RZ/T1グループ
パフォーマンスモニタ サンプルプログラム

R01AN4100JJ0110
Rev.1.10
2018.06.07

要旨

本アプリケーションノートは、RZ/T1 グループマイコンに搭載されている Cortex-R4 上で動作するパフォーマンスモニタ、およびキャッシュ操作機能の設定方法について説明します。

対象デバイス

RZ/T1

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

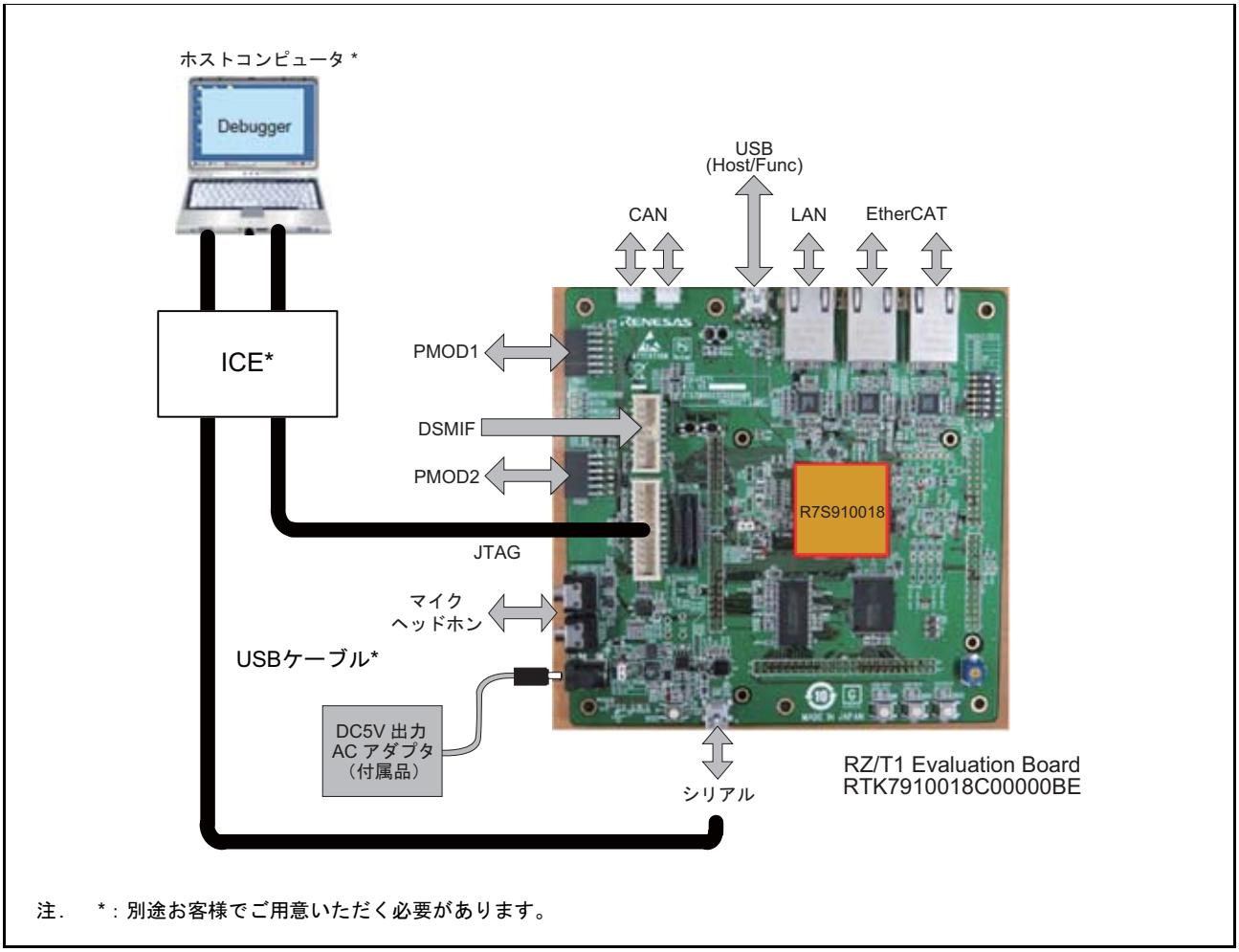
1.　仕様

RZ/T1 に搭載されている Cortex-R4 は、3 つのイベントカウントレジスタ、1 つのサイクルカウントレジスタを持つパフォーマンスモニタを内蔵しています。それぞれのイベントカウンタは個別に稼働でき、またカウンタごとにカウント対象とするイベントの割り当てが可能です。また、RZ/T1 の Cortex-R4 は命令キャッシュとデータキャッシュをそれぞれ 8Kbyte 内蔵しています。キャッシュラインサイズは 32byte となっており、各キャッシュはラインサイズ単位での Invalidate、Clean、Clean & Invalidate が可能です。

「表 1.1　主に使用する周辺機能と用途」、「図 1.1　動作環境」に本サンプルで使用する環境を示します。

表 1.1　　主に使用する周辺機能と用途

周辺機能	用途
クロック発生回路（CPG）	CPU クロックおよび低速オンチップオシレータで使用
FIFO 内蔵シリアルコミュニケーションインタフェース（SCIFA）	SCIFA の歩調同期式を使用し、RS-232C インタフェースによる COM ポート通信に使用
バスステートコントローラ（BSC）	サンプルで使用する SDRAM の設定に使用
コンペアマッチタイマ W（CMTW Unit0）	パフォーマンスモニタのサイクルカウントとの比較に使用



注.　*：別途お客様でご用意いただく必要があります。

図 1.1　　動作環境

2. 動作環境

本アプリケーションノートのサンプルプログラムは、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	RZ/T1グループ
動作周波数	CPUクロック（CPUCLK）：450MHz
動作電圧	電源電圧（I/O）：3.3V
統合開発環境	IARシステムズ製　Embedded Workbench® for Arm Version 8.20.2 Arm製　DS-5™ 5.26.2 Renesas製　e2studio 6.1.0
動作モード	SPIブートモード（シリアル・フラッシュ） 16ビットバスブートモード（NORフラッシュ）
ターミナルソフトの通信設定	• 通信速度：115200bps • データ長：8ビット • パリティ：なし • ストップビット長：1ビット • フロー制御：なし • 改行コード（受信）：CR • 改行コード（送信）：CR
使用ボード	• RZ/T1 Evaluation Board (RTK7910018C00000BE)
使用デバイス （ボード上で使用する機能）	• シリアルインタフェース（USB-Mini Bコネクタ J8） • NORフラッシュメモリ（CS0、CS1空間に接続） メーカ名：Macronix International Co., Ltd. 型名：MX29GL512FLT2I-10Q • シリアルフラッシュメモリ メーカ名：Macronix International Co., Ltd. 型名：MX25L51245G • SDRAM（CS2、CS3空間に接続） メーカ名：ISSI. 型名：S42S16320D
PC用USBシリアルポートドライバ	• RTK7910018C00000BE向け • RTK7910022C00000BR向け

3. 関連アプリケーションノート

本アプリケーションノートに関連するアプリケーションノートを以下に示します。併せて参照してください。

- RZ/T1 グループ初期設定アプリケーションノート（R01AN2554JJ）
- RZ/T1 グループ FIFO 内蔵シリアルコミュニケーションインタフェース（SCIFA）アプリケーションノート（R01AN2577JJ）
- RZ/T1 グループ CMTW & ELC サンプルプログラムアプリケーションノート（R01AN2600JJ）

注 . 本アプリケーションノートで記載しないレジスタに関しては、RZ/T1 グループ 初期設定アプリケーションノートで設定した値のまま使用します。

4.　周辺機能説明

動作モード、クロック発生回路（CPG）、FIFO 内蔵シリアルコミュニケーションインタフェース（SCIFA）、コンペアマッチタイマ W（CMTW）、リセット、汎用入出力ポートについての基本的な内容は、RZ/T1 グループ・ユーザーズマニュアルハードウェア編を参照してください。

パフォーマンスモニタ、キャッシュ操作機能の基本的な内容に関しましては Arm 社の Cortex-R4 テクニカルリファレンスマニュアルを参照してください。

5. ハードウェア説明

5.1 使用端子一覧

表 5.1 に使用端子と機能を示します。

表 5.1　使用端子と機能

端子名	入出力	内容
MD0	入力	動作モードの選択 MD0 = “L”, MD1 = “L”, MD2 = “L”（SPI ブートモード） MD0 = “L”, MD1 = “H”, MD2 = “L”（16 ビットバスブートモード）
MD1	入力	
MD2	入力	

6. ソフトウェア説明

本ソフトウェアには、パフォーマンスモニタ、キャッシュ操作機能を使用するためのドライバ、サイクル数を実時間に変換するユーティリティ関数を含んでいます。

6.1 機能概要

本ソフトウェアには、パフォーマンスモニタによりサイクル数、プロシージャからの復帰回数、データ読み出し命令の実行回数をカウントする機能が含まれています。また、CMTW によるカウントを平行して行い、パフォーマンスモニタによるサイクル数と CMTW によるカウントをそれぞれ実際の処理時間に変換して後述するシリアル出力機能によって出力します。

また、SDRAM に対してデータリードを行い、各キャッシュ操作機能をそれぞれ実行した後のメモリの状態を後述するシリアル出力機能によって出力します。

各処理の実行結果の出力として、FIFO 内蔵シリアルコミュニケーションインタフェース（SCIFA）の調歩同期式通信を用い、ホスト PC と RS-232 インタフェースの COM ポート通信を行い、ホスト PC 上のターミナルソフトウェアにて各サンプルプログラムの実行結果を出力します。

6.2 使用割り込み一覧

本サンプルプログラムでは割り込みを使用していません。

6.3 固定幅整数一覧

表 6.1 にサンプルプログラムで使用する固定幅整数を示します。サンプルコードで使用する固定幅整数は、標準ライブラリで定義されています。

表 6.1　サンプルプログラムで使用する固定幅整数

シンボル	内容
int8_t	8 ビット整数、符号あり
int16_t	16 ビット整数、符号あり
int32_t	32 ビット整数、符号あり
int64_t	64 ビット整数、符号あり
uint8_t	8 ビット整数、符号なし
uint16_t	16 ビット整数、符号なし
uint32_t	32 ビット整数、符号なし
uint64_t	64 ビット整数、符号なし

6.4 構造体 / 共用体一覧

本サンプルプログラムでは構造体 / 共用体を使用していません。

6.5　定数一覧

表 6.2、表 6.3 にサンプルプログラムで使用する定数を示します。

表6.2　パフォーマンスモニタサンプルプログラムで使用する設定用定数

定数名	設定値	内容
PMON_EVTCNT0 ～ PMON_EVTCNT2	(0u) ～ (1U)	設定を行うパフォーマンスモニタ番号の選択
PMON_EVT_SOFTINC	(0x00)	カウントイベント選択：ソフトウェアインクリメント
PMON_EVT_ICMISS	(0x01)	カウントイベント選択：命令キャッシュミス
PMON_EVT_DCMISS	(0x03)	カウントイベント選択：データキャッシュミス
PMON_EVT_DCACC	(0x04)	カウントイベント選択：データキャッシュアクセス
PMON_EVT_DRDEX	(0x06)	カウントイベント選択：データリード実行
PMON_EVT_DWDEX	(0x07)	カウントイベント選択：データライト実行
PMON_EVT_INSTEX	(0x08)	カウントイベント選択：アーキテクチャ的に実行された命令
PMON_EVT_DUALEX	(0x5E)	カウントイベント選択：アーキテクチャ的に実行された、デュアル発行された命令のペア
PMON_EVT_EXCENT	(0x09)	カウントイベント選択：取得された例外
PMON_EVT_EXCRET	(0x0A)	カウントイベント選択：アーキテクチャ的に実行された、例外からの復帰
PMON_EVT_CHCON	(0x0B)	カウントイベント選択：コンテキストIDの変更
PMON_EVT_CHPC	(0x0C)	カウントイベント選択：アーキテクチャ的に実行された、ソフトウェアによるPCの変更 ただし、例外からの復帰は除外されます
PMON_EVT_BIMM	(0x0D)	カウントイベント選択：アーキテクチャ的に実行された、B、BL、BLXイミディエート命令 実際に分岐が実行さない場合でもイベントが発生します
PMON_EVT_PROCRET	(0x0E)	カウントイベント選択：アーキテクチャ的に実行された、プロシージャからの復帰 ただし、例外によるものは除外されます
PMON_EVT_UNALIGN	(0x0F)	カウントイベント選択：アーキテクチャ的に実行された、アンアラインドアクセス
PMON_EVT_BMISSPRE	(0x10)	カウントイベント選択：予測に失敗したか、予測されなかった分岐
PMON_EVT_CYCLECNT	(0x11)	カウントイベント選択：サイクルカウント
CPUCKSEL_150M	(0x00u)	CPU周波数設定：150MHz
CPUCKSEL_300M	(0x01u)	CPU周波数設定：300MHz
CPUCKSEL_450M	(0x02u)	CPU周波数設定：450MHz
CPUCKSEL_600M	(0x03u)	CPU周波数設定：600MHz
TEST_TIME	(3u)	パフォーマンスモニタサンプルにおけるデータリード評価実行回数
TST_MEM_ADDR	(0x68000000)	パフォーマンスモニタサンプル、キャッシュ操作機能サンプルにおけるデータリード先アドレス（SDRAM）
TST_MEM_CNT	(1000000)	パフォーマンスモニタサンプルにおけるデータリード命令実行回数

表6.3　キャッシュ操作機能サンプルプログラムで使用する設定用定数

定数名	設定値	内容
CACHE_MAX_WAY	(0x3u)	キャッシュの最大way数
CACHE_WAY_SHIFT	(30u)	ビットシフト設定：[31:30]、way数指定用
CACHE_MAX_SET	(0x3Fu)	キャッシュの最大set数
CACHE_SET_SHIFT	(5u)	ビットシフト設定：[10:5]、set数指定用
CACHE_LINE_MASK	(0xFFFFFFE0u)	キャッシュ操作対象アドレス値マスク用
CACHE_LINE_SIZE	(0x00000020u)	キャッシュラインサイズ値
TST_MEM_SIZE	(0x00000060u)	キャッシュ操作機能サンプルにおける操作対象メモリサイズ（3キャッシュライン）
DUMP_ALIGN_MASK	(0xFFFFFFF0u)	アドレス値のアラインマスク用
DUMP_INCR_SIZE	(0x00000010u)	キャッシュ操作機能サンプルにおけるメモリダンプ時、ダンプ値の1回あたりの出力Byte数

6.6　変数一覧

本サンプルプログラムでは static 変数を使用していません。

6.7　関数一覧

表 6.4 に各サンプルプログラムで使用する共用関数、表 6.5 にパフォーマンスモニタサンプルプログラムで使用する関数、表 6.6 にキャッシュ操作機能サンプルプログラムで使用する関数の一覧をそれぞれ示します。

表6.4　共用関数一覧

関数名	概要
main	サンプルプログラムのメイン関数

表6.5　パフォーマンスモニタサンプル用関数一覧

関数名	概要
R_PMON_Open	パフォーマンスモニタ初期設定関数
R_PMON_Close	パフォーマンスモニタ終了設定関数
R_PMON_Start	イベントカウント開始関数
R_PMON_Stop	イベントカウント停止関数
R_PMON_StopAll	全カウンタの停止関数
R_PMON_SelectCount	カウンタ選択関数
R_PMON_GetCurCount	カウント取得関数
sample_pmon	パフォーマンスモニタサンプルのメイン関数
exec_tst_mem_dword_read	Read 命令実行評価関数
cyclecnt_to_nanosec	サイクルカウントの実行時間への変換関数

表6.6　キャッシュ操作機能サンプル用関数一覧

関数名	概要
R_CACHE_Inval_I_All	全命令キャッシュの Invalidate 関数
R_CACHE_Inval_I	キャッシュライン単位での命令キャッシュ Invalidate 関数
R_CACHE_Clean_D_All	全データキャッシュの Clean 関数
R_CACHE_Inval_D_All	全データキャッシュの Invalidate 関数
R_CACHE_CleanInval_D_All	全データキャッシュの Clean & Invalidate 関数
R_CACHE_Clean_D	キャッシュライン単位でのデータキャッシュ Clean 関数
R_CACHE_Inval_D	キャッシュライン単位でのデータキャッシュ Invalidate 関数
R_CACHE_CleanInval_D	キャッシュライン単位でのデータキャッシュ Clean & Invalidate 関数
sample_cache	キャッシュ操作機能サンプルのメイン関数
write_incr_data	評価用データ Write 関数
dump_memory	メモリダンプ関数

6.8　関数仕様

以下にサンプルコードの関数仕様を示します。

6.8.1　共用関数

関数名 main	
概　要	サンプルプログラムのメイン関数です。
宣　言	int main (void)
説　明	サンプルプログラムのメイン処理です。 ボードの初期設定を行った後、sample_pmon()、sample_cache() を呼び出し、サンプルプログラムを実行します。
引　数	なし
リターン値	0
補　足	なし

6.8.2　パフォーマンスモニタサンプル用関数

関数名 R_PMON_Open	
概　要	パフォーマンスモニタのカウント開始に必要な設定を行います。
宣　言	void R_PMON_Open(void);
説　明	パフォーマンスモニタのカウントを開始する際に必要な設定を行います。 • ユーザモードでのパフォーマンスモニタへのアクセスを許可 • カウンタのリセット • カウンタの Enable
引　数	なし
リターン値	なし
補　足	なし

関数名 R_PMON_Close	
概　要	パフォーマンスモニタのカウント終了に必要な設定を行います。
宣　言	void R_PMON_Close(void);
説　明	パフォーマンスモニタのカウントを終了する際に必要な設定を行います。 • ユーザモードでのパフォーマンスモニタへのアクセスを禁止 • カウンタの Disable
引　数	なし
リターン値	なし
補　足	なし

関数名 R_PMON_Start		
概　要	イベントのカウントをスタートします。	
宣　言	void R_PMON_Start (int32_t evtch, int32_t event, uint32_t ini_cnt);	
説　明	イベントのカウントを開始します。	
引　数	int32_t evtch	制御対象に指定するカウンタ番号 PMON_EVTCNT0 ～ PMON_EVTCNT2 から選択してください。
	int32_t event	カウント対象イベント 表 6.2 のカウントイベント選択：xxx~ と記載されている定数から選択してください。
	uint32_t ini_cnt	カウント初期値
リターン値	なし	
補　足	引数 evtch、event に規定された定義値以外を指定した場合の動作は不定となります。	

関数名 R_PMON_Stop		
概　要	イベントのカウントをストップします。	
宣　言	void R_PMON_Stop(int32_t evtch);	
説　明	イベントのカウントを停止します。	
引　数	int32_t evtch	制御対象に指定するカウンタ番号 PMON_EVTCNT0 ～ PMON_EVTCNT2 から選択してください。
リターン値	なし	
補　足	引数 evtch に規定された定義値以外を指定した場合の動作は不定となります。	

関数名 R_PMON_StopAll	
概　要	全カウンタのイベントカウントをストップします。
宣　言	void R_PMON_StopAll(void);
説　明	全カウンタのイベントカウントを停止します。
引　数	なし
リターン値	なし
補　足	なし

関数名 R_PMON_SelectCount		
概　要	現在のカウンタ番号を指定します。	
宣　言	void R_PMON_SelectCount(int32_t evtch);	
説　明	現在のカウンタ番号を指定します。カウンタ制御を行う際、制御対象とするカウンタを指定するために使用します。	
引　数	int32_t evtch	制御対象に指定するカウンタ番号 PMON_EVTCNT0 ～ PMON_EVTCNT2 から選択してください。
リターン値	なし	
補　足	引数 evtch に規定された定義値以外を指定した場合の動作は不定となります。	

関数名 R_PMON_GetCurCount		
概　要	現在のカウント値を取得します。	
宣　言	void R_PMON_GetCurCount(void);	
説　明	現在のカウント値を取得します。	
引　数	int32_t evtch	制御対象に指定するカウンタ番号 PMON_EVTCNT0 ～ PMON_EVTCNT2 から選択してください。
リターン値	イベントカウンタのカウント値	
補　足	取得対象となるカウンタは事前に R_PMON_SelectCount で選択されたカウンタとなります。 一度も R_PMON_SelectCount を実行していない場合、動作は不定となります。	

関数名 sample_pmon	
概　要	パフォーマンスモニタサンプルのメイン関数です。
宣　言	void sample_pmon (void);
説　明	パフォーマンスモニタサンプルのメイン関数です。各関数を実行し、その結果をシリアルターミナルに出力します。詳細は 6.8.2　パフォーマンスモニタサンプル用関数を参照してください
引　数	なし
リターン値	なし
補　足	なし

関数名 exec_tst_mem_dword_read	
概　要	リード命令実行評価関数です。
宣　言	void exec_tst_mem_dword_read (void);
説　明	リード命令実行回数取得のため、リード命令を 100 万回実行します。
引　数	なし
リターン値	なし
補　足	なし

関数名 cyclecnt_to_nanosec		
概　要	サイクル数カウント値を時間に変換する関数です。	
宣　言	void cyclecnt_to_nanosec (uint32_t cycle_cnt);	
説　明	パフォーマンスモニタのサイクル数カウント値を nsec 単位の値に変換します。	
引　数	uint32_t cycle_cnt	パフォーマンスモニタのサイクル数カウント値
リターン値	なし	
補　足	なし	

6.8.3　キャッシュ操作機能サンプル用関数

関数名 R_CACHE_Inval_I_All		
概　要	命令キャッシュをすべて Invalidate します。	
宣　言	void R_CACHE_Inval_I_All (void);	
説　明	すべての命令キャッシュを Invalidate します。	
引　数	なし	
リターン値	なし	
補　足	なし	

関数名 R_CACHE_Inval_I		
概　要	キャッシュライン単位で命令キャッシュを Invalidate します。	
宣　言	void R_CACHE_Inval_I (uint32_t sta_addr, uint32_t size);	
説　明	キャッシュライン単位で命令キャッシュを Invalidate します。	
引　数	uint32_t sta_addr	キャッシュ invalidate 開始アドレス
	uint32_t size	キャッシュ Invalidate サイズ
リターン値	なし	
補　足	引数 sta_addr にキャッシュラインサイズの境界に一致しないアドレスを指定した場合、指定したアドレスを含むキャッシュラインから Invalidate が開始します。 また、引数 size にキャッシュラインサイズの倍数でないサイズを指定した場合、指定した範囲を含むキャッシュラインまでの範囲で Invalidate します。	

関数名 R_CACHE_Clean_D_All	
概　要	データキャッシュをすべて Clean します。
宣　言	void R_CACHE_Clean_D_All (void);
説　明	すべてのデータキャッシュを Clean します。
引　数	なし
リターン値	なし
補　足	なし

関数名 R_CACHE_Inval_D_All	
概　要	データキャッシュをすべて Invalidate します。
宣　言	void R_CACHE_Inval_D_All (void);
説　明	すべてのデータキャッシュを Invalidate します。
引　数	なし
リターン値	なし
補　足	なし

関数名 R_CACHE_CleanInval_D_All		
概　要	データキャッシュをすべて Clean & Invalidate します。	
宣　言	void R_CACHE_CleanInval_D_All (void);	
説　明	すべてのデータキャッシュを Clean & Invalidate します。	
引　数	なし	
リターン値	なし	
補　足	なし	

関数名 R_CACHE_Clean_D		
概　要	キャッシュラインサイズ単位でデータキャッシュを Clean します。	
宣　言	void R_CACHE_Clean_D (uint32_t sta_addr, uint32_t size);	
説　明	キャッシュラインサイズ単位でデータキャッシュを Clean します。	
引　数	uint32_t sta_addr	キャッシュ Clean 開始アドレス
	uint32_t size	キャッシュ Clean サイズ
リターン値	なし	
補　足	引数 sta_addr にキャッシュラインサイズの境界に一致しないアドレスを指定した場合、指定したアドレスを含むキャッシュラインから Clean が開始します。 また、引数 size にキャッシュラインサイズの倍数でないサイズを指定した場合、指定した範囲を含むキャッシュラインまでの範囲で Clean します。	

関数名 R_CACHE_Inval_D		
概　要	キャッシュラインサイズ単位でデータキャッシュを Invalidate します。	
宣　言	void R_CACHE_Inval_D (uint32_t sta_addr, uint32_t size);	
説　明	キャッシュラインサイズ単位でデータキャッシュを Invalidate します。	
引　数	uint32_t sta_addr	キャッシュ Invalidate 開始アドレス
	uint32_t size	キャッシュ Invalidate サイズ
リターン値	なし	
補　足	引数 sta_addr にキャッシュラインサイズの境界に一致しないアドレスを指定した場合、指定したアドレスを含むキャッシュラインから Invalidate が開始します。 また、引数 size にキャッシュラインサイズの倍数でないサイズを指定した場合、指定した範囲を含むキャッシュラインまでの範囲で Invalidate します。	

関数名 R_CACHE_CleanInval_D		
概　要	キャッシュラインサイズ単位でデータキャッシュを Clean & Invalidate します。	
宣　言	void R_CACHE_CleanInval_D_All (uint32_t sta_addr, uint32_t size);	
説　明	キャッシュラインサイズ単位でデータキャッシュを Clean & Invalidate します。	
引　数	uint32_t sta_addr	キャッシュ Clean & Invalidate 開始アドレス
	uint32_t size	キャッシュ Clean & Invalidate サイズ
リターン値	なし	
補　足	引数 sta_addr にキャッシュラインサイズの境界に一致しないアドレスを指定した場合、指定したアドレスを含むキャッシュラインから Clean & Invalidate が開始します。また、引数 size にキャッシュラインサイズの倍数でないサイズを指定した場合、指定した範囲を含むキャッシュラインまでの範囲で Clean & Invalidate します。	

関数名 sample_cache	
概　要	キャッシュ操作機能サンプルのメイン関数です。
宣　言	void sample_cache(void);
説　明	キャッシュ操作機能サンプルのメイン関数です。各関数を実行し、その結果をシリアルターミナルに出力します。詳細は 6.8.3　キャッシュ操作機能サンプル用関数を参照してください
引　数	なし
リターン値	なし
補　足	なし

関数名 write_incr_data		
概　要	キャッシュ操作評価用データの書き込み関数です。	
宣　言	void write_incr_data (uint32_t addr, uint32_t size);	
説　明	キャッシュ操作を実行する準備として、インクリメントデータの write を行います。	
引　数	uint32_t addr	データ書き込み先アドレス
	uint32_t size	書き込みデータサイズ
リターン値	なし	
補　足	なし	

関数名 dump_memory		
概　要	メモリの内容をダンプします。	
宣　言	void dump_memory (uint32_t addr, uint32_t size);	
説　明	指定されたメモリの内容を読み込み、シリアルターミナルへ出力します。	
引　数	uint32_t addr	データダンプ元アドレス
	uint32_t size	ダンプデータサイズ
リターン値	なし	
補　足	なし	

6.9　サンプルプログラムの関数フロー

6.9.1　メイン関数

図 6.1 にメイン関数のフローを示します。

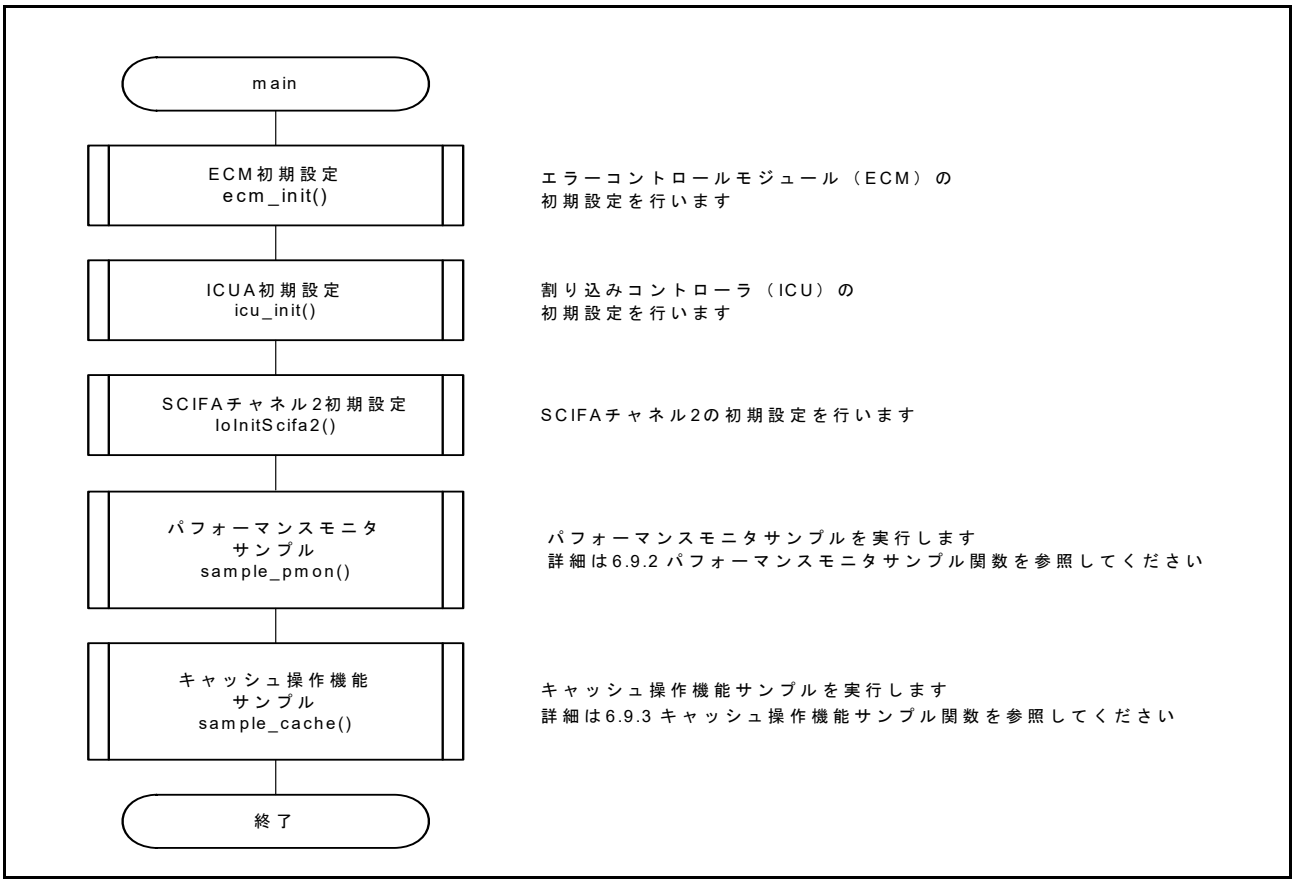


図 6.1　main のフロー

6.9.2　パフォーマンスモニタサンプル関数

図 6.2 に sample_pmon のフローを示します。

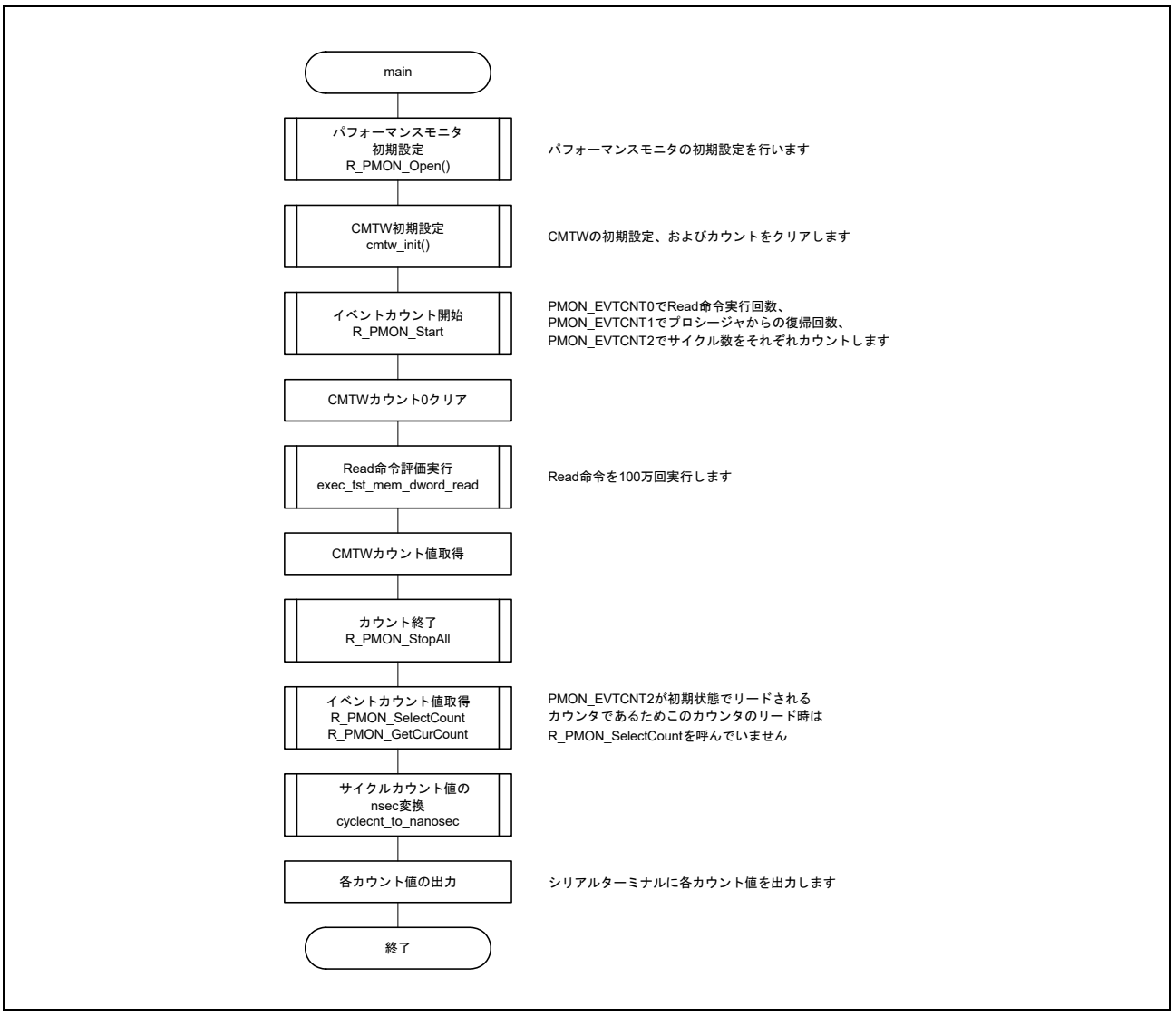


図 6.2　sample_pmon のフロー

6.9.3　キャッシュ操作機能サンプル関数

図 6.3 に sample_cache のフローを示します。

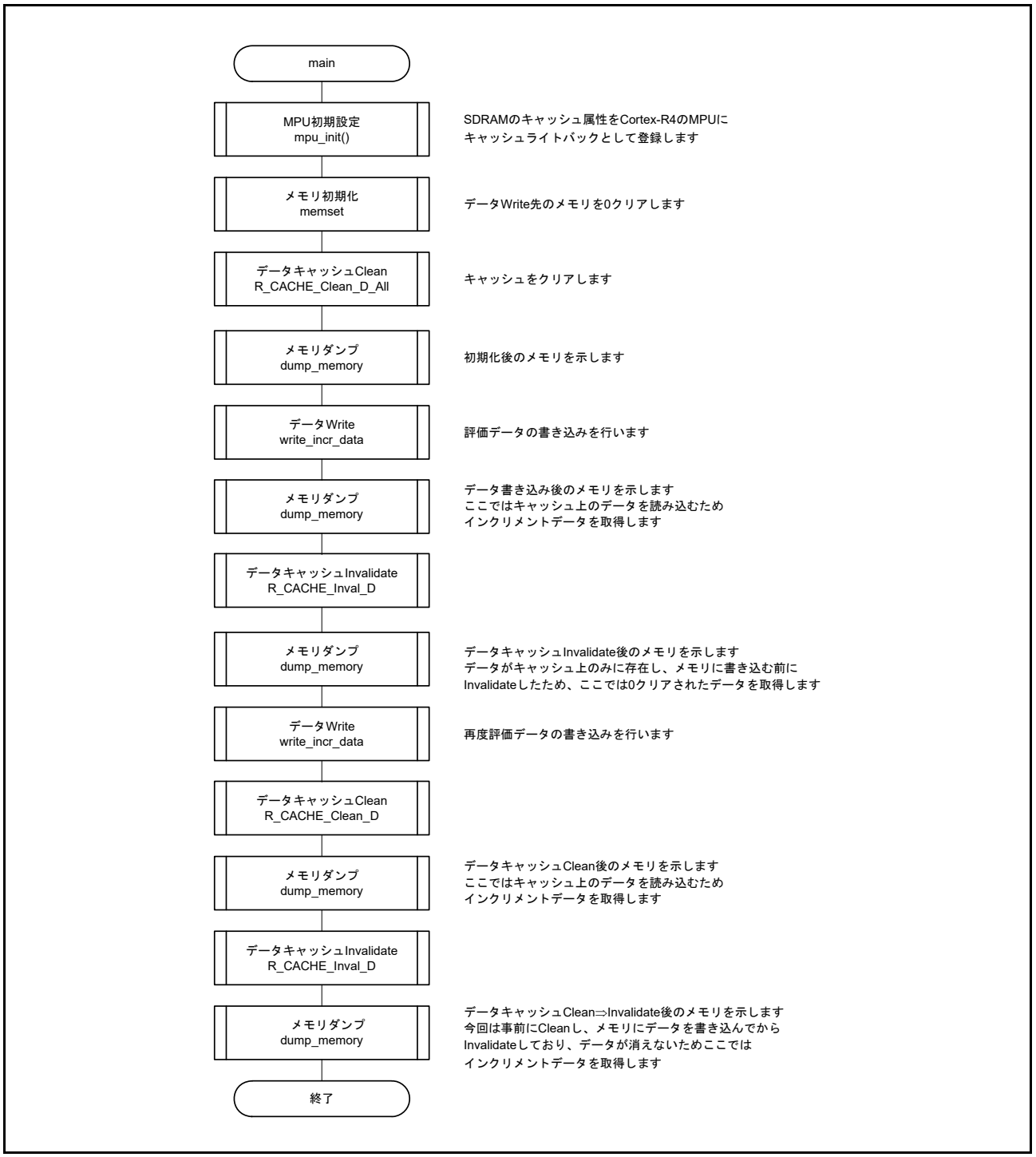


図 6.3　sample_cache のフロー

6.10　サンプルプログラム動作説明

PC のターミナルソフトウェア上でのサンプルプログラムの動作を示します。

6.10.1　プロジェクト設定

開発環境となる EWARM/DS-5/e2studio 上で使用されるプロジェクト設定については、RZ/T1 グループ 初期設定アプリケーションノートに記載しています。

6.10.2　使用準備

本サンプルプログラムでは、PC との通信動作を行いますので、その実行準備を説明します。

(1) ホスト PC にてターミナルソフトを起動し、シリアルポートの設定を次のように設定します。(Tera Term で COM3 使用の場合)

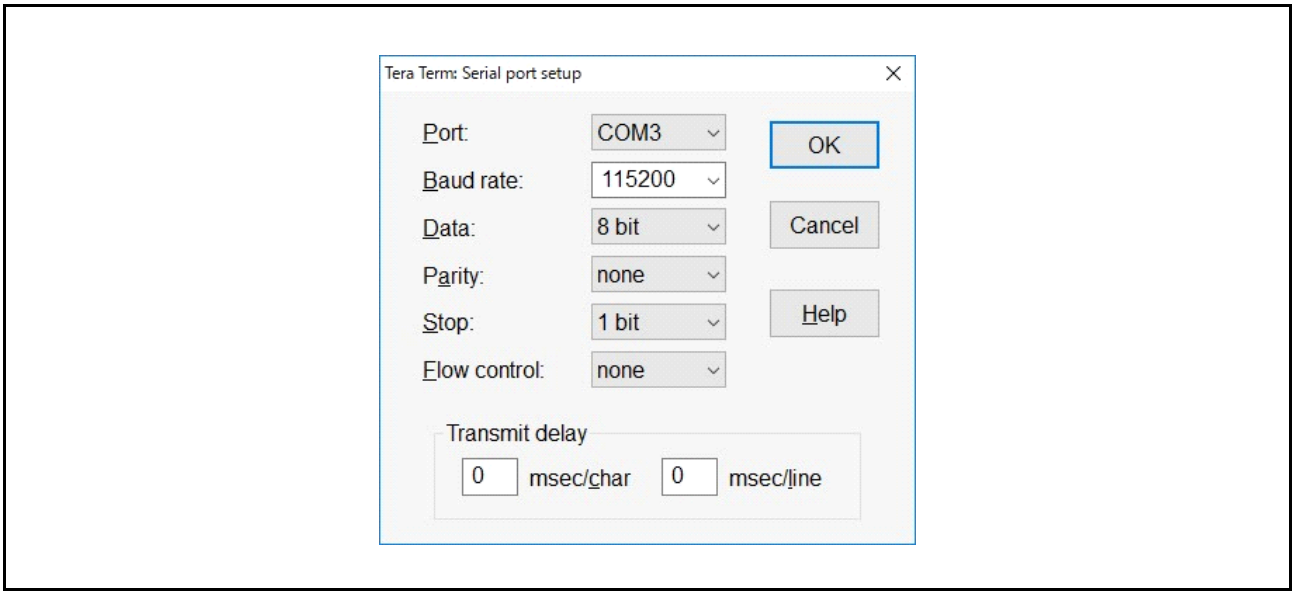


図 6.4　シリアルポートの設定

(2) サンプルプログラムを実行するとシリアル通信を開始し、下図に示すようにサンプルプログラムの実行結果がターミナルソフトに表示されます。

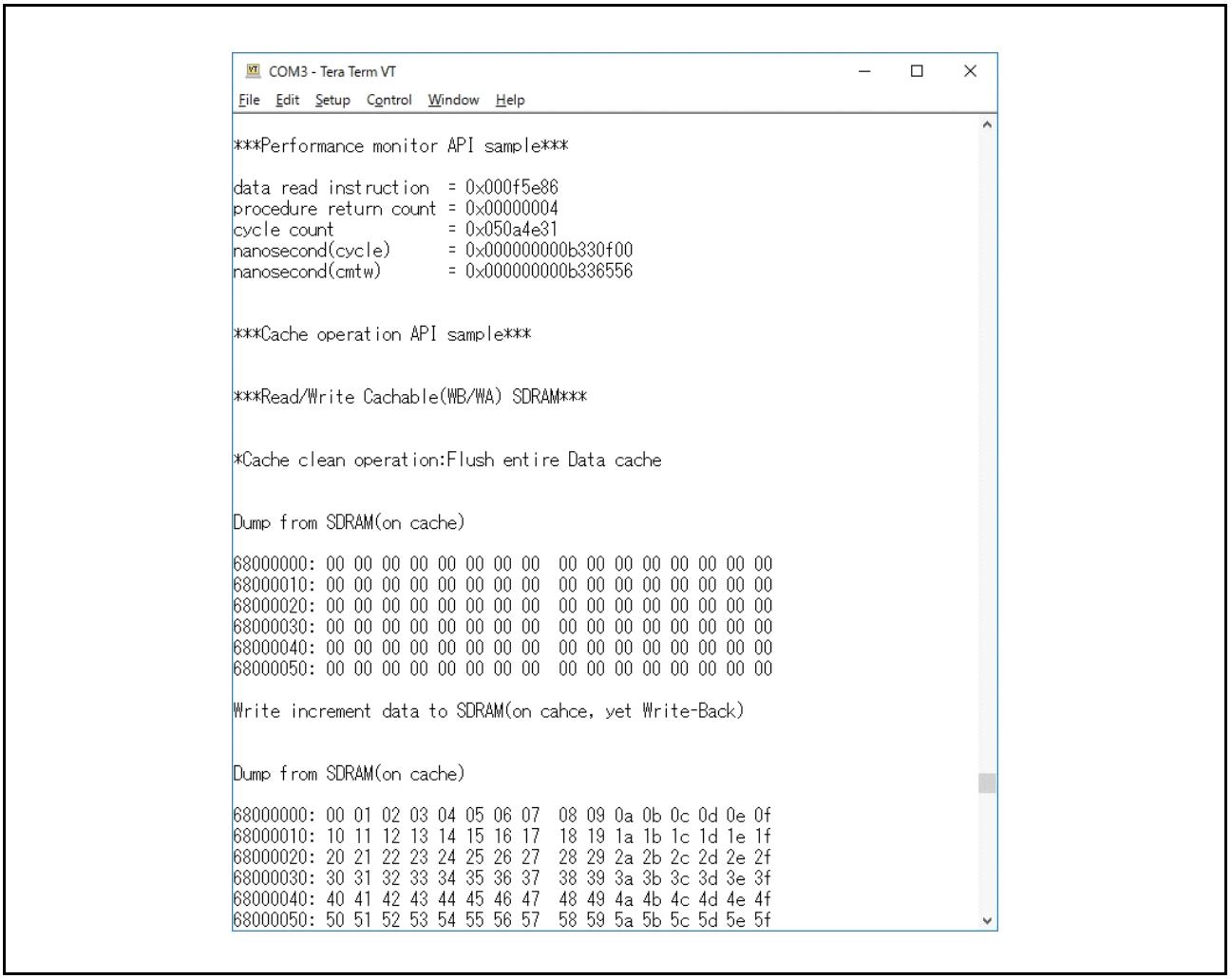


図 6.5　　サンプルプログラム実行後のターミナルソフトの表示

7.　サンプルプログラム

サンプルプログラムは、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。

8. 参考ドキュメント＆開発環境

- ユーザーズマニュアル：ハードウェア
  RZ/T1 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。）

- RZ/T1 Evaluation Board RTK7910018C00000BE 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。）

- 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。）

- IAR 統合開発環境
  IAR Embedded Workbench® for Arm に関しては、IAR システムズホームページから入手してください。
  （最新版を IAR システムズホームページから入手してください。）

- Arm 統合開発環境
  Arm Compiler toolchain、Arm DS-5 等に関しては、Arm ホームページから入手してください。
  （最新版を Arm ホームページから入手してください。）

- Renesas 統合開発環境
  e2studio に関しては、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。）
  コンパイラ＆ツールチェーン（GNUARM-NONE）に関しては、GNU TOOLS & SUPPORT Web サイト（https://gcc-renesas.com/）から入手してください。
  （最新版を GNU TOOLS & SUPPORT Web サイト（https://gcc-renesas.com/）から入手してください。）

9.　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	パフォーマンスモニタ サンプルプログラム アプリケーションノート

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2017.12.05	—	初版発行
1.10	2018.06.07	2. 動作環境	
		4	表2.1 動作環境　統合開発環境の内容変更
		4. 周辺機能説明	
		6	ARM→Armに変更
		8. 参考ドキュメント&開発環境	
		25	ARM→Armに変更

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. 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（予約領域）のアクセス禁止
   【注意】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、将来の機能拡張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。
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、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、内部 ROM、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。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)

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 エレクトロニクス株式会社 〒135-0061 東京都江東区豊洲3-2-24（豊洲フォレシア）

■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：https://www.renesas.com/contact/