

序

kameはRubyスクリプトによる自動化に対応した、C++で書かれたマルチスレッドで（ドライバを書けば）どんな測定にも対応できる測定プログラムです。

自分の固体NMRを用いた物性研究に使ってますが、特にNMR専用という訳ではなく汎用です。対応している測定器の組み合わせならばプログラムの変更が必要ないようになっています。グラフ表示がOpenGLにより速いこと、測定をRubyスクリプトで自動化できるのが特徴です。クロスプラットフォーム(Linux/Mac OS X/Windows)です。

ライセンスはオープンソースのGPL(GNU public licence) version 2 or laterです。

特徴

一般

マルチスレッド（同期にソフトウェアランザクショナルメモリ使用）

OpenGLによる高速なグラフ描画

スカラー量（温度、電圧等）の組み合わせは任意にグラフ化可能

ほぼすべての設定等の保存／読み込みが可能

Rubyスクリプトによりほぼすべて制御可能

通信エラー等は例外捕捉により安全に処理

装置からの取得データを全記録／後日再解析可能

NMR部分

National Instruments社のDAQデバイスをパルサー／オシロとすることができる

FFT/MUSIC/MEM等によるリアルタイムスペクトラム解析

緩和曲線(T_1 , T_2 , $T_{st.e.}$)もリアルタイムフィット

フーリエステップサムによる磁場／周波数スイープスペクトル測定

緩和・スペクトル測定において測定後でも窓関数（畳み込み）選択可能

必要ライブラリ	5
実行時	5
Mac OS Xへのインストール	5
ソースコードからインストールする	5
Linux PCへのインストール	6
準備	6
Fedoraのインストール	6
kameのインストール	9
バイナリからインストールする場合（簡単）	9
ソースコード(<i>tar</i> ボール)からインストールする場合	10
Windowsへのインストール	10
バイナリからインストールする場合（簡単）	10
ソースコード(<i>zip</i>)からインストールする場合	11
基本操作	13
テキストボックスの操作	13
グラフ・チャートの操作	14
コマンドラインオプション	14
--logging	14
--nooverpaint	14
--moduledir <path>	14
メニュー項目	15
ファイル	15
開く	15
保存	15
閉じる	15
ログを取る	15
終了	15

測定	15
停止	15
スクリプト	15
開始	15
新規ラインシエル	15
<i>Graphviz .dot</i> を保存	15
表示	15
ヘルプ	16
メッセージウィンドウ	16
タブ	16
ドライバ	16
グラフ	17
キャリブレーションテーブル	17
<i>Node Browser</i>	18
インターフェース	18
スカラエントリ	19
<i>Raw</i> ストリームリーダー	19
DCソース	20
デジタルマルチメータ(DMM)	20
デジタルストレージオシロスコープ(DSO)	21
ファンクションジェネレータ	21
液面計	22
ロックインアンプ/容量計	22
超伝導マグネット電源	22
標準信号発生器 (SG)	24
ネットワークアナライザー	24

温度計／温調器	25
カウンター	26
流量制御	26
モーター制御	26
ターボ分子ポンプ制御	27
PPMS制御	27
NMRパルサー	28
電流反転抵抗測定	29
NMR FID/エコー測定	30
NMR緩和率測定	31
NMR周波数掃引測定	32
NMR磁場掃引測定	33
NMR自動LCチューナー	34
例（各温度でスペクトラムを保存する）	35
LinuxでDAQmxを使っていて、頻繁にフリーズするときは？	36
GPIBで通信エラーが起こるときは？	36
シリアルポートの設定は？	36
シリアルポートが開けない時は？	36
kameで対応していない機器を使いたい時は？	37
NI製DAQmxデバイスを用いたNMRシステム(パルサー及びオシロスコープ ／アベレージャ)	37
Sシリーズデバイス	38
Mシリーズデバイスのパルサー用接続	38

必要ライブラリ

以下に必要なライブラリ等を列挙します。

実行時

kameはQtソフトウェアなので、Qtが必須です。

Ruby (スクリプト系のオブジェクト指向言語。測定の自動化と設定の保存/読み込みに使います。Macでは最初から入ってます。)

<http://www.ruby-lang.org/ja/>

GSL(GNU Scientific Library。非線形フィットと補完に使ってます。)

<http://www.gnu.org/software/gsl/>

FFTW(ver.3系列。FFTライブラリ。)

<http://www.fftw.org/>

linux-gpib(LinuxでGPIBを使用する場合に必要。)

<http://linux-gpib.sourceforge.net/>

National Instruments NI-488.2ドライバ(NIのGPIBドライバを使用する場合。)

National Instruments DAQmx (DAQデバイスを使用する場合のみ必要。)

<http://www.ni.com/dataacquisition/ja/nidaqmx.htm>

libtool-ltdl (Linux/Macのみ。GNU libtool dynamic module loader。起動時にモジュールを読み込む為に必要。)

libusb (Linux/Macのみ。OceanOptics/ThamwayのUSB機器の制御に必要。)

libdc1394 (Macのみ。Firewire接続カメラを使用する場合。)

Euresys egrabber (grablink/coaxlink接続カメラを使用する場合。)

zlib (Rawデータの圧縮用)

インストール

kameプログラム本体等は、以下のURLでダウンロードできます。

<https://kitag.issp.u-tokyo.ac.jp/>

[%e8%87%aa%e5%8b%95%e5%8c%96%e5%af%be%e5%bf%9c%e6%b8%ac%e5%ae%9a%e3%83%97%e3%83%ad%e3%82%b0%e3%83%a9%e3%83%a0kame/](https://kitag.issp.u-tokyo.ac.jp/~kitag/kame/)

Mac OS Xへのインストール

ソースコードからインストールする

SequiaとQt 6.8+XCode+XCode commandline tools+MacPortsで確認しています。MacPorts版のQtは入れないで下さい。

MacPorts及び、fftw-3, gsl, libtool, libusb, eigen3, (libdc1394)をインストールして下さい。qmake用に付属の.proファイルは標準の/opt/local以下にMacPortsが存在すると仮定しています。ユニバーサルバイナリでビルドする場合はfftw-3以外を+universalバリエーションで、fftw-3は+universal +clang13 -gfortranでインストールして下さい。

Qtの6以降を、MacPortsでなく本家の物をインストールして下さい。インストール時にQt 5 Compatibility Moduleを加えてください。

ThamwayのUSB機器を使用する際は、ビルド前にソースコード内のmodules/nmr/thamwayフォルダにfx2fw.bix, slow_dat.bin, fullspec_dat.binをコピーして下さい。

必要であればEuresys eGrabberをインストールして下さい。apple siliconではdext版で十分です。

Qt Creatorからkame.proを開き、ビルドして下さい。問題があれば、プロジェクトのビルドの構成でPATHを/opt/local/binにも通して下さい。

QMakeの最中にxcodebuildがどうこう言われる場合は、以下を参照：

<http://stackoverflow.com/questions/33728905/qt-creator-project-error-xcode-not-set-up-properly-you-may-need-to-confirm-t>

OS Xアップグレード後に問題が生じれば、Qt、XCode, commandline toolsを入れ直す必要があります。XCodeインストール後は、XCodeを起動してライセンス確認して下さい。commandline toolsは、xcode-select --installでインストールできます。

(Intel CPUの場合) Qt Creatorの「プロジェクトの実行設定」では、「DYLD_LIBRARY_PATH…追加する」のチェックボックスを外して下さい。そうしないと起動しません。

NI488.2の使用には以下の対応が必要なようです。そうでないと、kame起動時にクラッシュします。
<https://www.ni.com/ja-jp/support/documentation/bugs/22/ni-488-2-21-5-known-issues.html#>

(apple siliconの場合)Qt Creatorは8以降のものがベターです。Digilent/NI488.2の使用の際には拡張機能のセキュリティーポリシー変更が必要ですが、2022初頭現在では動作しないかもしれません

Linux PCへのインストール

(注：現在はサポートしていません)動作は32/64bitのx86版Fedora 14及び15で確認しています。

ただし、National Instruments社のDAQmxドライバを利用する場合は32bit版のFedora 14での動作を確認しています。

準備

以下では、kameの動作だけでなくコンパイルも可能な環境の構築を説明します。

Fedoraのインストール

DVDからのインストールをお勧めします。「KDEソフトウェア開発環境」等を選択して下さい。「Fedora Eclipse」もあると便利です。

/boot,/,/homeパーティションを分離してかつ空き領域を残しておくことで複数のLinuxを起動する際に便利です。例えば、別バージョンのFedoraをインストールする際に、空き領域を/にして、/bootと/homeを元のものと同じにすれば良いからです。ただし、その前に/boot/grub/grub.confの内容を写しておきましょう。更に言うなら、/,/homeはLVMにしておいた方が良いでしょう。

Windowsを残しておく場合は、あらかじめWindows側で領域を縮小してからFedoraをインストールすればいいですが、その時に、Windowsパーティション（通常最初の2つ）を残して残り領域にインストールしないとイケないです。

以下の作業の前に「ソフトウェアの更新」を行なって下さい。

OpenGLがきちんと動く環境を用意して下さい。(Fedora 15まで)nVidia製GPUの場合は、nVidiaのホームページからLinuxドライバをダウンロードしてインストールして下さい。その場合、kernelの更新ごとにインストールしなおす必要があります。

ライブラリのインストール

入っていない場合は、「ソフトウェアの追加/削除」から以下のパッケージを選択して下さい。もしくは、コマンドライン(su)でyum installして下さい。

gsl-devel, fftw-devel, atlas-sse2-devel, libgfortran, ruby, ruby-devel, kdelibs-devel, kernel-devel, libtool-ltdl-devel, rpmdevtools

以下のパッケージもあると便利です。

ccache, redhat-rpm-config

コンパイルする人は、一般ユーザでrpmbuildコマンドを利用できるようにするために、コマンドライン（一般ユーザ）でrpmdev-setuptreeを実行しましょう。そうすると~/rpmbuildフォルダにRPMパッケージが作られます。

Linux GPIBドライバのインストール

Linuxでは、National Instruments社のNI-488.2ドライバも利用可能ですが、私は確認してません。

kameと同じサイトに、Fedora用のRPMパッケージ及びSRPMパッケージを置いてあります。

詳細は、本家のドキュメントを参照

http://linux-gpib.sourceforge.net/doc_html/index.html

インストール

バイナリをインストールする場合（簡単）

linux-gpibと、kmod-linux-gpibのRPMをインストールします。コマンドラインでは、suでrpm -ivh等を使います。アップグレード時はrpm -Uvhです。

kmod-linux-gpibでは、kernelのバージョン(`uname -r`で確認できます)と一致していなければ意味有りません。なので、kernelは安易に更新しないようにしましょう。

SRPMからコンパイルしてインストールする場合

こんな感じでRPMパッケージを作りましょう

```
rpmbuild --define="kversion `uname -r`" --rebuild linux-gpib-kmod-{ver}.src.rpm
rpmbuild --rebuild linux-gpib-{ver}.src.rpm
```

あとは上記参照。

設定

/etc/gpib.confの編集

suで編集しましょう。vi /etc/gpib.confとかで。

```
board_type = "ni_pci"
```

の行を、ni_pciの部分を使っているデバイスに合わせましょう。

例えば、National Instruments社のUSB-GPIB-HSならば、

```
board_type = "ni_usb_b"
```

になります。

PCIまたはPCI ExpressのGPIBボードを使う場合

例えばこんな感じの項目を/etc/modprobe.d/modprobe.conf.localに追加する必要があります。

```
alias char-major-160 gpib_common
```

```
alias gpib0 tnt4882
```

```
install tnt4882 PATH=/sbin:/usr/sbin:/usr/local/sbin:$PATH;modprobe --ignore-install
```

```
tnt4882;sleep 3;gpib_config --minor 0
```

USBのGPIBデバイスを使う場合

余計な設定は不要ですが、調子が悪ければ、suで、

```
/usr/sbin/gpib_config --minor 0
```

を打つと良いかも。

(必要な場合のみ) National Instruments社のDAQmxドライバのインストール

DAQmx8.0.2のISOをダウンロードして、中身を適当な展開しましょう。

とりあえずインストールします。suで、

```
cd (NIDAQ802 directory)
```

```
chmod +x INSTALL
```

```
LANG=C ./INSTALL
```

```
/usr/local/bin/updateNIdrivers
```

最後のコマンドは、kernelを更新する度に実行する必要があります。

RTSIの設定 (複数のPCI/PCIe DAQデバイスを同期して使用する場合)

以下を参照。

<http://zone.ni.com/devzone/cda/tut/p/id/4620>

```
nidaqmxconfig --export daq.config
```

で設定内容を書き出します。そしてdaq.configに以下の内容を足します。

```
[DAQmxRTSICable RTSICable0]
```

```
RTSI.ConnDevs=Dev1,Dev2
```

設定内容を読み込みます。

```
nidaqmxconfig --import daq.config
```

上手く行っていれば、"Operation Successful"と出ます。

SE Linuxのポリシー設定

設定ユーティリティで許容 (permissive) にしてしまうか、

allow_execstackをチェックしましょう。

GUIでは、「管理」→「SELinux Management」で行います。system-config-selinuxがインストールされている必要があります。

kernelコマンドラインの設定

/boot/grub/grub.confをsuで編集します。

kernelのラインの最後に、以下を足します。

```
iommu=off vmalloc=256M
```

これは、IO仮想化のIOMMUをOFFにして、kernel用のメモリを増やします。

メモリを4GB超積んでいる場合は、さらに以下を足します。

```
mem=4096M
```

再起動しましょう。

DAQmxドライバは本当に困ったもんですね。

kameのインストール

バイナリからインストールする場合 (簡単)

suでrpmコマンドを使ってkameとkame-modules-standardをインストールします。kame-modules-nmrをインストールすれば、NMRモジュール、kame-modules-nidaqmxをインストールすれば、National Instruments社のDAQデバイスを使用するモジュールも使えるようになります。

kame-debuginfoをインストールすると、不運にもクラッシュした時にデバッガを用いて調査ができるようになります。

ソースコード(tarボール)からインストールする場合

RPMを作る場合

tools/mkrpm.shを参考にしてください。

DAQmxモジュールが不要であれば、rpmbuildコマンドに
--define="build_nidaqmx 0"

を追加して下さい。

パッケージを作らない場合

build用のディレクトリを作ります。

```
そのディレクトリにcdして、  
cmake ../(sourceディレクトリ)  
make  
make install
```

でインストールできますが、パッケージと混ざるとややこしいので推奨できません。

シリアルポートとlinux-gpibの為に、udevの設定が必要です。

Windowsへのインストール

バイナリからインストールする場合（簡単）

64bit版の場合

Qtの6.5以上(MinGW64bit構成を含む物、複数のバージョンを含まないこと)を**Qt5 compatibility support**を含めてインストールして下さい。

<https://www.qt.io/jp/>

複数バージョンのQtがあって起動に失敗する場合は、qtdir.txtの中で6.5以上のもののみを残して削除して下さい。

GPIBを使用する場合は、

National Instrumentsの488.2ドライバを（ANSI Cサポートやダイレクトエントリを外さないで）インストール

DAQmxを使用する場合は、

National InstrumentsのDAQmxドライバを（ANSI Cサポートを外さないで）インストールして下さい。

kame-win32-64bit***.zipを展開し、kame.batから起動して下さい。

ThamwayのUSB機器を使用する際は、実行フォルダにfx2fw.bix, slow_dat.bin, fullspec_dat.binをコピーして下さい。

32bit版の場合

Qtの5.7(MinGW32bit構成を含む物、複数のバージョンを含まないこと)をまずインストールして下さい。

<https://www.qt.io/jp/>

複数バージョンのQtがあって起動に失敗する場合は、qtdir.txtの中で5.7移行の行のみを残して削除して下さい。

GPIBを使用する場合は、National Instrumentsの488.2ドライバを（ANSI Cサポートやダイレクトエントリを外さないで）インストール

DAQmxを使用する場合は、National InstrumentsのDAQmxドライバを（ANSI Cサポートを外さないで）インストールして下さい。

kame-4**-win32.zipを展開し、kame.batから起動して下さい。

OpenGL2.1に対応していない古いマシンではグラフが正常に表示できません。古いintel内蔵グラフィックスマシンでkameが起動時かグラフを開く時に落ちる場合は、グラフィックドライバを更新した上で、設定画面で「トリプルバッファ」をオフにして「深度バッファ」を16bitにして下さい。

ThamwayのUSB機器を使用する際は、実行フォルダにfx2fw.bix, slow_dat.bin, fullspec_dat.binをコピーして下さい。

ソースコード(zip)からインストールする場合

上記のバイナリが動作する状態にまずして下さい。

64bit版の場合

次に、いくつかのライブラリを準備する必要があります。

msys2をデフォルトのc:/msys64にインストールしてください。

まず、アップデートします。

```
pacman -Syuu
```

その後、以下をインストールしてください。

```
pacman -S make
pacman -S mingw-w64-x86_64-zlib
pacman -S mingw-w64-x86_64-fftw
pacman -S mingw-w64-x86_64-gsl
pacman -S mingw-w64-x86_64-eigen3
```

ruby-2.8から3.2まではgccと相性が悪いため、自分でruby-2.6.10をコンパイルします。

ruby-2.6.*のソースを一つ上のディレクトリにrubyの名前で解凍して下さい。その後、msysを起動し、rubyのソースディレクトリにおいてmingwにパスを通し(PATH=\$PATH:/c/Qt-(適宜)/Tools/mingw(適宜)_64/binなど)、

```
CFLAGS="-g0" ./configure --enable-shared --disable-rubygems
make
```

でRubyのDLLを作ります。途中でエラーが出てもDLLができていればOKです

Qt Creatorからkame.proを開き、mingw64構成でビルドして下さい。gitからcloneする場合は、autoCRLF=falseにしておいた方が無難です。

実行するには、C:\msys64\mingw64\binの中にある、zlib.dll, llibgsl.dll, libgslcblas-0.dll, ibftw3-3.dll と、rubyフォルダの中のx86-msvc*-ruby*.dllを実行ファイルのあるフォルダ（通常はbuild-kame-***-Debug/Release）にコピーしてから、Qt Creatorから実行するかデバッグを開始して下さい。

関西弁にするには、kame_ja.qmを実行ファイルのあるフォルダにコピーしなければいけません。また、実行ファイルのあるフォルダにResourcesという名前のフォルダを作り、kame/script/rubylineshell.rbをコピーしてこなければ、ラインシェルが使いません。

32bit版の場合

次に、いくつかのライブラリを準備する必要があります。

fftw-3.3.4-dll32.zip（同様にkameのソースディレクトリの一つ上に、ディレクトリ名fftw3で展開）
<http://www.fftw.org/download.html>

Ruby及びmsys (<http://www.mingw.org/>) を適当なところからダウンロード、インストールし、rubyにパスを通して下さい。また、ruby-2.2.*のソースを一つ上のディレクトリにrubyの名前で解凍して下さい。その後、msys.batを起動し、rubyのソースディレクトリにおいてmingwにパスを通し(PATH=\$PATH:/c/QT/Qt-(適宜)/Tools/mingw4(適宜)_32/binなど)、
CFLAGS="-g0 " ./configure --enable-shared --disable-rubygems
make
でRubyのDLLを作ります。途中でエラーが出ててもDLLができていればOKです。

zlibに関しては、以下から"Compiled DLL"をダウンロードし、一つ上のディレクトリにzlibの名前で解凍して下さい。
<http://www.zlib.net/>

GSLに関しては新しいバージョンが必要なため、自身でコンパイルする必要があります。

<http://www.gnu.org/software/gsl/>

からgsl-2.4.tar.gzをダウンロードし、一つ上のディレクトリにgslの名前で解凍して下さい。上記Rubyの場合と同じように./configure;makeして下さい。その後、./libs/libgsl-23.dll及びcblas/.libs/libgsl-cblas-23.dllを実行フォルダにコピーして下さい。

Qt Creatorからkame.proを開き、mingw32構成でビルドして下さい。gitからcloneする場合は、autoCRLF=falseにしておいた方が無難です。

実行するには、zlib1.dll(C:\Program Files*\GnuWin32\binの中), llibgsl.dll, libgslcblas.dll (gslの中), ibftw3-3.dll (FFTW3の中), msvc*-ruby*.dll(rubyの中)を実行ファイルのあるフォルダ（通常はbuild-kame-***-Debug）にコピーしてから、Qt Creatorから実行するかデバッグを開始して下さい。

関西弁にするには、kame_ja.qmを実行ファイルのあるフォルダにコピーしなければいけません。また、実行ファイルのあるフォルダにResourcesという名前のフォルダを作り、kame/script/rubylineshell.rbをコピーしてこなければ、ラインシェルが使いません。

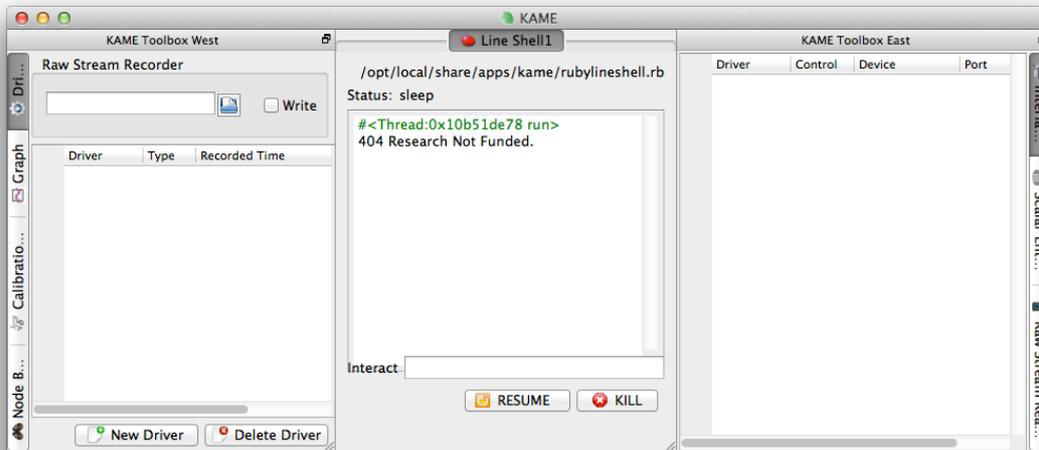
実行

Linuxではコマンドラインでkameと打つのがオススメです。

通常は関西弁で、LANG=C kameとやると英語で起動します。

エラー等のログが/tmp/kame.log (windowsでは実行フォルダ)に残りますので、問題が起これば確認してみてください。

基本操作



(初回) 「ドライバ」タブ中で測定器に対応するドライバを「新規」で追加。

(次回から) 「ファイル」→「開く」で設定を保存した.kamファイルを読み込む。

「インターフェース」タブのなかでドライバを「開始」。GPIB/シリアルポート等であればアドレス(ポート)を指定しておく(以下の章を参照)。

「ドライバ」タブまたは「インターフェース」タブでドライバ名等をクリックするとドライバ固有の設定画面が開く。(一部のドライバには設定画面はありません。)

「スカラエントリ」。このタブは、時系列でデータファイルにいくつかの測定値を保存するのに使用する。例えば、温度と電圧とか。「デルタ」を正にすると、変化分がその値を超えたときに一行出力する。負にすると、新しい値が読み込まれる度に一行出力する。該当する値をクリックすると、横軸が時間のチャートが現れる。

「グラフ」。X-Y(Z)のグラフを必要なだけ作れる。値は「スカラエントリ」から選ぶ。

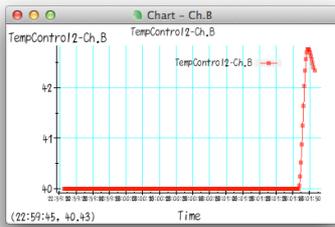
「ファイル」→「保存」とすると、ほとんどの設定を.kamファイルに保存します。随時行いましょう。

テキストボックスの操作

打ち間違いや装置への悪影響を防ぐため、ほとんどの設定は、エンターキーを打つまで反映されません。編集中は文字が青色になります。間違った形式で打つと赤色になります。エンターキーを打たずに別の場所を選択すると、入力結果は破棄されて元の値を表示します。

殆どの場合、テキストボックス中の値は、Rubyスクリプトから操作可能です。

グラフ・チャートの操作



OpenGLにより、3次元空間に半透明で描画されています。

左ダブルクリックすると、詳細なダイアログが開きます。

右ダブルクリックすると、操作ヒントが出ます。

プロット中で左クリックしながら範囲を選択すると、オートスケールを止めて拡大します。

プロット中で右クリックすると、オートスケールに戻ります。

プロット中でホイール操作でズームします。

軸上で右クリックしながら範囲を選択するとその軸のオートスケールを止めて拡大します。

軸上で右クリックすると、その軸をオートスケールします。

軸上でホイール操作すると、3次元的に傾けます。

中ボタンを押しながら動かすと、3次元的にグリグリします。

中クリックすると、視点が元に戻ります。

保存可能なデータがある場合、上か下にファイルを示すバーがあります。フォルダアイコンをクリックすると、保存すべきファイルを選べます。ただし、その時点では保存されません。「書き出し」を押した瞬間のデータが追記で保存されます。一度書きこむとアイコンが変化します。

コマンドラインオプション

普通は不要。

--logging

通信内容等もログに残します。メニューから「ログを取る」を選んでも一緒です。

—nooverpaint

グラフのフォント描画に2D描画でなく、renderText()を使います。

--moduledir <path>

モジュールを読み込むパスを追加します。ソースコードからインストールした場合に便利です。

メニュー項目

ファイル

開く

.kamファイルを読み込みます。中身はRubyスクリプトになっています。従って、スクリプト実行画面が中央部に開き、エラーが出れば赤く表示されます。

保存

現在開いている全ドライバや他の一部の設定を全て.kamファイルに保存します。内部的には、Rubyからコントロール可能なノードの内、保存属性が設定されているもののスナップショットが保存されます。

閉じる

全ドライバ、キャリブレーションテーブルが削除されます。誤って押さないようにしましょう。

ログを取る

測定器との通信ログ、ノードの追加/削除情報も/tmp/kame.log（Windowsでは実行フォルダのkame.log）に保存されます。ディスクを圧迫するので、不必要な時は止めましょう。

終了

測定中はキャンセルされます。保存を忘れないようにしましょう。

測定

停止

全測定を停止させます。うっかり押さないようにしましょう。

スクリプト

開始

自動測定のためのRubyスクリプトを読み込みます。標準では.seqファイルです。書き方は別章で説明します。標準（エラー）出力結果は、同フォルダの（ファイル名）.logに保存されます。

新規ラインシェ尔

Rubyスクリプトを一行ごとに打ち込める画面を開きます。起動時に既に1つ開いています。

Graphviz .dotを保存

生成した.dotファイルをgraphvizで処理すれば、ノードツリーをグラフィカルに把握できます。私も使わない機能です。

表示

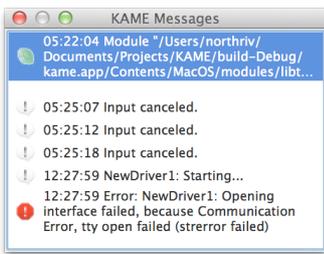
タブをクリックしても同じです。

ヘルプ

用意してません。このマニュアルを見て下さい。

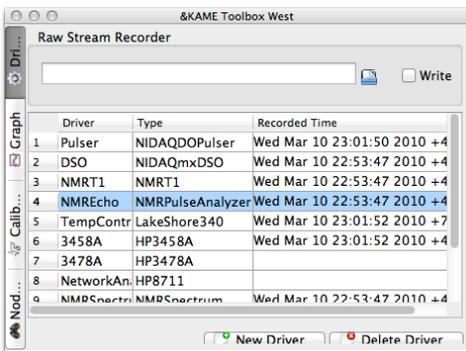
メッセージウィンドウ

起動時から左端にあります。主要なメッセージ、警告やエラーが表示されます。もう少し細かい情報は標準出力と/tmp/kame.log（Windowsでは実行フォルダのkame.log）に記録されます。



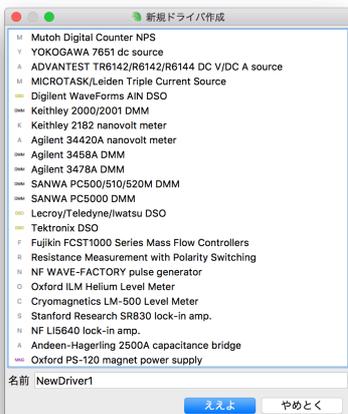
タブ

ドライバ



ドライバの追加・削除が出来ます。また、ドライバー一覧上でクリックするとドライバ固有の設定ウィンドウが出ます。レコード時刻は、最後に測定器から記録に値するデータが来た時間を表示します。

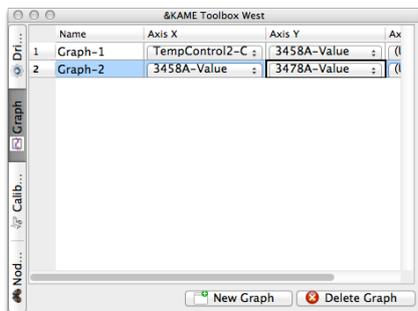
新規ドライバを押すと次のようなウィンドウが出て、必要なドライバに名前を付けて作成できます。名前は後から変更は難しいので、考えてつけて下さい。



Rawストリームレコーダ

生データを極力保存します。「書き込み」チェックボックスが入っている間、例えば、オシロスコープの波形が全て記録されます。GZIP形式で圧縮されますが、当然それなりのディスク容量を消費します。

グラフ

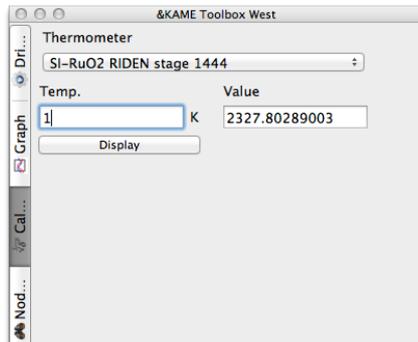


2次元／3次元グラフを生成します。X/Y/Z軸を「スカラエントリ」の中の項目から選んで下さい。

設定点数を超えた古いデータはグラフから消されます。

“Live”はメモリにレコードされた全データ、“Stored”は「テキストライタ」でディスクに記録されたデータを表示します。

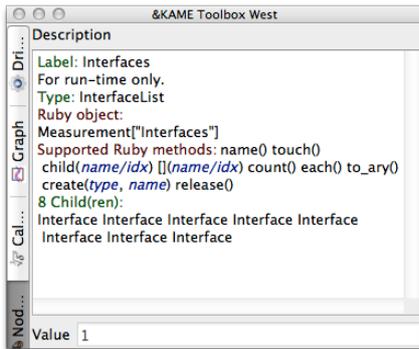
キャリブレーションテーブル



温度計の校正データを定義するのに使います。定義ファイルは今のところ、手動でRubyスクリプトを書く必要があります。サンプルがkame/Measurementsフォルダにあります。スプライン曲線による補完、またはチェビシェフ多項式による定義が可能です。

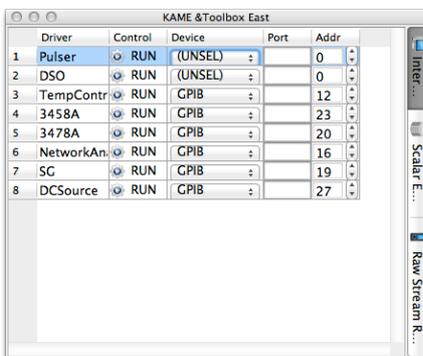
「表示」を押すと、校正曲線を表示します。

Node Browser



kameの中のデータは全てツリー構造に管理されています。Rubyスクリプトで制御する際、`"Measurement["Drivers"]["driver1"]["foo"].value=1.0`という感じで行います。画面上の入力部がRuby上でどう扱われるか表示します。スクリプトのサンプルは、kame/Sequencesフォルダにあります。

インターフェース



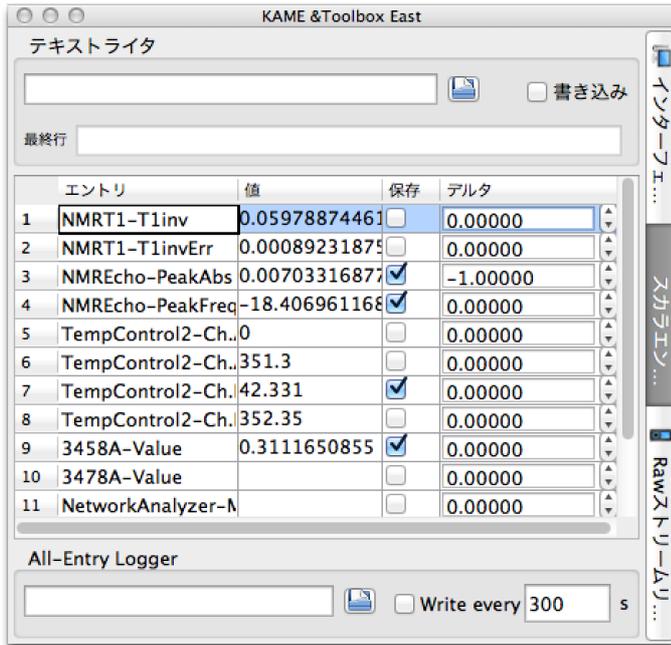
ドライバが測定器との通信経路を必要としている場合、ここに設定項目及び、起動/停止ボタンが出ます。通常は一ドライバに対し一つです。「デバイス」、「ポート」、「アドレス」をそれぞれ正しく設定して下さい。

GPIBの場合、ボードが一つであれば、「ポート」は空白で構いません。

シリアルポートの場合、RS-232Cではアドレスは不要ですが、ポートは/dev/ttyUSB0(Linux),/dev/tty.usbserial(Mac)等のデバイスファイルを指定します。RS-485ではスレーブのアドレスを設定して下さい。

TCP/IPの場合、「ポート」に(IPアドレス):(ポート番号)を設定して下さい。

スカラエントリ



ドライバからレコードされた結果の数値が一覧に出ます。

該当する値をクリックすると、横軸が時間のチャートが現れる。チャートの操作は、横軸をオートスケール出来ない以外はグラフと同じです。

テキストライタ

テキストファイルに一行ごとにスペース区切りで、「保存」を指定された値を出力します。改行はLFです。「書き込み」をチェックしないと出力しません。

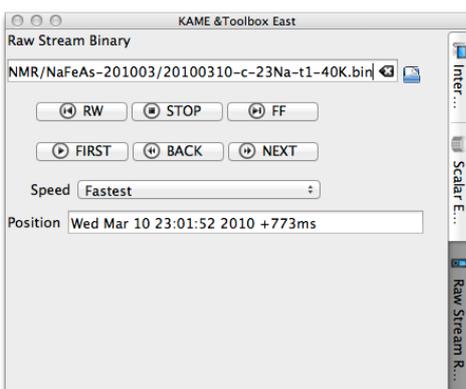
どのタイミングで一行が出力されるかは、「デルタ」で指定します。「デルタ」を正にすると、変化分がその値を超えたときに一行出力します。負にすると、その項目に新しい値が読み込まれる度に一行出力します。

項目の順番を変えるには、番号の上でCTRLキーを押しながらドラッグします。

ロガー

テキストファイルに一行ごとにスペース区切りで、指定した間隔で全ての値を出力します。

Rawストリームリーダー



上記のRawストリームレコーダで出力したファイルを読み込んで解析できます。巻戻し機能はバグっているので、何回も解析する場合は、「最初」に戻ってから「送り」で早送りして下さい。早送り速度は「速度」で指定します。「位置」はレコードが記録された時刻を表示しています。

ドライバ固有の設定

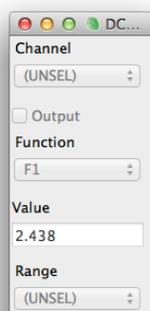
DCソース

YOKOGAWA 7651 DC source (GPIB)

ADVANTEST TR6144/R6142/R6144 DC source (GPIB)

MICROTASK/Leiden Triple Current Source (GPIB)

7651を使う場合は、他の装置よりも先に開始したほうが無難です。



デジタルマルチメータ (DMM)

Keithley 2000/2001 DMM (GPIB)

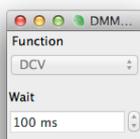
Keithley 2851A nanovolt meter (GPIB)

HP/Agilent 34420A nanovolt meter (GPIB)

HP/Agilent 3458A/3478A DMM (GPIB)

Keithley 6482 picoammeter (GPIB)

SANWA PC500/510/520M/PC5000 DMM (IR SerialPort)



「ウェイト」は、読み込み周期を設定します。

スカラエントリに値が送られます。

デジタルストレージオシロスコープ(DSO)

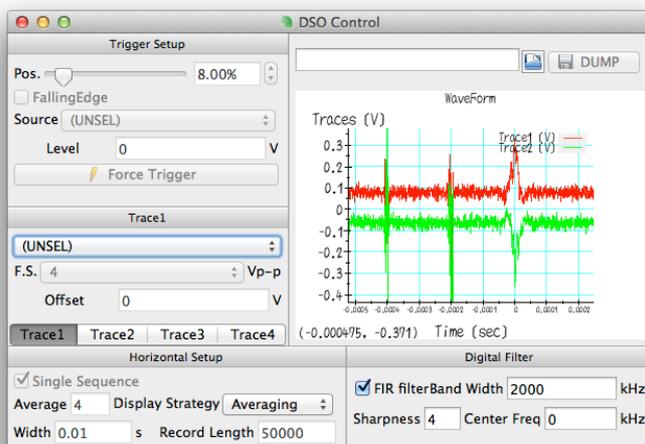
Tektronix DSO (GPIB)

Lecroy/Iwatsu DSO (GPIB)

DSO on NI-DAQ M,S series (NI-DAQmx)

Thamway A/D conversion DV14U25 (USB)

Digilent WaveForms AIN DSO (USB)



オシロスコープの/DIVでなく、縦軸はフルスケール「F.S.」で、水平軸は全長の「幅」で設定して下さい。

PCに連続取り込みするドライバでは、時間同期の取れているトリガ源（NMRパルサーなど）がソフトウェア的にトリガ位置を計算します。

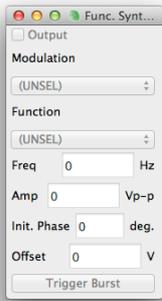
「シングルシーケンス」にすると、「アベレージ」回数分積算してはクリアするの繰り返しになります。

「表示の方法」を「Averaging」にすると適宜表示するので便利ですが、GPIBやCPUに大きな負担をかけます。「シングルシーケンス」モードで「Sequence」にすると指定積算回数に達した時にだけ表示します。

「デジタルフィルタ」、FIR(Finite Impulse Response)フィルタを掛けることができます。「中心周波数」(fc)から「バンド幅」(bw)で指定した帯域をそれぞれのチャンネルに掛けるので、 $fc-bw/2$ から $fc+bw/2$ の周波数を通すようになります。FIRは畳込みで高速に処理されます。

ファンクションジェネレータ

NF WAVE-FACTORY (GPIB)



液面計

Oxford ILM Helium levelmeter (GPIB, SerialPort)

Cryomagnetics LM-500 levelmeter (GPIB)

設定画面はありません。

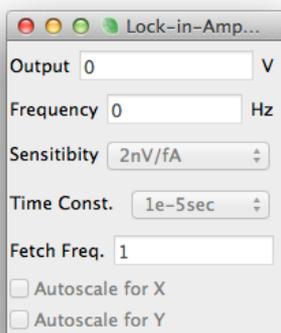
スカラエントリに値が送られます。

ロックインアンプ／容量計

Stanford Resrearch SR830 lock-in amplifier (GPIB)

NF LI5640 lock-in amplifier (GPIB)

Andeen-Hagerling 2500A capacitance bridge (GPIB)



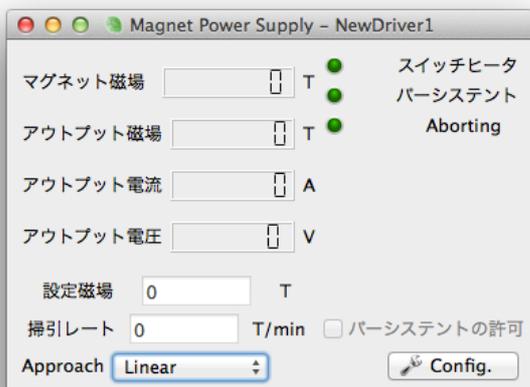
「取り込み頻度」はタイムコンスタントの逆数を基準にしています。

スカラエントリにX/Y値が送られます。

超伝導マグネット電源

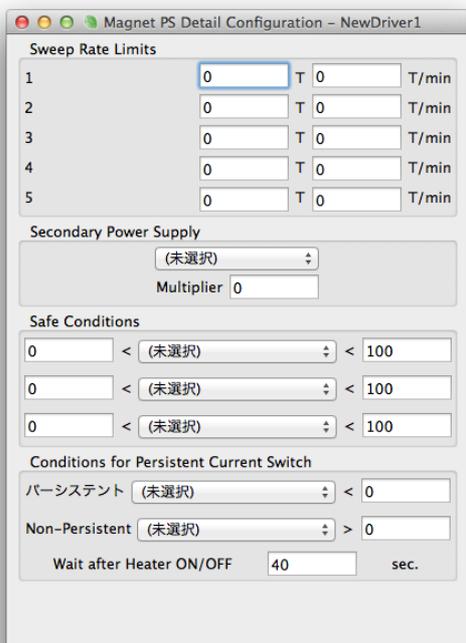
Oxford PS/IPS-120 magnet power supply (GPIB, SerialPort)

Cryogenic SMS10/30/120/150C magnet power supply (USB Serial Port)



パーシステントスイッチヒーターの制御は自動で行います。「パーシステントの許可」をチェックすると、磁場が設定値に到達した時にヒーターを切ります。”Approach”を”Oscillating”にすると、20%オーバーシュートしながら磁場をスイープします。

「設定」ボタンを押すと次の画面が開きます。



掃引レート及び、最高磁場に制限をかける場合は、「Sweep Rate Limits」に「最大磁場」-「レート」のペアを小さい順に必要な数だけ入れて下さい。

“Secondary Power Supply”はシムコイルの駆動に使います。

“Safe Conditions”は、スカラエントリの値を監視して、範囲から外れると磁場を0にします。

同様に、パーシステントスイッチ用の条件を付加することが出来ます。

“Wait after Heater ON/OFF”に、スイッチヒーターの待ち時間を入れて下さい。

標準信号発生器 (SG)

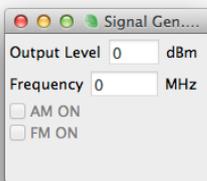
KENWOOD SG7130/SG7200 signal generator (GPIB)

DSTech. DPL-3.2XGF (SerialPort)

HP/Agilent 8643/8644/8648/8664/8665 signal generator (GPIB)

Rhode-Schwartz SML-01/02/03/SMV-03 signal generator (GPIB, SerialPort)

Thamway PROT NMR Control (TCP/IP, USB)



PROTの場合は、ゲイン等のNMR設定が制御可能です。

ネットワークアナライザー

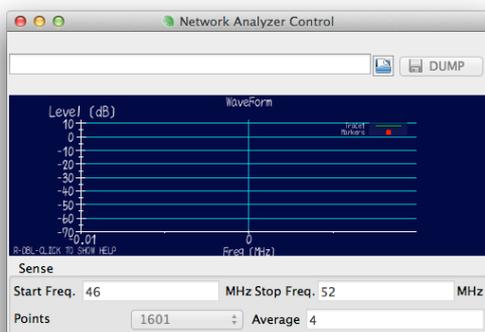
HP/Agilent 8711/8712/8713/8714 network analyzer (GPIB)

Agilent E5061/E5062 network analyzer (GPIB)

Copper Mountain TR1300/1,5048,4530 Network Analyzer (TCP/IP)

DG8SAQ VNWA3E w/ custom dll network analyzer (TCP/IP)

Thamway T300-1049A Impedance Analyzer (SerialPort)



スカラエントリにマーカーの値を出力します。

温度計／温調器

Cryocon M32/M62 temperature controller (GPIB)

LakeShore 340 temperature controller (GPIB)

LakeShore 350 temperature controller (GPIB)

LakeShore 370 temperature controller (GPIB)

Picowatt AVS-47 bridge (GPIB)

Oxford ITC-503 temperature controller (GPIB, SerialPort)

Neocera LTC-21 temperature controller (GPIB)

Keithley 2700 DMM w/ 7700 scanner (GPIB)

OMRON E5*C controller (Serial Port Modbus RTU 57600bps)



「温度計」を選択した場合、kameが抵抗値等から温度に変換します。温度計の校正曲線は「キャリブレーションテーブル」タブで確認できます。

「外部機器」を選ぶと、そのDCソースまたは流量制御弁を用いてPID制御を行います。

スカラーエントリに値（抵抗値等と温度）が送られます。

表示はされていませんが、"Stabilized"というノードに「設定温度」からの「現在の温度」のエラーを一定時間平均した値が保存されています。

カウンター

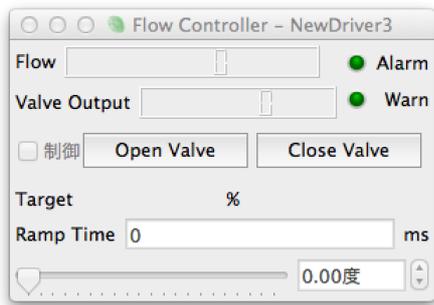
Mutoh NPS digital counter (SerialPort)

設定画面はありません。

スカラエントリに値が送られます。

流量制御

Fujikin FCST1000 Series Mass Flow Controllers (Serial Port 38400bps)



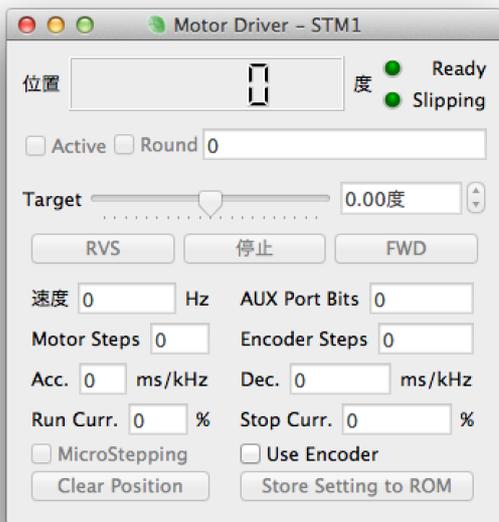
マスフローコントローラードライバです。

スカラエントリに値が送られます。

モーター制御

OrientalMotor FLEX CRK motor controller (Serial Port Modbus RTU 57600bps)

OrientalMotor FLEX AR/DG2 motor controller (Serial Port Modbus RTU 57600bps)



ステッピングモータードライバです。

「Active」で励磁のON/OFFを設定します。

スカラエントリに値が送られます。

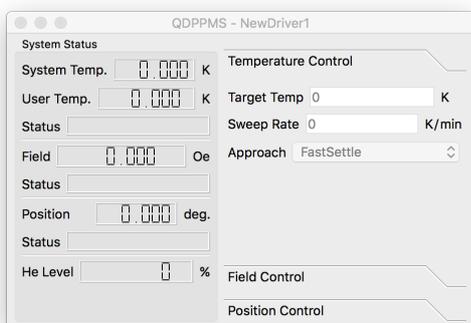
ターボ分子ポンプ制御

Pfeiffer Turbo molecular pump controller TC110 (Serial Port)

PPMS制御

Quantum Design PPMS low-level interface (GPIB, Serial Port 9600bps)

QuantumDesign社のPhysical Property Measurement System 6000コントローラを制御します。



シリアル通信の場合には、MultiVuのGPIB制御と同時に通信出来るようです。本体で、9600bps, フロー制御なし、改行コードCRLF、ビット長,パリティ,ストップビットは"8,N,1"に設定して下さい。

スカラエントリに値が送られます。

NMRパルサー

NMR pulser on NI-DAQ M series (NI-DAQmx)

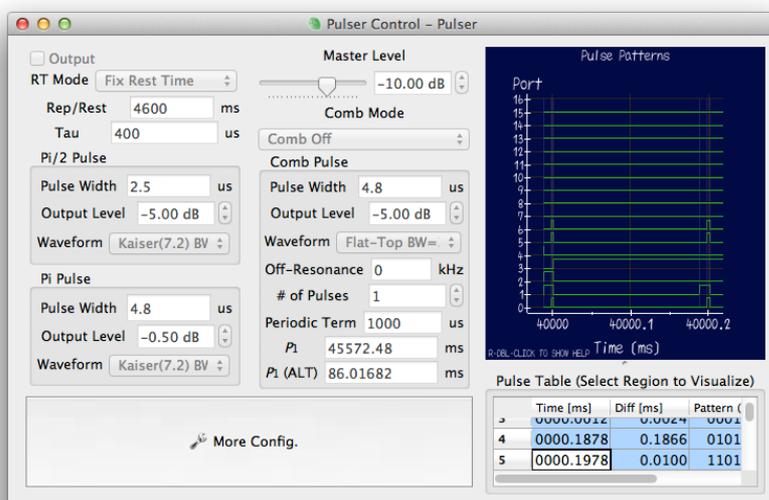
NMR handmade pulser on H8 (SerialPort)

NMR handmade pulser on SH2 (SerialPort)

NMR pulser Thamway N210-1026 PG32U40 (USB)

NMR pulser Thamway N210-1026S/T (GPIB, TCP/IP: experimental)

DAQmxについては付録も参照して下さい。



QAMに対応していない通常のパルサーの場合は、該当する部分（強度、波形等）は変更できません。

右側のグラフに、デジタル波形を表示します。表を選択すると選択範囲の波形を表示します。

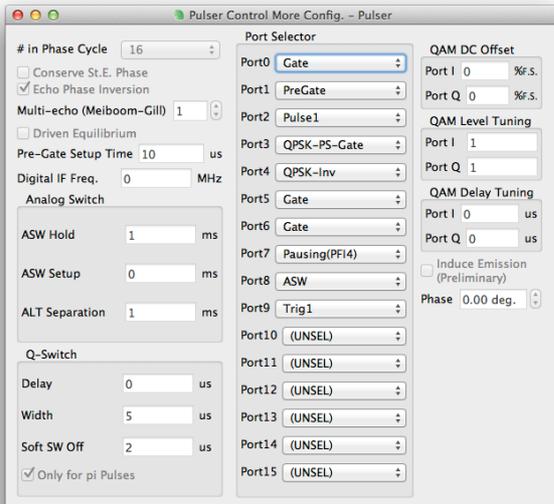
「RTモード」を”Fix Rep. Time”にすると、Saturation (Comb) Pulseまたは1st (Pi/2) Pulseの繰り返し時間を P_1 によらず”Rep/Rest”時間にします。”Fix Rest Time”の時は、 2τ または最後のパルスの中から”Rep/Rest”時間待ちます。

パルスの時間間隔等（”Tau”，” P_1 ”など）の基準はパルスの中心です。ただし、「コムパルス」にかんしては、最後のサチュレーションパルスの中心が基準です。

「コムモード」を”P1 ALT”にすると、 P_1 の値を交互に変化させます。その際、 T_1 を測定するには、後述のFID/エコー測定ドライバが2つ必要になります。”ASW”の信号を使ってレシーバーのゲーティングをする必要があります。

同様に、”Comb ALT”にすると、コムパルスあり・なしを交互に打ちます。

「もっと設定」を押すと次のような画面が出ます。



「フェーズサイクル数」は、4までは通常の $\pi/2, \pi$ パルスに関するサイクルで、それ以上は、Combパルスの位相も変化させます。

「Conserve St.E. Phase」をチェックすると、Stimulated Echo測定用のフェーズサイクルになります。Combパルスを2個に設定して、Combのパルス間隔を τ と同じに、 $\pi/2$ パルスの幅を0に、 π パルス幅をCombのパルス幅と同じに設定しましょう。

「マルチエコー」を2以上にすると、Carr-Purcell-Meiboom-Gill (CPMG)サイクルになります。CPMGでもフェーズサイクルします。FID/エコー測定のドライバでも適切な設定が必要です。

「ポートセレクタ」で、各ポートの機能を選んで下さい。

”Gate”はパルスのゲート。

”PreGate”はパワーアンプ用にセットアップ時間を設けたゲート。

”Trig1”は 2τ で立ち上がるトリガ用信号。

”Trig2”は $\pi/2$ パルスで立ち上がり π パルスで立ち下がるトリガ用信号。

”ASW”は、レシーバー用のゲート。 2τ の前後の設定期間にHレベルになります。

”QSW”はパルス直後だけHレベルになる、Qスイッチ用の出力。

”Pulse1”は $\pi/2$ パルス。

”Pulse2”は π パルス。

QPSK用のロジックは以下の通りです。

(位相) “QPSK-A” “QPSK-B” “QPSK-NonInv” “QPSK-Inv” “QPSK-PS-Gate”

0°	L	L	H	L	L
90°	H	L	H	L	H
180°	H	H	L	H	L
270°	L	H	L	H	H

ちなみに、いわゆる西のNMRグループは、“QPSK-A”、“QPSK-B”を、東のNMRグループは、“QPSK-PS-Gate”、“QPSK-Inv”のロジックを使用しています。

電流反転抵抗測定

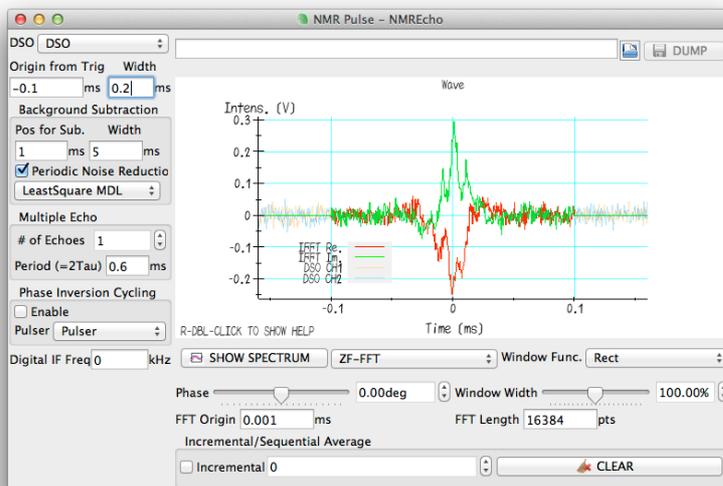
このドライバは、解析専用で、DCソースを反転させながら、DMMからの電圧データの差分を取ります。



スカラエントリに値が送られます。

NMR FID/エコー測定

このドライバは、解析専用で、DSOのデータをバックグラウンド除去し、切り出し、積算した後にスペクトラム解析します。



通常のエコーを観測する設定では、DSOのトリガーを 2τ (= "Trig1") に設定して、DSOを「シングルシーケンス」モードにして下さい。

「トリガーからの位置」を始点にして「幅」でDSOのデータから解析対象を切り出します。ただし、「マルチエコー」の場合は、さらに「エコー数」分平均化して表示します。

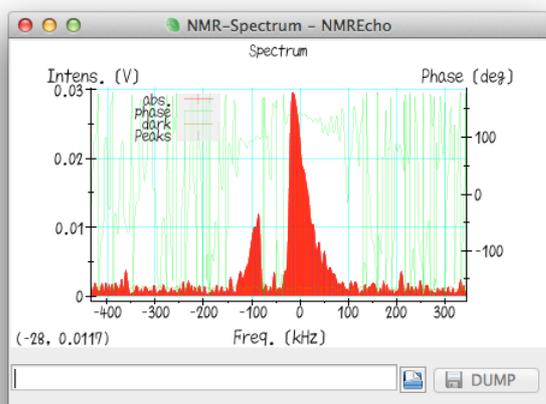
「バックグラウンド(BG)減算」の「幅」が0でない場合は、トリガーから「BG測定位置」を始点にして「幅」の平均レベルを引き算してデータを表示します。S/Nを損なわないために、減算用の「幅」は元データの「幅」よりも十分（数倍以上）長く取って下さい。

また、「周期性ノイズ除去」がチェックされている場合は、さらに、BG測定から卓越する周期性外部ノイズを幾つか推定し（最小二乗フィット）、その延長成分を減算します。ホワイトノイズかどうか判定する、情報量基準についてアルゴリズムを選択できます。この処理は、多数回のFFTを伴うので、CPUパワーを大きく必要とします。でもその価値はあります。周波数掃引の時は必ず使いましょう。

「フェーズ反転サイクル」をチェックすると、DSOで見てエコーの位相が反転する場合としない場合を交互に繰り返して差を取ります。時間依存するバックグラウンド成分を減算することができます。緩和率測定・スペクトラム測定には、偶数回数の平均の結果が引き渡されます。

「インクリメンタル」をチェックすると、一度画面がクリアされ、「消去」を押すまで積算を続けます。積算回数が表示されます。信号を探したり、FTを取る時に使うと便利です。

「スペクトラム表示」をクリックすると次の画面が出ます。



通常は、「ZF-FFT」(Zero-Filling Fast Fourier Transform)の結果が表示されます。FFTの時間軸原点は「FFT原点」でトリガー位置を基準にして入れて下さい。FFTは「窓関数」を掛けることも出来ます。

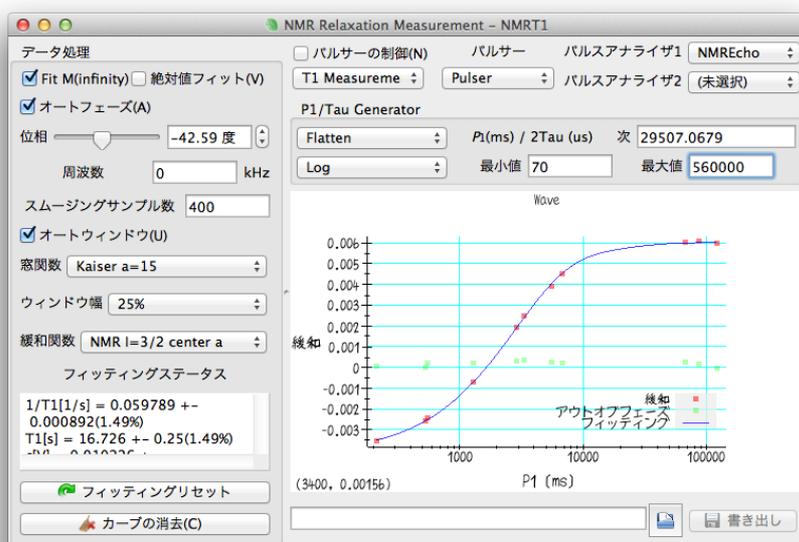
赤の部分“abs”が複素FFTでの絶対値です。FFT原点が合っていれば、“phase”はフラットになるはずで
す。“dark”はバックグラウンドから見積もられたノイズレベルです。“Peaks”は極大値を示します。

メイン画面の“IFFT Re.”/“IFFT Im.”には、スペクトラム解析の結果のIFFTが表示されています。つまり、ZF-FFTでは窓関数を掛けただけのものです。“DSO Re.”/“DSO Im.”にはDSOの波形からバックグラウンド処理をしたものが薄く表示されています。

スカラエントリには、最大値とその値をもつ周波数が送られます。

NMR緩和率測定

このドライバは、解析専用で、FID/エコー測定の結果から緩和率を測定します。パルサーの制御も行いません。FID/エコーがインクリメンタル平均になっているときは使用できません。窓関数の設定は引き継がず、結果を見ながら合わせる事ができます。



「パルサーの制御」をチェックすると、一度結果が消去され、データがレコードされる度に次のP₁等の値がパルサーにセットされます。「Random」の場合、その値は完全に乱数で決められます。「Flatten」の場合は、空いている部分を重点的に割り当てます。「P₁分散」を「Log」にすると対数スケールで見て「最小値」と「最大値」の間から均等に探します。「次」に次回使用予定の値が表示されているので、必要に応じて変更できます。

「フィッティングステータス」には、「緩和関数」で最小二乗フィット結果が表示されます。

「オートフェーズ」をチェックすると、もっとも変化量が大きくなるような位相が選択されます。

「スムージングサンプル数」はフィッティングの前に近い横軸の値をまとめる分割数を指定します。

「オートウィンドウ」をチェックすると、最もS/Nが大きくなるような窓関数を選択しますが、周波数依存性が物理的に重要な場合（超伝導状態等）は、これをOFFにして下さい。

「Fit M(infinity)」をチェックすると、 $t \rightarrow \infty$ の値もフィッティングパラメータにします。通常のT₂測定ではOFFにします。

「周波数」は、FID/エコー信号の解析対象の中心周波数をセットします。この値を変更するとこれまでの測定結果はクリアされます。

「フィッティングリセット」は、フィッティングが上手く行かないときに初期値を乱数で振りフィットをやり直すときに使います。

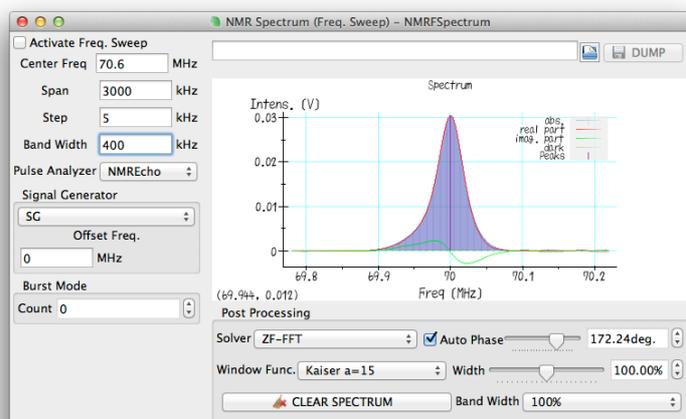
パルサーの設定が「P1 ALT」か「Comb ALT」の時はFID/エコー解析ドライバを2つセットする必要があります。

スカラエントリに値（ $1/T_{1,2, \text{st.e.}}$ と誤差）が送られます。

NMR周波数掃引測定

このドライバは、解析専用で、FID/エコー測定の結果から周波数掃引スペクトラムを測定します。SGの制御も行います。FID/エコーがインクリメンタル平均になっているときは使用できません。窓関数の設定は引き継がず、結果を見ながら合合わせることができます。

ZF-FFTの結果をずらしながら積算する、いわゆるFourier Step Sum(FSS)を行います。ただし、一度タイムドメインに戻すことで、窓関数の選択を後処理中で可能にしています。



「掃引開始」をチェックすると、結果をクリアーして、「中心周波数」 - 「スパン」 / 2から「ステップ」毎にSGの周波数を変更します。

「オートフェーズ」で実部が最大になるようにフェーズを選びますが、通常は周波数掃引では位相は意味がありません。

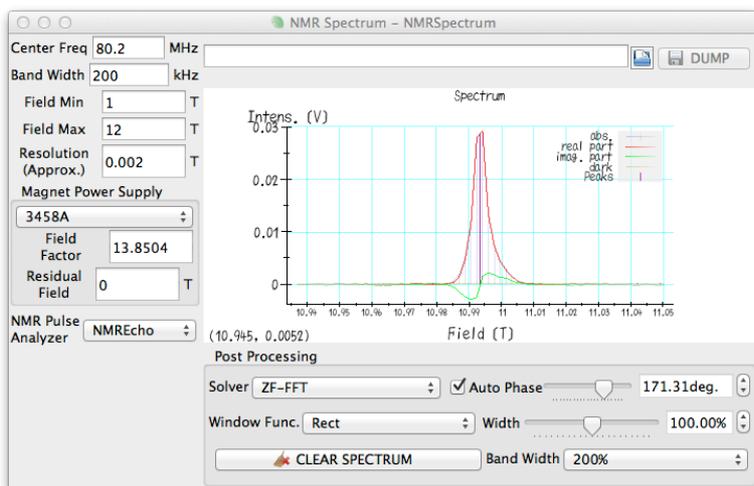
FSSの「バンド幅」は、測定開始前に数値で指定する他、後処理中でも"50%","100%","200%"の3種類から選択することが出来ます。

LC tuningでスイープ中に回路のチューニングをどうするか選択することが出来ます。"As is"ではなにもしません。"Auto Tune"ではオートチューナードライバを使用して"Step"毎ごとにチューニングを行います。"Await"では"Step"毎にパルスをOFFにしてユーザーがパルスを再度ONにするのを待ちます。

NMR磁場掃引測定

このドライバは、解析専用で、FID/エコー測定の結果から磁場掃引スペクトラムを測定します。磁場の値は、マグネット電源ドライバか、DMMの値から読み込みます。FID/エコーがインクリメンタル平均になっているときは使用できません。窓関数の設定は引き継がず、結果を見ながら合わせる事ができます。

ZF-FFTの結果をずらしながら積算する、いわゆるFourier Step Sum(FSS)を行います。ただし、一度タイムドメインに戻すことで、窓関数の選択を後処理中で可能にしています。



設定項目の説明については周波数掃引の項も参照して下さい。

内部動作としては、FSSの際に、周波数の原点を $-\log(H) \times \text{Center Freq}$ としています。これは、純粋なNMRの場合に磁場をずらしても正しく積算される仕掛けです。共鳴周波数が磁場に比例するか、 $\text{Center Freq} \gg 0.001 \times \text{Band Width}$ の場合に有効です。

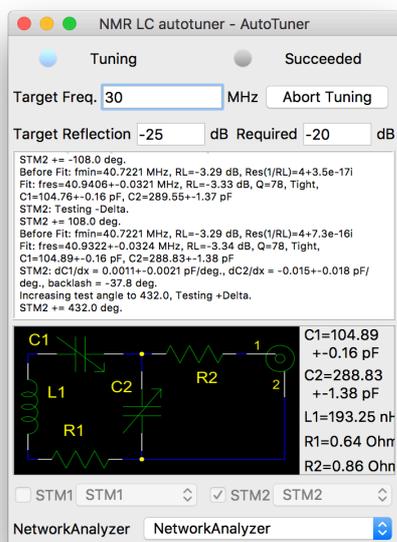
磁場の値は、読み込んだ値に"Field Factor"を掛けて、"Residual Field"の値を足したものを使います。

磁場のコントロールは、別のドライバ、または手動で行なって下さい。

"Resolution (Approx.)"は、横軸の精度を指定しますが、測定中に変更すると結果を一度クリアーする場合があります。

NMR自動LCチューナー

このドライバは、解析専用で、ベクトルネットワークアナライザをモニターさせながら、LC共振回路を調整します。LCR直列共振回路を仮定したフィッティングを行い、キャパシタ変位量とバックラッシュを推定しながらチューニングします。調整の前後に、モータードライバの"AUX"のビットを反転させるので、RFリレーの接続に利用できます。



スクリプトによる制御

Ruby言語による制御が可能です。kame内部では、Ruby専用のスレッドが常時走っており、モニタープログラム(kame/script/rubysupport.rb)がさらにRubyのスレッドを新しく作ってスクリプト(.seq)を読み込み実行します。

標準入力と標準出力はリダイレクトされてkameの中央部に表示されます。文字コードはUTF-8です。

エラーが起これば、バックトレースが赤色で表示されます。

Mathモジュールは標準で読み込まれています。

kameとの情報のやり取り／制御は、XNodeクラスのオブジェクトを介して行います。それらオブジェクトはツリー構造になっており、Measurementオブジェクトがルートです。

マウスポインタを制御対象に置くと“Node Browser”タブにRubyでのオブジェクト名とメソッド一覧及び現在の値が表示されます。

XNodeクラスは、Arrayと似たようなメソッドをサポートしています。

例えば、Measurement["Drivers"]["NMRT1"]["P1Min"]で制御対象のオブジェクト（この場合はXDoubleNodeクラス）にアクセスできます。

XValueNodeクラス及びその子クラスは、value=メソッドで代入、valueメソッドで値の取得が可能です。

Rubyによる制御はわずかなタイムラグがありますので、連続して値を代入する場合は適当なスリープを入れて下さい。

例（各温度でスペクトラムを保存する）

```
def wait_within(node, val, min_wait, timeout, incr = 1)
  tstart = Time.now
  print "Wait for stabilize '#{node.name}' within #{val}"
  sleep min_wait
  while tstart + timeout > Time.now
    if node.get().abs <= val then
      print "OK. #{Time.now - tstart} sec. lost"
      return true
    end
    sleep incr
  end
  print "Time out #{Time.now - tstart} sec."
  false
end
```

```
tempctl = Measurement["Drivers"]["TempControl2"]
pulser = Measurement["Drivers"]["Pulser"]
pulse = Measurement["Drivers"]["NMREcho"]
```

```

for temp,rept in [[10,100],[20,50],[40,25]]
  print "T=#{temp}K\n"
  pulser["RT"].value= rept
  pulser["Output"].value= true
  tempctl["TargetTemp"].value=temp

  pulse["ExAvgIncr"].value= false
  sleep(3)
  pulse["ExAvgIncr"].value= true
  pulse["Spectrum"]["FileName"].set("hogehege-#{temp}K-ft.dat")
  slp = [300.0, rept*100/1000.0].max()
  sleep(slp)
  pulse["Spectrum"]["Dump"].touch()

  wait_within(tempctl["Stabilized"], temp/100.0,30,36000)
end

```

FAQ

LinuxでDAQmxを使っていて、頻繁にフリーズするとき は？

現在のNational Instruments社DAQmx8.0.2ドライバでは、以下の制限事項というかバグがあります。
 32bit x86のみかつ、4GB以下のメモリ
 カーネルスタックサイズは8kB必要
 IOMMU (IO仮想化, VT-d等) は使えない
 SE Linuxの設定を行なってくれない。

これらを守らないと、突然フリーズします。他にも、マルチスレッド対応が完全でないようなので、問題があれば、CPUコア数を1個にしてみてください。

GPIBで通信エラーが起こるときは？

よくあるのは、アドレスが間違っているとか、ケーブルが長すぎるとかですが、まれに機器間の相性問題があります。Oxford社全般とYOKOGAWAの7651、cryoconは、IEEE488規格に準拠していないので、非常に沢山の機器をつないだ場合に問題が起こることがあります。装置を減らすか、一部をシリアル通信に切り替える等しましょう。

シリアルポートの設定は？

上記に特に指定がない限り、9600bpsノンパリティです。

シリアルポートが開けない時は？

ポート名は、WindowsならばCOM1などですが、Mac/Linuxでは/dev/tty.usbserial-***などです。

Linuxではパーミッションも確認して下さい。“Line in use”のエラーが出てmodem-managerが邪魔している場合は、NetworkManagerを停止・OFFにして、su -c 'killall modem-manager'としてください。

kameで対応していない機器を使いたい時は？

ドライバを作成する必要があります。ソースコード中のmodules以下のフォルダを参考にして作ることができますが、難しければ北川にコンタクトしてみてください。デジボルの機種を追加する程度であれば僅かな作業で出来ます。

付録

NI製DAQmxデバイスを用いたNMRシステム(パルサー及びオシロスコープ/アベレージャ)

私は、SシリーズのPCI-6111 (2ch12bit5MSps) をA-D変換器として利用しています。変調にQAMを使用する場合は、D-A変換部分も使います。上位機種のPCI-6115も利用できます。

内部動作としては、リアルタイムに全てリングバッファに取り込んで、ソフトウェアで積算しています。従って、いかなる繰り返し時間でも、取り込み範囲が重ならない限りは取りこぼしは有りません。また、積算回数に制限はありませんが、DSOドライバでは32bitで積算しているので65536を超える積算は、NMRエコードライバとの組合せで行なって下さい。

パルサーとしては、MシリーズのPCIe-6251(DIO 8ch10MHz)を使います。PCI-6220も遅いですが、使えます。PCIe-6251はディスコンになりますが、後継機種PCIe-6351についてはまだ動作確認していません。

その他必要な外部端子等としては、

PCI-6111用に
SH68-68-EP 1個
BNC-2110 1個

PCIe-6251用に
CB-68LPR 1個
SHC68-68EPM 1個
CA-1000 Box 1個
CA-1000 ラックマウント 1セット
CA-1000 BNCパネル 4個
CA-1000 カバー 5個

を使っています。

また、2つのボード間のクロック・トリガー同期用にRTSIケーブルの接続が必要です。

CA-1000 BNCパネルは絶縁BNCを使っていますが、これを非絶縁BNCに交換しないと、グリッチがNMRエコーに混じって観測されてしまいます。

Sシリーズデバイス

AI0,1(2ch)またはAI0-3(4ch)がA-D変換用の入力端子です。NMRではレシーバーのPSD出力に繋がります。0度と90度のポートが逆になってレシーバーもあるので注意。

ドライバで”National Instruments DAQ as DSO”を選択してください。

モジュレーターにQAMを使用する場合、パルサードライバとして、”NMR pulser NI-DAQ M Series with S Series”を選択して、2つ目のインターフェースにSシリーズを選んで下さい。AO0,1をQAMに繋がります。

Mシリーズデバイスのパルサー用接続

QAMを使用しない場合、パルサードライバとして、”NMR pulser NI-DAQ digital output only”を選んで下さい。

SシリーズをDSOとして使用する場合、DSOの「トリガー設定」で「ソース」にパルサーの適切なラインを選んで下さい。これはソフトウェアでトリガーが掛かるので、物理的に存在しないポートでも構いません。お勧めは、ポート8を”Trig1”に設定して、「ソース」を”Pulser名/line8”にすれば、 2τ の位置にトリガーが掛かります。ソフトウェアでトリガーをかけている都合上、DSOの縦横軸の設定を変更した後は、パルサーをOFF/ONしなさいないとトリガーがかかりません。

どんなパルサーにも言えますが、出力中に電源OFFした機器に繋いだり50 Ω 終端するのは止めましょう。ショートしているのとほぼ同じ事になります。

OFF時にはオープンステートにしますので、GATEとPRE-GATEのポートは数k Ω 程度でGNDに落として下さい。

以下の写真を参考に接続して下さい。

ポート7は、内部動作に予約しています（設定で変更できますが）。パルサーに信号変化がないときにカウントを停止させてボードを休ませる用途に使っています。その際、P0.7(J48)とPFI4(J41)を接続して下さい（写真中青い線）。また、NMRパルサーの設定画面の「もっと設定」内でポート7を”Pausing(PFI4)”に設定して下さい。

接続表は以下の通りです。

（ポート）	（DAQポート名）	（端子番号）
ポート0	P0.0	J52
ポート1	P0.1	J17
ポート2	P0.2	J49
ポート3	P0.3	J47
ポート4	P0.4	J19
ポート5	P0.5	J51
ポート6	P0.6	J16
ポート7	P0.7	J48 / PFI4 J41
GND	D-GND	4,7,9,12,13,15,18,35,36,44,50,53のいずれか

