cosylab

EPICS講習会 実習資料

Revision:	1.0
Status:	RELEASE
Repository:	https://internal.cosylab.com/svn/acc/projects/KEK
Project:	JCSL-KEK-EPICS-LEC-2013
Folder:	EPICSLectures2013/trunk/exercise
File:	DOC-KEK-EPICSCourse-exercises.odt
Owner:	Takashi Nakamoto
Last modification:	2013 年 6 月 24 日
Created:	2013 年 6 月 14 日

履歴

Revision	Date	Author	Section	Modification
0.1	2013-06-14	tnakamoto	1, 2	「実習 1」「実習 2」を作成
0.2	2013-06-21	tnakamoto	3, 4, 5, 6	「実習 3」～「実習 5」、「実習環境の解説」を作成
1.0	2013-06-24	tnakamoto	All	全体を見直し

公開範囲

本文書は、公開文書です。本文書の全体または一部は、「対象者」に挙げられている方であれば、どなたでも閲覧することができます。

目的

本文書は、KEK における EPICS 講習会の実習のための資料です。

対象者

KEK における EPICS 講習会の参加者

参考文献

[1] EPICS: Input/Output Controller Application Developer's Guide: http://www.aps.anl.gov/epics/base/R3-14/12-docs/AppDevGuide/

[2] EPICS Record Reference: http://www.aps.anl.gov/epics/wiki/index.php/RRM_3-14

[3] SNL Manual: http://www-csr.bessy.de/control/SoftDist/sequencer/Manual.pdf

[4] EPICS Quick Reference

凡例

本文書中において、以下のように”abco4$”で始まる行は、aboc4 上で実行する bash のコマンドを表しています。

```
abco4$ command
```

以下のように”abcob03$”で始まる行は、abcob03 上で実行する bash のコマンドを表しています。

```
abcob03$ command
```

bash コマンド中の”~”は自身のホームディレクトリを表しています。

以下のように、”>”で始まる行は、iocsh のコマンドを表しています。

```
> command
```

Table of Contents

1 実習1: コマンドによるPVへのアクセス

1.1 目的

本実習では、すでに用意されている IOC 上の PV に caget, caput, camonitor コマンドでアクセスします。この IOC は定電流電源を制御するためのものです(実際には電源をシミュレーションしています)。この電源の操作方法は非常に単純であり、電源の ON/OFF と電流値の設定を行うだけで動作します。本実習では、コマンドを使用してこの電源を操作することで、どのようにしてコマンドを用いて PV 値の get、put、monitor を行うことができるのかを学習します。

1.2 演習

以下のコマンドを abco4 上で実行し、電源の現在の ON/OFF の状況を確認してください。ただし、*user* は自身のユーザー名に置き換えてください。

```
abco4$ caget ET_user:PS1:ON_OFF_CMD
```

次に、この電源を ON してください。

```
abco4$ caput ET_user:PS1:ON_OFF_CMD ON
```

次に、電流値を 10mA に設定してください。

```
abco4$ caput ET_user:PS1:CURRENT_SP 10
```

電流値を設定した後に、実際に現在の電流値がどのようになっているのかを確認してください。

```
abco4$ caget ET_user:PS1:CURRENT_RB
```

電流値がどのように変化しているのかを監視してください。

```
abco4$ camonitor ET_user:PS1:CURRENT_RB
```

別の端末で、電流値を 20mA に設定してください。

```
abco4$ caput ET_user:PS1:CURRENT_SP 20
```

このとき、camonitor による値がどのように変化するのかを確認してください。

最後に、camonitor コマンドを Ctrl+C により終了させてください。

2 実習2: MEDMによる操作画面の作成

2.1 目的

本実習では、実習 1 で使用した電源の IOC を操作するための画面を作成します。実習 1 でコマンドで行ったことを、GUI により簡単にかつ直観的に行えるようにします。本実習で使用するウィジェットは以下のとおりです。

- Text
- Choice Button
- Slider
- Meter
- Strip Chart

なお、PV 名の中のユーザー名とデバイス名はハードコーディングせず、マクロを使用してください。例えば、

ET_user:PS1:ON_OFF_CMD

という PV にアクセスするのであれば、MEDM で PV を入力するところで、

ET_$(USER):$(NAME):ON_OFF_CMD

と入力し、$(USER)と$(NAME)はマクロで展開するようにしてください。マクロを展開するには、MEDM で実行モードで起動する際、マクロの値を指定することで

2.2 演習

ホームディレクトリに実習用のディレクトリを作成してください。

```
abco4$ mkdir -p ~/lecture/medm
```

作成したディレクトリに移動してください。

```
abco4$ cd ~/lecture/medm
```

MEDM を以下のコマンドで起動してください。

```
abco4$ medm
```

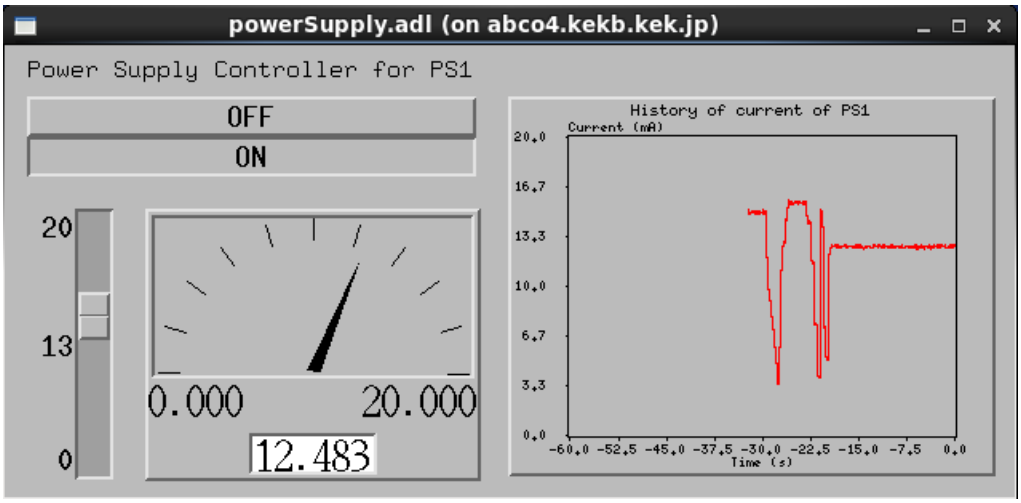
MEDM のメニューから File → New を選択し、新しい操作画面を作成してください。次に、Palettes → Object を選択し、適当なウィジェットを操作画面に配置してください。作成した操作画面は、powerSupply.adl というファイル名で保存してください。画面を作成し終えたら、一度、MEDM を終了してください。

次に、操作画面を実行します。以下のコマンドを実行してください。ただし、*user* は自身のユーザー名に置き換えてください。

```
abco4$ medm -x -macro “USER=user,NAME=PS1” powerSupply.adl
```

この方法により、MEDM は実行モードで実行され、$(USER)というマクロがユーザー名、$(NAME)というマクロが”PS1”という文字列に置き換えられます。

2.3 操作画面例



2.4 注意

本実習で作成した操作画面は、実習 4 でも使用します。

3 実習3: EPICS IOCの作成

3.1 目的

本実習では、どのように EPICS IOC を作成するのかを学びます。EPICS base に元々用意されているサンプルアプリケーションを元に EPICS IOC を作成します。具体的に、下記の事項を習得します。

- EPICS IOC の作成、ビルド、実行
- iocsh の使用方法
- アプリケーションを生成するのに使用する makeBaseApp.pl という Perl スクリプトの使用方法

3.2 演習

ここでは、myexampleApp という名前の EPICS アプリケーションと、iocmyexample という名前の IOC を作成します。

3.2.1 EPICS_HOST_ARCHの確認

以下のコマンドを実行し、EPICS_HOST_ARCH 環境変数に何が設定されているのかを確認してください。

```
abco4$ echo $EPICS_HOST_ARCH
```

ログインしているマシンの OS と CPU アーキテクチャ(例: linux-x86_64, win32-x86)が表示されるはずです。もし何も表示されない場合には、EPICS_HOST_ARCH を適切な値に設定してください。現在の OS と CPU アーキテクチャを確認するには、以下のコマンドを実行してください。

```
abco4$ /proj/epics/R314/R314123-CSA/base/startup/EpicsHostArch.pl
```

実行した結果、例えば”linux-x86_64”と表示された場合には、以下のようにして EPICS_HOST_ARCH 環境変数を設定してください。

```
abco4$ export EPICS_HOST_ARCH=linux-x86_64
```

3.2.2 サンプルアプリケーションの作成

以下のコマンドによりサンプルアプリケーションを~/lecture/myexample 以下に作成することができます。

```
abco4$ mkdir -p ~/lecture/myexample

abco4$ cd ~/lecture/myexample

abco4$ makeBaseApp.pl -t example myexample

abco4$ makeBaseApp.pl -i -t example myexample
```

最後のコマンドを実行すると、この IOC はどのアプリケーションをブートするのかを聞かれます。今回は、1 つしかアプリケーションがありませんので、そのまま Enter キーを押すだけです。makeBaseApp.pl の詳細な使用方法については[1]の 17 ページを参照してください。

3.2.3 作成されたファイルの確認

この段階で、~/lecture/myexample 以下にどのようなディレクトリやファイルが生成されたのかを確認し、自身のノートにメモをしてください。例えば、find コマンドを ~/lecture/myexample ディレクトリで実行すると、~/lecture/myexample ディレクトリ以下にあるディレクトリとファイルを確認することができます。

```
abco4$ cd ~/lecture/myexample
abco4$ find
```

この作業は、ビルドを行う前に行ってください。これにより、ビルドをするのに必要なファイルがどういったものであるかを確認することができます。

3.2.4 レコード名の変更

EPICS base に用意されているサンプルアプリケーションに含まれる PV の接頭辞は *user*Host となっています。これを KEKB 制御ネットワークにおける PV の命名規則に従い、ET_*user* という名前に置き換えます。以下の 2 つのファイルをエディタで開き、*user*Host という文字列を ET_*user* に変更してください。ただし、*user* は自身のユーザー名です。

- ~/lecture/myexample/myexampleApp/Db/userHost.substitutions
- ~/lecture/myexample/iocBoot/iocmyexample/st.cmd

3.2.5 アプリケーションのビルド

~/lecture/myexample ディレクトリにて、以下のコマンドを実行してください。

```
abco4$ make
```

3.2.6 ビルドにより生成されたファイルの確認

ここで、もう一度~/lecture/myexample 以下にどのようなディレクトリやファイルが生成されたのかを確認し、make 前と make 後でどのような違いがあるのかを確認してください。

3.2.7 IOCの起動

以下のコマンドを **abcob03 上で**実行し、IOC を起動してください。

```
abcob03$ cd ~/lecture/myexample/iocBoot/iocmyexample

abcob03$ ../../bin/linux-x86_64/myexample st.cmd
```

3.2.8 iocshコマンドの実行

IOC が起動したら、iocsh 上でいくつかのコマンド（例: dbl, dbpr, dbpf, dbgf 等)を試してみてください。簡単な使用方法であれば [4] に記述されています。

また、help の機能が付いているので、

```
> help
```

または、

```
> help <cmd>
```

でヘルプを確認してください。ここで、<cmd>は dbl などといったコマンド名です。help コマンドではワイルドカードを使えるので、

```
> help db*
```

とすれば、”db”で始まるコマンドの使用方法を見ることができます。

3.2.9 リモート端末からのPVの操作

実習 1 を参考に、abco4 から caget, caput, camonitor により PV の操作をしてみてください。

4 実習4: 電源のシミュレーション

4.1 目的

本実習では、電源を模擬する EPICS データベースを VisualDCT で作成します。この EPICS データベースで、以下の機能を実装してください。

- ON/OFF スイッチ: 1 で ON、0 で OFF とします。
- 電流のセットポイント値(設定値)の書き込み: 単位は mA とします。
- 電流のリードバック値(現在値)の読み出し: 電源が ON である場合に、セットポイント値を中心に値がランダムの上下するようにします。単位は mA とします。

操作画面には MEDM を使用してください。基本的には、実習 2 で作成したものがほぼそのまま使えるはずです。

なお、EPICS データベースを実装する際には、下記の点に注意してください。

- 適切なレコードタイプを使用してください。この実習では、ai、ao、bi、bo、calc(または calcout)を使うことになると思いますが、他にも便利そうなレコードがあれば使用しても構いません。
- レコード名について
  - レコード名は ET_*user*: (user は自身のユーザー名)で始まるようにしてください。
    これは、KEKB 制御ネットワークにおいて PV 名が重複しないようにするためのルールです。この接頭辞は、EPICS データベースでは ET_$(USER)などとしておき、$(USER)はマクロとして置き換えるようにするとよいでしょう。
  - レコード名には、ET_*user*:に続いて、電源の名前（例: PS1)を付してください。このとき、電源の名前にはマクロを使用し、ET_$(USER):$(NAME)などとしておくとよいでしょう。
  - マクロの展開は、st.cmd ファイル中の dbLoadRecrods コマンドで行ってください。マクロの展開により、同じ機能を持つ電源を異なる名前(例: PS1、PS2、PS3、...)で制御できるようにしてください。
  - レコード名は、命名規則に従ったものにしてください。自身が所属する加速器やグループの命名規則が分からない場合には、自身で命名規則を策定してください。例えば、電流を表す PV 名は”CURRENT”と命名することとし、セットポイント値には末尾に”_SP”、リードバック値には末尾に”_RB”と付けるようにしてみてください。

- 必要に応じて、以下に示すフィールドを設定するのを忘れないでください。各フィールドの詳細は[2]を参照してください。
  - SCAN
  - DESC
  - HOPR
  - LOPR
  - DRVH
  - DRVL
  - PREC
  - EGU
  - ONAM
  - ZNAM
  - …
- st.cmd を使用して abcob03 上で IOC を起動してください。IOC を起動したら、abco4 上で MEDM の操作画面により電源の操作が正しく行えることを確認してください。
  - 開発途中の段階では、caget や camonitor を使ってデバッグをしてみることも検討してください。

4.2 演習

4.2.1 空のEPICS IOCの作成

~/lecture/powerSupply ディレクトリに空の EPICS IOC を作成するところからはじめます。アプリケーション名は powerSupplyApp、IOC 名は iocpowerSupply とします。以下のコマンドを実行してください。

```
abco4$ mkdir -p ~/lecture/powerSupply

abco4$ cd ~/lecture/powerSupply

abco4$ makeBaseApp.pl -t ioc powerSupply

abco4$ makeBaseApp.pl -i -t ioc powerSupply
```

これらのコマンドを実行することにより、空の EPICS IOC が作成されます。

4.2.2 電源をシミュレートするEPICSデータベースの作成

~/lecturepowerSupply/powerSupplyApp/Db というディレクトリに powerSupply.vdb というファイルを VisualDCT で作成してください。

EPICS データベースを作成し終えたら、~/lecture/powerSupply/powerSupplyApp/Db にある Makefile をエディタで開き、”#DB += xxx.db “と記述されている行の次の行に、以下の命令を挿入してください。

```
DB += powerSupply.vdb
```

また、~/lecture/powerSupply/iocBoot/iocpowerSupply/st.cmd に dbLoadRecords ではじまる行を以下のように置換してください。ただし、*user* は自身のユーザー名としてください。

```
dbLoadRecords(“../../db/powerSupply.vdb”,”USER=user,NAME=PS1”)
```

4.2.3 ビルドと実行

以下のコマンドを実行してください。これにより、ビルドがなされます。

```
abco4$ cd ~/lecture/powerSupply

abco4$ make
```

4.2.4 IOCの起動

abcob03 上で、作成した IOC を起動してください。

```
abcob03$ cd ~/lecture/powerSupply/iocBoot/iocpowerSupply

abcob03$ ../../bin/linux-x86/powerSupply st.cmd
```

4.2.5 電源の操作

実習 1 または実習 2 と同じ方法で、実際に電源が操作できるのかどうかをチェックしてください。

4.3 追加演習

時間に余裕がある場合には、以下の機能も実装してみてください。

- 過電流となった場合(リードバック値がある値を超えたとき)に、自動的に OFF にするようにしてください。
- セットポイント値が変更された後に、リードバック値が少しずつ時間をかけてセットポイント値に近づくようにしてください。

- Busy 状態（リードバック値がセットポイント値に達するまで Busy とする)を表すレコードを作成してください。

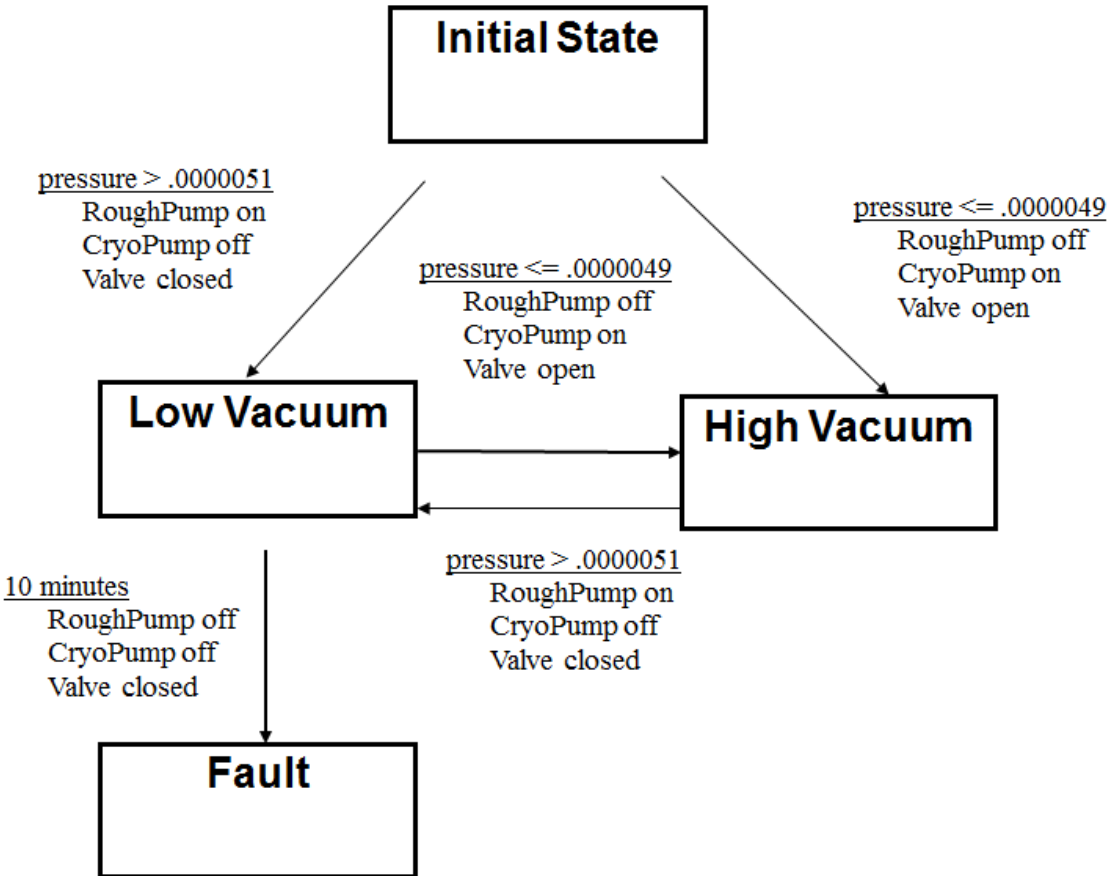
4.4 ヒント

- calc レコードの CALC フィールドでは、”RNDM”という関数が使用できます。この関数は、0～1 までの間の実数をランダムに返す関数です。

5 実習5: State Notation Lanugageの練習

5.1 目的

本実習の目標は、State Notation Language (SNL)を使用して、EPICS アプリケーションを作成するための流れを習得することです。ここで作成するアプリケーションは、下図の状態遷移図を実装した真空システムを模擬したものとします。



5.2 演習

5.2.1 空のEPICSアプリケーションの作成

まず最初に空の IOC を~/lecture/vacuum 以下に sncVacuumApp というアプリケーション名、iocsncVacuum という IOC 名で作成してください。具体的には、以下のコマンドを実行してください。

```
abco4$ mkdir -p ~/lecture/vacuum

aboc4$ cd ~/lecture/vacuum

abco4$ makeBaseApp.pl –t ioc sncVacuum

abco4$ makeBaseApp.pl –i –t ioc sncVacuum
```

次に、~/lecture/vacuum/configure/RELEASE をエディタで開き、”#SNCSEQ=...”と記述されている行を、以下の内容に置き換えてください。

```
SNCSEQ=/proj/epics/R314/R314123-CSA/modules/soft/seq-2.1.12
```

これは、Sequencerがどこにインストールされているのかを EPICS ビルドシステムに教えるためです。

5.2.2 データベースの作成

~/lecture/vacuum/sncVacuumApp/Db 以下に、sncVacuum.db というファイルをエディタで作成し、以下のレコードを作成してください。ただし、 *user* は自身のユーザー名に置き換えてください。

- ET_*user*:VAC:PRES (ai) – 圧力
- ET_*user*:VAC:CRYO (bo) – Cryo Pump の ON/OFF
- ET_*user*:VAC:ROUGH (bo) – Rough Pump の ON/OFF
- ET_*user*:VAC:VALVE (bo) – バルブの開閉
- ET_*user*:VAC:STAT (stringout) – 現在のステータスを表す文字列 (例: “Low Vacuum”)

なお、レコードを作る際には、マクロを使用するようにしてください。レコード間にリンクを張る必要はありません。

次に、~/lecture/vacuum/ sncVacuumApp/Db/Makefile をエディタで開き"#DB += xxx.db”と記述されている行の次の行に、以下の命令を挿入してください。

```
DB += sncVacuum.db
```

また、~/lecture/vacuum/ iocBoot/ sncVacuum /st.cmd 中の dbLoadRecords ではじまる行を以下のように置換してください。ただし、*user* は自身のユーザー名としてください。

```
dbLoadRecords("../../db/sncVacuum.db","USER=user")
```

5.2.3 SNLプログラムの作成

~/lecture/vacuum/sncVacuumApp/src に sncVacuum.stt というファイルを作成してください。このファイルの中身は以下のようになります。ただし、" ..."と記述されている部分は、自身でプログラミングをしてみてください。

```
program sncVacuum

double  pressure;
assign  pressure to “ET_{USER}:VAC:PRES”;
monitor pressure;

short   RoughPump;
assign  RoughPump to “ET_{USER}:VAC:ROUGH”;

short   CryoPump;
assign  CryoPump to “ET_{USER}:VAC:CRYO”;

short   Valve;
assign  Valve to “ET_{USER}:VAC:VALVE”;

string  CurrentState;
assign  CurrentState to “ET_{USER}:VAC:STAT”;

ss vacuum_control
{
   state init{
       entry {
         ...
       }

       when (pressure > .0000051){
         ...
       } state low_vacuum

       when (pressure <= .0000049){
         ...
       } state high_vacuum
   }

   state high_vacuum{
       entry {
         ...
       }

       when (pressure > .0000051){
         ...
       } state low_vacuum
   }

   state low_vacuum{
       entry {
         ...
       }
```

```
        when (pressure <= .0000049){
          ...
        } state high_vacuum

        when (delay(600.0)){
        } state fault
    }

    state fault {
        entry {
          ...
        }

        when () {
        } exit
    }
}
```

~/lecture/vacuum/sncVacuumApp/src に snc.dbd というファイルを作成し、中身を以下のようにしてください(1 行だけです)。

```
registrar(sncVacuumRegistrar)
```

~/lecture/vacuum/ sncVacuumApp/src/Makefile の”#sncVacuum_DBD...”などと書いてあるあたりに、以下の 3 行を追加してください。

```
sncVacuum_DBD += snc.dbd

sncVacuum_SRCS += sncVacuum.stt

sncVacuum_LIBS += seq pv
```

5.2.4 ビルドと実行

~/lecture/vacuum にて、以下のコマンドを実行してください。

```
abco4$ make
```

~/lecture/vacuum/iocBoot/iocsncExample/st.cmd の中の”iocInit()”の後(ファイルの末尾)に、以下の行を加えてください。

```
seq sncVacuum, “USER=user”
```

以下のコマンドを **abcob03 上で**実行し、EPICS IOC を起動してください。

```
abcob03$ cd iocBoot/iocsncVacuum

abcob03$ ../../bin/linux-x86/sncVacuum st.cmd
```

camonitorコマンドにて、ET_*user*:VAC:PRES以外のPVを監視してください。

```
abco4$ camonitor ET_user:VAC:CRYO ET_user:VAC:ROUGH ET_user:VAC:VALVE ET_user:VAC:STAT
```

caputコマンドにて、ET_*user*:VAC:PRESの値を変更することで、上記のPVがどのように変化するのかを確認してください。

iocshでは、以下のコマンドにより、SNLプログラムが現在どの状態であるかといったことを確認することができます。

```
> seqShow sncVacuum
```

6 実習環境の解説

6.1 開発環境(abco4)へのログイン

本実習では、EPICS の開発環境として、abco4 という KEKB 制御ネットワーク上にあるマシンを使用します。このマシンにログインするには、端末を立ち上げ、以下のコマンドを実行してください。ただし、*user* は自身のユーザー名に置き換えてください。

```
$ ssh -X user@abco4.kekb.kek.jp
```

パスワードの入力が求められますので、パスワードを入力し Enter キーを押下してください。ログインをしたら、EPICS に関連するコマンドを実行するために、PATH と EPICS に関する環境変数の設定をしなければなりません。通常は、これは制御システムの管理者により行われます。本実習では、以下のコマンドを実行し、実習環境向けの設定を行ってください。

```
abco4$ source /proj/epics/R314/R314123-CSA/set_path.bash
```

本実習では複数の端末を使用することになるので、3 つほど端末を立ち上げ、上記のコマンドを実行しておいてください。

6.2 実行環境(abcob03)へのログイン

本実習では、EPICS IOC の実行マシンとして abcob03 という KEKB 制御ネットワーク上にあるマシンを使用します。このマシンにログインするには、端末を立ち上げ、以下のコマンドを実行してください。ただし、*user* は自身のユーザー名に置き換えてください。

```
$ ssh user@abcob03.kekb.kek.jp
```

本実習では abcob03 の端末を 1 つ使用しますので、1 つ端末を立ち上げ、上記のコマンドを実行しておいてください。EPICS IOC を起動するとき(st.cmd を実行するとき)には、必ず abcob03 上で行ってください。

6.3 シェル

本資料では、シェルとして bash を使用することを前提としています。以下のコマンドを実行して、”/bin/bash”と表示されているのであれば、お使いのシェルは bash です。

```
abco4$ echo $SHELL
```

“/bin/bash”と表示されていない場合には、bash に切り替える(bash コマンドを実行)か、あるいは export などといった bash 向けのコマンドを、適宜、使用しているシェルに合わせて(例: csh であれば setenv コマンド)に読み替えてください。