Digital systemの現状

2010/3/5(金) 重力波研究交流会

宮川　治（東大宇宙線研）
＋
このシステムの開発に協力してくれたたくさんの人々

目次

- デジタルの基礎
- 一般ユーザー向けの説明
  - CLIOでやられていること
  - デジタル制御のためのツールの紹介
- 開発に興味がある人のための説明
  - 必要なハードウェア
  - リアルタイムコード
  - アナログ回路など
  - 値段

CLIOデジタルシステムの稼働状況

CLIOでデジタルシステムが稼働し始めた

- MCのアラインメント信号の取得
- ピコモータードライバ
- Per Armのロックアクイジション（透過光による線形化、閾値によるトリガー）
- Local SPIの制御

を2週間の間で実施した。明らかに作業効率が上がっている。

斎藤君談: 非常にやりやすいと感じました。 ～中略～ 全てデジタルで行うことができ、さらにモニター、データ測定もできるので、Center で作業をすることができます。100m 往復する回数が減り、体力的にも余裕ができると思います。

Digital system
基礎

デジタル固有のノイズ

- 問題点:ADC (analog to digital converter), DAC (digital to analog converter)でのノイズが大きい
  - Analog ~1nV/rHz
  - ADC/DACでのNoise ~数μV/rHz
- Whitening filter, Dewhitening filter
  - 通常信号はDCで大きく、高周波で小さくなりノイズに埋もれる。単純にゲインを上げるとDCでサチるので、DCでのゲインを変えず、高周波でゲインを上げ、信号を平坦化(Whitening)して、ADCに入れることにより、実施的にADCのノイズを回避する。逆に、DACから出た平坦な信号をもとのスペクトルに戻す(Dewhitening)ことにより、DACのノイズを回避する。
- Anti Aliasing(AA) filter, Anti Imaging(AI) filter
  - アナログの信号をあるサンプリング周波数でデジタル化するときに、その周波数以上にあるノイズがダウンコンバートされてノイズとなって現れる(Alias)
    - 例:100Hzのサンプリング周波数で、101Hzの大きな信号を取ろうとすると、当然101Hzは何も見えないのだが、そのビートの周波数の1Hzに大きな信号があるとデジタル上で勘違いしてしまう。
  - Aliasを防ぐために、サンプリング周波数より低いところ(通常はサンプリング周波数の半分より少し下)に急峻なローパスフィルターをいれる
  - Anti Imageはその逆で、DACから出てくる信号の高周波に低周波からの信号が乗っかってしまうので、それをアナログ側に出さないように、AAと同じような形のフィルターを通す。

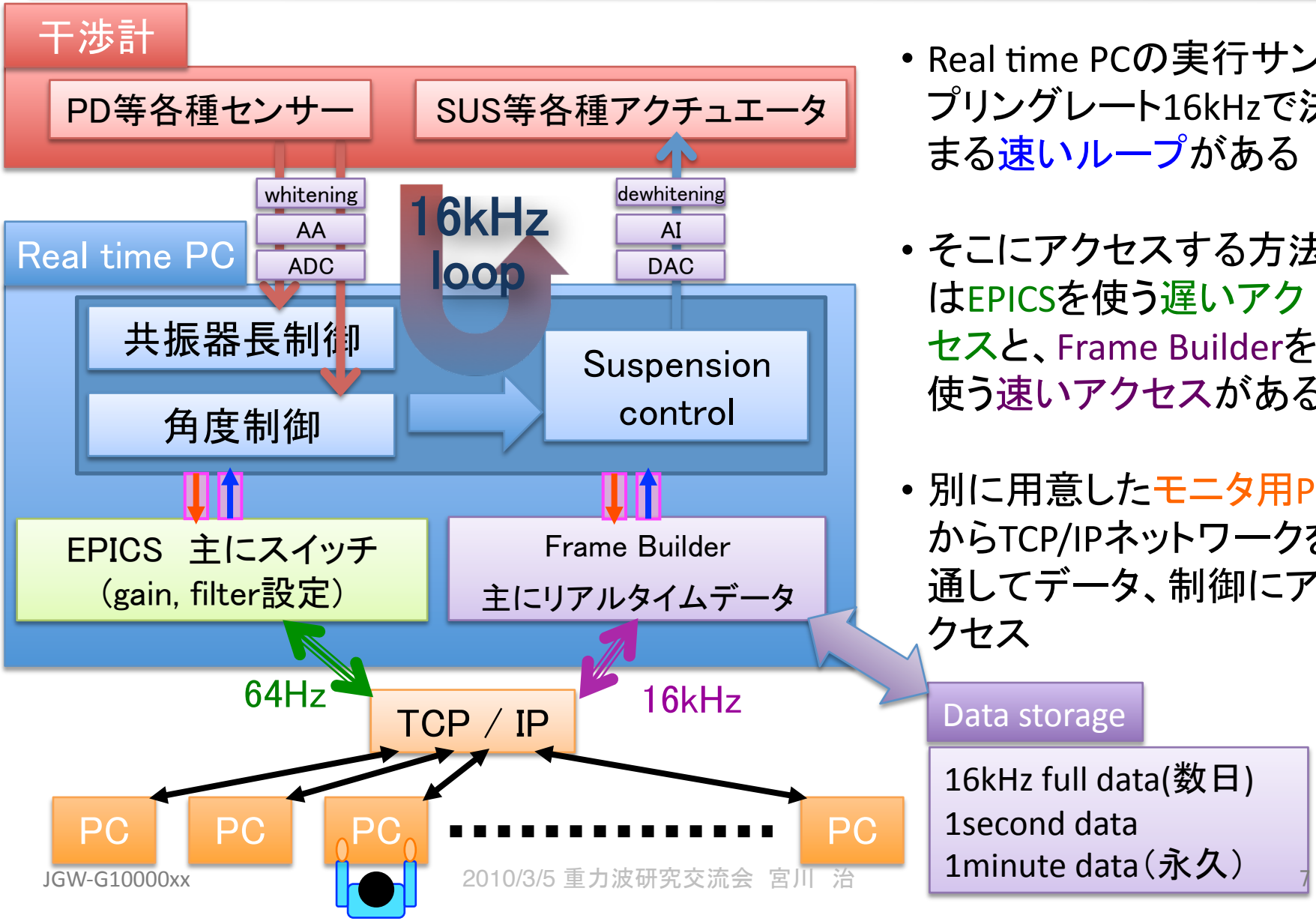
一度デジタルに入ってしまえば、あとはノイズは混入しないので、
デジタルの入り口と出口のノイズのみをきちんと考えてやればよい

デジタルシステムの利点と欠点

- ひと言で言うと、「干渉計へのアクセスのしやすさを提供する」
  - フィルターの設計制作、改変調整、切り替えが簡単
  - ゲイン、信号経路等の切り替えのスクリプト化による自動運転
  - 全干渉計パラメータの時系列での記憶
  - 開発された技術を簡単にコピーできる
- 干渉計全体を把握出来る人を増やす
  - 全体の見通しがよく、初心者でもほんの少しの訓練で干渉計が触れる
  - 誰がやっても最適化のレベルが毎回同じで、その状態が長く続く
  - 複雑なシステムになる程、効率的に信号を扱うことができる
- 感度向上のための時間の短縮につながる

- システムが巨大なので、構築するまでが大変
  - 構築したあとの作業効率は劇的に良くなる
- 遅い(16kHzサンプリングでUGF=300~400Hz)
  - LCGT、CLIOなら静かな環境なので高いUGFは必要なく、問題ない
- ADC、DACのノイズが大きい、alias, imageがある
  - whitening, dewhitening, anti alias, anti imaging filterで解決可能

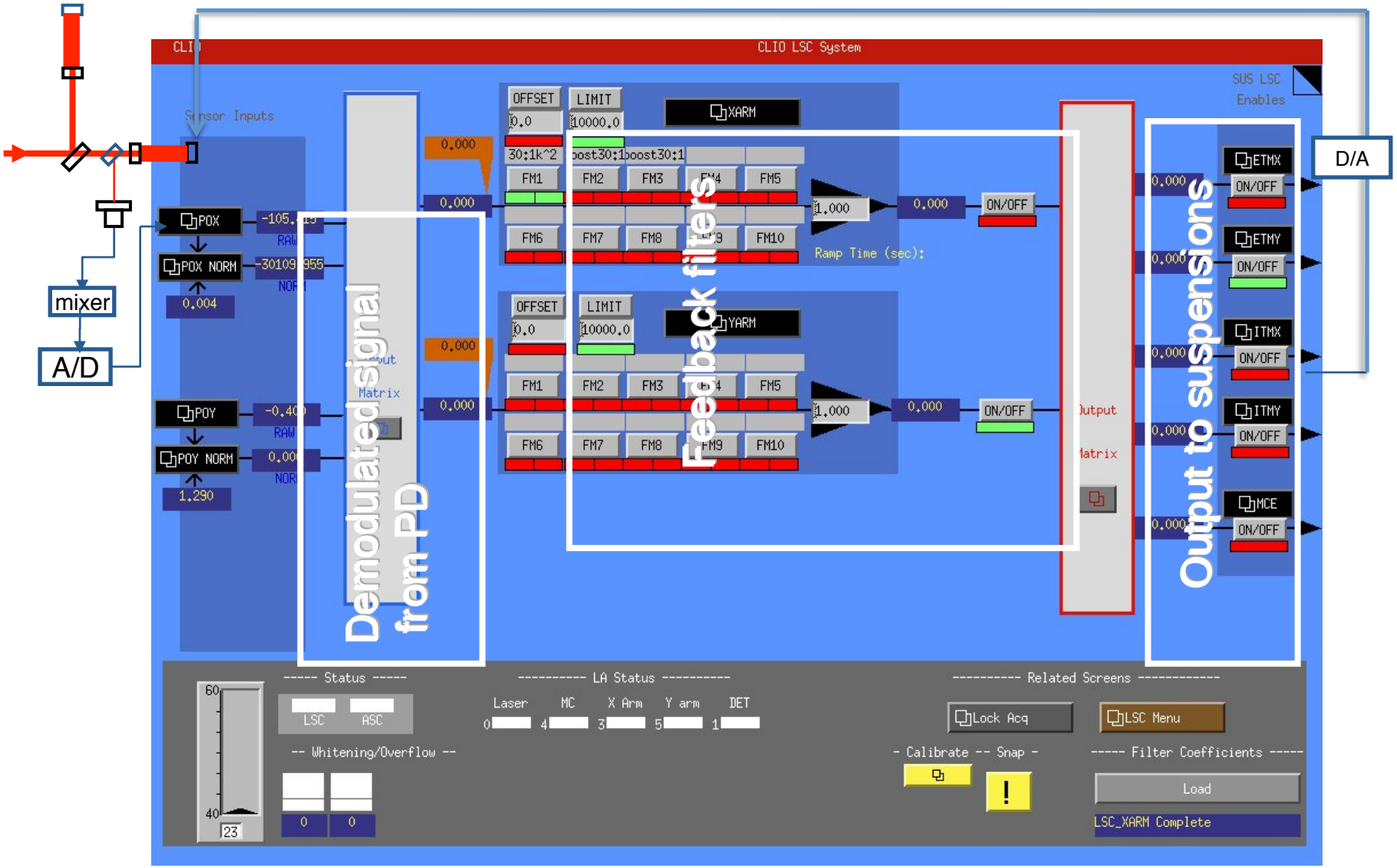
CLIOデジタルシステムの概念図



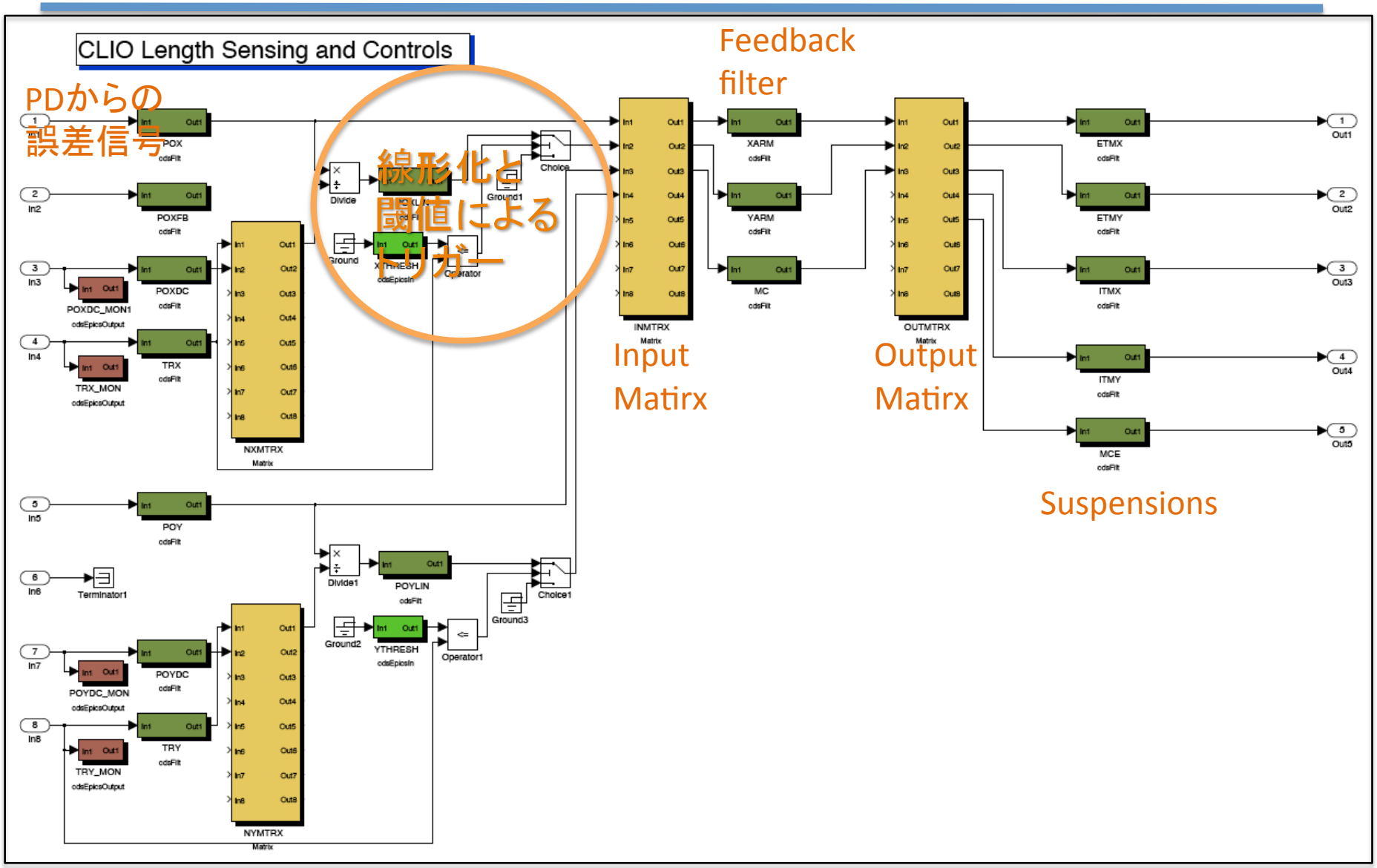
- Real time PCの実行サンプリングレート16kHzで決まる速いループがある
- そこにアクセスする方法はEPICSを使う遅いアクセスと、Frame Builderを使う速いアクセスがある
- 別に用意したモニタ用PCからTCP/IPネットワークを通してデータ、制御にアクセス

Digital system
一般ユーザーのための説明

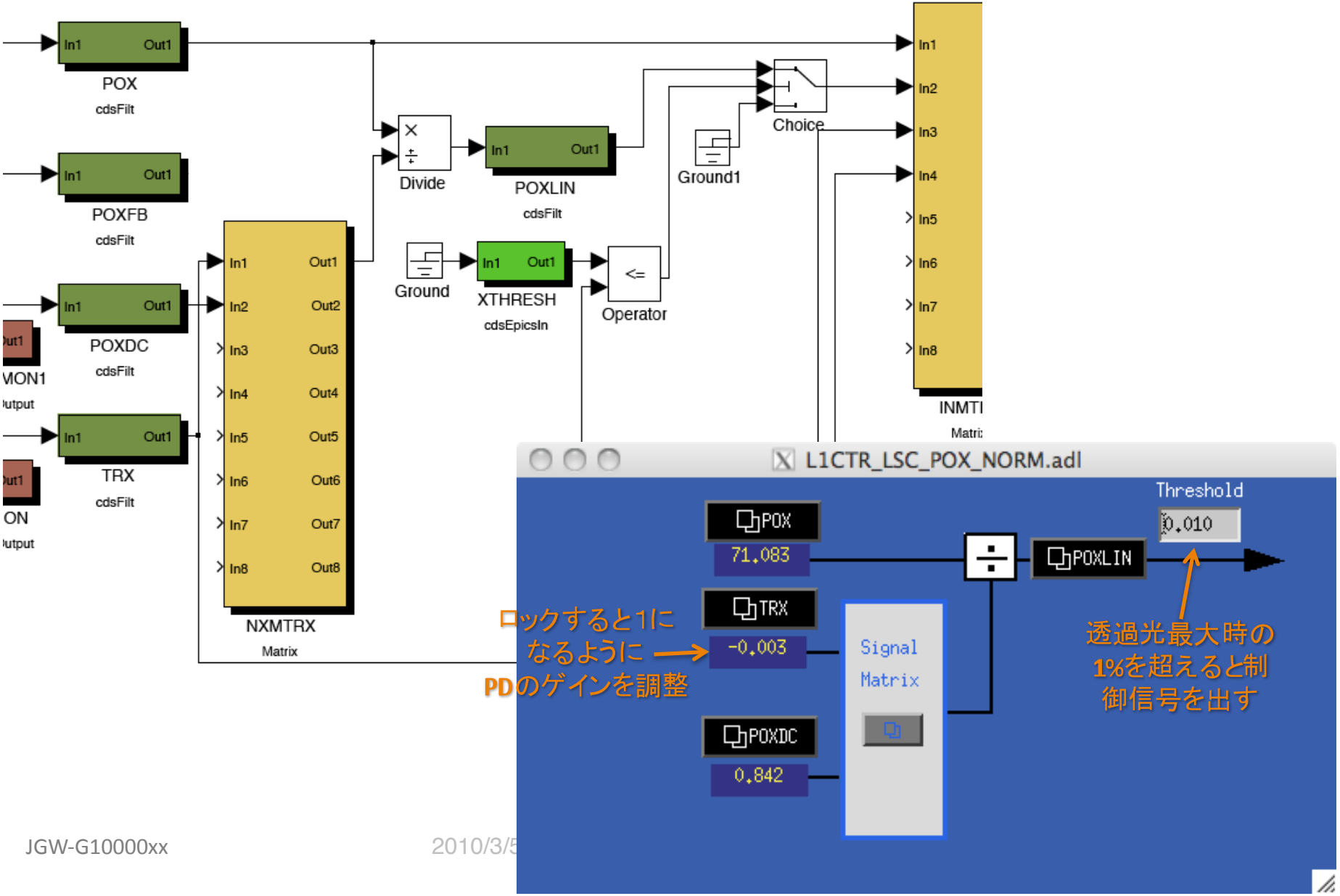
MEDM --Length sensing and control system



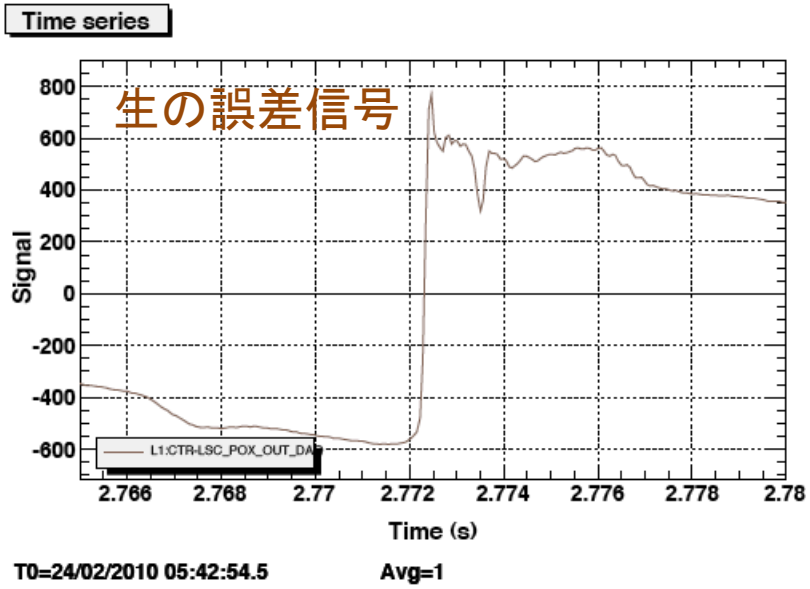
Real time code on Matlab, Simulink

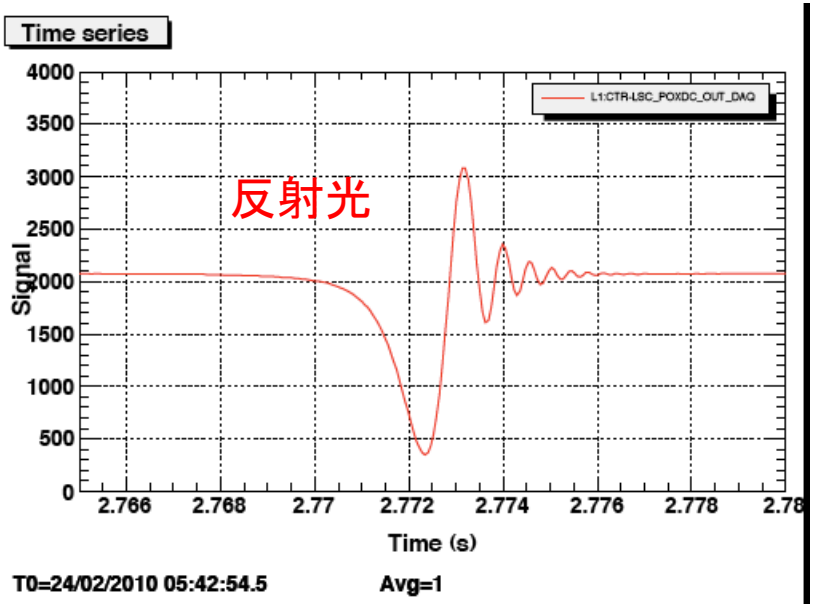


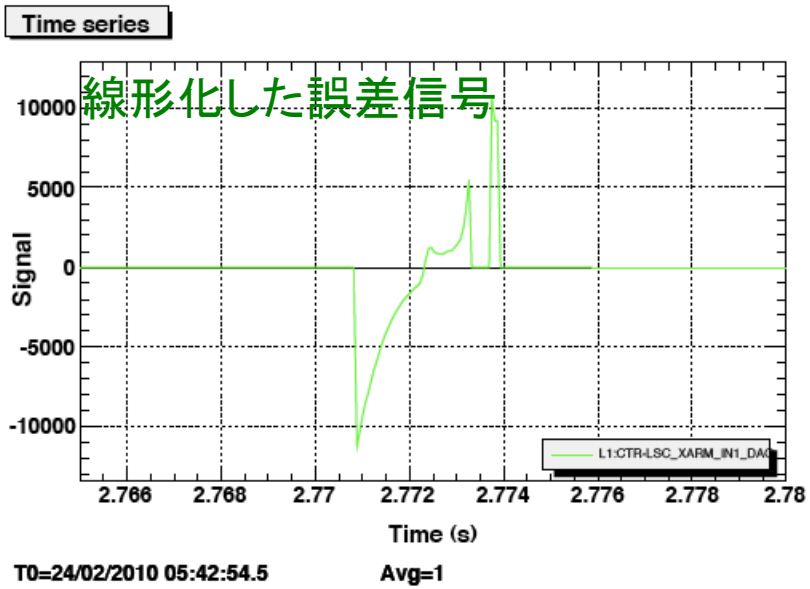
線形化と、閾値によるトリガー

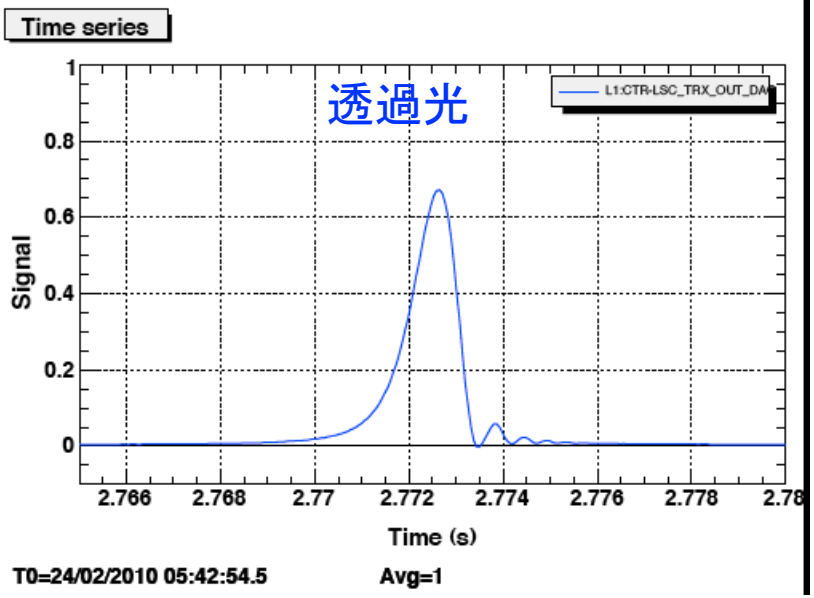


デジタル上の信号


Time series
生の誤差信号
Signal
Time (s)
L1:CTR-LSC_POX_OUT_DAQ
T0=24/02/2010 05:42:54.5
Avg=1


Time series
反射光
Signal
Time (s)
L1:CTR-LSC_POXDC_OUT_DAQ
T0=24/02/2010 05:42:54.5
Avg=1


Time series
線形化した誤差信号
Signal
Time (s)
L1:CTR-LSC_XARM_IN1_DAQ
T0=24/02/2010 05:42:54.5
Avg=1


Time series
透過光
Signal
Time (s)
L1:CTR-LSC_TRX_OUT_DAQ
T0=24/02/2010 05:42:54.5
Avg=1

Digital filter

Foton：デジタルフィルタ作成ツール

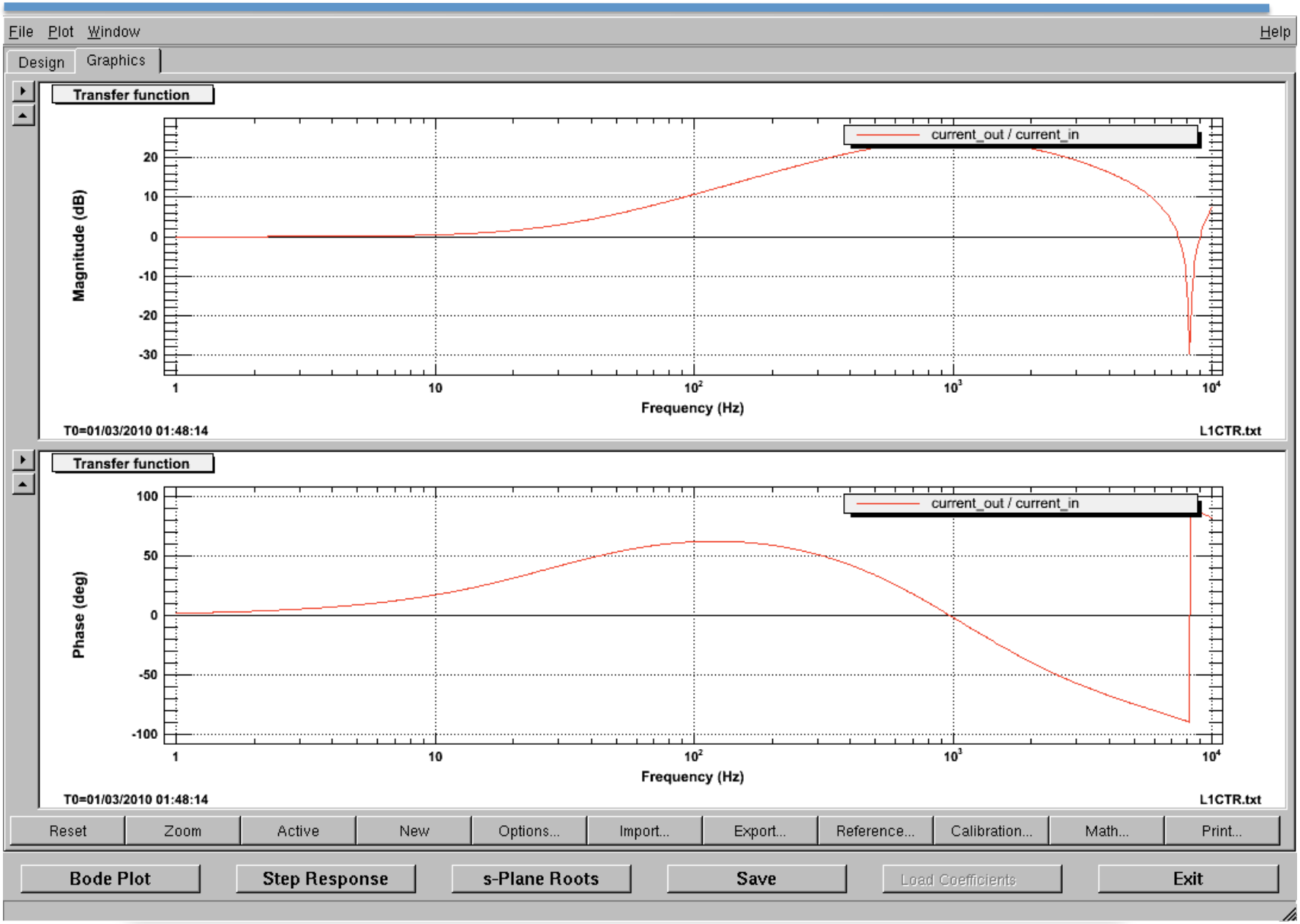
1. フィルタの形選択（ZPK、Notch、Resgain, BW、Elip、Chev、Comb…）
2. 変数を入力（周波数とQ値：ゼロポール、複素数）
3. その場でプロット（確認）

干渉計制御で必要となるフィルタ形状はほぼ全て網羅

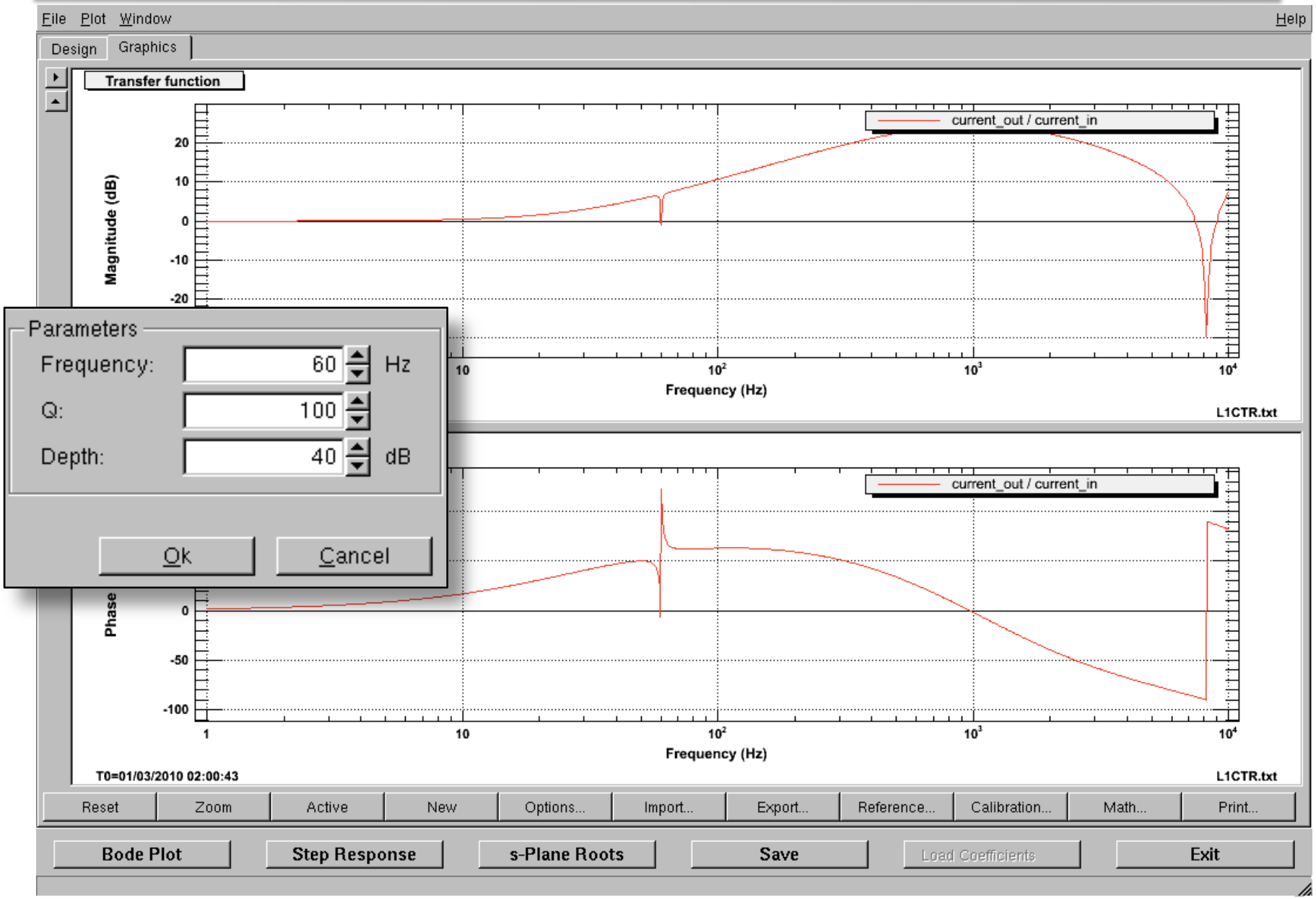
- アナログフィルタの難しい理論を知らなくてもすぐフィルタが出来る
- フィルタが決まればロックを落とすことなくループに挿入できる
- 欲しいフィルタを欲しい時に（設計から完成、ループへ挿入まで1～2分で出来る）

フィルタを微調整するには非常に重要

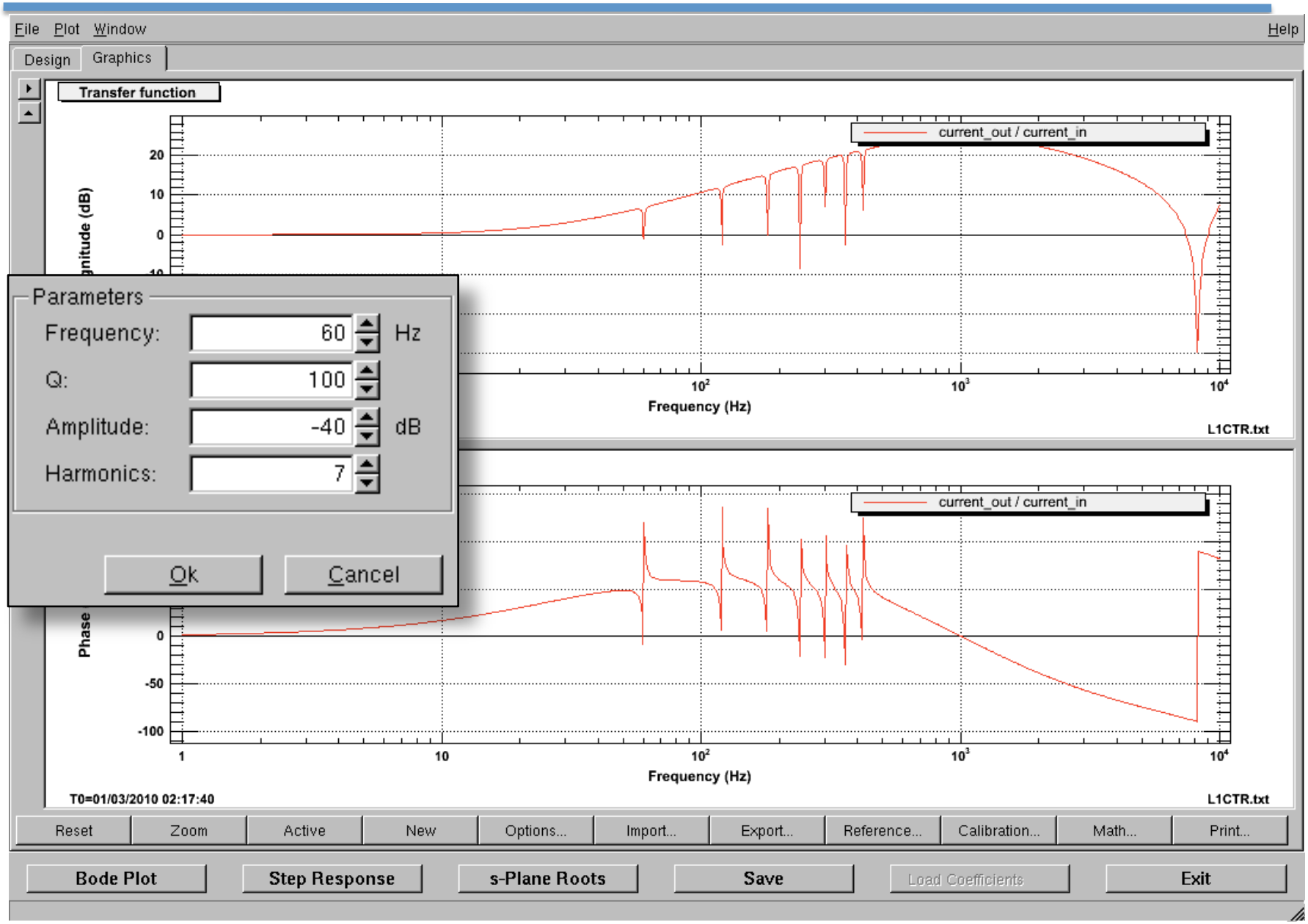
Fotonの例1:腕ロック用filter



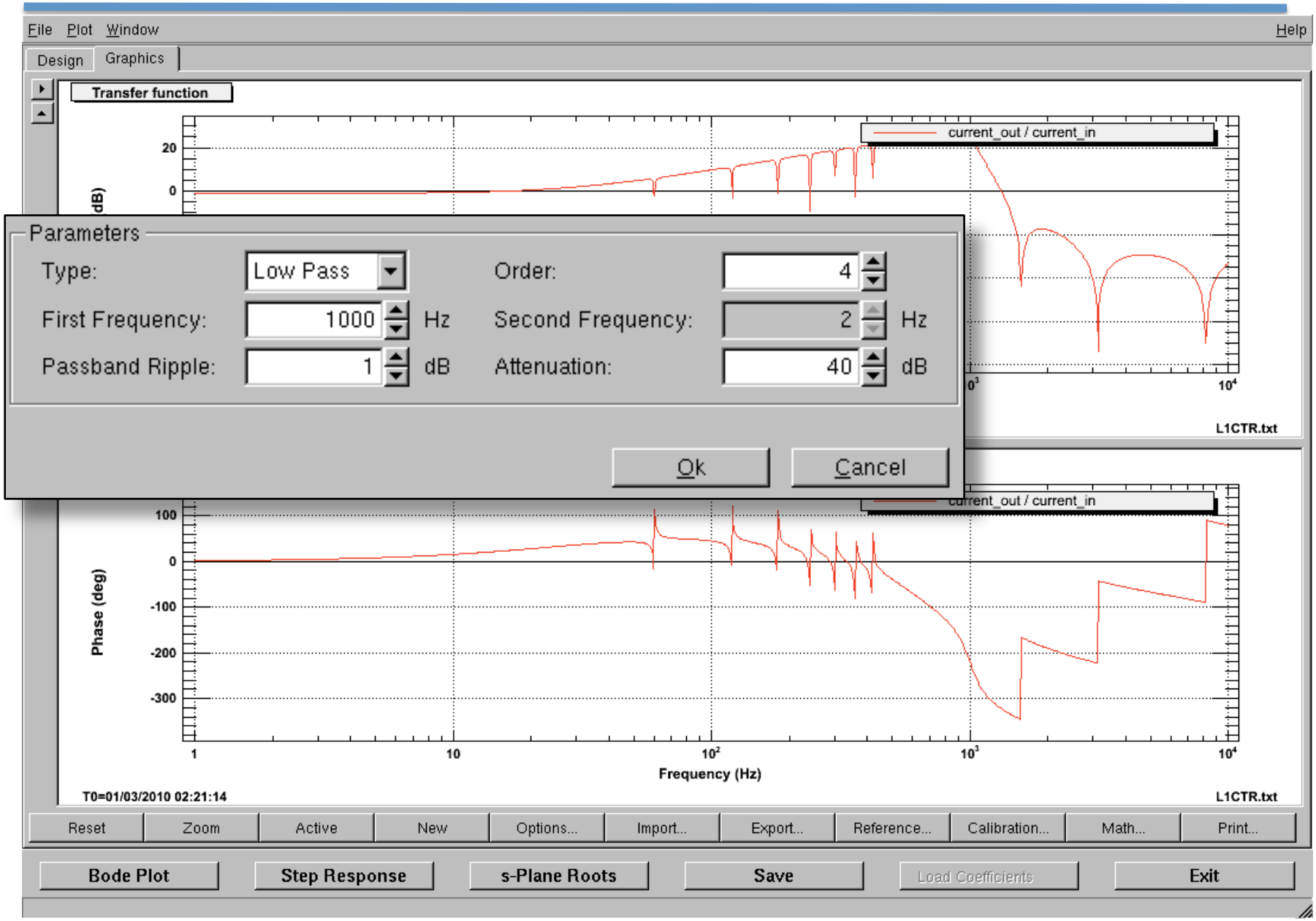
Fotonの例2: Notch filter



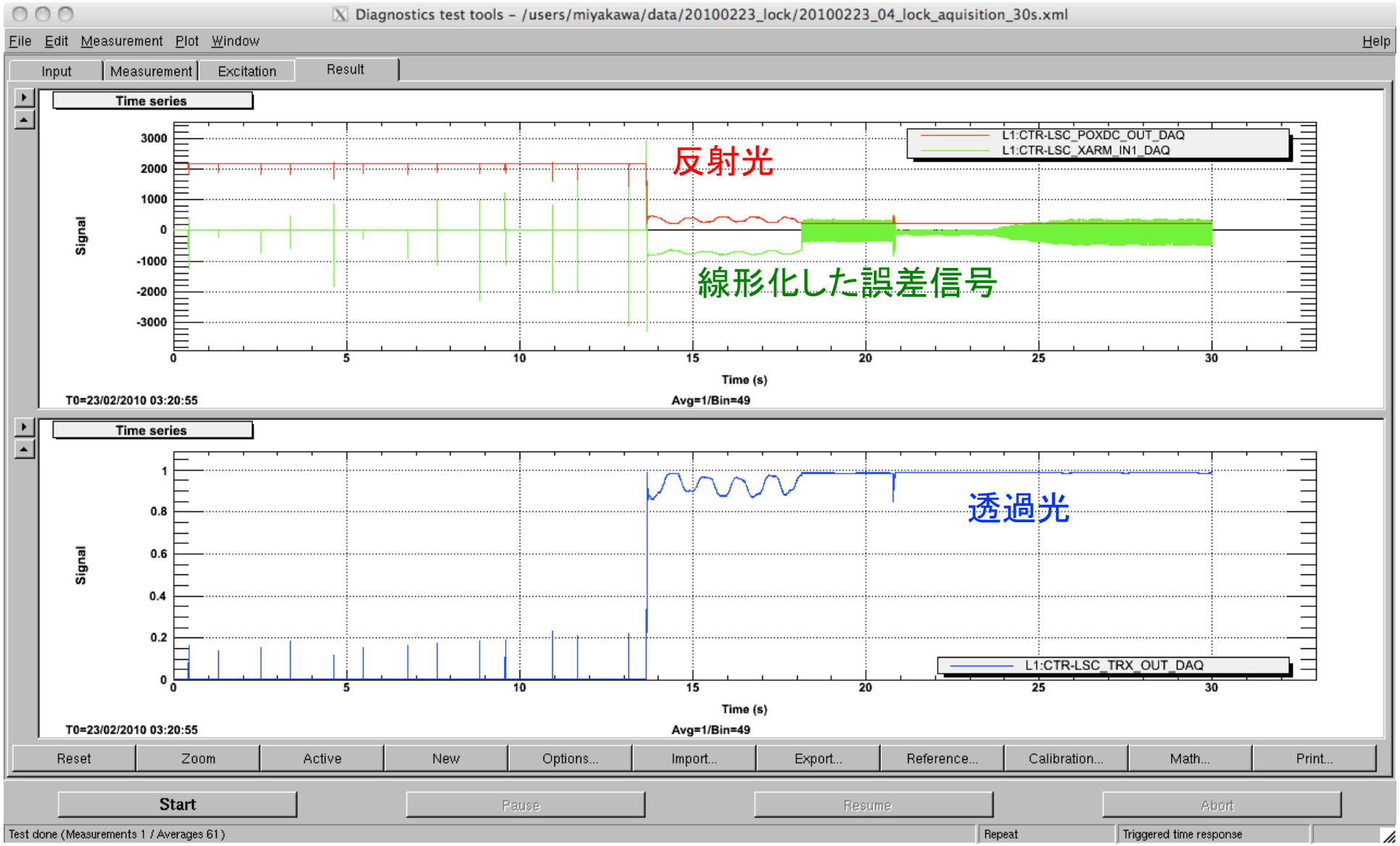
Fotonの例3: comb filter



Fotonの例4: elliptic filter



Per arm lock


Diagnostics test tools - /users/miyakawa/data/20100223_lock/20100223_04_lock_aquisition_30s.xml
File Edit Measurement Plot Window
Help
Input
Measurement
Excitation
Result
Time series
L1:CTR-LSC_POXDC_OUT_DAQ
L1:CTR-LSC_XARM_IN1_DAQ
反射光
線形化した誤差信号
Signal
Time (s)
T0=23/02/2010 03:20:55
Avg=1/Bin=49
Time series
透過光
L1:CTR-LSC_TRX_OUT_DAQ
Signal
Time (s)
T0=23/02/2010 03:20:55
Avg=1/Bin=49
Reset
Zoom
Active
New
Options...
Import...
Export...
Reference...
Calibration...
Math...
Print...
Start
Pause
Resume
Abort
Test done (Measurements 1 / Averages 61)
Repeat
Triggered time response

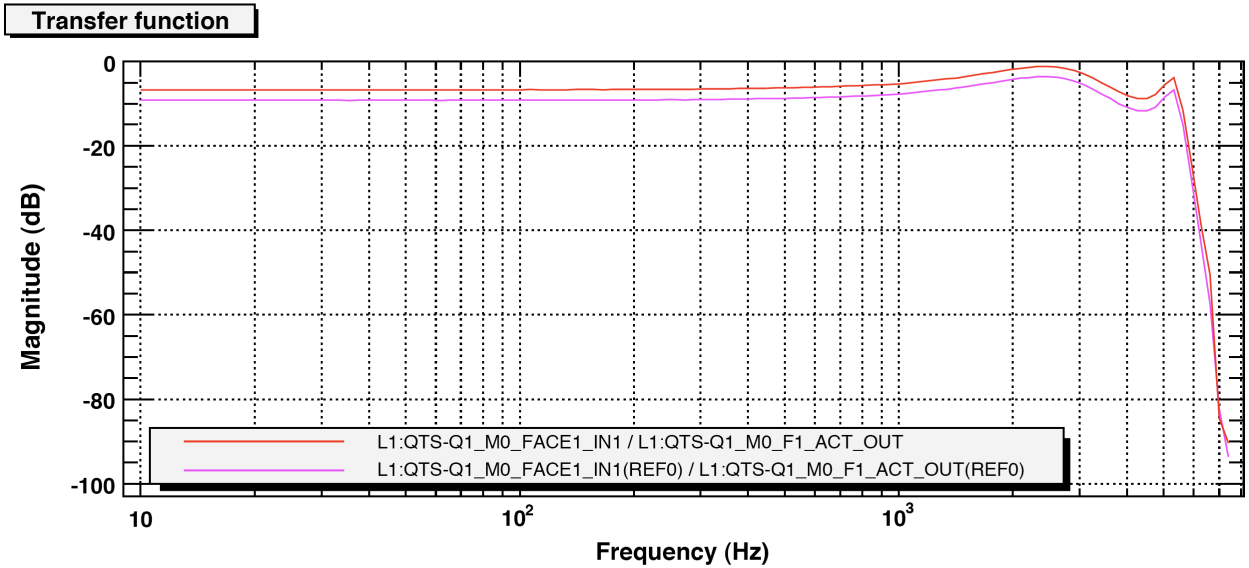
diaggui --FFT アナライザツール

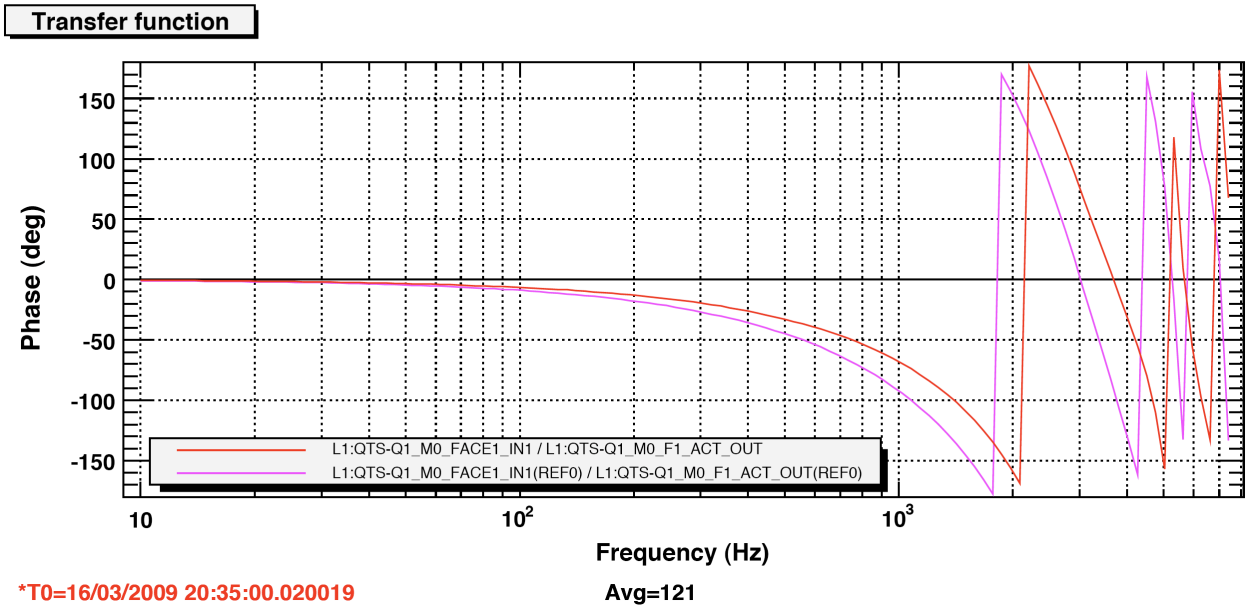
- デジタルFFTアナライザ
  - 入出力以外に各フィルタの前後等の演算過程の値を取り出せるように設定されている
  - コンピューター上でFFTアナライザを再現
  - 任意のポイントの測定が可能
    - ノイズレベル測定（FFT）
    - 任意の範囲でOpen loop、Close loop伝達関数測定（swept sine）

⇩

  - 無限個のFFTアナライザを手に入れたことになる
    - FFTアナライザのコストダウン
  - 重いFFTを持ち歩かなくて良い
    - ぶつけて壊すこともない
  - Endの測定もセンターから可能
    - 思い立ってから3km（LCGT）も移動するのは大変です

全ループの遅延特性





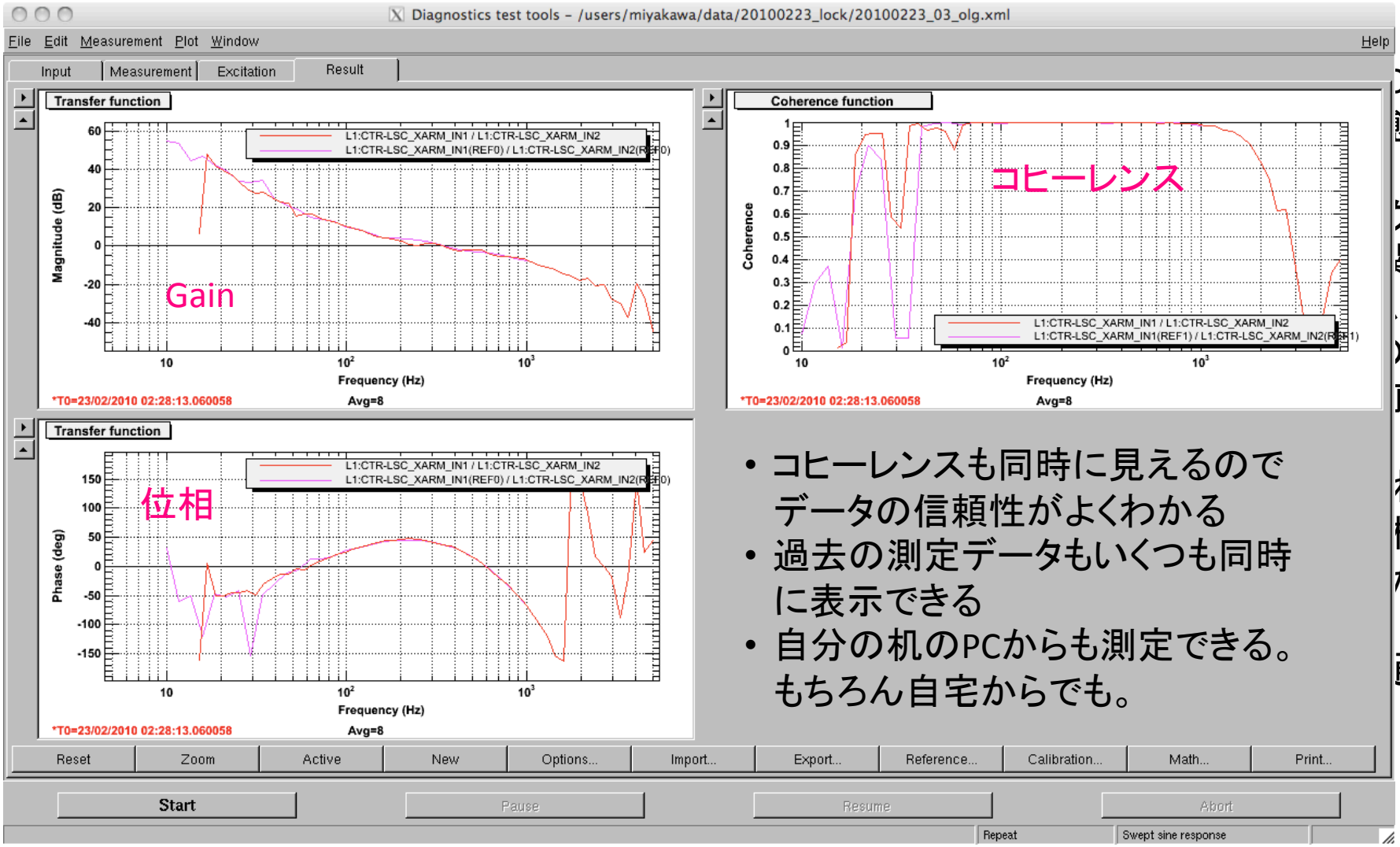
- デジタルの出力と入力をケーブルでつなぎ、デジタルの出口から、デジタルの入り口までの伝達関数を計算機内で測定

- ピンクが10kHzのアナログAA/AI有り
- 赤がアナログAA/AI無し

- 5kHzのローパスはソフトウェアAA/AI

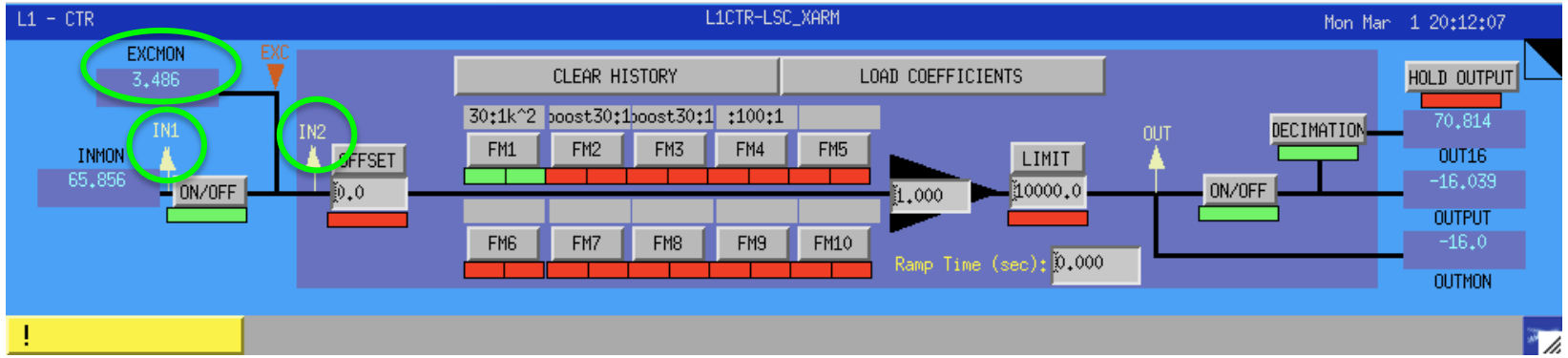
- AA/AI込みで、100Hzで10度程度、300Hzで30度程度の位相遅れ
- UGFは予想通り数100Hzまで

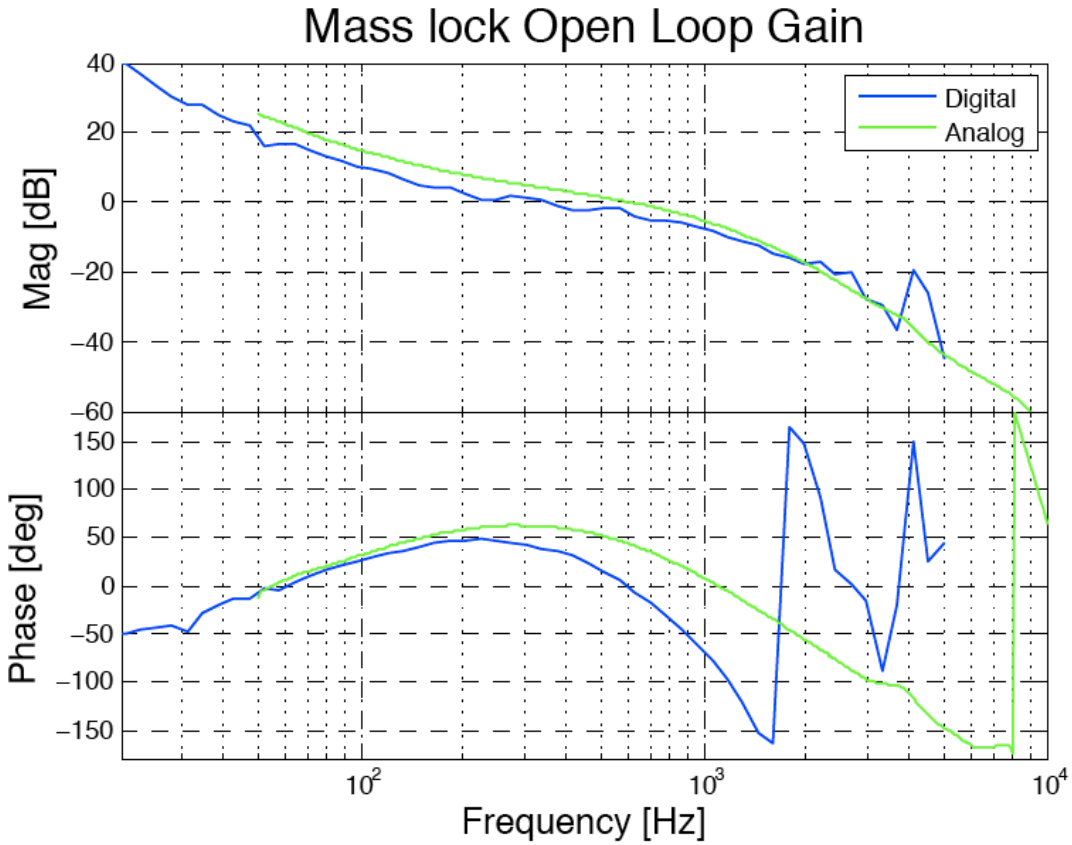
Open loop TF on diaggui



- コヒーレンスも同時に見えるのでデータの信頼性がよくわかる
- 過去の測定データもいくつも同時に表示できる
- 自分の机のPCからも測定できる。もちろん自宅からでも。

Open loop TF for Digital vs. Analog

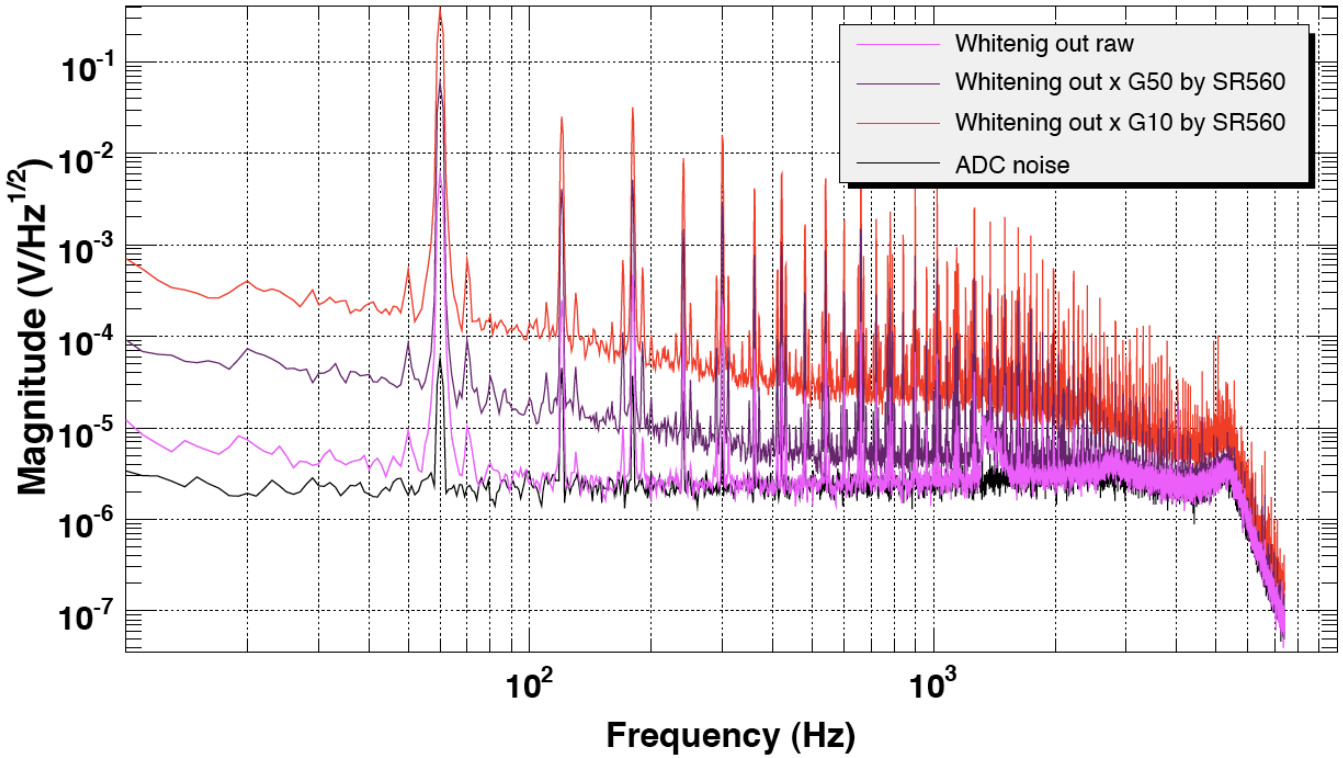




- Analog(緑)は普段CLIOの感度を測定している状態でとったもの
- Digital(青)とはアナログで制御した中に、伝達関数1のデジタルシステムを挿入して、デジタル上のExcitationの加算器の前後で測定したもの
- 前ページの位相の遅れに相当した分だけの位相遅れが加わっているのがわかる
- 実用的なUGFは予想通り400Hz程度が限界

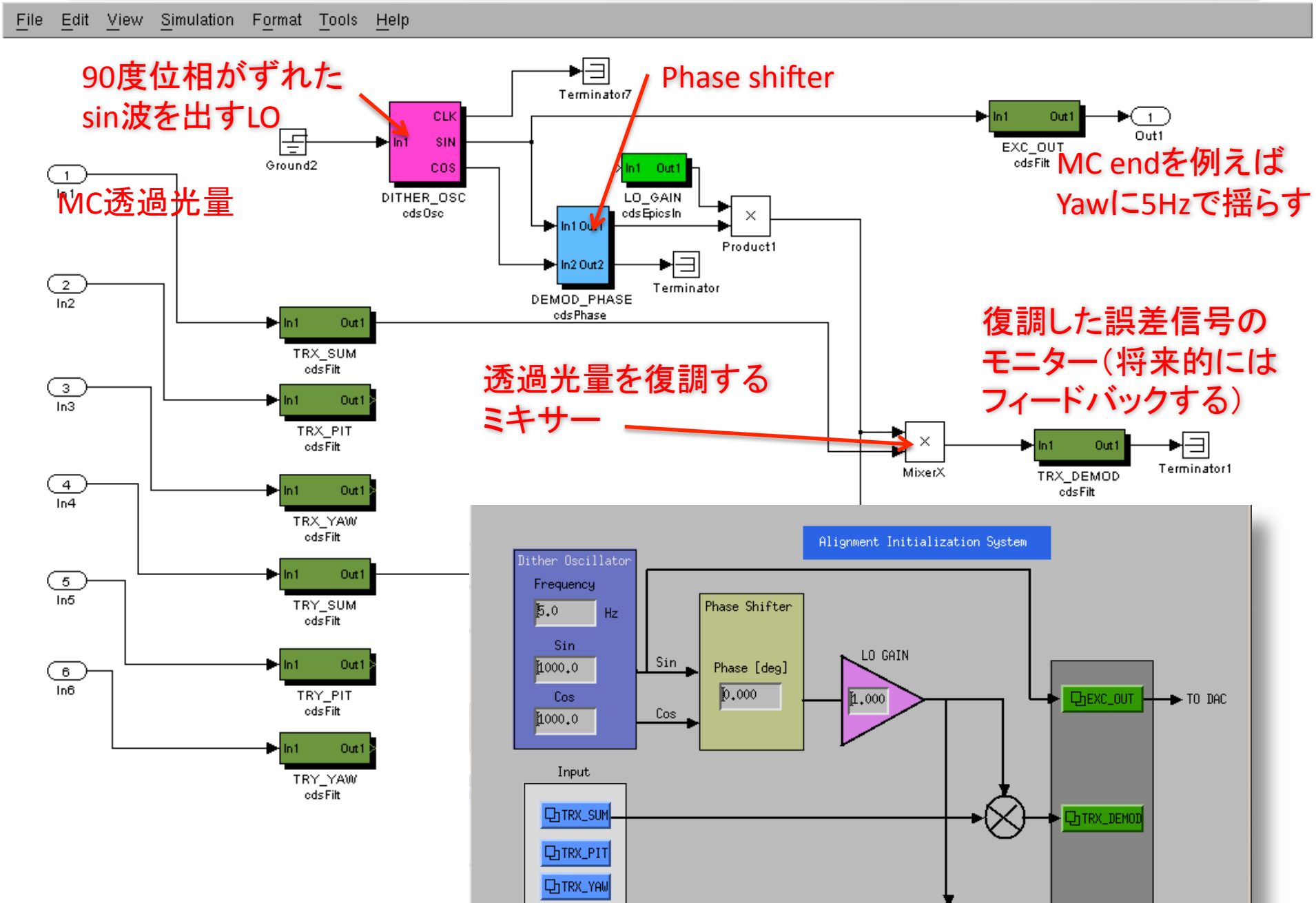
CLIOのノイズに対して

Feedback signal with Whitenig filte

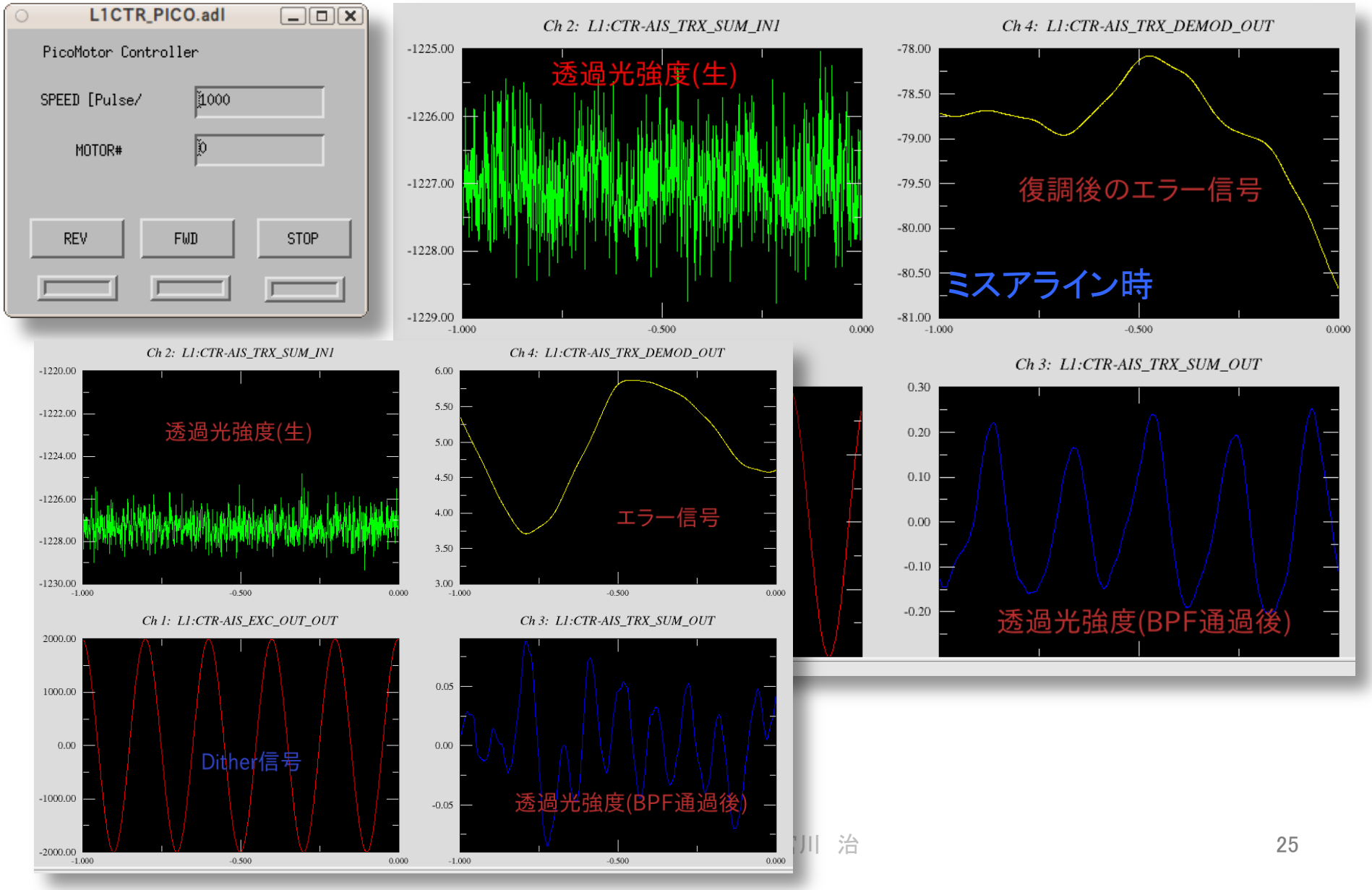


- アナログで制御して、そこから出てくるfeedback信号に普段感度をとるために使っているwhiteningをかけてADCに入れてみた。SR560で50倍してやればで60Hzでサチることなく、全帯域で意味のある信号が取れている。
- うまくwhitening してやればアナログの感度を汚すことなくデジタルで置き換えれる見通しが付いたことを意味する。

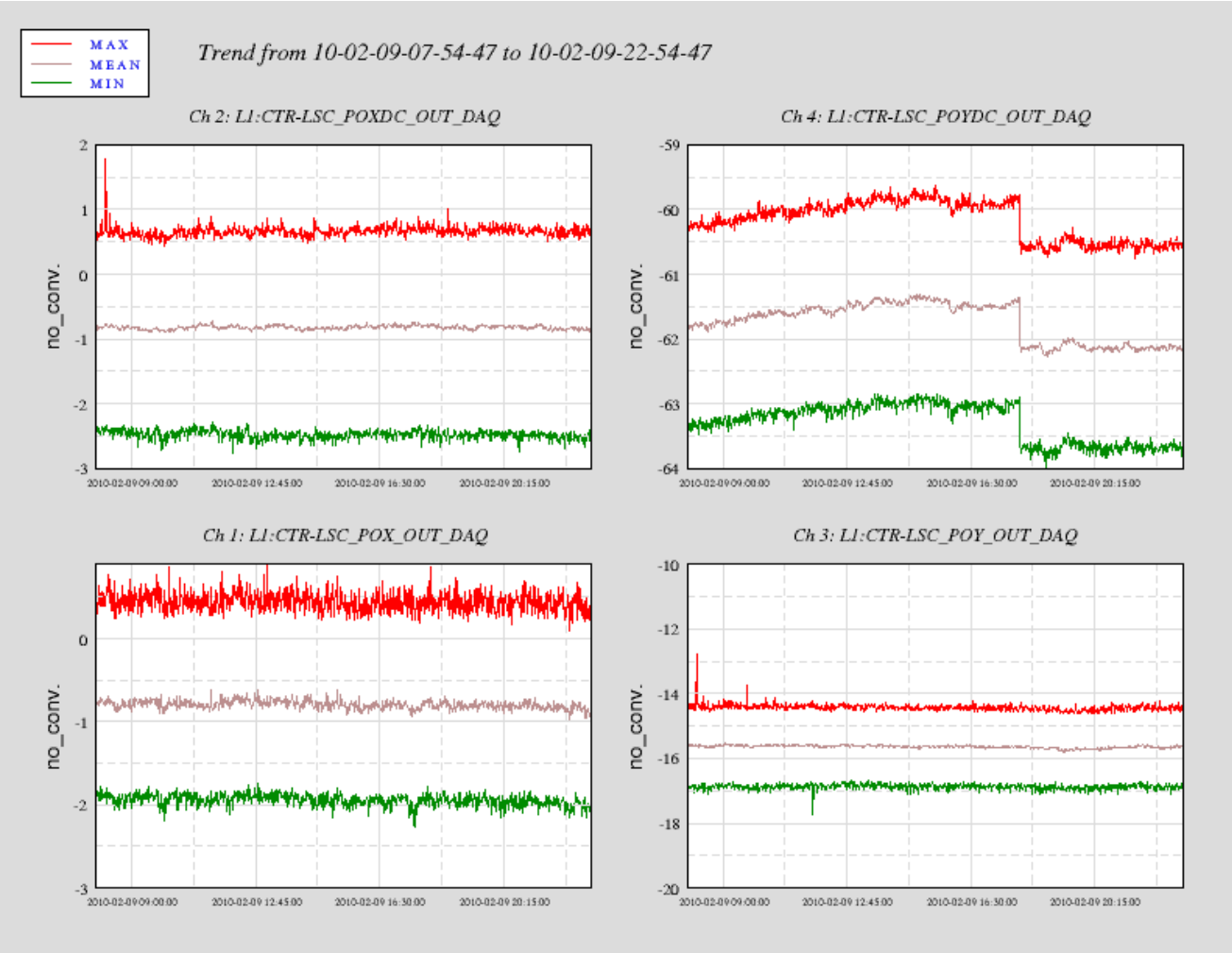
MCのミスアラインメント検出



ミスアラインメントに応じた誤差信号(含方向)

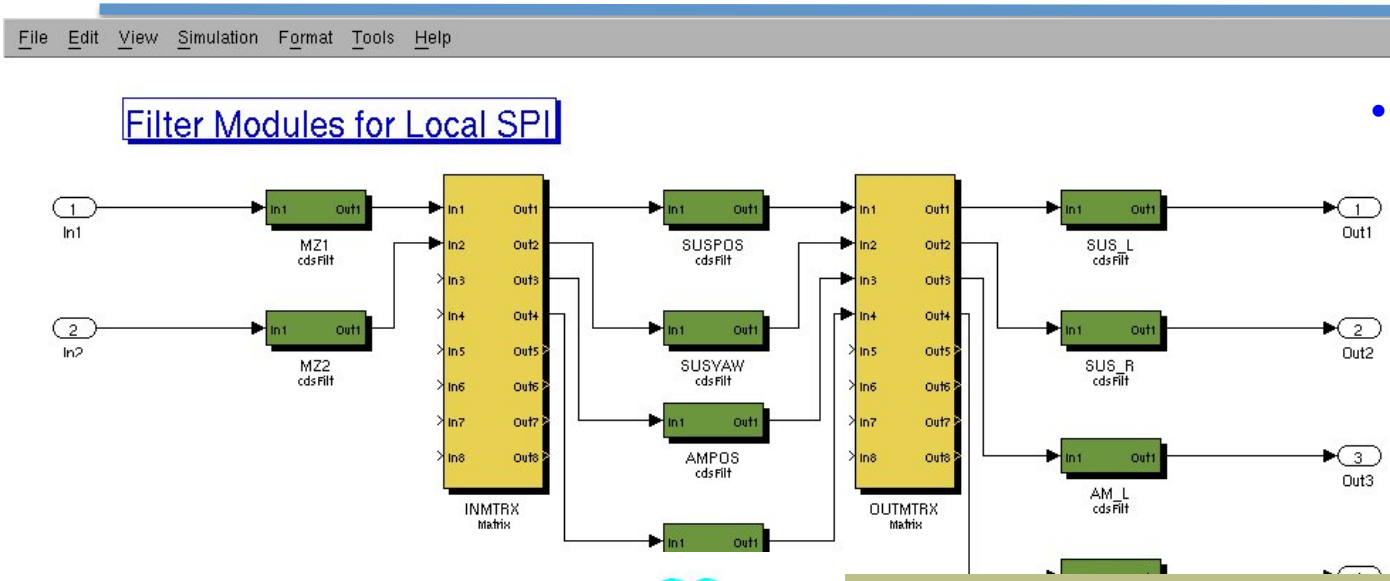


Long term data

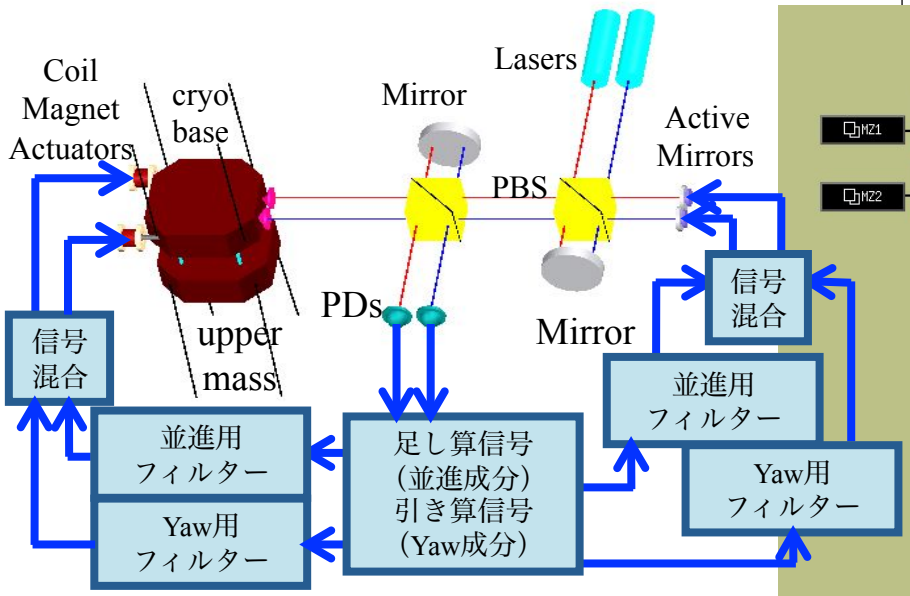


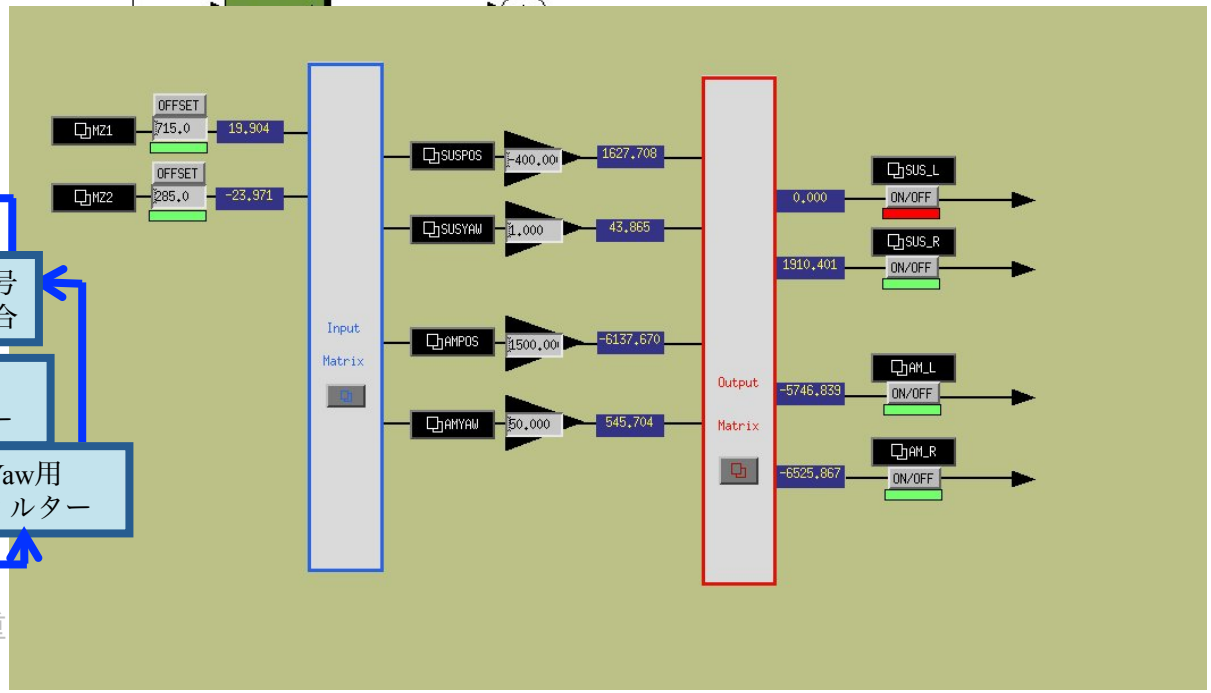
- DAQ channelについては過去のデータも見える。
- 例えばここでは15時間分のデータ。夜中3時頃に腕の反射光の電圧が突然変わっているのがわかる。
- 過去のデータは時系列のみでなく、FFTでも見える。

Local SPIの制御

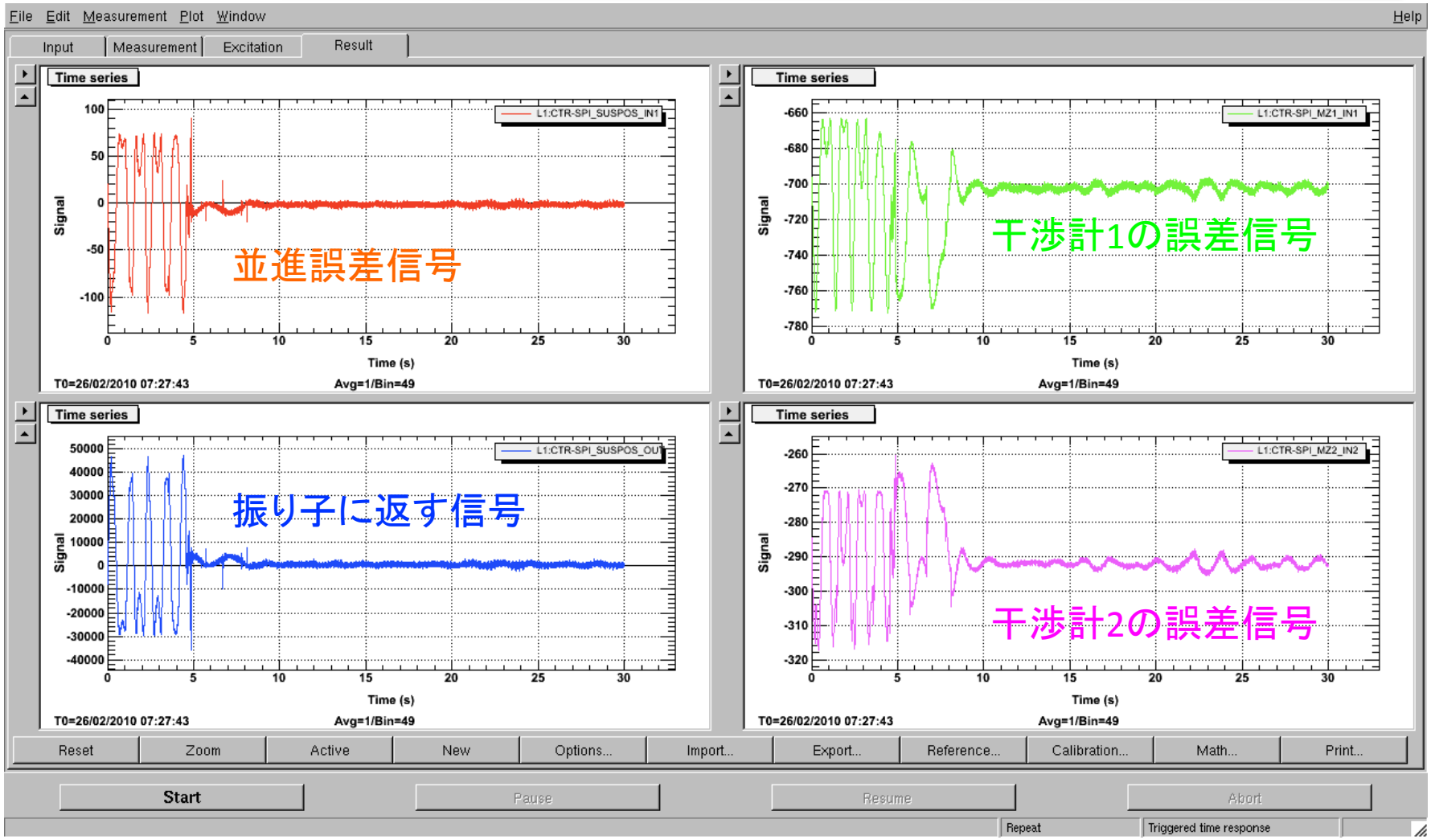


- 昼位から30分くらいで作り、午後には、ロックまでこぎついた。ゲインの微調整ができるので、最適なパラメーターに持っていきやすい。





LSPI


File Edit Measurement Plot Window
Help
Input
Measurement
Excitation
Result
Time series
L1:CTR-SPI_SUSPOS_IN1
Signal
Time (s)
T0=26/02/2010 07:27:43
Avg=1/Bin=49
並進誤差信号
L1:CTR-SPI_MZ1_IN1
干渉計1の誤差信号
L1:CTR-SPI_SUSPOS_OUT
振り子に返す信号
L1:CTR-SPI_MZ2_IN2
干渉計2の誤差信号
Reset
Zoom
Active
New
Options...
Import...
Export...
Reference...
Calibration...
Math...
Print...
Start
Pause
Resume
Abort
Repeat
Triggered time response

その他便利なツール群

- ezcaservo
  - 透過光やPDのオフセット自動調整等、ある目的の値にパラメータを自動的に近づける
- ezcademod
  - デジタル変調復調
  - (例)ロック後、鏡をデジタル的にゆらして変調 → そこから出る信号を復調→復調信号からアライメントを最適にするように制御
  - Matrixの成分やPhaseの最適化等もが可能
- その他様々な制御コマンドをスクリプト(sh, perl, python等)に記述することにより、干渉計オペレーションの自動化が可能
  - 例: ロックアクイジションスクリプト、ダウンスクリプト、アラインメント自動調整、復調位相自動調整、ビームセンタリング自動調整等
- burt
  - 干渉計パラメーターの記憶（例えば10分おき）と回復
- conlog
  - 干渉計パラメータがいつ変わったかを調べる
- Noisebudge
  - DiaggiuiとMatlabの組み合わせで、定期的にノイズ源を測定することにより、どこのノイズが感度に効くか推定できる

CLIOでこれから開発すべき項目

- Analog制御で出した感度をDigital制御で再現する
- Initial alignment自動調整
- 感度をモニタしながらのビームセンタリング
- 常時感度モニタ
- 干渉計制御自動化
- レーザーパワー、地面振動、温度、湿度等長期モニタ
- 定期的な自動Noise budget
- Arm信号を使ってのMC光路長制御

CLIOでのボランティア大募集

PDなどの光学系整備、回路、MEDMのスクリーン、
スクリプト書き、リアルタイムコード等
大量の仕事有り

Digital system
開発に興味がある人
のための説明

PCのスペック

Main PC

- SUPER MICRO社製PC、3GHz 4core Xeon x 2
- PCIe bus + Expansion chassis(20 PCIe bus, ただし使えるのは一つのPCにつきカード14枚くらいまでらしい)
- General Standard社製ADC/DACカード
- CONTEC社製Binary Outputカード(未テスト)
- CentOS 5
- Real time linux kernel (Wind River社製)
- Matlab + Simulink (リアルタイムコード生成)

その他PC(これから整備)

- NFS,NIS server
- Monitor用PC
- Gatewayを使ってfirewall内で運用

速いループ: 16kHz(128kHzまで可)

- Mail loop: xxx(モデル名)fe.rtl
  - Wind River社製のReal time Linux上で動く
  - MatlabのSimulink上のGUIで書き、cのコードに変換され、実行ファイルにコンパイルされる
  - 基本パーツはfilter, gain, switch, matrix, test point, excitation, 及び四則演算等

```
> more /cvs/cds/llo/target/l1ctr/startup.cmd
sudo  /cvs/cds/llo/target/l1ctr/ctrfe.rtl  >  /cvs/cds/llo/target/l1ctr/log.txt 2>& 1 &
```

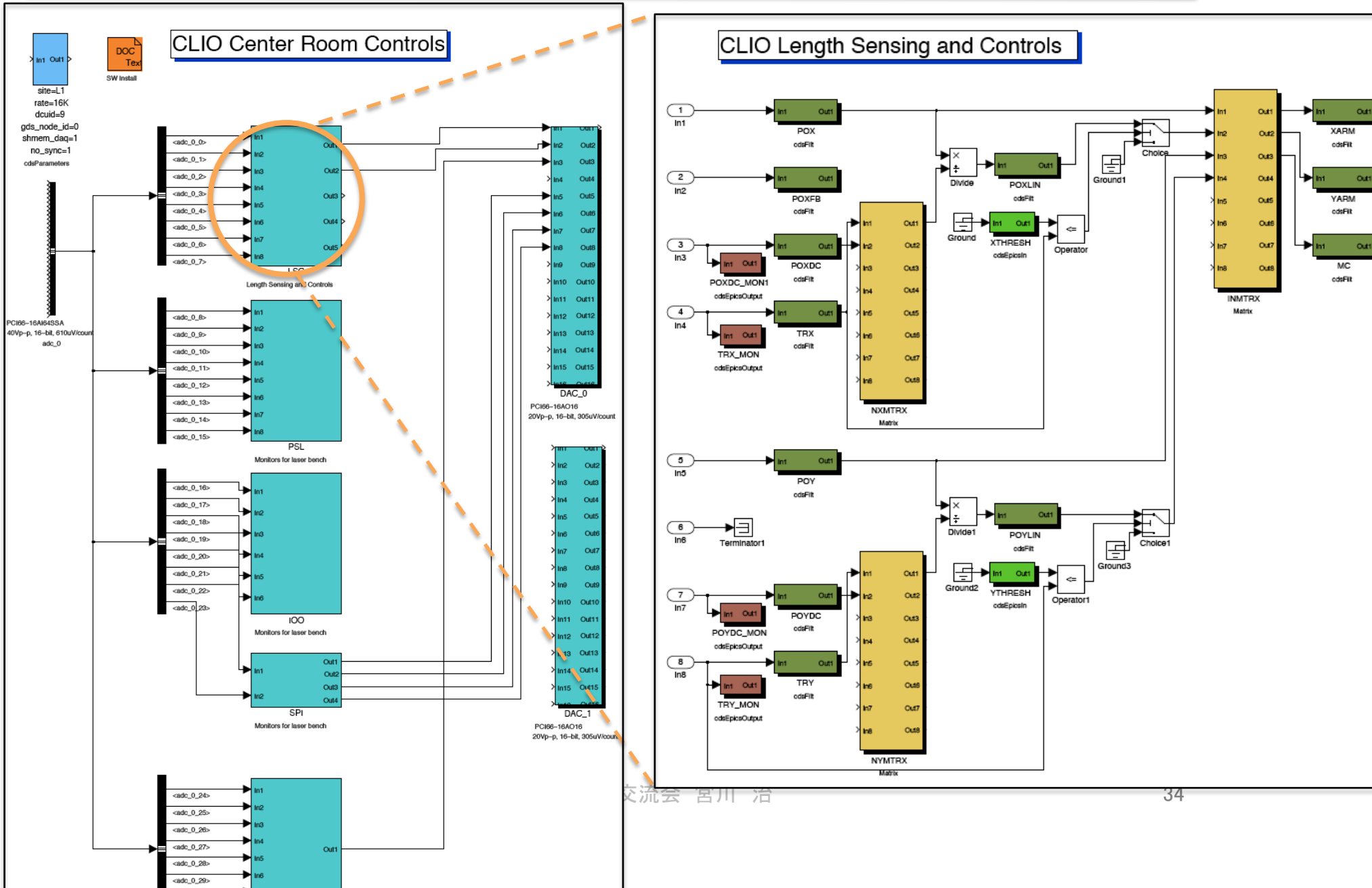
↑ctrというモデルの実際のリアルタイムルーチン

- コンパイルの様子

```
> make ctr
> make install-ctr
> make install-daq-ctr
> make install-screens-ctr
> startctr
```

でリアルタイムコードが走り出す。

Real time code on Matlab, Simulink



自動生成されるCのReal time code

実際のコントロールシグナル（filter bank, matrix, trigger, linearization etc.）のcのコード
cds/rts/src/fe/ctr/ctr.cが自動的に生成され、ctrfe.rtlがコンパイルされる

```
 .
 .
//Start of subsystem LSC ************************************************
// FILTER MODULE
lsc_pox = filterModuleD(dsp_ptr,dspCoeff,LSC_POX,dWord[0][0],0);

// FILTER MODULE
lsc_poxfb = filterModuleD(dsp_ptr,dspCoeff,LSC_POXFB,dWord[0][1],0);
.
.
for(ii=0;ii<1;ii++)
{
    lsc_nxmtrx[1][ii] =
    pLocalEpics->ctr.LSC_NXMTRX[ii][0] * lsc_trx + pLocalEpics->ctr.LSC_NXMTRX[ii][1] * lsc_poxdc;
}

// Relational Operator
lsc_operator = ((pLocalEpics->ctr.LSC_XTHRESH) <= (lsc_trx));

// DIVIDE
if(lsc_nxmtrx[1][0] != 0.0)
{
    lsc_divide = lsc_pox / lsc_nxmtrx[1][0];
}
else{
    lsc_divide = 0.0;
}
```

その他、速いループのプログラム

一般ユーザーは以下のことはあまり気にしなくていい

- epicsioc ←Epics本体

```
> more /cvs/cds/llo/target/l1ctrepics/startupL1
#!/bin/sh
./ctrepics ctrepicsL1.cmd >& iocL1.log &
echo ctrepics L1 IOC Server started
```

- daqd ←Data acquisition deamon
- NDS

```
> tail -3 /etc/inittab
fb:35:respawn:/cvs/cds/llo/target/fb/start_daqd.inittab
nds:35:respawn:/cvs/cds/llo/target/fb/start_nds.inittab
```

- tpman ←Test Point（後述）マネージャ
- awgtpman ←Arbitary wave form generator(計算機内の任意波形の発信器) マネージャ

```
> more /cvs/cds/llo/target/gds/startup_ctr.cmd
#!/bin/bash
cd /cvs/cds/llo/target/gds; sudo /cvs/cds/llo/target/gds/bin/awgtpman -s ctr  > ctr.log 2>& 1 &
```

Data channelの種類

Epics: 計算機上の情報(メモリ上のアドレスなど)と、人間ができる情報(チャンネル名)を結びつけてくれるシステム。ユーザーはチャンネル名だけ把握していれば、その情報元がどこにあるのかを気にしなくていい。

- Epics channel: リアルタイムデータを1/64秒ごとに読み取る遅いチャンネル。多数（数百）のチャンネルがあり、主にMEDMなどで使用。
- Test Point:リアルタイムに流れているデータを覗くために一時的に作るチャンネル。テストポイントの数には制限(24個)がある。数に応じてCPUに負荷がかかる。見えるのは現在のデータだけで、過去にさかのぼっては見えない。
- AWG: Excitation(計算機上の発信器、もちろんDACから信号を出すこともできる)を使うと、AWG Test Pointを一つ使用する。同じくテストポイントの数には制限(8個)がある。
- DAQ channel: test pointを無駄に使用しないように、あらかじめ決められたデータをディスク上にFrame dataとして保存しておく。データの読み込みにCPUパワーを食わない。過去のデータをさかのぼってみることができる。

Real time codeの情報

デジタルシステムに必要なアナログ回路開発

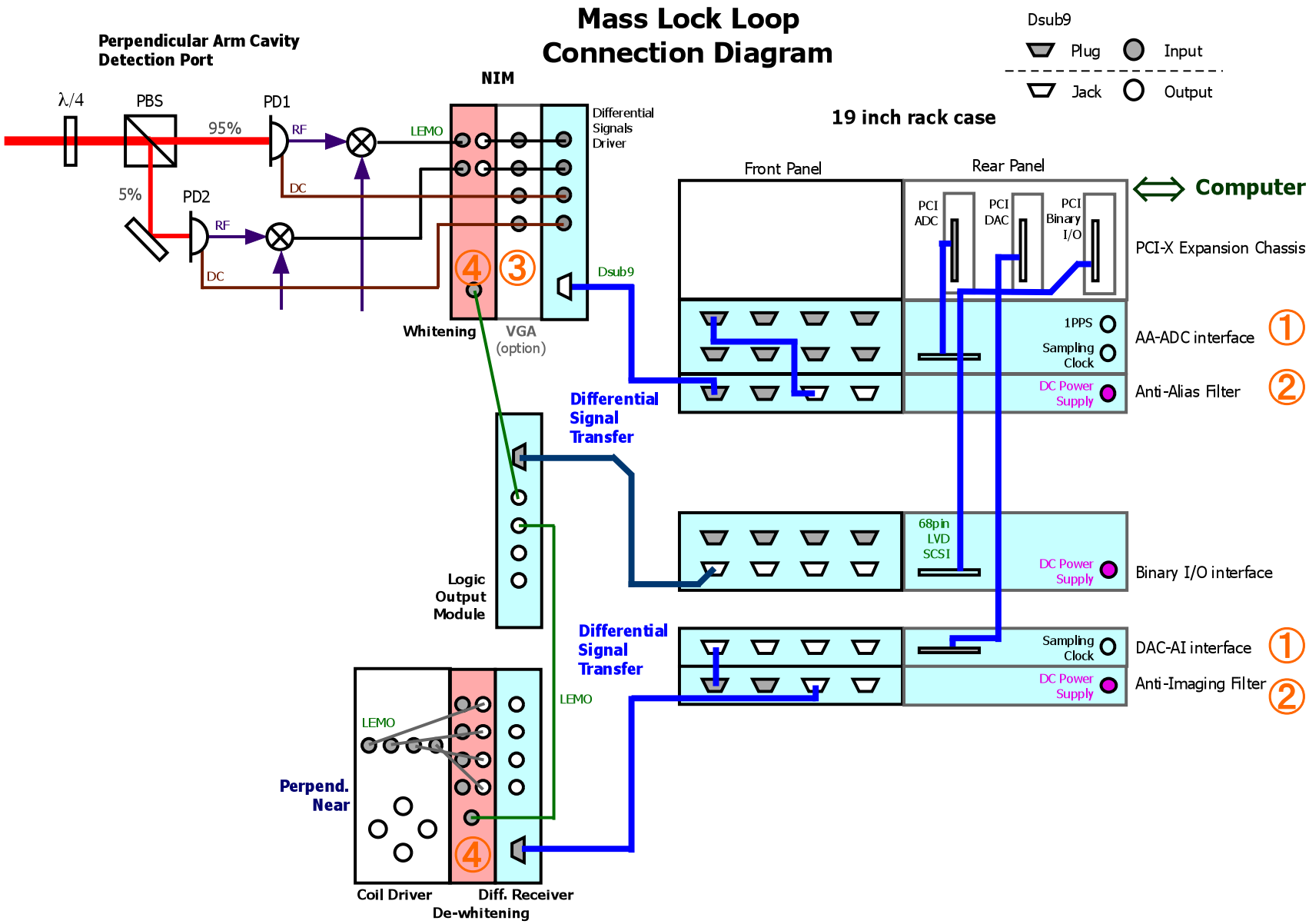
開発済み

1. ADC/DAC to AA/AI interface (Ultra SCSIケーブルで接続)
2. Anti Aliasing (AA) /Anti Imaging (AI) filter
3. Differential signal driver/receiver (ADC/DACは差動信号を扱うが、CLIOはシングルエンドのLEMOコネクタが一般のため)
4. Whitening/De-whitening filter（ADC/DACともにノイズが大きいため）

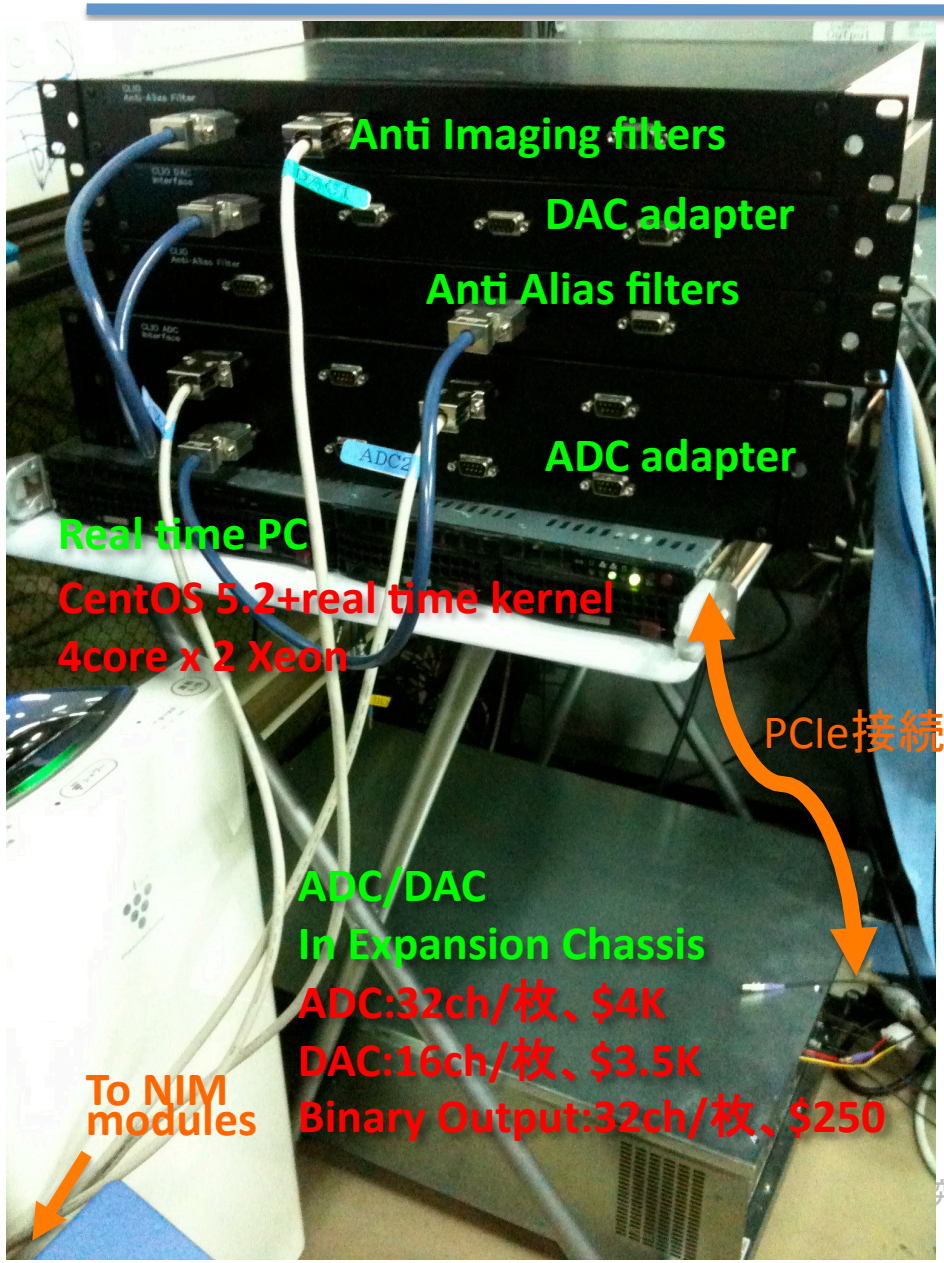
更に必要なもの

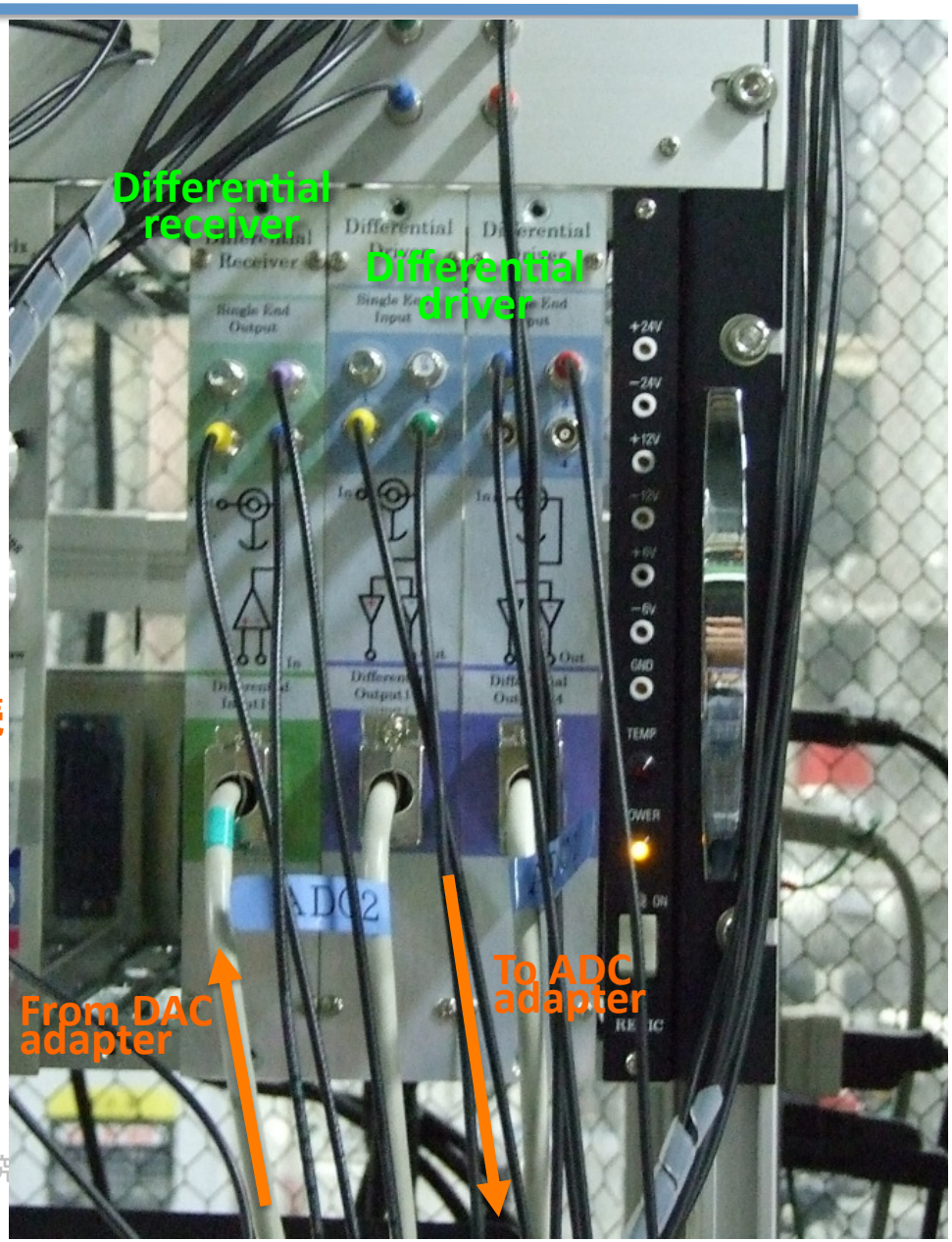
5. Binary output(0V-5Vのスイッチ)とそのインターフェース、及びそれをコントロールするソフト→Whitening/Dewhiteningが動くようになり、CLIOの感度を検証できる
6. Variable Gain Amp.
   - デジタルからのゲインコントロール部分を切り離した試作機を作る: 試作機完成
   - 後に、デジタルからコントロールできるものを作る: 未着手
7. Timing System
   - 現在使っているものは仮のもの、Master Slave構成のファイバーで接続された、遅延なども補正できるものが必要(LIGOから入手予定）

CLIO Digital Front-End



Pictures

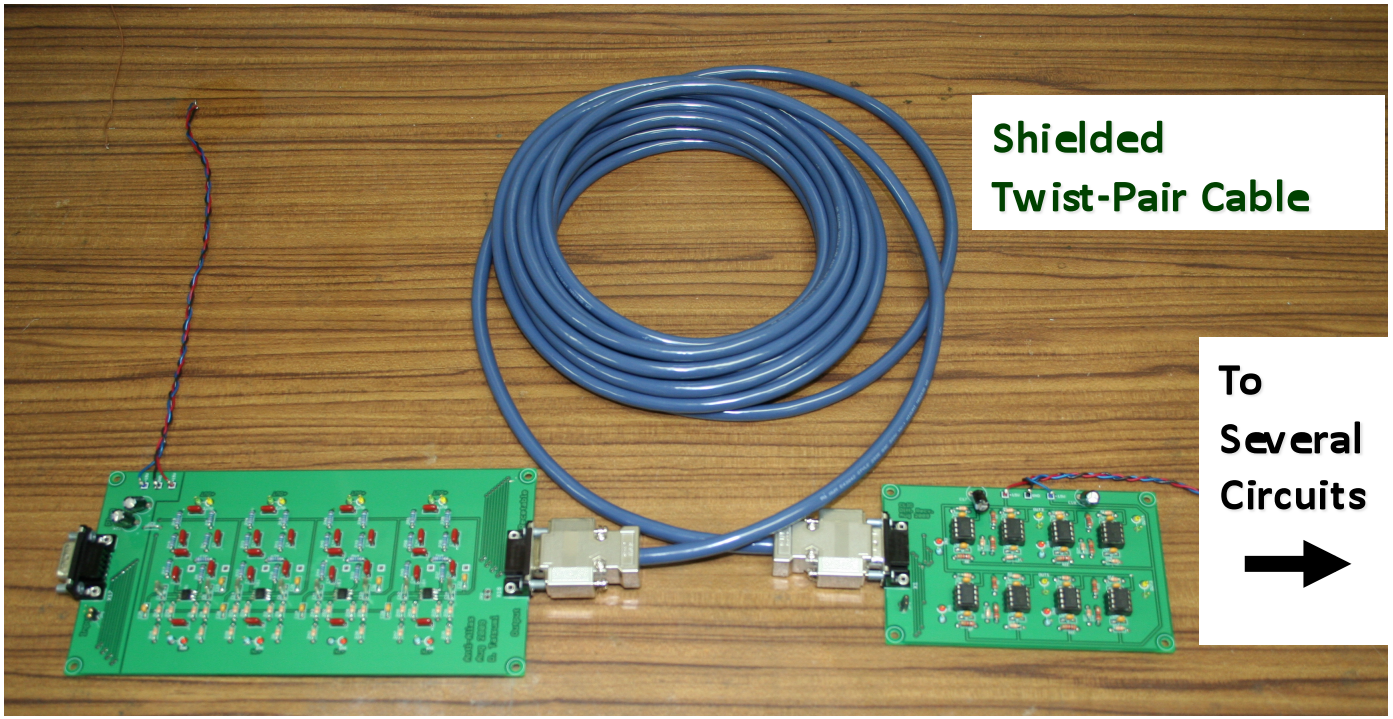

Anti Imaging filters
DAC adapter
Anti Alias filters
ADC adapter
Real time PC
CentOS 5.2+real time kernel
4core x 2 Xeon
PCIe接続
ADC/DAC
In Expansion Chassis
ADC:32ch/枚、$4K
DAC:16ch/枚、$3.5K
Binary Output:32ch/枚、$250
To NIM modules


Differential receiver
Differential driver
From DAC adapter
To ADC adapter

Differential Signal Transfer with AA/AI

- Cut off frequency of AA/AI:10kHz、回路はAA/AIで共通化
- ADC Sampling rate: 64kHz
- 計算機内で16kHzにdecimation（CPUパワー軽減のため）
- 電源からのLineノイズの影響を軽減するために差動伝送方式を採用

From DAC

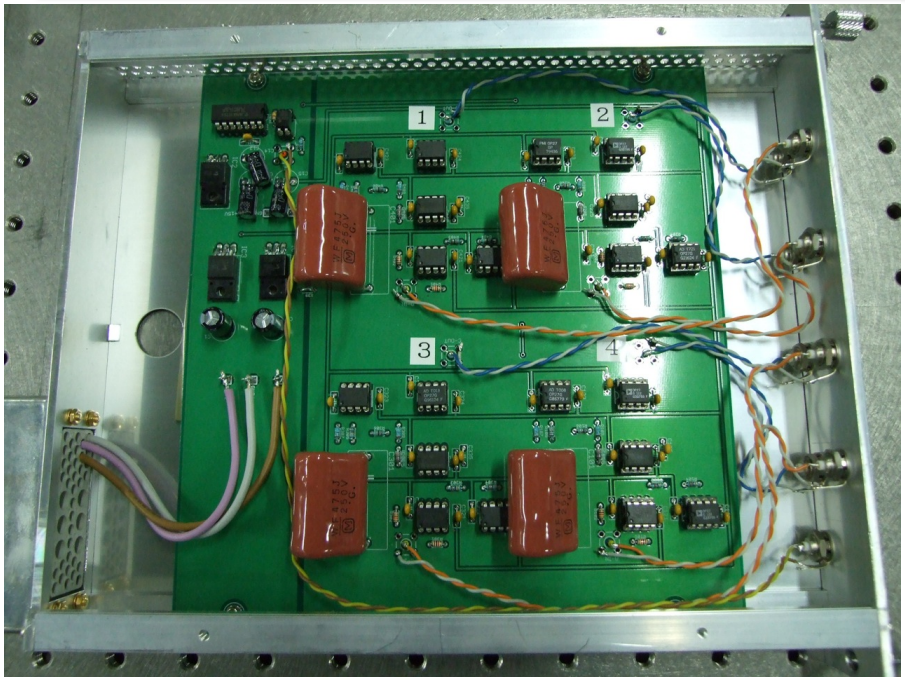


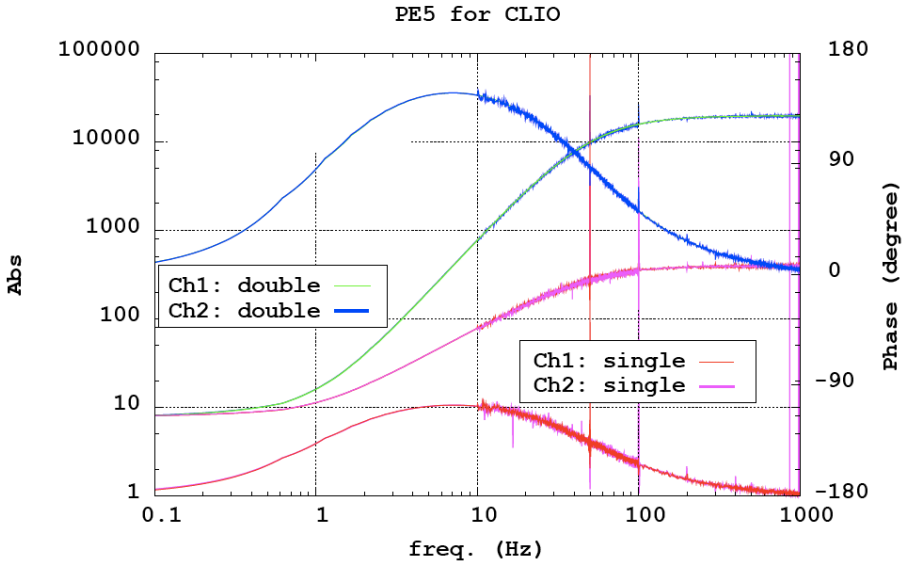
To Several Circuits

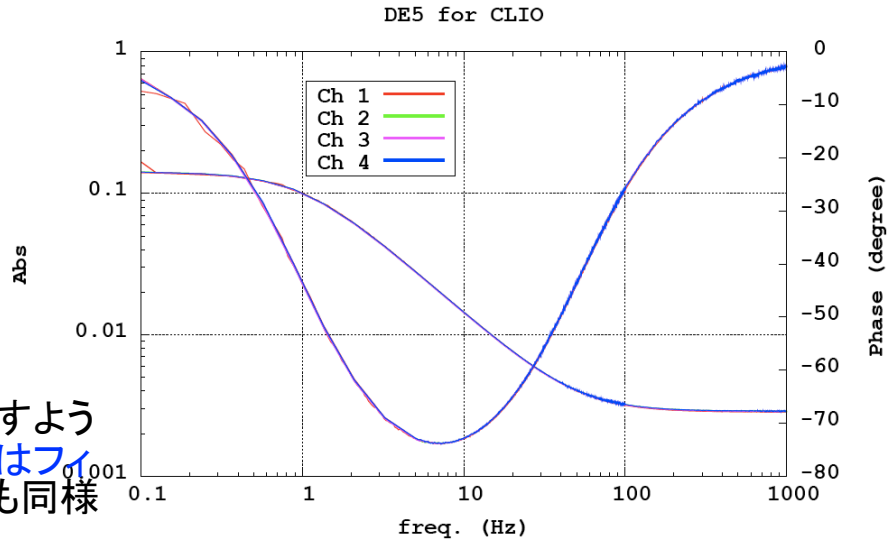
Anti-Alias (Smoothing) Filter + Driver

Signal Receiver

Whitening/dewhitening filter







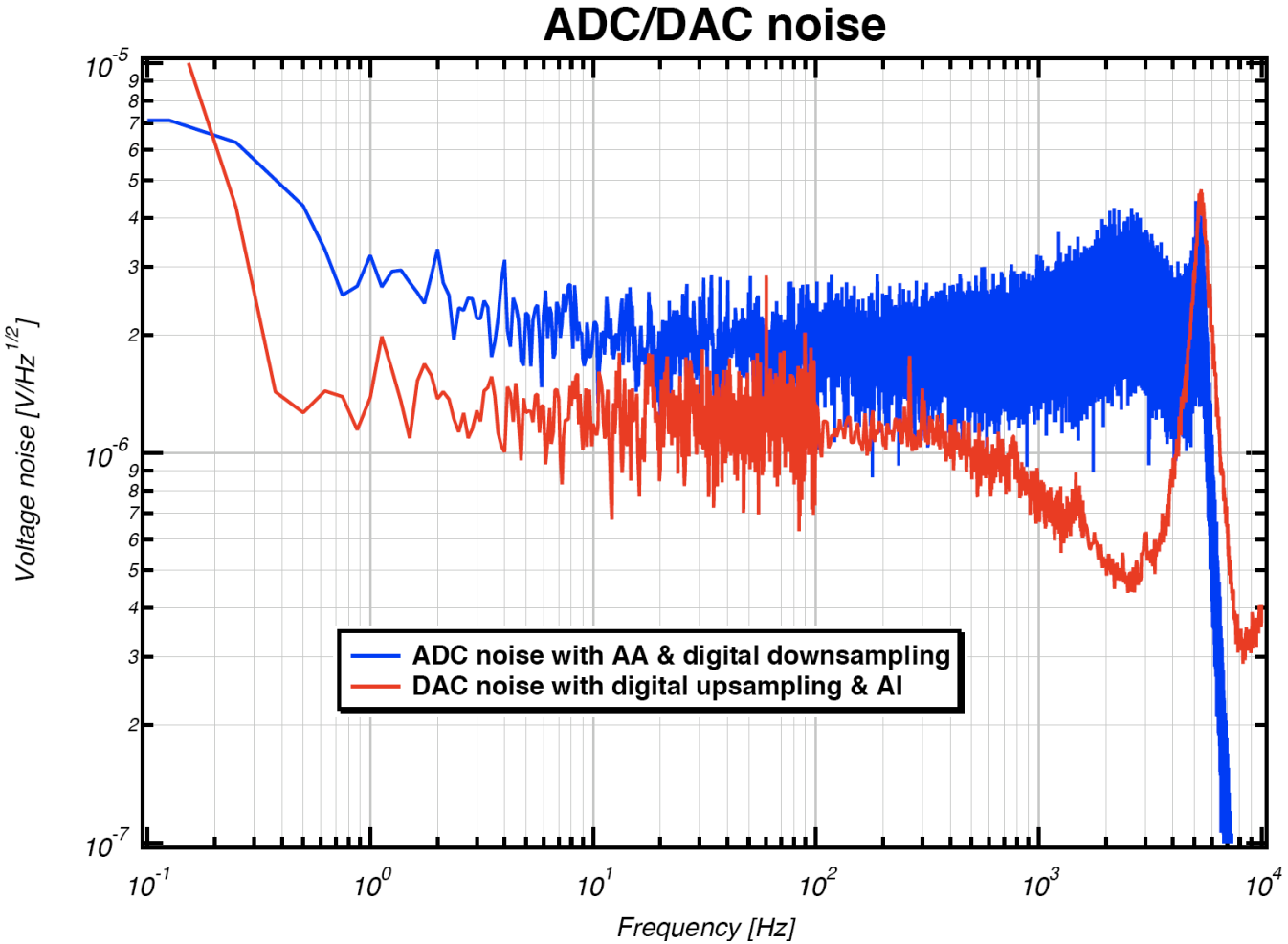
Pole 1Hz, Zero 50Hz, 一段 or 二段切り替え可能

Measured noise performance

- 3.5nV/rtHz @ 20Hz / Whitening filter input
- 5.0nV/rtHz @ 20Hz / De-Whitening filter output

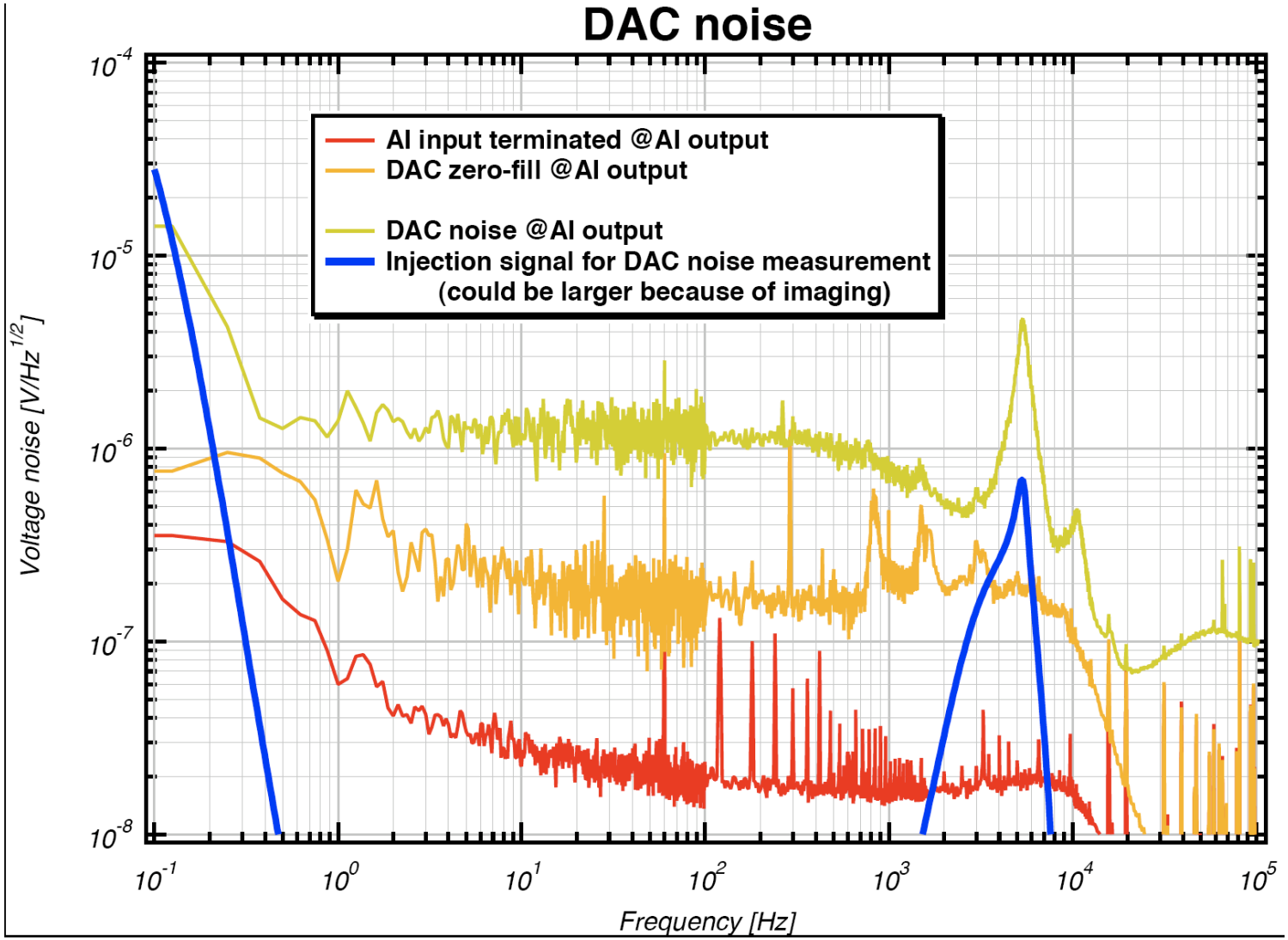
デジタル入力直後に、アナログwhitening filterを打ち消すようなdigital anti whitening filterを入れるので、計算機内ではフィルタリングされていない通常の信号として扱える、出力も同様にdigital anti dewhitening filterを通して出力

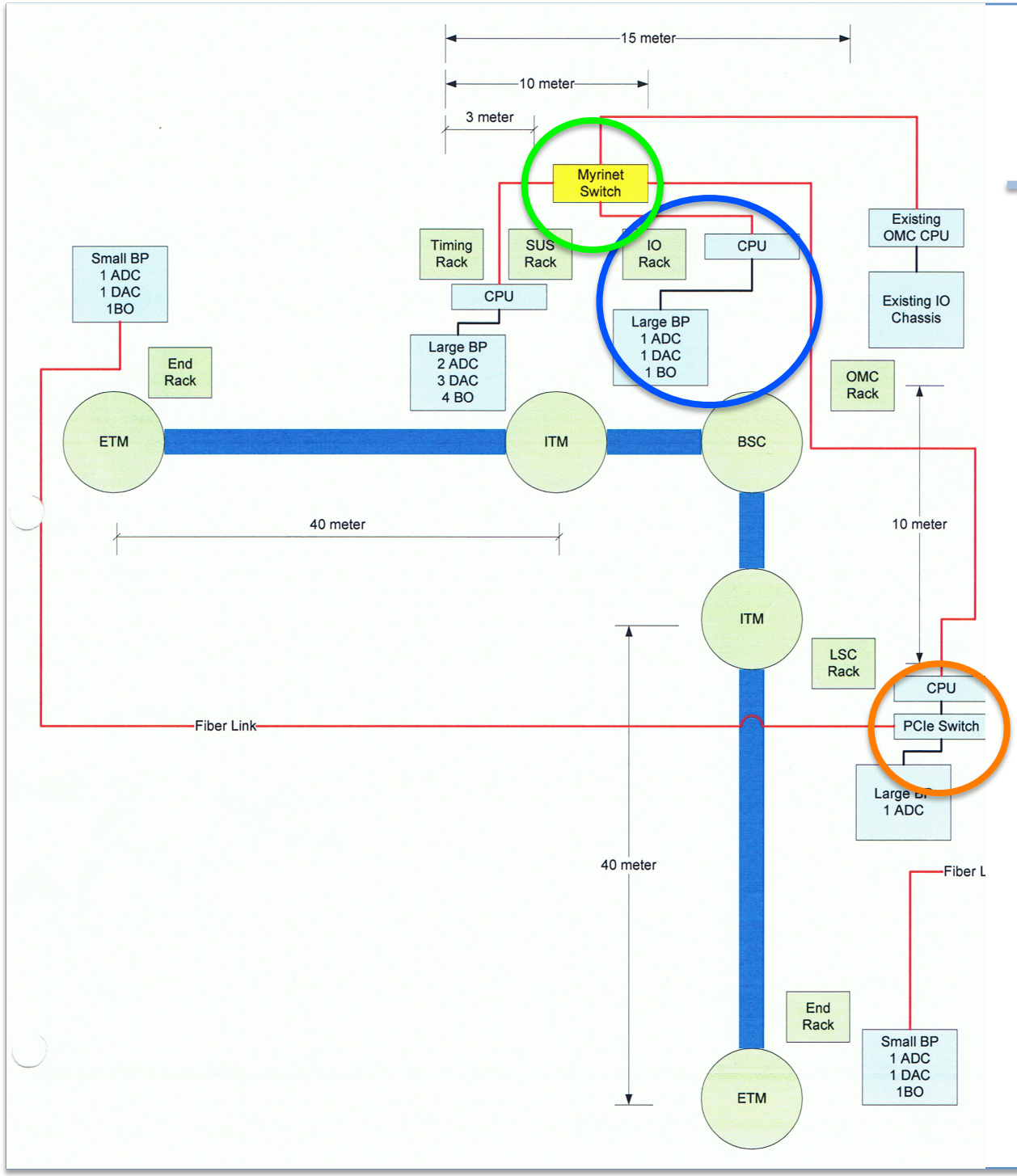
ADC/DACのノイズパフォーマンス



- ADC/DACともに数μV/√Hz程度とかなり良い
- ADCの数kHzの盛り上がりと5kHz以上がカットされているのはsoftware AAのため
- DACの1kHz以上は入力信号によるピークが出ているため正確でない
- 他に目立ったピークもなく、非常に素性の良いADC/DACである
- whitening、dewhitening filterを導入することにより、高周波でのノイズを軽減

DAC noise with AI filter





LCGTに向けて

大型化にともない、いくつか開発しなければならない技術がある（LIGOでも未解決）

- 今回のCLIO1st の規模
- Endまで届けるのに必要
  - PCIe switch
  - Optical adapter
  - Optical cable
  - Timing master/slave
- 複数台の計算機をつなぐのに必要
  - Myrinet switch?

- Kmクラスの接続
  - Reflective memory?
- 大容量データストレージ

cost

項目	値段(万円)	コメント
Multi core PCIe based PC	55/一台	
拡張ボックス	60/一台	最大8枚
ADC	40/枚	32ch/枚
DAC	35/枚	16ch/枚
Binary Output	2.5/枚	32ch/枚
Matlab/Simulink	12	
Real Time core	110	日本で買うと価格up

項目	値段(千円)	コメント
Analog回路	15/4ch	Inputからoutputまでを1chとする
Oscillator	30	
Timing master	20?	
Timing slave	10?/1station	
Fiber adapter, cable	20 for 100m	100mは拡張可能だが、kmはまだ未検証
Switching hub	60	

Example

1. Table top実験での例(ADC,DAC,BO各1枚の場合): 約400万円
   - PCIeを3枚させるPCを見つけて、フリーのReal time coreを使い、回路を自作し、発信器を既存のものにすれば150万円程度できるかもしれないが、まだ未検証
2. CLIOクラスでエンドルームに拡張boxをしない場合(センターからエンドまで普通のケーブルで持っていく、ADC,DAC,BO各3枚): 約550万円
3. CLIOクラスでエンドルームに拡張boxを設置(ファイバーケーブル、ADC,DAC,BO各4枚):約800万円

Summary

- CLIOでデジタル制御システムが稼働し始めた
- 実験の能率が劇的に上がることがすでに実証され始めている
- アナログのCLIOの感度をデジタル制御で出すことが次の当面の目標
- ビームセンタリングまでできるとCLIOでの実用度が一気に高まってくる

- LCGTのプロトタイプとして開発を続けていくことが大切である