SPring-8 と SACLA における BPM 信号処理システムへの MADOCA-to-EPICS ゲートウェイの適用

APPLICATION OF MADOCA-TO-EPICS GATEWAY FOR BPM SIGNAL PROCESSING SYSTEM AT SPRING-8 AND SACLA

清道明男#, A), 藤田貴弘 A), 前坂比呂和 B), 増田剛正 A)
Akio Kiyomichi #, A), Takahiro Fujita A), Hirokazu Maesaka B), Takemasa Masuda A)
A) Japan Synchrotron Radiation Research Institute (JASRI)
B) RIKEN SPring-8 Center

Abstract

We have developed the MADOCA-to-EPICS gateway for easy and rapid integration of EPICS ready devices into MADOCA, which is a control software framework of SPring-8 and SACLA. MADOCA uses equipment control software called Equipment Manager (EM) in the device control layer. The MADOCA-to-EPICS gateway is implemented as a general-purpose EM functions to handle EPICS IOCs using Channel Access (CA) protocol. Previously, there was a limit to the data collection rate due to the overhead of about 30 ms per access coming from CA reconnection processing. Therefore, we modified the gateway to avoid the CA reconnection and realized high-speed data collection by reducing the overhead to 1 ms or less. We have applied the gateway to the BPM processing system, Libera Brilliance+, installed in the SPring-8 storage ring for the evaluation towards the SPring-8 upgrade project. In addition, it has been applied to the Libera Brilliance Single Pass and Spark installed in the SSBT beam transport line, and the Libera Spark and Cavity BPM installed in SACLA. The gateway brings us the benefits to minimize the installation time and effort even for the different platform (CPU and OS) devices.

1. はじめに

SPring-8 および SACLA では基幹制御フレームワークとして MADOCA (Message And Database Oriented Control Architecture)[1]を採用している。MADOCA はクライアント・サーバモデルに基づいたメッセージ駆動型の分散型制御システムである。遠隔の VME 等のフロントエンド計算機に対してテキストベースのメッセージを送り、VME はその応答を同じくメッセージとして送り返す。一方、加速器施設でよく利用されている分散型の制御システムに EPICS[2]があり、これがあらかじめ組み込まれた機器も市販されている。このような EPICS 準拠デバイスを MADOCA 制御システムの元で簡便かつ迅速に導入できることを目的に、MADOCA-to-EPICS ゲートウェイ[3]を開発した。開発の過程において、複数の信号を繰り返し取得する際に接続のオーバヘッド時間が問題になったため、今回その改良を行った。SPring-8 および SACLA において導入した BPM 信号処理回路 Libera は EPICS が組み込まれており、これの制御に MADOCA-to-EPICS ゲートウェイを適用した。本報告ではゲートウェイの改良、現在の運用状況、および今後の改善点について報告する。

2. MADOCA-to-EPICS ゲートウェイの改良

2.1 MADOCA-to-EPICS ゲートウェイの基本機能

MADOCA は制御命令を行う GUI 部のレイヤーとハードウェアを制御する機器制御レイヤーに分かれており、Message Server (MS) を介してメッセージを送る。機器制御レイヤーでは Equipment Manager（EM）と呼ばれるソフトウェアが動作しており上位から抽象化された命令を受け取り、解釈して、物理デバイスとの対応付けを行い、デバイスドライバを呼び出して命令を末端の機器に伝える。そして機器からの応答を受理しこれを論理的な値へと抽象化して上位に返す。これら一連の制御を行うにあたり、EM はメッセージを解釈する関数・デバイスの制御を行う関数・メッセージとして抽象化する関数の3つの処理関数を持つ。そして受け取ったメッセージ（SVOC コマンド）と3つの処理関数、制御対象デバイスとの対応付けを EM 構成ファイル（config.tbl）で行う。

EPICS は各デバイス上で Input/Output Controller (IOC) が動作し、通信は Channel Access (CA)プロトコルで統一的に扱われる。IOC 上にはレコードのデータベースがあり、個々のレコードがデバイスやデバイスの制御を表している。レコードに対して値をセットする caput、レコード状態を返す caget といった CA プロトコルに基づいたコマンドで基本的には全ての制御を実現できる。

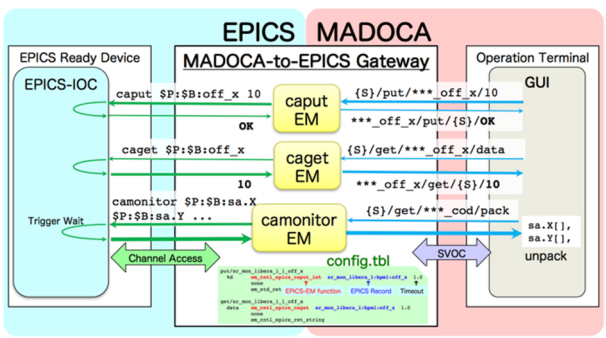

Figure 1: Scheme of MADOCA-to-EPICS gateway.

kiyomichi@spring8.or.jp

MADOCA-to-EPICS ゲートウェイは、EPICS における caget と caput および状態に変化があったときにレコード状態を返す camonitor の3つの CA コマンドに相当する処理とメッセージを抽象化する return 処理を EM 関数として実装している。EM 構成ファイル内にて EPICS レコード名を EM 関数の引数として指定するだけで、新たな開発をすることなく制御できる。基本構成を Figure 1、関数リストを Table 1 に示す。SPring-8 の MADOCA II [4]環境下では配列データをメッセージングで取り扱うことができるので msgpack ライブラリを用いて配列データをパックしてメッセージングで取得する。これらの msgpack ライブラリを用いる関数は“_msgpack_” を関数名に付けて MADOCA II 専用関数として管理している。SACLA の MADOCA 環境用にはファイルに保存する関数(em_cntl_epics_caget_waveform.c)を用意した。

Table 1: EM Functions of MADOCA-to-EPICS Gateway.

Type	EM functions
caput	em_cntl_epics_caput_char.c
	em_cntl_epics_caput_double.c
	em_cntl_epics_caput_int.c
	em_cntl_epics_caput_arraydouble.c
	em_cntl_epics_caput_arrayint.c
	em_cntl_epics_caput_msgpack_arraydouble.c
	em_cntl_epics_caput_msgpack_arrayint32.c
caget	em_cntl_epics_caget.c
	em_cntl_epics_caget_waveform.c
	em_cntl_epics_caget_msgpack_arraydouble.c
	em_cntl_epics_caget_msgpack_arrayint32.c
camonitor	em_cntl_epics_camonitor_msgpack_arrayint32.c
	em_cntl_epics_camonitor_msgpack_arraydouble.c
	em_cntl_epics_camonitor_msgpack_wfmarrayint32.c
	em_cntl_epics_camonitor_msgpack_wfmarraydouble.c
return	em_cntl_epics_ret_string.c
	em_cntl_epics_ret_clock.c

2.2 CA 接続オーバヘッドの解消

MADOCA-to-EPICS ゲートウェイの EM 関数は1回の実行で数十ミリ程度かかり、caget EM 関数を用いて多数の信号を繰り返し収集する際の収集レートに限界が生じていた。例えば100点ほどの信号を1秒周期で取得する需要があるが、これを満たすのが困難であった。EM 関数は caget コマンドをベースとしているため、毎回 CA 接続・データ取得・CA 切断を実行する仕様である。ボトルネックが何処にあるか調査したところ、CA の再接続処理において1アクセスあたり30ms 程度のオーバヘッドが生じていることが分かった。そこで、CA 接続の切断を行わずにレコードへの入出力を行うように改良することでデータ収集の高速化を図った。

CA 接続の切断を行わないよう管理するために CA コネクション情報を保存する。CA コネクションを作成する度に与えられる Channel ID を含む PV (Process Variables) 構造体を Linked list 形式で保存する。EM 関数の引数としてレコードが呼ばれた際にはリスト内のレコード名と比較してリスト内に存在しない場合には CA 接続処理の後にその情報を追加する。リストは static 変数として宣言しており、関数を抜けても情報は保存される。

これによって EM 関数で初めてレコードが呼ばれるときはこれまで通り数十ミリ程度かかるが、2回目以降は1ミリ秒以下に削減することができた。この改良により、数百点の信号でも 1 秒周期で十分間に合うデータ収集が可能となった。

3. BPM 信号処理システムへの適用

SPring-8 および SACLA では、Libera Brilliance+, Libera Brilliance SinglePass, Libera Spark, Libera Cavity といった幾種類かの BPM 信号処理システムを導入した。これらのデバイスは同じメーカーの製品ではあるが、製造年代や用途の違いより CPU や OS が異なり、それぞれ EPICS IOC および開発キットが提供されている。IOC を介さずに直接プラットフォーム上で動作する EM を開発する場合、それぞれに MADOCA 開発環境を準備する必要があり多くの時間と労力がかかるが、EPICS 準拠デバイスであることを活かして MADOCA-to-EPICS ゲートウェイを適用することで、導入までの時間と労力を最小限に抑えることができた。

3.1 SPring-8 蓄積リング COD 測定評価用 Libera

SPring-8-II に向けた BPM 信号処理システムの候補のひとつに Libera Brilliance+[5] がある。これのビーム信号を用いた性能評価のために SPring-8 蓄積リングへ設置し、MADOCA-to-EPICS ゲートウェイを適用して MADOCA 制御システムへ組み込んだ。

Libera Brilliance+ は Intel 系の CPU に Ubuntu Linux を搭載している。ソフトウェアは幾つかの階層に分かれており、ユーザは MCI (Measurement and Control Interface) を通して制御することができる。また、MCI を介した EPICS-IOC が提供されている。MADOCA-to-EPICS ゲートウェイを適用して、EPICS-IOC 経由での制御及びデータ収集系を Figure 2 のように構築した。

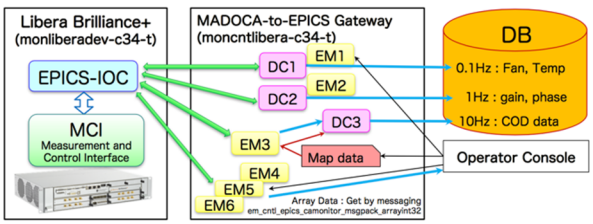

Figure 2: Scheme of control and data acquisition for Libera at SPring-8 SR.

MADOCA-to-EPICS ゲートウェイは Solaris10 の仮想ホストで運用する。仮想ホスト上では、以下に示す EM が6つと MADOCA II データ収集プロセスである Data Collector (DC)が3つ動作している。

- EM1, DC1: 0.1 Hz status, clock
- EM2, DC2: 1 Hz maxadc, gain, phase
- EM3, DC3: 10 Hz, Slow Acquisition (SA),COD
- EM4: Single Pass (SP) data by GUI

- EM5: Fast Acquisition (FA): 10 kHz, 128 kpt (10 sec)
- EM6: ADC, TBT window, PM data

定周期のデータ収集はDCを用いてデータベースに書き込む。周期の違いにより3つのDCに分けている。DC1およびDC2はファンや電源のステータスといったデバイス監視データ、DC3はBPMのSlow Acquisition (SA)データを収集する。ただし、ビーム位置データはLibera内部の各電極の電圧データをEM3経由で取得し、DC3上でBPMの非線形マップ関数を用いて演算して電圧データと合わせてデータベースへ書き出しを行う。Fast Acquisition (FA)およびSingle Pass(SP)データは一回の測定でのデータ量が大きいためGUI経由でファイルに保存する。FAデータのレートは10 kHzであるが、camonitor EMで約10秒分128kポイントのデータを溜めた1つの配列データにして、GUIで取得しファイルに保存する。

2016 年秋の運転からこの構成で運用し、安定的なデータ収集を行えている。

3.2 SPring-8 SSBT Stripline BPM

SPring-8 の SSBT(Synchrotron to Storage ring Beam Transport) ビーム輸送ラインの BPM では、Libera Brilliance+ の旧タイプである Libera Brilliance Single Pass [6] を 3 台使用している。Libera Brilliance Single Pass は ARMv5 系の CPU に Ubuntu Linux を搭載しており、ユーザは CSPI (Libera Control System Programing Interface : MCI の前バージョン) を通して制御することができる。また、CSPI を介した EPICS-IOC が提供されている。導入当初は ARM 用クロスコンパイラと CSPI 開発キットを用いて Libera 上で直接動作する EM を開発して運用していた。

2017 年より、ここに Libera Spark[7] を 2 台追加して計5台の BPM 信号処理システムを運用することとなった。Libera Spark の CPU は Zynq(ARMv7)で OS は i-tech Linux を搭載しており、ユーザ開発用に MCI および EPICS-IOC が提供されている。Libera 上の EM を開発する場合は、Single Pass とはバイナリの互換性は無く CSPI 開発キットとは別にクロスコンパイラと MCI 開発キットも用意する必要がある。さらに同じ MCI を用いる Brilliance+とも CPU が違うために互換性は無い。そのため、この方式の開発体制を維持していくのはコストがかかると判断した。そこで、Libera 上で直接動作する EM を開発するのでは無く、EPICS-IOC を利用して MADOCA-to-EPICS ゲートウェイを適用する方式に移行することにした。SSBT stripline 用 Libera の制御及びデータ収集の構成を Figure 3 に示す。

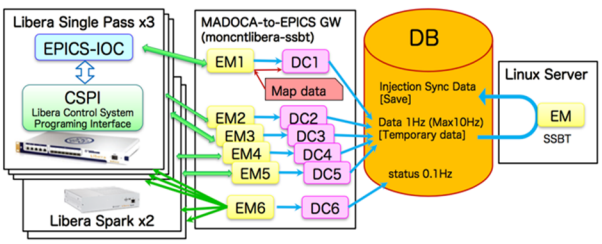

Figure 3: Scheme of control and data acquisition for Libera at SPring-8 SSBT stripline.

5台の Libera に対して1対1となるように EM および DC を用意し Solaris10の仮想ホストで運用する。EM 経由で SVOC コマンドの指定で AD データを1024点取得し、DC プロセスで演算とデータベースへ書き込みを行う。データ収集は1 Hz であるが、将来 SACLA から蓄積リングへのビーム入射の際には最大10 Hz の運用も予定している。また、各 Libera の死活監視用のプロセスを DC6として用意し、10秒周期で clock 情報を収集する。

データベースに書き込まれたデータは一定の日数経過後に間引かれる。蓄積リングへの Top-up 入射は、30~60 秒に1回の頻度で行われるが、この入射時のデータのみを別の信号名で永久保存する。入射の判別は各 BPM の電圧値で判別し入射フラグを立てることで行っている。データベースに書かれた後は、定期的に DB からデータを読み入射のあったデータを時間情報込みで DB に書き戻すよう、Linux サーバ上で動作する EM を別途開発して運用している。

2017 年4月の運転からこの方式に変更して運用開始し、安定的なデータ収集を行えている。また、加速器スタディにて 10Hz のデータ収集試験も行った。

3.3 SACLA BPM

SACLA の BPM にて Libera Spark および Libera Cavity BPM を導入するにあたり MADOCA-to-EPICS ゲートウェイを適用した。SACLA の制御システムは MADOCA であるので msgpack を用いた MADOCA II 専用関数は使用しない。そのため camonitor EM 関数は使用せず、caget/caput EM 関数のみで整備した。SACLA の Libera Spark / Libera Cavity BPM の制御及びデータ収集の構成を Figure 4 に示す。

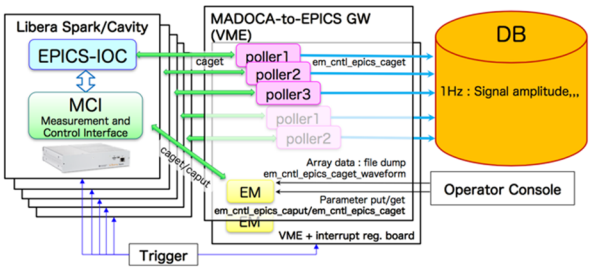

Figure 4: Scheme of control and data acquisition for Libera at SACLA.

Libera で読み出す BPM は SACLA の加速器終端からアンジュレータ入口までのビーム輸送部の計5箇所に設置している。MADOCA-to-EPICS ゲートウェイは Solaris10 を入れた VME 2台で運用する。1台は SACLA の同期データ収集系に組み込み、ビームに同期したトリガ信号を VME と Libera へ入力してデータ収集を行う。データ収集プロセス poller は Libera に1対1となるように分けている。Libera の信号は camonitor で読み出すことにより EPICS DB 上のデータが更新されるため、caget EM のみでは最新値を取得出来ない。そのため Libera 上で camonitor を実行することにより、caget で最新値を取得出来るようにした。

2017 年1月の運転から導入し、4月より同期データ収集を開始して、安定的なデータ収集を行えている。

4. 課題と機能拡張

MADOCA のデータ収集は上位から取りに行く pull 型であり、現行の MADOCA-to-EPICS ゲートウェイ EM は 1回1回 IOC に問い合わせることによるオーバヘッドが生じる。CA 接続情報を管理する改良によりデータ収集は 10 Hz 程度の収集レートまでは安定に動作したが、これ以上の高速化は取りこぼしが発生し安定運用は困難である。EPICS ではデータを IOC 側から配信するデータ収集も可能であるので、この機能を利用して10 Hz 以上データ収集を実現する高速化の機能拡張を検討した。

4.1 高速化機能拡張の方針

EPICS ではデータを IOC 側から配信することができる。IOC へデータ配信を促すコールバック関数を EM に実装することにより、常に EM 側に最新データを保持する仕組みを設計した。EM 上に EPICS のプロセス変数（PV）用バッファ領域を持たせて PV の更新情報を常にコピーし、MADOCA 側からは直接 IOC 上の PV 情報は扱わずコピーした PV 情報を参照する。このような構成にすることで EPICS 側は MADOCA 側の処理の影響を受けなくなるためシステムが単純化される。

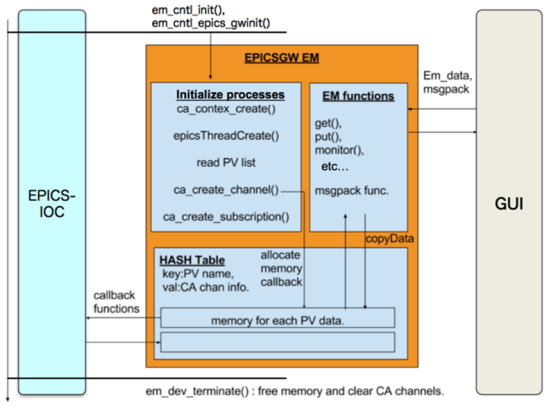

Figure 5: Scheme of extended MADOCA-to-EPICS gateway.

高速化機能拡張した MADOCA-to-EPICS ゲートウェイ EM の構成を Figure 5 に示す。EM 起動時の初期化プロセスにて CA 接続のための CA チャンネルの作成と PV の subscribe 登録を行うが、それぞれ対応するコールバック関数を用意している。この時、登録する PV 名とデータ型はあらかじめ指定する。CA チャンネル作成時に PV 情報を保存するためのメモリ領域を確保する。subscribe 登録時には PV 値が変更されたときにメモリ領域への書き込みを実行するコールバック関数が登録される。PV 情報はハッシュテーブルを用いて管理する。データ保存用のメモリはリングバッファとなっており、指定したデータ数を超えた場合に古いデータは新しいデータで上書きされる。

4.2 高速化機能拡張のプロトタイプ作成と評価

GUI 等の MADOCA クライアントからデータのやり取りを行うため、リングバッファ上のデータを読み出す EM 関数プロトタイプを作成した。作成したプロトタイプは camonitor 用関数で、メモリ上のデータを全て読み出し最後にリングバッファのデータ書き込み位置をリセットする。そのため毎回実行する度に未取得のデータのみを読み出せる。リングバッファに余裕のある範囲でデータの更新と読み出しが非同期で行うことが可能で、例えば 60 Hz のトリガ周期のデータを 60 回分バッファして 1 Hz で読み出す、といった使用法を想定している。

プロトタイプの評価試験を行うために、周期的にインクリメントされるレコードを持つ EPICS シミュレータを用いて読み出し試験を行った。レコードは最大 50 Hz で更新できる要素数 10000 の配列型データを使用する。クライアントで取得したデータが連続値であるならばデータ抜けは無い。レコードの実行周期を 1 Hz, 10 Hz, 50 Hz に変更してクライアントからデータ収集試験を行った。クライアントの収集周期をレコードの実行周期と合わせた同期データ収集では、50 Hz でクライアントプログラムの処理時間の遅延によりデータ取得ができず次に2つの配列データを取得するといったバラツキが生じたが、バッファの効果でデータの抜けは無かった。1回1回 IOC に問い合わせる現行版ではデータ抜けが生じる。レコードの実行周期 50 Hz に対して、クライアントの収集周期1 Hz および 10 Hz での非同期データ収集試験では安定的にデータを取得出来た。

高速データ収集についてはクライアントプログラムの処理内容に応じて、非同期データ収集で安定的なデータ取得が可能となる。

5. まとめ

MADOCA 制御システムにおいて EPICS 準拠デバイスを簡便かつ迅速に導入できるようにするために、MADOCA-to-EPICS ゲートウェイを開発した。CA 接続のオーバヘッドを解消しデータ収集の高速化を実現した。

SPring-8-II に向けた BPM 信号処理システムの評価試験や、現行の SPring-8 SSBT および SACLA の BPM 読み出しに導入した Libera 11台に適用し、制御やデータ収集を行うことができた。3種類のプラットフォームに対して、時間と労力を最小限に抑えた導入を実現した・

より高速なデータ収集に対応するために、データ更新時に IOC から送られるデータを受け取り EM 上にバッファする機能拡張のプロトタイプを作成しその有効性を確認した。今後の用途に合わせて本実装していきたい。

参考文献

[1] R. Tanaka *et al.*, “The first operation of control system at the SPring-8 storage ring”, Proceedings of ICALEPCS97, Beijing, China, 1997, p.1.
[2] http://www.aps.anl.gov/epics/
[3] A. Kiyomichi *et al.*, “Development of the MADOCA-to-EPICS Gateway”, Proceedings of the 13th Annual Meeting of Particle Accelerator Society of Japan, Chiba, Japan, Aug. 2016, p.656.
[4] T. Matsumoto *et al.*, “Development of New Control Framework MADOCA II at SPring-8”, Proceedings of the 10th Annual Meeting of Particle Accelerator Society of Japan, Nagoya, Japan, Aug. 2013, p.14.
[5] http://www.i-tech.si/accelerators-instrumentation/libera-brilliance-plus/
[6] http://cerncourier.com/cws/product/P000015259
[7] http://www.i-tech.si/accelerators-instrumentation/spark-el-hl/