Vol.2014-SLDM-165 No.28
Vol.2014-EMB-32 No.28
2014/3/16

ノーマリオフによる組み込みシステム低消費電力化の実装と評価

荒川 祐真[1]　長崎 健[1]　戸田 真志[2]　平田 圭二[1]　松原 仁[1]

概要：近年，小型な組み込みシステムが増加している．特にセンサーネットワークと呼ばれる分野では電池の交換にかかるコストや電池の大きさによるシステムの物理的な大きさなどからシステムの低消費電力化が求められている．本研究ではノーマリオフにより組み込みシステムの低消費電力化を図る．ノーマリオフとは端的には「使わないときは電気を切る」という考えで，待機時の消費電力 0 を目指す．しかし，CPU や RAM などの揮発性のデバイスは電気を切ると保持していた内部情報が消えてしまう．そこで，これらの内部情報を退避するための領域として不揮発性メモリ (NVRAM) を用いる．本研究ではこのようなノーマリオフを実現するためのハードウェアの枠組みを想定し，ソフトウェアの機能について提案・実装・電力評価を行う．

1. はじめに

近年，スマートフォンやタブレット型端末に代表されるように，小型な組み込みシステムの利用が増加している．中でも，スマートシティやスマートハウスを背景に，センサネットワークと呼ばれる分野での利用が注目されている．センサネットワークデバイスもまた小型な組み込みシステムの 1 つである．このようなデバイスは，人体などの移動体が装着する，自然環境に広く散布するといった利用が想定される．そのためセンサネットワークデバイスにおいては，内蔵電池の小型化や電池の交換にかかるコストの削減につながることから，システムの低消費電力化が重要である．

本論文ではセンサネットワークデバイスのためのノーマリオフ組み込みシステムを提案する．ノーマリオフとはシステムの待機時には電源を切るという考え [1] である．しかし，揮発性のデバイスは電源を切られると内部で保持していた情報が消えてしまう．そこで，システムが待機状態となる際に，これらの内部情報を不揮発性メモリ（NVRAM）へ退避する．また，システムの動作再開時に，退避した内部情報を NVRAM から復帰することでシステムのノーマリオフ化を実現する．本研究では，このようなノーマリオフを実現するための機能を OS 上に実装する．

[1] 公立はこだて未来大学
Future University Hakodate
[2] 熊本大学
Kumamoto University

2. 関連研究

2.1 既存の省電力技術

既存の省電力技術としては，プロセッサコアの間欠動作（パワーゲーティング）により低消費電力化を図る研究がある（木村ら [2]）．木村らは，プロセッサコアの各演算ユニットに対してパワーゲーティングが行えるプロセッサコア「Geyser」を用いて，パワーゲーティングの実施ポリシーを定める手法を提案している．そこで，木村らは各演算ユニットのパワーゲーティングにおける損益分岐点がプロセッサコアの温度情報により変化する点に着目し，損益分岐の条件に温度のパラメータを含めた条件式を提案している．

また，プロセッサ動作周波数を各プロセス毎に動的に変更することで省電力化を図る研究がある（宮川ら [3]）．宮川らの研究では，OS レベルの電力制御手法として電力制御スケジューラ PCCS（Power Consumption Controlling Scheduler）を提案し，プロトタイプを実装している．プロセッサの動作速度と消費電力量がトレードオフの関係にあることから，このスケジューラにより各プロセスの動作速度に基づいて動作周波数を変更している．

周期動作を行うセンサーネットワークデバイスを対象に，待機時の省電力化を図る研究がある（安部ら [4]）．安部らの研究では，待機時は周辺回路の電源を遮断し，MCU と無線モジュールを省電力モードへ移行することで省電力化を図る．ソフトウェア制御で周辺回路の電源を遮断

するために，周辺回路の電源を MCU から管理するための MOS-FET を設置している．

木村らの研究 [2] や宮川らの研究 [3] は CPU のみを省電力の対象としたものであり，安部らの研究 [4] は待機時であっても MCU は省電力モードではあるが電力を消費し続ける．これらの研究に対して，システム全体の省電力化を図る本手法と組み合わせることで，より省電力化が図れると考えられる．特に，本研究のノーマリオフおいては「CPU の電源も切る」という点がこれらの省電力手法とは異なる点である．

2.2 ノーマリオフを目指した研究

ノーマリオフを対象とした研究として，複数の周期タスクが存在するシステム上で，タスクスケジューリングによる省電力化を図る研究がある（岡本ら [5]）．これは，マイクロプロセッサを備えたセンサであるスマートセンサを対象とした研究である．そして，このようなシステムでは入力データのサンプリング周期とデータ送信（デッドライン）の周期は必ずしも一致せず，後者の周期の方が前者の周期よりはるかに大きい，すなわち時間制約がゆるいという点に着目している．そこで，システム上の複数のタスクについて，それらの実行順序を調整しアイドル時間をまとめることで，省電力モードへの状態遷移の回数を減らし，状態遷移に伴う電力オーバヘッドを削減する方法を提案している．

岡本らの手法は本研究においても有効なものであり，本研究と組み合わせることでよりシステムの低消費電力化が期待できる．ただし，岡本らの研究は理論を提案するものであり，その有効性についてはシミュレーション評価を行っている．対して，本研究はノーマリオフを実現するためのハードウェアの枠組みを想定し，ソフトウェアの機能について提案・実装するものである．特に，ソフトウェアの機能については，OS 上の省電力状態管理の枠組みを提案・実装している．

3. 想定するシステム構成

本論文で想定するシステムの構成を**図 1** に示す．システムはノーマリオフの領域とノーマリオンの領域で構成されている．ノーマリオン領域とは，システムのノーマリオフ化を実現するために最低限必要なデバイスや，システムの制約により周期的に電源を切ることが困難なデバイスが配置される領域である．ノーマリオフ領域は最低限のデバイスのみとして，CPU を含むシステムの大部分をノーマリオフ領域に配置し，低消費電力化を図る．本論文においてはノーマリオフ領域について消費電力の評価を行う．

なお，本論文において CPU はセンシングなどの周期処理を行うためのメインの CPU と，メインの CPU の電源を周期的に制御するためのみのノーマリオフ用の CPU が

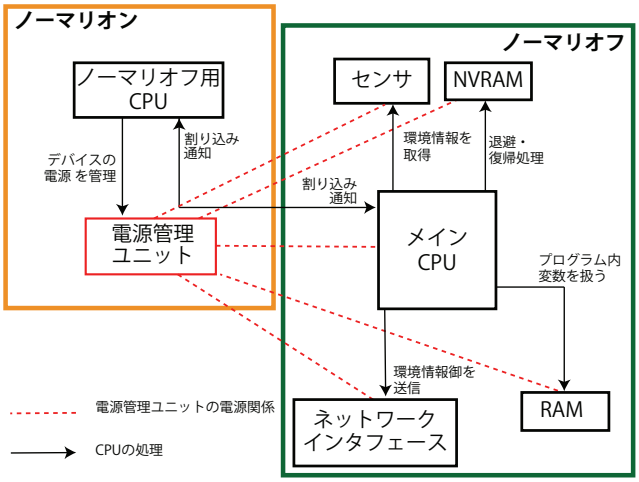

図 1 システム構成
Fig. 1 system structure

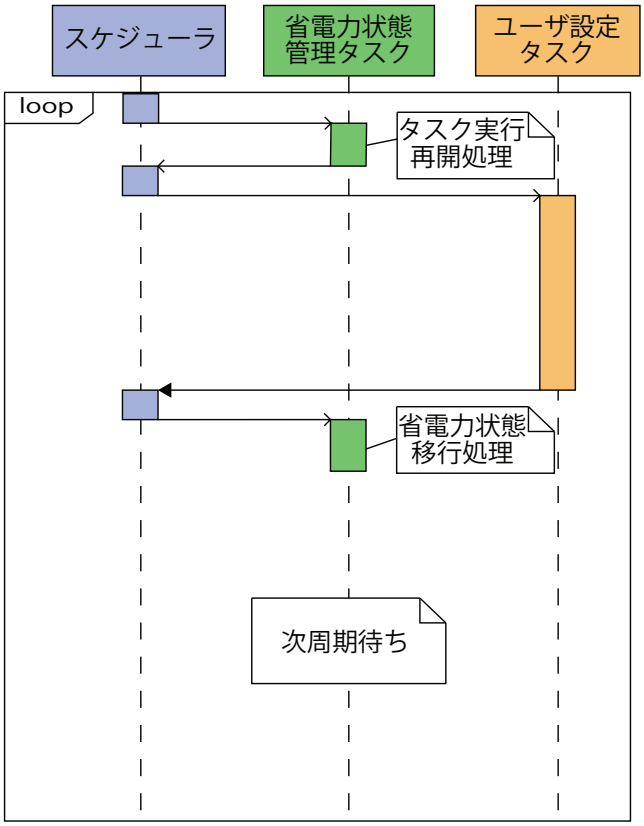

図 2 全体のシーケンス図
Fig. 2 Entire sequence

存在する．ノーマリオフ用の CPU は，低消費電力なものを用い，ノーマリオン領域に配置する．メインの CPU はノーマリオフ領域に配置し，ノーマリオフ化の対象とする．また，その他のデバイスとしてはノーマリオフ領域には RAM，センサ，ネットワークインタフェース，NVRAM が配置される．特に，NVRAM はノーマリオフを実現するために必要なデバイスで，揮発性デバイスの内部情報を退避・復帰するために使用する．

4. 提案システム

4.1 省電力状態管理タスクの概要

本論文では，OS 上の省電力化のための枠組みを「省電力状態管理タスク」というタスクとして実装する．省電力状態管理タスクは，ノーマリオフを含む省電力状態への移行と復帰を管理するタスクで，省電力状態への移行と復帰を行うタイミングでこのタスクが起床する．省電力状態への移行と復帰に関する処理は，デバイスドライバとシステムコールで実装する．そして，省電力状態管理タスクは，既存のシステムコールと新たに追加されたシステムコールを利用して省電力方針を記述する．このような構成とすることで，通常のタスクを記述する場合と同様に，省電力状態管理タスクの振る舞いを記述できる．そのため，本手法においては省電力のための設定を行う際にカーネル部の設計についての知識を必要としない．

なお，CPU と RAM の内部情報は OS の管理資源を単位として退避と復帰を行う．対象となる管理資源として，タスクに関しては，タスク管理ブロック（TCB）とタスクスタック領域，静的変数領域などで，その他にはメモリプール領域などがある．そのため，退避と復帰のシステムコールも OS の管理資源毎に作成する．

4.2 省電力状態管理タスクの振る舞い

省電力状態管理タスクの振る舞いについて**図 2** に示す．図 2 は，スケジューラと省電力状態管理タスク，ユーザ設定タスクの関わりを示している．図 2 の「ユーザ設定タスク」は，OS を利用するユーザ（組み込みシステム開発者）が設定するタスクである．このタスクは定期的に実行する処理を終えると，システムコールを通じて自タスクをタスク待機状態へ遷移させる．

図 2 では省電力状態管理タスクの基本的な振る舞いを示している．ノーマリオフのために必要な退避と復帰の処理は，それぞれ「全てのタスクがタスク待ち状態へ遷移した直後」と「システムの動作再開時」に実行されるべきである．このタイミングを保証するために，「省電力状態管理タスク」を平常時は最低優先度としておく．このように実装することで，他の全てのタスクが待ち状態へ遷移した後に退避の処理と電源管理ユニットへの電源遮断の通知をすることが可能となる．

表 1 ノーマリオフ評価環境
Table 1 Normally-off evaluation environment

デバイス名	メーカー名（型番）
メイン CPU	ルネサスエレクトロニクス（R5F563NEDDFC）
ノーマリオフ用 CPU	ルネサスエレクトロニクス（uPD78F1166AGC-UEU-AX）
温度センサ	Rohm（BD1020HFV）
FeRAM	RAMTRON（FM22L16）

表 2 ソースコードの追加行数
Table 2 Number of added lines

領域	追加／修正行数
ターゲット依存部	1080
CPU 依存部	206
カーネル（システムコール）	2057
デバイスドライバ	1362
省電力状態管理タスク	1041
総合計	5746

図 2 で「タスク実行再開処理」と示してある部分では，ノーマリオフ利用時は NVRAM から OS 資源の復帰処理を行う．そして，自身の優先度を最も低く設定し，ユーザが設定したタスクへディスパッチする．図 2 で「省電力状態移行処理」と示してある部分では，省電力状態へ移行するための処理として，ノーマリオフ利用時は OS 資源の NVRAM への退避処理を行う．そして，NVRAM へ省電力方針について設定し，ノーマリオフ用 CPU へ割り込み通知を行い，メイン CPU は省電力モードへ移行する．ノーマリオフ用 CPU は割り込み通知を受けると NVRAM からノーマリオフ方針を読み取り，電源遮断動作を行う．

このように，他のタスクがタスク待ち状態へ遷移したことをきっかけに動作するため，省電力状態管理タスクの追加に伴うユーザが設定したタスクの修正は不要である．また，本省電力手法はその実行タイミングから，タスク実行周期が長いほど，退避処理後に電源を遮断してから復帰処理を始めるまでの待機時間が長くなる．そのため，タスク実行周期が長いほど省電力効果が高くなる手法である．

5. 実験

5.1 実験設定

タスク設定について事前評価を行うためのシミュレーション実験と，電力消費について評価するための計測実験を行った．本論文の実験で使用したノーマリオフ評価環境を**表 1** に示す．メインの CPU として RX63N を，ノーマリオフ用の CPU として 78K0R を使用した．また，OS 資源の退避／復帰の領域に RX63N のデータフラッシュ領域か，あるいは FeRAM を使用し，実験でそれらを比較する．そして，OS には TOPPERS/ASP カーネルを使用し，システムコールや省電力状態管理タスクなどを追加した．な

お，以上の追加／修正を行ったソースコードの行数はおよそ**表 2** に示す通りである．主にカーネル部に対して修正は行っておらず，デバイスドライバとシステムコール，タスクの追加のみで機能が実現できているため，既存のシステムへの導入もし易い実装となっている．

本論文ではタスク設定としては温度センサの情報のみを周期的にネットワークインタフェースを通じて送信する状況を想定し，実験を行う．そこで，ユーザが設定するタスクは温度センサのタスク一つのみで，その動作は 1 秒周期で温度センサ値を取得し，1 分毎にまとめてネットワーク送信を行うものとする．そのため，今回の実験で利用する OS 資源は温度センサタスクのタスク資源のみで，1 分周期でネットワーク送信するために過去の温度センサ値をタスク内にもっているため温度センサタスクのタスク資源を退避／復帰の対象とする．

以降の実験では比較として RX63N のスリープモードと，いかなる省電力モードも利用しないアイドルループについても実験を行った．また，RX63N には低消費電力状態としてディープソフトウェアスタンバイモードがある．これは，CPU，内蔵周辺機能，RAM，および発振器の全ての機能を停止し，さらにこれらの内部電源の供給を停止するモード [6] である．ディープソフトウェアスタンバイモードについては，ノーマリオフ適用時の RX63N の省電力方針として電源を遮断した場合との比較として計測実験で評価を行う．

5.2 シミュレーション実験

ノーマリオフを利用する上でのタスク設定について評価するシミュレーション実験を行った．ノーマリオフを利用する際には，OS 資源の退避／復帰のために，データフラッシュ，あるいは FeRAM への書き込みと読み出しの電力消費が発生する．そのため，それらのオーバヘッドに対してノーマリオフによる電力削減効果が得られる損益分岐点（Break-Even Point:BEP）を評価する．ここで，待機時間の長さ t のときの損益分岐の条件式を式 1 に示す．条件式 1 は左辺がノーマリオフ利用時の待機時の消費電力量を，右辺がスリープモード利用時の待機時の消費電力量を表しており，この条件が成立する場合にノーマリオフを利用する．なお，E_{save} は退避にかかる消費電力量，E_{load} は復帰にかかる消費電力量，E_{recv} は復帰安定待ちの消費電力量，$E_S(t)$ はスリープモード利用時の消費電力量である．スリープモードは待機時間の長さに応じてスリープモード利用中の消費電力量は変化するため，消費電力量は待機時間の長さ t に依存する関数となっている．

$$E_{save} + E_{load} + E_{recov} < E_S(t) \quad (1)$$

本論文では，各種パラメータから損益分岐について評価を行うシミュレータを作成した．実験ではデータシートの値などから**表 3** に示すようにパラメータを設定した．

表 3 シミュレータに設定したパラメータ [6]
Table 3 Simulation parameters[6]

項目名	値	単位
電源電圧	3.3	V
CPU 消費電流		
-通常動作	52	mA
-スリープ	25	mA
退避／復帰データサイズ	550	bytes
パワーオン復帰 発振安定時間	1	ms
E2 データフラッシュ		
-書き込み時消費電流	7.2	mA
-バス幅	16	bits
-書き込み時間	0.25	ms
-消去ブロックサイズ	32	bytes
-ブロック消去時間	4	ms
FeRAM		
-消費電流	(Max) 18	mA
-バスアクセスサイクル	120	ns
-アクセス単位	1	バイト
A/D 変換器		
分解能	12	bits
変換時間	1	μs
アナログ電源電流		
-A/D 変換中	2.3	mA
-A/D 変換待機時	25	μA
リファレンス電源電流		
-A/D 変換中	0.6	mA
-A/D 変換待機時	1	mA

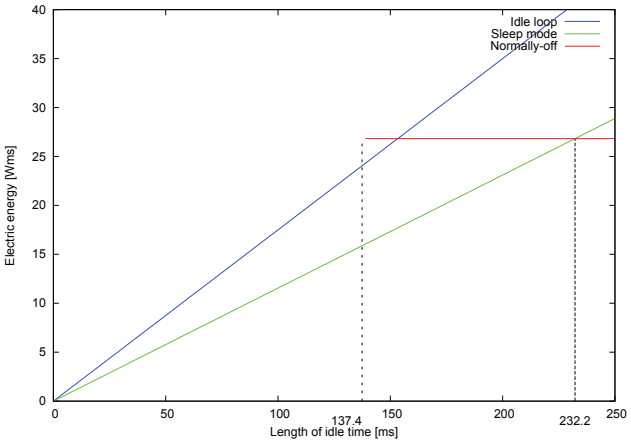

図 3 BEP(データフラッシュ使用時)
Fig. 3 BEP(when the system used dataflash)

BEP の評価結果を退避／復帰の領域としてデータフラッシュを使用した場合について**図 3** に，FeRAM を使用した場合について図 4 に示す．図 3 より，データフラッシュ利用時は CPU の既存の省電力状態であるスリープモードとの BEP が 232.2ms で，待機時間の長さがこれより長けれ

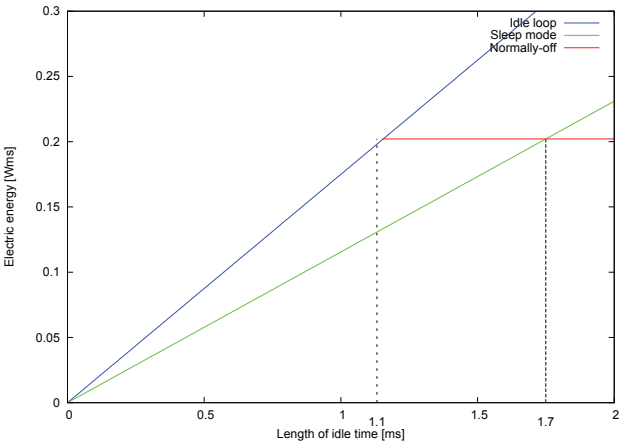

図 4 BEP(FeRAM 使用時)
Fig. 4 BEP(when the system used FeRAM)

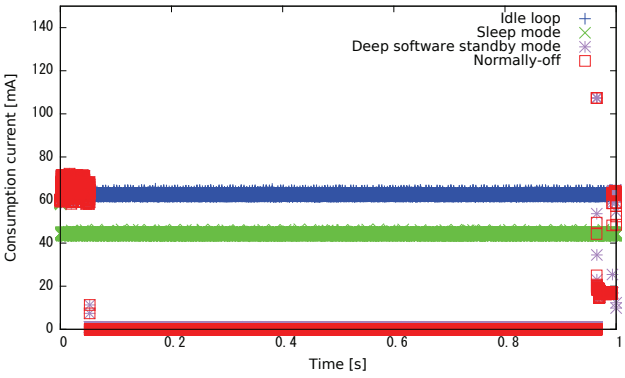

図 5 消費電流 (データフラッシュ使用時)
Fig. 5 Current consumption(when the system used dataflash)

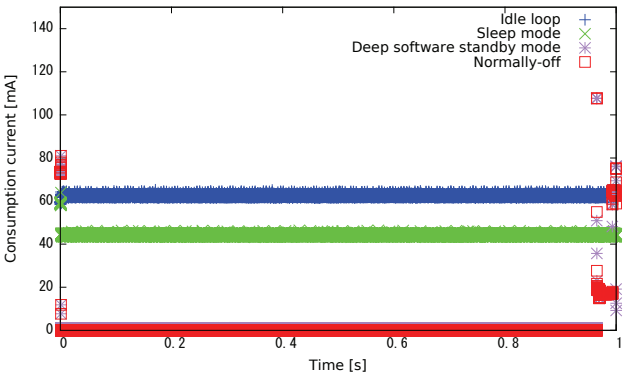

図 6 消費電流 (FeRAM 使用時)
Fig. 6 Current consumption(when the system used FeRAM)

ばノーマリオフ適用の効果が得られることが分かる．同様に**図 4** より，FeRAM 利用時は BEP が 1.7ms で，データフラッシュと FeRAM のいずれの場合においても今回設定するタスク実行周期 1 秒においては問題なくノーマリオフの省電力効果が得られることが確認できた．

なお，図 3 の 137.4ms と図 4 の 1.1ms に引かれている補助線は，待機時間の長さがこれより短い場合はノーマリオフを利用が不可能であることを表す．これは，主に退避処理と復帰処理，発振安定待ちにかかる時間によるもので，ノーマリオフはこの時点から描画されている．

5.3 計測実験

計測実験の結果を，OS 資源の退避／復帰にデータフラッシュ領域を使用した場合について**図 5** に，FeRAM を使用した場合について**図 6** に示す．図 5 と図 6 に関して，赤でプロットされているものはノーマリオフ適用時の結果で，紫もノーマリオフの方針ではあるが待機時に RX63N はディープソフトウェアスタンバイモードとした場合の結果，緑はスリープモードを利用した場合の結果で，青は待機時はいかなる省電力モードも利用しないアイドルループで待機した場合の結果である．なお，スリープモードとアイドルループ使用時は，退避／復帰の処理は無いため，どちらの図も同じ結果である．

実験結果の図 5 と図 6 では，0 秒の時点からタスクの実行が始まり，次に再びタスクを実行する直前までの 1 秒の周期を示している．これらの図から，ノーマリオフの方針を適用した場合（ノーマリオフ利用時とディープソフトウェアスタンバイ利用時）は，タスク実行と省電力状態管理タスクによる退避処理を終えた後の待機状態において，従来手法であるスリープモードと比較して電力消費を大幅に削減できていることが分かる．なお，ディープソフトウェアスタンバイモード利用時と提案手法のノーマリオフ利用時に関して，退避／復帰のデータサイズは同じであるため，電力消費に違いが表れる箇所は待機時のみである．待機時は，提案手法においては電力消費が 0 となっているが，ディープソフトウェアスタンバイモード利用時はわずかながらに電力を消費し続けている．また，退避／復帰の領域としてデータフラッシュを使用した場合の図 5 と FeRAM を使用した場合の図 6 を比較すると，復帰処理に比べ退避処理の方が変化が大きい．特にデータフラッシュを用いた場合においては復帰処理よりも退避処理の方がノーマリオフを利用する際の電力的・時間的オーバヘッドが大きいことが分かる．

図 5 と図 6 のそれぞれの手法について，1 周期の平均消費電流を求めた．結果を**表 4** に示す．表 4 から，提案手法のノーマリオフ利用時とスリープモード利用時を比較すると，ノーマリオフ利用時は退避／復帰領域にデータフラッシュを用いた場合は 90%程の省電力化，FeRAM を用いた

表 4 平均消費電流
Table 4 Average current consumption

省電力状態	平均消費電流 [mA]
ノーマリオフ (データフラッシュ)	4.4
ノーマリオフ (FeRAM)	1
ディープソフトウェアスタンバイ (データフラッシュ)	5.2
ディープソフトウェアスタンバイ (FeRAM)	1.9
スリープモード	44.4
アイドルループ	62.6

表 5 退避処理時間の比較
Table 5 Comparison of save processing time

退避領域	時間 [ms]
データフラッシュ	51.7
FeRAM	0.07

表 6 電池寿命
Table 6 Battery life

省電力状態	電池寿命 [h]
ノーマリオフ（データフラッシュ）	175.7
ノーマリオフ（FeRAM）	250.3
ディープソフトウェアスタンバイ（データフラッシュ）	164.3
ディープソフトウェアスタンバイ（FeRAM）	225.7
スリープモード	38.9
アイドルループ	28.7

場合は 98%程の省電力化を実現できていることが分かる．また，ディープソフトウェアスタンバイモードと比較すると，データフラッシュを用いた場合は 15%程，FeRAM を用いた場合は 47%程の省電力化を実現できた．そして，退避処理時間についてデータフラッシュ使用時と FeRAM 使用時の結果を**表 5** に示す．結果から，FeRAM 使用時はデータフラッシュ使用時に比べ，退避処理時間を 99%以上短縮できていることが分かる．

6. 考察

平均消費電流についての評価結果（表 4）から，アルカリの単 3 電池（公称容量 2000mAh）を 2 本直列で電源に用いた場合を想定した電池寿命を評価する．評価結果を**表 6** に示す．なお，評価の際にノーマリオン領域（78K0R）についても事前に計測を行い，計測結果である平均消費電流 7mA（消費電力 23.1mW）を加えて評価している．表 6 から，提案手法のノーマリオフ利用時はスリープモードと比較して，データフラッシュを用いた場合に 4.5 倍程，FeRAM を用いた場合に 6.4 倍程の電池寿命延長が確認できた．

また，平均消費電流の表 4 のディープソフトウェアスタンバイモード利用時に対する提案手法の消費電流の省電力割合は，データフラッシュ使用時に 90%で FeRAM 使用時に 98%と，FeRAM 使用時の方が高い結果となった．退避／復帰処理にノーマリオフ利用時とディープソフトウェアスタンバイモード利用時の違いは無いため，これは待機時の消費電力の違いによるものである．ノーマリオフ利用時は待機時の消費電力は 0 で待機時間の長さに依存する電力消費は無いが，ディープソフトウェアスタンバイモード利用時は待機時間の長さに応じて消費電力が増加する．このことから，データフラッシュに比べて FeRAM は退避処理にかかる時間は短く待機時間は長いことからこの結果となったことが分かる．以上から，提案手法は待機時間の長さが長くなれば，ディープソフトウェアスタンバイモードと比べて，より省電力効果が高まる手法であると言える．

7. まとめ

本論文では，センサネットワークデバイスを対象としたノーマリオフ化による低消費電力化について述べた．ノーマリオフを実現するハードウェアの構成について説明し，提案手法として省電力機能を OS 上に「省電力状態管理タスク」というタスクとして実装する手法を提案した．

実験ではシミュレーション実験と計測実験を行った．シミュレーション実験ではノーマリオフと既存の省電力手法として CPU のスリープモードの BEP を評価し，実験におけるタスク設定で十分にノーマリオフによる省電力効果を得られることを評価した．そして，計測実験では，結果からノーマリオフ利用時は FeRAM を用いることでスリープモードと比較して 98%程の省電力化が確認できた．

参考文献

[1] 安藤功兒：不揮発性デバイス：ノーマリオフコンピュータは実現できるか，電子情報通信学会誌， Vol. 93, No. 11, pp. 913–917 (2010).
[2] 木村一樹，近藤正章，天野英晴，宇佐美公良，中村　宏，佐藤未来子，並木美太郎：コア温度情報を用いた OS による細粒度パワーゲーティング制御方式の設計，技術報告 7, 東京農工大学, 電気通信大学, 慶應義塾大学, 芝浦工業大学, 東京大学, 東京農工大学, 東京農工大学 (2011).
[3] 宮川大輔，石川　裕：電力制御スケジューラのプロトタイプ実装，技術報告 86(2006-OS-103)，東京大学, 東京大学 (2006).
[4] 安部惠一，水野忠則，峰野博史：無線センサネットワーク向けの省電力型無線ノードの開発，技術報告 5，静岡大学創造科学技術大学院／浜松職業能力開発短期大学校, 愛知工業大学, 静岡大学創造科学技術大学院 (2011).
[5] 岡本和也，薦田登志矢，中田　尚，三輪　忍，佐藤洋平，植木　浩，林越正紀，清水　徹，　中村宏：周期実行システムにおける省電力スケジューリングの初期検討，技術報告 4，東京大学, 東京大学, 東京大学, 東京大学, ルネサスエレクトロニクス株式会社, ルネサスエレクトロニクス株式会社, ルネサスエレクトロニクス株式会社, ルネサスエレクトロニクス株式会社, 東京大学 (2012).
[6] ルネサスエレクトロニクス株式会社：RX63N グループ、RX631 グループ　ユーザーズマニュアル　ハードウェア編 (2013).