KEK 電子入射器の制御システム

古川 和朗*、上窪田 紀彦†
高エネルギー加速器研究機構 (KEK)

概要

大型加速器は寿命が長い上に、運転開始後も極限まで性能を高める必要があるために、制御系にもさまざまな機能拡張が可能となるような柔軟性が求められる。また、加速器モデルと現実の加速器を対比しながら運転できるような仕組みも必要となる。もちろん、加速器を長期間安定に運転するために、その信頼性・可用性も重要である。KEK の電子陽電子入射器では、600m にわたって分散配置された装置約千台（信号数約一万点）を総合的に監視・制御することで高品質なビームを生成し、KEKB、PF、PF-AR それぞれに適したビームを入射している。その制御系及びタイミング系の設計方針や実装、そして将来の方向性について解説する。

1 はじめに

制御システムは、加速器を運転する際にはある意味主役を演じることになる。つまり、加速器内の各機器の最大性能を引き出しつつ、オペレータに的確な情報を提供し、最終的に期待した性質を持ったビームを生成する仕事を受け持つ。

KEK の電子陽電子入射器は、KEKB リングへ 8 GeV の電子と 3.5 GeV の陽電子を、PF リング及び PF-AR リングへ 2.5 GeV または 3 GeV の電子をそれぞれ入射している [1]。リングにおける実験効率を向上させるために、入射ビームにも高い安定度が要求され、長期間にわたる高安定なビーム加速を実現するためののさまざまな機構が導入されている [2, 3]。

この解説では、第 2 節で一般的な加速器の制御について、第 3 節で KEK 電子入射器の制御の概要について、それぞれ説明する。その後、KEK 電子入射器の制御システムの詳細を解説するが、第 4 節では計算機、ネットワーク、フィールドコントローラを中心としたハードウェアについて、第 5 節ではこの制御システムのために開発された多階層制御ソフトウェアについて、第 6 節ではビーム運転で使用される応用プログラムについて、それぞれ扱う。また、第 7 節では制御グループが担当しているタイミングシステムについても解説する。最後に、第 8 節で今後の展開について議論していくことにする。

* <kazuro.furukawa@kek.jp>
† <norihiko.kamikubota@kek.jp>

2 加速器の制御

2.1 加速器の制御の歩み

一般に加速器の制御は加速器の特性、つまり加速器内の機器の数やパルス運転か連続運転か、などによって多少構成が異なるが、大きくは変わらない。しかし、その構成技術が、一般の加速器構成機器に比べて進歩の激しい計算機やネットワーク技術などに依存している部分が多く、また、それに伴って加速器制御に対する考え方も変わってきているため、過去に大きな変化を遂げてきた。全般的に言って、その重要度は増してきていると思われる。

2.1.1 1980 年代

大型加速器では、CERN の SPS がミニコンピュータのネットワークと NODAL という加速器用のインタプリタ言語によって運転を行い、ひとつの時代を開いたと思われる。日本では、KEK の電子入射器の初期の制御システムが機器を制御する数百台のマイクロコンピュータを独自の階層的なネットワークで接続し、計算機遠隔制御を実現させた [4]。その後 Fermilab の Tevatron が複合加速器を ACNET と呼ぶひとつのソフトウェアインターフェース (API) で制御を可能にし、また SLAC が SLC のための高速制御を可能にした [5]。日本でも TRISTAN が NODAL とデータモジュールと呼ぶソフトウェアを組み合わせ、機器指向の制御を実現した。

それまで、遠隔多重制御を行うために導入されてきた計算機が、徐々に本質的に必要になってきた年代である。例えば、SLAC の SLC は制御システムがなければ運転できなかったであろう。このように、制御システムの重要度が増してきたこともあり、1985 年からは Internatinal Conference on Accelerator and Large Experimental

Physics Control Systems (ICALEPCS) という国際会議が 2 年毎に開催されている[1]。

2.1.2 1990 年代

その後、一般の計算機やネットワークの規格の統一が進み、それらを活用した加速器制御も変化がはじまった。一時期は Unix ワークステーションなどのウィンドウシステム、TCP/IP と FDDI、イーサネットなどの標準ネットワーク、そして VME などのフロントエンド計算機を組み合わせることを “標準モデル” などと呼ぶようになった [6]。このころ、KEK の電子入射器の初期の制御システムの更新も行われ [7, 8, 9]、あとで述べるように、この標準モデルに近い形態になった。

それまでは、各加速器で独自の計算機やハードウェアを開発することが多かったが、これらの標準的な制御システムの形態によって、制御システムの課題は、いかに大きな制御システムを管理運営していくか、また、いかに各加速器間のソフトウェアやハードウェアの開発結果を共有していくか、ということに移っていくことになる。その中では、いかに一時的な流行 (Fad) に惑わされずに、しかし十分な機能を持った、寿命の長い制御システムを作り上げるか、ということや、資源を共有するためには、まず加速器自体の定義から始める必要がある、というような議論も行われた。

その後、SSC[2] が LANL と ANL/APS で開発された EPICS[10] を採用すると宣言したこともあって、EPICS を基礎とした資源の共有形態が模索されることになる。EPICS という共有可能な具体的な実装例を得たことにより、ソフトウェア資源の国際的な協力開発にはずみがつくことになった。この成果を、日本では KEKB リングの制御システムが積極的に利用している [11]。

ところで、制御システムにおいてもオブジェクト指向の設計やプログラミングなどソフトウェア技術の発展の成果の取り込みについては、有効性は認識されていたものの、独自の計算機環境や、実時間処理などのために制約を受け、大規模に行われることが少なかった。オブジェクト指向設計についてはなんらかの形で取り入れているところが多かったが、システム全体でオブジェクト指向プログラミングを活用することが難しかった。この点については、まず、上位層、つまりアプリケーションソフトウェアに近い部分から CORBA や、Jefferson-Lab の提唱する cdev[12] などを基礎としての利用が浸透しつつある。

また、安価になったパーソナルコンピュータ (PC) の技術を最大限利用しようとする動きも活発で、DESY の HERA[13] では種々のサブシステムを多数の Windows 計算機で統合している。Linux の信頼性が高まるにつれ、利用するシステムも増えてきている。

加速器のモデリングを現実の加速器の運転で直接対比させながら利用する試みについてもさまざまな形で行われている。しかし、制御グループと、運転またはビーム物理グループの間の垣根が高い場合もあり、実証試験と日々の運転は別、という例も少なくない。そのなかでも有効に使われたと思われるのは LEP の運転コンソールシステムである。また、KEKB の SADscript/Tk は大きな成功を収めており、入射器のためのものも含めてさまざまな運転用ソフトウェアが利用されている [14]。

KEK の電子入射器でもあとで述べるように、上に挙げた技術を適材適所に利用してシステムを構築し、また、改善し続けている。

2.2 制御システムの目的と構成

当初、遠隔制御を行うだけでも困難な時期もあったが、現在では基礎となる技術の進歩によってシステムの構築は容易になってきたかのように見える。しかし、非常に性格の異なるさまざまな加速器機器を統合して、期待したビームの生成に結び付けることは、まだ確立されたとは言えない。加速器や制御システムが大きくなるにつれ、加速器が目標とした成果を得るために、制御システムの理念や方法論を正しく持つことはますます重要になってきている。

2.2.1 制御システムに対する要求仕様

ひとことで言えば、加速器の制御の目標は、信頼性が高く、かつ柔軟な制御処理の機構を道具として加速器に提供するところにある。通常、そのために複数のサブシステムを統合して、全体の制御システムを構成することになる。ひとつのサブシステムは、複合加速器の場合、そのなかのひとつの加速器の制御を担当したり、真空システムのような機器グループを担当したりする。全体のシステムは、通常はサブシステムの詳細を隠して加速器全体を簡潔に表現するが、求められれば、個々の機器の詳細まで的確に提供できる必要がある。

このような制御システムの基本的な機能としては次のようなものがある。

- 加速器のオペレータや機器の担当者に対して、適当なグラフィカルインターフェースをもって、簡潔な情報と詳細な情報の双方をいつでも提供すること。
- 加速器機器に対して、適当な実時間処理を実行し、ビームフィードバックなどの加速器の性能を高め

[1] 当初はワークショップと呼んでいた。

[2] 残念ながら、その後 ICALEPCS93 の開催期間中にプロジェクトの中止が知らされた。

るための操作を可能にすること。

- 現実の加速器と対比させながら、加速器のモデリングやシミュレーションを行えるような環境を提供すること。
- 新しい問題が見つかったときに、解決のための新しい制御処理を効果的に追加するための環境が用意されていること。
- 個々の機器の間や時間との間でのちに相関解析が行うことができるように、情報を蓄積すること。
- 予期しない事態が起ったときに、適当な措置を講じたり、オペレータに報告したりする、いわゆるアラームの機能を持つこと。
- 関連する他の施設や研究者、技術者、利用者などとの間に適当な情報交換を行い、また技術や成果の共有の手段を提供すること。

概略の構成は図 1 のようなものとなる。これらを実現させるための実装方法は、さまざまなソフトウェア資源が共有されるようになった現在でも加速器によって異なっている。

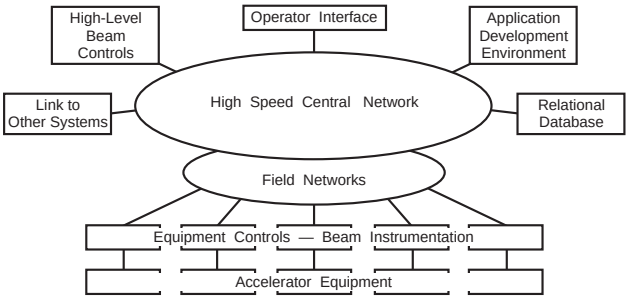

図 1: 典型的な制御システムの構成

2.2.2 制御システムの内部機能

上に挙げたような制御システムの基本機能を実現させるためには、アプリケーションソフトウェアに対して、次のような制御システムの内部機能（プリミティブ API）を持っている必要があると思われる。

- 要求されたときに、同期または非同期で情報を提供できる
- 定期的に情報を提供できる
- 状態が変わったときに報告できる
- 高速応答するために適当な方法で情報を一時保持できる
- 多数の情報を同時に処理できる
- 情報を長期保存できる

現存する制御システムはこれらのうち、すべてまたは一部の機能を実現している。機能が充分でない場合には、アプリケーションソフトウェアで機能を補わなければならない。

さらに、これらを実装する環境として、

- 基本言語としてのコンパイラ言語
- プロトタイプ開発に用いるスクリプト言語
- データベース環境
- グラフィカルユーザインターフェース (GUI)
- 全体の構成を 2 階層のみにするのか、多階層にするのか
- IP などの一対一の通信を基本とするのか、リフレクティブメモリなどにより情報を共有させるのか
- フィールドバスに何を使うのか

などを選択する必要がある。

2.2.3 線形加速器の制御システム

線形加速器の制御を考える際には、リング型加速器のようなビームの自己安定化の仕組みがないために、制御システムは必ず必要になる、ということを考慮しなくてはならない。また通常、線形加速器はパルス運転されるので、パルス毎の処理の可能性も考慮しておかなくてはならない。直線的に長いということが、装置に対して制限を与える場合もある。リング型加速器ではひとつの電源が複数の電磁石に接続されるなど、装置と名前の対応が複雑になりがちだが、線形加速器ではそれぞれの場所でエネルギーが異なることもあり、名前付けは比較的単純となる。しかし、そのエネルギーの測定が困難であることから、正確なモデリングも困難になる場合が多い。

3 KEK 電子入射器の制御の概要

3.1 入射器の制御の設計

KEK 電子入射器の制御システムは、古いシステムが 1982 年から運転に使用されたが [4]、1990 年頃から更新後のシステムの設計が始まり、1993 年に更新が実施されて [8, 9]、その後毎年改善されている。Unix 計算機をサーバとして現場の装置コントローラを統括するシステムになっており、それらの間の通信に TCP や UDP を元にした独自開発の RPC (Remote Procedure Call) を使用している。

設計においては、前節に述べたような条件を満たすように、いくつかの方針をおいた。つまり、

- できるだけ国際標準や業界標準の規格を採用する。

- 現場の装置コントローラは TCP/IP ネットワークに接続できることとする。

前者は、柔軟性や拡張性を維持するために重要と考えられ、また将来の更新を容易にすると期待される。後者は、以前は重視されていた統一された装置コントローラをやめることを意味するが、あとで述べるようなソフトウェア構成によって、装置に応じて、また最新の技術を利用しながらコントローラを選択できることになる。

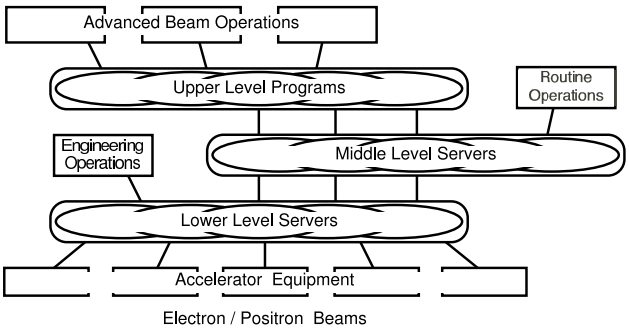

図 2: システム内の処理の論理的な構成

全体の模式図を図 2 と図 3 に示す。まず、この節で概要を示し、次節以降で詳細を解説する、制御システムと関連の深いタイミングシステムは第 7 節で解説するが、合わせて論じられることの多い人および加速器の保護安全システムについてはこの稿では扱わない。

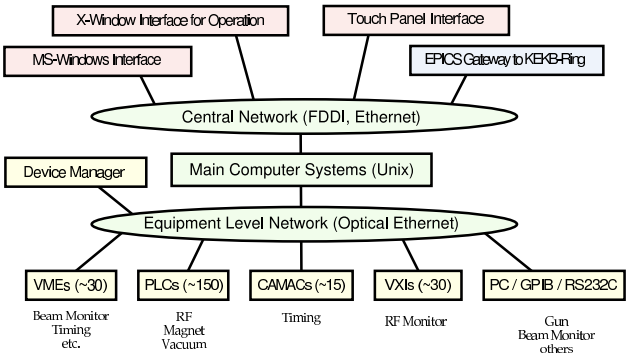

図 3: システムのハードウェアの物理的な構成

3.2 入射器の制御の全体構成

システムは大きくわけて図 2 のように 3 つのソフトウェアの階層から構成されている。下位層は、例えば 16bit の接点信号のような、異なる種類の制御ハードウェアに応じて用意されており、障害復旧などを含めたハードウェアの詳細を表現するが、通常は上位層からは隠蔽されている。

中間層には、電磁石などの抽象化された加速器装置を表現するサーバソフトウェアがおかれている。それぞれは実際のハードウェアを扱うためにいくつかの下位層のサーバを利用する。以下ではこの層を上位層と呼ぶことがある。

最上位の運転プログラムは実際の入射器のビーム運転を行う。先に述べた RPC によって制御ネットワーク上のどの計算機からでも、制御許可の登録があれば、運転操作を行うことができる。

さらに、他の環境と情報を交換するために、中間層のサーバの上にいくつかのゲートウェイソフトウェアが構築されている。例えば、KEKB リングで使われている EPICS 環境のための Channel Access サーバ [15] や、Web ブラウザ経由で蓄積情報を取り出すための CORBA サーバ [16] などが実際に運転に利用されている。

3.3 装置コントローラ

現場での装置の制御には、既に 20 年を経過した 8-bit マイクロコンピュータを内蔵したコントローラや、毎年のように現れる新しい技術を利用したコントローラなど多数を利用している。1993 年の制御システムの更新時には主なコントローラはそのまま使い続けたが、KEKB に向けた増強時には、多数のコントローラが更新された。それらは、個々の装置の要求仕様に合わせ、VME、VXI、CAMAC、PLC (Plogrammable Logic Controller)、PC などの技術で構成されている。速度的な要求が厳しくないものについては、TCP/IP 通信が可能な、PLC が多数導入されいくつかの意味で省力化に役立っている [17, 18]。

3.4 中央制御計算機とネットワーク

中央制御計算機として Unix クラスタサーバといくつかの Unix 計算機が主な中間層サーバと上位の運転ソフトウェアの処理を行っている。クラスタサーバは耐障害性、可用性の高いシステムの運用を可能にしているが、さらに制御ソフトウェアは複数の計算機間で冗長性を持って運用されている。

制御用の計算機ネットワークは、FDDI バックボーンと多数の Ethernet セグメントから構成されており、それらは星型の多階層トポロジで接続されている。現場のコントローラは、パルス運転をするクライストロンモジュレータのノイズを避けるために、すべて光 Ethernet (10BaseFL/100BaseFx) で接続されており、また、中央に近い階層の接続は冗長接続をしてある。

4 ハードウェア構成

4.1 歴史的経緯

1982 年から運転に使用された初代制御系は、ミニコンピュータ（三菱 MELCOM 70/30）8 台と約 300 の microprocessor ベースのローカルコントローラからなる分散処理系であり、ミニコンピュータ間は専用の光ファイバーネットワーク (Loop-1) で相互接続されていた [4, 20]。しかし、80 年代末にはミニコンピュータの保守や計算能力不足などの問題が顕在化し、1990 年頃から制御系を更新するべく調査・研究を始めた [7, 8]。1993 年 9 月には、計算機やネットワークなど基幹部を新しくした 2 代目制御系での運転を開始したが [21, 9, 22]、人員や予算の不足からローカルコントローラはそのまま継続して利用することとした。その後、基幹計算機や制御ソフトの拡張・改修を重ねつつ [23, 24]、ローカルコントローラも順次更新し [25, 18]、現在に至っている。

以上の経緯を表 1 に示す。

	1982 年-	1993 年-	現在 (2002 年)
主計算機	MELCOM 70/30	work-station	同左
(OS)	RealtimeOS	Unix	
network	専用光ファイバ (5Mbps)	Ethernet (10Mbps)	FDDI/switch ＋ Ethernet (10-100M)
(response)	100ms	1-10ms	0.1-数 ms
front-end and local-controller	MELCOM 70/30 ＋ CAMAC m.processor	VME-bus computer (680x0) m.processor	VME/VXI, CAMAC, PC(linux,Win) PLC

表 1: 入射器制御システムの歴史的変遷

4.2 設計方針と全体構成

1993 年の制御系の更新に当たっては、以下の設計方針に従った。

- ハード・ソフトとも業界標準を採用
  加速器およびその制御系は、通常 10 年以上継続して使用される。このため特定の計算機システムに依存する設計では、その計算機の親会社の方針の変化の影響を受け、好ましくない。また、個々の計算機寿命は 5–10 年と加速器寿命より短いため計算機を更新する日は必ず来る。業界標準の採用で、将来の計算機更新が楽になると期待できる。
- 十分高速な通信ネットワークを持つこと
  電子入射器は物理的に広い範囲 (500m) に機器が分散しているので、この距離スケールで十分高速な通信ネットワークが必要である。
- 拡張・変更に即応できること
  電子入射器は、運転用と同時に研究用の加速器でもあり、日々どこかで改良・テストが行われている。したがってその制御系も、機器の拡張・変更に対して即応できる flexibility があることが望まれる。

これらの点を考慮して設計・構築した制御系は、

- Main Computer の Unix 計算機
- front-end の VME-bus 計算機
- operator's interface の PC と touch panel

などの構成要素を、

- 標準 network である Ethetnet (TCP/IP protocol)

で相互接続した形態になった。その後の拡張・整備を経た結果、現在では図 3 のような構成に至ったわけである。

4.3 構成要素

図 3 で示される個々の構成要素を追ってみよう。

4.3.1 Main Computer Systems

1993 年の 2 代目制御系の運用は、Unix 計算機 1 台のみ（名前 peach、運転・開発兼用、maple 導入後は運転専用）で始まった。その後は多数の Unix 計算機を導入してきたが、現在は運転用 3 台と開発用 2 台の陣容になっている（表 2）。運転機が複数あるのは、負荷分散と redundancy 確保のためである。

名前	用途	機種	導入時期	状態
peach	運転	DECstn.(Ultrix)	Nov.1990	引退
lime	X表示	DECstn.(Ultrix)	Oct.1991	引退
maple	開発	DECstn.(Ultrix)	Oct.1992	引退
grape	運転	DECalpha(True64)	Mar.1994	引退
almond	開発	DECalpha(True64)	Mar.1996	
plum	運転	DECalpha(True64)	Mar.1996	
lychee	運転	Compaq(True64)	Aug.1999	
poplar	運転	Compaq(True64)	Aug.2001	
orange	開発	Compaq(True64)	Aug.2001	

表 2: 基幹 Unix 計算機の導入

基幹計算機システム用の運転プログラムやデータを安全に格納するために、1994 年には早くも 2 GBx 7 台のディスクアレイ装置 (RAID) が導入されている[3]。現

[3]ただし最初の RAID システムはたびたび故障し、大事なファイルが消えるなどして泣かされた。

在では 100GB クラスの RAID システムが２式あるが、いずれも２系統の親計算機 (Unix) を持たせ、計算機保守の際も同時に止めないことで完全な不停止サービスを実現している[4]。

4.3.2 Operator Interfaces and Applications

Operator Interfaces は、a) Windows PC（Visual Basic プログラム用）、b) タッチパネル、および c) X端末（X-window アプリ用）、の３種類の混成になっている。運転時の写真を図 4 に示す。

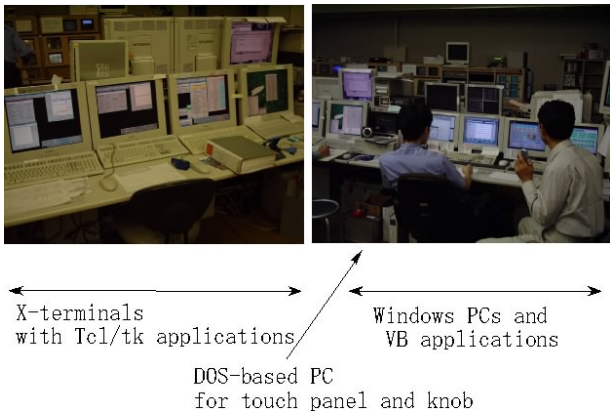

図 4: Operator’s interfaces (photo)

電子入射器では、80 年代末から DOS PC をオペレータコンソールとして導入する [26] など、早くから PC の利用を行ってきた。PC がコンソールに利用されたのは、運転業務に必要なグラフィックや漢字のある画面作成に、基幹計算機の Unix より PC のソフト開発環境が優れていると判断したからである。1996 年に DOS から Windows へ移行し、今日に至っている [19, 27, 28]。現在、PC コンソールは WindowsNT 4（一部 Windows2000）の PC 約 10 式からなる。加速器機器の全体監視や単純操作のほか、電子運転ログブック [29] が日々利用されている。

タッチパネルとノブを用いた操作システムは初代制御系から用意されていたが、現在使用されているのは DOS PC と PCTCP ソフトウエアライブラリを利用したもので、1991 年に最初のセットが導入された。その後表示器がカラー化され、６式が導入された [30]。現在でもノブを用いた直感的な操作はオペレータに好まれており、頻繁に利用されている。

KEKB 計画に対応した 1998 年以後の新規運転アプリケーションプログラムは、主に Tck/TK (Unix 計算機) で開発されX端末で表示された [45, 47, 51, 52]。これは、KEKB 電子入射器改造に伴って必要になったソフトウエアへの迅速な対応が、Windows 環境では難しかったからである。また、KEKB リング用シミュレーション用に開発されたX-window 向けの SAD script が電子入射器にも応用できるようになり、X端末で走る運転アプリケーションはどんどん増えていった。

ところで、EPICS ツールキットを使った KEKB リングの制御システムは電子入射器制御系とは独立なシステムである。しかし、リング側運転アプリケーションから電子入射器のデータが必要な場合がままあるため、Channnel Access Gateway が用意されている [15]（図 3 右上）。

4.3.3 Local Controllers

1993 年の制御系更新時点では、フロントエンド部を CAMAC から VME に移行、しかし micro-processor ベースのローカルコントローラはそのまま継続利用、という方針を取った（表 1 参照）。各セクタに１台づつ、計８台（１年後に９台）の VME-bus 計算機（68040 25MHz, OS-9 v2.4）が設置された。OS-9 はリアルタイム性とある程度の stand-alone 開発環境を合わせ持つ OS で、当時入手できた他のリアルタイム OS に比べ安価であったため採用した。各 VME で約 50 の運転プログラムがメモリに常駐（4MB 中 3.5MB 程度占有）し、cpu 稼働率は平均して 10%程度であった。

しかし、導入以来 15 年以上の時間がたってローカルコントローラの保守が困難になってきた。各機器のローカルコントローラは、単純な IO 数 100 点と簡単なロジックが必要である。また、上流の制御計算機と何らかの通信が出来なくてはならない。ローカルコントローラ更新時の機種選定は各機器の担当者が行ったが、結果的にクライストロン、電磁石電源、真空の３機器で PLC（横河 FA-M3）が選定された [17, 31, 32, 18]。PLC は、単純 IO では VME などと比較して安価であり、ラダーで簡単な前処理が可能である。また横河 FA-M3 は、ネットワーク通信機能を他社 PLC と比較検討した結果、我々の制御システムに組み込みやすい UDP プロトコルが実装されていた。1993 年以降の制御機器ごとのローカルコントローラの更新を、表 3 に整理した [5]。

なお、Trigger-delay の新ローカルコントローラにはネットワーク機能つき CAMAC が選定されたが、その後 VME module の導入に移行している。当初 CAMAC が選ばれたのは、必要な機能を持つ市販モジュールが CAMAC でしか見つからなかったためである [57]。また、BPM（ビームモニタ）用には VME が選定された。BPM の場合複雑な前処理ソフトが必要で、PLC では予

[4]計画停電時および重要な RAID 設定変更時には停止する

[5]更新の事情は [25, 18] でも説明されている。

機器	93 年以前	93 年-移行期 (移行時期)	現在
Klystron 70 台	CAMAC ― m.processor	VME ― m.processor ('97-'98)	PLC 70 台
Magnet 500 台	CAMAC ― m.processor	VME ― m.processor ('96-'00)	PLC 50 台
Vacuum 280 台	CAMAC ― m.processor	VME ― m.processor ('96-'97)	PLC 18 台
Trigger 16 式 140 信号	CAMAC ― m.processor	VME ― m.processor ('97-'02)	CAMAC, 11 台 VME 5 式
BPM 90 式	無し	('97 新規)	VME 19 台

表 3: 制御機器とローカルコントローラ

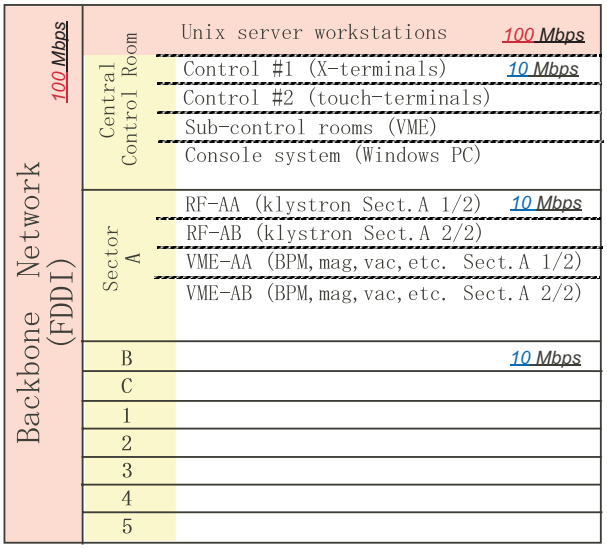

図 5: 制御ネットワークのセグメント

定する機能の実現は困難と判断した [33, 34]。さらに、クライストロン波形取り込みのためには VXI が [35]、また一部の特殊な用途向けに GPIB/RS232C のインターフェースも準備されている。

4.3.4 Network Systems

2 代目制御系用に 1993 年以降導入されたネットワークは、Ethernet や FDDI など国際標準規格に従ったものである。また、通信規約には業界標準の TCP/IP (TCP 及び UDP) を使用している[6]。標準品の採用により、ネットワークの保守や拡張を安価に行えるようになった。

現在の制御ネットワークは約５０のセグメントに分割され、電子入射器棟全域と KEK 所内の何ヶ所かをカバーしている。これらのセグメントは 100Mbps の FDDI バックボーンを持つ中央スイッチングハブで集中管理している。セグメントを分類すれば、a) FDDI に直結された Unix 計算機 (100Mbps)、b) 中央 Hub に直結した制御室周辺の計算機セグメント (10/100Mbps)、c) 中央からスター型配線（10BaseFL ＝光ファイバ回線）で接続された各セクタのセグメント (10Mbps)、などがある（図 5）。制御ネットワークについては、次節でさらに考察を行う。

4.4 制御ネットワーク

現在の制御系では、制御ネットワークの障害は電子入射器運転停止に直結してしまう。ネットワークを運転期間中安定に運用することは極めて重要で、そのためにいろいろな工夫がはかられている。

[6]初代制御システムでは、専用光回線と独自の通信規約を使用した。

4.4.1 ネットワーク形態

ネットワークがバス型であると[7]、障害箇所の特定が著しく困難になることを経験している。障害の場所特定や回復の容易さから、加速器制御のような大規模ネットワークのトポロジはスター型が望ましい。現制御系のネットワークも、基本的にスター型である。

また、制御機器グループ別にできるだけ独立のスター型ネットワークを用意し、一つの系統に障害が発生しても影響が他に及ばないようにすると同時に、トラフィックの分離も行っている。一つのグループネットワークの内部でも、スイッチングハブやブリッジを使用して、トラフィックの局所的な飽和による効率の低下を防ぐよう、ネットワークセグメントを構成する。この目的のためにもスター型トポロジは有利である。

4.4.2 冗長性

ネットワークに障害が発生した時、スター型トポロジの採用により障害発見は早くなるが、制御機能の一時停止は避けられない。そこで、経済的に許す限りネットワークに冗長性を持たせている。

基幹部分で使われている FDDI はその規格の中に 2 重化が含まれており、1 箇所の障害が全体に影響を与えることはない。制御系の主要部分は FDDI で結ばれているので、信頼性は高い[8]。

中央から各セクタへの接続についても、スター型配

[7]90 年代前半の Yellow cable (10Base5) はバス型の例。

[8]最近では大規模なネットワークは GbE (Giga-bit-Ethernet) を利用することが多いが、FDDI のような冗長性は無い。我々は FDDI の高信頼性を経験してきたが、維持費用の点から近い将来 GbE に移行せざるを得ないと考えている。

線を 2 重に張り、冗長性を持った光トランシーバに接続した。このような接続は、各機器のグループネットワークについてそれぞれ約 15 箇所の中継点で行っている。

4.4.3 光ファイバ使用による耐ノイズ性

電子入射器では、約 60 台の高出力クライストロンモジュレータがギャラリーに配置され、それぞれが強烈な電磁パルスノイズを出す。長い信号線を引かざるを得ない制御ネットワークでは、このノイズの影響は深刻である。このため、中央から各セクターの中継ボックスを経由して制御機器まで、ネットワーク配線は全て光ファイバ伝送 (10BaseFL 規格）で行っている。

4.4.4 研究所ネットワークとのリンク

電子入射器の制御ネットワークは、研究所ネットワーク (KEK 所内ネット) とは独立に運用され、物理的にも別物である。しかし制御ネットワークは利用できる場所が電子入射器棟に限定されるので、どうしても所内ネット側の計算機から電子入射器の状態を見たくなる場合がある。そこで、開発用計算機 1 台を所内ネットと制御ネットワークの両方に接続し、機器状態の読み出しのみ可能 (状態変更は出来ない) になるよう設定した。この結果、所内ネット側から加速器機器の状態読み出しが可能になっている。また、この計算機を通して、制御ネットワーク側からプリンターなど研究所共有の資源の利用を可能にしている。

4.4.5 運転時のネットワークトラフィック

ネットワークトラフィックは特定のセグメントに集中しているわけではない。一例として、表 4 に図 5 で示したセグメントのネットワークトラフィックの実測値を示すが、すべて 10Mbps に比べて 2-3 桁小さい[9], [10]。現状のトラフィック量であれば、既存のネットワーク容量で十分と言える。

network segment	traffic frames/s (kB/s)	devices in the segment
RF-1A	29 (2.8)	klystron (1sector 1st-half)
RF-CB	170 (16)	klystron (Csector 2nd-half)
VME-1B	35 (6.9)	BPM, vacuum (1sector 2nd-half)
VME-CA	26 (6.3)	BPM, vac, magnet (Csector 1st-half)

表 4: 典型的ネットワークトラフィック

[9] 1998 年 10 月の測定。現在はこの値より大きいと推測される。
[10] 表 4 中、RF-CB のトラフィックが多いのは、クライストロン波形監視プログラム（X-window ベース）が走っていたため。

4.5 履歴（*Archive*）システム

現在の制御系には、電子入射器のほぼ全部の制御機器信号（約 5000 点、10kB）を監視する履歴情報記録サブシステムが組み込まれている [36, 37]。信号の変動を約 1 秒の周期で監視し、変化があったときのみ記録をファイルに残している。現在 klystron、magnet、真空、などが対象で、BPM は準備中である。履歴サブシステムは 1993 年時点では VME/OS-9 ベースのシステムだったが、KEKB 計画で導入された PLC に更新された機器分が対応できなかった。1999 年夏に PLC 更新分を Linux PC でデータ収集する仕組みが整備され [28, 38]、履歴サブシステムも再開した。クライストロンを例にとると、1 台のクライストロン当り 18 個の信号 (全体では約 1200 の信号) を 1 秒弱の周期で監視している。履歴ファイルは各クライストロン毎に出来、クライストロン全体では 3 か月で数 GB、電子入射器全体で 4–5GB の大きさになる。

記録として残る履歴ファイルは ASCII 文字列の集合体の形である。その量が膨大なため、必要な情報をファイルから引き出すことは単純ではない。そこで、履歴情報を簡単にグラフ化できるよう、市販の graphic package[11] を採用して dev_hist と呼ばれる汎用ツールを開発した。dev_hist は制御用 Unix 計算機で動作する。メニュで表示すべき機器を選び、ボタンやメニューで表示期間を指定する。図 6 は、その外観である。dev_hist は 1996 年から使われ始め、もっぱらトラブルが起こった時に障害発生時刻前後の機器の状態を調査するのに利用されている。障害を吟味、同定するのに絶大な威力を発揮する場合がある。また、一定期間中の制御機器の傾向（トレンド）を調べることも可能である。

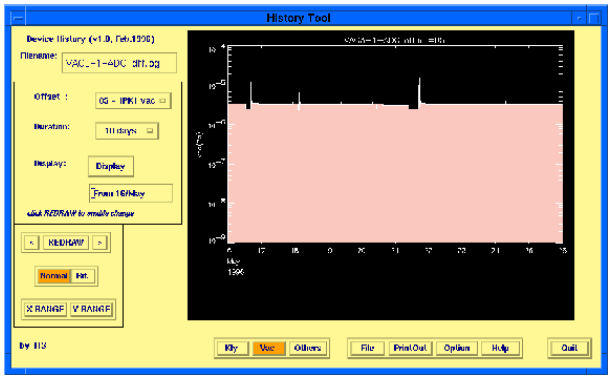

図 6: 機器履歴表示ツール dev_hist の外観

クライストロンと真空については、dev_hist のグラフ化ルーチンを呼ぶ Web 画面が 2000 年から利用可能に

[11] PV-WAVE。Visual Numerics 社の製品。

なった [38]。検索条件が dev_hist に比べ制限されるが、Web を採用したことで、「 誰でも (Web が使える職員なら)、どこでも (居室や自宅でも)、いつでも (制御グループ職員が居なくても)」クライストロン・真空の履歴情報が得られる。多くの電子入射器職員にとって、無くてはならないシステムになっている。

なお、歴史的経緯により、電子入射器にはクライストロンや真空以外にも異なる仕様の履歴システムが多数存在する[12]。これらを統一的に扱えるように、CORBA（あるいは XML）層を wrap して共通 API を整備し、上位アプリケーションを目的別に共通化しようという研究が続けられている [16, 40, 39]。

5 ソフトウェア構成

5.1 歴史的経緯

初代制御系では、ミニコンピュータと専用ネットワークは制御メッセージの通信システムの交換機と考えられた。例えば主制御卓のタッチパネルを操作すると、制御系（制御卓用ミニコン–ネットワーク–現場ミニコン）を経由して制御メッセージが現場のローカルコントローラに届き、目的の機器の設定が行われた。また、その結果としての現場機器の状態遷移は、逆方向に制御系を経由して状態表示プログラムに届いた。その round-trip 時間は、100ms と実測されている [4]。

初代制御系の時代、年月が経過するにつれ改造された仕様の異なるコントローラが追加導入されていった[13]が、個々のアプリケーションがコントローラの違いを記述して対応していた。当時はアプリケーション数が少なかったとはいえ、機器の追加や変更のたびにコーディングが必要で、迅速な対応は困難であった。また、当時のネットワーク伝送系の信頼度は今日的な感覚では低かった。機器操作がうまくいかない場合、コントローラにメッセージが届かないのか、あるいは状態が変わったというデータが上位に上がってこないのか、障害の場所の切り分けが困難だった。さらに、初代制御系のミニコンはメモリ制限から新規のプログラムを追加することはほとんど無理だった。このため現制御系の設計を始めた 1990 年頃には、初代制御系の能力不足を補うためのサブシステムが多数導入されたが [7, 8]、困ったことに多種多様な計算機機種が混在していた。

[12] 例えば、冷却水温度、ビーム電流記録、など。

[13] 例えば 1990 年ごろ、screen controller は 3 種類、magnet power-supply controller は 2 種類 (firmware の違いによりソフト的にはもっと多種類) あった。

5.2 設計方針と全体構成

現制御系の制御サービス（いわゆるサーバ層）は、加速器装置のコントローラの物理的な処理を表現する下位層 (Lower level servers)、及び加速器装置の論理的な処理を表現する上位層 (Middle level servers)、から構成されている。サーバ層が 2 層に分かれている点は、他の加速器制御系ではあまりみられない特徴である。下位層と上位層、及び上位層とアプリケーションソフトウェア層 (Upper level programs) の間は、ネットワークについて透過な RPC (Remote Procedure Call) によって接続されるが、ソフトウェア開発者は通常その存在を意識することはない。これらの階層構造は、図 2 に示されている。

上位層サーバは加速器構成要素に対応しており、Unix 計算機上で走っている。これら上位層のサーバは静的データベースに従って下位層のサーバと通信し、キャッシング・単位換算や障害処理を行なう。下位層は基本的 IO (VME module) やローカルコントローラに対応している。このようにサーバ層を 2 層構造にしたのは、初代制御系で問題になったコントローラ仕様の違いを下位サーバ層で隠蔽し、アプリケーション層に影響を与えないようにという考えからである。また、ローカルコントローラは 1993 年の新制御系の運用開始移行時間をかけて徐々に更新したが（4.3.3 節および表 3 参照）、それに伴って上位層は新しい下位層コントローラに対応した機能拡張を行う必要があった。しかし、上位層とアプリケーションソフトウェア層の間の取り決めについての変更は最小限に押えたため、開発済みのアプリケーションは新旧のローカルコントローラが混在した移行期もほとんど変更なしに継続して利用できた。これも 2 層構造の特徴が生かされた結果といえるだろう。現制御系で使用可能な上位・下位サーバの説明は、5.3.4 節に示す。

サーバやアプリケーションの間のプロセス間通信には、設計当時 defact standard になりつつあった TCP/IP プロトコルを使い、専用の RPC を開発した [41]。TCP/IP の採用は、前述した多種多様な計算機機種が混在した状況で相互通信を可能にするためにも最善の策と考えられた。

5.3 制御ソフトウェアの詳細

5.3.1 基本操作のモデル

制御系のソフトウェアは、図 7 に示す簡略化された基本操作モデルを原則としている [14]。

[14] この節の内容は [22] にも説明がある。

手順１　operator（ユーザー）は、制御しようとする特定の object（名前 *name* で区別される）に対し、命令 *com* を送る。必要ならばその *com* に付随した設定値 *value* も併せて送られるが、この場合は object の property が変更される。

手順２ａ　送られた *com* が指定した object に正常に作用したならば、*return-code* として 0 が operator に返される。*com* によっては object の property (*value*) も帰って来る。

手順２ｂ　エラーが起こればそのエラーに対応した負値の *return-code* が返される。この場合、operator は *return-code* を調べて何のエラーが発生したか知ることが出来る。

結局、operator(ユーザー) は制御系に対して (*com, name, value*) の組の制御メッセージを送ってその回答 (*return-code, value*) を得る、という手順を繰り返して実際の加速器を制御する。図 7 の例では、”TEST2”に対し *value* を 25 に設定 (SET) し、*return-code* 0(成功) を得ている。

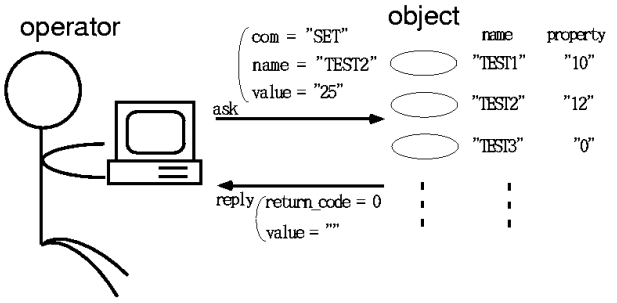

図 7: Software model of the control system.

この基本操作モデルでは、operator は制御したい対象をどう操作するかという点のみを考えればよく、途中経路 (Ethernet とか VME とか) のハードウェアについて考えたり知識を要求されたりすることはない。また、RPC 部分のネットワーク通信コードが単純になり、機器によらず共通化できている [42]。一方このモデルでは、あらかじめ整備された *com* のみで機器を操作することになり、設計によっては制限がうまれる[15]。また、round-trip が操作の基本単位なので、相手機器側に問題がある時やネットワークに障害がある場合は、返答待ちのタイムアウト処理に時間がかかってしまう問題がある[16]。

[15]例えば複数の property・object の同時一括変更がハードでサポートされていてもやりにくい。

[16]現制御系の制御単位を round-trip にしたのは、初代制御系では操作単位が one-way で、またネットワーク伝送系の信頼度の低さに泣かされていたので、これを反面教師としたという側面がある。

5.3.2　table と log (database)

制御系では、静的データベースとして管理 table、運転操作記録データベースとして log file を利用している（図 8）。

object の名前 (*name*) が実際にはどの VME のどのモジュール、何チャンネルであるかといった対応付けは、管理 table で管理されている。管理 table は通常の ASCII file で、エディタによって修正・追加する。サーバは、operator の制御対象 (object) に関するハードウェア情報をこの table から得る。管理 table は制御機器毎に存在するが、NFS を利用してすべての計算機で同じものを参照している。各セクタの VME 計算機は、自分が担当するセクタ分だけを元 table から切り出したローカル table を持っており、VME で走る運転ソフトはこちらを参照している。

制御系が実際の機器制御に伴って出力する情報（制御記録、エラー、など）は、操作記録 (log) に残される。情報記録のレベルは２つあり（エラー関連のみ情報を残すか、全てのアクセス記録を取るか）、制御機器の種類に応じていずれかを設定している。操作記録も、管理 table 同様 VME 側と Unix 計算機で共有している[17]。

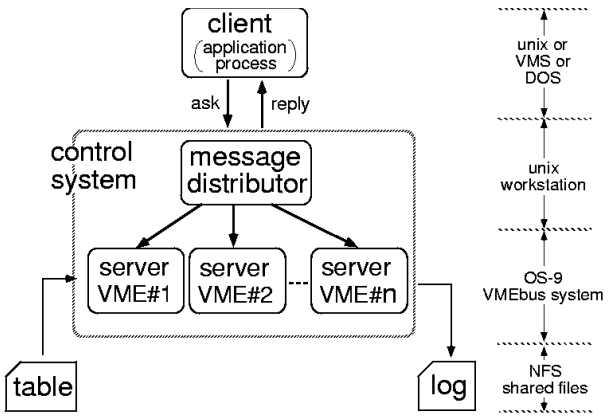

図 8: Control message distribution (Lower level)

5.3.3　制御メッセージの分配

上位層サーバの場合、client からの制御メッセージ（(*com, name, value*) の組）は直接サーバへ送られる。しかし、下位層サーバの場合、client からの制御メッセージは、２段階の処理プロセス（＝プログラム）がある（図 8）。client からの制御メッセージは、まず第１のプロセス (message distributor) に送られる。第１のプロセスは管理 table を参照して制御メッセージを目的の VME

[17]現在は、VME 側で発生した記録文字列は UDP で log マシン (Linux マシン) へ転送し、Unix と共通の log に書き込んでいる。

の第２のプロセスに転送するための郵便局であり、メッセージの転送以外何もしない。第２のプロセスはそれぞれの VME で実際に制御を行う server process である。第１プロセスと同じ管理 table を参照して、制御に必要なハードウェア情報を得ている。図 8 では片方向の矢印しか示していないが、server プロセスが用意した reply メッセージ（(*return-code, value*) の組）は同じ経路で client まで帰される。また、第１・第２ process が出力した制御情報（エラー情報など）は、同じ log に書き込まれる。この様に２段階に分割したのは、プロセス数が多くなっても単純な process の組合せで制御システムを構築した方が保守上有利と判断したためである。実際、message distributor は全ての制御要素で同じものが使用されている（管理 table と log だけが異なる）。server プロセスはおのおのの VME で独立に走るが、NFS により同じ実行イメージをロードしている。

> [参考] PLC の場合、事情はやや異なる。client からの制御メッセージは message distributer から各 PLC にメッセージが転送される。PLC の UDP 通信モジュールは、server として機能して reply を返す。これらの相関関係は図 8 と同じであるが、管理 table の内容は別途ラダーに組み込まれ、log への書き込みも起こらない。

さらに、上位層・下位層の制御メッセージの分配に関して、以下のような特長が指摘できる。

- socket の利用
  電子入射器では、歴史的に多種多様な計算機サブシステムが導入されてきた [7, 8] ため、プロセス間通信には異なる計算機機種間でも通信可能な TCP/IP プロトコル (socket 関数) を採用した [41]。また、socket 関連のパラメータを与えればサーバの RPC 部分のコードを半自動的に生成するテンプレートが開発され、利用されている [42]。
- ポート番号管理による非干渉性
  電子入射器では，制御の対照となる機器が多種類あるが、これまでに述べた message distributor, server, 管理 table, log はそれぞれの機器毎に独立に存在しており，異なる機器間ではお互いに干渉しない。この仕掛は，socket 通信で機器毎に異なるポート番号 (service name) を割り当てることで実現している。
- security
  message distributor は、socket 接続を受け付ける際に接続要求を出したホスト名を登録 table のものと照合している。登録 table に無いホストからの接続要求は拒否し、そのホスト名は log に記録される。また、ホストによって特定の命令（多くの場合”GET” など read-only コマンド）のみ受け付ける設定が可能である[18]。

5.3.4 制御可能な機器

制御系では、前述のように制御対象となる機器を２層に分類して考えている。

下位層は、VME module やローカルコントローラに対応する層である（表 5）。これらの要素は一部を除いてごく一般的なもので、電子入射器での制御以外にも広く利用可能と考えられる。

下位層の制御要素に対する命令 (*com*) は使用している VME モジュールに特有のハード仕様には依存しないように設計した。これは、将来保守上の理由などからモジュールの機種変更が必要になっても、これらの制御要素を利用する上位プログラムに影響が及ばない様にするためである。

上位層は、下位層の制御要素を利用して制御する機器である（表 6）。上位層の制御要素は下位層の機器に比べてより上位の概念で、電子入射器の加速器の構成要素に対応している。上位層のサーバは、下位層からみれば機器サービスを要求する client である。ただし 5.3.1 節で示される機器の制御手順は、両方の層で全く同じである。

制御要素	VMEmodule/PLC	コマンド・関数名
16bit D-out	PVME501/07	out16/outreg
16bit D-in	PVME501/04	in16/ingate
12bit ADC	PVME301,305	adc12
12bit DAC	PVME323	dac12
delay	TD4V	td4v
GPIB	DVME-GPIB	gpib
loop2	CAMAC-loop2	loopc
loop3	海津 8755	loop3
PLC	Yew FA-M3	plc

表 5: 制御可能な機器（下位層）

5.3.5 user’s interface

制御系とのインターフェースには、OS レベルのコマンドを利用する方法とＣ関数を利用する方法がある。これらは電子入射器運転用の Unix 計算機や VME 計算機

[18] この例が 5.4 節に示されている。

制御要素	下位層要素	コマンド・関数名
signal selector	D-out	coax
screen mon.	D-out/D-in	scrn
klystron	PLC	kly
magnet	PLC	mg
vacuum	PLC	vac
BPM	none	sp
interlock	PLC	intlk
trigger	td4v/loop3	trig

表 6: 制御可能な機器（上位層)

(OS-9) で利用できる[19]ほか、整備すれば PC(Linux, MS-DOS) でも利用可能である。

- コマンド利用
  制御系がインストールされた計算機では，機器を制御する OS レベルのコマンドが利用できる。コマンドは表 5・表 6 の機器それぞれに用意されている。簡単なテスト、速度が要求されない場合にはコマンド利用が便利である。

- C関数利用
  コマンドと同様C関数も表 5・表 6 の機器で用意されている。コマンド利用に比べ速度や複雑な条件判断が可能になる点で利点がある。ユーザーは、用意された制御ライブラリをリンクすることで電子入射器制御プログラムを開発できる。

表 5・表 6 の機器それぞれの利用の際の具体的な情報は、運転･開発マシン (表 2 参照) のオンラインマニュアルで得られる。一度概念を理解してしまえば、Unix/Linux 環境でこれらを参照するスタイルがお手軽である。また、コマンド・C関数を利用した運転用アプリケーションは、すでに多数開発され実際の運転で使用されている。運転ソフトウエアについては、6 節で詳述する。

5.4 Example(D-out)

この節では、機器制御の例を D-out service (16bit digital output) で具体的に解説してみよう。

- 管理 table
  D-out service の管理 table のうち入射部 VME(kannaduki) に関する部分を以下に示す。ここでは 2 枚の digital board (計 8 ポート) が定義されている。各 field は、左から a) ポート名、b)VME ホスト名、c) ボードアドレス、d) 各ボードでのチャンネル番号、を表している。

```
! [0] Injector
!     806 - PM screen
!     220,221 - PM cntroller
!     202 - CM   807 - WM
!name   nodename base-addr      channel
806     kannaduki fc4b0000      0
202     kannaduki fc4b0000      1
TEST0-1 kannaduki fc4b0000      2
807     kannaduki fc4b0000      3
TEST0-2 kannaduki fc4b0100      0
TEST0-3 kannaduki fc4b0100      1
220     kannaduki fc4b0100      2
221     kannaduki fc4b0100      3
```

- online manual
  D-out service を利用する際のコマンド（C関数）名は、表 5 で示したように out16(または outreg) である。運転 Unix 計算機などで、オンラインマニュアルを呼び出すことが出来る（図 9)。

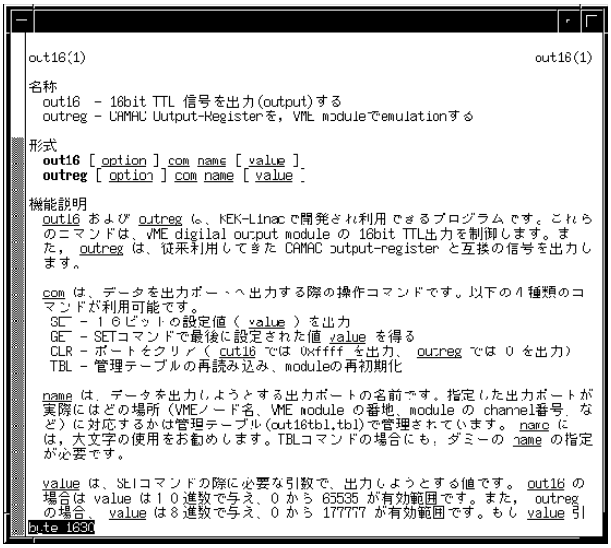

図 9: Online manual example

また、制御コマンドを引数無しで実行すれば簡単なヘルプが表示される。out16 の場合を以下に示す。

```
grape[164]% out16
out16: specify com and/or name

out16 v3.1b - digital output (16bit) control

Usage:   out16  com name [value] option ...
         com   - GET, SET, CLR, TBL
         name  - name of the port
         value - value to be set (for SET)
Option:  -h  help message
         -x    value in hexa
         -d    value in decimal
```

[19]VME 計算機では担当するセクタのみ制御出来る。

symbol definition(定義値)	意味
OUT_E_CM(-9)	共有メモリー使用不能
OUT_E_BUS(-10)	Bus-trap エラー（ボード不良）
OUT_E_COMMAND(-11)	指定したコマンド com がおかしい
OUT_E_NAME(-12)	指定した name 名は存在しない
OUT_E_TBL(-15)	管理 table が読めない
OUT_E_TBLPARAM(-14)	管理 table のパラメータ不良
OUT_E_MANYMODULE(-17)	管理 table の登録 board 枚数が多すぎ
OUT_E_OUTRANGE(-26)	指定値は有効範囲外
OUT_E_SCLIB(-81)	SCLIB(network 通信) のエラー
OUT_E_NETPROT(-82)	Linac 登録外、接続許可しない
OUT_E_SCOPEN(-83)	out16-daemon に接続できない
OUT_E_LIBRBUG(-20)	bug または version が合わない

表 7: error codes for out16/outreg functions

```
        -o    value in octal
        -lN   repeat N times
        -e    continue loop when eror
        -s    sleep 1 sec. after each call
        -sT   sleep T-sec. after each call
Example: out16 SET TEST-1 64
        out16 GET 200 -x -l100 -s
```

- コマンドによる操作例
  ここでは”TEST0-1”を、outreg コマンドを使って初期値 0 から 255 に変更する例を示す。数値は default では 8 進数表示、d-option で 10 進数表示である。

```
grape[169]% outreg get test0-1 -d   # 初期値
0
grape[170]% outreg set test0-1 255 -d
grape[171]% outreg get test0-1 -d   # 10進
255
grape[172]% outreg get test0-1      # 8進
377
grape[173]% outreg get test00-1     # 間違い例
out16 v3.1b: illegal portname specified
```

- Cプログラムからの制御例
  同様に、”TEST0-1”を 255 に変更するCプログラム例を示す。

```
main()
 {
    int     i, rtn;
    i = 255;                    # 255 設定
    rtn = outreg( "SET", "TEST0-1", &i );
    if( rtn ＜ 0 )   out\_errend( );
    printf( "suceeded" );
 }
```

  outreg(または out16) 関数の return 値は正常なら 0 だが、表 7 のようなエラー値が有り得る。この例では対応するエラーメッセージを表示して exit する out_errend() 関数を利用している。

- log
  「コマンドによる操作例」では grape からポート”TEST0-1”に対して読み書きを行った。この例では、log として以下に示す記録が残る。

```
15:24:14 grape> umdist-grape.1177:
    [331-80byte]-> com='SET' name='TEST0-1',
                   1 int's 0xff00,..
15:24:14 grape> umdist-kannaduki.4301:
                <- rtn=0 (no arguments)
15:35:22 grape> umdist-grape.1183:
                undefined name 'TEST00-1'
```

  このように制御系では全ての操作の記録を log に残すことが出来、時刻・依頼元 (この例では grape)・依頼先 (TEST0-1 at kannaduki) などを後からトレースすることが出来る。ただし、default 設定では D-out service は読みだし (*com*=”GET”等) は残さない設定なので、上の記録には現れていない。

- security check
  制御系は、依頼元ホストによって実行できる機能を制限している。例えば、開発用計算機 (maple) では、読みだし命令 (”GET”など) は実行できるが機器に何等かの変更を加える命令は実行できない様に設定している。以下の例では、maple から”TEST0-1”の値を読み出しているが変更には失敗している。

```
maple[200]% outreg get test0-1
377
maple[201]% outreg set test0-1 7
out16 v3.1b: fail to connect due to security
```

security check で引っかかった情報は操作記録と同様 log に記録される。

```
15:44:39 peach> umdist-maple.4464:
    [1918-80byte]-> com='SET' name='TEST0-1',
                    1 int's 0xfff8,..
15:44:40 peach> umdist-maple.4464:
    com specified is not allowd from maple
15:44:40 peach> umdist-maple.4464:
                  <- rtn=-82 (errr return)
```

5.5 制御メッセージの速度

典型的な例である D-out サービスで、制御メッセージの速度を測定した[20]。

表 8 は、client(Unix 計算機など) と VME 計算機の server 間の、UDP プロトコルでの往復 (round-trip) 時間を測定した結果である。6–8 ms かかっている大半 (5 ms 程度と推測) が、相対的に CPU 能力の低い VME 計算機での処理時間と考えられている[21]。また、Unix 計算機のみでの簡単な通信テストでは、100byte 程度の制御メッセージを UDP プロトコルで往復させるのに、ネットワーク越しの 2 計算機間で 1ms、同じ計算機内のプロセス間で 0.1ms 弱である。

client	server	round-trip
DECstation5000 (peach)	VME (saburo)	7 ms
PC9801BA(486,40MHz) (tpinj6)	VME (saburo)	8 ms
AlphaServer DS20E (poplar)	VME (hatsuhi)	6 ms

表 8: round-trip time (UDP, 80byte)

表 9 には運転状態での client と VME 間の制御メッセージ往復速度を示した。表 8 の条件と比較すると、a) message distrubutor が仲介する、b) security check が入っている、点が異なっている。mesage distributor が仲介することによる時間増は 1 ms 以下と見積られている。

5.6 運転中の機器サーバの負荷

運転中に蓄積されたサーバ log を解析すれば、それぞれの機器サーバがどれだけの量の制御メッセージ（トランザクション）を処理したか計算することができる。1998 年から 2001 年までの 4 年間の推移を表 10 に示す。

[20]outreg の GET を 1000-10000 回程度 loop させて時間を測定。
[21]なにぶん clock 25MHz ですから..

client	mess.dist.	server	round-trip
DECstation (peach)	DECstation (peach)	VME (saburo)	16 ms
PC9801BA (tpinj6)	DEC3000 (grape)	VME (saburo)	13 ms
AlphaServer (poplar)	AlphaServer (poplar)	VME (hatsuhi)	6 ms

表 9: round-trip time (UDP, 80byte)

機器 [transactions/s]	total Jun.98	total Jun.99	total Jun.00	total Jun.01
Klystron	5.3 tr./s	4.6	18	27
Magnet	63 tr./s	70	24	23
Vacuum	no data	no data	45 tr./s	2.1
Trigger	0.028 tr./s	0.035	0.035	1.6
BPM	32 tr./s	158	252	299

表 10: 1998-2001 年の機器サーバトランザクション

この表から、a) トランザクション総量は毎年増加して 2001 年には毎秒 350 に達していること、また b) BPM (beam-position monitor) の処理量が圧倒的に多いこと、などが分かる。さらに解析すれば、BPM へのトランザクションの半分以上が KEKB リング制御系からの寄与とわかり、トランザクション総量の増加は KEKB コミッショニング活動と深くかかわっているとわかる [43, 44]。

6 運転ソフトウェアとビーム制御

6.1 アプリケーションソフトウェア

これまで述べてきたように、入射器の制御システムは、階層化された多数の要素から構成されているが、さらに安定した信頼性の高い運転を実現するために、制御機器や計算機、ネットワーク、ソフトウェアの質を高める努力が払われてきた [45, 46]。

このような制御システムを基礎として、1997 年に KEKB 入射に向けた増強後のコミッショニングが開始されてからは、上位のクライアントソフトウェアとしてさまざまな運転用アプリケーションソフトウェアが構築されてきた [47]。それらの運転用ソフトウェアは主に、Tcl[48] または SADscript[14] というスクリプト言語で記述され、X-Window 上の Tk ウィジェットを用いて画面上で操作が行なわれている。その数はビームスタディや測定用のソフトウェアを含めて登録されているものだけで 150 ほどになっている。

6.2 情報の交換

コミッショニングの直前やその最中においては、加速器の装置やその情報が日々更新される。それらの基本情報は装置の担当グループが管理する場合が多いので、各担当グループやコミッショニンググループと制御システムの間で情報が円滑に交換できるよう注意している。例えば、居室のパーソナルコンピュータからデータベースを更新することができるように、標準のファイル共有プロトコル（Macintosh の AppleShare や Windows の NetBIOS-SMB）をファイアウォール上の Unix 計算機でサービスしている[22]。

担当グループから制御システムに渡る情報はスプレッドシート（Microsoft の Excel など）の形を取る場合が多い。現在は、新しい情報を実際の運転に使用するデータベースに反映させる手順は制御グループが担当している。

また、加速器のシミュレーションソフトウェアともさまざまな情報を交換しなければならない。入射器で以前から使用している TRANSPORT[49] は入出力をファイルで行なうので、その規約と合わせるために入力ファイルの準備や、出力ファイルの解釈を外部で行なわなくてはならない。そこで、運転パラメータなどの制御システムを通して得られる動的な情報と、静的なデータベースとから入力ファイルを生成し、計算を行なっている。

さらに、上に述べたように SAD も多数の運転用ソフトウェアで使用されている。SAD はビームシミュレーションソフトウェアではあるが、Mathematica 相当の SADscript というインタプリタ言語を持っており、また、KEKB のコミッショニング以降、X-Window の GUI を作る機能も追加されている。そこで SAD と入射器の制御システムの間では TRANSPORT と同様の比較的静的な情報の他に、直接に入射器の制御機能を通して相互作用して、柔軟な運転ソフトウェアが構築されている。

制御システムとシミュレーションソフトウェアの間の

[22]CAP や SAMBA という Freeware を使用している。

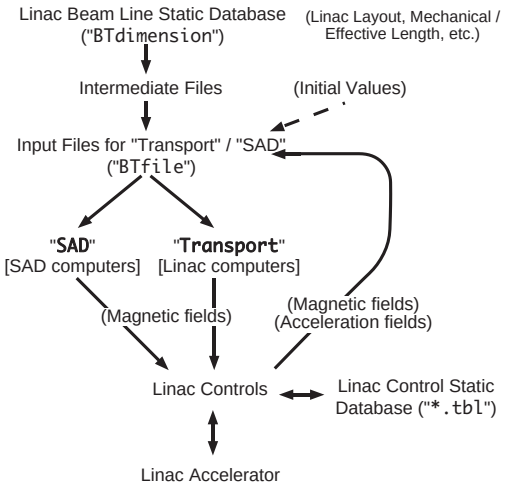

図 10: シミュレーションソフトウェアと制御システムの間の情報の流れ。

情報は加速器モデルとして意味のあるものになるように制御システム内でできるだけ変換をしている。例えば、加速電界とか、磁場勾配といった情報がやりとりされることになる。しかし、較正係数がまだ正確にわからないものも少からずあるので、ビームを使った較正などを進めているところである。

1998 年秋からの KEKB リングの運転においては入射器とリングの制御システムの間の協調も重要になった。KEKB リングの制御システムは EPICS [10] を採用したので、入射器の制御システムの RPC 規約と、EPICS の Channel Access 規約の変換をするための Channel Access Server というゲートウェイソフトウェアを準備している [15]。SAD で書かれた一部のソフトウェアや KEKB 全体のアラーム表示などにこのゲートウェイが利用されている。

また、条件が整えば、EPICS に限らない上位インターフェイスとして定義された cdev [12] や CORBA [16] の導入も将来の入射器の制御に有用であると考えており、実装を進めている。これらを利用することによって、対象指向プログラミングを行なうことができ、国際協力の見地からもソフトウェアやアイデアの交換が促進できるものと思われる。

6.3 オペレータインタフェース

KEKB 入射器のコミッショニングにおいては、当初 Windows の環境でソフトウェア開発が行なわれることが想定されていた。しかし、Windows のコンソールと制御システムとの間にゲートウェイがおかれていたことなどが障害となって、途中からソフトウェアの開発が停滞気味になってしまった。一方、ビームスタディや試験

には以前から X-Window 上の運転ソフトウェアが利用されていた [50]。

そこで、コミッショニンググループ内で新しく作るソフトウェアは X-Window 上で Tk ウィジェットを利用して主に Tcl と SADscript で作成することになり、制御グループも積極的に開発に関わった。ソフトウェア開発やデバッグのための環境や、共通に利用するライブラリルーチン、運転時のソフトウェア自体の障害記録、などが用意されている。

いづれの言語からも入射器の制御との接続は単純で、またいづれもインタプリタであるために開発、試験の繰り返し期間が短縮され、開発効率が大幅に向上したと思われる。特にこれまでシミュレーションプログラムを実時間で動作させることは、試験的には可能であっても、現実にはさまざまな困難があったが、KEKB のコミッショニング以降は複数の運転ソフトウェアで日常的に利用されるようになっている。

6.4 運転用アプリケーション

以上のような環境のもとで、次のような運転用アプリケーションソフトウェアが開発されてきている。

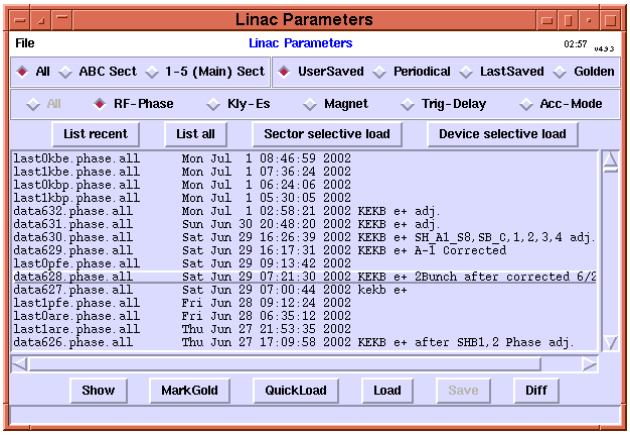

図 11: 運転パラメータを保存、設定、比較するためのパネル。多数のオプションを備えており、場所や種類によって機器を選択したり、ビームモードなどを区別したりできる。データベースは多数のプログラムで相互利用されている。

- ビーム位置モニタのビームによる較正。
- ビーム位置モニタのビームによる精度評価。
- 加速装置との対応付けをしたビーム軌道表示。
- ワイヤスキャナ上のビームの表示。
- ビームロス表示。
- エネルギーアナライザでのエネルギー分布の表示。
- エネルギー安定度の表示。
- 能動的及び受動的な各種パラメータの相関プロットと統計解析。

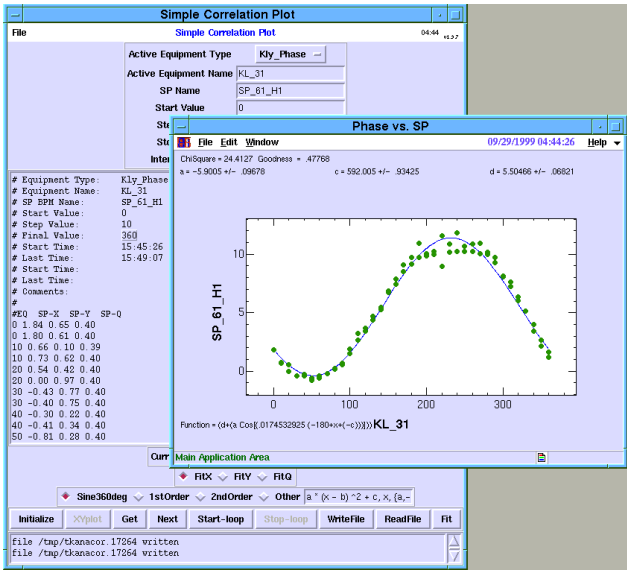

図 12: 相関プロットの例。さまざまな機器の情報の間でプロットを作ることができ、また、関数を選んで最小二乗法によるフィットが行える。

- バンチャ部シミュレータ。
- ビームオプティクス表示。
- Q-スキャンによるエミッタンス測定及びマッチング。
- 4 台のワイヤスキャナによるエミッタンス測定及びマッチング。
- Isochronous かつ Achromatic な Arc 部の評価と補正。
- 繰り返しによるビーム軌道補正。
- ローカルバンプによるウェイク場の補正。
- ビームモード切り替え。
- Arc 部と終端部でのエネルギーフィードバック。
- 多数の場所での軌道フィードバック。
- Downhill Simplex によるビームの最適化。

いずれのソフトウェアもだれもが操作できるように、X-Window のグラフィカルユーザインターフェースを備えている。それらの例を図 11、12 に示す。また、入射器の安定化に寄与しているビームモードスイッチとフィードバックパネルについて次項で詳しく説明する。

6.5 *KEKB* 入射器の安定化

KEKB 入射器のコミッショニングを開始したころは、3.5 GeV 陽電子発生に使われる 10 nC の一次電子の安定な加速のために努力が払われ、数々の技術改良により、これを達成することができた。しかし、さらに実際の運転を行う上では、高品質のビームの再現性の問題が重要となってきた。同じビームを別の時間に再現しようとすると、微妙な運転パラメータの調整を必要とし、時間

がかかる上、だれでも調整できるわけではなかった。

解析を進めるうちに、気温、水温などの環境の変化、モード間で 10 倍以上異なるビーム特性の切り替え、意識的に変化させた他の運転パラメータ、などに対して、各機器のパラメータの設定値からのずれが設計したときの許容値よりも大きい場合があることが指摘された。そこで、各パラメータの、ビームに対する変動の許容度が詳しく調べられ、パラメータ設定の際にその許容値を満足させるための方法が検討された。

例えば、コミッショニングは当初、通常運転とは別に部分的に行なわれたため、一部の電磁石は初期化が行なわれなかったり、消費時間を無視して消磁が行なわれたりした。しかし、実際の切り替え運転では限られた時間内に許容値内の磁場の設定が必要となるので、効率的な初期化の方法が開発された。具体的には、励磁特性を測定したときと同じ、ゼロと最大値の間の電流設定ループを一度だけ回るように設定を行なうこととした。

これらの他に、数多くの操作の中での単純な操作誤りに気付かず、問題の解析を困難にしている場合もしばしば見受けられた。それらは、ソフトウェアで自動化することによってできるだけ避けることにした [52]。

個々の機器の再現性の向上で対処できずに残ったビームの変動は、ビームを使ったエネルギーや軌道のフィードバックで対処することとした [51]。

6.5.1 ビーム・モード・スイッチ

上に述べたように、4 つのビームモード間の切り替えにおいては、各加速器機器のパラメータの再現性と信頼性が重要である。パラメータ切り替えのために用意されたソフトウェアは、現在では以下のような切り替え項目を持っている。

- 電磁石の簡易初期化
- 電磁石のパラメータ（主に電流値）設定
- rf のパラメータ（主に位相値）設定
- タイミングのパラメータ（主に待機モード）設定
- 電子銃のパラメータ設定
- 陽電子ターゲットとシケインの操作
- ビームモニタの測定モードとダイナミックレンジ切り替え
- ビームプロファイルモニタの操作
- 初期ビーム繰り返しの設定
- ビームトランスポートラインの選択
- 下流の加速器の運転システムへの通知
- 制御室での音声の発生
- 機器の状態、パラメータの差分表示と記録
- 各ビームモードのビームフィードバックの再起動

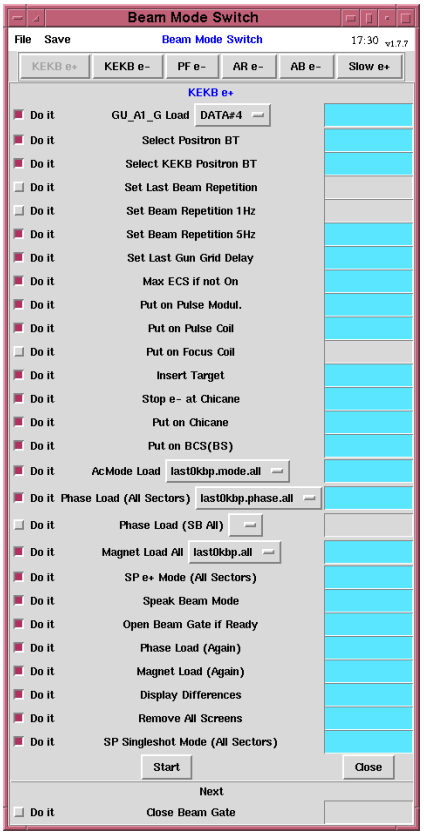

図 13: ビームモードスイッチパネルで、KEKB e^+ がを指定した状態。左の Check-button で項目の選択、非選択を変更でき、また Pull-down menu でパラメータファイルを選ぶことができる。右端は実行状態を表す。

各項目は、図 13 に示すようなパネルによってオペレータの判断でいつでも選択、非選択を変更でき、さらにその状態を保存しておくことができる。また、新しい項目の追加は簡単なデータベースの変更によって行なうことができる。もしも回復不可能な障害が起こった場合には、その旨が画面上に表示、記録され、障害が取除かれた時にオペレータが再試行することができる。

パラメータ設定と記した部分は、直前の同じビームモードで使用したパラメータを通常使用するが、他のパラメータをオペレータの判断で選択することも可能になっている。これらのパラメータはフィードバックや手動の調整で毎回異なるため全て記録を残しており、また再使用が可能である。

電磁石の初期化については磁場の再現性、急激な変化に対する電磁石電源の許容度、AC 電源容量、通信時の誤り率と制御システムの下位層、上位層での再試行、切り替えソフトウェア側での再試行、などについて繰り返し試験が行なわれ、改良されてきている。しかし、もっとも時間を消費する部分でもあるので、しばしば改善が行われている。

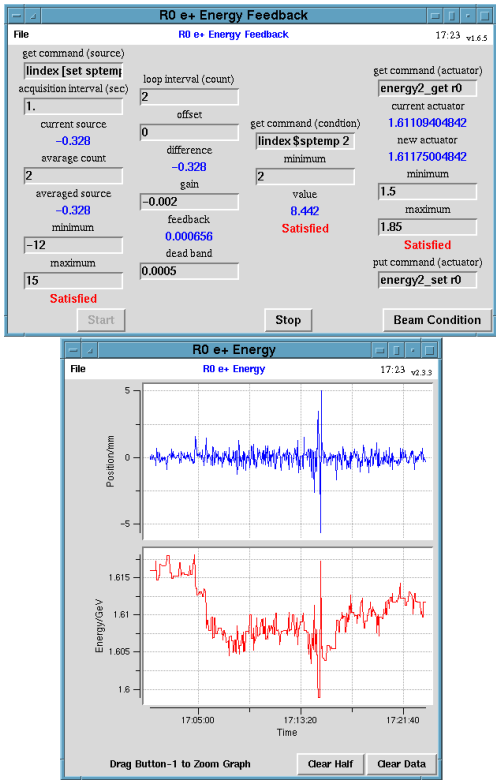

図 14: フィードバックの例として、R セクタのエネルギーフィードバックの設定パネルとグラフ。運転時であってもフィードバックの設定パラメータは簡単な前処理、後処理を含めて変更可能で、また、ソフトウェアのほとんどの部分は他のフィードバックと共通となっている。

6.5.2 ビーム・フィード バック・ループ

現在使用されているフィードバックは、加速器機器に閉じたフィードバック、ビームエネルギー のフィードバック、そしてビーム軌道のフィードバックに分類される。それらの基本的なソフトウェアは共通になっていて、単純な PID 制御を行う以下のような部分から構成されている。

- ビームモードやビーム電流などの条件の確認
- モニタ値の取得、時間移動平均、許容範囲の確認、その他の特別に指定された後処理
- 変換係数とフィードバックゲインを適用したフィードバック量の計算、許容範囲の確認
- 許容範囲の確認、特別に指定された前処理を施した値のアクチュエータへの設定
- 全体の制御とグラフ表示、記録、他のプログラムとのインターフェース

例えば、エネルギーフィードバックでは、分散の大きい場所でのビーム位置をモニタとして使い、(エネルギー幅を増大させないように) 2 台のクライストロンの

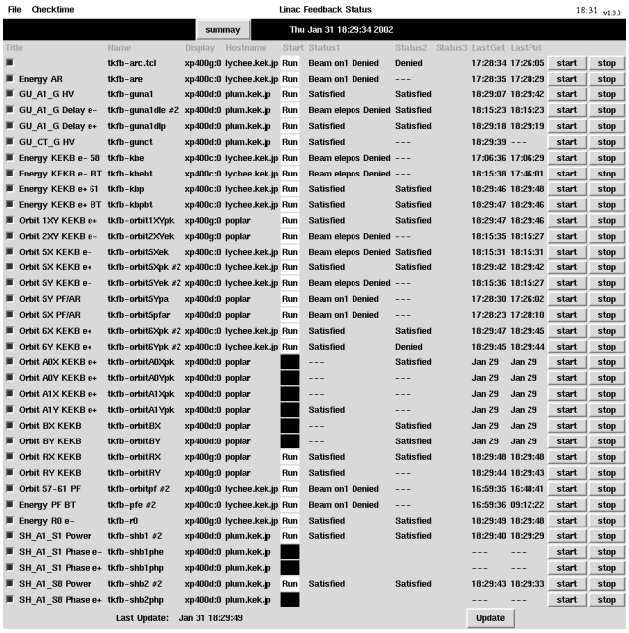

図 15: 動作しているフィードバックループの状態表示パネル。多数のエネルギーや軌道のフィードバックの動作状態を監視している。

rf 位相を逆方向に変更する操作をアクチュエータとして利用する。また、軌道フィードバックでは、1 ベータトロン波長内でのビーム位置の重み付き平均をモニタ値として使用する。

ビームのフィードバックはビーム位置モニタが 1Hz で読み出し可能なので、ソフトウェアの繰り返しはほぼ 1Hz で、振動を避けるためにゲインは低めで動作させている。

現在では、エネルギーフィードバックが 6 ヶ所、軌道フィードバックが 30 ヶ所、機器のフィードバックが 6 つ、常時使用されている。また、多数のフィードバックを管理するために、フィードバック状態表示やフィードバック記録のビューワなどのソフトウェアも用意されている。

6.6 ビーム運転ソフトウェアの効果

これらのソフトウェアはスクリプト言語で記述されているため、更新が容易で、しばしば運転中にも更新が行なわれる。また、他の運転用ソフトウェアと基本操作部分をライブラリルーチンとして共通化し、統一された操作環境が提供されている。

ビームモード切り替えソフトウェアは開発当初は毎日のように、また現在でも頻繁に改良が加えられ、信頼性が高まっている。この自動化によって、一日あたり約 50 回の切り替えも問題無く対応できるようになった。KEKB の蓄積ルミノシティに対して重要な切り替え時

間も通常 1 分程度になっている[23]。

また、フィードバックループを使用することによって、加速器の状態や場所などにもよるが、ビームの変動を長期間（6 時間程度）のものは 5 分の 1 程度に、短期間（1 分程度）のものは 約半分に減少させることができた。また、オペレータの操作を必要とせずにモード切り替え後も高品質なビームの維持が可能になった。また、加速器の異常を発見するための指標にもなっている。

これらのシステムによって、下流の加速器、特に KEKB Belle の実験効率に大きく寄与している。コミッショニングの詳細については、多数の報告がされているので、そちらを参照してほしい [53, 54, 2]。

7 タイミングシステム

7.1 入射器のタイミングシステム

KEK の電子陽電子入射器のタイミングシステムは、入射器内の電子銃、マイクロ波、及びビームモニタなどの 100 を超える機器に精度の高いタイミング信号を供給している。ここにおいても、入射器の他の部分と同様に、高い安定度を達成するためにさまざまな機構が導入されている [2]。特に、KEKB 入射のためには、以前の TRISTAN プロジェクトに比べると格段に精度の高いタイミング信号が必要となるため、全面的なシステムの再構築が行われた。リングの rf との同期精度の向上や、エネルギー増強のために入射器全体にわたって使用されたマイクロ波パルス圧縮器 (SLED) への信号供給に注意が払われている。

このようなタイミングシステムは、入射器内のさまざまな機器と関連を持って動作するが、主に制御グループが担当しているので、ここで解説をすることにする。

7.2 タイミングシステムの構成

入射器のタイミングシステムでは、ビームタイミング信号、マイクロ波発生用トリガ信号、ビームモニタ用トリガ信号、などが必要に応じて精度よく生成されている。それらは基本信号の発生と分配機構、15 ヶ所の副トリガステーションにおけるの遅延信号の生成機構などを通して加速器の各構成機器に供給されている。

7.2.1 基本クロック

以前の TRISTAN 蓄積リングへの入射には、300ps 程度のタイミング精度で十分であったため、入射器とリングの rf は非同期で、入射タイミングはそれらの間の 2 重同期回路で生成されていた。しかし、KEKB リングに入射されるビームはサブハーモニックバンチャ（SHB）を用いて、入射器の単バンチとして加速され、入射には 30ps 以下（リングの入射位相にして 5 度以下）の精度が求めらるので、利用される複数の rf 周波数は表 11、図 16 のように整数関係を持つことになった。

Purpose	Ratio	Frequency
Fundamental	-	10.38546 MHz
Linac SHB1	x11	114.24 MHz
Linac SHB2	x55	571.2 MHz
Linac Main	x275	2856 MHz
KEKB Ring	x49	508.8873 MHz

表 11: KEKB 入射器の基本周波数

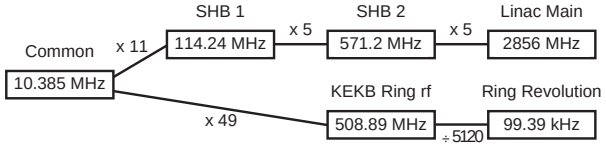

図 16: KEKB 用の各 rf 信号の関係。旋回周波数以外は入射器内で生成されている。

これらは入射器棟内に置かれた周波数分周逓倍器で生成され、マイクロ波ドライブシステムなどに分配されている [55]。タイミングシステムではこれらの周波数のうちあとで述べるように 114MHz と 571MHz をクロックとして利用しており、また、ビームタイミングは 10.39MHz から作られる。

これらの周波数は基本周波数（10MHz）にして 0.1Hz 単位で変更が可能で、KEKB リングの周長が日格差などで変動した際に軌道補正のソフトウェアによって変更が行われる。

PF や PF-AR の入射に際しては約 2ns 幅の複数バンチで加速しており、また条件が厳しくないので、リングの rf 周波数との同期は取っておらず、あとで述べるように旋回周波数とだけ同期を取っている。

これらの rf 周波数とは別に、運転用パルスモジュレータの繰り返し周波数（ビームの最大繰り返し周波数）である 50Hz は、電源等のノイズの影響を低減させるために、商用周波数に同期させて生成している。

7.2.2 ビームタイミング信号

ビームタイミングは 50Hz タイミングをリングの旋回周波数（PF は 1.6MHz、PF-AR は 0.8MHz）に同期させて作っている。ビーム繰り返しは最大で 50Hz である

[23] 主に電磁石の初期化の時間

が、分周して繰り返しを少くすることができる。さらに、必要に応じてリングのバケットを選択する遅延が追加される。

KEKB については上に述べたように、基本共通周波数 10.39MHz に同期した上で、5120 個のうちのひとつのバケットを選択するので、最大 0.5ms の遅延となる [56]。

7.2.3 クロック及びタイミング信号の分配

以前の TRISTAN プロジェクトにおいては、ビーム以外のパルスマイクロ波生成等のためのタイミング信号には 30ns 程度のジッターを持った非同期の遅延信号が主に使われていた。しかし、KEKB においては安定度の要求と SLED を使用した高電界加速のために精度の高いタイミング信号が必要になった。

そこで、入射器全体にクロック信号を分配し、クロックを計数することにより遅延信号を生成することにした。加速用マイクロ波のドライブラインには 2856MHz が使われているが、遅延信号を作るためには不都合なため、SHB2 に使用される 571MHz をクロックとして分配することにした。

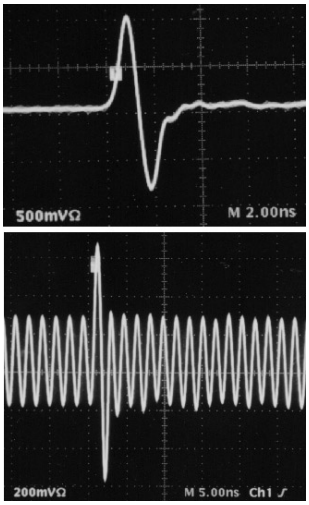

図 17: 50Hz パルス信号（上）、及び 571MHz クロックと 50Hz 信号が重畳された伝送信号（下）

50Hz の各パルスについてタイミンング信号が必要になるわけだが、571MHz クロックと 50Hz のタイミング信号を別々に伝送すると、信号伝播の遅れの差により、クロックのずれが懸念される。それを避けるため、クロックとタイミング信号を重畳させる機構を主トリガステーションに用意し、1 本の同軸ケーブルで伝送することにした（図 17）。その信号は同軸ケーブルから方向性結合器によって 1 次的な副トリガステーション 9 ヶ所、及び KEKB 入射用の電子銃ステーション (A1 電子銃) に導かれ、クロックとタイミング信号に再生される。さらに 2 次的な副トリガステーション 5 ヶ所にも導かれる。（表 12）この仕組みによって、遅延信号が各トリガステーションにおいて、ステップ 1.75ns、精度約 10ps で生成できることになる。

Station	Beam Station	1 次副 Station	2 次副 Station
場所	A1 電子銃	Sub-booster	副制御室
数	1	9	5
クロックの分離	TD4R	Trigger-Receiver	1 次副 Station より
遅延信号の発生	TD4R	TD4	TD4V
Field Bus	RS232C	CAMAC	VME
主な用途	ビーム	低レベル rf ビームモニタ	モジュレータ

表 12: タイミング信号の伝送と発生

7.2.4 遅延タイミング信号の発生

副トリガステーションでは、受け取った 50Hz タイミング信号を起点にして、571MHz クロックを計数することにより各機器に必要な遅延信号が生成される。従って、1.75ns を単位として遅延が選べることになる。遅延計数には Timing-Delay-4 (TD4) と呼ぶ ECL カウンタを内蔵したモジュールを使用しているが、各ステーションの都合により、VME、CAMAC、RS232C が制御接続に用いられ、それぞれ TD4V、TD4、TD4R と呼ばれている。カウンタは 16bit なので、最大 114 μ s の遅延を行うことができる。[24]

機器毎に必要なタイミングが異なるため、それぞれ別に TD4 を設置してあり、現在は合計約 150 台になっている。それらの TD4（及び loop3 遅延モジュール）は、それぞれに対応したドライバソフトウェアを通して、階層的な制御ソフトウェアで他の加速器機器と同様に統一的に管理され、運転ソフトウェアや加速器のオペレータからはハードウェアの違いを認識する必要はない [57, 47]。

7.2.5 パルスマイクロ波用タイミング信号

マイクロ波用タイミング信号としては、低レベルマイクロ波生成用のタイミングと、大電力クライストロンモジュレータの高圧パルスタイミングがある。

低レベルマイクロ波については、各 1 次副トリガステーションにおいて、パルスエンベロープと SLED の位相反転タイミングが作られ、サブブースタクライスト

[24] 一部の 2 次副トリガステーションでは、loop3 という入射器独自の通信規格に接続された非同期の遅延モジュールが使われているが、2002 年夏に TD4V に置き換わる。

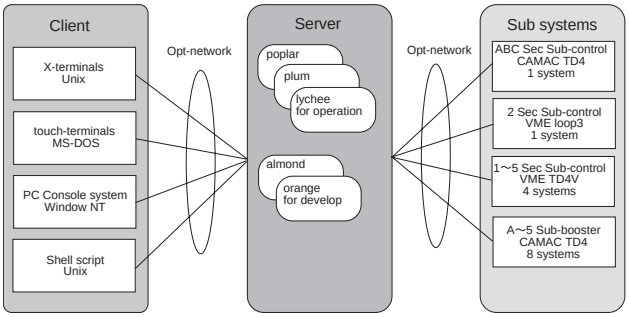

図 18: タイミング関連の制御ソフトウェアの構成、他の入射器の機器の制御と同様、複数の制御機器の違いをサーバソフトウェアで隠し、アプリケーションソフトウェアからは、信号遅延時間、待機モード選択、ビーム繰り返し、などについて均一な制御サービスを提供している。

ロンに供給される。そこで生成されたパルスマイクロ波がそのセクタ内の大電力クライストロンに送られることになる。この SLED の位相反転タイミングは、大電力マイクロ波の安定度に直接影響するが、上に述べたような機構により十分な精度を持って供給されている。

クライストロンモジュレータの高圧タイミングについては、各 2 次副トリガステーションにおいて、クライストロン毎に TD4V が用意され、個々に遅延を決めた上で信号がモジュレータられるようになっている。

大電力クライストロンは現在 59 台設置されているが、通常は数台が障害時の交換用として待機（スタンバイ）モードに置かれ、運転には使用しない。これらのスタンバイクライストロンもすぐに使用できるように、ビームとはずらしたタイミングで仮の運転状態にしておく必要がある。そのために、低レベルマイクロ波のエンベロープについては 57 μ s（TD4 の遅延レンジの半分）離れた 2 つのパルスを供給し、高圧タイミングでそのいずれかを選択する。その選択の組合せは、ビーム種別によって異なる。入射器のビーム運転モードは大きくわけて KEKB e^-、KEKB e^+、PF e^-、PF-AR e^-、の 4 つがあるが、それらのビームモードを変更したときにソフトウェアにより待機モードのクライストロンを切り替えている [52]。

7.2.6 ビームモニタ用タイミング信号

ビームモニタ用のタイミング信号も上と同様の仕組みを用いて、各 1 次副トリガステーションにおいて発生させている。しかし、ビームの繰り返しは 50Hz よりも低いこともあるので、50Hz タイミング信号の直前に遅いゲート信号を送って、ビームを区別できるようにしている。

ゲート信号は主トリガステーションで、ビーム繰り返し、またはそれより少ない 1Hz や 5Hz など 4 種類を生成して、対より線によって 1 次副トリガステーションに分配されている。副トリガステーション側の TD4 には、50Hz タイミング信号にそれぞれ必要なゲートをかけて、遅延信号が供給される。例えば、ビーム位置モニタのデータ収集用には観測モードによって、1Hz や 5Hz などの信号が配られる。

信号は、19 のビームモニタステーション、2 つのワイヤモニタステーション 4 つのストリークカメラステーションに送られている。これらのステーションからは、ビーム位置モニタ 90 台、ランダムシャッタカメラ 10 台、ワイヤスキャナ 14 台などへタイミング信号が接続されている。また、入射用のセプタム、キッカーのトリガもこの仕組みで用意されている。

7.3 システムの性能と今後

図 19 はストリークカメラで観測したビームのバンチ構造である。ビームの観測幅約 9ps はシミュレーションとよく一致する。このことは、タイミングシステムから供給されているビームタイミングとストリークカメラのタイミング、及び加速マイクロ波の間のジッタが 9ps よりも十分小さいということを意味しており、タイミングシステムの精度の高さを表している。

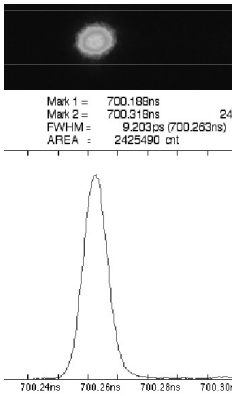

図 19: ストリークカメラによるビームバンチ構造の測定で 10nC ビームの幅が約 9ps に見えている

2001 年から、入射効率を増倍させるために、ひとつの rf パルス内で 2 つのビームバンチを加速する、いわゆる 2 バンチ加速がしばしば行われている [58]。このモードでは、ウェーク場の影響などを逃げながら、96ns 離れた 2 つのバンチを同等に加速するために、2 つのバンチのビーム特性をよく合致させる必要がある。入射器内には特に速いキッカーなどは設置されているわけではなく、SLED の位相反転タイミングなどを調整して、エネルギーアナライザやエミッタンスモニタなどで測定し

たビーム特性を合致させることになる。現在のところ、電子銃タイミング、バンチャ rf タイミング、そして各セクタの SLED 位相反転タイミングを調整することにより、2 バンチとも効率よく入射できることがわかっている [59]。

このようにタイミングシステムは順調に動作しているが、いくつか障害もあった。まず、メーカーから提供を受けた CAMAC ドライバソフトウェアの不具合が解消せず、制御が不能になることがあった。これに対しては、上位のソフトウェアを工夫することにより障害を避けることに成功している [57]。具体的には、一般に入射器内の機器の制御サーバソフトウェアでは冗長な複数のサーバが同等な働きをしているが、タイミング関連のサーバについては、CAMAC 部分についてだけ独立な 1 つのサーバに仕事を集中させることにしている。これによって、障害はなくなっているが、CAMAC ドライバソフトウェアの不具合が解消されれば、元の対称性のよい構成に戻したいと考えている。

また、TD4/TD4V のモジュール個体によって、約 2 週間に 1 回約 200ms の間出力が停止することが見つかった。KEKB のコミッショニング当初はこのことに気がつかず、また、頻度が低いためなかなか理解が進まなかったが、モジュール内の 2 ヶ所のコンパレータの不具合であることを突き止められ、2002 年夏には全て交換が完了する予定となっている。

システムの安定度が高まって来たので、今後をそれを維持するために監視システムの充実を検討している。オシロスコープによる波形とタイミングの監視、及び時間ディジタル変換器 (TDC) による各タイミングの監視の組み合わせとなる。ハードウェアは既に一部が設置されていて人手による監視は行っているが、常時監視を行うためにソフトウェアの開発を進めているところである。

第 8 節に書くような今後の改造についても検討しているが、現在のところ大きな変更は必要ないものと思っている。

8 まとめと今後

入射器の制御システムはこれまで述べてきたように、入射器の運転に欠くことのできないものになっている。しかし、加速器自体の要求仕様も変わってきており、最近では連続入射や 2 バンチ入射のための対応が行われている [59]。また、1993 年の制御システムの更新、1997 年からの KEKB 入射のためのコミッショニングを経て、制御システムも機能拡張が必要な時期に来ている。

8.1 ビームオプティクス

現在の入射器においてもまだまだ解決すべき問題は多く、特に大電流であることによるウェーク場の効果もあって、ビームオプティクスはなかなかモデルと一致しない部分も多い。しかし、加速勾配や磁場強度に使われる較正係数などの誤差の積み重ねも影響している可能性も否定できない。

これらの誤差を押さえ込むために、入射器の部分ごとにビームを使った評価を今まで以上に進める必要があると思われる。

8.2 他の制御システムとの協調

8.2.1 EPICS 環境の利用

さまざまな加速器制御のための機構が実装されてきたが、特に蓄積情報などの高速処理についてはまだまだ要求が多い。このような分野については、他の加速器においてもいくつかの実装が進んでおり特に EPICS のコミュニティにおいて協力して環境を作ろうとする動きがある [60, 61]。

そのような状況をふまえて、EPICS について考えると、入射器の制御システムも徐々に EPICS の利用を増やしていく必要がある。それを補強する要因は他にもある。

- 入射効率などの改善を目指して、現在以上に KEKB との間で密に情報を交換する必要が高くなる可能性がある。
- すでに Channel Access Server によって入射器の重要なパラメータ 2000 あまりが EPICS 環境にも提供されている [15, 62]。
- 入射器の制御でキャッシュ情報などの非同期の情報交換の仕組みが強化され、Channel Access Server の構築がより容易になっている。
- さらに多くのパラメータを EPICS 環境に公開すれば、アプリケーションソフトウェアからは入射器の制御システムは EPICS に見えるようになる。
- すでに入射器内でもワイヤスキャナは EPICS によって運用されている。
- 入射器に既存のコントローラの EPICS ドライバが他のプロジェクトのために開発されている [63]。

このように制御システムの最上位のアプリケーションソフトウェアと下位層の装置コントローラのソフトウェアは部分的に EPICS に移行することが可能となっている。これを進めることで、EPICS コミュニティで開発されたアプリケーションソフトウェアがそのまま利用でき

る。当面、このような仕組みによって、蓄積情報の処理の高速化を期待している。

入射器の場合は KEKB リング（と PF-AR リング）が EPICS を採用しているために EPICS の方向性を当面目指す、という意味合いが強いが、第 2 節でも述べたように、今後は技術や成果の共有が重要になるということを念頭におく必要がある。

8.2.2 CORBA の利用

さらにその後の制御システムの協力体制がどのようになるかわからないが、下位層は EPICS でも十分であったとしても、少くとも最上位については、次のような項目の検討が必要である。

- より誤りの少いソフトウェア開発を進めるために、オブジェクト指向のプログラミングなどの支援が受けやすい環境。
- 制御システム以外との密な情報交換のために、より一般的な情報技術が利用できる環境。
- 計算機プラットフォームなどに依存しない Java などの環境。

これらを考えて、将来の加速器の制御システムの上位層には、CORBA の利用が進むと考えられる。実際いくつかの加速器において CORBA の利用が検討されており、入射器においてもウェブブラウザ経由の情報提供に利用されている [16]。これが第 2 節に書いたような Fad に過ぎないかどうかはまだわからないが、少くとも現在のところ注目すべき技術であることは確かである。

8.3 SuperKEKB に向けて

KEK の電子入射器の将来計画として期待されているのが、現在の KEKB のルミノシティを 10 倍引き上げる、SuperKEKB 計画である。その計画では現在以上に入射器の役割が重要になる。制御システムとしても新しい装置に対応するなどの KEKB の増強時に経験したものと同様の機能拡張の他に、いくつか検討しておくことがある。

8.3.1 高速同期処理

現在の入射器では、主にビームポジションモニタ (BPM) の読み出しシステムの制限から、ビームフィードバックなどは一秒に一回しか動作できない [59]。しかし、試験的なビームの安定度の評価からは、50Hz まで重要かどうかはわからないものの、10Hz 付近に変動要因があることが見つかっている。入射器の安定度を高めるためには、より長期の変動への対策とともに、このような高速のビームフィードバックが重要になると思われる。もしダンピングリングが導入された場合には、大電流のビームの入射、出射のために、50Hz のフィードバック動作も必要になると思われる。

このような高速のエネルギー、軌道のフィードバックシステムは、上流から下流に向かって干渉するので、その相互作用を正しく評価しなくてはならない。そのため、入射器全体のビームフィードバックシステムをビーム繰り返しの 50Hz で同期して動作させる必要がある [64]。

このためには、50Hz で動作する低レベル rf システム、キッカー、データ収集システムなどを用意する必要がある。タイミングシステムは現在のものでもおそらく対応できると思われる。

そのようなシステムは SuperKEKB だけでなく、現在の入射器にも有効であると思われるので、その試験も始まっている。可能であれば、軌道やエネルギーだけでなく、エミッタンスやエネルギー幅の安定化も最終的な実験の効率には寄与が大きいと思われ、検討しているところである。

8.3.2 間欠ビーム測定

現在は、項目によって一日から一週間に一度、入射の無い時間帯に、rf のフェージングや、ビームオプティクスの再マッチングなどといった調整作業を行って、入射器の長期安定性を維持している。しかし、SuperKEKB においては連続入射が行われるため、このような作業の時間が確保できなくなる。

そこで現在検討されているのは、ビームパルスのうち一部だけ、例えば、1 秒に 1 パルスだけを選び測定や調整に使用する方法である。そのように選ばれたビームパルスはリングに入射されないように、ビームトランスポートの最後でビームダンプにけり出す必要がある。パルスを区別して加速器を多重に運転することになるので、パルスモジュレーションとか仮想加速器と呼ぶこともできる。

このような測定のためにも前項で上げた高速同期処理が使われ、また、調整の対象となる装置は 50Hz で動作する必要がある。連続入射が現在の KEKB でも有効なので、このような間欠測定システムも早くから導入できた方が好ましい。

8.4 終わりに

加速器の制御システムは、加速器内の装置、ビーム物理、ビーム運転など、全ての要素と関わりを持っているために、今後高度になる加速器ではさらに重要性が高ま

ると思われる。また最近、SASE FEL、ERL、リニアコライダ、大電流加速器、医療用加速器など、線形加速器が話題になることが多い。そういう意味で線形加速器の制御システムについては、まだまだ解決しなくてはならないことが多く現れると思われる。加速器の要求仕様を見失わず、ひとつひとつ問題を解決できるような制御システムを構築して、要求に答えていきたいと考えている。

なお、KEK の電子入射器に関連するレポートは一部ではあるがウェブに集めるようにしている。参考にされたい [65]。

参考文献

[1] A. Enomoto, "Upgrade to the 8-GeV Electron Linac for KEKB", *Proc. LINAC96*, Geneva, Switzerland, 1996, p.633.

[2] K. Furukawa *et al.*, "Towards Reliable Acceleration of High-Energy and High-Intensity Electron Beams", *Proc. LINAC2000*, Monterey, USA., 2000, p.630.

[3] T. Suwada *et al.*, "Present Status and Beam-Stability Issues of the KEKB Injector Linac", *Proc. PAC2001*, Chicago, USA., 2001, p.4083.

[4] K. Nakahara *et al.*, "Control System for the Photon Factory 2.5-GeV Electron Linac", Nucl. Instrum. Meth. A **251**(1986)327.

[5] R. Humphrey, "Lessons from the SLC for Future LC Control Systems", *Proc. ICALEPCS91*, Tsukuba, Japan, 1991, p.14.

[6] *Proc. ICALEPCS91*, Tsukuba, Japan, 1991. *Proc. ICALEPCS93*, Berlin, Germany, 1993.

[7] K. Furukawa *et al.*, "Recent Progress in the Control System for KEK 2.5-GeV e^-/e^+ Linac", Nucl. Instrum. Meth. A **293**(1990)16.

[8] K. Furukawa *et al.*, "Upgrade Plan for the Control System of the KEK e^-/e^+ Linac", *Proc. ICALEPCS91*, Tsukuba, 1991, p.89.

[9] N. Kamikubota *et al.*, "New Control System with VME and Workstations for the KEK e^-/e^+ Linac", Nucl. Instrum. Meth. A **352**(1994)131.

[10] L. Dalesio *et al.*, "The Experimental Physics and Industrial Control System Architecture: Past, Present, and Future", Nucl. Instrum. Meth. A **352**(1994)179.

[11] T. Katoh *et al.*, "Present status of the KEKB control system", *Proc. ICALEPCS97*, Beijing, China, 1997, p.15.

[12] J. Chen *et al.*, "CDEV: An Object-Oriented Class Library for Developing Device Control Applications", *Proc. ICALEPCS95*, Chicago, 1995, p.97.

[13] P. Duval, "The Use of PCs in Controlling DESY Accelerators", *Proc. ICALEPCS97*, Beijing, China, 1997, p.162.

[14] <URL:http://acc-physics.kek.jp/SAD/sad.html>

[15] K. Furukawa *et al.*, "Integration Feasibility of the Existing Linac Control System and Ring EPICS System at KEKB", *Proc. ICALEPCS95*, Chicago, USA., 1995, p.863.

[16] N. Kamikubota *et al.*, "Development of a CORBA Toolkit and its Evaluation", *Proc. ICALEPCS97*, Beijing, China, 1997, p.351.

[17] K. Furukawa *et al.*, "Microwave Control and Measurement System at the KEKB Linac", *Proc. ICALEPCS97*, Beijing, China, 1997, p.146.

[18] N. Kamikubota *et al.*, "Introduction of Modern Subsystems at the KEK Injector-Linac", *Proc. ICALEPCS2001*, San Jose, USA., 2001, p.328.

[19] I. Abe *et al.*, "MMI Object Analysis and the Distributed Components for a New Console in the KEK e-/e+ Linac", *Proc. ICALEPCS97*, Beijing, China, 1997, p.519.

[20] K. Nakahara, "Control System for the KEK Electron Linac", *OHO'85*, Tsukuba, 1985.

[21] N. Kamikubota *et al.*, "New Control System for the KEK Linac", *Proc. Linac Meeting in Japan*, Tsukuba, 1993, p351.

[22] N. Kamikubota *et al.*, "New Control System for the KEK Linac", *Proc. LINAC94*, Tsukuba, Japan, 1994, p.822.

[23] N. Kamikubota *et al.*, "Improvements to Realize a Higher Reliability of the KEK Linac Control System", *Proc. ICALEPCS95*, Chicago, USA., 1995, p.1052.

[24] N. Kamikubota *et al.*, "Techniques to Improve Reliability of the KEK-Linac Control System" *Proc. Linac Meeting in Japan*, Osaka, 1995, p209.

[25] N. Kamikubota *et al.*, "Evolution of the KEK Linac Control System by Introducing New Subsystems" *Proc. Linac Meeting in Japan*, Tsukuba, 2001, p273.

[26] K. Nakahara *et al.*, "An Operator Console System of the Photon Factory Injector Linac", Nucl. Instrum. Meth. A **293**(1990)446.

[27] I. Abe *et al.*, "PC-based Control System using ActiveX in the KEK e-/e+ Linac", *Proc. PCaPAC99*, Tsukuba, 1999, KEK-Proceedings 98-10.

[28] N. Kamikubota *et al.*, "Introducing PCs to Unix-based control systems", *Proc. PCaPAC2000*, Hamburg, 2000.

[29] M. Tanaka *et al.*, "Database system in the KEK Linac PC-based Control", *Proc. PCaPAC99*, Tsukuba, 1999, KEK-Proceedings 98-10.

[30] N. Kamikubota *et al.*, "PC as a touch-terminal controller", *Proc. PCaPAC99*, Tsukuba, 1999, KEK-Proceedings 98-10.

[31] A. Shirakawa *et al.*, "Renewal of Magnet Controller for e+/e- Linac", *Proc. Engineering and Technology in Basic Research*, Tsukuba, 1999, KEK-Proceedings 99-16.

[32] A. Shirakawa *et al.*, "Construction of Device Management Program for Vacuum Control System", *Proc. Linac Meeting in Japan*, Sendai, 1997, p213.

[33] N. Kamikubota *et al.*, "Data Acquisition of Beam-Position Monitors for the KEKB Injector-Linac", *Proc. ICALEPCS99*, Trieste, Italy, 1999, p.217.

[34] T. Suwada *et al.*, "Stripline-type Beam-position-monitor System for Single-bunch electron/positron Beams", Nucl. Instrum. Meth. A **440**(2000)307.

[35] H. Katagiri *et al.*, "RF Monitoring System in the Injector Linac", *Proc. ICALEPCS99*, Trieste, Italy, 1999, p.69.

[36] N. Kamikubota *et al.*, "Tool for Device Histories at the KEK Linac", *Proc. LINAC96*, Geneva, 1996, p.800.

[37] N. Kamikubota *et al.*, "Device Histories at the KEK Injector-Linac", *Proc. Linac Meeting in Japan*, Sendai, 1997, p204.

[38] N. Kamikubota *et al.*, "Presentation of Klystron History and Statistics by World-Wide-Web", *Proc. Linac Meeting in Japan*, Himeji, 2000, p252.

[39] N. Kamikubota *et al.*, "Accelerator Archive Databases as Distributed CORBA Objects", *Japan Physical Society Annual Meeting*, Okinawa, 2001.

[40] S. Kusano *et al.*, "Study of Sharable Applications Using Java and CORBA", *Proc. ICALEPCS99*, Trieste, 1999, p.535.

[41] N. Kamikubota *et al.*, "Network Communication Libraries for the Next Control System of the KEK e-/e+ Linac", *Proc. ICALEPCS91*, Tsukuba, 1991, p.318.

[42] N. Kamikubota *et al.*, "Software module for Network Servers", KEK-Linac Internal Report PFINJ-MC-32, 1992.

[43] N. Kamikubota *et al.*, "Control Transactions of the KEK Injector-linac Control System and the KEKB Commissioning", *Proc. Linac Meeting in Japan*, Sapporo, 1999, p119.

[44] N. Kamikubota *et al.*, "Growth of Control Transactions of the KEK Linac during the KEKB Commissioning", *Proc. APAC'02*, Beijing, 2001.

[45] K. Furukawa *et al.*, "Improvement of the KEK Linac Control System towards KEKB", *Proc. Linac Meeting in Japan*, Tokyo, 1996, p210.

[46] T. Obata *et al.*, "Reliable Controls with Diskless VME Computers at KEK Linac", *Proc. Linac Meeting in Japan*, Himeji, 2000, p.255.

[47] K. Furukawa *et al.*, "Accelerator Controls in KEKB Linac Commissioning", *Proc. ICALEPCS99*, Trieste, Italy, 1999, p.98.

[48] <URL:http://www.tcl.tk/>

[49] D.C. Carey *et al.*, "Third-Order TRANSPORT with MAD Input, A Computer Program for Designing Charged Particle Beam Transport Systems", FERMILAB-Pub-98/310, 1998.

[50] K. Furukawa *et al.*, "Control System for a Bunch Profile Monitor at the KEK e+/e- Linac", *Proc. LINAC94*, Tsukuba, Japan, 1994, p.819.

[51] K. Furukawa *et al.*, "Energy Feedback Systems at the KEKB Injector Linac", *Proc. ICALEPCS99*, Trieste, Italy, 1999, p.248.

[52] K. Furukawa *et al.*, "Beam Switching and Beam Feedback Systems at KEKB Linac", *Proc. LINAC2000*, Monterey, USA., 2000, p.633.

[53] A. Enomoto, *et al.*, "Commissioning of the KEKB 8-GeV e- / 3.5-GeV e+ Injector Linac", *Proc. PAC99*, Stockholm, Sweden, 1998, p.713.

[54] Y. Ogawa and Linac commissioning group, "Commissioning Status of the KEKB Linac", *Proc. PAC99*, New York, USA., 1999, p.2984.

[55] H. Hanaki *et al.*, "Low-Power rf Systems for the KEKB Injector Linac", *Proc. APAC98*, Tsukuba, 1998, p.139.

[56] E. Kikutani *et al.*, "The KEKB Bucket Selection System - Recent Progress and the Plan in the Near Future", *Proc. APAC2001*, Beijing, China, 2001, p.669.

[57] S. Kusano *et al.*, "Timing System Software for the KEK Injector Linac", to be published in *Proc. Linac Meeting in Japan*, Kyoto, 2002.

[58] Y. Ogawa *et al.*, "Two-Bunch Operation of the KEKB Linac for Doubling the Positron Injection Rate to the KEKB Ring", *Proc. APAC2001*, Beijing, China, 2001, p.112.

[59] K. Furukawa *et al.*, "Beam Feedback Systems And BPM Read-Out System for the Two-Bunch Acceleration at the KEKB Linac", *Proc. ICALEPCS2001*, San Jose, USA., 2001, p.266.

[60] R. Müller *et al.*, "Signal Archiving and Retrieval: Essential Long Term Performance Tuning Tool", *Proc. ICALEPCS2001*, San Jose, USA., 2001, p.662.

[61] K.U. Kasemir *et al.*, "Overview of the Experimental Physics and Industrial Control System Channel Archiver", *Proc. ICALEPCS2001*, San Jose, USA., 2001, p.526.

[62] M. Kaji and K. Furukawa, "Operation of KEKB Linac and Ring with EPICS", *Proc. Linac Meeting in Japan*, Tokyo, 1996, p207.

[63] K. Furukawa *et al.*, "Implementation of the EPICS Device Support for Network-Based Controllers", *Proc. ICALEPCS2001*, San Jose, USA., 2001, p.197.

[64] L. Hendrickson *et al.*, "Beam-Based Feedback Simulations for the NLC Linac", *Proc. LINAC2000*, Monterey, USA., 2000, p.74.

[65] <URL:http://www-linac.kek.jp/linac/>

目 次

1 はじめに **1**

2 加速器の制御 **1**

2.1 加速器の制御の歩み 1

2.2 制御システムの目的と構成 2

3 KEK 電子入射器の制御の概要 **3**

3.1 入射器の制御の設計 3

3.2 入射器の制御の全体構成 4

3.3 装置コントローラ 4

3.4 中央制御計算機とネットワーク 4

4 ハード ウェア構成 **5**

4.1 歴史的経緯 5

4.2 設計方針と全体構成 5

4.3 構成要素 5

4.4 制御ネットワーク 7

4.5 履歴（Archive）システム 8

5 ソフト ウェア構成 **9**

5.1 歴史的経緯 9

5.2 設計方針と全体構成 9

5.3 制御ソフトウェアの詳細 9

5.4 Example(D-out) 12

5.5 制御メッセージの速度 14

5.6 運転中の機器サーバの負荷 14

6 運転ソフト ウェアとビーム制御 **14**

6.1 アプリケーションソフトウェア 14

6.2 情報の交換 15

6.3 オペレータインタフェース 15

6.4 運転用アプリケーション 16

6.5 KEKB 入射器の安定化 16

6.6 ビーム運転ソフトウェアの効果 18

7 タイミングシステム **19**

7.1 入射器のタイミングシステム 19

7.2 タイミングシステムの構成 19

7.3 システムの性能と今後 21

8 まとめと今後 **22**

8.1 ビームオプティクス 22

8.2 他の制御システムとの協調 22

8.3 SuperKEKB に向けて 23

8.4 終わりに 23

参考文献 **24**