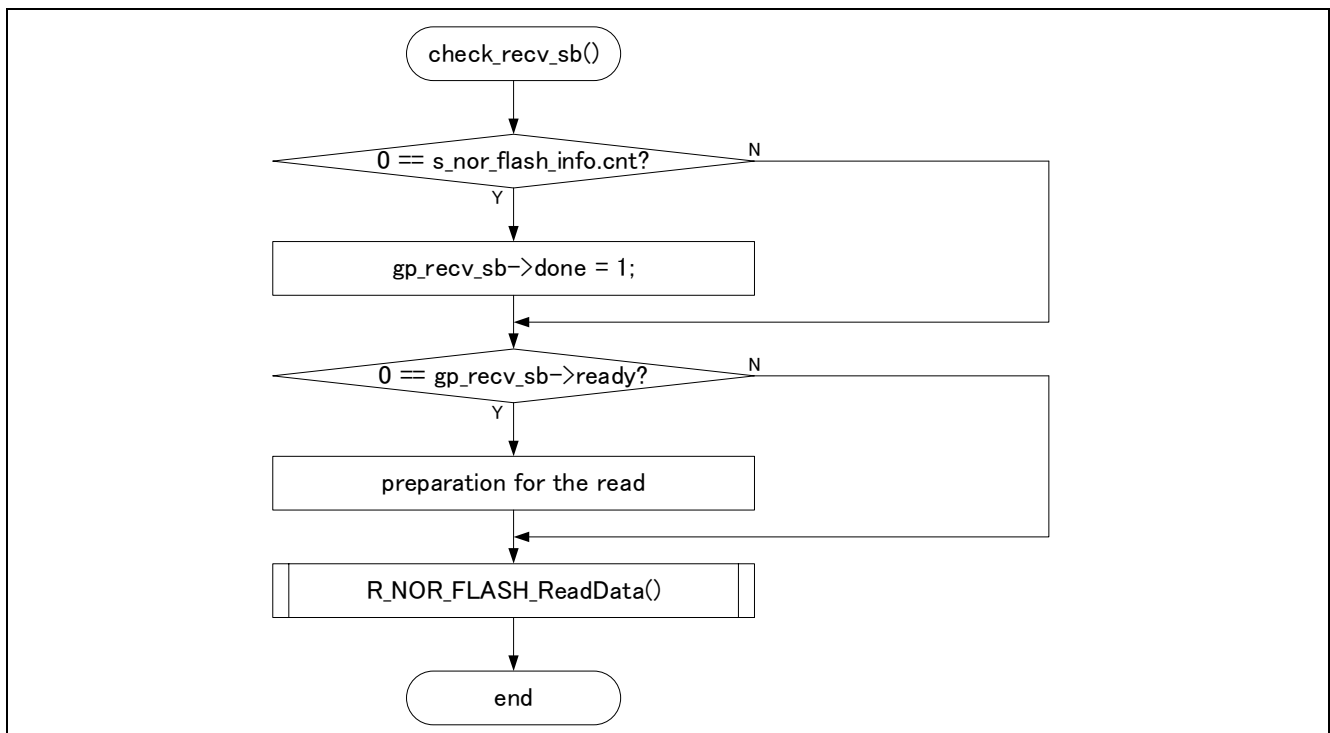
図 5-6 wait_sw ()



5.8.9.6 check_rcv_sb ()

図 5-7 に check_rcv_sb () のフローチャートを示します。

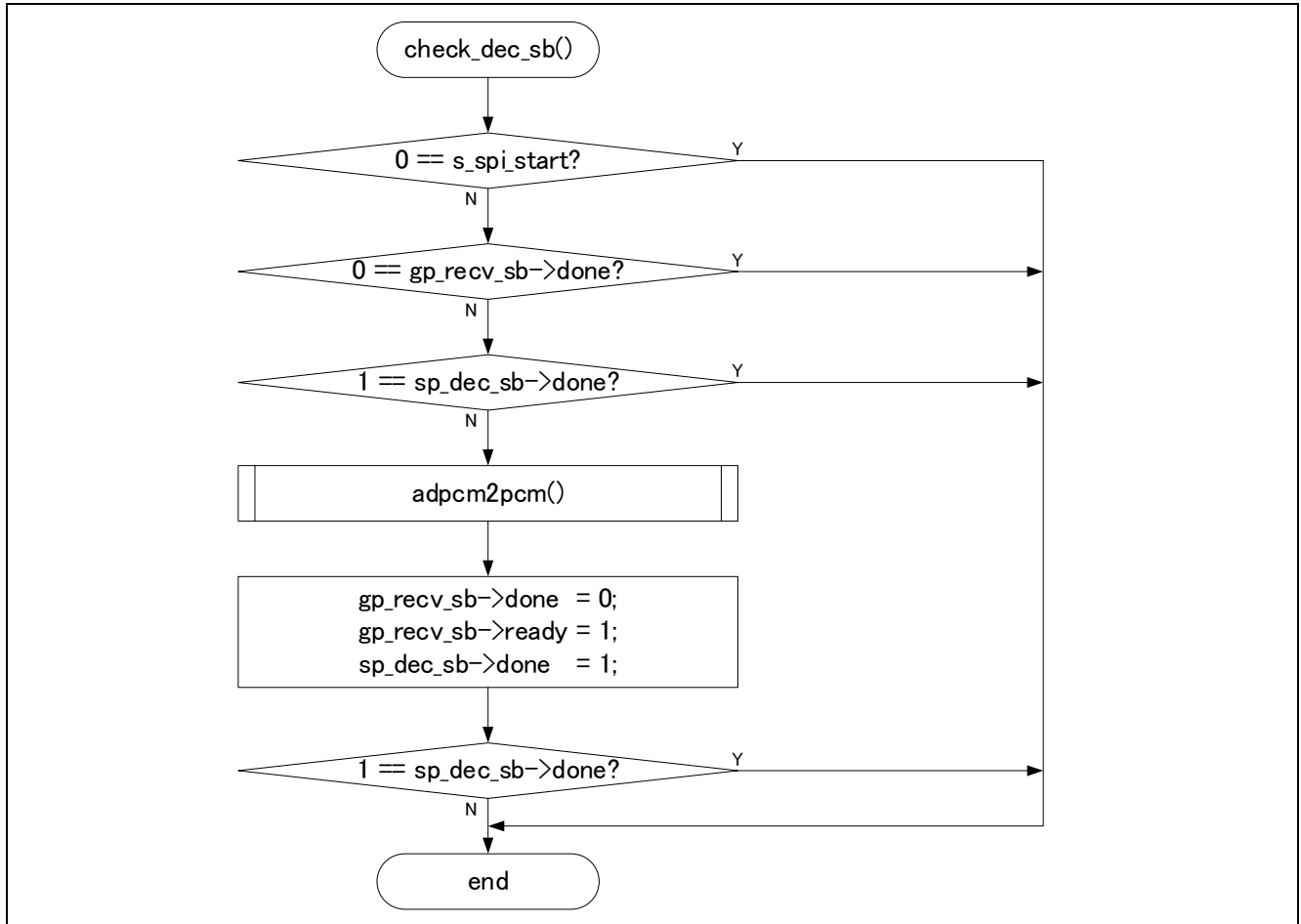
図 5-7 check_rcv_sb ()



5.8.9.7 check_dec_sb ()

図 5-8 に check_dec_sb () のフローチャートを示します。

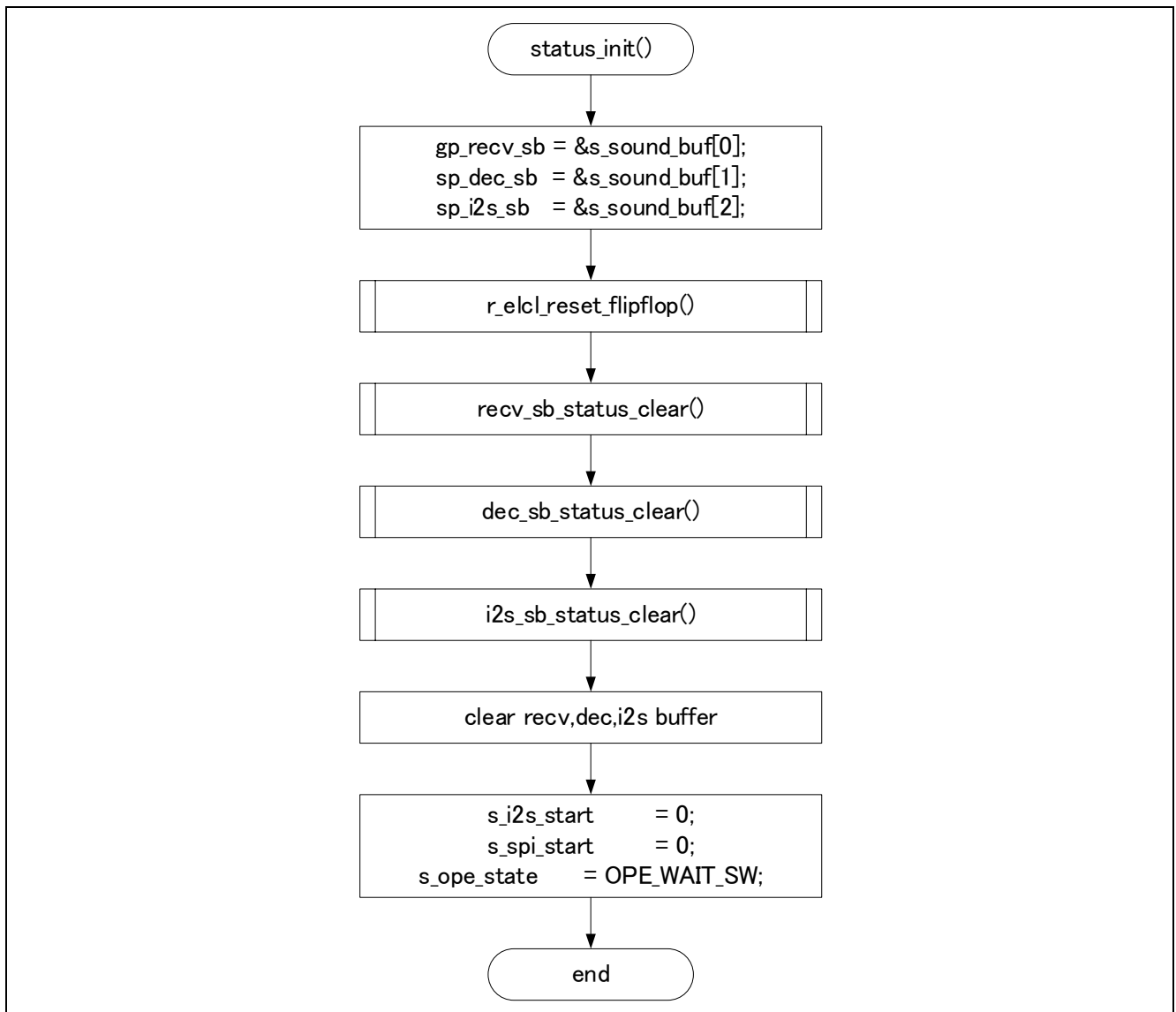
図 5-8 check_dec_sb ()



5.8.9.8 status_init ()

図 5-9 に status_init () のフローチャートを示します。

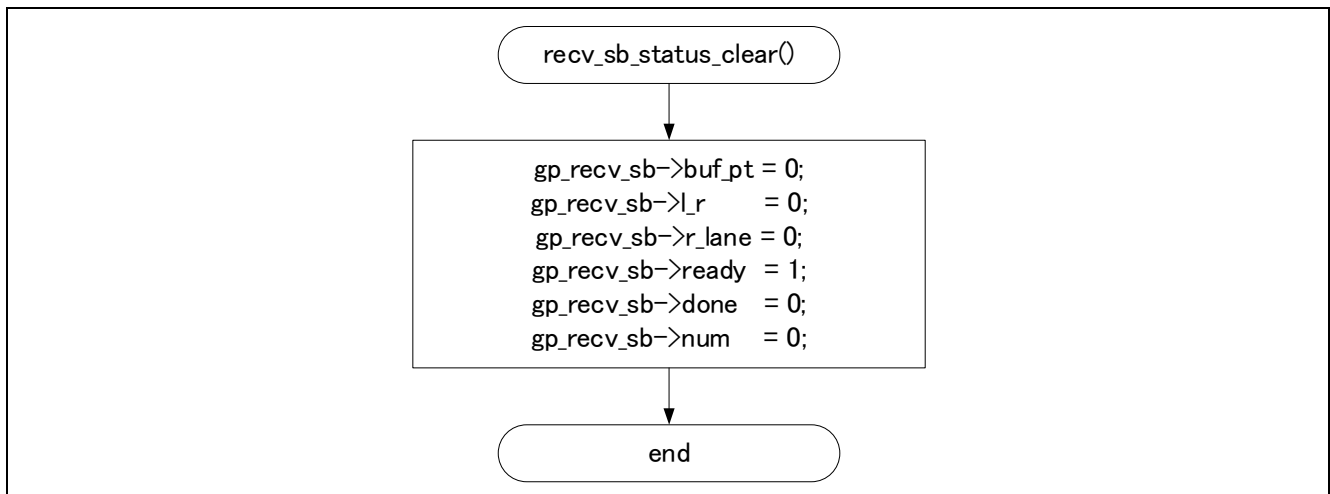
図 5-9 status_init ()



5.8.9.9 recv_sb_status_clear ()

図 5-10 に recv_sb_status_clear () のフローチャートを示します。

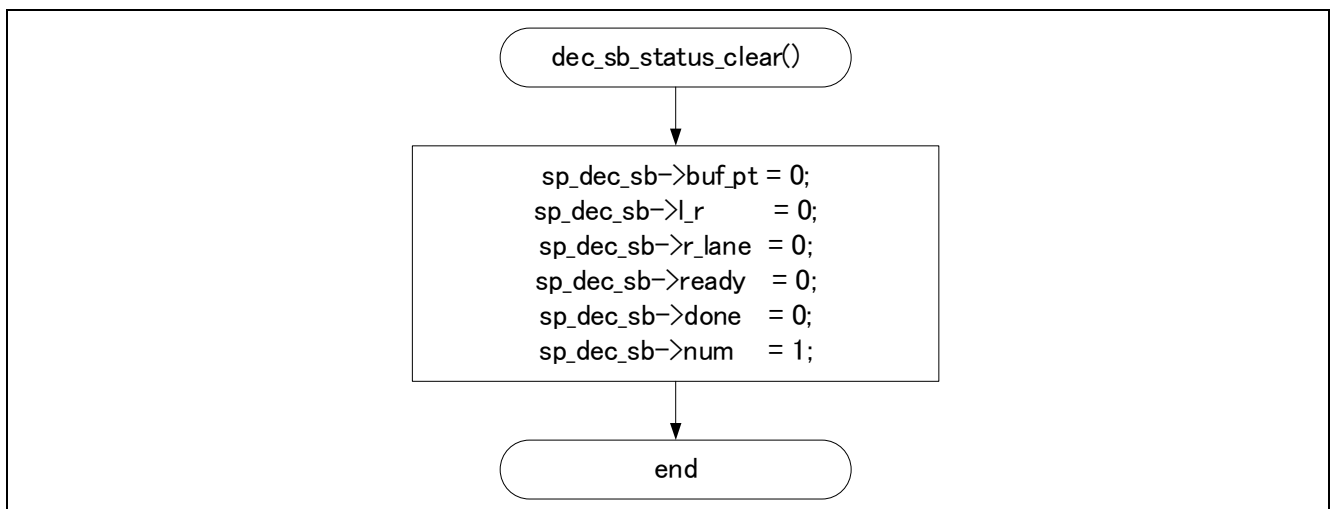
図 5-10 recv_sb_status_clear ()



5.8.9.10 dec_sb_status_clear ()

図 5-11 に dec_sb_status_clear () のフローチャートを示します。

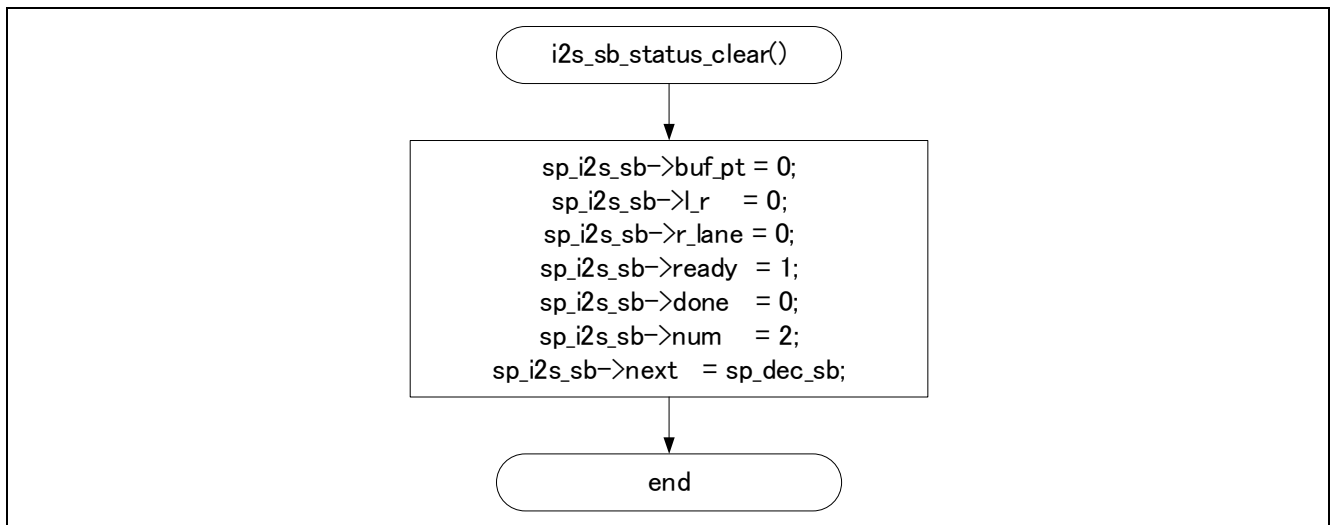
図 5-11 dec_sb_status_clear ()



5.8.9.11 i2s_sb_status_clear ()

図 5-12 に i2s_sb_status_clear () のフローチャートを示します。

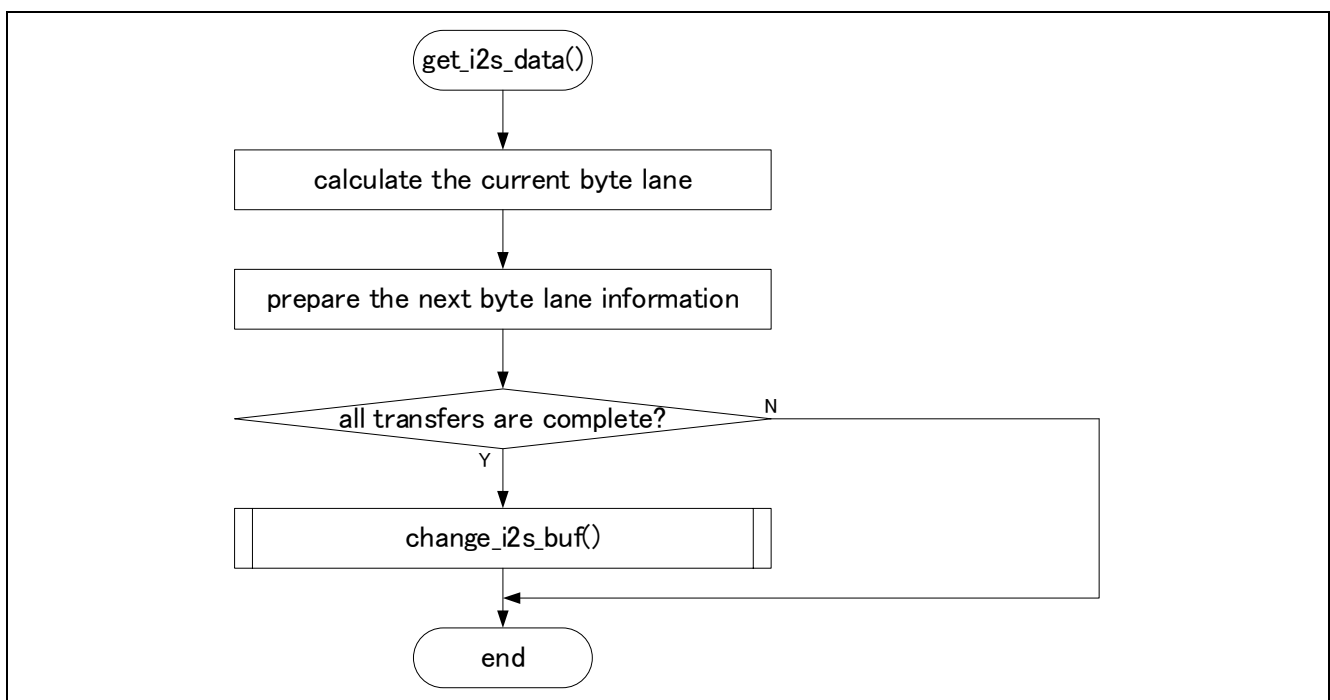
図 5-12 i2s_sb_status_clear ()



5.8.9.12 get_i2s_data ()

図 5-13 に get_i2s_data () のフローチャートを示します。

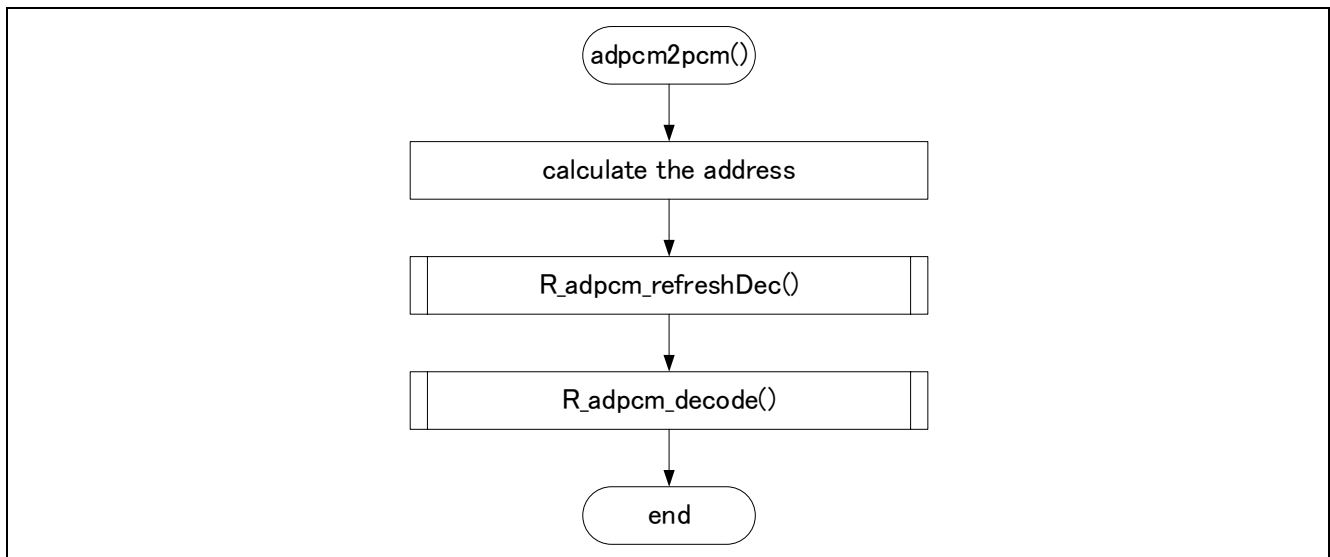
図 5-13 get_i2s_data ()



5.8.9.13 adpcm2pcm ()

図 5-14 に adpcm2pcm () のフローチャートを示します。

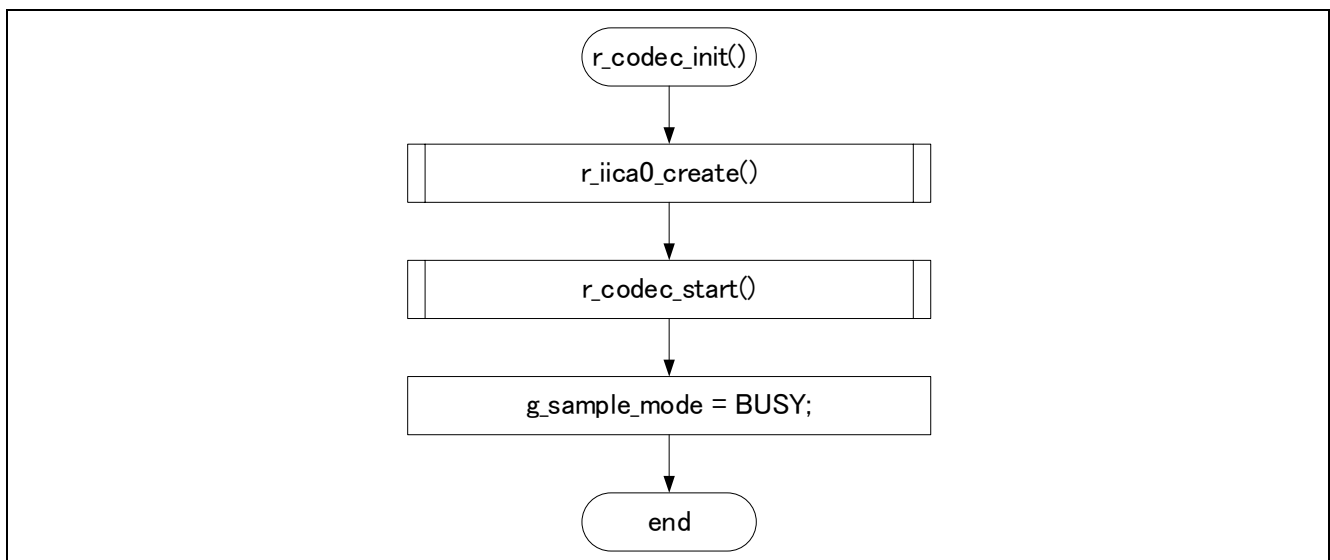
図 5-14 adpcm2pcm ()



5.8.9.14 r_codec_init ()

図 5-15 に `r_codec_init ()` のフローチャートを示します。

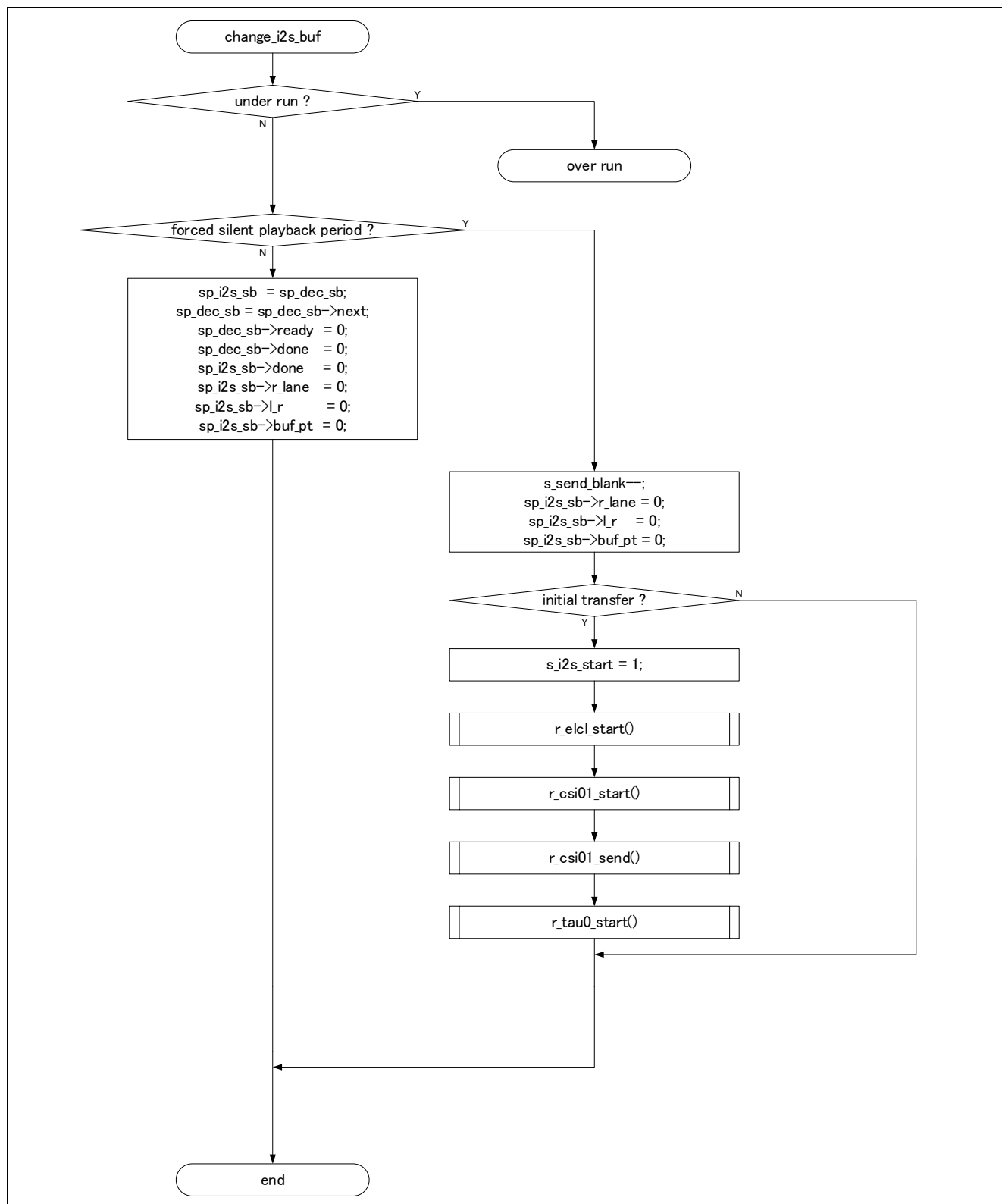
図 5-15 `r_codec_init ()`



5.8.9.15 change_i2s_buf ()

図 5-16 に change_i2s_buf () のフローチャートを示します。

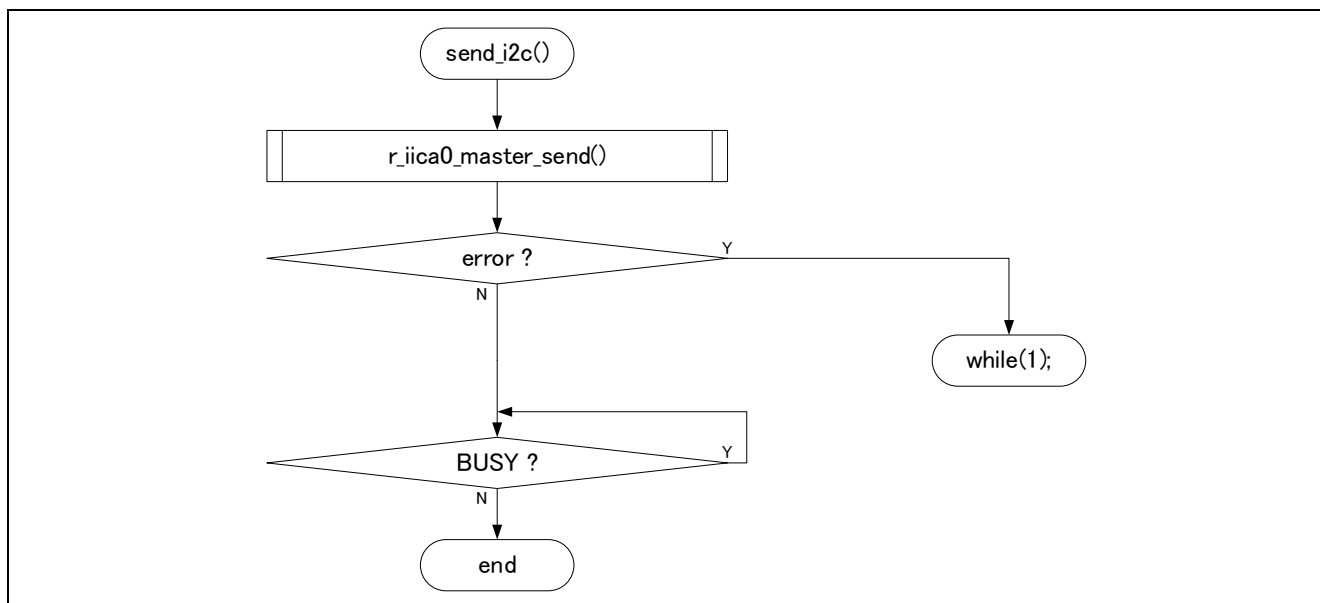
図 5-16 change_i2s_buf ()



5.8.9.16 send_i2c ()

図 5-17 に send_i2c () のフローチャートを示します。

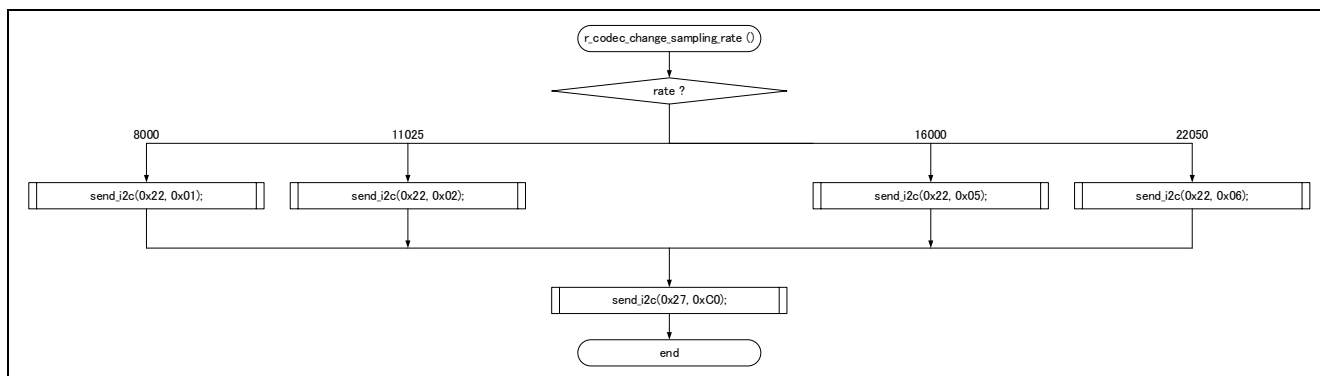
図 5-17 send_i2c ()



5.8.9.17 r_codec_change_sampling_rate ()

図 5-18 に `r_codec_change_sampling_rate ()` のフローチャートを示します。

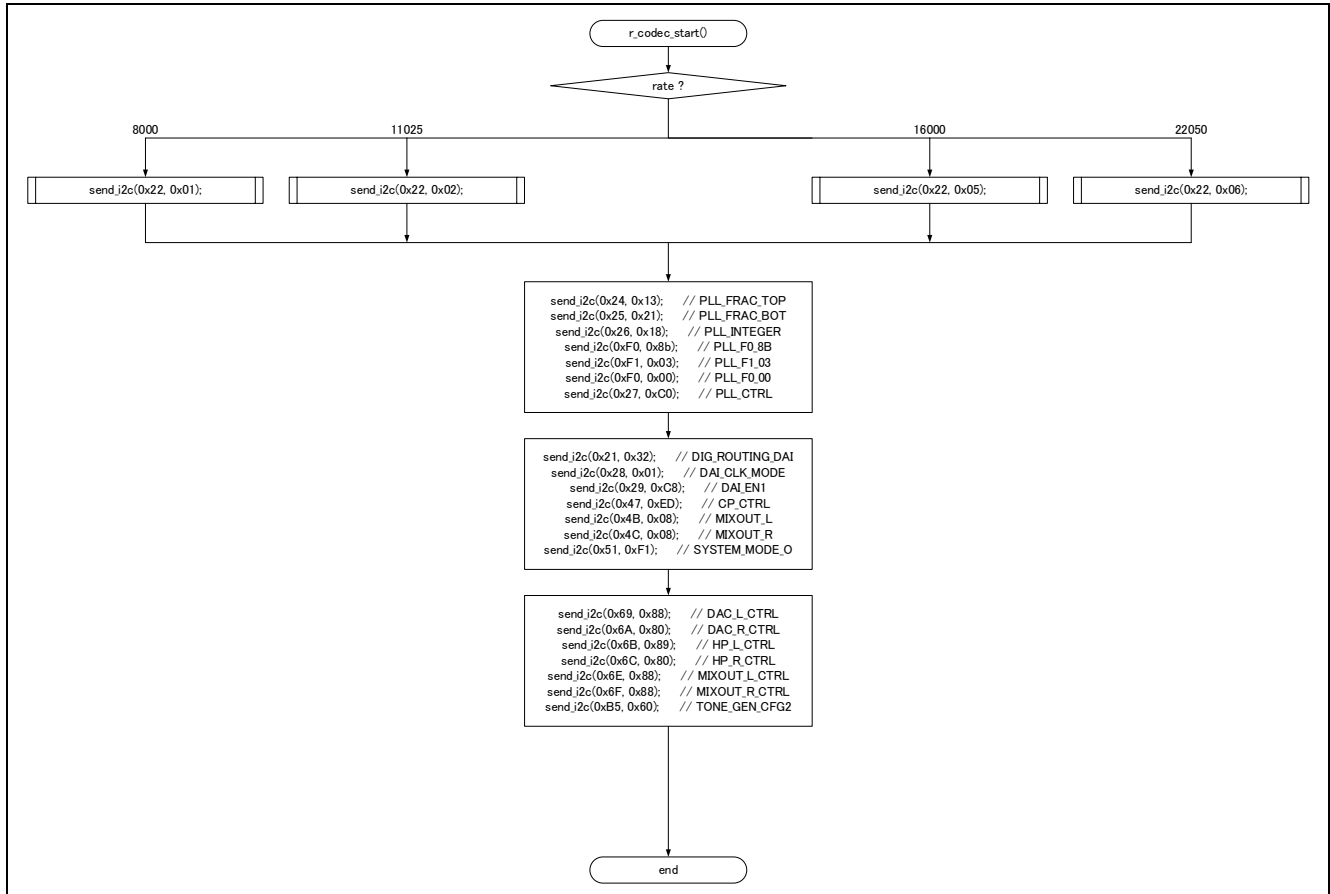
図 5-18 r_codec_change_sampling_rate ()



5.8.9.18 r_codec_start ()

図 5-19 に r_codec_start () のフローチャートを示します。

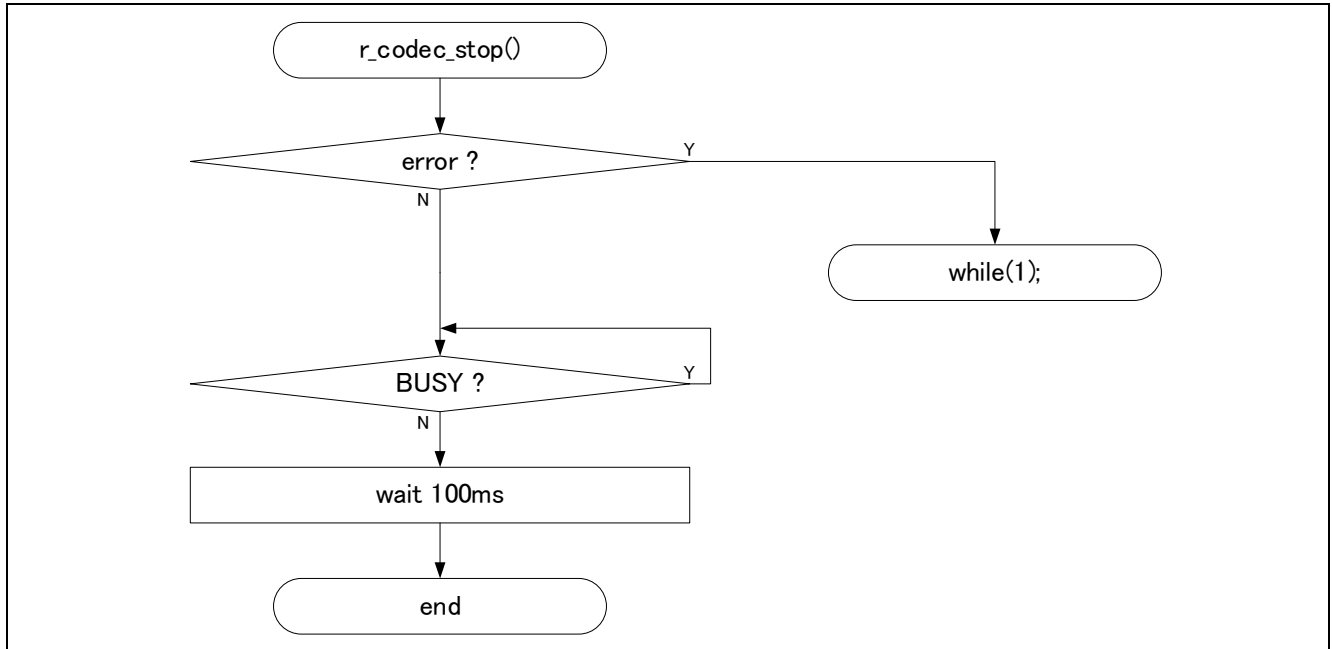
図 5-19 r_codec_start ()



5.8.9.19 r_codec_stop ()

図 5-20 に r_codec_stop () のフローチャートを示します。

図 5-20 r_codec_stop ()



5.8.10 r01an7416_adpcm_player.scfg

サンプルコードで使用しているスマート・コンフィグレータの設定ファイルです。スマート・コンフィグレータで設定されている全ての機能が含まれています。サンプルコードの設定は以下の通りです。

表 5-8 スマート・コンフィグレータの設定値

タグ名	コンポーネント	内容
クロック	-	動作モード：高速メインモード 2.4 (V) ~ 5.5 (V) EV _{DD} 設定：1.8V ≤ EV _{DD0} < 5.5V 高速オンチップ・オシレータ：32MHz f _{IHP} ：32MHz f _{CLK} ：32000kHz（高速オンチップ・オシレータ） f _{SXP} ：32.768kHz（低速オンチップ・オシレータ）
システム	-	オンチップ・デバッグ動作設定：COM ポート 疑似 RRM/DMM 機能設定：使用する Start/Stop 関数機能設定：使用しない トレース機能設定：使用する セキュリティ ID 設定：セキュリティ ID を設定する セキュリティ ID：0x00000000000000000000 セキュリティ ID 認証失敗時の設定：フラッシュ・メモリのデータを消去しない
コンポーネント	r_bsp	Start up select：Enable (use BSP startup) Control of invalid memory access detection：Disable RAM guard space (GRAM0-1)：Disabled Guard of control registers of port function (GPORT)：Disabled Guard of registers of interrupt function (GINT)：Disabled Guard of control registers of clock control function, voltage detector, and RAM parity error detection function (GCSC)：Disabled Data flash access control (DFLEN)：Disables Initialization of peripheral functions by Code Generator/Smart Configurator：Enable API functions disable：Enable Parameter check enable：Enable Setting for starting the high-speed on-chip oscillator at the times of release from STOP mode and of transitions to SNOOZE mode：High-speed Enable user warm start callback (PRE)：Unused Enable user warm start callback (POST)：Unused Watchdog Timer refresh enable：Unused
	Config_LVD0	動作モード設定：リセット・モード 電圧検出設定：リセット発生電圧 (V _{LVD0})：1.86 (V)
	Config_INTC	INTP0：使用する 有効エッジ：立下リエッジ 優先順位：レベル 3（低優先順位）

表 5-9 スマート・コンフィグレータの設定値

タグ名	コンポーネント	内容
コンポーネント	Config_TAU0_0	コンポーネント : インターバル・タイマ 動作モード : 16 ビット・カウンタ・モード リソース : TAU0_0 動作クロック : CK00 クロックソース : f _{CLK} インターバル時間 : 0.163us カウント開始時に INTTM00 割り込みを発生する 割り込み設定 : 使用しない
	Config_TAU0_1	コンポーネント : 外部イベント・カウンタ リソース : TAU0_1 動作クロック : CK00 クロックソース : f _{CLK} 入力ソース設定 : ELCL 動作モード設定 : 16 ビット 外部イベント・エッジ選択 (TI01) : 立ち上がりエッジ カウント値 : 32 割り込み設定 : 使用しない
	Config_TAU0_7	コンポーネント : インターバル・タイマ 動作モード : 16 ビット・カウンタ・モード リソース : TAU0_7 動作クロック : CK01 クロックソース : f _{CLK} /2 ⁸ インターバル時間 : 1ms 割り込み設定 : 使用する 優先順位 : レベル 2
	Config_Through	コンポーネント : ELCL Through Common setting : L3L0 Detail setting : L3L0 Input signal selector : ELISEL_5 , TO00 Application : Through Output signal selector : P51

表 5-10 スマート・コンフィグレータの設定値

タグ名	コンポーネント	内容
コンポーネント	Config_PCLBUZ0	出力クロック : $f_{MAIN}/2^3$
	Config_PORT	コンポーネント : ポート ポート選択 : PORT5 P53 : 出力 (1 を出力)
	Config_CSI20	転送クロックモード : 内部クロック (マスタ) 動作クロック : CK10 クロックソース : f_{CLK} 転送モード : シングル転送モード データビット長設定 : 8 ビット データ転送方向設定 : MSB データ送受信タイミング設定 : タイプ 1 ボーレート : 1000000bps 通信完了割り込み優先順位 : レベル 3 (低優先順位) コールバック機能設定 : 送信完了、受信完了、オーバーランエラー
	r_nor_flash	Parameter check : use system default ENABLE CHECKING OF THE WEL BIT. : Check WEL bit Select serial flash memory device : Renesas Electronics AT25QF Select serial flash memory capacity : 64M bit CS Port Number : PORT1 CS Bit Number : BIT7 Data transfer mode : CPU transfer (Software transfer) SPI(CSI) channel number : Channel 4 DTC Control Data Number : 0

5.8.10.1 クロック

サンプルコードで使用するクロックの設定を行います (設定値は表 5-8 を参照してください)。

5.8.10.2 システム

サンプルコードのオンチップ・デバッグ設定を行います。

「オンチップ・デバッグ動作設定」、「セキュリティ ID 認証失敗時の設定」は、「表 5-3 オプション・バイト設定」の「オンチップ・デバッグ動作許可」に影響を与えます。設定を変更する際は注意してください。

5.8.10.3 r_bsp

サンプルコードのスタートアップの設定を行います。

5.8.10.4 Config_LVD0

サンプルコードの電源管理の設定を行います。

「表 5-3 オプション・バイト設定」の「LVD0 の設定」に影響を与えます。設定を変更する際は注意してください。

5.8.10.5 Config_INTC

サンプルコードの INTP0 の設定を行います。

サンプルコードでは、ユーザスイッチ (SW1) の判定に使用します。

5.8.10.6 Config_TAU00

サンプルコードの TAU00 の設定を行います。

サンプルコードでは、「インターバル・タイマ」として使用し、BCLK を生成します。サンプリング周波数は、サンプルプログラム内で自動で設定されます。

5.8.10.7 Config_TAU01

サンプルコードの TAU01 の設定を行います。

サンプルコードでは、「外部イベント・カウンタ」として使用し、LRCLK を生成します。サンプルコードでのデータサイズは 32 ビット固定です。

5.8.10.8 Config_TAU07

サンプルコードの TAU07 の設定を行います。

サンプルコードでは 1ms のインターバル・タイマとして使用します。

5.8.10.9 Config_Through

サンプルコードの BCLK の出力端子を変更するために設定を行います。

BCLK として使用する TO00 の出力先を P51 に設定します。

5.8.10.10 Config_PORT

サンプルコードのポートの設定を行います。

サンプルコードでは LED1 の制御に P53 を使用します。

5.8.10.11 Config_CSI20

NOR FLASH 用 SPI を制御するために使用します。

5.8.10.12 r_nor_flash

NOR FLASH 用ミドルウェアです。

5.8.10.13 Config_PCLBUSZ0

ARD-AUDIO-DA7212 ヘクロックを供給するために使用します。

6. サンプルコード

サンプルコードは、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。

7. 参考ドキュメント

RL78/G23 ELCL、SPI による I²S 通信 アプリケーションノート (R01AN6420)

Serial NOR Flash Memory 制御モジュール Software Integration System (R01AN7243)

RL78 ファミリ 音声録音・再生システム(独自 ADPCM コーデック) M3S-S2-Tiny 導入ガイド (R20AN0122)

RL78/G23 ユーザーズマニュアル ハードウェア編 (R01UH0896)

RL78 ファミリ ユーザーズマニュアル ソフトウェア編 (R01US0015)

RL78 スマート・コンフィグレータ ユーザーガイド : CS+編 (R20AN0580)

RL78 スマート・コンフィグレータ ユーザーガイド : e² studio 編 (R20AN0579)

RL78 スマート・コンフィグレータ ユーザーガイド : IAR 編 (R20AN0581)

(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。)

テクニカルアップデート／テクニカルニュース

(最新版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。)

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。

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	Oct.21.24	-	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、入力信号や入出力ブルアップ電源を入れないでください。入力信号や入出力ブ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くと、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、ノイズ輻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、これに限られません。）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た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。