7Y-08

複数 BLE サービスの同時運用に対応した IoT センサにおける並行タスクの制御機構の設計と実装

馬場　健輔†　　内藤　克浩‡

愛知工業大学大学院†　愛知工業大学‡

1．はじめに

近年，センサと無線通信機構を搭載したデバイスである IoT センサと連携するスマートフォンアプリが普及している[1]．IoT センサはアプリごとに専用機を設計すること一般的である．そのため，複数種類のアプリが同時実行される環境では，アプリごとに IoT センサを準備する必要があり，コストの観点から改善の余地が残されている．

そこで我々は，複数アプリから同時にセンサ値を要求されるような状況に対応可能な IoT センサモジュールを実現するフレームワークを提案している．本フレームワークには，複数センサを制御する機能と，ユーザ設定に基づく挙動をとるプロセスであるユーザ定義タスクとの並行処理が求められる．しかし，現行の IoT センサモジュールはユーザ定義タスクを統括的に管理する機能を有していないため，ユーザ定義タスクの実行頻度にばらつきが生じてしまう．

そこで本稿では，IoT センサモジュールフレームワークにおける並行タスクの制御機構であるタスクマネージャを提案する．タスクマネージャは，FreeRTOS の機能の一つであるマルチタスキング機能とセマフォを用いることで，タスクの排他制御・管理を行う．書き込まれるユーザ定義タスクを動的配列に追加し，ユーザ定義タスクの実行順序を管理することで，ユーザ定義タスクの実行頻度を均一化し処理の確実な実行を担保する．タスクマネージャは，IoT センサモジュールフレームワーク内に常駐しているタスクと並行処理可能とするため，タスクマネージャ自体もタスクとして定義している．検証にてタスクマネージャが複数タスクを適切に管理可能であるのか評価する．

†Kensuke Baba, Aichi Institute of Technology Graduate School of Business Administration and Computer Science
‡Katsuhiro Naito, Aichi Institute of Technology

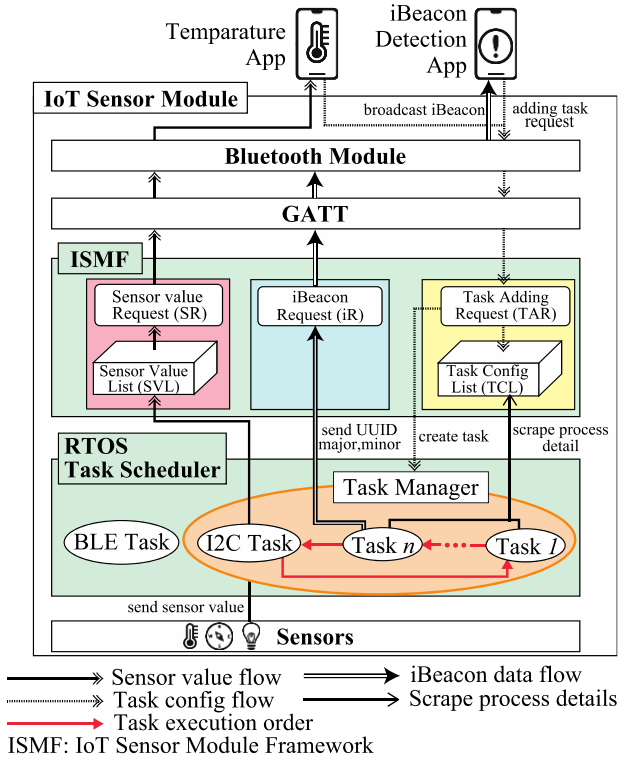

図１　システムモデル

2．提案システム

2.1　IoT センサモジュールフレームワーク

図 1 に本提案のシステムモデルを示す．提案システムは Generic Attribute Profile(GATT)と IoT センサモジュールフレームワーク，タスクマネージャから構成されている．マイコンとスマートフォン間の通信には BLE が利用されることが多い．そのため，提案システムでは GATT を採用している．フレームワークは GATT とタスクマネージャを繋ぐ API を担っている．フレームワークは Sensor value Request (SR)，Task Adding Request (TAR)と iBeacon Request (iR) の 3 つの関数群によって構成されている．SR はユーザからセンサ値の要求があった際に最新のセンサ値を格納している Sensor Value List (SVL) からセンサ値を取得し GATT に格納することで，

ユーザにセンサ値を提供する関数である．TAR はユーザ定義タスクの書き込み要求があった際に，ユーザ定義タスクを構造体の配列である Task Config List（TCL）に動的に格納し，ユーザ定義タスクを生成しタスクマネージャに共有する関数である． iR はユーザ定義タスクから iBeacon の発信要求があった際に，ユーザ定義タスクから iBeacon の UUID の情報などを受け取り，指定された形式の iBeacon を発信する関数である．

2.2 ユーザ定義タスクの分類

想定しているユーザ定義タスクは iBeacon 発信タスク，センサ値に基づく iBeacon 発信タスクとセンサ設定変更タスクの 3 つに分類される．iBeacon 発信タスクはユーザが指定した UUID, Major, Minor の iBeacon を発信するタスクである． TCL から UUID などの情報を取得し iR に渡すことで，iBeacon を発信する．センサ値に基づく iBeacon 発信タスクは，ユーザが指定した条件を満たした場合に iBeacon を発信可能とする．条件の例として，温度が 26℃の以上の時に iBeacon を発信などがある．条件を満たしていた場合に TCL から情報を取得し iR に渡すことで，iBeacon を発信する．センサ設定変更タスクは，各種センサの周波数や電源のオン・オフを指定可能なタスクである．TCL から各種センサへの設定情報を取得しセンサに書き込むことで，センサの設定変更を行う．

2.3 タスクマネージャ

タスクマネージャは，TAR によって生成されたユーザ定義タスクを動的配列に追加し，管理・実行する役割を担っている．タスクマネージャには，ユーザ定義タスクの他に最新のセンサ値を常に取得し Sensor Value List (SVL)に格納する I2C タスクが存在する．これら複数のタスクは， FreeRTOS が提供する Multi Task Scheduler を用いることで，並行処理と排他制御を行う．タスクマネージャは ユーザ定義タスクを繰り返し順次処理することで，複数サービスに同時対応する．具体的な処理として，まず，実行するユーザ定義タスク以外のユーザ定義タスクをスリープさせる．その後，実行するユーザ定義タスクはセマフォを取得し処理を実行する．実行終了後にセマフォをリリースし，タスクをスリープさせることで，タスクマネージャは次のユーザ定義タスクを起動させる．上記の手法でユーザ定義タスクの確実な実行を担保可能としている．また，タスクマネージャ自体がタスクとなっているため，BLE の処理を行うタスクである BLE タスクと並行動作可能である．それ故，タスクマネージャの処理を行いつつ BLE 通信や iBeacon の発信を行うことが可能である．

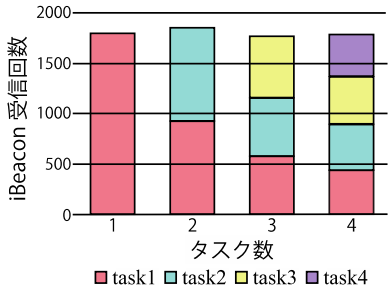

図 2 ユーザ定義タスク数の増加に応じた iBeacon 受信頻度の変化

3. 検証および評価

タスクマネージャにおけるユーザ定義タスクの実行頻度を均一化する機能の有効性を検証した．1 個から 4 個の iBeacon 発信タスクを書き込み，それぞれのユーザ定義タスクの個数における受信した iBeacon の種類と回数を確認することで実行頻度の均一性を確認した．実装デバイスとして [2]を利用した．検証時間として，それぞれのユーザ定義タスク数ごとに 30 分間 iBeacon を受信し，ログを取得した．

図 2 に，ユーザ定義タスク数の増加に応じた iBeacon 受信頻度の変化のグラフを示す．グラフから，タスク数 2 個の時は 2 種類の iBeacon を受信し，受信回数はタスク数 1 個の時と比較して約半分受信していた．タスク数が 3，4 個の時も同様に受信回数がそれぞれのタスクごとに約三分の一，約四分の一となっており，タスクの実行頻度が均一であることを確認した．

4. まとめ

本稿では，IoT センサモジュールにおける並行タスクの制御機構であるタスクマネージャの提案を行った．提案システムでは FreeRTOS を用いることで複数のユーザ定義タスクの排他制御・管理を行った．検証評価の結果，本提案がユーザ定義タスクの実行頻度を均一に行っていることを確認した．

参考文献

[1] Islam MM, Rahaman A, Islam MR. “Development of smart healthcare monitoring system in IoT environment,” SN Comput Sci. 2020;1(3):185, May 2020.

[2] Naito, Katsuhiro; Osaki, Shotaro, “Prototyping of New Concept BLE Device with Multiple Functions”, The 27th World Multi-Conference on Systemics, Cybernetics and Informatics: WMSCI 2023, September 2023.