Vol.2017-SLDM-181 No.40
Vol.2017-EMB-46 No.40
2017/11/8

安全プロセッサのための高信頼性RTOSの開発及びその評価

斎藤 弘樹[1]　富岡 洋一[1]　北道 淳司[1]

概要：近年，組込みシステムでは技術発展に伴いシステムの高度化・複雑化が進んでいる．本稿ではシステムの故障検知機能を持つプロセッサに対する RTOS の開発について述べる．そのようなシステムにおける動作中の故障や誤動作は人命や経済などに対し悪響を及ぼす可能性があり，システムの信頼性向上が要求される．本稿ではシステムの故障検知機能を持つプロセッサに対する RTOS の開発について述べる．研究対象とした MCU TMS570LS3137 はリアルタイムアプリケーション向けの MCU であり，システムの故障検知機能を持つ．開発した RTOS は TOPPERS/HRP2 kernel をベースに高信頼性が要求されるシステム向けの機能を拡張した．拡張した機能はメモリ保護機能や時間保護機能，故障検知機能である．また，開発した RTOS の初期化時にシステムのセルフテスト機能を組込み，システムの故障検知機能を評価した．さらに開発した RTOS でのタスクのディスパッチや共有資源の獲得などの時間計測プログラムを作成し RTOS の性能評価を行った．

Developmet of a Highly Reliable Real-Time Operating System for a Safety Processor and it's Evaluation

Saito Hiroki[1]　Tomioka Yoichi[1]　Kitamichi Junji[1]

Abstract: Presently, the sophistication and complexity of embedded systems are increasing as technology evolves. In embedded systems, failures and malfunctions are including the loss of human lives and economic disasters. Therefore, most of embedded systems are required highly reliable. In this paper, we report a development of RTOS for a processor with fault-detection function. The target MCU is TMS570LS3137. This MCU has a fault-detection function for real-time applications. The developed RTOS is extended for safety functions in the TMS570LS3137 for TOPPERS/HRP2 kernel. The extended functions are a memory protection function, a time protection function and a fault-detection function. The self-test function is incorporated in a startup routine of the developed RTOS and we evaluate the fault-detection function of the system. And, we develop application programs to measure time sush as task dispatching and acquisition of resources and the performance of RTOS is evaluated.

1. はじめに

組込みシステムは，車載機器，電力供給システムあるいは医療機器など多くのリアルタイムアプリケーションを支えている．また，技術発展に伴いシステムの高度化・複雑化が進んでおり故障や誤動作は人命や経済に悪影響を及ぼす原因になる．したがってシステム自体の信頼性向上が必要となる．現在システムの信頼性を向上させる手法としてロックステッププロセッサや Triple Modular Redundant(TMR) アーキテクチャなどプロセッサに対するフォルトトレランントデザイン [1] などの採用がある．ロックステッププロセッサ [1] の場合，2 つのプロセッサの演算結果を比較し結果の一致不一致を判断することでプロセッサの故障を判定することが出来る．TMR では 3 つあるプロセッサの結果を比較することで 1 つのプロセッサに故障があったとしても動作し続けることが出来る．他の方法として，故障が発生した場合故障検知機能や故障診断機能を用いた解析方法 [2] がある．故障検知機能を持つアプリケーションの多くは故障原因の特定のためにパラメータ推移法 [2] やオブザーバベース法 [2] を使用している．

[1] 会津大学大学院 コンピュータ理工学研究科
The Uniersity of Aizu, Graduate School of Computer Science and Engineering

ソフトウェアによる解決策として Real-Time Operating System(RTOS)[3] が効果的である．RTOS を用いる事でアプリケーションの開発期間の短縮やタスク，リソースおよび時間の管理が容易になる．また多種多様な RTOS が存在するため要求仕様に適した RTOS を使用することが可能である．しかし RTOS に対する故障検知機能の実装は不十分であるため高信頼性 RTOS の開発が信頼性向上に繋ると考えられる．本稿では故障検知機能やメモリ保護機能を持つ Micro Controller Unit(MCU) のための高信頼性 RTOS の開発について述べる．対象 MCU は Texas Instruments 社の TMS570LS3137[4] であり，ロックステップデュアル CPU やメモリの不正アクセスを防ぐ Memory Protection Unit(MPU)，MCU の故障を検知する Error Signaling Module(ESM) などを持つ．現在 TMS570LS3137 に対していくつかの RTOS が開発されている (FreeRTOS，SAFERTOS[5]，μC/OS-Ⅱ[6] 等)．FreeRTOS や SAFERTOS ではタスクを生成する Application Program Interface(API) に対して開発者がメモリ領域の開始番地やメモリサイズ，アクセス属性を設定する必要がある．なおカーネルはメモリ配置を静的に行わない．したがって開発者は MPU のアライメント制約を満すような設計や設定情報の決定が必要となる．μC/OS-Ⅱ でも同様，開発者が MPU の領域レジスタに対するメモリ領域の開始番地やメモリサイズ，アクセス属性を設定する必要がある．またこのカーネルではメールボックス機能が実装してあり，ポインタを引数とするタスク間の通信を行うためアクセス出来ない領域に間接的にアクセス出来る可能性がありメモリの不正アクセスが発生する可能性がある．

開発した RTOS は TOPPERS/HRP2 kernel[7] をベースとしている．この RTOS は高信頼性組込みシステムのために開発されメモリ保護機能やオーバランハンドラ機能を持つ．またタスクは静的にメモリ配置されコンフィギュレータによりメモリ保護に要求される情報（リージョン番号，開始番地，メモリサイズ等）が定義される．したがって開発者は MPU のアライメント制約について考えなくてよい．今回 TOPPERS/HRP2 kernel に対して，TMS570LS3137 の持つ故障検知機能の拡張を行い，RTOS の実装を行った．またシステム評価のために提案されている RTOS の性能評価方法 [3]，[8] をベースに評価プログラムの開発を行った．プログラムを用いて，故障検知機能や RTOS 機能の性能ついて評価し開発した RTOS を搭載したシステムの信頼性や性能について述べる．

2. TOPPERS/HRP2 KERNEL

ベースとする TOPPERS(Toyohashi OPen Platform for Embedded Real-Time Systems)/HRP2 kernel[7] について述べる．HRP2(High Reliable system Profile 2) kerenl は特に信頼性・安全性の要求が高い組込みシステム向けとして提案され，アクセス保護機能や時間保護機能など保護機能を強化したオープンソースのカーネルである．

2.1 アクセス保護機能

HRP2 kernel ではタスクに対して配置するメモリ領域，アクセス属性を指定することにより指定されたタスク以外からの不正アクセスを防ぐことが出来る．またカーネルがアクセス保護の対象とするメモリ領域をメモリオブジェクトと呼び，先頭アドレス，メモリサイズ，アクセス保護属性によって構成される．

このカーネルにおけるメモリ保護機能は実行中に発生する不正メモリアクセスを検出する．この機能では各メモリオブジェクトに対して読出し，書込み，実行の操作がどの保護ドメインから許可されているかをそれぞれ設定する．保護ドメインは保護機能を提供するために用いるカーネルオブジェクト（カーネルが管理対象とするソフトウェア）の集合である．カーネルではサービスコールと呼ばれるインターフェースがある．ポインタを使った間接的な不正アクセスを防ぐために TOPPERS の拡張サービス保護機能を用いることで下位のソフトウェアから上位のソフトウェアを呼び出す場合アクセス権限に応じて実行するか不正アクセスエラーを返すかの判断を行う．これによりソフトウェアが特権モードで実行された場合，ハードウェアで実現できないアクセス管理を可能にする．

2.2 時間保護機能

オーバランハンドラ機能はタスクが使用したプロセッサ時間が指定時間を越えた場合に起動される機能である．この機能はオーバランハンドラの動作状態と残りプロセッサ時間から構成されている．この機能により時間制約の厳しいタスクに対して制約を守れているかどうかをユーザが認識する事が出来る．

3. TMS570LS3137 MCU

TMS570LS3137[4] は ARM Cortex-R4F プロセッサを搭載しておりリアルタイム制約の厳しいアプリケーション向けの MCU である．Flash や RAM が ECC 対応であるなどメモリに信頼性向上のための機能が実装してある．またペリフェラルにパリティチェック機能が搭載されている．ARM Cortex-R4F プロセッサ及びペリフェラル，ターゲットボードについて述べる．

3.1 ARM Cortex-R4F プロセッサ

対象 MCU には ARM Cortex-R4F プロセッサが搭載されている．このプロセッサはロックステップで動作するデュアル CPU やメモリ保護ユニットなどリアルタイム性の要求が高いアプリケーション向けの機能を持つ．

このプロセッサは Master CPU と Checker CPU 及び CPU Compare Module for Cortex-R4F(CCM-R4F) から構成されている．Master CPU の演算結果は CPU の出力

値と CCM-R4F の入力値として使用される．一方 Checker CPU の演算結果は CCM-R4F の入力値として使用される．CCM-R4F では 2 つの入力値を比較し結果が不一致だった場合，コンペアエラーを出力する．

メモリ保護ユニットは外部メモリに対してメモリ領域を分割し領域ごとに保護属性を設定することでアクセス保護を行う．MPU は 12 個のメモリ領域をサポートしている．またこのユニットは実アドレスに対して保護属性の設定を行うことが出来る．したがって外部メモリへのアクセス時に MMU のような仮想アドレスからのアドレス変換を必要としない．これによりアドレス変換ミスが発生しないためメモリアクセスのリアルタイム性や信頼性を保持することが可能となる．ARM Cortex-R4F プロセッサにおけるメモリ保護ユニットの制御はシステム制御コプロセッサのレジスタを用いて行う．設定後 MPU を有効にすることでメモリ保護が行われる．

3.2 ペリフェラル

開発した RTOS においてサポートしたペリフェラルの説明を行う．

Real Time Interrupt(RTI) module はタイマ機能を提供するモジュールである．このモジュールには 2 つの 32bit カウンタが搭載されており，スケジューリングで必要なタイムベースを定義することが可能である．また OS 用の Tick を生成するために構成可能なコンペアレジスタがある．このレジスタはタイマカウンタの値と設定値が一致した場合割込みを発生させる．これにより周期的な Tick を発生させることが出来る．

Error Signaling Module(ESM) は MCU 上で発生する様々なエラーを管理する．これらのエラーは割当てられている深刻度に応じて処理される．特徴として 128 個のエラーチャンネルをサポートし，それぞれ 3 つのグループ（グループ 1，2，3）に分割される．グループ 1 は最も深刻度の低いエラーから構成されている．このグループは各ペリフェラルのパリティチェックエラーやメモリの修正可能エラーなどが分類されエラー時にマスク可能割込みを発生させる．発生させる割込み要求は通常割込み要求 Interrupt ReQuest(IRQ) か高速割込み要求 Fast Interrupt reQuest(FIQ) をユーザが指定する．グループ 2 は CPU コンペアエラーやメモリ修正不可能エラー等から構成されており，エラー時マスク不可能割込みを発生させる．グループ 3 は ECC メモリの修正不可能エラーやプログラマブルヒューズのロードエラー等が分類される．このグループではエラーが発生した時外部割込みは発生せずエラーピン端子に low レベルをドライブする．

Vectored Interrupt Manager(VIM) module はデバイスにある割込みソースの優先度の設定や制御を行う．対象 MCU には 95 個の割込みチャンネルがある．VIM では各チャンネルに対して，IRQ あるいは FIQ をユーザが指定することが可能である．また割込みの許可も構成可能である．信頼性向上のための特徴としてソフトエラーに対してパリティ保護された割込みテーブルを持つ．

3.3 ターゲットボード

今回対象 MCU を搭載しているボード TMS570LS31 HERCULES Development Kit を使用した．このボードには Integrated USB JTAG Emulator や 8MB SDRAM，Programmable LED，Error pin LED，リセットボタンが搭載されている．

4. 開発した RTOS の実装

開発環境及び開発した RTOS について述べる．

4.1 開発環境

TMS570LS3137 はビックエンディアンをサポートしている．開発にあたり Texas Instruments 社の統合開発環境のフリーバージョンには ARM Cortex-R4F CPU ビックエンディアン用の gcc コンパイラが含まれていないため対象 MCU 用の gcc コンパイラを開発した．使用したパッケージは以下の通りである．

・gcc 4.9	・binutils 2.24	・cloog 0.18.0	・expat 2.0.1
・gmp 4.3.2	・isl 0.11.1	・libelf 0.8.13	・libiconv 1.14
・mpc 0.8.1	・mpfr 2.4.2	・newlib 2.2.0	・zlib 1.2.8

gcc パッケージ内の multilib のオプションで armv7 アーキテクチャのビッグエンディアンをサポートするように変更し gcc コンパイラを生成した．また開発した RTOS を対象 MCU 上で動作させる環境として Open On-Chip Debugger(OpenOCD) 0.9.0 を使用した．これは RTOS の実行ファイルを MCU 上でデバックするデバッカであり，Flash への書込みやメモリ操作が可能となる．また gdb 7.8.0 を gcc コンパイラと共に作成した．これらは CentOS 6.9 を搭載した PC 上に構築した．

4.2 拡張機能

この MCU ではシステムリセット後に CPU レジスタやメモリ，PLL の初期化の他に各モジュールのセルフテストを行う．したがって開発した RTOS のスタートアップルーチン内にセルフテスト機能を組込んだ．

次にシステム制御コプロセッサのレジスタを用いた HRP2 kernel のメモリ保護機能を開発をした．8 つの領域を使用しタスク毎に開始アドレスや属性，サイズを設定する．また各設定情報を保持するコンテキストブロックを開発し，設定情報の取得を可能にした．

HRP2 kernel のタイマ機能及びオーバランハンドラ機能を Real Time Interrupt(RTI) module のコンペアレジスタを使って有効にした．

ターゲット依存部としてシリアル通信を要求するため Serial Communication Interface(SCI) Module を組込んだ．

RTI 及び SCI で発生した割込みは Vectored Interrupt Manager(VIM) module によって管理される．

4.3 故障検知機能 API

故障検知機能は対象 MCU の Error Signaling Module(ESM) を有効にすることで実装した．

ESM ではシステムのエラー状態を管理し割当てられた優先度に応じて，処理する．ここで，対象 MCU ボードではそれぞれのエラーが発生した場合 CPU チップの出力ピン Error pin に接続されている LED を点灯する．これによりグループ 3 のエラーは LED によりエラー状態を示すことが可能となる．

ESM を用いた故障検知機能を有効にする API を開発した．開発した API を RTOS 内に組込むことで信頼性の高い RTOS を実装することが出来る．ここで，グループ 3 に分類されるエラーは割込みが発生しないので，割込みハンドラ API の開発は行っていない．またグループ 1 およびグループ 2 のエラー割込みは VIM によって管理される．

1) 故障検知機能初期化 API

故障検知機能初期化 API では ESM の初期化を行う．内容としてエラーステータスレジスタ及び Error pin のリセットを行う．処理後グループ 2 のエラーは割込み許可状態となる．同様にグループ 3 のエラーはエラー検出後 Error pin に自動出力される．

2) 低優先度故障検知開始 API

低優先度故障検知機能開始 API ではグループ 1 のエラーの割込みを許可する．内容として低優先度故障検知の割込み優先度の設定及び割込みの許可を行う．エラー割込みの設定と共に不使用のペリフェラルからのエラー割込みを禁止する．

3) 低優先度故障検知停止 API

低優先度故障検知停止 API ではグループ 1 のエラーの割込みを禁止する．この API を使用することでグループ 1 内のエラー割込みをすべて禁止するか特定のエラーの割込みを禁止するかを設定できる．

4) 低優先度故障検知終了 API

低優先度故障検知終了 API ではグループ 1 のエラー割込みをすべて禁止する．

5) 低優先度故障割込みハンドラ

低優先度故障割込みハンドラはグループ 1 に該当するエラーが発生した時呼出される．このハンドラは通常割込み要求 (IRQ) で処理される．このハンドラの機能として割込み発生時のエラーのチャンネル番号を取得する．エラー処理は低優先度故障割込みハンドラの処理部で行われる．

6) 中優先度故障割込みハンドラ

中優先度故障割込みハンドラはグループ 2 に該当するエラーが発生した時呼出される．このハンドラは高速割込み要求 (FIQ) で起動するため高速処理される．このハンドラの機能として割込み発生時のエラーのチャンネル番号を取得する．エラー処理は中優先度故障割込みハンドラの処理部で行われる．

7) 中優先度及び低優先度故障割込みハンドラの処理部

中優先及び低優先故障割込みハンドラの処理部を使用することで発生したエラーのチャンネル番号を引数として処理プログラムを書くことが可能である．処理プログラムの例として該当ペリフェラルの停止などが考えられる．

8) CCM-R4F セルフテスト API

CCM-R4F セルフテスト API はデュアル CPU の演算結果を比較する Compare module に対してテストを行う．故障が検知された場合 CCM-R4F セルフテストエラーが発生し中優先度故障割込みハンドラが呼出される．

5. 開発した RTOS の評価

開発した RTOS を搭載したシステムの評価方法および評価結果について述べる．

5.1 評価方法

開発した RTOS の評価にあたり以下の評価プログラムを作成した．

5.1.1 故障検知機能評価プログラム

スタートアップルーチン内のセルフテスト機能および故障検知機能 API の評価プログラムを作成した．このプログラムはメインタスク *main_task* 及びボード上の LED を初期化するタスク *init_task*，CCM-R4F セルフテスト API から構成される．このプログラムは *main_task* が *init_task* を起動し LED が初期化され，その後 *main_task* 内で CCM-R4F セルフテスト API が動作し CCM-R4F に故障がなかった場合ボード上の LED が点灯するように作成した．

5.1.2 性能評価プログラム

RTOS の性能評価方法として RHEALSTONE benchmark program[8] がある．このベンチマークではタスクのプリエンプションやスイッチング，割込み応答，セマフォ入れ替え，デットロック解除，データグラム処理の時間性能計測手法が提案されている．Tran Nguyen Bao Anh[3] はタスクスイッチングやセマフォ獲得/返却及び通過，メッセージキュー獲得/返却及び通過，固定長メモリサイズ獲得/返却，割込み応答の時間性能評価手法を提案した．これらの提案手法をベースに開発した RTOS の持つ機能を評価した．評価にあたり対象 MCU の動作周波数は 100MHz とした．これは TMS570LS3137 を搭載している TMS570LS31x Hercules USB Development Kit における最大動作周波数である．また，時間性能計測のために，HRP2 kernel の性能評価用システム時間参照機能 *get_utm* 及び RTI module の *free-running counter* を使用した．これらの時間分解能は 1μ 秒である．システム状態はタイマ割込みを有効にしている．評価項目は以下の **1)** から **12)**

の通り．

1) タスクスイッテング時間では実行中の低優先度タスクが高優先度タスクを実行可能にするサービスコールを呼んだ時から高優先度タスクが実行状態になった時間までを計測する．開発した RTOS では実行するタスクのディスパッチの有無が選択可能である．またタスクやドメインの切替え時にメモリ保護ユニットの設定変更が行われる．したがって下の各条件で時間測定を行いディスパッチ時のデータ退避やメモリ保護ユニットの設定変更に伴う性能を評価する．

表 1 タスクスイッチングの評価条件
Table 1 Evaluation conditions of task switching

ディスパッチ	遷移元タスク	遷移先タスク
ディスパッチ無	システムタスク	-
	ユーザタスク	-
ディスパッチ有	システムタスク	システムタスク
	システムタスク	ユーザタスク
	ユーザタスク	ユーザタスク (遷移元と同ドメイン)
	ユーザタスク	ユーザタスク (遷移元と異ドメイン)

2) 割込み応答時間では割込みが発生した時から割込みハンドラが起動した時までの時間を計測する．

3) 割込みハンドラからのタスク応答時間では割込みハンドラでタスクを実行可能にするサービスコールを呼んだ時からタスクが実行状態になった時間までを計測する．

4) セマフォ獲得/返却時間ではセマフォの獲得サービスコールを呼ぶ前と呼んだ後返却サービスコールを呼ぶ前と呼んだ後の時間を計測する．

5) タスク間のセマフォ通過時間ではセマフォ返却待ちのタスクがある状態で他のタスクによってセマフォが返却された時から返却待ちのタスクがセマフォを獲得する時までの時間を計測する．

6) データキュー登録/獲得時間ではデータキューのデータ登録サービスコールを呼ぶ前と呼んだ後データ獲得サービスコールを呼ぶ前と呼んだ後の時間を計測する．

7) タスク間のデータキュー通過時間ではデータ獲得待ちのタスクがある状態で他のタスクによってデータキューにデータが登録された時から獲得待ちのタスクがデータを獲得する時までの時間を計測する．

8) 固定メモリブロック獲得/返却時間ではメモリブロックの獲得サービスコールを呼ぶ前と呼んだ後返却サービスコールを呼ぶ前と呼んだ後の時間を計測する．

9) イベントフラグ登録/解除時間ではイベントフラグを用いたイベントの登録サービスコールを呼ぶ前と呼んだ後解除サービスコールを呼ぶ前と呼んだ後の時間を計測する．

10) タスク間のイベントフラグ通過時間ではイベント待ちのタスクがある状態他のタスクによってイベントフラグによってイベント登録された時からイベント待ちのタスクがイベント待ち解除した時までの時間を計測する．

11) ミューテックスロック/ロック解除時間ではミューテックスのロックサービスコールを呼ぶ前と呼んだ後ロック解除サービスコールを呼ぶ前と呼んだ後の時間を計測する．

12) タスク間のミューテックス通過時間ではミューテックスロック待ちのタスクある状態で他のタスクによってミューテックスのロック解除される時からロック待ちのタスクがロックした時までの時間を計測する．

これらの評価項目のためのプログラムを作成した．割込みハンドラ内では *get_utm* が使用できないため評価項目 **2)**，**3)** では，*free-running counter* を使用し他項目では *get_utm* を使用した．

次に CPU の動作周波数を 100MHz から 10KHz まで変化させることによる CPU の性能評価も行った．動作周波数を変更するためにスタートアップルーチン内の PLL 初期化時，動作周波数を決定するプログラムを組込んだ．開発した RTOS では周期ハンドラ機能からタスクを起動する場合，ディスパッチなしのタスクスイッチを行う．したがって，**1) タスクスイッテング時間**の実行中の低優先度タスクがシステムタスクで高優先度タスクを実行可能にするサービスコールを呼んだ時ディスパッチなしで高優先度タスクが実行状態になった時間までを計測するタスクスイッチング計測プログラムを使用した．動作周波数の変更に伴い *get_utm* の時間分解能が変化するため ARM Cortex-R4F プロセッサが持つ Performance Monitoring Unit(PMU)[9] を使用した．PMU はシステム制御コプロセッサのレジスタを用いて CPU のクロックサイクルを計測することができる．評価プログラムの *get_utm* を PMU のクロックサイクル参照処理に変更した．変更後クロック数を計測し動作周波数の逆数を掛けることで時間を算出した．

RTOS の性能評価および CPU の性能評価で作成したプログラムは計測を 100 回行い平均値と最大値を求めた．

5.2 評価結果

まず，故障検知機能評価プログラムを動作させたことで，(1) スタートアップルーチン内で行われたシステムのセルフテスト機能によってシステムに故障がなかった，(2) システムリセット後 RTOS のスケジューラが動作し *main_task* が起動した，(3) *init_task* が起動されタスクのディスパッチが行われた，(4) CCM-R4F セルフテスト API を動作させた結果 CCM-R4F module に故障がなかった，ことを確認した．

次に RTOS の機能性能を評価する．まず *get_utm* 及び *free-running counter* のジッターの計測を 100 回行った結果 *get_utm* が 5.17μ 秒， *free-running counter* が 0.27μ 秒であった．評価結果からジッターの値を除くことで RTOS 機能の時間性能を算出した．計測結果は表 2 である．測定

結果から各機能は 20μ 秒以下で処理されることが分かった．またシステムタスクからユーザタスクに切替わる時，メモリ保護ユニットの設定変更に平均 3.35μ 秒かかり異なるユーザドメイン間での切替えでは平均 4.74μ 秒かかることが分かった．

表 2　RTOS 機能の測定結果
Table 2　Measurement results of RTOS functions

評価機能				結果の平均値 [μ 秒]	結果の最大値 [μ 秒]
タスクスイッチング	ディスパッチ	遷移元タスク	遷移先タスク		
	ディスパッチ無	システムタスク	-	2.60	6.83
		ユーザタスク	-	5.03	9.83
	ディスパッチ有	システムタスク	システムタスク	6.26	10.83
		システムタスク	ユーザタスク	9.61	13.83
		ユーザタスク	ユーザタスク (遷移元と同ドメイン)	9.25	14.83
		ユーザタスク	ユーザタスク (遷移元と異ドメイン)	11.00	15.83
割込み応答				3.75	4.73
割込みハンドラからのタスク応答				5.65	6.73
セマフォ獲得				3.95	8.83
セマフォ返却				3.60	8.83
タスク間のセマフォ通過				7.69	12.83
データキュー登録				5.04	8.83
データキュー獲得				4.65	5.83
タスク間のデータキュー通過				7.88	12.83
固定メモリブロック獲得				4.40	8.83
固定メモリブロック返却				5.41	9.83
イベントフラグ登録				3.55	7.83
イベントフラグ解除				3.30	4.83
タスク間のイベントフラグ通過				10.61	16.83
ミューテックスロック				4.44	8.83
ミューテックスロック解除				4.31	5.83
タスク間のミューテックス通過				8.16	8.83

最後に動作周波数を変化させたことによる CPU 性能の結果を示す．まず PMU のクロックサイクル参照機能ジッター計測を 100 回行った平均クロック数は 13 クロックであった．CPU 動作周波数変化プログラムの時間結果からジッターの平均値に動作周波数の逆数を掛けた値を除くことで CPU の動作周波数を変化させた時の時間性能を算出した．算出した結果は表 3 である．

表 3　動作周波数に伴うタスクスイッチング時間
Table 3　Measurement results of task switching under rarions operating frequency

動作周波数	平均値 [μ 秒]	最大値 [μ 秒]
100 MHz	2.87	3.06
10 MHz	29.30	32.00
1 MHz	282.81	315.00
100 KHz	2834.20	3200.00
10 KHz	26270.00	31800.00

開発した RTOS では周期ハンドラやアラームハンドラに設定できる最小値は 1m 秒である．周期ハンドラの起動周期を 1m 秒にし CPU の動作周波数を 100KHz 以下にした場合，周期ハンドラからのタスクスイッチが 1m 秒を超過ことがわかった．一方動作周波数が 100MHz の場合，平均で周期時間の 0.287%しかタスクスイッチングに費されない．

6. おわりに

本稿では，故障検知機能を持つ MCU で動作する信頼性の高い RTOS の開発について述べた．開発した RTOS は TOPPERS/HRP2 kernel をベースに故障検知機能を拡張した．RTOS で動作する評価プログラムを作成し動作結果からスタートアップルーチン内でセルフテストが行われ，システムの故障が存在しないことを確認した．また、故障検知機能 API を有効にすることで故障検知が可能であり，故障割込みハンドラの処理部をユーザが定義することで故障対処が可能である．したがって，開発した RTOS を使用しセルフテスト機能や故障検知機能 API を有効にすることで，故障が発生してもシステム全体の信頼性向上につながる．また，RTOS の機能や CPU の性能の評価結果から動作周波数を 100MHz にした場合，各 RTOS 機能が 20μ 秒以下で処理されること，100KHz 以下にした場合タスクスイッチング時間がタイマ機能の最小周期を超過する可能性があることを確認した．これらの結果は開発した RTOS を使用するアプリケーションプログラムを設計する時の参考値となることが考えられる．

参考文献

[1] Baleani, M., Ferrari, A., Mangeruca, L., Sangiovanni-Vincentelli, A., Peri, M. and Pezzini, S.: Fault-tolerant platforms for automotive safety-critical applications, *Proceedings of the 2003 international conference on Compilers, architecture and synthesis for embedded systems*, ACM, pp. 170–177 (2003).

[2] Isermann, R. and Balle, P.: Trends in the application of model-based fault detection and diagnosis of technical processes, *Control engineering practice*, Vol. 5, No. 5, pp. 709–719 (1997).

[3] Anh, T. N. B. and Tan, S.-L.: Real-time operating systems for small microcontrollers, *IEEE micro*, Vol. 29, No. 5, pp. 30–45 (2009).

[4] TEXAS INSTRUMENTS Inc.: *TMS570LS31x/21x 16/32-Bit RISC Flash Microcontroller Technical Reference Manual*, http://www.tij.co.jp/jp/lit/ug/spnu499b/spnu499b.pdf (2013).

[5] WITTENSTEIN High Interrity Systems: *SAFERTOS User Manual for the Code Composer Studio TMS570 MPU Product Variant*, https://www.highintegritysystems.com/downloads/manuals_and_datasheets/Sample_SafeRTOS_User_Manual.pdf (2011).

[6] Labrosse, J. J: *μC/OS-II The Real-Time Kernel User's Manual*, https://doc.micrium.com/download/attachments/10753158/100-uC-OS-II-003.pdf (2015).

[7] NPO 法人 TOPPERS プロジェクト：TOPPERS 第 3 世代カーネル (ITRON 系) 統合仕様書, release 3.0.0 edition, https://www.toppers.jp/docs/tech/tgki_spec-300.pdf (2016).

[8] Kar, R. P. and Porter, K.: Rhealstone-a real-time benchmarking proposal, *Dr Dobbs Journal*, Vol. 14, No. 2, p. 14 (1989).

[9] ARM: *Cortex-R4 and Cortex-R4F Technical Reference Manual*, http://infocenter.arm.com/help/topic/com.arm.doc.ddi0363fj/DDI0363FJ_cortexr4_trm.pdf (2009).