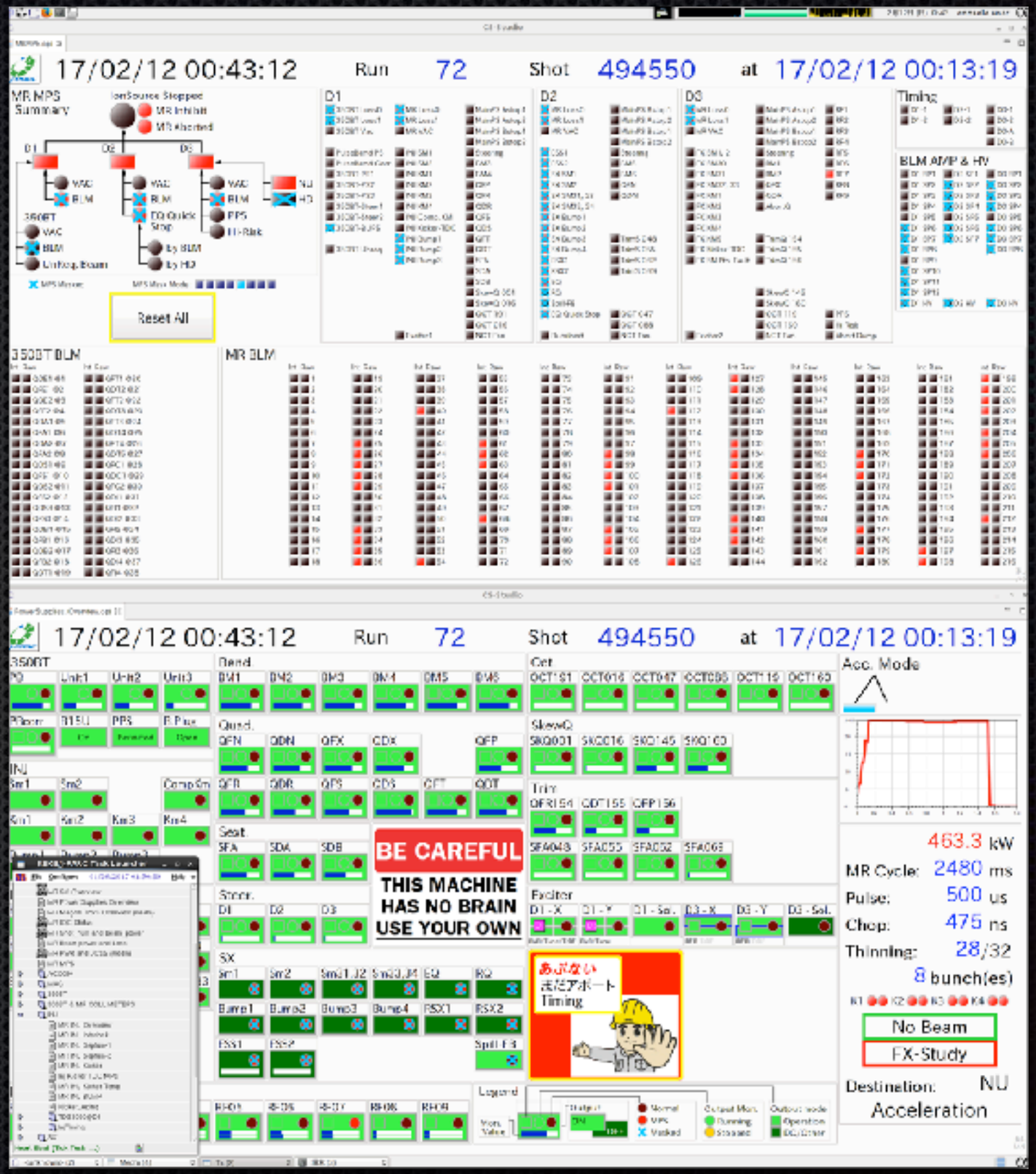
What is EPICS?

2017.02.17 @ OCU by Noboru Yamamoto

目次

- EPICSとは
- EPICS 7 (=3+4)
- CSS
- IOC
- EPICS 演習について



CSSによる表示画面の例

EPICSとは？

- Experimental Physics and Industrial Control System
  - 国際的なEPICSの開発コミュニティ
  - Open Source なネットワーク分散環境向け制御アプリケーション開発フレームワーク
    - EPICS version 3 とEPICS Version 4がある。
    - 統合されてEPICS 7（=3+4)となる見込み

EPICS はxxxではない

- EPICSは制御システムではない。
  - EPICSは制御システムを構築するためのフレームワーク。
  - EPICSを導入しただけで、装置の制御ができるわけではない。制御対象に合わせた作業が必要。
- EPICSは商用のソフトではない。
  - 自ら手を動かす必要がある。
  - サードパーティからサービスを購入することは不可能ではないが。
  - EPICS ReadyなHWを販売しているケースもある。
- EPICSは「DAQ」システムでは無い。
  - 素粒子・原子核実験のようなタイプの実験データの取り扱いにBestとはいえない。

EPICSのユーザ

- 加速器: ANL-APS, SNS, FRIB, BNL, Australian Synchrotron, DESY, Diamond LS, ESS, FNAL, GSI, SLAC, …
  KEKB, PF-AR, J-PARC, RIBF…
- 物理実験: J-PARC NU, APS-BL
- 天文学: Keck II, LIGO, Gemini, KAGRA
- 核融合： ITER

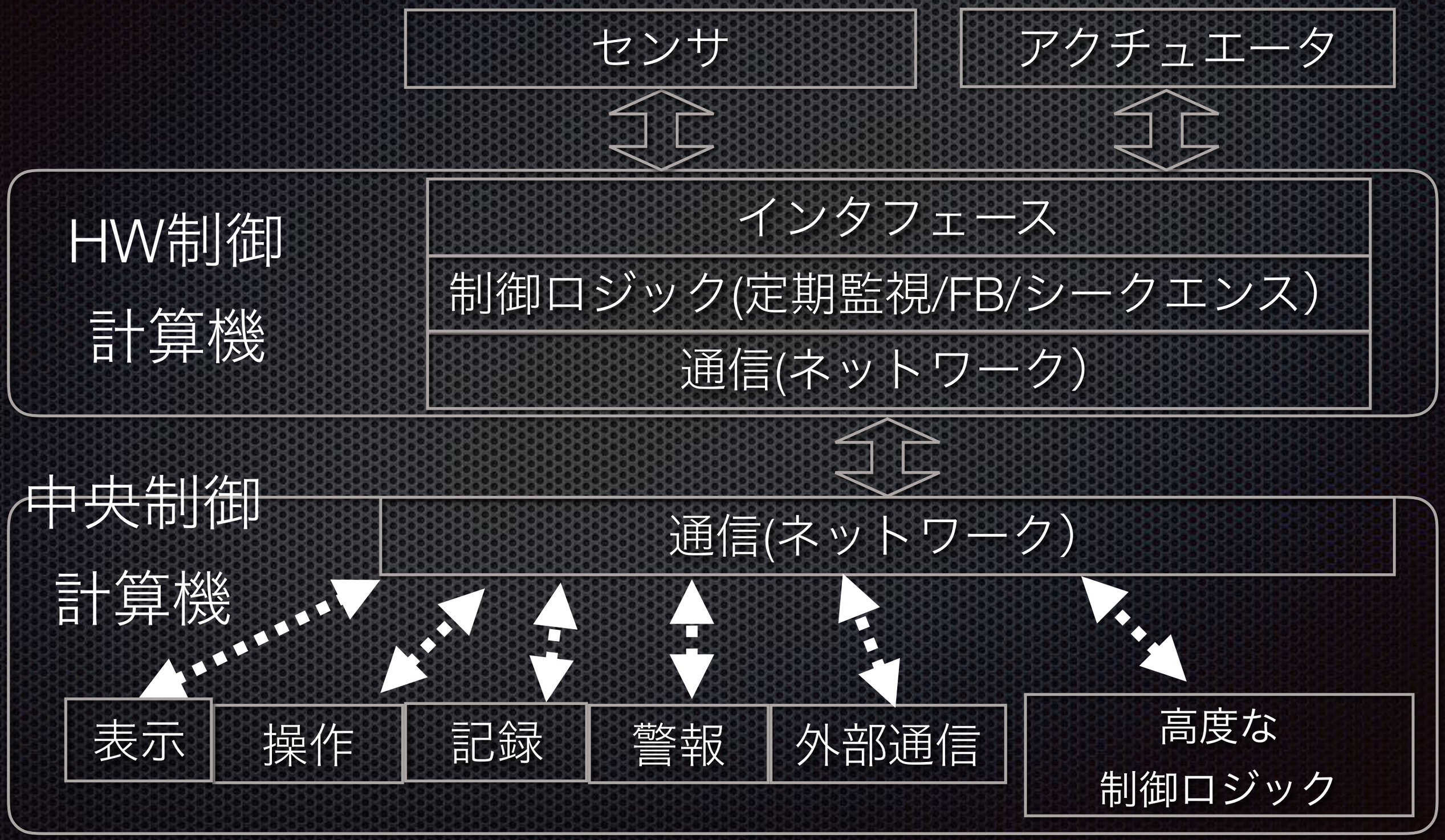
http://www.aps.anl.gov/epics/sites.php

EPICS の10のイカシタところ

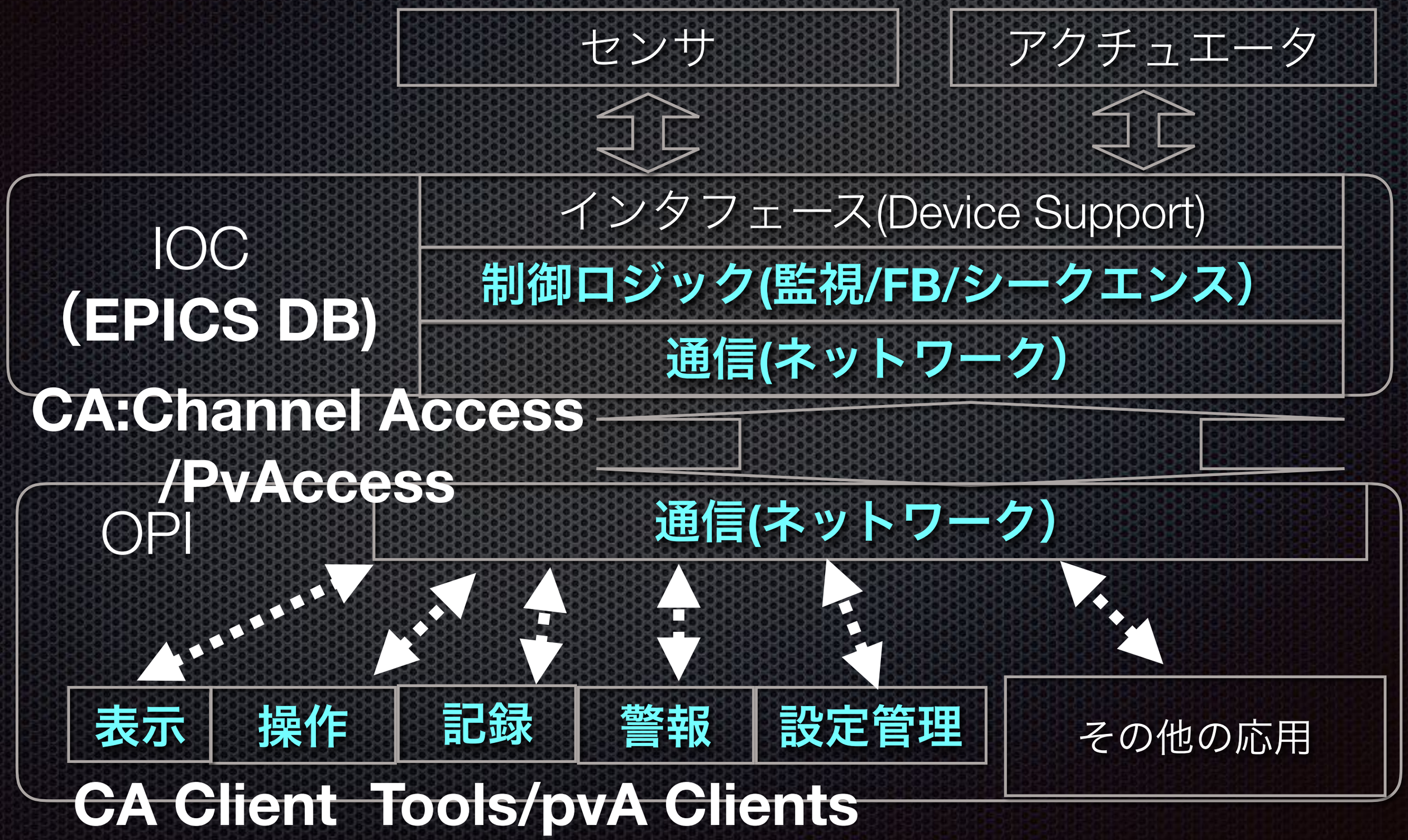
1. 無料である。将来のアップグレードについても課金は発生しない。
2. オープンソースのソフトウェアである。ソースコードはWebから(無料で）ダウンロードできる。
   改変、改良も自由(EPICS ライセンス）
3. たくさんのユーザがいる。：安定した運用の実績がある。
4. 制御ポイントにつけられた名前(CA name)を知っていれば良い。
5. たくさんのソフトウェアから好きなものを選べば良い....
6. ... さもなければ、自分で作り上げることもできる。
7. 退屈な部分はすでに実装されている。
8. すぐ近くに多くの専門家がいる。： 色々相談もできる。
9. 良いコントリビューションは国際的に受け入れられる。 ：国際的な活躍の場がある。
10. 制御点数が10個でも、100万個でも対応できる。: Scalability

”Ten Really Neat Things About EPICS”@http://www.aps.anl.gov/epics/neat.php

ネットワーク分散制御のパターン



EPICS制御のパターン



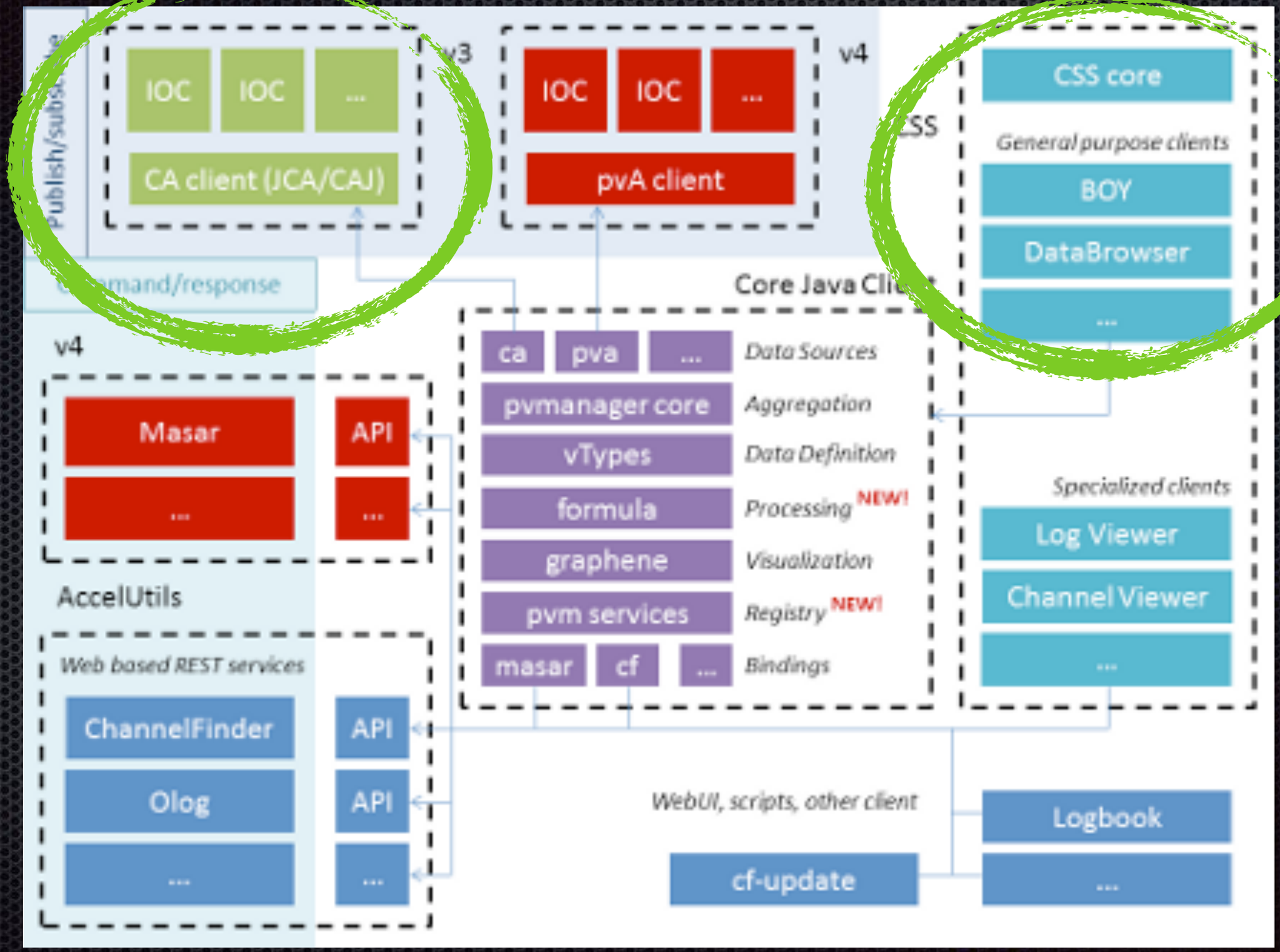
Network プロトコル

- EPICS はネットワーク標準：TCP/IP, UDP/IPの上に構築されている。
- EPICS version 3 ではCA：Channel Access
  - Int/Float/StringなどのPrimitive DataType + α(Timestamp, severity, status, range-information,..)
  - get/put/monitor + callback
- EPICS Version 4 では　PvAccess
  - Composite Datatyes："Normative Types”(NTypes)
  - get/put/monitor + put-with-get, an RPC-type callback
  - CA serverにもTransparentにアクセス可能

EPICS 7

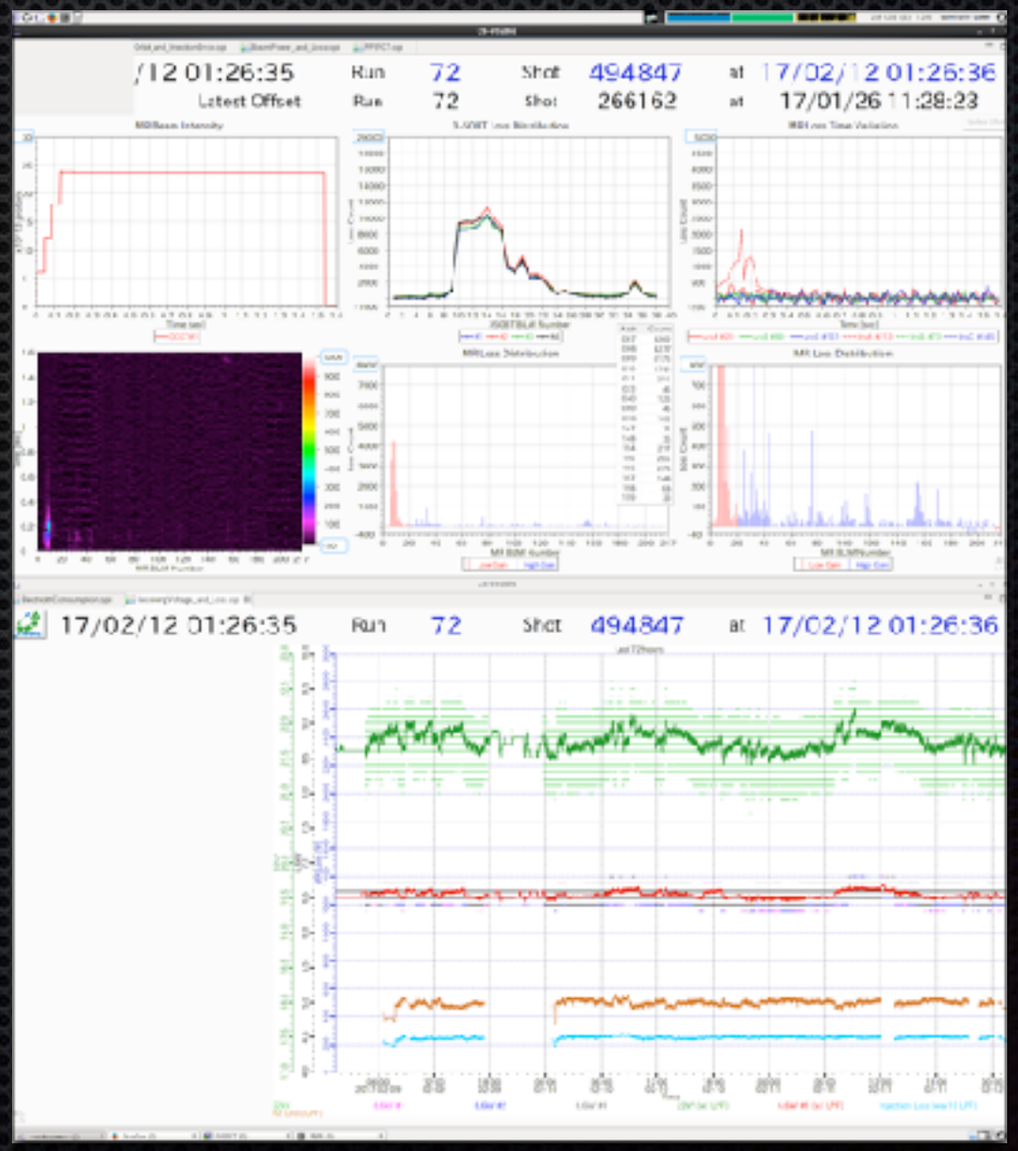
- EPICS V3 とEPICS V4は共存できる。
  - EPICS V3: HWのサポートが豊富
  - EPICS V4:高機能なアプリケーションを実現できる。
    - 例えば、CSS (Control System Studio)

CSS の概要

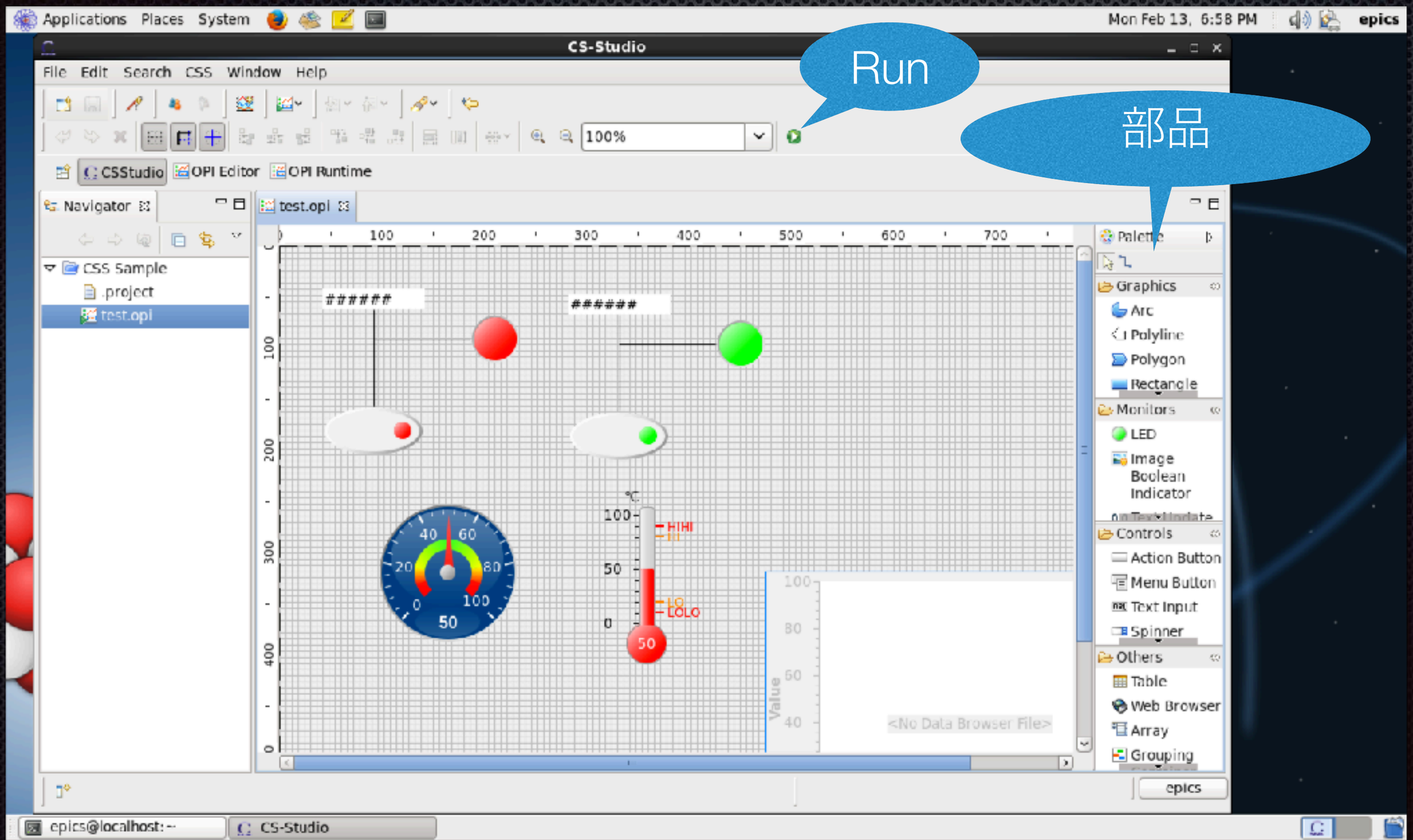

IOC
IOC
...
CA client (JCA/CAJ)
v3
IOC
IOC
...
pvA client
v4
CSS core
General purpose clients
BOY
DataBrowser
...
Core Java Cli
v4
ca
pva
...
Data Sources
pvmanager core
Aggregation
vTypes
Data Definition
formula
Processing NEW!
graphene
Visualization
pvm services
Registry NEW!
masar
cf
...
Bindings
Masar
API
...
...
AccelUtils
Specialized clients
Log Viewer
Channel Viewer
...
Web based REST services
ChannelFinder
API
Olog
API
...
...
WebUI, scripts, other client
Logbook
cf-update
...

CSS/BOY,DataViewer

- 制御対象のデータを様々な形で表示。操作も。
  - 豊富な部品
  - Channel 名
  - 各部品の設定
- Archiveデータと統合
- Script(JAVA/Python)による拡張



CSS編集画面


Applications Places System
Mon Feb 13, 6:58 PM
epics
CS-Studio
File Edit Search CSS Window Help
Run
部品
100%
CSStudio
OPI Editor
OPI Runtime
Navigator
test.opi
CSS Sample
.project
test.opi
Palette
Graphics
Arc
Polyline
Polygon
Rectangle
Monitors
LED
Image
Boolean
Indicator
Controls
Action Button
Menu Button
Text Input
Spinner
Others
Table
Web Browser
Array
Grouping
<No Data Browser File>
epics
epics@localhost:~
CS-Studio

CA Applicationを作るには？

Languages support EPICS CA/PvAccess

- C++
- Java
- Python/Perl
- LabView/Matlab/Octave/Scilab
- C#/ node.js /websocket(?)
- SAD (KEK作成の加速器物理向けプログラム)

IOC:Input Output Controller

IOC

- HWとのインタフェースとなる計算機
  - Unix(Linux, MacOSX, BSD) / VxWorks(商用RT-OS)/ RTEMS(Open Source RT-OS)/Windows
  - EPICS Database
    - HW Scanning / Data Processing / HW Control
    - 割り込み処理 / アラーム処理
  - CA/PvAccess server
    - 上位計算機との通信

EPICS Database

- EPICS データベースはRelationなデータベースではない。
- HW制御に必要なデータをon Memoryで管理している。
- EPICSではHWとの入出力は全てこのDatabase経由で行われる。(HWのアドレスなどは全てDBで隠蔽される)
- DatabaseはRecordの集まりとして構築される。

EPICS Record

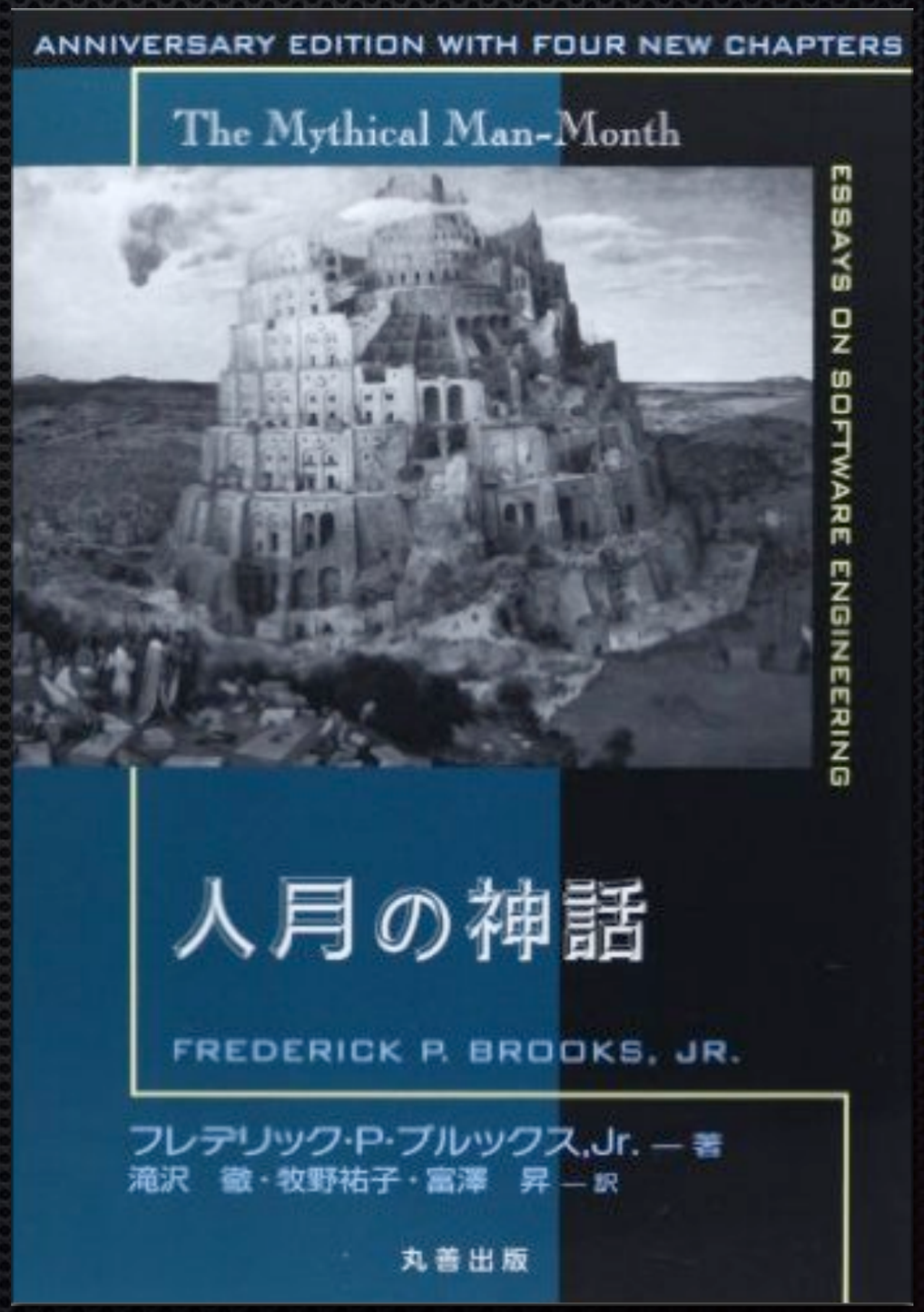
- 一つのRecordがHWの一つの入力点あるいは出力点に対応している。
- Recordは固有の形、それに応じた属性値及び「動作」を持っている。(Record Processing)
- 例：AI (Analogue Input)レコード
  - VAL：最終の読み取り値／HOPR,LOPR: 動作範囲の上限下限
  - INP: 入力アドレス／DTYP: デバイスのタイプ
  - 動作：指定されたアドレスからのデータの読み込み

閑話休題

- ”Zen of Python” says:
  - There should be one-- and preferably only one --obvious way to do it.
- 一方、EPICSでは同じことを実現する複数の方法が提供されていることがある。
  - 歴史的事情／それぞれの方法の適合範囲の違い
  - システムの発展に伴い、Refactoringは必要かも。

狼男を撃つ銀の弾は無い

人月の神話 【新装版】
Jr FrederickP.Brooks

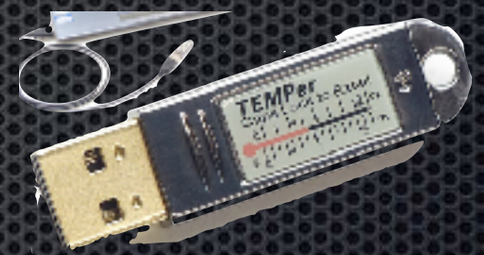


制御ロジックの実装

- Record リンク
  - Database内複数のレコードを連携(Link)して動作させる。
  - Database 組み込み機能。C/C++などのプログラムは不要
- 新規レコード型の導入: C/C++による開発
- Sequencer プログラム :有限状態機械による記述からプログラム作成
- C/C++/Java/Perl/Pythonなどによるアプリケーション開発

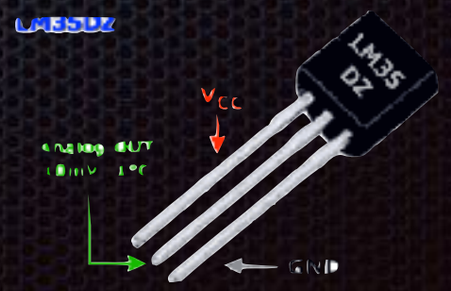
Databaseの例

- 温度を測定したい！
  - 電子温度計: Network/USB/Serial

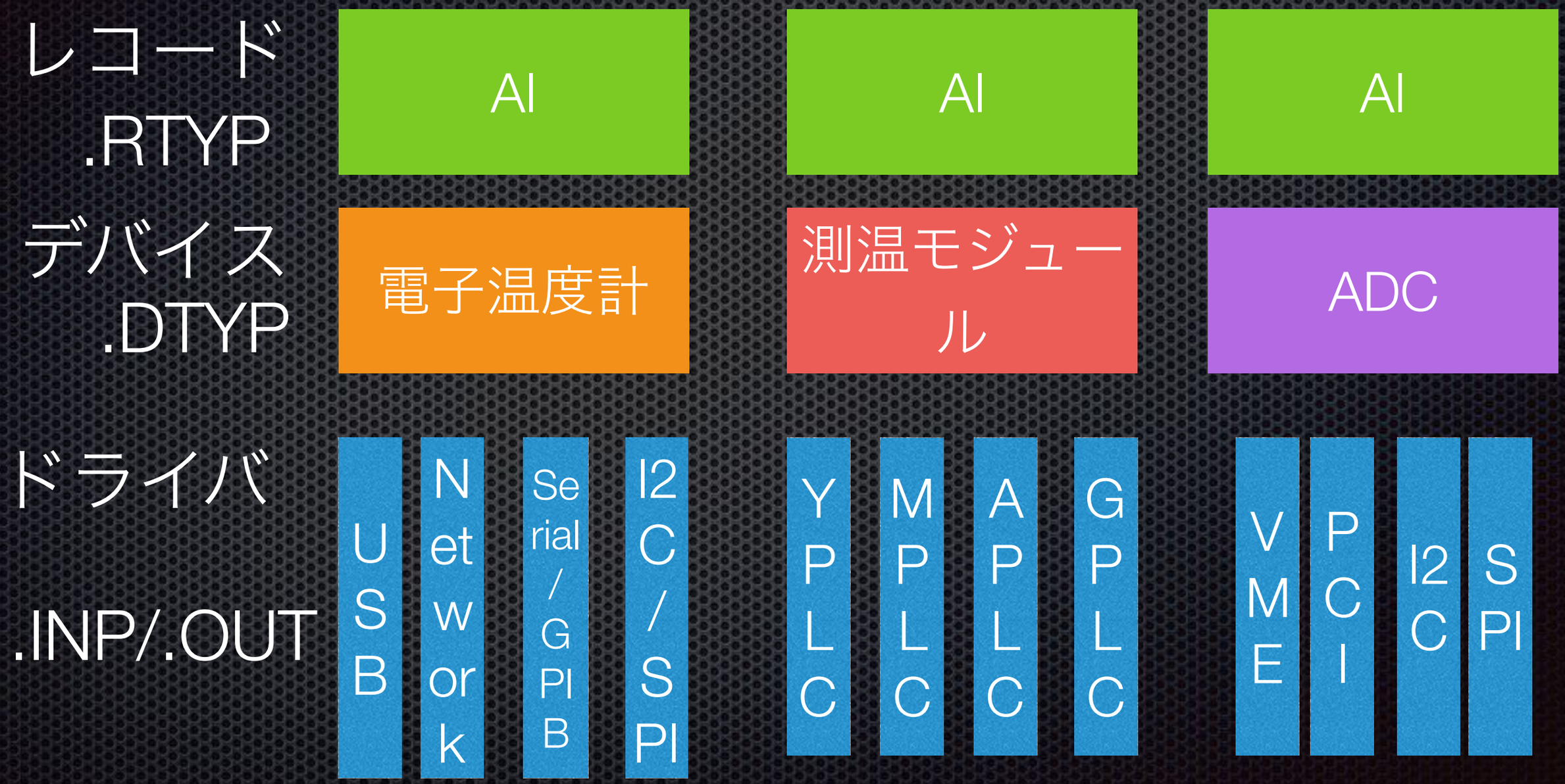


  - 温度センサ+測温モジュール＋計測用計算機(PLCなど)

  - 温度センサとADC (VME / PCI / I2C / SPI / USB….)

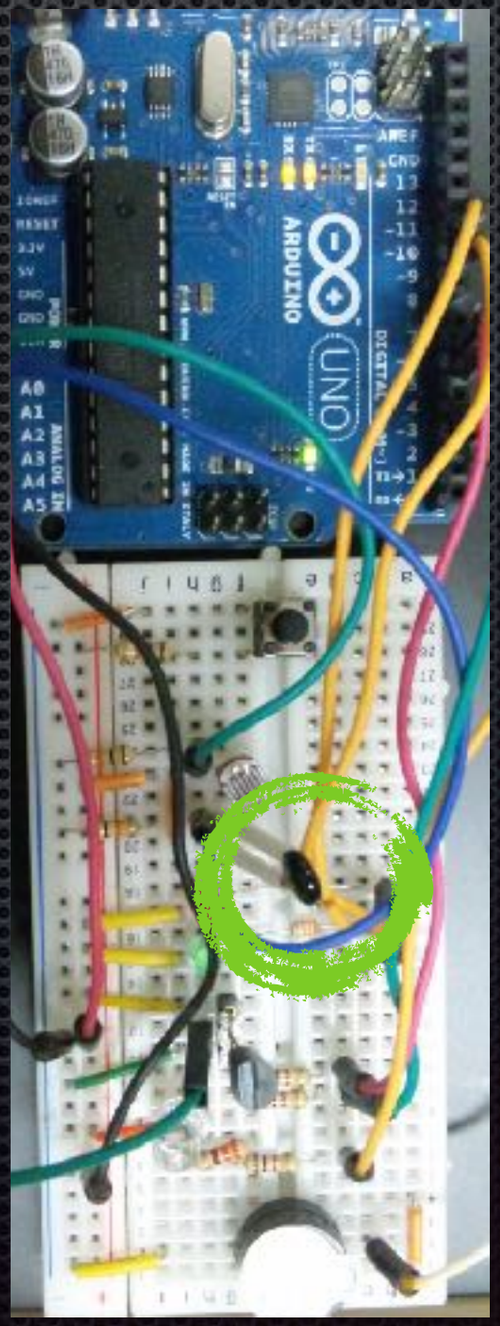


Databaseの例（続き）



温度測定の一例

- 温度によって抵抗が変化する素子(Thermistor)
- ArduinoのADC機能を使い、電圧信号を取得
- ArduinoとCPU(IOC)との間はUSB(CDC、CPUからはSerialポートと見える)
- StreamDeviceライブラリを使う。



EPICS Databaseの例

```
record(ai, "$(P):THERM")
{
    field(DESC, "Thermistor")
    field(DTYP, "stream")
    field(INP,  "@devArduino.proto ain(1) $(TTY)")
    field(SCAN, "$(ascan)")
    field(LINR, "LINEAR")
    field(ASLO, "0.004887585532746823")
    field(HOPR,"5.5")
    field(LOPR,"0.0")
}
```

IOC起動スクリプト(抜粋)

```
# Serialポートを選択
drvAsynSerialPortConfigure "TTY", "/dev/serial/by-id/usb-RTcorporation_GAINERmini_v1.1-if00"
...
## Load record instances
dbLoadRecords("../Db/devArduinoProto.db","P=myArduino,R=0,TTY=TTY,ascan=.1 second")
...
iocInit()
```

EPICSを使い始めるには？

- EPICS home page
  - http://www.aps.anl.gov/epics/
    - source code
    - Distributions: Debian, VirtualBoxImage
    - Device Support, including StreamDevice

EPICS　IOC実演

まとめ

- EPICSはOpen Sourceの計算機による機器制御システム構築のためのフレームワークである。
  - EPICSを利用することで、制御システムを（比較的）簡単に構築できる。
- EPICSは小さな実験装置から、加速器や素粒子実験など機器までカバーできる。
  - 導入を考えている方はご相談ください。

One more thing.....

Spring 2017 EPICS Collaboration Meeting

About the Meeting

The Spring 2017 EPICS Collaboration meeting is held by Research Reactor Institute, Kyoto University (KURRI) and takes place at KURRI from 15-19 May 2017.

5/15(Mon)
Seminar, Tutorial, Training

5/16(Tue)-5/18(Thu)
Collaboration meeting - Main tracks

5/19(Fri)
Optional day

EPICS (Experimental Physics and Industrial Control System) is a set of open source software tools, libraries and applications developed collaboratively and used worldwide to create distributed soft real-time control systems for scientific instruments such as particle accelerators, telescopes, and large scientific experiments. These collaboration meetings provide a chance for developers and managers from the various different sites to come together and discuss their work and progress, and make plans for the future. They give a chance to see what is being done at other laboratories, and to review the specifications for new tools or enhancements to existing ones in order to maximize their usefulness to the whole community and avoid duplication of effort.

For further information, please contact the EPICS 2017 local team at epics2017##rri.kyoto-u.ac.jp (Replace ## with an atmark).

News

Jan. 27, 2017
This web site is opened.

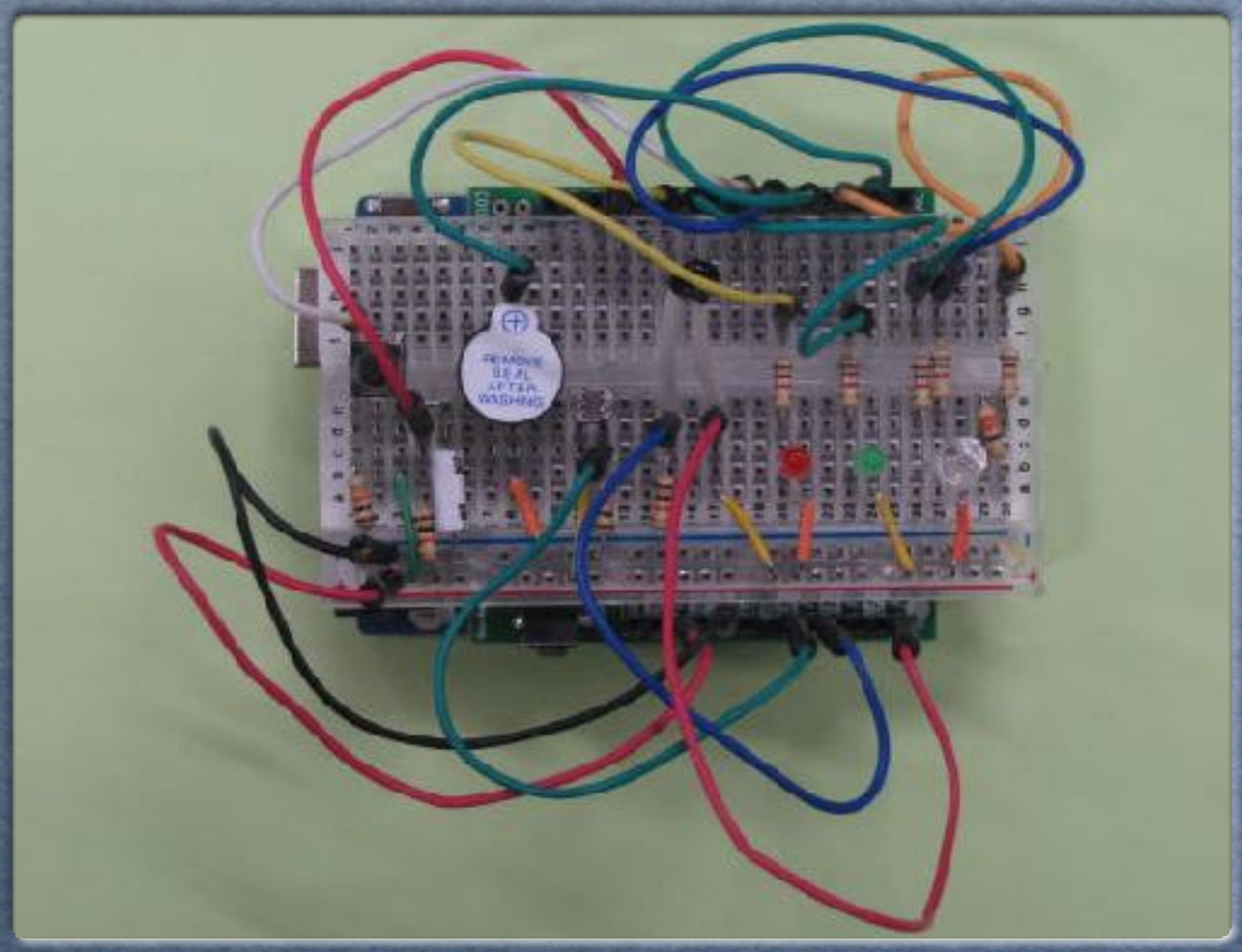
pageTop

Copyright © 2017 KURRI All Rights Reserved.

EPIC 講習会

講習の内容

- EPICSによる制御システムの動作を講習用の装置を使い経験する。
  - IOC/OPIとしてRaspberry Piを使用
  - 操作デバイスとしてArduinoとそれに接続されたいくつかの 素子(LED, Thermistor, CdS, push switch, Buzzer) を使う。



Includes
1 TriColor LED(RGB)
2 LEDs（Red&Green)
1 Buzzer
1 Push Switch
1 Thermistor(温度)
1 CdS(明るさ)
1 level Switch(傾き検知)

~ 5,000JPY
including Arduino

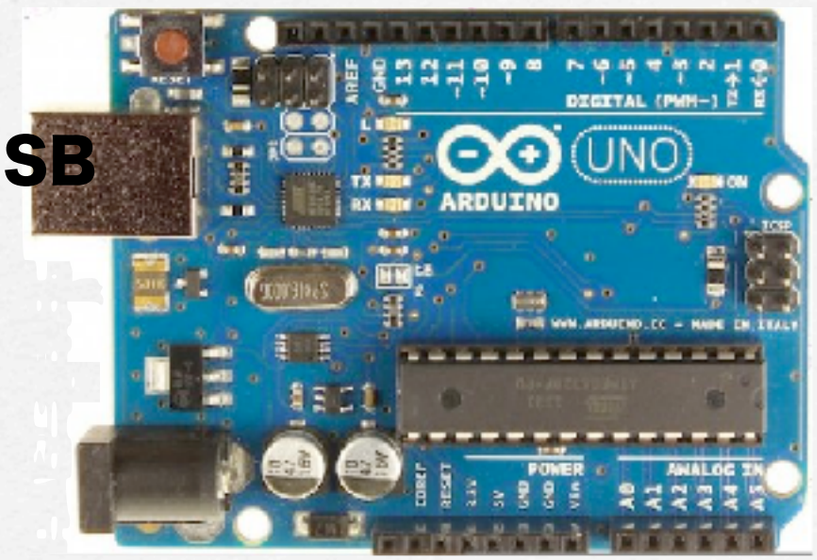
Arduino shield(ext. board)
with Bread board
for EPICS training

Arduino

Arduino Uno

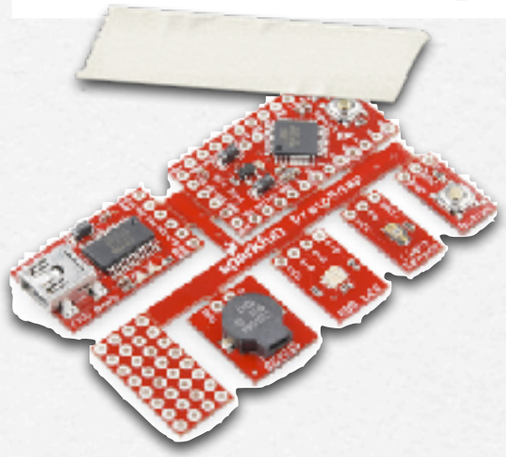
DI/DO/AO ports

USB

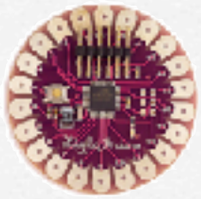


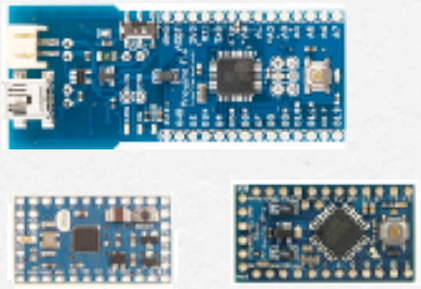
AI ports

Arduino Leonard
USB micro
Latest ver. of Arduino



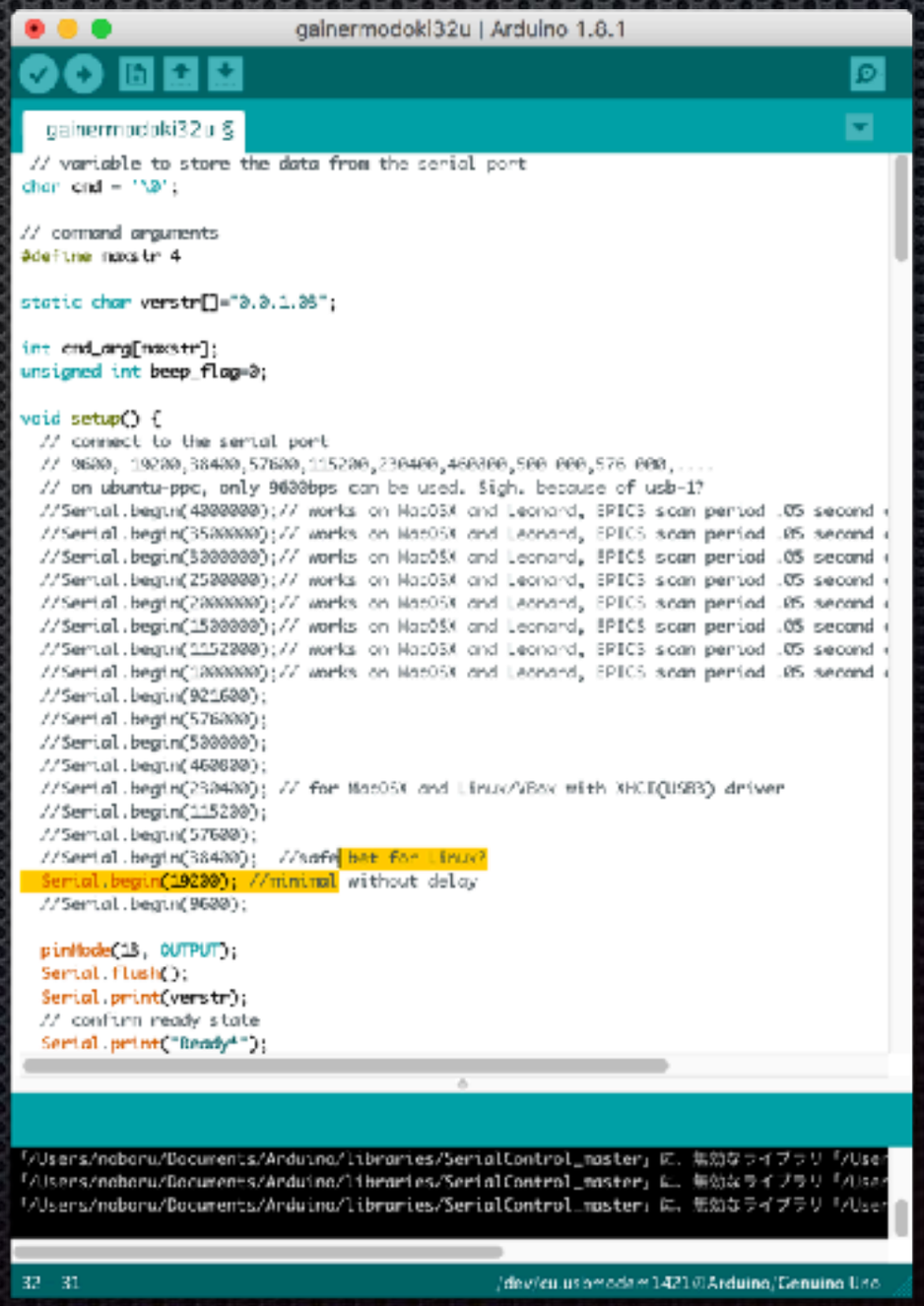
Arduino ProtoSnap-Mini
inclues 3color LED,
Buzzer, Light sensor
and Push Button





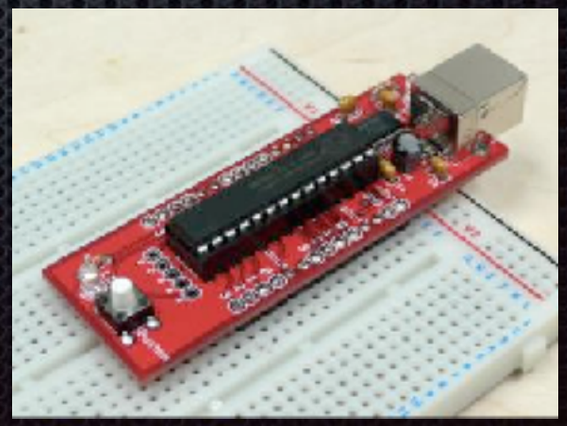
Arduino

- IO board with AVR AT−mega series micro CPU: various type
  - 単独での動作も可能
  - digital IO/ analogue IO
- Arduino language
- Arduino-IDE
  - コンパイル
  - Binary code download



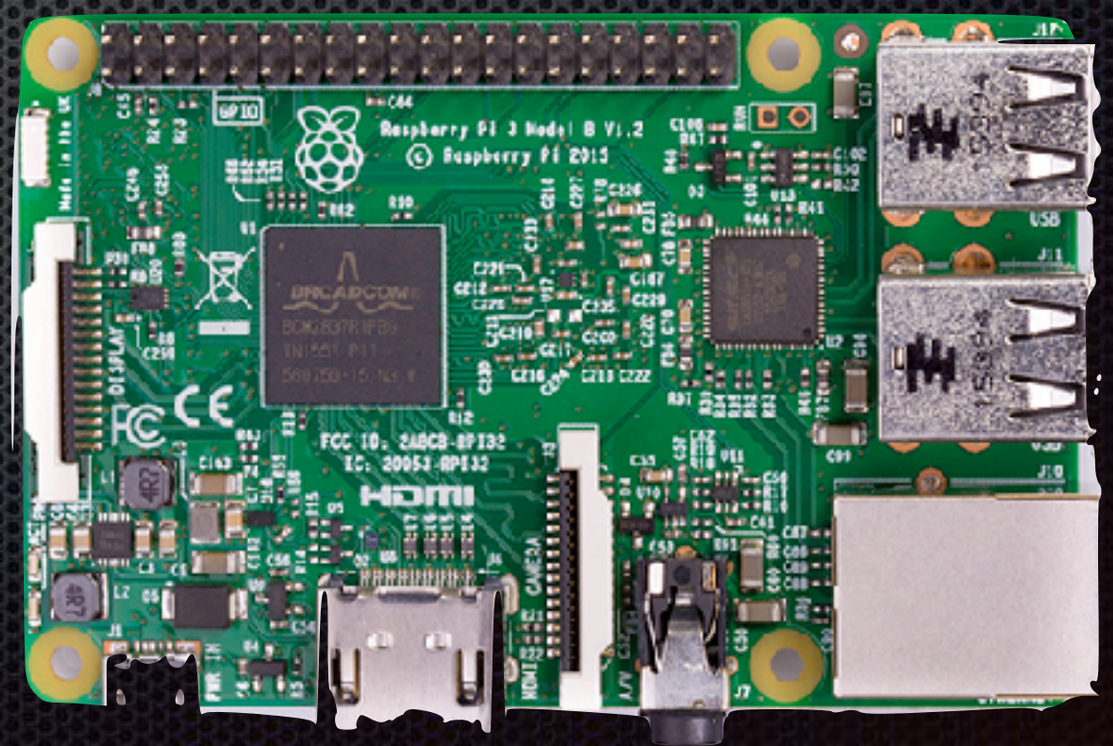
Gainermodoki.ino

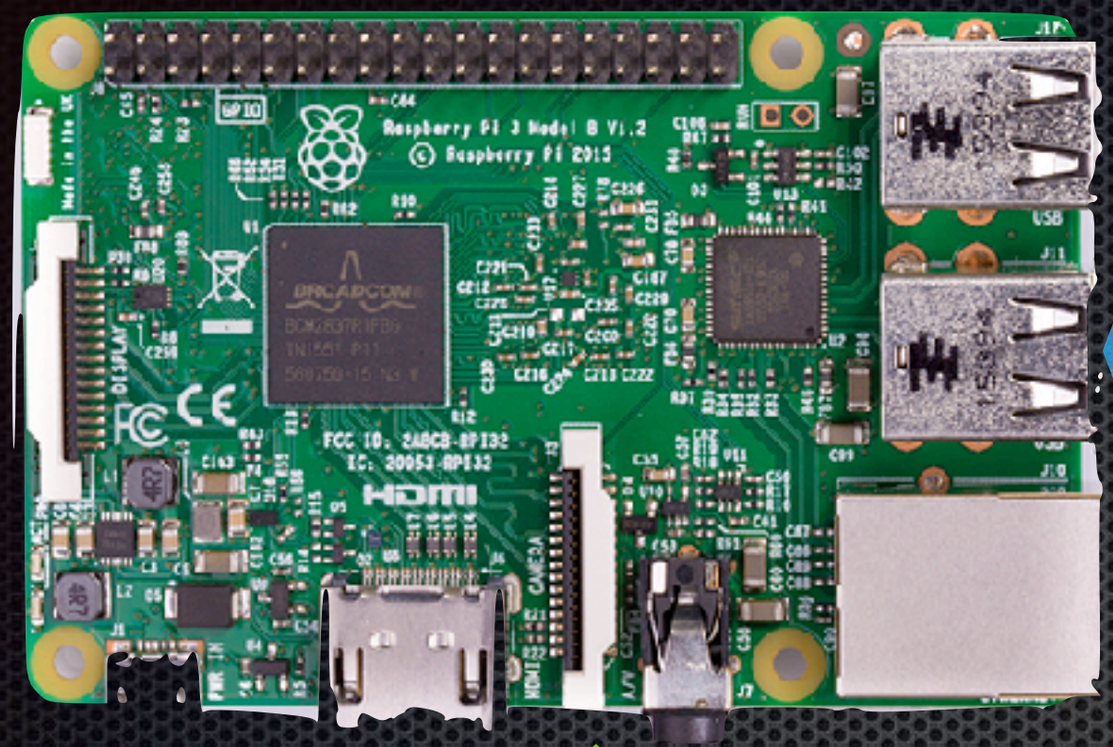
- Arduino Sketch(program in Arduino Language)
- read a simple command from serial/usb connection
- return response to PC through serial/usb connection.
- “modoki”:a Japanes word for “pseudo-” or “-like”
  - Gainer: a small USB I/O board produced in Japan.



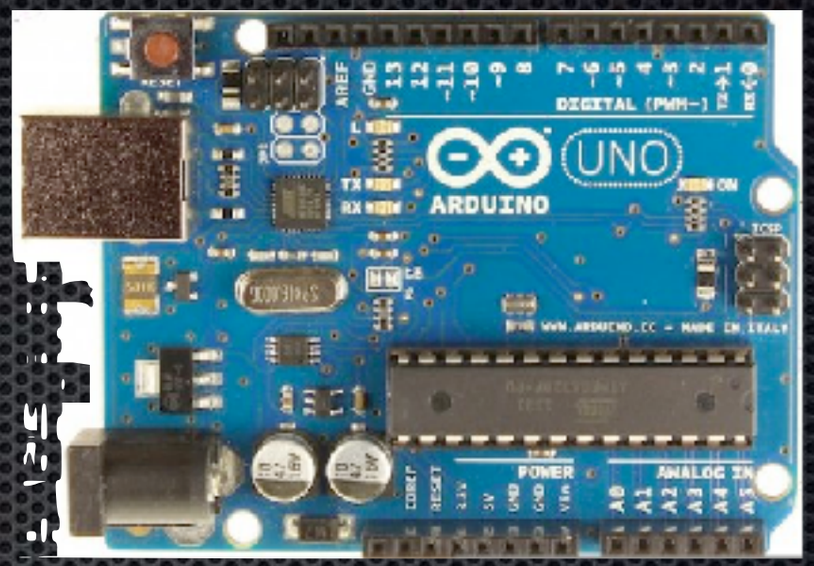
Raspberry PI

- カードサイズコンピュータ：ARMアーキテクチャの統合型CPU
- Linuxなどの OSを使用可能：KB,Mouse, Display,電源を繋げばPC
- ネットワーク機能(Ethernet / WiFi)、USB port 搭載
- Digital IO(GP-IO)が可能
- I2C/SPIなども利用可能





USB
serial



StreamDevice

IOC:Database

TCP/IP

CA Client

RPi as IOC

- cd /opt/epics/modules/soft/StreamDeivce/apps/script
- ./ArduinoProt.cmd を確認／編集
  - ArduinoをUSB経由で接続
  - Serial portを確認：
    - /dev/ttyACMXX,
    - /dev/serial/by-id/…
    - /dev/serial/by-path/…

StreamDevice

- Serial, TCP/IP, VXI-11, GP-IBなどStreamでコマンドのやりとりによって制御する装置をEPICSに組み込むための汎用ライブラリ
- 装置ごとの違いは
  - Streamの初期化コマンド：drvAsynXXXtConfigureXXX
  - protocolファイル
    によって吸収される。

protocol file / Database file

```
Terminator = CR LF;

setCurrent {
    out "CURRENT %.2f";
    @init {
        out "CURRENT?";
        in "CURRENT %f A";
    }
}
```

```
record (ao, "PS1:I-set")
{
    field (DESC, "Set current of PS1")
    field (DTYP, "stream")
    field (OUT,  "@ExamplePS.proto setCurrent PS1”)
    # @file protocol busname
    field (EGU,  "A")
    field (PREC, "2")
    field (DRVL, "0")
    field (DRVH, "60")
    field (LOPR, "0")
    field (HOPR, "60")
}
```

```
# in IOC command file
drvAsynIPPortConfigure ("PS1", "192.168.164.10:23")
```

EPICS の環境変数

- EPICS_CA_ADDR_LIST
  - サブネットを超えたIOC/OPIの通信では設定が必要。
- EPICS_CA_MAX_ARRAY_BYTES
  - CAのパケットサイズは32kBがデフォルト、長いarrayデータを取り扱うためには、IOC側／OPI側の双方で、適切なサイズの設定が必要。
- EPICS_TS_MIN_WEST ：-540 for JAPAN
  - Localの時刻表示を正しく行うために。(EPICSの時刻管理はUnixの時刻管理とは異なっている。)

IOC-shの起動

- Terminalを開いて
  - cd /opt/epics/modules/soft/StreamDevice/app/script
  - sudo ./ArduinoProto.cmd

IOC-shコマンド

- dbl [<record type>]
- dbgrep <pattern>
- dbpr <record name> [<level:0.. 4>]
- dbgf <record name>
- dbpf <record name> <new value>
- help

Python-CA : Install

- cd /opt/epics/modules/soft/PythonCA....
- python setup.py build
- sudo python setup.py install

Python-CA:簡単な使い方

- import ca
- ca.Get(‘myArduino:LED:R’)
- ca.Info(‘myArduino:LED:R’)
- ca.Put(‘myArduino:LED:R’, 0.9)
- ca.Monitor(‘myArduino:LED:R’)

python-CA: channel object

- import ca
- chobj=ca.channel(‘myArduino:LED:R’)
- ca.flush(); print chobj.isConnected()
- chobj.get(); print chobj.updated; ca.pend_event(0.1);print chobj.updated
- print chobj.val
- chobj.put(0.7); ca.flush()
- chobj.monitor(chobj.update_val)

EPICS開発環境の準備

EPICSアプリケーション開発環境

- EPICSの標準配布に含まれる

  makeBaseApp.pl
  スクリプトを用いる。

- EPICS　on Ubuntu では、
  - %makeBaseApp.plで実行可能
  - % which makeBaseApp.pl
    を試してみましょう。

参照文書

- EPICS: Input / Output Controller
- Application Developer's Guide

- Release 3.14.7, 13 Dec 2004
- Release 3.14.8, 02 Dec 2005
- Release 3.14.9, 05 Feb. 2007
- Release 3.14.10, 27 Oct. 2008

- http://www.aps.anl.gov/epics/base/R3-14/10.php
  などから入手可能

- EPICS-Baseについての決定版文書。ただし、一挙にすべてを読むには適さない。
  疑問点／不明点があればこの文書にまずはあたって見る。

開発環境の役割

- EPICSアプリケーションの開発
  以下の様に様々な要素がある。それぞれの要素毎に適切な環境変数等を設定。ディレクトリ構造も仮定することで、効率化をはかる。これにより、コンパイラ／リンクオプションの指定等が単純化される。
  - データベース
  - OPI
  - Device Support/Record Support
  - Sequencer プログラム
  - Caクライアント
- EPICSアプリケーションの実行
  - IOC/IOCSHの起動スクリプト等を半自動的に生成してくれる。

makeBaseApp

- makeBaseApp.plが作成する環境ではTopのアプリケーションディレクトリの下に設定ファイル／アプリケーションソース／実行用スクリプト等を配置する。（後で詳述）
- makeBaseApp.plを実行する前に、このTopとなるディレクトリを作成する。
- mkdir ~/EpicsADE
- cd ~/EpicsADE
- Topディレクトリ内で次の二つのコマンドを実行する。
- makeBaseApp.pl -t example mySample
- makeBaseApp.pl -t example -i mySample
- example: テンプレートの名前。利用可能なテンプレートの一覧は

makeBaseApp

- makeBaseApp.pl -t example water
  によって以下のディレクトリやファイルが作成される。:
- <top>/
- Makefile
- configure/
- ...
- SampleApp
  src/
- ...
- Db/
- ...
- makeBaseApp.pl -t example -i heater
  では、以下のディレクトリとファイルが作成される。
- <top>/iocBoot

<top> Directory Structureの作成

- Topディレクトリでmake BaseApp.plを使って初めてアプリケーションディレクトリを作成したら、
- cd ~/EpicsADE
- make EPICS_HOST_ARCH=linux-x86 install
- を一度だけ実行する
- EPICS_HOST_ARCH環境変数は、アプリケーションを開発するホストのOSとCPUアーキテクチャを示す文字列。EPICSでサポートされているホストには、
  - linux-x86 :intel x86 上のlinux
  - darwin-ppcx86 : MacOSXのUniversal binary
  - win32-x86 : Windows上のMS C++を使った開発
  - linux-x86_64 : 64bits Linux
  - win32-x86-cygwin: cygwin環境をWindowsで使う場合。

  などがある。

RELASE

- configure/RELEASEファイルでインストールされているEPICSの環境等の設定を行う
  - EPICS_BASE=$(TOP)/../base
  - SNCSEQ=$(EPICS_BASE)/../modules/soft/seq/seq-current
  - ASYN=$(EPICS_BASE)/../modules/soft/asyn/asyn-current
    - StreamDeviceはAsynに依存しているので、この環境変数の設定が必須。
  - STREAM=$(EPICS_BASE)/../modules/soft/StreamDevice

<top> Directory Structure

- <top>　アプリケーション作成の作業用ディレクトリ
  - 例ではEpicsADE
- makeの前の<top>ディレクトリは：
- <top>/
- configure/　設定ファイル等。通常は変更の必要なし
- RELEASE　EPICS Baseのインストール先等を設定
- xxxApp/　起動スクリプト以外の総てのソースコードは此処に
- src/　C/C++/SNL等のソースコード
- xxxSrc/　複数の xxxSrc ディレクトリが可能
- Db/　Databases, templates & substitutions files
- xxxDb　複数のxxxDbディレクトリが可能
- ...　adl, edm,などのディレクトリがあってもよい。

<top> Install Directories

- makeの実行後の<top>ディレクトリは:
- <top>/
- bin/ コンパイル済みオブジェクトおよび実行ファイル
- linux-x86/ ターゲット毎にディレクトリ
- vxWorks-68040/
- lib/ ライブラリ用
- linux-x86/ binと同じくターゲット毎にディレクトリ
- include/ ヘッダファイルがインストールされる。
- dbd/ menu, recordtype, device, driverなどMakeが書き換えるので、編集不可
- db/ DB 設定ファイル, templatesなどdbdと同じく編集不可
- javalib/ .jar ファイル

<top>/configure files

- 変更する可能性のあるファイル
  - CONFIG - Makeの設定の上書きが可能。通常は、makeBaseApp.pl実行時の設定でよいはず。
    CROSS_COMPILER_TARGET_ARCHS = vxWorks-68040
  - CONFIG_APP
    その他の設定。
  - RELEASE
    アプリケーション構築に使われるベースの場所
    EPICS_BASE=/opt/epics/R3.14.7/base
- その他のファイルは変更する必要がない。不用意に変更するとmakeのルールが壊れてしまいます。
  - RULES*
    EPICS baseのMAKEルールあるいは其処へのリンク
  - Makefile
    makeコマンドの設定ファイル。

<top>/xxxApp/src files

- base.dbd - baseが提供するレコード／デバイスなどの定義
  - 不要なレコード型／デバイスサポート等を取り除くことが可能
  - EPICSベースが提供するmenu, recordtype, device, drivers へのリンクが記述されている。

- include “menuGlobal.dbd”
  include “menuConvert.dbd”
  include “aiRecord.dbd”
  ...
- include "waveformRecord.dbd"
- device(ai,CONSTANT,devAiSoft,"Soft Channel")
- ...
  device(waveform,CONSTANT,devWfSoft,"Soft Channel")

xxxApp/src の続き

- Makefileにはソースコードからアプリケーションを作るのに必要な情報を記述する。
- TOP=../..
- include $(TOP)/configure/CONFIG
- DBDINC += xxxRecord
- DBD += Sample.dbd
- PROD_IOC = Sample
- Sample_SRCS += xxxRecord.c devXxxSoft.c SampleMain.cpp
- Sample_SRCS += Sample_registerRecordDeviceDriver.cpp
- Sample_LIBS += iocsh miscloc rsrvloc dbtoolsloc asloc dbloc
- Sample_LIBS += registryloc dbStaticloc ca Com
- include $(TOP)/configure/RULES
- SampleInclude.dbd
- include “base.dbd”
- include “xxxRecord.dbd”

src/Makefile

- StreamDevice Supportを使うために, Makefileに以下の行を追加する。
  - <appName>_LIBS += stream
  - <appName>_LIBS += asyn
  - <appName>_DBD += stream.dbd
  - <appName>_DBD += asyn.dbd
- ただし、<appName>はmakeBaseApp.plで作成したアプリケーション名に合わせて、変更すること。

xxxApp/xxxDb/Makefile

- データベースはMakefileに登録が必要
  - DB += xxx.db
    makeで<top>/dbにコピーされる
- 複数の同じ型のデバイスがある場合には、そのデータベースはテンプレートとsubstitution ファイルを使うと便利。
  - テンプレートの一例：yyy.template
- record(ai,”$(USER):aiExample$(NO)”){
  … }
    - 対応する substitutionファイルは：yyy.substitutions
- file yyy.template {
  {USER=“mrk”,NO=“1”}
  {USER=“mrk”,NO=“2”}
- }
    - Makefileに以下の２行を追加
      - make時にmsiコマンドを使って展開(CD-ROMに含まれず)

iocBoot/iocxxx

- iocshを起動するためのスクリプト(st.cmd)が用意されている。
- iocBoot/iocxxx/Makefile
  Creates cdCommands file for vxWorks targets. Make sure that
  vxWorksのターゲット向けのdbCommandsが設定されるためには,
  Makeファイルへの登録が必要
    - ARCH = <arch>
        - is defined correctly in the Makefile
    - For a vxWorks target the cdCommands output file looks like
- startup="<full path to iocxxx>"
- appbin="<full path to top/bin/arch>"
- …
    - A vxWorks st.cmd file looks like
- < cdCommands
  cd appbin
  ld < iocCore
  ld < xxxLib
  cd startup

make の実行

- makeはあらかじめ指定された依存関係（Makefile)に従って、コンパイルなどを実行する。
- <top> 以下のどのサブディレクトリでもmakeを実行可能
  - <top>
    - make clean uninstall
      総てのバイナリ等を消去して、ソースコードだけを残す。
    - make
      更新が必要なファイルを更新し、しかるべきところにインストール
  - configure
    - make
      RELEASEに従ってターゲット毎に必要なファイルを更新
  - iocBoot/iocxxx
    - make
      envPathファイルを作成する。

Make Targets (commands) cont.

- xxxApp/xxxSrc
  - O.<arch>にコンパイル済みのオブジェクトを作成
  - <top>/lib/<arch>, <top>/bin/<arch>, <top>/db, <top>/db, <top>/includeなどにファイルをコピー
  - make
    総ての更新が必要なコンポーネントを再構築する。
  - make <arch>
    特定の<arch>について再構築する。
    例：make vxWorks-ppc604
  - make clean
    すべての O.<arch> を消去する
- XxxApp/xxxDb
  - make
    データベースのインストールなどを行う。

Application Templates

- makeBaseApp は templates に基づいてアプリケーションディレクトリやその中のファイルを作成している。
- アプリケーションテンプレートをよういすることで、新しいアプリケーションタイプを定義できる。
- Base には以下のファイルが含まれている。
- <base>/templates/makeBaseApp/top/
  Makefile
  configure
- exampleApp
  exampleBoot
  simpleApp
  simpleBoot
- テンプレートファイルはmakeBaseApp.plの実行時にその中のマクロが置換される。
- “makeBaseApp.pl -l”でインストールされているテンプレートのリストが表示される。

iocsh

- Iocsh
  - EPICS database/CA server program：EPICS IOCの実体
  - st.cmd : iocshの起動スクリプト
    - Database のダウンロードなどの初期設定
      - dbLoadDatabase/dbLoadTemplate
    - シークエンサープログラムの起動
      - seq sncExample
    - iocの起動 : iocInit
- iocshを起動するには、次のコマンドを使う

  <top>/bin/<arch>/<application> st.cmd
- あるいは、

  chmod +x st.cmd
  ./st.cmd
  とする。”chmod +x st.cmd”は一度実行しておけば次回からは省略できる。

iocshのデバグのために

- iocshのコマンドのヘルプ:

  epics> help
  epics> help <command>

- よく使うコマンド

  epics> thread
  epics> dbl
  epics> dbpr “<record_name>”
  epics> scanppl
  epics> dbgf “<channel name>”
  epics> dbpf “<channel name>”, <value>
  epics> …

ホスト側でのテストのためのコマンド

- base/bin/<arch>
  - caConnTest
    usage: caConnTest < channel name > [< count >] [< delay sec >]
  - caEventRate
    usage: caEventRate < PV name > [subscription count]
  - ca_test
    usage: ca_test <PV name> [optional value to be written]
  - caget
  - camonitor
  - caput
  - catime
  - cainfo
- これらのコマンドは主にテスト用に使われる。定常的な運転に使われるアプリケーションはこれらのコマンドは使わないことが望ましい。
  - これらのコマンドは実行毎にSearch-Open-get/put/monitor-Closeを繰り返す。これは非効率であり、またネットワークに余分な負荷となるので。

Handsonの準備

- CDでPCをブートする
- ログイン画面がでたら、sluserとしてログインする。
- X–terminalを開く

IOCshを起動してみる

- cd ~/EpicsADE
- makeBaseApp.pl -t example <Application Name>
- makeBaseApp.pl -i -t example <Application Name>
- make
- Run Iocsh:
  - do:
    - cd iocBoot/ioc<Application Name>
    - ../../bin/linux-x86/<Application Name> st.cmd
  - or
    - cd iocBoot/ioc<Application Name>
    - chmod +x ./st.cmd
    - ./st.cmd

IOC コマンドを試す。

- In inocsh try:
  起動したiocshのプロンプト(epics>)で以下のコマンドを試してみる。
  - help
  - dbl
  - dbpr <your channel name>
  - scanppl
    - Scan Periodic Process List. scaniol, scanpelもある。
  - thread
    値が更新されないとき,threadの
- In Unix shell try:
  新しいterminal windowを開いて
  - caget
  - cainfo <your channel name>
  - camonitor <your channel name>
  - caput <your channel name> <new value>