EPICS 内蔵小型デバイスサーバの開発

DEVELOPMENT OF COMPACT DEVICE SERVER WITH EPICS

帯名 崇#, A), 路川 徹也 B)
Takashi Obina#, A), Tetsuya Michikawa B)
A) KEK, high Energy Accelerator Research Institute
B) East Japan Institute of Technology Co., Ltd.

Abstract

Large accelerator control system must handle a lot of instruments that are located in wide-spread area in many buildings. Many kinds of software framework for such purpose have been developed, such as EPICS, TANGO, MADOCA, etc. On the other hand, most of commercial products available on the market do not embed such a software tools by default. We have developed a compact device server to control four devices with RS232C interfaces. The device server works as an EPICS Input/Output Controller (IOC). In this paper, details of design, fabrication and results of long-term test will be reported.

1. はじめに

加速器をはじめとした巨大な実験装置を制御するには数多くのデバイスを遠隔地から操作する必要がある。これらのデバイスを有機的かつ機能的に制御するためには、なんらかの制御フレームワークが必要となるため、EPICS[1], MADOCA, TANGO など多くのソフトウェアが開発されてきた。その一方で、市販の製品を購入した際には RS232C や GPIB など標準的なインターフェースを持つ製品はあるものの、最初から加速器制御フレームワークに対応している製品はほとんど無い。このような状況を打開するため、我々は RS232C インターフェースを持つ小型デバイスサーバを開発し、各種デバイスを制御フレームワークに容易に対応させることを目指した。本稿では EPICS を内蔵したデバイスサーバの開発について、ハードウェアおよびソフトウェアの詳細と加速器での運用実績について報告する。

2. 開発の動機と方針

2.1 開発の動機

KEK においては多くの加速器で EPICS と呼ばれるネットワーク分散型ソフトウェアフレームワークを使用している。詳細は参考文献[1]に譲るが、基本的な考え方として全体をハードウェア制御層、ネットワーク層、プレゼンテーション層の３階層構造に分けるものである。図１に概略を示すように、ハードウェアを制御するための入出力コントローラ（Input/Output controller, IOC）が個別の機器に依存する部分を担当し、最上位のプレゼンテーション層に位置する各種ユーザーインターフェース（OPI）や制御プログラム群とはチャンネルアクセス（Channel Access, CA）と呼ばれるプロトコルを使用して状態変数（Process Variables, PV）をやりとりすることで制御を行っている。IOC としては VME をはじめとしたバスシステムが多く用いられるほか、PLC や Linux/Windows 等を搭載した PC が使用されることもある。

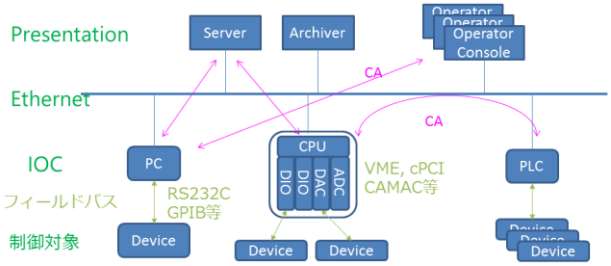

Figure 1: Overview of EPICS control system. Channel Access (CA) protocol is used to communicate among IOCs, control program or operator interfaces (OPIs).

なにか新しいハードウェアを市販品として購入した場合にその機器が直接 CA プロトコルを理解していれば制御の面では最も簡便で理想的である。これならば電源を投入して Ethernet に接続するだけですぐに EPICS 環境で制御できるような状況になるため、追加で何もする必要がない。しかしながら現実的にはそのような機器は市場にはほとんどなく、搭載されているインターフェース（例えば RS232C や GPIB、デジタル I/O、アナログ I/O 等）に応じて IOC を設置し、そこで稼働するソフトウェアはハードあるいはソフトウェア担当者が個別の機器に応じて開発しているのが現状である。

ここではさらに具体的に RS232C の機器制御を例に挙げる。RS232C は非常に古くから存在するシリアルインターフェースであり、多くの機器が標準的に装備している。制御用コンピュータ（以下では IOC とする）にはシリアルインターフェースが１～２個程度はじめから装着されていることも多く、対象物が少ないときにはこれで十分であった。対象物の数が多い場合には IOC に増設ボードを装着するこ

takashi.obina@kek.jp

ともよく行われてきた。しかし、このような構成をとった場合の大きな欠点として IOC と対象物との距離が近く（数 m ～ 数 10 m 程度まで）なければいけないこと、特殊なハードウェアが必要になるため IOC が故障したときの対応が非常に困難になること等が挙げられる。そのほかデバッグも当該 IOC 上でしかできないという欠点もあった。近年ではこれらの問題を解決するためにシリアル―Ethernet メディアコンバータを使用することが多い。IOC とメディアコンバータとの間はソケット通信を使用してデータのやり取りを行うため、IOC の設置場所はネットワーク上のどこにあっても良く、距離の問題は無い。さらに、１つのマシンから複数のメディアコンバータとの通信を行うこともできるため、特殊なハードウェアが不要になって耐障害性の面で有利になった。この構成での欠点は主として２つある。１つはメディアコンバータと IOC 間のネットワーク通信と CA プロトコルが流れるネットワークとは論理的には全く別の通信であるにもかかわらず、物理メディアとしてはおなじ Ethernet 線を使うため冗長であり、ネットワークトラフィックの増加につながりやすいことである。もう１つは、CA プロトコルは標準化されているため、初期化方法やなんらかの障害によってコネクションが切断された場合の再接続方法などが規定されているのに対し、メディアコンバータとの通信は機器に依存する部分であるため障害後の復旧に問題が生じる場合があることである。特に後者はメディアコンバータの製品によって違いがあり、多種多数の機器が導入されることは管理コストの増加につながってしまう。

以上の問題を解決し、少しでも理想形に近づけるため、我々は EPICS を内蔵した小型デバイスサーバを開発することにした。

2.2 開発の方針

最初に決定すべき項目はこのデバイスサーバ全てを最初から開発するのか、あるいは市販品をベースにした開発をするのか、であった。これには最終的に必用なハードウェアの数量と、開発にかかるコスト（金銭面と時間、労力まで含めて）とに大きく関わる。現状のマンパワーでベースとなるハードウェアと OS まで含めたソフトウェア開発を遂行することはほぼ不可能であり、当面必要となる数量も１０個以下と大量ではなかったため、今回の目的に適した市販の小型ボードを選定する方針とした。今後の応用を常に念頭におき、できるかぎり拡張性・柔軟性に富むハードウェアであることを条件に入れている。

近年の回路集積技術の進歩の恩恵を受けて、小型デバイスサーバに使用可能なハードウェアは各社から多くの製品が出ている。比較対象にしたのは以下の製品である：1) BeagleBone Black[2], 2) Raspberry Pi[3], 3) Arduino Uno[4], 4) Armadillo[5], 5) Suzaku[6], 6) Galileo[7]。これらの製品のうち、比較的業務用途といえる製品はアットマークテクノ社製の Armadillo および Suzaku である。I/O ピンの数は十分であり、EPICS を稼働させた実績もいくつか存在している[8-10]。設計年代が少し古いということもあって CPU 能力は控え目でメモリもあまり多く搭載していないこと、価格もほかの製品に比べて高価であることなどの理由で今回の候補からは除外した。Galileo は設計を始めた段階ではまだ市販されていなかったが、CPU 能力は十分であるがサイズが大きいこと、消費電力の面からも組み込み用途というよりは小型の PC という傾向が強い。Intel プロセッサに限定されるアプリケーションがあれば今後候補に上ると考えている。1)～3)の３種類のボードはいずれも個人レベルでも購入可能な程度に安価であるため近年非常に人気が高く、ユーザー間の議論や情報共有が広く行われている製品である。これらについての比較表を表１に示す。Arduino Uno は最も小型かつ安価であるがホスト PC に USB 接続して使用することが前提になっており OS やイーサネットインターフェースを搭載することに問題が出てくる。Raspberry Pi と BeagleBone Black は似通ったスペックである。最終的に I/O ピン数が多いこと、ハードウェアで RS232C ポートが５ポートサポートされることを重視して、BeagleBone Black （以下 BBB と記述）を採用することとした。

Table 1: Comparison of three boards

	BeagleBone Black(Rev.C)	Raspberry Pi (ModelB)	Arduino Uno
CPU	TI AM3359 (ARM Cortex-A8)	Broadcom BCM2835 (ARM11)	Atmel ATMega 328
Speed	1 GHz	700 MHz	16 MHz
RAM	512MB DDR3L@400MHz	512MB SDRAM@400MHz	2 KB
Storage	Onboard eMMC 4GB microSD slot	SD slot	32 KB Flash
I/O(GPIO)	65	8	20
ADC	7	n/a	8
Ethernet	10/100 x1	10/100 x1	n/a
OS	Debian(default) Angstrome Linux Fedora etc	Raspbian(Debian) Pidora(Fedora) ARCH linux etc	n/a
Size [mm]	86.4 x 53.3	85.6 x 54	75 x 53.3

BBB の特徴としてはソフトウェアのみならずハードウェアまで含めてオープンソース化されていることが挙げられ、ライセンス形態はクリエイティブコモンズとなっている。ドキュメント類も充実しており、“Cape”と呼ばれる拡張ボード規格についてもよく整備されている[11]。市販品の Cape も数多く販売されるなど、活動は活発に行われている。BBB は実際の生産品に組み込むというよりはプロトタイプ製作としての用途を前提にした製品であるが、自己責任において機器に組み込んで使用することは妨げていない[12]。

2.3 製作ターゲット

今回製作するデバイスサーバの最初の目標は EPICS を内蔵した 4 ポート RS232C サーバである。これは後述するコンパクト ERL 加速器における放射線モニタで使用することを主眼にしている。前述したように BBB は SoC チップ内に UART (Universal Asynchronous Receiver Transmitter)回路を内蔵してお

り、ブート時のコンフィグレーションでこれらのポートを使うように指定するのみでシリアル通信が可能となる。

BBB を起動するとき、デフォルトでは HDMI 出力有効でシリアルポートは１ポートのみ使用可能な状態となる。BBB には定義済みの動作モードが８種類用意されていてピンごとにモードを指定可能である。Cape ボードを装着し、EEPROM に書かれたファームウェアに対応するドライバを読み込むように設定することで電源投入時にモードを決定できる。４ポートの RS232C を使うには使用する全ピンをモード６に設定するのが簡単である。また、BBB の I/O 電圧レベルは 3.3 V であるため RS232C 用のレベルコンバータも搭載した Cape を設計した。回路図および基板図面を図２に示す。

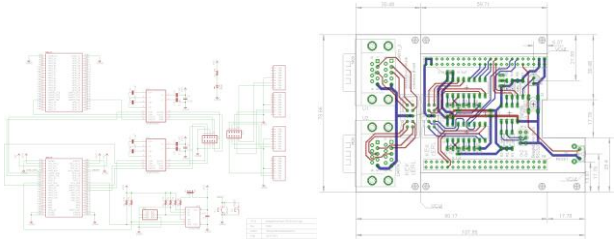

Figure 2: Circuit and board schematic of mezzanine card. Simple double-sided board is enough to mount all parts.

基板製作はインターネット経由でプリント基板メーカに発注した[13]。基板サイズが小さいため、安価かつ１週間以内には基板が送られてくる。オプションとして部品実装まで発注することもできるが、DIP パッケージの場合は価格が高い。量産するならば表面実装パーツでの設計にする必要がある。

シャーシについても安価に製作するため、特殊なサイズを避けてタカチ社製の標準ケース kc5-13-10 をベースにして追加工することにした[14]。加工用 CAD 図面を作成して送るのみで製作可能である。

図３に製作したデバイスサーバの写真を示す。子基板である Cape は BBB の上に装着される。Dsub コネクタのサイズが大きいため全体のサイズを極端に小さくすることは困難である。特殊なシリアルコネクタを採用すれば小型化は可能であるが、本来の目的である「簡便に接続できるデバイスサーバ」という主旨を優先して Dsub-9pin コネクタを４個配置した。

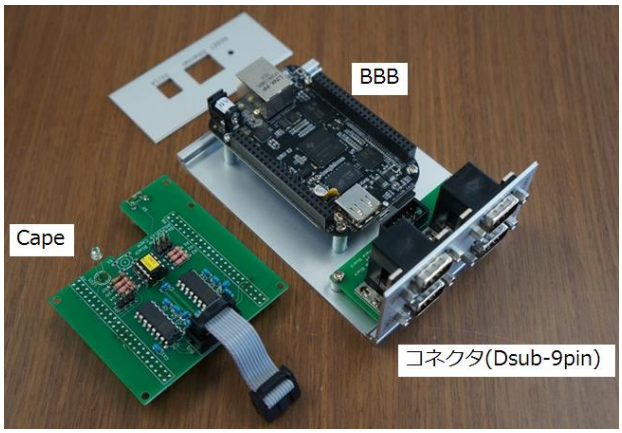

Figure 3: Photograph of 4-port RS232C device server. A “Cape” mezzanine board is installed on top of BBB.

3. ソフトウェア開発

3.1 OS と開発環境のセットアップ

BBB のデフォルト OS は Angstrom Linux である。CentOS や Debian などよく使用される Linux と設定が異なるため、手間取ることもあったがコミュニティの情報などを参照して解決することができた。最新のリビジョン(Rev C)ではデフォルト OS が Debian に変更されており、こちらはほぼ苦労することなく EPICS をインストール可能であった。

通常、OS はオンボード eMMC から立ち上げることができるが、今回作成した Cape で使用する UART のピンが eMMC と重なっているという問題があった。そこで当面は microSD カードを装着してそこからブートする方針とした。各種プログラムは BBB に ssh でログインした後にセルフコンパイルを行う。通常、組み込み機器の CPU は遅く、クロスコンパイルを行うことが一般的であるが BBB ではさほどストレスなくコンパイル・実行が出来る。EPICS Base 全体をコンパイルするような場合にはさすがに時間がかかるが、許容範囲内である。Base のほかにもよく使用する asyn, sequencer, stream device などすべて問題なくコンパイルと実行が出来た。

3.2 EPICS Application 開発

ALOKA 社製の放射線モニタ MAR-782[15]を１つのデバイスサーバあたり最大４台接続する。RS232C 経由であるため Linux では /dev/ttyO1 等のデバイスファイル経由でアクセスする。メッセージベースの非同期制御を簡単に取り扱うことのできる Asyn Driver[16]と Stream Device[17] を使用した。

IOC アプリケーションプログラムは NFS マウントしたサーバ上においている。SD カード上に保存することも可能ではあるが、できる限り個々のデバイスサーバの個性をなくすためにこの方針とした。また、ブート時に自動的に IOC が起動するように設定してある。

4. 結果

図４にデバイスサーバと KEK にてコミッショニングが進行中のコンパクト ERL (cERL) [18]の平面図と放射線モニタの配置図を示す。cERL のシールド壁外形は約 60 m × 20 m 程度である。計測器には ALOKA 社製 MAR-782 を使用している。これは放射線センサと本体部分とを分離することが可能であるため、センサ部をシールド内の特にレベルが高いと予想される場所に設置し、約 40m 程度の専用線で本体と接続している。外部インターフェースは RS232C あるいは RS422 が選択可能であるため、今回は RS232C に設定して、直近の場所にデバイスサーバを設置した。同時に加速器制御専用のネッ

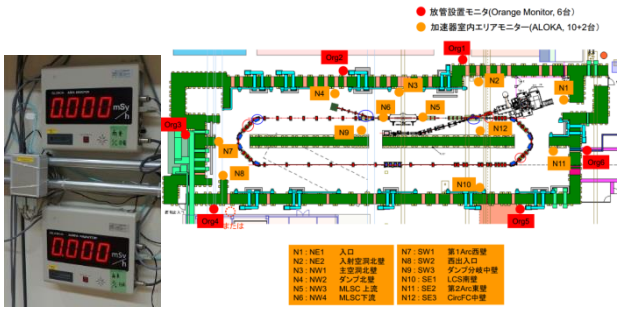

Figure 4: Left figure shows area monitor (ALOKA MAR-782) and a device server installed on the outside wall of accelerator shield. Location of radiation sensors are shown as a yellow circle in the right figure.

トワークに接続して EPICS IOC として通信を行っている。MAR-782 は指定した閾値を超えるとリレー出力を出すポートを備えているため、インターロック目的には接点を監視するシステムを別系統で用意している。加速器制御室は同じ建物の２階に位置している。

測定結果の例を図５に示す。このときは午前中に超伝導空洞のコンディショニングを実施し、パルスビームによるビーム調整に続いて連続ビーム(CW、6.5 μA 程度)によるビーム調整スタディを行ったときの記録である。シールド内各所での放射線レベルを測定することでビームロスの発生源の調査とチューニングを行うことができる。

デバイスサーバの設置後、約半年間以上にわたって連続運用を実施している。ビーム運転その間、一度も再起動を実施する必用もなく順調に連続稼働した。

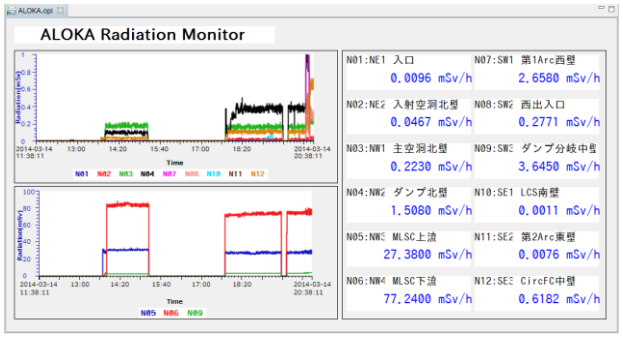

Figure 5: Example plot of radiation level inside the cERL accelerator shield, when CW beam tuning was carried out. CSS program is used to show the data.

4 ポート RS232C 版のほかに、NIM ケースに格納した多機能タイプの開発も実施した。これは主としてデジタル I/O とアナログ入力として使用することを目的としている。ケースは加速器制御でよく利用されている NIM モジュールとしたが、NIM ビン電源が無い環境でも単体で使用可能にするため、背面の切り替えスイッチによって AC アダプタから給電することも可能にしている。BBB の ADC 入力は 0 ～ 1.8 V / 12bit であり、このままでは不便であったため、前段にスケール調整用のバッファ回路を配置して±10V の電圧入力を読み込む仕様とした。デジタルは最大 60bit の 3.3V TTL レベル I/O である。ソフトウェアのデバッグは完了し、問題なく動作することを確認した。これから実際の設置場所に配置して運用する予定である。

Figure 6: BBB installed in NIM chassis. 60 bit Digital I/O, 1 port RS232C, 6 channels ADC are available.

5. まとめと今後の予定

市販の名刺サイズシングルボードコンピュータである BeagleBone Black をベースにして EPICS 内蔵の小型デバイスサーバを開発した。最初の応用例として 4 ポートの RS232C ホストとして稼働する IOC を製作して KEK-cERL において連続運転をおこない、機器制御とモニタに関して十分な性能と安定性を持っていることを確認した。開発したデバイスサーバは拡張性にも優れているため、今後新しいデバイスの制御が必要になったときにも迅速なプロトタイピングを行うことができる。今後の開発ターゲットとしては以下のような候補を検討している

- 1 port モデル製作、4 port タイプの量産、
- USB ホスト（既に実績あり）
- PoE 対応
- 様々な I/O に使用 (Digital/Analog)
- Ethernet 2 port タイプ
- Gigabit Ethernet
- 単純な表示端末や入退室の表示用 PC 置き換え

デバイスサーバが大量に必要な場合は BBB を使用するよりはこれをベースにしてボードを一から製作した方が安価になる可能性もあるが、当面必要な数量を考えると BBB をそのまま利用していく方が有利だと考えている。しかし、最近では本製品の人気のため入手困難になっている状況も現れている。いずれ解消すると思われるが、このような状況に陥ることを避けるため、新たなプラットホームを自作するという選択肢も常に考慮しておく必要があると思われる。

BBB をベースにしたプラットホームは他の制御系、たとえば導入部で例を挙げた TANGO, MADOCA のほか PF ビームラインで使用されている STARS/COACK[19]などへも容易に移植可能である。

このたび我々が開発した拡張ボードの回路図や基板図面、ケース図面などのハードウェア情報やソフトウェア開発の詳細な記録は wiki サイト[20-21]にて公開しておりダウンロードして利用可能である。より多くの研究所や個人によるハードウェア/ソフトウェアの開発および情報共有を進めて行くことで開発コミュニティが広がることを期待している。

参考文献

[1] EPICS, http://www.aps.anl.gov/epics/
[2] BeagleBone Black, http://beagleboard.org/black
[3] Raspberry Pi, http://www.raspberrypi.org/
[4] Arduino, http://www.arduino.cc/
[5] Armadillo, http://armadillo.atmark-techno.com/
[6] Suzaku, http://suzaku.atmark-techno.com/
[7] Galileo, http://www.intel.co.jp/content/www/jp/ja/do-it-yourself/galileo-maker-quark-board.html
[8] T. T. Nakamura, et al., Proc. IPAC10, Kyoto (2010) 2683
[9] S. Kusano, et al., Proc. 8th Annual Meeting of Particle Accelerator Society of Japan (2011) MOPS095
[10] EPICS Users wiki, EPICS on Armadillo development, http://cerldev.kek.jp/trac/EpicsUsersJP/wiki/armadillo/crossenv
[11] BeagleBone Capes http://elinux.org/Beagleboard:BeagleBone_Capes
[12] BeagleBone Black “System Reference Manual” http://elinux.org/Beagleboard:BeagleBoneBlack
[13] プリント基板センターPB、http://pcb-center.com/ P 板.com, http://www.p-ban.com/ 等
[14] タカチ製作所 http://www.takachi-el.co.jp/
[15] ALOKA MAR-782, http://www.hitachi-aloka.co.jp/english/products/data/radiation-011-MAR781782
[16] EPICS Asyn Driver, http://www.aps.anl.gov/epics/modules/soft/asyn/
[17] EPICS Stream Device 2 http://epics.web.psi.ch/software/streamdevice/
[18] ERL 開発室 http://pfwww.kek.jp/ERLoffice/index.html
[19]STARS, http://stars.kek.jp/
[20] EpicsUsers wiki, http://cerldev.kek.jp/trac/EpicsUsersJP/
[21] ALOKA MAR-782 用 IOC 開発, EpicsUsersJP wiki, http://cerldev.kek.jp/trac/EpicsUsersJP/wiki/epics/streamdevice/Aloka_MAR_783