Easy 言語　文法説明書

2008.9.25　　1.0.2 版

本製品を正しくお使いいただくために、ご使用前には必ず
この説明書をよくお読みください。

CrestI　**株式会社クレスト**

FirstMyCom で用いられる Easy 言語の文法を説明します。

1. Easy ファイルの書き方

全て、1 バイトコードの ASCII 文字で記載します。 英大文字小文字は厳密に区別されます。英文字は、定義済みキーワードと文字列定数以外、原則的に英小文字をもちいます。フリーフォーマット記述が可能ですが、複数行に跨る文、1 行の中に複数の文を記述することは出来ません。

2. 変数と定数

2.1. 変数

変数は、予め組み込まれている 2 個の符号無し整数型(16 ビット)を使用します。変数を追加することはできません。

変数名は、英大文字の 1 文字で表され、A B とそれぞれ決められています。

2.2. 定数

2.2.1. 符号無し整数型

16 ビットの符号無し整数型が使えます。常に 10 進数で記述します。

2.2.2. 文字列型

1 バイトコードの ASCII 文字の文字列が使えます。文字列は、"(ダブルクオーテーション)で括ります。例えば、”Hello World” などと記述します。文字列内に"(ダブルクオーテーション)を含めることは出来ません。

2.2.3. 論理型

論理型は、文中の条件式（CONDITION）で用います。真 TRUE 、偽 FALSE が使えます。

3. 演算子

3.1. 算術演算子

加算 + ，減算 － ，掛け算 ＊ ，割り算 / が使えます。

１つの文の中に複数の算術演算子を使うことは出来ません。

= を用いて、左辺に値を代入できます。

3.2. 論理演算子

論理和 ｜ ，論理積 & ，排他的論理和 ^ ，２の補数(not) ~ が使えます。

１つの文の中に複数の論理演算子を使うことは出来ません。

= を用いて、左辺に値を代入できます。

3.3. 条件演算子

等価 == ， 不等号 > ， >=, <, <=, が使えます。

１つの文の中に複数の条件演算子を使うことは出来ません。

4. 文

4.1. 宣言

4.1.1. conf 文

構文

```
conf [KEYWORD]
```

説明

FirstMyCon の実行時に適用する周辺デバイスの構成を定義します。必ず他の文より先に記述します。

[KEYWORD] は、次のものから選びます。

RS232C 、 ADC 、 PWM 、 TIMER 、 DIN、DOUT

4.2. 制御構造

4.2.1. if...then...else 文

構文

```
if CONDITION1
then
        STATEMENT-BLOCK1
[elseif CONDITION2
  then
        STATEMENT-BLOCK2]
....
[else
        STATEMENT-BLOCK3]
endif
```

説明

CONDITION1 が真の時は、then 以降の STATEMENT-BLOCK1 を 実行し、CONDITION1 が偽の時で CONDITION2 が真の時は、 STATEMENT-BLOCK2 を実行します。 CONDITION1 が偽で、CONDITION2 も偽の時は、else の後の STATEMENT-BLOCK3 を実行します。 elseif のブロックは、省略可で、複数記述できます。 else のブロックは、省略または、 一つだけ記述します。

4.2.2. while...wend 文

構文

```
while [ CONDITION ]
      STATEMENT-BLOCK
wend
```

説明

CONDITION が真である間、STATEMENT-BLOCK を実行します。 CONDITION は、STATEMENT-BLOCK の前に評価さます。while... wend 文を入れ子にすることは出来ません。CONDITION は、唯一のキーワード TRUE を記述するか、省略します。省略した場合は、CONDITION が真と解釈されます。これにより、while...wend 文は、STATEMENT-BLOCK を無条件ループで実行します。

4.2.3. break 文

構文

```
break
```

説明

while... wend 文の中で使用して、無条件にループを脱出します。

4.2.4. end 文

構文

```
end
```

説明

プログラムの実行を終了します。

4.3. 標準入出力

4.3.1. print 文

構文

print 変数、または、定数 ［ , 出力行 ］

説明

指定された変数、または定数の内容を LCD に表示します。 文字列定数は、″(ダブルクオーテーション)で囲まれている部分を表示します。最大の
出力可能な文字数は 16 文字です。
第二引数にて、出力する行を指定できます。1、または、2 を指定します。省略した場合は、1 とみなされます。

4.3.2. led 文

構文

led 変数、または、定数（符号無し整数型のみ）

説明

LED を点灯または消灯します。指定された変数、または定数の値が、0、及び、偶数で消灯、奇数で点灯します。

4.3.3. input 文

構文

input 変数

説明

conf 文で定義されたデータ入力可能な周辺デバイスから値を入力します。
データ入力可能な周辺デバイスには、次のものがあります。
RS232C 、 ADC 、 DIN、TIMER これらは、排他定義となっています。

[RS232C] 1 バイトのシリアル入力データを取得します。データ受信するまで処理は停止します。
[ADC] Vref=1.5V で、範囲 FSR=±0.75V のアナログデータを AD 変換した結果を取得します。変換式は次の通りです。

$$Nadc = 1023 \times \frac{Vin - Vr-}{Vr+ - Vr-}$$

[DIN] 1 または 0 を入力します。 1 ： OFF = 回路 OPEN 0 ： ON = 回路 SHORT
[TIMER] 1 MHz でフリーランしている 16 ビットカウンタのタイマ値を読み出します。

4.3.4. output 文

構文

output 変数、または、定数（符号無し整数型のみ）

説明

conf 文で定義されたデータ出力可能な周辺デバイスから値を出力します。
データ入力可能な周辺デバイスには、次のものがあります。
PWM 、 DOUT これらと RS232C は、排他定義となっています。

[PWM] Pulse Width Modulation 出力を開始します。1 ミリ秒周期のパルス幅を 1024 等分した PWM duty cycle を出力値として指定します。有効な指定範囲は 0～1023 となります。
[DOUT] 出力値のビット 0 の 1 または 0 を出力します。 1 = 3.3V 0 = 0V の電圧が出力されます。

制限

RS232C の出力は未サポートです。

4.4. その他

4.4.1. // 文

構文

// [....]

説明

// から改行までは、注釈として無視されます。

5. 関数

5.1. 遅延　sleep 文

構文

sleep 変数、または、定数（符号無し整数型のみ））

説明

指定された値の時間（ミリ秒）の間、プログラムの実行を遅延します。

６．プログラミング例

6.1. DIN 入力、LCD 表示、DOUT 反転出力

1 秒毎に DIN から入力したデジタル値を LCD 表示に表示し、DOUT へ反転出力します。
16 回でプログラムは終了します。

```
conf  DIN
conf  DOUT
print "Digital in->out"
A = 0
while
    input B
    print B, 2
    B = ~ B
    output B
    A = A + 1
    if A <= 16 then
        break
    endif
    sleep( 1000 )
wend
end
```

6.2. RS232C 入力、LCD 表示

RS232C から入力したデータを LCD 表示に表示します。1 バイト毎に 10 進数で表示します。
無限ループとなりプログラムは終了しません。

```
conf  RS232C
print "Serial input"
while
     input A
     print A, 2
wend
end
```