

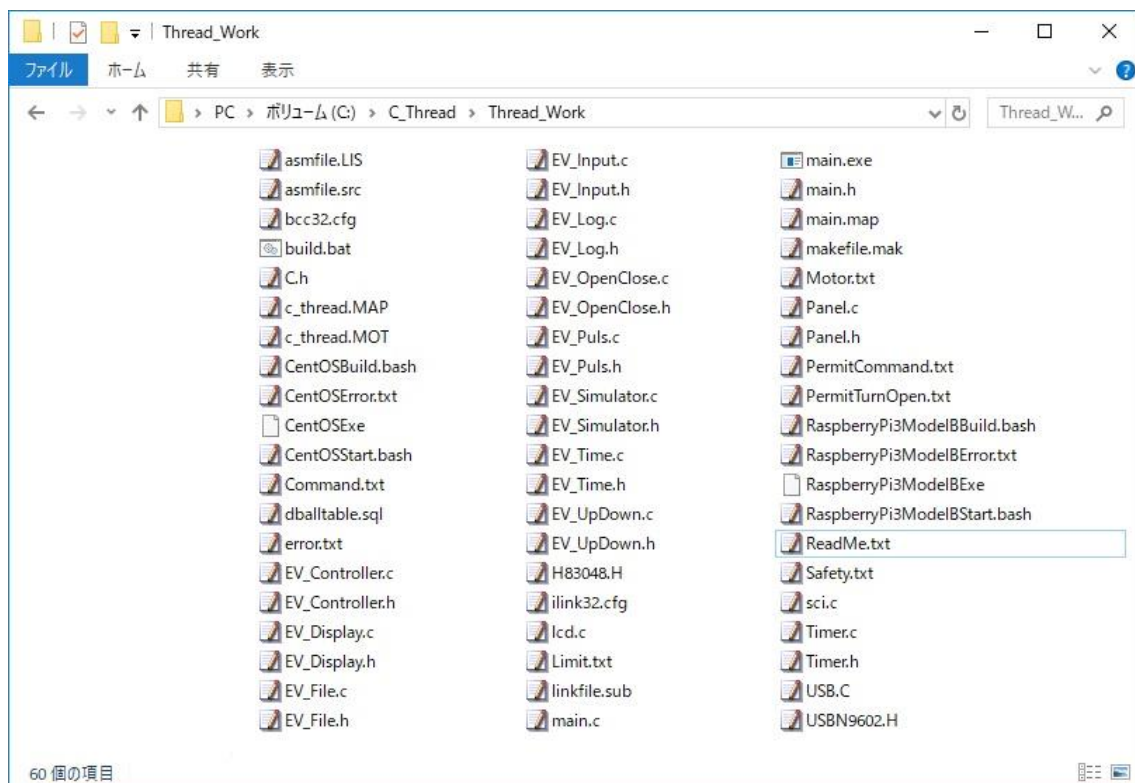
発明の巻

C 言語の疑似スレッド最新版(ライセンスフリー) の
サポートパック(324,000 円)

スレッドとは、コンピュータープログラミング上の、
並列処理の機能です。スレッドをサポートしているプ
ログラミング言語と、スレッドをサポートしていない
プログラミング言語があります。高度な機能のスレッ
ドをわざとサポートしない言語があるのは、パソコン
以外の環境で実行するために、低レベルマイコン対応
のプログラミング言語でいるから(例えば C 言語)で
す。代わりにマイコンにはインターバルタイマがあり
ます。今回、マイコンのインターバルタイマを使用し
て、並行処理タイプのスレッド(疑似スレッドと命名)
を作成しました。C 言語のスレッドです。低レベル環
境に移植可能です。(16bit 以上)サンプルに、エレベ
ーターのプログラムが入っています。「コントローラ」
「シミュレータ」の 2 つのプログラムを並行処理技術
により連携しながら同時実行するように開発しまし
た。プロセスが 1 個でスレッド(プログラム)が複数で
動かします。LINUX よりも前の世代のマイコンはプロ

セスが1個しか入らないでしょうから。C, Assembly, Batch Shell Script 使用。2018年5月8日版プログラムです。初期デバッグ対応します。日本語で、日本国内で、初期サポート対応可能です。改造は自己責任でお願いします。324,000円の中身は初期サポート代です。サンプルコードは開発終了品です。購入後、ご自由に、使用・改造・配布、O.K.です。

お問い合わせ先：info@hidemine.ciao.jp



発明アルゴリズムに疑似スレッドと名付けました。本物に偽物と名付けました。特許申請はしないで、無料公開をいたしました。発明という手段で人類としてのノルマを達成し、また、生きた証が発生いたしました。篠宮氏を逆賊に祭り上げようとする人種がいます。篠宮氏は辞退いたします。秋篠宮家様は本物の皇族ですが、篠宮氏は偽物です。

タイトル(title):

C 言語の疑似スレッド

サブタイトル(subtitle):

C 言語の偽物のスレッド

The thread at the imitation of the C language.

似ているが独創的な別物

The resembling but original singleton.

C 言語によるスレッドという概念の模倣

Copying a concept,

the thread, by the C language.

解説

C言語のプロジェクト Thread について、

このプロジェクトは予告なくデバッグ目的で更新されることがあります。

このプロジェクトは、お客様が改造して、よりお客様に使いやすいプログラムにするために、提供されるサンプルプログラムであり、お客様の好みに完成させてください。

=====
このプロジェクトは、4通りの方法でコンパイル 実行 できます。

1つ目の方法：

Windows版 Borland BCC C/C++ のダウンロードとインストール

bcc コンパイラー で検索します

C++Builder のホームページを開きます

C++ Compiler 5.5 / Turbo Debugger (日本語) (コンパイラのみで軽い)

(テキストエディタと組み合わせて使用します)(フリーソフト)

(ユーザー登録が必要)(メールアドレスが必要)をダウンロードして

解凍します

freecommandlinetools2.exeをインストールします

freeturbodebugger.exeをインストールします

makefile.mak build.bat は

複数のファイルを1個のプロジェクトとしてコンパイルするための

ファイルです

error.txt はコンパイルエラーを表示するファイルです

main.exe をダブルクリックすると BCC実行ソフト (このソフト) が
起動します

2つ目の方法：

Windows版 AKI-H8 3052F USB開発セット の購入

メーカー：ルネサスエレクトロニクスさん

販売者：秋月電子通商さん

通販コード：K-00182

商品名：AKI-H8 3052F USB開発セット

商品価格：税込7,000円 (2018年3月6日現在)

AKI-H8 3052F USB開発セット で検索します

AKI-H8 3052F USB開発セット を 秋月電子通商さん の通販サイトで購入
するか、それとも ご自宅の最寄りの電子パーツ専門店 で注文購入します

AC/DCコンバータ(ACアダプター) と、RS-232Cケーブル と、

延長USBコード を、ご自宅の最寄りの電子パーツ専門店 で注文購入
します

AC/DCコンバータ の電流電圧 は お店の 人 に聞いてください

RS-232C は、パソコンに端子がある必要があり、また、パソコンと

AKI-H8基板 の両方のコネクタ のオス端子・メス端子 を確認して

購入してください

延長USBコード も オス端子・メス端子 を確認してください

カラー短絡ソケット6mm 青 2228CG-BU で検索します

カラー短絡ソケット6mm 青 2228CG-BU のような 短絡コネクタ を、
ご自宅の最寄りの電子パーツ専門店で2個 注文購入します
(100円か200円支払うと何個かまとめて購入できます)

USB開発セットのマニュアルに従い、usbフォルダのmain.cを
コンパイルして、H8WriteTurboをインストールして、AKI-H8基板
への書き込み時に、短絡ソケット(短絡コネクタ)を使用して、
マニュアルに従いモードを調節して、usbフォルダのusbtest.MOT
をAKI-H8基板へ書き込み、走らせてみます
押しボタンを押すと、sw1 sw2 sw3 sw4などと液晶パネルに
表示されれば大丈夫です。

C_Threadフォルダの中の、Thread_Workフォルダの中の、main.c
を、コンパイルします(build.bat を、編集で中身を確認後、
ダブルクリックします)

C_Threadフォルダの中の、Thread_Workフォルダの中の、c_thread.MOT
を、モードを調節しながら、AKI-H8基板へ書き込み、走らせてみます
液晶パネル・押しボタン・LEDなどが機能していれば、成功です

asmfile.src linkfile.sub build.bat は
複数のファイルを1個のプロジェクトとしてコンパイルするための
ファイルです

Puls.h Puls.c の記述により、
PortB bit 0 1 2 3 4 5 (AKI-H8-CN1 pin 16 17 18 19 20) から、
エレベーターのモーター指令信号の出力(OUTPUT)があります

error.txt はコンパイルエラーを表示するファイルです

3つ目の方法：

LINUX CentOS6 GCC の利用

LINUX の GCC を使用するときには、文字コードを utf8 にして
改行コードを <LF>のみ(UNIX) にして名前を付けて保存してください

データベース MySQL を使用しておりますので、dballtable.sql
を解析して、MySQL の準備をしてから、挑戦してみてください

MySQL をインストールしたら、所定の場所に、<mysql.h> があると
思います

CentOSBuild.bash は

複数のファイルを1個のプロジェクトとしてコンパイルするための
ファイルです

CentOSError.txt はコンパイルエラーを表示するファイルです

LINUX CentOS6 で CentOSStart.bash をダブルクリックすると、
GCC実行ソフト CentOSExe (このソフト) が起動します

私は、CentOS にしましたが、CentOSBuild.bash CentOSStart.bash
が読める方は、お好きな LINUX で挑戦してみてください

4つ目の方法：

LINUX Raspberry Pi 3 Model B (ラズベリーパイ) Raspbian GCC の利用

LINUX の GCC を使用するときは、文字コードを utf8 にして
改行コードを <LF>のみ(UNIX) にして名前を付けて保存してください

データベース MySQL を使用しておりますので、dballtable.sql
を解析して、MySQL の準備をしてから、挑戦してみてください

以下のコードを試して、<mysql.h> をインストールしてください

```
sudo apt-get install libmariadbclient-dev
```

RaspberryPi3ModelBBuild.bash は

複数のファイルを1個のプロジェクトとしてコンパイルするための
ファイルです

RaspberryPi3ModelBError.txt はコンパイルエラーを表示するファイルです

LINUX Raspbian で RaspberryPi3ModelBStart.bash

をダブルクリックすると、GCC実行ソフト RaspberryPi3ModelBExe
(このソフト) が起動します

Puls.h Puls.c の記述により、

GPIO 16 17 18 19 20 (pin 36 11 12 35 38) から、

エレベーターのモーター指令信号の出力(OUTPUT) があります

=====

asmfile.src について

リセットベクトルの 転送先ラベル が `_start` になっています

ずっと下の方の `_start` の ラベル から処理を開始して、

`jsr @_main` で C言語の関数 `main` を呼び出しています

C言語の関数 `main` は、 `void main(void);` という形で、

`main.c` に記述があります

その後、 `int_error: rte` で `rte` (`return`と同じ意味) で終了しています

リセットベクトルに続く1番から60番までの 割り込みベクトル について、

使用しない 割り込みベクトル は ラベル `int_error` に転送されます

`;26 OVIO _INT_OVIO: .DATA.L _ITU_OVI_0 ;タイマ0割り込み`

で、 タイマ0割り込み は、 ラベル `_ITU_OVI_0` に転送されます

ラベル `_ITU_OVI_0` から開始して、 スタック 退避をして、

`jsr @_InterruptITU0` で、 C言語の関数 `InterruptITU0`

を呼び出しています

C言語の関数 `InterruptITU0` は `void InterruptITU0(void);`

という形で、 `Timer.h` `Timer.c` に記述があります

戻ってくると、再び スタック を戻して、 `rte` です

ファイルの先頭に、

`.IMPORT _main`

`.IMPORT _InterruptITU0`

という記述があり、 C言語の関数 を参照しています

.EXPORT _EnableInterrupt,_DisableInterrupt

_EnableInterrupt: andc.b #H'3f,ccr rts

_DisableInterrupt: orc.b #H'c0,ccr rts

で、C言語から

_EnableInterrupt (割り込み許可)

_DisableInterrupt (割り込み禁止)

を呼び出せるようにしています

C言語の Panel.h に 外部参照プロトタイプ宣言 があります

```
extern void EnableInterrupt(void);
```

```
extern void DisableInterrupt(void);
```

C言語からの呼び出し名は、

```
EnableInterrupt();
```

```
DisableInterrupt();
```

です

=====
main.c の 関数 Run の ID==31 を見てください

Thread Ready GO! で開始して競馬のコースが8コースあります

Thread Ready GO! There are 8 cources on a race.

ゴールまで14歩です

There are 14 cells to a GOAL.

<1>コースは'r'ボタンが鞭で<2>コースは'l'ボタンが鞭です

For the <1> course, You click a 'R' button.

For the <2> course, You click a 'L' button.

スレッドを使用しています

Thread *th[8]; でオブジェクト宣言しています

th[i] = new Thread(i + 1); で初期値設定しています

この2行は Java で次と同じ意味です

```
Thread th[] = new Thread[8];
```

```
th[i] = new Thread(i + 1);
```

```
void Repaint(void)
```

```
{
```

```
    ...
```

```
}
```

```
void Run(Thread *This)
```

```
{
```

```
    ...
```

```
}
```

```
void Init(Thread *This)
```

```
{
```

```
    ...
```

```
}
```

```
void Destroy(Thread *This)
```

```
{
```

```
    ...
```

```
}
```

はそれぞれ Java で次と同じ意味です

```
public void paint(Graphics g)
```

```
{
```

```
    ...
```

```
}
```

```
public void run()
```

```
{
```

```
    ...  
}  
public void init()  
{  
    ...  
}  
public void destroy()  
{  
    ...  
}
```

`delete_(th[i]);` でオブジェクトを消去しています

この1行は C++ で次と同じ意味です

```
delete th[i];
```

スレッド20が走り始めたら、

0以外の数字キーを押してみてください

その数字に20を加えた番号のスレッドが、キーを押す度毎に、

起動・消去を繰り返します

20を含めて、全部スレッドが消去されると、終了です

これらのスレッドに関する仕様は Timer.c に記述しました

=====

2階建エレベーターEVについて

使用方法

EV_Simulator にエレベーターが表示されます

EV_Controller にエレベーターの動作が表示されます

EV_Input の使用方法

u キーを押すとエレベーターが2階に上昇して扉が開きます

d キーを押すとエレベーターが1階に下降して扉が開きます

o キーを押すと扉が開きます

c キーを押すと扉が閉じます

s キーを押すと籠が非常停止します

r キーを押すと籠が非常停止から復帰します

Y キーを押すとエレベーターが2階に上昇して扉が開きます

H キーを押すと2階で扉が閉じます

y キーを押すとエレベーターが1階に下降して扉が開きます

h キーを押すと1階で扉が閉じます

籠が無い階で H h キーを押しても籠は動作しません

籠が無い階で Y y キーを押したとき籠の扉が開いていると、

籠は動作しません

開いた状態の扉は一定時間後自動で閉じます

EV_Time.h に #define OPENTIMEOUT 10 と書いてあるので 10秒 です

閉まりかけの時に開く動作をするキーを押すと扉が反転して開きます

動作説明

全体の動作説明

モーターの情報は Motor.txt にあります

エレベーター塔内のリミットスイッチの情報は Limit.txt にあります

EV_Simulator はエレベーターの次の位置を出力していて Safety.txt

Motor.txt Limit.txt を採取して Safety.txt Limit.txt に書き込んで

エレベーターの画面表示もしています

EV_Controller はエレベータを制御していて Command.txt Limit.txt

を採取して PermitCommand.txt Motor.txt に書き込んでいます

EV_Controllerの動作説明

エレベーターには現在位置情報(Limit.txt)があります

最簡形の2階建ての場合通常系には5個の位置状態があります

下の階の停止状態

下の階の低速区域

中間の高速区域

上の階の低速区域

上の階の停止状態

5個の区域の境界に合計4個のセンサーがあります

4個のセンサーがエレベーターの現在位置を取得しています

4個のセンサーからの信号はメンバ変数(Positionクラスの

*p_UnderSlow *p_UnderStop *p_UpperSlow *p_UpperStop)

に読み込みます

昇りのメソッド(UpMotorクラスのOnUpMotor)と降りのメソッド

(DownMotorクラスのOnDownMotor)を使って

モーターに出力(Motor.txtに出力)します

全く同じ様にドアも通常系で4個のセンサーがありドアの開閉ではエレベーターの昇降と全く同じクラス構造です

後はインスタンス(Position P UpMotor UPMT DownMotor DNMT Door DR OpenMotor OPMT CloseMotor CLMT)

を宣言して仕様に合わせてメソッドを呼び出すだけでO.K.です

終了方法

エレベーターが通常停止しているときに q キーを押します

メンテナンス

異常終了した場合、終了後、Thread_Work フォルダの次のファイルをチェックしてください

Safety.txt

Safety.txt を開いて r にして上書き保存してください

r は通常動作を意味します

s は非常停止を意味します

h は復帰を意味します

Y はスターデルタのスター起動を意味します

Command.txt

Command.txt を開いて q にして上書き保存してください

q は終了を意味します

u は上昇を意味します

d は下降を意味します

o は開を意味します

c は閉を意味します

Y は上階呼びを意味します

y は下階呼びを意味します

H は上階閉を意味します

h は下階閉を意味します

N は信号無しを意味します

PermitCommand.txt

PermitCommand.txt を開いて c にして上書き保存してください

N は命令入力禁止を意味します

c は命令入力許可を意味します

PermitTurnOpen.txt

PermitTurnOpen.txtを開いて N にして上書き保存してください

N は反転開信号入力禁止を意味します

o は反転開信号入力許可を意味します

Motor.txt

Motor.txt を開いて s にして上書き保存してください

s はモーター停止を意味します

j はモーター上昇回転開始を意味します

u はモーター低速上昇回転を意味します

U はモーター高速上昇回転を意味します

k はモーター下降回転開始を意味します

d はモーター低速下降回転を意味します

D はモーター高速下降回転を意味します

h はモーター開回転開始を意味します

o はモーター低速開回転を意味します

O はモーター高速開回転を意味します

t はモーター閉回転開始を意味します

c はモーター低速閉回転を意味します

C はモーター高速閉回転を意味します

Limit.txt

Limit.txt を開いて ynnnyynn にして上書き保存してください

ynnnyynn は籠が下階停止状態で扉が閉停止状態を意味します

ynnnnyynn は籠が下階停止状態で扉が閉低速区域を意味します

ynnnnnnyynn は籠が下階停止状態で扉が中間高速区域を意味します

ynnnnnyynn は籠が下階停止状態で扉が開低速区域を意味します

ynnnnnyyynn は籠が下階停止状態で扉が開停止状態を意味します

nynnyynn は籠が下階低速区域で扉が閉停止状態を意味します

nnnnyynn は籠が中間高速区域で扉が閉停止状態を意味します

nnnyynn は籠が上階低速区域で扉が閉停止状態を意味します

nnyyyynn は籠が上階停止状態で扉が閉停止状態を意味します

nnyynnyynn は籠が上階停止状態で扉が閉低速区域を意味します

nnyynnnnyynn は籠が上階停止状態で扉が中間高速区域を意味します

nnyynnyynn は籠が上階停止状態で扉が開低速区域を意味します

nnyynnyyynn は籠が上階停止状態で扉が開停止状態を意味します

参照ライブラリ

```

/*****
/*      H8/3048F Include File                               */
/*****

struct st_sam {
    void          *MAR;          /* MAR          */
    unsigned int  ETCR;         /* ETCR         */
    unsigned char IOAR;         /* IOAR         */
    unsigned char DTCR;         /* DTCR         */
};

struct st_fam {
    void          *MARA;         /* MARA         */
    unsigned int  ETCRA;        /* ETCRA        */
    unsigned char IOARA;        /* IOAR         */
    unsigned char DTCRA;        /* DTCRA        */
    void          *MARB;         /* MARB         */
    unsigned int  ETCRB;        /* ETCRB        */
    unsigned char IOARB;        /* IOAR         */
    unsigned char DTCRB;        /* DTCRB        */
};

struct st_itu {
    unsigned char TSTR;         /* TSTR         */
    unsigned char TSNC;         /* TSNC         */
    unsigned char TMDR;         /* TMDR         */
    unsigned char TFCR;         /* TFCR         */
    char          wk[44];       /*              */
    unsigned char TOER;         /* TOER         */
    unsigned char TOCR;         /* TOCR         */
};

struct st_itu0 {
    unsigned char TCR;          /* TCR          */
    unsigned char TIOR;         /* TIOR         */
    unsigned char TIER;         /* TIER         */
    unsigned char TSR;          /* TSR          */
    unsigned int  TCNT;         /* TCNT         */
    unsigned int  GRA;          /* GRA          */
    unsigned int  GRB;          /* GRB          */
};

struct st_itu3 {
    unsigned char TCR;          /* TCR          */
    unsigned char TIOR;         /* TIOR         */
    unsigned char TIER;         /* TIER         */
    unsigned char TSR;          /* TSR          */
    unsigned int  TCNT;         /* TCNT         */
    unsigned int  GRA;          /* GRA          */
    unsigned int  GRB;          /* GRB          */
    unsigned int  BRA;          /* BRA          */
    unsigned int  BRB;          /* BRB          */
    char          wk[2];        /*              */
};

struct st_tpc {
    unsigned char TPMR;         /* TPMR         */
    unsigned char TPCR;         /* TPCR         */
};

```

```

unsigned char  NDERB;      /* NDERB      */
unsigned char  NDERA;      /* NDERA      */
unsigned char  NDRB1;     /* NDRB (H'A4) */
unsigned char  NDRA1;     /* NDRA (H'A5) */
unsigned char  NDRB2;     /* NDRB (H'A6) */
unsigned char  NDRA2;     /* NDRA (H'A7) */
};

struct st_rfsbc {          /* struct RFSHC */
    unsigned char  RFSHCR; /* RFSHCR      */
    unsigned char  RTMCSR; /* RTMCSR      */
    unsigned char  RTCNT;  /* RTCNT       */
    unsigned char  RTCOR;  /* RTCOR       */
};

struct st_sci {           /* struct SCI   */
    unsigned char  SMR;    /* SMR         */
    unsigned char  BRR;    /* BRR         */
    unsigned char  SCR;    /* SCR         */
    unsigned char  TDR;    /* TDR         */
    unsigned char  SSR;    /* SSR         */
    unsigned char  RDR;    /* RDR         */
    char          wk[2];   /*             */
};

struct st_p1 {           /* struct P1    */
    unsigned char  DDR;    /* P1DDR      */
    char          wk;      /*             */
    unsigned char  DR;     /* P1DR       */
};

struct st_p2 {           /* struct P2    */
    unsigned char  DDR;    /* P2DDR      */
    char          wk1;     /*             */
    unsigned char  DR;     /* P2DR       */
    char          wk2[20]; /*             */
    unsigned char  PCR;    /* P2PCR      */
};

struct st_p4 {           /* struct P4    */
    unsigned char  DDR;    /* P4DDR      */
    char          wk1;     /*             */
    unsigned char  DR;     /* P4DR       */
    char          wk2[18]; /*             */
    unsigned char  PCR;    /* P4PCR      */
};

struct st_p5 {           /* struct P5    */
    unsigned char  DDR;    /* P5DDR      */
    char          wk1;     /*             */
    unsigned char  DR;     /* P5DR       */
    char          wk2[16]; /*             */
    unsigned char  PCR;    /* P5PCR      */
};

struct st_p6 {           /* struct P6    */
    unsigned char  DDR;    /* P6DDR      */

```

```

char          wk;          /* */
unsigned char DR;         /* P6DR */
};

struct st_p7 {             /* struct P7 */
    unsigned char DR;     /* P7DR */
};

struct st_p8 {             /* struct P8 */
    unsigned char DDR;    /* P8DDR */
    char          wk;     /* */
    unsigned char DR;     /* P8DR */
};

struct st_p9 {             /* struct P9 */
    unsigned char DDR;    /* P9DDR */
    char          wk;     /* */
    unsigned char DR;     /* P9DR */
};

struct st_da {             /* struct D/A */
    unsigned char STCR;   /* DASTCR */
    char          wk[127]; /* */
    unsigned char DR0;    /* DADR0 */
    unsigned char DR1;    /* DADR1 */
    unsigned char CR;     /* DACR */
};

struct st_ad {             /* struct A/D */
    unsigned int  DRA;    /* ADDRA */
    unsigned int  DRB;    /* ADDRb */
    unsigned int  DRC;    /* ADDRc */
    unsigned int  DRD;    /* ADDRd */
    unsigned char CSR;    /* ADCSR */
    unsigned char CR;     /* ADCR */
};

struct st_bsc {            /* struct BSC */
    unsigned char CSCR;   /* CSCR */
    char          wk1[140]; /* */
    unsigned char ABWCR;  /* ABWCR */
    unsigned char ASTCR;  /* ASTCR */
    unsigned char WCR;    /* WCR */
    unsigned char WCER;   /* WCER */
    char          wk2[3];  /* */
    unsigned char BRCR;   /* BRCR */
};

struct st_intc {           /* struct INTC */
    unsigned char ISCR;   /* ISCR */
    unsigned char IER;    /* IER */
    unsigned char ISR;    /* ISR */
    char          wk;     /* */
    unsigned char IPRA;   /* IPRA */
    unsigned char IPRB;   /* IPRB */
};

```



```

#define DMAC0A (*(volatile struct st_sam *)0xFFFF20) /* DMAC 0A Addr */
#define DMAC0B (*(volatile struct st_sam *)0xFFFF28) /* DMAC 0B Addr */
#define DMAC1A (*(volatile struct st_sam *)0xFFFF30) /* DMAC 1A Addr */
#define DMAC1B (*(volatile struct st_sam *)0xFFFF38) /* DMAC 1B Addr */
#define DMAC0 (*(volatile struct st_fam *)0xFFFF20) /* DMAC 0 Addr */
#define DMAC1 (*(volatile struct st_fam *)0xFFFF30) /* DMAC 1 Addr */
#define ITU (*(volatile struct st_itu *)0xFFFF60) /* ITU Address*/
#define ITU0 (*(volatile struct st_itu0 *)0xFFFF64) /* ITU0 Address*/
#define ITU1 (*(volatile struct st_itu0 *)0xFFFF6E) /* ITU1 Address*/
#define ITU2 (*(volatile struct st_itu0 *)0xFFFF78) /* ITU2 Address*/
#define ITU3 (*(volatile struct st_itu3 *)0xFFFF82) /* ITU3 Address*/
#define ITU4 (*(volatile struct st_itu3 *)0xFFFF92) /* ITU4 Address*/
#define TPC (*(volatile struct st_tpc *)0xFFFFA0) /* TPC Address*/
#define RFSHC (*(volatile struct st_rfshc *)0xFFFFAC) /* RFSHC Address*/
#define SCI0 (*(volatile struct st_sci *)0xFFFFB0) /* SCI0 Address*/
#define SCI1 (*(volatile struct st_sci *)0xFFFFB8) /* SCI1 Address*/
#define P1 (*(volatile struct st_p1 *)0xFFFFC0) /* P1 Address*/
#define P2 (*(volatile struct st_p2 *)0xFFFFC1) /* P2 Address*/
#define P3 (*(volatile struct st_p1 *)0xFFFFC4) /* P3 Address*/
#define P4 (*(volatile struct st_p4 *)0xFFFFC5) /* P4 Address*/
#define P5 (*(volatile struct st_p5 *)0xFFFFC8) /* P5 Address*/
#define P6 (*(volatile struct st_p6 *)0xFFFFC9) /* P6 Address*/
#define P7 (*(volatile struct st_p7 *)0xFFFFCE) /* P7 Address*/
#define P8 (*(volatile struct st_p8 *)0xFFFFCD) /* P8 Address*/
#define P9 (*(volatile struct st_p9 *)0xFFFFD0) /* P9 Address*/
#define PA (*(volatile struct st_p1 *)0xFFFFD1) /* PA Address*/
#define PB (*(volatile struct st_p1 *)0xFFFFD4) /* PB Address*/
#define DA (*(volatile struct st_da *)0xFFFF5C) /* D/A Address*/
#define AD (*(volatile struct st_ad *)0xFFFFE0) /* A/D Address*/
#define BSC (*(volatile struct st_bsc *)0xFFFF5F) /* BSC Address*/
#define FLMCR (*(volatile unsigned char *)0xFFFF40) /* FLMCR Address*/
#define EBR1 (*(volatile unsigned char *)0xFFFF42) /* EBR1 Address*/
#define EBR2 (*(volatile unsigned char *)0xFFFF43) /* EBR2 Address*/
#define RAMCR (*(volatile unsigned char *)0xFFFF48) /* RAMCR Address*/
#define DIVCR (*(volatile unsigned char *)0xFFFF5D) /* DIVCR Address*/
#define MSTCR (*(volatile unsigned char *)0xFFFF5E) /* MSTCR Address*/
#define MDCR (*(volatile unsigned char *)0xFFFFF1) /* MDCR Address*/
#define SYSCR (*(volatile unsigned char *)0xFFFFF2) /* SYSCR Address*/
#define INTC (*(volatile struct st_intc *)0xFFFFF4) /* INTC Address*/
#define st_itu1 st_itu0 /* Change Struct ITU1 */
#define st_itu2 st_itu0 /* Change Struct ITU2 */
#define st_itu4 st_itu3 /* Change Struct ITU4 */
#define st_p3 st_p1 /* Change Struct P3->P1 */
#define st_pa st_p1 /* Change Struct PA->P1 */
#define st_pb st_p1 /* Change Struct PB->P1 */

```

```
/*=====
                                     N9604 Address
=====*/
#define    USB9602R    (*(volatile unsigned char *)0x400003)
#define    USB9602D    (*(volatile unsigned char *)0x400001)
```

```
/*=====
                                     N9604 Define
=====*/
#define    USB_CLKDIV    0x04    /* CLKOUT = 48MHz/4 = 12MHz */
```

```
/* USB1.0リクエスト */
```

```
#define    USB_GET_STATUS        0
#define    USB_CLEAR_FEATURE    1
#define    USB_SET_FEATURE      3
#define    USB_SET_ADDRESS      5
#define    USB_GET_DESCRIPTOR   6
#define    USB_SET_DESCRIPTOR   7
#define    USB_GET_CONFIGURATION 8
#define    USB_SET_CONFIGURATION 9
#define    USB_GET_INTERFACE    10
#define    USB_SET_INTERFACE    11
#define    USB_SYNCH_FRAME      12
```

```
/* ディスクリプタ名 */
```

```
#define    USB_DEVICE        1
#define    USB_CONFIGURATION 2
```

```

#define USB_XSTRING          3
#define USB_INTERFACE        4
#define USB_ENDPOINT         5
#define USB_HID              0x21
#define USB_HIDREPORT        0x22
#define USB_HIDPHYSICAL      0x23

```

```
/* HIDリクエスト */
```

```

#define USB_GET_REPORT       0x01
#define USB_GET_IDLE        0x02
#define USB_GET_PROTOCOL    0x03
#define USB_SET_REPORT      0x09
#define USB_SET_IDLE        0x0A
#define USB_SET_PROTOCOL    0x0B

```

```

/*=====
                                N9604 Register
=====*/

```

```

#define USB_MCNTL           0x00 /*Main control register */
#define USB_CCONF           0x01 /*Clk. config. register */
#define USB_TCR             0x02 /*Xcvr config. register */
#define USB_RID             0x03 /*Rev. ID  register */
#define USB_FAR             0x04 /*Func address register */
#define USB_NFSR           0x05 /*Node func st register */
#define USB_MAEV           0x06 /*Main event  register */
#define USB_MAMSK          0x07 /*Main mask  register */
#define USB_ALTEV          0x08 /*Alt. event  register */
#define USB_ALTMSK         0x09 /*ALT mask  register */

```

```

#define USB_TXEV          0x0A /*TX event register */
#define USB_TXMSK        0x0B /*TX mask register */
#define USB_RXEV          0x0C /*RX event register */
#define USB_RXMSK        0x0D /*RX mask register */
#define USB_NAKEV        0x0E /*NAK event register */
#define USB_NAKMSK       0x0F /*NAK mask register */
#define USB_FWEV          0x10 /*FIFO warning register */
#define USB_FWMSK        0x11 /*FIFO warning mask */
#define USB_FNH           0x12 /*Frame nbr hi register */
#define USB_FNL           0x13 /*Frame nbr lo register */
#define USB_DMACNTRL     0x14 /*DMA control register */

#define USB_EPC0          0x20 /*Endpoint0 register */
#define USB_TXD0          0x21 /*TX data register 0 */
#define USB_TXS0          0x22 /*TX status register 0 */
#define USB_TXC0          0x23 /*TX command register 0 */

#define USB_RXD0          0x25 /*RX data register 0 */
#define USB_RXS0          0x26 /*RX status register 0 */
#define USB_RXC0          0x27 /*RX command register 0 */

#define USB_EPC1          0x28 /*Endpoint1 register */
#define USB_TXD1          0x29 /*TX data register 1 */
#define USB_TXS1          0x2A /*TX status register 1 */
#define USB_TXC1          0x2B /*TX command register 1 */

#define USB_EPC2          0x2C /*Endpoint2 register */
#define USB_RXD1          0x2D /*RX data register 1 */
#define USB_RXS1          0x2E /*RX status register 1 */

```

```

#define USB_RXC1          0x2F /*RX command register 1 */

#define USB_EPC3          0x30 /*Endpoint3 register */
#define USB_TXD2          0x31 /*TX data register 2 */
#define USB_TXS2          0x32 /*TX status register 2 */
#define USB_TXC2          0x33 /*TX command register 2 */

#define USB_EPC4          0x34 /*Endpoint4 register */
#define USB_RXD2          0x35 /*RX data register 2 */
#define USB_RXS2          0x36 /*RX status register 2 */
#define USB_RXC2          0x37 /*RX command register 2 */

#define USB_EPC5          0x38 /*Endpoint5 register */
#define USB_TXD3          0x39 /*TX data register 3 */
#define USB_TXS3          0x3A /*TX status register 3 */
#define USB_TXC3          0x3B /*TX command register 3 */

#define USB_EPC6          0x3C /*Endpoint6 register */
#define USB_RXD3          0x3D /*RX data register 3 */
#define USB_RXS3          0x3E /*RX status register 3 */
#define USB_RXC3          0x3F /*RX command register 3 */

/*----- MCNTRL bits -----*/

#define USB_SRST          0x01 /*software reset */
#define USB_DBG           0x02 /*debug mode */
#define USB_VGE           0x04 /*voltage regulator enable*/
#define USB_NAT           0x08 /*node attached */
#define USB_INT_DIS       0x00 /*interrupts disabled */
#define USB_INT_L_O       0x40 /*act lo ints, open drain */

```

```

#define USB_INT_H_P      0x80 /*act hi ints, push pull */
#define USB_INT_L_P      0xC0 /*act lo ints, push pull */

/*----- FAR bits -----*/
#define USB_AD_EN        0x80 /*address enable */

/*----- NFSR bits -----*/
#define USB_RST_ST       0x00 /*reset state */
#define USB_RSM_ST       0x01 /*resume state */
#define USB_OPR_ST       0x02 /*operational state */
#define USB_SUS_ST       0x03 /*suspend state */

/*----- MAEV, MAMSK bits -----*/
#define USB_WARN         0x01 /*warning bit has been set*/
#define USB_ALT          0x02 /*alternate event */
#define USB_TX_EV        0x04 /*transmit event */
#define USB_FRAME        0x08 /*SOF packet received */
#define USB_NAK          0x10 /*NAK event */
#define USB_ULD          0x20 /*unlock locked detected */
#define USB_RX_EV        0x40 /*receive event */
#define USB_INTR_E       0x80 /*master interrupt enable */

/*----- ALTEV, ALTMSK bits -----*/
#define USB_EOP          0x08 /*end of packet */
#define USB_SD3          0x10 /*3 ms suspend */
#define USB_SD5          0x20 /*5 ms suspend */
#define USB_RESET_A      0x40 /*reset detected */
#define USB_RESUME_A     0x80 /*resume detected */

```

/*----- TXEV, TXMSK bits -----*/

```
#define USB_TXFIFO0      0x01 /*TX_DONE, FIFO 0 */
#define USB_TXFIFO1      0x02 /*TX_DONE, FIFO 1 */
#define USB_TXFIFO2      0x04 /*TX_DONE, FIFO 2 */
#define USB_TXFIFO3      0x08 /*TX_DONE, FIFO 3 */
#define USB_TXUDRN0      0x10 /*TX_URUN, FIFO 0 */
#define USB_TXUDRN1      0x20 /*TX_URUN, FIFO 1 */
#define USB_TXUDRN2      0x40 /*TX_URUN, FIFO 2 */
#define USB_TXUDRN3      0x80 /*TX_URUN, FIFO 3 */
```

/*----- RXEV, RXMSK bits -----*/

```
#define USB_RXFIFO0      0x01 /*RX_DONE, FIFO 0 */
#define USB_RXFIFO1      0x02 /*RX_DONE, FIFO 1 */
#define USB_RXFIFO2      0x04 /*RX_DONE, FIFO 2 */
#define USB_RXFIFO3      0x08 /*RX_DONE, FIFO 3 */
#define USB_RXOVRN0      0x10 /*RX_OVRN, FIFO 0 */
#define USB_RXOVRN1      0x20 /*RX_OVRN, FIFO 1 */
#define USB_RXOVRN2      0x40 /*RX_OVRN, FIFO 2 */
#define USB_RXOVRN3      0x80 /*RX_OVRN, FIFO 3 */
```

/*----- NAKEV, NAKMSK bits -----*/

```
#define USB_NAK_I0      0x01 /*IN NAK, FIFO 0 */
#define USB_NAK_I1      0x02 /*IN NAK, FIFO 1 */
#define USB_NAK_I2      0x04 /*IN NAK, FIFO 2 */
#define USB_NAK_I3      0x08 /*IN NAK, FIFO 3 */
#define USB_NAK_O0      0x10 /*OUT NAK, FIFO 0 */
#define USB_NAK_O1      0x20 /*OUT NAK, FIFO 1 */
```

```

#define USB_NAK_O2          0x40 /*OUT NAK, FIFO 2 */
#define USB_NAK_O3          0x80 /*OUT NAK, FIFO 3 */

/*----- EPCX bits -----*/
#define USB_EP_EN          0x10 /*enables endpt. (1-6) */
#define USB_ISO            0x20 /*set for isochr. (1-6) */
#define USB_DEF            0x40 /*force def. adr (0 only) */
#define USB_STALL          0x80 /*force stall handshakes */

/*----- TXCx bits -----*/
#define USB_TX_EN          0x01 /*transmit enable */
#define USB_TX_LAST        0x02 /*last data in FIFO */
#define USB_TX_TOGL        0x04 /*specifies PID used */
#define USB_FLUSH          0x08 /*flushes all FIFO data */
#define USB_IGNIOS         0x80 /* */

/*----- TXSx bits -----*/
#define USB_TX_DONE        0x20 /*transmit done */
#define USB_ACK_STAT       0x40 /*ack status of xmission */

/*----- RXCx bits -----*/
#define USB_RX_EN          0x01 /*receive enable */
#define USB_IGN_OUT        0x02 /*ignore out tokens */
#define USB_IGN_SETUP      0x04 /*ignore setup tokens */

/*----- RXS0 bits -----*/
#define USB_RX_LAST        0x10 /*indicates RCOUNT valid */
#define USB_RX_TOGL        0x20 /*last pkt was DATA1 PID */
#define USB_SETUP_RX       0x40 /*setup packet received */

```



```
#define USB_RX_ERR 0x80 /*last packet had an error*/
```

```
/*
```

```
USB N9604 コントロール
```

```
(C)2002 C.I.M
```

```
*/
```

```
#include <stdio.h>
```

```
#include <string.h>
```

```
#include "usbn9602.h"
```

```
#include "h83048.h"
```

```
extern void PrintSCI(const char *fmt, ...); /* sci.c */
```

```
static void RegisterSet();
```

```
static void ResetUSB();
```

```
static void WakeupUSB();
```

```
static void rx0();
```

```
static void rx1();
```

```
static void tx0();
```

```
static void tx1();
```

```
static void nako0();
```

```
static void nako1();
```

```
static void naki0();
```

```
static void naki1();
```

```
static void clrfeature();
```

```

static void setfeature();

static void getdescriptor();

static void send_desc_sub(void *ptr,int size);

static void send_desc();

static void getstatus();

static void setconfiguration();

static void SetStallUSB(int adr);

static void ClearStallUSB(int adr);

static void FlushRXC(int no);

static void FlushTXC(int no);

static void TxToggle(int no);

static void WriteUSB(int adr,int data);

static unsigned char ReadUSB(int adr);

static int ReadUSBBurst(int adr,int adr2,char *buff,int cnt);

static int WriteUSBBurst(int adr,int adr2,char *buff,int cnt);

/*-----*/

static int SendTX1();

/*-----*/

int get_inbufflen(void);

void init_usbbuff(void);

int write_inbuff(char *p,int size);

int get_outbufflen(void);

int write_buff(char *p,int size);

int read_outbuff(char *p,int size);

/*-----*/

```

```

static unsigned char usbevent;      /* USB割り込みイベント */
static unsigned char SETADDR;      /* アドレスセット */
static unsigned char configno;     /* コンフィグレーションNO */
static unsigned char usbbuff[64];  /* 読み込みバッファ */
static unsigned char rx1buff[64];
static unsigned char rx2buff[64];
static unsigned char STALLD;       /* ECPの状態 */
static unsigned char DATA0_1;     /* USB_TXTGLのフラグ */
static char          senddesc;     /* 1 = ディスクリプタ送信中 */
static int           desc_size;    /* ディスクリプタ送信サイズ */
static char          *desc_ptr;    /* ディスクリプタポインタ */

static const unsigned char epctbl[8] =
{USB_EPC0,USB_EPC1,USB_EPC2,USB_EPC3,USB_EPC4,USB_EPC5,USB_EPC6,USB_EPC0};
static int txcreg[4] = {USB_TXC0,USB_TXC1,USB_TXC2,USB_TXC3};
static int rxcreg[4] = {USB_RXC0,USB_RXC1,USB_RXC2,USB_RXC3};

/*-----*/
/*-----*/

static const unsigned char dev_desc[] = {
    0x12,      /* length of this desc. */
    0x01,      /* デバイス・ディスクリプタ 1 */
    0x00,0x01, /* USB Version 1.0 */
    0x00,      /* device class クラス無し */
    0x00,      /* device subclass */
    0x00,      /* device protocol */
    0x08,      /* EP0の最大パケットサイズ */

```

```

0xfe,0xff,          /* vendor ID サンプルなのでとりあえず */
0x10,0x00,         /* product ID */
0x01,0x00,         /* revision ID */
0x01,              /* index of manuf. string */
0x01,              /* index of prod. string */
0x02,              /* index of ser. # string */
0x01               /* bNumConfigurations */
};

/* コンフィグレーションディスクリプタ */
static const unsigned char cfg_desc[] = {
    0x09,           /* length of this desc. */
    0x02,           /* コンフィグレーション・ディスクリプタ */
    9+9+7*3,       /* インターフェース／エンドポイントディスクリプタ等の合計長 CFG + IF +
EP*3 */
    0x00,           /*
*/
    0x01,           /* インターフェース数 1 */
    0x01,           /* コンフィグレーションは 1 */
    0x00,           /* index of config. string */
    0xc0,           /* attr.: self powered D6=自己電源 */
    100,           /* ;max power (100 mA) */
};

/*static const unsigned char if_desc[] = {*/
    0x09,           /* length of this desc. */
    0x04,           /* INTERFACE descriptor */
    0x00,           /* interface number */
    0x00,           /* alternate setting */
    0x03,           /* # of (non 0) endpoints */
    0x00,           /* interface class */
};

```

```

0x00,          /* interface subclass          */
0x00,          /* interface protocol          */
0x03,          /* index of intf. string      */
/*},*/
/*static const unsigned char endp_desc[] = {*/
/* pipe 0 */
7,             /* length of this desc.       */
5,             /* ENDPOINT descriptor        */
0x81,          /* address (IN)                */
0x02,          /* attributes (BULK)          */
0x40,0x00,     /* max packet size (64)       */
255,          /* interval (ms)              */
/* pipe 1 */
7,             /* length of this desc.       */
5,             /* ENDPOINT descriptor        */
0x02,          /* address (OUT)               */
0x02,          /* attributes (BULK)          */
0x40,0x00,     /* max packet size (64)       */
255,          /* interval (ms)              */

/* pipe 2 (not use) */
7,             /* length of this desc.       */
5,             /* ENDPOINT descriptor        */
0x83,          /* address (IN)                */
0x02,          /* attributes (BULK)          */
0x40,0x00,     /* max packet size (64)       */
255,          /* interval (ms)              */
};

```

```
static const char lang_data[] = {
    4,3,9,4    /* LANGID (English)    */
};
```

```
static const char mfg_str[] = {
    18,3,
    'U',0,'S',0,'B',0,' ',0,'T',0,'E',0,'S',0,'T',0,
};
```

```
static const char nbr_str[] = {
    8,3,
    '1',0,',',0,'0',0,
};
```

```
static const char int_str[] = {
    34,3,
    'U',0,'S',0,'B',0,' ',0,
    'T',0,'E',0,'S',0,'T',0,' ',0,'P',0,'R',0,'O',0,'G',0,'R',0,'A',0,'M',0,
};
```

```
static void wait(int c)
{
    int    i,j;
    for(j=0;j<c;j++)
    {
        for(i=0;i<0x682;i++)
```

```

    {
    }
}

/*-----*/

/* USB初期化 */
void InitUSB()
{
    init_usbbuff();

    ResetUSB();

    RegisterSet();

    WakeupUSB();

/*    PrintSCI(" REV = %d¥n",ReadUSB(USB_RID)); */

/*    PrintSCI(" CLOCK = %02X¥n",ReadUSB(USB_CCONF)); */

}

static void RegisterSet()
{
    STALLD = 0;

    senddesc = 0;

    DATA0_1 = 0;

    SETADDR = 0;

    WriteUSB(USB_FAR,USB_AD_EN+0);          /* アドレス初期化    */
    WriteUSB(USB_EPC0,USB_EP_EN);          /* EP0をイネーブル    */
    WriteUSB(USB_NAKMSK,USB_NAK_O0);       /* NAK MASKをセット*/
    WriteUSB(USB_TXMSK,USB_TXFIFO0+USB_TXFIFO1+USB_TXFIFO2+USB_TXFIFO3); /* TX MASK

```


をセツト*/

```
WriteUSB(USB_RXMSK,USB_RXFIFO0+USB_RXFIFO1+USB_RXFIFO2+USB_RXFIFO3); /* RX MASK
```

をセツト*/

```
WriteUSB(USB_ALTMSK,USB_SD3+USB_RESET_A); /* ALT MASKをセツト*/
```

```
WriteUSB(USB_MAMSK,USB_INTR_E+USB_RX_EV+USB_NAK+USB_TX_EV+USB_ALT); /*
```

MAIN MASKをセツト*/

```
FlushTXC(0);
```

```
FlushRXC(1);
```

```
FlushTXC(1);
```

```
WriteUSB(USB_TXC1,0);
```

```
WriteUSB(USB_RXC1,0);
```

```
WriteUSB(USB_RXC0,USB_RX_EN); /* RX0をイネ-ブル */
```

}

static void ResetUSB()

{

```
WriteUSB(USB_MCNTRL,USB_SRST+USB_VGE); /* USBリセット 3.3V供給 */
```

```
wait(100); /* 100msec */
```

```
WriteUSB(USB_MCNTRL,USB_INT_L_P+USB_VGE); /* 割込みはactive low push pull */
```

```
WriteUSB(USB_CCONF,USB_CLKDIV-1); /* 48MHz/4 = 12MHz */
```

}

static void WakeupUSB()

{

```
WriteUSB(USB_NFSR,USB_OPR_ST); /* 動作可にする */
```

```
WriteUSB(USB_MCNTRL,USB_INT_L_P+USB_NAT+USB_VGE); /* USBのノ-トを動作可にする */
```

}

/* USBポートデータ表示 */

```
/* ※リードすると、ステータスが変わるレジスタもあるので注意 */
```

```
void DispUSBPort()
```

```
{
```

```
    int    i,j;
```

```
    PrintSCI("00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F¥n");
```

```
    for(j=0;j<4;j++)
```

```
    {
```

```
        for(i=0;i<16;i++)
```

```
            PrintSCI("%02X ",ReadUSB(i+j*16));
```

```
            PrintSCI("¥n");
```

```
    }
```

```
}
```

```
/*-----*/
```

```
/* USB割り込み */
```

```
#ifdef __GNUC__
```

```
void usb_int() __attribute__((interrupt_handler));
```

```
#endif
```

```
void usb_int()
```

```
{
```

```
    unsigned char  nakeevent,rxevent,txevent,altevent;
```

```
    char  reg;
```

```
    usbevent = ReadUSB(USB_MAEV);
```

```
    if( usbevent & USB_NAK )
```

```
    {
```

```
nakevent = ReadUSB(USB_NAKEV);
if( nakevent & USB_NAK_O0 )
{
    nako0();
}
if( nakevent & USB_NAK_O1 )
{
    nako1();
}
else if( nakevent & USB_NAK_I0 )
{
    naki0();
}
else if( nakevent & USB_NAK_I1 )
{
    naki1();
}
}
else if( usbevent & USB_RX_EV )
{
    rxevent = ReadUSB(USB_RXEV);
    if( rxevent & USB_RXFIFO0 )
    {
        rx0();
    }
    else if( rxevent & USB_RXFIFO1 )
    {
        rx1();
    }
}
```

```

}
else if( usbevent & USB_TX_EV )
{
    txevent = ReadUSB(USB_TXEV);
    if( txevent & USB_TXFIFO0 )
    {
        tx0();
    }
    else if( txevent & USB_TXFIFO1 )
    {
        tx1();
    }
}
else if( usbevent & USB_ALT )
{
    altevent = ReadUSB(USB_ALTEV);
    if( altevent & USB_RESET_A )
    {
        /* リセット */
        RegisterSet();
        WakeupUSB();
    }
    else if( altevent & USB_SD3 )
    {
        /* サスペンド */
        /* ALTMSKをセット */
        WriteUSB(USB_ALTMSK,USB_RESUME_A+USB_RESET_A);
        /* ノードをサスペンド */
        WriteUSB(USB_NFSR,USB_SUS_ST);
    }
}

```

```

}
else if( altevent & USB_RESUME_A )
{
    /* リジューム */
    /* ALTMSKをセット */
    WriteUSB(USB_ALTMSK,USB_SD3+USB_RESET_A);
    /* ノード を動作可能にする */
    WriteUSB(USB_NFSR,USB_OPR_ST);
}
}
}
}

```

```

/*=====

```

RXイベントの処理

```

=====*/

```

```

/* RX0(system) */

```

```

/*

```

リクエストコードの取得

0 byte

D7 ... データ方向 0=ホスト->デバイス, 1=デバイス->ホスト

D6-D5 ... タイプ

0:標準, 1:クラス, 2:ベンダ, 3:予約

D4-D0 ... 受信側

0:デバイス, 1:インターフェイス, 2:エンドポイント, 3:その他

1 byte

特定のリクエスト

2 byte

value

2 byte

index

2 byte

length

*/

static void rx0()

{

unsigned char rxstat;

rxstat = ReadUSB(USB_RXS0);

if(rxstat & USB_SETUP_RX)

{

ReadUSBBurst(USB_RXD0,USB_RXS0,(char*)usbbuff,8);

FlushRXC(0);

FlushTXC(0);

ClearStallUSB(USB_EPC0);

if((usbbuff[0] & 0x60) == 0)

{

/* 標準リクエスト */

switch(usbbuff[1])

{

case USB_CLEAR_FEATURE :

clrfeature();

break;

case USB_GET_CONFIGURATION :

WriteUSB(USB_TXD0,configno);

break;

case USB_GET_DESCRIPTOR :

getdescriptor();

```

        break;
    case USB_GET_STATUS :
        getstatus();
        break;
    case USB_GET_INTERFACE :
        WriteUSB(USB_TXD0,0);
        break;
    case USB_SET_ADDRESS :
        WriteUSB(USB_EPC0,USB_DEF);
        SETADDR = usbbuff[2];USB_AD_EN;
        WriteUSB(USB_FAR,SETADDR);
        break;
    case USB_SET_CONFIGURATION :
        setconfiguration();
        break;
    case USB_SET_FEATURE :
        setfeature();
        break;
    case USB_SET_INTERFACE :
        if( usbbuff[2] != 0 )
            SetStallUSB(USB_EPC0);
        break;
    default :
        /* 未定義 */
        SetStallUSB(USB_EPC0);
        break;
    }
}
else if( (usbbuff[0] &0x60 ) == 0x20 )

```

```

{
    /* クラスリクエスト */
    SetStallUSB(USB_EPC0);
}
else if( (usbbuff[0] &0x60 ) == 0x40 )
{
    /* ベンダリクエスト */
    SetStallUSB(USB_EPC0);
}
else
{
    /* 未定義 */
    SetStallUSB(USB_EPC0);
}
/* SETUPなのでデータの有無に関係無くDATA1として送信 */
DATA0_1 |= 1;
TxToggle(0);
}
else
{
    if( senddesc )
    {
        senddesc = 0;
    }
    FlushTXC(0);
    WriteUSB(USB_RXC0,USB_RX_EN);
}
}

```



```

/*-----*/

/* RX1 受信 */

static void rx1()

{

    int          cnt;

    unsigned char  rxstat;

    rxstat = ReadUSB(USB_RXS1);          /* RX1ステータス取得 */

    /* SETUP,ERROR/パケットでない */

    if( !(rxstat & (USB_SETUP_RX|USB_RX_ERR)) )

    {

        /* ホストからデータの受信 */

        /* FIFOからデータ取得 */

        cnt = ReadUSBBurst(USB_RXD1,USB_RXS1,(char*)rx1buff,64);

        /* リングバッファに書き込み */

        write_inbuff((char*)rx1buff,cnt);

    }

    FlushRXC(1);          /* バッファをフラッシュ */

    WriteUSB(USB_RXC1,USB_RX_EN);    /* 受信可に設定          */

}

```

```

/*-----*/

```

```

/* RX2 受信(not use) */

```

```

static void rx2()

```

```

{

    unsigned char  rxstat;

    rxstat = ReadUSB(USB_RXS2);

}

```

```

/*=====

```

TXイベントの処理

```
=====*/  
/* TX0 送信終了 */  
static void tx0()  
{  
    unsigned char  txstat;  
    txstat = ReadUSB(USB_TXS0);  
    if( (txstat & USB_ACK_STAT) && (txstat & USB_TX_DONE) )  
    {  
        /* ok */  
        FlushTXC(0);  
        if( senddesc )  
        {  
            send_desc();  
            TxToggle(0);          /* TX0送信可 */  
        }  
        else  
        {  
            WriteUSB(USB_RXC0,USB_RX_EN); /* RX0受信可 */  
        }  
    }  
    else  
    {  
        /* error ? */  
    }  
}  
  
/*-----*/
```

```
/* TX1送信終了 */
```

```
static void tx1()
```

```
{
```

```
    unsigned char  txstat;
```

```
    txstat = ReadUSB(USB_TXS1);
```

```
    if( (txstat & USB_ACK_STAT) && (txstat & USB_TX_DONE) )
```

```
    {
```

```
        /*
```

```
            送信終了後に次の送信データを送信するようにする
```

```
            送信データが無い場合、HOSTには0バイトで送る
```

```
        */
```

```
        SendTX1();
```

```
    }
```

```
    else
```

```
    {
```

```
        /*
```

```
            送ったサイズより小さい読み込みが行われた場合、こちらにくる場合がある
```

```
        */
```

```
    }
```

```
}
```

```
/*=====
```

```
                NAKイベントの処理
```

```
=====*/
```

```
/*
```

```
    NAKイベントは、エラーが発生した場合など再送信する場合処理する
```

```
    NAK0しかENABLEにしていないので、それ以外は処理無し
```

```
*/
```

```
static void nako0()
```

```

{
}

static void nako1()
{
}

static void naki0()
{
}

static void naki1()
{
}

/*=====
標準リクエストの処理
=====*/

/* 選択機器 */
static void clrfeature()
{
    if( (usbbuff[0] & 3) == 2 )
    {
        /* エンドポイントデータ */
        if((usbbuff[3]&7) != 0)
            ClearStallUSB(epctbl[usbbuff[3]&7]);
        STALLD &= -1 ^ (1<<(usbbuff[3]&7));
    }
}

static void setfeature()

```

```

{
    if( (usbbuff[0]&3) == 2 )      /* ENDPOINT */
    {
        /* エンドポイントデータ */
        if((usbbuff[3]&7) != 0 )
            SetStallUSB(epctbl[usbbuff[3]&7]);
        STALLD |= (1<<(usbbuff[3]&7));
    }
}

/*-----*/
/* ディスクリプタを返す */
static void getdescriptor()
{
    DATA0_1 &= 0xfe;
    switch( usbbuff[3] )
    {
        case  USB_DEVICE :
            send_desc_sub((void*)dev_desc,dev_desc[0]);
            break;
        case  USB_CONFIGURATION :
            {
                send_desc_sub((void *)cfg_desc,cfg_desc[2]);
                break;
            }
        case  USB_XSTRING :
            {
                switch( usbbuff[2] )
                {

```

```

        case 0 :
            send_desc_sub((void *)lang_data,lang_data[0]);
            break;
        case 1 :
            send_desc_sub((void *)mfg_str,mfg_str[0]);
            break;
        case 2 :
            send_desc_sub((void *)nbr_str,nbr_str[0]);
            break;
        case 3 :
            send_desc_sub((void *)int_str,int_str[0]);
            break;
    }
    break;
}
default :
{
}
}
}

```

```

static void send_desc_sub(void *ptr,int size)

```

```

{
    desc_size = (usbbuff[7] << 8) + usbbuff[6];
    /* 受信要求バッファ以上はデータを送らない */
    if( desc_size > size ) desc_size = size;
    desc_ptr = ptr;
    senddesc = 1; /* ディスクリプタ送信中フラグを立てる */
}

```

```

    send_desc();
}

static void send_desc()
{
    int    sz;

    sz = 8;

    if( desc_size == 0 ) return;
    if( desc_size <= 8 ) sz = desc_size;

    sz = WriteUSBurst(USB_TXD0,USB_TXS0,desc_ptr,sz);

    desc_size -= sz;

    desc_ptr += sz;

    if( desc_size == 0 ) senddesc = 0;
}

/*-----*/

/* ステータス */

static void getstatus()
{
    int    data,ep;

    data = usbbuff[0]&3;

    if( (data == 0) || (data == 1) )    /* DEVICE,INTERFACE */
    {
        WriteUSB(USB_TXD0,0);
        WriteUSB(USB_TXD0,0);
    }

    else if( data== 2)                /* エンドポイント */
    {

        ep = usbbuff[3]&7;

        /* epのSTALL状態を送信 */

```

```

    if( STALLD & (1<<ep) ) WriteUSB(USB_TXD0,1);
    else
        WriteUSB(USB_TXD0,0);
}
else
{
    WriteUSB(USB_TXD0,0);
}
}

/*-----*/
static void setconfiguration()
{
    configno = usbbuff[2];
    if( configno == 0 )
    {
        WriteUSB(USB_EPC1,0);        /* EPC1を使用不可 */
        WriteUSB(USB_EPC2,0);        /* EPC2を使用不可 */
        WriteUSB(USB_EPC3,0);        /* EPC3を使用不可 */
        WriteUSB(USB_EPC4,0);        /* EPC4を使用不可 */
        WriteUSB(USB_EPC5,0);        /* EPC5を使用不可 */
        WriteUSB(USB_EPC6,0);        /* EPC6を使用不可 */
    }
    else
    {
        STALLD = 0;
        FlushTXC(1);
        /* EPC1をアドレス1としてイネーブル */
        WriteUSB(USB_EPC1,USB_EP_EN+01);
        /* TX1送信可 */
    }
}

```



```
WriteUSB(USB_TXC1,USB_TX_EN|USB_TX_LAST);
```

```
FlushRXC(1);
```

```
/* EPC2をアドレス2としてイネ-ブル */
```

```
WriteUSB(USB_EPC2,USB_EP_EN+02);
```

```
/* RX1受信可 */
```

```
WriteUSB(USB_RXC1,USB_RX_EN);
```

```
/*
```

USB_TX_LASTを立てると、READ時ストールしなくなる。

ただし、データを送る前の最初のREADは0byteになります。

host側は複数のリクエストを同時発行できないので、

リクエストがストールするのはまずいの回避。

```
*/
```

```
}
```

```
}
```

```
/*=====
```

汎用ルーチン

```
=====*/
```

```
/* STALLのセットとクリア */
```

```
static void SetStallUSB(int adr)
```

```
{  
    WriteUSB(adr,ReadUSB(adr) | 0x80);
```

```
}
```

```
static void ClearStallUSB(int adr)
```

```
{  
    WriteUSB(adr,ReadUSB(adr)&0x7f);
```

```
}
```

```
/* FIFOのフラッシュ */
```

```
static void FlushRXC(int no)
```

```
{  
    WriteUSB(rxcreg[no],USB_FLUSH);  
}
```

```
static void FlushTXC(int no)
```

```
{  
    int    d;  
    d = ReadUSB(txcreg[no]);  
    d != USB_FLUSH;  
    WriteUSB(txcreg[no],d);  
}
```

```
/* 送信終了フラグセット */
```

```
/* reg = USB_TXC0~6 */
```

```
static void TxToggle(int no)
```

```
{  
    unsigned char d;  
    d = USB_TX_EN;  
    if( DATA0_1 & (1<<no) ) d != USB_TX_TOGL;  
    else                d &= ~USB_TX_TOGL;  
    d != USB_TX_LAST;  
    WriteUSB(txcreg[no],d);  
    DATA0_1 ^= (1<<no);  
}
```

```
/*-----*/
```

```
/* USBのアドレスから読み込み */
```

```
static unsigned char ReadUSB(int adr)
```

```
{  
    USB9602R = (unsigned char)adr;  
    return( USB9602D );  
}
```

```
/* USBのアドレスへ書き込み */
```

```
static void WriteUSB(int adr,int data)
```

```
{  
    USB9602R = (unsigned char)adr;  
    USB9602D = (unsigned char)data;  
}
```

```
/* バースト転送 */
```

```
static int ReadUSBBurst(int adr,int adr2,char *buff,int cnt)
```

```
{  
    int    i;  
    int    rcnt;  
    USB9602R = (unsigned char)adr;  
    for(rcnt=0,i=0;i<cnt;i++)  
    {  
        if( (ReadUSB(adr2) & 0xf) == 0 )    break;  
  
        USB9602R = (unsigned char)adr;  
        *buff = USB9602D;  
        buff++;  
        rcnt++;  
    }
```

```

    }
    return(rcnt);
}

static int WriteUSBurst(int adr,int adr2,char *buff,int cnt)
{
    int    i,scnt;
    for(scnt=0,i=0;i<cnt;i++)
    {
        if( (ReadUSB(adr2) & 0x1f) == 0 )    break;

        USB9602R = (unsigned char)adr;
        USB9602D = *buff;
        buff++;
        scnt++;
    }
    return(scnt);
}

```

```

/*=====

```

サンプルプログラム

```

=====*/

```

```

/* TX1送信ルーチン */

```

```

/*

```

USBから一方的に送信できないため、今回は、

TX1送信終了時にバッファ(outbuff)にあるデータを送信します。

よって、HOSTからは定期的にREADを行う。

それ以外のタイミングではEPC1のFIFOバッファのサイズに注意する。

```

*/

```

```

static int SendTX1()
{
    int    c,cnt,sz,i;

    cnt = 0;
    /* FIFOは最大64byte */
    sz = read_outbuff((char*)rx2buff,64);
    FlushTXC(1); /* 送信バッファをフラッシュ */
    if( sz != 0 )
    {
        /* バースト転送 */
        cnt = WriteUSBurst(USB_TXD1,USB_TXS1,(char*)rx2buff,sz);
    }
    TxToggle(1); /* 送信終了処理 */
    return( cnt ); /* 送信データ数を返す sz==cntのはず */
}

```

```

/*-----*/

```

```

/*

```

送受信バッファ

inbuffがHOSTから送られてきたデータのバッファ

outbuffがHOSTへ送るデータ用

このサンプルではリングバッファを超えた分は捨てられます。

今回は、256バイト確保しています。バッファがあふれないように

メイン側で処理してください。

```

*/

```

```

#define    USBBUFFLEN    256          /* バッファのサイズ */

```

```

static int    inpos,inlen;          /* 入力バッファ位置、サイズ */

```

```

static int    outpos,outlen;       /* 出力バッファ位置、サイズ */

```

```

static char  inbuff[USBBUFFLEN];      /* 入力リングバッファ */
static char  outbuff[USBBUFFLEN];    /* 出力リングバッファ */

/*-----*/
/*          バッファの初期化          */
/*-----*/

void init_usbbuff()
{
    inpos = inlen = 0;
    outpos = outlen = 0;
}

/*-----*/
/*          HOSTからの受信バッファへ書き込み          */
/*          char      *p      バッファポインタ          */
/*          int      size  書き込みサイズ          */
/*          戻り値          書き込んだサイズ          */
/*-----*/

int write_inbuff(char *p,int size)
{
    int      i;
    INTC.IER &= (-1^0x20);          /* IRQ5 Disable */
    for(i=0;i<size;i++)
    {
        if( inlen >= USBBUFFLEN )    break;
        inbuff[inpos] = *p;
        inpos = (inpos + 1)%USBBUFFLEN;
        inlen++;
    }
}

```

```

    p++;
}

INTC.IER |= 0x20;          /* IRQ5 Enable */

return(i);
}

/*-----*/

/*          送信バッファへ書き込み          */
/* char    *p      バッファポインタ          */
/* int     size   書き込みサイズ            */
/* 戻り値          書き込んだサイズ          */
/*-----*/

int write_buff(char *p,int size)
{
    int    i;

    INTC.IER &= (-1^0x20);    /* IRQ5 Disable */

    for(i=0;i<size;i++)
    {
        if( outlen >= USBBUFFLEN )    break;

        outbuff[outpos%USBBUFFLEN] = *p;
        outpos = (outpos + 1)%USBBUFFLEN;
        outlen++;

        p++;
    }

    INTC.IER |= 0x20;          /* IRQ5 Enable */

    return(i);
}

/*-----*/

/*          送信バッファから読み込み          */

```

```

/* char      *p      バッファポインタ          */
/* int       size   バッファ最大サイズ        */
/* 戻り値     読み込んだサイズ                */
/*-----*/

```

```
int read_outbuff(char *p,int size)
```

```

{
    int i;
    INTC.IER &= (-1^0x20);          /* IRQ5 Disable */
    for(i=0;outlen>0;i++)
    {
        if( i >= size ) break;
        p[i] = outbuff[ (USBBUFFLEN+outpos-outlen)%USBBUFFLEN ];
        outlen--;
    }
    INTC.IER |= 0x20;              /* IRQ5 Enable */
    return(i);
}

```

```
/*-----*/
```

```

/*          受信バッファから読み込み          */
/* char      *p      バッファポインタ          */
/* int       size   バッファ最大サイズ        */
/* 戻り値     読み込んだサイズ                */
/*-----*/

```

```
int read_buff(char *p,int size)
```

```

{
    int i;
    INTC.IER &= (-1^0x20);          /* IRQ5 Disable */
    for(i=0;inlen>0;i++)
    {

```



```

    if( i >= size ) break;

    p[i] = inbuff[ (USBBUFFLEN+inpos-inlen)%USBBUFFLEN ];
    inlen--;
}

INTC.IER |= 0x20;          /* IRQ5 Enable */

return(i);
}

/*-----*/
/*          受信バッファのサイズ取得          */
/*-----*/

int get_inbufflen()
{
    return( inlen%USBBUFFLEN );
}

/*-----*/
/*          送信バッファのサイズ取得          */
/*-----*/

int get_outbufflen()
{
    return( outlen%USBBUFFLEN );
}

```

```
/*
```

SCI処理

(C)2002 C.I.M

```
*/
```

```
#include <stdio.h>
```

```
#include <stdarg.h>
```

```
#include "h83048.h"
```

```
static char buff[80]; /* 文字列展開用バッファ(必要なら増やす) */
```

```
/*=====
```

SCI初期化

```
-----
```

```
9600bps パリティ無し STOP1
```

```
=====*/
```

```
void InitSCI()
```

```
{
```

```
int i;
```

```
SCI1.SCR = 0;
```

```
SCI1.SMR = 0; /* パリティ無し STOP1 */
```

```
SCI1.BRR = 80; /* 9600bps 3052 */
```

```
for(i=0;i<280;i++) {} /* wait */
```

```
SCI1.SCR = 0x30; /* TE = 1 , RE = 1 */
```

```
i = SCI1.SSR;
```

```
SCI1.SSR = 0x80; /* Clear Error Flag (TDRE=1) */
```

```
}
```

```
/*=====
```

SCI出力

```
-----
```

```
=====*/
```

```
void PutSCI(char c)
```

```
{  
    unsigned char i;  
    while( 1 )  
    {  
        i = SCI1.SSR;  
        if( i & 0x80 )    break;  
    }  
    SCI1.TDR = c;  
    SCI1.SSR = i&0x7f;  
}
```

```
/*=====
```

SCI入力

```
-----
```

データを受信するまで待ちつづけます。

```
=====*/
```

```
char GetSCI()
```

```
{  
    unsigned char i;  
    char          c;  
    while( 1 )  
    {
```

```

        i = SCI1.SSR;
        if( i & 0x40 )    break;
    }
    c = SCI1.RDR;
    SCI1.SSR = i&0xbf;
    return(c);
}

```

```

/*=====

```

SCI入力データチェック

```

-----

```

SCIにデータがあるかチェックします。

戻り値 1 = データあり、0 = データなし

```

=====*/

```

```

int ScanSCI()

```

```

{
    if( SCI1.SSR & 0x40 ) return(1);
    return(0);
}

```

```

/*=====

```

SCI文字列出力

```

-----

```

書式はprintf()と同じです。バッファは80文字分しか取っていないので、

必要ならば、増やしてください。

```

=====*/

```

```

void PrintSCI(char *fmt, ...)

```

```

{
    int    i;

```

```
va_list arg;
va_start(arg, fmt);
*buff = '\0';
vsprintf(buff,fmt,arg);
va_end(arg);
```

```
for(i=0;;i++)
```

```
{
```

```
    if( buff[i] == 0 )    break;
```

```
    /* 改行コードは2バイトにして送信 */
```

```
    if( buff[i] == '\n' ) PutSCI('\r');
```

```
    PutSCI(buff[i]);
```

```
}
```

```
}
```

```
/*
```

```
    LCD处理
```

```
    (C)2002 C.I.M
```

```
*/
```

```
#include <stdio.h>
```

```
#include <stdarg.h>
```

```
#include "h83048.h"
```

```
void ClearLCD();
```

```
/* PortB (write) b0..3 = LCD(LED) out , bit4 = LCD RS , bit7 = LCD E */
```

```
#define LCD_RS 0x10
```

```
#define LCD_E 0x80
```

```
#define LCDMASK 0x60
```

```
static void wait(int c)
```

```
{
    int i,j;
    for(j=0;j<c;j++)
    {
        for(i=0;i<0x682;i++) {}
    }
}
```

```
/*=====
LCD BYTE 出力
=====*/
```

```
/*
 今回の基板は4bit接続なので、下位4bitのみ出力
*/
```

```
static void LCDOut8(short rs,short code)
{
    int    stat;
    int    pb;

    pb = PB.DR;          /* 現在のPBDを退避 */

    if( rs )    stat = (pb & LCDMASK) | LCD_RS;
    else        stat = (pb & LCDMASK);

    PB.DR = code | stat | LCD_E;
    PB.DR = code | stat;
    PB.DR = pb;          /* 元のPBDに復帰 */
    wait(4);
}
```

```
/*=====
LCD BYTE 出力(4bit)
=====*/
```

```
void LCDOut4(int rs,int code)
{
    int    stat;
    int    pb;
    char  lb,hb;
```

```
pb = PB.DR;          /* 現在のPBDを退避 */
```

```
if( rs )    stat = (pb & LCDMASK) | LCD_RS;
```

```
else        stat = (pb & LCDMASK);
```

```
hb = ((code>>4)&0xf) | stat;
```

```
lb = (code&0xf) | stat;
```

```
PB.DR = hb | LCD_E;
```

```
PB.DR = hb;
```

```
PB.DR = lb | LCD_E;
```

```
PB.DR = lb;
```

```
PB.DR = pb;          /* 元のPBDに復帰 */
```

```
wait(4);
```

```
}
```

```
/*=====
```

LCD コントロール

```
-----
```

LCD初期化、表示、クリア

```
=====*/
```

```
void InitLCD()
```

```
{
```

```
int    i;
```

```
wait(15);
```



```

for(i=0;i<3;i++)
{
    LCDOut8(0,0x3);
}
LCDOut8(0,0x2);

LCDOut4(0,0x28); /* bit4=8/4bit , bit3=1/2line , bit2=large/small */
LCDOut4(0,0x10); /* bit3=Display/Cursor , bit2=Right/Left */
LCDOut4(0,0x0e); /* bit2=display , bit1=cursor , bit0=blink */
LCDOut4(0,0x06);

LCDOut4(0,0x01); /* クリア */
LCDOut4(0,0x02); /* カーソルホーム */

}

/*=====
LCD クリア
-----
LCD初期化、表示、クリア
=====*/

void ClearLCD()
{
    LCDOut4(0,0x01); /* クリア */
    LCDOut4(0,0x02); /* カーソルホーム */
}

```

```
/*=====
```

LCDキャラクタ表示

```
-----
```

'\n','\r','\f'はLCDクリア処理を行います。

```
=====*/
```

```
void PutLCD(char c)
```

```
{
```

```
    if( c == '\f' ) ClearLCD();
```

```
    else if( c == '\n' ) ClearLCD();
```

```
    else if( c == '\r' ) ClearLCD();
```

```
    else LCDOut4(1,c);
```

```
}
```

```
/*=====
```

LCDカーソル移動

```
-----
```

x = 0~15

y = 0,1

```
=====*/
```

```
void LocateLCD(int x,int y)
```

```
{
```

```
    LCDOut4(0,0x80 + y*0x40 + x);
```

```
}
```

```
/*=====
```

LCD文字列表示

```
-----
```

パラメータはprintf()と同じです。64文字を超えないように設定してく

ださい。'%f'はLCDクリア、'%n'は改行。

=====*/

```
void PrintLCD(char *fmt,...)
{
    int    i;
    static char  buff[64];
    va_list arg;
    va_start(arg, fmt);
    *buff = '\0';
    vsprintf(buff,fmt,arg);
    va_end(arg);
    for(i=0;;i++)
    {
        if( buff[i] == 0 ) break;
        else if( buff[i] == '\n' ) LocateLCD(0,1); /* 改行 */
        else if( buff[i] == '\r' ) LCDOut4(0,0x2); /* カーソルホーム */
        else if( buff[i] == '%f' ) ClearLCD(); /* LCDクリア */
        else LCDOut4(1,buff[i]); /* データ出力 */
    }
}
```

本文

```

/* C.h */

#define USE_THREAD
/* #define USE_BCC */
/* #define USE_LINUX */
/* #define CENTOS */
/* #define USE_RASPBIAN */
#ifdef USE_CENTOS
#define USE_LINUX
#endif
#ifdef USE_RASPBIAN
#define USE_LINUX
#endif
#ifdef USE_LINUX
#define USE_BCC
#endif
#include <stdio.h> /* printf() */
#include <string.h> /* strcmp(), strlen() */
#include <stdlib.h> /* calloc(), free(), rand() */
#ifndef USE_BCC
/* AKI-H8 3052F USB */
#include "h83048.h"
#define SLEEP_PER_SEC 16000.0
#define NOTUSE_FILES
#else
/* time_t, tm, time(), clock(), CLOCKS_PER_SEC */
#include <time.h>
#ifdef USE_LINUX
#include <termios.h> /* kbhit() */
#include <unistd.h> /* kbhit() */
#include <fcntl.h> /* kbhit() */
#ifdef USE_CENTOS
#include <mysql.h> /* LINUX CentOS GCC MySQL */
#endif
#ifdef USE_RASPBIAN
#include <mysql/mysql.h> /* LINUX Raspbian GCC MySQL */
#endif
#ifdef USE_RASPBIAN
#include <wiringPi.h> /* LINUX Raspberry Pi 3 Model B I/O */
#endif
#else
#include <conio.h> /* kbhit(), getche() */
#endif
#define SLEEP_PER_SEC 100000000.0
#endif
#ifdef USE_LINUX
#define CLEAR system("clear")
#else
#define CLEAR system("cls")
#endif
#define OK 0
#define NG 1
#define ONE_MORE_TIME 2
#define ON 1
#define OFF 0

```

```
/* Panel.h */
```

```
#ifndef USE_BCC
```

```
/*=====
```

```
外部参照
```

```
=====*/
```

```
/* asmfile.src 内に定義 */
```

```
extern void EnableInterrupt(void);
```

```
extern void DisableInterrupt(void);
```

```
/* lcd.c */
```

```
extern void InitLCD(void);
```

```
extern void PrintLCD(char *fmt,...);
```

```
extern void PutLCD(char c);
```

```
/* usb.c */
```

```
extern void InitUSB(void);
```

```
extern void DispUSBPort(void);
```

```
/* バッファ処理 */
```

```
extern int get_inbufflen(void);
```

```
extern int write_buff(char *p,int size);
```

```
extern int read_buff(char *p,int size);
```

```
/* sci.c */
```

```
extern void InitSCI(void);
```

```
extern void PrintSCI(char *fmt, ...);
```

```
extern int ScanSCI(void);
```

```
extern char GetSCI(void);
```

```
/* main.c内定義 */
```

```
void H8init(void);
```

```
int SetLED(int no,int onoff);
```

```
int GetSW(int no);

#endif

/* 表示を表す列挙体宣言 */

enum PrintF

{

    Panel,

    ClsPnl,

    InputCommand,

    Monitor

};

/* 画面クリア */

void Clear(void);

/* 表示を表す関数のプロトタイプ宣言 */

void PrintF(int mode, char *str);

#ifdef USE_LINUX

int kbhit(void);

#endif
```

```
/* Panel.c */
```

```
#include "C.h"
```

```
#include "Panel.h"
```

```
/* 画面クリア */
```

```
void Clear(void)
```

```
{
```

```
#ifndef USE_BCC
```

```
    static char buff[64];
```

```
    sprintf(buff,"%f");
```

```
    PrintLCD(buff);
```

```
#else
```

```
    CLEAR;
```

```
#endif
```

```
}
```

```
/* 表示のための関数 */
```

```
void PrintF(int mode, char *str)
```

```
{
```

```
#ifndef USE_BCC
```

```
    static char buff[64];
```

```
#endif
```

```
    switch(mode)
```

```
{
```

```
    case ClsPnl:
```

```
#ifndef USE_BCC
```

```
        Clear();
```



```
if(strcmp(str, "\n\r0") == 0)
{
    sprintf(str, "%s", "\n\rf");
}
sprintf(buff,"%s", str);
PrintSCI("%s",buff);
PrintLCD(str);
#endif

break;

case Panel:

#ifndef USE_BCC

if(strcmp(str, "\n\r0") == 0)
{
    sprintf(str, "%s", "\n\rf");
}

sprintf(buff,"%s", str);
PrintSCI("%s",buff);
PrintLCD(str);

#else

printf("%s", str);

#endif

break;

case InputCommand:

#ifdef USE_BCC

printf("%s", str);

#endif

break;

case Monitor:

#ifndef USE_BCC
```

```

sprintf(buff,"%s", str);
PrintSCI("%s",buff);
write_buff(buff,strlen(buff)+1);
#else
    printf("%s", str);
#endif
    break;
default:
    break;
}
return;
}

#ifdef USE_LINUX
int kbhit(void)
{
    struct termios oldt, newt;

    int ch;

    int oldf;

    tcgetattr(STDIN_FILENO, &oldt);
    newt = oldt;
    newt.c_lflag &= ~(ICANON | ECHO);
    tcsetattr(STDIN_FILENO, TCSANOW, &newt);
    oldf = fcntl(STDIN_FILENO, F_GETFL, 0);
    fcntl(STDIN_FILENO, F_SETFL, oldf | O_NONBLOCK);

    ch = getchar();
}

```

```
tcsetattr(STDIN_FILENO, TCSANOW, &oldt);
```

```
fcntl(STDIN_FILENO, F_SETFL, oldf);
```

```
if (ch != EOF) {
```

```
    ungetc(ch, stdin);
```

```
    return 1;
```

```
}
```

```
return 0;
```

```
}
```

```
#endif
```

```
/* Timer.h */
```

```
#ifndef Panel_h
```

```
#define Panel_h
```

```
#include "Panel.h"
```

```
#endif
```

```
/* 疑似スレッド定義 */
```

```
#ifdef USE_THREAD
```

```
/* 疑似スレッドに使用する定数の宣言 */
```

```
#define INITCLOCKNO -1
```

```
#define STOPCLOCKNO -2
```

```
/* 構造体宣言 */
```

```
typedef struct tag_Thread
```

```
{
```

```
    /* 疑似スレッドID */
```

```
    int ID;
```

```
    /* 指定開始時 */
```

```
    double preClock;
```

```
    /* woviClockがpreClockからsetClock秒増えたらRunを呼ぶ */
```

```
    double setClock;
```

```
    /* Runが呼ばれた回数を調べるために使用(countUpNextRunが呼ばれた回数) */
```

```
    long count;
```

```
    /* List機能 */
```

```
    struct tag_Thread *previous;
```

```
    struct tag_Thread *next;
```

```
}Thread;
```

```
/* 疑似メソッドとwovi用関数のプロトタイプ宣言 */  
/* 宣言の順番は以下の通り */  
  
#endif  
  
double getClock(void);  
  
void SleepMSec(long ms); /* ミリ秒待ち関数 */  
  
#ifdef USE_THREAD  
  
void nextRun(Thread *This, long ms);  
  
void countUpNextRun(Thread *This, long ms);  
  
void Run(Thread *This); /* main.cで内容を定義します */  
  
void Init(Thread *This); /* main.cで内容を定義します */  
  
void Destroy(Thread *This); /* main.cで内容を定義します */  
  
Thread *new_Thread(int id);  
  
void delete_(Thread *This);  
  
void Start(Thread *This);  
  
void Stop(Thread *This);  
  
int Thread_checkAllDelete(void);  
  
int Thread_checkStayAnother(void);  
  
Thread *Thread_getThread(int id);  
  
Thread *Thread_Start(int id);  
  
void Thread_Toggle(int id);  
  
/* タイマ関数 */  
  
void woviRun(void); /* Runを呼ぶタイミング */  
  
void wovilnit(void); /* タイマ初期化関数 */  
  
#ifndef USE_BCC  
  
void InitITU(void); /* タイマ割り込み用 */  
  
void InterruptITU0(void); /* タイマ割り込み用 */  
  
#endif  
  
#ifdef USE_LINUX
```

```
void wovi(double threadPerSec); /* タイマ関数 */  
  
#else  
  
void wovi(void); /* タイマ関数 */  
  
#endif  
  
void initWOVI(void); /* タイマ初期化関数 */  
  
#endif  
  
#ifdef USE_BCC  
  
void PrintCurrentTime(void); /* 現在日時表示 */  
  
void myDateTime(long *mydate, long *mytime); /* 現在日時取得 */  
  
#endif
```

```
/* Timer.c */
```

```
#include "C.h"
```

```
#include "Timer.h"
```

```
/* 時間を表す外部変数宣言 */
```

```
double woviClock;
```

```
/* 疑似スレッド定義 */
```

```
#ifdef USE_THREAD
```

```
/* wovi用疑似インスタンス宣言 */
```

```
Thread woviThreadFirst;
```

```
Thread woviThreadLast;
```

```
#endif
```

```
/* 時刻取得 */
```

```
double getClock(void)
```

```
{
```

```
    return woviClock;
```

```
}
```

```
/* ミリ秒待ち関数 */
```

```
void SleepMSec(long ms)
```

```
{
```

```
#ifndef USE_BCC
```

```
    double start;
```

```
    double set;
```

```
    double end;
```

```

start = getClock();
set = ((double) ms) / 1000;
end = start;
while(end < start + set)
{
    end = getClock();
}
#else
double cnt;
double set;
cnt = 0.0;
set = ((double) ms) / 1000;
while(cnt < set)
{
    cnt += 1.0 / SLEEP_PER_SEC;
}
#endif
return;
}

#ifdef USE_THREAD
/* スレッドのvoid Sleep(int ms)の代用 */
void nextRun(Thread *This, long ms)
{
    This->preClock = getClock();
    This->setClock = (((double) ms) / 1000);
    return;
}

```



```
/* スレッドのvoid Sleep(int ms)の代用 */  
void countUpNextRun(Thread *This, long ms)  
{  
    nextRun(This, ms);  
    This->count++;  
}
```

```
#ifndef USE_BCC
```

```
Thread th101;
```

```
Thread th102;
```

```
Thread th111;
```

```
Thread th112;
```

```
Thread th113;
```

```
Thread th114;
```

```
Thread th119;
```

```
Thread th120;
```

```
Thread th121;
```

```
Thread th122;
```

```
Thread th123;
```

```
Thread th130;
```

```
Thread th131;
```

```
Thread th141;
```

```
Thread th142;
```

```
Thread th143;
```

```
Thread th144;
```

```
Thread th145;
```

```
#endif
```

```
/* スレッドのコンストラクタの代用 */
```

```
Thread *new_Thread(int id)
{
    Thread *List;
    Thread *new_List;
    List = &woviThreadFirst;
    while(List->next->next != NULL)
    {
        List = List->next;
    }
#ifdef USE_BCC
    if(id == 1)
    {
        new_List = &th101;
    }
    else if(id == 2)
    {
        new_List = &th102;
    }
    else if(id == 11)
    {
        new_List = &th111;
    }
    else if(id == 12)
    {
        new_List = &th112;
    }
    else if(id == 13)
    {
        new_List = &th113;
    }
#endif
}
```

```
}  
else if(id == 14)  
{  
    new_List = &th114;  
}  
else if(id == 19)  
{  
    new_List = &th119;  
}  
else if(id == 20)  
{  
    new_List = &th120;  
}  
else if(id == 21)  
{  
    new_List = &th121;  
}  
else if(id == 22)  
{  
    new_List = &th122;  
}  
else if(id == 23)  
{  
    new_List = &th123;  
}  
else if(id == 30)  
{  
    new_List = &th130;
```

```
}  
else if(id == 31)  
{  
    new_List = &th131;  
}  
else if(id == 41)  
{  
    new_List = &th141;  
}  
else if(id == 42)  
{  
    new_List = &th142;  
}  
else if(id == 43)  
{  
    new_List = &th143;  
}  
else if(id == 44)  
{  
    new_List = &th144;  
}  
else if(id == 45)  
{  
    new_List = &th145;  
}  
  
#endif  
  
#ifdef USE_BCC  
    new_List = (Thread *)calloc(1, sizeof(Thread));  
  
#endif
```

```

if(new_List == NULL)
{
    Printf(Pannel, "¥n");
    Printf(Pannel, "calloc failed");
    return NULL;
}

new_List->previous = List;
new_List->next = List->next;
new_List->next->previous = new_List;
List->next = new_List;
new_List->preClock = INITCLOCKNO;
new_List->setClock = 0;
new_List->ID = id;
new_List->count = 0;
/* スレッドのvoid init(void)の代用 */
Init(new_List);
return new_List;
}

/* スレッドのデストラクタの代用 */
void delete_(Thread *This)
{
    Destroy(This);
    This->previous->next = This->next;
    This->next->previous = This->previous;
#ifdef USE_BCC
    free(This);
#endif
    return;
}

```

```
}

/* スレッドのvoid start(void)の代用 */
```

```
void Start(Thread *This)
```

```
{
    woviClock = getClock();
    This->preClock = woviClock;
    return;
}
```

```
/* スレッドのvoid stop(void)の代用 */
```

```
void Stop(Thread *This)
```

```
{
    This->preClock = STOPCLOCKNO;
    return;
}
```

```
int Thread_checkAllDelete(void)
```

```
{
    if(woviThreadFirst.next->next == NULL)
    {
        return OK;
    }
    else
    {
        return NG;
    }
}
```

```

int Thread_checkStayAnother(void)
{
    Thread *checkthread = woviThreadFirst.next->next;
    int i = 0;
    while(checkthread != NULL)
    {
        checkthread = checkthread->next;
        i = i + 1;
    }
    return i;
}

```

```

Thread *Thread_getThread(int id)
{
    Thread *th;

    if(woviThreadFirst.next->next == NULL)
    {
        return NULL;
    }
    else
    {
        th = woviThreadFirst.next;
        do
        {
            if(th->ID == id)
            {
                return th;
            }
        }
    }
}

```

```
    }
    else
    {
        th = th->next;
    }
}while(th->next != NULL);
}
return NULL;
}
```

Thread *Thread_Start(int id)

```
{
    Thread *th;

    th = Thread_getThread(id);
    if(th == NULL)
    {
        th = new_Thread(id);
        Start(th);
    }
    else if(th->preClock == STOPCLOCKNO)
    {
        Start(th);
    }
    return th;
}
```

void Thread_Toggle(int id)

```
{
```



```

Thread *th;

th = Thread_getThread(id);
if(th == NULL)
{
    th = new_Thread(id);
    Start(th);
}
else if(th->preClock == STOPCLOCKNO)
{
    Start(th);
}
else
{
    delete_(th);
}
return;
}

```

/ タイマ関数 */*

/ Runを呼ぶタイミング */*

```
void woviRun(void)
```

```

{
    Thread *List;
    Thread *next_List;
    List = &woviThreadFirst;
    List = List->next;
    while(List->next != NULL)
    {

```

```

next_List = List->next;

if((List->preClock != INITCLOCKNO) && (List->preClock != STOPCLOCKNO))
{
    if(woviClock >= List->preClock + List->setClock)
    {
        List->preClock = woviClock;
        /* スレッドのvoid run(void)の代用 */
        Run(List);
    }
}

List = next_List;
}

return;
}

```

/* タイマ初期化関数 */

/* 関数main の冒頭で、 */

/* スレッド構造体リストの両端を初期化します。 */

void wovilnit(void)

```

{
    woviThreadFirst.previous = NULL;
    woviThreadFirst.next = &woviThreadLast;
    woviThreadLast.previous = &woviThreadFirst;
    woviThreadLast.next = NULL;
    return;
}

```

#ifndef USE_BCC

/* タイマ割り込み用 */

```

/* 関数main の冒頭で、 */
/* タイマ割り込み の専用設定をして、 */
/* タイマ0割り込み を立ち上げます。 */
void InitITU(void)
{
    ITU.TSTR = 0x01; /* timer 0 enable */
    ITU.TSNC = 0;
    ITU.TMDR = 0x0;
    ITU.TFCR = 0x0;
    ITU.TOER = 0x0;
    ITU.TOCR = 0xff;
    ITU0.TCR = 0x00; /* 分周なし */
    ITU0.TIOR = 0x88;
    ITU0.TIER = 0x04; /* オーバーフロー割り込み許可 */
    /* AKI-H8 3052F USB の演算速度は 25MHz なので、 */
    /* 25Kカウント すると、 1ms です。 */
    ITU0.TCNT = 0xffff - 25000; /* 1 msec interval */
    ITU0.GRA = 0;
    ITU0.GRB = 0;
}

/* タイマ割り込み用 */
/* 周りの関数が 関数main から呼び出されているのに、 */
/* この関数だけは、 asmfile.src の タイマ0割り込み から */
/* 直接呼び出されています。 */
void InterruptITU0(void)
{
    ITU0.TSR &= 0xfb;
}

```

```

/* AKI-H8 3052F USB の演算速度は 25MHz なので、 */
/* 25Kカウント すると、 1ms です。 */
ITU0.TCNT = 0xffff - 25000; /* 1 msec interval */
/* woviClock はシステムリセット時からの秒数時計です。 */
woviClock += 0.001;
return;
}

#endif

/* タイマ関数 */
#ifdef USE_LINUX
void wovi(double threadPerSec)
#else
void wovi(void)
#endif
{
#ifdef USE_BCC
    /* woviClock は秒数時計 */
#endif
#ifdef USE_LINUX
    /* threadPerSec で受け取る数値が1に近づく程、 */
    /* スピードが上がります。 */
    /* threadPerSec で受け取る数値が大きくなる程、 */
    /* スピード下がります。 */
    /* タイマ割り込み を使用できない時に */
    /* threadPerSec を使用します。 */
    woviClock += 1.0 / threadPerSec;
#else
    /* もしくは、 <time.h> の clock() を使用します。 */
    woviClock = (double) (clock() / CLOCKS_PER_SEC);

```

```
#endif

#endif

    woviRun(); /* スレッドのためのRunを呼ぶタイミング */

    return;

}
```

```
/* タイマ初期化関数 */
```

```
void initWOVI(void)
```

```
{

    /* 関数main の冒頭で、 */

    /* 秒数時計woviClock を */

    /* 0.0秒 で初期化します。 */

    woviClock = 0.0;

    /* タイマ割り込み用 */
```

```
#ifndef USE_BCC
```

```
    /* タイマ0割り込み を立ち上げます。 */

    InitITU();

    /* asmfile.src の 割り込み許可ラベル を呼びます。 */

    EnableInterrupt();
```

```
#endif
```

```
    /* スレッド構造体リストの両端を初期化します。 */

    wovilnit();

    return;
```

```
}
```

```
#endif
```

```
#ifdef USE_BCC
```

```
/* 現在日時表示 */
```

```
void PrintCurrentTime(void)
```

```
{  
  
time_t timer;  
  
struct tm *t_st;  
  
/* 現在時刻の取得 */  
  
time(&timer);  
  
/* 現在時刻を構造体に変換 */  
  
t_st = localtime(&timer);  
  
printf("%d", t_st->tm_year+1900);  
if(t_st->tm_mon+1 < 10)  
{  
    printf("0%d", t_st->tm_mon+1);  
}  
else  
{  
    printf("%d", t_st->tm_mon+1);  
}  
if(t_st->tm_mday < 10)  
{  
    printf("0%d", t_st->tm_mday);  
}  
else  
{  
    printf("%d", t_st->tm_mday);  
}  
  
printf(" ");  
  
if(t_st->tm_hour < 10)
```

```

{
    printf("0%d", t_st->tm_hour);
}
else
{
    printf("%d", t_st->tm_hour);
}
if(t_st->tm_min < 10)
{
    printf("0%d", t_st->tm_min);
}
else
{
    printf("%d", t_st->tm_min);
}
if(t_st->tm_sec < 10)
{
    printf("0%d", t_st->tm_sec);
}
else
{
    printf("%d", t_st->tm_sec);
}

return;
}

void myDateTime(long *mydate, long *mytime)
{

```

```
time_t timer;

struct tm *t_st;

/* 現在時刻の取得 */
time(&timer);

/* 現在時刻を構造体に変換 */
t_st = localtime(&timer);

*mydate = t_st->tm_year + 1900;
*mydate *= 100;
*mydate += (t_st->tm_mon + 1);
*mydate *= 100;
*mydate += t_st->tm_mday;

*mytime = t_st->tm_hour;
*mytime *= 100;
*mytime += t_st->tm_min;
*mytime *= 100;
*mytime += t_st->tm_sec;

return;

}

#endif
```



```
/* EV_Time.h */
```

```
#ifndef Panel_h
```

```
#define Panel_h
```

```
#include "Panel.h"
```

```
#endif
```

```
#ifndef Timer_h
```

```
#define Timer_h
```

```
#include "Timer.h"
```

```
#endif
```

```
#define OPENTIMEOUT 10
```

```
/*=====
```

```
時間を表す構造体
```

```
=====*/
```

```
struct EV_Time{
```

```
    double TimeTemp;
```

```
    double *p_TimeTemp;
```

```
    int Permit;
```

```
    int *p_Permit;
```

```
    int tmpTimeSafety;
```

```
    Thread *th;
```

```
};
```

```
/*=====
```

```
時間を表すプロトタイプ宣言
```

=====*/

```
void EV_Time(struct EV_Time *This, Thread *th);  
void SetCurrentTime(struct EV_Time *This);  
int GetCurrentTime(struct EV_Time *This);  
void WaitSecond(struct EV_Time *This, int num_Second);  
void SetPermit(struct EV_Time *This, int P);  
int GetPermit(struct EV_Time *This);  
void Checkfmove(int *p_check, int *p_fmove, int tmp);  
void Wait_ms(struct EV_Time *This, int num);
```

```
/* EV_Time.c */
```

```
#include "C.h"
```

```
#include "EV_Time.h"
```

```
/*=====
```

```
時間を表す関数
```

```
=====*/
```

```
void EV_Time(struct EV_Time *This, Thread *th)
```

```
{
```

```
    This->TimeTemp = 0;
```

```
    This->p_TimeTemp = &This->TimeTemp;
```

```
    SetCurrentTime(This);
```

```
    This->p_Permit = &This->Permit;
```

```
    This->Permit = OFF;
```

```
    This->tmpTimeSafety = 0;
```

```
    This->th = th;
```

```
    /* 戻る */
```

```
    return;
```

```
}
```

```
void SetCurrentTime(struct EV_Time *This)
```

```
{
```

```
    This->p_TimeTemp = &This->TimeTemp;
```

```
    *This->p_TimeTemp = getClock();
```

```
    /* 戻る */
```

```

    return;
}

int GetCurrentTime(struct EV_Time *This)
{
    This->p_TimeTemp = &This->TimeTemp;
    return (int) (getClock() - *This->p_TimeTemp);
}

void WaitSecond(struct EV_Time *This, int num_Second)
{
    nextRun(This->th, (num_Second * 1000));
    /* 戻る */
    return;
}

void SetPermit(struct EV_Time *This, int P)
{
    This->p_Permit = &This->Permit;
    if(P == ON) This->Permit = ON;
    else if(P == OFF) This->Permit = OFF;

    /* 戻る */
    return;
}

int GetPermit(struct EV_Time *This)
{
    This->p_Permit = &This->Permit;

```

```
    return This->Permit;
}

void Checkfmove(int *p_check, int *p_fmove, int tmp)
{
    if(*p_check != tmp){
        *p_fmove = OFF;
        *p_check = tmp;
    }

    /* 戻る */
    return;
}
```

```
void Wait_ms(struct EV_Time *This, int num){

    nextRun(This->th, num);

    /* 戻る */
    return;
}
```

```
/* EV_File.h */
```

```
#ifndef Panel_h
```

```
#define Panel_h
```

```
#include "Panel.h"
```

```
#endif
```

```
/*=====
```

```
   ファイルを表す構造体
```

```
=====*/
```

```
#ifdef NOTUSE_FILES
```

```
typedef struct tag_Handle_EV_Status
```

```
{
```

```
    char safety;
```

```
    char *p_limit;
```

```
    char limit[9];
```

```
    char motor;
```

```
    char command;
```

```
    char permitcommand;
```

```
    char permitturnopen;
```

```
}Handle_EV_Status;
```

```
typedef struct tag_EV_Status
```

```
{
```

```
    Handle_EV_Status *p_status;
```

```
}EV_Status;
```

```
#endif
```

```
/* ファイルストリーム */
```

```
struct EV_File
```

```
{
```

```
    FILE *fp;
```

```
};
```

```
/*=====
```

```
   ファイルを表すプロトタイプ宣言
```

```
=====*/
```

```
#ifndef NOTUSE_FILES
```

```
void new_EV_Status(EV_Status *This);
```

```
#endif
```

```
void EV_File(struct EV_File *This);
```

```
int Write(struct EV_File *This, char *filename, char ch);
```

```
int WriteString(struct EV_File *This, char *filename, char *str);
```

```
int Read(struct EV_File *This, char *filename, char *p_ch);
```

```
int ReadString(struct EV_File *This, char *filename, char *str, int strlength);
```

```
int PermitCommand_Read(struct EV_File *This, char *p_PermitCommand);
```

```
int PermitCommand_Write(struct EV_File *This, char PermitCommand);
```

```
int Command_Read(struct EV_File *This, char *p_Command);
```

```
int Command_Write(struct EV_File *This, char Command);
```

```
int PermitTurnOpen_Read(struct EV_File *This, char *p_PermitTurnOpen);
```

```
int PermitTurnOpen_Write(struct EV_File *This, char PermitTurnOpen);
```

```
void Motor_Write(struct EV_File *This, char Motor);
```

```
char Motor_Read(struct EV_File *This);
```

```
void Limit_Read(struct EV_File *This, char *str);
```

```
/* EV_File.c */
```

```
#include "C.h"
```

```
#include "EV_File.h"
```

```
/*=====
```

```
ファイル不使用時大域オブジェクト宣言
```

```
=====*/
```

```
#ifdef NOTUSE_FILES
```

```
Handle_EV_Status status;
```

```
#endif
```

```
/*=====
```

```
ファイルを表す関数
```

```
=====*/
```

```
#ifdef NOTUSE_FILES
```

```
void new_EV_Status(EV_Status *This)
```

```
{
```

```
    This->p_status = &status;
```

```
    status.safety = 'r';
```

```
    status.p_limit = &status.limit[0];
```

```
    strcpy(status.p_limit, "yynnyynn¥0");
```

```
    status.motor = 's';
```

```
    status.command = 'N';
```

```
    status.permitcommand = 'N';
```

```
    status.permitturnopen = 'N';
```

```
}
```

```
#endif
```



```

void EV_File(struct EV_File *This)
{
    /* 初期値 */
    This->fp = NULL;

    /* 戻る */
    return;
}

#ifdef NOTUSE_FILES

int Write(struct EV_File *This, char *filename, char ch)
{
    switch(filename[0])
    {
        case 'S':
            status.safety = ch;
            break;
        case 'M':
            status.motor = ch;
            break;
        case 'C':
            status.command = ch;
            break;
        case 'P':
            switch(filename[6])
            {
                case 'C':
                    status.permitcommand = ch;

```

```

        break;
    case 'T':
        status.permitturnopen = ch;
        break;
    default:
        break;
    }
    break;
default:
    break;
}
return OK;
}
#else
int Write(struct EV_File *This, char *filename, char ch)
{
    int Ret = OK;
    if((This->fp = fopen(filename, "w")) == NULL){
        Ret = NG;
    }
    else if(fputc((int) ch, This->fp) == ch){
        fclose(This->fp);
        Ret = OK;
    }
    else{
        fclose(This->fp);
        Ret = NG;
    }
    return Ret;
}

```

```

}

#endif

#ifdef NOTUSE_FILES

int WriteString(struct EV_File *This, char *filename, char *str)
{
    switch(filename[0])
    {
        case 'L':
            status.p_limit = &status.limit[0];
            strcpy(status.p_limit, str);
            break;
        default:
            break;
    }
    return OK;
}

#else

/* 文字列書き込み */

int WriteString(struct EV_File *This, char *filename, char *str)
{
    int Ret = OK;
    if((This->fp = fopen(filename, "w")) == NULL){
        Ret = NG;
    }

    /* ¥nは追記されない */
    else if(fputs(str, This->fp) >= 0){
        fclose(This->fp);
        /* 書き込み成功 */
    }
}

```

```

    Ret = OK;
}
else{
    fclose(This->fp);
    /* 書き込み失敗 */
    Ret = NG;
}
return Ret;
}
#endif

#ifdef NOTUSE_FILES
int Read(struct EV_File *This, char *filename, char *p_ch)
{
    switch(filename[0])
    {
    case 'S':
        *p_ch = status.safety;
        break;
    case 'M':
        *p_ch = status.motor;
        break;
    case 'C':
        *p_ch = status.command;
        break;
    case 'P':
        switch(filename[6])
        {

```

```
case 'C':
    *p_ch = status.permitcommand;
    break;
case 'T':
    *p_ch = status.permitturnopen;
    break;
default:
    break;
}
```

```
break;
```

```
default:
```

```
break;
```

```
}
```

```
return OK;
```

```
}
```

```
#else
```

```
int Read(struct EV_File *This, char *filename, char *p_ch)
```

```
{
```

```
int Ret = OK;
```

```
if((This->fp = fopen(filename, "r")) == NULL){
```

```
Ret = NG;
```

```
}
```

```
else if((*p_ch = fgetc(This->fp)) == EOF){
```

```
fclose(This->fp);
```

```
Ret = ONE_MORE_TIME;
```

```
}
```

```
else if(*p_ch == '¥n'){
```

```
fclose(This->fp);
```

```
Ret = ONE_MORE_TIME;
```

```

}
else if(*p_ch == 'N'){
    fclose(This->fp);
    Ret = ONE_MORE_TIME;
}
else{
    fclose(This->fp);
    Ret = OK;
}
return Ret;
}
#endif

#ifdef NOTUSE_FILES

int ReadString(struct EV_File *This, char *filename, char *str, int strlength)
{
    switch(filename[0])
    {
        case 'L':
            status.p_limit = &status.limit[0];
            strcpy(str, status.p_limit);
            break;
        default:
            break;
    }
    return OK;
}

#else

int ReadString(struct EV_File *This, char *filename, char *str, int strlength)

```

```

{
    int Ret = OK;
    if((This->fp = fopen(filename, "r")) == NULL){
        Ret = NG;
    }
    else if(fgets(str, strlen, This->fp) == NULL){
        fclose(This->fp);
        Ret = ONE_MORE_TIME;
    }
    else if(str[0] == '\n'){
        fclose(This->fp);
        Ret = ONE_MORE_TIME;
    }
    else if(str[0] == 'N'){
        fclose(This->fp);
        Ret = ONE_MORE_TIME;
    }
    else{
        fclose(This->fp);
        Ret = OK;
    }
    return Ret;
}
#endif

```

```

int PermitCommand_Read(struct EV_File *This, char *p_PermitCommand)
{
    int Ret = OK;
    switch(Read(This, "PermitCommand.txt", p_PermitCommand)){

```

```
case NG:
    Printf(ClsPnl, "¥nReading Error");
    Ret = NG;
    break;
```

```
case ONE_MORE_TIME:
    Ret = ONE_MORE_TIME;
    break;
```

```
default:
    Ret = OK;
    break;
```

```
}
```

```
return Ret;
```

```
}
```

```
int PermitCommand_Write(struct EV_File *This, char PermitCommand)
```

```
{
```

```
int Ret = OK;
```

```
switch(Write(This, "PermitCommand.txt¥0", PermitCommand)){
```

```
case NG:
```

```
    Printf(ClsPnl, "¥nWriting Error");
```

```
    Ret = NG;
```

```
    break;
```

```
case OK:
```

```
    Ret = OK;
```

```
    break;
```

```
default:
```

```
    Ret = NG;
```

```
    break;
```



```

    }
    return Ret;
}

int Command_Read(struct EV_File *This, char *p_Command)
{
    int Ret = OK;
    switch(Read(This, "Command.txt¥0", p_Command)){
    case NG:
        Printf(ClsPnl, "¥nReading Error");
        Ret = NG;
        break;
    case ONE_MORE_TIME:
        Ret = ONE_MORE_TIME;
        break;
    default:
        Ret = OK;
        break;
    }
    return Ret;
}

```

```

int Command_Write(struct EV_File *This, char Command)
{
    int Ret = OK;
    switch(Write(This, "Command.txt¥0", Command)){
    case NG:
        Printf(ClsPnl, "¥nWriting Error");
        Ret = NG;
    }
}

```

```
        break;
case OK:
    Ret = OK;
    break;
default:
    Ret = NG;
    break;
}
return Ret;
}
```

```
int PermitTurnOpen_Read(struct EV_File *This, char *p_PermitTurnOpen)
{
    int Ret = OK;
    switch(Read(This, "PermitTurnOpen.txt¥0", p_PermitTurnOpen)){
    case NG:
        Printf(ClsPnl, "¥nReading Error");
        Ret = NG;
        break;
    case ONE_MORE_TIME:
        Ret = ONE_MORE_TIME;
        break;
    default:
        Ret = OK;
        break;
    }
    return Ret;
}
```

```

int PermitTurnOpen_Write(struct EV_File *This, char PermitTurnOpen)
{
    int Ret = OK;
    switch(Write(This, "PermitTurnOpen.txt¥0", PermitTurnOpen)){
    case NG:
        Printf(ClsPnl, "¥nWriting Error");
        Ret = NG;
        break;
    case OK:
        Ret = OK;
        break;
    default:
        Ret = NG;
        break;
    }
    return Ret;
}

```

```

void Motor_Write(struct EV_File *This, char Motor)
{
    switch(Write(This, "Motor.txt¥0", Motor)){
    case NG:
        Printf(ClsPnl, "¥nWriting Error");
        break;
    case OK:
        return;
        break;
    default:
        break;
    }
}

```

```

}

/* 戻る */
return;
}

char Motor_Read(struct EV_File *This)
{
    char ch;
    char *p_ch;
    ch = '\0';
    p_ch = &ch;

    switch(Read(This, "Motor.txt\0", p_ch)){
    case NG:
        break;
    case ONE_MORE_TIME:
        return '\0';
        break;
    case OK:
        return ch;
        break;
    default:
        break;
    }
    return '\0';
}

```

```
void Limit_Read(struct EV_File *This, char *str)
{
    switch(ReadString(This, "Limit.txt¥0", str, 9)){
    case NG:
        Printf(ClsPnl, "¥nReading Error");
        break;
    case ONE_MORE_TIME:
        return;
        break;
    default:
        return;
        break;
    }

    /* 戻る */
    return;
}
```

```
/* EV_UpDown.h */
```

```
#ifndef Panel_h
```

```
#define Panel_h
```

```
#include "Panel.h"
```

```
#endif
```

```
#ifndef Timer_h
```

```
#define Timer_h
```

```
#include "Timer.h"
```

```
#endif
```

```
#ifndef EV_Time_h
```

```
#define EV_Time_h
```

```
#include "EV_Time.h"
```

```
#endif
```

```
#ifndef EV_File_h
```

```
#define EV_File_h
```

```
#include "EV_File.h"
```

```
#endif
```

```
/*=====
```

```
上昇下降を表す構造体
```

```
=====*/
```

```
struct Position
```

```
{
```

```
    int m_UNSL;
```

```
int m_UNST;
int m_UPSL;
int m_UPST;
/* 下降減速位置 */
int *p_UnderSlow;
/* 下降停止位置 */
int *p_UnderStop;
/* 上昇減速位置 */
int *p_UpperSlow;
/* 上昇停止位置 */
int *p_UpperStop;
/* Sleep用 */
int fstop;
int *p_fstop;
int fmove;
int *p_fmove;
};
```

```
/* 上昇 */
```

```
struct UpMotor
```

```
{
    struct EV_File SF;
    struct EV_File MF;
};
```

```
/* 下降 */
```

```
struct DownMotor
```

```
{
    struct EV_File SF;
```

```

    struct EV_File MF;
};

/* エレベーターの位置仮想ログ */
struct WaitPositionChangeLog
{
    struct EV_File LF;
    char strLimit[9];
    char *p_strLimit;
    /* Sleep用 */
    int tmp;
};

/*=====
  上昇下降を表すプロトタイプ宣言
  =====*/

void Position(struct Position *This);
/* 上昇 */
void UpMotor(struct UpMotor *This);
void OnUpMotor(struct UpMotor *This, struct Position *P, char *p_Safety);
/* 下降 */
void DownMotor(struct DownMotor *This);
void OnDownMotor(struct DownMotor *This, struct Position *P, char *p_Safety);
/* 経過時間 */
void WaitPositionChangeLog(struct WaitPositionChangeLog *This);
void OnInitWaitPositionChangeLog(struct WaitPositionChangeLog *This, struct Position *P);
void OnWaitUpPositionChangeLog(struct WaitPositionChangeLog *This, struct Position *P);
void OnWaitDownPositionChangeLog(struct WaitPositionChangeLog *This, struct Position *P);
/* 上昇 */

```



```
void Up(struct Position *P, struct UpMotor UPMT, struct WaitPositionChangeLog WPCL, char  
*p_Safety);
```

```
/* 下降 */
```

```
void Down(struct Position *P, struct DownMotor DNMT, struct WaitPositionChangeLog WPCL, char  
*p_Safety);
```

```
/* EV_UpDown.c */
```

```
#include "C.h"
```

```
#include "EV_UpDown.h"
```

```
/*=====
```

```
上昇下降を表す関数
```

```
=====*/
```

```
/*
```

```
 * Position
```

```
*/
```

```
void Position(struct Position *This)
```

```
{
```

```
    /* 初期値 */
```

```
    This->m_UNSL = OFF;
```

```
    This->m_UNST = OFF;
```

```
    This->m_UPSL = OFF;
```

```
    This->m_UPST = OFF;
```

```
    This->p_UnderSlow = &This->m_UNSL;
```

```
    This->p_UnderStop = &This->m_UNST;
```

```
    This->p_UpperSlow = &This->m_UPSL;
```

```
    This->p_UpperStop = &This->m_UPST;
```

```
    /* Sleep用 */
```

```
    This->fstop = OFF;
```

```
    This->p_fstop = &This->fstop;
```

```
    This->fmove = OFF;
```

```
    This->p_fmove = &This->fmove;
```

```

    /* 戻る */
    return;
}

/*
 * UpMotor
 */
void UpMotor(struct UpMotor *This)
{
    EV_File(&This->SF);
    EV_File(&This->MF);

    /* 戻る */
    return;
}

/*
 * OnUpMotor
 */
/* 上昇 */
void OnUpMotor(struct UpMotor *This, struct Position *P, char *p_Safety)
{
    if(*P->p_UpperStop == ON){
        /* Sleep用 */
        if(*P->p_fstop == OFF){
            *P->p_fstop = ON;
            /* 現在実行中の命令を外部に報告 */
            Motor_Write(&This->MF, 's');
            Printf(ClsPnl, "STOP");
        }
    }
}

```

```

}
if(*p_Safety == 'Y'){
    *p_Safety = 'r';
    Write(&This->SF, "Safety.txt¥0", 'r');
}
}
else if(*P->p_UpperSlow == ON){
    /* Sleep用 */
    *P->p_fstop = OFF;
    if(*P->p_fmove == OFF){
        *P->p_fmove = ON;
        /* 現在実行中の命令を外部に報告 */
        Motor_Write(&This->MF, 'u');
        Printf(ClsPnl, "UP");
    }
    else if(*p_Safety == 'Y'){
        Motor_Write(&This->MF, 'j');
        *p_Safety = 'r';
        Write(&This->SF, "Safety.txt¥0", 'r');
    }
}
}
else if(*P->p_UnderSlow == OFF){
    /* Sleep用 */
    *P->p_fstop = OFF;
    if(*P->p_fmove == OFF){
        *P->p_fmove = ON;
        /* 現在実行中の命令を外部に報告 */
        Motor_Write(&This->MF, 'U');
        Printf(ClsPnl, "UP Speedy");
    }
}
}

```

```

}

else if(*p_Safety == 'Y'){
    Motor_Write(&This->MF, 'j');
    *p_Safety = 'r';
    Write(&This->SF, "Safety.txt¥0", 'r');
}

}

else if(*P->p_UnderStop == OFF){
    /* Sleep用 */
    *P->p_fstop = OFF;
    if(*P->p_fmove == OFF){
        *P->p_fmove = ON;
        /* 現在実行中の命令を外部に報告 */
        Motor_Write(&This->MF, 'u');
        Printf(ClsPnl, "UP");
    }
    else if(*p_Safety == 'Y'){
        Motor_Write(&This->MF, 'j');
        *p_Safety = 'r';
        Write(&This->SF, "Safety.txt¥0", 'r');
    }
}

else if(*P->p_UnderStop == ON){
    /* Sleep用 */
    *P->p_fstop = OFF;
    if(*P->p_fmove == OFF){
        *P->p_fmove = ON;
        /* 現在実行中の命令を外部に報告 */
        Motor_Write(&This->MF, 'j');

```

```
        Printf(ClsPnl, "UP Start");
    }
    if(*p_Safety == 'Y'){
        *p_Safety = 'r';
        Write(&This->SF, "Safety.txt¥0", 'r');
    }
}
```

```
/* 戻る */
```

```
return;
```

```
}
```

```
/*
```

```
* DownMotor
```

```
*/
```

```
void DownMotor(struct DownMotor *This)
```

```
{
```

```
    EV_File(&This->SF);
```

```
    EV_File(&This->MF);
```

```
/* 戻る */
```

```
return;
```

```
}
```

```
/*
```

```
* OnDownMotor
```

```
*/
```

```
/* 下降 */
```

```
void OnDownMotor(struct DownMotor *This, struct Position *P, char *p_Safety)
```

```
{
```

```
    if(*P->p_UnderStop == ON){
```

```
        /* Sleep用 */
```

```
        if(*P->p_fstop == OFF){
```

```
            *P->p_fstop = ON;
```

```
            /* 現在実行中の命令を外部に報告 */
```

```
            Motor_Write(&This->MF, 's');
```

```
            Printf(ClsPnl, "STOP");
```

```
        }
```

```
        if(*p_Safety == 'Y'){
```

```
            *p_Safety = 'r';
```

```
            Write(&This->SF, "Safety.txt¥0", 'r');
```

```
        }
```

```
    }
```

```
    else if(*P->p_UnderSlow == ON){
```

```
        /* Sleep用 */
```

```
        *P->p_fstop = OFF;
```

```
        if(*P->p_fmove == OFF){
```

```
            *P->p_fmove = ON;
```

```
            /* 現在実行中の命令を外部に報告 */
```

```
            Motor_Write(&This->MF, 'd');
```

```
            Printf(ClsPnl, "DOWN");
```

```
        }
```

```
        else if(*p_Safety == 'Y'){
```

```
            Motor_Write(&This->MF, 'k');
```

```
            *p_Safety = 'r';
```

```
            Write(&This->SF, "Safety.txt¥0", 'r');
```

```
        }
```

```

}
else if(*P->p_UpperSlow == OFF){
    /* Sleep用 */
    *P->p_fstop = OFF;
    if(*P->p_fmove == OFF){
        *P->p_fmove = ON;
        /* 現在実行中の命令を外部に報告 */
        Motor_Write(&This->MF, 'D');
        Printf(ClsPnl, "DOWN Speedy");
    }
    else if(*p_Safety == 'Y'){
        Motor_Write(&This->MF, 'k');
        *p_Safety = 'r';
        Write(&This->SF, "Safety.txt¥0", 'r');
    }
}
else if(*P->p_UpperStop == OFF){
    /* Sleep用 */
    *P->p_fstop = OFF;
    if(*P->p_fmove == OFF){
        *P->p_fmove = ON;
        /* 現在実行中の命令を外部に報告 */
        Motor_Write(&This->MF, 'd');
        Printf(ClsPnl, "DOWN");
    }
    else if(*p_Safety == 'Y'){
        Motor_Write(&This->MF, 'k');
        *p_Safety = 'r';
        Write(&This->SF, "Safety.txt¥0", 'r');
    }
}

```



```

    }
}
else if(*P->p_UpperStop == ON){
    /* Sleep用 */
    *P->p_fstop = OFF;
    if(*P->p_fmove == OFF){
        *P->p_fmove = ON;
        /* 現在実行中の命令を外部に報告 */
        Motor_Write(&This->MF, 'k');
        Printf(ClsPnl, "DOWN Start");
    }
    if(*p_Safety == 'Y'){
        *p_Safety = 'r';
        Write(&This->SF, "Safety.txt¥0", 'r');
    }
}

/* 戻る */
return;
}

/*
 * WaitPositionChangeLog
 */
/* エレベーターの位置仮想ログ */
void WaitPositionChangeLog(struct WaitPositionChangeLog *This)
{
    This->p_strLimit = &This->strLimit[0];
}

```

```

    /* 戻る */
    return;
}

/*
 * OnInitWaitPositionChangeLog
 */
/* エレベーターの位置仮想ログ */
void OnInitWaitPositionChangeLog(struct WaitPositionChangeLog *This, struct Position *P)
{
    /* センサ初期値 */
    /* リミットスイッチの読み込み */
    This->p_strLimit = &This->strLimit[0];
    Limit_Read(&This->LF, This->p_strLimit);
    This->tmp = ((This->strLimit[0] == 'y') ? ON : OFF);
    Checkfmove(P->p_UnderStop, P->p_fmove, This->tmp);
    This->tmp = ((This->strLimit[1] == 'y') ? ON : OFF);
    Checkfmove(P->p_UnderSlow, P->p_fmove, This->tmp);
    This->tmp = ((This->strLimit[2] == 'y') ? ON : OFF);
    Checkfmove(P->p_UpperSlow, P->p_fmove, This->tmp);
    This->tmp = ((This->strLimit[3] == 'y') ? ON : OFF);
    Checkfmove(P->p_UpperStop, P->p_fmove, This->tmp);

    /* 戻る */
    return;
}

/*

```

```
* OnWaitUpPositionChangeLog
```

```
*/
```

```
/* エレベーターの位置仮想ログ */
```

```
void OnWaitUpPositionChangeLog(struct WaitPositionChangeLog *This, struct Position *P)
```

```
{
```

```
    /* リミットスイッチの読み込み */
```

```
    This->p_strLimit = &This->strLimit[0];
```

```
    Limit_Read(&This->LF, This->p_strLimit);
```

```
    This->tmp = (This->strLimit[0] == 'y') ? ON : OFF;
```

```
    Checkfmove(P->p_UnderStop, P->p_fmove, This->tmp);
```

```
    This->tmp = (This->strLimit[1] == 'y') ? ON : OFF;
```

```
    Checkfmove(P->p_UnderSlow, P->p_fmove, This->tmp);
```

```
    This->tmp = (This->strLimit[2] == 'y') ? ON : OFF;
```

```
    Checkfmove(P->p_UpperSlow, P->p_fmove, This->tmp);
```

```
    This->tmp = (This->strLimit[3] == 'y') ? ON : OFF;
```

```
    Checkfmove(P->p_UpperStop, P->p_fmove, This->tmp);
```

```
    /* 戻る */
```

```
    return;
```

```
}
```

```
/*
```

```
* OnWaitDownPositionChangeLog
```

```
*/
```

```
/* エレベーターの位置仮想ログ */
```

```
void OnWaitDownPositionChangeLog(struct WaitPositionChangeLog *This, struct Position *P)
```

```
{
```

```
    /* リミットスイッチの読み込み */
```

```
    This->p_strLimit = &This->strLimit[0];
```

```
Limit_Read(&This->LF, This->p_strLimit);
This->tmp = (This->strLimit[0] == 'y') ? ON : OFF;
Checkfmove(P->p_UnderStop, P->p_fmove, This->tmp);
This->tmp = (This->strLimit[1] == 'y') ? ON : OFF;
Checkfmove(P->p_UnderSlow, P->p_fmove, This->tmp);
This->tmp = (This->strLimit[2] == 'y') ? ON : OFF;
Checkfmove(P->p_UpperSlow, P->p_fmove, This->tmp);
This->tmp = (This->strLimit[3] == 'y') ? ON : OFF;
Checkfmove(P->p_UpperStop, P->p_fmove, This->tmp);
```

```
/* 戻る */
```

```
return;
```

```
}
```

```
/*
```

```
* Up
```

```
*/
```

```
/* 上昇 */
```

```
void Up(struct Position *P, struct UpMotor UPMT, struct WaitPositionChangeLog WPCL, char
*p_Safety)
```

```
{
```

```
/* 到着まで */
```

```
if(*P->p_UpperStop == OFF){
```

```
    /* エレベーターの位置仮想ログ */
```

```
    OnWaitUpPositionChangeLog(&WPCL, P);
```

```
/* 上昇 */
```

```
    OnUpMotor(&UPMT, P, p_Safety);
```

```
/* エレベーターの位置仮想ログ */
```

```
OnWaitUpPositionChangeLog(&WPCL, P);
```

```
}
```

```
/* 戻る */
```

```
return;
```

```
}
```

```
/*
```

```
* Down
```

```
*/
```

```
/* 下降 */
```

```
void Down(struct Position *P, struct DownMotor DNMT, struct WaitPositionChangeLog WPCL, char  
*p_Safety)
```

```
{
```

```
/* 到着まで */
```

```
if(*P->p_UnderStop == OFF){
```

```
/* エレベーターの位置仮想ログ */
```

```
OnWaitDownPositionChangeLog(&WPCL, P);
```

```
/* 下降 */
```

```
OnDownMotor(&DNMT, P, p_Safety);
```

```
/* エレベーターの位置仮想ログ */
```

```
OnWaitDownPositionChangeLog(&WPCL, P);
```

```
}
```

```
/* 戻る */
```

```
return;
```

}

```
/* EV_OpenClose.h */
```

```
#ifndef Panel_h
```

```
#define Panel_h
```

```
#include "Panel.h"
```

```
#endif
```

```
#ifndef Timer_h
```

```
#define Timer_h
```

```
#include "Timer.h"
```

```
#endif
```

```
#ifndef EV_Time_h
```

```
#define EV_Time_h
```

```
#include "EV_Time.h"
```

```
#endif
```

```
#ifndef EV_File_h
```

```
#define EV_File_h
```

```
#include "EV_File.h"
```

```
#endif
```

```
#ifndef EV_UpDown_h
```

```
#define EV_UpDown_h
```

```
#include "EV_UpDown.h"
```

```
#endif
```

```
/*=====
```

開閉を表す構造体

=====*/

struct Door

```
{  
    int m_CLSL;  
    int m_CLST;  
    int m_OPSL;  
    int m_OPST;  
    /* 閉減速位置 */  
    int *p_CloserSlow;  
    /* 閉停止位置 */  
    int *p_CloserStop;  
    /* 開減速位置 */  
    int *p_OpennerSlow;  
    /* 開停止位置 */  
    int *p_OpennerStop;  
    /* Sleep用 */  
    int fstop;  
    int *p_fstop;  
    int fmove;  
    int *p_fmove;  
};
```

/* 開 */

struct OpenMotor

```
{  
    struct EV_File SF;  
    struct EV_File MF;  
};
```



```

/* 閉 */
struct CloseMotor
{
    struct EV_File SF;
    struct EV_File MF;
};

/* エレベーターの位置仮想ログ */
struct WaitDoorChangeLog
{
    struct EV_File TOF;
    char chTurnOpen;
    char *p_chTurnOpen;
    struct EV_File LF;
    char strLimit[9];
    char *p_strLimit;
    /* Sleep用 */
    int tmp;
};

/*=====
  開閉を表すプロトタイプ宣言
=====*/

void Door(struct Door *This);

/* 開 */
void OpenMotor(struct OpenMotor *This);
void OnOpenMotor(struct OpenMotor *This, struct Door *DR, char *p_Safety);

/* 閉 */

```

```
void CloseMotor(struct CloseMotor *This);

void OnCloseMotor(struct CloseMotor *This, struct Door *DR, char *p_Safety);

/* 經過時間 */

void WaitDoorChangeLog(struct WaitDoorChangeLog *This);

void OnInitWaitDoorChangeLog(struct WaitDoorChangeLog *This, struct Door *DR);

void OnWaitOpenDoorChangeLog(struct WaitDoorChangeLog *This, struct Door *DR);

void OnWaitCloseDoorChangeLog(struct WaitDoorChangeLog *This, struct Door *DR);

/* 開 */

void Open(struct Door *DR, struct OpenMotor OPMT, struct WaitDoorChangeLog WDCL, char
*p_Safety);

/* 閉 */

void Close(struct Door *DR, struct CloseMotor CLMT, struct WaitDoorChangeLog WDCL, char
*p_Safety);
```

```
/* EV_OpenClose.c */
```

```
#include "C.h"
```

```
#include "EV_OpenClose.h"
```

```
/*=====
```

```
  開閉を表す関数
```

```
=====*/
```

```
/*
```

```
 * Door
```

```
*/
```

```
void Door(struct Door *This)
```

```
{
```

```
    /* 初期値 */
```

```
    This->m_CLSL = OFF;
```

```
    This->m_CLST = OFF;
```

```
    This->m_OPST = OFF;
```

```
    This->m_OPST = OFF;
```

```
    This->p_CloserSlow = &This->m_CLSL;
```

```
    This->p_CloserStop = &This->m_CLST;
```

```
    This->p_OpennerSlow = &This->m_OPST;
```

```
    This->p_OpennerStop = &This->m_OPST;
```

```
    /* Sleep用 */
```

```
    This->fstop = OFF;
```

```
    This->p_fstop = &This->fstop;
```

```
    This->fmove = OFF;
```

```
    This->p_fmove = &This->fmove;
```

```

    /* 戻る */
    return;
}

/*
 * OpenMotor
 */
void OpenMotor(struct OpenMotor *This)
{
    EV_File(&This->SF);
    EV_File(&This->MF);

    /* 戻る */
    return;
}

/*
 * OnOpenMotor
 */
/* 開 */
void OnOpenMotor(struct OpenMotor *This, struct Door *DR, char *p_Safety)
{
    if(*DR->p_OpennerStop == ON){
        /* Sleep用 */
        if(*DR->p_fstop == OFF){
            *DR->p_fstop = ON;
            /* 現在実行中の命令を外部に報告 */
            Motor_Write(&This->MF, 's');
            Printf(ClsPnl, "STOP");
        }
    }
}

```

```

}

if(*p_Safety == 'Y'){
    *p_Safety = 'r';
    Write(&This->SF, "Safety.txt¥0", 'r');
}

}

else if(*DR->p_OpennerSlow == ON){
    /* Sleep用 */
    *DR->p_fstop = OFF;
    if(*DR->p_fmmove == OFF){
        *DR->p_fmmove = ON;
        /* 現在実行中の命令を外部に報告 */
        Motor_Write(&This->MF, 'o');
        Printf(ClsPnl, "OPEN");
    }
}

else if(*p_Safety == 'Y'){
    Motor_Write(&This->MF, 'h');
    *p_Safety = 'r';
    Write(&This->SF, "Safety.txt¥0", 'r');
}

}

else if(*DR->p_CloserSlow == OFF){
    /* Sleep用 */
    *DR->p_fstop = OFF;
    if(*DR->p_fmmove == OFF){
        *DR->p_fmmove = ON;
        /* 現在実行中の命令を外部に報告 */
        Motor_Write(&This->MF, 'O');
        Printf(ClsPnl, "OPEN Speedy");
    }
}

```

```

}

else if(*p_Safety == 'Y'){
    Motor_Write(&This->MF, 'h');
    *p_Safety = 'r';
    Write(&This->SF, "Safety.txt¥0", 'r');
}

}

else if(*DR->p_CloserStop == OFF){
    /* Sleep用 */
    *DR->p_fstop = OFF;
    if(*DR->p_fmove == OFF){
        *DR->p_fmove = ON;
        /* 現在実行中の命令を外部に報告 */
        Motor_Write(&This->MF, 'o');
        Printf(ClsPnl, "OPEN");
    }
    else if(*p_Safety == 'Y'){
        Motor_Write(&This->MF, 'h');
        *p_Safety = 'r';
        Write(&This->SF, "Safety.txt¥0", 'r');
    }
}

else if(*DR->p_CloserStop == ON){
    /* Sleep用 */
    *DR->p_fstop = OFF;
    if(*DR->p_fmove == OFF){
        *DR->p_fmove = ON;
        /* 現在実行中の命令を外部に報告 */
        Motor_Write(&This->MF, 'h');

```

```
        Printf(ClsPnl, "OPEN Start");
    }
    if(*p_Safety == 'Y'){
        *p_Safety = 'r';
        Write(&This->SF, "Safety.txt¥0", 'r');
    }
}
```

```
/* 戻る */
```

```
return;
```

```
}
```

```
/*
```

```
* CloseMotor
```

```
*/
```

```
void CloseMotor(struct CloseMotor *This)
```

```
{
```

```
    EV_File(&This->SF);
```

```
    EV_File(&This->MF);
```

```
/* 戻る */
```

```
return;
```

```
}
```

```
/*
```

```
* OnCloseMotor
```

```
*/
```

```
/* 閉 */
```

```

void OnCloseMotor(struct CloseMotor *This, struct Door *DR, char *p_Safety)
{
    if(*DR->p_CloserStop == ON){
        /* Sleep用 */
        if(*DR->p_fstop == OFF){
            *DR->p_fstop = ON;
            /* 現在実行中の命令を外部に報告 */
            Motor_Write(&This->MF, 's');
            Printf(ClsPnl, "STOP");
        }
        if(*p_Safety == 'Y'){
            *p_Safety = 'r';
            Write(&This->SF, "Safety.txt¥0", 'r');
        }
    }
    else if(*DR->p_CloserSlow == ON){
        /* Sleep用 */
        *DR->p_fstop = OFF;
        if(*DR->p_fmmove == OFF){
            *DR->p_fmmove = ON;
            /* 現在実行中の命令を外部に報告 */
            Motor_Write(&This->MF, 'c');
            Printf(ClsPnl, "CLOSE");
        }
        else if(*p_Safety == 'Y'){
            Motor_Write(&This->MF, 't');
            *p_Safety = 'r';
            Write(&This->SF, "Safety.txt¥0", 'r');
        }
    }
}

```



```

}
else if(*DR->p_OpennerSlow == OFF){
    /* Sleep用 */
    *DR->p_fstop = OFF;
    if(*DR->p_fmove == OFF){
        *DR->p_fmove = ON;
        /* 現在実行中の命令を外部に報告 */
        Motor_Write(&This->MF, 'C');
        Printf(ClsPnl, "CLOSE Speedy");
    }
    else if(*p_Safety == 'Y'){
        Motor_Write(&This->MF, 't');
        *p_Safety = 'r';
        Write(&This->SF, "Safety.txt¥0", 'r');
    }
}
else if(*DR->p_OpennerStop == OFF){
    /* Sleep用 */
    *DR->p_fstop = OFF;
    if(*DR->p_fmove == OFF){
        *DR->p_fmove = ON;
        /* 現在実行中の命令を外部に報告 */
        Motor_Write(&This->MF, 'c');
        Printf(ClsPnl, "CLOSE");
    }
    else if(*p_Safety == 'Y'){
        Motor_Write(&This->MF, 't');
        *p_Safety = 'r';
        Write(&This->SF, "Safety.txt¥0", 'r');
    }
}

```

```

    }
}
else if(*DR->p_OpennerStop == ON){
    /* Sleep用 */
    *DR->p_fstop = OFF;
    if(*DR->p_fmove == OFF){
        *DR->p_fmove = ON;
        /* 現在実行中の命令を外部に報告 */
        Motor_Write(&This->MF, 't');
        Printf(ClsPnl, "CLOSE Start");
    }
    if(*p_Safety == 'Y'){
        *p_Safety = 'r';
        Write(&This->SF, "Safety.txt¥0", 'r');
    }
}
}

/* 戻る */
return;
}

/*
 * WaitDoorChangeLog
 */
/* エレベーターの位置仮想ログ */
void WaitDoorChangeLog(struct WaitDoorChangeLog *This)
{
    This->p_chTurnOpen = &This->chTurnOpen;
    This->p_strLimit = &This->strLimit[0];
}

```

```

    /* 戻る */
    return;
}

/*
 * OnInitWaitDoorChangeLog
 */
/* エレベーターの位置仮想ログ */
void OnInitWaitDoorChangeLog(struct WaitDoorChangeLog *This, struct Door *DR)
{
    /* センサ初期値 */
    /* リミットスイッチの読み込み */
    This->p_strLimit = &This->strLimit[0];
    Limit_Read(&This->LF, This->p_strLimit);
    This->tmp = (This->strLimit[4] == 'y') ? ON : OFF;
    Checkmove(DR->p_CloserStop, DR->p_fmove, This->tmp);
    This->tmp = (This->strLimit[5] == 'y') ? ON : OFF;
    Checkmove(DR->p_CloserSlow, DR->p_fmove, This->tmp);
    This->tmp = (This->strLimit[6] == 'y') ? ON : OFF;
    Checkmove(DR->p_OpennerSlow, DR->p_fmove, This->tmp);
    This->tmp = (This->strLimit[7] == 'y') ? ON : OFF;
    Checkmove(DR->p_OpennerStop, DR->p_fmove, This->tmp);

    /* 戻る */
    return;
}

```

```

/*
 * OnWaitOpenDoorChangeLog
 */
/* エレベーターの位置仮想ログ */
void OnWaitOpenDoorChangeLog(struct WaitDoorChangeLog *This, struct Door *DR)
{
    /* リミットスイッチの読み込み */
    This->p_strLimit = &This->strLimit[0];
    Limit_Read(&This->LF, This->p_strLimit);
    This->tmp = (This->strLimit[4] == 'y') ? ON : OFF;
    Checkfmove(DR->p_CloserStop, DR->p_fmove, This->tmp);
    This->tmp = (This->strLimit[5] == 'y') ? ON : OFF;
    Checkfmove(DR->p_CloserSlow, DR->p_fmove, This->tmp);
    This->tmp = (This->strLimit[6] == 'y') ? ON : OFF;
    Checkfmove(DR->p_OpennerSlow, DR->p_fmove, This->tmp);
    This->tmp = (This->strLimit[7] == 'y') ? ON : OFF;
    Checkfmove(DR->p_OpennerStop, DR->p_fmove, This->tmp);

    /* 戻る */
    return;
}

/*
 * OnWaitCloseDoorChangeLog
 */
/* エレベーターの位置仮想ログ */
void OnWaitCloseDoorChangeLog(struct WaitDoorChangeLog *This, struct Door *DR)
{
    /* リミットスイッチの読み込み */

```

```
This->p_strLimit = &This->strLimit[0];
Limit_Read(&This->LF, This->p_strLimit);
This->tmp = (This->strLimit[4] == 'y') ? ON : OFF;
Checkfmove(DR->p_CloserStop, DR->p_fmove, This->tmp);
This->tmp = (This->strLimit[5] == 'y') ? ON : OFF;
Checkfmove(DR->p_CloserSlow, DR->p_fmove, This->tmp);
This->tmp = (This->strLimit[6] == 'y') ? ON : OFF;
Checkfmove(DR->p_OpennerSlow, DR->p_fmove, This->tmp);
This->tmp = (This->strLimit[7] == 'y') ? ON : OFF;
Checkfmove(DR->p_OpennerStop, DR->p_fmove, This->tmp);
```

```
/* 戻る */
```

```
return;
```

```
}
```

```
/*
```

```
* Open
```

```
*/
```

```
/* 開 */
```

```
void Open(struct Door *DR, struct OpenMotor OPMT, struct WaitDoorChangeLog WDCL, char
*p_Safety)
```

```
{
```

```
/* 到着まで */
```

```
if(*DR->p_OpennerStop == OFF)
```

```
{
```

```
/* エレベーターの位置仮想ログ */
```

```
OnWaitOpenDoorChangeLog(&WDCL, DR);
```

```
/* 開 */
```

```
OnOpenMotor(&OPMT, DR, p_Safety);
```

```
/* エレベーターの位置仮想ログ */
```

```
OnWaitOpenDoorChangeLog(&WDCL, DR);
```

```
}
```

```
/* 戻る */
```

```
return;
```

```
}
```

```
/*
```

```
* Close
```

```
*/
```

```
/* 閉 */
```

```
void Close(struct Door *DR, struct CloseMotor CLMT, struct WaitDoorChangeLog WDCL, char  
*p_Safety)
```

```
{
```

```
/* 到着まで */
```

```
if(*DR->p_CloserStop == OFF)
```

```
{
```

```
/* エレベーターの位置仮想ログ */
```

```
OnWaitCloseDoorChangeLog(&WDCL, DR);
```

```
/* 閉 */
```

```
OnCloseMotor(&CLMT, DR, p_Safety);
```

```
/* エレベーターの位置仮想ログ */
```

```
OnWaitCloseDoorChangeLog(&WDCL, DR);
```

```
}
```

```
/* 戻る */
```

```
return;
```

```
}
```

```
/* EV_Display.h */
```

```
#ifndef Panel_h
```

```
#define Panel_h
```

```
#include "Panel.h"
```

```
#endif
```

```
#ifndef Timer_h
```

```
#define Timer_h
```

```
#include "Timer.h"
```

```
#endif
```

```
#ifndef EV_Time_h
```

```
#define EV_Time_h
```

```
#include "EV_Time.h"
```

```
#endif
```

```
#ifndef EV_File_h
```

```
#define EV_File_h
```

```
#include "EV_File.h"
```

```
#endif
```

```
/*=====
```

```
シミュレータを表す関数のプロトタイプ宣言
```

```
=====*/
```

```
void Displnput(void);
```

```
void Disp(char ch, char str[9]);
```



```
/* EV_Display.c */
```

```
#include "C.h"
```

```
#include "EV_Display.h"
```

```
void DisplInput(void)
```

```
{
```

```
    /* 入力指示 */
```

```
    Printf(InputCommand, "%nUP = 'u', DOWN = 'd', OPEN = 'o', CLOSE = 'c'");
```

```
    Printf(InputCommand, "%nEMERGENCY = 's', RECOVERY = 'r'");
```

```
    Printf(InputCommand, "%n1st Floor CALL = 'y', 2nd Floor CALL = 'Y'");
```

```
    Printf(InputCommand, "%n1st Floor CLOSE = 'h', 2nd Floor CLOSE = 'H'");
```

```
    Printf(InputCommand, "%nQUIT = 'q'");
```

```
    Printf(InputCommand, "%nCOMMAND>");
```

```
}
```

```
/*
```

```
 * 表示関数
```

```
*/
```

```
void Disp(char ch, char str[9])
```

```
{
```

```
#ifdef USE_BCC
```

```
    int i;
```

```
    /* 画面クリア */
```

```
    CLEAR;
```

```
    if((((ch == 'o') || (ch == 'h')) && (str[3] == 'y') && (str[7] == 'y'))
```

```
        || ((ch == 't') && (str[3] == 'y') && (str[7] == 'n') && (str[6] == 'y'))
```

```
        || ((ch == 's') && (str[3] == 'y') && (str[7] == 'y')))
```

```

{
    for(i = 0; i < 4; i++)
    {
        Printf(Monitor, "¥n0000    0000");
    }
    for(i = 4; i < 12; i++)
    {
        Printf(Monitor, "¥n        ");
    }
}

else if((((ch == 'O') || (ch == 'h')) && (str[3] == 'y') && (str[7] == 'n') && (str[6] == 'y'))
    || (((ch == 'c') || (ch == 't')) && (str[3] == 'y') && (str[7] == 'n') && (str[6] == 'n') && (str[5] ==
'n') && (str[4] == 'n'))
    || ((ch == 's') && (str[3] == 'y') && (str[7] == 'n') && (str[6] == 'y'))
{
    for(i = 0; i < 4; i++)
    {
        Printf(Monitor, "¥n 0000    0000 ");
    }
    for(i = 4; i < 12; i++)
    {
        Printf(Monitor, "¥n        ");
    }
}

else if((ch == 's') && (str[3] == 'y') && (str[7] == 'n') && (str[6] == 'n') && (str[5] == 'n') && (str[4]
== 'n'))
{
    for(i = 0; i < 4; i++)
    {

```

```

        Printf(Monitor, "¥n 0000 0000 ");
    }
    for(i = 4; i < 12; i++)
    {
        Printf(Monitor, "¥n      ");
    }
}
else if((((ch == 'o') || (ch == 'h')) && (str[3] == 'y') && (str[7] == 'n') && (str[6] == 'n') && (str[5] ==
'n') && (str[4] == 'n'))
    || (((ch == 'C') || (ch == 't')) && (str[3] == 'y') && (str[5] == 'y') && (str[4] == 'n'))
    || ((ch == 's') && (str[3] == 'y') && (str[5] == 'y') && (str[4] == 'n')))
{
    for(i = 0; i < 4; i++)
    {
        Printf(Monitor, "¥n 0000 0000 ");
    }
    for(i = 4; i < 12; i++)
    {
        Printf(Monitor, "¥n      ");
    }
}
else if((((ch == 'u') || (ch == 'j')) && (str[3] == 'y') && (str[4] == 'y'))
    || ((ch == 'k') && (str[3] == 'n') && (str[2] == 'y') && (str[4] == 'y'))
    || ((ch == 'h') && (str[3] == 'y') && (str[5] == 'y') && (str[4] == 'n'))
    || (((ch == 'c') || (ch == 't')) && (str[3] == 'y') && (str[4] == 'y'))
    || ((ch == 's') && (str[3] == 'y') && (str[4] == 'y')))
{
    for(i = 0; i < 4; i++)
    {

```

```

        Printf(Monitor, "%n 00000000 ");
    }
    for(i = 4; i < 12; i++)
    {
        Printf(Monitor, "%n      ");
    }
}
else if((((ch == 'o') || (ch == 'h')) && (str[3] == 'n') && (str[2] == 'y') && (str[7] == 'y'))
    || ((ch == 't') && (str[3] == 'n') && (str[2] == 'y') && (str[7] == 'n') && (str[6] == 'y'))
    || ((ch == 's') && (str[3] == 'n') && (str[2] == 'y') && (str[7] == 'y')))
{
    for(i = 0; i < 1; i++)
    {
        Printf(Monitor, "%n      ");
    }
    for(i = 1; i < 5; i++)
    {
        Printf(Monitor, "%n0000    0000");
    }
    for(i = 5; i < 12; i++)
    {
        Printf(Monitor, "%n      ");
    }
}
else if((((ch == 'O') || (ch == 'h')) && (str[3] == 'n') && (str[2] == 'y') && (str[7] == 'n') && (str[6] ==
'y'))
    || (((ch == 'c') || (ch == 't')) && (str[3] == 'n') && (str[2] == 'y') && (str[7] == 'n') && (str[6] ==
'n') && (str[5] == 'n') && (str[4] == 'n'))
    || ((ch == 's') && (str[3] == 'n') && (str[2] == 'y') && (str[7] == 'n') && (str[6] == 'y')))

```

```

{
    for(i = 0; i < 1; i++)
    {
        Printf(Monitor, "%n      ");
    }
    for(i = 1; i < 5; i++)
    {
        Printf(Monitor, "%n 0000  0000 ");
    }
    for(i = 5; i < 12; i++)
    {
        Printf(Monitor, "%n      ");
    }
}
else if((ch == 's') && (str[3] == 'n') && (str[2] == 'y') && (str[7] == 'n') && (str[6] == 'n') && (str[5]
== 'n') && (str[4] == 'n'))
{
    for(i = 0; i < 1; i++)
    {
        Printf(Monitor, "%n      ");
    }
    for(i = 1; i < 5; i++)
    {
        Printf(Monitor, "%n 0000  0000 ");
    }
    for(i = 5; i < 12; i++)
    {
        Printf(Monitor, "%n      ");
    }
}

```

```

    }
}
else if((((ch == 'o') || (ch == 'h')) && (str[3] == 'n') && (str[2] == 'y') && (str[7] == 'n') && (str[6] ==
'n') && (str[5] == 'n') && (str[4] == 'n'))
    || (((ch == 'C') || (ch == 't')) && (str[3] == 'n') && (str[2] == 'y') && (str[5] == 'y') && (str[4] ==
'n'))
    || ((ch == 's') && (str[3] == 'n') && (str[2] == 'y') && (str[5] == 'y') && (str[4] == 'n'))))
{
    for(i = 0; i < 1; i++)
    {
        Printf(Monitor, "%n      ");
    }
    for(i = 1; i < 5; i++)
    {
        Printf(Monitor, "%n 0000 0000 ");
    }
    for(i = 5; i < 12; i++)
    {
        Printf(Monitor, "%n      ");
    }
}
else if((((ch == 'U') || (ch == 'j')) && (str[3] == 'n') && (str[2] == 'y') && (str[4] == 'y'))
    || (((ch == 'd') || (ch == 'k')) && (str[3] == 'n') && (str[2] == 'n') && (str[1] == 'n') && (str[0] ==
'n') && (str[4] == 'y'))
    || ((ch == 'h') && (str[3] == 'n') && (str[2] == 'y') && (str[5] == 'y') && (str[4] == 'n'))
    || (((ch == 'c') || (ch == 't')) && (str[3] == 'n') && (str[2] == 'y') && (str[4] == 'y'))
    || ((ch == 's') && (str[3] == 'n') && (str[2] == 'y') && (str[4] == 'y'))))
{
    for(i = 0; i < 2; i++)

```

```

    {
        Printf(Monitor, "%n      ");
    }
    for(i = 2; i < 6; i++)
    {
        Printf(Monitor, "%n 00000000 ");
    }
    for(i = 6; i < 12; i++)
    {
        Printf(Monitor, "%n      ");
    }
}
else if((((ch == 'o') || (ch == 'h')) && (str[3] == 'n') && (str[2] == 'n') && (str[1] == 'n') && (str[0] ==
'n') && (str[7] == 'y'))
    || ((ch == 't') && (str[3] == 'n') && (str[2] == 'n') && (str[1] == 'n') && (str[0] == 'n') && (str[7]
== 'n') && (str[6] == 'y'))
    || ((ch == 's') && (str[3] == 'n') && (str[2] == 'n') && (str[1] == 'n') && (str[0] == 'n') && (str[7]
== 'y'))))
{
    for(i = 0; i < 4; i++)
    {
        Printf(Monitor, "%n      ");
    }
    for(i = 4; i < 8; i++)
    {
        Printf(Monitor, "%n0000      0000");
    }
    for(i = 8; i < 12; i++)
    {

```

```

        Printf(Monitor, "%n      ");
    }
}
else if((((ch == 'O') || (ch == 'h')) && (str[3] == 'n') && (str[2] == 'n') && (str[1] == 'n') && (str[0] ==
'n') && (str[7] == 'n') && (str[6] == 'y'))
    || (((ch == 'c') || (ch == 't')) && (str[3] == 'n') && (str[2] == 'n') && (str[1] == 'n') && (str[0] ==
'n') && (str[7] == 'n') && (str[6] == 'n') && (str[5] == 'n') && (str[4] == 'n'))
    || ((ch == 's') && (str[3] == 'n') && (str[2] == 'n') && (str[1] == 'n') && (str[0] == 'n') && (str[7]
== 'n') && (str[6] == 'y')))
{
    for(i = 0; i < 4; i++)
    {
        Printf(Monitor, "%n      ");
    }
    for(i = 4; i < 8; i++)
    {
        Printf(Monitor, "%n 0000    0000 ");
    }
    for(i = 8; i < 12; i++)
    {
        Printf(Monitor, "%n      ");
    }
}
else if((ch == 's') && (str[3] == 'n') && (str[2] == 'n') && (str[1] == 'n') && (str[0] == 'n') && (str[7]
== 'n') && (str[6] == 'n') && (str[5] == 'n') && (str[4] == 'n'))
{
    for(i = 0; i < 4; i++)
    {
        Printf(Monitor, "%n      ");
    }
}

```



```

}
for(i = 4; i < 8; i++)
{
    Printf(Monitor, "%n 0000 0000 ");
}
for(i = 8; i < 12; i++)
{
    Printf(Monitor, "%n      ");
}
}
else if((((ch == 'o') || (ch == 'h')) && (str[3] == 'n') && (str[2] == 'n') && (str[1] == 'n') && (str[0] ==
'n') && (str[7] == 'n') && (str[6] == 'n') && (str[5] == 'n') && (str[4] == 'n'))
    || (((ch == 'C') || (ch == 't')) && (str[3] == 'n') && (str[2] == 'n') && (str[1] == 'n') && (str[0] ==
'n') && (str[5] == 'y') && (str[4] == 'n'))
    || ((ch == 's') && (str[3] == 'n') && (str[2] == 'n') && (str[1] == 'n') && (str[0] == 'n') && (str[5]
== 'y') && (str[4] == 'n')))
{
    for(i = 0; i < 4; i++)
    {
        Printf(Monitor, "%n      ");
    }
    for(i = 4; i < 8; i++)
    {
        Printf(Monitor, "%n 0000 0000 ");
    }
    for(i = 8; i < 12; i++)
    {
        Printf(Monitor, "%n      ");
    }
}

```

```

    }
}
else if((((ch == 'h') && (str[3] == 'n') && (str[2] == 'n') && (str[1] == 'n') && (str[0] == 'n') && (str[5]
== 'y') && (str[4] == 'n'))
    || (((ch == 'c') || (ch == 't')) && (str[3] == 'n') && (str[2] == 'n') && (str[1] == 'n') && (str[0] ==
'n') && (str[4] == 'y'))
    || ((ch == 's') && (str[3] == 'n') && (str[2] == 'n') && (str[1] == 'n') && (str[0] == 'n') && (str[4]
== 'y'))))
{
    for(i = 0; i < 4; i++)
    {
        Printf(Monitor, "%xn      ");
    }
    for(i = 4; i < 8; i++)
    {
        Printf(Monitor, "%xn  00000000  ");
    }
    for(i = 8; i < 12; i++)
    {
        Printf(Monitor, "%xn      ");
    }
}
else if((((ch == 'o') || (ch == 'h')) && (str[1] == 'y') && (str[0] == 'n') && (str[7] == 'y'))
    || ((ch == 't') && (str[1] == 'y') && (str[0] == 'n') && (str[7] == 'n') && (str[6] == 'y'))
    || ((ch == 's') && (str[1] == 'y') && (str[0] == 'n') && (str[7] == 'y'))))
{
    for(i = 0; i < 6; i++)
    {
        Printf(Monitor, "%xn      ");
    }
}

```

```

}
for(i = 6; i < 10; i++)
{
    Printf(Monitor, "%n0000    0000");
}
for(i = 10; i < 12; i++)
{
    Printf(Monitor, "%n        ");
}
}
else if((((ch == 'O') || (ch == 'h')) && (str[1] == 'y') && (str[0] == 'n') && (str[7] == 'n') && (str[6] ==
'y'))
    || (((ch == 'c') || (ch == 't')) && (str[1] == 'y') && (str[0] == 'n') && (str[7] == 'n') && (str[6] ==
'n') && (str[5] == 'n') && (str[4] == 'n'))
    || ((ch == 's') && (str[1] == 'y') && (str[0] == 'n') && (str[7] == 'n') && (str[6] == 'y'))))
{
    for(i = 0; i < 6; i++)
    {
        Printf(Monitor, "%n        ");
    }
    for(i = 6; i < 10; i++)
    {
        Printf(Monitor, "%n 0000    0000 ");
    }
    for(i = 10; i < 12; i++)
    {
        Printf(Monitor, "%n        ");
    }
}
}

```

```

else if((ch == 's') && (str[1] == 'y') && (str[0] == 'n') && (str[7] == 'n') && (str[6] == 'n') && (str[5]
== 'n') && (str[4] == 'n'))
{
    for(i = 0; i < 6; i++)
    {
        Printf(Monitor, "%n      ");
    }
    for(i = 6; i < 10; i++)
    {
        Printf(Monitor, "%n 0000 0000 ");
    }
    for(i = 10; i < 12; i++)
    {
        Printf(Monitor, "%n      ");
    }
}
else if((((ch == 'o') || (ch == 'h')) && (str[1] == 'y') && (str[0] == 'n') && (str[7] == 'n') && (str[6] ==
'n') && (str[5] == 'n') && (str[4] == 'n'))
    || (((ch == 'C') || (ch == 't')) && (str[1] == 'y') && (str[0] == 'n') && (str[5] == 'y') && (str[4] ==
'n'))
    || ((ch == 's') && (str[1] == 'y') && (str[0] == 'n') && (str[5] == 'y') && (str[4] == 'n'))))
{
    for(i = 0; i < 6; i++)
    {
        Printf(Monitor, "%n      ");
    }
    for(i = 6; i < 10; i++)
    {
        Printf(Monitor, "%n 0000 0000 ");
    }
}

```

```

}
for(i = 10; i < 12; i++)
{
    Printf(Monitor, "%n    ");
}
}
else if((((ch == 'u') || (ch == 'j')) && (str[3] == 'n') && (str[2] == 'n') && (str[1] == 'n') && (str[0] ==
'n') && (str[4] == 'y'))
    || (((ch == 'D') || (ch == 'k')) && (str[1] == 'y') && (str[0] == 'n') && (str[4] == 'y'))
    || ((ch == 'h') && (str[1] == 'y') && (str[0] == 'n') && (str[5] == 'y') && (str[4] == 'n'))
    || (((ch == 'c') || (ch == 't')) && (str[1] == 'y') && (str[0] == 'n') && (str[4] == 'y'))
    || ((ch == 's') && (str[1] == 'y') && (str[0] == 'n') && (str[4] == 'y')))
{
    for(i = 0; i < 6; i++)
    {
        Printf(Monitor, "%n    ");
    }
    for(i = 6; i < 10; i++)
    {
        Printf(Monitor, "%n  00000000  ");
    }
    for(i = 10; i < 12; i++)
    {
        Printf(Monitor, "%n    ");
    }
}
else if((((ch == 'o') || (ch == 'h')) && (str[0] == 'y') && (str[7] == 'y'))
    || ((ch == 't') && (str[0] == 'y') && (str[7] == 'n') && (str[6] == 'y'))

```

```

    || ((ch == 's') && (str[0] == 'y') && (str[7] == 'y')))
{
    for(i = 0; i < 8; i++)
    {
        Printf(Monitor, "%n      ");
    }
    for(i = 8; i < 12; i++)
    {
        Printf(Monitor, "%n0000    0000");
    }
}

else if((((ch == 'O') || (ch == 'h')) && (str[0] == 'y') && (str[7] == 'n') && (str[6] == 'y'))
    || (((ch == 'c') || (ch == 't')) && (str[0] == 'y') && (str[7] == 'n') && (str[6] == 'n') && (str[5] ==
'n') && (str[4] == 'n'))
    || ((ch == 's') && (str[0] == 'y') && (str[7] == 'n') && (str[6] == 'y')))
{
    for(i = 0; i < 8; i++)
    {
        Printf(Monitor, "%n      ");
    }
    for(i = 8; i < 12; i++)
    {
        Printf(Monitor, "%n 0000    0000 ");
    }
}

else if((ch == 's') && (str[0] == 'y') && (str[7] == 'n') && (str[6] == 'n') && (str[5] == 'n') && (str[4]
== 'n'))
{
    for(i = 0; i < 8; i++)

```

```

{
    Printf(Monitor, "%n      ");
}
for(i = 8; i < 12; i++)
{
    Printf(Monitor, "%n 0000 0000 ");
}
}
else if((((ch == 'o') || (ch == 'h')) && (str[0] == 'y') && (str[7] == 'n') && (str[6] == 'n') && (str[5] ==
'n') && (str[4] == 'n'))
    || (((ch == 'C') || (ch == 't')) && (str[0] == 'y') && (str[5] == 'y') && (str[4] == 'n'))
    || ((ch == 's') && (str[0] == 'y') && (str[5] == 'y') && (str[4] == 'n')))
{
    for(i = 0; i < 8; i++)
    {
        Printf(Monitor, "%n      ");
    }
    for(i = 8; i < 12; i++)
    {
        Printf(Monitor, "%n 0000 0000 ");
    }
}
else if(((ch == 'j') && (str[1] == 'y') && (str[0] == 'n') && (str[4] == 'y'))
    || (((ch == 'd') || (ch == 'k')) && (str[0] == 'y') && (str[4] == 'y'))
    || ((ch == 'h') && (str[0] == 'y') && (str[5] == 'y') && (str[4] == 'n'))
    || (((ch == 'c') || (ch == 't')) && (str[0] == 'y') && (str[4] == 'y'))
    || ((ch == 's') && (str[0] == 'y') && (str[4] == 'y')))
{
    for(i = 0; i < 8; i++)

```

```
{
    Printf(Monitor, "%n    ");
}
for(i = 8; i < 12; i++)
{
    Printf(Monitor, "%n 00000000 ");
}
}
#endif
/* 入力指示 */
DisplInput();
return;
}
```



```
/* EV_Input.h */
```

```
#ifndef Panel_h
```

```
#define Panel_h
```

```
#include "Panel.h"
```

```
#endif
```

```
#ifndef Timer_h
```

```
#define Timer_h
```

```
#include "Timer.h"
```

```
#endif
```

```
#ifndef EV_Time_h
```

```
#define EV_Time_h
```

```
#include "EV_Time.h"
```

```
#endif
```

```
#ifndef EV_File_h
```

```
#define EV_File_h
```

```
#include "EV_File.h"
```

```
#endif
```

```
/*=====
```

```
  入力を表す関数のプロトタイプ宣言
```

```
=====*/
```

```
char GetChar(char Ret);
```

```
/*=====
```

入力を表す構造体

```
=====*/  
struct EV_Input  
{  
    /* 安全 */  
    char Safety;  
    char *p_Safety;  
    char Command;  
    char PermitCommand;  
    char *p_PermitCommand;  
    char PermitTurnOpen;  
    char *p_PermitTurnOpen;  
    char ch;  
    char *p_ch;  
    char str[9];  
    char *p_str;  
  
    /* ファイルストリーム */  
    struct EV_File SF;  
    struct EV_File CF;  
    struct EV_File PCF;  
    struct EV_File PTOF;  
    struct EV_File LF;  
    struct EV_File MF;  
};  
  
/*=====*/  
入力を表すコンストラクタとメソッドのプロトタイプ宣言  
=====*/
```

```
void EV_Input(struct EV_Input *This);
```

```
void OnInput(struct EV_Input *This, Thread *th);
```

```
/* EV_Input.c */
```

```
#include "C.h"
```

```
#include "EV_Input.h"
```

```
/*=====
```

```
    入力を表す関数
```

```
=====*/
```

```
char GetChar(char Ret)
```

```
{
```

```
#ifndef USE_BCC
```

```
    char sw[4];
```

```
    int i;
```

```
    int j;
```

```
    /* スイッチワーク初期化 */
```

```
    sw[0] = sw[1] = sw[2] = sw[3] = 0;
```

```
    /* ボタンが押された時にLCD,SCI,USBにメッセージを送る */
```

```
    for(j=0;j<4;j++)
```

```
    {
```

```
        i = GetSW(j);
```

```
        if( ((sw[j]^1) & i) ) /* sw = off->onで条件成立 */
```

```
        {
```

```
            switch(j){
```

```
            case 0:
```

```
                Ret = 'u'; /* sw0 Up 2階で開く */
```

```
                break;
```

```
            case 1:
```

```
                Ret = 'd'; /* sw1 Down 1階で開く */
```

```
        break;
    case 2:
        Ret = 'c'; /* sw2 Close 閉じる */
        break;
    case 3:
        Ret = 'q'; /* sw3 Quit 終了 */
        break;
    default:
        break;
}
}
```

```
}
}
```

```
#else
```

```
#ifdef USE_LINUX
```

```
    if(kbhit())
```

```
    {
```

```
        Ret = (char) getchar();
```

```
    }
```

```
#else
```

```
#ifdef USE_MSVS2005
```

```
    if(_kbhit())
```

```
    {
```

```
        Ret = (char) _getche();
```

```
    }
```

```
#else
```

```
    if(kbhit())
```

```
    {
```

```
        Ret = (char) getche();
```

```
    }
```

```
#endif
```

```
#endif
```

```
#endif
```

```
    return Ret;
```

```
}
```

```
/*=====
```

```
    入力を表すコンストラクタとメソッド
```

```
=====*/
```

```
void EV_Input(struct EV_Input *This)
```

```
{
```

```
    /* 初期化 */
```

```
    This->Command = '¥0';
```

```
    This->p_ch = &This->ch;
```

```
    This->p_str = &This->str[0];
```

```
    This->str[8] = '¥0';
```

```
    /* 安全初期化 */
```

```
    This->p_Safety = &This->Safety;
```

```
    /* 入力許可初期化 */
```

```
    This->p_PermitCommand = &This->PermitCommand;
```

```
    /* 反転開許可初期化 */
```

```
    This->p_PermitTurnOpen = &This->PermitTurnOpen;
```

```
    EV_File(&This->SF);
```

```
    EV_File(&This->CF);
```

```
    EV_File(&This->PCF);
```

```
    EV_File(&This->PTOF);
```

```
    EV_File(&This->MF);
```

```

/* 安全入力 */
if(Read(&This->SF, "Safety.txt¥0", This->p_Safety) == NG) return;
/* 入力許可 */
if(PermitCommand_Read(&This->PCF, This->p_PermitCommand) == NG) return;
/* 反転開許可 */
if(PermitTurnOpen_Read(&This->PTOF, This->p_PermitTurnOpen) == NG) return;
/* モーター命令解読 */
if(Read(&This->MF, "Motor.txt¥0", This->p_ch) == NG) return;
/* リミットスイッチの前状態読み込み */
ReadString(&This->LF, "Limit.txt¥0", This->p_str, 9);
}

```

```

void OnInput(struct EV_Input *This, Thread *th)

```

```

{
    if(Command_Read(&This->CF, &This->Command) == NG) return;
    This->Command = GetChar(This->Command);

    /* 安全入力 */
    if(Read(&This->SF, "Safety.txt¥0", This->p_Safety) == NG) return;
    /* 入力許可 */
    if(PermitCommand_Read(&This->PCF, This->p_PermitCommand) == NG) return;
    /* 反転開許可 */
    if(PermitTurnOpen_Read(&This->PTOF, This->p_PermitTurnOpen) == NG) return;
    /* モーター命令解読 */
    if(Read(&This->MF, "Motor.txt¥0", This->p_ch) == NG) return;
    /* リミットスイッチの前状態読み込み */
    ReadString(&This->LF, "Limit.txt¥0", This->p_str, 9);

    switch(This->Command){

```

case 's':

```
/* 命令入力 */  
Write(&This->SF, "Safety.txt", 's');  
Motor_Write(&This->MF, 's');  
This->Command = 'N';  
Command_Write(&This->CF, 'N');  
PermitCommand_Write(&This->PCF, 'N');  
break;
```

case 'r':

```
/* 命令入力 */  
Write(&This->SF, "Safety.txt", 'h');  
This->Command = 'N';  
Command_Write(&This->CF, 'N');  
PermitCommand_Write(&This->PCF, 'c');  
break;
```

case 'q':

```
/* 命令入力 */  
Command_Write(&This->CF, This->Command);  
PrintF(ClsPnl, "QUIT");  
delete_(th);  
break;
```

default:

```
if(This->PermitCommand == 'c'){  
    switch(This->Command){  
    case 'o':  
        if(This->str[4] != 'y'){  
            PermitTurnOpen_Write(&This->PTOF, 'o');  
        }  
    }  
}
```



```

if(This->Safety == 'h'){
    /* 命令入力 */
    This->Safety = 'Y';
    Write(&This->SF, "Safety.txt", 'Y');
    if((This->ch == 's') && (This->Safety == 'Y') && (This->str[7] == 'n')){
        Motor_Write(&This->MF, 'h');
    }
}

/* 命令入力 */
Command_Write(&This->CF, This->Command);
PermitCommand_Write(&This->PCF, 'N');
break;
case 'c':
    if(This->str[4] != 'y'){
        PermitTurnOpen_Write(&This->PTOF, 'o');
    }
    if(This->Safety == 'h'){
        /* 命令入力 */
        This->Safety = 'Y';
        Write(&This->SF, "Safety.txt", 'Y');
        if((This->ch == 's') && (This->Safety == 'Y') && (This->str[4] == 'n')){
            Motor_Write(&This->MF, 't');
        }
    }
}

/* 命令入力 */
Command_Write(&This->CF, This->Command);
PermitCommand_Write(&This->PCF, 'N');
break;
case 'u':

```

```

if(This->str[4] != 'y'){
    PermitTurnOpen_Write(&This->PTOF, 'o');
}
if(This->Safety == 'h'){
    /* 命令入力 */
    This->Safety = 'Y';
    Write(&This->SF, "Safety.txt", 'Y');
    if((This->ch == 's') && (This->Safety == 'Y') && (This->str[0] == 'y') && (This-
>str[4] == 'n')){
        Motor_Write(&This->MF, 't');
    }
    else if((This->ch == 's') && (This->Safety == 'Y') && (This->str[3] == 'n')){
        Motor_Write(&This->MF, 'j');
    }
    else if((This->ch == 's') && (This->Safety == 'Y') && (This->str[3] == 'y') && (This-
>str[7] == 'n')){
        Motor_Write(&This->MF, 'h');
    }
}
/* 命令入力 */
Command_Write(&This->CF, This->Command);
PermitCommand_Write(&This->PCF, 'N');
break;
case 'd':
    if(This->str[4] != 'y'){
        PermitTurnOpen_Write(&This->PTOF, 'o');
    }
    if(This->Safety == 'h'){
        /* 命令入力 */

```

```

This->Safety = 'Y';

Write(&This->SF, "Safety.txt", 'Y');

if((This->ch == 's') && (This->Safety == 'Y') && (This->str[3] == 'y') && (This-
>str[4] == 'n')){

    Motor_Write(&This->MF, 't');

}

else if((This->ch == 's') && (This->Safety == 'Y') && (This->str[0] == 'n')){

    Motor_Write(&This->MF, 'k');

}

else if((This->ch == 's') && (This->Safety == 'Y') && (This->str[0] == 'y') && (This-
>str[7] == 'n')){

    Motor_Write(&This->MF, 'h');

}

}

/* 命令入力 */

Command_Write(&This->CF, This->Command);

PermitCommand_Write(&This->PCF, 'N');

break;

case 'y':

if((This->str[0] != 'y') && (This->str[4] != 'y')){

    Printf(Pannel, "¥nA Basket isn't 1st Floor");

}

else{

if(This->Safety == 'h'){

    /* 命令入力 */

    This->Safety = 'Y';

    Write(&This->SF, "Safety.txt", 'Y');

if((This->ch == 's') && (This->Safety == 'Y') && (This->str[0] == 'n')){

        Motor_Write(&This->MF, 'k');

}

}

}

}

```

```

    }
    else if((This->ch == 's') && (This->Safety == 'Y') && (This->str[0] == 'y') &&
(This->str[7] == 'n')){
        Motor_Write(&This->MF, 'h');
    }
}
/* 命令入力 */
Command_Write(&This->CF, This->Command);
PermitCommand_Write(&This->PCF, 'N');
}
break;
case 'Y':
    if((This->str[3] != 'y') && (This->str[4] != 'y')){
        Printf(Panel, "¥nA Basket isn't 2nd Floor");
    }
    else{
        if(This->Safety == 'h'){
            /* 命令入力 */
            This->Safety = 'Y';
            Write(&This->SF, "Safety.txt", 'Y');
            if((This->ch == 's') && (This->Safety == 'Y') && (This->str[3] == 'n')){
                Motor_Write(&This->MF, 'j');
            }
            else if((This->ch == 's') && (This->Safety == 'Y') && (This->str[3] == 'y') &&
(This->str[7] == 'n')){
                Motor_Write(&This->MF, 'h');
            }
        }
    }
}

```

```

        /* 命令入力 */
        Command_Write(&This->CF, This->Command);
        PermitCommand_Write(&This->PCF, 'N');
    }
    break;
case 'h':
    if(This->str[0] != 'y'){
        Printf(Panel, "¥nA Basket isn't 1st Floor");
    }
    else{
        if(This->str[4] != 'y'){
            PermitTurnOpen_Write(&This->PTOF, 'o');
        }
        if(This->Safety == 'h'){
            /* 命令入力 */
            This->Safety = 'Y';
            Write(&This->SF, "Safety.txt", 'Y');
            if((This->ch == 's') && (This->Safety == 'Y') && (This->str[4] == 'n')){
                Motor_Write(&This->MF, 't');
            }
        }
        /* 命令入力 */
        Command_Write(&This->CF, This->Command);
        PermitCommand_Write(&This->PCF, 'N');
    }
    break;
case 'H':
    if(This->str[3] != 'y'){
        Printf(Panel, "¥nA Basket isn't 2nd Floor");
    }

```

```

}
else{
    if(This->str[4] != 'y'){
        PermitTurnOpen_Write(&This->PTOF, 'o');
    }
    if(This->Safety == 'h'){
        /* 命令入力 */
        This->Safety = 'Y';
        Write(&This->SF, "Safety.txt", 'Y');
        if((This->ch == 's') && (This->Safety == 'Y') && (This->str[4] == 'n')){
            Motor_Write(&This->MF, 't');
        }
    }
    /* 命令入力 */
    Command_Write(&This->CF, This->Command);
    PermitCommand_Write(&This->PCF, 'N');
}
break;
default:
    break;
}
}
else if(This->PermitTurnOpen == 'o'){
    if((This->str[0] == 'y') && (This->str[7] != 'y')){
        switch(This->Command){
            case 'o':
            case 'd':
            case 'y':
                if(This->Safety == 'h'){

```

```

        /* 命令入力 */
        This->Safety = 'Y';
        Write(&This->SF, "Safety.txt", 'Y');
        if((This->ch == 's') && (This->Safety == 'Y') && (This->str[0] == 'y') && (This-
>str[7] == 'n')){

                Motor_Write(&This->MF, 'h');
        }
}

/* 命令入力 */
Command_Write(&This->CF, This->Command);
PermitTurnOpen_Write(&This->PTOF, 'N');
break;
default:
        break;
}
}
else if((This->str[3] == 'y') && (This->str[7] != 'y')){
        switch(This->Command){
        case 'o':
        case 'u':
        case 'Y':
                if(This->Safety == 'h'){
                        /* 命令入力 */
                        This->Safety = 'Y';
                        Write(&This->SF, "Safety.txt", 'Y');
                        if((This->ch == 's') && (This->Safety == 'Y') && (This->str[3] == 'y') && (This-
>str[7] == 'n')){

                                Motor_Write(&This->MF, 'h');
                        }
                }
        }
}

```

```
    }  
    /* 命令入力 */  
    Command_Write(&This->CF, This->Command);  
    PermitTurnOpen_Write(&This->PTOF, 'N');  
    break;  
default:  
    break;  
}  
}  
}  
break;  
}  
return;  
}
```



```
/* EV_Controller.h */
```

```
#ifndef Panel_h
```

```
#define Panel_h
```

```
#include "Panel.h"
```

```
#endif
```

```
#ifndef Timer_h
```

```
#define Timer_h
```

```
#include "Timer.h"
```

```
#endif
```

```
#ifndef EV_Time_h
```

```
#define EV_Time_h
```

```
#include "EV_Time.h"
```

```
#endif
```

```
#ifndef EV_File_h
```

```
#define EV_File_h
```

```
#include "EV_File.h"
```

```
#endif
```

```
#ifndef EV_UpDown_h
```

```
#define EV_UpDown_h
```

```
#include "EV_UpDown.h"
```

```
#endif
```

```
#ifndef EV_OpenClose_h
```

```
#define EV_OpenClose_h
```

```
#include "EV_OpenClose.h"
```

```
#endif
```

```
/*=====
制御盤を表す構造体宣言
=====*/
```

```
struct EV_Controller
```

```
{
```

```
/* エレベーターの現在位置オブジェクトの宣言 */
```

```
struct Position P;
```

```
struct Position *p_P;
```

```
/* 上昇オブジェクトの宣言 */
```

```
struct UpMotor UPMT;
```

```
/* 下降オブジェクトの宣言 */
```

```
struct DownMotor DNMT;
```

```
/* エレベーターの位置仮想ログ */
```

```
struct WaitPositionChangeLog WPCL;
```

```
/* エレベーターの現在位置オブジェクトの宣言 */
```

```
struct Door DR;
```

```
struct Door *p_DR;
```

```
/* 開オブジェクトの宣言 */
```

```
struct OpenMotor OPMT;
```

```
/* 閉オブジェクトの宣言 */
```

```
struct CloseMotor CLMT;
```

```
/* エレベーターの位置仮想ログ */  
struct WaitDoorChangeLog WDCL;
```

```
/* 経過時間テンポラリ */  
struct EV_Time T;
```

```
/* 安全 */
```

```
char Safety;
```

```
char *p_Safety;
```

```
/* Limit */
```

```
char str[9];
```

```
char *p_str;
```

```
/* 命令 */
```

```
char Command;
```

```
char *p_Command;
```

```
char PermitCommand;
```

```
char *p_PermitCommand;
```

```
char PermitTurnOpen;
```

```
char *p_PermitTurnOpen;
```

```
/* ファイルストリーム */
```

```
struct EV_File SF;
```

```
struct EV_File LF;
```

```
struct EV_File CF;
```

```
struct EV_File PCF;
```

```
struct EV_File PTOF;
```

```
struct EV_File MF;
```

```
};
```

```
/*=====
  制御を表すコンストラクタとメソッドのプロトタイプ宣言
=====*/
void EV_Controller(struct EV_Controller *This, Thread *th);
void OnController(struct EV_Controller *This, Thread *th);
```

```
/* EV_Controller.c */
```

```
#include "C.h"
```

```
#include "EV_Controller.h"
```

```
/*=====
```

```
制御関数
```

```
=====*/
```

```
void EV_Controller(struct EV_Controller *This, Thread *th)
```

```
{
```

```
    /* 初期化 */
```

```
    This->p_P = &This->P;
```

```
    Position(This->p_P);
```

```
    UpMotor(&This->UPMT);
```

```
    DownMotor(&This->DNMT);
```

```
    WaitPositionChangeLog(&This->WPCL);
```

```
    This->p_DR = &This->DR;
```

```
    Door(This->p_DR);
```

```
    OpenMotor(&This->OPMT);
```

```
    CloseMotor(&This->CLMT);
```

```
    WaitDoorChangeLog(&This->WDCL);
```

```
    EV_Time(&This->T, th);
```

```
    EV_File(&This->SF);
```

```
    EV_File(&This->LF);
```

```
    EV_File(&This->CF);
```

```
    EV_File(&This->PCF);
```

```
    EV_File(&This->PTOF);
```

```
    EV_File(&This->MF);
```

```

/* 安全初期化 */
This->p_Safety = &This->Safety;

/* Limit初期化 */
This->p_str = &This->str[0];

/* 命令初期化 */
This->p_Command = &This->Command;
This->p_PermitCommand = &This->PermitCommand;
This->p_PermitTurnOpen = &This->PermitTurnOpen;

/* モーター停止命令 */
Motor_Write(&This->MF, 's');

/* エレベーターの位置仮想ログ */
/* 初期値 */
OnInitWaitPositionChangeLog(&This->WPCL, This->p_P);

/* エレベーターの位置仮想ログ */
/* 初期値 */
OnInitWaitDoorChangeLog(&This->WDCL, This->p_DR);

/* リミットスイッチの前状態読み込み */
ReadString(&This->LF, "Limit.txt¥0", This->p_str, 9);

/* 命令初期化 */
if(Command_Write(&This->CF, 'N') == NG) return;
if(PermitCommand_Write(&This->PCF, 'c') == NG) return;
if(PermitTurnOpen_Write(&This->PTOF, 'N') == NG) return;
}

```

```

/*
 * 主制御関数
 */
void OnController(struct EV_Controller *This, Thread *th)
{
    /* 安全入力 */
    if(Read(&This->SF, "Safety.txt¥0", This->p_Safety) == NG) return;
    /* リミットスイッチの前状態読み込み */
    ReadString(&This->LF, "Limit.txt¥0", This->p_str, 9);
    /* 命令入力 */
    if(Command_Read(&This->CF, This->p_Command) == NG) return;
    /* 入力許可 */
    if(PermitCommand_Read(&This->PCF, This->p_PermitCommand) == NG) return;
    /* 反転開許可 */
    if(PermitTurnOpen_Read(&This->PTOF, This->p_PermitTurnOpen) == NG) return;

    switch(This->Command){
        /* 終了命令ならば */
        case 'q':
            Motor_Write(&This->MF, 's');
            delete_(th);
            break;
        /* 非常停止命令ならば */
        case 's':
            SetPermit(&This->T, OFF);
            break;
        /* 復帰命令ならば */
        case 'r':
            break;
    }
}

```

```

/* 上階呼命令ならば */
case 'Y':
    if((*This->p_P->p_UnderStop == ON) && (*This->p_DR->p_CloserStop == OFF)){
        break;
    }
/* 上昇命令ならば */
case 'u':
    if(*This->p_P->p_UpperStop == ON){
        /* 開完了時 */
        if(*This->p_DR->p_OpennerStop == ON){
            PermitCommand_Write(&This->PCF, 'c');
            SetPermit(&This->T, ON);
            Command_Write(&This->CF, 'N');
            Clear();
            Printf(Pannel, "Hello EV    ");
            break;
        }
        else{
            SetPermit(&This->T, OFF);
            SetCurrentTime(&This->T);
            /* 開 */
            Open(This->p_DR, This->OPMT, This->WDCL, This->p_Safety);
        }
    }
}
else if(*This->p_DR->p_CloserStop == ON){
    /* 閉完了時 */
    SetPermit(&This->T, OFF);
    SetCurrentTime(&This->T);
    PermitTurnOpen_Write(&This->PTOF, 'N');
}

```



```

    /* 上昇 */
    Up(This->p_P,This->UPMT,This->WPCL, This->p_Safety);
}
else if(*This->p_DR->p_CloserStop == OFF){
    SetPermit(&This->T, OFF);
    SetCurrentTime(&This->T);
    /* 閉 */
    Close(This->p_DR,This->CLMT,This->WDCL, This->p_Safety);
}
else{
    SetPermit(&This->T, OFF);
    SetCurrentTime(&This->T);
}
break;
/* 下階呼命令ならば */
case 'y':
    if((*This->p_P->p_UpperStop == ON) && (*This->p_DR->p_CloserStop == OFF)){
        break;
    }
/* 下降命令ならば */
case 'd':
    if(*This->p_P->p_UnderStop == ON){
        /* 開完了時 */
        if(*This->p_DR->p_OpennerStop == ON){
            PermitCommand_Write(&This->PCF, 'c');
            SetPermit(&This->T, ON);
            Command_Write(&This->CF, 'N');
            Clear();
        }
    }
}

```

```

        Printf(Panel, "Hello EV    ");
        break;
    }
    else{
        SetPermit(&This->T, OFF);
        SetCurrentTime(&This->T);
        /* 開 */
        Open(This->p_DR,This->OPMT,This->WDCL, This->p_Safety);
    }
}
else if(*This->p_DR->p_CloserStop == ON){
    /* 閉完了時 */
    SetPermit(&This->T, OFF);
    SetCurrentTime(&This->T);
    PermitTurnOpen_Write(&This->PTOF, 'N');
    /* 下降 */
    Down(This->p_P,This->DNMT,This->WPCL, This->p_Safety);
}
else if(*This->p_DR->p_CloserStop == OFF){
    SetPermit(&This->T, OFF);
    SetCurrentTime(&This->T);
    /* 閉 */
    Close(This->p_DR,This->CLMT,This->WDCL, This->p_Safety);
}
else{
    SetPermit(&This->T, OFF);
    SetCurrentTime(&This->T);
}
break;

```

```
/* 開命令ならば */
```

```
case 'o':
```

```
/* 開完了時 */
```

```
if(*This->p_DR->p_OpennerStop == ON){
```

```
    PermitCommand_Write(&This->PCF, 'c');
```

```
    SetPermit(&This->T, ON);
```

```
    Command_Write(&This->CF, 'N');
```

```
    Clear();
```

```
    Printf(Panel, "Hello EV  ");
```

```
    break;
```

```
}
```

```
else{
```

```
    SetPermit(&This->T, OFF);
```

```
    SetCurrentTime(&This->T);
```

```
/* 開 */
```

```
    Open(This->p_DR, This->OPMT, This->WDCL, This->p_Safety);
```

```
}
```

```
break;
```

```
/* 閉命令ならば */
```

```
case 'c':
```

```
    SetPermit(&This->T, OFF);
```

```
    SetCurrentTime(&This->T);
```

```
/* 閉完了時 */
```

```
if(*This->p_DR->p_CloserStop == ON){
```

```
    PermitTurnOpen_Write(&This->PTOF, 'N');
```

```
    Command_Write(&This->CF, 'N');
```

```
    PermitCommand_Write(&This->PCF, 'c');
```

```
    Clear();
```

```
    Printf(Panel, "Hello EV  ");
```

```

        break;
    }
    else{
        /* 閉 */
        Close(This->p_DR,This->CLMT,This->WDCL, This->p_Safety);
    }
    break;
/* 閉命令ならば */
case 'H':
    if(*This->p_P->p_UpperStop == ON){
        SetPermit(&This->T, OFF);
        SetCurrentTime(&This->T);
        /* 閉完了時 */
        if(*This->p_DR->p_CloserStop == ON){
            Command_Write(&This->CF, 'N');
            PermitTurnOpen_Write(&This->PTOF, 'N');
            PermitCommand_Write(&This->PCF, 'c');
            Clear();
            Printf(Panel, "Hello EV    ");
            break;
        }
    }
    else{
        /* 閉 */
        Close(This->p_DR,This->CLMT,This->WDCL, This->p_Safety);
    }
}

break;
/* 閉命令ならば */

```

case 'h':

```
if(*This->p_P->p_UnderStop == ON){
```

```
    SetPermit(&This->T, OFF);
```

```
    SetCurrentTime(&This->T);
```

```
    /* 閉完了時 */
```

```
if(*This->p_DR->p_CloserStop == ON){
```

```
    Command_Write(&This->CF, 'N');
```

```
    PermitTurnOpen_Write(&This->PTOF, 'N');
```

```
    PermitCommand_Write(&This->PCF, 'c');
```

```
    Clear();
```

```
    Printf(Pannel, "Hello EV    ");
```

```
    break;
```

```
}
```

```
else{
```

```
    /* 閉 */
```

```
    Close(This->p_DR, This->CLMT, This->WDCL, This->p_Safety);
```

```
}
```

```
}
```

```
break;
```

default:

```
break;
```

```
}
```

```
if((GetCurrentTime(&This->T) >= OPENTIMEOUT) && (*This->p_DR->p_OpennerStop == ON) &&  
(GetPermit(&This->T) == ON)){
```

```
    SetPermit(&This->T, OFF);
```

```
    SetCurrentTime(&This->T);
```

```
    PermitTurnOpen_Write(&This->PTOF, 'o');
```

```
    PermitCommand_Write(&This->PCF, 'N');
```

```
/* 閉 */
```

```
Command_Write(&This->CF, 'c');
```

```
}
```

```
return;
```

```
}
```

```
/* EV_Puls.h */
```

```
#ifndef Panel_h
```

```
#define Panel_h
```

```
#include "Panel.h"
```

```
#endif
```

```
#ifndef Timer_h
```

```
#define Timer_h
```

```
#include "Timer.h"
```

```
#endif
```

```
#ifndef EV_Time_h
```

```
#define EV_Time_h
```

```
#include "EV_Time.h"
```

```
#endif
```

```
#ifndef EV_File_h
```

```
#define EV_File_h
```

```
#include "EV_File.h"
```

```
#endif
```

```
/* Raspberry Pi 3 Model B I/O */
```

```
#ifdef USE_RASPBIAN
```

```
#define GPIO16 16
```

```
#define GPIO17 17
```

```
#define GPIO18 18
```

```
#define GPIO19 19
```

```
#define GPIO20 20
```

```
#endif
```

```
/*=====
```

```
送信を表す構造体
```

```
=====*/
```

```
struct EV_Puls
```

```
{
```

```
    char ch;
```

```
    char *p_ch;
```

```
    char Command;
```

```
    char *p_Command;
```

```
    /* ファイルストリーム */
```

```
    struct EV_File CF;
```

```
    struct EV_File MF;
```

```
};
```

```
/*=====
```

```
送信を表す関数のプロトタイプ宣言
```

```
=====*/
```

```
void EV_Set(int addressDataSet, int dataSet, int addressClockSet,
```

```
int clockSet);
```

```
void EV_EnableSet(void);
```

```
int EV_AddressSet(Thread *th, int base, int address);
```

```
int EV_DataSet(Thread *th, int base, int data);
```

```
void EV_AddressDataSet(struct EV_Puls *puls, Thread *th,
```

```
int address_1, int address_2, int address_3, int address_4,
```

```
int data_1, int data_2, int data_3, int data_4);
```



```
/*=====
送信を表すコンストラクタとメソッドのプロトタイプ宣言
=====*/
void EV_Puls(struct EV_Puls *This, Thread *th);
void OnPuls(struct EV_Puls *This, Thread *th);
```

```
/* EV_Puls.c */
```

```
#include "C.h"
```

```
#include "EV_Puls.h"
```

```
void EV_Set(int addressDataSet, int dataSet, int addressClockSet,  
int clockSet){
```

```
#ifndef USE_BCC
```

```
    /* address data set */
```

```
    if(addressDataSet == 0){
```

```
        PB.DR &= 0xfe;
```

```
    }
```

```
    else if(addressDataSet == 1){
```

```
        PB.DR |= 0x01;
```

```
    }
```

```
    /* data set */
```

```
    if(dataSet == 0){
```

```
        PB.DR &= 0xfd;
```

```
    }
```

```
    else if(dataSet == 1){
```

```
        PB.DR |= 0x02;
```

```
    }
```

```
    /* address clock set */
```

```
    if(addressClockSet == 0){
```

```
        PB.DR &= 0xfb;
```

```
    }
```

```
    else if(addressClockSet == 1){
```

```
        PB.DR |= 0x04;
```

```

}

/* clock set */
if(clockSet == 0){
    PB.DR &= 0xf7;
}

else if(clockSet == 1){
    PB.DR |= 0x08;
}

/* disable set */
PB.DR &= 0xef;

#endif

#ifdef USE_RASPBIAN

/* address data set */
digitalWrite(GPIO16, addressDataSet);

/* data set */
digitalWrite(GPIO17, dataSet);

/* address clock set */
digitalWrite(GPIO18, addressClockSet);

/* clock set */
digitalWrite(GPIO19, clockSet);

/* disable set */
digitalWrite(GPIO20, 0);

#endif

return;
}

void EV_EnableSet(void){

#ifdef USE_BCC

/* address data set */

```

```

PB.DR &= 0xfe;

/* data set */

PB.DR &= 0xfd;

/* address clock set */

PB.DR &= 0xfb;

/* clock set */

PB.DR &= 0xf7;

/* enable set */

PB.DR |= 0x10;

#endif

#ifdef USE_RASPBIAN

/* address data set */

digitalWrite(GPIO16, 0);

/* data set */

digitalWrite(GPIO17, 0);

/* address clock set */

digitalWrite(GPIO18, 0);

/* clock set */

digitalWrite(GPIO19, 0);

/* enable set */

digitalWrite(GPIO20, 1);

#endif

return;

}

int EV_AddressSet(Thread *th, int base, int address){

int Ret;

Ret = NG;

if(th->count == base){

```

```

    EV_Set(address, 0, 0, 0);

    th->count++;

    Ret = OK;
}
else if(th->count == base + 1){
    EV_Set(address, 0, 1, 0);

    th->count++;

    Ret = OK;
}
return Ret;
}

```

```

int EV_DataSet(Thread *th, int base, int data){
    int Ret;

    Ret = NG;

    if(th->count == base){
        EV_Set(0, data, 0, 0);

        th->count++;

        Ret = OK;
    }
    else if(th->count == base + 1){
        EV_Set(0, data, 0, 1);

        th->count++;

        Ret = OK;
    }
    return Ret;
}

```

```

void EV_AddressDataSet(struct EV_Puls *puls, Thread *th,

```

```

int address_1, int address_2, int address_3, int address_4,
int data_1, int data_2, int data_3, int data_4){
    if(EV_AddressSet(th, 0, address_1) == OK);
    else if(EV_AddressSet(th, 2, address_2) == OK);
    else if(EV_AddressSet(th, 4, address_3) == OK);
    else if(EV_AddressSet(th, 6, address_4) == OK);
    else if(EV_DataSet(th, 8, data_1) == OK);
    else if(EV_DataSet(th, 10, data_2) == OK);
    else if(EV_DataSet(th, 12, data_3) == OK);
    else if(EV_DataSet(th, 14, data_4) == OK);
    else{
        EV_EnableSet();
        th->count = 0;
        switch(puls->Command){
            case 'q':
#endif USE_BCC
                /* address data set */
                PB.DR &= 0xfe;
                /* data set */
                PB.DR &= 0xfd;
                /* address clock set */
                PB.DR &= 0xfb;
                /* clock set */
                PB.DR &= 0xf7;
                /* disable set */
                PB.DR &= 0xef;
#endif

```

```

#endif

```

```

#ifdef USE_RASPBIAN

```

```

        /* address data set */
        digitalWrite(GPIO16, 0);
        /* data set */
        digitalWrite(GPIO17, 0);
        /* address clock set */
        digitalWrite(GPIO18, 0);
        /* clock set */
        digitalWrite(GPIO19, 0);
        /* disable set */
        digitalWrite(GPIO20, 0);
#endif

        delete_(th);
        break;
    default:
        break;
    }
}
return;
}

void EV_Puls(struct EV_Puls *This, Thread *th){
    This->p_ch = &This->ch;
    This->p_Command = &This->Command;
    EV_File(&This->MF);
    EV_File(&This->CF);
#ifdef USE_BCC
    PB.DDR = 0xff; /* bit7..0 out */
    PB.DR |= 0xff;
#endif
}

```

```

#ifdef USE_RASPBIAN
    if (wiringPiSetupGpio() == -1) exit(NG);
    pinMode(GPIO16, OUTPUT);
    pinMode(GPIO17, OUTPUT);
    pinMode(GPIO18, OUTPUT);
    pinMode(GPIO19, OUTPUT);
    pinMode(GPIO20, OUTPUT);
#endif

    return;
}

void OnPuls(struct EV_Puls *This, Thread *th){
    if(th->count == 0){
        /* 命令入力 */
        Command_Read(&This->CF, This->p_Command);
        /* モータ一命令解読 */
        Read(&This->MF, "Motor.txt¥0", This->p_ch);
    }
    switch(This->ch){
    case 's':
        EV_AddressDataSet(This, th, 0, 1, 1, 1, 0, 0, 0, 0);
        break;
    case 'j':
        EV_AddressDataSet(This, th, 0, 1, 1, 1, 0, 0, 0, 1);
        break;
    case 'u':
        EV_AddressDataSet(This, th, 0, 1, 1, 1, 0, 0, 1, 0);
        break;
    case 'U':

```



```
    EV_AddressDataSet(This, th, 0, 1, 1, 1, 0, 0, 1, 1);  
    break;  
case 'k':  
    EV_AddressDataSet(This, th, 0, 1, 1, 1, 0, 1, 0, 0);  
    break;  
case 'd':  
    EV_AddressDataSet(This, th, 0, 1, 1, 1, 0, 1, 0, 1);  
    break;  
case 'D':  
    EV_AddressDataSet(This, th, 0, 1, 1, 1, 0, 1, 1, 0);  
    break;  
case 'h':  
    EV_AddressDataSet(This, th, 0, 1, 1, 1, 0, 1, 1, 1);  
    break;  
case 'o':  
    EV_AddressDataSet(This, th, 0, 1, 1, 1, 1, 0, 0, 0);  
    break;  
case 'O':  
    EV_AddressDataSet(This, th, 0, 1, 1, 1, 1, 0, 0, 1);  
    break;  
case 't':  
    EV_AddressDataSet(This, th, 0, 1, 1, 1, 1, 0, 1, 0);  
    break;  
case 'c':  
    EV_AddressDataSet(This, th, 0, 1, 1, 1, 1, 0, 1, 1);  
    break;  
case 'C':  
    EV_AddressDataSet(This, th, 0, 1, 1, 1, 1, 1, 0, 0);  
    break;
```

default:

break;

}

return;

}

```
/* EV_Simulator.h */
```

```
#ifndef Panel_h
```

```
#define Panel_h
```

```
#include "Panel.h"
```

```
#endif
```

```
#ifndef Timer_h
```

```
#define Timer_h
```

```
#include "Timer.h"
```

```
#endif
```

```
#ifndef EV_Time_h
```

```
#define EV_Time_h
```

```
#include "EV_Time.h"
```

```
#endif
```

```
#ifndef EV_File_h
```

```
#define EV_File_h
```

```
#include "EV_File.h"
```

```
#endif
```

```
#ifndef EV_Display_h
```

```
#define EV_Display_h
```

```
#include "EV_Display.h"
```

```
#endif
```

```
/*=====
```

シミュレータを表す構造体

```
=====*/  
struct EV_Simulator  
{  
    char ch;  
    char *p_ch;  
    char ch2;  
    char *p_ch2;  
    char ch3;  
    char *p_ch3;  
    char str[9];  
    char *p_str;  
  
    /* 時間管理 */  
    struct EV_Time T;  
  
    /* ファイルストリーム */  
    struct EV_File SF;  
    struct EV_File CF;  
    struct EV_File MF;  
    struct EV_File LF;  
};  
  
/*=====*/  
シミュレータを表すコンストラクタとメソッドのプロトタイプ宣言  
=====*/  
void EV_Simulator(struct EV_Simulator *This, Thread *th);  
void OnSimulator(struct EV_Simulator *This, Thread *th);
```

```
/* EV_Simulator.c */
```

```
#include "C.h"
```

```
#include "EV_Simulator.h"
```

```
/*=====
```

```
シミュレータ関数
```

```
=====*/
```

```
void EV_Simulator(struct EV_Simulator *This, Thread *th)
```

```
{
```

```
    /* 初期化 */
```

```
    EV_Time(&This->T, th);
```

```
    SetCurrentTime(&This->T);
```

```
    This->p_ch = &This->ch;
```

```
    This->p_ch2 = &This->ch2;
```

```
    This->p_ch3 = &This->ch3;
```

```
    This->p_str = &This->str[0];
```

```
    This->str[8] = '¥0';
```

```
    EV_File(&This->SF);
```

```
    EV_File(&This->CF);
```

```
    EV_File(&This->MF);
```

```
    EV_File(&This->LF);
```

```
    /* モーター命令解読 */
```

```
    if(Read(&This->MF, "Motor.txt¥0", This->p_ch) == NG) return;
```

```
    /* リミットスイッチの前状態読み込み */
```

```
    if(ReadString(&This->LF, "Limit.txt¥0", This->p_str, 9) == NG) return;
```

```
    /* 籠表示 */
```

```

Disp(This->ch, This->str);
}

void OnSimulator(struct EV_Simulator *This, Thread *th)
{
    /* 終了条件 */
    Read(&This->CF, "Command.txt¥0", This->p_ch3);
    if(This->ch3 == 'q'){
        Clear();
        delete_(th);
        return;
    }

    /* モーター命令解読 */
    Read(&This->MF, "Motor.txt¥0", This->p_ch);

    /* リミットスイッチの前状態読み込み */
    ReadString(&This->LF, "Limit.txt¥0", This->p_str, 9);

    /* 停止条件 */
    Read(&This->SF, "Safety.txt¥0", This->p_ch2);
    if(This->ch2 == 's'){
        Write(&This->CF, "Command.txt¥0", 'N');
        /* 籠表示 */
        Disp(This->ch, This->str);
        return;
    }

    /* リミットスイッチの新状態作成 */

```

```
if(This->ch == 'u'){
    if(This->str[0] == 'y');
    else if(This->str[1] == 'y') This->str[1] = 'n';
    else if(This->str[2] == 'n');
    else if(This->str[3] == 'n') This->str[3] = 'y';
}
else if(This->ch == 'U'){
    if(This->str[0] == 'y');
    else if(This->str[1] == 'y');
    else if(This->str[2] == 'n') This->str[2] = 'y';
    else if(This->str[3] == 'n');
}
else if(This->ch == 'j'){
    if(This->str[0] == 'y') This->str[0] = 'n';
    else if(This->str[1] == 'y') This->str[1] = 'n';
    else if(This->str[2] == 'n') This->str[2] = 'y';
    else if(This->str[3] == 'n') This->str[3] = 'y';
}
else if(This->ch == 'd'){
    if(This->str[3] == 'y');
    else if(This->str[2] == 'y') This->str[2] = 'n';
    else if(This->str[1] == 'n');
    else if(This->str[0] == 'n') This->str[0] = 'y';
}
else if(This->ch == 'D'){
    if(This->str[3] == 'y');
    else if(This->str[2] == 'y');
    else if(This->str[1] == 'n') This->str[1] = 'y';
    else if(This->str[0] == 'n');
```

```

}
else if(This->ch == 'k'){
    if(This->str[3] == 'y') This->str[3] = 'n';
    else if(This->str[2] == 'y') This->str[2] = 'n';
    else if(This->str[1] == 'n') This->str[1] = 'y';
    else if(This->str[0] == 'n') This->str[0] = 'y';
}
else if(This->ch == 'o'){
    if(This->str[4] == 'y');
    else if(This->str[5] == 'y') This->str[5] = 'n';
    else if(This->str[6] == 'n');
    else if(This->str[7] == 'n') This->str[7] = 'y';
}
else if(This->ch == 'O'){
    if(This->str[4] == 'y');
    else if(This->str[5] == 'y');
    else if(This->str[6] == 'n') This->str[6] = 'y';
    else if(This->str[7] == 'n');
}
else if(This->ch == 'h'){
    if(This->str[4] == 'y') This->str[4] = 'n';
    else if(This->str[5] == 'y') This->str[5] = 'n';
    else if(This->str[6] == 'n') This->str[6] = 'y';
    else if(This->str[7] == 'n') This->str[7] = 'y';
}
else if(This->ch == 'c'){
    if(This->str[7] == 'y');
    else if(This->str[6] == 'y') This->str[6] = 'n';
    else if(This->str[5] == 'n');
}

```



```

        else if(This->str[4] == 'n') This->str[4] = 'y';
    }
else if(This->ch == 'C'){
    if(This->str[7] == 'y');
    else if(This->str[6] == 'y');
    else if(This->str[5] == 'n') This->str[5] = 'y';
    else if(This->str[4] == 'n');
}
else if(This->ch == 't'){
    if(This->str[7] == 'y') This->str[7] = 'n';
    else if(This->str[6] == 'y') This->str[6] = 'n';
    else if(This->str[5] == 'n') This->str[5] = 'y';
    else if(This->str[4] == 'n') This->str[4] = 'y';
}

/* リミットスイッチの新状態書き込み */
WriteString(&This->LF, "Limit.txt¥0", This->str);

/* 籠表示 */
Disp(This->ch, This->str);

return;
}

```

```
/* EV_Log.h */
```

```
#ifndef Panel_h
```

```
#define Panel_h
```

```
#include "Panel.h"
```

```
#endif
```

```
#ifndef Timer_h
```

```
#define Timer_h
```

```
#include "Timer.h"
```

```
#endif
```

```
#ifndef EV_Time_h
```

```
#define EV_Time_h
```

```
#include "EV_Time.h"
```

```
#endif
```

```
#ifndef EV_File_h
```

```
#define EV_File_h
```

```
#include "EV_File.h"
```

```
#endif
```

```
/*=====
```

```
記録を表す構造体
```

```
=====*/
```

```
struct EV_Log{
```

```
    MYSQL *conn;
```

```
    MYSQL_RES *resp;
```

```

MYSQL_ROW row;
char str_sql[255];
long mydate;
long mytime;
char ch_safety;
char str_limit[9];
char ch_command;
char ch_permitcommand;
char ch_permitturnopen;
char ch_motor;
char *p_ch_safety;
char *p_str_limit;
char *p_ch_command;
char *p_ch_permitcommand;
char *p_ch_permitturnopen;
char *p_ch_motor;
struct EV_File SF;
struct EV_File LF;
struct EV_File CF;
struct EV_File PCF;
struct EV_File PTOF;
struct EV_File MF;
};

/*=====
記録を表すプロトタイプ宣言
=====*/

void EV_Log(struct EV_Log *This); /* database接続 */
void delete_EV_Log(struct EV_Log *This); /* database切断 */

```

```
void OnLog(struct EV_Log *This, Thread *th); /* database記録 */
```

```
/* EV_Log.c */
```

```
#include "C.h"
```

```
#include "EV_Log.h"
```

```
#define DBSERVER "localhost"
```

```
#define DBID "pi"
```

```
#define DBPASSWORD "raspberry"
```

```
#define DBNAME "ev001"
```

```
#define DBPORTNO 3306
```

```
/* database接続 */
```

```
void EV_Log(struct EV_Log *This)
```

```
{
```

```
    This->conn = NULL;
```

```
    This->resp = NULL;
```

```
    This->p_ch_safety = &This->ch_safety;
```

```
    This->p_str_limit = &This->str_limit[0];
```

```
    This->p_ch_command = &This->ch_command;
```

```
    This->p_ch_permitcommand = &This->ch_permitcommand;
```

```
    This->p_ch_permitturnopen = &This->ch_permitturnopen;
```

```
    This->p_ch_motor = &This->ch_motor;
```

```
    EV_File(&This->SF);
```

```
    EV_File(&This->LF);
```

```
    EV_File(&This->CF);
```

```
    EV_File(&This->PCF);
```

```
    EV_File(&This->PTOF);
```

```
    EV_File(&This->MF);
```

```

memset(&This->str_sql[0], 0x00, sizeof(This->str_sql));
/* mysql接続 */
This->conn = mysql_init(NULL);
if(!mysql_real_connect(This->conn, DBSERVER, DBID, DBPASSWORD, DBNAME, DBPORTNO, NULL,
0))
{
    /* error */
    printf("%s¥n", mysql_error(This->conn));
    exit(NG);
}
return;
}

```

```

/* database切斷 */
void delete_EV_Log(struct EV_Log *This)
{
    mysql_free_result(This->resp);
    mysql_close(This->conn);
    return;
}

```

```

/* database記録 */
void OnLog(struct EV_Log *This, Thread *th)
{
    myDateTime(&This->mydate, &This->mytime);
    Read(&This->SF, "Safety.txt¥0", This->p_ch_safety);
    ReadString(&This->LF, "Limit.txt¥0", This->p_str_limit, 9);
    Read(&This->CF, "Command.txt¥0", This->p_ch_command);
    Read(&This->PCF, "PermitCommand.txt¥0", This->p_ch_permitcommand);
}

```

```
Read(&This->PTOF, "PermitTurnOpen.txt¥0", This->p_ch_permitturnopen);
Read(&This->MF, "Motor.txt¥0", This->p_ch_motor);
sprintf(This->str_sql, "insert into s_log(lg_ymd, lg_hms, lg_safety, lg_limit, lg_command,
lg_permitcommand, lg_permitturnopen, lg_motor) values ( %ld, %ld, '%c', '%s', '%c', '%c', '%c', '%c');",
This->mydate, This->mytime, This->ch_safety, This->p_str_limit, This->ch_command, This-
>ch_permitcommand, This->ch_permitturnopen, This->ch_motor);
mysql_query(This->conn, This->str_sql);
if(This->ch_command == 'q'){
    delete_(th);
}
return;
}
```

```
/* main.h */
```

```
#ifndef Panel_h  
#define Panel_h  
#include "Panel.h"  
#endif
```

```
#ifndef Timer_h  
#define Timer_h  
#include "Timer.h"  
#endif
```

```
#ifndef EV_Time_h  
#define EV_Time_h  
#include "EV_Time.h"  
#endif
```

```
#ifndef EV_File_h  
#define EV_File_h  
#include "EV_File.h"  
#endif
```

```
#ifndef EV_UpDown_h  
#define EV_UpDown_h  
#include "EV_UpDown.h"  
#endif
```

```
#ifndef EV_OpenClose_h  
#define EV_OpenClose_h  
#include "EV_OpenClose.h"  
#endif
```

```
#ifndef EV_Display_h  
#define EV_Display_h  
#include "EV_Display.h"  
#endif
```

```
#ifndef EV_Input_h  
#define EV_Input_h  
#include "EV_Input.h"  
#endif
```

```
#ifndef EV_Controller_h  
#define EV_Controller_h  
#include "EV_Controller.h"  
#endif
```

```
#ifndef EV_Puls_h  
#define EV_Puls_h  
#include "EV_Puls.h"  
#endif
```

```
#ifndef EV_Simulator_h  
#define EV_Simulator_h  
#include "EV_Simulator.h"  
#endif
```



```
#ifdef USE_LINUX
#ifndef EV_Log_h
#define EV_Log_h
#include "EV_Log.h"
#endif
#endif

#ifdef USE_THREAD
typedef struct tag_Count
{
#ifndef USE_BCC
    int cnt[2];
#else
    int cnt[8];
#endif
}Count;
#endif

int main(void);
```

```
/* main.c */

#include "C.h"
#include "main.h"

#ifdef USE_THREAD
Count Cnt;
int i_cnt, j_cnt;
#else
/* 擬似スレッドの擬似インスタンス宣言 */
Thread* th[2];
/* 擬似スレッドの擬似インスタンス宣言 */
Thread* th[8];
#endif
/* 擬似スレッドの擬似インスタンス宣言 */
Thread *th1[4];
Thread *th19;
Thread *th20;
Thread *th41;
Thread *th42;
Thread *th43;
Thread *th44;
Thread *th45;
#endif
#ifdef NOTUSE_FILES
EV_Status s;
#endif
```

```
/*=====
  入力オブジェクト宣言
=====*/
```

```
struct EV_Input in;
```

```
/*=====
  制御オブジェクト宣言
=====*/
```

```
struct EV_Controller cntrl;
```

```
/*=====
  送信オブジェクト宣言
=====*/
```

```
struct EV_Puls puls;
```

```
/*=====
  シミュレータオブジェクト宣言
=====*/
```

```
struct EV_Simulator simu;
```

```
#ifdef USE_LINUX
```

```
/*=====
  記録オブジェクト宣言
=====*/
```

```
struct EV_Log lg;
```

```
#endif
```

```
int main(void)
```

```

{
#ifdef USE_BCC
    char sw[4];
    int i;
    int j;
    int f;
    int cnt;
    static char buff[64];
#endif

#ifdef USE_THREAD
    Thread *th30;
    Thread *th31;
#endif

#ifdef USE_BCC
    for(i=0;i<0x7fff;i++) {}
    H8init(); /* H8 レジスタ初期化 */
    InitSCI(); /* SCI1初期化(serial) */
    InitLCD(); /* LCD初期化 */

    /* LED OFF */
    SetLED(0,0);
    SetLED(1,0);
    SetLED(2,0);
    SetLED(3,0);

    /*-----*/
    /* USB初期化 */
    InitUSB();
    INTC.ISCR &= (-1^0x20); /* IRQ_5 センソコントロ-ル Active Low */

```

```

INTC.IER |= 0x20; /* IRQ5 Enable */

/*-----*/

EnableInterrupt(); /* 割り込み許可 ccr */

f = 0;

PrintSCI("CPU MODE %02X\n",MDCR); /* MODE 6 */
PrintLCD("%fReady!3052"); /* %fはLCDクリアに利用 */

/* スイッチワーク初期化 */
sw[0] = sw[1] = sw[2] = sw[3] = 0;

#else

    printf("\nHello BCC");

#endif

#ifdef NOTUSE_FILES

    /* ファイル初期化 */
    new_EV_Status(&s);

#endif

#ifdef USE_THREAD

    /* タイマー初期化 */
    initWOVI();

    /* 2秒待機 */
    SleepMSec(2000);

    /* LEDTEST */
    th30 = new_Thread(30);
    th31 = new_Thread(31);
    Start(th30);
    Start(th31);
    for(;;)
    {

```

```

        /* タイマー呼び出し */
#ifdef USE_CENTOS
        wovi(25000000.0);
#else
#ifdef USE_RASPBIAN
        wovi(25000000.0);
#else
        wovi();
#endif
#endif
        if(Thread_checkAllDelete() == OK)
        {
            break;
        }
    }
#endif

    Clear();
    Printf(Pannel, "NEXT      ");
    /* 2秒待機 */
    SleepMSec(2000);
    Clear();
#ifdef USE_BCC
    for(;;)
    {
        /*-----*/
        /* ボタンが押された時にLCD,SCI,USBにメッセージを送る */
        for(j=0;j<4;j++)
        {

```

```

i = GetSW(j);
if( ((sw[j]^1) & i) ) /* sw = off->onで条件成立 */
{
    SetLED(j,1); /* LED押した瞬間点灯 */
    sprintf(buff,"sw%u",j+1);
    PrintSCI("%s¥n",buff);
    /* NULL(0x00)まで送信 */
    write_buff(buff,strlen(buff)+1);
    PrintLCD(buff);
}
else SetLED(j,0);
sw[j] = i;
}

/*-----*/
/* HOSTからのシリアル入力をLCD,USBに送る */
if(ScanSCI()) /* SCIに受信データあり? */
{
    i = GetSCI(); /* シリアル入力 */
    PutLCD(i); /* LCD出力 */
    buff[0] = i;
    write_buff(buff,1); /* USB出力 */
}

/*-----*/
/* USBからデータを受信した場合、そのままHOSTへリダイレクトする */
if(get_inbufflen()) /* 受信データあり? */
{
    /* データ取得(buffサイズは64byteまで) */

```

```
    cnt = read_buff(buff,64);  
    PrintLCD("%f"); /* LCDクリア */  
    PrintLCD(buff); /* LCDへ表示 */  
    PrintSCI(buff); /* シリアル出力 */  
    write_buff(buff,cnt); /* USBへリダイレクト */  
}
```

```
/*-----*/
```

```
/* 動作確認のため点滅 */
```

```
SetLED(3,f);
```

```
f ^= 1;
```

```
for(i=0;i<10000;i++) {} /* 適当にウェイト */
```

```
}
```

```
#else
```

```
printf("END");
```

```
/* 5秒待機 */
```

```
SleepMSec(5000);
```

```
return OK;
```

```
#endif
```

```
}
```

```
#ifdef USE_THREAD
```

```
/*
```

```
* 擬似スレッドの擬似メソッド関数
```

```
*/
```

```
/* public void paint(Graphics g)の代用 */
```

```
void Repaint(void)
```

```
{
```

```
#ifndef USE_BCC
```



```

    int i;

#else

    int i,j;

#endif

    Clear();

#ifndef USE_BCC

    for(i = 0; i < Cnt.cnt[0]; i++)
    {
        Printf(Panel, " ");
    }
    Printf(Panel, "<1>");
    Printf(Panel, "¥n");
    for(i = 0; i < Cnt.cnt[1]; i++)
    {
        Printf(Panel, " ");
    }
    Printf(Panel, "<2>");

#else

    for(i = 0; i < 8; i++)
    {
        for(j = 0; j < Cnt.cnt[i]; j++)
        {
            printf(" ");
        }
        printf("<%d>", (i + 1));
        for(j = 0; j < 13 - Cnt.cnt[i]; j++)
        {
            printf(" ");
        }
    }

```

```

        printf("|");
        printf("¥n");
    }
#endif

    return;
}

/*
 * 疑似スレッドの疑似メソッド関数
 */
/* スレッドのpublic void run()の代用 */
void Run(Thread *This)
{
    int i;
    Thread *th1;
#ifdef USE_BCC
    char key = '¥0';
#else
    int j;
    char sw[4];
    /* スイッチワーク初期化 */
    sw[0] = sw[1] = sw[2] = sw[3] = 0;
#endif
    if(This->ID == 1)
    {
        Repaint();
#ifdef USE_BCC
        Cnt.cnt[0]++;

```

```

        nextRun(This, (((rand() % 9) + 10) * 100));
#else
    if(kbhit())
    {
#ifdef USE_LINUX
        key = (char) getchar();
#else
        key = (char) getche();
#endif
    }
    if(key == 'r')
    {
        Cnt.cnt[0]++;
    }
    nextRun(This, (((rand() % 9) + 10) * 15));
    while(kbhit())
    {
#ifdef USE_LINUX
        key = (char) getchar();
#else
        key = (char) getche();
#endif
    }
}
else if(This->ID == 2)
{
    Repaint();
}
#endif USE_BCC

```

```

        Cnt.cnt[1]++;
        nextRun(This, (((rand() % 9) + 10) * 100));
#else
        if(kbhit())
        {
#ifdef USE_LINUX
                key = (char) getchar();
#else
                key = (char) getche();
#endif
        }
        if(key == 'l')
        {
                Cnt.cnt[1]++;
        }
        nextRun(This, (((rand() % 9) + 10) * 15));
        while(kbhit())
        {
#ifdef USE_LINUX
                key = (char) getchar();
#else
                key = (char) getche();
#endif
        }
}
#endif
}
#endif
#ifdef USE_BCC
        else if(((This->ID) >= 3) && ((This->ID) <= 8))
        {

```

```

        Repaint();
        Cnt.cnt[(This->ID) - 1]++;
        nextRun(This, (((rand() % 9) + 10) * 40));
    }
#endif

else if(This->ID == 11)
{
    if(This->count == 1)
    {
        Clear();
        Printf(Panel, "<1>1st    ");
        countUpNextRun(This, (1900 * 1));
    }
    else if(This->count == 2)
    {
        Clear();
        Printf(Panel, "<1>2nd");
        Printf(Panel, "<1>Stop ");
        Stop(This);
    }
    else if(This->count == 3)
    {
        Clear();
        Printf(Panel, "<1>3rd    ");
        countUpNextRun(This, (1500 * 1));
    }
    else if(This->count == 4)
    {
        Clear();

```

```

        Printf(Panel, "<1>Stop ");
        Stop(This);
    }
}
else if(This->ID == 12)
{
    if(This->count == 1)
    {
        Clear();
        Printf(Panel, "<2>1st ");
        countUpNextRun(This, (1700 * 2));
    }
    else if(This->count == 2)
    {
        Clear();
        Printf(Panel, "<2>2nd ");
        countUpNextRun(This, (1700 * 2));
    }
    else if(This->count == 3)
    {
        Clear();
        Printf(Panel, "<2>3rd ");
        countUpNextRun(This, (1700 * 2));
    }
    else
    {
        Clear();
        Printf(Panel, "<2>Stop");
    }
}

```

```

        Stop(This);
        delete_(This);
    }
}
else if(This->ID == 13)
{
    if(This->count == 1)
    {
        Clear();
        Printf(Pannel, "<3>1st    ");
        countUpNextRun(This, (1700 * 3));
    }
    else if(This->count == 2)
    {
        Clear();
        Printf(Pannel, "<3>2nd    ");
        countUpNextRun(This, (1700 * 3));
    }
    else if(This->count == 3)
    {
        Clear();
        Printf(Pannel, "<3>3rd    ");
        countUpNextRun(This, (1700 * 3));
    }
    else
    {
        Clear();
        Printf(Pannel, "<3>Stop");
        Stop(This);
    }
}

```

```

        delete_(This);
    }
}
else if(This->ID == 14)
{
    if(This->count == 1)
    {
        Clear();
        Printf(Panel, "<4>1st ");
        countUpNextRun(This, (1500 * 4));
    }
    else if(This->count == 2)
    {
        Clear();
        Printf(Panel, "<4>2nd");
        countUpNextRun(This, (1500 * 4));

        Printf(Panel, "<1>Start ");
        th11 = Thread_Start(11);
        countUpNextRun(th11, (1500 * 1));
    }
    else if(This->count == 3)
    {
        Clear();
        Printf(Panel, "<4>3rd ");
        countUpNextRun(This, (1500 * 4));
    }
    else if(This->count == 4)
    {

```



```

        th11 = Thread_getThread(11);
        if(th11 != NULL)
        {
            delete_(th11);
        }

        Printf(Panel, "<4>Sto");
        Stop(This);
        delete_(This);
    }
}
else if(This->ID == 19)
{
#ifdef USE_LINUX
    nextRun(This, 4000);
#else
    nextRun(This, 1);
#endif

#ifdef USE_BCC
    /* ボタンが押された時にLCD,SCI,USBにメッセージを送る */
    for(j=0;j<4;j++)
    {
        i = GetSW(j);
        if( ((sw[j]^1) & i) ) /* sw = off->onで条件成立 */
        {
            Thread_Toggle(j + 20);
            nextRun(This, 1000);
        }
    }
}

```

```
#else
```

```
key = '¥0';
```

```
key = GetChar(key);
```

```
if(key == '1')
```

```
{
```

```
    Thread_Toggle(21);
```

```
}
```

```
else if(key == '2')
```

```
{
```

```
    Thread_Toggle(22);
```

```
}
```

```
else if(key == '3')
```

```
{
```

```
    Thread_Toggle(23);
```

```
}
```

```
else if(key == '4')
```

```
{
```

```
    Thread_Toggle(24);
```

```
}
```

```
else if(key == '5')
```

```
{
```

```
    Thread_Toggle(25);
```

```
}
```

```
else if(key == '6')
```

```
{
```

```
    Thread_Toggle(26);
```

```
}
```

```
else if(key == '7')
{
    Thread_Toggle(27);
}
else if(key == '8')
{
    Thread_Toggle(28);
}
else if(key == '9')
{
    Thread_Toggle(29);
}
else if(key == '0')
{
    Thread_Toggle(20);
}
```

```
#endif
```

```
}
else if(This->ID == 20)
{
    Printf(Panel, "0");
    countUpNextRun(This, 2000);
}
else if(This->ID == 21)
{
    Printf(Panel, "1");
    countUpNextRun(This, 2000);
}
else if(This->ID == 22)
```

```

{
    Printf(Panel, "2");
    countUpNextRun(This, 2000);
}
else if(This->ID == 23)
{
    Printf(Panel, "3");
    countUpNextRun(This, 2000);
}
#endif USE_BCC
else if((This->ID >= 24) && (This->ID <= 29))
{
    printf("%d", This->ID - 20);
    nextRun(This, 2000);
}
#endif
#endif USE_BCC
else if(This->ID == 30)
{
    if(This->count == 0)
    {
        This->count++;
        PB.DR &= 0x0e;
        nextRun(This, 1000);
    }
    else if(This->count == 1)
    {
        This->count--;
        PB.DR |= 0x01;
    }
}

```

```

        nextRun(This, 1000);
    }
}
#endif

else if(This->ID == 31)
{
    if(This->count == 0)
    {
        /* 第1部分 */
#ifdef USE_BCC
        printf("%nThread Ready GO! There are 8 cources on a race.");
        printf("%nThere are 14 cells to a GOAL.");
        printf("%nFor the <1> cource, You click a 'R' button.");
        printf("%nFor the <2> cource, You click a 'L' button.");
#endif
#endif

        countUpNextRun(This, 0);

    #else

        /* 5秒待機 */
        countUpNextRun(This, 5000);

    #endif

    }

    else if(This->count == 1)
    {
        /* 擬似スレッド開始 */
        Printf(Pannel, "%n");
        Printf(Pannel, "Thread Ready GO!");
        /* 2秒待機 */
        countUpNextRun(This, 2000);
    }
}

```

```

    }
    else if(This->count == 2)
    {
#ifdef USE_BCC
        /* 擬似スレッドの擬似インスタンス初期化 */
        for(i = 0; i < 2; i++)
        {
            th[i] = new_Thread(i + 1);
        }
#else
        /* 擬似スレッドの擬似インスタンス初期化 */
        for(i = 0; i < 8; i++)
        {
            th[i] = new_Thread(i + 1);
        }
#endif

        countUpNextRun(This, 1);
    }
    else if(This->count == 3)
    {
#ifdef USE_BCC
        if(Cnt.cnt[0] >= 13)
        {
            i_cnt = 1;
            This->count++;
        }
        else if(Cnt.cnt[1] >= 13)
        {

```

```

        i_cnt = 2;
        This->count++;
    }
#else
    i_cnt = Cnt.cnt[0];
    j_cnt = 0;
    for(i = 1; i < 8; i++)
    {
        if(i_cnt < Cnt.cnt[i])
        {
            i_cnt = Cnt.cnt[i];
            j_cnt = i;
        }
    }
    if(i_cnt >= 13) This->count++;
#endif

    nextRun(This, 1);
}
else if(This->count == 4)
{
    Clear();
    if(i_cnt == 1)
    {
        Printf(Panel, "GOAL!<1>WON  ");
    }
    else if(i_cnt == 2)
    {
        Printf(Panel, "GOAL!<2>WON  ");
    }
}

```

```

#ifdef USE_BCC
    else
    {
        printf("GOAL!¥n<%d>WON", (j_cnt + 1));
    }
#endif

#ifndef USE_BCC
    delete_(th[0]);
    delete_(th[1]);
#else
    for(i = 0; i < 8; i++)
    {
        delete_(th[i]);
    }
#endif

#ifdef USE_LINUX
    This->count = 15;
#else
    This->count++;
#endif

    /* 2秒待機 */
    nextRun(This, 2000);
}
else if(This->count == 5)
{
    Clear();
    Printf(Pannel, "NEXT    ");
    /* 2秒待機 */
    countUpNextRun(This, 2000);
}

```



```

}
else if(This->count == 6)
{
    Clear();
    /* 第2部分 */
    Printf(Pannel, "CountUp    ");
    /* 2秒待機 */
    countUpNextRun(This, 2000);
}
else if(This->count == 7)
{
    /* 疑似スレッド開始 */
    Clear();
    /* 疑似スレッドの疑似インスタンス初期化 */
    for(i = 0; i < 4; i++)
    {
        th1[i] = new_Thread(i + 11);
    }
    countUpNextRun(This, 1);
}
else if(This->count == 8)
{
    if(Thread_checkStayAnother() == 2)
    {
        This->count++;
    }
    nextRun(This, 1);
}
else if(This->count == 9)

```

```

{
    /* 2秒待機 */
    countUpNextRun(This, 2000);
}
else if(This->count == 10)
{
    Clear();
    Printf(Pannel, "NEXT ");
    /* 2秒待機 */
    countUpNextRun(This, 2000);
}
else if(This->count == 11)
{
    Clear();
    /* 第3部分 */
    Printf(Pannel, "Toggle ");
    /* 2秒待機 */
    countUpNextRun(This, 2000);
}
else if(This->count == 12)
{
    th19 = new_Thread(19);
    Start(th19);
    th20 = new_Thread(20);
    Start(th20);
    countUpNextRun(This, 1);
}
else if(This->count == 13)

```

```

{
    if(Thread_checkStayAnother() == 3)
    {
        delete_(th19);
        This->count++;
    }
    nextRun(This, 1);
}
else if(This->count == 14)
{
    countUpNextRun(This, 2000);
}
else if(This->count == 15)
{
    Clear();
    Printf(Pannel, "NEXT    ");
    /* 2秒待機 */
    countUpNextRun(This, 2000);
}
else if(This->count == 16)
{
    Clear();
    /* 第4部分 */
    Printf(Pannel, "Hello EV    ");
    /* 2秒待機 */
    countUpNextRun(This, 2000);
}
else if(This->count == 17)
{

```

```
/* 擬似スレッドの擬似インスタンス初期化 */
```

```
th41 = new_Thread(41);
```

```
th42 = new_Thread(42);
```

```
th43 = new_Thread(43);
```

```
th44 = new_Thread(44);
```

```
#ifdef USE_LINUX
```

```
th45 = new_Thread(45);
```

```
#endif
```

```
delete_(This);
```

```
/* LEDTEST */
```

```
delete_(Thread_getThread(30));
```

```
}
```

```
}
```

```
else if(This->ID == 41)
```

```
{
```

```
nextRun(This, 100);
```

```
OnInput(&in, This);
```

```
}
```

```
else if(This->ID == 42)
```

```
{
```

```
nextRun(This, 100);
```

```
OnController(&cntrl, This);
```

```
}
```

```
else if(This->ID == 43)
```

```
{
```

```
nextRun(This, 110);
```

```
OnPuls(&puls, This);
```

```

    }

    else if(This->ID == 44)
    {
        nextRun(This, 2000);
        OnSimulator(&simu, This);
    }

#ifdef USE_LINUX
    else if(This->ID == 45)
    {
        nextRun(This, 2000);
        OnLog(&lg, This);
    }
#endif

    return;
}

/* スレッドのコンストラクタのpublic void init()の代用 */
void Init(Thread *This)
{
    if(This->ID == 1)
    {
        Cnt.cnt[0] = 0;
        nextRun(This, (((rand() % 9) + 10) * 30));
    }
    else if(This->ID == 2)
    {
        Cnt.cnt[1] = 0;
        nextRun(This, (((rand() % 9) + 10) * 30));
    }
}

```

```
else if((This->ID >= 3) && (This->ID <= 8))
{
    Cnt.cnt[(This->ID) - 1] = 0;
    nextRun(This, (((rand() % 9) + 10) * 200));
}
else if(This->ID == 11)
{
    Printf(Panel, "<1>Init");
    countUpNextRun(This, (1500 * 1));
}
else if(This->ID == 12)
{
    Printf(Panel, "<2>Init ");
    countUpNextRun(This, (1500 * 2));
}
else if(This->ID == 13)
{
    Printf(Panel, "¥n");
    Printf(Panel, "<3>Init");
    countUpNextRun(This, (1500 * 3));
}
else if(This->ID == 14)
{
    Printf(Panel, "<4>Init ");
    countUpNextRun(This, (1500 * 4));
}
else if(This->ID == 20)
{
```

```

    Clear();
    Printf(Panel, "<0>Init    ");
    Printf(Panel, "¥n");
    countUpNextRun(This, 2000);
}
else if(This->ID == 21)
{
    Clear();
    Printf(Panel, "<1>Init    ");
    Printf(Panel, "¥n");
    countUpNextRun(This, 2000);
}
else if(This->ID == 22)
{
    Clear();
    Printf(Panel, "<2>Init    ");
    Printf(Panel, "¥n");
    countUpNextRun(This, 2000);
}
else if(This->ID == 23)
{
    Clear();
    Printf(Panel, "<3>Init    ");
    Printf(Panel, "¥n");
    countUpNextRun(This, 2000);
}
#endif USE_BCC
else if((This->ID >= 24) && (This->ID <= 29))
{

```

```

        printf("¥n<%d>Init¥n", This->ID - 20);
        nextRun(This,2000);
    }
#endif

#ifndef USE_BCC
    else if(This->ID == 30)
    {
        PB.DDR = 0xff; /* bit7..0 out */
        PB.DR |= 0xff;
    }
#endif

    else if(This->ID == 41)
    {
        nextRun(This, 100);
        EV_Input(&in);
    }
    else if(This->ID == 42)
    {
        nextRun(This, 100);
        EV_Controller(&cntrl, This);
    }
    else if(This->ID == 43)
    {
        nextRun(This, 110);
        EV_Puls(&puls, This);
    }
    else if(This->ID == 44)
    {
        nextRun(This, 2000);
    }

```



```

        EV_Simulator(&simu, This);
    }

#ifdef USE_LINUX
    else if(This->ID == 45)
    {
        nextRun(This,1);
        EV_Log(&lg);
    }
#endif

    return;
}

/* スレッドのデストラクタの代用 */
void Destroy(Thread *This)
{
    if(This->ID == 11)
    {
        Clear();
        Printf(Pannel, "<1>Destroy");
    }
    else if(This->ID == 12)
    {
        Printf(Pannel, "<2>Destro");
    }
    else if(This->ID == 13)
    {
        Printf(Pannel, "<3>Destro");
    }
    else if(This->ID == 14)

```

```
{
    Printf(Panel, "¥n");
    Printf(Panel, "<4>Destroy  ");
}
if(This->ID == 20)
{
    Clear();
    Printf(Panel, "<0>Destroy  ");
    Printf(Panel, "¥n");
}
else if(This->ID == 21)
{
    Clear();
    Printf(Panel, "<1>Destroy  ");
    Printf(Panel, "¥n");
}
else if(This->ID == 22)
{
    Clear();
    Printf(Panel, "<2>Destroy  ");
    Printf(Panel, "¥n");
}
else if(This->ID == 23)
{
    Clear();
    Printf(Panel, "<3>Destroy  ");
    Printf(Panel, "¥n");
}
```

```

#ifdef USE_BCC

    else if((This->ID >= 24) && (This->ID <= 29))
    {
        printf("¥n<%d>Destroy¥n", This->ID - 20);
    }

#endif

#ifdef USE_LINUX

    else if(This->ID == 45)
    {
        delete_EV_Log(&lg);
    }

#endif

    return;
}

#endif

#ifdef USE_BCC

/*=====

                                LEDコントロール

-----

int SetLED(int no,int onoff)

int    no        LEDナンバー 0~3
int    onoff     0=OFF,1=ON
戻り値          以前のLEDの状態 (0=OFF,else=ON)

LEDをコントロールします。

=====*/

int SetLED(int no,int onoff)

```

```

{
    int f;
    f = PB.DR&(1<<no);
    if( onoff == 0 ) PB.DR |= (1<<no); /* off (1) */
    else PB.DR &= 0xff^(1<<no); /* on (0) */
    return( f );
}

```

```

/*=====

```

SW状態取得

```

-----

```

```

int GetSW(int no)

```

```

int      no          SWナンバー 0~3
戻り値   SWの状態(0=OFF,else=ON)

```

SWの状態を取得します。

```

=====*/

```

```

int GetSW(int no)

```

```

{
    return( ((PA.DR&(1<<no))?0:1) );
}

```

```

/*=====

```

H8初期化

```

-----

```

BUSモードや、ポートの初期化

```

P1  bit1  BUS      USB A0

```

P3	BUS	USB D7..0
P6 bit4	BUS	USB RD
P6 bit5	BUS	USB WR
P8 bit2	BUS	USB CS
P9 bit5	BUS	USB INT(IRQ5)
P9 bit3	BUS	RS232C
P9 bit1	BUS	RS232C
PA bit0..3	IN	SW0..3
PB bit0..3	OUT	LED0..3 LCD DB4..7
PB bit4	OUT	LCD RS
PB bit7	OUT	LCD E

=====*/

```
void H8init()
```

```
{
```

```
    BSC.ABWCR = 0x06; /* 8bit BUS MODE */
```

```
    P1.DDR = 0xff; /* all OUT */
```

```
    P2.DDR = 0xff; /* all OUT */
```

```
    P2.PCR = 0x00; /* Pull up off */
```

```
    P5.DDR = 0xff; /* all OUT */
```

```
    P5.PCR = 0x00; /* Pull up off */
```

```
    P6.DDR = 0xff; /* all OUT */
```

```
    P9.DDR = 0xdf; /* Bit5 IN */
```

```
    P8.DDR = 0xff; /* all OUT */
```

```
    PA.DDR = 0xf0; /* bit7..4 out , bit3..0 in */
```

```
    PB.DDR = 0xff; /* bit7..0 out */
```

```
}
```

```
#endif
```


実行環境

```
drop database if exists ev001;
```

```
create database ev001;
```

```
use ev001;
```

```
GRANT SELECT ON ev001.* TO pi@localhost IDENTIFIED BY 'raspberry';
```

```
GRANT UPDATE ON ev001.* TO pi@localhost IDENTIFIED BY 'raspberry';
```

```
GRANT INSERT ON ev001.* TO pi@localhost IDENTIFIED BY 'raspberry';
```

```
GRANT DELETE ON ev001.* TO pi@localhost IDENTIFIED BY 'raspberry';
```

```
DROP TABLE IF EXISTS s_log;
```

```
CREATE TABLE s_log (
```

```
lg_ymd decimal(8) not null comment '日付',
```

```
lg_hms decimal(6) not null comment '時刻',
```

```
lg_safety varchar(1) not null comment '安全スイッチ',
```

```
lg_limit varchar(9) not null comment 'リミットスイッチ',
```

```
lg_command varchar(1) not null comment '命令入力',
```

```
lg_permitcommand varchar(1) not null comment '命令許可',
```

```
lg_permitturnopen varchar(1) not null comment '反転開許可',
```

```
lg_motor varchar(1) not null comment 'モーター出力',
```

```
PRIMARY KEY(lg_ymd,lg_hms)
```

```
)
```

```
ENGINE = INNODB
```

```
Default Charset = UTF8
```

```
COLLATE = UTF8_BIN
```

```
COMMENT='ログ'
```

```
;
```



```
-I"c:\borland\Bcc55\include"  
-L"c:\borland\Bcc55\lib"
```

-L"c:\borland\Bcc55\Lib"

```

# makefile.mak
CC = bcc32
LL = ilink32
INCLUDE = -I"C:\borland\bcc55\Include"
LIB = -L"C:\borland\bcc55\Lib"
CFLAGS = -O2 -w -tWC -D"USE_BCC"
LFLAGS = /Tpe
TARGET = main.exe
OBJS = Panel.obj Timer.obj EV_Time.obj EV_File.obj EV_UpDown.obj EV_OpenClose.obj EV_Display.obj
EV_Input.obj EV_Controller.obj EV_Puls.obj EV_Simulator.obj main.obj
$(TARGET): $(OBJS)
    $(LL) $(LFLAGS) $(LIB) \
    $(OBJS) c0x32.obj,$(TARGET),,cw32.lib import32.lib
main.obj : main.c main.h EV_Simulator.h EV_Puls.h EV_Controller.h EV_Input.h EV_Display.h EV_OpenClose.h
EV_UpDown.h EV_File.h EV_Time.h Timer.h Panel.h C.h
    $(CC) $(CFLAGS) -c main.c
EV_Simulator.obj : EV_Simulator.c EV_Simulator.h EV_Display.h EV_File.h EV_Time.h Timer.h Panel.h C.h
    $(CC) $(CFLAGS) -c EV_Simulator.c
EV_Puls.obj : EV_Puls.c EV_Puls.h EV_File.h EV_Time.h Timer.h Panel.h C.h
    $(CC) $(CFLAGS) -c EV_Puls.c
EV_Controller.obj : EV_Controller.c EV_Controller.h EV_OpenClose.h EV_UpDown.h EV_File.h EV_Time.h
Timer.h Panel.h C.h
    $(CC) $(CFLAGS) -c EV_Controller.c
EV_Input.obj : EV_Input.c EV_Input.h EV_File.h EV_Time.h Timer.h Panel.h C.h
    $(CC) $(CFLAGS) -c EV_Input.c
EV_Display.obj : EV_Display.c EV_Display.h EV_File.h EV_Time.h Timer.h Panel.h C.h
    $(CC) $(CFLAGS) -c EV_Display.c
EV_OpenClose.obj : EV_OpenClose.c EV_OpenClose.h EV_UpDown.h EV_File.h EV_Time.h Timer.h Panel.h
C.h
    $(CC) $(CFLAGS) -c EV_OpenClose.c
EV_UpDown.obj : EV_UpDown.c EV_UpDown.h EV_File.h EV_Time.h Timer.h Panel.h C.h
    $(CC) $(CFLAGS) -c EV_UpDown.c
EV_File.obj : EV_File.c EV_File.h Panel.h C.h
    $(CC) $(CFLAGS) -c EV_File.c
EV_Time.obj : EV_Time.c EV_Time.h Timer.h Panel.h C.h
    $(CC) $(CFLAGS) -c EV_Time.c
Timer.obj : Timer.c Timer.h Panel.h C.h
    $(CC) $(CFLAGS) -c Timer.c
Panel.obj : Panel.c Panel.h C.h
    $(CC) $(CFLAGS) -c Panel.c
clean:
    del *.obj
    del main.tds
    del main.ilc
    del main.ild
    del main.ilf
    del main.ils

```

```
; asmfile.src
```

```
.CPU 300HA
```

```
.SECTION V, CODE, LOCATE=H'000000
```

```
; C言語の関数 を参照
```

```
.IMPORT _main ; C言語の関数main を参照
```

```
.IMPORT _usb_int ; C言語の関数usb_int を参照
```

```
.IMPORT _InterruptITU0 ; C言語の関数InterruptITU0 を参照
```

```
;-----
```

```
; リセットベクタ の 転送先ラベル が _start になっています
```

```
; リセットベクタ
```

```
.DATA.L _start
```

```
;-----
```

```
; リセットベクタ に続く1番から60番までの 割り込みベクタ
```

```
; について、使用しない 割り込みベクタ は ラベルint_error
```

```
; に転送されます
```

```
; 割り込みベクタ
```

```
; 1 Reserved
```

```
_INT_Reserved1: .DATA.L int_error
```

```
; 2 Reserved
```

```
_INT_Reserved2: .DATA.L int_error
```

```
; 3 Reserved
```

```
_INT_Reserved3: .DATA.L int_error
```

```
; 4 Reserved
```

```
_INT_Reserved4: .DATA.L int_error
```

; 5 Reserved

_INT_Reserved5: .DATA.L int_error

; 6 Reserved

_INT_Reserved6: .DATA.L int_error

; 7 NMI

_INT_NMI: .DATA.L int_error

; 8 TRAP

_INT_TRAP1: .DATA.L int_error

; 9 TRAP

_INT_TRAP2: .DATA.L int_error

; 10 TRAP

_INT_TRAP3: .DATA.L int_error

; 11 TRAP

_INT_TRAP4: .DATA.L int_error

; 12 IRQ0

IRQ0: .DATA.L int_error

; 13 IRQ1

IRQ1: .DATA.L int_error

; 14 IRQ2

IRQ2: .DATA.L int_error

; 15 IRQ3

IRQ3: .DATA.L int_error

; 16 IRQ4

IRQ4: .DATA.L int_error

; 17 IRQ5

IRQ5: .DATA.L usb_interrupt ; USB割り込み

; 18 Reserved

_INT_Reserved18: .DATA.L int_error

; 19 Reserved

_INT_Reserved19: .DATA.L int_error
; 20 WOVI
_INT_WOVI: .DATA.L int_error
; 21 CMI
_INT_CMI: .DATA.L int_error
; 22 Reserved
_INT_Reserved22: .DATA.L int_error
; 23 Reserved
_INT_Reserved23: .DATA.L int_error
; 24 IMIA0
_INT_IMIA0: .DATA.L int_error
; 25 IMIB0
_INT_IMIB0: .DATA.L int_error
; タイマ0割り込み は、 ラベル_ITU_OVI_0 に転送されます
; 26 OVI0
_INT_OVI0: .DATA.L _ITU_OVI_0 ; タイマ0割り込み
; 27 Reserved
_INT_Reserved27: .DATA.L int_error
; 28 IMIA1
_INT_IMIA1: .DATA.L int_error
; 29 IMIB1
_INT_IMIB1: .DATA.L int_error
; 30 OVI1
_INT_OVI1: .DATA.L int_error
; 31 Reserved
_INT_Reserved31: .DATA.L int_error
; 32 IMIA2
_INT_IMIA2: .DATA.L int_error
; 33 IMIB2

_INT_IMIB2: .DATA.L int_error

; 34 OVI2

_INT_OVI2: .DATA.L int_error

; 35 Reserved

_INT_Reserved35: .DATA.L int_error

; 36 IMIA3

_INT_IMIA3: .DATA.L int_error

; 37 IMIB3

_INT_IMIB3: .DATA.L int_error

; 38 OVI3

_INT_OVI3: .DATA.L int_error

; 39 Reserved

_INT_Reserved39: .DATA.L int_error

; 40 IMIA4

_INT_IMIA4: .DATA.L int_error

; 41 IMIB4

_INT_IMIB4: .DATA.L int_error

; 42 OVI4

_INT_OVI4: .DATA.L int_error

; 43 Reserved

_INT_Reserved43: .DATA.L int_error

; 44 DEND0A

_INT_DEND0A: .DATA.L int_error

; 45 DEND0B

_INT_DEND0B: .DATA.L int_error

; 46 DEND1A

_INT_DEND1A: .DATA.L int_error

; 47 DEND1B

_INT_DEND1B: .DATA.L int_error

; 48 Reserved

_INT_Reserved48: .DATA.L int_error

; 49 Reserved

_INT_Reserved49: .DATA.L int_error

; 50 Reserved

_INT_Reserved50: .DATA.L int_error

; 51 Reserved

_INT_Reserved51: .DATA.L int_error

; 52 ERI0

_INT_ERI0: .DATA.L int_error

; 53 RXI0

_INT_RXI0: .DATA.L int_error

; 54 TXI0

_INT_TXI0: .DATA.L int_error

; 55 TEI0

_INT_TEI0: .DATA.L int_error

; 56 ERI1

_INT_ERI1: .DATA.L int_error

; 57 RXI1

_INT_RXI1: .DATA.L int_error

; 58 TXI1

_INT_TXI1: .DATA.L int_error

; 59 TEI1

_INT_TEI1: .DATA.L int_error

; 60 ADI

_INT_ADI: .DATA.L int_error

;-----


```
.SECTION P, CODE, ALIGN=2
```

```
; _start のラベルから処理を開始
```

```
; リセットベクタの転送先
```

```
_start:
```

```
mov.l #H'0FFFF10, er7
```

```
; 初期化付きデータを使用する場合、RAMに転送する
```

```
; c_thread.MAPのメモリアドレス使用状況を見る
```

```
; メモリアドレスが重複するとコンパイルエラーになる
```

```
; linkfile.subもD(99C0), C(9A00)等必要があれば合わせる
```

```
mov.l #H'99C0, er0 ; 転送元(99C0)
```

```
mov.l #H'0FFE000, er1 ; 転送先
```

```
mov.l #DATA_END, er2 ; 転送終了
```

```
init_loop:
```

```
cmp.l er1, er2
```

```
beq init_end
```

```
mov.b @er0+, r3l
```

```
mov.b r3l, @er1
```

```
inc.l #1, er1
```

```
bra init_loop
```

```
init_end:
```

```
; C言語の関数mainを呼び出しています
```

```
; C言語の関数mainは、void main(void);という形で、
```

```
; main.cに記述があります
```

```
jsr @_main
```

```
; 割り込み未使用
```

```
int_error:
```

```
; rte (returnと同じ意味)で終了
```

```
rte
```

;------

; USB割り込み からの転送先

usb_interrupt:

; スタック 退避

push.l er0

push.l er1

push.l er2

push.l er3

push.l er4

push.l er5

push.l er6

; C言語の関数usb_int を呼び出しています

jsr @_usb_int

; スタック 戻

pop.l er6

pop.l er5

pop.l er4

pop.l er3

pop.l er2

pop.l er1

pop.l er0

; 終了

rte

;------

; タイマ0割り込み からの転送先

_ITU_OVI_0:

; スタック 退避

push.l er0

push.l er1

push.l er2

push.l er3

push.l er4

push.l er5

push.l er6

; C言語の関数InterruptITU0 を呼び出しています

; C言語の関数InterruptITU0 は void InterruptITU0(void);

; という形で、 Timer.h Timer.c に記述があります

jsr @_InterruptITU0

; スタック 戻

pop.l er6

pop.l er5

pop.l er4

pop.l er3

pop.l er2

pop.l er1

pop.l er0

; 終了

rte

;

; C言語から

; _EnableInterrupt (割り込み許可)

; _DisableInterrupt (割り込み禁止)

; を呼び出せるようにしています

; C言語の Panel.h に 外部参照プロトタイプ宣言 があります

; extern void EnableInterrupt(void);

; extern void DisableInterrupt(void);

```
; C言語からの呼び出し名は、
; EnableInterrupt();
; DisableInterrupt();
; です
; 割り込み許可、禁止ルーチン
.EXPORT _EnableInterrupt,_DisableInterrupt
_EnableInterrupt:
andc.b #H'3f,ccr
rts
_DisableInterrupt:
orc.b #H'c0,ccr
rts

;-----
.SECTION D,DATA

.SECTION B,DATA
DATA_END: .RES.W 1

.END
```

OUTPUT c_thread

PRINT c_thread

INPUT asmfile, main, EV_Simulator, EV_Puls, EV_Controller, EV_Input, EV_Display, EV_OpenClose,
EV_UpDown, EV_File, EV_Time, Timer, Panel, sci, lcd, usb

LIB c:\h8\akic\c38hab

START R(0FFE000), P(200), D(99C0), C(9A00)

ROM (D, R)

EXIT

```
@rem build.bat
C:
set bccDir="C:\borland\bcc55\Bin"
set akih8asmDir="c:\h8\akiasm"
set akih8cDir="c:\h8\akic"
set path=%bccDir%;%path%
set path=%asih8cDir%;%asih8asmDir%;%path%
set CurrentDir="%~dp0"
cd %CurrentDir%
del error.txt
make -f makefile.mak >> error.txt
make -f makefile.mak clean >> error.txt
cc38h.exe -cpu=300ha -include=%asih8cDir% usb.c >> error.txt
cc38h.exe -cpu=300ha -include=%asih8cDir% sci.c >> error.txt
cc38h.exe -cpu=300ha -include=%asih8cDir% lcd.c >> error.txt
cc38h.exe -cpu=300ha -include=%asih8cDir% Panel.c >> error.txt
cc38h.exe -cpu=300ha -include=%asih8cDir% Timer.c >> error.txt
cc38h.exe -cpu=300ha -include=%asih8cDir% EV_Time.c >> error.txt
cc38h.exe -cpu=300ha -include=%asih8cDir% EV_File.c >> error.txt
cc38h.exe -cpu=300ha -include=%asih8cDir% EV_UpDown.c >> error.txt
cc38h.exe -cpu=300ha -include=%asih8cDir% EV_OpenClose.c >> error.txt
cc38h.exe -cpu=300ha -include=%asih8cDir% EV_Display.c >> error.txt
cc38h.exe -cpu=300ha -include=%asih8cDir% EV_Input.c >> error.txt
cc38h.exe -cpu=300ha -include=%asih8cDir% EV_Controller.c >> error.txt
cc38h.exe -cpu=300ha -include=%asih8cDir% EV_Puls.c >> error.txt
cc38h.exe -cpu=300ha -include=%asih8cDir% EV_Simulator.c >> error.txt
cc38h.exe -cpu=300ha -include=%asih8cDir% main.c >> error.txt
a38h.exe asmfile.src >> error.txt
l38h.exe -subcommand=linkfile.sub >> error.txt
c38h.exe c_thread.abs >> error.txt
del %CurrentDir%*.obj >> error.txt
del %CurrentDir%c_thread.abs >> error.txt
error.txt
exit
```

```
#!/bin/bash
cd `dirname $0`
find $PWD -name CentOSBuild.bash
rm ./CentOSError.txt
gcc -D"USE_CENTOS" -Wall -o CentOSExe main.c EV_Log.c EV_Simulator.c EV_Puls.c EV_Controller.c
EV_Input.c EV_Display.c EV_OpenClose.c EV_UpDown.c EV_File.c EV_Time.c Timer.c Panel.c -
I/usr/include/mysql -L/usr/lib/mysql -lmysqlclient &>>./CentOSError.txt
exit
```

```
#!/bin/bash
cd `dirname $0`
find $PWD -name CentOSStart.bash
./CentOSExe
exit
```



```
#!/bin/bash
cd `dirname $0`
find $PWD -name RaspberryPi3ModelBBuild.bash
rm ./RaspberryPi3ModelBError.txt
gcc -D"USE_RASPBIAN" -Wall -o RaspberryPi3ModelBExe main.c EV_Log.c EV_Simulator.c EV_Puls.c
EV_Controller.c EV_Input.c EV_Display.c EV_OpenClose.c EV_UpDown.c EV_File.c EV_Time.c Timer.c
Panel.c -I/usr/local/include -L/usr/local/lib -lwiringPi -lmariadbclient &>>./RaspberryPi3ModelBError.txt
exit
```

```
#!/bin/bash
cd `dirname $0`
find $PWD -name RaspberryPi3ModelBStart.bash
./RaspberryPi3ModelBExe
exit
```

実行環境状態ファイル

yynnyynn

N

出力ファイル

Start	Length	Name	Class
0001:00401000	000010C78H	_TEXT	CODE
0002:00412000	0000039A4H	_DATA	DATA
0003:004159A4	000000B70H	_BSS	BSS
0004:00000000	0000000A4H	_TLS	TLS

PAGE 1

PROGRAM NAME =

```
1          1 ; asmfile.src
2          2
3          3 .CPU 300HA
4 000000    4 .SECTION V,CODE,LOCATE=H'000000
5          5
6          6 ; C言語の関数 を参照
7          7 .IMPORT _main ; C言語の関数main を参照
8          8 .IMPORT _usb_int ; C言語の関数usb_int を参照
9          9 .IMPORT _InterruptITU0 ; C言語の関数InterruptITU0 を参照
10         10
11         11 ;-----
12         12 ; リセットベクタの転送先ラベルが_startになっています
13         13 ; リセットベクタ
14 000000 00000000    14 .DATA.L _start
15         15
16         16 ;-----
17         17 ; リセットベクタに続く1番から60番までの割り込みベクタ
18         18 ; について、使用しない割り込みベクタはラベルint_error
19         19 ; に転送されます
20         20 ; 割り込みベクタ
21         21 ; 1 Reserved
22 000004 00000000    22 _INT_Reserved1: .DATA.L int_error
23         23 ; 2 Reserved
24 000008 00000000    24 _INT_Reserved2: .DATA.L int_error
```

25	25	; 3 Reserved
26	00000C 00000000	26 _INT_Reserved3: .DATA.L int_error
27	27	; 4 Reserved
28	000010 00000000	28 _INT_Reserved4: .DATA.L int_error
29	29	; 5 Reserved
30	000014 00000000	30 _INT_Reserved5: .DATA.L int_error
31	31	; 6 Reserved
32	000018 00000000	32 _INT_Reserved6: .DATA.L int_error
33	33	; 7 NMI
34	00001C 00000000	34 _INT_NMI: .DATA.L int_error
35	35	; 8 TRAP
36	000020 00000000	36 _INT_TRAP1: .DATA.L int_error
37	37	; 9 TRAP
38	000024 00000000	38 _INT_TRAP2: .DATA.L int_error
39	39	; 10 TRAP
40	000028 00000000	40 _INT_TRAP3: .DATA.L int_error
41	41	; 11 TRAP
42	00002C 00000000	42 _INT_TRAP4: .DATA.L int_error
43	43	; 12 IRQ0
44	000030 00000000	44 IRQ0: .DATA.L int_error
45	45	; 13 IRQ1
46	000034 00000000	46 IRQ1: .DATA.L int_error
47	47	; 14 IRQ2
48	000038 00000000	48 IRQ2: .DATA.L int_error
49	49	; 15 IRQ3
50	00003C 00000000	50 IRQ3: .DATA.L int_error
51	51	; 16 IRQ4
52	000040 00000000	52 IRQ4: .DATA.L int_error
53	53	; 17 IRQ5

54 000044 00000000 54 IRQ5: .DATA.L usb_interrupt ; USB割り込み

55 55 ; 18 Reserved

56 000048 00000000 56 _INT_Reserved18: .DATA.L int_error

57 57 ; 19 Reserved

*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 *** 05/08/18 00:58:15

PAGE 2

PROGRAM NAME =

58 00004C 00000000 58 _INT_Reserved19: .DATA.L int_error

59 59 ; 20 WOVI

60 000050 00000000 60 _INT_WOVI: .DATA.L int_error

61 61 ; 21 CMI

62 000054 00000000 62 _INT_CMI: .DATA.L int_error

63 63 ; 22 Reserved

64 000058 00000000 64 _INT_Reserved22: .DATA.L int_error

65 65 ; 23 Reserved

66 00005C 00000000 66 _INT_Reserved23: .DATA.L int_error

67 67 ; 24 IMIA0

68 000060 00000000 68 _INT_IMIA0: .DATA.L int_error

69 69 ; 25 IMIB0

70 000064 00000000 70 _INT_IMIB0: .DATA.L int_error

71 71 ; タイマ0割り込みは、ラベル_ITU_OVI_0に転送されます

72 72 ; 26 OVIO

73 000068 00000000 73 _INT_OVIO: .DATA.L _ITU_OVI_0 ; タイマ0割り込み

74 74 ; 27 Reserved

75 00006C 00000000 75 _INT_Reserved27: .DATA.L int_error

76 76 ; 28 IMIA1

77 000070 00000000 77 _INT_IMIA1: .DATA.L int_error

78 78 ; 29 IMIB1

79	000074	00000000	79	_INT_IMIB1: .DATA.L int_error
80			80	;30 OVI1
81	000078	00000000	81	_INT_OVI1: .DATA.L int_error
82			82	;31 Reserved
83	00007C	00000000	83	_INT_Reserved31: .DATA.L int_error
84			84	;32 IMIA2
85	000080	00000000	85	_INT_IMIA2: .DATA.L int_error
86			86	;33 IMIB2
87	000084	00000000	87	_INT_IMIB2: .DATA.L int_error
88			88	;34 OVI2
89	000088	00000000	89	_INT_OVI2: .DATA.L int_error
90			90	;35 Reserved
91	00008C	00000000	91	_INT_Reserved35: .DATA.L int_error
92			92	;36 IMIA3
93	000090	00000000	93	_INT_IMIA3: .DATA.L int_error
94			94	;37 IMIB3
95	000094	00000000	95	_INT_IMIB3: .DATA.L int_error
96			96	;38 OVI3
97	000098	00000000	97	_INT_OVI3: .DATA.L int_error
98			98	;39 Reserved
99	00009C	00000000	99	_INT_Reserved39: .DATA.L int_error
100			100	;40 IMIA4
101	0000A0	00000000	101	_INT_IMIA4: .DATA.L int_error
102			102	;41 IMIB4
103	0000A4	00000000	103	_INT_IMIB4: .DATA.L int_error
104			104	;42 OVI4
105	0000A8	00000000	105	_INT_OVI4: .DATA.L int_error
106			106	;43 Reserved
107	0000AC	00000000	107	_INT_Reserved43: .DATA.L int_error

108 108 ;44 DEND0A
109 0000B0 00000000 109 _INT_DEND0A: .DATA.L int_error
110 110 ;45 DEND0B
111 0000B4 00000000 111 _INT_DEND0B: .DATA.L int_error
112 112 ;46 DEND1A
113 0000B8 00000000 113 _INT_DEND1A: .DATA.L int_error
114 114 ;47 DEND1B

*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 *** 05/08/18 00:58:15

PAGE 3

PROGRAM NAME =

115 0000BC 00000000 115 _INT_DEND1B: .DATA.L int_error
116 116 ;48 Reserved
117 0000C0 00000000 117 _INT_Reserved48: .DATA.L int_error
118 118 ;49 Reserved
119 0000C4 00000000 119 _INT_Reserved49: .DATA.L int_error
120 120 ;50 Reserved
121 0000C8 00000000 121 _INT_Reserved50: .DATA.L int_error
122 122 ;51 Reserved
123 0000CC 00000000 123 _INT_Reserved51: .DATA.L int_error
124 124 ;52 ERI0
125 0000D0 00000000 125 _INT_ERI0: .DATA.L int_error
126 126 ;53 RXI0
127 0000D4 00000000 127 _INT_RXI0: .DATA.L int_error
128 128 ;54 TXI0
129 0000D8 00000000 129 _INT_TXI0: .DATA.L int_error
130 130 ;55 TEI0
131 0000DC 00000000 131 _INT_TEI0: .DATA.L int_error


```

132          132 ;56 ERI1
133 0000E0 00000000      133  _INT_ERI1: .DATA.L int_error
134          134 ;57 RXI1
135 0000E4 00000000      135  _INT_RXI1: .DATA.L int_error
136          136 ;58 TXI1
137 0000E8 00000000      137  _INT_TXI1: .DATA.L int_error
138          138 ;59 TEI1
139 0000EC 00000000      139  _INT_TEI1: .DATA.L int_error
140          140 ;60 ADI
141 0000F0 00000000      141  _INT_ADI: .DATA.L int_error
142          142
143          143 ;-----
144 000000      144  .SECTION P,CODE,ALIGN=2
145          145 ;_start のラベルから処理を開始
146          146 ;リセットベクタの転送先
147 000000      147  _start:
148 000000 7A0700FFFF10    148  mov.l #H'0FFFF10,er7
149          149 ;初期化付きデータを使用する場合、RAMに転送する
150          150 ;c_thread.MAPのメモリアドレス使用状況を見る
151          151 ;メモリアドレスが重複するとコンパイルエラーになる
152          152 ;linkfile.subもD(99C0),C(9A00)等必要があれば合わせる
153 000006 7A00000099C0    153  mov.l #H'99C0, er0 ;転送元(99C0)
154 00000C 7A0100FFE000    154  mov.l #H'0FFE000, er1 ;転送先
155 000012 7A0200000000    155  mov.l #DATA_END, er2 ;転送終了
156 000018          156  init_loop:
157 000018 1F92          157  cmp.l er1, er2
158 00001A 58700008          158  beq init_end
159 00001E 6C0B          159  mov.b @er0+, r3l
160 000020 689B          160  mov.b r3l, @er1

```

```

161 000022 0B71      161  inc.l #1, er1
162 000024 40F2      162  bra init_loop
163 000026          163  init_end:
164          164  ;C言語の関数main を呼び出しています
165          165  ;C言語の関数main は、 void main(void); という形で、
166          166  ;main.c に記述があります
167 000026 5E000000      167  jsr @_main
168          168  ;割り込み未使用
169 00002A          169  int_error:
170          170  ;rte (returnと同じ意味) で終了
171 00002A 5670      171  rte

```

*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 *** 05/08/18 00:58:15

PAGE 4

PROGRAM NAME =

```

172          172
173          173  ;-----
174          174  ;USB割り込み からの転送先
175 00002C          175  usb_interrupt:
176          176  ;スタック 退避
177 00002C 01006DF0      177  push.l er0
178 000030 01006DF1      178  push.l er1
179 000034 01006DF2      179  push.l er2
180 000038 01006DF3      180  push.l er3
181 00003C 01006DF4      181  push.l er4
182 000040 01006DF5      182  push.l er5
183 000044 01006DF6      183  push.l er6
184          184  ;C言語の関数usb_int を呼び出しています
185 000048 5E000000      185  jsr @_usb_int

```

```

186          186 ;スタック戻
187 00004C 01006D76      187  pop.l er6
188 000050 01006D75      188  pop.l er5
189 000054 01006D74      189  pop.l er4
190 000058 01006D73      190  pop.l er3
191 00005C 01006D72      191  pop.l er2
192 000060 01006D71      192  pop.l er1
193 000064 01006D70      193  pop.l er0
194          194 ;終了
195 000068 5670          195  rte
196          196
197          197 ;-----
198          198 ;タイマ0割り込みからの転送先
199 00006A          199  _ITU_OVI_0:
200          200 ;スタック退避
201 00006A 01006DF0      201  push.l er0
202 00006E 01006DF1      202  push.l er1
203 000072 01006DF2      203  push.l er2
204 000076 01006DF3      204  push.l er3
205 00007A 01006DF4      205  push.l er4
206 00007E 01006DF5      206  push.l er5
207 000082 01006DF6      207  push.l er6
208          208 ;C言語の関数InterruptITU0 を呼び出しています
209          209 ;C言語の関数InterruptITU0 は void InterruptITU0(void);
210          210 ;という形で、Timer.h Timer.c に記述があります
211 000086 5E000000      211  jsr @_InterruptITU0
212          212 ;スタック戻
213 00008A 01006D76      213  pop.l er6
214 00008E 01006D75      214  pop.l er5

```

```

215 000092 01006D74      215  pop.l er4
216 000096 01006D73      216  pop.l er3
217 00009A 01006D72      217  pop.l er2
218 00009E 01006D71      218  pop.l er1
219 0000A2 01006D70      219  pop.l er0
220          220  ;終了
221 0000A6 5670          221  rte
222          222
223          223  ;-----
224          224  ;C言語から
225          225  ;_EnableInterrupt (割り込み許可)
226          226  ;_DisableInterrupt (割り込み禁止)
227          227  ;を呼び出せるようにしています
228          228  ;C言語の Panel.h に 外部参照プロトタイプ宣言 があります

```

*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 *** 05/08/18 00:58:15

PAGE 5

PROGRAM NAME =

```

229          229  ;extern void EnableInterrupt(void);
230          230  ;extern void DisableInterrupt(void);
231          231  ;C言語からの呼び出し名は、
232          232  ;EnableInterrupt();
233          233  ;DisableInterrupt();
234          234  ;です
235          235  ;割り込み許可、禁止ルーチン
236          236  .EXPORT _EnableInterrupt,_DisableInterrupt
237 0000A8          237  _EnableInterrupt:
238 0000A8 063F          238  andc.b #H'3f,ccr

```

```

239 0000AA 5470      239   rts
240 0000AC          240   _DisableInterrupt:
241 0000AC 04C0     241   orc.b #H'c0,ccr
242 0000AE 5470     242   rts
243                243
244                244   ;-----
245 000000          245   .SECTION D,DATA
246                246
247 000000          247   .SECTION B,DATA
248 000000 00000002  248   DATA_END: .RES.W 1
249                249
250                250   .END

*****TOTAL ERRORS    0
*****TOTAL WARNINGS  0

*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***   05/08/18 00:58:15

```

PAGE 6

*** CROSS REFERENCE LIST

NAME	SECTION	ATTR	VALUE	SEQUENCE
B	B	SCT	00000000	247*
D	D	SCT	00000000	245*
DATA_END	B		00000000	155 248*
IRQ0	V		00000030	44*
IRQ1	V		00000034	46*
IRQ2	V		00000038	48*
IRQ3	V		0000003C	50*
IRQ4	V		00000040	52*

IRQ5	V	00000044	54*		
P	P	SCT 00000000	144*		
V	V	SCT 00000000	4*		
_DisableInterrupt		P EXPT 000000AC	236	240*	
_EnableInterrupt		P EXPT 000000A8	236	237*	
_INT_ADI	V	000000F0	141*		
_INT_CMI	V	00000054	62*		
_INT_DEND0A	V	000000B0	109*		
_INT_DEND0B	V	000000B4	111*		
_INT_DEND1A	V	000000B8	113*		
_INT_DEND1B	V	000000BC	115*		
_INT_ERI0	V	000000D0	125*		
_INT_ERI1	V	000000E0	133*		
_INT_IMIA0	V	00000060	68*		
_INT_IMIA1	V	00000070	77*		
_INT_IMIA2	V	00000080	85*		
_INT_IMIA3	V	00000090	93*		
_INT_IMIA4	V	000000A0	101*		
_INT_IMIB0	V	00000064	70*		
_INT_IMIB1	V	00000074	79*		
_INT_IMIB2	V	00000084	87*		
_INT_IMIB3	V	00000094	95*		
_INT_IMIB4	V	000000A4	103*		
_INT_NMI	V	0000001C	34*		
_INT_OVI0	V	00000068	73*		
_INT_OVI1	V	00000078	81*		
_INT_OVI2	V	00000088	89*		
_INT_OVI3	V	00000098	97*		
_INT_OVI4	V	000000A8	105*		

_INT_RXI0	V	000000D4	127*
_INT_RXI1	V	000000E4	135*
_INT_Reserved1	V	00000004	22*
_INT_Reserved18	V	00000048	56*
_INT_Reserved19	V	0000004C	58*
_INT_Reserved2	V	00000008	24*
_INT_Reserved22	V	00000058	64*
_INT_Reserved23	V	0000005C	66*
_INT_Reserved27	V	0000006C	75*
_INT_Reserved3	V	0000000C	26*
_INT_Reserved31	V	0000007C	83*
_INT_Reserved35	V	0000008C	91*
_INT_Reserved39	V	0000009C	99*
_INT_Reserved4	V	00000010	28*
_INT_Reserved43	V	000000AC	107*
_INT_Reserved48	V	000000C0	117*
_INT_Reserved49	V	000000C4	119*

*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 *** 05/08/18 00:58:15

PAGE 7

*** CROSS REFERENCE LIST

NAME	SECTION	ATTR	VALUE	SEQUENCE
_INT_Reserved5	V		00000014	30*
_INT_Reserved50	V		000000C8	121*
_INT_Reserved51	V		000000CC	123*
_INT_Reserved6	V		00000018	32*
_INT_TEIO	V		000000DC	131*

```

_INT_TEI1          V      000000EC  139*
_INT_TRAP1        V      00000020  36*
_INT_TRAP2        V      00000024  38*
_INT_TRAP3        V      00000028  40*
_INT_TRAP4        V      0000002C  42*
_INT_TXI0         V      000000D8  129*
_INT_TXI1         V      000000E8  137*
_INT_WOVI         V      00000050  60*
_ITU_OVI_0        P      0000006A  73 199*
_InterruptITU0    IMPT 00000000   9 211
_main             IMPT 00000000   7 167
_start           P      00000000  14 147*
_usb_int         IMPT 00000000   8 185
init_end         P      00000026  158 163*
init_loop        P      00000018  156* 162
int_error        P      0000002A  22 24 26 28 30 32 34 36 38 40 42 44
                 46 48 50 52 56 58 60 62 64 66 68 70
                 75 77 79 81 83 85 87 89 91 93 95 97
                 99 101 103 105 107 109 111 113 115 117 119 121
                 123 125 127 129 131 133 135 137 139 141 169*
usb_interrupt    P      0000002C  54 175*

```

*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 *** 05/08/18 00:58:15

PAGE 8

*** SECTION DATA LIST

SECTION	ATTRIBUTE	SIZE	START
---------	-----------	------	-------

V	ABS-CODE	00000F4	000000
P	REL-CODE	00000B0	
D	REL-DATA	0000000	
B	REL-DATA	0000002	

S00E0000635F7468726561644D4F54C8
S1130000000022A0000022A0000022A67
S1130010000022A0000022A0000022A2D
S1130020000022A0000022A0000022A1D
S10B0030000022A0000022A6D
S1130038000022A0000022A0000022C03
S1130048000022A0000022A0000022AF5
S1130058000022A0000022A0000022AE5
S10B0068000026A0000022AF5
S1130070000022A0000022A0000022ACD
S1130080000022A0000022A0000022ABD
S1130090000022A0000022A0000022AAD
S10B00A0000022A0000022AFD
S11300A8000022A0000022A0000022A95
S11300B8000022A0000022A0000022A85
S11300C8000022A0000022A0000022A75
S10B00D8000022A0000022AC5
S11300E0000022A0000022A0000022A5D
S10700F0000022ADD
S11302007A0700FFFF107A0000099C07A0100FF0F
S1130210E0007A0200FFE0101F92587000086C0B98
S1130220689B0B7140F25E0002B0567001006DF0E6
S113023001006DF101006DF201006DF301006DF439
S113024001006DF501006DF65E004F5001006D7603
S113025001006D7501006D7401006D7301006D7215
S113026001006D7101006D70567001006DF00100A9
S11302706DF101006DF201006DF301006DF40100F9
S11302806DF501006DF65E00493001006D760100E9
S11302906D7501006D7401006D7301006D720100D5
S11302A06D7101006D705670063F547004C0547038
S11302B05E0060667A37000000A790B000119338B
S11302C00FF47A0600FFE1E419550B5579257FFFA
S11302D04DF85C000FBC5E004ACC5E004C4C0D3800
S11302E00D305C000F380D380DB05C000F300D3849
S11302F0790000025C000F260D38790000035C00D2
S11303000F1C5E004DF27FF472507FF570505E005B
S113031002A80D3228F117506DF07A000009A0000
S113032001006DF05E004B1C0B970B877A000000F9
S11303309A0F01006DF05E004D200B9718886EC870
S113034000036EC800026EC8000168C87A0000FF8F
S1130350E04E5E003EB65E00495C7A0000007D0C6
S11303605E0044147900001E5E00450A01006FF030
S113037000047900001F5E00450A0F8501006F70BD
S113038000045E0046E60FD05E0046E65E00495874
S10403905E0B
S11303910047320D0046F45E0049767A0100009A67
S11303A11B0D305E00499E7A00000007D05E0044B9
S11303B1145E00497619550D505C000EAE0D0D0DFE
S11303C15117F10AC16819D90117D1660147540DB3
S11303D1B80D505C000E460D5009B06DF07A000067
S11303E1009A2C01006DF00FE05E0059080B970B8A
S11303F18701006DF67A0000009A3101006DF05E0D
S1130401004B1C0B970B970FE05E0059AC09B00D25
S1130411010FE05E00578401006DF65E004D200B75
S11304219740080D380D505C000DF20DD017F50FF4
S1130431C10AD168980B557925000458D0FF785E1D
S1130441004B0C0D0047165E004AFC17D00D055EEC

S1130451004CD068ED0DB10FE05E0057845E00588B
S1130461A60D004738790100400FE05E0058480DA2
S1060471057A0105
S113047400009A3501006DF15E004D200B970100D9
S11304846DF65E004D200B9701006DF65E004B1C6C
S11304940B970D510FE05E0057840D28790000037C
S11304A45C000D760D20795000010D0219550B5592
S11304B4792527104DF85A0003B654705E00606620
S11304C47A0500FFE01219665E00497619EE400EC4
S11304D47A0100009A370D605E00499E0B5E6950F5
S11304E41D0E4DEC7A0100009A390D605E00499EA1
S11304F47A0100009A3D0D605E00499E19EE400E9C
S11305047A0100009A370D605E00499E0B5E6F50BE
S10B051400021D0E4DEA7A01FD
S113051C00009A3F0D605E00499E5E00604454707B
S113052C5E0060667A370000000A7A03000000015F
S113053C7A04000007D019550F8618886EF800034B
S113054C6EF800026EF8000168F86960792000010A
S113055C46365C00FF5E7A0000FFE01269010B5126
S113056C69815E0058CE17F07902000901D053203F
S113057C7918000A79000064528017F00F810FE09C
S113058C5E0044A65A000E88696079200002463644
S113059C5C00FF207A0000FFE01469010B516981B4
S11305AC5E0058CE17F07902000901D05320791858
S11305BC000A79000064528017F00F810FE05E008F
S11305CC44A65A000E8869607920000B586000B667
S11305DC01006F6000127A200000000146205E00CB
S11305EC49767A0100009A430D505E00499E7A01C8
S11305FC0000076C0FE05E0044EA5A000E8801000D
S109060C6F6000127A206A
S10A06120000000246265E12
S11306190049767A0100009A540D505E00499E7A8A
S11306290100009A5B0D505E00499E0FE05E004792
S1130639145A000E8801006F6000127A200000002E
S113064903461E5E0049767A0100009A660D505EE4
S113065900499E7A01000005DC0FE05E0044EA4090
S11306692401006F6000127A200000000446165E20
S11306790049767A0100009A770D505E00499E0F72
S1130689E05E0047145A000E8869607920000C580F
S11306996000A801006F6000127A20000000014683
S10A06A9205E0049767A018F
S11306B000009A880D505E00499E7A0100000D48A3
S11306C00FE05E0044EA5A000E8801006F600012DA
S11306D07A200000000246205E0049767A0100007D
S11306E09A990D505E00499E7A0100000D480FE073
S11306F05E0044EA5A000E8801006F6000127A20FF
S113070000000003461E5E0049767A0100009AAA3
S11307100D505E00499E7A0100000D480FE05E0017
S113072044EA401C5E0049767A0100009ABB0D50F2
S11007305E00499E0FE05E0047140FE05E7F
S113073D0046B25A000E8869607920000D5860009A
S113074DA801006F6000127A200000000146205EB0
S113075D0049767A0100009AC30D505E00499E7AD6
S113076D01000013EC0FE05E0044EA5A000E88010D
S113077D006F6000127A200000000246205E0049DF
S113078D767A0100009AD40D505E00499E7A0100DD
S113079D0013EC0FE05E0044EA5A000E8801006F6F

S11307AD6000127A2000000003461E5E0049767A2F
S11307BD0100009AE50D505E00499E7A0100001379
S11207CDEC0FE05E0044EA401C5E0049767A01BF
S11307DC00009AF60D505E00499E0FE05E00471430
S11307EC0FE05E0046B25A000E8869607920000E55
S11307FC586000E401006F6000127A2000000001D1
S113080C46205E0049767A0100009AFE0D505E0088
S113081C499E7A01000017700FE05E0044EA5A000B
S113082C0E8801006F6000127A200000000246401F
S113083C5E0049767A0100009B0F0D505E00499EC5
S113084C7A01000017700FE05E0044EA7A010000A1
S113085C9B160D505E00499E7900000B5E0047AC61
S10C086C0F867A01000005DC5E31
S11308750044EA5A000E8801006F6000127A2000D6
S1130885000003461E5E0049767A0100009B210D98
S1130895505E00499E7A01000017700FE05E004428
S11308A5EA403801006F6000127A20000000044618
S11308B52A7900000B5E0047720F8247060FA05E80
S11308C50046B27A0100009B320D505E00499E0F2F
S11308D5E05E0047140FE05E0046B25A000E8869D9
S11308E5607920001346440FB10FE05E0044A6195A
S11308F5DD0DD05C0009700DD117F10FF20A926876
S11309052ADA0117D2660247160DD0791000145E54
S11309150047EA7A01000003E80FE05E0044A60BF6
S11309255D792D00044DCA5A000E8869607920004F
S10809351446187A01CD
S113093A00009B390D505E00499E0FC10FE05E0017
S113094A44EA5A000E8869607920001546187A012C
S113095A00009B3B0D505E00499E0FC10FE05E00F5
S113096A44EA5A000E8869607920001646187A010B
S113097A00009B3D0D505E00499E0FC10FE05E00D3
S113098A44EA5A000E8869607920001746187A01EA
S113099A00009B3F0D505E00499E0FC10FE05E00B1
S11309AA44EA5A000E8869607920001E465E0100F7
S11309BA6F600012462401006F6000120B70010081
S11309CA6FE0001228D6E80E38D67A01000003E851
S10609DA0FE05ECA
S11309DD0044A65A000E8801006F6000127A2000B1
S11309ED0000015860049401006F6000121B700138
S11309FD006FE000127FD670007A01000003E80F4C
S1130A0DE05E0044A65A000E8869607920001F58E5
S1130A1D6003E601006F600012460C1A910FE05E51
S1130A2D0044EA5A000E8801006F6000127A20001C
S1130A3D00000146247A0100009A3D0D505E0049E5
S1130A4D9E7A0100009B410D505E00499E0FC10F20
S1130A5DE05E0044EA5A000E8801006F6000127ACE
S1130A6D2000000002463019DD0DD00B505E00450D
S1130A7D0A0DD217F210321032010078A06BA000CC
S1130A8DFFFE01A0B5D792D00024DDE0FB10FE05E15
S1130A9D0044EA5A000E0201006F6000127A200032
S10A0AAD00000346566B2015
S1130AB400FFE0127920000D4D1A790000016BA0AC
S1130AC400FFE01601006F6000120B7001006FE07D
S1130AD4001240246B2000FFE0147920000D4D1810
S1130AE4790000026BA000FFE01601006F600012A2
S1130AF40B7001006FE000120FB10FE05E0044A61B
S1130B045A000E0201006F6000127A2000000004F4

S1130B1446685E0049766B2000FFE01679200001E9
S1130B24460E7A0100009B520D505E00499E401808
S1130B346B2000FFE01679200002460C7A010000C6
S1130B449B630D505E00499E01006B2000FFE01A79
S10B0B545E0046B201006B20B4
S1130B5C00FFE01E5E0046B201006F6000120B70D6
S1130B6C01006FE000120FC10FE05E0044A65A00B3
S1130B7C0E0201006F6000127A2000000005461C73
S1130B8C5E0049767A0100009A1B0D505E00499E67
S1130B9C0FC10FE05E0044EA5A000E0201006F60C1
S1130BAC00127A2000000006461C5E0049767A018A
S1130BBC00009B740D505E00499E0FC10FE05E0058
S1130BCC44EA5A000E0201006F6000127A20000002
S1130BDC000746365E00497619DD0DD07910000BFF
S1130BEC5E00450A0DD217F210321032010078A0C4
S1050BFC6BA0E9
S1130BFE00FFE0220B5D792D00044DDC0FB10FE0F9
S1130C0E5E0044EA5A000E0201006F6000127A2061
S1130C1E0000000846245E00474C79200002460E71
S1130C2E01006F6000120B7001006FE000120FB134
S1130C3E0FE05E0044A65A000E0201006F60001220
S1130C4E7A2000000009460C0FC10FE05E0044EA53
S1130C5E5A000E0201006F6000127A200000000A93
S1130C6E461C5E0049767A0100009A1B0D505E0009
S1130C7E499E0FC10FE05E0044EA5A000E020100C6
S1130C8E6F6000127A200000000B461C5E0049764E
S1130C9E7A0100009B850D505E00499E0FC10FE047
S1040CAE5EE4
S1130CAF0044EA5A000E0201006F6000127A20001E
S1130CBF00000C4634790000135E00450A01006BF7
S1130CCFA000FFE0325E0046E6790000145E0045A7
S1130CDF0A01006BA000FFE0365E0046E60FB10F7E
S1130CEFE05E0044EA5A000E0201006F6000127AC0
S1130CFF200000000D46305E00474C79200003466C
S1130D0F1A01006B2000FFE0325E0046B201006F54
S1130D1F6000120B7001006FE000120FB10FE05E65
S1130D2F0044A65A000E0201006F6000127A2000E1
S1110D3F00000E460C0FC10FE05E0044EA5A9E
S1130D4D000E0201006F6000127A200000000F46B2
S1130D5D1C5E0049767A0100009A1B0D505E004916
S1130D6D9E0FC10FE05E0044EA5A000E0201006FB0
S1130D7D6000127A2000000010461A5E0049767A50
S1130D8D0100009B960D505E00499E0FC10FE05E62
S1130D9D0044EA406001006F6000127A20000000F9
S1130DAD114652790000295E00450A01006BA0002F
S1130DBDFFE03A7900002A5E00450A01006BA000AE
S1130DCDFFE03E7900002B5E00450A01006BA00099
S10B0DDDFFE0427900002C5EE7
S1130DE500450A01006BA000FFE0460FE05E0046E8
S1130DF5B27900001E5E0047725E0046B25A000ECD
S1130E0588696079200029461A7A01000000640F79
S1130E15E05E0044A60FE17A0000FFE0525E002683
S1130E250E406069607920002A461A7A01000000A5
S1130E35640FE05E0044A60FE17A0000FFE0905ED8
S1130E45001C9A403E69607920002B461A7A0100FE
S1130E5500006E0FE05E0044A60FE17A0000FFE19B
S1130E65885E0019DC401C69607920002C46140F4C

S1130E75C10FE05E0044A60FE17A0000FFE19C5E2E
S1130E850013A87A170000000A5E00604454705EE0
S1130E950060660F86790400097A0500FFE0126990
S1130EA560792000014626190069D05E0058CE17E7
S1130EB5F001D053407918000A7900001E528017BB
S1130EC5F00F810FE05E0044A65A00112269607994
S1130ED5200002462819006FD000025E0058CE1785
S1130EE5F001D053407918000A7900001E5280178B
S1130EF5F00F810FE05E0044A65A001122696C7958
S1130F052C00034D36792C00084E3069601B5017B1
S1130F15F010300A85190069D05E0058CE17F0012C
S1130F25D053407918000A790000C8528017F00F92
S1130F35810FE05E0044A65A0011227A040000499D
S1130F459E195569607920000B461A7A0100009BAA
S1130F55A70D505D407A01000005DC0FE05E0044FB
S1120F65EA5A00112269607920000C461A7A01BA
S1130F7400009BAF0D505D407A0100000BB80FE0F9
S1130F845E0044EA5A00112269607920000D462468
S1130F947A0100009A3D0D505D407A0100009BB92F
S1130FA40D505D407A01000011940FE05E0044EAA5
S1130FB45A00112269607920000E461A7A01000052
S1130FC49BC10D505D407A01000017700FE05E0075
S1130FD444EA5A0011227A03000007D06960792099
S1130FE4001446245E0049767A0100009BCB0D5021
S1130FF45D407A0100009A3D0D505D400FB10FE052
S11310045E0044EA5A0011226960792000154624DF
S10410145E7A
S11310150049767A0100009BDC0D505D407A0100A2
S1131025009A3D0D505D400FB10FE05E0044EA5A52
S113103500112269607920001646245E0049767AFC
S11310450100009BED0D505D407A0100009A3D0DB6
S1131055505D400FB10FE05E0044EA5A001122696A
S1131065607920001746245E0049767A0100009BCB
S1131075FE0D505D407A0100009A3D0D505D400F15
S1131085B10FE05E0044EA5A00112269607920003D
S10D10951E460AF8FF38D438D65A75
S113109F00112269607920002946187A01000000A7
S11310AF640FE05E0044A67A0000FFE0525E002466
S11310BFE2406069607920002A461A7A0100000035
S11310CF640FE05E0044A60FE17A0000FFE0905E3C
S11310DF001AFA403E69607920002B461A7A010004
S11310EF00006E0FE05E0044A60FE17A0000FFE1FF
S11310FF885E00199E401C69607920002C46140FEE
S113110FB10FE05E0044A60FE17A0000FFE19C5EA1
S113111F0012BE5E00604454705E0060667A040085
S113112F00499E19660F8569507920000B46105EA2
S108113F0049767A016E
S113114400009C0F0D605D40404469507920000C01
S1131154460C7A0100009C1A0D605D4040306950D2
S11311647920000D460C7A0100009C240D605D403B
S1131174401C69507920000E46147A0100009A3D00
S11311840D605D407A0100009C2E0D605D40695046
S113119479200014461A5E0049767A0100009C3FC8
S11311A40D605D407A0100009A3D0D605D4040642E
S11311B4695079200015461A5E0049767A010000C9
S11311C49C500D605D407A0100009A3D0D605D40C6
S11311D44042695079200016461A5E0049767A0126

S11311E400009C610D605D407A0100009A3D0D6032
S11011F45D40402069557925001746185EBF
S11312010049767A0100009C730D605D407A01000C
S1131211009A3D0D605D405E00604454706DF66D53
S1131221F50D067909000128D617500D950CE91A19
S1131231094B04101540F866050D8846121A0E4B2A
S113124104101940F80D9029D6148939D640141A7F
S11312510E4B04101940F8795900FF0D9029D61649
S11312618939D60D506D756D7654706DF60D06285E
S1131271D31750790100011A0E4B04101140F8667F
S113128110470419114004790100010D106D7654C2
S113129170F80638ECF8FF38C038C1188838D8F828
S11312A1FF38C8188838DBF8FF38C9F8DF38D0F8B9
S11012B1FF38CDF8F038D1F8FF38D4547071
S1139A00435055204D4F444520253032580A000C11
S1139A1052656164792133303532004E4558542004
S1139A202020202020202020202000737725754F
S1139A300025730A000C0020003C313E000A003C64
S1139A40323E003C313E31737420202020202000
S1139A50202020003C313E326E64003C313E537482
S1139A606F70202020003C313E3372642020202080
S1139A70202020202020003C313E53746F70202092
S1139A8020202020202020003C323E3173742020EF
S1139A9020202020202020003C323E326E6420F3
S1139AA02020202020202020003C323E337264DE
S1139AB0202020202020202020003C323E5374F0
S1139AC06F70003C333E3173742020202020200F
S1139AD0202020003C333E326E64202020202020B2
S1139AE020202020003C333E33726420202020209D
S1099AF0202020202000CD
S1139AF63C333E53746F70003C343E317374202004
S1139B062020202020202020003C343E326E64009A
S1139B163C313E53746172742020003C343E3372F0
S1139B2664202020202020202020003C343E5387
S1139B36746F00300031003200330054687265617F
S1139B466420526561647920474F2100474F414C99
S1139B56213C313E574F4E202020202000474F41C5
S1139B664C213C323E574F4E202020202000436F8D
S1139B76756E745570202020202020202000544C
S1139B866F67676C65202020202020202020007E
S1139B9648656C6C6F204556202020202020200D
S1139BA6003C313E496E6974003C323E496E69742D
S1139BB62020003C333E496E6974003C343E496EB6
S1139BC669742020003C303E496E697420202020B1
S1139BD62020202020003C313E496E69742020203D
S1099BE6202020202020B6
S1139BEC003C323E496E69742020202020202026
S1139BFC20003C333E496E697420202020202015
S1139C0C2020003C313E44657374726F79003C3202
S1139C1C3E44657374726F003C333E4465737472D7
S1139C2C6F003C343E44657374726F792020209E
S1139C3C2020003C303E44657374726F79202020E1
S1139C4C202020003C313E44657374726F792020D0
S1139C5C20202020003C323E44657374726F7920BF
S1139C6C202020202020003C333E44657374726F07
S10C9C7C79202020202020200083
S11312BE5E0060660F860F957A00000000200AE03C

S11312CE0FD15E0042AA7A00000000200AE05E0001
S11312DE42EE01006FE600027A00000000060AE00B
S11312EE01006FE000087A000000000C0AE0010024
S11312FE6FE0000E7A00000000120AE001006FE0BA
S113130E001C18886EE8001A7A0500003F0A7A005E
S113131E000000380AE05D507A000000003C0AE04D
S113132E5D507A00000000400AE05D507A00000034
S113133E00440AE05D5001006F60000201006DF091
S113134E7A00000000400AE07A0100009C865E00ED
S113135E3F9E0B9779200001473A790000096DF003
S113136E01006F60001C01006DF07A000000004464
S113137E0AE07A0100009C915E003FF80B970B8701
S113138E79200001470E7A01000000120AE1686815
S113139E5E002E045E00604454705E0060660F862D
S10913AE0F9501006F60C2
S11313B4000E01006DF07A000000003C0AE07A019F
S11313C400009C9C5E003F9E0B976E68000CA87106
S11313D4460E5E0049760FD05E0046B25A00177877
S11313E401006F60000201006DF07A00000000400C
S11313F40AE07A0100009C865E003F9E0B97790009
S113140400096DF001006F60001C01006DF07A00AB
S10B1414000000440AE07A0124
S113141C00009C915E003FF80B970B8701006F60F7
S113142C000801006DF07A00000000380AE07A0130
S113143C00009CA95E003F9E0B976E680006A87384
S113144C461CF84E6DF07A000000003C0AE07A016D
S113145C00009C9C5E003F140B875A00176A686857
S113146CA87546366E680012A879587002D06E685B
S113147C0013A879460AF86E6EE800135A00174A4F
S113148C6E680014A86E470E6E680015A86E4606AB
S113149CF8796EE800155A00174A6868A855462E65
S11314AC6E680012A879587002946E680013A879BC
S11314BC5870028A6E680014A86E4608F8796EE8B4
S11314CC001440066E680015A86E5A00174A686827
S11314DCA86A46466E680012A879460AF86E6EE84A
S11314EC00125A00174A6E680013A879460AF86E60
S11314FC6EE800135A00174A6E680014A86E46086B
S109150CF8796EE80014FB
S1131512400E6E680015A86E4606F8796EE800154F
S11315225A00174A6868A86446366E680015A87997
S1131532587002146E680014A879460AF86E6EE8B1
S113154200145A00174A6E680013A86E470E6E689D
S11015520012A86E4606F8796EE800125AE2
S113155F00174A6868A844462E6E680015A8795884
S113156F7001D86E680014A879587001CE6E6800A8
S113157F13A86E4608F8796EE8001340066E6800EC
S113158F12A86E5A00174A6868A86B46466E680021
S113159F15A879460AF86E6EE800155A00174A6EB9
S11315AF680014A879460AF86E6EE800145A0017FB
S11315BF4A6E680013A86E4608F8796EE800134068
S11315CF0E6E680012A86E4606F8796EE800125A7E
S11315DF00174A6868A86F46366E680016A87958D0
S11315EF7001586E680017A879460AF86E6EE80006
S11315FF175A00174A6E680018A86E470E6E6800D8
S113160F19A86E4606F8796EE800195A00174A684A
S113161F68A84F462E6E680016A8795870011C6E85
S113162F680017A879587001126E680018A86E46E3

S113163F08F8796EE8001840066E680019A86E5A0C
S109164F00174A6868A8B9
S11316556846466E680016A879460AF86E6EE80075
S1131665165A00174A6E680017A879460AF86E6E6F
S1131675E800175A00174A6E680018A86E4608F85E
S1131685796EE80018400E6E680019A86E4606F8D4
S1131695796EE800195A00174A6868A86346346EDC
S11316A5680019A8795870009C6E680018A87946D7
S11316B50AF86E6EE800185A00174A6E680017A8F4
S11316C56E470E6E680016A86E4606F8796EE8003A
S11316D51640726868A84346286E680019A87947BA
S11316E5646E680018A879475C6E680017A86E4693
S11316F508F8796EE8001740066E680016A86E4074
S1131705446868A874463E6E680019A8794608F8C7
S11317156E6EE80019402E6E680018A8794608F821
S11317256E6EE80018401E6E680017A86E4608F82E
S1131735796EE80017400E6E680016A86E4606F827
S1081745796EE80016B7
S113174A7A00000000120AE001006DF07A0000003E
S10E175A00440AE07A0100009C915E4D
S1131765003F660B977A01000000120AE168685E84
S10C1775002E045E006044547070
S1139C864D6F746F722E74787400004C696D69742D
S1139C962E7478740000436F6D6D616E642E747854
S1129CA67400005361666574792E74787400003E
S113177E0D0046067FD67200400A7920000146040A
S113178E7FD670000D8846067FD67210400A7928E0
S113179E000146047FD670100D1146067FD67220C7
S11317AE400A7921000146047FD670200D99460622
S11317BE7FD67230400A7929000146047FD67030F5
S11317CE7FD6724054707FD672007FD672107FD64A
S11317DE72207FD672307FD6704054705E00606682
S11317EE0F850D1C0D94790E000119660DC017F0AF
S11317FE01006F5100121F81461E0D690D610D68A8
S113180E0D405C00FF6A01006F5000120B70010067
S113181E6FD000120D6E402E0B5C0DC017F0010041
S113182E6F5100121F81461E0D69790100010D686B
S113183E0D405C00FF3A01006F5000120B70010067
S113184E6FD000120D6E0DE05E00604454705E00AA
S113185E60660F850D1C0D94790E000119660DC07F
S109186E17F001006F51A9
S113187400121F81461E0D690D610D480D605C0049
S1131884FEF801006F5000120B7001006FD00012BC
S11318940D6E402E0B5C0DC017F001006F5100124A
S11318A41F81461E790900010D610D480D605C001E
S11318B4FEC801006F5000120B7001006FD00012BC
S11318C40D6E0DE05E00604454705E0060660F852B
S11318D40F966F79001819110FE05C00FF080D00D3
S11318E4587000B06F79001A790100020FE05C00B0
S11318F4FEF40D005870009C6F79001C79010004FC
S11319040FE05C00FEE00D00587000886F79001E44
S1131914790100060FE05C00FECC0D0047766F7979
S11319240020790100080FE05C00FF2C0D004764E0
S11319346F7900227901000A0FE05C00FF1A0D00A1
S113194447526F7900247901000C0FE05C00FF0813
S11319540D0047406F7900267901000E0FE05C000B
S1091964FEF60D00472E04

S113196A5C00FE661A8001006FE000126E580006E2
S113197AA871461A7FD672007FD672107FD672205C
S113198A7FD672307FD672400FE05E0046B25E00A9
S113199A6044547001006DF60F8601006FE6000281
S11319AA7A00000000060AE001006FE000087A00EE
S11319BA000000100AE05E003F0A7A000000000CF3
S11319CA0AE05E003F0AF8FF38D438D601006D7684
S11319DA54705E0060660F850F9401006F40001219
S11319EA46307A000000000C0AD001006F5100084B
S11319FA5E0040C001006F50000201006DF07A00E2
S1131A0A000000100AD07A0100009CB65E003F9ED7
S1131A1A0B9719EE790600016858A843587000AC71
S1131A2AA8444758A84F4770A8554738A863477C26
S1131A3AA8644740A868474CA86A471CA86B472C68
S1131A4AA86F474AA873470CA8744756A875470E48
S1091A5A5A001AF46DFEB0
S1131A6040026DF66DFE400A6DFE6DF640046DF6A4
S1131A706DF66DFE401E6DFE6DFE6DF640166DF645
S1131A806DFE6DF6400E6DFE6DF66DF640066DF65D
S1131A906DF66DF66DFE40266DFE6DFE6DFE6DF608
S1131AA0401C6DF66DFE6DFE6DF640126DFE6DF61B
S1131AB06DFE6DF640086DF66DF66DFE6DF66DF616
S1131AC06DF66DF66DFE0FC10FD05C00FE007A1748
S1131AD000000010401E6DFE6DFE6DF66DF66DF696
S1131AE06DF66DF66DFE0FC10FD05C00FDE07A1749
S10D1AF0000000105E006044547013
S10E9CB64D6F746F722E747874000001
S1131AFA5E0060660F860F9501006FE60024010001
S1131B0A6F6000245E00366E7A00000000280AE047
S1131B1A5E0036E27A00000000300AE05E003942D5
S1131B2A7A00000000380AE05E003BA27A00000057
S1131B3A004C0AE001006FE000705E002E067A0096
S1131B4A000000740AE05E002E7A7A000000007C2E
S1131B5A0AE05E0030DA7A00000000840AE05E00E0
S1131B6A333A7A00000000A20AE00FD15E0042AACB
S1131B7A7A0500003F0A7A00000000E00AE05D509F
S1131B8A7A00000000E40AE05D507A00000000E8F1
S1131B9A0AE05D507A00000000EC0AE05D507A002A
S1131BAA000000F00AE05D507A00000000F40AE049
S1131BBA5D507A00000000BA0AE001006FE000BC41
S1131BCA7A00000000C00AE001006FE000CA7A0050
S1131BDA000000CE0AE001006FE000D07A000000A6
S1071BEA00D40AE036
S1131BEE01006FE000D67A00000000DA0AE001007F
S1131BFE6FE000DC7A00000000F40AE0F9735E0087
S1131C0E41E07A00000000380AE001006F61002411
S1131C1E5E003BBC7A00000000840AE001006F61A5
S1131C2E00705E003362790000096DF001006F6091
S1131C3E00CA01006DF07A00000000E40AE07A01A8
S1131C4E00009CC25E003FF80B970B877A000000E2
S1131C5E00E80AE0F94E5E00410C792000014726A8
S1131C6E7A00000000EC0AE0F9635E00407C792004
S1131C7E000147127A00000000F00AE0F94E5E0000
S1131C8E419C792000015E00604454705E006066E2
S1131C9E790D0001F463FC4E19550F860F93010065
S1131CAE6F6000BC01006DF07A00000000E00AE0F6
S1051CBE7A01A6

S1131CC000009CCD5E003F9E0B9779200001587069
S1131CD0079E790000096DF001006F6000CA0100E2
S1131CE06DF07A00000000E40AE07A0100009CC273
S1131CF05E003FF80B970B877A00000000E80AE0CC
S1131D0001006F6100D05E0040C07920000158706F
S1131D10075E7A00000000EC0AE001006F6100D664
S1131D205E00403079200001587007447A000000BB
S1131D3000F00AE001006F6100DC5E004150792091
S1131D4000015870072A6E6800CEA8485870051025
S1131D50A8594762A86358700458A86458700222AF
S1131D60A868587005C4A86F58700398A8714718DD
S1131D70A8725870067EA8734726A875475EA8798F
S1131D80587001D85A0023F47A00000000F40AE0E6
S1131D90F9735E0041E00FB05E0046B25A0023F4CF
S1131DA07A00000000A20AE00D515E0043725A005F
S1091DB023F45A0023F4A2
S1131DB601006F60002401006F00000C69007920A8
S1131DC60001461201006F60007001006F00000CF5
S1131DD669015870061801006F60002401006F0046
S1131DE60014690079200001464E01006F600070FF
S1131DF601006F0000146900792000015860034A4E
S1131E067A00000000EC0AE00C495E00407C7A0090
S1131E16000000A20AE00DD15E0043727A000000C2
S1131E2600E80AE00CC95E00410C5E0049767A01BF
S1131E3600009CD95A0023A201006F6000700100C4
S1131E466F00000C69007920000146767A000000D5
S1131E5600A20AE00D515E0043727A00000000A260
S1131E660AE05E0042EE7A00000000F00AE00CC9C8
S1131E765E00419C7A00000000380AE07A370000D1
S1131E8600140FF17A02000000145E00600E7A005F
S1131E96000000280AE07A37000000080FF17A02F2
S1131EA6000000085E00600E01006F6100BC0100C7
S1131EB66F6000245E003E267A170000001C5A005D
S1131EC623F401006F60007001006F00000C6901CC
S1131ED646667A00000000A20AE00D515E004372D6
S1131EE67A00000000A20AE05E0042EE7A000000DB
S1131EF600840AE07A370000001E0FF17A02000020
S1131F06001E5E00600E7A000000007C0AE07A374D
S1131F16000000080FF17A02000000085E00600E60
S1091F2601006F6100BC25
S1131F2C01006F6000705E0036267A1700000026F1
S1131F3C401A7A00000000A20AE00D515E004372C1
S1131F4C7A00000000A20AE05E0042EE5A0023F47D
S1131F5C01006F60002401006F00001469007920F8
S1131F6C0001461201006F60007001006F00000C4D
S1131F7C69015870047201006F60002401006F0046
S1131F8C000C690079200001465001006F6000705D
S1131F9C01006F00001469007920000146387A00B3
S1131FAC000000EC0AE00C495E00407C7A00000063
S10A1FBC00A20AE00DD15E53
S1131FC30043727A00000000E80AE00CC95E004196
S1131FD30C5E0049767A0100009CD95A0023A25A69
S1131FE300215001006F60007001006F00000C6955
S1131FF3007920000146767A00000000A20AE00D72
S1132003515E0043727A00000000A20AE05E0042C0
S1132013EE7A00000000F00AE00CC95E00419C7AEE
S113202300000000380AE07A37000000140FF17A49

S113203302000000145E00600E7A00000000300A04
S1132043E07A37000000080FF17A02000000085E0F
S113205300600E01006F6100BC01006F6000245E2D
S1132063003E6E7A170000001C5A0023F401006F30
S113207360007001006F00000C690146667A00007E
S11320830000A20AE00D515E0043727A00000000D3
S1132093A20AE05E0042EE7A00000000840AE07ABE
S11320A3370000001E0FF17A020000001E5E00607D
S10620B30E7A009F
S11320B60000007C0AE07A37000000080FF17A027C
S11320C6000000085E00600E01006F6100BC0100A5
S11320D66F6000705E0036267A1700000026401AED
S10E20E67A00000000A20AE00D515E2A
S11320F10043727A00000000A20AE05E0042EE5A39
S11321010023F401006F60007001006F0000146987
S1132111007920000146387A00000000EC0AE00C47
S1132121495E00407C7A00000000A20AE00DD15E06
S11321310043727A00000000E80AE00CC95E004126
S11321410C5E0049767A0100009CD95A0023A27AD9
S113215100000000A20AE00D515E0043727A000004
S11321610000A20AE05E0042EE7A00000000840A49
S1132171E07A370000001E0FF17A020000001E5EB4
S113218100600E7A00000000740AE07A3700000054
S1132191080FF17A02000000085E00600E01006F73
S11321A16100BC01006F6000705E0035DE5A0023E0
S11321B1207A00000000A20AE00D515E0043727A0A
S11321C100000000A20AE05E0042EE01006F600021
S11321D17001006F000000C69007920000146387A14
S10821E100000000F006
S11321E60AE00CC95E00419C7A00000000E80AE0A0
S10621F60CC95EB0
S11321F900410C7A00000000EC0AE00C495E004043
S11322097C5E0049767A0100009CD95A0023A27AA0
S113221900000000840AE07A370000001E0FF17AFB
S1132229020000001E5E00600E7A000000007C0AB6
S1132239E07A37000000080FF17A02000000085E17
S113224900600E01006F6100BC01006F6000705EE9
S11322590036265A00232001006F60002401006F15
S11322690000146900792000015860017E7A00009A
S11322790000A20AE00D515E0043727A00000000DB
S1132289A20AE05E0042EE01006F60007001006F78
S1132299000000C69007920000146387A000000002B
S11322A9E80AE00CC95E00410C7A00000000F00A5C
S11322B9E00CC95E00419C7A00000000EC0AE00CC6
S11322C9495E00407C5E0049767A0100009CD95A38
S11322D90023A27A00000000840AE07A3700000094
S10822E91E0FF17A0253
S11322EE0000001E5E00600E7A000000007C0AE013
S11222FE7A37000000080FF17A02000000085E33
S113230D00600E01006F6100BC01006F6000705E24
S113231D0036267A17000000265A0023F401006FB9
S113232D60002401006F000000C6900792000015842
S113233D6000B47A00000000A20AE00D515E004374
S113234D727A00000000A20AE05E0042EE01006F07
S113235D60007001006F000000C69007920000146D8
S113236D3C7A00000000E80AE00CC95E00410C7ADB
S113237D00000000F00AE00CC95E00419C7A0000E9

S113238D0000EC0AE00C495E00407C5E0049767A61
S113239D0100009CD90D505E00499E404A7A000011
S11323AD0000840AE07A370000001E0FF17A020064
S11323BD00001E5E00600E7A000000007C0AE07AC9
S11323CD3700000080FF17A02000000085E00607C
S11323DD0E01006F6100BC01006F6000705E00367E
S11323ED267A17000000267A00000000A20AE05E9C
S10823FD00430C7920F0
S1132402000A4D6A01006F60007001006F00001442
S113241269007920000146567A00000000A20AE012
S11324225E0043A67920000146447A00000000A220
S11324320AE00D515E0043727A00000000A20AE036
S11324425E0042EE7A00000000F00AE0F96F5E00DF
S1102452419C7A00000000EC0AE00CC95E1A
S113245F00407C7A00000000E80AE00C495E00416E
S10A246F0C5E006044547091
S1139CC24C696D69742E747874000053616665740F
S1139CD2792E747874000048656C6C6F20455620A9
S10B9CE2202020202020200097
S11324765E0060661B970FF40C8D18886EC8000308
S11324866EC800026EC8000168C819660D605E005A
S1132496126C0D0E0D6117F10AC16819D90117D116
S11324A6660147260D600C004620A800470EA801CA
S11324B6470EA802470EA803470E400EFD75400AB5
S11324C6FD644006FD634002FD710B567926000448
S11324D64DBA0CD80B975E00604454705E0060667C
S11324E60F8618886EE800067A00000000120AE0DC
S11324F601006FE000147A00000000180AE00100F2
S11325066FE0002218886EE8002001006FE60002E3
S11325167A00000000070AE001006FE000087A0075
S11325260000000C0AE001006FE0000E7A050000CF
S11325363F0A7A00000000260AE05D507A00000098
S1132546002A0AE05D507A000000002E0AE05D5082
S11325567A00000000320AE05D507A000000003A7B
S10725660AE05D50D7
S113256A01006F60000201006DF07A00000000268E
S113257A0AE07A0100009CEA5E003F9E0B977920ED
S113258A0001477A7A000000002E0AE001006F6119
S113259A00085E0040307920000147627A0000009B
S11325AA00320AE001006F61000E5E00415079209B
S11325BA0001474A01006F60001401006DF07A00C0
S11325CA0000003A0AE07A0100009CF65E003F9E92
S11325DA0B97792000014726790000096DF0010065
S11325EA6F60002201006DF07A00000000360AE0F5
S11325FA7A0100009D015E003FF80B970B875E008E
S113260A604454705E006066FC74F568FD4E0F8684
S113261A0F937A01000000060AE17A000000002AFB
S113262A0AE05E0040C079200001587007766E68A0
S113263A00065C00FE366EE8000601006F600002C9
S113264A01006DF07A00000000260AE07A0100001A
S109265A9CEA5E003F9EB6
S11326600B9779200001587007447A000000002E70
S11326700AE001006F6100085E004030792000012C
S11326805870072A7A00000000320AE001006F61E7
S1062690000E5ED8
S1132693004150792000015870071001006F60005A
S11326A31401006DF07A000000003A0AE07A010099

S11326B3009CF65E003F9E0B97792000015870063D
S11326C3EA790000096DF001006F60002201006DDB
S11326D3F07A00000000360AE07A0100009D015EF3
S11326E3003FF80B970B876E680006A871587000BC
S11326F38EA872474CA873586000ACF8736DF07AD8
S113270300000000260AE07A0100009D0C5E003FF2
S1132713140B877A000000003A0AE0F9735E004164
S1132723E0F84E6EE800067A000000002A0AE0F99A
S11327334E5E00410C7A000000002E0AE0F94E4081
S1132743360C586DF07A00000000260AE07A010087
S1132753009D0C5E003F140B87F84E6EE800067A6B
S1132763000000002A0AE0F94E5E00410C7A0000E3
S113277300002E0AE0F9635E00407C5A002DAE7A16
S1082783000000002A24
S11327880AE06E6900065E00410C7A0100009D179D
S10E2798790000015E00499E0FB05E57
S11327A30046B25A002DAE6E680007A86358600452
S11327B3BEF46F19DD6E680006A8485870041CA8A0
S11327C359587002F2A86358700080A864587001C6
S11327D394A86858700376A86F4710A875587000BB
S11327E3C8A8795870023A5A002DAE6E68001CA827
S11327F379470E7A00000000320AE00C495E00417B
S11328039C6868A868586003B0F85968E86DF07A63
S113281300000000260AE07A0100009D0C5E003FE1
S1132823140B876E680012A873461C6868A8594680
S1132833166E68001FA86E460E7A000000003A0A5F
S1132843E00C595E0041E05A002BBC6E68001CA8E3
S113285379470E7A00000000320AE00C495E00411A
S11328639C6868A868463EF85968E86DF07A0000EA
S11328730000260AE07A0100009D0C5E003F140B62
S1132883876E680012A873461C6868A85946166EBB
S109289368001CA86E465C
S11328990E7A000000003A0AE00CC95E0041E05AD2
S11328A9002BBC6E68001CA879470E7A0000000053
S11328B9320AE00C495E00419C6868A868586000C8
S11328C99AF85968E86DF07A00000000260AE07A60
S11328D90100009D0C5E003F140B876E680012A86F
S11328E97346266868A85946206E680018A8794671
S11328F9186E68001CA86E46107A000000003A0A98
S1072909E00CC95EB4
S113290D0041E040526E680012A873461E6868A825
S113291D5946186E68001BA86E46107A0000000019
S113292D3A0AE0F96A5E0041E0402C6E680012A895
S113293D7346246868A859461E6E68001BA879461D
S113294D166E68001FA86E460E7A000000003A0A44
S113295DE00C595E0041E05A002BBC6E68001CA8C8
S113296D79470E7A00000000320AE00C495E0041FF
S113297D9C6868A8685860009AF85968E86DF07A01
S113298D00000000260AE07A0100009D0C5E003F66
S113299D140B876E680012A87346266868A85946FB
S11329AD206E68001BA87946186E68001CA86E4639
S11329BD107A000000003A0AE00CC95E0041E040C5
S11329CD526E680012A873461E6868A85946186EA1
S11329DD680018A86E46107A000000003A0AE0F964
S11329ED6B5E0041E0402C6E680012A873462468AC
S10829FD68A859461E05
S1132A026E680018A87946166E68001FA86E460EF7

S1132A127A000000003A0AE00C595E0041E05A00D5
S1132A222BBC7A03000000180AE36838A87947121E
S1132A326E380004A879470A7A0100009D1C5A00E7
S1132A422BEC6868A868466CF85968E86DF07A0060
S1132A52000000260AE07A0100009D0C5E003F148C
S1132A620B876E680012A873461E6868A859461839
S1132A726E680018A86E46107A000000003A0AE059
S1132A82F96B5E0041E0402C6E680012A873462485
S1132A926868A859461E6E680018A87946166E68BB
S1132AA2001FA86E460E7A000000003A0AE00C5995
S1082AB25E0041E05A43
S1132AB7002BBC7A03000000180AE36E380003A852
S1132AC77947126E380004A879470A7A0100009DF6
S1132AD7365A002BEC6868A868466CF85968E86DA5
S1132AE7F07A00000000260AE07A0100009D0C5EE0
S1132AF7003F140B876E680012A873461E6868A808
S1132B075946186E68001BA86E46107A000000002D
S1132B173A0AE0F96A5E0041E0402C6E680012A8A9
S1132B277346246868A859461E6E68001BA8794631
S1132B37166E68001FA86E460E7A000000003A0A58
S1132B47E00C595E0041E0406C6E680018A87947B5
S1132B570A7A0100009D1C5A002BEC6E68001CA822
S1132B6779470E7A00000000320AE00C495E004103
S1132B779C6868A868463EF85968E86DF07A0000D3
S1132B870000260AE07A0100009D0C5E003F140B4B
S1132B97876E680012A873461C6868A85946166EA4
S1092BA768001CA86E4645
S1132BAD0E7A000000003A0AE00CC95E0041E07A9B
S1132BB000000002A0AE06E6900065E00410C7AEF
S1132BCD000000002E0AE00CD95E00407C5A002D57
S10E2BD0AE6E68001BA87947107A0158
S1132BE800009D360DD05E00499E5A002DAE6E68DA
S1132BF8001CA879470E7A00000000320AE00C494D
S1132C085E00419C6868A868463EF85968E86DF01C
S1132C187A00000000260AE07A0100009D0C5E009D
S1132C283F140B876E680012A873461C6868A8597E
S1132C3846166E68001CA86E460E7A000000003A1D
S1132C480AE00CC95E0041E07A000000002A0AE0AD
S1132C586E6900065E00410C7A000000002E0AE04F
S1132C680CD95E00407C5A002DAE6E68000CA86F2C
S1132C78586001327A03000000180AE36838A8791B
S1132C885860008C6E380007A879587000826E6807
S1132C980006A864470CA86F4708A87947045A0098
S1132CA82DAE6868A8684646F85968E86DF07A005A
S1132CB8000000260AE07A0100009D0C5E003F1424
S1132CC80B876E680012A87346246868A859461EC5
S1092CD86E680018A879E4
S1132CDE46166E68001FA86E460E7A000000003A74
S1132CEE0AE00C595E0041E07A000000002A0AE077
S1132CFE6E6900065E00410C7A00000000320AE0A5
S10A2D0E0CD95E00419C5A41
S1132D15002DAE7A03000000180AE36E380003A8FD
S1132D2579586000846E380007A879477C6E68007F
S1132D3506A859470AA86F4706A8754702406A6857
S1132D4568A8684646F85968E86DF07A00000000FF
S1132D55260AE07A0100009D0C5E003F140B876E86
S1132D65680012A87346246868A859461E6E680051

S1132D751BA87946166E68001FA86E460E7A0000DA
S1132D8500003A0AE00C595E0041E07A00000000B9
S1132D952A0AE06E6900065E00410C7A0000000015
S1122DA5320AE00CD95E00419C5E00604454701A
S1139CEA5361666574792E74787400004D6F746FCE
S1139CFA722E74787400004C696D69742E747874CA
S1139D0A00005361666574792E747874005155495D
S1139D1A54000A41204261736B65742069736E278C
S1139D2A742031737420466C6F6F72000A412042AB
S1139D3A61736B65742069736E277420326E6420B5
S1099D4A466C6F6F72000E
S1132DB45E0060667A050000499E790600027A0186
S1132DC400009D500D605D507A0100009D7F0D60F1
S1132DD45D507A0100009DA00D605D507A010000F2
S1132DE49DCC0D605D507A0100009DFA0D605D502D
S1132DF47A0100009E060D605D505E0060445470CD
S1052E0440AEDB
S1139D500A5550203D202775272C20444F574E206D
S1139D603D202764272C204F50454E203D20276F50
S1139D70272C20434C4F5345203D20276327000ABF
S1139D80454D455247454E4359203D202773272CC7
S1139D90205245434F56455259203D2027722700F4
S1139DA00A31737420466C6F6F722043414C4C2010
S1139DB03D202779272C20326E6420466C6F6F720A
S1139DC02043414C4C203D20275927000A3173740E
S1139DD020466C6F6F7220434C4F5345203D202724
S1139DE068272C20326E6420466C6F6F7220434CC0
S1139DF04F5345203D20274827000A5155495420F9
S1139E003D20277127000A434F4D4D414E443E00EC
S1132E0601006DF60F86190069E06FE000026FE0BE
S1132E1600046FE0000601006FE600087A00000078
S1132E2600020AE001006FE0000C7A0000000004D3
S1132E360AE001006FE000107A00000000060AE0D5
S1132E4601006FE0001419006FE000187A0000001B
S1132E5600180AE001006FE0001A19006FE0001E77
S1132E667A000000001E0AE001006FE00020010066
S1132E766D76547001006DF60F865E003F0A7A0088
S1132E86000000040AE05E003F0A01006D765470FC
S1132E965E0060660F850F9601006F7400180100CF
S1132EA66F600014690079200001465201006F60CB
S1132EB6001A6901462801006F66001A79000001AD
S1132EC669E07A00000000040AD0F9735E0041E06D
S1132ED67A0100009E10790000015E00499E684851
S1132EE6A859586001E8F87268C86DF07A010000C5
S1082EF69E150FD05EE4
S1132EFB003F140B875A0030D4FB6801006F60004E
S1132F0B10690079200001466E01006F60001A19E9
S1132F1B11698101006F6000206901462C01006F6C
S1132F2B6600207900000169E07A00000000040AC2
S1132F3BD0F96F5E0041E07A0100009E2179000019
S1132F4B015E00499E5A0030D46848A85958600165
S1132F5B787A00000000040AD00CB95E0041E0F857
S1132F6B7268C86DF07A0100009E150FD05E003FAA
S1132F7B140B875A0030D401006F600008690146B7
S1132F8B6E01006F60001A1911698101006F6000F7
S1132F9B206901462C01006F66002079000001694E
S10F2FABE07A00000000040AD0F94F5E39

S1132FB70041E07A0100009E26790000015E004986
S1132FC79E5A0030D46848A859586001007A000017
S1132FD70000040AD00CB95E0041E0F87268C86DBE
S1132FE7F07A0100009E150FD05E003F140B875A3D
S1132FF70030D401006F60000C6901466C01006F5B
S113300760001A1911698101006F60002069014688
S11330172C01006F6600207900000169E07A000047
S11330270000040AD0F96F5E0041E07A0100009EB8
S113303721790000015E00499E5A0030D46848A8F0
S113304759586000887A00000000040AD00CB95E62
S11330570041E0F87268C86DF07A0100009E150F11
S1133067D05E003F140B87406401006F60000C695A
S11330770079200001465601006F60001A19116993
S11330878101006F6000206901462801006F660017
S1133097207900000169E07A00000000040AD00CDF
S10830A7B95E0041E0E9
S10530AC7A01A4
S11330AE00009E32790000015E00499E6848A859CF
S11330BE4614F87268C86DF07A0100009E150FD0A1
S11330CE5E003F140B875E006044547001006DF682
S11330DE0F865E003F0A7A00000000040AE05E00DD
S11330EE3F0A01006D7654705E0060660F850F9681
S11330FE01006F74001801006F60000C69007920E5
S113310E0001465201006F60001A69014628010052
S113311E6F66001A7900000169E07A00000000046E
S113312E0AD0F9735E0041E07A0100009E10790027
S113313E00015E00499E6848A859586001E8F8727C
S113314E68C86DF07A0100009E150FD05E003F1423
S113315E0B875A003334FB7401006F60000869005B
S113316E79200001466E01006F60001A1911698102
S113317E01006F6000206901462C01006F6600207C
S113318E7900000169E07A00000000040AD0F963B7
S109319E5E0041E07A012E
S11031A400009E3D790000015E00499E5A27
S11331B10033346848A859586001787A0000000048
S11331C1040AD00CB95E0041E0F87268C86DF07A68
S11331D10100009E150FD05E003F140B875A003388
S11331E13401006F6000106901466E01006F6000D9
S11331F11A1911698101006F6000206901462C01D0
S1133201006F6600207900000169E07A0000000088
S1133211040AD0F9435E0041E07A0100009E43793C
S11332210000015E00499E5A0033346848A859588A
S11332316001007A00000000040AD00CB95E00416D
S1133241E0F87268C86DF07A0100009E150FD05E38
S1133251003F140B875A00333401006F6000146977
S113326101466C01006F60001A1911698101006F39
S11332716000206901462C01006F6600207900007F
S11332810169E07A00000000040AD0F9635E00419D
S1133291E07A0100009E3D790000015E00499E5ADB
S11332A10033346848A859586000887A0000000048
S11332B1040AD00CB95E0041E0F87268C86DF07A77
S11332C10100009E150FD05E003F140B874064017F
S11332D1006F600014690079200001465601006FF8
S11332E160001A1911698101006F600020690146AC
S11332F12801006F6600207900000169E07A00006F
S11333010000040AD00CB95E0041E07A0100009E7E
S113331150790000015E00499E6848A8594614F897

S11333217268C86DF07A0100009E150FD05E003FF0
S1133331140B875E006044547001006DF60F867AAA
S113334100000000040AE001006FE000067A0000BB
S113335100000E0AE001006FE0001801006D7654D1
S1133361705E0060660F860F957A000000000E0AFA
S1133371E001006FE000187A000000000A0AE00192
S1133381006F6100185E0042666E680012A87946FC
S109339106790000014073
S11333970219006FE0001C7A04000043C46F600049
S11333A71C6DF001006F51002001006F50000C5D90
S11333B7400B876E680013A8794606790000014021
S11333C70219006FE0001C6DF001006F510020012E
S11333D7006F5000085D400B876E680014A879469C
S11333E70679000001400219006FE0001C6DF0012F
S11333F7006F51002001006F5000105D400B876E76
S1133407680015A879460679000001400219006F84
S1133417E0001C6F66001C6DF601006F5100200170
S1133427006F5000145D400B875E00604454705E6C
S11334370060660F860F957A000000000E0AE00110
S1133447006FE000187A000000000A0AE001006F2D
S11334576100185E0042666E680012A87946067915
S1133467000001400219006FE0001C7A04000043CA
S1133477C46F60001C6DF001006F51002001006FE5
S108348750000C5D4044
S113348C0B876E680013A879460679000001400289
S113349C19006FE0001C6DF001006F51002001005A
S11334AC6F5000085D400B876E680014A8794606C0
S11334BC79000001400219006FE0001C6DF001005F
S11334CC6F51002001006F5000105D400B876E6838
S11334DC0015A879460679000001400219006FE037
S11334EC001C6F66001C6DF601006F51002001007B
S11234FC6F5000145D400B875E00604454705E98
S113350B0060660F860F957A000000000E0AE0013B
S113351B006FE000187A000000000A0AE001006F58
S113352B6100185E0042666E680012A87946067940
S113353B000001400219006FE0001C7A04000043F5
S113354BC46F60001C6DF001006F51002001006F10
S113355B50000C5D400B876E680013A87946067903
S113356B000001400219006FE0001C6DF001006FB9
S113357B51002001006F5000085D400B876E6800FF
S113358B14A879460679000001400219006FE00088
S113359B1C6DF001006F51002001006F5000105D96
S11335AB400B876E680015A8794606790000014029
S11335BB0219006FE0001C6F66001C6DF601006FB3
S11335CB51002001006F5000145D400B875E0060BB
S11335DB4454705E0060660F860F9501006F6000A8
S11335EB14690146307A00000000200AF00FE15CF9
S10635FB00FE3894
S11335FE01006DF57A000000001C0AF00FE15C007B
S113360EF8860B977A00000000200AF00FE15C00A9
S113361EFE165E00604454705E0060660F860F9562
S113362E01006F60000C690146307A000000002033
S113363E0AF00FE15C00FEC401006DF57A00000094
S113364E001C0AF00FE15C00FA9E0B977A00000053
S113365E00200AF00FE15C00FEA25E00604454708D
S1139E1053544F50005361666574792E74787400FF
S1139E20004F50454E004F50454E205370656564BA

S1139E3079004F50454E20537461727400434C4F68
S1139E40534500434C4F5345205370656564790077
S10F9E50434C4F5345205374617274005F
S113366E01006DF60F86190069E06FE000026FE04E
S113367E00046FE0000601006FE600087A00000008
S113368E00020AE001006FE0000C7A000000000463
S113369E0AE001006FE000107A00000000060AE065
S11336AE01006FE0001419006FE000187A000000AB
S11336BE00180AE001006FE0001A19006FE0001E07
S11336CE7A000000001E0AE001006FE000200100F6
S11336DE6D76547001006DF60F865E003F0A7A0018
S11336EE000000040AE05E003F0A01006D7654708C
S11336FE5E0060660F850F9601006F74001801005F
S113370E6F600014690079200001465201006F605A
S113371E001A6901462801006F66001A790000013C
S113372E69E07A00000000040AD0F9735E0041E0FC
S113373E7A0100009E5C790000015E00499E684894
S113374EA859586001E8F87268C86DF07A01000054
S108375E9E610FD05E27
S1133763003F140B875A00393CFB6A01006F60006A
S113377310690079200001466E01006F60001A1979
S113378311698101006F6000206901462C01006FFC
S11337936600207900000169E07A00000000040A52
S11337A3D0F9755E0041E07A0100009E6D79000057
S11337B3015E00499E5A00393C6848A85958600184
S11337C3787A00000000040AD00CB95E0041E0F8E7
S11337D37268C86DF07A0100009E610FD05E003FEE
S11337E3140B875A00393C01006F600008690146D6
S11337F36E01006F60001A1911698101006F600087
S1133803206901462C01006F6600207900000169DD
S10F3813E07A00000000040AD0F9555EC2
S113381F0041E07A0100009E70790000015E0049CB
S113382F9E5A00393C6848A859586001007A000035
S113383F0000040AD00CB95E0041E0F87268C86D4D
S113384FF07A0100009E610FD05E003F140B875A80
S113385F00393C01006F60000C6901466C01006F79
S113386F60001A1911698101006F60002069014618
S113387F2C01006F6600207900000169E07A0000D7
S113388F0000040AD0F9755E0041E07A0100009E42
S113389F6D790000015E00499E5A00393C6848A8C3
S11338AF59586000887A00000000040AD00CB95EF2
S11338BF0041E0F87268C86DF07A0100009E610F55
S11338CFD05E003F140B87406401006F60000C69EA
S11338DF0079200001465601006F60001A19116923
S11338EF8101006F6000206901462801006F6600A7
S11338FF207900000169E07A00000000040AD00C6F
S108390FB95E0041E078
S10539147A0133
S113391600009E7A790000015E00499E6848A85916
S11339264614F87268C86DF07A0100009E610FD0E4
S11339365E003F140B875E006044547001006DF611
S11339460F865E003F0A7A00000000040AE05E006C
S11339563F0A01006D7654705E0060660F850F9610
S113396601006F74001801006F60000C6900792074
S11339760001465201006F60001A690146280100E2
S11339866F66001A7900000169E07A0000000004FE
S11339960AD0F9735E0041E07A0100009E5C79006B

S11339A600015E00499E6848A859586001E8F8720C
S11339B668C86DF07A0100009E610FD05E003F1467
S11339C60B875A003B9CFB6B01006F600008690084
S11339D679200001466E01006F60001A1911698192
S11339E601006F6000206901462C01006F6600200C
S11339F67900000169E07A00000000040AD0F96446
S1093A065E0041E07A01BD
S1103A0C00009E83790000015E00499E5A70
S1133A19003B9C6848A859586001787A0000000067
S1133A29040AD00CB95E0041E0F87268C86DF07AF7
S1133A390100009E610FD05E003F140B875A003BC3
S1133A499C01006F6000106901466E01006F600000
S1133A591A1911698101006F6000206901462C015F
S1133A69006F6600207900000169E07A0000000018
S1133A79040AD0F9445E0041E07A0100009E887986
S1133A890000015E00499E5A003B9C6848A85958AA
S1133A996001007A00000000040AD00CB95E0041FD
S1133AA9E0F87268C86DF07A0100009E610FD05E7C
S1133AB9003F140B875A003B9C01006F6000146997
S1133AC901466C01006F60001A1911698101006FC9
S1133AD96000206901462C01006F6600207900000F
S1133AE90169E07A00000000040AD0F9645E00412C
S1133AF9E07A0100009E83790000015E00499E5A25
S1133B09003B9C6848A859586000887A0000000067
S1133B19040AD00CB95E0041E0F87268C86DF07A06
S1133B290100009E610FD05E003F140B87406401C2
S1133B39006F600014690079200001465601006F87
S1133B4960001A1911698101006F6000206901463B
S1133B592801006F6600207900000169E07A0000FE
S1133B690000040AD00CB95E0041E07A0100009E0E
S1133B7994790000015E00499E6848A8594614F8E3
S1133B897268C86DF07A0100009E610FD05E003F34
S1133B99140B875E006044547001006DF60F867A3A
S1133BA900000000040AE001006FE0000E01006D4F
S1133BB97654705E0060660F860F957A00000000E8
S1133BC9040AE001006FE0000E01006F61000E0FAF
S1133BD9E05E0042666E680004A879460679000033
S1133BE901400219006FE000127A04000043C46F18
S1083BF96000126DF0F5
S1133BFE01006F51002001006F50000C5D400B87D8
S1133C0E6E680005A8794606790000014002190086
S1133C1E6FE000126DF001006F51002001006F5034
S1133C2E00085D400B876E680006A879460679008A
S1133C3E0001400219006FE000126DF001006F5198
S1133C4E002001006F5000105D400B876E68000767
S1133C5EA879460679000001400219006FE00012B0
S1133C6E6F6600126DF601006F51002001006F5058
S1133C7E00145D400B875E00604454705E00606606
S1133C8E0F860F957A00000000040AE001006FE032
S1133C9E000E01006F61000E0FE05E0042666E685B
S1133CAE0004A879460679000001400219006FE06E
S1133CBE00127A04000043C46F6000126DF001001D
S1133CCE6F51002001006F50000C5D400B876E6832
S1133CDE0005A879460679000001400219006FE03D
S1073CEE00126DF060
S1133CF201006F51002001006F5000085D400B87E7
S1133D026E680006A8794606790000014002190090

S1133D126FE000126DF001006F51002001006F503F
S1133D2200105D400B876E680007A879460679008C
S1133D320001400219006FE000126F6600126DF677
S1133D4201006F51002001006F5000145D400B878A
S10A3D525E00604454705E43
S1133D590060660F860F957A00000000040AE001EF
S1133D69006FE0000E01006F61000E0FE05E00427C
S1133D79666E680004A879460679000001400219B5
S1133D89006FE000127A04000043C46F6000126DF3
S1133D99F001006F51002001006F50000C5D400BD2
S1133DA9876E680005A87946067900000140021963
S1133DB9006FE000126DF001006F51002001006FE8
S1133DC95000085D400B876E680006A8794606799E
S1133DD9000001400219006FE000126DF001006F4D
S1133DE951002001006F5000105D400B876E680081
S1133DF907A879460679000001400219006FE0001F
S1133E09126F6600126DF601006F51002001006FF9
S1133E195000145D400B875E00604454705E00607F
S1133E29660F860F9501006F600014690146307AA9
S1133E3900000000200AF00FE15C00FE4401006D60
S1063E49F57A0004
S1133E4C0000001C0AF00FE15C00F8A60B977A0047
S1133E5C000000200AF00FE15C00FE225E006044CB
S1133E6C54705E0060660F860F9501006F60000C46
S1133E7C690146307A00000000200AF00FE15C0073
S1133E8CFECA01006DF57A000000001C0AF00FE178
S1133E9C5C00FABE0B977A00000000200AF00FE1D9
S10D3EAC5C00FEA85E006044547041
S1139E5C53544F50005361666574792E74787400B3
S1139E6C005550005550205370656564790055506A
S1139E7C20537461727400444F574E00444F574E35
S1139E8C2053706565647900444F574E20537461B9
S1069E9C727400DA
S1133EB601006DF60F867A0000FFE224010069E037
S1133EC67A0600FFE224F87268E87A00000000062A
S1133ED60AE001006FE000027A0100009EA00100E3
S1133EE66F6000025E005990F8736EE8000FF84E9B
S1133EF66EE800106EE80011F84E6EE8001201003D
S1133F066D7654700F811A800100699054705E00BB
S1133F1660660F946E7D00197A0600FFE2246849F5
S1133F26A9434716A94D470CA9504714A953462838
S1133F3668ED40246EED000F401E6EED0010401834
S1133F466E480006A8434706A8544708400A6EED84
S1133F56001140046EED001219005E0060445470B7
S1133F6601006DF60F966869A94C46247A0600FF90
S1133F76E2247A00000000060AE001006FE0000276
S1133F8601006F71000801006F6000025E00599026
S1133F96190001006D7654705E0060660F9401008F
S1093FA66F7500187A0696
S1133FAC00FFE2246849A9434716A94D470CA950C1
S1133FBC4716A953462E686E400A6E6E000F4004D6
S1133FCC6E6E001068DE401C6E480006A843470660
S1133FDCA854470A400E6E6E001168DE40066E6EE2
S1133FEC001268DE19005E006044547001006DF627
S1133FFC0F966869A94C46247A0600FFE2247A00DE
S113400C000000060AE001006FE0000201006F618E
S113401C000201006F7000085E0059901900010046

S113402C6D7654705E0060660F850F9601006DF619
S113403C7A0100009EAA0FD05C00FF560B970C0070
S113404C4624A8014706A8024716401A7A01000025
S113405C9EBD790000015E00499E7906000140086F
S113406C79060002400219660D605E0060445470CC
S104407C5EE2
S113407D0060660F850C9E6DF67A0100009EAA0FF7
S113408DD05C00FE820B870C00461CA8004714A8C9
S113409D0146147A0100009ECC790000015E0049AF
S11340AD9E400419664004790600010D605E0060B0
S11340BD4454705E0060660F850F9601006DF67AAD
S11340CD0100009EDB0FD05C00FEC60B970C004673
S11340DD24A8014706A8024716401A7A0100009E3C
S11340EDBD790000015E00499E7906000140087903
S11340FD060002400219660D605E00604454705E56
S113410D0060660F850C9E6DF67A0100009EDB0F35
S113411DD05C00FDF20B870C00461CA8004714A8C9
S113412D0146147A0100009ECC790000015E00491E
S113413D9E400419664004790600010D605E00601F
S113414D4454705E0060660F850F9601006DF67A1C
S104415D015D
S113415E00009EE80FD05C00FE360B970C00462441
S113416EA8014706A8024716401A7A0100009EBD11
S113417E790000015E00499E790600014008790628
S113418E0002400219660D605E00604454705E00CA
S113419E60660F850C9E6DF67A0100009EE80FD0C7
S11341AE5C00FD620B870C00461CA8004714A80197
S11341BE46147A0100009ECC790000015E00499EF0
S11341CE400419664004790600010D605E006044E8
S11341DE54705E0060660F850C9E6DF67A010000CA
S11341EE9EFC0FD05C00FD1E0B870C004616A8002C
S11341FE4712A801460E7A0100009ECC79000001F9
S113420E5E00499E5E00604454705E0060661B87CC
S113421E0F8518886EF800017A06000000010AF671
S113422E01006DF67A0100009EFC0FD05C00FD606C
S113423E0B970C004618A800470EA8014710A802BA
S109424E470C400A400882
S110425440066E780001400218880B875E5B
S113426100604454705E0060660F850F967900000C
S1134271096DF001006DF67A0100009F070FD05C14
S113428100FD740B970B870C004618A8014706A87D
S1134291024710400E7A0100009EBD790000015EC5
S10C42A100499E5E006044547064
S1139EA079796E6E79796E6E00005065726D6974A2
S1139EB0436F6D6D616E642E74787400000A526591
S1139EC06164696E67204572726F72000A57726926
S1139ED074696E67204572726F7200436F6D6D61B6
S1139EE06E642E74787400005065726D69745475D5
S1139EF0726E4F70656E2E74787400004D6F746FC0
S1139F00722E74787400004C696D69742E747874C1
S1059F1000004C
S11342AA5E0060660F860F957A0000009F120FE189
S11342BA5E005FF001006FE600080FE055267A0002
S11342CA0000000C0AE001006FE0000E19006FE025
S11342DA000C19006FE0001201006FE500145E0084
S11342EA604454705E0060660F8601006FE6000842
S11342FA0FE201006DF25E0044020B975E00604418

S113430A54705E0060667A37000000100F86010061
S113431A6FE600087A02000000080AF201006DF253
S113432A5E0044020B9701006F6600080FA10FE2BB
S113433A0FF05E005A925E005F047A1700000010C5
S113434A5E00604454705E0060660F850D16790E38
S113435A03E852E617F60FE101006F5000145E00FE
S113436A44A65E006044547001006DF60F867A001D
S113437A0000000C0AE001006FE0000E7921000141
S113438A460A790000016FE0000C400A0D11460647
S109439A19006FE0000CA6
S11343A001006D76547001006DF60F867A000000EF
S11343B0000C0AE001006FE0000E6F60000C0100CA
S11343C06D7654705E0060660F860F9569606F713D
S11343D000181D10470A190069D06F70001869E0B2
S11343E05E00604454705E0060660F850D1617F61C
S11343F00FE101006F5000145E0044A65E006044AC
S10544005470F3
S10B9F120000000000000000044
S11344027A0000FFE23801006F7100045E005FF082
S113441254705E0060667A37000000200F867A02CD
S1134422000000180AF201006DF255D40B970FE158
S11344320FF05E005FAA0F817A0200009F2A7A00C2
S1134442000000100AF05E005D6C7A0000000018A4
S11344520AF07A01000000080AF15E005FF04010E2
S11344627A02000000080AF201006DF255920B97DE
S11344727A01000000180AF17A02000000100AF221
S11344820FF05E005AC20F817A00000000080AF0A2
S11344925E005F9A0D0046C87A17000000205E0096
S11344A2604454705E0060661B971B970F860F95DE
S11344B27A02000000020AE201006DF25C00FF4092
S11344C20B970FD10FF05E005FAA0F817A020000F3
S11344D29F2A7A000000000A0AE05E005D6C0B97D7
S11344E20B975E00604454705E0060660F860F9502
S10744F20FE055B0CF
S11344F601006F6000120B7001006FE000125E0096
S1134506604454705E0060660D057A0400FFE24065
S1134516400601006F44001A01006F40001A0100B3
S11145266F01001A46EC79250001460A7A0659
S113453400FFE27C5A00463879250002460A7A06CF
S113454400FFE29A5A0046387925000B460A7A0698
S113455400FFE2B85A0046387925000C460A7A0669
S113456400FFE2D65A0046387925000D460A7A063A
S113457400FFE2F45A0046387925000E460A7A060B
S113458400FFE3125A00463879250013460A7A06D7
S108459400FFE3305AB3
S113459900463879250014460A7A0600FFE34E5A85
S11345A900463879250015460A7A0600FFE36C5A56
S11345B90046387925001646087A0600FFE38A4043
S11345C96E7925001746087A0600FFE3A84060794B
S11345D925001E46087A0600FFE3C64052792500E6
S11345E91F46087A0600FFE3E4404479250029467B
S11345F9087A0600FFE40240367925002A46087A3C
S11346090600FFE42040287925002B46087A060096
S1134619FFE43E401A7925002C46087A0600FFE498
S10E46295C400C7925002D46067A0644
S113463400FFE47A0FE6461C7A0100009F1A190072
S11346445E00499E7A0100009F1C19005E00499E8A

S11346541A80405401006FE4001601006F40001AF1
S113466401006FE0001A01006F86001601006FC697
S1134674001A7A0000009F327A01000000020AE166
S11346845E005FF07A0000009F3A7A010000000A9E
S11346940AE15E005FF069E51A8001006FE0001231
S11346A40FE05E000E940FE05E006044547001005E
S11346B46DF60F865E00112801006F60001601007D
S11346C46F61001A01006F81001A01006F60001A04
S11346D401006F61001601006F81001601006D7601
S11346E454705E0060660F867A0200FFE2380100B0
S11346F46DF25C00FD080B977A0000FFE2387A0143
S1134704000000020AE15E005FF05E006044547042
S113471401006DF60F867A0000009F427A010000C3
S108472400020AE15E42
S1134729005FF001006D76547001006B2000FFE219
S11347395A01006F01001A460419005470790000E8
S113474901547001006B2100FFE25A01006F11004F
S11347591A1988400C01006F11001A0D800B500DB6
S1134769080F9146F00D8054706DF50D0501006B2E
S11347792000FFE25A01006F01001A472001006B74
S11347892100FFE25A69101D5046040F90401001A1
S1134799006F11001A01006F10001A46E81A806DA4
S11347A97554705E0060660D0555BE0F86460E0D85
S11347B9505C00FD4C0F865C00FF22401C7A000010
S11347C90000020AE07A0100009F425E005F7E0D4D
S11347D90047060FE05C00FF040FE05E00604454ED
S11347E9705E0060660D0555800F86460E0D505CA0
S11347F900FD0E0F865C00FEE440247A00000000F1
S1134809020AE07A0100009F425E005F7E0D0047C5
S1084819080FE05C0044
S113481EFEC640060FE05C00FE8A5E0060445470E4
S10D482E5E0060661B971B977A0576
S113483800FFE24001006F56001A407201006F65E5
S1134848001A7A00000000020AE07A0100009F3291
S11348585E0060360D0047547A00000000020AE04B
S11348687A0100009F425E0060360D00473E7A01E0
S1134878000000020AE17A020000000A0AE20FF0CF
S11348885E005AC20F817A0000FFE2385E005F8A39
S11348980D0047187A0000FFE2387A010000000291
S11348A80AE15E005FF00FE05E00052C0FD6010001
S11348B86F60001A46860B970B975E00604454702E
S11348C81A8001006BA000FFE2567A0000FFE25E47
S10D48D801006BA000FFE25A7A0012
S11348E200FFE24001006BA000FFE2741A800100A6
S11348F26BA000FFE2785470F801386018883861C1
S11349023862188838633890F8FF38911888386409
S1134912F8883865F804386679009E576B80FF681B
S113492219006B80FF6A19006B80FF6C54700100E1
S11349326DF27F67722079009E576B80FF687A0160
S113494200FFE2387A0200009F4A0F905E005AC2CB
S113495201006D7254705A00482E7A0000009F3A8B
S11349627A0100FFE2385E005FF0558C5E0002A818
S10749725A0048C8D4
S1139F1A0A0063616C6C6F63206661696C65640037
S1139F2A408F400000000000BFF000000000000066
S1139F3A0000000000000000C000000000000054
S10B9F4A3F50624DD2F1A9FC66

S11349767A0000009F5201006DF07A0000FFE49870
S11349865E0059080B977A0000FFE49801006DF06A
S11349965E004D200B9754705E0060667A0500FF3B
S11349A6E4D80D040F960C4458600114AC00476814
S11349B6AC014710AC0258700106AC03587000C234
S11349C65A004AC655AA7A0100009F540FE05E00BA
S11349D659660D00461E7A0000009F5A01006DF0CD
S11349E67A0000009F5701006DF00FE05E00590842
S11349F60B970B9701006DF67A0000009F57010095
S1134A066DF00FD05E0059080B970B9701006DF5FB
S10D4A167A0000009F5740507A0118
S1134A2000009F540FE05E0059660D00461E7A0099
S1134A3000009F5A01006DF07A0000009F570100AB
S1134A406DF00FE05E0059080B970B9701006DF6B0
S1134A507A0000009F5701006DF00FD05E005908E7
S1134A600B970B9701006DF57A0000009F5701002B
S1134A706DF05E004B1C0B970B9701006DF65E000B
S1134A804D200B974040403E01006DF67A00000038
S1134A909F5701006DF00FD05E0059080B970B97DD
S1134AA001006DF57A0000009F5701006DF05E0074
S1134AB04B1C0B970B970FD05E0059AC0B500D019D
S10A4AC00FD05E0057845E76
S1084AC700604454707F
S10E9F520C000A00002573000A0C003D
S1134ACC188838BA38B8F85038B919000B5079200F
S1134ADC01184DF8F83038BA28BCF88038BC54703B
S1134AEC0C8820BC737047FA38BBE07F30BC547021
S1134AFC20BC736047FA29BDE0BF30BC0C985470DE
S1134B0C7EBC736047067900000154701900547021
S1134B1C5E0060667A0600FFE5187A050000000463
S1134B2C7A00000000180AF00AD07308470E7A00C6
S1134B3C000000180AF00AD00B70400A7A0000003B
S1134B4C00180AF00AD00F85189968E901006DF076
S1134B5C01006F71001C0FE05E0059C80B971955CB
S1134B6C0D5017F00AE0680947220D5017F00AE0C0
S1134B7C6808A80A4606F80D5C00FF640D5017F090
S1134B8C0AE068085C00FF580B5540D45E00604493
S1054B9C547050
S1134B9E1911400C19880B58792806824DF80B51C0
S1134BAE1D014DF0547001006DF50D090D8D28D6C4
S1134BBE17500D010D99470C0D197969006079494C
S1134BCE001040060D19796900600DD50D90148DF6
S1134BDE0CD8C88038D63DD639D67900000455B0E6
S1134BEE01006D7554706DF56DF40D090D8528D6A4
S1134BFE17500D010D99470C0D197969006079490C
S1134C0E001040060D19796900600D54119411942A
S1134C1E11941194EC0F0D90148CED0F148D0CC890
S1134C2EC88038D63CD60CD8C88038D63DD639D66AF
S1134C3E790000045C00FF586D746D7554700100AB
S1134C4E6DF619EE7900000F5C00FF4419667908C2
S1134C5E00030DE05C00FF4E0B56792600034DEE6C
S1134C6E790800020DE05C00FF3C19667908002804
S1134C7E0D605C00FF70790800100D605C00FF662C
S1094C8E7908000E0D6021
S1134C945C00FF5C790800060D605C00FF52790834
S1134CA400010D605C00FF48790800020D605C00A0
S1134CB4FF3E01006D7654707908000119005C0011

S1134CC4FF2E7908000219005A004BF46DF60C8E7E
S1134CD4AE0C460455E2401CAE0A460455DA4014B1
S1134CE4AE0D460455D2400C17D60D687900000169
S1134CF45C00FEFC6D76547001006DF60D0E0D869E
S1134D040D6079080040528009E0791000800D0895
S1134D1419005C00FEDA01006D7654705E00606673
S1134D24790C000119447A0600FFE5687A0500004E
S1134D3400047A00000000180AF00AD07308470E32
S1134D447A00000000180AF00AD00B70400A7A00B7
S1134D54000000180AF00AD00F85189968E90100C9
S1134D646DF001006F71001C0FE05E0059C80B97D2
S1134D7419550D5017F00AE0680947560D5017F0FE
S1094D840AE06808A80A1A
S1134D8A460A0DC80D405C00FF68403C0D5017F001
S1134D9A0AE06808A80D460C790800020D405C0079
S1134DAAFE4840240D5017F00AE06808A80C46068E
S1134DBA5C00FEFE40120D5017F00AE0680817D097
S1134DCA0D080DC05C00FE220B5540A05E00604436
S1054DDA547010
S1134DDC1911400C19880B58792806824DF80B5180
S1134DEC1D014DF054705C0009145C0000B45504B3
S1134DFC5A004ED601006DF618886AA800FFE66BC0
S1134E0C6AA800FFE66D18886AA800FFE66C6AA81A
S1134E1C00FFE5A979080080790000045C0008080C
S1134E2C79080010790000205C0007FC7906000F5C
S1134E3C790800100D605C0007EE0D687900000B1B
S1134E4C5C0007E40D687900000D5C0007DA790853
S1134E5C0050790000095C0007CE790800D6790070
S1134E6C00075C0007C219660D605C000710790E21
S1134E7C00010DE05C0006E60DE05C0007000D6828
S1134E8C7900002B5C0007A00D687900002F5C00F3
S1134E9C07960DE8790000275C00078C01006D76FE
S1134EAC54707908000519005C00077C79000064D4
S1134EBC5C00FF1C790800C419005C00076A7908C0
S1094ECC00037900000160
S1134ED25A00563479080002790000055C00075233
S1134EE2790800CC19005A0056345E0060667A04D1
S1134EF20000A0107A0000009FDF01006DF05E0049
S1134F024B1C0B9719EE19660DE5790D001052D55E
S1134F120D6009505C00070C17506DF001006DF431
S1134F225E004B1C0B970B870B56792600104DE046
S1134F327A000000A01601006DF05E004B1C0B9777
S1134F420B5E792E00044DBE5E0060445470010076
S1074F526DF67A0675
S1134F5600FFE5A8790000065C0006C468E87C60EB
S1134F66734047367900000E5C0006B40C8E734816
S1134F7647045C0002C8735E47085C0002C25A001D
S1134F865044730E47085C0002B85A005044731E1F
S1134F9647045C0002AE5A0050447C6073604722AB
S1134FA67900000C5C0006780C8E730847085C00D9
S1134FB600925A005044731E587000825C0001C66A
S1134FC6407C7C607320471E7900000A5C00065013
S1134FD60C8E730847065C00020A4062731E475E26
S1134FE65C00023E40587C6073104752790000080B
S1134FF65C00062C0C8E7368470A5C00FDFC5C00A3
S1135006FECE403A734E471A790800C0790000096C
S11350165C00061A79080003790000055C00060E99

S1135026401C737E471879080050790000095C001C
S113503605FC79080002790000055C0005F0010013
S10850466D7654705E5D
S113504B00606619DD790000265C0005CE0C8E73BB
S113505B68587001047A0600FFE5AB790000086D10
S113506BF00FE179080026790000255C0005CC0BD5
S113507B870DD05C0004E80DD05C000502790500B8
S113508B200D505C0004C06868E8605860009C6E9B
S113509B680001473CA801471EA8034772A80547B0
S11350AB3EA8064726A8084716A809475CA80A473F
S11350BB26A80B4760406C5C0001865A0051526A6C
S11350CB2800FFE5AA17500D08400E5C0002244090
S11350DB765C00035440700DD8790000214024798D
S11350EB0800400D505C0005406E6E0002CE806AD6
S11350FBAE00FFE5A96A2800FFE5A917500D087953
S113510B0000045C000522403E5C00038E40385CCB
S113511B00018840326E680002472C0D505C00047E
S113512B0E40240D505C000406401C6868E860A820
S108513B2046080D50A1
S11351405C0003F6400C686EEE60AE400D505C00F0
S113515003E87A0000FFE66C7D0070000DD05C0070
S1135160045440226A2800FFE66D470818886AA89D
S113517000FFE66D0DD05C00041079080001790092
S10A518000275C0004AE5E92
S113518700604454705E0060667900002E5C000482
S11351978E0C8EE8C017504626790000406DF07AD2
S11351A70500FFE5EB0FD17908002E7900002D5C90
S11351B700048C0B870D060D010FD05C0005627987
S11351C70000015C00039C790800017900002F5C53
S11351D700045A5E0060445470790000365C000492
S11351E73E54706DF6790000225C0004320C8E7316
S11351F768472A7358472619005C0003866A280004
S1135207FFE66D470C5C0001C819005C0003A04072
S11352170C79080001790000275C0004106D7654AF
S1135227706DF67900002A5C0003F40C8E736847EF
S113523708735847045C0004826D765470547054A5
S113524770547054705E0060667A0600FFE5AB68C1
S113525768E803A80246426E680003E80747186E2A
S1135267690003E9071751177178106A2800009F2F
S10852775E17505C000E
S113527C02D46E680003E8071750790100011A087D
S113528C4B04101140F817117A0000FFE66B680A03
S113529C169A688A5E00604454705E0060667A06F3
S11352AC00FFE5AB6868E803A80246406E6800039C
S11352BCE80747186E690003E9071751177178104F
S11352CC6A2800009F5E17505C0002626E68000340
S11352DCE8071750790100011A084B04101140F824
S11352EC7A0000FFE66B680A149A688A5E006044D1
S11152FC54707A0000FFE66C7D0072006A2891
S113530A00FFE5AEA801470AA8024716A8034722E9
S113531A54706A2900009F6617517A0000009F663D
S113532A40686A2900009F7A17517A0000009F7823
S113533A40586A2800FFE5AD470EA801471AA8029C
S113534A4726A803473254706A2900009F9F17D142
S113535A7A0000009F9F40326A2900009FA317D159
S113536A7A0000009FA340226A2900009FB517D143
S10D537A7A0000009FB540126A2973

S113538400009FBD17D17A0000009FBD55025470E1
S11353945E0060660F840D187A0600FFE66E6A28C5
S11353A400FFE5B217500C8018886A2900FFE5B1A5
S11353B41751091069E01D804F0269E801006BA4CD
S11353C400FFE670F8016AA800FFE66D55065E006B
S11353D4604454705E0060667A0600FFE66E7905E9
S11353E40008696047446960792000084E026965D2
S11353F47A0400FFE6706DF5010069417908002223
S1135404790000215C0002780B870D05696119019D
S113541469E117F0010069410A81010069C169600A
S1135424460818886AA800FFE66D5E00604454705D
S10454345E16
S11354350060666A2800FFE5ABE80317500D051900
S1135445EE790600210D0047067925000146100F68
S1135455E05C0001DA0DE80D605C0001D2403C79A7
S1135465250002462E6A2800FFE5AEE80717500D12
S113547505790100011A084B04101140F86A280048
S1135485FFE66B1750660147067908000140060DD4
S1135495E840020DE80D605C0001945E0060445431
S11354A5706DF66A2800FFE5AD6AA800FFE5AA4618
S11354B53C19660D68790000285C0001720D687956
S11354C500002C5C0001680D68790000305C000168
S11354D55E0D68790000345C0001540D68790000A5
S11354E5385C00014A0D687900003C404018886A21
S11354F5A800FFE66B790600010D605C0000867964
S1135505080011790000285C00012479080003795B
S113551500002B5C0001180D60554A7908001279CB
S108552500002C5C00F6
S113552A01080D687900002F5C0000FE6D76547047
S113553A6DF60D065C0000E4C88017500D080D6077
S113554A5C0000E66D7654706DF60D065C0000CCC7
S113555AE87F17500D080D605C0000CE6D7654701D
S113556A01006DF60D0617F6103679080008786003
S113557A6B2000FFE0085C0000B001006D765470F8
S113558A5E0060660D0617F610367A0500FFE00026
S113559A0AE569505C00008417500D06703E0D68D9
S11355AA69505C0000845E00604454705E0060666B
S11355BA1B971B977A0500FFE66C0D0EFE01790017
S11355CA00010DE11A094B04101040F868591751EC
S11355DA66104704702E4002722E701E17560DE095
S11355EA17F0103001006FF000040D6801006F71AD
S11355FA000478106B2000FFE000010069F1552ACE
S113560A790000011B5E4B04101040F86859158994
S109561A68D90B970B9702
S10456205E28
S113562100604454706AA8004000036A28004000E7
S11356310154700D016AA9004000030D806AA8009E
S113564140000154705E0060667A03004000030F5E
S1135651840F9568BC19EE196640180DC055C6E84C
S11356610F471868BC6A280040000168D80B750B06
S11356715E0B566F7000181D064DE00DE05E006075
S11356814454705E0060660D0C0D830F956F7400BA
S11356911819EE1966401E0D30558AE81F471A0D79
S11356A1C06AA80040000368586AA8004000010BC3
S11356B1750B5E0B561D464DDE0DE05E00604454D6
S11356C17001006DF619EE7A0000FFE62B790100F7
S11356D1405C0001140D06790000015C00FEAA0D77

S11356E16647166DF67A0100FFE62B7908002A79E1
S11356F1000029558E0B870D0E790000015C00FE19
S1135701B40DE001006D76547019006BA000FFE643
S1065711766BA011
S113571400FFE67419006BA000FFE67A6BA000FF9C
S1135724E67854705E0060667A0400FFE6767A05D4
S113573400FFE6740F830D1E7FF5725019994030F4
S11357446940792001004C2C695017F068397800BE
S11357546AA900FFE67C69500B5017F07901010038
S113576401D0531069D869400B5069C00B730B59AE
S10E57741DE94DCC7FF570500D905ED9
S113577F00604454705E006066790E01007A040085
S113578FFFE67A7A0500FFE6780F830D127FF57235
S113579F50199940346940792001004C32695017F0
S11357AFF001D053E00D8017F0683978006AA90033
S11357BFFFE77C69500B5017F001D053E069D869AC
S11357CF400B5069C00B730B590D201D094DC67F3C
S11357DF570500D905E00604454705E0060667A01
S11357EF0500FFE67A0F840D1E7FF5725019664090
S11357FF381DE64C386B2000FFE678695119107994
S113580F10010017F07901010001D053100D80171B
S113581FF00D6117F10AC178006A2800FFE77C6871
S113582F9869501B5069D00B5669504EC47FF57061
S113583F500D605E00604454705E0060667A050030
S113584FFFE6760F840D1E7FF57250196640381DE3
S113585FE64C386B2000FFE674695119107910017B
S108586F0017F07901B0
S1135874010001D053100D8017F00D6117F10AC117
S113588478006A2800FFE67C689869501B5069D049
S11058940B5669504EC47FF570500D605ED9
S11358A100604454706B2000FFE67617F079010124
S11358B10001D053100D8054706B2000FFE67A175E
S11058C1F07901010001D053100D805470E7
S1139F5E20282C3034383C20120100010000000868
S1139F6EFEEFF100001000101020109022700010199
S1139F7E00C064090400000300000003070581020A
S1139F8E4000FF070502024000FF07058302400061
S1139F9EFF0403090412035500530042002000542A
S1139FAE00450053005400080331002E00300022F8
S1139FBE0355005300420020005400450053005443
S1139FCE002000500052004F004700520041004D48
S1049FDE007F
S11399C00023002B0033003B0027002F0037003F0C
S1139FDF30302030312030322030332030342030B5
S1139FEF352030362030372030382030392030417B
S1139FFF2030422030432030442030452030460A61
S10CA00F002530325820000A003C
S11358CE01006DF67A0600FFE8B8010069607A01FF
S11358DE41C64E6D5E007F1E7A1000003039010006
S11358EE69E06960198817707A01000080005E0014
S10D58FE7EF80D1001006D76547062
S11359085E0060660F857A06000000047A000000D6
S113591800180AF00AE07308470E7A00000000181E
S11359280AF00AE00B70400A7A00000000180AF037
S11359380AE00F8601006DF001006F70001C010082
S11359486DF001006DF51A91790000015E00608E1B
S11159587A170000000C0D065E0060445470C8

S11359665E0060660F860F9540040B760B756869BB
S113597668581C8946040C9946F068681750685D98
S10D5986175519505E006044547079
S11359905E0060660F840FC60F950FE00B766C599F
S10F59A0688946F60FC05E006044547036
S11359AC5E0060660F851AE640020B760FD00B750E
S10F59BC680946F60FE05E00604454707A
S11359C85E0060660F850F9601006F710018010075
S11359D86DF101006DF601006DF51A917900000172
S11359E85E00608E7A170000000C5E0060445470FD
S11359F80FC446120FD5460E792107FF46120FB37F
S1135A084604156E4A0A1AC47A0500000008186687
S1135A185A005C8C0D9946120FC4461A0FD54616C8
S1135A280FB3461E16E6580002660FA246360FB39A
S1135A3846325800025A0FA246140FB346105800B4
S1135A48024E0CE60D190FA40FB55A005C981035D9
S1135A581234792C00104C0C1B5910351234792C44
S1135A6800104DF410331232792A00104C0C1B51DC
S1135A7810331232792A00104DF4580000CA1AC4A0
S1135A881AD5186619995800020601006DF201002B
S1135A986DF301006F23000401006DF30100692316
S1135AA8795B800001006DF30FF2550E0B970B978E
S1135AB801006D7301006D7254706DF601006DF590
S1135AC801006DF401006DF301006DF201006DF149
S1135AD801006DF0010069140100692510341035C7
S1095AE81FD44218451013
S1135AEE01006F14000401006F2500041FD4440647
S1135AFE0F940FA10FC26816682E0FA00100691430
S1135B0E01006F1500040100690201006F03000418
S1135B1E796C000F796A000F69191119111911198E
S1135B2E1119796907FF69011111111111111160
S1135B3E796107FF792907FF5870FEAE0D11587072
S1135B4EFEC103512341035123410351234103396
S1135B5E12321033123210331232794C0080794ADA
S1135B6E00800D901910474C792000364E3E792057
S1135B7E00204D0E793000200FB34702700A0FA399
S1135B8E1AA2792000104D14793000100D380DB380
S1135B9E0D2B0DA219AA0D884702700B0C884F14FA
S1135BAE113213334402700B1B5040F01AA27A03C6
S1135BBE000000010C6815E84B2A0AB544020B7469
S1135BCE0AA4792C01005850008011341335440275
S1075BDE700D0B59DF
S1135BE2792907FF5860006E1AC41AD5580000A617
S1135BF21FA446061FB55870FE8A1AB544087A34A4
S1135C020000000145061AA4450440121AA41735E0
S1135C1217347A150000000144020B740CE60FC41A
S1135C2246080FD41AD57919FFE00DCC460C0D4C5A
S1135C320DD40D5D19557919FFF0792C0100450832
S1135C42113413350B5940F2792C00804C0810356E
S1135C5212341B5940F20D994E10113413350D991C
S1135C624A08113413350B5940F4732D471C0CD8D1
S1135C72E80B47167A15000000844020B74792CCE
S1135C8201004D06113413350B59113413351134F8
S1135C92133511341335796C000F101910191019BB
S1135CA21019649C101C1006131C01006D70010076
S1135CB2698401006F85000401006D7101006D723A
S1135CC201006D7301006D7401006D756D76547082

S1135CD201F0640158600080792407FF5860006C6A
S1135CE25800007401F064235860006C40520F8521
S1135CF201F06415460E0D44464601F06423464006
S1135D025800005410311230792800104C0C1B5CDF
S1135D1210311230792800104DF4580000BA0FA543
S1135D2201F06435472610331232792A00104C0CE5
S1135D321B5410331232792A00104DF45800009E7E
S1135D421AC4100E131C1AD55800018E790E07FFC0
S1135D521AC41AD5580001621AC418EE790E07FF45
S1135D6219DD790500085800015001006DF60100A4
S1135D726DF501006DF401006DF301006DF2010098
S1135D826DF101006DF0681E6826156E691C111C09
S1135D92111C111C111C796C07FF692411141114B5
S1135DA211141114796407FF01006D1001006911C8
S1135DB27968000F01006F23000401006922796AE8
S1095DC2000F792C07FF1E
S1135DC85870FF06792407FF5870FF120DCC5870DE
S1135DD8FF160D445870FF40194C791C03FF0DCE74
S1135DE8792EFFCC58D0FF5279480010794A001019
S1135DF81AC47A0500000100103312321031123030
S1135E08441C0AB144087A100000000145040AA0A2
S1135E1840040AA004011235123444E0401C1AB1AC
S1135E2844087A300000000145041AA040041AA06F
S1135E3804011235DD01123444C2730D46080AB158
S1135E4844020B700AA001F064014702700D792C1B
S1135E5800804C06103512341B5E792E07FF58C09C
S1135E68FEE40DEE4E2E113413354402700D0DEE83
S1135E784A040B5E40F0732D47180CD8E80B471201
S1135E887A1500000008440A0B74792C00804D022F
S1135E980B5E4020732D471C0CD8E80B47167A1568
S1135EA80000000844020B74792C01004D061134DC
S1095EB813350B5E1134EB
S1135EBE13351134133511341335796C000F101E4D
S1135ECE101E101E101E64EC101C100E131C01006D
S1135EDE6D700100698401006F85000401006D710E
S1135EEE01006D7201006D7301006D7401006D751B
S1095EFE01006D765470F3
S1135F0401006DF201006DF101006F010004010055
S1135F1469000D821112111211121112796207FF15
S1135F240D8A7968000F793203FF4B4A79220053B3
S1135F344E44794800107912FFEC4B227922002059
S1135F444C0C0D224F1E103112301B5240F40F9093
S1135F547912FFE00D224F0C10301B5240F6113022
S1135F640B524BFA796A8000470217B001006D7136
S10D5F7401006D7254701A8040F2B0
S10F5F7E5E007E3CA8014702190054702D
S1135F8A5E007E3C0C884E0419005470F80154706C
S1135F9A5E007E3C0C884604F80154701900547064
S1135FAA01006DF201006DF11AA20F9147224A0610
S1135FBA7902080017B17912041F1B52103144FAEF
S1135FCA1031121210311212103112121031121230
S1135FDA698201006F8100026F8A000601006D71F8
S1095FEA01006D7254700A
S1135FF001006DF2010069020100699201006F0264
S1116000000401006F92000401006D725470E1
S113600E6DF301006DF201006DF101006DF06C0B8B
S113601E689B0B011B7246F601006D7001006D71DA

S10B602E01006D726D735470E3
S11160365E007E3C1A08470479000001547096
S113604401006F76001401006D7201006FF20010FD
S113605401006D7201006D7301006D7401006D75B3
S1056064547073
S113606601006DF501006DF401006DF301006DF2A1
S113607601006F72001001006DF201006FF600144B
S10B608601006F720004547065
S113608E5E0060667A37000000B27A0300FFE89084
S113609E7A0400FFE8800D0201006FF100AA0100EF
S11360AE6F7600CE01006F7500D20D00466C0100B5
S11360BE6F7000CA460E790004526BA000FFE8BC55
S11360CE5A00616001006F7000CA6E080010E80785
S11360DE17D0460C790005146BA000FFE8BC407284
S11360EE01006F7000CA6E08001073284610730803
S11360FE470C790105166BA100FFE8BC4054010063
S113610E6F7000CA6E0800107368464601006F7008
S113611E00AA01006BA000FFE8B40D2017F00100E8
S113612E6BA000FFE87C01006F7000CA01006BA03A
S113613E00FFE8821A8001006BA000FFE886188832
S113614E68C8F8306EF800885C001CBA0D005870F1
S113615E0A1C7900FFFF5A006B9E6868A8255860D9
S113616E09CC0B76188868C80FF001006BA000FFEE
S107617EE8940FF09F
S113618201006BA000FFE8981A8001006BA000FFDA
S1096192E89C01006BA074
S113619800FFE8A01A8001006BA000FFE8A401003B
S11361A86BA000FFE8A81A8001006BA000FFE8AC11
S11361B840306868A8204718A8234720A82B470A17
S11361C8A82D461C7D40701040167D40703040104D
S11361D87C407330460A7D40704040047D40702007
S11361E80B766868A82D47CAA82B47C6A82047C2BC
S11361F8A82347BEF9206AA900FFE88A6869A9307D
S1136208460AF9306AA900FFE88A0B761A8001006A
S11362186BA000FFE88C6868A82A586000800FD03C
S11362280BF001006FF000A07308470A01006F70BC
S113623800A00B70400601006F7000A00F851BF0D3
S113624801006FF000967308470A01006F7000960B
S11362581B70400601006F700096690017F001007B
S11362686BA000FFE88C4C167D40701001006B207A
S113627800FFE88C17B001006BA000FFE88C010059
S10962886B2000FFE88C0F
S113628E7A20000002004F0E7D4070007900051841
S113629E6BA000FFE8BC0B76686817D0796000FF2F
S11362AE17F078006A280000A02073285870008623
S11362BE1A8001006BA000FFE88C4064686817D059
S11362CE7930003017F001006FF000A07A01000062
S11062DE02001A810F907A010000000A5E91
S11362EB007EF801006B2100FFE88C1F904D240109
S11362FB006B2000FFE88C7A010000000A5E007F30
S113630B1E01006F7100A00A9001006BA000FFE853
S113631B8C400E7D407000790005186BA000FFE8E0
S113632BBC0B76686817D0796000FF17F078006AAA
S113633B280000A020732846867A00FFFFFFF0189
S113634B0069B06868A82E586001000B766868A8CE
S113635B2A466C0FD00BF001006FF00096730847C1
S113636B0A01006F7000960B70400601006F7000FE

S113637B960F851BF001006FF000A07308470A010D
S113638B006F7000A01B70400601006F7000A069C6
S113639B0017F0010069B04C0A7A00FFFFFFFFF0101
S11363AB0069B0010069307A20000002004F0E7DB6
S11363BB407000790005186BA000FFE8BC0B7668F2
S11363CB6817D0796000FF17F078006A280000A0E7
S10863DB207328477642
S11363E01A80010069B04058686817D079300030CE
S11363F017F001006FF000A07A01000002001A817B
S11364000F907A010000000A5E007EF801006931F6
S11364101F904D1C010069307A010000000A5E00E4
S11364207F1E01006F7100A00A90010069B0400E49
S11364307D407000790005186BA000FFE8BC0B7667
S1136440686817D0796000FF17F078006A280000A9
S1136450A020732846927A000000002001006BA060
S113646000FFE8B06868A8684708A86C4704A84C10
S11364704610686817D017F001006BA000FFE8B062
S11364800B766868A825587004A4A84558700212B2
S1136490A8475870020CA8584742A863587002E4F2
S11364A0A8644738A865587001F8A866587001F2C7
S11364B0A867587001ECA8694722A86E5870043089
S11364C0A86F4718A87058700398A873587002F8FB
S10964D0A8754708A87837
S10D64D647045A00696A01006B20B5
S11364E000FFE8B07A200000006C46440FD00B9008
S11364F001006FF000AA7308470A01006F7000AA39
S11365000B70400601006F7000AA0F851B900100FD
S11365106FF000967308470A01006F7000961B70B6
S1136520400601006F700096010069025A0066720E
S113653001006B2000FFE8B07A200000006858607B
S1136540009A6868A875470CA8584708A8784704B4
S1136550A86F46420FD00BF001006FF000AA73083A
S1136560470A01006F7000AA0B70400601006F70AC
S113657000AA0F851BF001006FF000967308470A0D
S113658001006F7000961B70400601006F7000964B
S11365906900177040400FD00BF001006FF00096B8
S11365A07308470A01006F7000960B7040060100E4
S11365B06F7000960F851BF001006FF000AA73083F
S11365C0470A01006F7000AA1B70400601006F703C
S10965D000AA690017F0A8
S11365D60F825A0066726868A875470CA858470860
S11365E6A8784704A86F46420FD00BF001006FF05E
S11365F600967308470A01006F7000960B704006F9
S113660601006F7000960F851BF001006FF000AA62
S11366167308470A01006F7000AA1B704006010049
S11366266F7000AA6900177040400FD00BF001008D
S11366366FF000AA7308470A01006F7000AA0B7077
S1136646400601006F7000AA0F851BF001006FF072
S113665600967308470A01006F7000961B70400688
S113666601006F700096690017F00F820100693010
S11366767A20FFFFFFFFF460A7A00000000010100AF
S113668669B0686817D06DF07A01000000020AF15C
S11366960FA05C00050E0B875A0068DE01006B2015
S11366A600FFE8B07A200000004C46420FD00B9062
S11366B60B9001006FF000AA7308470A01006F7080
S10966C600AA0B70400660
S11366CC01006F7000AA0F851B901B9001006FF0E7

S11366DC00967308470A01006F7000961B70404ABE
S11366EC01006F70009640420FD00B900B9001008D
S11366FC6FF000AA7308470A01006F7000AA0B70B1
S113670C400601006F7000AA0F851B901B900100BF
S113671C6FF000967308470A01006F7100961B71A6
S113672C400601006F7100960F907A010000008003
S113673C0AF15E005FF0010069307A20FFFFFFF72
S113674C460A7A0000000006010069B0686917D197
S113675C01006F70008401006DF001006F70008404
S113676C01006DF07A00000000080AF05C00078657
S113677C0B970B975A0068DE0FD00BF001006FF0EC
S113678C00AA7308470A01006F7000AA0B70400639
S113679C01006F7000AA0F851BF001006FF00096CB
S11367AC7308470A01006F7000961B7040060100C6
S10967BC6F7000966E08E9
S11367C200015A0069300FD00B9001006FF000AA4C
S11367D27308470A01006F7000AA0B70400601009C
S11367E26F7000AA0F851B9001006FF0009673086B
S11367F2470A01006F7000961B70400601006F701C
S11368020096010069000F825E0059AC01006FF02F
S113681200AA010069317A21FFFFFFF471201003D
S113682269311F814C0A0100693001006FF000AA2F
S11368320FF001006BA000FFE8981A8001006BA023
S113684200FFE8A401006B2000FFE88C01006F71D8
S110685200AA1A9001006BA000FFE8AC5AE9
S113685F0069780FD00B9001006FF0009673084713
S113686F0A01006F7000960B70400601006F7000F5
S113687F960F851B9001006FF000AA7308470A015A
S113688F006F7000AA1B70400601006F7000AA0111
S113689F0069007A6000FFFFFFF01006FF0009601AF
S11368AF0069307A20FFFFFFF460A7A00000000DD
S11368BF01010069B0790000786DF07A01000000E2
S11368CF020AF101006F7000985C0002CE0B870F74
S11368DFF00F825E0059AC01006FF000AA5A0069F5
S11368EF780FD00B9001006FF000A07308470A01D7
S11368FF006F7000A00B70400601006F7000A00FB7
S113690F851B900F81730847060F901B7040020F72
S113691F90010069006B2100FFE8886981404AF804
S113692F2568F80FF00F827A000000000101006F55
S113693FF000AA0FF101006BA100FFE8981A910173
S106694F006BA136
S113695200FFE8A401006B2100FFE88C1A8101000B
S11369626BA100FFE8AC400E790005186BA000FF95
S1136972E8BC7D4070006868A86E587001B86A2848
S113698200FFE8807308586001AC7C407310463600
S113699201006B2000FFE8AC4F2C40207A0100007D
S11369A200017A0000FFE88A5C0013B87A0000FF56
S11369B2E8AC010069011B710100698101006B20D0
S11369C200FFE8AC4ED601006B2000FFE8A046248E
S11369D201006B2000FFE8A4461A01006B2000FFB0
S11369E2E8A8461001006F7100AA0FA05C0013749F
S11369F25A006AFC01006B2000FFE8940FA11A9071
S1136A0201006FF0009C01006FF000A04F30010005
S1136A126F71009C0FA05C00134A40227A000000B1
S1136A2200880AF07A01000000015C0013367A0044
S1136A3200FFE8A0010069011B71010069810100E7
S1136A426B2000FFE8A04ED401006B2000FFE89802

S1136A5201006B2100FFE8941A9001006FF0009C83
S1136A6201006F7100A00A8101006FF100A00F8085
S1136A724F3601006F71009C01006B2000FFE89408
S1136A825C0012E040227A00000000880AF07A01DA
S1136A92000000015C0012CC7A0000FFE8A40100B0
S1136AA269011B710100698101006B2000FFE8A4E9
S1096AB24ED401006F71D8
S1116AB800AA01006F7000A01A8101006B207C
S1136AC600FFE8985C00129840227A0000000088D4
S1136AD60AF07A01000000015C0012847A0000FFCC
S1136AE6E8A8010069011B710100698101006B209F
S1136AF600FFE8A84ED47C407310473601006B2094
S1136B0600FFE8AC4F2C40207A01000000017A0018
S1136B1600FFE88A5C0012487A0000FFE8AC010037
S1136B2669011B710100698101006B2000FFE8AC5C
S1136B364ED60B76404001006FF600A6400E0100CC
S1136B466F7000A60B7001006FF000A601006F7056
S1136B5600A668086EF8009547086E780095A82584
S1136B6646DC01006F7100A61AE10FE05C0011F02C
S1136B7601006F7600A6686847145C0012900D004A
S1136B86460C6A2800FFE88073085870F5D45C0049
S1136B96125E6B2000FFE8887A17000000B25E00E1
S1136BA6604454705E0060667A37000000247A05FC
S1096BB6000000040AF5D3
S1136BBC01006FF0001801006FF1002001006FF16C
S1136BCC001C18AA689A6F72003C0C22463CAA5807
S1136BDC472CAA644712AA69470EAA6F4712AA75D3
S1136BEC4706AA78471840227A000000000A40069C
S1136BFC7A000000000801006FF00014400C7A00CA
S1136C0C0000001001006FF000146F70003C79203D
S1136C1C0064470679200069461201006F70001862
S1136C2C4C0A01006F74001817B4400601006F740E
S1136C3C00181AE61AB30FC4467401006B2000FF48
S1136C4CE890476AF83068D87A0600000001406281
S1136C5C0FD00AE00F82010069F00FC001006F71C1
S1136C6C00145E0085220100697068890FA0680812
S1136C7CA8094E0C0FD00AE0680989306889401EB8
S1136C8C6F70003C79200078460A0FD00AE068093F
S1136C9C895740080FD00AE06809893768890FC003
S1096CAC01006F710014EA
S1116CB25E0085220F840B760FC4469E6A286F
S1136CC000FFE880732847626F70003C0C00465A4F
S1136CD0A858471EA86F4706A8784716404C0100DE
S1136CE06F70001847440FE00B760AD0F9306889BB
S1136CF0403801006F700018473001006F700020AA
S1136D000B7001006FF000201B70F93068890100DF
S1136D106F7000200B7001006FF000201B706E7904
S1136D20003D68897A03000000020FB00AE00F8378
S1136D3001006FF600146F70003C79200064470671
S1136D4079200069467201006B2000FFE890460835
S1136D5001006F700018476001006F7000184D420A
S1136D606A2800FFE880733847160B7301006F70C1
S1136D7000200B7001006FF000201B70F92B4036D0
S1136D806A2800FFE8807348472E0B7301006F7079
S1136D9000200B7001006FF000201B70F920688940
S1136DA040160B7301006F7000200B7001006FF031
S1096DB000201B70F92D09

S1136DB668897A0600FFE89801006F70001C680C6A
S1136DC6AC2B4708AC2D4704AC20460E01006F7070
S1136DD6001C0B70010069E040466A2800FFE8804A
S1136DE6732847326F70003C0C004634A858470A94
S1136DF6A86F4714A8784702402601006F70001C4D
S1136E060BF0010069E0401801006F70001C0B7065
S1136E16010069E0400A01006F70001C010069E08F
S1136E267A0400FFE890010069401FB04F14010087
S1136E366946010069441AB401006BA400FFE8A483
S1136E46400C0FB61A8001006BA000FFE8A47A0479
S1136E5600FFE8AC01006F70001C680BAB2B470802
S1136E66AB2D4704AB20463801006B2000FFE88CAE
S1136E761FE04F2C6A2800FFE88AA830462201004B
S10D6E866B2000FFE88C1AE07A018C
S1136E9000FFE8A4010069120A82010069921A80C6
S1136EA0010069C0401E01006B2000FFE88C1FE059
S1136EB04F0C01006B2000FFE88C1AE040021A809F
S1136EC0010069C001006F7600141B76401A0100AF
S1136ED06F7000200B7001006FF000201B700FE13A
S1136EE01B760AD1681968890FE64CE201006F70BE
S1136EF00020189968897A17000000245E00604416
S1136F0054705E0060667A37000000647A0300FF05
S1136F10E8807A04000000040AF47A0500FFE89090
S1136F200F866FF1005001006FF6005CF83068C8FF
S1136F307A000000007C0AF07A010000A0185E00CD
S1136F4060360D00587000B87A000000002F0AF078
S1136F5001006DF07A000000002E0AF001006DF0D0
S1136F6001006F70008801006DF001006F700088F0
S1136F7001006DF07A01000000320AF17A0000008E
S1096F8000115E007F3EDC
S1136F867A17000000100D024752188868E80100BE
S1136F96695001006B2100FFE88C1F904F0601002A
S1136FA66950400801006B2000FFE88C01006BA0CC
S1136FB600FFE8AC7A0600FFE88A0D207920FFFF80
S1136FC6470C79200001460CF82B5A007CEEF82D6D
S1046FD65A5D
S1136FD7007CEEF82A68E85A007CF07A000000008B
S1136FE71101006DF00FC10B717A00000000260A32
S1136FF7F05E0083240B97400CF8016EF8002F18FE
S1137007886EC800017A000000000101006FF000DC
S113701760400E01006F7000600B7001006FF0009D
S11370276001006F7000600AC06808A83047E40178
S1137037006F7000607A20000000014F7A01006F33
S11370477000600AC05E0059AC0F8201006F7000C8
S1137057601B7001006F71002A0A8101006FF10044
S11370672A7A000000000101006FF0005840300148
S1137077006F7000600AC001006F7100580AC16891
S113708708689801006F7000580B7001006FF000DB
S11370975801006F7000600B7001006FF000600112
S11370A7006F7000580FA11F904FC401006F70004D
S11370B7580AC0189968890FC00B7001006FF00058
S10870C7445E0059AC1A
S11370CC01006FF000601A8001006FF0005801009E
S11370DC6F7000607A01000000025E007EF8010010
S11370EC6FF0004001006F70004401006F7100608D
S11370FC0A901B7001006FF0003C404E01006F7052
S113710C004401006F7100580A9001006FF0004CAD

S113711C68086EF8005701006F71003C01006F7234
S113712C00581AA101006FF1004801006F72004C66
S113713C681968A901006F710048689801006F71A4
S113714C00580B7101006FF1005801006F7000586B
S113715C01006F7100401F904DA201006F70006021
S113716C461AF8306EC800017A00000000010100D5
S113717C6FF000601A8001006FF0002A6F700050EE
S113718C79200067470679200047463C7D307050D4
S113719C01006F70002A01006F7100600A901B7070
S11371AC7A20FFFFFFFC4D0C01006B2100FFE890E0
S10971BC1F904F0C6F70E1
S11371C200501BD06FF000504008790000666FF04A
S11371D200506E78002FA80146246838E8186EF82C
S11371E20057A8084706A810470A401A0FE00B7673
S11371F2F92B40100FE00B76F920688940080FE065
S11372020B76F92D68896F700050792000665860FB
S1137212077201006F70002A58D001861A80401A43
S11372220FE00B7601006F7100580AC10B716819E8
S1137232688901006F7000580B7001006FF00058ED
S113724201006F7100601F904DD61A800F820100FA
S11372526BA600FFE89801006F70002A01006BA083
S113726200FFE8A47C307350460A01006B2000FF44
S1137272E8904E067C307320470E0FE00B76F92E12
S113728268890FA00B700F827C307350470C7C30DF
S1137292735047247C307320471E01006BA600FF06
S11372A2E89C0100695001006BA000FFE8A80100FF
S10972B269500FA10A81DF
S11372B80F9201006F70002A0FA10A9001006F71ED
S11372C800600A9001006FF0006001006F71005CBC
S11372D86819A92B4708A92D4704A920465201007C
S11372E86B2000FFE88C01006F7100601F904F4016
S11372F86A2800FFE88AA830463601006F70005CF0
S10973080B7001006BA0F5
S113730E00FFE89401006B2000FFE88C01006F7111
S113731E00601A9001006BA000FFE8A01A80010024
S113732E6BA000FFE8AC5A007CEC01006B2000FF61
S113733EE88C01006F7100601F904F4C01006B20B1
S113734E00FFE88C01006F7100601A9001006BA0C2
S113735E00FFE8AC6E78002FA80146186E78002F58
S113736EA801586009786E780057A8084706A81038
S113737E5860096A7A0000FFE8AC010069011B71CD
S113738E010069815A007CEC1A8001006BA000FF9A
S113739EE8AC5A007CEC01006F70006001006F7165
S11373AE002A0A8101006FF1002A010069520AA125
S11373BE01006FF100521F814C120F914D0E01000F
S11373CE6F7100520B710FC05C00092201006F70C8
S11373DE002A58F0028A6848A83046147A00000042
S11373EE000101006FF0004C01006FF00052402CC1
S10973FE1A8001006FF08C
S1137404004C1A8001006FF0005201006F70002AD3
S11374140B7001006FF0002A01006F7000600B70A5
S113742401006FF000601A8001006FF00058403AC9
S11374340FE00B7601006F71004C0AC1681968896B
S113744401006F70002A1B7001006FF0002A010015
S11374546F7000580B7001006FF0005801006F70DB
S1137464004C0B7001006FF0004C01006F70002A98
S11374744EBE0FE00B76F92E688940240FE00B769D

S113748401006F71004C0B7101006FF1004C1B7113
S11374940AC168196889010069501B70010069D029
S11374A401006F7000580B7001006FF00058010069
S11374B46F70004C01006F7100521A9001006F71DC
S11374C400601F904C0A01006B2000FFE8904EAC53
S11374D47C307350470C7C307350472E7C307320C0
S11374E447280100695001006F7100580A810100A7
S10774F46FF10058D9
S10774F801006BA67B
S11374FC00FFE8980100695001006BA000FFE8A4AD
S113750C40226E68FFFA830461A40101B7601001C
S113751C6F7000581B7001006FF000586E68FFF0E
S113752CA83047E87C307320464A6E68FFFA82ECC
S113753C464201006B2000FFE8A447067C307350E1
S113754C47321B7601006F70005801006B2100FF5E
S113755CE8A41A901B7001006FF000581A80010008
S113756C6BA000FFE8A401006F70005C01006BA02E
S113757C00FFE89801006F70005C6808A82B4708AF
S113758CA82D4704A820465001006B2000FFE88C6F
S113759C01006F7100581F904F3E6A2800FFE88A64
S11375ACA830463401006F70005C0B7001006BA0B7
S11375BC00FFE89401006B2000FFE88C01006F7161
S11375CC00581A9001006BA000FFE8A01A8001007C
S11375DC6BA000FFE8AC406201006B2000FFE88C5D
S10975EC01006F7100585D
S10B75F21F904F4601006B20BE
S11375FA00FFE88C01006F7100581A9001006BA01C
S113760A00FFE8AC6E78002FA80146146E78002FAD
S113761AA80146286E780057A8084704A810461CF4
S113762A7A0000FFE8AC010069011B71010069815E
S113763A400A1A8001006BA000FFE8AC01006B202E
S113764A00FFE89801006F71005C1F905860069272
S113765A01006B2000FFE89401006BA000FFE8988B
S113766A5A007CEC6848A83046147A0000000001EE
S113767A01006FF0004C01006FF00052402A1A809B
S113768A01006FF0004C01006FF0005201006F70AF
S113769A002A0B7001006FF0002A01006F7000606E
S11376AA0B7001006FF0006001006F70002A4F1425
S11376BA7A000000000101006FF0004C0FE00B7626
S11376CA684940060FE00B76F93068897A000000B2
S11376DA000101006FF000580FE10B76FA2E689A49
S10976EA01006F7000585F
S11376F00B7001006FF0005801006F70002A58C032
S1137700009E01006BA600FFE89801006F70002A3D
S113771017B0010069511F814F2001006F70002ACB
S113772017B001006BA000FFE8A401006F70002AEE
S113773001006F7100581A8140180100695001005F
S11377406BA000FFE8A40100695001006F710058AD
S11377500A8101006FF1005801006F70002A0100D7
S113776069510A81010069D140360FE00B760100AF
S11377706F71004C0AC168196889010069501B7058
S1137780010069D001006F7000580B7001006FF0A9
S1137790005801006F70004C0B7001006FF0004C3B
S11377A001006F70004C01006F7100521A900100CC
S11377B06F7100601F904C0A01006B2000FFE8907E
S11377C04EA87C307350470C7C30735047347C3068
S11377D07320472E010069504F4A01006BA600FF3A

S10977E0E89C0100695062
S10777E601006BA090
S11377EA00FFE8A80100695001006F7100580A817F
S11377FA01006FF1005840226E68FFFA830461A55
S113780A40101B7601006F7000581B7001006FF067
S113781A00586E68FFFA83047E87C307320467033
S113782A6E68FFFA82E466801006B2000FFE8A4DC
S113783A460A01006B2000FFE8A847067C30735014
S113784A474E1B7601006F70005801006B2100FF41
S113785AE8A41A9001006B2100FFE8A81A901B7094
S113786A01006FF000581A8001006BA000FFE8A81E
S113787A01006BA000FFE8A401006F70005C010027
S113788A6BA000FFE8981A8001006BA000FFE89C38
S113789A01006F70005C6808A82B4708A82D4704ED
S11378AAA820465001006B2000FFE88C01006F718D
S11378BA00581F904F3E6A2800FFE88AA8304634D2
S11378CA01006F70005C0B7001006BA000FFE8946D
S10778DA01006B201B
S11378DE00FFE88C01006F7100581A9001006BA035
S11378EE00FFE8A01A8001006BA000FFE8AC406225
S11378FE01006B2000FFE88C01006F7100581F9090
S113790E4F4601006B2000FFE88C01006F71005899
S113791E1A9001006BA000FFE8AC6E78002FA8014F
S113792E46146E78002FA80146286E780057A808D3
S113793E4704A810461C7A0000FFE8AC0100690159
S113794E1B7101006981400A1A8001006BA000FFC0
S113795EE8AC01006B2000FFE89801006F71005C3A
S113796E1F90461001006B2000FFE89401006BA0EE
S113797E00FFE8985A007CEC010069500B7001007F
S113798E6F7100601F904C0C010069510BF10FC019
S113799E5C00035A6848A830460E7A0000000001C6
S11379AE01006FF0004840241A8001006FF0004878
S11379BE01006F70002A1B7001006FF0002A010096
S11379CE6F7000600B7001006FF0006001006F704C
S10979DE006001006F715F
S11379E400481A9001006F72002A0A8201006FF2A4
S11379F4002A0FE00B760AC1681968897A0000002F
S1137A04000101006FF000580FE10B76FA2E689A1B
S1137A1401006F7000580B7001006FF000580100F3
S1137A246F700048402E0FE00B7601006F71004C1D
S1137A340AC168196889010069501B70010069D083
S1137A4401006F7000580B7001006FF000580100C3
S1137A546F70004C0B7001006FF0004C01006F71EC
S1137A6400601F904E0A01006B2000FFE8904EB6A1
S1137A747C307350470C7C307350472E7C3073201A
S1137A84472801006BA600FFE89801006950010034
S1057A946BA0E2
S1137A9600FFE8A40100695001006F7100580A81D4
S1137AA601006FF1005840226E68FFFA830461AA6
S1137AB640101B7601006F7000581B7001006FF0B9
S1137AC600586E68FFFA83047E87C307320464AAB
S1137AD66E68FFFA82E464201006B2000FFE8A454
S1137AE647067C30735047321B7601006F7000588F
S1137AF601006B2100FFE8A41A901B7001006FF0D0
S1137B06005801006F70005C01006BA000FFE8984D
S1137B161A8001006BA000FFE8A46F700050792063
S1137B26006546080FE00B76F96540060FE00B7615

S1137B36F945688901006F7000580B7001006FF0FA
S1137B46005801006F70002A4D0A0FE00B76F92BDF
S1137B56688940160FE00B76F92D688901006F706E
S1137B66002A17B001006FF0002A01006F70005859
S1137B760B7001006FF0005801006F70002A7A2025
S1097B86000000644D0E37
S1137B8C0BF601006F7000580B800B7040140FE064
S1137B9C0B76F9306889F83068E801006F7000588B
S1137BAC0BF001006FF000580FE00B7001006FF049
S1137BBC003040360FE01B76010069F001006F7056
S1137BCC002A7A010000000A5E007EF88930010069
S1137BDC6970688901006F70002A7A010000000A3D
S1137BEC5E007EF801006FF0002A01006F70002A1E
S1137BFC4EC201006F76003001006F70005C6808A4
S1137C0CA82B4708A82D4704A820465001006B2039
S1137C1C00FFE88C01006F7100581F904F3E6A28DB
S1137C2C00FFE88AA830463401006F70005C0B70CB
S1137C3C01006BA000FFE89401006B2000FFE88CAF
S1137C4C01006F7100581A9001006BA000FFE8A0AF
S1137C5C1A8001006BA000FFE8AC406201006B20AE
S1137C6C00FFE88C01006F7100581F904F46010014
S1057C7C6B2078
S1137C7E00FFE88C01006F7100581A9001006BA091
S1137C8E00FFE8AC6E78002FA80146146E78002F23
S1137C9EA80146286E780057A8084704A810461C6A
S1137CAE7A0000FFE8AC010069011B7101006981D4
S1137CBE400A1A8001006BA000FFE8AC01006B20A4
S1137CCE00FFE89801006F71005C1F9046100100E1
S1137CDE6B2000FFE89401006BA000FFE898188862
S1137CEE68E87A17000000645E00604454705E001A
S1137CFE60660F850F960FD00AE06808A8344F52BE
S1137D0E0FD00AE0F93068891B760FD00AE06809B4
S1137D1E8901688940200FD00AE06808A8394F14FA
S1137D2E0FD10AE1681888F668986E18FFFF88016C
S1137D3E6E98FFFF1B760FE64EDC6E580001A839D6
S1137D4E4F106E58000188F66ED8000168588801EE
S1137D5E68D85E00604454705E0060661B977A03B9
S1097D6E00FFE8827A0623
S1137D7400FFE886010069F00F9501006B2100FF05
S1137D84E87C7A2100000001462601006DF50F818D
S1137D94010069305E0084C80B97010069300AD082
S1137DA4010069B0010069600AD0010069E0403A4A
S1137DB419444030010069700B70010069F01B70B5
S10F7DC4680817D00100693101006B2230
S1137DD000FFE8B45D207920FFFF471201006960CE
S1137DE00B70010069E00B5417F41FD44DCA0B97B5
S1137DF05E006044547001006B2000FFE87C7A2031
S1137E0000000001460C01006B2000FFE882189976
S1137E106889547001006B2000FFE87C7A20000021
S1137E20000146041900547001006B2000FFE88232
S10F7E306E090010E94017D10D105470CA
S10BA018000000000000000003D
S1137E3C01006DF501006DF401006DF301006DF2AD
S1137E4C01006DF101006D120100691301006F0156
S1137E5C0004010069000D840D8C796C7FF0792C82
S1137E6C7FF047540DAC796C7FF0792C7FF047563B
S1137E7C0D8C65AC4B5E1AB144087A3000000001DE

S1137E8C45141AA0451001F06410475A0C444B0AD0
S1137E9C79000002400C0C444BF6190040047900A5
S1137EACFFFF01006D7101006D7201006D73010024
S1137EBC6D7401006D7554700F85796D000F01F0B1
S1137ECC641546DA409E0FA5796D000F01F06435F9
S1137EDC46CC409C01F064207A607FFFFFFF46ACE8
S10F7EEC01F0643146A67900000140B6A5
S1137EF86DF20D820C2A4A0217B00D994A04D280FA
S1137F0817B15E0085220C224A0217B00CAA4A0256
S1097F1817B16D725470F5
S1137F1E01006DF301006DF20D8252120D935203A7
S1137F2E52100928093801006D7201006D735470E7
S113A0202020202020202020202060606060602020ED
S113A030202020202020202020202020202020201D
S113A04048101010101010101010101010101010D5
S113A0508484848484848484848484848484848101010101075
S113A06010818181818181810101010101010101CE
S113A07001010101010101010101010101010101082
S113A080108282828282828202020202020202029F
S113A09002020202020202020202020202020202047
S113A0A00000000000000000000000000000000AD
S113A0B000000000000000000000000000000009D
S113A0C000000000000000000000000000000008D
S113A0D000000000000000000000000000000007D
S113A0E000000000000000000000000000000006D
S113A0F000000000000000000000000000000005D
S113A10000000000000000000000000000000004C
S109A11000000000000046
S10DA1160000000000000000000003C
S1137F3E5E0060667A370000003C7A040000000C95
S1137F4E0AF47A05000000140AF501006FF00038F8
S1137F5E01006FF1003401006F73005C7A020000C0
S1137F6E000801006FF2002C19226FF2002A7A0624
S1137F7E000000540AF6686AEA7F46620FE00B704F
S1137F8E7A01000000075E00855E0D0047501A80DF
S1137F9E401A01006F70003401006F7100380A90AF
S1137FAE1899688901006F7000380B7001006FF02B
S1137FBE00387A20000000084DD81A80010069B0FD
S1137FCE7C607370470A01006F700060F9FF400810
S1137FDE01006F700060F90168895A0083167A00F8
S1137FEE0000001C0AF001006DF07A010000002869
S1137FFE0AF101006F70005C01006DF001006F70FB
S113800E005C01006DF001006F70006C5E0088EE85
S113801E7A170000000C01006F7000606808A8FF5B
S107802E460C7A007F
S1138032000000540AF07D0072706F7000247920F2
S113804207FF464A6E680001A8F0463A0FE00BF0BC
S11380527A01000000065E00855E0D0047280100DC
S11380626F7000606808A8014608790000015A0091
S1138072831801006F7000606808A8FF461079003A
S1138082FFFF5A008318790000025A0083186F70A9
S11380920024793003FF6FF000244D1E01006F703E
S11380A2005801006DF001006F70005801006DF07F
S11380B25E00867E0B970B97401E01006F7000587F
S11380C201006DF001006F70005801006DF05E0059
S11380D2867E0B970B971B500D0618886EF80023AC
S11380E20B560D6017F001006F7100381A900100F2

S11380F269B0010069F10FD1010069705E00879ACE
S113810201006F70003801006F71002C5E007EF871
S113811201006FF000301AE67A0000000008010047
S10981226F7100301A909A
S11381280F82401401006F7000300AE00AD00FD1AB
S11381380AE1680868980B760FA01F864DE6400A87
S11381480FD00AE0189968890B767A260000000890
S11381584DEE6F7000326F78002E52806F71003AC7
S113816819016DF17A01000000080FD05E00872223
S11381780B8701006F7600381B760FC10FE05E0096
S1138188879A0FE001006F71002C5E007EF80100F2
S11381986FF000301AE67A000000000801006F71E2
S11381A800301A900F82401401006F7000300AE00B
S11381B80AC00FC10AE1680868980B760FA01F86EA
S11381C84DE6400A0FC00AE0189968890B767A26AB
S11381D8000000084DEE6F7000326F78002E528059
S11381E86F71003A19011B516DF17A010000000803
S11381F80FC05E0087220B8701006F7000340100F7
S11382086DF00100693101006DF17A01000000246D
S10982180AF16F70002C57
S113821E5E0085860B970B977A06000000040AF61C
S113822E7A000000000801006DF001006F71003844
S113823E0FE05E0084C80B977A000000000801006F
S10A824E6DF00FD10FE05E9C
S11382550089920B970D004F12790000016FF00012
S11382652A010069300B705A00830E7A0000000062
S11382750801006DF001006F7100380FE05E0084A6
S1138285C80B977A000000000801006DF00FD10FAD
S1138295E05E0089920B970D004632010069300BB1
S11382A570010069B001006F70003401006DF001C9
S11382B500693301006DF37A01000000240AF16FB0
S11382C570002C5E0085860B970B9740447A00005F
S11382D500000801006DF001006F7100380FE05ECA
S11382E50084C80B977A000000000801006DF00FA9
S11382F5C10FE05E0089920B970D004C146F70005F
S11383052A460E010069301B70010069B05A0082CC
S11283150019007A170000003C5E0060445470AA
S11383245E0060667A370000002C0FF30F860100AD
S11383346FF100107A000000000101006FF00018D3
S113834401006F7500440A95188868D87A00000004
S1138354000801006DF00FE17A000000000C0AF040
S11383645E0084C80B9701006F7000445A0084B404
S11383740FA00B707A01000000055E007EF80B70FD
S113838410307A06000000081A867A0000000008FC
S11383941AE001006DF00FA11031103110317A1180
S11383A40000A1200AE10FB00AE05E0084C80B9725
S11383B47A000000000801006FF000281A80010011
S11383C46FF0002001006FF0001C7A040000000825
S11383D41AE401006FF400145A00847601006F70EC
S11383E4001401006DF00FB10AE17A000000000CE3
S11383F40AF00AE05E0089920B970D004D3C0100E0
S11384046DF40FB00AE001006DF07A010000001072
S10984140AF10AE11A80DF
S113841A5E0088420B970B9717F001006FF0001864
S113842A01006F70002801006F7100200A810100AA
S113843A6FF1002001006F7000287A01000000022A
S113844A5E007EF801006FF00028473079000001D2

S113845A6DF00FB00AE00FC15E0087C80B870100F9
S113846A6F70001C0B7001006FF0001C01006F702D
S113847A001C7A200000000458D0FF5A01006F70D4
S113848A0018461C6E78002388306CD84004F830F4
S113849A68D81B7501006F7000101F8544F04012E5
S11384AA6E78002388306CD80FA01B700F8258C0D7
S11184BAFEB87A170000002C5E006044547078
S113A12000000000000008000000000000050D4
S113A130000000000000320000000000001F409A
S113A14000000000000138800000000000C350012
S113A15000000000007A12000000000004C4B400F4
S113A16000000002FAF08000000001DCD6500003
S113A1700000012A05F200000000BA43B74000B7
S113A18000000746A5288000000048C27395000020
S113A1900002D79883D20000001C6BF52634000020
S10BA1A0011C37937E08000047
S11384C85E0060660F850F9401006F7600181FC564
S11384D847401FC544200FD30FC21AC440120FB020
S11384E80B730FA10B710F921B71681968890B74B9
S11384F81FE445EA401C0FD30AE30AE40FC21AC477
S1138508400C0FA01B700F8268086CB80B741FE433
S10D851845F00FD05E00604454707C
S113852201006DF20D9946100D82177253120DA8B8
S113853253100D810D28401E0F920D81177179087A
S11385420010121012311AA144020AA11B5846F25A
S10F855212101710177001006D725470A6
S113855E5E0060660F840F951AE6400E0FC00AE0A8
S113856E680947041900400A0B761FD64DEE7900B1
S10B857E00015E00604454702B
S11385865E0060667A370000000C7A050000000181
S11385960AF50D0C0F967904000801006DF57A01B2
S11385A60000000E0AF101006F70002817B05E008C
S11385B68A580B9701006DF66DFC7A000000010D7
S11385C60AF00FD15E0089D00B970B87790C003F19
S11385D66F70000A190C0DC017F001D053400D0E31
S11385E67900004219C017F001006DF07A0100000E
S11385F600090FD05E008CA40B977906000840147F
S11386060D6019E017F00AD00D6117F10AD1680859
S113861668981B561DE64CE8400C0D6017F00AD00F
S1138626189968891B560D664CF00DE05240190CDB
S11386366DFC7A0100000090FD05E0087C80B8726
S11386467900003F6FF0000A7A01000000400FD066
S11386565E008BBA7A00000000801006DF00FD1AE
S113866601006F70002C5E0084C80B977A17000018
S1098676000C5E006044ED
S105867C547035
S113867E01006DF27A370000001A7A00000000182C
S113868E0AF001006F71002601006DF101006F7198
S113869E002601006DF17A01000000080AF10100C5
S11386AE6DF15E008D407A170000000C6F7100189B
S11386BE0FF05E008FC00F817A020000A1A87A002E
S11386CE000000080AF05E0090AC7A000000001073
S11386DE0AF001006F71000C01006DF101006F7162
S11386EE000C01006DF17A01000000080AF101008F
S11386FE6DF15E008E467A170000000C7A000000C2
S113870E00100AF05E005F047A170000001A0100E1
S107871E6D725470B1

S10BA1A83FD34395810624DD3A
S11387225E0060661B971B87010069F00F946F79E7
S1138732001E790D0008199D4754790100FF0DD2DF
S11387421A0A4B04101140F80C9B18DD1B740FC658
S11387524028010069740AE468450CB816850FC005
S11387620D915E008F7C684814D868C80DD01A0832
S11387724B04110540F80C5D1B760FE64CD40CDD5F
S11387824706790100014002191117F10F900B8777
S10B87920B975E006044547074
S113879A5E0060660F860F957A00000000080100EC
S11387AA6DF01036103610367A010000A1B00AE1D6
S11187BA0FD05E0084C80B975E0060445470BD
S113A1B0000000000000000100000000000000596
S113A1C00000000000000001900000000000007DF6
S113A1D000000000000000271000000000000C35C8
S113A1E000000000000003D09000000000001312DC7
S113A1F00000000000005F5E100000000001DCD6532
S113A20000000000009502F90000000002E90EDDE5
S113A210000000000E8D4A510000000048C27395F3
S113A220000000016BCC41E9000000071AFD498DD5
S113A2300000002386F26FC1000000B1A2BC2EC54E
S113A2400000003782DACE9D900001158E460913D7A
S113A250000056BC75E2D6310001B1AE4D6E2EF54D
S113A260000878678326EAC9002A5A058FC295ED4C
S113A27000D3C21BCECCEDA10422CA8B0A00A425B5
S113A28014ADF4B7320334B96765C793FA10079D69
S11387C85E0060667A370000000A0F83010069F1D2
S11387D86F790022790C0008199C4752FAFF0DC0E3
S11387E81A084B04110A40F80CA218CC1AE64026C2
S11387F80FB50AE568540C2816840FD00D915E0056
S11388088F9E685814C868D80DC01A084B04100402
S113881840F80C4C0B76010069701F864DD20CCCC6
S11388284706790100014002191117F10F907A17D1
S10D88380000000A5E006044547063
S11388425E0060667A370000000C01006FF00004DE
S11388520F9601006F7500280AD61B7601006F730D
S113886200240AD31B737A02000000011AC4404495
S11388726868175068391751191017F01AC00100A8
S113888269F04C14010069717A11000001000100C2
S113889269F1790000014002190017F00F84010009
S11388A2697047041A800F826E78000368E81B75AB
S11388B21B761B730FD546B87A2400000001460EBF
S11388C201006F70000446067900FFFF40120FA0FB
S11388D27A200000000146041900400479000001D7
S10F88E27A170000000C5E006044547024
S11388EE5E0060667A03000000070F820F95010099
S11388FE6F7600207A04000000180AF4694179614A
S113890E80007921800046067901FFFF400479013A
S113891E00010FA06889694079607FF01190119072
S113892E1190119069D07A000000000701006DF0DC
S113893E0B740FC10FE05E0084C80B977900000320
S113894E6DF00FB10FE05E0087220B8769504F0663
S113895E7D607070402869500B5069D07D607270D5
S113896E4016790000016DF00FB10FE05E00872213
S113897E0B8769501B5069D07C60737047E45E00AF
S107898E604454707A
S11389925E0060660F860F9301006F7400180FE587

S11389A20FC44604190040201AE6400A6C586C3979
S11389B21C9846060B761FC645F21B75685817505E
S11189C21B73683B175319305E0060445470FA
S11389D05E0060667A37000000100FF60F850F9473
S11389E069506F710028091069D001006DF601000C
S11389F06F71002E0FC05E0093880B977C607370BD
S1138A00470869500B5069D0400C7A0100000010F0
S1138A100FE05E0093367A000000004201006DF023
S1138A207A01000000100FE05E008CA40B976E68C3
S1138A300008E8E06EE800087A000000009010081
S1138A406DF00FE10FC05E0084C80B977A1700002A
S10B8A5000105E006044547045
S1138A585E0060667A37000000187A050000000996
S1138A680AF50F840F920FC44D087A030000000122
S1138A7840147A03FFFFFFF0FC07A01FFFFFFFD8
S1138A885E007F1E0F840FC07A010000001B5E008A
S1138A987EF80F860FC07A010000001B5E007EF887
S1138AA80F947A2300000001462E7A000000000884
S1138AB801006DF00FE11031103110317A1100000F
S1138AC8A2900FD05E0084C80B970FE01030780097
S1138AD86B210000A36040320FC446021B767A0064
S1138AE80000000801006DF00FE110311031103162
S1138AF87A110000A2F80FD05E0084C80B970FE02C
S1138B08103078006B210000A37A6FF100120FC4B4
S1138B1847747A23FFFFFFF460A7A000000001B11
S1138B281AC00F8401006F70003001006DF00FA1AF
S1138B380FC05E0095160B970FE646087A230000D0
S1078B48000147627C
S1138B4C01006F70003001006DF00FA069006DF033
S1138B5C7A00000000180AF00FD15E0089D00B9741
S1138B6C0B877A01000000400FD05E008BBA79208E
S1138B7C0001460E6F7000120B506FF000127D5007
S1138B8C70700FA06F71001269817A0000000008E9
S1138B9C01006DF00FD101006F7000345E0084C8CA
S1118BAC0B977A17000000185E0060445470A7
S113A29080000000000000000CECB8F27F4200F3A8F
S113A2A0A70C3C40A64E6C5286F0AC99B4E8DAFD9C
S113A2B0DA01EE641A708DEAB01AE745B101E9E4F8
S113A2C08E41ADE9FBEBEC27DE5D3EF282A242E8135
S113A2D0B9A74A0637CE2EE195F83D0A1FB69CD999
S113A2E0F24A01A73CF2DCD0C3B8358109E84F0735
S113A2F09E19DB92B4E31BA99E74D1B791E07E480B
S113A300C428D05AA4751E4CF2D56790AB41C2A4A1
S113A310964E858C91BA2655BA121A4650E4DDEC56
S113A320E65829B3046B0AFA8E938662882AF53FAE
S113A330B080392CC4349DEDDA7F5BF59096684983
S113A340873E4F75E2224E68A76C582338ED2623CB
S113A350CF42894A5DCE35EB8049A4AC0C5811AE8F
S113A3600000005900B3010D016601C0021A027317
S113A37002CD0327038003DA0434FFA6FF4CFEF269
S109A380FE99FE3FFDE51E
S111A386FD8CFD32FCD8FC7FFC25FBCBFB726B
S1138BBA5E0060667A370000000C0F840F957A0610
S1138BCA000000081AB30FD00FE15E007EF80AC056
S1138BDA0F820FD00FE15E007EF8790000801A0938
S1138BEA4B04119040F86EF8000511086EF8000B5B
S1138BFA11086EF800041B75010069F50FD00FE127

S1138C0A5E007EF80AC00F85010069700FE15E00FD
S1138C1A7EF8790600801A094B04119640F80FA0D2
S1138C2A68086E790005169847280FA068066E78BB
S1138C3A000B166846086E780004166847087A031C
S1138C4A0000001400C685816E847067A03000042
S1138C5A00017A2300000001463240140CE8175041
S1138C6A17106859168968D9100E46041B75FE0138
S1138C7A685816E847041FC544E21FC545086858E3
S1138C8A14E868D8400679000001400219007A17EF
S10D8C9A0000000C5E0060445470FB
S1138CA45E0060661B971B970F82010069F17A03CC
S1138CB4000000801006F7000200FB15E007EF811
S1138CC40FA60A8601006F7000200FB15E007EF8C4
S1138CD4790400801A094B04119440F8686816C893
S1138CE446500CCD1855400414D5110D0CDD46F82F
S1138CF4686816584708686814C868E840340FA0C1
S1138D04010069710A90010069F001006F7000208D
S1138D140FB15E007EF80FA50A85400C686847080A
S1138D24685814C868D8400A0B76010069701F8616
S10F8D3445EA0B970B975E0060445470F7
S1138D405E0060660F857A040000000701006F7003
S1138D50002001006DF001006F70002001006DF034
S1138D605E0092E40B970B970D06474879260001A6
S1138D70460A790004B86BA000FFE8BC792600021C
S1138D80460A7900044C6BA000FFE8BC19667A0020
S1138D900000001C0AF00D6117F10A90F9FF6889C1
S1138DA00B56792600084DE67A000000001C0AF0F5
S1138DB05A008E367A000000001C0AF07A01000087
S1138DC0A3945E005F7E0D00470C190069D07A0002
S1138DD00000A39440607A060000001C0AF6696351
S1138DE01113111311131113796307FF793303FE61
S1138DF069D36950791003FE462869500B5069D036
S1138E0069607960800F69E00FE30B73400E0FC157
S1138E100FB05E00933669501B5069D06960734888
S1138E2047EC69607960800F79403FE069E07A0040
S1098E300000001C0AF023
S1138E3601006F7100185E005FF05E0060445470BD
S10BA39400000000000000000BE
S1138E465E0060660F8601006F70002001006DF002
S1138E5601006F70002001006DF05E0092E40B9735
S1138E660B970D05474079250001460A790004B89A
S1138E766BA000FFE8BC79250002460A7900044C82
S1138E866BA000FFE8BC19667A000000001C0AF01C
S1138E960D6117F10A90F9FF68890B5679260008C8
S1138EA64DE65A008F647A050000001C0AF56955E1
S1138EB61115111511151115796507FF793503FF7D
S1138EC60D5C4C0E7A000000A39C0FE15E005FF080
S1138ED640607A000000001C0AF00FE15E005FF0BC
S1138EE6792C00344C4C0FE50B950BF579090034BE
S1138EF619C90D9017F07901001001D053100D090F
S1138F06400A0FD01BF5191169811B590D994EF2B1
S1138F167900003419C017F07901001001D05310FD
S1138F267909FFFF1B584B04101940F869506690E6
S1078F3669D07A0180
S1138F3A0000001C0AF10FE20F905E005A920FE044
S1138F4A7A010000A39C5E005F9A0D00470C7A0029
S1138F5A0000001C0AF07D0070707A000000001CFB

S1138F6A0AF001006F7100185E005FF05E00604452
S1058F7A54702E
S10BA39C00000000000000000B6
S1138F7C01006DF20D1A4F14792A00084D04FA0002
S1138F8C4008680A100A1B5A4EFA688A01006D726F
S1058F9C54700C
S1138F9E01006DF20D1A4F14792A00084D04FA00E0
S1138FAE4008680A110A1B5A4EFA688A01006D724C
S1058FBE5470EA
S1138FC001006DF119990D11471A4A067909080034
S1138FD017917919040F1B59101144FA10311031EC
S1138FE010311031010069811A9101006F81000471
S1098FF001006D715470D5
S1138FF60F8501F064155860009A792407FF47141A
S11390060D44586000820FA501F06435586000785E
S1139016580000800FA501F064355860007640685B
S11390260FA501F06435466C0DCC465C0F8501F047
S113903664154654405E0F8501F06415473C1031B4
S11390461230792800104C0C1B5C10311230792831
S113905600104DF4580000C40FA501F06435471AFB
S113906610331232792A00104C0C1B54103312326F
S1139076792A00104DF4580000A81AA21AB3687D85
S1139086100D131A58000228790107FF1AA21AB302
S1139096580001FA790107FF1AA27A0300000008B3
S11390A668FA580001E801006DF601006DF501004C
S11390B66DF401006DF301006DF201006DF1010025
S11390C66DF07A3700000018691D692565D569F5C5
S11390D6691C111C111C111C111C796C07FF6924D6
S10990E611141114111412
S11390EC1114796407FF01006D100100691179688F
S11390FC000F01006F23000401006922796A000F3D
S113910C792C07FF5870FEE2792407FF5870FF0A89
S113911C0DCC5870FF1A0D445870FF36094C793C2E
S113912C03FF792C07FF58C0FF58792CFFCB58D07D
S113913CFF426FFC000279480010794A00100100CD
S113914C6FF000040F9601006FF2000801006FF33B
S113915C000C0D1552356FF500160D34529452B1A7
S113916C0A946511990009D44406791C00019900ED
S113917C6FF400140DC50D1D18990D0452340AC556
S113918C99000DE452B40AC599000D6452240AC522
S113919C99006FF500120DD50D1D18990D845234DD
S11391AC0AC50D0452B40AC599000DE452240AC52C
S11391BC99000D6452A40AC599006FF500100DD5E2
S11391CC0D1D18990DB352830AB50D0452240AC50B
S10991DC99000DE452A40A
S11391E20AC59900528209D24404791A0001091A64
S11391F26F74000852040AC26F7400085284094A49
S11392020D5B6F73001001006F7400126F7D001607
S113921219556F71000211321333133413350DA034
S113922279600100471211321333133413350B5192
S1139232792107FF5870FE5401F064454702700B11
S11392420D114E2E113213334402700B0D114A04C9
S11392520B5140F0732B47180CB8E80B47127A13E3
S113926200000008440A0B72792A00804D020B5158
S11392724020732B471C0CB8E80B47167A130000E7
S1139282000844020B72792A01004D06113213338E
S11392920B51113213331132133311321333796AEF

S11392A2000F1011101110111011641A101A68789E
S11392B21008131A7A170000001801006D700100DC
S11392C2698201006F83000401006D7101006D72F8
S10792D201006D73B4
S11192D601006D7401006D7501006D7654701A
S11392E401006DF57A01000000080AF16818E87FAF
S11392F4A87F460A0B716818E8F0A8F04704190020
S1139304402A6818F01050006898460A0B711AD561
S1139314400E681847067900000140100B710B7565
S11393247A25000000064DEA7900000201006D75FC
S1059334547070
S11393365E0060660F840F931AD51B730FB6402821
S11393460FC30AE30FB26838E880175017700F830C
S11393560FA06809100968890FD547080FC00AE0EE
S11393667D0070000FB51B760FE64CD40FD547066C
S1139376790100014002191117F10F905E00604454
S105938654701E
S11393885E0060667A37000000387A050000000442
S11393980AF50F840F967A000000001001006DF0A3
S11393A8191101006F7000545E0095FA0B970FE3D3
S11393B80B930BF37A160000000701006FF60034D5
S11393C8790600030FC00B900BF001006FF0002823
S11393D87A140000000701006FF4002C5A0095046A
S11393E86838460C01006F7000346809587000FA39
S11393F87A000000001001006DF019110FD05E0013
S113940895FA0B9701006F70002801006FF0003088
S113941801006F74002C790E0003407C01006F700B
S113942800301A916809FA0810311A0A4EFA1A809C
S1139438684801F064101A916839FA0810311A0A59
S11394484EFA01006F720034010069F01A8068282F
S113945801F06401010069705E007F1E01006FF076
S113946800247A000000000401006DF07A01000076
S109947800280AF10DE2D9
S113947E096217F210320AD20FA05E0095B20B9753
S113948E1B5E01006F7000301BF001006FF00030A7
S113949E1BF40DEE4C807926000346247A0000005F
S11394AE000A01006DF00D6417F40FC110310AD1DB
S11394BE01006F7000540AC00AC05E0084C84022C7
S11394CE7A000000000A01006DF00D6417F40FC15D
S11394DE10310AD101006F7000540AC00AC05E0039
S11394EE95B20B971B561BF301006F7000341BF0E4
S11394FE01006FF000340D6658C0FEDE7A170000CF
S10B950E00385E006044547054
S11395165E0060667A370000000C0F860F940D60BC
S11395267910003F69C00FF10FE05E00879A7A0059
S11395360000000801006DF0191101006F7000288A
S11395465E0095FA0B971A800F8219550FF6401095
S11395560B760FA00B700F8269407930000869C043
S1139566686847ECFB804004110B0B55686816B816
S113957647F66DF57A03000000080FA01A830FB1B2
S11395860FE05E0087220B876940195069C001000E
S11395966DF30FE101006F7000285E0084C80B971E
S10F95A67A170000000C5E006044547053
S11395B25E0060660F860F9401006F7500180AD66D
S11395C21B760AD41B740FC319CC40226868175048
S11395D268391751091009C00D0468E817F47900B6
S11395E2010001D053040D4C1B751B761B730FD561

S10B95F246DA5E006044547088
S11395FA5E0060661B870F8469F101006F73001AAE
S113960A0FC51AE6400C0FD00B756E7900016889F5
S113961A0B761FB645F00FC00B875E00604454708B
S9030000FD

LINK COMMAND LINE

LNK -subcommand=linkfile.sub

LINK SUBCOMMANDS

OUTPUT c_thread
 PRINT c_thread
 INPUT asmfile, main, EV_Simulator, EV_Puls, EV_Controller, EV_Input, EV_Display, EV_OpenClose,
 EV_UpDown, EV_File, EV_Time, Timer, Panel, sci, lcd, usb
 LIB c:\h8\akic\c38hab
 START R(0FFE000), P(200), D(99C0), C(9A00)
 ROM (D, R)
 EXIT

*** LINKAGE EDITOR LINK MAP LIST ***

SECTION NAME	START	END	LENGTH
	UNIT NAME		MODULE NAME

ATTRIBUTE : CODE NOSHR

V	H'00000000	- H'000000F3	H'000000F4
		asmfile	asmfile
* TOTAL ADDRESS *	H'00000000	- H'000000F3	H'000000F4

ATTRIBUTE : CODE NOSHR

P	H'00000200	- H'000002AF	H'000000B0
		asmfile	asmfile
	H'000002B0	- H'000012BD	H'0000100E
		main	main
	H'000012BE	- H'0000177D	H'000004C0
		EV_Simulator	EV_Simulator
	H'0000177E	- H'00001AF9	H'0000037C
		EV_Puls	EV_Puls
	H'00001AFA	- H'00002475	H'0000097C
		EV_Controller	EV_Controller
	H'00002476	- H'00002DB3	H'0000093E
		EV_Input	EV_Input
	H'00002DB4	- H'00002E05	H'00000052
		EV_Display	EV_Display

```

H'00002E06 - H'0000366D H'00000868
              EV_OpenClose          EV_OpenClose
H'0000366E - H'00003EB5 H'00000848
              EV_UpDown             EV_UpDown
H'00003EB6 - H'000042A9 H'000003F4
              EV_File               EV_File
H'000042AA - H'00004401 H'00000158
              EV_Time               EV_Time
H'00004402 - H'00004975 H'00000574
              Timer                 Timer
H'00004976 - H'00004ACB H'00000156
              Panel                 Panel
H'00004ACC - H'00004B9D H'000000D2
              sci                   sci
H'00004B9E - H'00004DDB H'0000023E
              lcd                   lcd
H'00004DDC - H'000058CD H'00000AF2
              usb                   usb
H'000058CE - H'00005907 H'0000003A
              rand                  rand
H'00005908 - H'00005965 H'0000005E
              sprintf               sprintf
H'00005966 - H'0000598F H'0000002A
              strcmp                strcmp

```

H8/300H LINKAGE EDITOR (Evaluation software) Ver.1.0 PAGE : 2

*** LINKAGE EDITOR LINK MAP LIST ***

SECTION NAME	START	END	LENGTH
	UNIT NAME	MODULE NAME	

ATTRIBUTE : CODE NOSHR

P	H'00005990	- H'000059AB	H'0000001C
		strcpy	strcpy
	H'000059AC	- H'000059C7	H'0000001C
		strlen	strlen
	H'000059C8	- H'000059F7	H'00000030
		vsprintf	vsprintf
	H'000059F8	- H'00005CD1	H'000002DA
		add3	add3
	H'00005CD2	- H'00005F03	H'00000232
		divd3	divd3
	H'00005F04	- H'00005F7D	H'0000007A
		dtoa3	dtoa3
	H'00005F7E	- H'00005F89	H'0000000C
		eqd3	eqd3
	H'00005F8A	- H'00005F99	H'00000010
		ged3	ged3
	H'00005F9A	- H'00005FA9	H'00000010
		ltd3	ltd3
	H'00005FAA	- H'00005FEF	H'00000046

H'00005FF0	-	ltod3 H'0000600D	ltod3 H'0000001E
		mv83	mv83
H'0000600E	-	H'00006035	H'00000028
		mvn3	mvn3
H'00006036	-	H'00006043	H'0000000E
		ned3	ned3
H'00006044	-	H'00006065	H'00000022
		spregld3	spregld3
H'00006066	-	H'0000608D	H'00000028
		spregsv3	spregsv3
H'0000608E	-	H'00007E3B	H'00001DAE
		_fmtout	_fmtout
H'00007E3C	-	H'00007EF7	H'000000BC
		cmpd3	cmpd3
H'00007EF8	-	H'00007F1D	H'00000026
		divl3	divl3
H'00007F1E	-	H'00007F3D	H'00000020
		mull3	mull3
H'00007F3E	-	H'00008323	H'000003E6
		_dti	_dti
H'00008324	-	H'000084C7	H'000001A4
		_its	_its
H'000084C8	-	H'00008521	H'0000005A
		memcpy	memcpy
H'00008522	-	H'0000855D	H'0000003C
		divul3	divul3

H8/300H LINKAGE EDITOR (Evaluation software) Ver.1.0 PAGE : 3

*** LINKAGE EDITOR LINK MAP LIST ***

SECTION NAME	START	-	END	LENGTH	MODULE NAME
	UNIT NAME				

ATTRIBUTE : CODE NOSHR

P	H'0000855E	-	H'00008585	H'00000028	
			_allzero	_allzero	
	H'00008586	-	H'0000867D	H'000000F8	
			_calcnpw	_calcnpw	
	H'0000867E	-	H'00008721	H'000000A4	
			_log10	_log10	
	H'00008722	-	H'00008799	H'00000078	
			_lsfts	_lsfts	
	H'0000879A	-	H'000087C7	H'0000002E	
			_pow5	_pow5	
	H'000087C8	-	H'00008841	H'0000007A	
			_rsfts	_rsfts	
	H'00008842	-	H'000088ED	H'000000AC	
			_sub	_sub	
	H'000088EE	-	H'00008991	H'000000A4	
			_unpack	_unpack	

```

H'00008992 - H'000089CF H'0000003E
             memcmp             memcmp
H'000089D0 - H'00008A57 H'00000088
             _mult64           _mult64
H'00008A58 - H'00008BB9 H'00000162
             _power            _power
H'00008BBA - H'00008CA3 H'000000EA
             _rnd              _rnd
H'00008CA4 - H'00008D3F H'0000009C
             _setsbit          _setsbit
H'00008D40 - H'00008E45 H'00000106
             frexp             frexp
H'00008E46 - H'00008F7B H'00000136
             modf              modf
H'00008F7C - H'00008F9D H'00000022
             dslc3             dslc3
H'00008F9E - H'00008FBF H'00000022
             dsruc3            dsruc3
H'00008FC0 - H'00008FF5 H'00000036
             itod3             itod3
H'00008FF6 - H'000092E3 H'000002EE
             muld3             muld3
H'000092E4 - H'00009335 H'00000052
             _duchek           _duchek
H'00009336 - H'00009387 H'00000052
             _lsft             _lsft
H'00009388 - H'00009515 H'0000018E
             _mult             _mult
H'00009516 - H'000095B1 H'0000009C
             _pow10            _pow10

```

H8/300H LINKAGE EDITOR (Evaluation software) Ver.1.0 PAGE : 4

*** LINKAGE EDITOR LINK MAP LIST ***

SECTION NAME	START	END	LENGTH
	UNIT NAME	MODULE NAME	

ATTRIBUTE : CODE NOSHR

P	H'000095B2	- H'000095F9	H'00000048
		_add	_add
	H'000095FA	- H'00009629	H'00000030
		memset	memset

* TOTAL ADDRESS * H'00000200 - H'00009629 H'0000942A

ATTRIBUTE : DATA NOSHR ROM

D	H'000099C0	- H'000099C0	H'00000000
		asmfile	asmfile
	H'000099C0	- H'000099CF	H'00000010

usb usb

* TOTAL ADDRESS * H'000099C0 - H'000099CF H'00000010

ATTRIBUTE : DATA NOSHR

C H'00009A00 - H'00009C84 H'00000285
main main
H'00009C86 - H'00009CB4 H'0000002F
EV_Simulator EV_Simulator
H'00009CB6 - H'00009CC0 H'0000000B
EV_Puls EV_Puls
H'00009CC2 - H'00009CE9 H'00000028
EV_Controller EV_Controller
H'00009CEA - H'00009D4F H'00000066
EV_Input EV_Input
H'00009D50 - H'00009E0F H'000000C0
EV_Display EV_Display
H'00009E10 - H'00009E5B H'0000004C
EV_OpenClose EV_OpenClose
H'00009E5C - H'00009E9E H'00000043
EV_UpDown EV_UpDown
H'00009EA0 - H'00009F11 H'00000072
EV_File EV_File
H'00009F12 - H'00009F19 H'00000008
EV_Time EV_Time
H'00009F1A - H'00009F51 H'00000038
Timer Timer
H'00009F52 - H'00009F5C H'0000000B
Panel Panel
H'00009F5E - H'0000A017 H'000000BA
usb usb

H8/300H LINKAGE EDITOR (Evaluation software) Ver.1.0 PAGE : 5

*** LINKAGE EDITOR LINK MAP LIST ***

SECTION NAME	START - END	LENGTH
	UNIT NAME	MODULE NAME

ATTRIBUTE : DATA NOSHR

C H'0000A018 - H'0000A01F H'00000008
_fmtout _fmtout
H'0000A020 - H'0000A11F H'00000100
_ctype _ctype
H'0000A120 - H'0000A1A7 H'00000088
_its _its
H'0000A1A8 - H'0000A1AF H'00000008
_log10 _log10
H'0000A1B0 - H'0000A28F H'000000E0
_pow5 _pow5

```

H'0000A290 - H'0000A393 H'00000104
              _power                _power
H'0000A394 - H'0000A39B H'00000008
              frexp                  frexp
H'0000A39C - H'0000A3A3 H'00000008
              modf                   modf

```

* TOTAL ADDRESS * H'00009A00 - H'0000A3A3 H'000009A4

ATTRIBUTE : DATA NOSHR RAM

```

R           H'00FFE000 - H'00FFE000 H'00000000
              asmfile                asmfile
H'00FFE000 - H'00FFE00F H'00000010
              usb                    usb

```

* TOTAL ADDRESS * H'00FFE000 - H'00FFE00F H'00000010

ATTRIBUTE : DATA NOSHR

```

B           H'00FFE010 - H'00FFE011 H'00000002
              asmfile                asmfile
H'00FFE012 - H'00FFE223 H'00000212
              main                   main
H'00FFE224 - H'00FFE237 H'00000014
              EV_File                EV_File
H'00FFE238 - H'00FFE497 H'00000260
              Timer                  Timer
H'00FFE498 - H'00FFE517 H'00000080
              Panel                  Panel
H'00FFE518 - H'00FFE567 H'00000050
              sci                     sci
H'00FFE568 - H'00FFE5A7 H'00000040
              lcd                    lcd

```

H8/300H LINKAGE EDITOR (Evaluation software) Ver.1.0 PAGE : 6

*** LINKAGE EDITOR LINK MAP LIST ***

SECTION NAME	START	END	LENGTH
	UNIT NAME	MODULE NAME	

ATTRIBUTE : DATA NOSHR

```

B           H'00FFE5A8 - H'00FFE87B H'000002D4
              usb                    usb
H'00FFE87C - H'00FFE8B7 H'0000003C
              _fmtout                _fmtout
H'00FFE8B8 - H'00FFE8BB H'00000004
              _rnext                 _rnext
H'00FFE8BC - H'00FFE8BD H'00000002

```

_errno

_errno

* TOTAL ADDRESS * H'00FFE010 - H'00FFE8BD H'000008AE

H8/300H LINKAGE EDITOR (Evaluation software) Ver.1.0 PAGE : 1

*** LINKAGE EDITOR EXTERNALLY DEFINED SYMBOLS LIST ***

SYMBOL NAME	ADDR	TYPE
\$ADDD\$3	H'00005AC2	DAT
\$CMPD\$3	H'00007E3C	DAT
\$DIVD\$3	H'00005D6C	DAT
\$DIVL\$3	H'00007EF8	DAT
\$DIVUL\$3	H'00008522	DAT
\$DSL\$3	H'00008F7C	DAT
\$DSRUC\$3	H'00008F9E	DAT
\$DTOL\$3	H'00005F04	DAT
\$EQD\$3	H'00005F7E	DAT
\$GED\$3	H'00005F8A	DAT
\$ITOD\$3	H'00008FC0	DAT
\$LTD\$3	H'00005F9A	DAT
\$LTOD\$3	H'00005FAA	DAT
\$MULD\$3	H'000090AC	DAT
\$MULL\$3	H'00007F1E	DAT
\$MV8\$3	H'00005FF0	DAT
\$MVN\$3	H'0000600E	DAT
\$NED\$3	H'00006036	DAT
\$SUBD\$3	H'00005A92	DAT
\$sp_regld\$3	H'00006044	DAT
\$sp_regsv\$3	H'00006066	DAT
_Checkfmove	H'000043C4	ENT
_Clear	H'00004976	ENT
_ClearLCD	H'00004CBC	ENT
_Close	H'00003626	ENT
_CloseMotor	H'000030DA	ENT
_Cnt	H'00FFE012	DAT
_Command_Read	H'000040C0	ENT
_Command_Write	H'0000410C	ENT
_Destroy	H'00001128	ENT
_DisableInterrupt	H'000002AC	DAT
_Disp	H'00002E04	ENT
_DispInput	H'00002DB4	ENT
_DispUSBPort	H'00004EEC	ENT
_Door	H'00002E06	ENT
_Down	H'00003E6E	ENT
_DownMotor	H'00003942	ENT
_EV_AddressDataSet	H'000018CE	ENT
_EV_AddressSet	H'000017EA	ENT
_EV_Controller	H'00001AFA	ENT
_EV_DataSet	H'0000185C	ENT
_EV_EnableSet	H'000017D4	ENT
_EV_File	H'00003F0A	ENT
_EV_Input	H'000024E2	ENT
_EV_Puls	H'0000199E	ENT

_EV_Set	H'0000177E	ENT
_EV_Simulator	H'000012BE	ENT
_EV_Time	H'000042AA	ENT
_EnableInterrupt	H'000002A8	DAT
_GetChar	H'00002476	ENT
_GetCurrentTime	H'0000430C	ENT

H8/300H LINKAGE EDITOR (Evaluation software) Ver.1.0 PAGE : 2

*** LINKAGE EDITOR EXTERNALLY DEFINED SYMBOLS LIST ***

SYMBOL NAME	ADDR	TYPE
_GetPermit	H'000043A6	ENT
_GetSCI	H'00004AFC	ENT
_GetSW	H'0000126C	ENT
_H8init	H'00001292	ENT
_Init	H'00000E94	ENT
_InitITU	H'000048FA	ENT
_InitLCD	H'00004C4C	ENT
_InitSCI	H'00004ACC	ENT
_InitUSB	H'00004DF2	ENT
_InterruptITU0	H'00004930	ENT
_LCDOut4	H'00004BF4	ENT
_Limit_Read	H'00004266	ENT
_LocateLCD	H'00004CFC	ENT
_Motor_Read	H'00004218	ENT
_Motor_Write	H'000041E0	ENT
_OnCloseMotor	H'000030F6	ENT
_OnController	H'00001C9A	ENT
_OnDownMotor	H'0000395E	ENT
_OnInitWaitDoorChangeLog	H'00003362	ENT
_OnInitWaitPositionChangeLog	H'00003BBC	ENT
_OnInput	H'0000260E	ENT
_OnOpenMotor	H'00002E96	ENT
_OnPuls	H'000019DC	ENT
_OnSimulator	H'000013A8	ENT
_OnUpMotor	H'000036FE	ENT
_OnWaitCloseDoorChangeLog	H'0000350A	ENT
_OnWaitDownPositionChangeLog	H'00003D58	ENT
_OnWaitOpenDoorChangeLog	H'00003436	ENT
_OnWaitUpPositionChangeLog	H'00003C8A	ENT
_Open	H'000035DE	ENT
_OpenMotor	H'00002E7A	ENT
_PermitCommand_Read	H'00004030	ENT
_PermitCommand_Write	H'0000407C	ENT
_PermitTurnOpen_Read	H'00004150	ENT
_PermitTurnOpen_Write	H'0000419C	ENT
_Position	H'0000366E	ENT
_Printf	H'0000499E	ENT
_PrintLCD	H'00004D20	ENT
_PrintSCI	H'00004B1C	ENT
_PutLCD	H'00004CD0	ENT
_PutSCI	H'00004AEC	ENT
_Read	H'00003F9E	ENT

_ReadString	H'00003FF8	ENT
_Repaint	H'000004C0	ENT
_Run	H'0000052C	ENT
_ScanSCI	H'00004B0C	ENT
_SetCurrentTime	H'000042EE	ENT
_SetLED	H'0000121E	ENT
_SetPermit	H'00004372	ENT
_SleepMSec	H'00004414	ENT
_Start	H'000046E6	ENT

H8/300H LINKAGE EDITOR (Evaluation software) Ver.1.0 PAGE : 3

*** LINKAGE EDITOR EXTERNALLY DEFINED SYMBOLS LIST ***

SYMBOL NAME	ADDR	TYPE
_Stop	H'00004714	ENT
_Thread_Start	H'000047AC	ENT
_Thread_Toggle	H'000047EA	ENT
_Thread_checkAllDelete	H'00004732	ENT
_Thread_checkStayAnother	H'0000474C	ENT
_Thread_getThread	H'00004772	ENT
_Up	H'00003E26	ENT
_UpMotor	H'000036E2	ENT
_WaitDoorChangeLog	H'0000333A	ENT
_WaitPositionChangeLog	H'00003BA2	ENT
_WaitSecond	H'00004350	ENT
_Wait_ms	H'000043E6	ENT
_Write	H'00003F14	ENT
_WriteString	H'00003F66	ENT
__add	H'000095B2	ENT
__allzero	H'0000855E	ENT
__calcpw	H'00008586	ENT
__ctype	H'0000A020	DAT
__dti	H'00007F3E	ENT
__duchek	H'000092E4	ENT
__errno	H'00FFE8BC	DAT
__fmtout	H'0000608E	ENT
__its	H'00008324	ENT
__log10	H'0000867E	ENT
__lsft	H'00009336	ENT
__lsfts	H'00008722	ENT
__mult	H'00009388	ENT
__mult64	H'000089D0	ENT
__pow10	H'00009516	ENT
__pow5	H'0000879A	ENT
__power	H'00008A58	ENT
__rnd	H'00008BBA	ENT
__rnext	H'00FFE8B8	DAT
__rsfts	H'000087C8	ENT
__setsbit	H'00008CA4	ENT
__sub	H'00008842	ENT
__unpack	H'000088EE	ENT
_cntrl	H'00FFE090	DAT
_countUpNextRun	H'000044EA	ENT

delete	H'000046B2	ENT
_frexp	H'00008D40	ENT
_getClock	H'00004402	ENT
_get_inbufflen	H'000058A6	ENT
_get_outbufflen	H'000058BA	ENT
_i_cnt	H'00FFE016	DAT
_in	H'00FFE052	DAT
_initWOVI	H'0000495C	ENT
_init_usbbuff	H'0000570A	ENT
_j_cnt	H'00FFE018	DAT
_main	H'000002B0	ENT
_memcpy	H'00008992	ENT

H8/300H LINKAGE EDITOR (Evaluation software) Ver.1.0 PAGE : 4

*** LINKAGE EDITOR EXTERNALLY DEFINED SYMBOLS LIST ***

SYMBOL NAME	ADDR	TYPE
_memcpy	H'000084C8	ENT
_memset	H'000095FA	ENT
_modf	H'00008E46	ENT
_new_EV_Status	H'00003EB6	ENT
_new_Thread	H'0000450A	ENT
_nextRun	H'000044A6	ENT
_puls	H'00FFE188	DAT
_rand	H'000058CE	ENT
_read_buff	H'00005848	ENT
_read_outbuff	H'000057EA	ENT
_s	H'00FFE04E	DAT
_simu	H'00FFE19C	DAT
_sprintf	H'00005908	ENT
_status	H'00FFE224	DAT
_strcmp	H'00005966	ENT
_strcpy	H'00005990	ENT
_strlen	H'000059AC	ENT
_th	H'00FFE01A	DAT
_th1	H'00FFE022	DAT
_th101	H'00FFE27C	DAT
_th102	H'00FFE29A	DAT
_th111	H'00FFE2B8	DAT
_th112	H'00FFE2D6	DAT
_th113	H'00FFE2F4	DAT
_th114	H'00FFE312	DAT
_th119	H'00FFE330	DAT
_th120	H'00FFE34E	DAT
_th121	H'00FFE36C	DAT
_th122	H'00FFE38A	DAT
_th123	H'00FFE3A8	DAT
_th130	H'00FFE3C6	DAT
_th131	H'00FFE3E4	DAT
_th141	H'00FFE402	DAT
_th142	H'00FFE420	DAT
_th143	H'00FFE43E	DAT
_th144	H'00FFE45C	DAT

_th145	H'00FFE47A	DAT
_th19	H'00FFE032	DAT
_th20	H'00FFE036	DAT
_th41	H'00FFE03A	DAT
_th42	H'00FFE03E	DAT
_th43	H'00FFE042	DAT
_th44	H'00FFE046	DAT
_th45	H'00FFE04A	DAT
_usb_int	H'00004F50	ENT
_vsprintf	H'000059C8	ENT
_wovi	H'00004958	ENT
_woviClock	H'00FFE238	DAT
_woviInit	H'000048C8	ENT
_woviRun	H'0000482E	ENT
_woviThreadFirst	H'00FFE240	DAT

H8/300H LINKAGE EDITOR (Evaluation software) Ver.1.0 PAGE : 5

*** LINKAGE EDITOR EXTERNALLY DEFINED SYMBOLS LIST ***

SYMBOL NAME	ADDR	TYPE
_woviThreadLast	H'00FFE25E	DAT
_write_buff	H'00005784	ENT
_write_inbuff	H'00005728	ENT

MAKE Version 5.2 Copyright (c) 1987, 2000 Borland

```
bcc32 -O2 -w -tWC -D"USE_BCC" -c Panel.c
```

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

Panel.c:

```
bcc32 -O2 -w -tWC -D"USE_BCC" -c Timer.c
```

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

Timer.c:

```
bcc32 -O2 -w -tWC -D"USE_BCC" -c EV_Time.c
```

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

EV_Time.c:

```
bcc32 -O2 -w -tWC -D"USE_BCC" -c EV_File.c
```

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

EV_File.c:

警告 W8004 EV_File.c 74: 'Ret' に代入した値は使われていない(関数 Write)

警告 W8004 EV_File.c 108: 'Ret' に代入した値は使われていない(関数 WriteString)

警告 W8071 EV_File.c 166: 変換によって有効桁が失われる(関数 Read)

警告 W8004 EV_File.c 162: 'Ret' に代入した値は使われていない(関数 Read)

警告 W8004 EV_File.c 203: 'Ret' に代入した値は使われていない(関数 ReadString)

警告 W8004 EV_File.c 229: 'Ret' に代入した値は使われていない(関数 PermitCommand_Read)

警告 W8004 EV_File.c 247: 'Ret' に代入した値は使われていない(関数 PermitCommand_Write)

警告 W8004 EV_File.c 265: 'Ret' に代入した値は使われていない(関数 Command_Read)

警告 W8004 EV_File.c 283: 'Ret' に代入した値は使われていない(関数 Command_Write)

警告 W8004 EV_File.c 301: 'Ret' に代入した値は使われていない(関数 PermitTurnOpen_Read)

警告 W8004 EV_File.c 319: 'Ret' に代入した値は使われていない(関数 PermitTurnOpen_Write)

警告 W8066 EV_File.c 343: 実行されないコード(関数 Motor_Write)

警告 W8066 EV_File.c 364: 実行されないコード(関数 Motor_Read)

警告 W8066 EV_File.c 367: 実行されないコード(関数 Motor_Read)

警告 W8066 EV_File.c 382: 実行されないコード(関数 Limit_Read)

警告 W8066 EV_File.c 385: 実行されないコード(関数 Limit_Read)

```
bcc32 -O2 -w -tWC -D"USE_BCC" -c EV_UpDown.c
```

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

EV_UpDown.c:

```
bcc32 -O2 -w -tWC -D"USE_BCC" -c EV_OpenClose.c
```

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

EV_OpenClose.c:

```
bcc32 -O2 -w -tWC -D"USE_BCC" -c EV_Display.c
```

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

EV_Display.c:

```
bcc32 -O2 -w -tWC -D"USE_BCC" -c EV_Input.c
```

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

EV_Input.c:

```
bcc32 -O2 -w -tWC -D"USE_BCC" -c EV_Controller.c
```

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

EV_Controller.c:

```
bcc32 -O2 -w -tWC -D"USE_BCC" -c EV_Puls.c
```

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

EV_Puls.c:

警告 W8057 EV_Puls.c 53: パラメータ 'addressDataSet' は一度も使用されない(関数 EV_Set)

警告 W8057 EV_Puls.c 53: パラメータ 'dataSet' は一度も使用されない(関数 EV_Set)

警告 W8057 EV_Puls.c 53: パラメータ 'addressClockSet' は一度も使用されない(関数 EV_Set)

警告 W8057 EV_Puls.c 53: パラメータ 'clockSet' は一度も使用されない(関数 EV_Set)

警告 W8057 EV_Puls.c 182: パラメータ 'th' は一度も使用されない(関数 EV_Puls)

```
bcc32 -O2 -w -tWC -D"USE_BCC" -c EV_Simulator.c
```

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

EV_Simulator.c:

警告 W8019 EV_Simulator.c 68: コードは効果を持たない(関数 OnSimulator)

警告 W8019 EV_Simulator.c 86: コードは効果を持たない(関数 OnSimulator)

警告 W8019 EV_Simulator.c 104: コードは効果を持たない(関数 OnSimulator)

警告 W8019 EV_Simulator.c 122: コードは効果を持たない(関数 OnSimulator)

```
bcc32 -O2 -w -tWC -D"USE_BCC" -c main.c
```

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

main.c:

警告 W8004 main.c 309: 'key' に代入した値は使われていない(関数 Run)

```
ilink32 /Tpe -L"C:¥borland¥bcc55¥Lib" Panel.obj Timer.obj EV_Time.obj EV_File.obj
```

```
EV_UpDown.obj EV_OpenClose.obj EV_Display.obj EV_Input.obj EV_Controller.obj EV_Puls.obj
```

```
EV_Simulator.obj main.obj c0x32.obj,main.exe,cw32.lib import32.lib
```

Turbo Incremental Link 5.00 Copyright (c) 1997, 2000 Borland

MAKE Version 5.2 Copyright (c) 1987, 2000 Borland

```
del *.obj
```

```
del main.tds
```

```
del main.ilc
```

```
del main.ild
```

```
del main.ilf
```

```
del main.ils
```

H8/300H ASSEMBLER (Evaluation software) Ver.1.0

```
*****TOTAL ERRORS    0
```

```
*****TOTAL WARNINGS  0
```

H8/300H LINKAGE EDITOR (Evaluation software) Ver.1.0

: OUTPUT c_thread

: PRINT c_thread

: INPUT asmfile, main, EV_Simulator, EV_Puls, EV_Controller, EV_Input, EV_Display, EV_OpenClose,
EV_UpDown, EV_File, EV_Time, Timer, Panel, sci, lcd, usb

: LIB c:¥h8¥akic¥c38hab

: START R(OFFE000), P(200), D(99C0), C(9A00)

: ROM (D, R)

: EXIT

LINKAGE EDITOR COMPLETED

H8/300H OBJECT CONVERTER (Evaluation software) Ver.1.0

OBJECT CONVERTER COMPLETED

著作者:

しのみや ひでみね

篠宮 英峰