

ForceSensor コンポーネント・マニュアル

最終更新日：2010/1/14

目次

1	ForceSensor コンポーネント概要	2
2	動作環境	2
2.1	ソフトウェア	2
2.2	ハードウェア	2
2.3	ビルドに関して	2
3	データポートにより提供している機能	3
3.1	フィルタ No. やデータ出力範囲を変更する方法	4
3.2	フィルタ No. やデータ出力範囲を変更する際の注意	5
4	サービスポートにより提供している機能	6
4.1	要求した瞬間の力・モーメントを取得	7
4.2	ある期間での力・モーメントの最大値・最小値を取得	7
4.3	オフセット関連	7
4.4	データ出力範囲関連	8
4.5	フィルタ No. 関連	8
4.6	座標変換関連	9
5	ForcrSensor コンポーネントの状態遷移	10
5.1	onActivated	10
5.2	onExecute	10
5.3	onDeactivated	10
5.4	onError	10
6	その他	11

1 ForceSensor コンポーネント概要

「ForceSensor コンポーネント」は力センサを用いるコンポーネントで、力 3 成分、モーメント 3 成分を通常 1ms 周期ごとにデータポートから出力します。ロボット機能要素としてニッタ株式会社製の力センサ「IFS-70M35A25-M50B」を用いています。データ出力周期は ART に依存しており、ART で可能な範囲内で任意に設定できます。データポートだけの使用でも十分な能力を有しますが、さらにサービスポートを使うことで、様々な機能を利用することができます。

また、このコンポーネントを作成する際に [1] を参考にしています。

- 名称 : ForceSensor
- ロボット機能要素 : ニッタ株式会社製の力センサ「IFS-70M35A25-M50B」
- 出力データ : 6 個の double 型配列 [Fx[N], Fy[N], Fz[N], Mx[Nm], My[Nm], Mz[Nm]]
- データ出力周期 : 1ms(変更可能, 6 節参照)
- データ出力範囲 : [± 200 [N], ± 200 [N], ± 400 [N], ± 20 [Nm], ± 20 [Nm], ± 20 [Nm]](変更可能)

2 動作環境

本コンポーネントが動作を確認している動作環境は以下の通りです。

2.1 ソフトウェア

OS	Vine Linux 3.2 + ART-Linux 2.4.36
RT ミドルウェア	OpenRTM-aist 0.4.1 C++ 版
コンパイラ	gcc 3.3.2
依存ライブラリ	OpenRTM-aist に関するもの

本コンポーネントはソースファイルとして配布しています。適宜コンパイルを行い利用してください。

また、本コンポーネントは ART 環境で使用することを前提として作成されています。ART 環境でのコンポーネントの利用方法は OpenRTM の公式サイトを参照してください [2]。

2.2 ハードウェア

制御ボードの設置や機器の接続等はニッタ株式会社製の力センサ「IFS-70M35A25-M50B」付属のマニュアルを参照してください。ISA ボードのベースアドレスは 0x314 としてください。

2.3 ビルドに関して

特に特殊な操作は必要ありません。ファイルを展開したディレクトリに移動し、make を行って下さい。

```
make -f Makefile.ForceSensor
```

3 データポートにより提供している機能

概要で説明した通り，データポートからは力 3 成分とモーメント 3 成分の合計 6 成分が，double 型の配列として通常 1ms 周期ごとに出力されています．単位はそれぞれ力が [N]，モーメントが [Nm] となっています．

- 出力データ : 6 個の double 型配列 [$F_x[N]$, $F_y[N]$, $F_z[N]$, $M_x[Nm]$, $M_y[Nm]$, $M_z[Nm]$]
- データ出力周期 : 1ms(変更可能，6 節参照)
- データ出力範囲 : [$\pm 200[N]$, $\pm 200[N]$, $\pm 400[N]$, $\pm 20[Nm]$, $\pm 20[Nm]$, $\pm 20[Nm]$](変更可能)

図 1 に各出力データの方向を示します．座標原点はセンサ本体のちょうど中心になります．

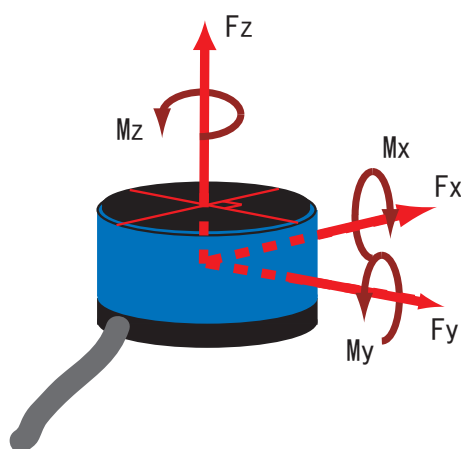


図 1 出力データの方向

さらに，出力データにはノイズ除去のための 7 段階のフィルタをかけることができます．フィルタはローパスフィルタであり，カットオフ周波数が低ければ低いほど出力データに遅延が生じることになります．各フィルタとカットオフ周波数，データの遅延の関係を表 1 に示します．フィルタ No. が 1 大きくなると，カットオフ周波数はだいたい 1/4 に，データ遅延はだいたい 4 倍になります．

表 1 フィルタ No. とカットオフ周波数，データ遅延の関係

フィルタ No.	カットオフ周波数 [Hz]	データ遅延 [ms]
0	なし	0
1	500	2
2	125	8
3	31.25	32
4	7.81	128
5	1.95	512
6	0.49	2048

通常はフィルタ 1 をかける仕様になっていますが，任意に変更することもできます．

3.1 フィルタ No. やデータ出力範囲を変更する方法

RTCLink 上で ForceSensor コンポーネントをクリックすると、通常図 2 に赤丸で示した位置にコンポーネント内部のパラメータを変更する画面 (ConfigurationView) が現れます。filter_No. がフィルタ No. に、fullscale_**が各データの出力範囲の正の限界値にそれぞれ対応しており、あらかじめ標準で使うであろう数値がセットされています。数値を変更する場合、value に任意の数字を入力後、Apply ボタンをクリックしてください。フィルタやデータの出力範囲が変更されます。

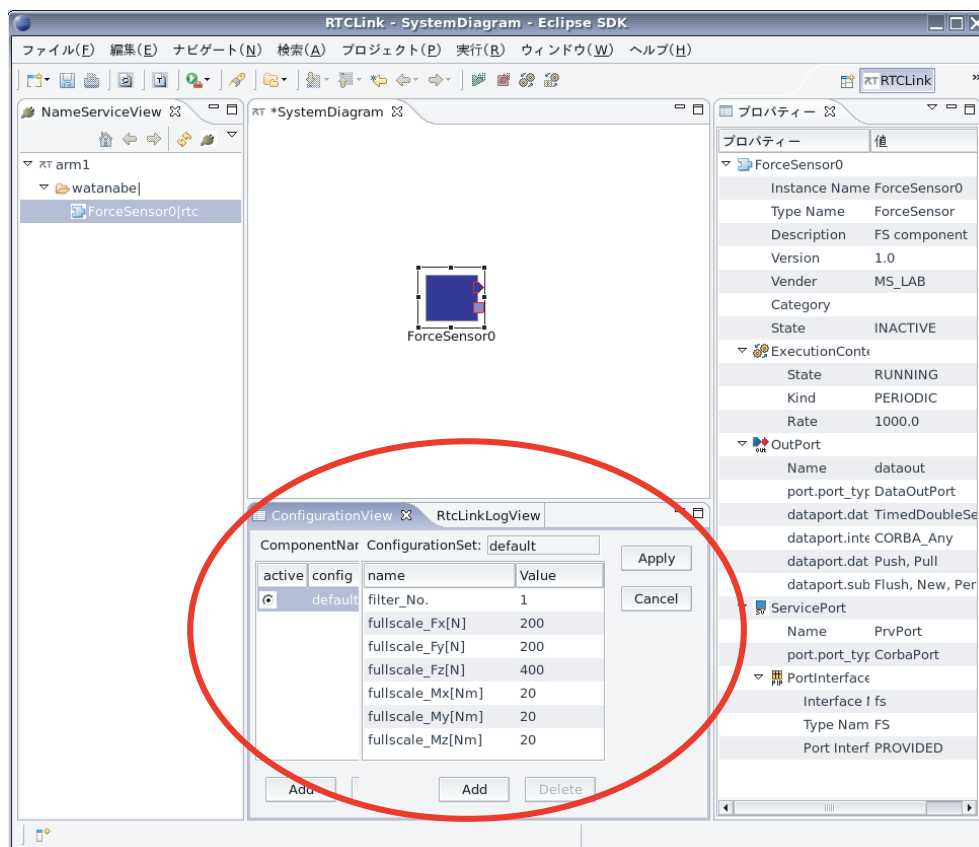


図 2 RTCLink 画面

但し、これらの数値を変更する場合、いくつかの注意点があります。3.2 節を参考にしてください。

3.2 フィルタ No. やデータ出力範囲を変更する際の注意

ForceSensor コンポーネントの内部パラメータを ConfigurationView から変更する際、いくつか注意点があります。

1. フィルタ No. を変更する際、入力数値は必ず 0～6 の整数にしてください。
小数や範囲外の数値が入力された場合、正常な動作は保障できません。^{*1}
2. データ出力範囲を変更する場合、限界下限を必ず守ってください。また、入力数値は必ず整数にしてください。
データの出力範囲には限界下限、推奨下限、推奨上限が決められています。表 2 を参考にしてください。推奨上下限はハード側で推奨されている値です。出力範囲をあまり小さくしすぎるとデータはすぐに飽和してしまうし、逆に大きくしすぎると分解能が失われてしまいます。これらが守られない場合、正常な動作は保障できません。^{*2}なお、ここで設定できる fullscale とはデータ出力範囲の正の限界値のことであり、負の限界値にはこれをマイナスにしたものが設定されます。
また、プログラム内部の計算の都合上、ConfigurationView で設定した出力範囲と実際に力センサに設定されている出力範囲は厳密には異なります。例として出力範囲を ± 200 [N] に設定した場合、実際に力センサに設定される限界値は ± 199.999 [N] といった感じになるはずですが、実際に力センサに設定されている出力範囲を取得したい際は、4.4 節を参照してください。
3. フィルタ No. やデータ出力範囲を変更する際は、必ず Activate 前に変更を行ってください。
これはコンポーネントの仕様で、Activate のタイミングでフィルタ No. やデータ出力範囲を力センサ本体に設定しているからです。Activate 後にこれらの数値を ConfigurationView 上から変更しても、一度 onDeactivated 状態にし、再度 Activate するまで変更した数値は力センサ本体に設定されません。

表 2 データ出力範囲の上下限

成分	限界下限範囲	推奨下限範囲	標準範囲	推奨上限範囲
Fx[N]	± 2	± 200	± 200	± 1040
Fy[N]	± 2	± 200	± 200	± 1020
Fz[N]	± 3	± 400	± 400	± 2030
Mx[Nm]	± 1	± 9	± 20	± 35
My[Nm]	± 1	± 9	± 20	± 32
Mz[Nm]	± 1	± 10	± 20	± 37

^{*1} 範囲外の数値が入力された場合は、コンポーネントは ERROR 状態になります。

^{*2} 推奨値を超えた範囲が設定された場合、コンポーネントはコンソール上に *Configuration_Parameter Over* と表示してから動作を始めます。限界値を超えた範囲が設定された場合は、コンポーネントは ERROR 状態になります。

4 サービスポートにより提供している機能

ForceSensor コンポーネントは、サービスポートにより以下に示すような様々なサービスを利用することができます。

```
short Get_Wrench(inout DoubleArray Force);
short Get_DigitalWrench(inout DoubleArray Force);
short Reset_Peak();
short Get_MaxPeak(inout DoubleArray Force);
short Get_MinPeak(inout DoubleArray Force);
short Set_Offset();
short Get_Offset(inout DoubleArray Offset);
short Set_Fullscale(inout ShortArray Force);
short Get_Fullscale(inout DoubleArray Force);
short Set_Filter(inout short num);
short Get_Filter(inout short num);
short Set_ToolCoordinate(inout DoubleArray Tool);
short Get_ToolCoordinate(inout DoubleArray Tool);
```

引数の (inout DoubleArray Force) などはサービスポートで関数を宣言する際の特異な表現です。ForceSensor コンポーネントでは typedef double DoubleArray [6];, typedef short ShortArray [6]; と定義しており、例えば (inout DoubleArray Force) は (double Force[6]) と同じ意味になります。

それぞれの関数の説明を以下で行います。

4.1 要求した瞬間の力・モーメントを取得

```
short Get_Wrench(inout DoubleArray Force);  
short Get_DigitalWrench(inout DoubleArray Force);
```

Get_Wrench(inout DoubleArray Force)^{*3}は、サービスが呼び出された瞬間の力とモーメントの値を引数 Force に返します。これを使うことで、データポートから垂れ流されている力とモーメントだけでなく、ある瞬間の力とモーメントを知ることができます。

short Get_DigitalWrench(inout DoubleArray Force) も、サービスが呼び出された瞬間の力とモーメントの値を引数 Force に返します。しかし、上と違ってこちらは単位付きの物理量に変換する前のデジタル値を得ることができます。得られるデジタル値は、設定したデータ出力範囲の限界で-32767～32766 となります。

これらのサービスが終了したとき、サービスは 0 を返します。

4.2 ある期間での力・モーメントの最大値・最小値を取得

```
short Reset_Peak();  
short Get_MaxPeak(inout DoubleArray Force);  
short Get_MinPeak(inout DoubleArray Force);
```

コンポーネントを Activate した瞬間から、力センサは自身にかかる力・モーメント各成分方向の最大値、最小値を記憶しています。

Get_MaxPeak(inout DoubleArray Force), Get_MinPeak(inout DoubleArray Force) は、それぞれ呼び出された時点での各成分方向の最大値、最小値を引数 Force に返します。

Reset_Peak() は、それまで力センサ内に記憶されていた最大値、最小値をリセットします。

これらのサービスが終了したとき、サービスは 0 を返します。

4.3 オフセット関連

```
short Set_Offset();  
short Get_Offset(inout DoubleArray Offset);
```

Set_Offset() は、呼び出された時点で力センサにかかっている力、モーメントを、オフセット（基準点）として力センサに設定します。

Get_Offset(inout DoubleArray Offset) は、呼び出された時点で力センサに設定されているオフセットを、引数 Offset に返します。

これらのサービスが終了したとき、サービスは 0 を返します。

^{*3} Wrench とは工具のレンチのことではなく、力とモーメントの両方を意味する言葉らしいです。

4.4 データ出力範囲関連

```
short Set_Fullscale(inout ShortArray Force);  
short Get_Fullscale(inout DoubleArray Force);
```

Set_Fullscale(inout ShortArray Force) は、引数 Force で与えられている配列を、データ出力範囲として力センサに設定します。ForceSensor コンポーネントは Activate 時に ConfigurationView の数値を出力範囲として設定しますが、その後このサービスを用いることで、新たな出力範囲を設定することができます。^{*4}なお、このサービスを利用してのデータ出力範囲の変更と ConfigurationView を利用しての出力範囲の変更との間に違いはありません。ConfigurationView 上でできることをサービスとしても提供しているだけです。

引数 Force は図 3 を参考にして必ず short 型の整数配列で与えてください。また、設定の上下限は表 2 を参考にしてください。限界下限を超えた数値を与えた場合、データ出力範囲は変更されず、サービスは-1 を返します。適切な数値が与えられた場合、サービスは 0 を返します。

	Force[0]	Force[1]	Force[2]	Force[3]	Force[4]	Force[5]
short Force[6]	fullscale	fullscale	fullscale	fullscale	fullscale	fullscale
	▪ Fx[N]	▪ Fy[N]	▪ Fz[N]	▪ Mx[Nm]	▪ My[Nm]	▪ Mz[Nm]

図 3 Set_fullscale 時に与える配列とその中身

Get_Fullscale(inout DoubleArray Force) は、呼び出された時点で力センサに設定されているデータ出力範囲を、引数 Force に返します。3.2 節で「必ずしも設定した出力範囲までデータが得られるとは限りません」と書きましたが、このサービスを呼ぶことで得られる数値が、実際に力センサに設定されているデータ出力範囲です。

このサービスが終了したとき、サービスは 0 を返します。

4.5 フィルタ No. 関連

```
short Set_Filter(inout short num);  
short Get_Filter(inout short num);
```

Set_Filter(inout short num) は、引数 num で与えられている数値を、フィルタ No. として設定します。ForceSensor コンポーネントは Activate 時に ConfigurationView の数値をフィルタ No. として設定しますが、その後このサービスを用いることで、新たなフィルタ No. を設定することができます。^{*5}なお、このサービスを利用してのフィルタの変更と ConfigurationView を利用してのフィルタの変更との間に違いはありません。ConfigurationView 上でできることをサービスとしても提供しているだけです。

引数 num は必ず short 型の 0~6 の整数で与えてください。それ以外の整数が与えられた場合、フィルタ No. は変更されず、サービスは-1 を返します。適切な数値が与えられた場合、サービスは 0 を返します。

^{*4} このサービスを用いて出力範囲を変更しても、ConfigurationView 上に表示されている出力範囲は変更されません。

^{*5} このサービスを用いてフィルタを変更しても、ConfigurationView 上に表示されているフィルタ No. は変更されません。

Get_Filter(inout short num) は、呼び出された時点で設定されているフィルタ No. を、引数 num に返します。

このサービスが終了したとき、サービスは 0 を返します。

4.6 座標変換関連

```
short Set_ToolCoordinate(inout DoubleArray Tool);
short Get_ToolCoordinate(inout DoubleArray Tool);
```

この力センサは、センサ座標の原点および方向を変更するために、座標変換を行うことができます。この機能を使うと、たとえばセンサ座標をツール座標にあわせることができるようになります。[1]

Set_ToolCoordinate(inout DoubleArray Tool) は、座標変換を行うためのサービスです。引数 Tool に力センサ座標系から見たツール座標系の位置と姿勢を書き込み、それを力センサに設定します。

引数 Tool は図 4 を参考にして与えてください。また、入力数値の上下限は表 3 を参考にしてください。ここで、Tx,Ty,Tz は小数でもかまいませんが、反映されるのは小数点以下第 1 位までです。Rz,Ry,Rx も小数でもかまいませんが、プログラム内部で Amount = (int)(Amount*32768/180) として再計算されるため、あまり小数点以下深いところまでは反映されません。Tool 配列で限界値を超えた値が与えられたとき、座標変換は適用されず、サービスは-1 を返します。適切な数値が与えられた場合、サービスは 0 を返します。

	Tool [0]	Tool [1]	Tool [2]	Tool [3]	Tool [4]	Tool [5]
double Tool [6]	Tx [mm]	Ty [mm]	Tz [mm]	Rz [deg]	Ry [deg]	Rx [deg]

図 4 Set_ToolCoordinate 時に与える配列とその中身

表 3 座標変換の種類と変換量

成分	成分の意味	数値の上下限
Tx	位置：x 軸に沿って座標を平行移動	± 3276[mm]
Ty	位置：y 軸に沿って座標を平行移動	± 3276[mm]
Tz	位置：z 軸に沿って座標を平行移動	± 3276[mm]
Rz	姿勢：z 軸を中心に座標を回転	± 180[deg]
Ry	姿勢：(新たな)y 軸を中心に座標を回転	± 180[deg]
Rx	姿勢：(新たな)x 軸を中心に座標を回転	± 180[deg]

Get_ToolCoordinate(inout DoubleArray Tool) は、座標変換を行うために与えられた最新の Tool 配列を、引数 Tool に返します。

このサービスが終了したとき、サービスは 0 を返します。

5 ForcrSensor コンポーネントの状態遷移

ここでは、各状態遷移時に ForceSensor コンポーネントがどのような処理を行っているのかを、簡単に紹介します。とはいえ、onInitialize、onFinalize 状態ではコンポーネントは力センサに対して特に処理を行わないため、ここでは除外します。

5.1 onActivated

まず、Activate 時点でコンポーネントが認識している ConfigurationView の値をコンソール上に出力します。値を変更していた場合、正しいかどうかを確認してください。

次に、ConfigurationView に書き込まれている値が適切な範囲であるかを判断します。範囲外であった場合、コンポーネントは onError 状態になります。範囲内であれば、それらの値を力センサに設定します。ここでデータ出力範囲とフィルタ No. を力センサに設定するので、ConfigurationView での数値の変更は必ず Activate 前に行ってください。

最後にオフセットの設定と、最大値、最小値のリセットを行います。Activate 時点で力センサにかかっている力・モーメントを、オフセットとして設定します。(力センサがマニピュレータの手先についているなど、力センサにかかる重力方向が Activate 後に変化する場合、状況に応じてその都度オフセットをかけ直したほうが良いと思います。)

5.2 onExecute

ForceSensor コンポーネントでは、onExecute は通常 1ms 周期で周期実行されます(周期は ART 依存で変更可能、6 節参照)。毎周期ごとに力センサの各成分のデータを読み取り、単位付きの物理量 [N]、[Nm] に変換してデータポートから 6 個の配列として出力しています。

5.3 onDeactivated

座標変換されていた場合、座標変換をリセットします。これにより、別のシステムでこの力センサを使用する際に、以前のシステムでの座標変換が残ってしまっている、という事態を防ぎます。

5.4 onError

RTCLink 上の ConfigurationView に書き込まれた値が範囲外だった場合、ForceSensor コンポーネントは onError 状態になります。この状態から脱出する方法は特に実装していないので、素直に RTCLink 上でコンポーネントの Reset を行った後、ConfigurationView の数値を適切なもの書き換え、改めて Activate を行ってください。

6 その他

ForceSensor コンポーネントの使用する `rtc.conf` ファイルには、以下のように書かれています。

```
corba.nameservers: arm1
naming.formats: %h %n.rtc
logger.enable:NO

exec_cxt.periodic.type: ArtExecutionContext
exec_cxt.periodic.rate: 1000
manager.modules.load_path: ./
manager.modules.abs_path_allowed: yes
```

この中で、ForceSensor コンポーネントを使用する上で特に重要な二つについて説明します。

```
corba.nameservers: arm1
```

ForceSensor コンポーネントを起動させるホストの PC 名が記述してあります。arm1 上で起動させることを想定しているので arm1 と記述してありますが、別の PC 上で起動させる場合は名前を書き換えてください。

```
exec_cxt.periodic.rate: 1000
```

onExecute の実行周期が [Hz] で記述してあります。今は 1ms での周期実行を想定しているので 1000[Hz] と記述してありますが、例えば 5ms で周期実行させたい場合は 200[Hz] に変更してください。

なお、力センサ本体は [Fx,Fy,Fz,Mx,My,Mz] のデータのセットを 125 μ s 間隔で出力しています。ここから考えると、`exec_cxt.periodic.rate` に設定できる数値の、力センサ本体の使用から見た上限値は 8000 であるということになります。ただし、実際にはこの数値は ART にも依存しているため、ART で実現可能な実行周期より早くなるような数値は設定できません。

参考文献

- [1] ニッタ株式会社：Intellijent Force Sensor レシーバーボード 取扱説明書
- [2] ARTLinux 用実行コンテキスト -OpenRTM-aist- : <http://www.is.aist.go.jp/rt/OpenRTM-aist/html/TIPS2FARTLinuxE794A8E5AE9FE8A18CE382B3E383B3E38386E382ADE382B9E38388.html>, 2010/01/14 閲覧