IoT デバイスでセンサのアドレス取得による不具合の判別

牧 丈晴[1]　串田 高幸[1]

概要：マイクロコントローラとセンサ間でのデータ通信には I^2C（Inter-Integrated Circuit）が用いられる．マイクロコントローラは，センサからセンサデータを取得した後，プログラムを通してセンサデータを出力している．しかし，センサデータが取れていない時に，センサとプログラムのどちらに異常があるのか分からない課題がある．その問題を解決するために，本研究ではセンサの I^2C アドレスの取得結果の可否の判定によってセンサデータ取得の可否を判断する．判定結果をログ出力してサーバーに転送してファイルに保存する．判定方法は，I^2C アドレスの取得ができるかの可否により決める．評価手法としてセンサが外れてからログとして出力されるまでの時間を評価する．評価実験ではセンサのログを 5 秒おきで計 15 回出力する．センサの回路の線を物理的に抜き取った記録と，出力したログに記述されている I^2C アドレスの取得結果を比較する．結果，最初の 30 秒間は上記 2 つの記録の誤差は出なかった．そのことから本研究の提案手法であるセンサの I^2C アドレスの取得結果の可否の判定が出来ていることが分かる．

1. はじめに

背景

近年では物をインターネットに繋げて活用していく IoT（Internet of Things）が注目を浴びている．IoT デバイスはスマートフォンや通信機器から普及していき，現在では，家電，自動車，農業の環境で活用されている [1]．IoT デバイスにはデータを計測するためのセンサと，センサの管理やデータの変換サーバーと通信を行うためのマイクロコントローラが搭載されている．マイクロコントローラがセンサからデータを受信するために I^2C プロトコルの通信が用いられる [2]．I^2C はシリアル・データライン (SDA) とシリアル・クロックライン (SCL) の 2 本のバスラインを使用するシリアル通信の方式である[*1]．バスに接続されている各デバイスを固有のアドレスによって指定でき，マスター/スレイブモデルで通信をする．

課題

本研究における課題は IoT デバイスがセンサデータの取得に失敗した場合，センサに異常があるのか，プログラムに異常があるのか分からないことである．IoT デバイスはマイクロコントローラにセンサを搭載したデバイスと定義する．IoT デバイスの使用例として温室管理があげられる．温室管理を行うための IoT デバイスは温度データを収集して，サーバーにデータを送信する [3,4]．温度データを収集する IoT デバイスを運用するには，温度を測定するためのセンサと，センサを制御するためのプログラムが実装されているマイクロコントローラが必要である．マイクロコントローラは，センサからセンサデータを取得した後，プログラムを通してセンサデータを出力している．そのためセンサとプログラムのどちらかにでも異常があると，IoT デバイスからセンサデータを出力できない．システムの異常を特定するためにはログが用いられる．ログの例は HTTP サーバーのアクセスログがあげられる．ログを用いてシステムの状態を把握し，問題の状況や原因を推測する．例えば，HTTP サーバーのアクセスログは HTTP レスポンスステータスコードが 500 番台のとき，サーバーエラーを示す．しかし IoT デバイスはログを出力しないため，異常の発信元を特定するための作業が複雑である．例として実装されたプログラムの確認，マイクロコントローラが通電しているかの確認，センサが通電しているかの確認があげられる．

各章の概要

第 2 章では，本論文の関連研究を説明する．第 3 章では，本研究の課題を解決するための提案手法を説明する．第 4 章では，章の提案手法を実現するための実装と実験方法を説明する．第 5 章では，第 3 章の提案手法が課題を解決しているかを判断するための評価手法と分析手法説明する．第 6 章では，提案，実験，評価が本研究の課題を解決しているかを議論する．第 7 章では，本研究の課題解決による

[1] 東京工科大学コンピュータサイエンス学部
〒 192–0982 東京都八王子市片倉町 1404-1

[*1] https://www.nxp.com/docs/ja/user-guide/UM10204.pdf

貢献を説明する．

2. 関連研究

センサユニットを管理するマスターユニットを作成して，センサユニットの消費電力の削減と故障時にデータの損失を防ぐ研究がある [5]．故障発生時にデータ損失を防ぐことを目的としているため，故障を直すために故障の原因を特定する手段が不足している．

検出性能の低下予想に基づいて IoT デバイスの修理をスケジューリングによって，地理的に分散した環境での IoT デバイスのメンテナンスにかかる工数を削減する研究がある [6]．メンテナンスの工数削減を目的としているため，IoT デバイスの故障を直すための原因の特定する手段が不足している．

3. 提案方式

提案方式

本研究では，IoT デバイスの異常を I^2C アドレスを用いて特定する手法を提案する．本手法の目的はマイクロコントローラがセンサの I^2C アドレスを取得できるかの可否により，IoT デバイスに装着されているセンサの通電による異常を特定することである．前提としてマイクロコントローラが正常に稼働しているとする．マイクロコントローラがセンサの I^2C アドレスを取得できない場合，マイクロコントローラとセンサの I^2C 通信が失敗していることを表している．I^2C 通信の失敗がセンサ側にある場合，原因として IoT デバイスに装着しているセンサが指定した場所とは異なる場所に装着されているため，通電できていなことになる．そのためマイクロコントローラが I^2C アドレスを取得できない場合，センサが通電していないことを示している．

本提案では，マイクロコントローラがセンサの通電を確認した結果をログ出力した後に，ログを出力してサーバーに転送する．上記の全体図を図 1 に示す．図 1 は IoT デバイス，サーバーから構成されるおり，IoT デバイスにはマイクロコントローラと温度センサが搭載されている．IoT デバイスはマイクロコントローラと温度センサを搭載したデバイスをあらわす．温度センサは測定したセンサデータを I^2C 通信でマイクロコントローラに送信するセンサをあらわす．マイクロコントローラは温度センサの制御とサーバーにデータを送信するソフトウェアが，実装されている集積回路をあらわす．サーバーはログデータを受信してファイルに保存するソフトウェアが動作している．マイクロコントローラが温度センサから I^2C アドレス取得を試みる．取得結果によってマイクロコントローラは温度センサが通電しているかを判定する．I^2C アドレスを取得成功した場合はログに I^2C アドレスを出力する．また取得失敗した場合は，温度センサが通電していないことをログ出力する．ログには，センサのアドレスか通電していないことを示すテキストと，I^2C アドレス取得実行時刻を含むデータが記述されている．マイクロコントローラが上記のログを出力してサーバーに送信する．

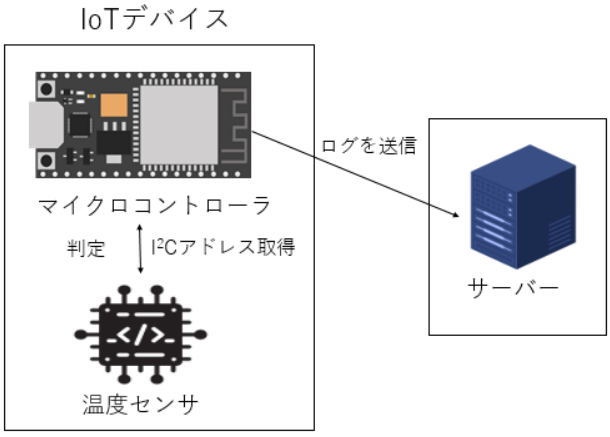

図 1 提案の全体図

提案の流れ図を図 2 に示す．図 2 の流れを以下に説明する．

(1) I^2C アドレスを取得
(2) I^2C アドレスを送信
(3) I^2C アドレスを取得できているかを判定して，判定結果をログ出力
(4) ログを送信

(1) の「I^2C アドレスを取得」はマイクロコントローラがセンサから I^2C アドレスを取得する命令を実行する．(2) の「I^2C アドレスを送信」はセンサから I^2C アドレスを取得する．また、この際センサから I^2C アドレスが取得できない場合がある．(3) の「I^2C アドレスを取得できているかを判定して，判定結果をログ出力」はマイクロコントローラがセンサから I^2C アドレスを取得できているかを判定する．I^2C アドレスの取得に成功した場合は，ログに I^2C アドレスを出力する．I^2C アドレスの取得に失敗した場合は，ログにテキストでセンサがないことを出力する．(4)「ログを送信」は (3) で出力したログデータをサーバーに転送してサーバー内でファイルに保存される．

ユースケース・シナリオ

本研究の提案を用いたユースケースシナリオについて説明する．ユースケースシナリオでは IoT デバイスの実装時を想定している．実装時に試運転で IoT デバイスを稼働する際に，センサデータが取得できなかったときに使用する．本提案のシステムを導入することで，センサデータがないときにセンサに異常があるのかプログラムに異常があるのか，サーバーに送信されたログを閲覧することで判断できる．センサに異常があるかプログラムに異常があるか判別

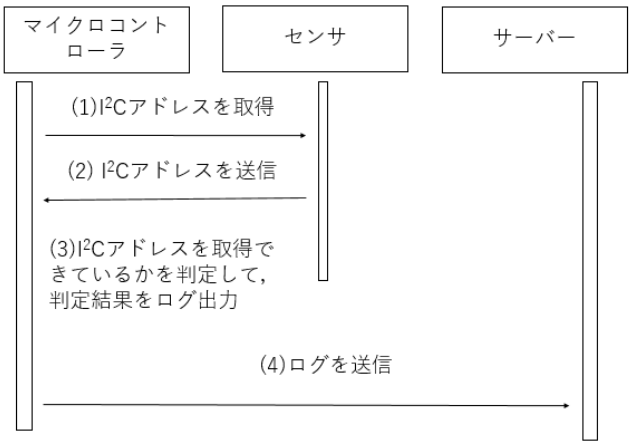

図 2 提案の流れ図

できることで．エラーが起きたときにエラーを解消するための作業工数を減らすことができる．

4. 実装と実験方法

実装

本研究の提案手法であるセンサの通電の確認とログを出力するソフトウェアをマイクロコントローラに実装した．図 3 は通電の確認のフローチャートである．「I^2C アドレスの取得実行」はマイクロコントローラがセンサの I^2C アドレスを取得するプログラムを実行する．「I^2C アドレス取得できた」ではマイクロコントローラがセンサから I^2C アドレスを取得できているかを判定する．判定結果が Yes の場合「I^2C アドレスをログ出力する」．判定結果が No の場合「センサが通電していないことをログ出力する」．

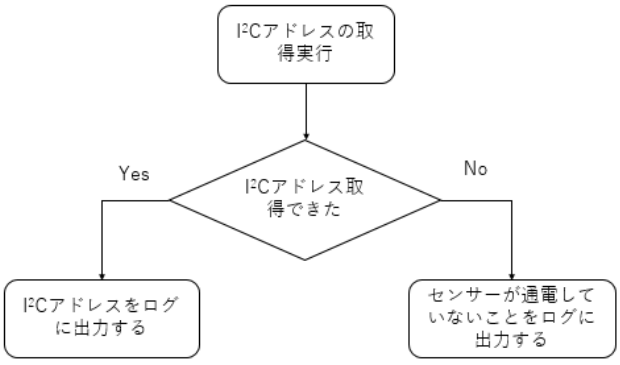

図 3 通電の確認のフローチャート

図 4 はログ生成するソフトウェアの構成図である．構成はセンサとマイクロコントローラ，サーバーの 3 つがある．マイクロコントローラ内は 4 つのソフトウェアがあり，以下に記述する．

- マイクロコントローラの名称
- I^2C アドレス取得実行時刻
- MicroPython のバージョン
- I^2C アドレス取得結果

図 4 にある I^2C アドレス取得結果はセンサからデータを取得するソフトウェアである．I^2C アドレス取得結果はマイクロコントローラがセンサの I^2C アドレスを取得し，センサの通電の可否を判定して出力するソフトウェアである．図 4 にあるマイクロコントローラの名称と I^2C アドレス取得実行時刻と MicroPython のバージョンは，マイクロコントローラ内でデータを取得するソフトウェアである．マイクロコントローラの名称は本提案を実行しているマイクロコントローラ機器名（例えば ESP32）を取得するソフトウェアである．I^2C アドレス取得実行時刻は I^2C アドレス取得結果を実行した際の，時刻を出力するソフトウェアである．MicroPython のバージョンはマイクロコントローラに実装しているプログラム言語を出力するソフトウェアである．上記 4 つのソフトウェアからのデータをログ出力する．出力したログメッセージを HTTP でサーバーに送信する．サーバーは受信したログメッセージをログファイルに保存する．

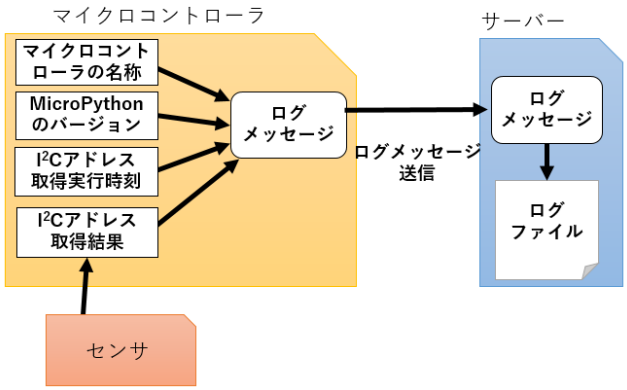

図 4 ソフトウェア構成図

実験環境

表 1 は実験対応表である．実験に使用した機器はサーバー側とクライアント側に分かれる．サーバー側は VMware ESXi を使用し，VM でサーバーを構築した．VM は Ubuntu20.04.2LTS で構築した．サーバー側の実装は Python 3.8.10 を使用し，Flask2.0.2 で HTTP サーバーを構築した．クライアント側から送信されたログはテキストファイルで保存される．クライアント側はマイクロコントローラとして ESP32 と温度センサとして S-5851A を使用する．実装した言語は MicroPython 1.17.0 である．ログは HTTP でサーバー側に送信した．送信したログを図 5 に示す．ログに含まれる情報は I^2C アドレス取得実行時刻である「2022/2/9 18:41:55」と I^2C アドレス取得結果である「72」，MicroPython のバージョンである「micropython(1.17.0)」，マイクロコントローラの名称である「esp32」が記載されている．また I^2C アドレス取得結果でセンサが通電していないと判定されたログを図 6 に示

す．ログに含まれる情報は情報は I^2C アドレス取得時刻である「2022/2/9 18:42:1」と I^2C アドレス取得結果である「No sensor」，MicroPython のバージョンである「micropython(1.17.0)」，マイクロコントローラの名称である「esp32」が記載されている．

表 1 実験対応表

VM	Ubuntu
HTTP サーバー	Flask
マイクロコントローラ	ESP32
温度センサ	S-5851A

図 5 センサが通電している場合のログ出力例

図 6 センサが通電していない場合のログ出力例

5. 評価手法と分析手法

回路から外れたことによってセンサデータが取れなかったときに，ログの出力でセンサが外れたことを判断する状況を想定して実験する．センサが外れてからログとして出力されるまでの時間を評価する．センサの回路の線を物理的に抜き取りログと実際のセンサの差を記録する．またログの出力は 5 秒に 1 回とする．

図 7 は実験結果を示す．凡例はセンサのつけ外しとログの出力がある．横軸は時間を表し，単位は秒である．縦軸は True or False すなわち，センサが通電しているかの可否をグラフに示している．図 7 より，35 秒以降にセンサのつけ外しとログの出力に差が生じている．具体的にはセンサが外れているにもかかわらずログの出力はセンサがついていると出力している．センサのつけ外しとログの出力に差が生じる理由は定期的にログの出力をしているのではなく，ログの出力をしてから 5 秒待ってログの出力をしているためである．つまり，ログを出力する時間と 5 秒を足した時間の周期でログを出力していることになる．証拠として 65 秒以降にログの出力がセンサがついていることを示していることがあげられる．実験終了直後にセンサを付けたことによってログの出力もセンサがついていることを示した．

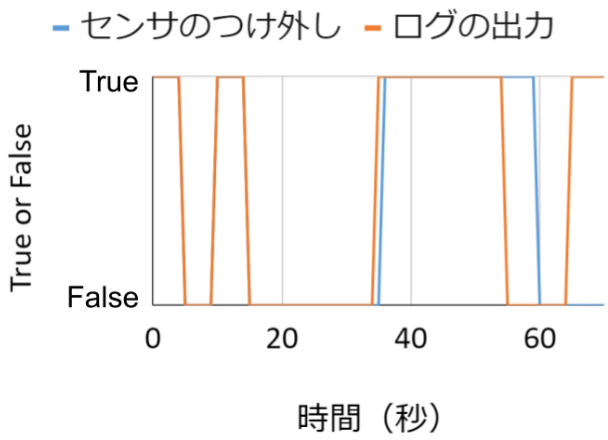

図 7 実験結果図

6. 議論

評価手法と実験にて本提のソフトウェアを繰り返し実行した際，実行時間と待機時間の周期でログを出力したため，定期的にログの出力ができなかった．現状稼働時間が長くなればなるほど，ログを出力する時間のずれが大きくなる．プログラムの実行時間に左右されないように，実行周期を指定するときは一定時間ごとに時刻を指定して，指定した時刻にとログを出力する．指定した時刻にログを出力することで，ログを取りたい時刻とログを出力する時刻に差が生じることが無くなる．

本研究では I^2C アドレスの取得結果を用いて，センサの通電状況を判定して結果をログとして出力できた．これにより IoT デバイスの異常が，センサとプログラムのどちらが原因で発生しているのか分かった．しかし今回の提案ではセンサに異常があるのは判定できるが，センサの異常個所を特定することは出来ない．センサの異常個所の特定方法として，電流センサを用いた通電の確認があげられる．本研究ではセンサから I^2C アドレスが取得できない場合，センサが通電していないと判定している．しかし I^2C アドレスの取得に失敗してセンサが通電していない場合，センサではなく回路の断線によりセンサのアドレスが取得できない．センサに異常があるのか回路に異常があるのかを判断するために電流センサを用いた異常個所の特定をする．

7. おわりに

本論文では IoT デバイスがセンサデータ取得失敗した際に，プログラムとセンサのどちらに異常があるのか分からない課題をあげた．課題に対し，マイクロコントローラがセンサの I^2C アドレスの取得結果を用いた異常の判定と判定結果をログ出力した．評価手法としてセンサが外れてからログとして出力されるまでの時間を評価する．評価実験ではセンサのログを 5 秒おきで計 15 回出力する．センサの回路の線を物理的に抜き取った記録と，出力したログに

記述されている I^2C アドレスの取得結果を比較する．結果，最初の 30 秒間は上記 2 つの記録の誤差は出なかった．そのことから本研究の提案手法であるセンサの I^2C アドレスの取得結果の可否の判定が出来ていることが分かる．

参考文献

[1] Gubbi, J., Buyya, R., Marusic, S. and Palaniswami, M.: Internet of Things (IoT): A vision, architectural elements, and future directions, *Future generation computer systems*, Vol. 29, No. 7, pp. 1645–1660 (2013).

[2] Mankar, J., Darode, C., Trivedi, K., Kanoje, M. and Shahare, P.: Review of I2C protocol, *International Journal of Research in Advent Technology*, Vol. 2, No. 1 (2014).

[3] Saraswathi, D., Manibharathy, P., Gokulnath, R., Sureshkumar, E. and Karthikeyan, K.: Automation of Hydroponics Green House Farming using IOT, *2018 IEEE International Conference on System, Computation, Automation and Networking (ICSCA)*, pp. 1–4 (online), DOI: 10.1109/ICSCAN.2018.8541251 (2018).

[4] Sihombing, P., Karina, N. A., Tarigan, J. T. and Syarif, M. I.: Automated hydroponics nutrition plants systems using arduino uno microcontroller based on android, *Journal of Physics: Conference Series*, Vol. 978, p. 012014 (online), DOI: 10.1088/1742-6596/978/1/012014 (2018).

[5] Choubey, P. K., Pateria, S., Saxena, A., SB, V. P. C., Jha, K. K. and PM, S. B.: Power efficient, bandwidth optimized and fault tolerant sensor management for IOT in Smart Home, *2015 IEEE International Advance Computing Conference (IACC)*, pp. 366–370 (online), DOI: 10.1109/IADCC.2015.7154732 (2015).

[6] Anu, H., Chen, J., Shi, W., Hou, J., Liang, B. and Qin, B.: An Approach to Recommendation of Verbosity Log Levels Based on Logging Intention, *2019 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pp. 125–134 (online), DOI: 10.1109/ICSME.2019.00022 (2019).