# RaspberryPi 用拡張ボード PiRT-Unit を用いた I2C デバイスの RT コンポーネント実装

○関山守(産総研)　安藤慶昭(産総研)　原功(産総研)　神徳徹雄(産総研)
片見剛人((有)ウィン電子工業)　深澤篤史((有)ウィン電子工業)　谷川民生(産総研)

## 1. 背景と目的

携帯電話のような可搬型個人用情報機器に内蔵されるセンサのデータを情報処理の対象として活用するような事例が増える[1]なかで，ユーザーが必要とするセンサを自分で持ち歩いたり，測定対象物に結合させたりすることができるような可搬状態にしたうえで，データを取得することに対する要求も増加している．この要求を満たすシステムを提案・実現する際には初期導入費用・手間や作業などに必要とする労力等のような導入コストが低減されていることが望ましい．

センサを可搬とする手段としてはArduino，Armadillo のような接続用IOピンを持つマイコンボードを用いる事が考えられる．マイコンボードを用いてセンサの制御とデータ取得を行うには若干の電子工作とマイコンボード上で動作するドライバの記述が必要である．だがマイコンボードの価格は千差万別であるためマイコンボードを用いることで費用面での導入コストが低減されるわけではない．また若干の電子工作とドライバの記述という必要事項は導入コストとしては必ずしも低いものではない. さらにRTミドルウェア[2]の環境下でセンサを可搬状態にする場合には，RTミドルウェアの動作環境を整備する必要がある．この事項により導入コストがさらに増加すると考えるのが一般的であった．RasberryPi財団が発売している低価格LinuxマイコンボードRaspberryPi(図1)は前述の状況を一変させた．基板上に存在するGPIOピンはユーザーが希望する信号が入出力可能である．GPIOピンの一部はI2C, SPIおよびシリアル通信等の一般的な規格用途として割り当てられている．この結果センサを接続するために必要となる電子工作が簡単化された．RaspberryPi 用の OS として Debian 系 Linux である Raspbianを用いる事が推奨されているが，Raspbian上ではI2C, SPI等の通信規格については標準ドライバが用意されているためドライバ開発のためのコストは不要となる．さらにRaspbian上でRTミドルウェアの動作環境を整備する事については，OpenRTM-aistの公式サイト上に必要な情報とソフトウェアが準備されており簡単に導入する事が可能となっている．以上を踏まえるとRaspberryPiを用いる事によってセンサを可搬状態でRTコンポーネント化するための導入コストは非常に低減される，という事が理解できる．

本稿では, I2CデバイスについてRaspberryPiを用いてのRTコンポーネント化，そして可搬化するまでに必要となる事柄について論ずる．可搬化のためのデバイスの試験と評価は必須であり，そのための簡便な手法の提案は有意であるため以下に詳述する．

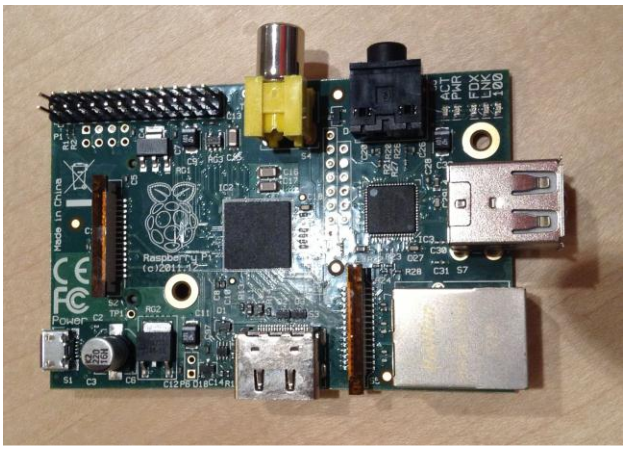

図 1　RaspberryPi

## 2. PiRT-Unit による I2C デバイス評価環境

### 2.1 RaspberryPi による I2C デバイス操作

I2C はフィリップス社が提唱したデバイス間シリアル通信方式であり，センサを含む多数のデバイスがデバイス間通信の方式として採用している．この通信方式では複数のデバイスがデータ用およびクロック用の 2 本のオープンコレクタ信号線に対してバス結合しており，10Kbits/sec～3.4Mbits/sec までの速度でデータ授受を行う．I2C 対応のデバイスは通信の為の ID としてそれぞれが固有の 7bit アドレスを持っている．このアドレスを用いる事により同一バス上では最大 112 個の I2C デバイスのノードが通信する事が可能である．データ用信号線を用いて制御のためのコマンド送受信と実際のデータの授受を行う設計となっている．

RaspberryPi は I2C 通信規格対応のデータ用およびクロック用専用入出力ピンを GPIO ポートなどに有しており，通信速度はデフォルトで 100Kbits/sec となっている．RaspberryPi を用いて I2C デバイスを制御するためにはハードウェア的に双方を結合する事と RaspberryPi 上で I2C 用のドライバソフトウェアを有効化する必要がある．ハードウェア的な結合については I2C 専用入出力ピンに対して対象となる I2C デバイスをバス結合させるだけで条件を満たす．(図 2)この際にオープンコレクタ信号線での結合に必要なプルアップ抵抗は RaspberryPi 基板上に実装

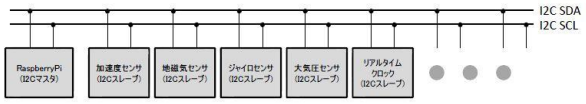


図 2　I2C バス

されているのでユーザーが準備する必要はない．I2C 用のドライバソフトウェアの有効化については OS である Raspbian 内のドライバモジュール設定を変更する事が必要である．

ハードウェア的な結合とドライバモジュール設定の変更を行うだけでファイルシステム上に /dev/i2c-0, /dev/i2c-1 という名称の I2C デバイス制御用のデバイスファイルが生成される．このデバイスファイルへのアクセスを介してデバイスの制御とデータの授受が可能になる． すなわちデバイスの RT コンポーネント化を行う際には RT コンポーネントのソースコード内に上記デバイスファイルへの読み書きといったアクセスを記述する事になる．

RaspberryPi を用いて I2C デバイスを RT コンポーネント化するために必要な事項は I2C デバイスの種類によらず一般化されており，RT コンポーネント化は簡便に実現が可能であり，ユーザーは自分が使用したい分野の I2C デバイスを複数の中から用途，価格，性能に応じて選定することに注力することができる．

### 2.2 PiRT-Unit

I2C デバイス選定にかかる作業量が減少することにより選定のための試験および評価を十全に行うことが可能となる．RaspberryPi を用いて試験・評価を行う場合には GPIO ポートから必要となる I2C 用のデータ信号線とクロック信号線を評価対象の I2C デバイスに接続することになる．

PiRT-Unit(図 3)は産総研が開発した RaspberryPi 用 IO 拡張基板である．この基板は SPI 経由での AD/DA 変換ポート，XBee モジュールおよび RS232C 接続シリアル通信コネクタ等を有し，基板を用いる事で GPIO ポート経由の IO を簡単に利用する事が可能となる．PiRT-Unit を用いて I2C デバイス選定の手法を

図 3　PiRT-Unit を搭載した RaspberryPi

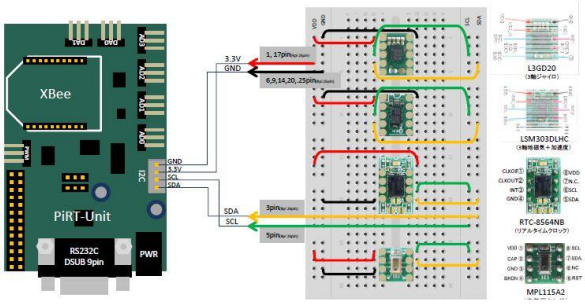


図 4　評価回路の単純化

さらに簡単化する．

PiRT-Unit 上には I2C 接続用 4 ピンコネクタが存在する．I2C 用のデータ信号線とクロック信号線の他に基板から I2C デバイスへ 3.3V/GND の電源を供給することを目的とした接続用コネクタである．これら 2 本の信号線と 2 本の電源線をブレッドボードのバス部分に接続することで，ブレッドボード上での複数 I2C デバイスの同時評価を単純な回路の形で実現する事ができる．(図 4)構成回路の単純化は評価の際のミスの低減と作業の効率化に資するため重要である．

なお PiRT-Unit は RaspberryPi による I2C デバイス制御において必須の基板ではない．だが I2C デバイスの選定・試験実装のような評価段階においてはこの基板を利用することにより評価手法が単純化・簡単化する．ゆえに安心かつ効率的に I2C デバイスの評価を行う際には PiRT-Unit があると便利である．

## 3.　I2C デバイスの RT コンポーネント実装

PiRT-Unit とブレッドボードを使用して，試験的に複数の I2C デバイスを同時に RaspberryPi に接続した状態で RT コンポーネント化した実例を報告する．RaspberryPi 上での RT ミドルウェアの環境としては OpenRTM-aist の公式サイトより提供されている C++ 版 OpenRTM-aist 1.1.0-RELEASE を用いた．

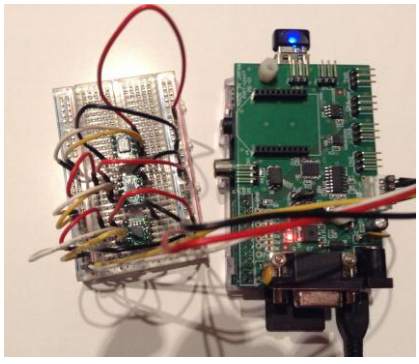

図 5　試験用回路

I2C デバイスとしては ST マイクロエレクトロニクス社の LSM303DLHC（３軸地磁気センサ＋３軸加速

度センサモジュール）および L3GD20（3 軸ジャイロセンサジュール）を用いた．LSM303DLHC は I2C バス上では地磁気センサと加速度センサがそれぞれ別個のモジュールとして扱われる．(図 5)

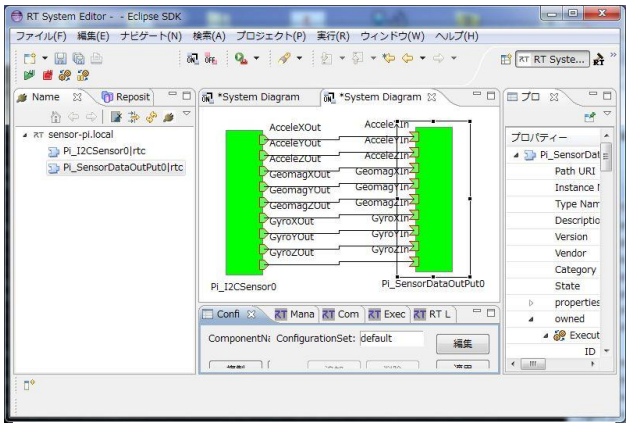


図 6　試験用 RT コンポーネント

RT コンポーネントの作成については，Windows 上で動作する RTC ビルダーを用いてひな形を作成後に Windows 上のエディターでソースコードの内容を編集した．その後，RT コンポーネントのプロジェクトをフォルダ（ディレクトリ）ごと RaspberryPi にコピーする．以降の作業はすべて RaspberryPi 上で行うことができる．RaspberryPi 上で Cmake, make を行い，RT コンポーネントを生成する．（図 6）

RT コンポーネントの動作内容はそれぞれの I2C デバイスのイニシャライズを onInitialize ルーチンで行い，周期実行を意図して onExecute ルーチンにて各センサデバイスの 3 軸データ数値を取得，さらに各データを 32bit 幅の整数値に変換して TimedLong のデータポートから出力を行うところまで行うものとした．RT コンポーネントの実行周期については rtc.conf 内の exec_cxt.periodic.rate の値を 1Hz とすることで指定した．なお実際のデータ取得周期は体感でおおよそ 1 秒であった．動作例を図に示す．表示している値はセンサが出力している実測値であり補正を行っていない．(図 7)

## 4.　考察

一台の RaspberryPi と一つの I2C バスを用いて一度にデータを取得できる I2C デバイスの個数について考えてみる．この場合は通信速度と授受データの個数，解像度が問題となる．今回の 2 個 3 種類の I2C デバイスにおいては 3 軸 1 セットのデータを受信する為にはデバイス間通信の調停やデバイス制御コマンドを考慮に入れると 20 バイト程度のデータ授受が必要となる．単バイト送受での調停も含めて考えると，20 バイトのデータ授受にかかるクロック数は 200～220 クロックと見積もると，今回のシステムにおいて 3 個のセンサデータを更新する場合にはデータ授受だけで 700 クロックほどかかる見込みとなる．実際にはセンサの応答には時間がかかるためそれも考慮に入れる必要がある．今回用いた 2 個 3 種類の I2C デバイスの応答時間は 5 $\mu$ 秒と非常に小

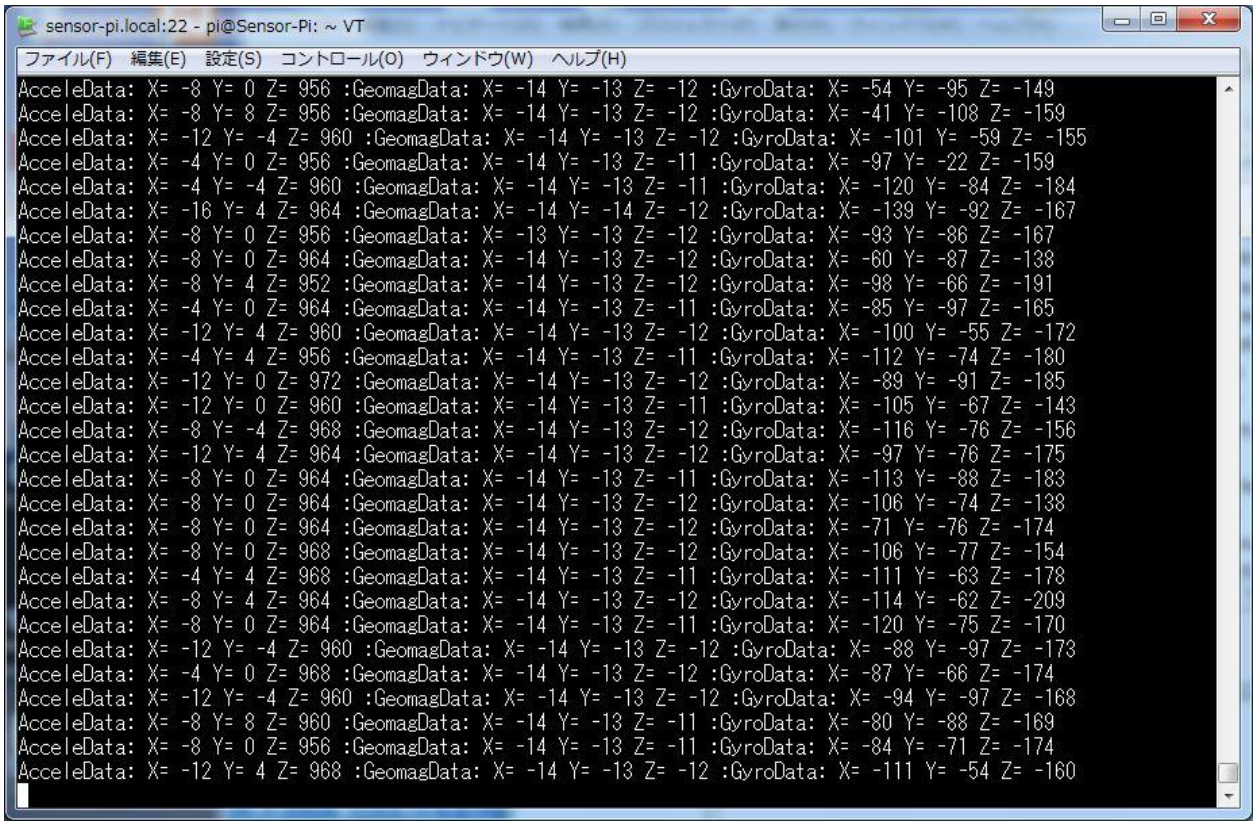


図 7　センサ値出力実行例

さい値であるため，RaspberryPi の I2C クロック周波数がデフォルトで 100KHz であることを考えると応答時間は 1 クロック未満であり全体のデータ授受と比較しても影響はほぼ無いと言える．今回の例のように RT コンポーネントの動作周期を 1Hz として，そのタイミングでセンサデータを取得することを考えると現在の 130 倍の数のセンサを用いることも可能となる．

必要となるセンサの種類に応じてセンサデータの取得周期は変化させることが望ましい．この場合の RT コンポーネント化において注意しなければならないことは，I2C バスの排他制御である．これはそのまま I2C デバイスファイルの排他制御を実現する事で実現できると考える．

目的とするセンサが I2C デバイスである場合は上記の事例のように非常に簡単に RT コンポーネント化が可能である．また I2C デバイスの試験実装・評価・選定が PiRT-Unit を用いる事で簡便かつ効率的に行うことが可能であることがわかる．I2C デバイスの評価を効率的に行うことが可能となるため，I2C デバイスであるセンサならば最終的な可搬化実装も低コストでの実現が可能となる．

RaspberryPi の価格と可搬化・RT コンポーネント化の手法の低コスト性をふまえると，RaspberryPi によって可搬化される I2C デバイスは可搬型個人用情報機器の高性能な代替となりうる．なぜならば，必要とするセンサは複数の候補の中から簡単に選ぶことができ，自分が必要とする形で RT コンポーネント化が可能であるからである．さらには RaspberryPi のサイズなどを考えると後付の装着型測定用センサというような新しい用途でのセンサ活用が簡単に実現できる．

## 5. まとめ

センサを可搬の状態で RT コンポーネント化する場面において，センサが I2C デバイスであるならば RaspberryPi を用いることで簡単にその目的を達成する事が可能である．可搬化・コンポーネント化を行う前段階として同一種類のセンサから使用するセンサを評価･選定する場合には PiRT-Unit を用いる事で評価・選定の効率を上げることが可能である．実際の I2C デバイスの RT コンポーネント実装例を挙げ，PiRT-Unit の効果とセンサ可搬化・RT コンポーネント化について論じた．

RaspberryPi は安価であり，この手法による可搬化･RT コンポーネント化はセンサの用途と活用の幅を広げる事が可能であると考えている．今後は RaspberryPi を実際のセンサデータロギングに活用することでこの手法の有効性を示していく．

## 参 考 文 献

[1] 西尾信彦，他:“実世界に広がる装着型センサを用いた行動センシングとその応用”，情報処理学会学会誌，vol. 54, no. 6, pp.562-605, 2013.

[2] Noriaki ANDO, Takashi SUEHIRO, Kosei KITAGAKI, Tetsuo KOTOKU, Woo-Keun Yoon, "RT-Middleware: Distributed Component Middleware for RT (Robot Technology)", 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS2005), pp.3555-3560, 2005.08, Edmonton, Canada.