MATLAB®で信号処理
～各種フィルタ設計を題材として～

MathWorks Japan
アプリケーションエンジニアリング部
竹本佳充

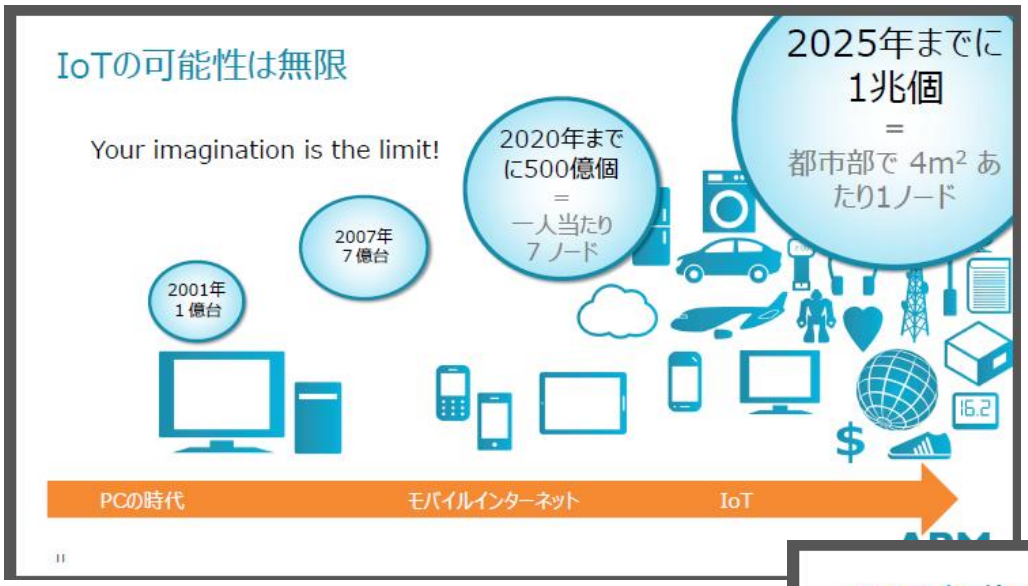
Agenda

- Section1: フィルタとは？
- Section2: Case study
- Section3: フィルタの実装
- Section4: フィルタ設計FAQ
- まとめ

Agenda

- Section1: フィルタとは？
  - 背景
  - フィルタの用途・課題
  - ユーザの声
  - フィルタの種類
  - MATLABによるフィルタ設計手法

背景：IoT時代のセンサ信号処理

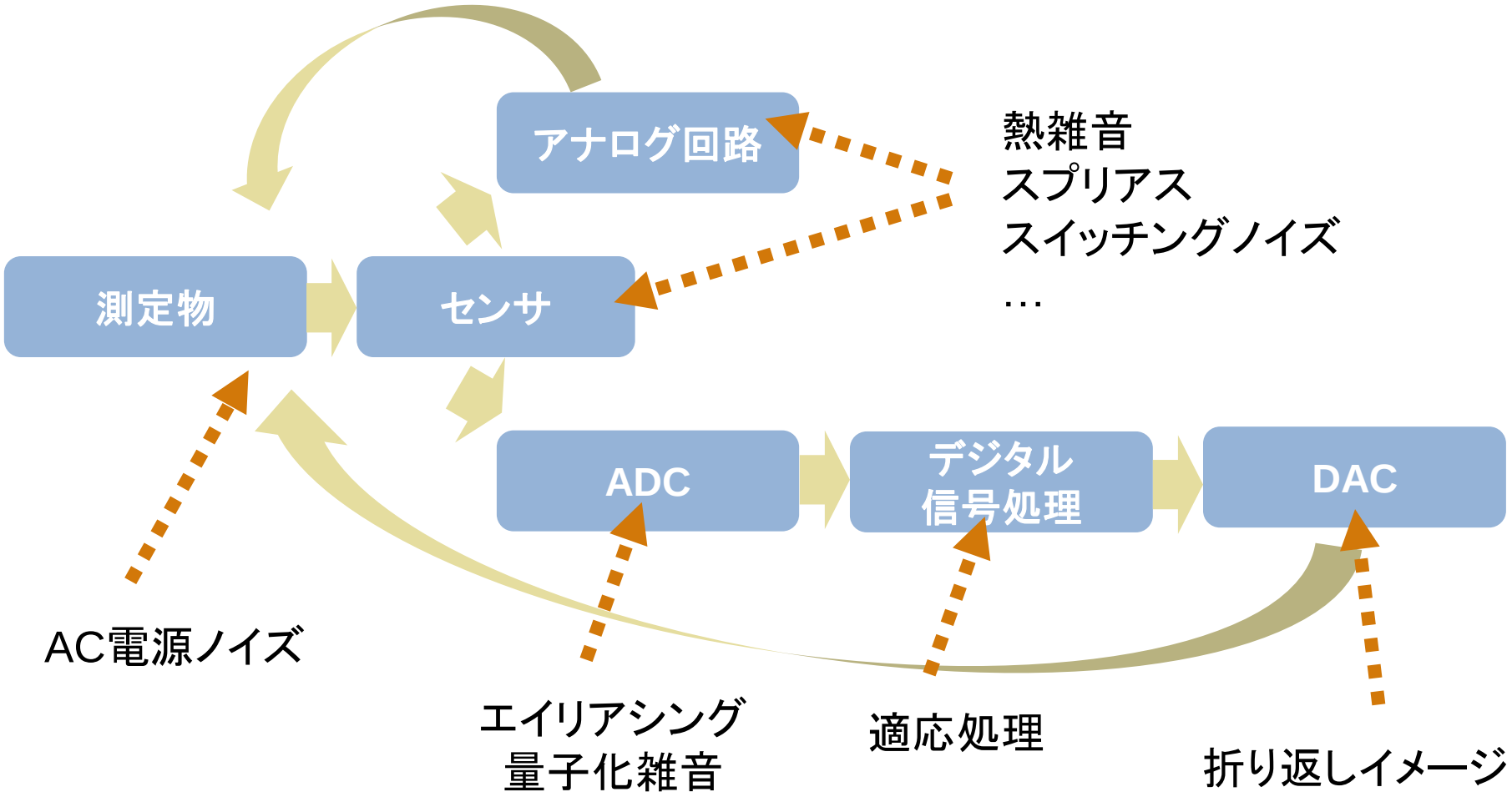


- センサの組み込み製品への応用
- ノード数の飛躍的な増加
- 2020年東京オリンピック

センサ信号処理は
キーテクノロジー

モデルベースで実践！組み込みDSP開発セミナー
（2014/9/10実施、ARM, STMicroelectronics, MathWorks三社共催セミナー）ARM社資料抜粋

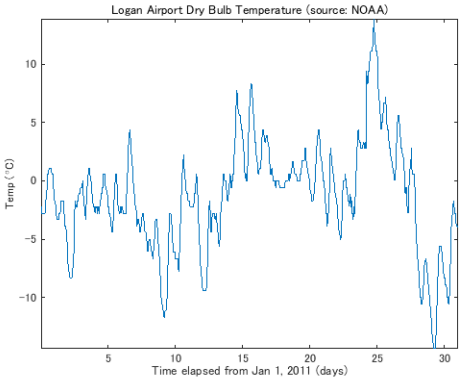
背景: センサシステムにおけるフィルタリング

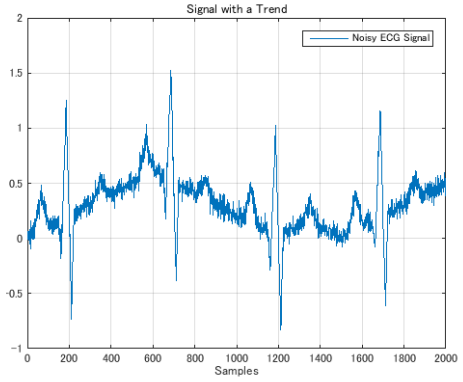


主にフィルタ処理を題材とします

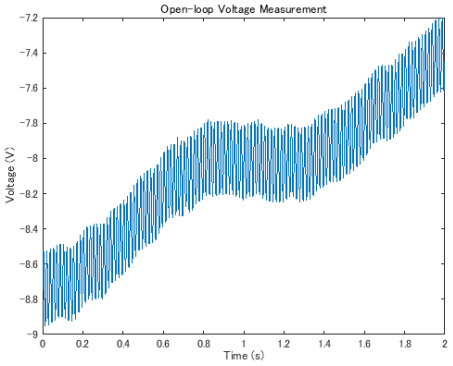
フィルタの用途・課題

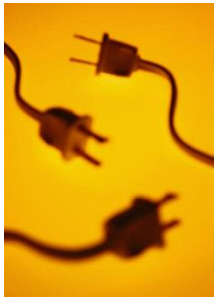
トレンド成分
⇒トレンド成分のみ取得したい
⇒トレンド成分は除去したい



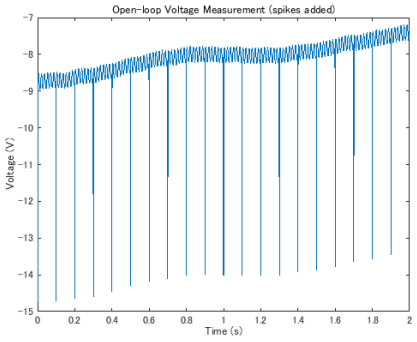


電源ノイズ
取得データに重畳
⇒信号処理で切るしかない





スパイクノイズ
通常のフィルタでは除去しきれない
⇒選択するフィルタに工夫が必要

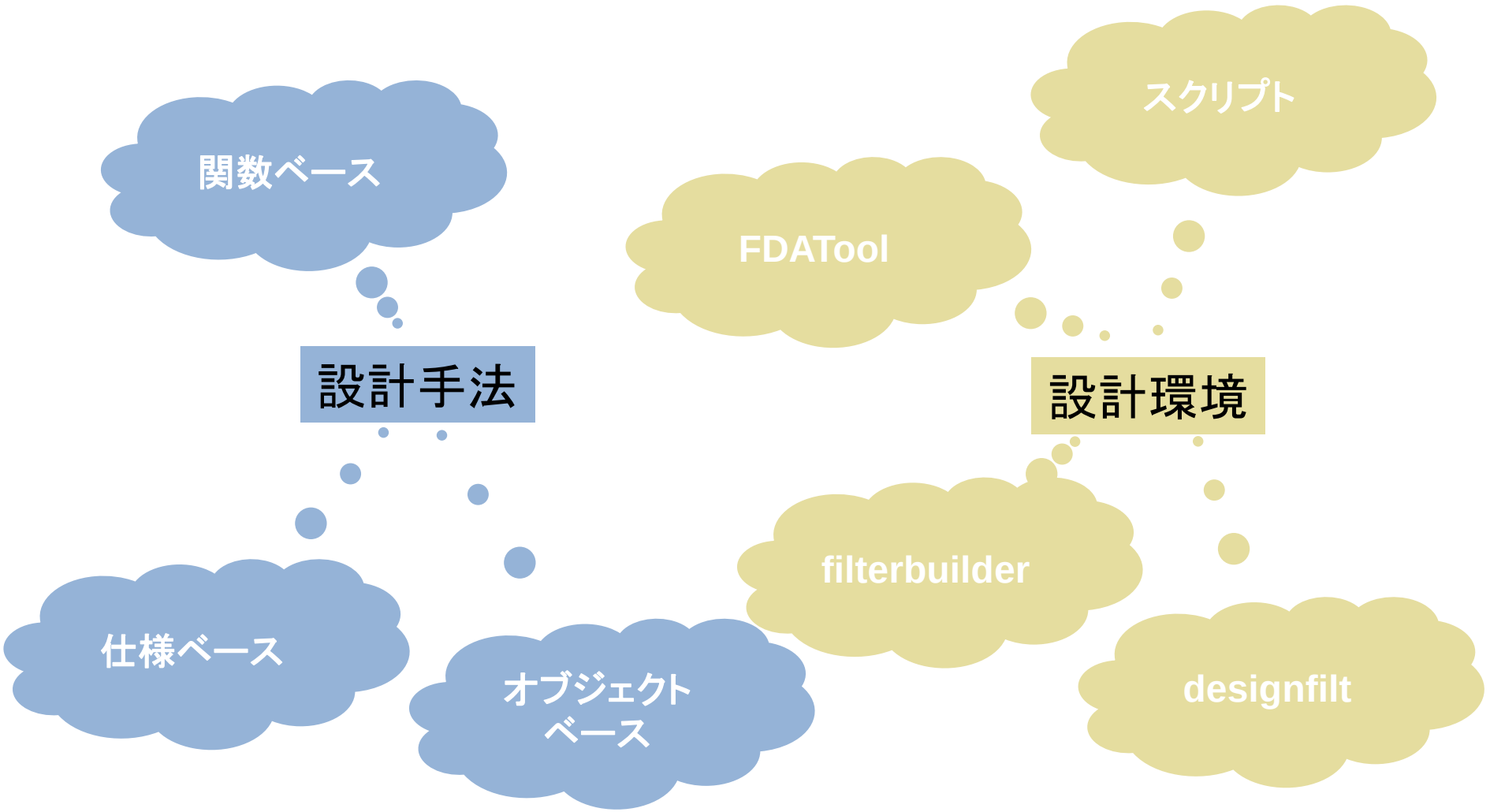


フィルタの用途・課題（cont'd）

- データ解析
  - 多様なフィルタ手法を検討・比較したい
  - 試行錯誤を減らして効率を上げたい
  - 既存技術はツールにまかせ、独自アルゴリズム検討に集中したい
  - 大規模データを扱いたい
- システム設計
  - テストベンチ作成に時間がかかる
  - 量子化の影響を検証したい
  - 安定性を評価・確保したい
- 実装
  - 次数が高すぎてH/Wに実装できない
  - シミュレータで計算した係数を他言語にマニュアルで移植している

すべてMATLAB/Simulinkで解決・実現可能

ユーザーの声


関数ベース
スクリプト
FDATool
設計手法
設計環境
仕様ベース
オブジェクト
ベース
filterbuilder
designfilt

ユーザーの声(cont'd)

```
>> Hs = fdesign.lowpass
Hs =
            Response: 'Lowpass'
       Specification: 'Fp,Fst,Ap,Ast'
         Description: {4x1 cell}
NormalizedFrequency: true
               Fpass: 0.45
               Fstop: 0.55
               Apass: 1
               Astop: 60
>> H = design(Hs)
H =
     FilterStructure: 'Direct-Form FIR'
          Arithmetic: 'double'
           Numerator: [1x43 double]
PersistentMemory: false
```

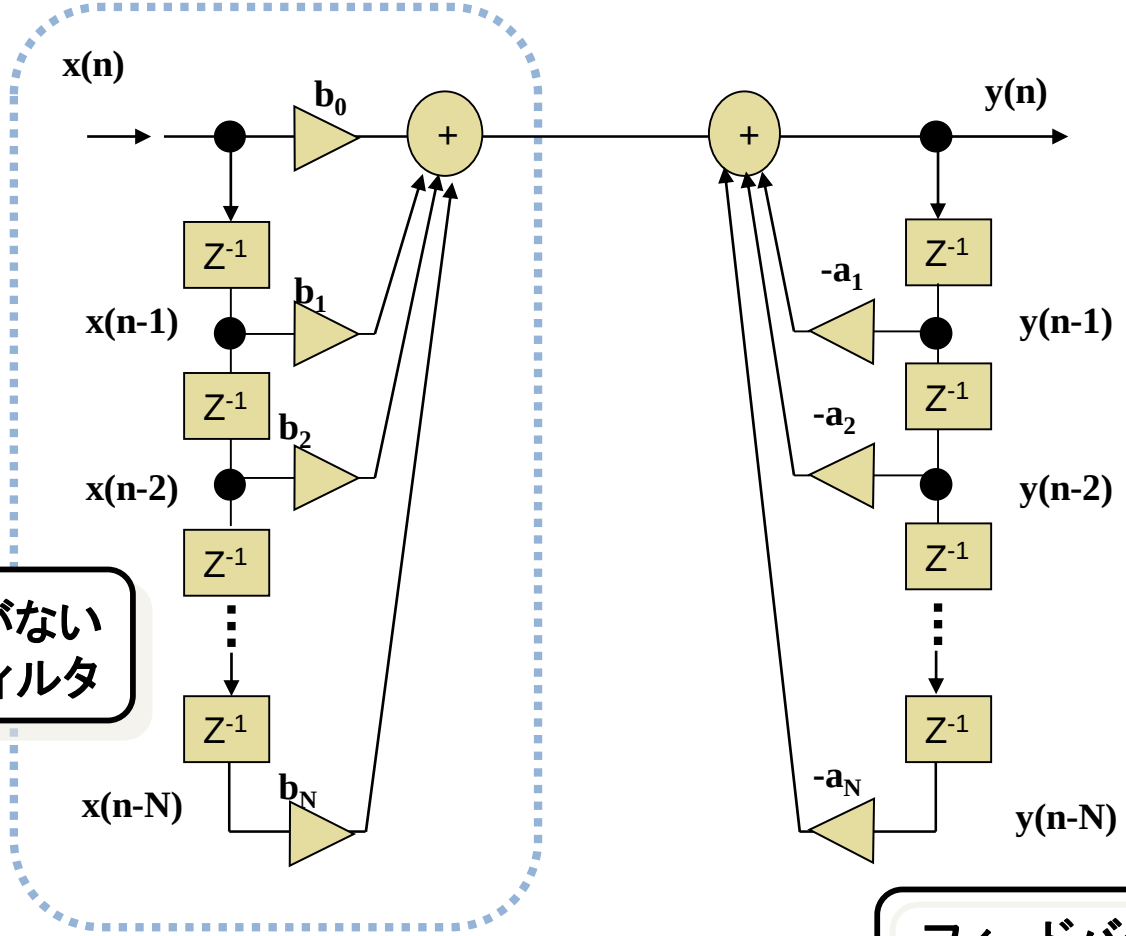
何の略称？

```
>> SOS = H.sosMatrix;
>> y = sosfilt(SOS,x);
```

フィルタの構造によって、
適用する関数が違う

2種類のオブジェクトを
管理する必要がある

MATLABによるフィルタ設計: フィルタの構造

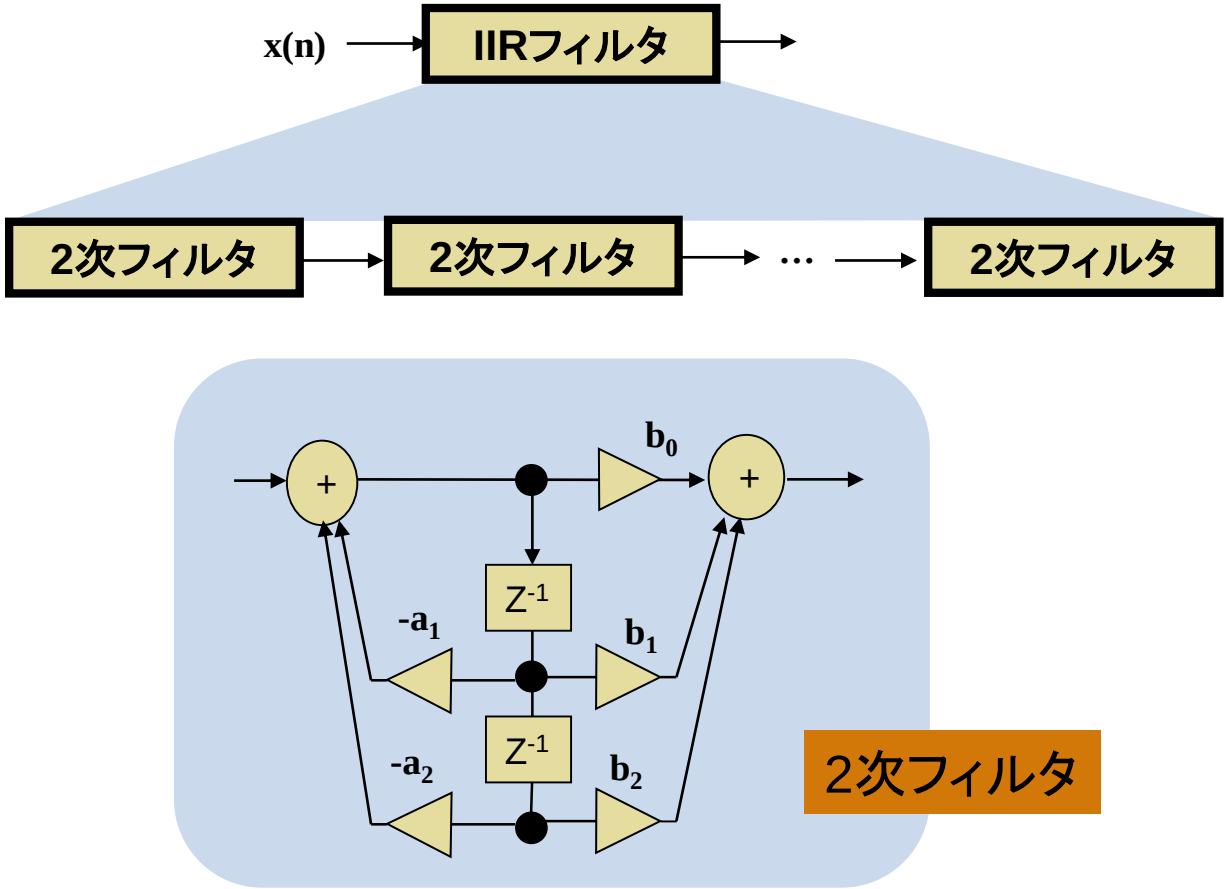


フィードバックがない
タイプ⇒FIRフィルタ

フィードバックをもつ
タイプ⇒IIRフィルタ

フィルタの設計⇒係数を決めること

MATLABによるフィルタ設計:IIR（SOS型）



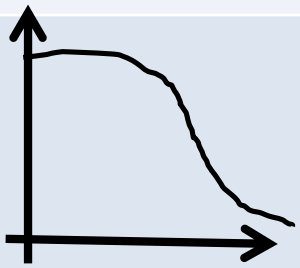
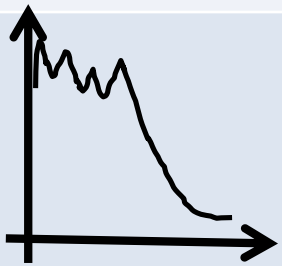
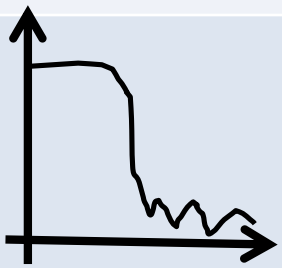
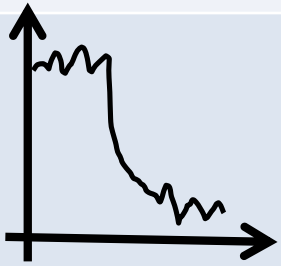
SOS　(Second Order Section, Biquad)型

⇒IIRフィルタの係数感度低減の効果

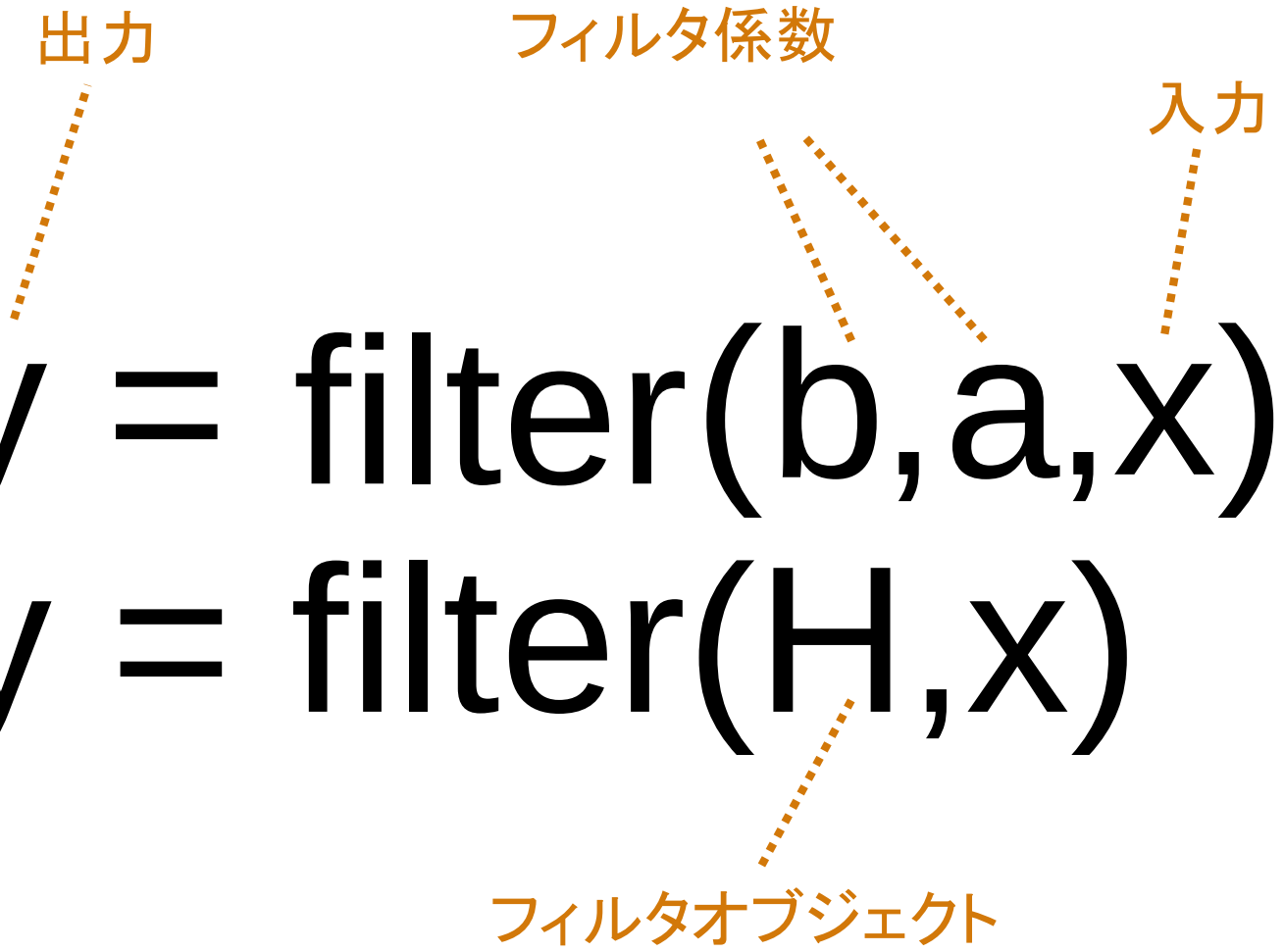
FIRフィルタとIIRフィルタとの比較

項目	FIR (Finite Impulse Response)	IIR (Infinite Impulse Response)
インパルス応答	有限	無限
次数	多い	**少ない**
フィードバック構造	なし	あり
安定性	**安定**	不安定
群遅延特性	**線形**	非線形
H/Wコスト	高い	**低い**
速度	遅い	**早い**

IIRフィルタの種類と特徴

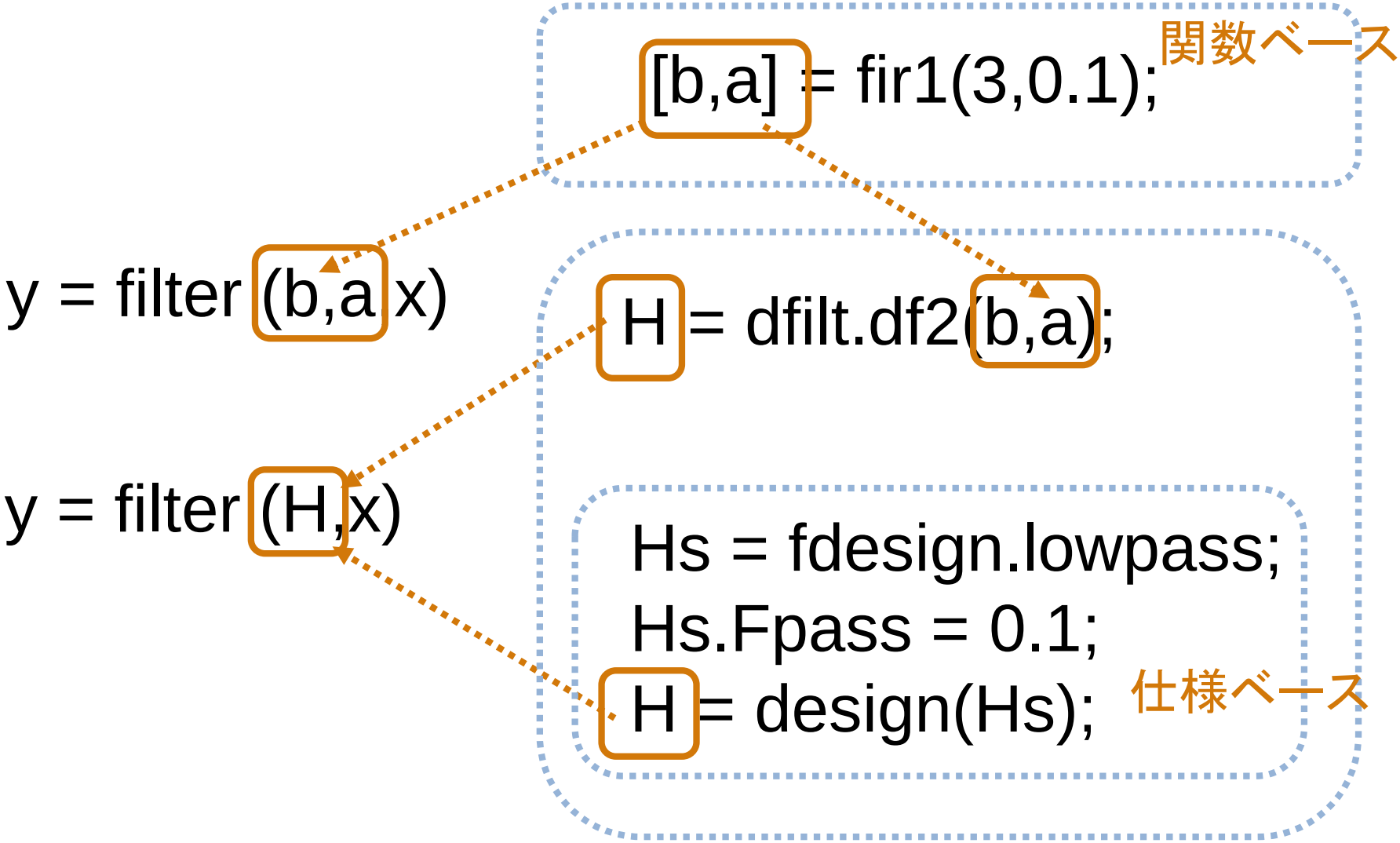
項目	バタワース	チェビシェフⅠ型	チェビシェフⅡ型	楕円
通過域リップル	なし	あり	なし	あり
遮断域リップル	なし	なし	あり	あり
群遅延特性	○	△	△	×
特徴	リップル特性を持たないため、**平坦な周波数特性**が得られる	通過域リップルを大きくすることで、急峻なロールオフ特性が得られる	遮断域の減衰量を小さく指定することで、急峻なロールオフ特性が得られる	通過域と遮断域双方のリップルを制御可能なため、**最も急峻なロールオフ特性**が得られる
波形イメージ				

MATLABによるフィルタ設計: フィルタの実行



filter関数でフィルタリング

MATLABによるフィルタ設計: 設計



関数ベースとオブジェクトベース

MATLABによるフィルタ設計: 手法

- **関数ベース**
  - 従来手法
  - 記述がシンプル
  - 直感的

- **オブジェクトベース、仕様ベース**
  - **フィルタ属性の管理**
  - メソッド選択の試行錯誤削減
  - 仕様オブジェクト作成、フィルタオブジェクト作成の2ステップが必要

- **designfilt**
  - R2014a新機能
  - 仕様ベースの設計手法の手順を簡略化

  (オブジェクト生成作業を1ステップ化、プロパティ名の明確化)
  - フィルタ設計フローのアシスト(足りない引数の候補推定等)

MATLABによるフィルタ設計: オプション

信号処理用
Simulinkブロックライブラリ
- FFT
- 各種フィルタ
- スペクトラムアナライザ
- ライブ音声処理

フィルタ設計用
MATLAB関数ライブラリ
- 適応フィルタ
- マルチレートフィルタ
- 固定小数点フィルタ

R2011aより統合
- Filter Design Toolboxのフィルタ設計機能を踏襲
- 旧Signal Processing Blocksetの機能を、MATLAB環境で実行可能

DSP System Toolboxに各種応用的フィルタ設計機能

MATLABによるフィルタ設計: オプション構成

- **MATLAB**
  - filter処理
  - 多項式フィッティング（トレンド抽出）
- **Signal Processing Toolbox**
  - ベーシックなフィルタ設計
    - FIR, IIR（バタワース、チェビシェフ…）
    - メディアンフィルタ
  - フィルタ設計GUI（FDATool, filterbuilder）
- **DSP System Toolbox**
  - 応用的なフィルタ設計
    - 適応フィルタ
    - マルチレートフィルタ
  - 各種可視化（スペクトラムアナライザ）
  - 音声信号処理
- **Simulink**
  - ブロック線図環境
- **Embedded Coder**
  - 組み込みコード生成

Agenda

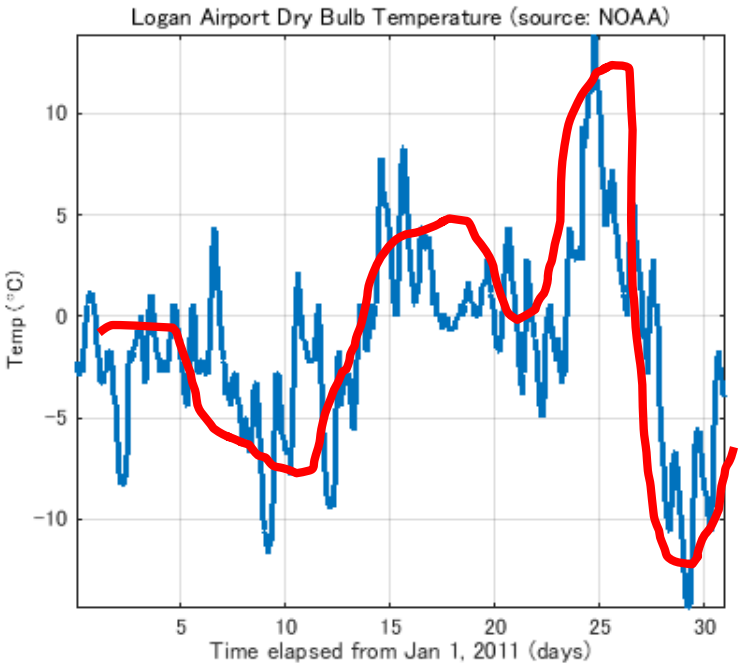
- Section1: フィルタとは？
- **Section2: Case study**
- Section3:フィルタの実装
- Section4:フィルタ設計FAQ
- まとめ

Agenda

- Section2: Case study
  - Case1: 気温データ
    - （トレンド抽出、アベレージング）
  - Case2: 心拍データ
    - （トレンド成分の除去、ピーク値検索、平滑化）
  - Case3: センサーデータ
    - （リサンプリング、メディアンフィルタ）
  - Case4: 変位データ・加速度データ
    - （微分・積分処理）
  - Case5: ノイズキャンセリング
    - （適応フィルタ）
  - Case6: ΔΣADコンバータ
    - （アンチエイリアシング、マルチレート処理）

Case1: 気温データ

2011年1月のボストンの気温データ



目的:

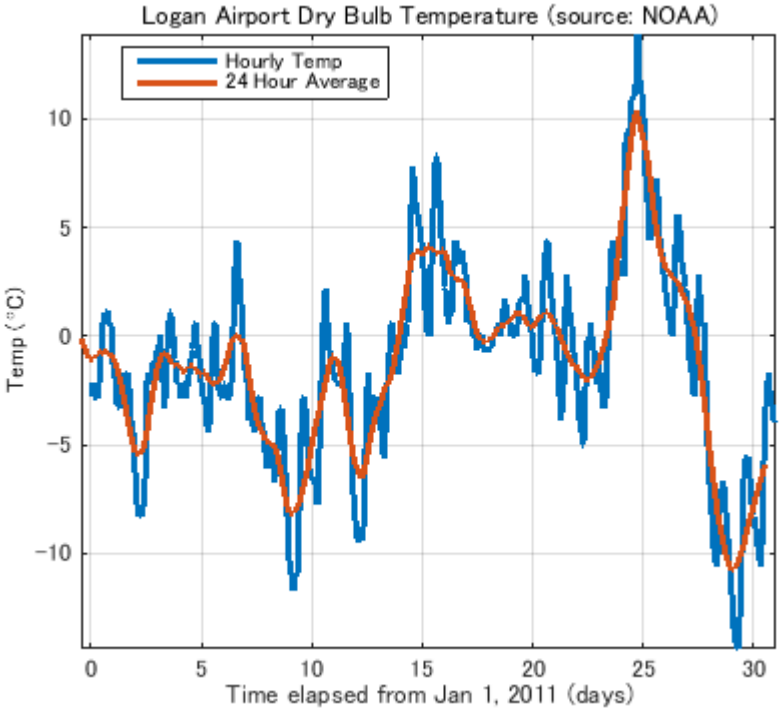
- 1ヶ月の気温変動傾向
- 24時間の気温変動傾向

課題:

- 時刻の影響を除去
- 24時間の変動平均

Case1: 気温データ(トレンド抽出)

トレンド成分の重ね描き




```
>>coeff = ones(1, 24)/24;
>>out   = filter(coeff, 1, in);
```

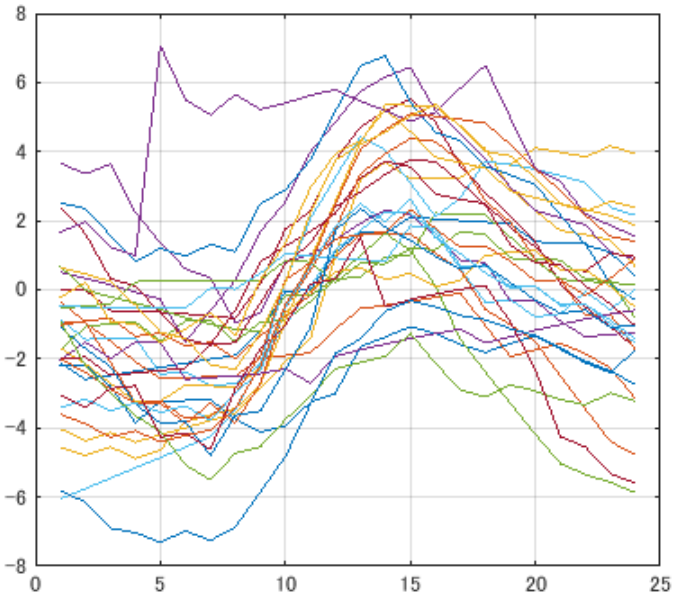
フィルタ係数

フィルタリング

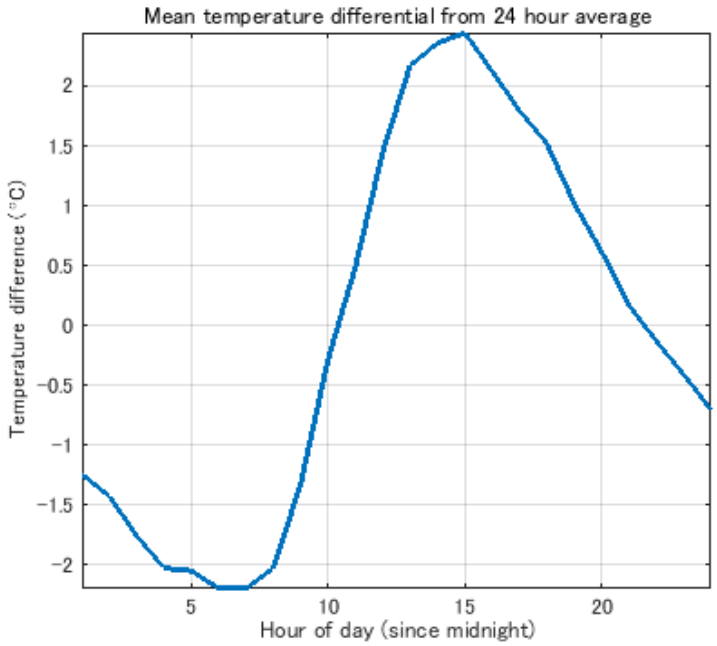
移動平均フィルタでスムージング

Case1: 気温データ(アベレージング)

24時間データ重ね描き(31日分)



24時間データ(31日分平均)




```
>>out = reshape(in,24,31).' ;
>>plot(1:24, mean(out))
```

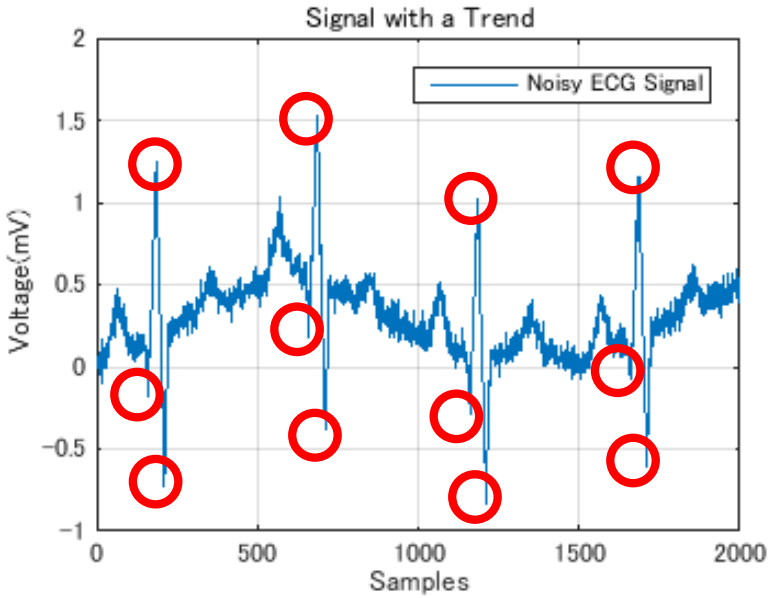
24行31列に並べ替え

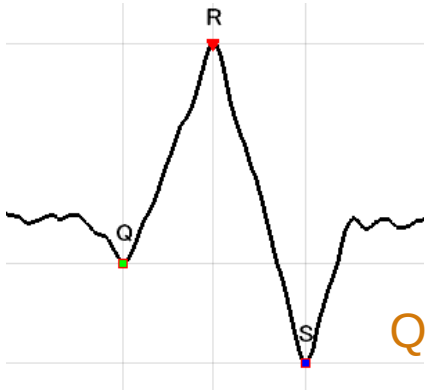
全行の平均値を求める

行列処理で記述は簡潔に

Case2: 心電図データ

心電図データ





QRS群

目的:

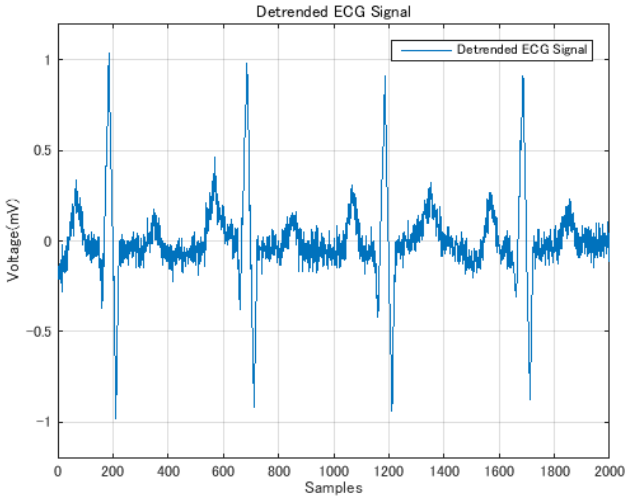
- Q波、R波、S波を得る
- 以下の情報を得る
  - 立ち上がり時間
  - 立ち下がり時間
  - 立ち上がり振幅
  - 立ち下がり振幅

課題:

- 信号劣化要因の除去
- ピーク情報の取得方法

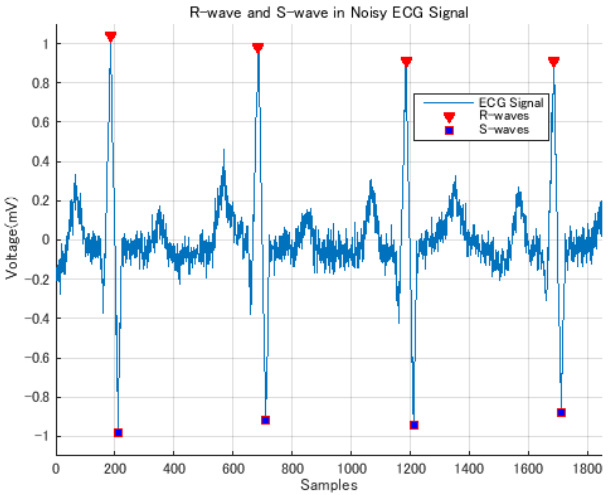
Case2: 心電図データ(トレンド除去、ピーク値探索)

トレンド除去後の心電図データ




```
>>[p,s,mu] = polyfit((t,in,6);
>>f_y      = polyval(p,t,[],mu);
>>out      = in- f_y;
```

R波とS波
(Q波は雑音との識別困難)

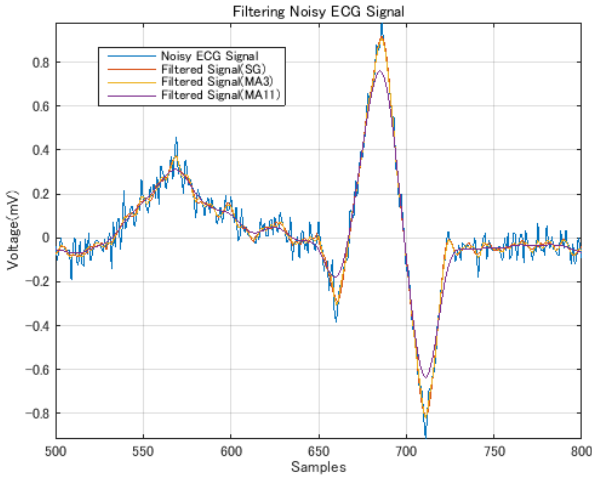



```
>>[~, R] = findpeaks(in,...
      'MinPeakHeight',0.5);
```

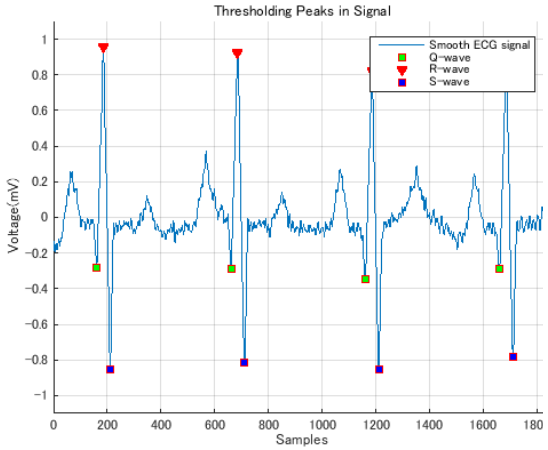
多項式近似でトレンド除去
findpeaks関数でピーク値探索

Case2: 心電図データ(平滑化とピーク値再探索)

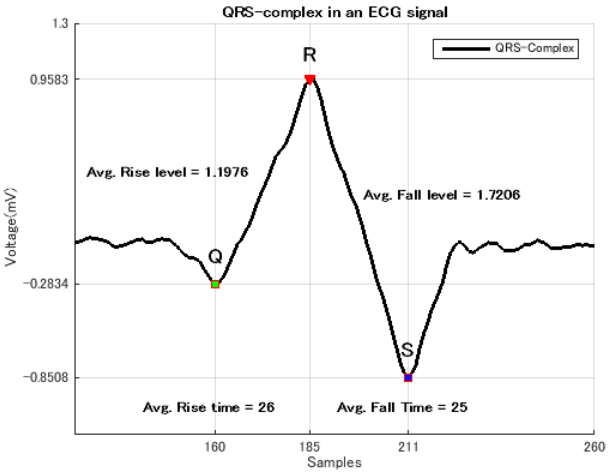
Savitzky-Golay法および
MA法による平滑化

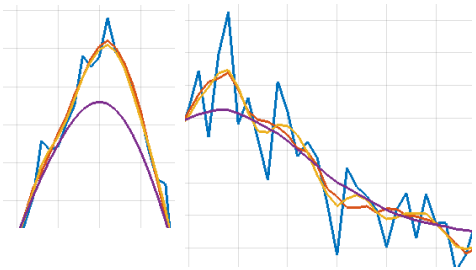


QRS群



QRS群各種統計値





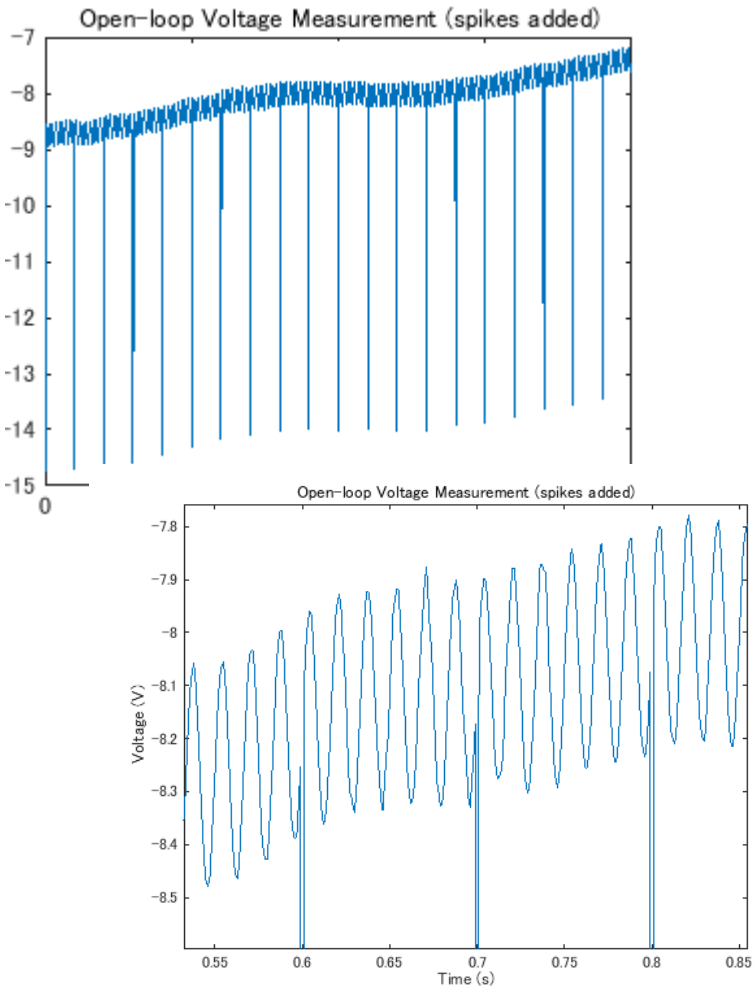
```
>>out = sgolayfilt(in,7,21);
```

- MA次数大:振幅減衰
- MA次数小:雑音残る

目的に適した手法選択

Case3: センサーデータ

アナログセンサー電圧データ
（ACノイズ、スパイクノイズ）



目的:

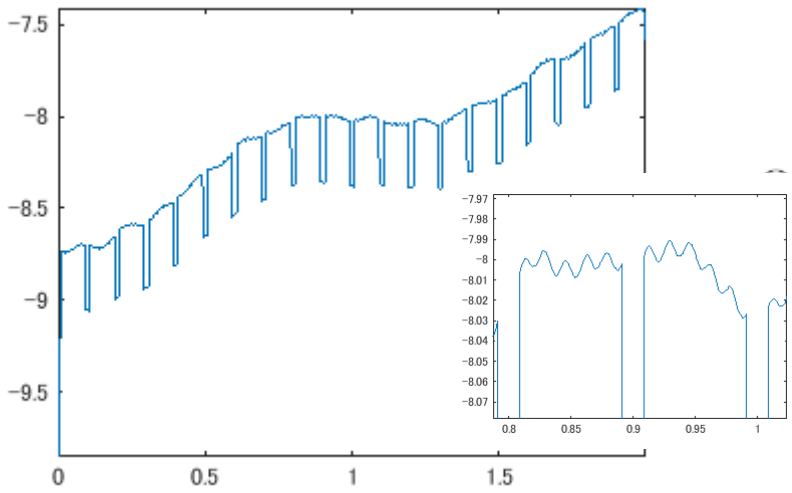
- 取得したセンサー信号から不要成分を除去

課題:

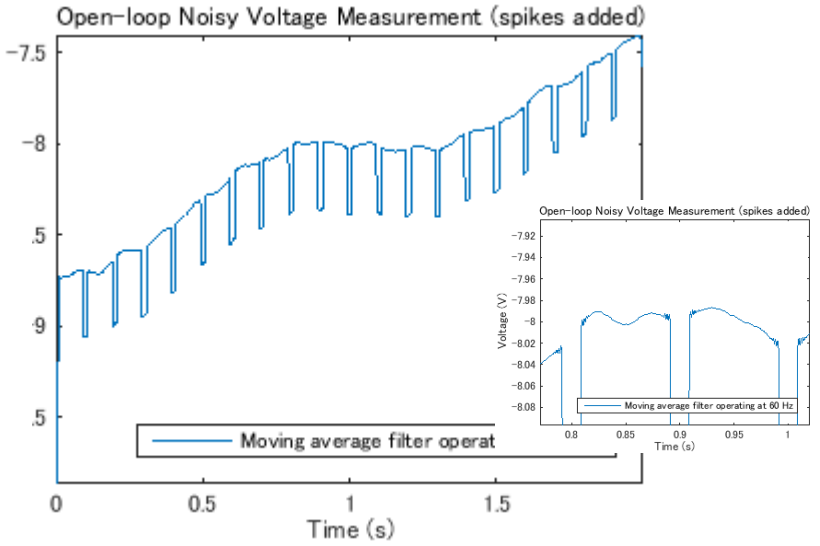
- ACラインからのノイズ除去
- スパイクノイズ除去

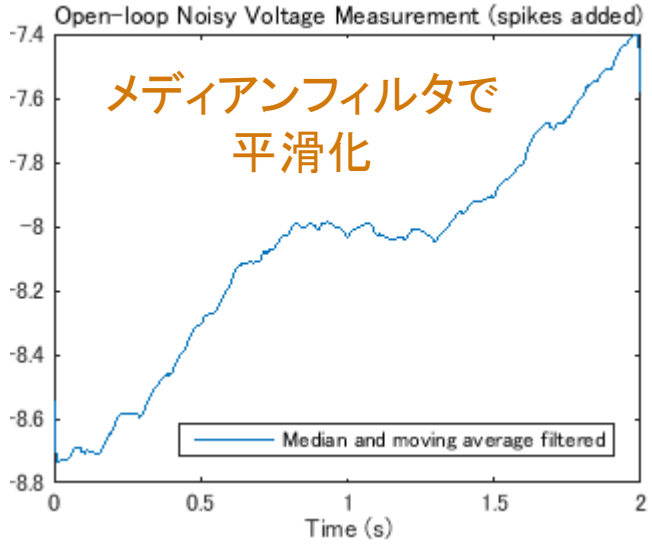
Case3: センサーデータ(メディアンフィルタ)

Savitzky-Golay法で平滑化



リサンプリング後に
Savitzky-Golay法で平滑化



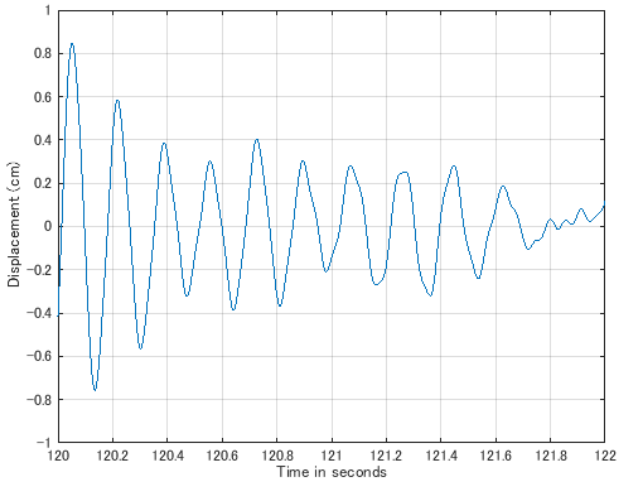



```
>>out1 = resample(in1,fs1,fs2);
>>out2 = medfilt1(in2,3);
```

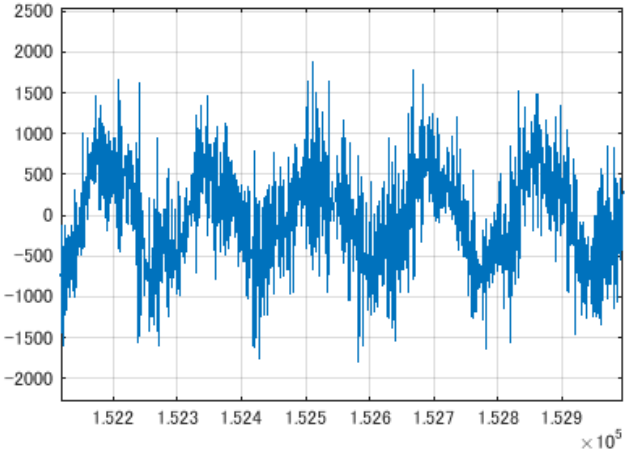
スパイクノイズには
メディアンフィルタが効果的

Case4: 変位データ、加速度データ

地震の条件下における
3 階建て構造物の 1 階変位データ



加速度データ



目的:

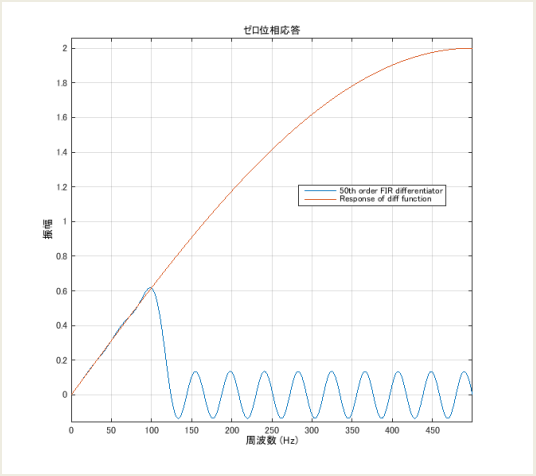
- 変位センサーから得られた床の位置データから、速度と加速度情報を取得
- 加速度データから、速度と変位情報を取得

課題:

- フィルタによる微分処理
- フィルタによる積分処理

Case4: 変位データ、加速度データ（微分・積分フィルタ）

振幅応答比較
（diff関数vs微分フィルタ）



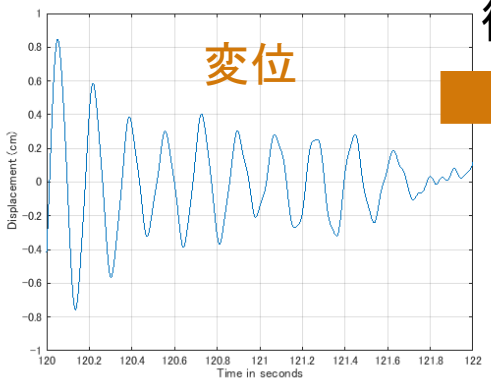
diff関数の伝達関数

$H(z) = 1 - z^{-1}$

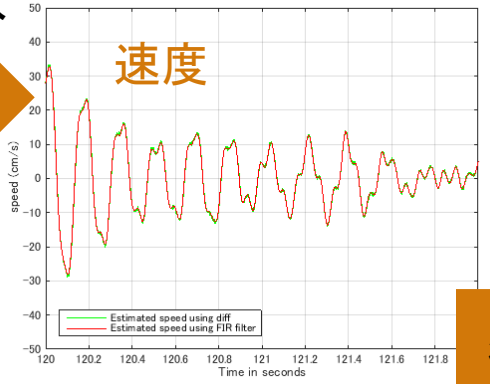
⇒高周波成分を強調

⇒微分フィルタが効果的

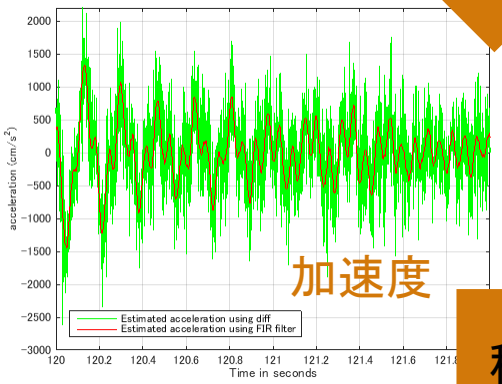
**フィルタにより
微分・積分が可能**



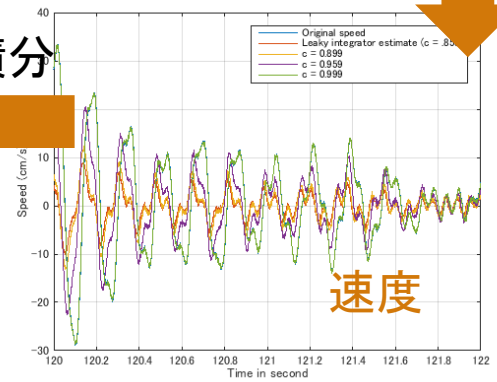
微分



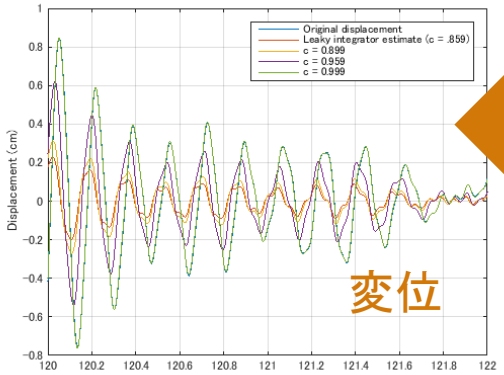
微分



積分



積分



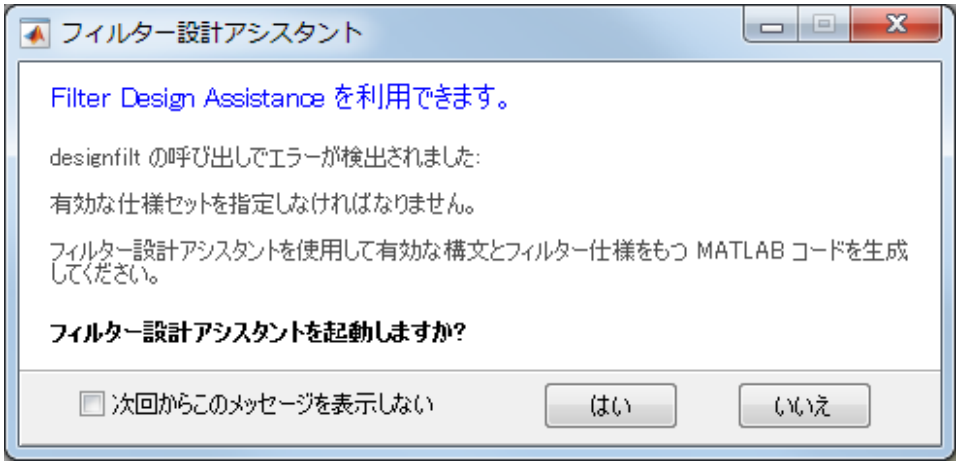
Case4: 変位データ、加速度データ（cont'd）

R2014a

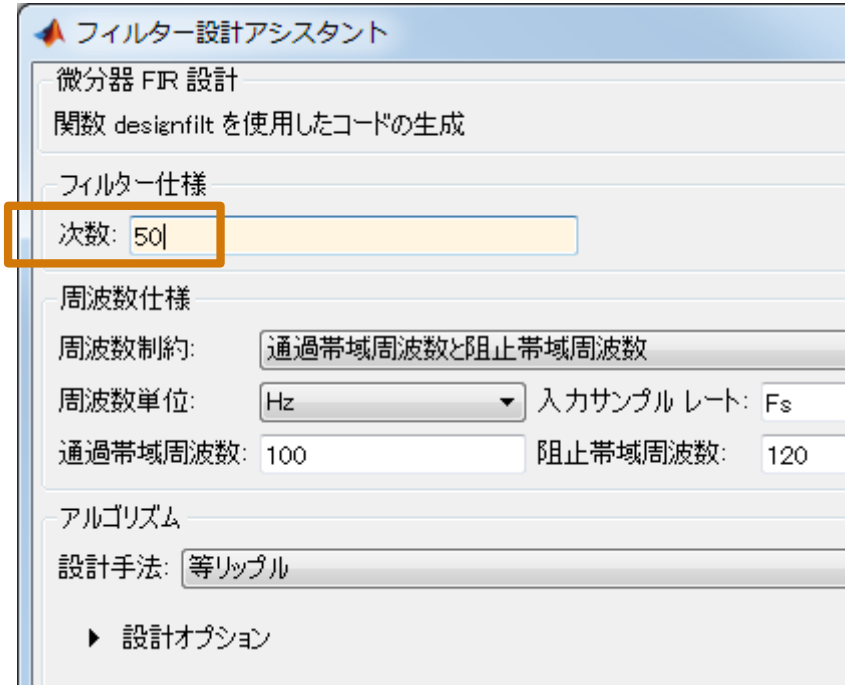
①フィルタオブジェクト定義

```
>> df = designfilt('differentiatorfir',...
                   'PassbandFrequency',100,…
                   'StopbandFrequency',120,...
                   'SampleRate',Fs);
```

②設計に必要な要求仕様が不足



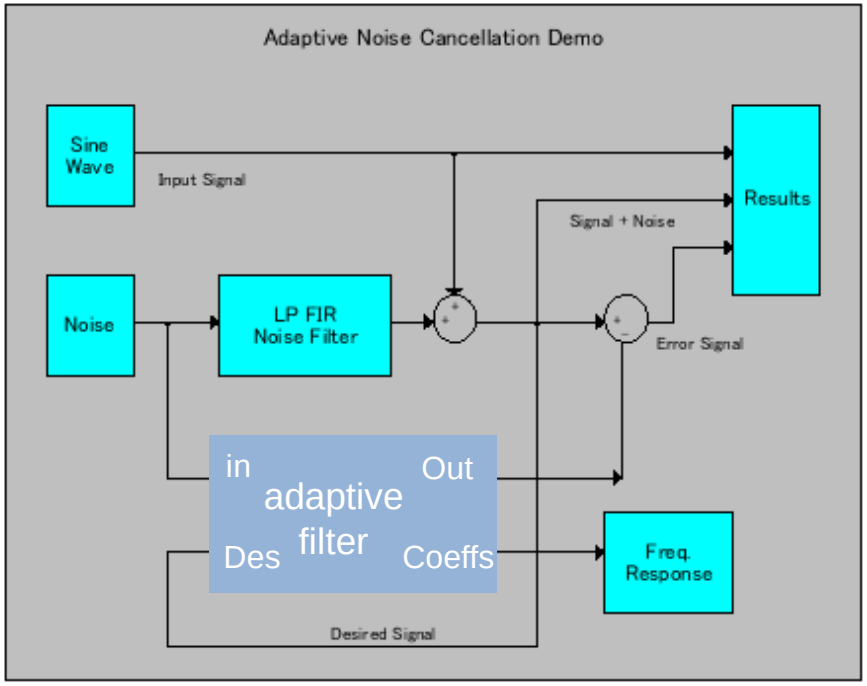
③不足情報の補完を促すGUI



④必要な仕様条件を補完し、自動実行

```
df = designfilt('differentiatorfir',…
                'FilterOrder', 50,…
                'PassbandFrequency', 100,…
                'StopbandFrequency', 120,…
                'SampleRate', Fs);
```

Case5: ノイズキャンセリング



目的:

- 雑音が通過した経路を推定
- 推定した経路情報を用いて、雑音の重畳した信号から雑音のみ除去

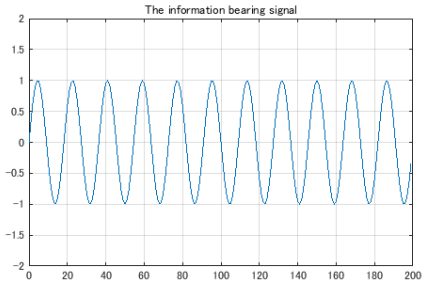
課題:

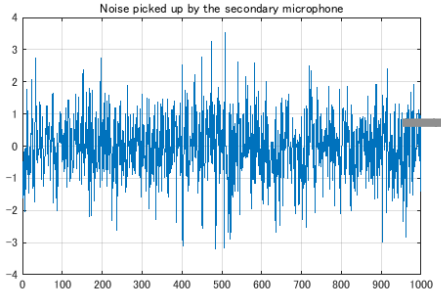
- MATLABによる適応アルゴリズムの実現
- Simulinkによる適応アルゴリズムの実現

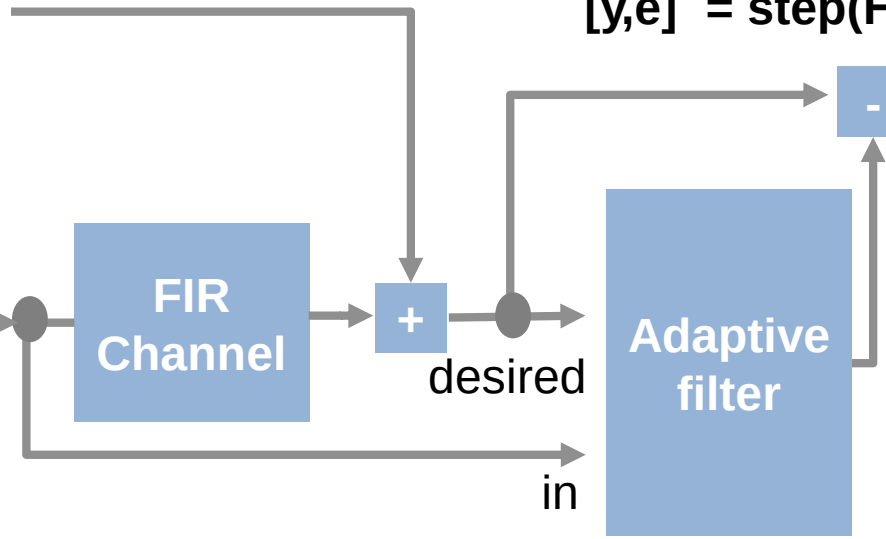
Case5: ノイズキャンセリング（MATLABによるRLS）

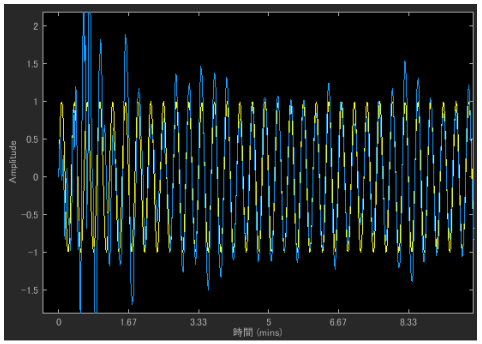
- n: noise
- s: signal

```
Hd = dsp.FIRFilter;
Hadapt = dsp.RLSFilter(M);
d = step(Hd,n) + s;
[y,e]  = step(Hadapt,n,d);
```









Dsp.RLSFilter により実現

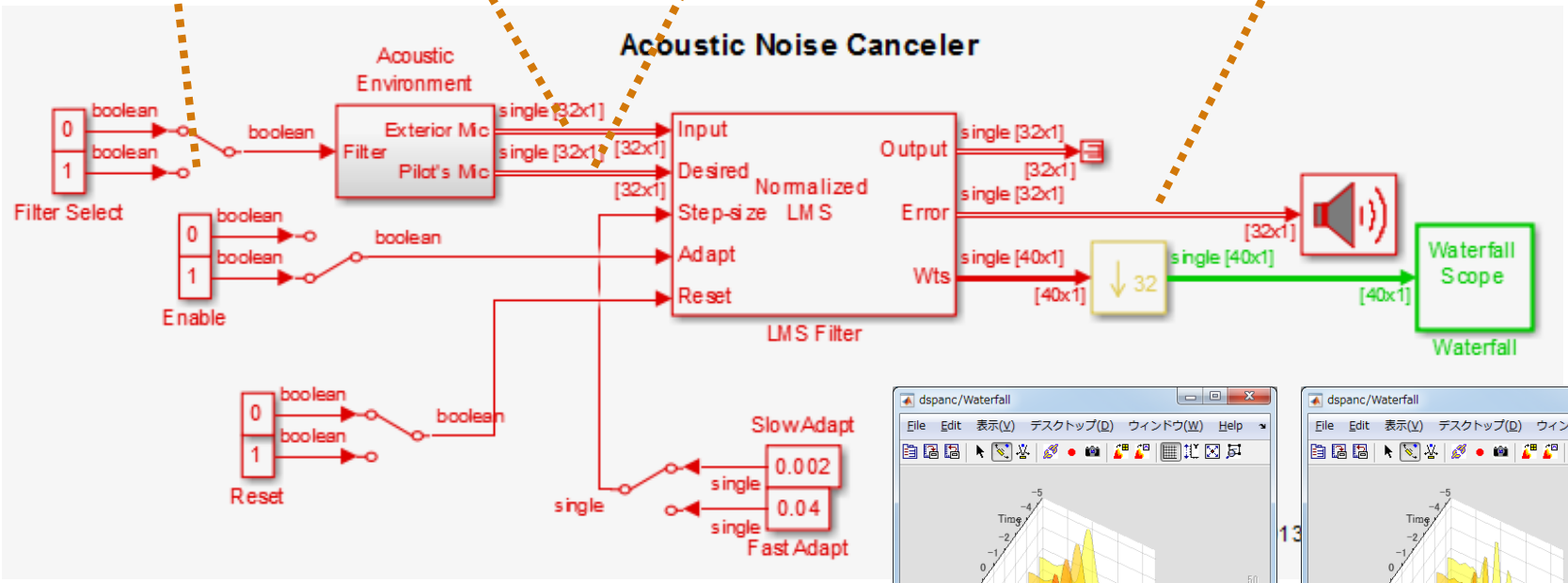
Case5: ノイズキャンセリング（SimulinkによるLMS）

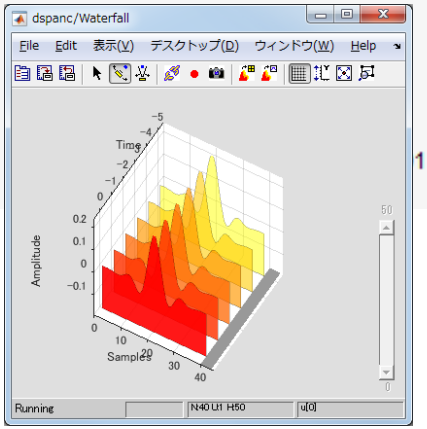
雑音環境
切り替え

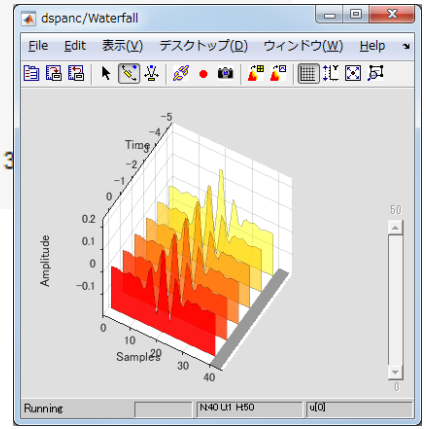
雑音取得用
外部マイク

pilotマイク

誤差信号
（取得信号）

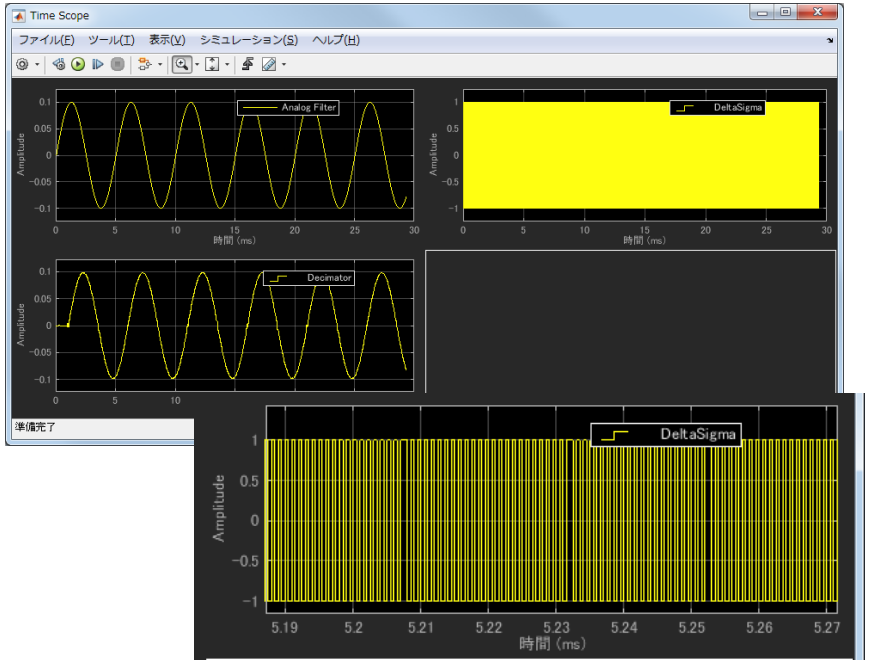






**LMSブロック
により実現**

Case6: ΔΣ型ADコンバータ

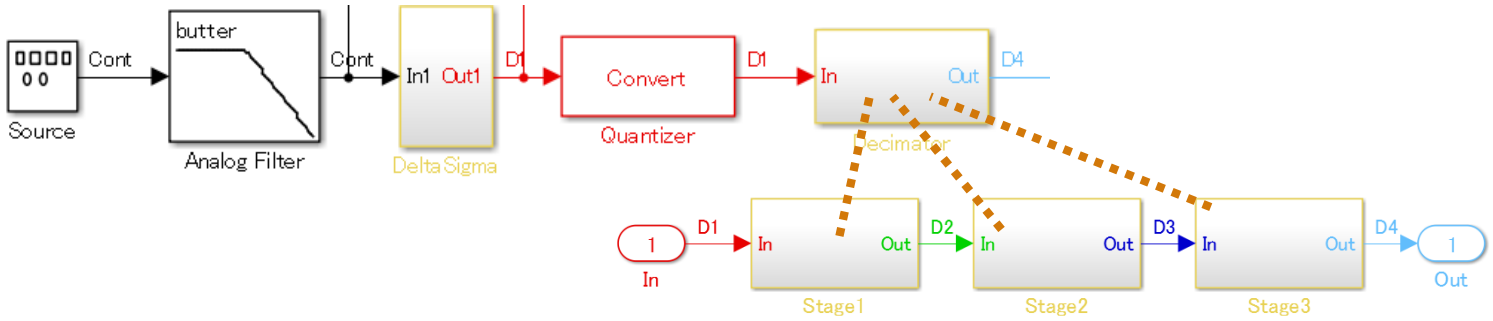


目的:

- アナログ部アーキテクチャ検討
- デジタル部レート変換、固定小数点解析

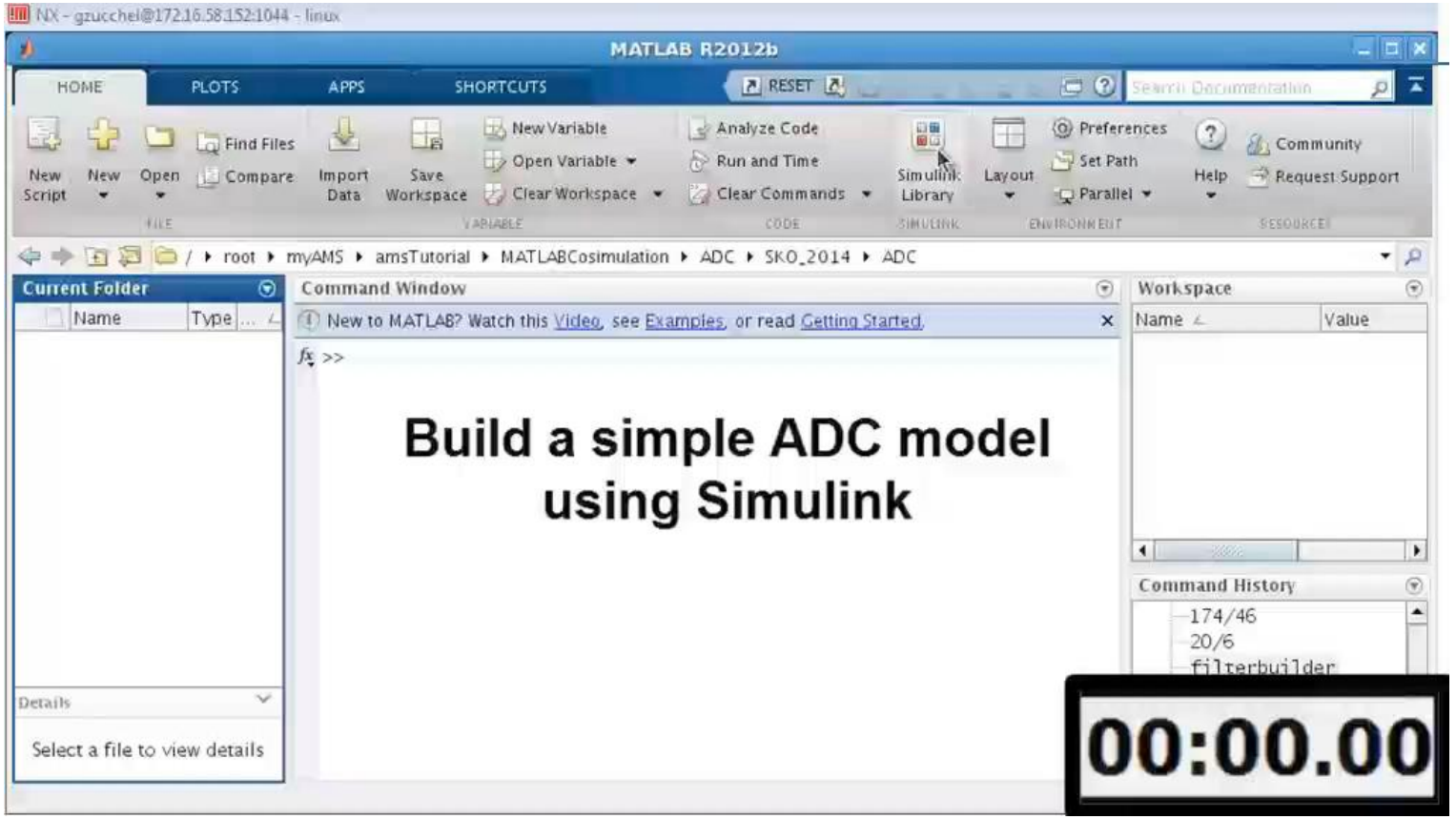
課題:

- アナログミックスドシグナルのハンドリング
- フィルタ設計ワークフロー

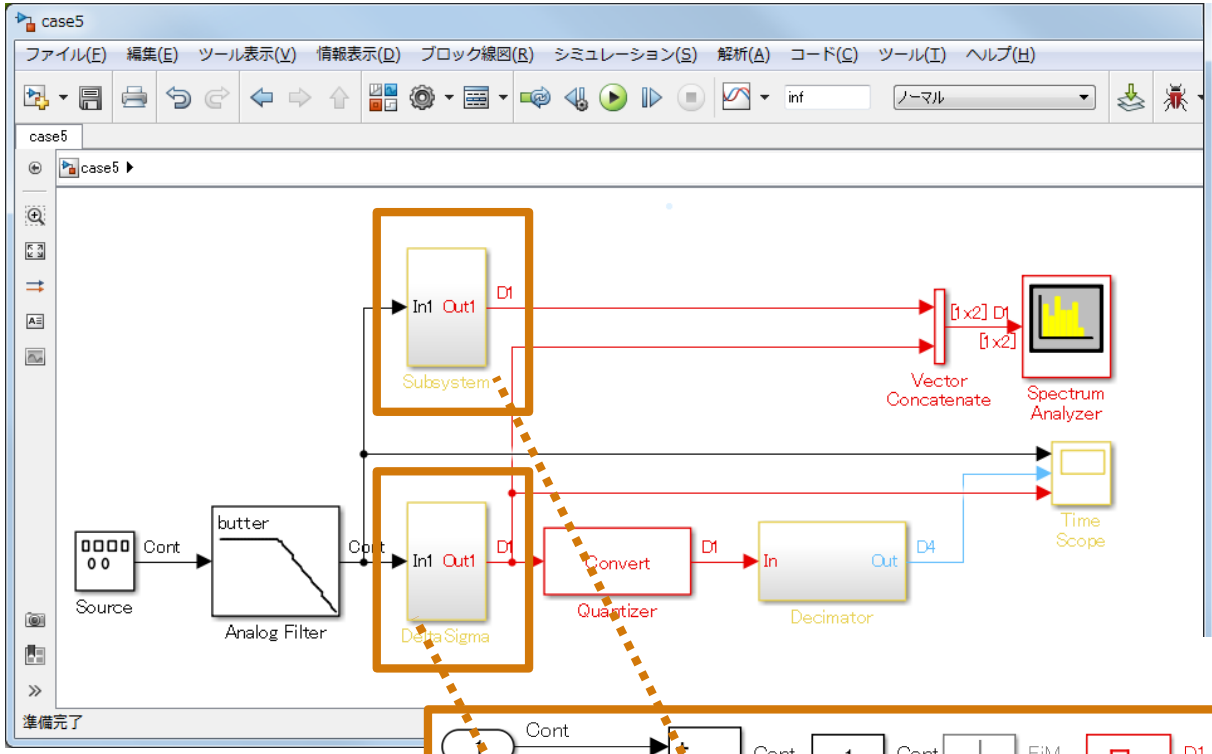


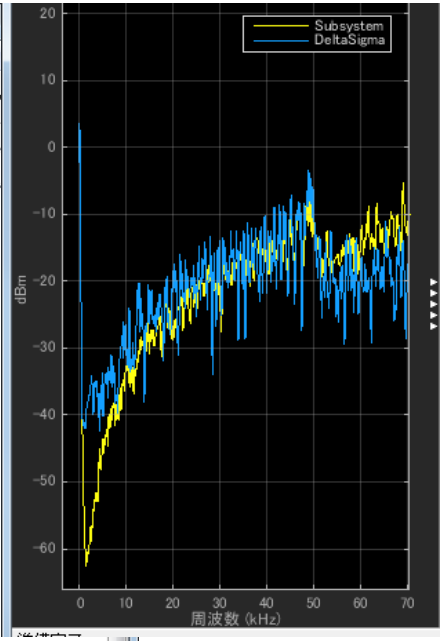
Demo: Simulinkによる信号処理システムモデリング例

- ΔΣADコンバータ
  - アナログフィルタ（アンチエイリアシング）
  - デジタルフィルタ（デシメーション）

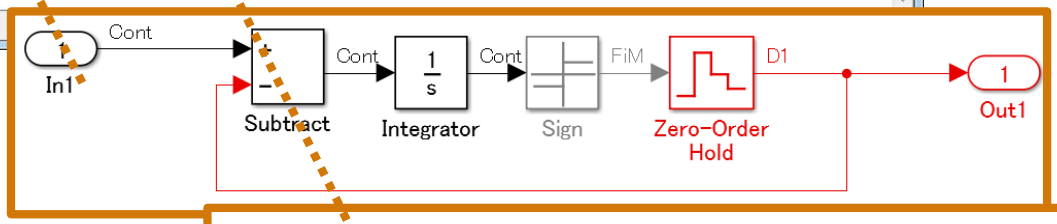


Case6: ΔΣ型ADコンバータ（アナログ処理部）

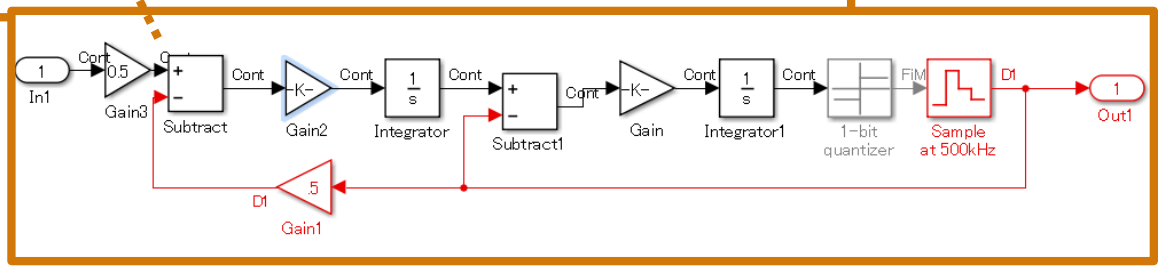




1st order

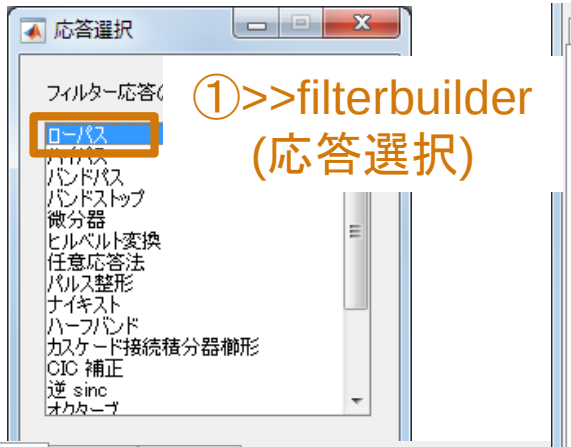


2nd order

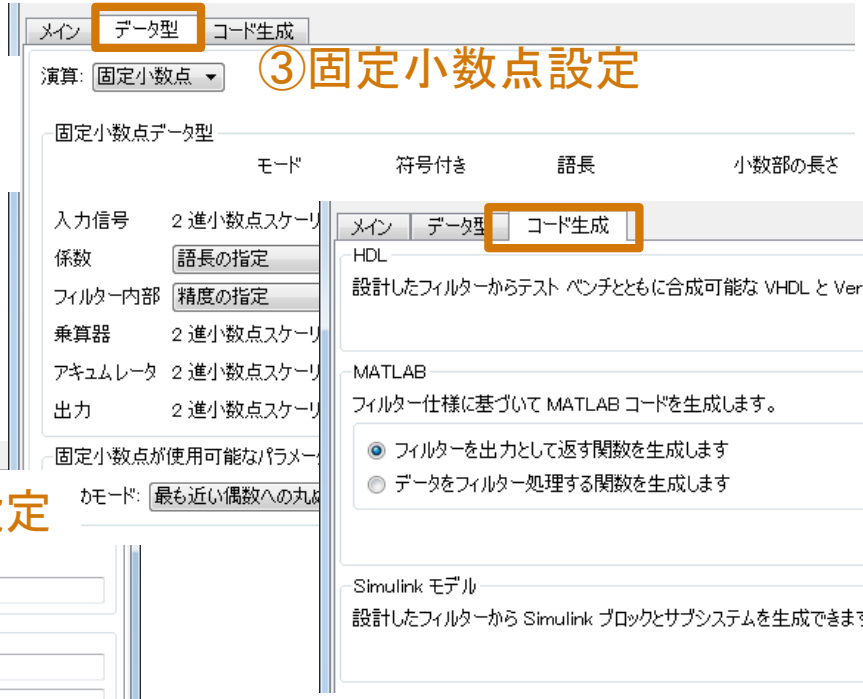


アーキテクチャ検討には Simulinkが最適

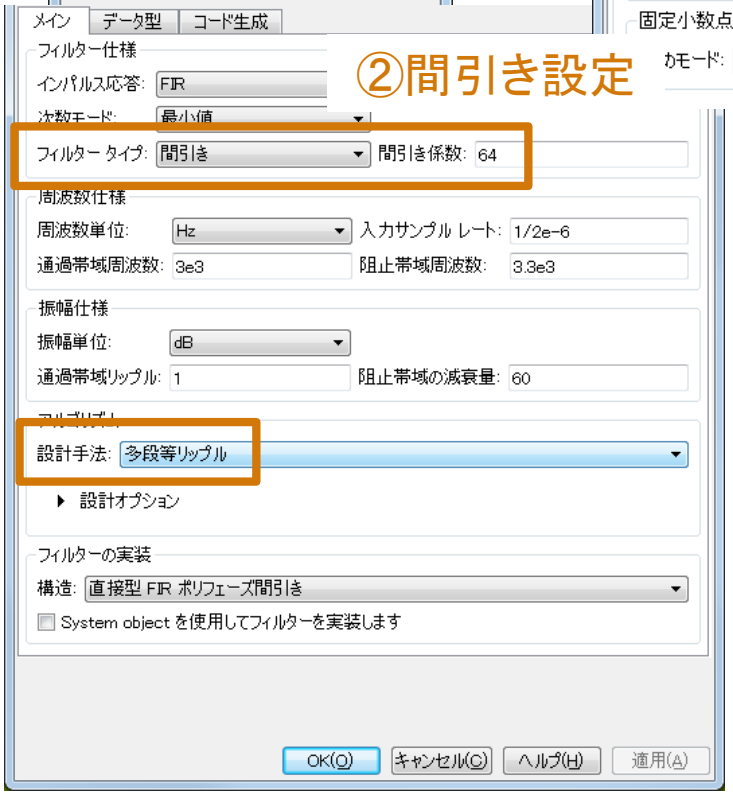
Case6: ΔΣ型ADコンバータ（デジタル処理部）



①>>filterbuilder
(応答選択)

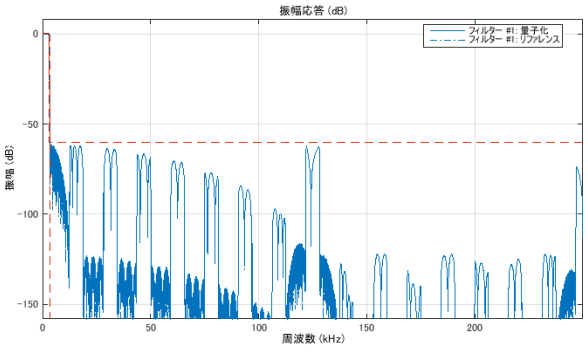


③固定小数点設定



②間引き設定

デシメーションフィルタの
周波数特性



④Simulinkブロック
自動生成

**filterbuilderによる
フィルタ設計フロー**

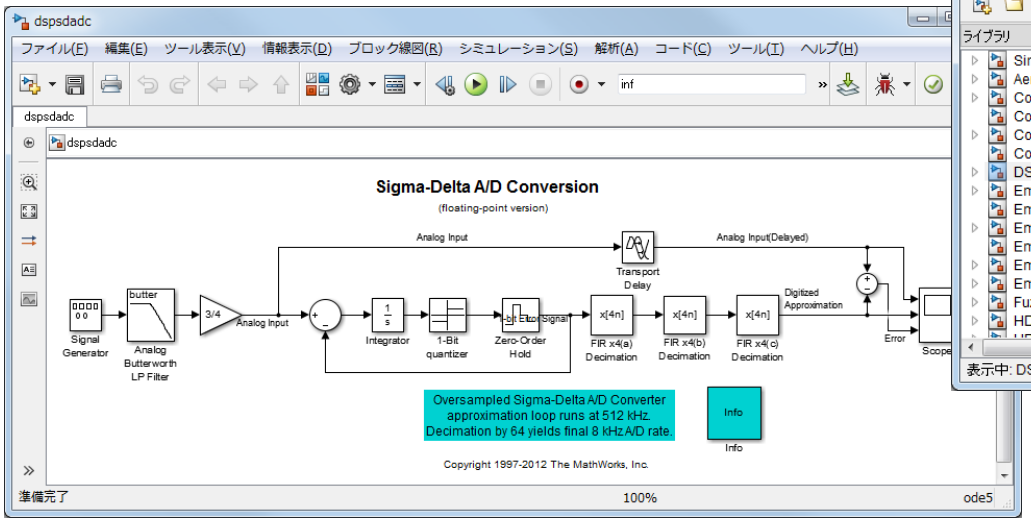
Agenda

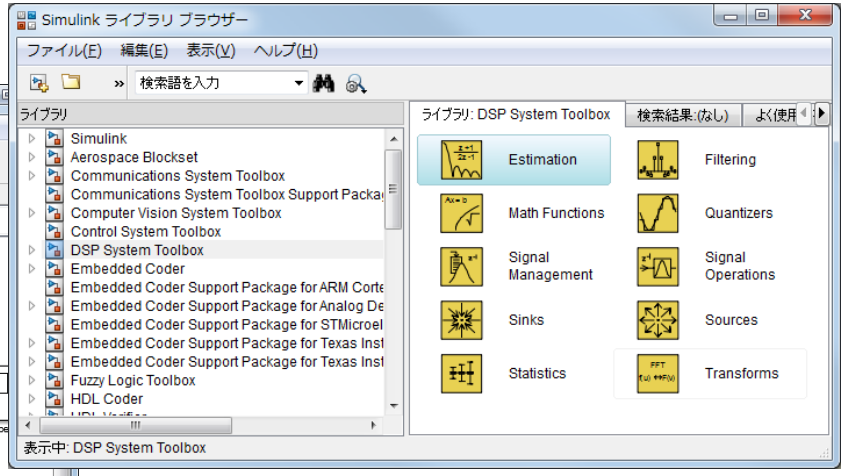
- Section1: フィルタとは？
- Section2: Case study
- **Section3: フィルタの実装**
- Section4: フィルタ設計FAQ
- まとめ

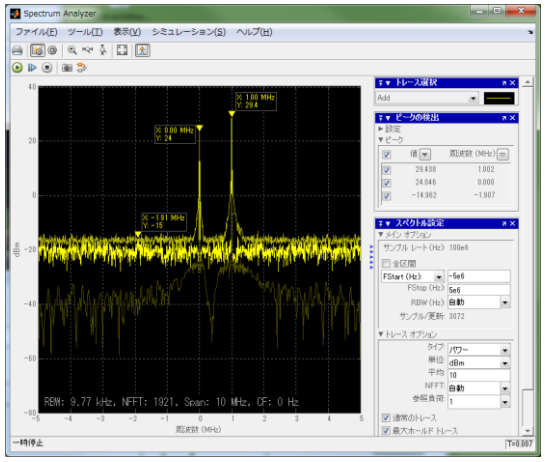
Agenda

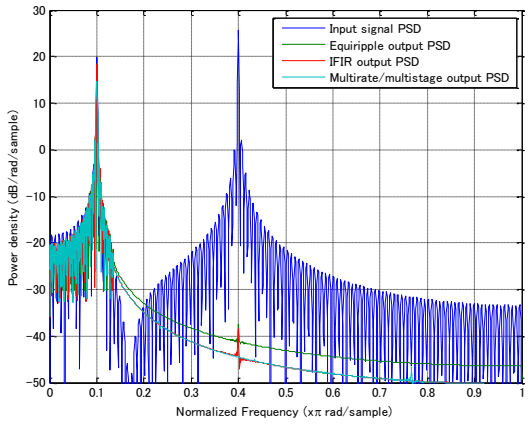
- Section3: フィルタの実装
  - 実装用コード生成環境
  - CMSIS/Ne10ライブラリ対応状況
  - デモ
  - STMicro Discoveryボード

信号処理システム設計ライブラリ
DSP System Toolbox™









- 高度なフィルタ設計
  - マルチレート, 適応フィルタ, 固定小数点 (*Fixed-Point Designer*™)
- スペクトル解析・スペアナ表示
- 行列・統計処理
- FFT/DCT/DWT
- ARM® Cortex®-M CMSIS, Cortex-A Ne10対応 (Embedded Coder™)

CMSIS/Ne10ライブラリサポート状況

R2013b R2014b

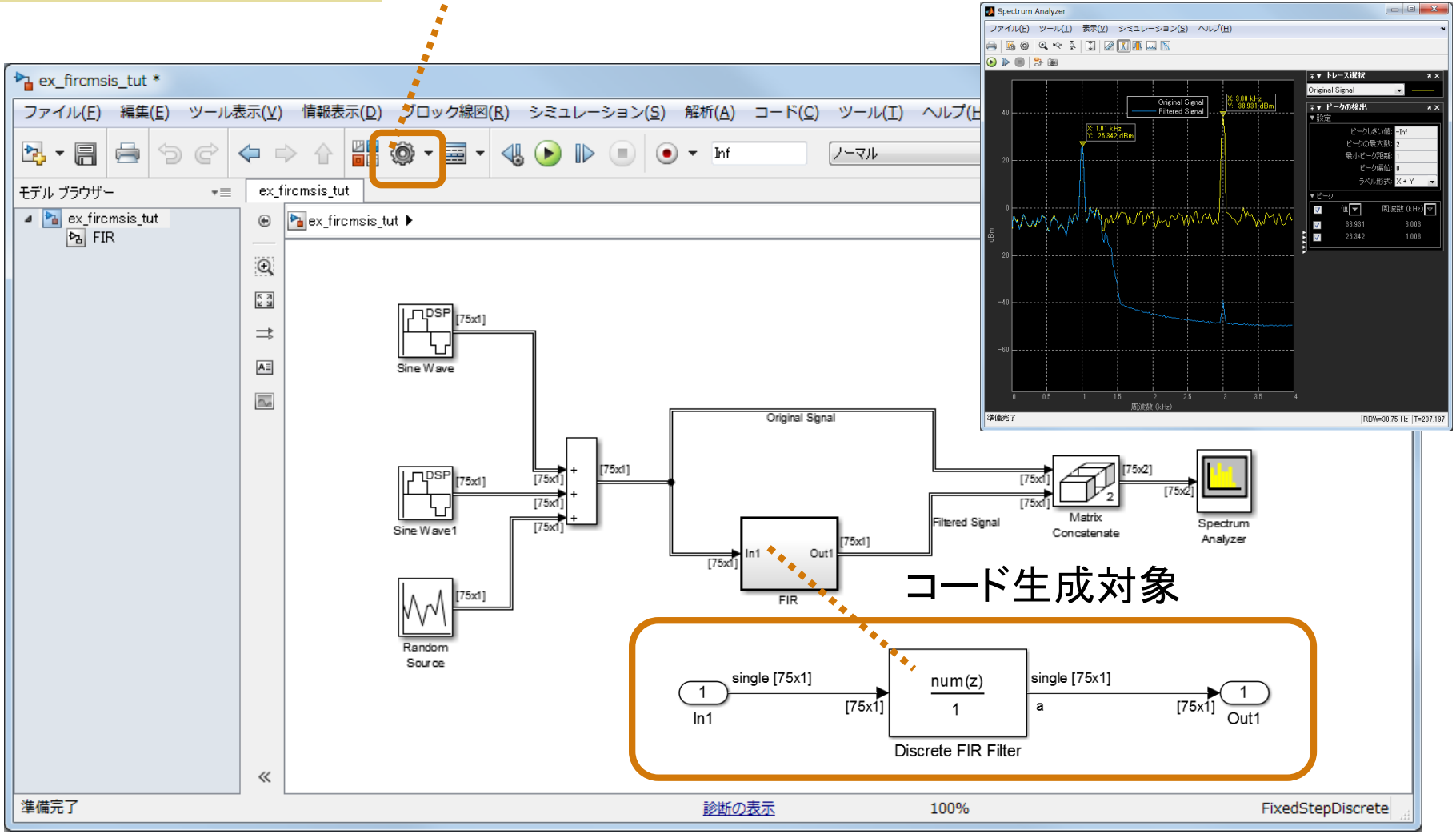
機能	概要	CMSIS	Ne10
Discrete FIR Filter	Model FIR filters	○	○
FIR Decimation	Filter and downsample input signals	○	○
FIR Interpolation	Upsample and filter input signals	○	○
LMS Filter	Compute output, error, and weights using LMS adaptive algorithm	○	
Biquad Filter	Model biquadratic IIR (SOS) filters	○	
FFT	Fast Fourier transform (FFT) of input	○	○
IFFT	Inverse fast Fourier transform (IFFT) of input	○	○
Correlation	Cross-correlation of two inputs	○	
Convolution	Convolution of two inputs	○	
Mean	Find mean value of input or sequence of inputs	○	
RMS	Compute root-mean-square value of input or sequence of inputs	○	
Variance	Compute variance of input or sequence of inputs	○	
Standard Deviation	Find standard deviation of input or sequence of inputs	○	

ブロック・システムオブジェクトに対応

Case1:浮動小数点FIRフィルタ

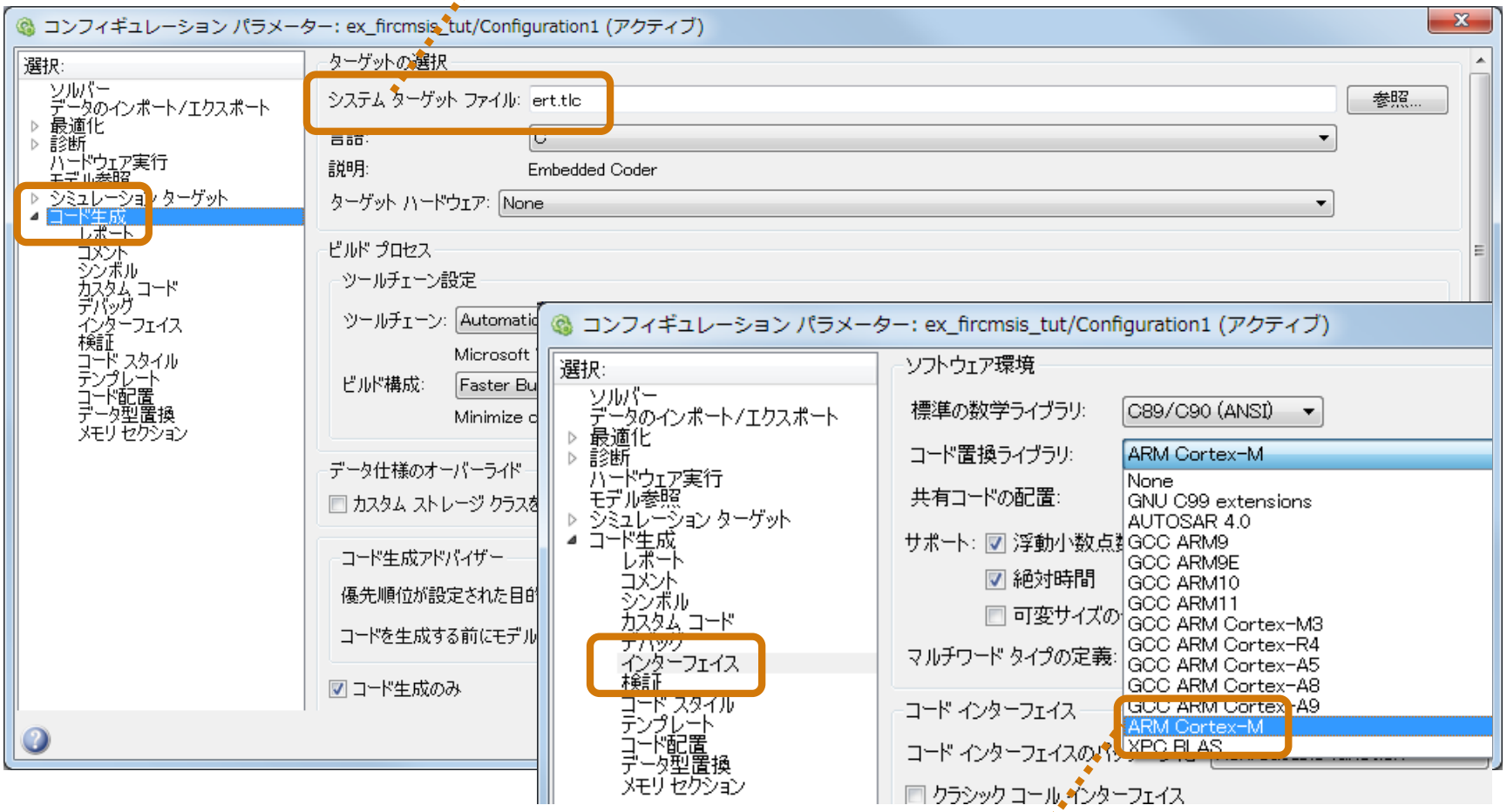
>>ex_fircmsis_tut

コンフィギュレーションパラメータ



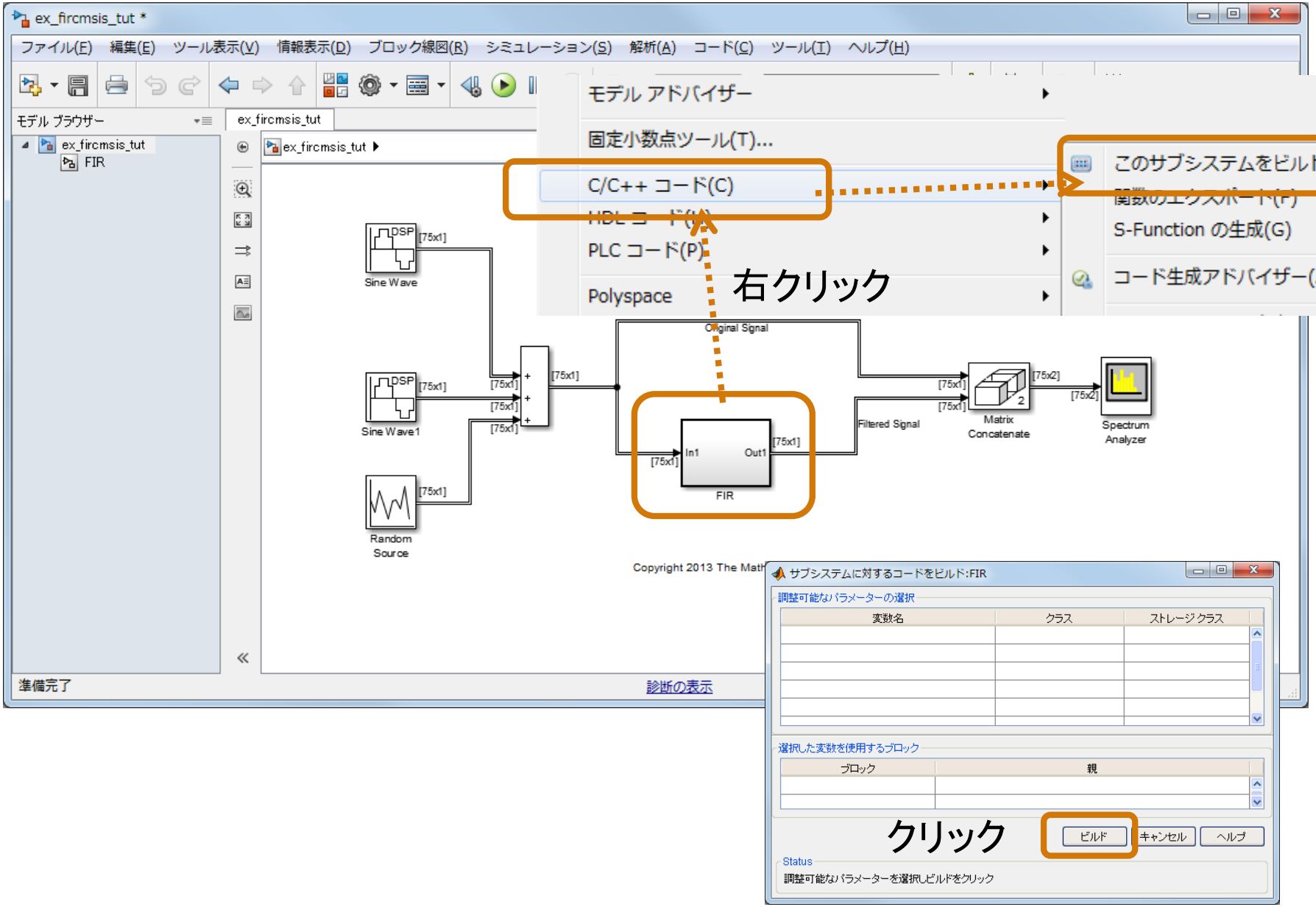
Case1:浮動小数点FIRフィルタ(Cortex-M)

システムターゲットファイルに、“ert.tlc”を選択

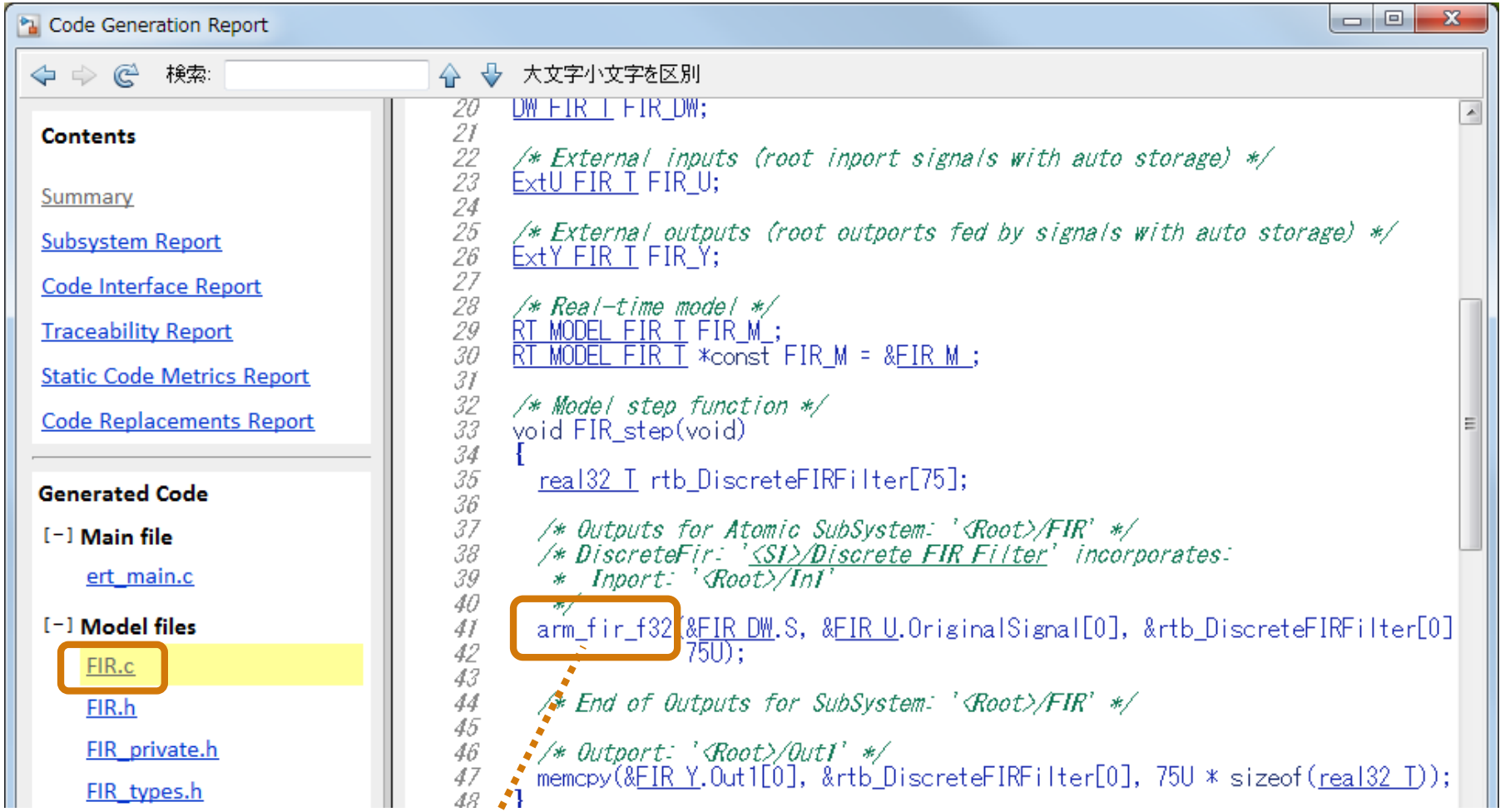


インターフェイス>コード置換ライブラリに、”ARM Cortex-M”を選択

Case1:浮動小数点FIRフィルタ（Cortex-M）



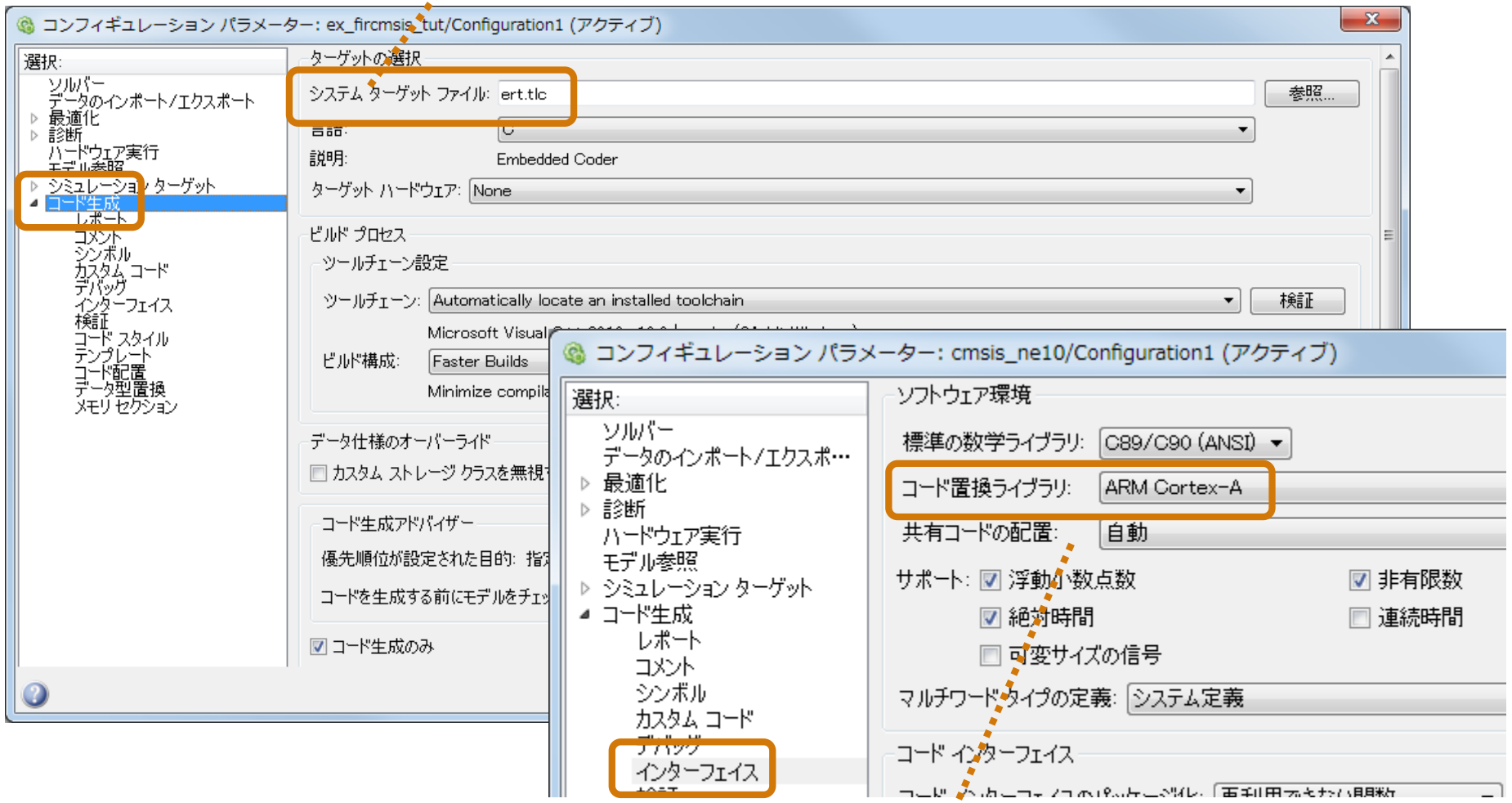
Case1:浮動小数点FIRフィルタ(Cortex-M)



FIRフィルタ部について、
CMSIS関数"arm_fir_f32"を生成

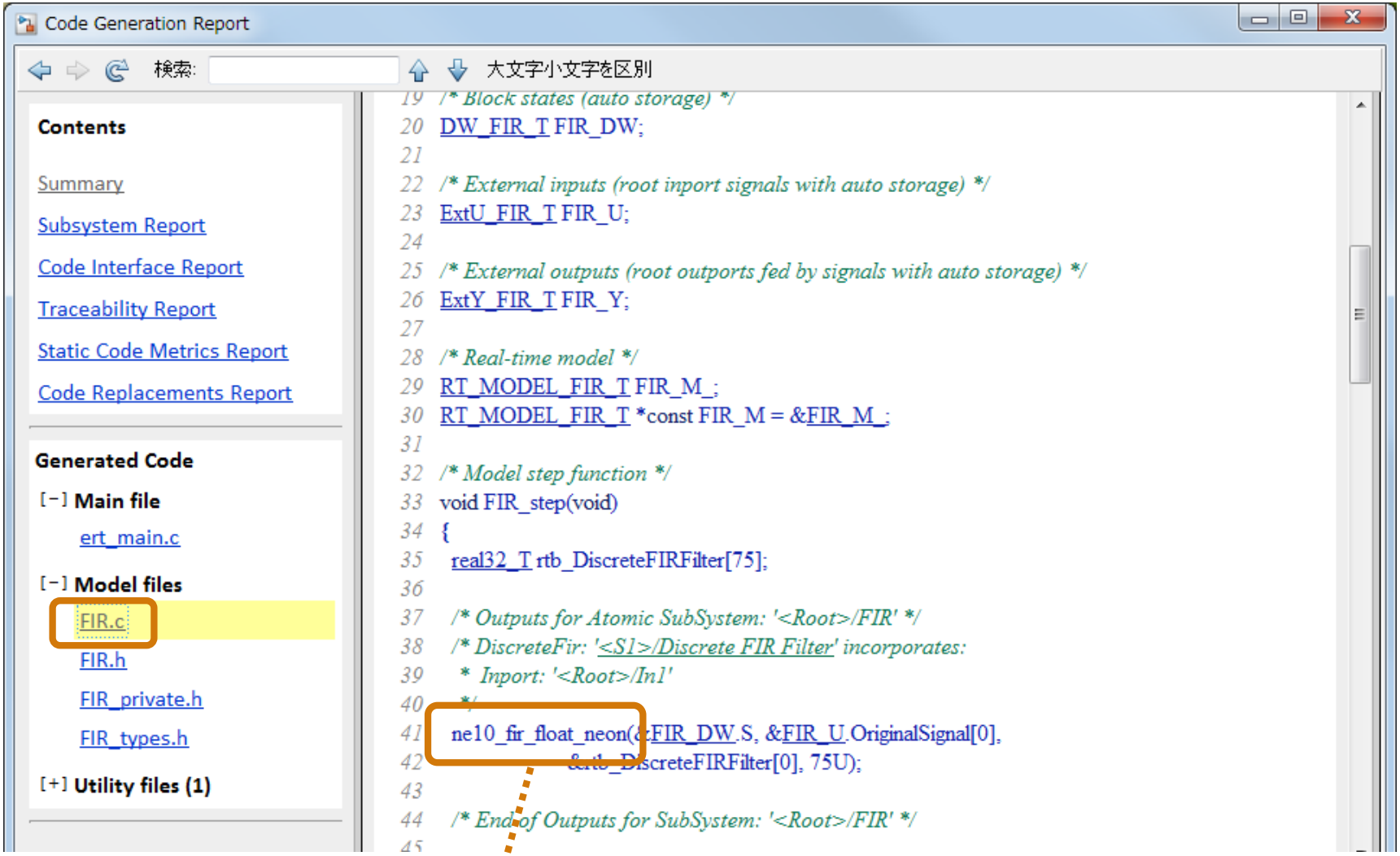
Case1:浮動小数点FIRフィルタ（Cortex-A）

システムターゲットファイルに、“ert.tlc”を選択



インターフェイス>コード置換ライブラリに、”ARM Cortex-A”を選択

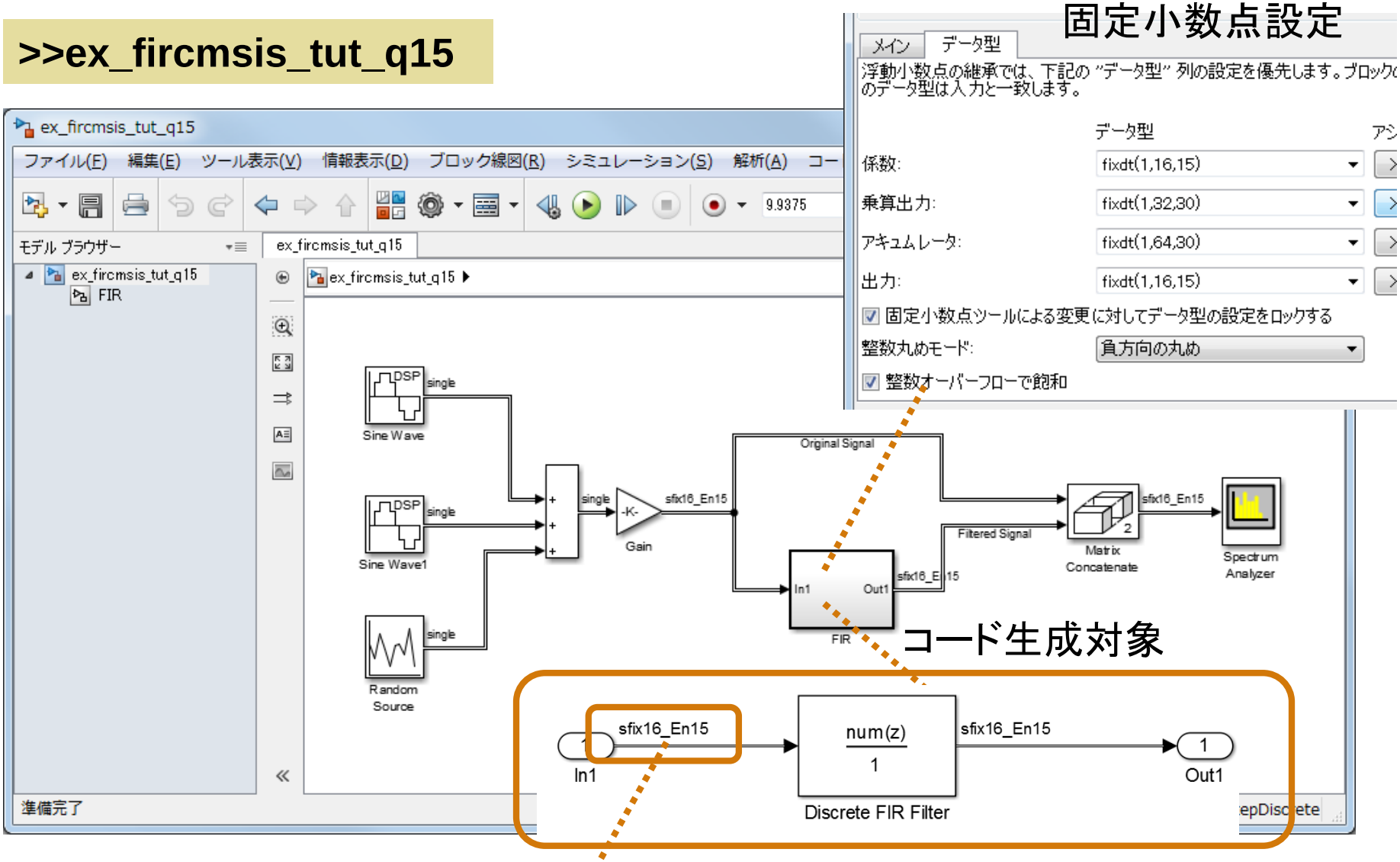
Case1:浮動小数点FIRフィルタ（Cortex-A）



FIRフィルタ部について、
”ne10_fir_float_neon”を生成

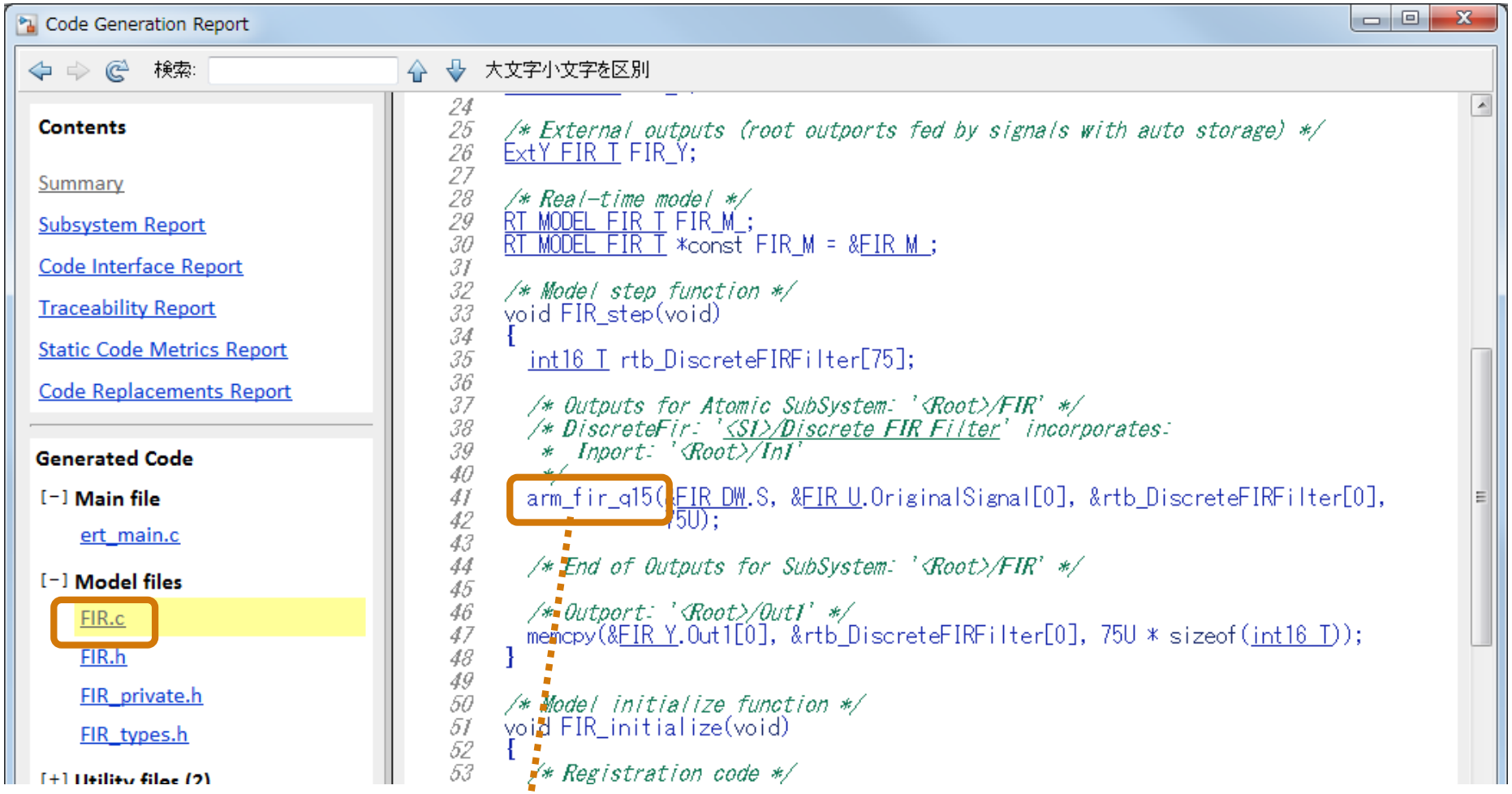
Case2:固定小数点FIRフィルタ

>>ex_fircmsis_tut_q15



sfix16_En15: 符号付、語長16bit、小数部15bit

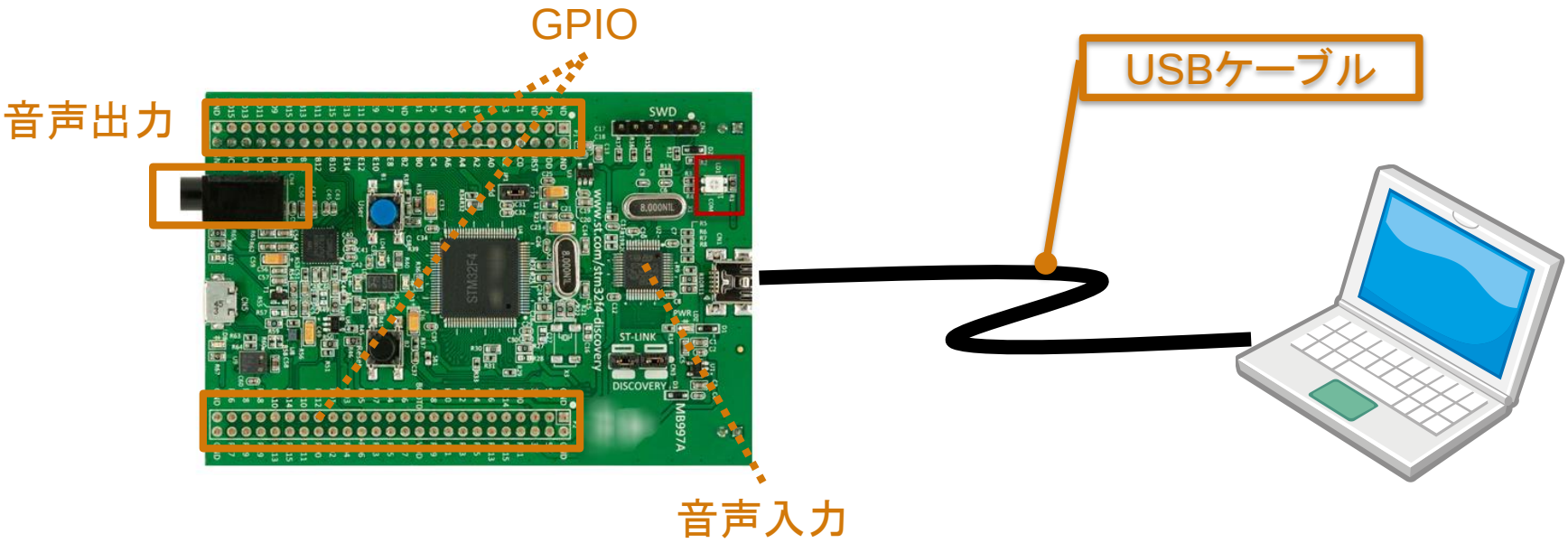
Case2:固定小数点FIRフィルタ



FIRフィルタ部について、
CMSIS関数”arm_fir_q15”を生成

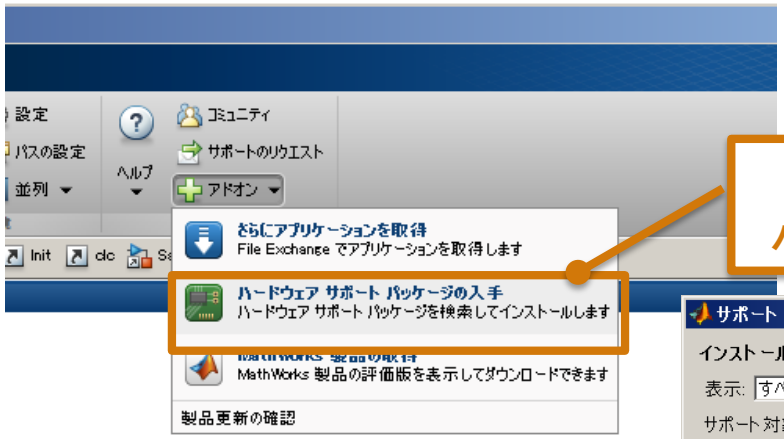
STM32F4-Discovery®

- Simulinkで作成したモデルを直接プロセッサーにダウンロードして実行
- ダウンロードにはST-LINKを使用
- 接続はUSBのみ*（電源供給、ダウンロード、PIL）

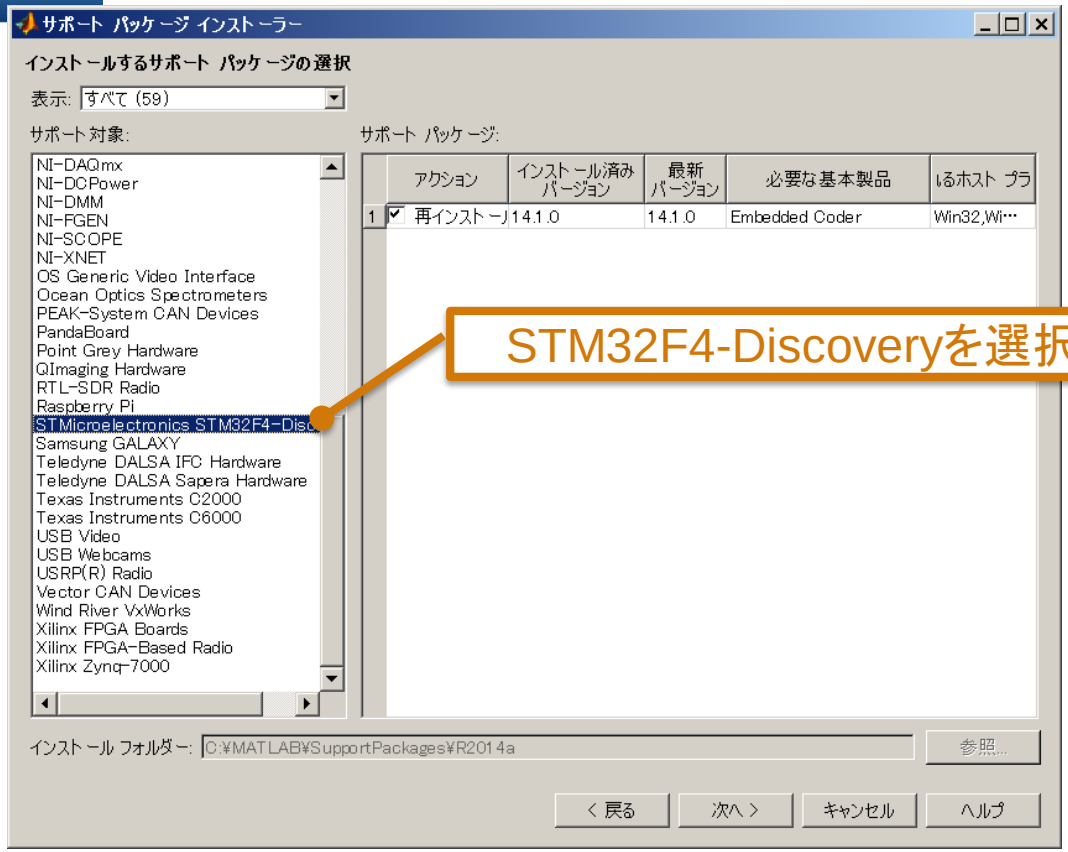


*高速PILシミュレーションおよびエクスターナルモードでの実行には、シリアル通信用アダプターを使用

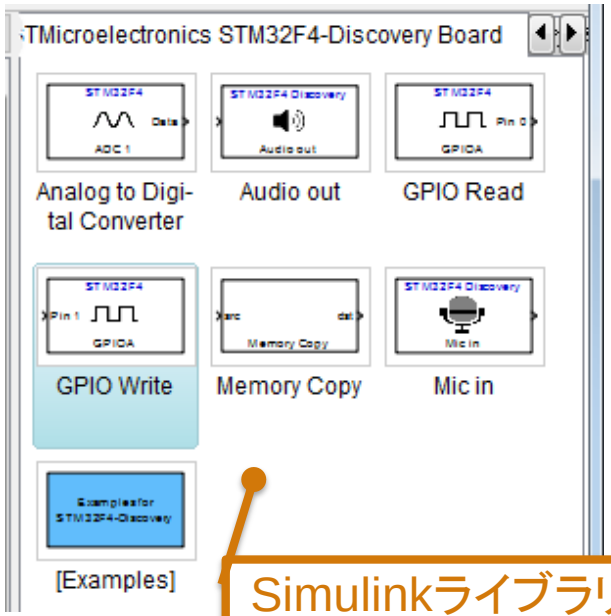
ライブラリインストール



アドオン⇒
ハードウェアサポートパッケージの入手



STM32F4-Discoveryを選択



Simulinkライブラリ

チュートリアル、例題

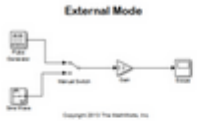
インストールされたサポート パッケージ

DSP System Toolbox Support Package for ARM Cortex-M Processors
MATLAB Support Package for USB Webcams
Simulink Support Package for Arduino Hardware
Communications System Toolbox Support Package for RTL-SDR Radio
Embedded Coder Support Package for ARM Cortex-M Processors
Simulink Support Package for Raspberry Pi Hardware
MATLAB Support Package for Arduino Hardware
Simulink Support Package for Arduino Due Hardware
Embedded Coder Support Package for STMicroelectronics STM32F4-Discovery Board
DSP System Toolbox Support Package for ARM Cortex-A Processors
Embedded Coder Support Package for ARM Cortex-A Processors
MATLAB Support Pack

Tutorials

Tutorials

Getting Started with Embedded Coder Support Package for STMicroelectronics STM32F4-Discovery
Uses: Embedded Coder



Code Verification and Validation with PIL and External Mode
Uses: Embedded Coder

Code Optimization using CMSIS DSP Library
Uses: Embedded Coder, DSP System Toolbox

Examples

Push Button and LED
Uses: Embedded Coder

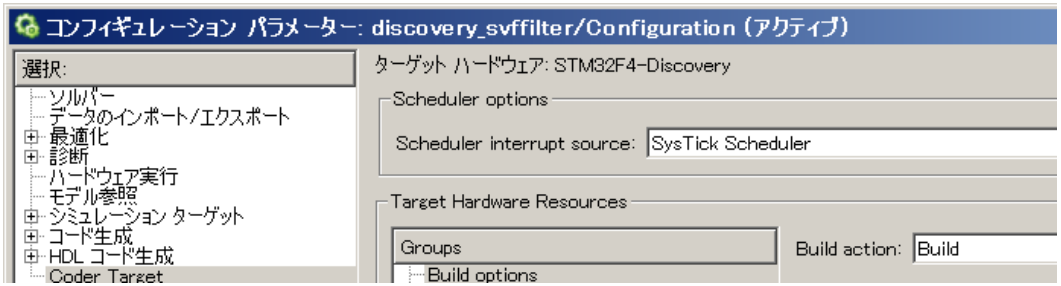
Parametric Audio Equalizer for STM32F4-Discovery Board
Uses: Embedded Coder, DSP System Toolbox

Asynchronous Scheduling
Uses: Embedded Coder

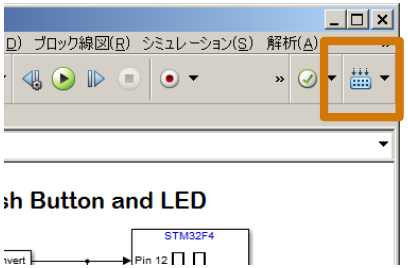
Examples

コード生成～マニュアル・ダウンロード手順

1. 例「Push Button and LED」をオープン
2. [Build action]を[Build]に設定

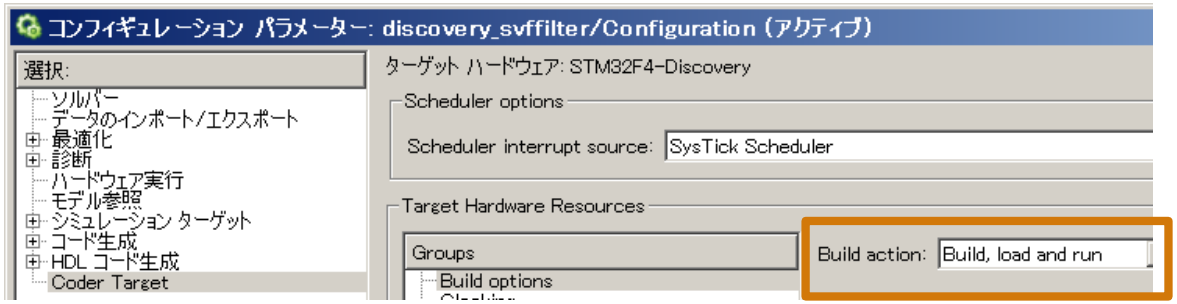


3. コード生成ボタン（モデルのビルド）を押下すると
ビルドが行われ、カレントディレクトリに**.hexが生成される。
4. ボードにUSB接続したら
STM32 ST-Link Utilityをオープン
5. [Target]メニュー⇒[Connect]
6. [Target]メニュー⇒[Program & Verify]
を選択し、生成された*.hexファイルを指定しダウンロード。

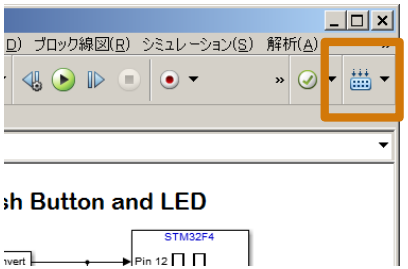


コード生成～オート・ダウンロード手順

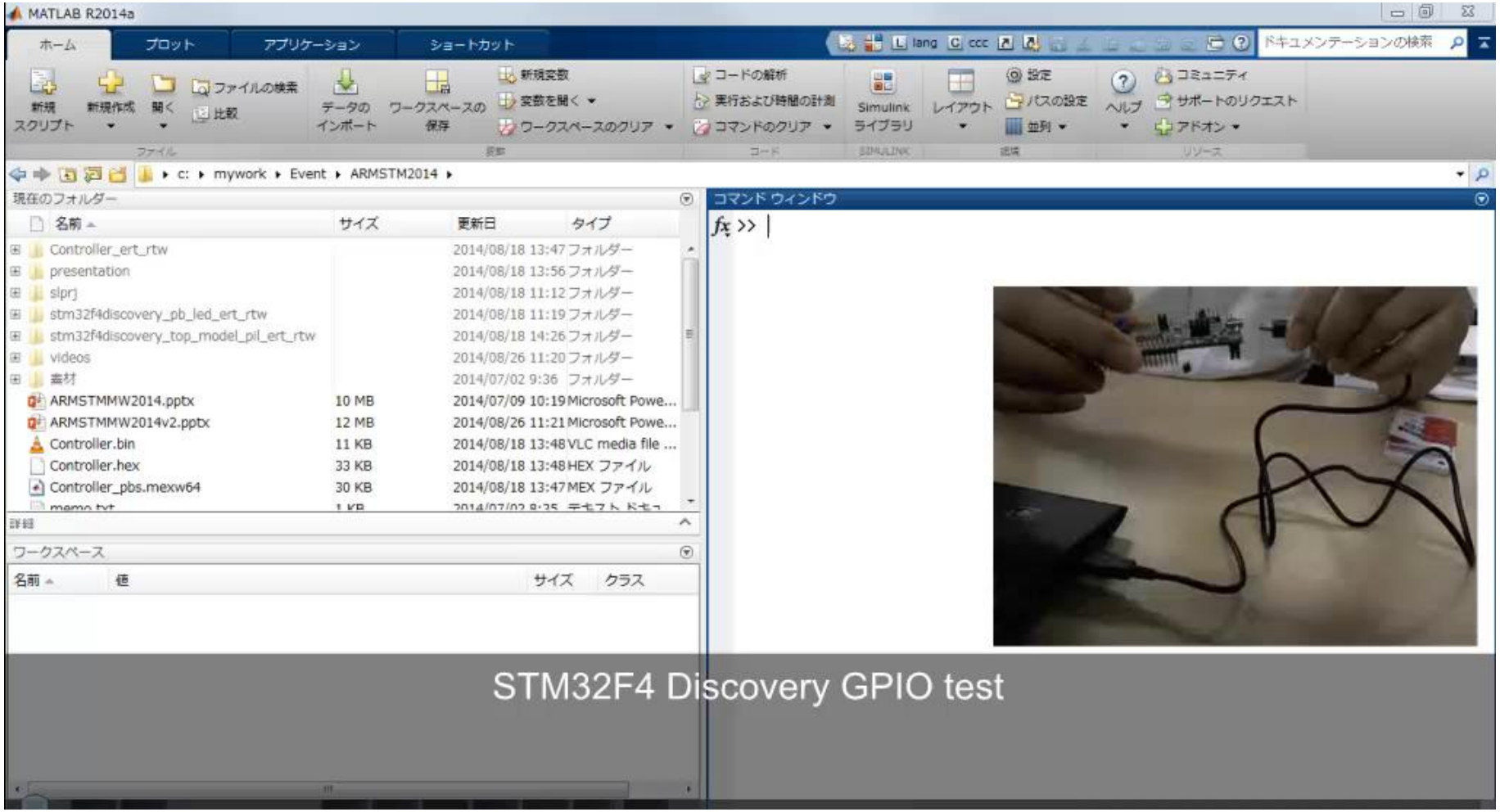
1. 例「Push Button and LED」をオープン
2. コンフィギュレーションパラメータの[Build action]を[Build, load and run]に設定



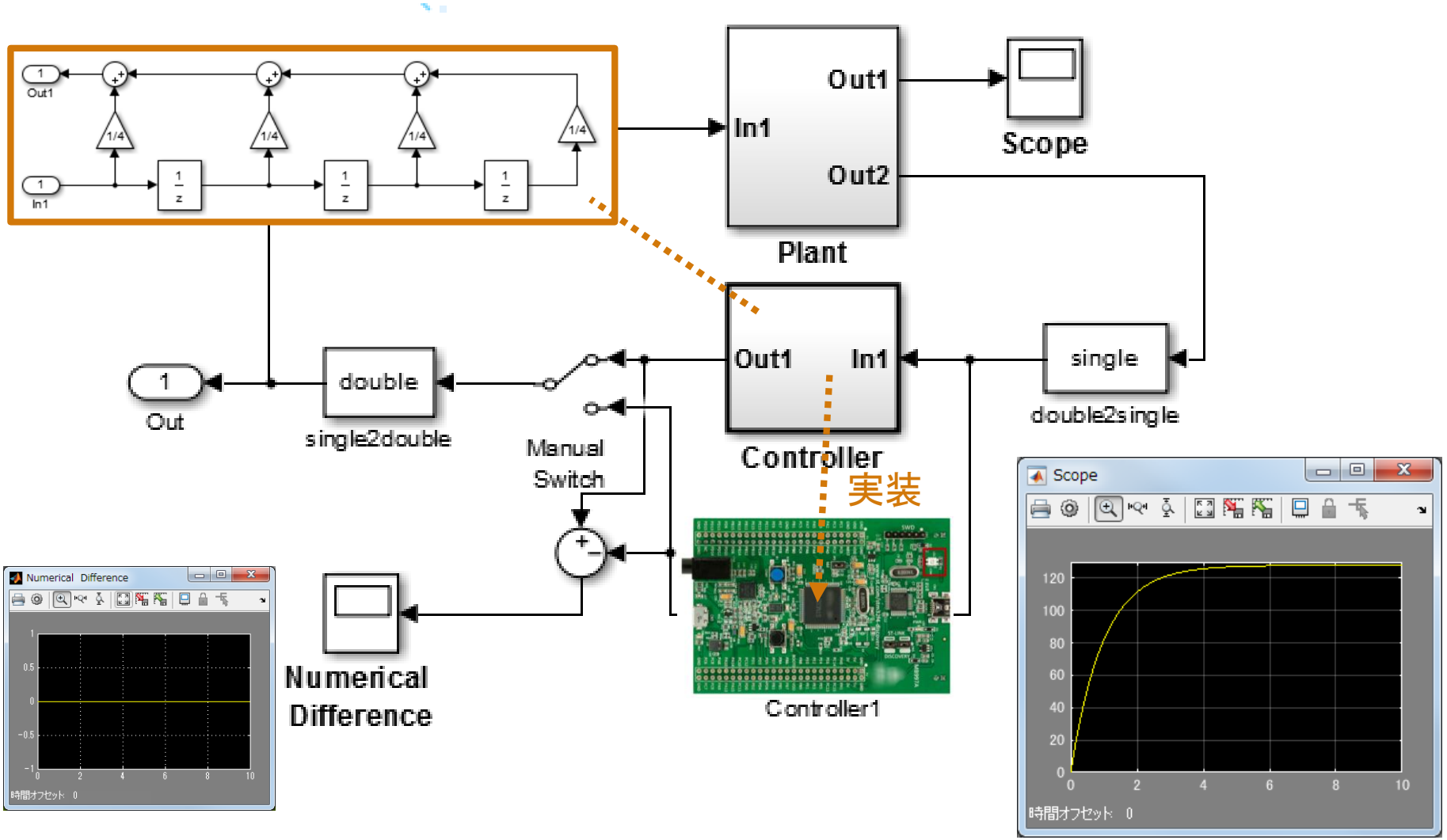
3. コード生成ボタン（モデルのビルド）を押下すると
ビルドが行われ、カレントディレクトリに**.hexが生成される。
4. プログラムのダウンロードまで自動的に行われる。



Demo（GPIOテスト）



PIL (Processor in the loop)



PIL (cont'd)

PILは2種類のインターフェースを選択可能

- ST-LinkによるPIL
  - Discovery-PC間のケーブル1本で実施できるので手軽
  - 通信速度が遅い
  - ベンチマーク結果（Tutorial: Code Verification and …）

    ```
    >> tic, sim(gcs), toc
    ```

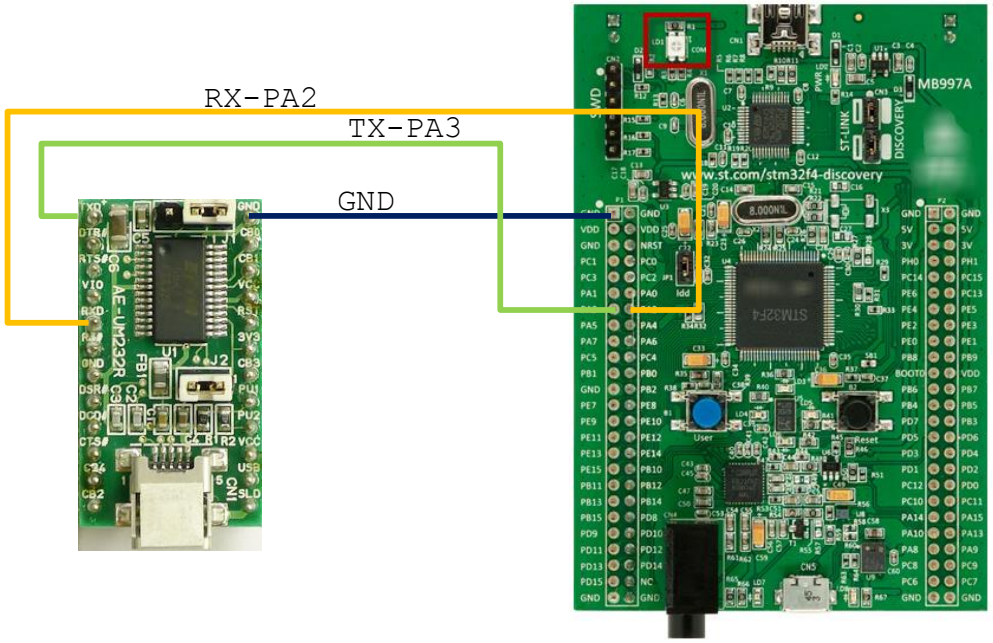
    経過時間は 207.729494 秒です。
- USBシリアルインターフェースによるPIL
  - PC-USB Serial-Discovery-PCで接続の必要がある
  - 通信速度が速い
  - ベンチマーク結果（同上）：ST-Linkより8倍高速

    ```
    >> tic, sim(gcs), toc
    ```

    経過時間は 25.948735 秒です。

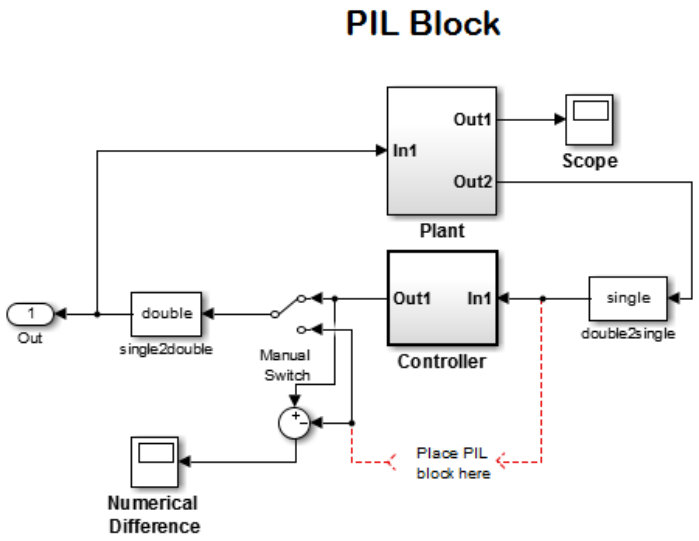
PIL/エクスターナルモード基板配線図

FT-232側ピン	STM32F4 Discovery側ピン
GND	GND
RX(RXD)	PA2
TX(TXD)	PA3

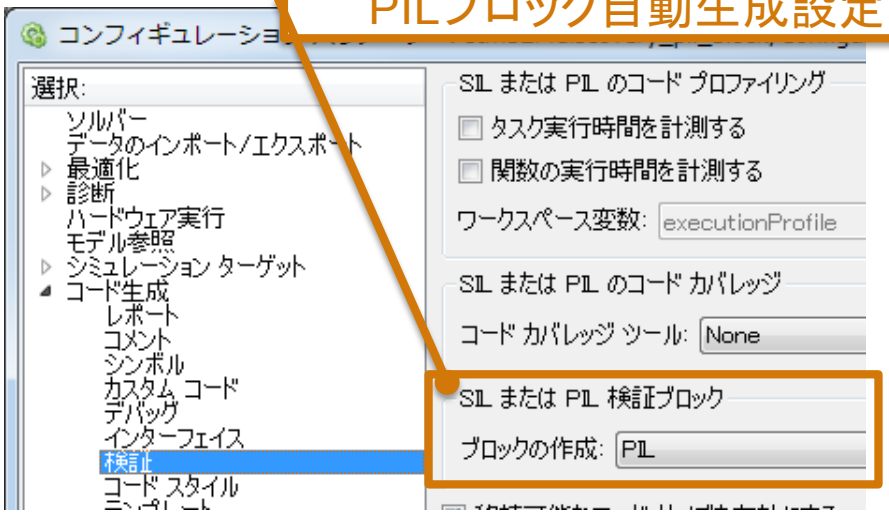


PIL (Processor in the loop)

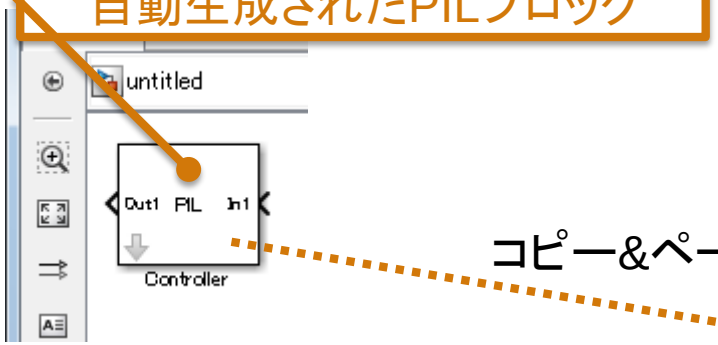
>>stm32f4discovery_pil_block



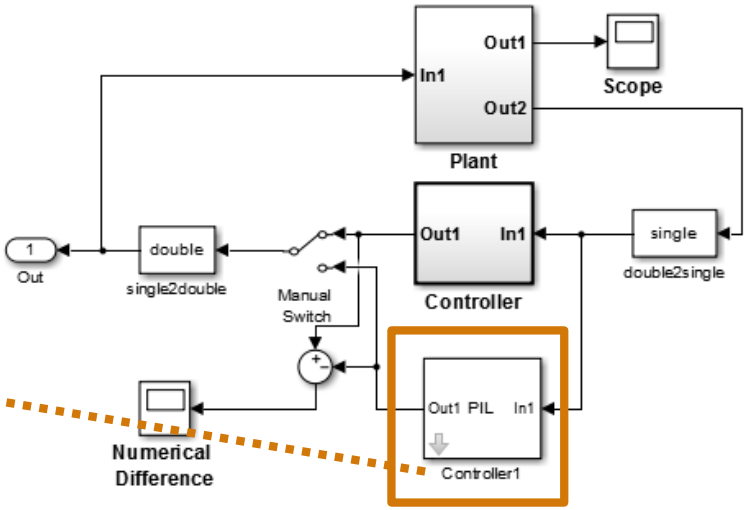
PILブロック自動生成設定



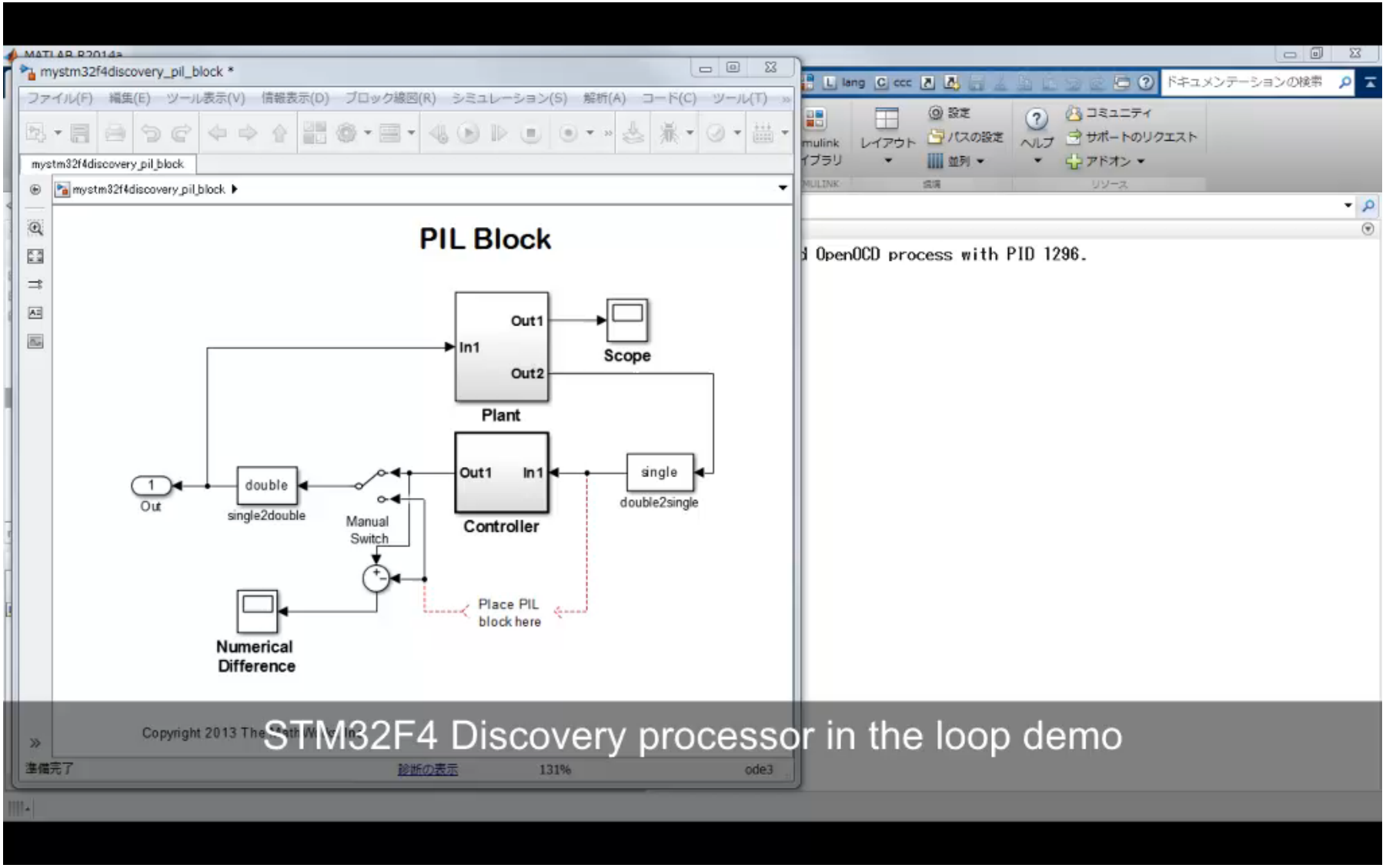
自動生成されたPILブロック



コピー&ペースト



Demo（PIL）


mystm32f4discovery_pil_block *
PIL Block
Out1
In1
Out2
Plant
Scope
1
Out
double
single2double
Manual
Switch
Out1
In1
Controller
single
double2single
Place PIL
block here
Numerical
Difference
OpenOCD process with PID 1296.
STM32F4 Discovery processor in the loop demo

Agenda

- Section1: フィルタとは？
- Section2: Case study
- Section3: フィルタの実装
- **Section4: フィルタ設計FAQ**
- まとめ

Agenda

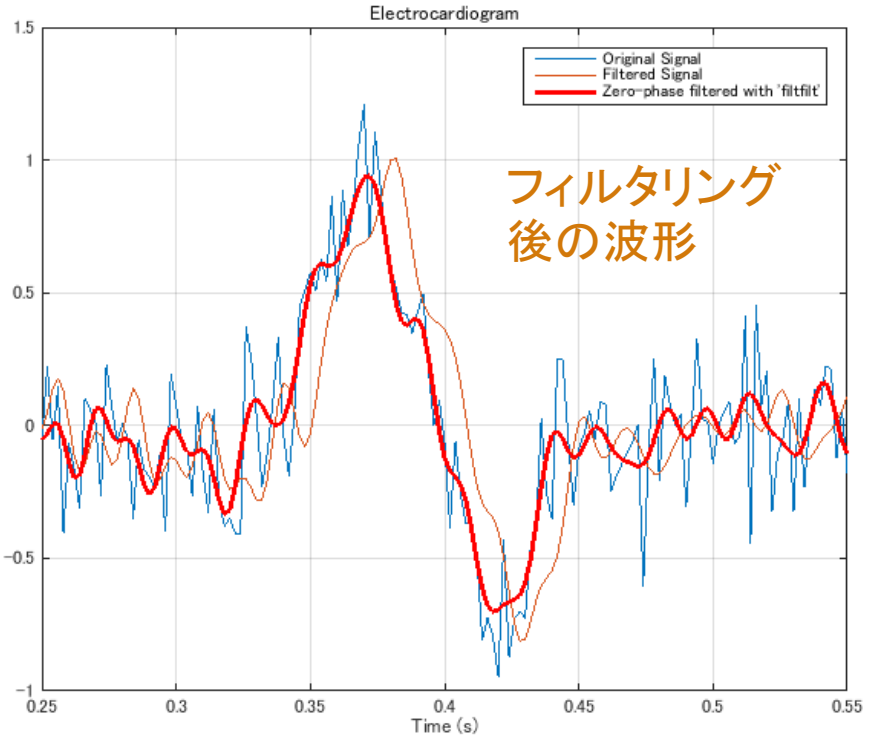
- **Section4: フィルタ設計FAQ**
  - フィルタの遅延補正の方法は？
  - 各種設計環境の違いは？
  - アナログフィルタは設計できますか？
  - 紙のデータにフィルタをかけるには？
  - 信号に欠損がある場合や、
    サンプリングが不等間隔のデータを扱うには？
  - テストベンチの効果的な作成方法は？
  - テキストやバイナリファイルの
    ストリーミング処理を実現するには？

Q:フィルタの遅延補正の方法は？

```
d = designfilt( 'lowpassiir' );
%%
grpdelay(d,N,Fs)
%%
out1 = filter(d,xn);

out2 = filtfilt(d,xn);
```

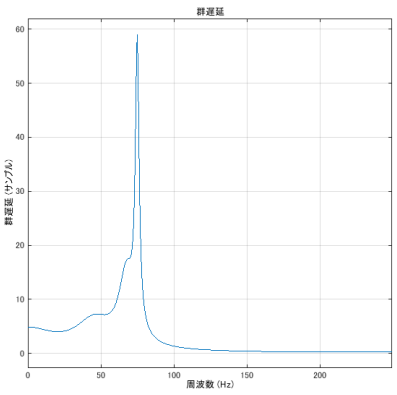
通常のフィルタ処理

遅延補正フィルタ



順方向と逆方向でフィルタリングすることで、
遅延補正および波形歪低減の効果

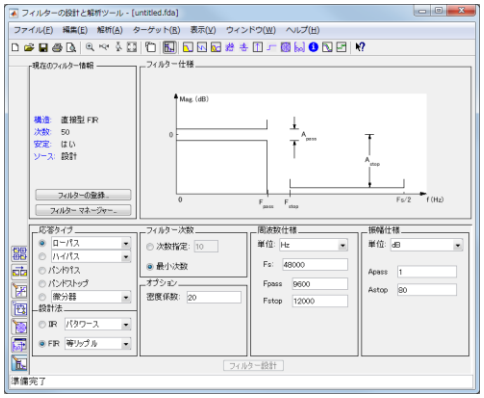
filtfilt関数で実現



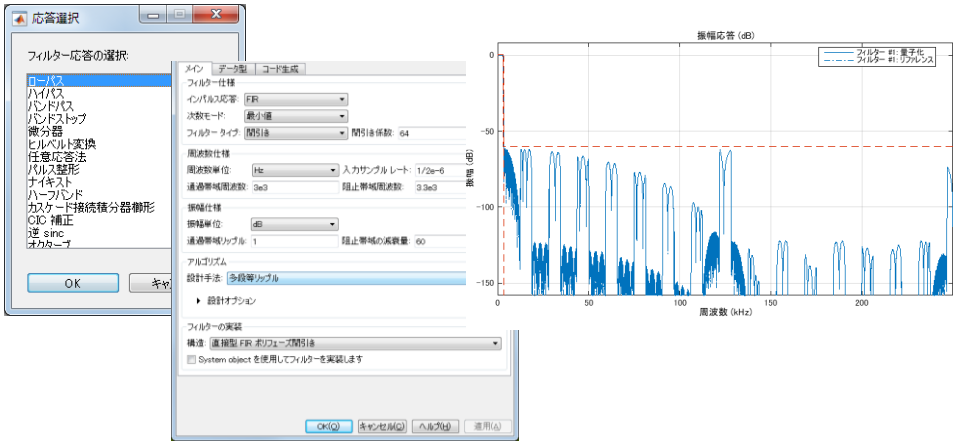
群遅延特性

Q:各種フィルタ設計環境の違いは？

- **FDATool**
  - 従来手法
  - フィルタの登録・カスケード
  - フィルタの変換（HPF⇒LPF等）
  - 固定小数点化
  - Cヘッダ、HDLコード生成



- **filterbuilder**
  - 仕様ベース
  - メソッド選択の試行錯誤削減
  - 固定小数点化、コード生成等、FDAToolの主機能を踏襲



- **Designfilt GUI**
  - R2014a新機能（filterbuilderがベース）
  - 仕様ベースの設計手法の手順を簡略化

(従来は2ステップのオブジェクト生成作業が必要)

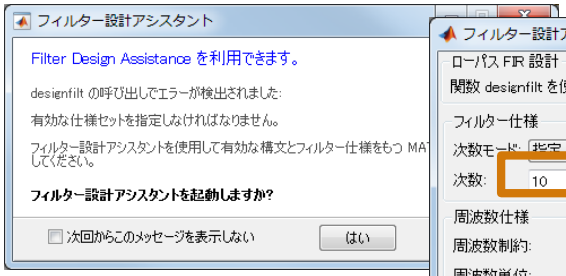
  - フィルタ設計フローのアシスト（足りない引数の候補推定等）

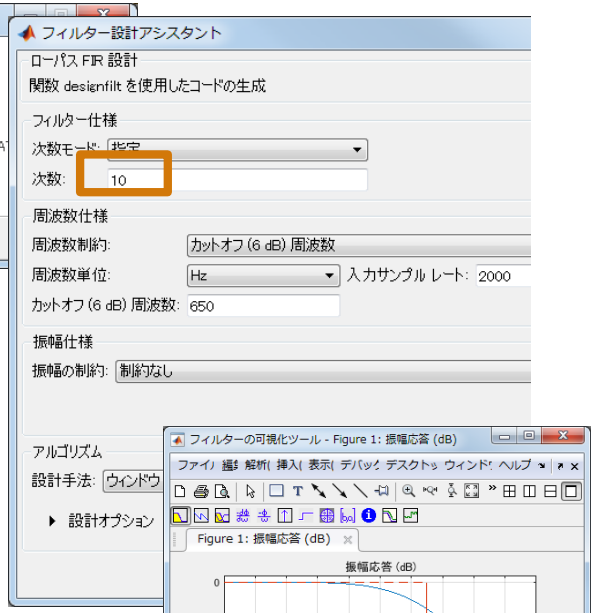
Q:各種フィルタ設計環境の違いは？(cont'd) R2014a

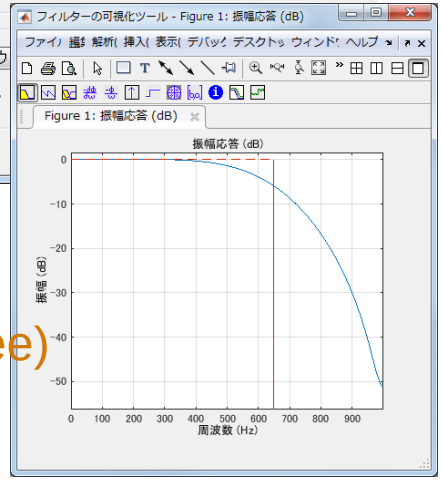
①サンプリング2000[Hz]、カットオフ650[Hz]のローパスFIRフィルタを作成

```
>> dee = designfilt('lowpassfir', 'CutoffFrequency', 650, 'SampleRate', 2000)
```

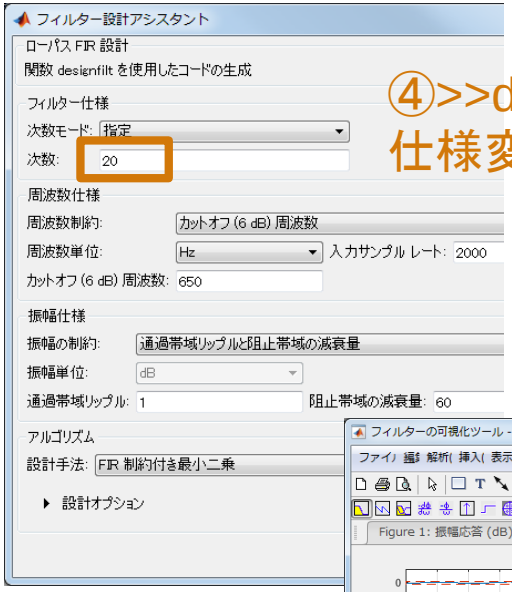
②足りない引数を補完するGUI



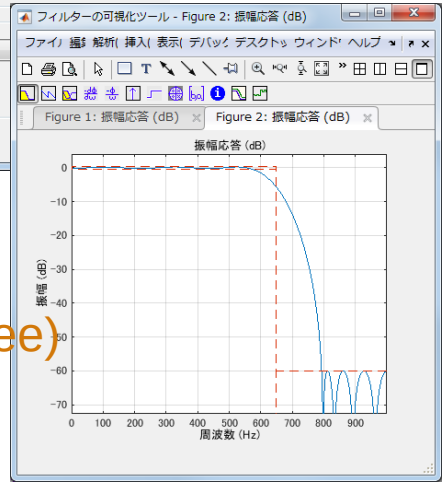




③>>fvtool(dee)
特性確認



④>>designfilt(dee)
仕様変更



⑤>>fvtool(dee)
特性確認

Q:アナログフィルタは設計できますか？

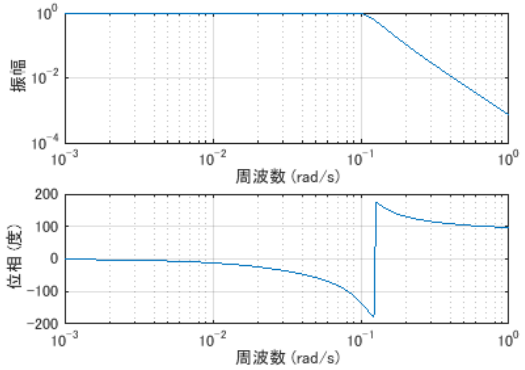
Case1. 仕様が決まっている場合

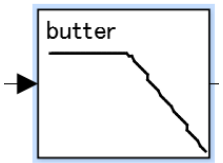
Signal Processing Toolbox™

bilinear	アナログ-デジタル フィルター変換用の双一次変換
besselap	ベッセル アナログ ローパス フィルターのプロトタイプ
besself	ベッセル アナログ フィルターの設計
buttap	バタワース フィルターのプロトタイプ
butter	バタワース フィルターの設計
cheb1ap	チェビシェフ I 型アナログ ローパス フィルターのプロトタイプ
cheb2ap	チェビシェフ II 型アナログ ローパス フィルターのプロトタイプ
cheby1	チェビシェフ I 型フィルターの設計 (通過帯域リップル)
cheby2	チェビシェフ II 型フィルターの設計 (阻止帯域リップル)
ellip	楕円フィルターの設計
ellipap	楕円アナログ ローパス フィルターのプロトタイプ

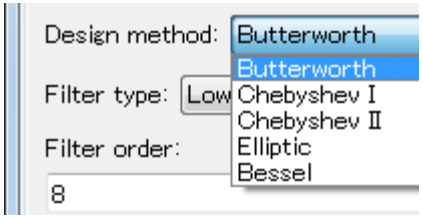
e.g.）

```
>> [b,a]=cheby1(3,0.5,0.1,'s');
>> freqs(b,a)
```



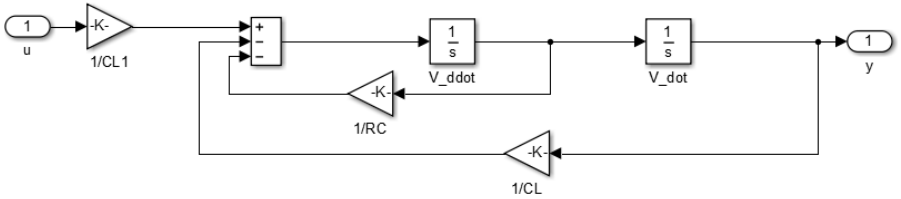


Analog Filter



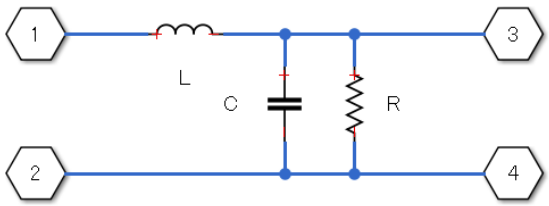
DSP System Toolbox

Case2. 回路が決まっている場合



伝達関数表現（Simlink）

$$G(s) = \frac{\frac{1}{CL}}{s^2 + \frac{1}{RC}s + \frac{1}{CL}}$$



回路表現（Simscape™）

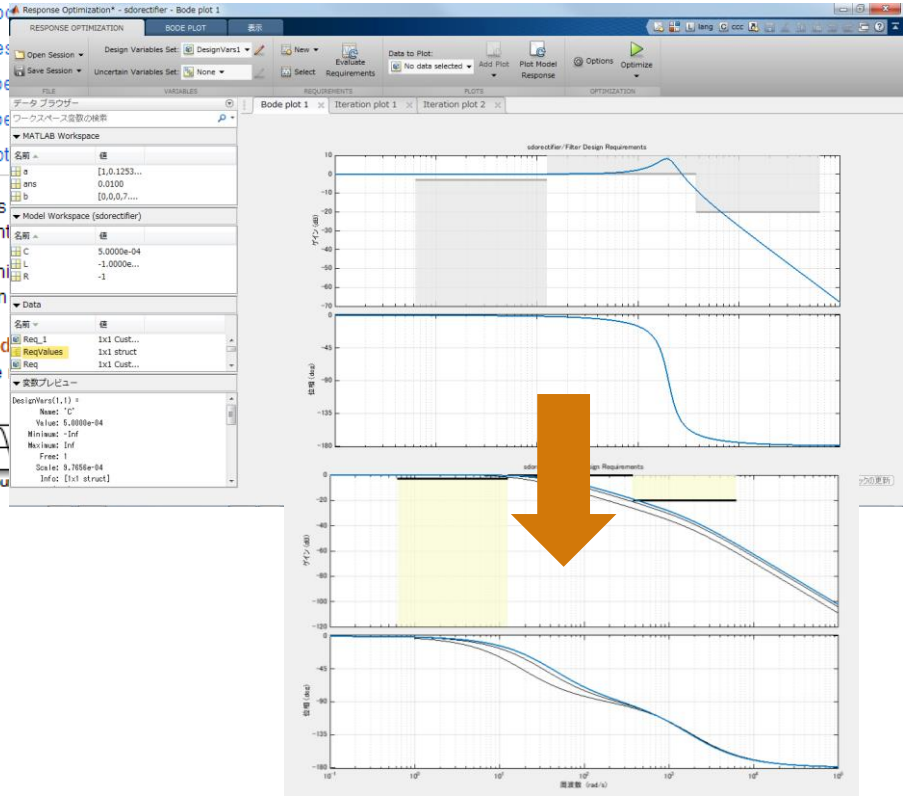
Q:アナログフィルタは設計できますか（cont'd）？

Simulink Design Optimization™ によるRLCフィルタ定数最適化

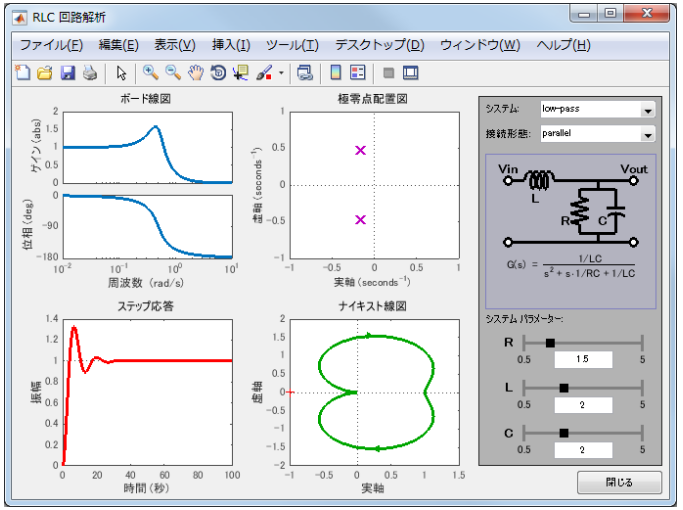
Simulink Design Optimization　Getting Started with Simulink Design Optimization

Design Optimization Using Frequency-Domain Check Blocks (GUI)

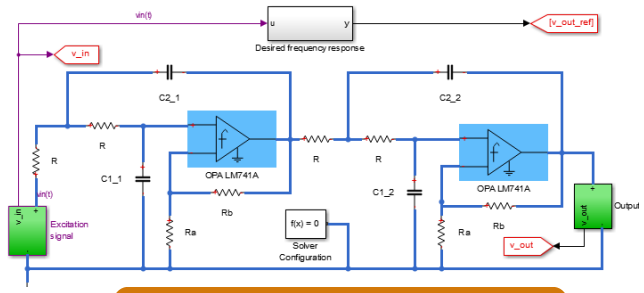
On this page…



Control System Toolbox™ のRLCフィルタ設計GUI

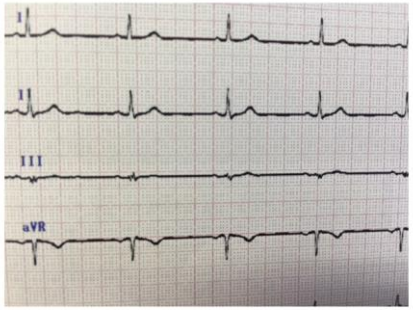


SimElectronics™ によるアクティブフィルタ

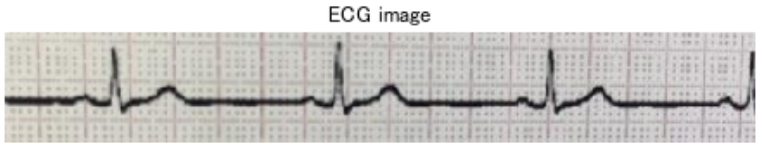


各種手法で実現

Q:紙のデータにフィルタをかけるには？



JPEG画像データ

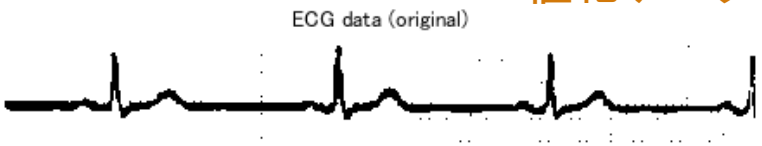



```
...
HR2 = imread('HR_s2.jpg');
...
HR2b = im2bw(HR2,0.55);
...
for k = 1:515;
    HR2f = find(HR2b(:,k));
    HR2mean(k) = mean(HR2f);
end
...
```

バイナリイメージ変換

列の重心（データ部のインデックス平均値）

二値化データ



スムージング＆ピークサーチ

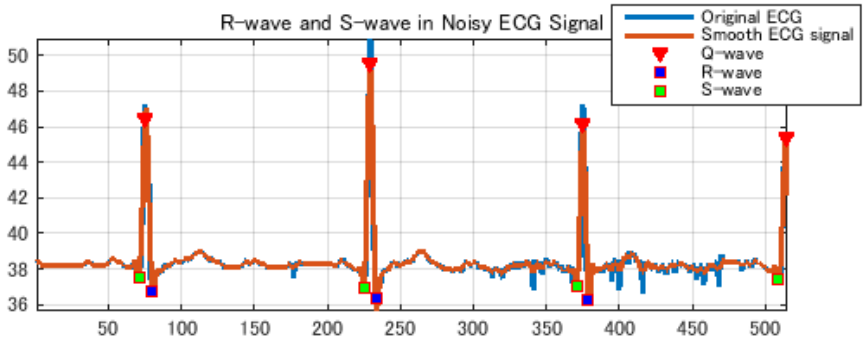


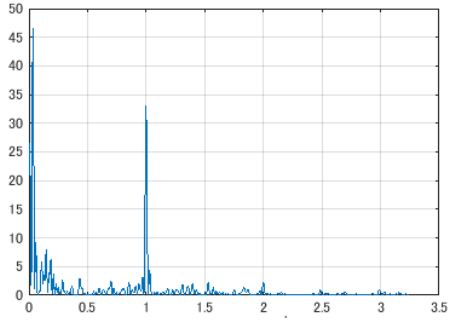
Image Processing Toolbox™の関数を応用

画像処理オプションをお試し下さい

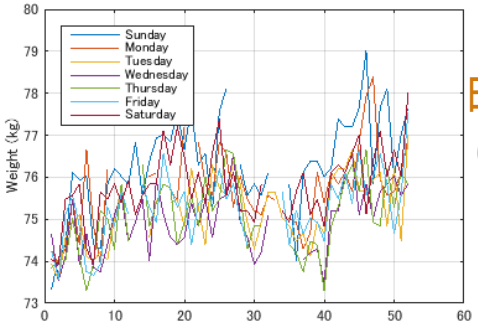
Q:信号に欠損がある場合や、サンプリングが不等間隔のデータを扱うには？

R2014b

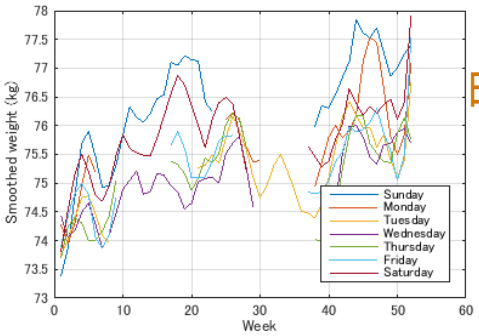
```
%% Lomb-Scargle法によるピリオドグラム推定
[p,f] = plomb(wgt,7,'normalized');
plot(f,p)
xlabel('Frequency (week^[-1])')
grid
%% 一周期（7日分）のデータで束ねる
wgd = reshape(wgt(1:7*52),[7 52])';
plot(wgd)
xlabel('Week')
ylabel('Weight (kg)')
grid
q = legend(datestr(datenum(2012,1,1:7),'dddd'));
q.Location = 'NorthWest';

%% スムージングフィルタ処理
wgs = sgolayfilt(wgd,3,7);
plot(wgs)
xlabel('Week')
ylabel('Smoothed weight (kg)')
grid
```



plomb関数
実行結果



時系列データ
（オリジナル）



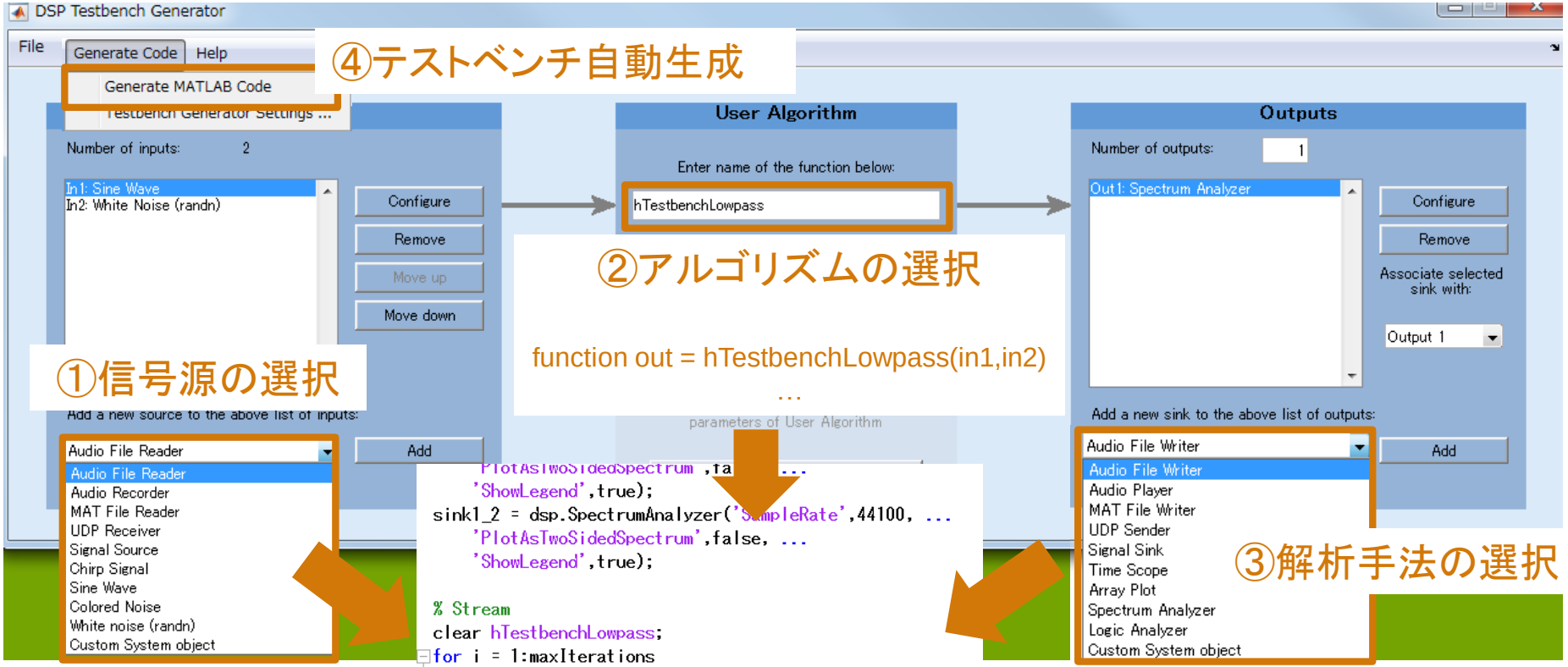
時系列データ
（フィルタ後）

plomb関数をお試し下さい

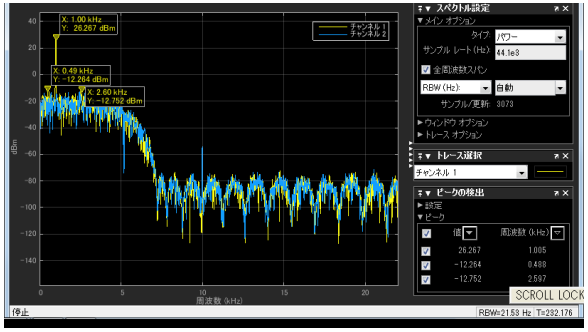
Q:テストベンチの効果的な作成方法は？

R2014b

>>testbenchGeneratorExampleApp



⑤自動生成されたテストベンチと実行結果



Appsで自動生成可能

Q: ストリーミング処理を実現するには？

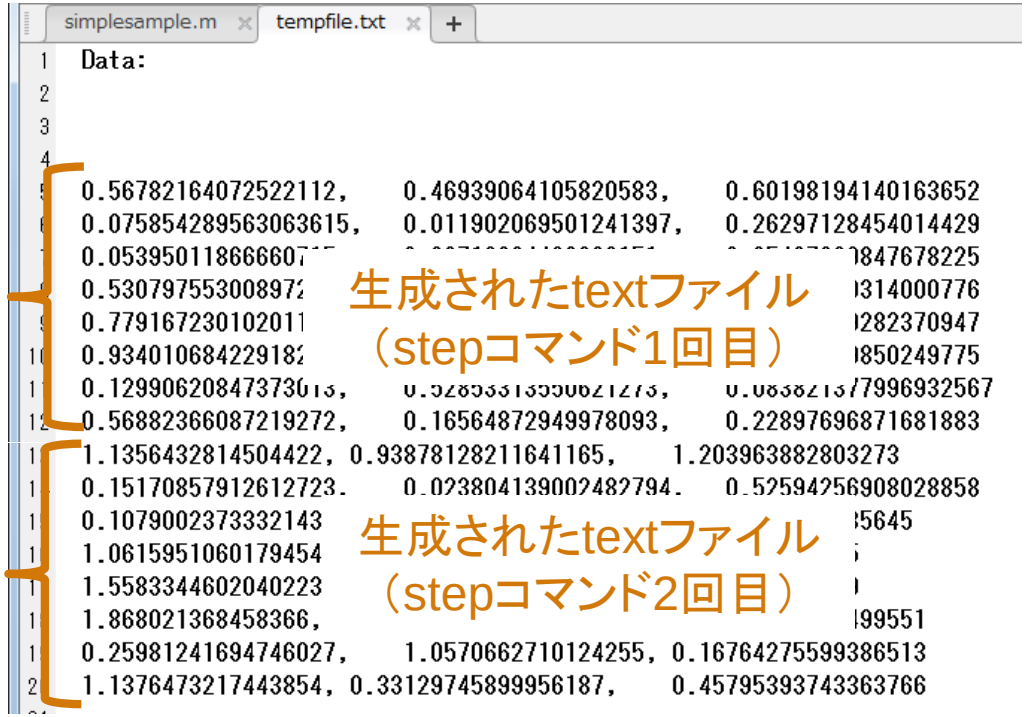
R2014b

- dspdemo.BinaryFileReader
- dspdemo.BinaryFileWriter
- dspdemo.TextFileReader
- dspdemo.TextFileWriter

e.g.)

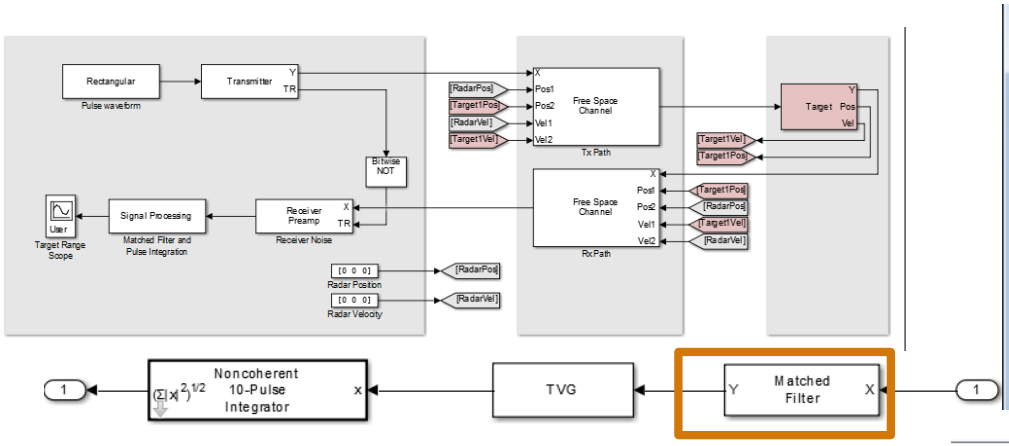
```
wobj = dspdemo.TextFileWriter
data = rand(8,3);
step(wobj,data)
%
step(wobj,2*data)
```

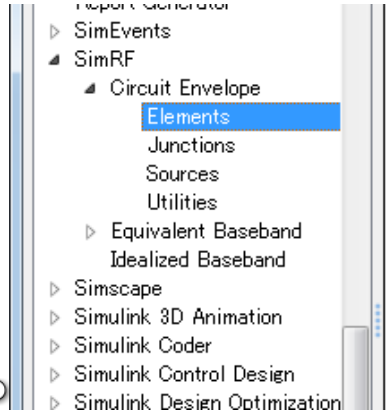
```
classdef TextFileReader < matlab.System &
                matlab.system.mixin.FiniteSource
properties (Nontunable)
                Filename        = 'tempfile.txt'
                HeaderLines  = 4
end

properties
                DataFormat  = '%g'
                Delimiter      = ','
                SamplePerFrame = 1024
end

methods(Access = protected)
                function setupImpl(obj)
                ...
                end
...
end
```

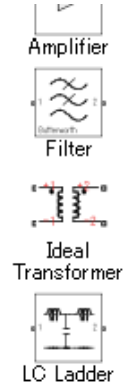


カスタマイズ可能

Q:各種アプリケーションへの応用例は？

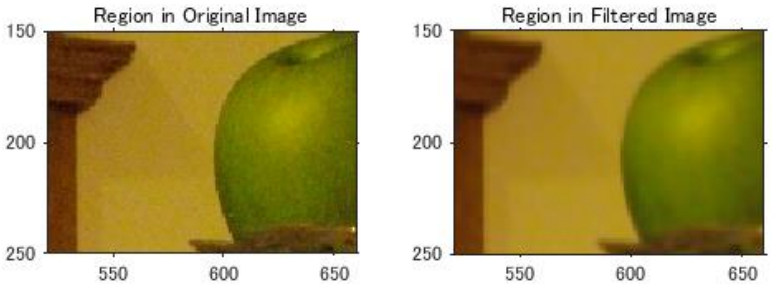




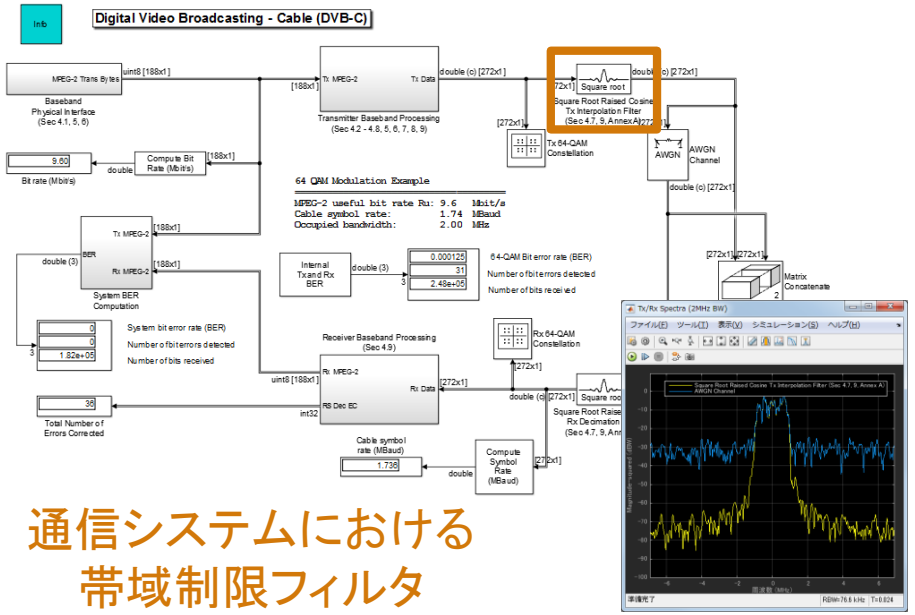


高周波用
フィルタブロック

レーダーシステムにおける
マッチドフィルタ



ガイド付きフィルタによる
エッジを保持したノイズ除去



通信システムにおける
帯域制限フィルタ

各種アプリケーションで応用

Agenda

- Section1: フィルタとは？
- Section2: Case study
- Section3: フィルタの実装
- Section4: フィルタ設計FAQ
- まとめ

まとめ

- **Case study**
  - 種々のケースにおけるフィルタの適用例
  - ベーシックな信号処理には**Signal Processing Toolbox**
  - 応用的な信号処理、Simulink用には**DSP System Toolbox**
- **フィルタの実装**
  - ARM Cortex-M, Cortex-Aの信号処理系ライブラリ対応
  - STMicro社Discoveryへの簡易実装
- **フィルタ設計FAQ**
  - 各種新機能
    - 不等間隔データ対応
    - テストベンチ自動生成等の各種新機能
    - ストリーミング処理

帰ったら
是非お試しください!

デモブースのご案内

信号処理 / 画像処理・
コンピュータビジョン　ソリューション