EPSON
EXCEED YOUR VISION

FSA サンプルプログラム
マニュアル

-姿勢検出-

セイコーエプソン株式会社

Rev.1.0

本資料のご使用につきましては、次の点にご留意願います。

本資料の内容については、予告無く変更することがあります。

1. 本資料の一部、または全部を弊社に無断で転載、または、複製など他の目的に使用することは堅くお断りいたします。

2. 本資料に掲載される応用回路、プログラム、使用方法等はあくまでも参考情報であり、これらに起因する第三者の知的財産権およびその他の権利侵害あるいは損害の発生に対し、弊社はいかなる保証を行うものではありません。また、本資料によって第三者または弊社の知的財産権およびその他の権利の実施権の許諾を行うものではありません。

3. 特性値の数値の大小は、数直線上の大小関係で表しています。

4. 製品および弊社が提供する技術を輸出等するにあたっては「外国為替および外国貿易法」を遵守し、当該法令の定める手続きが必要です。大量破壊兵器の開発等およびその他の軍事用途に使用する目的をもって製品および弊社が提供する技術を費消、再販売または輸出等しないでください。

5. 本資料に掲載されている製品は、生命維持装置その他、きわめて高い信頼性が要求される用途を前提としていません。よって、弊社は本（当該）製品をこれらの用途に用いた場合のいかなる責任についても負いかねます。

6. 本資料に掲載されている会社名、商品名は、各社の商標または登録商標です。

目　次

1. 概要

本 FSA サンプルプログラムは、3 軸ジャイロセンサー、3 軸加速度センサー、3 軸地磁気センサーを用いて姿勢検出を実現します。
下記に示すような特徴を備えています。

1）エプソンオリジナルのアルゴリズムによる、高精度姿勢検出を実現します。

2）FSA による高効率な演算処理によって、低クロックでの処理を実現します。

3）工場出荷時のキャリブレーションを不要とするフィルタリング処理を行います。

4）お客様の環境に応じた初期パラメータの設定が可能です。

5）基板に実装されたセンサーの座標軸とは異なる任意の座標軸を設定することが可能です。

本 FSA サンプルプログラム全体の構成を図 1-1 に示します。

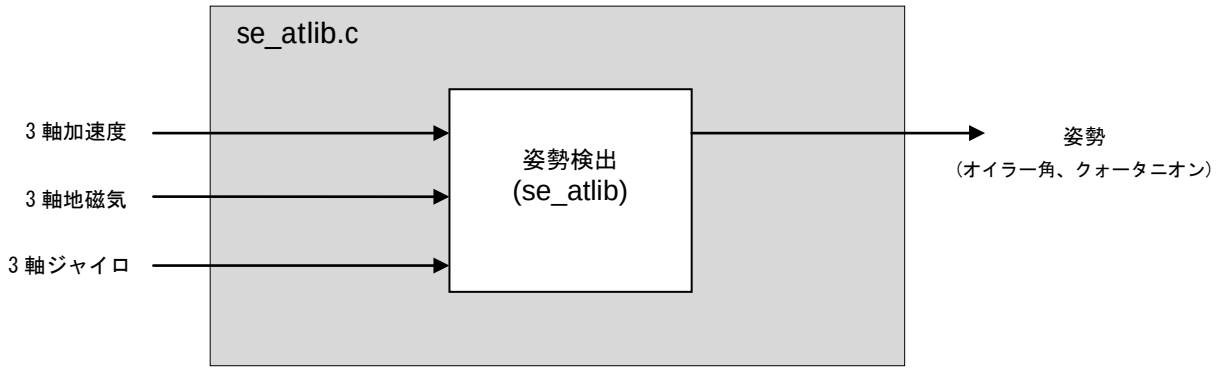


図 1-1　姿勢検出の FSA ライブラリ構成

以下に本サンプルプログラムがサポートする関数一覧を示します。

表 1-1　姿勢検出サンプルプログラムのサポート関数一覧

関数	概仕様
se_ATInitialize	姿勢検出に必要な初期化を行う関数
se_ATEstimate	姿勢検出を行う関数
se_ATCALInitialize	AT キャリブレーションに必要な初期化を行う関数
se_ATCALExecute	AT キャリブレーションを行う関数
se_ATSetOffset	ジャイロセンサーのオフセット値をセットする関数
se_ATGetOffset	ジャイロセンサーのオフセット値を取得する関数
se_ATSetAttitude	任意の座標軸を設定する関数
se_ATSetAttitudeParam	座標軸設定パラメータをセットする関数
se_ATGetAttitudeParam	座標軸設定パラメータを取得する関数

2. サンプルプログラム構成ファイル

姿勢検出サンプルプログラムは、以下のファイルで構成されています。

表 2-1 構成ファイル一覧

パッケージ名	ファイル	内容
atlib	se_atlib.o	姿勢検出ライブラリオブジェクト（MCU 処理部）
	se_atlib_fsa.o	姿勢検出ライブラリオブジェクト（FSA 処理部）
	se_atlib.h	姿勢検出ライブラリヘッダファイル
	se_symbols_atlib.def	リンク時に必要な定義ファイル

3. 構造体

以下に姿勢検出で用いる型定義一覧を示します。これらの型定義は se_atlib.h 内で定義されています。

AT_DATA 構造体

```
typedef struct {
    short           *psData;
    long            lR;
    long            lDOER;
    long            *plQuat;
    long            *plEuler;
} AT_DATA;
```

本サンプルプログラムに必要な情報をまとめた構造体です。各要素の詳細は以下の通りです。

psData
3 軸ジャイロセンサー、3 軸加速度センサー、3 軸地磁気センサーのセンサー値を配置する領域へのポインタを指定してください。但し、FSA がアクセス可能な領域を割り当てる必要があります。確保する領域の順序は、以下の通りです。

表 3-1 センサーデータ配列

オフセット	サイズ	説明
0x00	16bit	加速度センサーX 軸
0x02	16bit	加速度センサーY 軸
0x04	16bit	加速度センサーZ 軸
0x06	16bit	地磁気センサーX 軸
0x08	16bit	地磁気センサーY 軸
0x0a	16bit	地磁気センサーZ 軸
0x0c	16bit	ジャイロセンサーX 軸
0x0e	16bit	ジャイロセンサーY 軸
0x10	16bit	ジャイロセンサーZ 軸

lR
本サンプルプログラムでは使用しません。設定は不要です。

lDOER
本サンプルプログラムでは使用しません。設定は不要です。

plQuat
本サンプルプログラムで推定される姿勢をクォータニオンで保持する領域へのポインタです。本ポインタを参照することで、端末の姿勢を確認することができます。クォータニオンの固定小数点位置は 28 です。物理量を表す単位はありません。

表 3-2 姿勢データ配列（クォータニオン）

オフセット	サイズ	説明
0x00	32bit	クォータニオン q0
0x04	32bit	クォータニオン q1
0x08	32bit	クォータニオン q2
0x0c	32bit	クォータニオン q3

plEuler
本サンプルプログラムで推定される姿勢をオイラー角（ロール・ピッチ・ヨー角）で保持する領域へのポインタです。本ポインタを参照することで、端末の姿勢を確認することができます。NULL ポインタが割り当てられる場合、オイラー角は出力されません。オイラー角の単位は radian で、固定小数点位置は 29 です。

表 3-3 姿勢データ配列（オイラー角）

オフセット	サイズ	説明
0x00	32bit	ロール角
0x04	32bit	ピッチ角
0x08	32bit	ヨー角

4. マクロ定義

スケールファクタ値の設定

本サンプルプログラムではスケールファクタの設定値の算出を簡易化するために、以下のマクロ関数を定義しています。

表 3-1 スケールファクタ値の設定マクロ

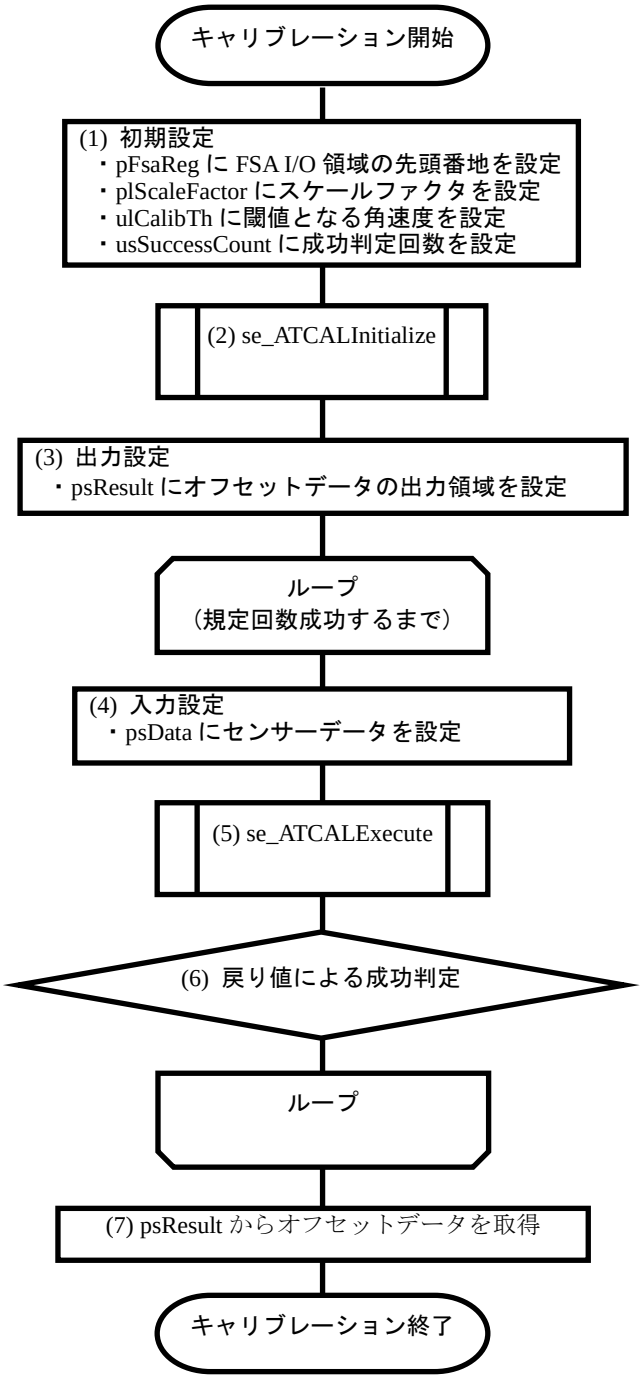
マクロ名	説明
SET_GYR_SF(sensitivity) sensitivity：ジャイロセンサー感度、単位は dps/LSB	se_ATInitilize関数でplScaleFactor[0]に設定するジャイロのスケールファクタ値を算出する [1]。
SET_ACC_SF(sensitivity) sensitivity：加速度センサーの感度、単位は G/LSB	se_ATInitilize 関数で plScaleFactor[1]に設定する加速度のスケールファクタ値を算出する。

[1] 例えば、14.375 LSB/dps のジャイロセンサーの場合、plScaleFactor[0] = SET_GYR_SF(1/14.375) とします。
同様に、256 LSB/G の加速度センサーの場合、plScaleFactor[1] = SET_ACC_SF(1/256)とします。

5. サンプルプログラムの制御

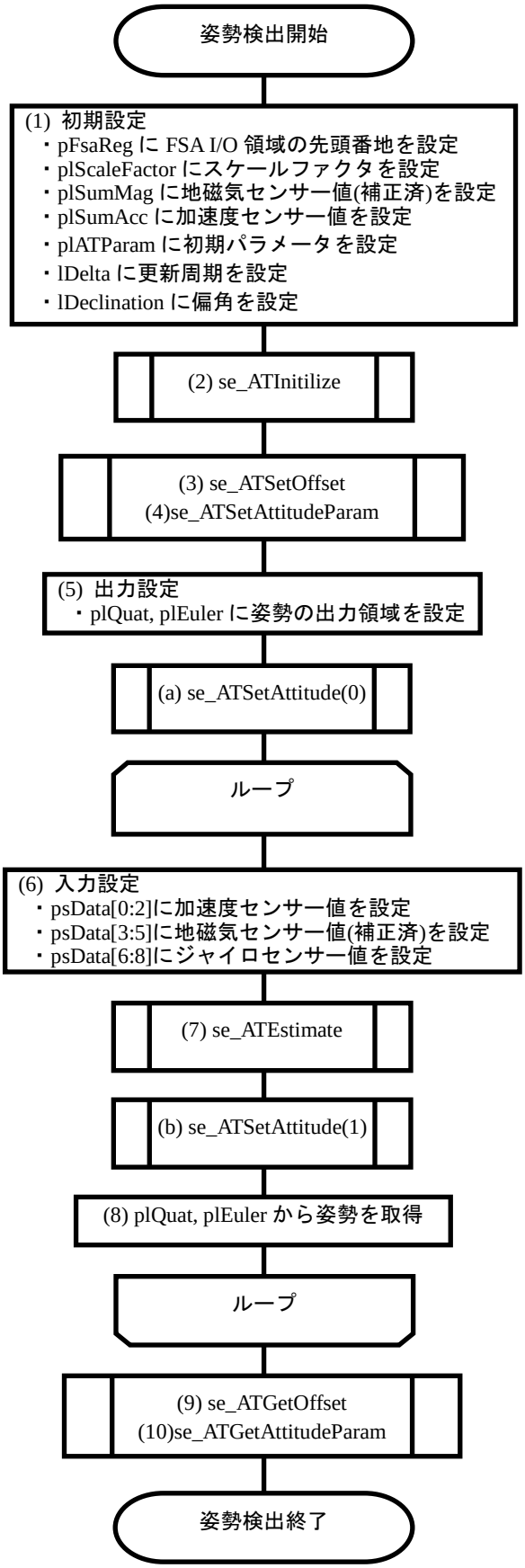
本サンプルプログラムのフ制御フローを、下図を使って説明します。

5.1. AT キャリブレーション



(1) キャリブレーションに必要な初期設定を行います。
(2) 初期化を行います。本関数はキャリブレーション実行時に初回のみ実行します。
(3) キャリブレーションの出力を設定します。オフセットデータの出力領域を確保します。
(4) キャリブレーションの入力を設定します。センサーデータを設定します。
(5) 上記(4)で設定された入力を元にキャリブレーションを行います。
(6) 上記(5)の戻り値からキャリブレーションが完了したかチェックします。
(7) 必要に応じて、上記(3)の出力領域からオフセットデータを取得してください。

5.2.姿勢検出


姿勢検出開始
(1) 初期設定
・pFsaReg に FSA I/O 領域の先頭番地を設定
・plScaleFactor にスケールファクタを設定
・plSumMag に地磁気センサー値(補正済)を設定
・plSumAcc に加速度センサー値を設定
・plATParam に初期パラメータを設定
・lDelta に更新周期を設定
・lDeclination に偏角を設定
(2) se_ATInitilize
(3) se_ATSetOffset
(4)se_ATSetAttitudeParam
(5) 出力設定
・plQuat, plEuler に姿勢の出力領域を設定
(a) se_ATSetAttitude(0)
ループ
(6) 入力設定
・psData[0:2]に加速度センサー値を設定
・psData[3:5]に地磁気センサー値(補正済)を設定
・psData[6:8]にジャイロセンサー値を設定
(7) se_ATEstimate
(b) se_ATSetAttitude(1)
(8) plQuat, plEuler から姿勢を取得
ループ
(9) se_ATGetOffset
(10)se_ATGetAttitudeParam
姿勢検出終了

(1) 初期設定を行います。地磁気データはキャリブレーションで補正した値を設定します。
(2) 初期化および初期姿勢の算出を行います。本関数は姿勢検出の実行時に初回のみ実行します。
(3) 前述の AT キャリブレーションで求めたオフセットデータをセットします。
(4) 必要に応じて、前回起動時に(10)で取得した座標軸設定パラメータをセットします。
(5) 姿勢検出の出力を設定します。姿勢の出力領域を必要に応じて確保します。
(6) 姿勢検出の入力を設定します。地磁気データは地磁気キャリブレーション済みの値を設定します。
(7) 上記(6)で設定された入力を元に姿勢検出を行います。
(8) 必要に応じて、上記(5)の出力領域から姿勢を取得してください。
(9) 必要に応じて、姿勢検出で更新されたオフセットデータを取得してください。
(10)必要に応じて、(a)および(b)で定義した座標軸設定パラメータを取得してください。

任意の座標軸を定義して姿勢検出を行う場合は、下記の処理を実施してください。

(a) 定義する座標軸上のロール角、ピッチ角、ヨー角のすべてが 0 度となる姿勢 A を記録します。
(b) 姿勢 A から、定義する座標軸上のピッチ角に相当する角度に、－30 度から－60 度程度変化させた姿勢 B を記録します。この姿勢 A から姿勢 B への変化を、定義する座標軸上における Y 軸を中心とした回転と見做して、最終的な座標軸が定義されます。

6. API 関数仕様

se_ATCALInitialize

インクルード
```
#include “se_atlib.h”
```

形式
```
int se_ATCALInitialize (FSAREG *pFsaReg,
                        long *plScaleFactor,
                        unsigned long ulCalibTh,
                        unsigned short usSuccessCount);
```

引数

pFsaReg	FSA の I/O 領域の先頭番地へのポインタ
plScaleFactor	ジャイロセンサーのスケールファクタへのポインタ
ulCalibTh	AT キャリブレーション閾値
usSuccessCount	AT キャリブレーション完了判定回数

戻り値

正常に終了した場合は 0、それ以外（入力が NULL ポインタなど）は-1 を返します。

説明

静止時誤差(オフセット)を補正するための AT キャリブレーションに必要な初期化を行います。AT キャリブレーションは、角速度の変動によって端末の静止状態を判定し、キャリブレーションを実施するため、*ulCalibTh* には静止状態を判定する閾値を角速度で設定してください。角速度の単位は dps で、固定小数点位置は 28 です。*usSuccessCount* にはキャリブレーションの完了を判断するための、静止判定回数を設定します。*plScaleFactor* にはジャイロセンサーのスケールファクタを設定します。

se_ATCALExecute

インクルード
```
#include “se_atlib.h”
```

形式
```
int se_ATCALExecute (short *psData,
                     short *psResult);
```

引数

psData	AT 構造体の *psData* へのポインタ
psResult	3 軸ジャイロセンサーのオフセット値へのポインタ

戻り値

キャリブレーションが成功した場合は 0、成功しなかった場合は-1 を返します。

説明

AT キャリブレーションを実施します。
psData に、AT 構造体の *psData* を 3 軸ジャイロセンサー値を設定した状態で割り当ててください。内部ではそこから角速度を算出し、se_ATCALInitialize 関数で指定した閾値と判定回数に従ってキャリブレーションを実施します。そしてキャリブレーションが成功した時点での平均値をジャイロセンサーの静止時オフセット値として *psResult* にセットします。これらの参照先は FSA がアクセス可能な領域に確保する必要があります。なお、本関数の出力は se_ATSetOffset 関数の入力とすることが可能です。

se_ATSetOffset

インクルード
```
#include “se_atlib.h”
```

形式　int se_ATSetOffset(short *psOffset)

引数

psOffset　ジャイロセンサーのオフセット値へのポインタ

戻り値

正常に終了した場合は 0、それ以外（入力が NULL ポインタなど）は-1 を返します。

説明

psOffset で与えられた値をジャイロセンサーのオフセット値としてセットします。
本関数の入力値は、se_ATCALExecute 関数、もしくは se_ATGetOffset 関数から得られた値をセットしてください。

se_ATGetOffset

インクルード
```
#include “se_atlib.h”
```

形式　int se_ATGetOffset (short *psOffset)

引数

psOffset　ジャイロセンサーのオフセット値へのポインタ

戻り値

正常に終了した場合は 0、それ以外（入力が NULL ポインタなど）は-1 を返します。

説明

se_ATEstimate 関数内で推定したジャイロセンサーのオフセット値を *psOffset* にセットします。
本関数の出力は se_ATGetOffset 関数の入力とすることができます。

se_ATSetAttitude

インクルード

```
#include “se_atlib.h”
```

形式　int se_ATSetAttitude (int iAttNum)

引数

iAttNum　記録する姿勢の番号（0：基準となる姿勢 A、1：所定動作後の姿勢 B）

戻り値

正常に終了した場合は 0、それ以外は-1 を返します。

説明

本関数は、任意の座標軸を基準に姿勢検出を行う際に、基準となる座標軸の設定を行うための関数です。具体的な手順は下記の通りです。

１）*iAttNum* を 0 として本関数をコールし、この時の姿勢 A を記録します。姿勢 A は、設定座標軸上のロール角、ピッチ角、ヨー角のすべてが 0 度の姿勢に相当します。

２）姿勢 A から、設定座標軸上のピッチ角に相当する角度に、－30 度から－60 度程度姿勢を変化させた状態で、*iAttNum* を 1 として本関数をコールし、この時の姿勢 B を記憶します。

上記１）、２）の操作により得られた姿勢 A から姿勢 B の変化を、設定座標軸上における Y 軸を中心とする回転と見做して、最終的な座標軸が設定されます。
本関数により座標軸を設定した以降は、se_ATEstimate関数で得られる姿勢は、設定された座標軸を基準とした値となります[2]。
座標軸の設定をやり直す場合は、上記の１）、２）の手順を再度実行してください。また、設定した座標軸を解除する場合は、se_ATInitialize 関数により初期化を実施してください。

[2] 座標軸軸設定パラメータの算出は se_ATSetAttitude 関数による座標軸の設定を受け、se_ATEstimate 関数で行われます。従って、se_ATGetAttitudeParam 関数で座標軸設定パラメータを取得する場合は、se_ATEstimate 関数をコールした後に取得してください。

se_ATSetAttitudeParam

インクルード
```
#include “se_atlib.h”
```

形式　`int se_ATSetAttitudeParam (char *pcParam)`

引数

pcParam	座標軸設定パラメータへのポインタ

戻り値
正常に終了した場合は 0、それ以外は-1 を返します。

説明
座標軸設定パラメータをセットします。
事前に se_ATGetAttitudeParam 関数で取得した座標軸設定パラメータを *pcParam* にセットし、本関数をコールすることで、se_ATSetAttitude 関数で設定した座標軸を復元します。なお、この座標軸設定パラメータの保持に必要な領域は 38Byte です。
本関数をコールする場合、se_ATInitialize 関数より後に実行してください。

復元後の初期姿勢について
se_ATSetAttitudeParam 関数コールする際の姿勢は任意です。本関数コールした後、復元した座標軸に合わせた姿勢が出力されます。但し、se_ATGetAttitudeParam 関数コール時と磁場環境が異なる場合、ヨー角方向に回転した姿勢が出力される可能性があります。

se_ATGetAttitudeParam

インクルード
```
#include “se_atlib.h”
```

形式　`int se_ATGetAttitudeParam (char *pcParam)`

引数

pcParam	座標軸設定パラメータへのポインタ

戻り値
正常に終了した場合は 0、それ以外は-1 を返します。

説明
座標軸設定パラメータを取得します。
本関数を実行することで *pcParam* に関数コール時の座標軸設定パラメータを取得します。取得したパラメータは se_ATSetAttitudeParam 関数からセットすることで、se_ATSetAttitude 関数で設定した座標軸を復元することが可能です。なお、この座標軸設定パラメータの保持に必要な領域は 38Byte です。

se_ATInitialize

インクルード

```
#include “se_atlib.h”
```

形式

```
int se_ATInitialize(
        void    *pFsaReg,
        long    *plSumAcc,
        long    *plSumMag,
        long    *plScaleFactor,
        long    *plATParam,
        long    lDeclination,
        long    lDelta
)
```

引数

pFsaReg	FSA の I/O 領域の先頭番地へのポインタ
plSumAcc	加速度センサー値へのポインタ
plSumMag	地磁気センサー値へのポインタ
plScaleFactor	ジャイロ・加速度センサーのスケールファクタ値へのポインタ
plATPalam	姿勢検出の初期パラメータへのポインタ
lDeclination	磁気偏角
lDelta	姿勢検出の更新周期

戻り値

正常に終了した場合は 0、それ以外（入力が NULL ポインタなど）は-1 を返します。

説明

姿勢の推定に必要な初期化をします。
pFsaReg はFSAのI/O領域の先頭番地を設定してください [3]。
plSumAcc および *plSumMag* はそれぞれ 3 軸加速度センサー値と 3 軸地磁気センサー値を設定します。但し、センサーのサンプリング時のノイズを抑えるために、16 サンプルの合計値を設定してください。また、FSA がアクセス可能な領域に確保する必要があります。
plScaleFactor は使用するセンサーのスケールファクタ値を設定します。前述のマクロ関数を使用して、ジャイロセンサー、加速度センサーの順でスケールファクタ値を設定してください。
plATParam は姿勢検出の初期パラメータを設定します。パラメータは使用するセンサーなどによって異なるため、環境に合わせたパラメータを設定してください [4]。
*lDeclination*は地磁気における偏角を設定します。偏角はdegree単位で、固定小数点位置を 23 として設定してください [5]。
*lDelta*は姿勢検出の更新周期を設定します。更新周期はsec単位で、固定小数点位置を 28 として設定してください [6]。
なお、本関数をコールせずに se_ATEstimate 関数をコールした場合の動作は保証されません。

[3] FSA の I/O 領域は機種によって異なります。詳細は各テクニカルマニュアルを参照してください。
[4] 設定するパラメータ(24Byte)の詳細については、弊社営業窓口までお問い合わせください。
[5] 例えば東京であれば現在の偏角は約 7 度であるため、 (long)(-7*(2^23)+0.5)を設定します。
[6] 例えば更新周期が 128Hz の場合、(long)((1/128)*(2^28)+0.5)となります。

se_ATEstimate

インクルード

```
#include “se_atlib.h”
```

形式　`int se_ATEstimate(AT_DATA *pstATData)`

引数

pstATData　AT_DATA 構造体のポインタ

戻り値

正常に終了した場合は 0、それ以外（入力が NULL ポインタなど）は-1 を返します。

説明

端末の姿勢を推定します。
サンプリングしたジャイロ・加速度・地磁気センサー値を、前述のセンサーデータ配列に従って、AT_DATA 構造体の *pstATData* に設定してください。
推定された姿勢はクォータニオンおよびオイラー角（ピッチ・ロール・ヨー角）で表現され、それぞれ AT_DATA 構造体の *plQuat* および *plEuler* に返します。但し、オイラー角を返す領域に NULL ポインタが割り当てられている場合、クォータニオンのみを出力します。
なお、se_ATSetAttitude 関数によって所定の座標軸が設定された場合、出力は設定された座標軸を基準とした姿勢となります。

なお、4G を超える衝撃や 0.5G 未満の自由落下状態となったとき、出力の姿勢が一瞬乱れる場合があります。

7. 要求メモリ

下表に要求メモリサイズをまとめます。入出力データ保持する領域は別途必要です。

表 6-1 要求メモリサイズ（byte）

メモリ種別	C17	C33	FSA	FSA スタック
ROM	9386	6206	5620	-
RAM	196	196	2796	900

(*1) FSA スタックメモリとは、FSA が一時的な作業領域として使用するメモリです。
具体的には、FSA がアクセス可能なメモリの最上位の領域が使用されます。

改訂履歴表

付－1

Rev. No.	日付	ページ	種別	改訂内容（旧内容を含む） および改訂理由
Rev.1.0	2015/04/03	-	新規	新規作成

セイコーエプソン株式会社
マイクロデバイス事業部　デバイス営業部

東京　〒191-8501　東京都日野市日野 421-8
TEL（042）587-5313（直通）　FAX（042）587-5116

大阪　〒541-0059　大阪市中央区博労町 3-5-1　エプソン大阪ビル 15F
TEL（06）6120-6000（代表）　FAX（06）6120-6100

ドキュメントコード：412957100
2015 年 4 月 作成