V850E2/ML4
性能評価ソフトウェア

R01AN1228JJ0100
Rev.1.00
2012.08.02

要旨

本アプリケーションノートでは、V850E2/ML4のタイマ・アレイ・ユニットA（TAUA）を使用し、ユーザ作成タスク（関数）の性能を評価するサンプルコードについて説明しています。

対象デバイス

V850E2/ML4

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あわせて変更し、十分評価してください。

目次

1. 仕様

本アプリケーションノートでは、ユーザ作成タスク（関数）の性能を評価します。本サンプルコードにユーザ作成タスクを組み込み、シリアル端末からユーザ作成タスクの選択と起動を行います。ユーザ作成タスクの開始から終了までに要したサイクル数を測定することで性能評価を行います。

図 1.2の 仕様概略フローで示すように、V850E2/ML4 CPUボードは複数ある評価処理の中から実行するものを指定するコマンド（評価コマンド）が送られてくるのを待ちます。評価コマンドが送られてきたら対応する評価処理を実行して結果をシリアル出力から送信します。

性能評価を行うときは、ホストPCから評価コマンドをシリアルケーブル経由でV850E2/ML4 CPUボードに転送します。このため、V850E2/ML4 CPUボードの他にシリアルケーブルとハイパーターミナル等のシリアル通信アプリケーションソフトがインストールされたホストPCが必要です。性能評価の詳細については 5.1性能評価を参照してください。

表 1.1に 使用する周辺機能と用途を、図 1.1に システム構成図を、図 1.2に 仕様概略フローを示します。

表1.1　使用する周辺機能と用途

周辺機能	用途
タイマ・アレイ・ユニット A（TAUA）	タスク処理時間測定
アシンクロナス・シリアル・インタフェース J（UARTJ）	ホスト PC からの評価コマンド受信および評価結果送信
割り込み機能	UARTJ 受信割り込みと TAUA 割り込みで使用

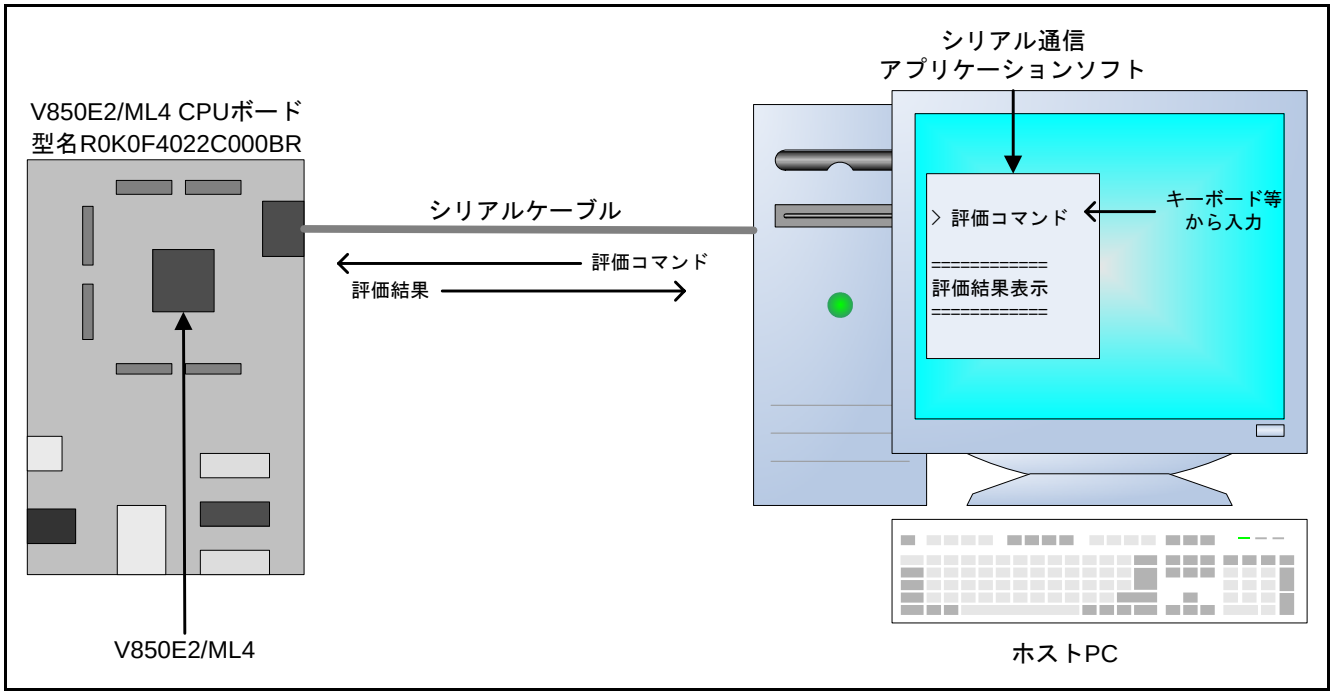

図1.1　システム構成図

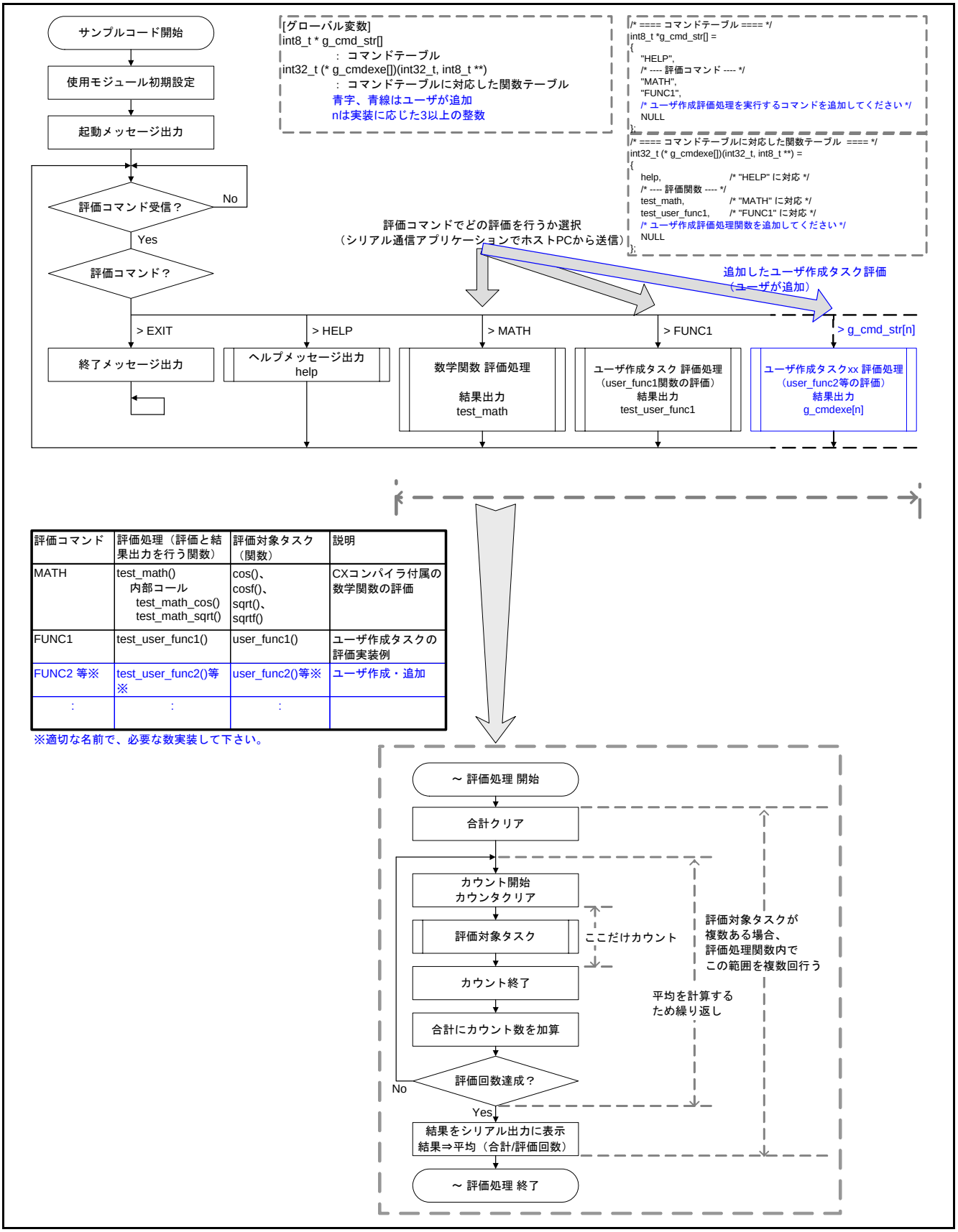
サンプルコード開始
使用モジュール初期設定
起動メッセージ出力
評価コマンド受信？
No
Yes
評価コマンド？
[グローバル変数]
int8_t * g_cmd_str[]
：コマンドテーブル
int32_t (* g_cmdexe[])(int32_t, int8_t **)
：コマンドテーブルに対応した関数テーブル
青字、青線はユーザが追加
nは実装に応じた3以上の整数
/* ==== コマンドテーブル ==== */
int8_t *g_cmd_str[] =
{
"HELP",
/* ---- 評価コマンド ---- */
"MATH",
"FUNC1",
/* ユーザ作成評価処理を実行するコマンドを追加してください */
NULL
};
/* ==== コマンドテーブルに対応した関数テーブル ==== */
int32_t (* g_cmdexe[])(int32_t, int8_t **) =
{
help, /* "HELP" に対応 */
/* ---- 評価関数 ---- */
test_math, /* "MATH" に対応 */
test_user_func1, /* "FUNC1" に対応 */
/* ユーザ作成評価処理関数を追加してください */
NULL
};
評価コマンドでどの評価を行うか選択
（シリアル通信アプリケーションでホストPCから送信）
追加したユーザ作成タスク評価
（ユーザが追加）
> EXIT
> HELP
> MATH
> FUNC1
> g_cmd_str[n]
終了メッセージ出力
ヘルプメッセージ出力
help
数学関数 評価処理
結果出力
test_math
ユーザ作成タスク 評価処理
（user_func1関数の評価）
結果出力
test_user_func1
ユーザ作成タスクxx 評価処理
（user_func2等の評価）
結果出力
g_cmdexe[n]
評価コマンド
評価処理（評価と結果出力を行う関数）
評価対象タスク（関数）
説明
MATH
test_math()
内部コール
test_math_cos()
test_math_sqrt()
cos()、
cosf()、
sqrt()、
sqrtf()
CXコンパイラ付属の数学関数の評価
FUNC1
test_user_func1()
user_func1()
ユーザ作成タスクの評価実装例
FUNC2 等※
test_user_func2()等※
user_func2()等※
ユーザ作成・追加
：
：
：
※適切な名前で、必要な数実装して下さい。
～ 評価処理 開始
合計クリア
カウント開始
カウンタクリア
評価対象タスク
ここだけカウント
カウント終了
合計にカウント数を加算
評価回数達成？
No
Yes
結果をシリアル出力に表示
結果⇒平均（合計/評価回数）
～ 評価処理 終了
評価対象タスクが
複数ある場合、
評価処理関数内で
この範囲を複数回行う
平均を計算する
ため繰り返し

図1.2　仕様概略フロー

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	V850E2/ML4
動作周波数	内部システム・クロック（f_{CLK}） ：200MHz P バス・クロック（f_{PCLK}） ：66.667MHz
動作電圧	Vcc：3.3V
統合開発環境	ルネサス エレクトロニクス製 CubeSuite+ Ver.1.02.01
C コンパイラ	ルネサス エレクトロニクス製 CX コンパイラパッケージ Ver.1.21
	コンパイルオプション -C f4022 -o DefaultBuild¥v850e2ml4_eval.lmf -Xobj_path=Defaultbuild -g -I%ProjectDir%¥inc -Xdef_ver +Xide -Xmap=DefaultBuild¥v850e2ml4_eval.map -Xhex=DefaultBuild¥v850e2ml4_eval.hex
動作モード	通常動作モード
サンプルコードのバージョン	1.00
使用ボード	R0K0F4022C000BR
使用ツール	シリアル通信アプリケーション

3. ハードウェア説明

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
P2_13/TXD0F	出力	シリアルポートへの出力として使用
P2_12/RXD0F	入力	シリアルポートからの入力として使用

4. ソフトウェア説明

4.1 動作概要

本サンプルコードは、V850E2/ML4 のタイマ・アレイ・ユニット A（TAUA）を使用し、ユーザ作成タスク（関数）の性能を評価します。

ユーザ作成タスクを組み込んだ性能評価ソフトウェアに、ホスト PC のシリアル通信アプリケーション（シリアル端末）から評価コマンドを転送して、評価を実行します。ユーザ作成タスクの実行期間中に、TAUA が要する PCLK サイクル数を測定して、CPU のクロックサイクル数を計算することで性能評価を行います。性能評価を行った結果はシリアル出力から送信してシリアル端末に表示します。

図 4.1に 性能評価概要を示します。

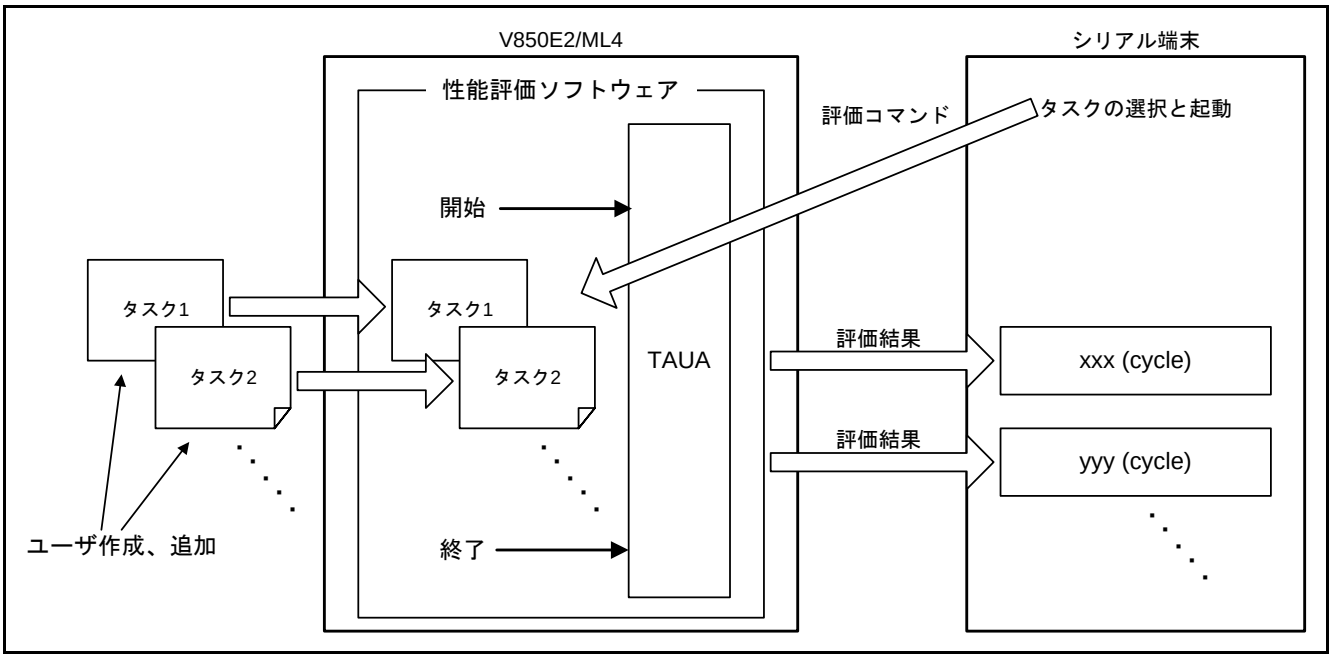

図4.1 性能評価概要

4.2 ファイル構成

表 4.1に サンプルコードで使用するファイルを示します。なお、統合開発環境で自動生成されるファイルは除きます。

表4.1 サンプルコードで使用するファイル

ファイル名	概要	備考
main.c	メイン処理モジュール	
io_taua0_timer.c	タイマ処理モジュール	
io_uartj0_stdio.c	シリアル入出力処理モジュール	
test_math.c	数学関数評価モジュール	
test_user.c	ユーザ作成タスク評価モジュール	
user_func1.c	ユーザ作成モジュール	
io_taua0_timer.h	タイマ処理ヘッダ	
io_uartj0_stdio.h	シリアル入出力処理ヘッダ	
user_func1.h	ユーザ作成モジュールヘッダ	
r_typedefs.h	固定幅整数型定義ヘッダ	

4.3 定数一覧

表 4.2に サンプルコードで使用する定数を示します。

表4.2 サンプルコードで使用する定数

定数名	設定値	内容
MAX_ARGNUM	8	引数の最大数
MAX_ARGLENGTH	256	引数の文字数の最大数
BUF_SIZE_WAIT	256	文字列入力待ちバッファサイズ
BUF_SIZE_UARTJ0_STDOUT	512	出力バッファサイズ
BUF_SIZE_UARTJ0_STDIN	256	入力バッファサイズ
PI	3.141592653589	円周率
EVAL_NUM	256	MATH コマンド処理評価実行回数
EVAL_TIMES	10	FUNC1 コマンド処理評価実行回数
LOOP_TIMES	100	ユーザ作成処理内における空ループ実行回数

4.4 変数一覧

表 4.3に グローバル変数を示します。

表4.3 グローバル変数

型	変数名	内容	使用関数
int8_t	g_buf_wait[WAIT_BUF_SIZE]	文字列入力待ちバッファ	main、 cmdline_pars
int8_t	g_arg [MAX_ARGNUM][MAX_ARGLENGTH]	コマンド文字列群格納領域	command_exe、 cmdline_split、 cmdline_pars
int8_t	* g_cmd_str[]	コマンドテーブル	cmdline_pars
int32_t	(* g_cmdexe[])(int32_t, int8_t **)	コマンドテーブルに対応した処理関数テーブル	cmdline_pars
double	g_buf_double_result[EVAL_NUM]	cos 関数と sqrt 関数の計算結果格納バッファ	test_math、 test_math_cos、 test_math_sqrt
float	g_buf_float_result[EVAL_NUM]	cosf 関数と sqrtf 関数の計算結果格納バッファ	test_math、 test_math_cos、 test_math_sqrt
uint32_t	g_cnt_int_taua0	割り込み処理実行回数計測用カウンタ	io_int_taua0_count_time、 io_taua0_start_timer、 io_taua0_get_counter
uint8_t	g_buf_uartj0_stdout [UARTJ0_STDOUT_BUF_SIZE]	出力データバッファ	io_uartj0_printf
uint8_t	g_buf_uartj0_stdin [UARTJ0_STDIN_BUF_SIZE]	入力データバッファ	io_int_uartj0_recv、 io_uartj0_fgets
int32_t	g_cnt_uartj0_stdin	入力データ数カウンタ	io_int_uartj0_recv、 io_uartj0_fgets

4.5 関数一覧

表 4.4に 関数を示します。

表4.4 関数

関数名	概要
main	メイン処理
command_exe	コマンドライン文字列の実行処理
cmdline_split	コマンドライン文字列の分離処理
cmdline_pars	コマンド文字列群の解析実行処理
help	ヘルプコマンド処理
io_init_uartj0	UARTJ0 初期化処理
io_int_uartj0_recv	UARTJ0 受信割り込み処理
io_uartj0_printf	UARTJ0 フォーマット指定シリアル出力処理
io_uartj0_fgets	UARTJ0 行読み込みシリアル入力処理
io_init_taua0	TAUA0 初期化処理
io_int_taua0_count_timer	TAUA0 割り込み処理
io_taua0_start_timer	TAUA0 タイマカウント動作開始処理
io_taua0_stop_timer	TAUA0 タイマカウント動作停止処理
io_taua0_get_counter	TAUA0 タイマカウント取得処理
test_math	数学関数ライブラリ速度評価処理
test_math_cos	余弦演算速度評価処理
test_math_sqrt	平方根演算速度評価処理
test_user_func1	ユーザ作成処理速度評価処理
user_func1	ユーザ作成処理

4.6 関数仕様

サンプルコードの関数仕様を示します。

main	
概　要	メイン処理
ヘッダ	
宣　言	void main(void)
説　明	初期化処理を行った後、コマンド入力待ちを行い、入力されたコマンドを実行します。 コマンド実行は command_exe 関数をコールして行います。
引　数	なし
リターン値	なし

command_exe	
概　要	コマンドライン文字列の実行処理
ヘッダ	
宣　言	int32_t command_exe(int8_t * buf)
説　明	引数で指定されたコマンドライン文字列に対応する処理を実行します。 cmdline_split 関数をコールしてコマンドライン文字列の分離処理を行い、コマンド文字列と 0 個以上の引数文字列を含む、文字列の配列（コマンド文字列群）に格納した後、cmdline_pars 関数をコールして、そのコマンド文字列群の解析と実行を行います。
引　数	int8_t * buf　：コマンドライン文字列
リターン値	>0　：コマンド処理に依存 0　：正常終了 -1　：EXIT コマンド検出

cmdline_split	
概　要	コマンドライン文字列の分離処理
ヘッダ	
宣　言	int32_t cmdline_split(int8_t * cmdline, int8_t * argv[])
説　明	cmdline で指定されたコマンドライン文字列をスペースで分離し、分離して得られたコマンド文字列と引数文字列を最大 MAX_ARGNUM 個まで配列 argv に格納します。 cmdline で指定されたコマンドライン文字列の先頭に '>' があるとき、 '>' の次の文字から分離処理を開始します。分離処理では 2 個以上のスペースが続いたときは 1 個のスペースとして処理します。ただし、ダブルクォーテーションで囲まれた文字列は 1 個の単一文字列とします。また、格納した文字列の数をリターンします。
引　数	int8_t * cmdline　：コマンドライン文字列 int8_t * argv[]　：分離結果（コマンド文字列群）の格納アドレス
リターン値	コマンド文字列群の文字列数

cmdline_pars	
概　要	コマンド文字列群の解析実行処理
ヘッダ	
宣　言	int32_t cmdline_pars(int32_t argc, int8_t * argv[])
説　明	引数 argv で指定されたコマンド文字列群を解析して対応したコマンド処理関数を実行します。 実行するコマンドは、コマンドテーブル g_cmd_str[]で登録したコマンドです。 コマンドを追加/削除する場合は、コマンドテーブル g_cmd_str[]とコマンドテーブルに対応した関数テーブル g_cmdexe[]を修正してください。
引　数	int32_t argc　：コマンド文字列群の文字列数 int8_t * argv[]　：コマンド文字列群
リターン値	>0　：コマンド処理に依存 0　：コマンドテーブル以外の正常終了 -1　：EXIT コマンド検出

help	
概　要	ヘルプコマンド処理
ヘッダ	
宣　言	int32_t help(int32_t argc, int8_t ** argv)
説　明	評価コマンドの説明を UARTJ0 のシリアル出力により行います。
引　数	int32_t argc　：コマンド文字列群の文字列数 int8_t ** argv　：コマンド文字列群
リターン値	0　：正常終了 -1　：エラー

io_init_uartj0	
概　要	UARTJ0 初期化処理
ヘッダ	
宣　言	void io_init_uartj0(void)
説　明	UARTJ0 をシリアル入出力用に設定するための初期化処理を行います。
引　数	なし
リターン値	なし

io_int_uartj0_recv	
概　要	UARTJ0 受信割り込み処理
ヘッダ	
宣　言	void io_int_uartj0_recv(void)
説　明	UARTJ0 の受信データを受信データバッファに格納します。
引　数	なし
リターン値	なし

io_uartj0_printf	
概　要	UARTJ0 フォーマット指定シリアル出力処理
ヘッダ	
宣　言	int32_t io_uartj0_printf(const int8_t format[], ...)
説　明	引数指定のフォーマットに従って、UARTJ0 により文字列をシリアル出力します。
引　数	const int8_t format[], …　：出力フォーマット文字列とフォーマットに従うデータ
リターン値	出力バイト数

io_uartj0_fgets		
概　要	UARTJ0 行読み込みシリアル入力処理	
ヘッダ		
宣　言	int8_t * io_uartj0_fgets(int8_t * s, int32_t n, FILE * stream)	
説　明	UARTJ0 シリアル入力により 1 行の文字列を読み込みます。	
引　数	int8_t * s	：読み込みデータ格納アドレス
	int32_t n	：読み込みサイズ指定
	FILE * stream	：ファイルポインタ（stdin 固定）
リターン値	NULL	：引数 stream の値が stdin でないとき、何もせずに終了
	NULL 以外	：引数 s の値（正常終了）

io_init_taua0	
概　要	TAUA0 初期化処理
ヘッダ	
宣　言	void io_init_taua0(void)
説　明	PCLK クロックサイクル数測定タイマとして使用するための TAUA0 の初期化処理を行います。
引　数	なし
リターン値	なし

io_int_taua0_count_timer	
概　要	TAUA0 割り込み処理
ヘッダ	
宣　言	void io_int_taua0_count_timer(void)
説　明	割り込み処理実行回数計測用カウンタをインクリメントします。
引　数	なし
リターン値	なし

io_taua0_start_timer	
概　要	TAUA0 タイマカウント動作開始処理
ヘッダ	
宣　言	void io_taua0_start_timer(void)
説　明	割り込み処理実行回数計測用カウンタをクリアし、TAUA0 タイマカウント動作を開始します。
引　数	なし
リターン値	なし

io_taua0_stop_timer	
概　要	TAUA0 タイマカウント動作停止処理
ヘッダ	
宣　言	void io_taua0_stop_timer(void)
説　明	TAUA0 タイマカウント動作を停止します。
引　数	なし
リターン値	なし

io_taua0_get_counter	
概　要	TAUA0 タイマカウント取得処理
ヘッダ	
宣　言	uint32_t io_taua0_get_counter(void)
説　明	割り込み処理実行回数計測用カウンタと TAUA0CNT0 レジスタからタイマカウント動作中 にかかった CPU クロックサイクル数を計算します。
引　数	なし
リターン値	タイマカウント中にかかったサイクル数

test_math		
概　要	数学関数ライブラリ速度評価処理	
ヘッダ		
宣　言	int32_t test_math (int32_t argc, int8_t ** argv)	
説　明	数学関数ライブラリ演算速度評価を行います。 余弦演算速度評価処理関数と平方根演算速度評価処理関数をコールします。	
引　数	int32_t argc	コマンド文字列群の文字列数
	int8_t ** argv	コマンド文字列群
リターン値	1	

test_math_cos	
概　要	余弦演算速度評価処理
ヘッダ	
宣　言	void test_math_cos (void)
説　明	余弦演算速度評価を行います。MATH ライブラリの cos 関数と cosf 関数を 256 回実行し、TAUA により測定した 1 回ごとの処理時間（CPU クロックサイクル数）の平均を計算します。 計算結果は io_uartj0_printf 関数をコールして出力します。
引　数	なし
リターン値	なし

test_math_sqrt	
概　要	平方根演算速度評価処理
ヘッダ	
宣　言	void test_math_sqrt (void)
説　明	平方根関数演算速度評価を行います。MATH ライブラリの sqrt 関数と sqrtf 関数を 256 回実行し、TAUA により測定した 1 回ごとの処理時間（CPU クロックサイクル数）の平均を計算します。 計算結果は io_uartj0_printf 関数をコールして出力します。
引　数	なし
リターン値	なし

test_user_func1		
概　要	ユーザ作成処理速度評価処理	
ヘッダ		
宣　言	int32_t test_user_func1 (int32_t argc, int8_t * argv[])	
説　明	user_func1 関数を 10 回実行し、TAUA により測定した 1 回ごとの処理時間（CPU クロックサイクル数）の平均を計算します。 計算結果は io_uartj0_printf 関数をコールして出力します。	
引　数	int32_t argc	コマンド文字列群の文字列数
	int8_t ** argv	コマンド文字列群
リターン値	1	

user_func1	
概　要	ユーザ作成処理
ヘッダ	
宣　言	void user_func1(void)
説　明	ユーザ作成関数の実装例です。サンプルコードでは、空ループを 100 回行います。
引　数	なし
リターン値	なし

4.7 フローチャート

4.7.1 メイン処理

図 4.2に メイン処理のフローチャートを示します。

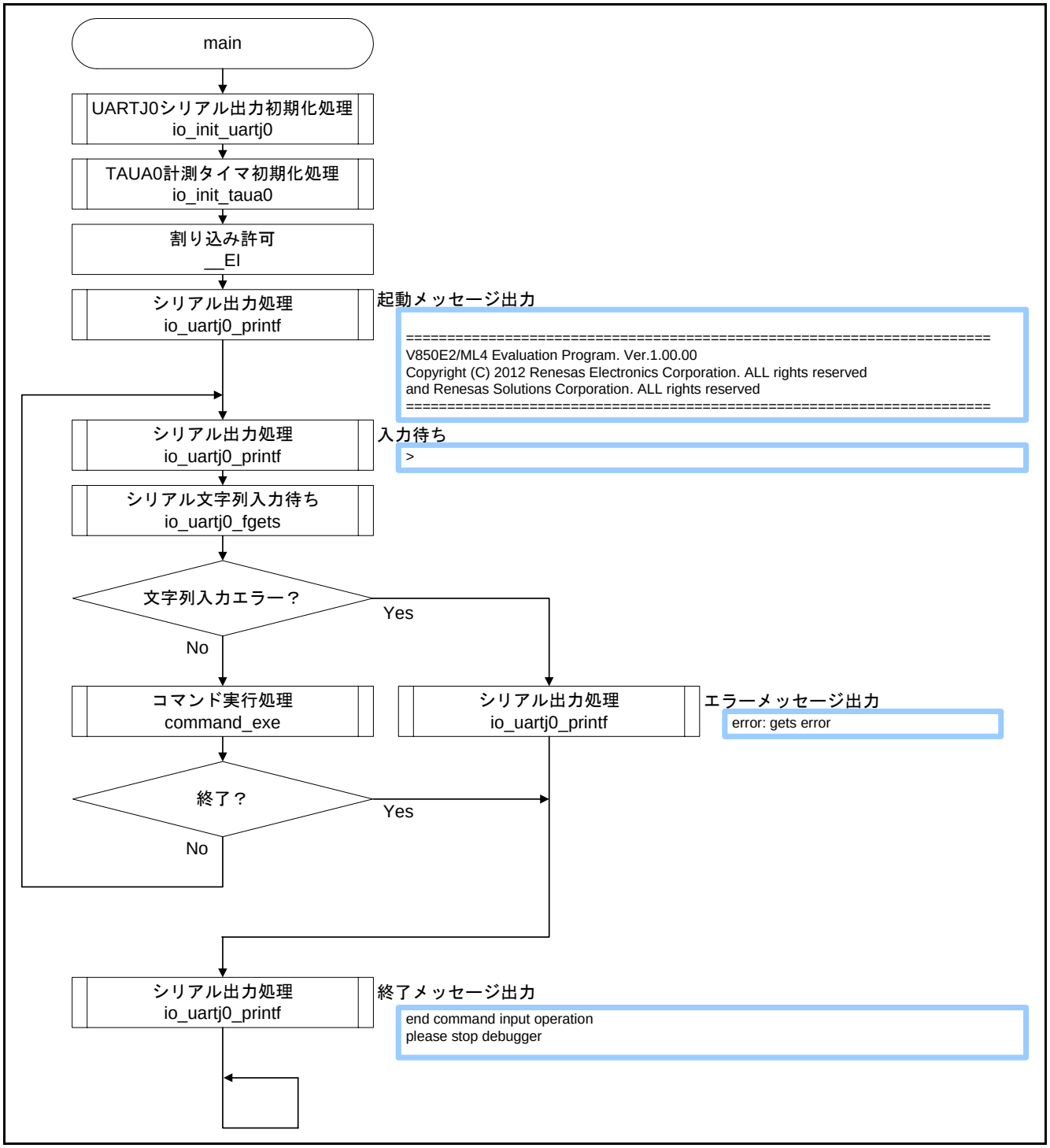

図4.2 メイン処理

4.7.2 コマンドライン文字列の実行処理

図 4.3に コマンドライン文字列の実行処理のフローチャートを示します。

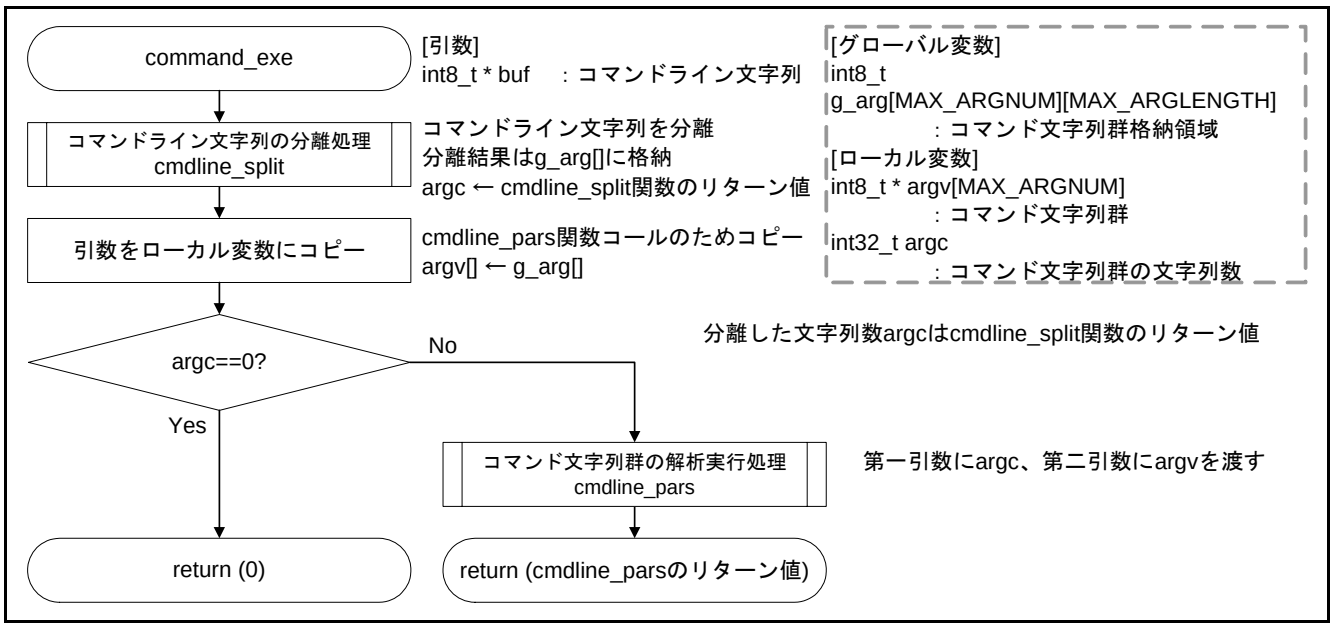

図4.3 コマンドライン文字列の実行処理

4.7.3 コマンドライン文字列の分離処理

図 4.4、図 4.5に コマンドライン文字列の分離処理のフローチャートを示します。

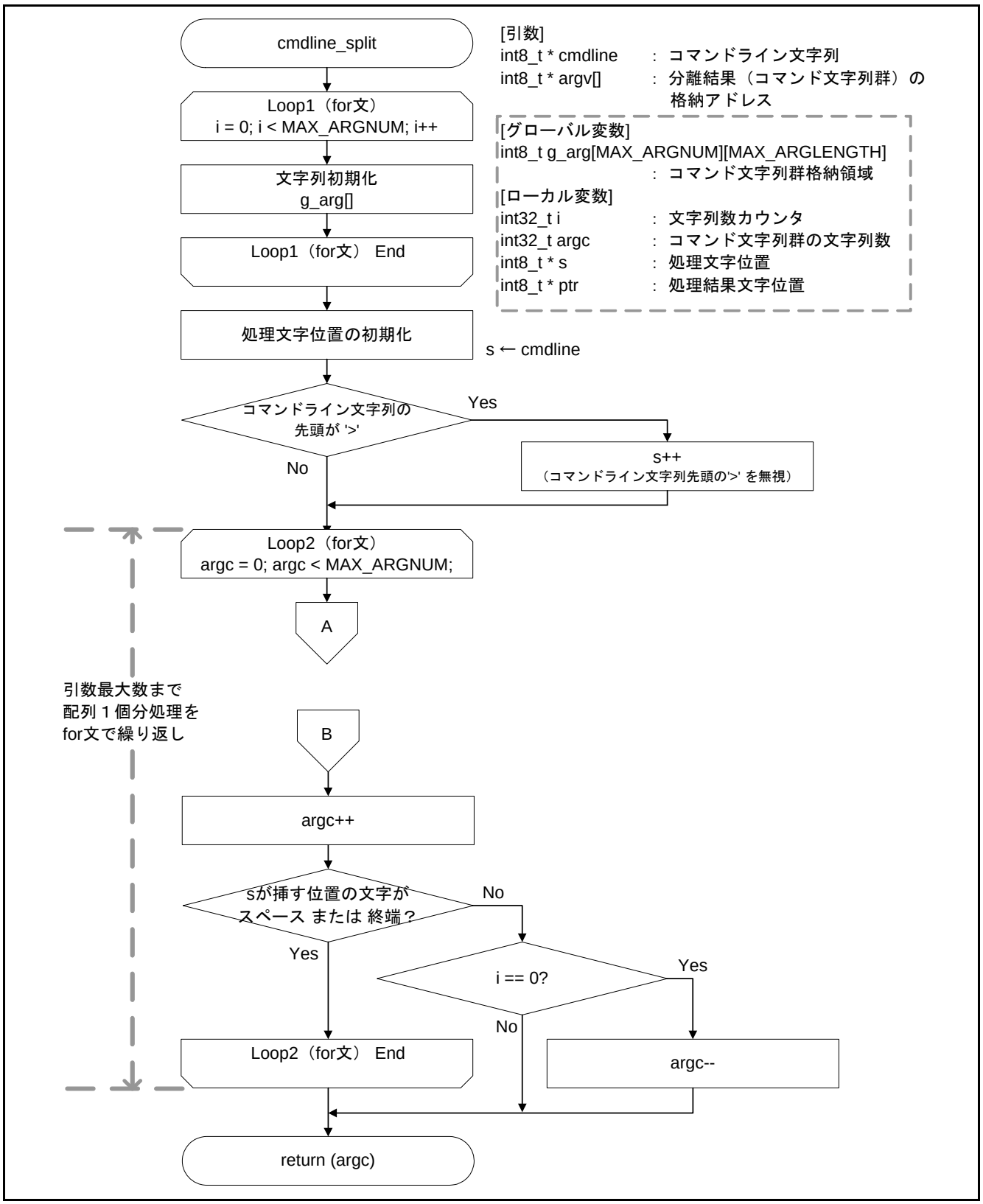

図4.4 コマンドライン文字列の分離処理

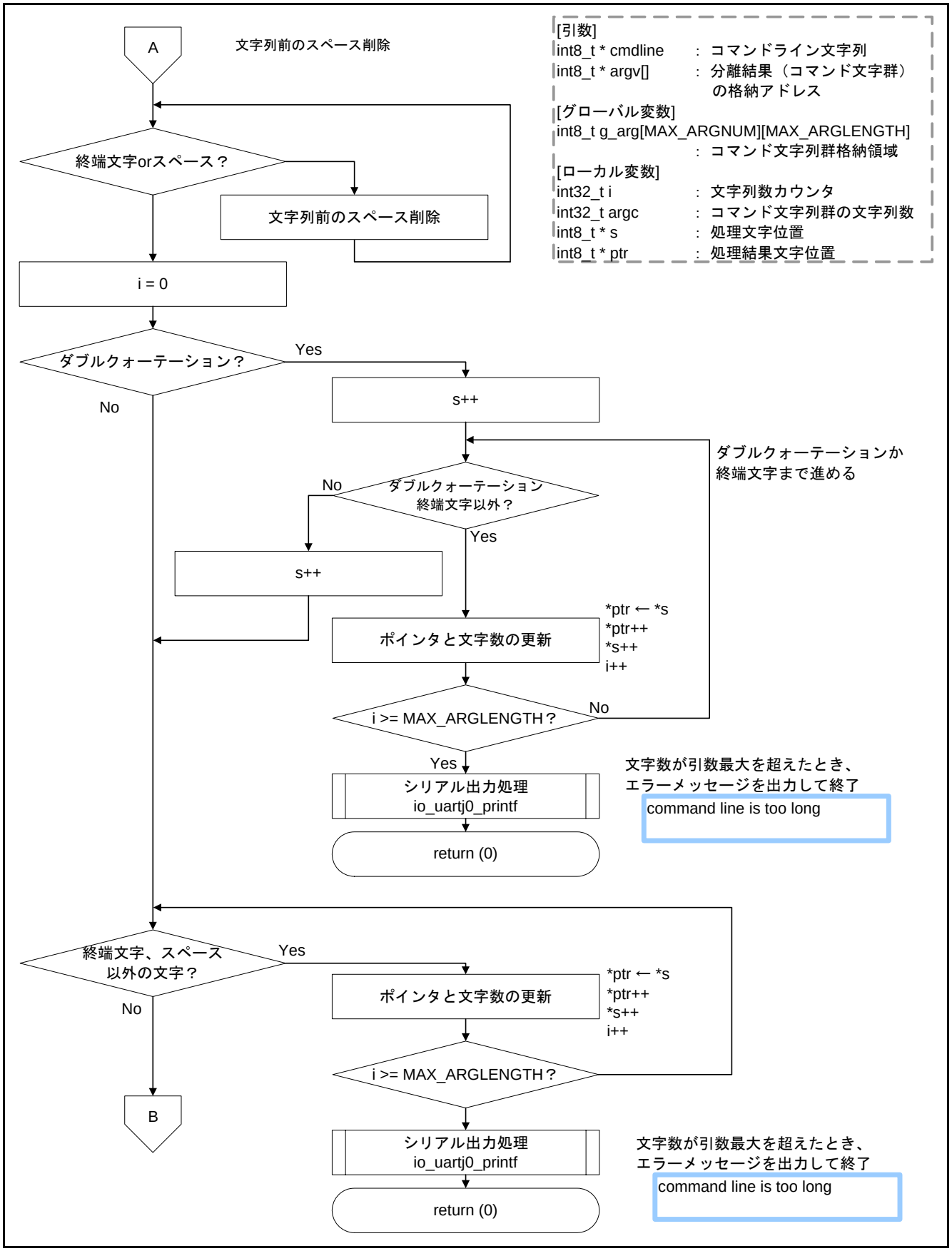

図4.5 コマンドライン文字列の分離処理

4.7.4 コマンド文字列群の解析実行処理

図 4.6に コマンド文字列群の解析実行処理のフローチャートを示します。

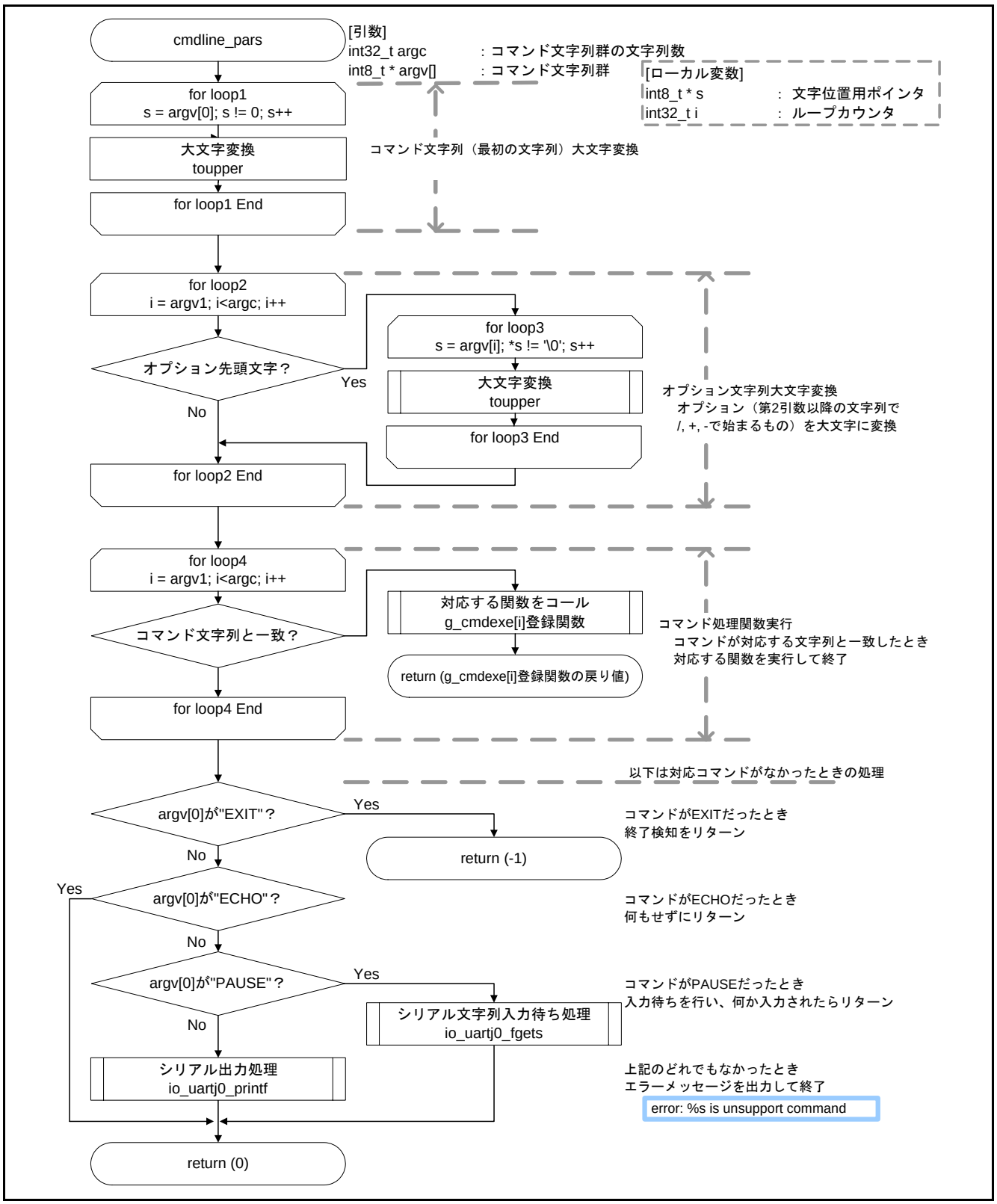

図4.6 コマンド文字列群の解析実行処理

4.7.5 ヘルプコマンド処理

図 4.7に ヘルプコマンド処理のフローチャートを示します。

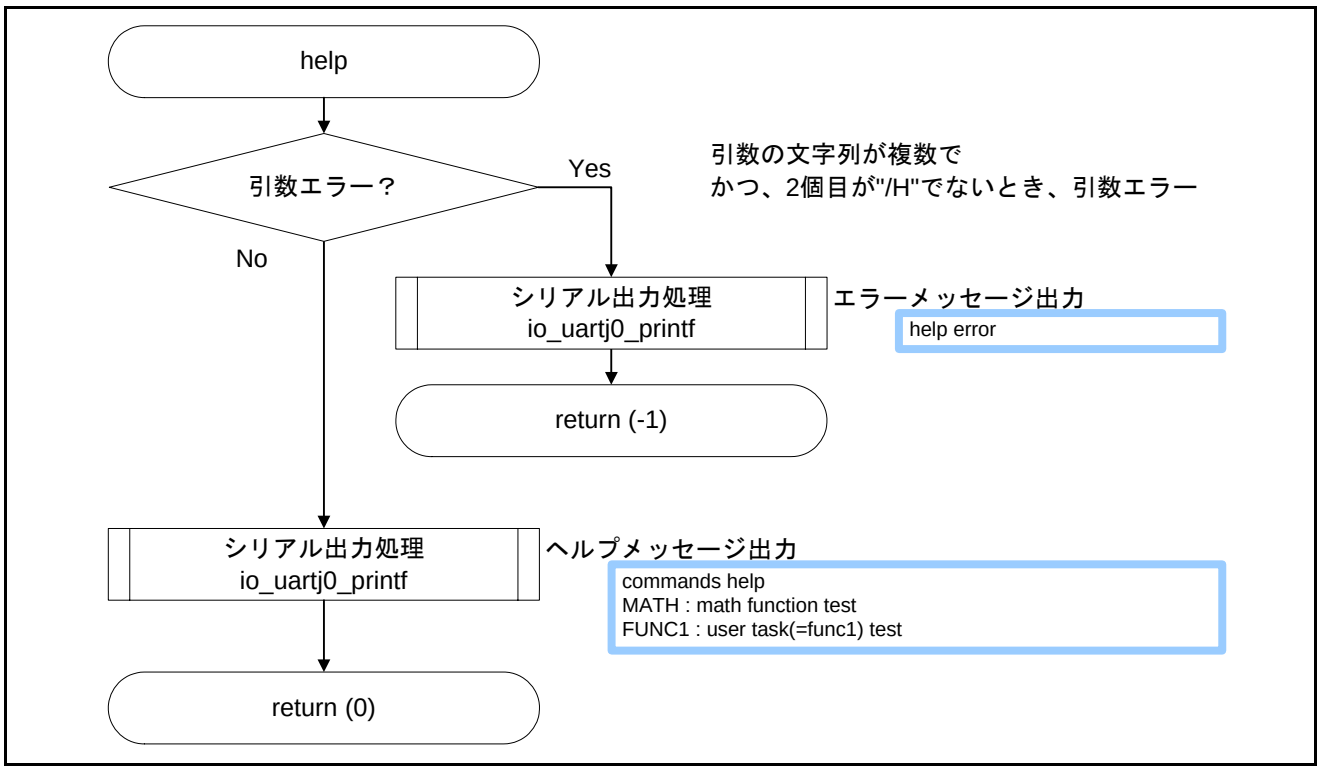

図4.7 ヘルプコマンド処理

4.7.6 UARTJ0 初期化処理

図 4.8に UARTJ0 初期化処理のフローチャートを示します。

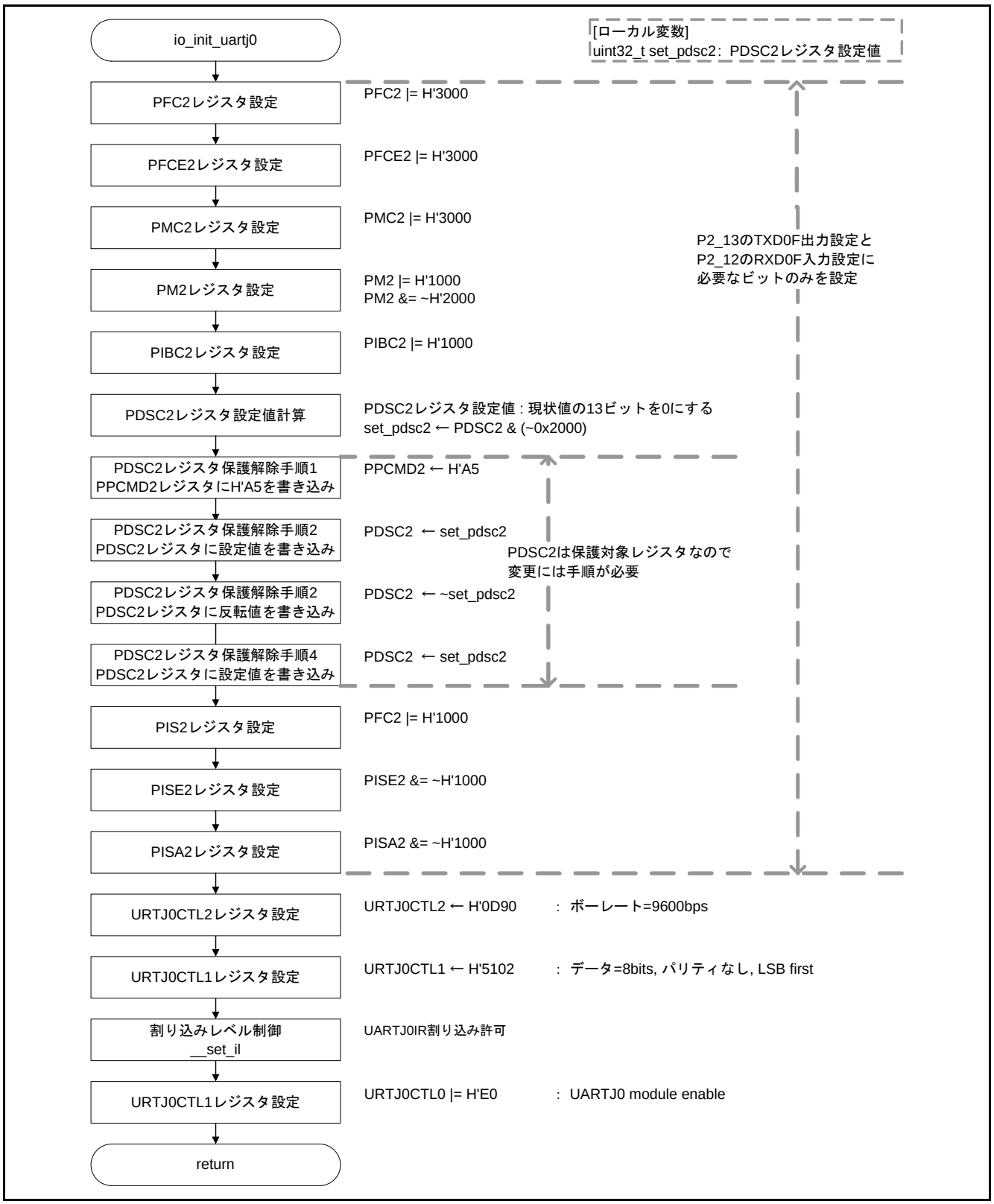

図4.8 UARTJ0 初期化処理

4.7.7 UARTJ0 受信割り込み処理

図 4.9に UARTJ0 受信割り込み処理のフローチャートを示します。

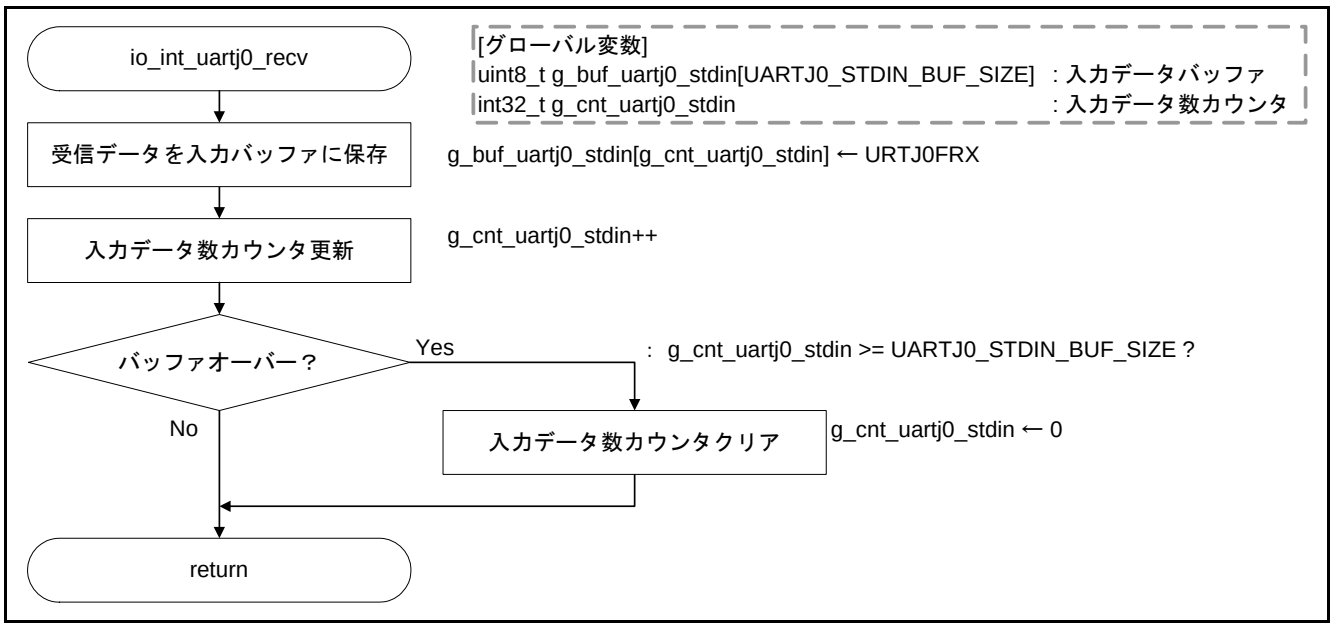

図4.9 UARTJ0 受信割り込み処理

4.7.8 UARTJ0 フォーマット指定シリアル出力処理

図 4.10に UARTJ0 フォーマット指定シリアル出力処理のフローチャートを示します。

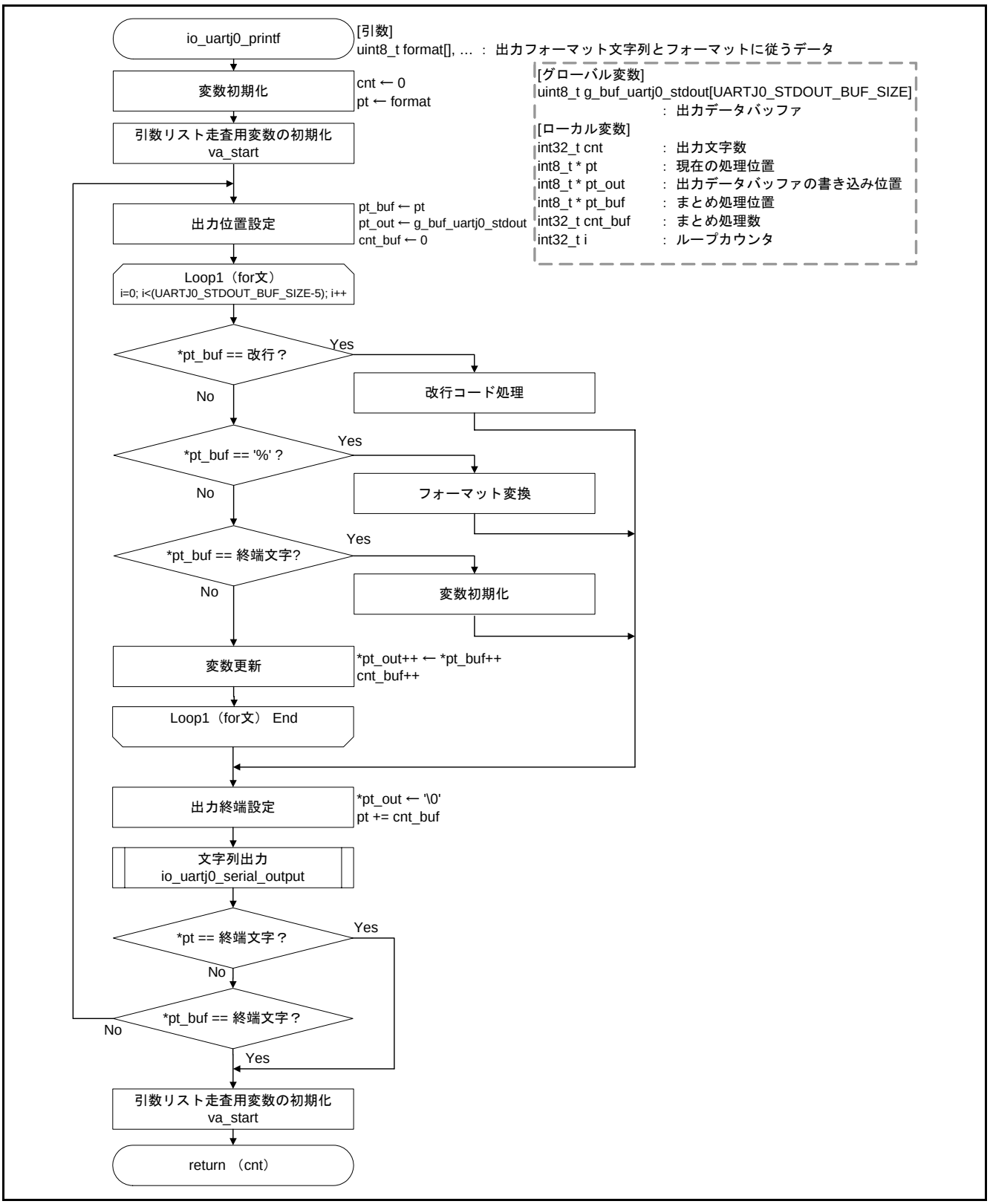

図4.10 UARTJ0 フォーマット指定シリアル出力処理

4.7.9 UARTJ0 行読み込みシリアル入力処理

図 4.11に UARTJ0 行読み込みシリアル入力処理のフローチャートを示します。

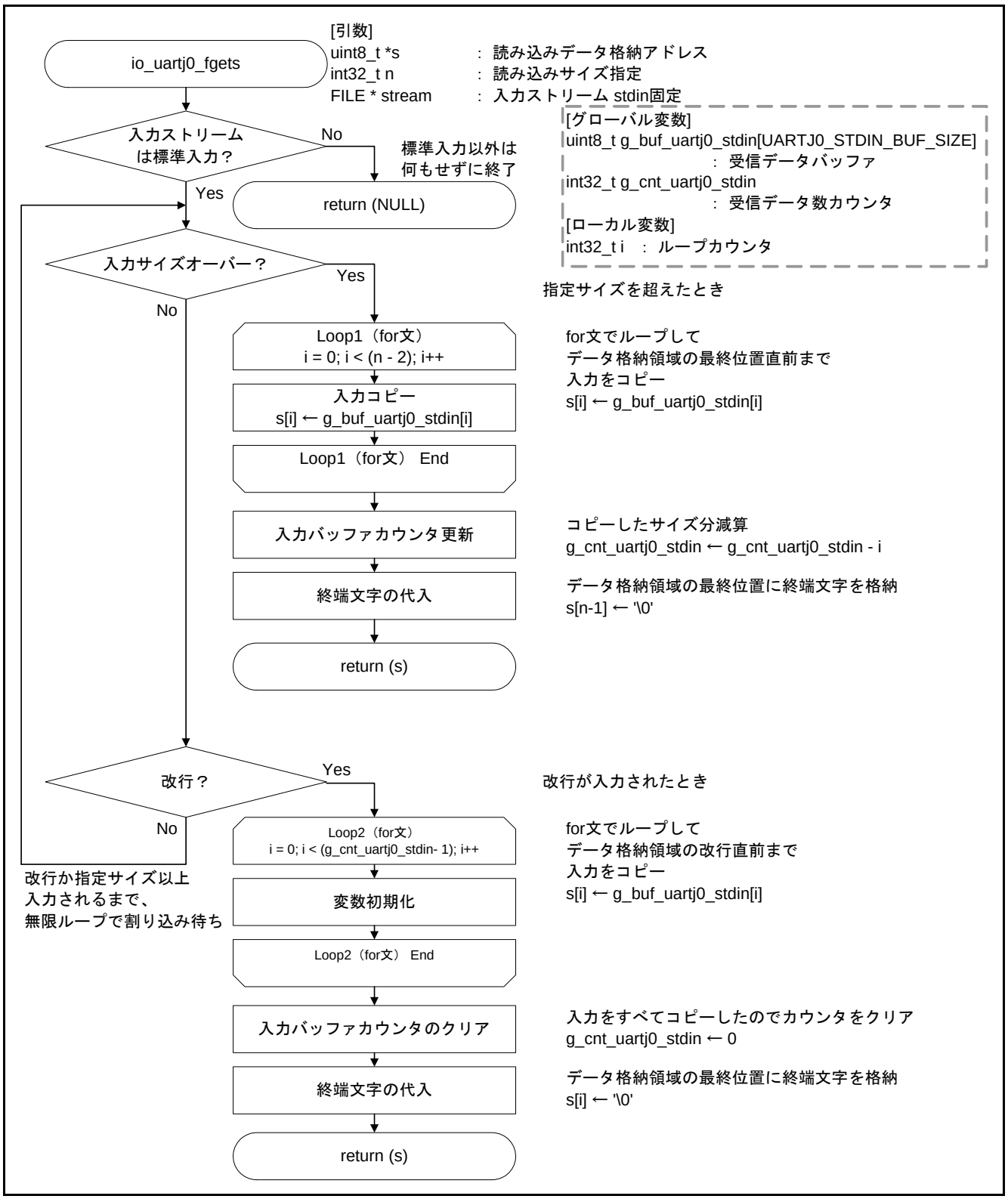

図4.11 UARTJ0 行読み込みシリアル入力処理

4.7.10 TAUA0 初期化処理

図 4.12に TAUA0 初期化処理のフローチャートを示します。

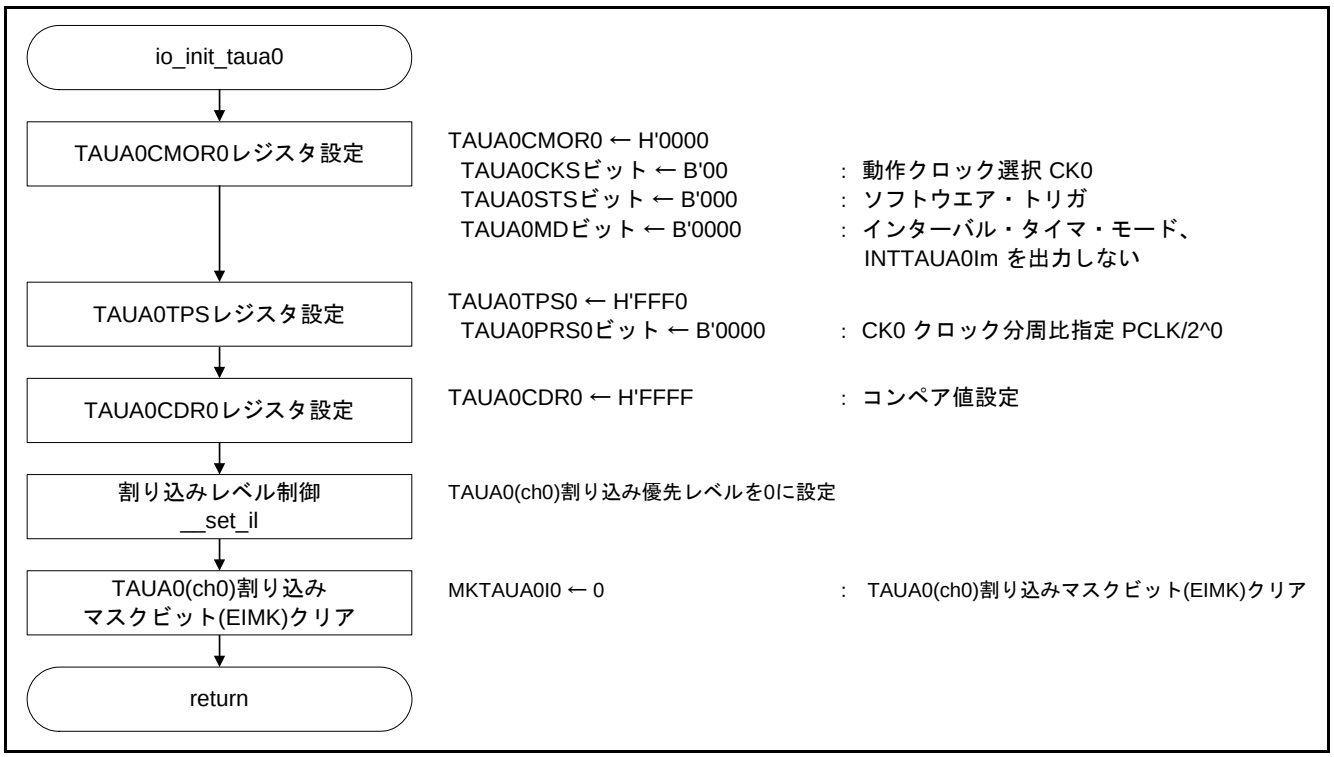

図4.12 TAUA0 初期化処理

4.7.11 TAUA0 割り込み処理

図 4.13に TAUA0 割り込み処理のフローチャートを示します。

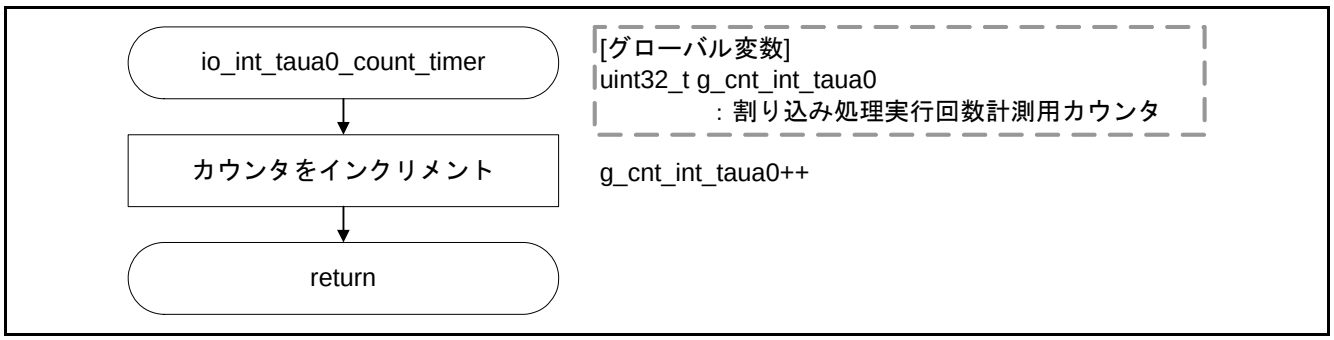

図4.13 TAUA0 割り込み処理

4.7.12 TAUA0 タイマカウント動作開始処理

図 4.14に TAUA0 タイマカウント動作開始処理のフローチャートを示します。

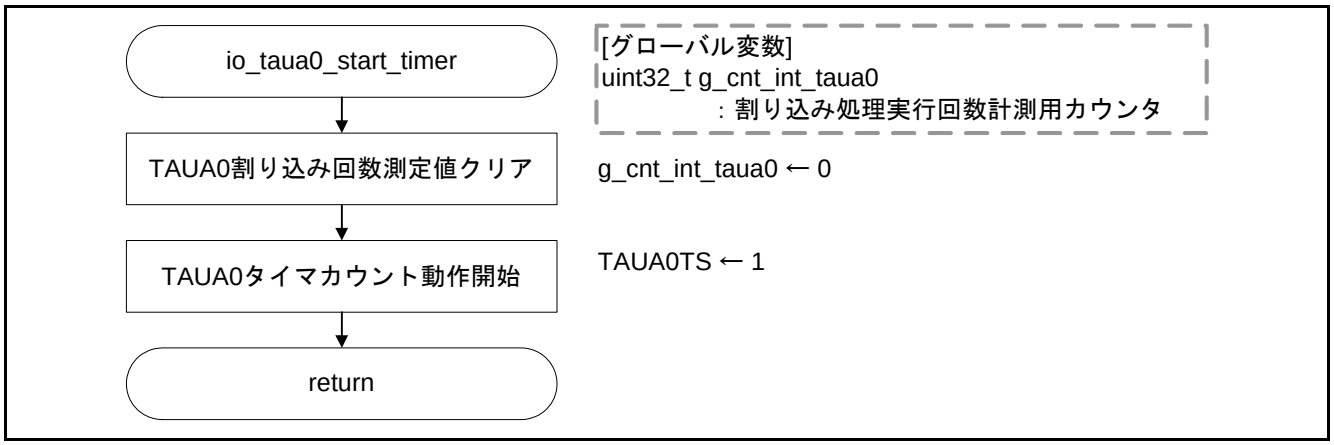

図4.14 TAUA0 タイマカウント動作開始処理

4.7.13 TAUA0 タイマカウント動作停止処理

図 4.15に TAUA0 タイマカウント動作停止処理のフローチャートを示します。

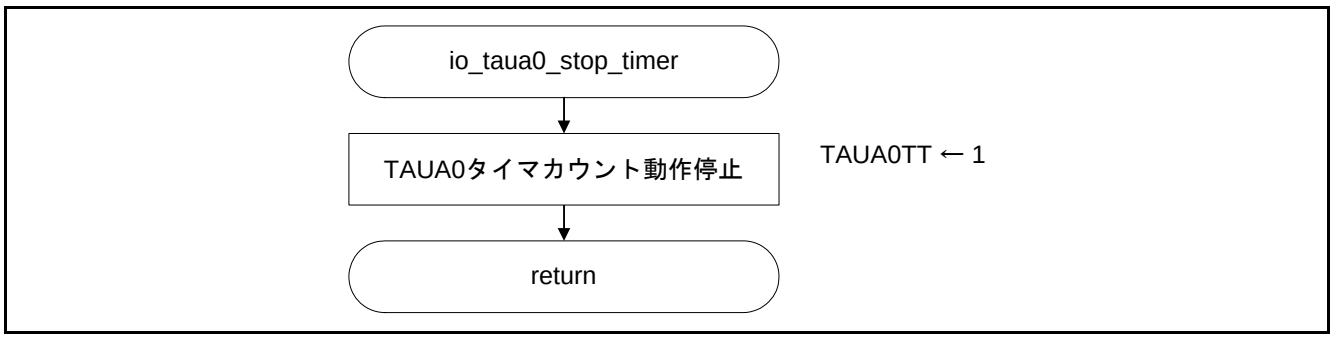

図4.15 TAUA0 タイマカウント動作停止処理

4.7.14 TAUA0 タイマカウント取得処理

図 4.16に TAUA0 タイマカウント取得処理のフローチャートを示します。

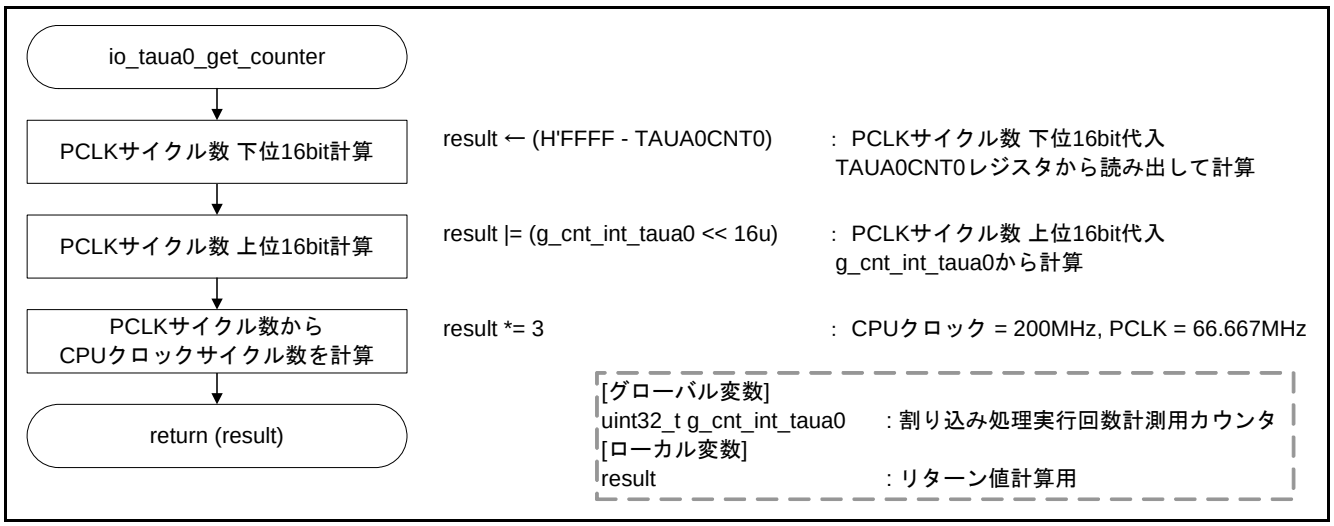

図4.16 TAUA0 タイマカウント取得処理

4.7.15 数学関数ライブラリ速度評価処理

図 4.17に 数学関数ライブラリ速度評価処理のフローチャートを示します。

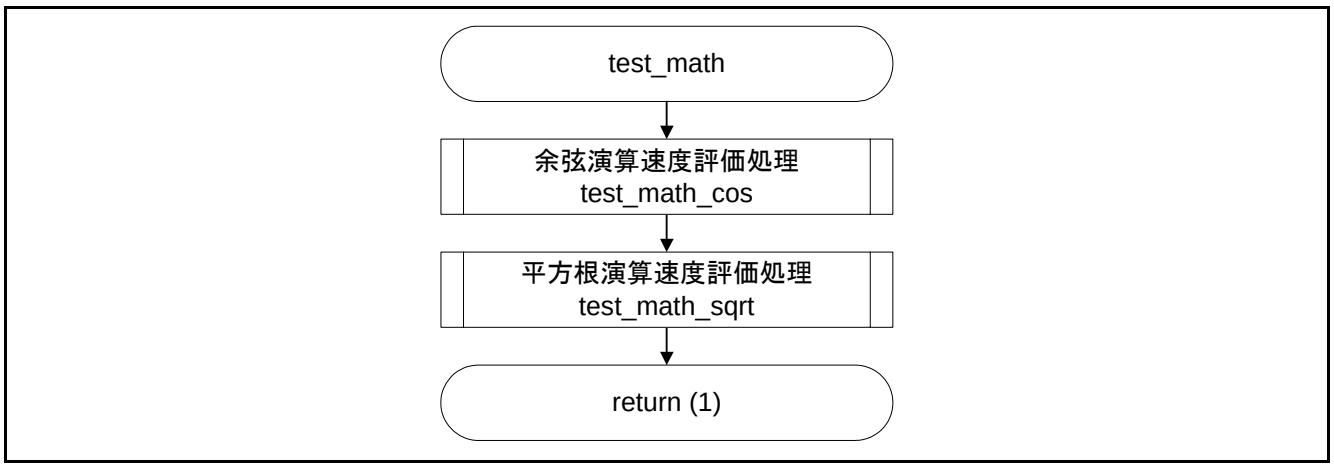

図4.17 数学関数ライブラリ速度評価処理

4.7.16 余弦演算速度評価処理

図 4.18に 余弦演算速度評価処理のフローチャートを示します。

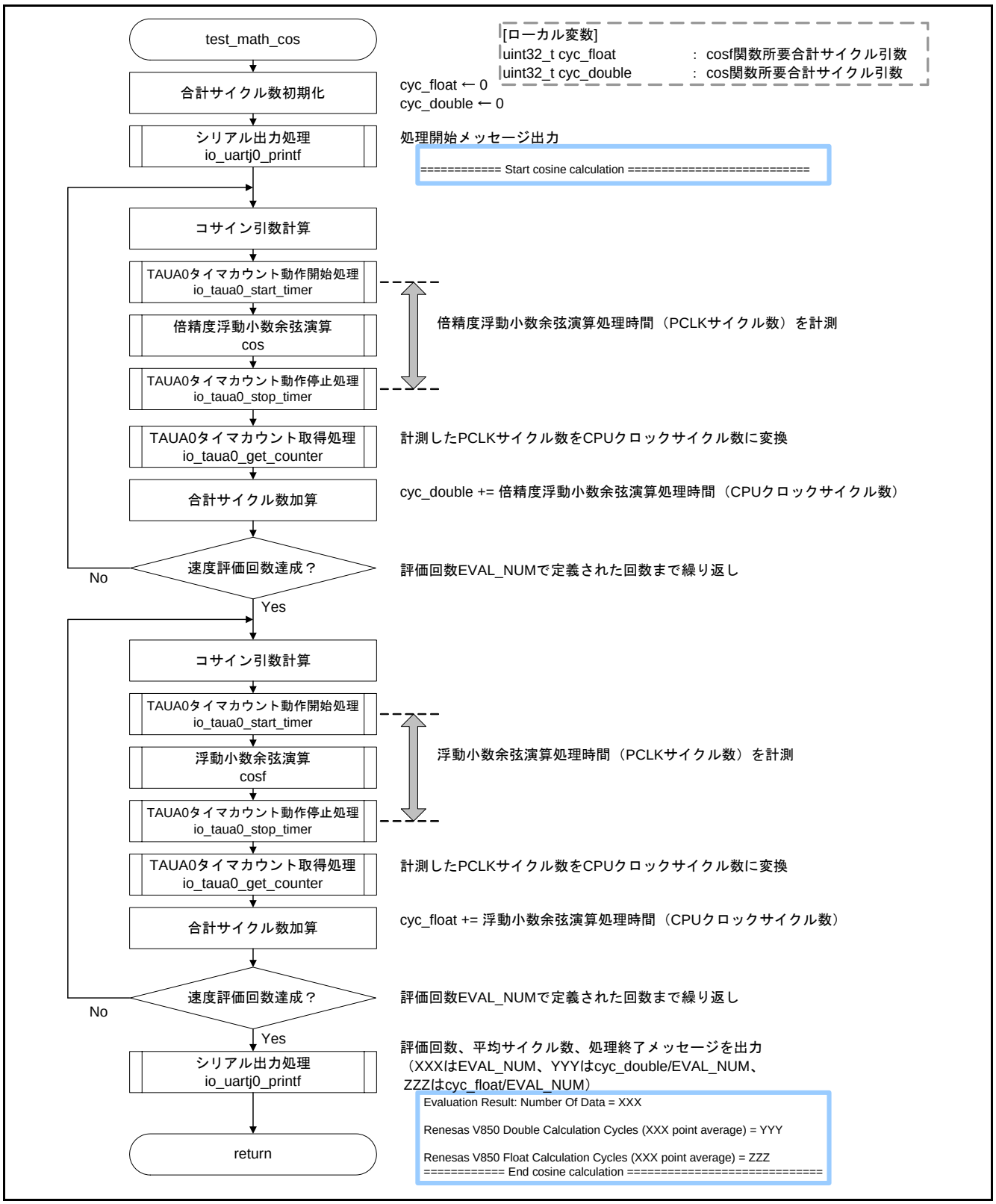

図4.18 余弦演算速度評価処理

4.7.17 平方根演算速度評価処理

図 4.19に 平方根演算速度評価処理のフローチャートを示します。

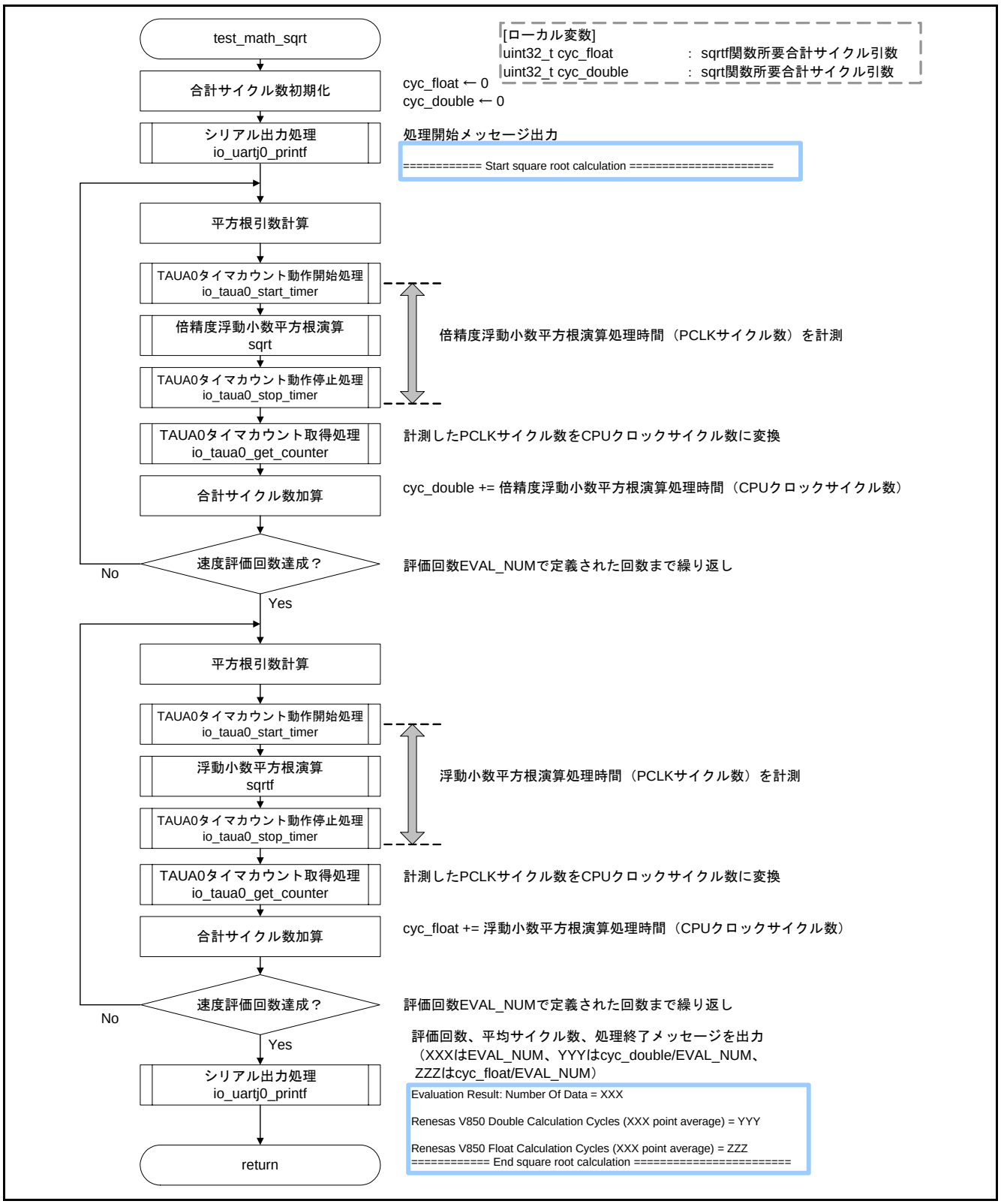

図4.19 平方根演算速度評価処理

4.7.18 ユーザ作成処理速度評価処理

図 4.20に ユーザ作成処理速度評価処理のフローチャートを示します。

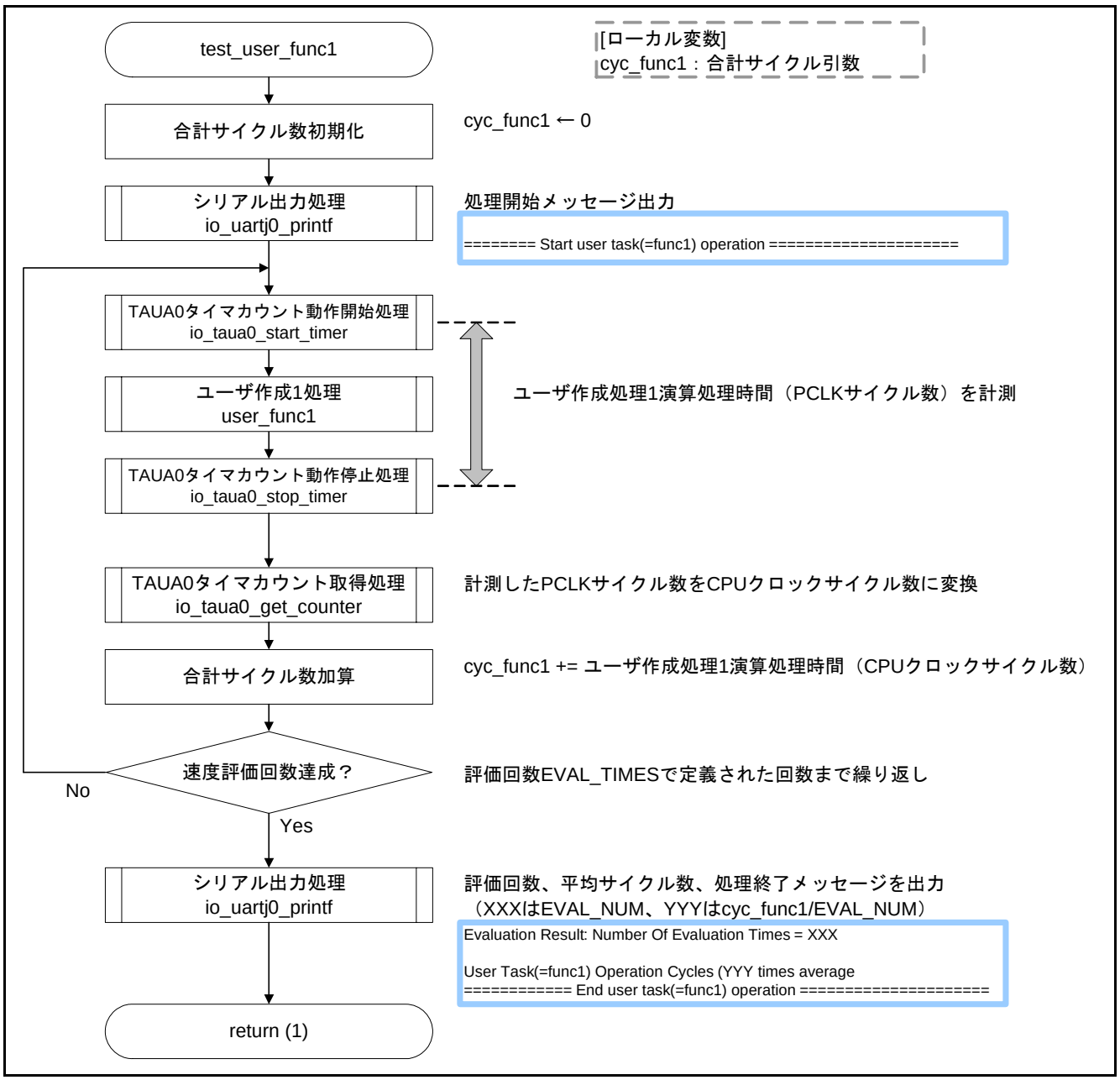

図4.20 ユーザ作成処理速度評価処理

4.7.19 ユーザ作成処理

図 4.21に ユーザ作成処理のフローチャートを示します。

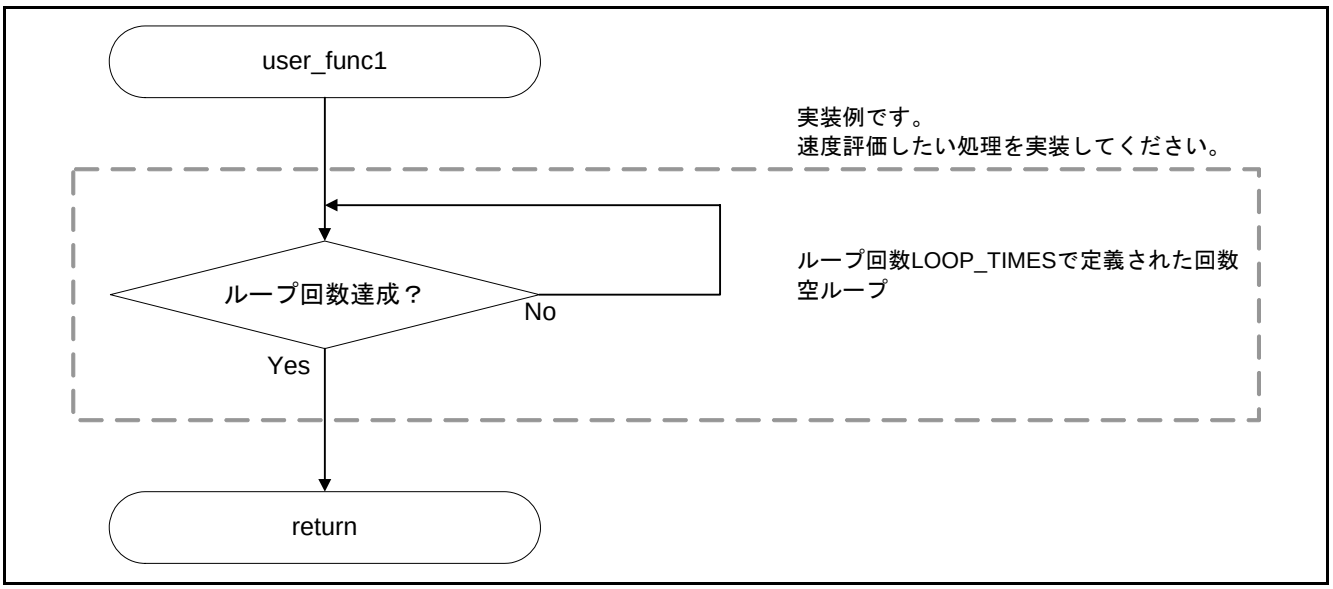

図4.21 ユーザ作成処理

5. 応用例

応用例としてサンプルコードによる性能評価の方法とユーザ作成タスクの追加方法を説明します。

5.1 性能評価

評価方法について説明します。

1. ホスト PC と V850E2/ML4 CPU ボードをシリアルケーブルで接続してください。なお、シリアルポートを搭載しない PC の場合、USB・シリアル変換ケーブルを使用してください。
2. ホスト PC 上のシリアル通信アプリケーションを起動し、通信設定を下記のようにしてください。

   転送レート 9600bps、データ長 8、パリティなし、1 ストップビット、フロー制御 Xon/Xoff
3. プログラムをロードして実行してください。
4. プログラム実行後、シリアル通信アプリケーションに下記が表示されます。

```
======================================================================
V850E2/ML4 Evaluation Program. Ver.1.00.00
Copyright (C) 2012 Renesas Electronics Corporation. ALL rights reserved
and Renesas Solutions Corporation. ALL rights reserved
======================================================================
>
```

図5.1 性能評価起動ログ

5. シリアル通信アプリケーションのコンソール上で“HELP”と入力すると評価コマンドの種類が表示されます。下記はサンプルコードに含まれる数学関数演算とユーザ作成タスク例として 100 回のループ処理（=user_func1）のみを組み込んだ場合の表示例です。必要に応じ、お客様が評価したいタスクと対応するコマンドを追加してください。

```
> HELP
commands help
MATH : math function test
FUNC1 : user task(=func1) test
>
```

図5.2 性能評価 HELP ログ

6. シリアル通信アプリケーションのコンソール上で“MATH”と入力すると、サンプルコードに含まれる V850 の組み込みライブラリ（=math.h）を使用した数学関数演算を行います。そして、演算に要したサイクル数が表示します。下記では、V850 の組み込みライブラリにより数学関数演算を 256 回繰り返した場合の平均値を表示しています。

```
> MATH
============ Start cosine calculation ============================
Evaluation Result: Number Of Data = 256

Renesas V850 Double Calculation Cycles (256 point average) = 341

Renesas V850 Float Calculation Cycles (256 point average) = 75
============ End cosine calculation ==============================

============ Start square root calculation ======================
Evaluation Result: Number Of Data = 256

Renesas V850 Double Calculation Cycles (256 point average) = 56

Renesas V850 Float Calculation Cycles (256 point average) = 38
============ End square root calculation ========================
>
```

図5.3 性能評価 MATH ログ

7. シリアル通信アプリケーションのコンソール上で”FUNC1”と入力すると、ユーザ作成タスク例としてサンプルコードに含まれる 100 回のループ処理を行います。そして、処理に要したサイクル数が表示されます。下記では、100 回のループ処理を 10 回繰り返した場合の平均値を表示しています。

```
> FUNC1
======== Start user task(=func1) operation =====================
Evaluation Result: Number Of Evaluation Times = 10

User Task(=func1) Operation Cycles (10 times average) = 1221
============ End user task(=func1) operation ====================
>
```

図5.4 性能評価 FUNC1 ログ

【注】 評価時の注意事項

サンプルコードでの測定はタイマで行っておりますが、CPU のタイマ開始と終了処理にも数サイクル数を要します。そこで、タスクを実行しない場合のサイクル数を測定し、タスクを実行した場合のサイクル数から減じる必要があります。

例えば、サンプルコード数学関数演算では、test_math.c の EMPTY_LOOP を有効（#undef EMPTY_LOOP をコメントアウト）にし実行すれば、タスクを実行しない場合のサイクル数が測定できます。

```
/* ---- 空ループ演算 ---- */
/* 演算の結果は空ループの演算結果との引き算 */
#define EMPTY_LOOP
//#undef EMPTY_LOOP /* この行をコメントアウトすると空ループの演算を行います。 */
```

図5.5 空ループ実装例

5.2 ユーザ作成タスクの追加

ユーザ作成タスクの追加方法を説明します。

1. 追加したいタスクを C 言語の関数（以下、user_func2 とする）として作成してください。
2. user_func2 を test_user.c に組み込んでください。組み込み方法は test_user.c の test_user_func1 の中の user_func1 と同様に、user_func2 の実行前に io_taua0_start_timer 関数、実行後に io_taua0_stop_timer 関数を実行するという方法で作成します。

```
        for (i = 0; i < EVAL_TIMES; i++)
        {
                io_taua0_start_timer();
#ifndef EMPTY_LOOP
                user_func2();
#endif
                io_taua0_stop_timer();

                cyc_func1 += io_taua0_get_counter();
        }
```

図5.6 ユーザ作成タスク評価処理組み込み例

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

ユーザーズマニュアル：ハードウェア

V850E2/ML4 ユーザーズマニュアル ハードウェア編 （R01UH0262JJ）
（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：開発環境

CubeSuite+ V1.00.00 統合開発環境 ユーザーズマニュアル コーディング編（CX コンパイラ）
（R20UT0554JJ）
CubeSuite+ V1.00.00 統合開発環境 ユーザーズマニュアル ビルド編（CX コンパイラ）
（R20UT0558JJ）
（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。）

ホームページとサポート窓口

ルネサス エレクトロニクスホームページ
http://japan.renesas.com

お問合せ先
http://japan.renesas.com/contact/

改訂記録	V850E2/ML4　アプリケーションノート　性能評価ソフトウェア

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2012.08.02	—	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、本文を参照してください。なお、本マニュアルの本文と異なる記載がある場合は、本文の記載が優先するものとします。

1. 未使用端子の処理

   【注意】未使用端子は、本文の「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。未使用端子は、本文「未使用端子の処理」で説明する指示に従い処理してください。

2. 電源投入時の処置

   【注意】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. リザーブアドレスのアクセス禁止

   【注意】リザーブアドレスのアクセスを禁止します。

   アドレス領域には、将来の機能拡張用に割り付けられているリザーブアドレスがあります。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。

4. クロックについて

   【注意】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。

5. 製品間の相違について

   【注意】型名の異なる製品に変更する場合は、事前に問題ないことをご確認下さい。

   同じグループのマイコンでも型名が違うと、内部メモリ、レイアウトパターンの相違などにより、特性が異なる場合があります。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。

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、お客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
2. 本資料に記載されている情報は、正確を期すため慎重に作成したものですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
3. 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に関し、当社は、何らの責任を負うものではありません。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. 当社製品を改造、改変、複製等しないでください。かかる改造、改変、複製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）、交通用信号機器、
   防災・防犯装置、各種安全装置等
   当社製品は、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（原子力制御システム、軍事機器等）に使用されることを意図しておらず、使用することはできません。たとえ、意図しない用途に当社製品を使用したことによりお客様または第三者に損害が生じても、当社は一切その責任を負いません。なお、ご不明点がある場合は、当社営業にお問い合わせください。
6. 当社製品をご使用の際は、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他の保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障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、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、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。お客様が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. 本資料に記載されている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。また、当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途に使用しないでください。当社製品または技術を輸出する場合は、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。
10. お客様の転売等により、本ご注意書き記載の諸条件に抵触して当社製品が使用され、その使用から損害が生じた場合、当社は何らの責任も負わず、お客様にてご負担して頂きますのでご了承ください。
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。

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その総株主の議決権の過半数を直接または間接に保有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。

ルネサス エレクトロニクス株式会社

■営業お問合せ窓口 http://www.renesas.com

※営業お問合せ窓口の住所・電話番号は変更になることがあります。最新情報につきましては、弊社ホームページをご覧ください。

ルネサス エレクトロニクス販売株式会社　〒100-0004　千代田区大手町2-6-2（日本ビル） (03)5201-5307

■技術的なお問合せおよび資料のご請求は下記へどうぞ。
総合お問合せ窓口：http://japan.renesas.com/contact/