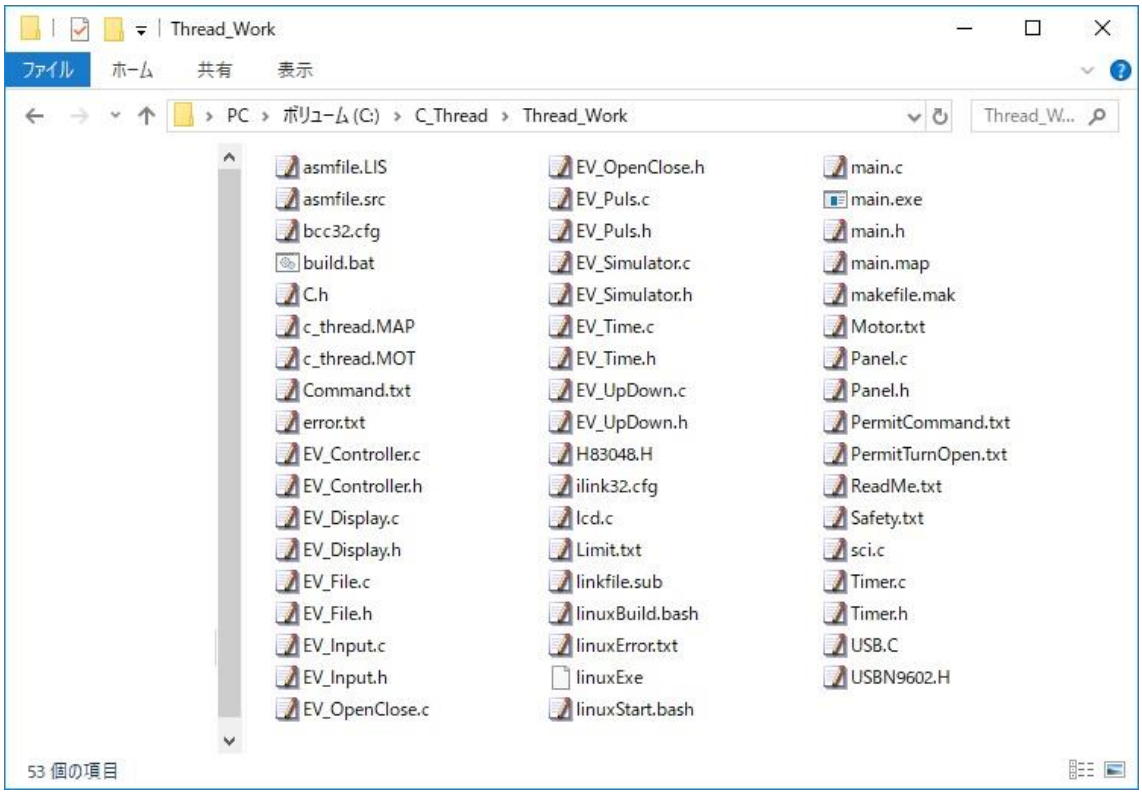
発明の巻

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)が複数で動かせます。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 4 月 7 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。

==========

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

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

Linux CentOS6 GCC の利用

私は、 CentOS にしましたが、 linuxBuild.bash ・ linuxStart.bash

が読める方は、お好きな Linux で挑戦してみてください

==========

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 OVI0

_INT_OVI0: .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> cource, You click a 'R' button.

For the <2> cource, 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)で使って

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 yynnyynn にして上書き保存してください

yynnyynn は籠が下階停止状態で扉が閉停止状態を意味します

yynnnynn は籠が下階停止状態で扉が閉低速区域を意味します

yynnnnnn は籠が下階停止状態で扉が中間高速区域を意味します

yynnnnyn は籠が下階停止状態で扉が開低速区域を意味します

yynnnnyy 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

nnynyynn 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

nnyynynn は籠が上階停止状態で扉が閉低速区域を意味します

nnyynnnn は籠が上階停止状態で扉が中間高速区域を意味します

nnyynnyn は籠が上階停止状態で扉が開低速区域を意味します

nnyynnyy は籠が上階停止状態で扉が開停止状態を意味します

==========

Linux CentOS6 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

makefile.mak build.bat asmfile.src linkfile.sub linuxBuild.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。

error.txt linuxError.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実行ソフトが起動します。

c_thread.MOT を AKI-H8 3052F USB に書き込みます。

linuxStart.bash をダブルクリックすると、
GCC実行ソフト linuxExe が起動します。

参照ライブラリ

```
/************************************************************************/
/*      H8/3048F Include File                                           */
/************************************************************************/
struct st_sam {                         /* struct DMAC  */
      void              *MAR;           /* MAR          */
      unsigned int      ETCR;           /* ETCR         */
      unsigned char     IOAR;           /* IOAR         */
      unsigned char     DTCR;           /* DTCR         */
};

struct st_fam {                         /* struct DMAC  */
      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 {                         /* struct 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 {                        /* struct 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 {                        /* struct 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 {                         /* struct 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_rfshc {                   /* 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_MCNTRL              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 TxTogle(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をｲﾈｰﾌﾞﾙ         */
        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
```

```
をｾｯﾄ */
    WriteUSB(USB_RXMSK,USB_RXFIFO0+USB_RXFIFO1+USB_RXFIFO2+USB_RXFIFO3);  /* RX MASK
をｾｯﾄ */
    WriteUSB(USB_ALTMSK,USB_SD3+USB_RESET_A);          /* ALT MASKをｾｯﾄ */
    WriteUSB(USB_MAMSK,USB_INTR_E+USB_RX_EV+USB_NAK+USB_TX_EV+USB_ALT);          /*
MAIN MASKをｾｯﾄ */
    FlushTXC(0);
    FlushRXC(1);
    FlushTXC(1);
    WriteUSB(USB_TXC1,0);
    WriteUSB(USB_RXC1,0);
    WriteUSB(USB_RXC0,USB_RX_EN);                  /* RX0をｲﾈｰﾌﾞﾙ        */
}

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のﾉｰﾄﾞを動作可にする */
}

/* 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   nakevent,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);
        /* ﾉｰﾄﾞをｻｽﾍﾟﾝﾄﾞ */
        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);
            /* ﾉｰﾄﾞを動作可能にする */
            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なのﾃﾞｰﾀの有無に関係無くDATA1として送信 */
        DATA0_1 ¦= 1;
        TxTogle(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();
            TxTogle(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 = WriteUSBBurst(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をｱﾄﾞﾚｽ1としてｲﾈｰﾌﾞﾙ */
        WriteUSB(USB_EPC1,USB_EP_EN+01);
        /* TX1送信可 */
```

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


        FlushRXC(1);
        /* EPC2をｱﾄﾞﾚｽ2としてｲﾈｰﾌﾞﾙ */
        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 TxTogle(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 WriteUSBBurst(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 = WriteUSBBurst(USB_TXD1,USB_TXS1,(char*)rx2buff,sz);
	}
	TxTogle(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 ﾊﾟﾘﾃｨ無し STOP1
=================================================================*/
void InitSCI()
{
    int         i;
    SCI1.SCR = 0;
    SCI1.SMR = 0;                   /* ﾊﾟﾘﾃｨ無し 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処理

                                                (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=coursor , 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 */
#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
#include "h83048.h"
#define SLEEP_PER_SEC 16000.0
#define NOTUSE_FILES
#else
#include <time.h> /* time_t, tm, time(), clock(), CLOCKS_PER_SEC */
#ifdef USE_LINUX
#include <termios.h> /* kbhit() */
#include <unistd.h> /* kbhit() */
#include <fcntl.h> /* kbhit() */
#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¥0") == 0)
  {
      sprintf(str, "%s", "¥n¥f");
  }
  sprintf(buff,"%s", str);
  PrintSCI("%s",buff);
  PrintLCD(str);
#endif
  break;
 case Panel:
#ifndef USE_BCC
  if(strcmp(str, "¥n¥0") == 0)
  {
      sprintf(str, "%s", "¥n¥f");
  }
  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 woviInit(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
void wovi(double threadPerSec); /* タイマ関数 */
```

```
void initWOVI(void); /* タイマ初期化関数 */
#endif
#ifdef USE_BCC
void PrintCurrentTime(void); /* 現在日時表示 */
#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;
#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;
    }
#ifndef 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;
    }
```

```
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;
    }
#endif
#ifdef USE_BCC
    new_List = (Thread *)calloc(1, sizeof(Thread));
#endif
    if(new_List == NULL)
    {
        PrintF(Panel, "¥n");
        PrintF(Panel, "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 woviInit(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 */
void wovi(double threadPerSec)
{
#ifdef USE_BCC
	/* woviClock は秒数時計 */
#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、 */
	/* スピードが下がります。 */
	/* 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;
    }
    #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
==========================================================*/
#ifdef 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, strlength, 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¥0", 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);
/* 経過時間 */
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_OPSL = OFF;
	This->m_OPST = OFF;
	This->p_CloserSlow = &This->m_CLSL;
	This->p_CloserStop = &This->m_CLST;
	This->p_OpennerSlow = &This->m_OPSL;
	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_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_CloserSlow == 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 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_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_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;
	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;
}
```

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

```
/* EV_Display.c */

#include "C.h"
#include "EV_Display.h"

void DispInput(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, "¥n              ");
        }
        for(i = 4; i < 8; i++)
        {
            PrintF(Monitor, "¥n    00000000    ");
        }
        for(i = 8; i < 12; i++)
        {
            PrintF(Monitor, "¥n              ");
        }
    }
    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, "¥n              ");
```

```
        }
        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 */
    DispInput();
    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;
            }
        }
    }
#elif USE_LINUX
    if(kbhit())
    {
        Ret = (char) getchar();
    }
#elif USE_MSVS2005
    if(_kbhit())
    {
        Ret = (char) _getche();
    }
#else
    if(kbhit())
    {
        Ret = (char) getche();
    }
#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(Panel, "¥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(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');
```

```
            /* 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(Panel, "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

/*==========================================================================
  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){
#ifdef USE_BCC
	printf(" ");
	/* address data set */
	printf("%d", addressDataSet);
	/* data set */
	printf("%d", dataSet);
	/* address clock set */
	printf("%d", addressClockSet);
	/* clock set */
	printf("%d", clockSet);
	/* disable set */
	printf("0");
#else
	/* address data set */
	if(addressDataSet == 0){
		PB.DR &= 0x07;
	}
	else if(addressDataSet == 1){
		PB.DR ¦= 0x08;
	}
	/* data set */
```

```
    if(dataSet == 0){
        PB.DR &= 0x0b;
    }
    else if(dataSet == 1){
        PB.DR |= 0x04;
    }
    /* address clock set */
    if(addressClockSet == 0){
        PB.DR &= 0x0d;
    }
    else if(addressClockSet == 1){
        PB.DR |= 0x02;
    }
    /* clock set */
    if(clockSet == 0){
        PB.DR &= 0x0e;
    }
    else if(clockSet == 1){
        PB.DR |= 0x01;
    }
#endif
    return;
}

void EV_EnableSet(void){
#ifdef USE_BCC
    printf(" ");
    /* address data set */
    printf("0");
```

```
        /* data set */
        printf("0");
        /* address clock set */
        printf("0");
        /* clock set */
        printf("0");
        /* enable set */
        printf("1");
#else
        /* address data set */
        PB.DR &= 0x07;
        /* data set */
        PB.DR &= 0x0b;
        /* address clock set */
        PB.DR &= 0x0d;
        /* clock set */
        PB.DR &= 0x0e;
#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':
                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);
    #ifndef USE_BCC
        PB.DDR = 0xff; /* bit7..0 out */
        PB.DR ¦= 0xff;
```

```
#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);
		/* モーター命令解読 */
		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;
}
```

```
/* 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_THREAD
typedef struct tag_Count
{
#ifndef USE_BCC
	int cnt[2];
#else
	int cnt[8];
#endif
}Count;
#endif
```

```
/* main.c */

#include "C.h"
#include "main.h"

#ifdef USE_THREAD
Count Cnt;
int i_cnt, j_cnt;
#ifndef USE_BCC
/* 擬似スレッドの擬似インスタンス宣言 */
Thread* th[2];
#else
/* 擬似スレッドの擬似インスタンス宣言 */
Thread* th[8];
#endif
/* 疑似スレッドの疑似インスタンス宣言 */
Thread *th1[4];
Thread *th19;
Thread *th20;
Thread *th41;
Thread *th42;
Thread *th43;
Thread *th44;
#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;

void main(void)
{
#ifndef 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
#ifndef 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 ｾﾝｽｺﾝﾄﾛｰﾙ 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 */
        wovi(5000000.0);
        if(Thread_checkAllDelete() == OK)
        {
            break;
        }
    }
#endif
```

```
    Clear();
    PrintF(Panel, "NEXT          ");
    /* 2秒待機 */
    SleepMSec(2000);
    Clear();
#ifndef 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++) {} /* 適当にウエイト */
    }
#else
    printf("END");
    /* 5秒待機 */
```

```
	SleepMSec(5000);
	return;
#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 *th11;
#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();
#ifndef 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) * 30));
            while(kbhit())
```

```
            {
#ifdef USE_LINUX
                key = (char) getchar();
#else
                key = (char) getche();
#endif
            }
#endif
        }
        else if(This->ID == 2)
        {
            Repaint();
#ifndef 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) * 30));
```

```
            while(kbhit())
            {
#ifdef USE_LINUX
                key = (char) getchar();
#else
                key = (char) getche();
#endif
            }
#endif
        }
#ifdef USE_BCC
        else if(((This->ID) >= 3) && ((This->ID) <= 8))
        {
                Repaint();
                Cnt.cnt[(This->ID) - 1]++;
                nextRun(This, (((rand() % 9) + 10) * 200));
        }
#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(Panel, "<3>1st         ");
            countUpNextRun(This, (1700 * 3));
        }
        else if(This->count == 2)
        {
            Clear();
            PrintF(Panel, "<3>2nd         ");
```

```
            countUpNextRun(This, (1700 * 3));
        }
        else if(This->count == 3)
        {
            Clear();
            PrintF(Panel, "<3>3rd         ");
            countUpNextRun(This, (1700 * 3));
        }
        else
        {
            Clear();
            PrintF(Panel, "<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
#ifndef 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);
    }
#ifdef USE_BCC
    else if((This->ID >= 24) && (This->ID <= 29))
    {
        printf("%d", This->ID - 20);
        nextRun(This, 2000);
    }
#endif
```

```
#ifndef 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
#ifndef USE_BCC
```

```
            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(Panel, "¥n");
            PrintF(Panel, "Thread Ready GO!");
            /* 2秒待機 */
            countUpNextRun(This, 2000);
        }
        else if(This->count == 2)
        {
#ifndef 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)
        {
#ifndef 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
            /* 2秒待機 */
```

```
        countUpNextRun(This, 2000);
    }
    else if(This->count == 5)
    {
        Clear();
        PrintF(Panel, "NEXT          ");
        /* 2秒待機 */
        countUpNextRun(This, 2000);
    }
    else if(This->count == 6)
    {
        Clear();
        /* 第2部分 */
        PrintF(Panel, "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(Panel, "NEXT          ");
	/* 2秒待機 */
	countUpNextRun(This, 2000);
}
else if(This->count == 11)
{
	Clear();
	/* 第3部分 */
	PrintF(Panel, "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(Panel, "NEXT          ");
        /* 2秒待機 */
        countUpNextRun(This, 2000);
    }
    else if(This->count == 16)
    {
```

```
            Clear();
            /* 第4部分 */
            PrintF(Panel, "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);

            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);
    }
    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);
    }
#ifdef 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);
    }
```

```
	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(Panel, "<1>Destroy");
	}
	else if(This->ID == 12)
	{
		PrintF(Panel, "<2>Destro");
	}
	else if(This->ID == 13)
	{
		PrintF(Panel, "<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
    return;
}
#endif
```

```
#ifndef 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

```
-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
 .IMPORT _main
 .IMPORT _usb_int
 .IMPORT _InterruptITU0

 .DATA.L _start ;リセットベクトル
;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
;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:
 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:
 jsr @_main

; 割り込み未使用
int_error:
 rte

usb_interrupt:
 push.l er0
 push.l er1
 push.l er2
 push.l er3
 push.l er4
 push.l er5
 push.l er6
 jsr @_usb_int
 pop.l er6
 pop.l er5
 pop.l er4
 pop.l er3
 pop.l er2
 pop.l er1
 pop.l er0
```

```
 rte

_ITU_OVI_0:
 push.l er0
 push.l er1
 push.l er2
 push.l er3
 push.l er4
 push.l er5
 push.l er6
 jsr @_InterruptITU0
 pop.l er6
 pop.l er5
 pop.l er4
 pop.l er3
 pop.l er2
 pop.l er1
 pop.l er0
 rte

;------------------------------------------------------
; 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=%akih8cDir%;%akih8asmDir%;%path%
set CurrentDir="%~dp0"
cd %CurrentDir%
del error.txt
make -f makefile.mak >> error.txt
make -f makefile.mak clean >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% Panel.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% Timer.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% EV_Time.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% EV_File.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% EV_UpDown.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% EV_OpenClose.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% EV_Display.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% EV_Input.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% EV_Controller.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% EV_Puls.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% EV_Simulator.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% main.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% usb.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% sci.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% lcd.c >> error.txt
a38h asmfile.src >> error.txt
l38h -subcommand=linkfile.sub >> error.txt
c38h c_thread >> 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 linuxBuild.bash
rm ./linuxError.txt
gcc -D"USE_LINUX" -o linuxExe main.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 &>>./linuxError.txt
exit
```

```
#!/bin/bash
cd `dirname $0`
find $PWD -name linuxStart.bash
./linuxExe
exit
```

実行環境状態ファイル

r

nnyyyynn

q

C

N

S

出力ファイル

```
 Start         Length     Name                   Class
0001:00401000 000010CA4H _TEXT                  CODE
0002:00412000 0000039C0H _DATA                  DATA
0003:004159C0 000000B6CH _BSS                   BSS
0004:00000000 0000000A4H _TLS                   TLS
```

```
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***     04/07/18 13:06:57
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     .IMPORT _main
  6                   6     .IMPORT _usb_int
  7                   7     .IMPORT _InterruptITU0
  8                   8
  9 000000 00000000   9     .DATA.L _start ;リセットベクトル
 10                  10    ;1 Reserved
 11 000004 00000000  11     _INT_Reserved1: .DATA.L int_error
 12                  12    ;2 Reserved
 13 000008 00000000  13     _INT_Reserved2: .DATA.L int_error
 14                  14    ;3 Reserved
 15 00000C 00000000  15     _INT_Reserved3: .DATA.L int_error
 16                  16    ;4 Reserved
 17 000010 00000000  17     _INT_Reserved4: .DATA.L int_error
 18                  18    ;5 Reserved
 19 000014 00000000  19     _INT_Reserved5: .DATA.L int_error
 20                  20    ;6 Reserved
 21 000018 00000000  21     _INT_Reserved6: .DATA.L int_error
 22                  22    ;7 NMI
 23 00001C 00000000  23     _INT_NMI: .DATA.L int_error
 24                  24    ;8 TRAP
```

```
25 000020 00000000          25    _INT_TRAP1: .DATA.L int_error
26                          26    ;9 TRAP
27 000024 00000000          27    _INT_TRAP2: .DATA.L int_error
28                          28    ;10 TRAP
29 000028 00000000          29    _INT_TRAP3: .DATA.L int_error
30                          30    ;11 TRAP
31 00002C 00000000          31    _INT_TRAP4: .DATA.L int_error
32                          32    ;12 IRQ0
33 000030 00000000          33    IRQ0: .DATA.L int_error
34                          34    ;13 IRQ1
35 000034 00000000          35    IRQ1: .DATA.L int_error
36                          36    ;14 IRQ2
37 000038 00000000          37    IRQ2: .DATA.L int_error
38                          38    ;15 IRQ3
39 00003C 00000000          39    IRQ3: .DATA.L int_error
40                          40    ;16 IRQ4
41 000040 00000000          41    IRQ4: .DATA.L int_error
42                          42    ;17 IRQ5
43 000044 00000000          43    IRQ5: .DATA.L usb_interrupt ;USB割り込み
44                          44    ;18 Reserved
45 000048 00000000          45    _INT_Reserved18: .DATA.L int_error
46                          46    ;19 Reserved
47 00004C 00000000          47    _INT_Reserved19: .DATA.L int_error
48                          48    ;20 WOVI
49 000050 00000000          49    _INT_WOVI: .DATA.L int_error
50                          50    ;21 CMI
51 000054 00000000          51    _INT_CMI: .DATA.L int_error
52                          52    ;22 Reserved
53 000058 00000000          53    _INT_Reserved22: .DATA.L int_error
```

```
  54                    54    ;23 Reserved
  55 00005C 00000000          55    _INT_Reserved23: .DATA.L int_error
  56                    56    ;24 IMIA0
  57 000060 00000000          57    _INT_IMIA0: .DATA.L int_error
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***     04/07/18 13:06:57
PAGE    2
PROGRAM NAME =


  58                    58    ;25 IMIB0
  59 000064 00000000          59    _INT_IMIB0: .DATA.L int_error
  60                    60    ;26 OVI0
  61 000068 00000000          61    _INT_OVI0: .DATA.L _ITU_OVI_0 ;タイマ0割り込み
  62                    62    ;27 Reserved
  63 00006C 00000000          63    _INT_Reserved27: .DATA.L int_error
  64                    64    ;28 IMIA1
  65 000070 00000000          65    _INT_IMIA1: .DATA.L int_error
  66                    66    ;29 IMIB1
  67 000074 00000000          67    _INT_IMIB1: .DATA.L int_error
  68                    68    ;30 OVI1
  69 000078 00000000          69    _INT_OVI1: .DATA.L int_error
  70                    70    ;31 Reserved
  71 00007C 00000000          71    _INT_Reserved31: .DATA.L int_error
  72                    72    ;32 IMIA2
  73 000080 00000000          73    _INT_IMIA2: .DATA.L int_error
  74                    74    ;33 IMIB2
  75 000084 00000000          75    _INT_IMIB2: .DATA.L int_error
  76                    76    ;34 OVI2
  77 000088 00000000          77    _INT_OVI2: .DATA.L int_error
  78                    78    ;35 Reserved
```

```
 79 00008C 00000000          79    _INT_Reserved35: .DATA.L int_error
 80                          80    ;36 IMIA3
 81 000090 00000000          81    _INT_IMIA3: .DATA.L int_error
 82                          82    ;37 IMIB3
 83 000094 00000000          83    _INT_IMIB3: .DATA.L int_error
 84                          84    ;38 OVI3
 85 000098 00000000          85    _INT_OVI3: .DATA.L int_error
 86                          86    ;39 Reserved
 87 00009C 00000000          87    _INT_Reserved39: .DATA.L int_error
 88                          88    ;40 IMIA4
 89 0000A0 00000000          89    _INT_IMIA4: .DATA.L int_error
 90                          90    ;41 IMIB4
 91 0000A4 00000000          91    _INT_IMIB4: .DATA.L int_error
 92                          92    ;42 OVI4
 93 0000A8 00000000          93    _INT_OVI4: .DATA.L int_error
 94                          94    ;43 Reserved
 95 0000AC 00000000          95    _INT_Reserved43: .DATA.L int_error
 96                          96    ;44 DEND0A
 97 0000B0 00000000          97    _INT_DEND0A: .DATA.L int_error
 98                          98    ;45 DEND0B
 99 0000B4 00000000          99    _INT_DEND0B: .DATA.L int_error
100                         100    ;46 DEND1A
101 0000B8 00000000         101    _INT_DEND1A: .DATA.L int_error
102                         102    ;47 DEND1B
103 0000BC 00000000         103    _INT_DEND1B: .DATA.L int_error
104                         104    ;48 Reserved
105 0000C0 00000000         105    _INT_Reserved48: .DATA.L int_error
106                         106    ;49 Reserved
107 0000C4 00000000         107    _INT_Reserved49: .DATA.L int_error
```

```
 108                    108    ;50 Reserved
 109 0000C8 00000000        109    _INT_Reserved50: .DATA.L int_error
 110                    110    ;51 Reserved
 111 0000CC 00000000        111    _INT_Reserved51: .DATA.L int_error
 112                    112    ;52 ERI0
 113 0000D0 00000000        113    _INT_ERI0: .DATA.L int_error
 114                    114    ;53 RXI0
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***     04/07/18 13:06:57
PAGE    3
PROGRAM NAME =

 115 0000D4 00000000        115    _INT_RXI0: .DATA.L int_error
 116                    116    ;54 TXI0
 117 0000D8 00000000        117    _INT_TXI0: .DATA.L int_error
 118                    118    ;55 TEI0
 119 0000DC 00000000        119    _INT_TEI0: .DATA.L int_error
 120                    120    ;56 ERI1
 121 0000E0 00000000        121    _INT_ERI1: .DATA.L int_error
 122                    122    ;57 RXI1
 123 0000E4 00000000        123    _INT_RXI1: .DATA.L int_error
 124                    124    ;58 TXI1
 125 0000E8 00000000        125    _INT_TXI1: .DATA.L int_error
 126                    126    ;59 TEI1
 127 0000EC 00000000        127    _INT_TEI1: .DATA.L int_error
 128                    128    ;60 ADI
 129 0000F0 00000000        129    _INT_ADI: .DATA.L int_error
 130                    130
 131                    131    ;------------------------------------------------------
```

```
132 000000                132     .SECTION P,CODE,ALIGN=2
133 000000                133     _start:
134 000000 7A0700FFFF10       134     mov.l #H'0FFFF10,er7
135                       135
136                       136     ;初期化付きデータを使用する場合、RAMに転送する
137                       137     ; c_thread.MAP のメモリアドレス使用状況を見る
138                       138     ; メモリアドレスが重複するとコンパイルエラーになる
139                       139     ; linkfile.sub も D(99C0), C(9A00) 等必要があれば合わせる
140 000006 7A00000099C0        140     mov.l #H'99C0, er0 ; 転送元(99C0)
141 00000C 7A0100FFE000      141     mov.l #H'0FFE000, er1 ; 転送先
142 000012 7A0200000000      142     mov.l #DATA_END, er2 ; 転送終了
143 000018                143     init_loop:
144 000018 1F92           144     cmp.l er1, er2
145 00001A 58700008           145     beq init_end
146 00001E 6C0B           146     mov.b @er0+, r3l
147 000020 689B           147     mov.b r3l, @er1
148 000022 0B71           148     inc.l #1, er1
149 000024 40F2           149     bra init_loop
150 000026                150     init_end:
151 000026 5E000000           151     jsr @_main
152                       152
153                       153     ; 割り込み未使用
154 00002A                154     int_error:
155 00002A 5670           155     rte
156                       156
157 00002C                157     usb_interrupt:
158 00002C 01006DF0           158     push.l er0
159 000030 01006DF1           159     push.l er1
160 000034 01006DF2           160     push.l er2
```

```
 161 000038 01006DF3          161     push.l er3
 162 00003C 01006DF4          162     push.l er4
 163 000040 01006DF5          163     push.l er5
 164 000044 01006DF6          164     push.l er6
 165 000048 5E000000          165     jsr @_usb_int
 166 00004C 01006D76          166     pop.l er6
 167 000050 01006D75          167     pop.l er5
 168 000054 01006D74          168     pop.l er4
 169 000058 01006D73          169     pop.l er3
 170 00005C 01006D72          170     pop.l er2
 171 000060 01006D71          171     pop.l er1
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***     04/07/18 13:06:57
PAGE    4
PROGRAM NAME =


 172 000064 01006D70          172     pop.l er0
 173 000068 5670              173     rte
 174                          174
 175 00006A                   175     _ITU_OVI_0:
 176 00006A 01006DF0          176     push.l er0
 177 00006E 01006DF1          177     push.l er1
 178 000072 01006DF2          178     push.l er2
 179 000076 01006DF3          179     push.l er3
 180 00007A 01006DF4          180     push.l er4
 181 00007E 01006DF5          181     push.l er5
 182 000082 01006DF6          182     push.l er6
 183 000086 5E000000          183     jsr @_InterruptITU0
 184 00008A 01006D76          184     pop.l er6
 185 00008E 01006D75          185     pop.l er5
```

```
186 000092 01006D74          186     pop.l er4
187 000096 01006D73          187     pop.l er3
188 00009A 01006D72          188     pop.l er2
189 00009E 01006D71          189     pop.l er1
190 0000A2 01006D70          190     pop.l er0
191 0000A6 5670              191     rte
192                  192
193                  193    ;---------------------------------------------------------
194                  194    ; 割り込み許可、禁止ルーチン
195                  195
196                  196     .EXPORT _EnableInterrupt,_DisableInterrupt
197 0000A8               197    _EnableInterrupt:
198 0000A8 063F          198     andc.b #H'3f,ccr
199 0000AA 5470          199     rts
200 0000AC               200    _DisableInterrupt:
201 0000AC 04C0          201     orc.b #H'c0,ccr
202 0000AE 5470          202     rts
203                  203
204                  204    ;---------------------------------------------------------
205 000000               205     .SECTION D,DATA
206                  206
207 000000               207     .SECTION B,DATA
208 000000 00000002         208    DATA_END: .RES.W 1
209                  209
210                  210     .END
*****TOTAL ERRORS      0
*****TOTAL WARNINGS    0
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***     04/07/18 13:06:57
PAGE    5
```

*** CROSS REFERENCE LIST

NAME SECTION ATTR VALUE SEQUENCE

B B SCT 00000000 207*

D D SCT 00000000 205*

DATA_END B 00000000 142 208*

IRQ0 V 00000030 33*

IRQ1 V 00000034 35*

IRQ2 V 00000038 37*

IRQ3 V 0000003C 39*

IRQ4 V 00000040 41*

IRQ5 V 00000044 43*

P P SCT 00000000 132*

V V SCT 00000000 4*

_DisableInterrupt P EXPT 000000AC 196 200*

_EnableInterrupt P EXPT 000000A8 196 197*

_INT_ADI V 000000F0 129*

_INT_CMI V 00000054 51*

_INT_DEND0A V 000000B0 97*

_INT_DEND0B V 000000B4 99*

_INT_DEND1A V 000000B8 101*

_INT_DEND1B V 000000BC 103*

_INT_ERI0 V 000000D0 113*

_INT_ERI1 V 000000E0 121*

_INT_IMIA0 V 00000060 57*

_INT_IMIA1 V 00000070 65*

```
_INT_IMIA2              V       00000080  73*
_INT_IMIA3              V       00000090  81*
_INT_IMIA4              V       000000A0  89*
_INT_IMIB0              V       00000064  59*
_INT_IMIB1              V       00000074  67*
_INT_IMIB2              V       00000084  75*
_INT_IMIB3              V       00000094  83*
_INT_IMIB4              V       000000A4  91*
_INT_NMI                V       0000001C  23*
_INT_OVI0               V       00000068  61*
_INT_OVI1               V       00000078  69*
_INT_OVI2               V       00000088  77*
_INT_OVI3               V       00000098  85*
_INT_OVI4               V       000000A8  93*
_INT_RXI0               V       000000D4  115*
_INT_RXI1               V       000000E4  123*
_INT_Reserved1          V       00000004  11*
_INT_Reserved18         V       00000048  45*
_INT_Reserved19         V       0000004C  47*
_INT_Reserved2          V       00000008  13*
_INT_Reserved22         V       00000058  53*
_INT_Reserved23         V       0000005C  55*
_INT_Reserved27         V       0000006C  63*
_INT_Reserved3          V       0000000C  15*
_INT_Reserved31         V       0000007C  71*
_INT_Reserved35         V       0000008C  79*
_INT_Reserved39         V       0000009C  87*
_INT_Reserved4          V       00000010  17*
_INT_Reserved43         V       000000AC  95*
```

_INT_Reserved48 V 000000C0 105*

_INT_Reserved49 V 000000C4 107*

*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 *** 04/07/18 13:06:57

PAGE 6

*** CROSS REFERENCE LIST

NAME SECTION ATTR VALUE SEQUENCE

_INT_Reserved5 V 00000014 19*

_INT_Reserved50 V 000000C8 109*

_INT_Reserved51 V 000000CC 111*

_INT_Reserved6 V 00000018 21*

_INT_TEI0 V 000000DC 119*

_INT_TEI1 V 000000EC 127*

_INT_TRAP1 V 00000020 25*

_INT_TRAP2 V 00000024 27*

_INT_TRAP3 V 00000028 29*

_INT_TRAP4 V 0000002C 31*

_INT_TXI0 V 000000D8 117*

_INT_TXI1 V 000000E8 125*

_INT_WOVI V 00000050 49*

_ITU_OVI_0 P 0000006A 61 175*

_InterruptITU0 IMPT 00000000 7 183

_main IMPT 00000000 5 151

_start P 00000000 9 133*

_usb_int IMPT 00000000 6 165

init_end P 00000026 145 150*

init_loop P 00000018 143* 149

```
int_error               P       0000002A   11   13   15   17   19   21   23   25   27   29   31   33
                                   35   37   39   41   45   47   49   51   53   55   57   59
                                   63   65   67   69   71   73   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  154*
usb_interrupt           P       0000002C   43  157*
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***    04/07/18 13:06:57
PAGE    7

*** 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
S1130000000002000000022A0000022A0000022A67
S1130010000022A0000022A0000022A0000022A2D
S1130020000022A0000022A0000022A0000022A1D
S10B00300000022A0000022A6D
S1130038000022A0000022A0000022A0000022C03
S1130048000022A0000022A0000022A0000022AF5
S1130058000022A0000022A0000022A0000022AE5
S10B00680000026A0000022AF5
S1130070000022A0000022A0000022A0000022ACD
S1130080000022A0000022A0000022A0000022ABD
S1130090000022A0000022A0000022A0000022AAD
S10B00A00000022A0000022AFD
S11300A80000022A0000022A0000022A0000022A95
S11300B80000022A0000022A0000022A0000022A85
S11300C80000022A0000022A0000022A0000022A75
S10B00D80000022A0000022AC5
S11300E00000022A0000022A0000022A0000022A5D
S10700F00000022ADD
S11302007A0700FFFF107A00000099C07A0100FF0F
S1130210E0007A0200FFE0101F92587000086C0B98
S1130220689B0B7140F25E0002B0567001006DF0E6
S113023001006DF101006DF201006DF301006DF439
S113024001006DF501006DF65E004F4001006D7613
S113025001006D7501006D7401006D7301006D7215
S113026001006D7101006D70567001006DF00100A9
S11302706DF101006DF201006DF301006DF40100F9
S11302806DF501006DF65E00492001006D760100F9
S11302906D7501006D7401006D7301006D720100D5
S11302A06D7101006D705670063F547004C0547038
S11302B05E0060567A370000000A790B000119339B
S11302C00FF47A0600FFE1E019550B5579257FFFFE
S11302D04DF85C000FCA5E004ABC5E004C3C0D3812
S11302E00D305C000F460D380DB05C000F3E0D382D
S11302F0790000025C000F340D38790000035C00C4
S11303000F2A5E004DE27FF472507FF570505E005D
S113031002A80D3228F117506DF07A0000009A0000
S113032001006DF05E004B0C0B970B877A00000009
S11303309A0F01006DF05E004D100B9718886EC880
S113034000036EC800026EC8000168C87A0000FF8F
S1130350E04A5E003EB65E00494C7A00000007D0DA
S11303605E0044147900001E5E00450A01006FF030
S1130370000047900001F5E00450A0F8501006F70BD
S113038000045E0046D60FD05E0046D601006B2007
S113039000009C8A01006DF001006B2000009C8628
S11303A001006DF05E0049480B970B975E004722F2
S11303B00D0046D85E0049667A0100009A1B0D3095
S11303C05E00498E7A00000007D05E0044145E0090
S11303D0496619550D505C000EA00D0D0D5117F116
S11303E00AC16819D90117D1660147540DB80D50D8
S11303F05C000E380D5009B06DF07A0000009A2CA5
S113040001006DF00FE05E0058F80B970B870100B9
S11304106DF67A0000009A3101006DF05E004B0C1E
S11304200B970B970FE05E00599C09B00D010FE08D
S11304305E00577401006DF65E004D100B97400887
S11304400D380D505C000DE40DD017F50FC10AD126
S113045068980B557925000458D0FF785E004AFC54

S10804600D0047165ECC
S1130465004AEC17D00D055E004CC068ED0DB10FC9
S1130475E05E0057745E0058960D00473879010019
S1130485400FE05E0058380D057A0100009A3501EA
S1130495006DF15E004D100B9701006DF65E004D8A
S11304A5100B9701006DF65E004B0C0B970D510F6A
S11304B5E05E0057740D28790000035C000D680D9C
S11304C520795000010D0219550B55792527104D3B
S11304D5F85A0003D254705E0060567A0500FFE0B7
S11304E51219665E00496619EE400E7A0100009AFC
S11304F5370D605E00498E0B5E69501D0E4DEC7A1B
S11005050100009A390D605E00498E7A01F5
S113051200009A3D0D605E00498E19EE400E7A018D
S113052200009A370D605E00498E0B5E6F50000229
S11305321D0E4DEA7A0100009A3F0D605E00498E5E
S11305425E00603454705E0060567A370000000A21
S11305527A03000000017A04000007D019550F86C0
S113056218886EF800036EF800026EF8000168F84E
S1130572696079200001463 65C00FF5E7A0000FF65
S1130582E01269010B5169815E0058BE17F07902CE
S1130592000901D053207918000A790000645280BF
S11305A217F00F810FE05E0044A65A000E966960B1
S11305B27920000246365C00FF207A0000FFE01437
S11305C269010B5169815E0058BE17F07902000977
S11305D201D053207918000A79000064528017F081
S11305E20F810FE05E0044A65A000E9669607920DF
S11305F2000B586000B601006F6000127A20000001
S1080602000146205E2B
S10806070049667A01C1
S113060C00009A430D505E00498E7A010000076C7E
S113061C0FE05E0044EA5A000E9601006F60001270
S113062C7A200000000246265E0049667A0100002B
S113063C9A540D505E00498E7A0100009A5B0D505E
S113064C5E00498E0FE05E0047045A000E960100CF
S113065C6F6000127A2000000003461E5E0049669C
S113066C7A0100009A660D505E00498E7A010000F3
S113067C05DC0FE05E0044EA402401006F600012C9
S111068C7A200000000446165E0049667A01DB
S113069A00009A770D505E00498E0FE05E00470412
S11306AA5A000E9669607920000C586000A8010070
S11306BA6F6000127A200000000146205E0049663E
S11306CA7A0100009A880D505E00498E7A01000073
S11306DA0D480FE05E0044EA5A000E9601006F606F
S11306EA00127A200000000246205E0049667A0161
S11306FA00009A990D505E00498E7A0100000D4858
S113070A0FE05E0044EA5A000E9601006F60001281
S113071A7A2000000003461E5E0049667A01000043
S108072A9AAA0D505EC8
S113072F00498E7A0100000D480FE05E0044EA4055
S113073F1C5E0049667A0100009ABB0D505E0049AA
S113074F8E0FE05E0047040FE05E0046A25A000ED4
S113075F9669607920000D586000A801006F600052
S113076F127A200000000146205E0049667A0100DC
S113077F009AC30D505E00498E7A01000013EC0FEF
S113078FE05E0044EA5A000E9601006F6000127A91
S113079F200000000246205E0049667A0100009A9D
S11307AFD40D505E00498E7A01000013EC0FE05E0A

S10707BF0044EA5AAB
S11307C3000E9601006F6000127A200000000346BA
S11307D31E5E0049667A0100009AE50D505E0049EA
S11307E38E7A01000013EC0FE05E0044EA401C5EC6
S11307F30049667A0100009AF60D505E00498E0F98
S1130803E05E0047040FE05E0046A25A000E9669BD
S1130813607920000E586000E401006F6000127AD3
S113082320000000146205E0049667A0100009A19
S1130833FE0D505E00498E7A01000017700FE05ED3
S11308430044EA5A000E9601006F6000127A2000FA
S10E085300000246405E0049667A0187
S113085E00009B0F0D505E00498E7A010000177049
S113086E0FE05E0044EA7A0100009B160D505E0015
S113087E498E7900000B5E00479C0F867A010000BB
S113088E05DC5E0044EA5A000E9601006F6000120A
S113089E7A2000000003461E5E0049667A010000BE
S11308AE9B210D505E00498E7A01000017700FE0F8
S11308BE5E0044EA403801006F6000127A200000A7
S11308CE0004462A7900000B5E0047620F8247063A
S11308DE0FA05E0046A27A0100009B320D505E000F
S11208EE498E0FE05E0047040FE05E0046A25AFA
S11308FD000E9669607920001346440FB10FE05E38
S113090D0044A619DD0DD05C0009620DD117F10F5E
S113091DF20A92682ADA0117D2660247160DD079C8
S113092D1000145E0047DA7A01000003E80FE05E61
S113093D0044A60B5D792D00044DCA5A000E96692D
S113094D607920001446187A0100009B390D505E22
S113095D00498E0FC10FE05E0044EA5A000E9669FE
S113096D607920001546187A0100009B3B0D505EFF
S113097D00498E0FC10FE05E0044EA5A000E9669DE
S113098D607920001646187A0100009B3D0D505EDC
S10F099D00498E0FC10FE05E0044EA5ACF
S11309A9000E9669607920001746187A0100009BAA
S11309B93F0D505E00498E0FC10FE05E0044EA5AB5
S11309C9000E9669607920001E465E01006F600083
S11309D912462401006F6000120B7001006FE000E2
S11309E91228D6E80E38D67A01000003E80FE05E34
S11309F90044A65A000E9601006F6000127A200087
S1130A0900000158600486010006F6000121B700129
S1130A19006FE000127FD670007A01000003E80F2F
S1130A29E05E0044A65A000E9669607920001F58BB
S1130A396003D801006F600012460C1A910FE05E43
S1130A490044EA5A000E9601006F6000127A2000F2
S1130A5900000146247A0100009A3D0D505E0049C9
S1130A698E7A0100009B410D505E00498E0FC10F24
S1090A79E05E0044EA5AAE
S1130A7F000E9601006F6000127A200000000246FC
S1130A8F3019DD0DD00B505E00450A0DD217F21051
S1130A9F321032010078A06BA000FFE01A0B5D79D2
S1130AAF2D00024DDE0FB10FE05E0044EA5A000E37
S1130ABF1001006F6000127A200000000346566B8E
S1130ACF2000FFE0127920000D4D1A790000016B11
S1130ADFA000FFE01601006F6000120B7001006FA2
S1130AEFE0001240246B2000FFE0147920000D4D2D
S1130AFF18790000026BA000FFE01601006F600081
S1130B0F120B7001006FE000120FB10FE05E004493
S1130B1FA65A000E1001006F6000127A2000000029

S1130B2F04465A5E0049666B2000FFE016792000E9
S1130B3F01460E7A0100009B520D505E00498E4014
S1060B4F186B20FD
S1130B5200FFE01679200002460C7A0100009B6335
S1130B620D505E00498E01006B2000FFE01A5E000B
S1130B7246A201006B2000FFE01E5E0046A20FC1E9
S1130B820FE05E0044EA5A000E1001006F6000128B
S1130B927A2000000005461C5E0049667A010000C7
S1130BA29A1B0D505E00498E0FC10FE05E0044EAAE
S1130BB25A000E1001006F6000127A200000000636
S10B0BC2461C5E0049667A013E
S1130BCA00009B740D505E00498E0FC10FE05E005A
S1130BDA44EA5A000E1001006F6000127A200000E6
S1130BEA000746365E00496619DD0DD07910000B01
S1130BFA5E00450A0DD217F210321032010078A0B6
S1130C0A6BA000FFE0220B5D792D00044DDC0FB1D0
S1130C1A0FE05E0044EA5A000E1001006F600012F2
S1130C2A7A200000000846245E00473C792000022F
S1130C3A460E01006F6000120B7001006FE0001294
S1130C4A0FB10FE05E0044A65A000E1001006F6058
S1130C5A00127A2000000009460C0FC10FE05E0063
S1130C6A44EA5A000E1001006F6000127A20000055
S1130C7A000A461C5E0049667A0100009A1B0D5061
S1040C8A5E08
S1130C8B00498E0FC10FE05E0044EA5A000E1001BB
S1130C9B006F6000127A200000000B461C5E0049B7
S1130CAB667A0100009B850D505E00498E0FC10FC4
S1130CBBE05E0044EA5A000E1001006F6000127AE6
S1130CCB200000000C4634790000135E00450A0136
S1130CDB006BA000FFE0325E0046D6790000145E85
S1130CEB00450A01006BA000FFE0365E0046D60FFD
S1130CFBB10FE05E0044EA5A000E1001006F600072
S1130D0B127A200000000D46305E00473C7920002C
S10A0D1B03461A01006B20DF
S1130D2200FFE0325E0046A201006F6000120B700A
S1130D3201006FE000120FB10FE05E0044A65A00FB
S1130D420E1001006F6000127A200000000E460CA4
S1130D520FC10FE05E0044EA5A000E1001006F60FB
S1130D6200127A200000000F461C5E0049667A01D9
S1130D7200009A1B0D505E00498E0FC10FE05E000A
S1130D8244EA5A000E1001006F6000127A2000003C
S1130D920010461A5E0049667A0100009B960D50C8
S1130DA25E00498E0FC10FE05E0044EA406001001D
S1130DB26F6000127A20000001146527900002968
S10B0DC25E00450A01006BA06D
S1130DCA00FFE03A7900002A5E00450A01006BA0A1
S1130DDA00FFE03E7900002B5E00450A01006BA08C
S1130DEA00FFE0427900002C5E00450A01006BA077
S1130DFA00FFE0460FE05E0046A27900001E5E0097
S1130E0A47625E0046A25A000E966960792000295D
S1130E1A461A7A01000000640FE05E0044A60FE15F
S1130E2A7A0000FFE04E5E00260E4060696079207A
S1130E3A002A461A7A01000000640FE05E0044A605
S10C0E4A0FE17A0000FFE08C5E69
S1130E53001C9A403E69607920002B461A7A0100F0
S1130E6300006E0FE05E0044A60FE17A0000FFE18D
S1130E73845E0019DC401C69607920002C46140F42

S1130E83C10FE05E0044A60FE17A0000FFE1985E24
S1130E930013B67A170000000A5E00603454705ED4
S1130EA30060560F86790400097A0500FFE0126992
S1130EB3607920000014626190069D05E0058BE17E9
S1130EC3F001D053407918000A7900001E528017AD
S1130ED3F00F810FE05E0044A65A00113069607978
S1130EE3200002462819006FD000025E0058BE1787
S1130EF3F001D053407918000A7900001E5280177D
S1130F03F00F810FE05E0044A65A001130696C793B
S1130F132C00034D36792C00084E3069601B5017A3
S1130F23F010300A85190069D05E0058BE17F0012E
S1130F33D053407918000A790000C8528017F00F84
S1070F43810FE05ED9
S1070F470044A65A5F
S1130F4B0011307A040000498E195569607920002D
S1130F5B0B461A7A0100009BA70D505D407A0100E6
S1130F6B0005DC0FE05E0044EA5A0011306960793A
S1130F7B20000C461A7A0100009BAF0D505D407A9E
S1130F8B0100000BB80FE05E0044EA5A0011306910
S1130F9B607920000D46247A0100009A3D0D505DC7
S1130FAB407A0100009BB90D505D407A010000119E
S1130FBB940FE05E0044EA5A001130696079200017
S1130FCB0E461A7A0100009BC10D505D407A010059
S1130FDB0017700FE05E0044EA5A0011307A0300E9
S10F0FEB0007D069607920001446245EE2
S1130FF70049667A0100009BCB0D505D407A0100E2
S1131007009A3D0D505D400FB10FE05E0044EA5A70
S113101700113069607920001546245E0049667A1D
S11310270100009BDC0D505D407A0100009A3D0DE5
S1131037505D400FB10FE05E0044EA5A001130697A
S113104760792000164624 5E0049667A0100009BFA
S1131057ED0D505D407A0100009A3D0D505D400F44
S1131067B10FE05E0044EA5A00113069607920004D
S10710771746245E93
S113107B0049667A0100009BFE0D505D407A01002A
S113108B009A3D0D505D400FB10FE05E0044EA5AEC
S113109B00113069607920001E460AF8FF38D438F6
S11310ABD65A001130696079200029461 87A01005D
S11310BB0000640FE05E0044A67A0000FFE04E5E82
S11310CB0024E2406069607920002A461A7A010005
S11310DB0000640FE05E0044A60FE17A0000FFE01E
S11310EB8C5E001AFA403E69607920002B461A7A0F
S11310FB010000006E0FE05E0044A60FE17A0000D2
S113110BFFE1845E00199E401C69607920002C4628
S110111B140FB10FE05E0044A60FE17A004F
S113112800FFE1985E0012CC5E00603454705E00EC
S113113860567A040000498E19660F856950792034
S1131148000B46105E0049667A0100009C0F0D6093
S11311585D40404469507920000C460C7A01000038
S11311689C1A0D605D40403069507920000D460C93
S11311787A0100009C240D605D40401C6950792071
S1131188000E46147A0100009A3D0D605D407A0115
S113119800009C2E0D605D40695079200014461AAA
S11311A85E0049667A0100009C3F0D605D407A014C
S11311B800009A3D0D605D40406469507920001538
S10B11C8461A5E0049667A0134
S11311D000009C500D605D407A0100009A3D0D6057

S11311E05D4040426950792000164 61A5E00496608
S11311F07A0100009C610D605D407A0100009A3D18
S11312000D605D4040206955792500174 6185E0042
S113121049667A0100009C730D605D407A0100000D
S11312209A3D0D605D405E00603454706DF66DF55F
S11312300D067909000128D617500D950CE91A09F6
S11312404B04101540F866050D8846121A0E4B0420
S1131250101940F80D9029D6148939D640141A0E66
S11312604B04101940F8795900FF0D9029D61689BF
S113127039D60D506D756D7654706DF60D0628D305
S1131280175079010001 1A0E4B04101140F8661033
S1131290470419114004790100010D106D76547053
S11312A0F80638ECF8FF38C038C1188838D8F8FF8A
S11312B038C8188838DBF8FF38C9F8DF38D0F8FFAA
S10912C038CDF8F038D12F
S10912C6F8FF38D4547058
S1139A00435055204D4F444520253032580A000C11
S1139A1052656164792133303532004E4558542004
S1139A202020202020202020202020000737725754F
S1139A300025730A000C0020003C313E000A003C64
S1139A40323E003C313E3173742020202020202000
S1139A50202020003C313E326E64003C313E537482
S1139A606F70202020003C313E3372642020202080
S1139A70202020202020003C313E53746F70202092
S1139A802020202020202020003C323E3173742020EF
S1139A9020202020202020202020003C323E326E6420F3
S1139AA02020202020202020202020003C323E337264DE
S1139AB0202020202020202020202020003C323E5374F0
S1139AC06F70003C333E3173742020202020202020200F
S1139AD0202020003C333E326E64202020202020B2
S1139AE020202020003C333E33726420202020209D
S1099AF0202020202000CD
S1139AF63C333E53746F70003C343E317374202004
S1139B062020202020202020202020003C343E326E64009A
S1139B163C313E53746172742020003C343E3372F0
S1139B2664202020202020202020202020003C343E5387
S1139B36746F00300031003200330054687265617F
S1139B4664205265616479204 74F2100474F414C99
S1139B56213C313E574F4E202020202000474F41C5
S1139B664C213C323E574F4E202020202000436F8D
S1139B76756E74557020202020202020202000544C
S1139B866F67676C652020202020202020202020007E
S1139B9648656C6C6F2045562020202020202020200D
S1139BA6003C313E496E6974003C323E496E69742D
S1139BB62020003C333E496E6974003C343E496EB6
S1139BC669742020003C303E496E697420202020B1
S1139BD62020202020003C313E496E69742020203D
S1099BE6202020202020B6
S1139BEC003C323E496E6974202020202020202026
S1139BFC20003C333E496E697420202020202020 15
S1139C0C2020003C313E44657374726F79003C3202
S1139C1C3E44657374726F003C333E4465737472D7
S1139C2C6F003C343E44657374726F79202020209E
S1139C3C2020003C303E44657374726F79202020E1
S1139C4C202020003C313E44657374726F792020D0
S1139C5C20202020003C323E44657374726F7920BF
S1139C6C202020202020003C333E44657374726F07

S1139C7C79202020202020200000415312D0000006
S1059C8C0000D3
S11312CC5E0060560F860F957A00000000200AE03E
S11312DC0FD15E0042AA7A00000000200AE05E00F3
S11312EC42EE01006FE600027A00000000060AE0FD
S11312FC01006FE000087A000000000C0AE0010016
S113130C6FE0000E7A00000000120AE001006FE0AB
S113131C001C18886EE8001A7A0500003F0A7A0050
S113132C000000380AE05D507A000000003C0AE03F
S113133C5D507A00000000400AE05D507A00000026
S113134C00440AE05D5001006F60000201006DF083
S113135C7A00000000400AE07A0100009C8E5E00D7
S113136C3F9E0B9779200001473A790000096DF0F5
S113137C01006F60001C01006DF07A000000004456
S113138C0AE07A0100009C995E003FF80B970B87EB
S113139C79200001470E7A01000000120AE1686807
S11313AC5E002E045E00603454705E0060560F863F
S10913BC0F9501006F60B4
S11313C2000E01006DF07A000000003C0AE07A0191
S11313D200009CA45E003F9E0B976E68000CA871F0
S11313E2460E5E0049660FD05E0046A25A0017867B
S11313F201006F60000201006DF07A0000000040FE
S11314020AE07A0100009C8E5E003F9E0B977900F2
S113141200096DF001006F60001C01006DF07A009D
S10B1422000000440AE07A0116
S113142A00009C995E003FF80B970B8701006F60E1
S113143A000801006DF07A00000000380AE07A0122
S113144A00009CB15E003F9E0B976E680006A8736E
S113145A461CF84E6DF07A000000003C0AE07A015F
S113146A00009CA45E003F140B875A001778686833
S113147AA87546366E680012A879587002D06E684D
S113148A0013A879460AF86E6EE800135A00175833
S113149A6E680014A86E470E6E680015A86E46069D
S11314AAF8796EE800155A0017586868A855462E49
S11314BA6E680012A879587002946E680013A879AE
S11314CA5870028A6E680014A86E4608F8796EE8A6
S11314DA001440066E680015A86E5A00175868680B
S11314EAA86A46466E680012A879460AF86E6EE83C
S11314FA00125A0017586E680013A879460AF86E44
S113150A6EE800135A0017586E680014A86E46084E
S109151AF8796EE80014ED
S1131520400E6E680015A86E4606F8796EE8001541
S11315305A0017586868A86446366E680015A8797B
S1131540587002146E680014A879460AF86E6EE8A3
S113155000145A0017586E680013A86E470E6E6881
S11015600012A86E4606F8796EE800125AD4
S113156D0017586868A844462E6E680015A8795868
S113157D7001D86E680014A879587001CE6E68009A
S113158D13A86E4608F8796EE8001340066E6800DE
S113159D12A86E5A0017586868A86B46466E680005
S11315AD15A879460AF86E6EE800155A0017586E9D
S11315BD680014A879460AF86E6EE800145A0017ED
S11315CD586E680013A86E4608F8796EE80013404C
S11315DD0E6E680012A86E4606F8796EE800125A70
S11315ED0017586868A86F46366E680016A87958B4
S11315FD7001586E680017A879460AF86E6EE800F8
S113160D175A0017586E680018A86E470E6E6800BB

S113161D19A86E4606F8796EE800195A001758682E
S113162D68A84F462E6E680016A8795870011C6E77
S113163D680017A879587001126E680018A86E46D5
S113164D08F8796EE8001840066E680019A86E5AFE
S109165D0017586868A89D
S11316636846466E680016A879460AF86E6EE80067
S1131673165A0017586E680017A879460AF86E6E53
S1131683E800175A0017586E680018A86E4608F842
S1131693796EE80018400E6E680019A86E4606F8C6
S11316A3796EE800195A0017586868A86346346EC0
S11316B3680019A8795870009C6E680018A87946C9
S11316C30AF86E6EE800185A0017586E680017A8D8
S11316D36E470E6E680016A86E4606F8796EE8002C
S11316E31640726868A84346286E680019A87947AC
S11316F3646E680018A879475C6E680017A86E4685
S113170308F8796EE8001740066E680016A86E4065
S1131713446868A874463E6E680019A8794608F8B9
S11317236E6EE80019402E6E680018A8794608F813
S11317336E6EE80018401E6E680017A86E4608F820
S1131743796EE80017400E6E680016A86E4606F819
S1081753796EE80016A9
S11317587A00000000120AE001006DF07A00000030
S10E176800440AE07A0100009C995E37
S1131773003F660B977A01000000120AE168685E76
S10C1783002E045E006034547072
S1139C8E4D6F746F722E74787400004C696D697425
S1139C9E2E7478740000436F6D6D616E642E74784C
S1129CAE7400005361666574792E747874000036
S113178C0D00460828D6E80738D6400A7920000110
S113179C46047FD670300D88460828D6E80B38D619
S11317AC400A7928000146047FD670200D114608A3
S11317BC28D6E80D38D6400A7921000146047FD695
S11317CC70100D99460828D6E80E38D6547079292E
S11317DC000146047FD67000547028D6E80738D62B
S11317EC28D6E80B38D628D6E80D38D628D6E80EF6
S11317FC38D654705E0060560F850D1C0D94790E0F
S113180C000119660DC017F001006F5100121F8102
S113181C461E0D690D610D680D405C00FF620100F1
S113182C6F5000120B7001006FD000120D6E402E22
S113183C0B5C0DC017F001006F5100121F81461E87
S113184C0D69790100010D680D405C00FF32010048
S113185C6F5000120B7001006FD000120D6E0DE073
S113186C5E00603454705E0060560F850D1C0D9441
S109187C790E000119665C
S11318820DC017F001006F5100121F81461E0D6932
S11318920D610D480D605C00FEF001006F500012F7
S11318A20B7001006FD000120D6E402E0B5C0DC049
S11318B217F001006F5100121F81461E79090001C2
S11318C20D610D480D605C00FEC001006F500012F7
S11318D20B7001006FD000120D6E0DE05E006034DC
S11318E254705E0060560F850F966F7900181911B8
S11318F20FE05C00FF080D005870009A6F79001A20
S1131902790100020FE05C00FEF40D0058700086BE
S11319126F79001C790100040FE05C00FEE00D000A
S1131922477746F79001E790100060FE05C00FECE5A
S11319320D0047626F790020790100080FE05C0017
S1131942FF2E0D0047506F7900227901000A0FE044

S11319525C00FF1C0D00473E6F7900247901000CE7
S11319620FE05C00FF0A0D00472C6F790026790116
S1091972000E0FE05C0013
S1131978FEF80D00471A5C00FE641A8001006FE050
S113198800126E580006A87146060FE05E0046A2D4
S11319985E006034547001006DF60F8601006FE637
S11319A800027A00000000060AE001006FE0000868
S11319B87A00000000100AE05E003F0A7A00000087
S11319C8000C0AE05E003F0AF8FF38D438D601005D
S11319D86D7654705E0060560F850F9401006F405A
S11319E8001246307A000000000C0AD001006F5143
S11319F800085E0040C001006F50000201006DF056
S1131A087A00000000100AD07A0100009CBE5E0034
S1131A183F9E0B9719EE790600016858A843587042
S1131A2800ACA8444758A84F4770A8554738A8633F
S1131A38477CA8644740A868474CA86A471CA86B1A
S1131A48472CA86F474AA873470CA8744756A8752C
S1131A58470E5A001AF46DFE40026DF66DFE400AF9
S1091A686DFE6DF6400463
S1131A6E6DF66DF66DFE401E6DFE6DFE6DF6401647
S1131A7E6DF66DFE6DF6400E6DFE6DF66DF640065F
S1131A8E6DF66DF66DF66DFE40266DFE6DFE6DFE0A
S1131A9E6DF6401C6DF66DFE6DFE6DF640126DFE1D
S1131AAE6DF66DFE6DF640086DF66DF66DFE6DF618
S1131ABE6DF66DF66DF66DFE0FC10FD05C00FE1662
S1131ACE7A1700000010401E6DFE6DFE6DF66DF66A
S1131ADE6DF66DF66DF66DFE0FC10FD05C00FDF663
S10F1AEE7A17000000105E006034547092
S10E9CBE4D6F746F722E7478740000F9
S1131AFA5E0060560F860F9501006FE60024010011
S1131B0A6F6000245E00366E7A00000000280AE047
S1131B1A5E0036E27A00000000300AE05E003942D5
S1131B2A7A00000000380AE05E003BA27A00000057
S1131B3A004C0AE001006FE000705E002E067A0096
S1131B4A000000740AE05E002E7A7A000000007C2E
S1131B5A0AE05E0030DA7A00000000840AE05E00E0
S1131B6A333A7A00000000A20AE00FD15E0042AACB
S1131B7A7A0500003F0A7A00000000E00AE05D509F
S1131B8A7A00000000E40AE05D507A00000000E8F1
S1131B9A0AE05D507A00000000EC0AE05D507A002A
S1131BAA000000F00AE05D507A00000000F40AE049
S1131BBA5D507A00000000BA0AE001006FE000BC41
S1131BCA7A00000000C00AE001006FE000CA7A0050
S1131BDA000000CE0AE001006FE000D07A000000A6
S1071BEA00D40AE036
S1131BEE01006FE000D67A00000000DA0AE001007F
S1131BFE6FE000DC7A00000000F40AE0F9735E0087
S1131C0E41E07A00000000380AE001006F61002411
S1131C1E5E003BBC7A00000000840AE001006F61A5
S1131C2E00705E003362790000096DF001006F6091
S1131C3E00CA01006DF07A00000000E40AE07A01A8
S1131C4E00009CCA5E003FF80B970B877A000000DA
S1131C5E00E80AE0F94E5E00410C792000014726A8
S1131C6E7A00000000EC0AE0F9635E00407C792004
S1131C7E000147127A00000000F00AE0F94E5E0000
S1131C8E419C792000015E00603454705E00605602
S1131C9E790D0001F463FC4E19550F860F93010065

S1131CAE6F6000BC01006DF07A00000000E00AE0F6
S1051CBE7A01A6
S1131CC000009CD55E003F9E0B9779200001587061
S1131CD0079E790000096DF001006F6000CA0100E2
S1131CE06DF07A00000000E40AE07A0100009CCA6B
S1131CF05E003FF80B970B877A00000000E80AE0CC
S1131D0001006F6100D05E0040C07920000158706F
S1131D10075E7A00000000EC0AE001006F6100D664
S1131D205E00403079200001587007447A000000BB
S1131D3000F00AE001006F6100DC5E004150792091
S1131D4000015870072A6E6800CEA8485870051025
S1131D50A8594762A86358700458A86458700222AF
S1131D60A868587005C4A86F58700398A8714718DD
S1131D70A8725870067EA8734726A875475EA8798F
S1131D80587001D85A0023F47A00000000F40AE0E6
S1131D90F9735E0041E00FB05E0046A25A0023F4DF
S1131DA07A00000000A20AE00D515E0043725A005F
S1091DB023F45A0023F4A2
S1131DB601006F60002401006F00000C69007920A8
S1131DC60001461201006F60007001006F00000CF5
S1131DD669015870061801006F60002401006F0046
S1131DE600146900792000014640100 6F600070FF
S1131DF601006F000014690079200001586003 4A4E
S1131E067A00000000EC0AE00C495E00407C7A0090
S1131E16000000A20AE00DD15E0043727A000000C2
S1131E2600E80AE00CC95E00410C5E0049667A01CF
S1131E3600009CE15A0023A201006F6000700100BC
S1131E466F00000C69007920000146767A000000D5
S1131E5600A20AE00D515E0043727A00000000A260
S1131E660AE05E0042EE7A00000000F00AE00CC9C8
S1131E765E00419C7A00000000380AE07A370000D1
S1131E8600140FF17A02000000145E005FFE7A0070
S1131E96000000280AE07A37000000080FF17A02F2
S1131EA6000000085E005FFE01006F6100BC0100D8
S1131EB66F6000245E003E267A170000001C5A005D
S1131EC623F401006F60007001006F00000C6901CC
S1131ED646667A00000000A20AE00D515E004372D6
S1131EE67A00000000A20AE05E0042EE7A000000DB
S1131EF600840AE07A370000001E0FF17A02000020
S1131F06001E5E005FFE7A000000007C0AE07A375E
S1131F16000000080FF17A02000000085E005FFE71
S1091F2601006F6100BC25
S1131F2C01006F6000705E0036267A1700000026F1
S1131F3C401A7A00000000A20AE00D515E004372C1
S1131F4C7A00000000A20AE05E0042EE5A0023F47D
S1131F5C01006F60002401006F00001469007920F8
S1131F6C0001461201006F60007001006F00000C4D
S1131F7C69015870047201006F60002401006F0046
S1131F8C000C690079200001465001006F6000705D
S1131F9C01006F000014690079200001463 87A00B3
S1131FAC000000EC0AE00C495E00407C7A00000063
S10A1FBC00A20AE00DD15E53
S1131FC30043727A00000000E80AE00CC95E004196
S1131FD30C5E0049667A0100009CE15A0023A25A71
S1131FE300215001006F60007001006F00000C6955
S1131FF3007920000146767A00000000A20AE00D72
S1132003515E0043727A00000000A20AE05E0042C0

S1132013EE7A00000000F00AE00CC95E00419C7AEE
S1132023000000000380AE07A37000000140FF17A49
S1132033020000000145E005FFE7A00000000300A15
S1132043E07A37000000080FF17A02000000085E0F
S1132053005FFE01006F6100BC01006F6000245E3E
S1132063003E6E7A170000001C5A0023F401006F30
S1132073600070001006F00000C690146667A00007E
S11320830000A20AE00D515E0043727A00000000D3
S1132093A20AE05E0042EE7A00000000840AE07ABE
S11320A3370000001E0FF17A020000001E5E005F7E
S10620B3FE7A00AF
S11320B60000007C0AE07A37000000080FF17A027C
S11320C6000000085E005FFE01006F6100BC0100B6
S11320D66F6000705E0036267A1700000026401AED
S10E20E67A00000000A20AE00D515E2A
S11320F10043727A00000000A20AE05E0042EE5A39
S11321010023F401006F60007001006F0000146987
S113211100792000146387A00000000EC0AE00C47
S1132121495E00407C7A00000000A20AE00DD15E06
S11321310043727A00000000E80AE00CC95E004126
S11321410C5E0049667A0100009CE15A0023A27AE1
S113215100000000A20AE00D515E0043727A000004
S11321610000A20AE05E0042EE7A00000000840A49
S1132171E07A370000001E0FF17A020000001E5EB4
S1132181005FFE7A00000000740AE07A3700000065
S1132191080FF17A02000000085E005FFE01006F84
S11321A16100BC01006F6000705E0035DE5A0023E0
S11321B1207A00000000A20AE00D515E0043727A0A
S11321C100000000A20AE05E0042EE01006F600021
S11321D17001006F00000C69007920000146387A14
S10821E100000000F006
S11321E60AE00CC95E00419C7A00000000E80AE0A0
S10621F60CC95EB0
S11321F900410C7A00000000EC0AE00C495E004043
S11322097C5E0049667A0100009CE15A0023A27AA8
S113221900000000840AE07A370000001E0FF17AFB
S1132229020000001E5E005FFE7A000000007C0AC7
S1132239E07A37000000080FF17A02000000085E17
S1132249005FFE01006F6100BC01006F6000705EFA
S11322590036265A00232001006F60002401006F15
S113226900001469007920000015860017E7A00009A
S11322790000A20AE00D515E0043727A00000000DB
S1132289A20AE05E0042EE01006F60007001006F78
S113229900000C69007920000146387A000000002B
S11322A9E80AE00CC95E00410C7A00000000F00A5C
S11322B9E00CC95E00419C7A00000000EC0AE00CC6
S11322C9495E00407C5E0049667A0100009CE15A40
S11322D90023A27A00000000840AE07A3700000094
S10822E91E0FF17A0253
S11322EE0000001E5E005FFE7A000000007C0AE024
S11222FE7A37000000080FF17A02000000085E33
S113230D005FFE01006F6100BC01006F6000705E35
S113231D0036267A17000000265A0023F401006FB9
S113232D60002401006F00000C6900792000015842
S113233D6000B47A00000000A20AE00D515E004374
S113234D727A00000000A20AE05E0042EE01006F07
S113235D60007001006F00000C69007920000146D8

S113236D3C7A00000000E80AE00CC95E00410C7ADB
S113237D00000000F00AE00CC95E00419C7A0000E9
S113238D0000EC0AE00C495E00407C5E0049667A71
S113239D0100009CE10D505E00498E404A7A000019
S11323AD0000840AE07A370000001E0FF17A020064
S11323BD00001E5E005FFE7A000000007C0AE07ADA
S11323CD37000000080FF17A02000000085E005F7D
S11323DDFE01006F6100BC01006F6000705E00368E
S11323ED267A17000000267A00000000A20AE05E9C
S10823FD00430C7920F0
S1132402000A4D6A01006F60007001006F00001442
S113241269007920000146567A00000000A20AE012
S11324225E0043A67920000146447A00000000A220
S11324320AE00D515E0043727A00000000A20AE036
S11324425E0042EE7A00000000F00AE0F96F5E00DF
S1102452419C7A00000000EC0AE00CC95E1A
S113245F00407C7A00000000E80AE00C495E00416E
S10A246F0C5E0060345470A1
S1139CCA4C696D69742E7478740000536166657407
S1139CDA792E747874000048656C6C6F20455620A1
S10B9CEA20202020202020008F
S11324765E0060561B970FF40C8D18886EC8000318
S11324866EC800026EC8000168C819660D605E005A
S1132496127A0D0E0D6117F10AC16819D90117D108
S11324A6660147260D600C004620A800470EA801CA
S11324B6470EA802470EA803470E400EFD75400AB5
S11324C6FD644006FD634002FD710B567926000448
S11324D64DBA0CD80B975E00603454705E0060569C
S11324E60F8618886EE800067A00000000120AE0DC
S11324F601006FE000147A00000000180AE00100F2
S11325066FE0002218886EE8002001006FE60002E3
S11325167A00000000070AE001006FE000087A0075
S11325260000000C0AE001006FE0000E7A050000CF
S11325363F0A7A00000000260AE05D507A00000098
S1132546002A0AE05D507A000000002E0AE05D5082
S11325567A00000000320AE05D507A000000003A7B
S10725660AE05D50D7
S113256A01006F60000201006DF07A00000000268E
S113257A0AE07A0100009CF25E003F9E0B977920E5
S113258A0001477A7A000000002E0AE001006F6119
S113259A00085E0040307920000147627A0000009B
S11325AA00320AE001006F61000E5E00415079209B
S11325BA0001474A01006F60001401006DF07A00C0
S11325CA0000003A0AE07A0100009CFE5E003F9E8A
S11325DA0B977920000147267900000096DF0010065
S11325EA6F60002201006DF07A00000000360AE0F5
S11325FA7A0100009D095E003FF80B970B875E0086
S113260A603454705E006056FC74F568FD4E0F86A4
S113261A0F937A01000000060AE17A000000002AFB
S113262A0AE05E0040C079200001587007766E68A0
S113263A00065C00FE366EE8000601006F600002C9
S113264A01006DF07A00000000260AE07A0100001A
S109265A9CF25E003F9EAE
S11326600B977920000158700744 7A000000002E70
S11326700AE001006F6100085E004030792000012C
S11326805870072A7A00000000320AE001006F61E7
S1062690000E5ED8

S1132693004150792000015870071001006F60005A
S11326A31401006DF07A000000003A0AE07A010099
S11326B3009CFE5E003F9E0B977920000158700635
S11326C3EA790000096DF001006F60002201006DDB
S11326D3F07A00000000360AE07A0100009D095EEB
S11326E3003FF80B970B876E680006A871587000BC
S11326F38EA872474CA873586000ACF8736DF07AD8
S1132703000000000260AE07A0100009D145E003FEA
S1132713140B877A000000003A0AE0F9735E004164
S1132723E0F84E6EE800067A000000002A0AE0F99A
S11327334E5E00410C7A000000002E0AE0F94E4081
S1132743360C586DF07A00000000260AE07A010087
S1132753009D145E003F140B87F84E6EE800067A63
S1132763000000002A0AE0F94E5E00410C7A0000E3
S113277300002E0AE0F9635E00407C5A002DAE7A16
S1082783000000002A24
S11327880AE06E6900065E00410C7A0100009D1F95
S10E2798790000015E00498E0FB05E67
S11327A30046A25A002DAE6E680007A86358600462
S11327B3BEF46F19DD6E680006A8485870041CA8A0
S11327C359587002F2A86358700080A864587001C6
S11327D394A86858700376A86F4710A875587000BB
S11327E3C8A8795870023A5A002DAE6E68001CA827
S11327F379470E7A00000000320AE00C495E00417B
S11328039C6868A868586003B0F85968E86DF07A63
S1132813000000000260AE07A0100009D145E003FD9
S1132823140B876E680012A873461C6868A8594680
S1132833166E68001FA86E460E7A000000003A0A5F
S1132843E00C595E0041E05A002BBC6E68001CA8E3
S113285379470E7A00000000320AE00C495E00411A
S11328639C6868A868463EF85968E86DF07A0000EA
S11328730000260AE07A0100009D145E003F140B5A
S1132883876E680012A873461C6868A85946166EBB
S109289368001CA86E465C
S11328990E7A000000003A0AE00CC95E0041E05AD2
S11328A9002BBC6E68001CA879470E7A0000000053
S11328B9320AE00C495E00419C6868A868586000C8
S11328C99AF85968E86DF07A00000000260AE07A60
S11328D90100009D145E003F140B876E680012A867
S11328E97346266868A85946206E680018A8794671
S11328F9186E68001CA86E46107A000000003A0A98
S1072909E00CC95EB4
S113290D0041E040526E680012A873461E6868A825
S113291D5946186E68001BA86E46107A0000000019
S113292D3A0AE0F96A5E0041E0402C6E680012A895
S113293D7346246868A859461E6E68001BA879461D
S113294D166E68001FA86E460E7A000000003A0A44
S113295DE00C595E0041E05A002BBC6E68001CA8C8
S113296D79470E7A00000000320AE00C495E0041FF
S113297D9C6868A8685860009AF85968E86DF07A01
S113298D00000000260AE07A0100009D145E003F5E
S113299D140B876E680012A87346266868A85946FB
S11329AD206E68001BA87946186E68001CA86E4639
S11329BD107A000000003A0AE00CC95E0041E040C5
S11329CD526E680012A873461E6868A85946186EA1
S11329DD680018A86E46107A000000003A0AE0F964
S11329ED6B5E0041E0402C6E680012A873462468AC

S10829FD68A859461E05
S1132A026E680018A87946166E68001FA86E460EF7
S1132A127A000000003A0AE00C595E0041E05A00D5
S1132A222BBC7A03000000180AE36838A87947121E
S1132A326E380004A879470A7A0100009D245A00DF
S1132A422BEC6868A868466CF85968E86DF07A0060
S1132A52000000260AE07A0100009D145E003F1484
S1132A620B876E680012A873461E6868A859461839
S1132A726E680018A86E46107A000000003A0AE059
S1132A82F96B5E0041E0402C6E680012A873462485
S1132A926868A859461E6E680018A87946166E68BB
S1132AA2001FA86E460E7A000000003A0AE00C5995
S1082AB25E0041E05A43
S1132AB7002BBC7A03000000180AE36E380003A852
S1132AC77947126E380004A879470A7A0100009DF6
S1132AD73E5A002BEC6868A868466CF85968E86D9D
S1132AE7F07A00000000260AE07A0100009D145ED8
S1132AF7003F140B876E680012A873461E6868A808
S1132B075946186E68001BA86E46107A000000002D
S1132B173A0AE0F96A5E0041E0402C6E680012A8A9
S1132B277346246868A859461E6E68001BA8794631
S1132B37166E68001FA86E460E7A000000003A0A58
S1132B47E00C595E0041E0406C6E680018A87947B5
S1132B570A7A0100009D245A002BEC6E68001CA81A
S1132B6779470E7A00000000320AE00C495E004103
S1132B779C6868A868463EF85968E86DF07A0000D3
S1132B870000260AE07A0100009D145E003F140B43
S1132B97876E680012A873461C6868A85946166EA4
S1092BA768001CA86E4645
S1132BAD0E7A000000003A0AE00CC95E0041E07A9B
S1132BBD000000002A0AE06E6900065E00410C7AEF
S1132BCD000000002E0AE00CD95E00407C5A002D57
S10E2BDDAE6E68001BA87947107A0158
S1132BE800009D3E0DD05E00498E5A002DAE6E68E2
S1132BF8001CA879470E7A00000000320AE00C494D
S1132C085E00419C6868A868463EF85968E86DF01C
S1132C187A00000000260AE07A0100009D145E0095
S1132C283F140B876E680012A873461C6868A8597E
S1132C3846166E68001CA86E460E7A000000003A1D
S1132C480AE00CC95E0041E07A000000002A0AE0AD
S1132C586E6900065E00410C7A000000002E0AE04F
S1132C680CD95E00407C5A002DAE6E68000CA86F2C
S1132C78586001327A03000000180AE36838A8791B
S1132C885860008C6E380007A879587000826E6807
S1132C980006A864470CA86F4708A87947045A0098
S1132CA82DAE6868A8684646F85968E86DF07A005A
S1132CB8000000260AE07A0100009D145E003F141C
S1132CC80B876E680012A87346246868A859461EC5
S1092CD86E680018A879E4
S1132CDE46166E68001FA86E460E7A000000003A74
S1132CEE0AE00C595E0041E07A000000002A0AE077
S1132CFE6E6900065E00410C7A00000000320AE0A5
S10A2D0E0CD95E00419C5A41
S1132D15002DAE7A03000000180AE36E380003A8FD
S1132D2579586000846E380007A879477C6E68007F
S1132D3506A859470AA86F4706A8754702406A6857
S1132D4568A8684646F85968E86DF07A00000000FF

S1132D55260AE07A0100009D145E003F140B876E7E
S1132D65680012A87346246868A859461E6E680051
S1132D751BA87946166E68001FA86E460E7A0000DA
S1132D8500003A0AE00C595E0041E07A00000000B9
S1132D952A0AE06E6900065E00410C7A0000000015
S1122DA5320AE00CD95E00419C5E00603454702A
S1139CF25361666574792E74787400004D6F746FC6
S1139D02722E74787400004C696D69742E747874C1
S1139D1200005361666574792E7478740051554955
S1139D2254000A41204261736B65742069736E2784
S1139D32742031737420466C6F6F72000A412042A3
S1139D4261736B65742069736E277420326E6420AD
S1099D52466C6F6F720006
S1132DB45E0060567A050000498E790600027A01A6
S1132DC400009D580D605D507A0100009D870D60E1
S1132DD45D507A0100009DA80D605D507A010000EA
S1132DE49DD40D605D507A0100009E020D605D501C
S1132DF47A0100009E0E0D605D505E0060345470D5
S1052E0440AEDB
S1139D580A5550203D202775272C20444F574E2065
S1139D683D202764272C204F50454E203D20276F48
S1139D78272C20434C4F5345203D20276327000AB7
S1139D88454D455247454E4359203D202773272CBF
S1139D98205245434F56455259203D2027722700EC
S1139DA80A31737420466C6F6F722043414C4C2008
S1139DB83D202779272C20326E6420466C6F6F7202
S1139DC82043414C4C203D20275927000A31737406
S1139DD820466C6F6F7220434C4F5345203D20271C
S1139DE868272C20326E6420466C6F6F7220434CB8
S1139DF84F5345203D20274827000A5155495420F1
S1139E083D20277127000A434F4D4D414E443E00E4
S1132E0601006DF60F86190069E06FE000026FE0BE
S1132E1600046FE0000601006FE600087A00000078
S1132E2600020AE001006FE0000C7A0000000004D3
S1132E360AE001006FE000107A00000000060AE0D5
S1132E4601006FE0001419006FE000187A0000001B
S1132E5600180AE001006FE0001A19006FE0001E77
S1132E667A000000001E0AE001006FE00020010066
S1132E766D76547001006DF60F865E003F0A7A0088
S1132E86000000040AE05E003F0A01006D765470FC
S1132E965E0060560F850F9601006F7400180100DF
S1132EA66F6000146900792000001465201006F60CB
S1132EB6001A6901462801006F66001A79000001AD
S1132EC669E07A00000000040AD0F9735E0041E06D
S1132ED67A0100009E18790000015E00498E684859
S1132EE6A859586001E8F87268C86DF07A010000C5
S1082EF69E1D0FD05EDC
S1132EFB003F140B875A0030D4FB6801006F60004E
S1132F0B106900792000001466E01006F60001A19E9
S1132F1B11698101006F6000206901462C01006F6C
S1132F2B6600207900000169E07A00000000040AC2
S1132F3BD0F96F5E0041E07A0100009E2979000011
S1132F4B015E00498E5A0030D46848A85958600175
S1132F5B787A00000000040AD00CB95E0041E0F857
S1132F6B7268C86DF07A0100009E1D0FD05E003FA2
S1132F7B140B875A0030D401006F600008690146B7
S1132F8B6E01006F60001A1911698101006F6000F7

S1132F9B206901462C01006F66002079000001694E
S10F2FABE07A00000000040AD0F94F5E39
S1132FB70041E07A0100009E2E790000015E00497E
S1132FC78E5A0030D46848A859586001007A000027
S1132FD70000040AD00CB95E0041E0F87268C86DBE
S1132FE7F07A0100009E1D0FD05E003F140B875A35
S1132FF70030D401006F60000C6901466C01006F5B
S113300760001A1911698101006F60002069014688
S11330172C01006F6600207900000169E07A000047
S11330270000040AD0F96F5E0041E07A0100009EB8
S113303729790000015E00498E5A0030D46848A8F8
S113304759586000887A00000000040AD00CB95E62
S11330570041E0F87268C86DF07A0100009E1D0F09
S1133067D05E003F140B87406401006F60000C695A
S113307700792000014656 01006F60001A19116993
S11330878101006F60002069014628 01006F660017
S113309720790000169E07A00000000040AD00CDF
S10830A7B95E0041E0E9
S10530AC7A01A4
S11330AE00009E3A790000015E00498E6848A859D7
S11330BE4614F87268C86DF07A0100009E1D0FD099
S11330CE5E003F140B875E006034547001006DF692
S11330DE0F865E003F0A7A00000000040AE05E00DD
S11330EE3F0A01006D7654705E0060560F850F9691
S11330FE01006F74001801006F60000C69007920E5
S113310E0001465201006F60001A69014628010052
S113311E6F66001A7900000169E07A00000000046E
S113312E0AD0F9735E0041E07A0100009E1879001F
S113313E00015E00498E6848A859586001E8F8728C
S113314E68C86DF07A0100009E1D0FD05E003F141B
S113315E0B875A003334FB7401006F60000869005B
S113316E79200001466E01006F60001A1911698102
S113317E01006F6000206901462C01006F6600207C
S113318E7900000169E07A00000000040AD0F963B7
S109319E5E0041E07A012E
S11031A400009E45790000015E00498E5A2F
S11331B10033346848A859586001787A0000000048
S11331C1040AD00CB95E0041E0F87268C86DF07A68
S11331D10100009E1D0FD05E003F140B875A003380
S11331E13401006F6000106901466E01006F6000D9
S11331F11A1911698101006F6000206901462C01D0
S1133201006F6600207900000169E07A0000000088
S1133211040AD0F9435E0041E07A0100009E4B7934
S11332210000015E00498E5A0033346848A859589A
S11332316001007A00000000040AD00CB95E00416D
S1133241E0F87268C86DF07A0100009E1D0FD05E30
S1133251003F140B875A00333401006F6000146977
S113326101466C01006F60001A1911698101006F39
S1133271600020690146 2C01006F6600207900007F
S11332810169E07A00000000040AD0F9635E00419D
S1133291E07A0100009E45790000015E00498E5AE3
S11332A10033346848A859586000887A0000000048
S11332B1040AD00CB95E0041E0F87268C86DF07A77
S11332C10100009E1D0FD05E003F140B8740640177
S11332D1006F60001469007920000146560 1006FF8
S11332E160001A1911698101006F600020690146AC
S11332F12801006F6600207900000169E07A00006F

S11333010000040AD00CB95E0041E07A0100009E7E
S113331158790000015E00498E6848A8594614F89F
S11333217268C86DF07A0100009E1D0FD05E003FE8
S1133331140B875E006034547001006DF60F867ABA
S113334100000000040AE001006FE000067A0000BB
S113335100000E0AE001006FE0001801006D7654D1
S1133361705E0060560F860F957A000000000E0A0A
S1133371E001006FE000187A000000000A0AE00192
S1133381006F6100185E0042666E680012A87946FC
S109339106790000014073
S11333970219006FE0001C7A04000043C46F600049
S11333A71C6DF001006F51002001006F50000C5D90
S11333B7400B876E680013A8794606790000014021
S11333C70219006FE0001C6DF001006F510020012E
S11333D7006F5000085D400B876E680014A879469C
S11333E706790000001400219006FE0001C6DF0012F
S11333F7006F51002001006F5000105D400B876E76
S1133407680015A879460679000001400219006F84
S1133417E0001C6F66001C6DF601006F5100200170
S1133427006F5000145D400B875E00603454705E7C
S11334370060560F860F957A000000000E0AE00120
S1133447006FE000187A000000000A0AE001006F2D
S1133457610018 5E0042666E680012A87946067915
S113346700000 1400219006FE0001C7A04000043CA
S1133477C46F60001C6DF001006F51002001006FE5
S108348750000C5D4044
S113348C0B876E680013A879460679000001400289
S113349C19006FE0001C6DF001006F51002001005A
S11334AC6F5000085D400B876E680014A8794606C0
S11334BC79000001400219006FE0001C6DF001005F
S11334CC6F51002001006F5000105D400B876E6838
S11334DC0015A879460679000001400219006FE037
S11334EC001C6F66001C6DF601006F51002001007B
S11234FC6F5000145D400B875E00603454705EA8
S113350B0060560F860F957A000000000E0AE0014B
S113351B006FE000187A000000000A0AE001006F58
S113352B6100185E0042666E680012A87946067940
S113353B000001400219006FE0001C7A04000043F5
S113354BC46F60001C6DF001006F51002001006F10
S113355B50000C5D400B876E680013A87946067903
S113356B000001400219006FE0001C6DF001006FB9
S113357B51002001006F5000085D400B876E6800FF
S113358B14A879460679000001400219006FE00088
S113359B1C6DF001006F51002001006F5000105D96
S11335AB400B876E680015A8794606790000014029
S11335BB0219006FE0001C6F66001C6DF601006FB3
S11335CB51002001006F5000145D400B875E0060BB
S11335DB3454705E0060560F860F9501006F6000C8
S11335EB14690146307A00000000200AF00FE15CF9
S10635FB00FE3894
S11335FE01006DF57A000000001C0AF00FE15C007B
S113360EF8860B977A00000000200AF00FE15C00A9
S113361EFE165E00603454705E0060560F860F9582
S113362E01006F60000C690146307A000000002033
S113363E0AF00FE15C00FEC401006DF57A00000094
S113364E001C0AF00FE15C00FA9E0B977A00000053
S113365E00200AF00FE15C00FEA25E00603454709D

S1139E1853544F50005361666574792E74787400F7
S1139E28004F50454E004F50454E205370656564B2
S1139E3879004F50454E20537461727400434C4F60
S1139E48534500434C4F5345205370656564790 06F
S10F9E58434C4F53452053746172740057
S113366E01006DF60F86190069E06FE000026FE04E
S113367E00046FE0000601006FE600087A00000008
S113368E00020AE001006FE0000C7A000000000463
S113369E0AE001006FE000107A00000000060AE065
S11336AE01006FE0001419006FE000187A000000AB
S11336BE00180AE001006FE0001A19006FE0001E07
S11336CE7A000000001E0AE001006FE000200100F6
S11336DE6D76547001006DF60F865E003F0A7A0018
S11336EE000000040AE05E003F0A01006D7654708C
S11336FE5E0060560F850F9601006F74001801006F
S113370E6F60001469007920000146520 1006F605A
S113371E001A6901462801006F66001A790000013C
S113372E69E07A00000000040AD0F9735E0041E0FC
S113373E7A0100009E64790000015E00498E68489C
S113374EA859586001E8F87268C86DF07A01000054
S108375E9E690FD05E1F
S1133763003F140B875A00393CFB6A01006F60006A
S1133773106900792000014 66E01006F60001A1979
S113378311698101006F6000206901462C01006FFC
S1133793660020790000016 9E07A00000000040A52
S11337A3D0F9755E0041E07A0100009E757900004F
S11337B3015E00498E5A00393C6848A85958600194
S11337C3787A00000000040AD00CB95E0041E0F8E7
S11337D37268C86DF07A0100009E690FD05E003FE6
S11337E3140B875A00393C01006F600008690146D6
S11337F36E01006F60001A1911698101006F600087
S1133803206901462C01006F6600207900000169DD
S10F3813E07A00000000040AD0F9555EC2
S113381F0041E07A0100009E78790000015E0049C3
S113382F8E5A00393C6848A859586001007A000045
S113383F0000040AD00CB95E0041E0F87268C86D4D
S113384FF07A0100009E690FD05E003F140B875A78
S113385F00393C01006F60000C6901466C01006F79
S113386F60001A1911698101006F60002069014618
S113387F2C01006F6600207900000169E07A0000D7
S113388F0000040AD0F9755E0041E07A0100009E42
S113389F75790000015E00498E5A00393C6848A8CB
S11338AF59586000887A00000000040AD00CB95EF2
S11338BF0041E0F87268C86DF07A0100009E690F4D
S11338CFD05E003F140B87406401006F60000C69EA
S11338DF007920000146560 1006F60001A19116923
S11338EF8101006F60002069014628 01006F6600A7
S11338FF207900000169E07A00000000040AD00C6F
S108390FB95E0041E078
S10539147A0133
S113391600009E82790000015E00498E6848A8591E
S11339264614F87268C86DF07A0100009E690FD0DC
S11339365E003F140B875E006034547001006DF621
S11339460F865E003F0A7A00000000040AE05E006C
S11339563F0A01006D7654705E0060560F850F9620
S113396601006F74001801006F60000C6900792074
S1133976000146520 1006F60001A690146280100E2

S11339866F66001A7900000169E07A0000000004FE
S11339960AD0F9735E0041E07A0100009E64790063
S11339A600015E00498E6848A859586001E8F8721C
S11339B668C86DF07A0100009E690FD05E003F145F
S11339C60B875A003B9CFB6B01006F600008690084
S11339D679200001466E01006F60001A1911698192
S11339E601006F6000206901462C01006F6600200C
S11339F67900000169E07A00000000040AD0F96446
S1093A065E0041E07A01BD
S1103A0C00009E8B790000015E00498E5A78
S1133A19003B9C6848A859586001787A0000000067
S1133A29040AD00CB95E0041E0F87268C86DF07AF7
S1133A390100009E690FD05E003F140B875A003BBB
S1133A499C01006F6000106901466E01006F600000
S1133A591A1911698101006F6000206901462C015F
S1133A69006F6600207900000169E07A0000000018
S1133A79040AD0F9445E0041E07A0100009E90797E
S1133A890000015E00498E5A003B9C6848A85958BA
S1133A996001007A00000000040AD00CB95E0041FD
S1133AA9E0F87268C86DF07A0100009E690FD05E74
S1133AB9003F140B875A003B9C01006F6000146997
S1133AC901466C01006F60001A1911698101006FC9
S1133AD96000206901462C01006F6600207900000F
S1133AE90169E07A00000000040AD0F9645E00412C
S1133AF9E07A0100009E8B790000015E00498E5A2D
S1133B09003B9C6848A859586000887A0000000067
S1133B19040AD00CB95E0041E0F87268C86DF07A06
S1133B290100009E690FD05E003F140B87406401BA
S1133B39006F60001469007920000146560 1006F87
S1133B4960001A1911698101006F6000206901463B
S1133B592801006F6600207900000169E07A0000FE
S1133B690000040AD00CB95E0041E07A0100009E0E
S1133B799C790000015E00498E6848A8594614F8EB
S1133B897268C86DF07A0100009E690FD05E003F2C
S1133B99140B875E006034547001006DF60F867A4A
S1133BA900000000040AE001006FE0000E01006D4F
S1133BB97654705E0060560F860F957A00000000F8
S1133BC9040AE001006FE0000E01006F61000E0FAF
S1133BD9E05E0042666E680004A879460679000033
S1133BE901400219006FE000127A04000043C46F18
S1083BF96000126DF0F5
S1133BFE01006F51002001006F50000C5D400B87D8
S1133C0E6E680005A879460679000001400219008 6
S1133C1E6FE000126DF001006F51002001006F5034
S1133C2E00085D400B876E680006A879460679008A
S1133C3E0001400219006FE000126DF001006F5198
S1133C4E002001006F5000105D400B876E68000767
S1133C5EA879460679000001400219006FE00012B0
S1133C6E6F6600126DF601006F51002001006F5058
S1133C7E00145D400B875E00603454705E00605626
S1133C8E0F860F957A00000000040AE001006FE032
S1133C9E000E01006F61000E0FE05E0042666E685B
S1133CAE0004A879460679000001400219006FE06E
S1133CBE00127A04000043C46F6000126DF001001D
S1133CCE6F51002001006F50000C5D400B876E6832
S1133CDE0005A879460679000001400219006FE03D
S1073CEE00126DF060

S1133CF201006F51002001006F5000085D400B87E7
S1133D026E680006A879460679000001400219009O
S1133D126FE000126DF001006F51002001006F503F
S1133D2200105D400B876E680007A879460679008C
S1133D320001400219006FE000126F6600126DF677
S1133D4201006F51002001006F5000145D400B878A
S10A3D525E00603454705E53
S1133D590060560F860F957A00000000040AE001FF
S1133D69006FE0000E01006F61000E0FE05E00427C
S1133D79666E680004A879460679000001400219B5
S1133D89006FE000127A04000043C46F6000126DF3
S1133D99F001006F51002001006F50000C5D400BD2
S1133DA9876E680005A879460679000001400219 63
S1133DB9006FE000126DF001006F51002001006FE8
S1133DC95000085D400B876E680006A8794606799E
S1133DD9000001400219006FE000126DF001006F4D
S1133DE951002001006F5000105D400B876E680081
S1133DF907A879460679000001400219006FE0001F
S1133E09126F6600126DF601006F51002001006FF9
S1133E195000145D400B875E00603454705E00608F
S1133E29560F860F9501006F600014690146307AB9
S1133E3900000000200AF00FE15C00FE4401006D60
S1063E49F57A0004
S1133E4C0000001C0AF00FE15C00F8A60B977A0047
S1133E5C000000200AF00FE15C00FE225E006034DB
S1133E6C54705E0060560F860F9501006F60000C56
S1133E7C690146307A00000000200AF00FE15C0073
S1133E8CFECA01006DF57A000000001C0AF00FE178
S1133E9C5C00FABE0B977A00000000200AF00FE1D9
S10D3EAC5C00FEA85E006034547051
S1139E6453544F50005361666574792E74787400AB
S1139E74005550005550205370656564790055506 2
S1139E842053746172740044 F574E00444F574E2D
S1139E9420537065656479004 4F574E20537461B1
S1069EA4727400D2
S1133EB601006DF60F867A0000FFE220010069E03B
S1133EC67A0600FFE220F87268E87A00000000062E
S1133ED60AE001006FE000027A0100009EA80100DB
S1133EE66F6000025E005980F8736EE8000FF84EAB
S1133EF66EE800106EE80011F84E6EE8001201003D
S1133F066D7654700F811A800100699054705E00BB
S1133F1660560F946E7D00197A0600FFE220684909
S1133F26A9434716A94D470CA9504714A953462838
S1133F3668ED40246EED000F401E6EED0010401834
S1133F466E480006A8434706A8544708400A6EED84
S1133F56001140046EED001219005E0060345470C7
S1133F6601006DF60F966869A94C46247A0600FF90
S1133F76E2207A00000000060AE001006FE000027A
S1133F8601006F71000801006F6000025E00598036
S1133F96190001006D7654705E0060560F9401009F
S1093FA66F7500187A0696
S1133FAC00FFE2206849A9434716A94D470CA950C5
S1133FBC4716A953462E686E400A6E6E000F4004D6
S1133FCC6E6E001068DE401C6E480006A843470660
S1133FDCA854470A400E6E6E001168DE40066E6EE2
S1133FEC001268DE19005E006034547001006DF637
S1133FFC0F966869A94C46247A0600FFE2207A00E2

S113400C000000060AE001006FE0000201006F618E
S113401C000201006F7000085E0059801900010056
S113402C6D7654705E0060560F850F9601006DF629
S113403C7A0100009EB20FD05C00FF560B970C0068
S113404C4624A8014706A8024716401A7A01000025
S113405C9EC5790000015E00498E79060001400877
S113406C79060002400219660D605E0060345470DC
S104407C5EE2
S113407D0060560F850C9E6DF67A0100009EB20FFF
S113408DD05C00FE820B870C00461CA8004714A8C9
S113409D0146147A0100009ED4790000015E0049A7
S11340AD8E400419664004790600010D605E0060C0
S11340BD3454705E0060560F850F9601006DF67ACD
S11340CD0100009EE30FD05C00FEC60B970C00466B
S11340DD24A8014706A8024716401A7A0100009E3C
S11340EDC5790000015E00498E790600014008790B
S11340FD060002400219660D605E00603454705E66
S113410D0060560F850C9E6DF67A0100009EE30F3D
S113411DD05C00FDF20B870C00461CA8004714A8C9
S113412D0146147A0100009ED4790000015E004916
S113413D8E400419664004790600010D605E00602F
S113414D3454705E0060560F850F9601006DF67A3C
S104415D015D
S113415E00009EF00FD05C00FE360B970C00462439
S113416EA8014706A8024716401A7A0100009EC509
S113417E790000015E00498E790600014008790638
S113418E0002400219660D605E00603454705E00DA
S113419E60560F850C9E6DF67A0100009EF00FD0CF
S11341AE5C00FD620B870C00461CA8004714A80197
S11341BE46147A0100009ED4790000015E00498EF8
S11341CE400419664004790600010D605E006034F8
S11341DE54705E0060560F850C9E6DF67A010000DA
S11341EE9F040FD05C00FD1E0B870C004616A80023
S11341FE4712A801460E7A0100009ED479000001F1
S113420E5E00498E5E00603454705E0060561B87FC
S113421E0F8518886EF800017A06000000010AF671
S113422E01006DF67A0100009F040FD05C00FD6063
S113423E0B970C004618A800470EA8014710A802BA
S109424E470C400A400882
S110425440066E780001400218880B875E5B
S1134261006034547 05E0060560F850F967900002C
S1134271096DF001006DF67A0100009F0F0FD05C0C
S113428100FD740B970B870C004618A8014706A87D
S1134291024710400E7A0100009EC5790000015EBD
S10C42A100498E5E006034547084
S1139EA879796E6E79796E6E00005065726D69749A
S1139EB8436F6D6D616E642E74787400000A526589
S1139EC86164696E67204572726F72000A5772691E
S1139ED874696E67204572726F7200436F6D6D61AE
S1139EE86E642E74787400005065726D69745475CD
S1139EF8726E4F70656E2E74787400004D6F746FB8
S1139F08722E74787400004C696D69742E747874B9
S1059F18000044
S11342AA5E0060560F860F957A0000009F1A0FE191
S11342BA5E005FE001006FE600080FE055267A0012
S11342CA0000000C0AE001006FE0000E19006FE025
S11342DA000C19006FE0001201006FE500145E0084

S11342EA603454705E0060560F8601006FE6000862
S11342FA0FE201006DF25E0044020B975E00603428
S113430A54705E0060567A37000000100F86010071
S113431A6FE600087A02000000080AF201006DF253
S113432A5E0044020B9701006F6600080FA10FE2BB
S113433A0FF05E005A825E005EF47A1700000010E6
S113434A5E00603454705E0060560F850D16790E58
S113435A03E852E617F60FE101006F5000145E00FE
S113436A44A65E006034547001006DF60F867A002D
S113437A0000000C0AE001006FE0000E7921000141
S113438A460A790000016FE0000C400A0D11460647
S109439A19006FE0000CA6
S11343A001006D76547001006DF60F867A000000EF
S11343B0000C0AE001006FE0000E6F60000C0100CA
S11343C06D7654705E0060560F860F9569606F714D
S11343D000181D10470A190069D06F70001869E0B2
S11343E05E00603454705E0060560F850D1617F63C
S11343F00FE101006F5000145E0044A65E006034BC
S10544005470F3
S10B9F1A00000000000000003C
S11344027A0000FFE23401006F7100045E005FE096
S113441254705E0060567A37000000200F867A02DD
S1134422000000180AF201006DF255D40B970FE158
S11344320FF05E005F9A0F817A0200009F327A00CA
S1134442000000100AF05E005D5C7A0000000018B4
S11344520AF07A01000000080AF15E005FE04010F2
S11344627A02000000080AF201006DF255920B97DE
S11344727A01000000180AF17A02000000100AF221
S11344820FF05E005AB20F817A00000000080AF0B2
S11344925E005F8A0D0046C87A17000000205E00A6
S11344A2603454705E0060561B971B970F860F95FE
S11344B27A02000000020AE201006DF25C00FF4092
S11344C20B970FD10FF05E005F9A0F817A02000003
S11344D29F327A000000000A0AE05E005D5C0B97DF
S11344E20B975E00603454705E0060560F860F9522
S10744F20FE055B0CF
S11344F601006F6000120B7001006FE000125E0096
S1134506603454705E0060560D057A0400FFE23C89
S1134516400601006F44001A01006F40001A0100B3
S11145266F01001A46EC79250001460A7A0659
S113453400FFE2785A00462879250002460A7A06E3
S113454400FFE2965A0046287925000B460A7A06AC
S113455400FFE2B45A0046287925000C460A7A067D
S113456400FFE2D25A0046287925000D460A7A064E
S113457400FFE2F05A0046287925000E460A7A061F
S113458400FFE30E5A00462879250013460A7A06EB
S108459400FFE32C5AB7
S1134599004628792500144 60A7A0600FFE34A5A99
S11345A900462879250015460 87A0600FFE3684086
S11345B96E7925001646087A0600FFE3864060797E
S11345C925001746087A0600FFE3A440527925001F
S11345D91E46087A0600FFE3C240447925001F46B8
S11345E9087A0600FFE3E04036792500294 6087A70
S11345F90600FFE3FE40287925002A46087A0600CB
S1134609FFE41C401A7925002B46087A0600FFE4CB
S11346193A400C7925002C46067A0600FFE4580F28
S1134629E6461C7A0100009F2219005E00498E7A32

S1044639017C
S113463A00009F2419005E00498E1A80405401002D
S113464A6FE4001601006F40001A01006FE0001AC0
S113465A01006F86001601006FC6001A7A00000077
S113466A9F3A7A01000000020AE15E005FE07A00E5
S113467A00009F427A010000000A0AE15E005FE03F
S113468A69E51A8001006FE000120FE05E000EA2D6
S113469A0FE05E006034547001006DF60F865E0011
S11346AA113601006F60001601006F61001A0100E4
S11346BA6F81001A01006F60001A01006F61001612
S11346CA01006F81001601006D7654705E0060561A
S11346DA0F867A0200FFE23401006DF25C00FD18D6
S11346EA0B977A0000FFE2347A01000000020AE124
S11346FA5E005FE05E006034547001006DF60F8661
S113470A7A0000009F4A7A01000000020AE15E0073
S113471A5FE001006D76547001006B2000FFE256E2
S109472A01006F01001AFB
S1134730460419005470790000001547001006B2184
S113474000FFE25601006F11001A1988400C0100A6
S11347506F11001A0D800B500D080F9146F00D805C
S113476054706DF50D0501006B2000FFE25601004A
S11347706F01001A472001006B2100FFE256691008
S11347801D5046040F90401001006F11001A0100E4
S11347906F10001A46E81A806D7554705E006056FB
S11347A00D0555BE0F86460E0D505C00FD5C0F8651
S11347B05C00FF22401C7A00000000020AE07A013C
S11347C000009F4A5E005F6E0D0047060FE05C002D
S11347D0FF040FE05E00603454705E0060560D0508
S11347E055800F86460E0D505C00FD1E0F865C0043
S11347F0FEE440247A00000000020AE07A0100008F
S11348009F4A5E005F6E0D0047080FE05C00FEC626
S113481040060FE05C00FE8A5E00603454705E0068
S113482060561B971B977A0500FFE23C01006F5609
S1074830001A4072B5
S113483401006F65001A7A00000000020AE07A01A1
S113484400009F3A5E0060260D0047547A00000082
S109485400020AE07A01F4
S113485A00009F4A5E0060260D00473E7A01000071
S113486A00020AE17A020000000A0AE20FF05E007F
S113487A5AB20F817A0000FFE2345E005F7A0D00BC
S113488A47187A0000FFE2347A01000000020AE1C5
S113489A5E005FE00FE05E0005480FD601006F601F
S11348AA001A46860B970B975E00603454701A8081
S11348BA01006BA000FFE2527A0000FFE25A0100F6
S11348CA6BA000FFE2567A0000FFE23C01006BA0F6
S10D48DA00FFE2701A8001006BA0DA
S11348E400FFE2745470F801386018883861386244
S11348F4188838633890F8FF389118883864F88832
S11349043865F804386679009E576B80FF68190090
S11349146B80FF6A19006B80FF6C547001006DF2A9
S11349247F67722079009E576B80FF687A0100FFCE
S1134934E2347A0200009F520F905E005AB20100E3
S11349446D7254705A00481E7A0000009F427A0127
S113495400FFE2345E005FE0558C5E0002A85A005B
S105496448B84E
S1139F220A0063616C6C6F63206661696C6564002F
S1139F32408F400000000000BFF00000000000005E

S1139F4200000000000000000C0000000000000004C
S10B9F523F50624DD2F1A9FC5E
S11349667A0000009F5A01006DF07A0000FFE4769A
S11349765E0058F80B977A0000FFE47601006DF0AD
S11349865E004D100B9754705E0060567A0500FF6B
S1134996E4B60D040F960C4458600114AC00476846
S11349A6AC014710AC0258700106AC03587000C244
S11349B65A004AB655AA7A0100009F5C0FE05E00D2
S11349C659560D00461E7A0000009F6201006DF0E5
S11349D67A0000009F5F01006DF00FE05E0058F85B
S11349E60B970B9701006DF67A0000009F5F01009D
S11349F66DF00FD05E0058F80B970B9701006DF51D
S10D4A067A0000009F5F40507A0120
S1134A1000009F5C0FE05E0059560D00461E7A00B1
S1134A2000009F6201006DF07A0000009F5F0100AB
S1134A306DF00FE05E0058F80B970B9701006DF6D1
S1134A407A0000009F5F01006DF00FD05E0058F800
S1134A500B970B9701006DF57A0000009F5F010033
S1134A606DF05E004B0C0B970B9701006DF65E002B
S1134A704D100B974040403E01006DF67A00000058
S1134A809F5F01006DF00FD05E0058F80B970B97F6
S1134A9001006DF57A0000009F5F01006DF05E007C
S1134AA04B0C0B970B970FD05E00599C0B500D01CD
S10A4AB00FD05E0057745E96
S1084AB700603454709F
S10E9F5A0C000A00002573000A0C0035
S1134ABC188838BA38B8F85038B919000B5079201F
S1134ACC01184DF8F83038BA28BCF88038BC54704B
S1134ADC0C8820BC737047FA38BBE07F30BC547031
S1134AEC20BC736047FA29BDE0BF30BC0C985470EE
S1134AFC7EBC736047067900000154701900547032
S1134B0C5E0060567A0600FFE4F67A0500000004A6
S1134B1C7A00000000180AF00AD07308470E7A00D6
S1134B2C000000180AF00AD00B70400A7A0000004B
S1134B3C00180AF00AD00F85189968E901006DF086
S1134B4C01006F71001C0FE05E0059B80B971955EB
S1134B5C0D5017F00AE0680947220D5017F00AE0D0
S1134B6C6808A80A4606F80D5C00FF640D5017F0A0
S1134B7C0AE068085C00FF580B5540D45E006034B3
S1054B8C547060
S1134B8E1911400C19880B58792806824DF80B51D0
S1134B9E1D014DF0547001006DF50D090D8D28D6D4
S1134BAE17500D010D99470C0D197969006079495C
S1134BBE001040060D19796900600DD50D90148D06
S1134BCE0CD8C88038D63DD639D67900000455B0F6
S1134BDE01006D7554706DF56DF40D090D8528D6B4
S1134BEE17500D010D99470C0D197969006079491C
S1134BFE001040060D19796900600D54119411943B
S1134C0E11941194EC0F0D90148CED0F148D0CC8A0
S1134C1EC88038D63CD60CD8C88038D63DD639D6BF
S1134C2E790000045C00FF586D746D7554700100BB
S1134C3E6DF619EE7900000F5C00FF4419667908D2
S1134C4E00030DE05C00FF4E0B56792600034DEE7C
S1134C5E790800020DE05C00FF3C19667908002814
S1134C6E0D605C00FF70790800100D605C00FF663C
S1094C7E7908000E0D6031
S1134C845C00FF5C790800060D605C00FF52790844

S1134C9400010D605C00FF48790800020D605C00B0
S1134CA4FF3E01006D7654707908000119005C0021
S1134CB4FF2E7908000219005A004BE46DF60C8E9E
S1134CC4AE0C460455E2401CAE0A460455DA4014C1
S1134CD4AE0D460455D2400C17D60D687900000179
S1134CE45C00FEFC6D76547001006DF60D0E0D86AE
S1134CF40D6079080040528009E0791000800D08A6
S1134D0419005C00FEDA01006D7654705E00605693
S1134D14790C000119447A0600FFE5467A05000080
S1134D2400047A00000000180AF00AD07308470E42
S1134D347A00000000180AF00AD00B70400A7A00C7
S1134D44000000180AF00AD00F85189968E90100D9
S1134D546DF001006F71001C0FE05E0059B80B97F2
S1134D6419550D5017F00AE0680947560D5017F00E
S1094D740AE06808A80A2A
S1134D7A460A0DC80D405C00FF68403C0D5017F011
S1134D8A0AE06808A80D460C790800020D405C0089
S1134D9AFE4840240D5017F00AE06808A80C46069E
S1134DAA5C00FEFE40120D5017F00AE0680817D0A7
S1134DBA0D080DC05C00FE220B5540A05E00603456
S1054DCA547020
S1134DCC1911400C19880B58792806824DF80B5190
S1134DDC1D014DF054705C0009145C0000B45504C3
S1134DEC5A004EC601006DF618886AA800FFE64902
S1134DFC6AA800FFE64B18886AA800FFE64A6AA86F
S1134E0C00FFE58779080080790000045C0008083E
S1134E1C79080010790000205C0007FC7906000F6C
S1134E2C790800100D605C0007EE0D687900000B2B
S1134E3C5C0007E40D687900000D5C0007DA790863
S1134E4C0050790000095C0007CE790800D6790080
S1134E5C00075C0007C219660D605C000710790E31
S1134E6C00010DE05C0006E60DE05C0007000D6838
S1134E7C7900002B5C0007A00D687900002F5C0003
S1134E8C07960DE8790000275C00078C01006D760E
S1134E9C547079080005190 05C00077C79000064E4
S1134EAC5C00FF1C790800C419005C00076A7908D0
S1094EBC00037900000170
S1134EC25A00562479080002790000055C00075253
S1134ED2790800CC19005A0056245E0060567A0401
S1134EE20000A0187A0000009FE701006DF05E0049
S1134EF24B0C0B9719EE19660DE5790D001052D57F
S1134F020D6009505C00070C17506DF001006DF441
S1134F125E004B0C0B970B870B56792600104DE066
S1134F227A000000A01E01006DF05E004B0C0B978F
S1134F320B5E792E00044DBE5E0060345470010096
S1074F426DF67A0685
S1134F4600FFE586790000065C0006C468E87C601D
S1134F567340473679000 00E5C0006B40C8E734826
S1134F6647045C0002C8735E47085C0002C25A002D
S1134F765034730E47085C0002B85A005034731E4F
S1134F8647045C0002AE5A0050347C6073604722CB
S1134F967900000C5C0006780C8E730847085C00E9
S1134FA600925A005034731E587000825C0001C68A
S1134FB6407C7C607320471E7900000A5C00065023
S1134FC60C8E730847065C00020A4062731E475E36
S1134FD65C00023E40587C607310475279000 0081B
S1134FE65C00062C0C8E7368470A5C00FDFC5C00B3

S1134FF6FECE403A734E471A790800C0790000097D
S11350065C00061A79080003790000055C00060EA9
S1135016401C737E4718790800507900000095C002C
S113502605FC79080002790000055C0005F0010023
S10850366D7654705E6D
S113503B00605619DD790000265C0005CE0C8E73DB
S113504B68587001047A0600FFE589790000086D42
S113505BF00FE179080026790000255C0005CC0BE5
S113506B870DD05C0004E80DD05C000502790500C8
S113507B200D505C0004C06868E8605860009C6EAB
S113508B680001473CA801471EA8034772A80547C0
S113509B3EA8064726A8084716A809475CA80A474F
S11350AB26A80B4760406C5C0001865A0051426A8C
S11350BB2800FFE58817500D08400E5C00022440C2
S11350CB765C00035440700DD8790000214024799D
S11350DB0800400D505C0005406E6E0002CE806AE6
S11350EBAE00FFE5876A2800FFE58717500D0879A7
S11350FB0000045C000522403E5C00038E40385CDC
S113510B00018840326E680002472C0D505C00048E
S113511B0E40240D505C000406401C6868E860A830
S108512B2046080D50B1
S11351305C0003F6400C686EEE60AE400D505C0000
S113514003E87A0000FFE64A7D0070000DD05C00A2
S1135150045440226A2800FFE64B470818886AA8CF
S113516000FFE64B0DD05C00041079080001790 0C4
S10A517000275C0004AE5EA2
S1135177006034547 05E0060567900002E5C0004B2
S11351878E0C8EE8C017504626790000406DF07AE2
S11351970500FFE5C90FD17908002E7900002D5CC2
S11351A700048C0B870D060D010FD05C0005627997
S11351B70000015C00039C790800017900002F5C63
S11351C700045A5E00603454707900003 65C0004B2
S11351D73E54706DF6790000225C0004320C8E7326
S11351E768472A7358472619005C0003866A280014
S11351F7FFE64B470C5C0001C819005C0003A040A5
S11352070C79080001790000275C0004106D7654BF
S1135217706DF67900002A5C0003F40C8E736847FF
S1135227087358470 45C0004826D765470547054B5
S1135237705470547 05E0060567A0600FFE5896803
S113524768E803A80246426E680003E80747186E3A
S1135257690003E9071751177178106A2800009F3F
S10852676617505C0016
S113526C02D46E680003E8071750790100011A088D
S113527C4B04101140F817117A0000FFE649680A35
S113528C169A688A5E00603454705E0060567A0623
S113529C00FFE5896868E803A80246406E680003CE
S11352ACE80747186E690003E9071751177178105F
S11352BC6A2800009F6617505C0002626E68000348
S11352CCE8071750790100011A084B04101140F834
S11352DC7A0000FFE649680A149A688A5E00603413
S11152EC54707A0000FFE64A7D0072006A28C3
S11352FA00FFE58CA801470AA8024716A80347221C
S113530A54706A2900009F6E17517A0000009F6E3D
S113531A40686A2900009F8217517A0000009F8023
S113532A40586A2800FFE58B470EA801471AA802CE
S113533A4726A803473254706A2900009FA717D14A
S113534A7A0000009FA740326A2900009FAB17D159

S113535A7A0000009FAB40226A2900009FBD17D143
S10D536A7A0000009FBD40126A297B
S113537400009FC517D17A0000009FC555025470E1
S11353845E0060560F840D187A0600FFE64C6A2807
S113539400FFE59017500C8018886A2900FFE58FF9
S11353A41751091069E01D804F0269E801006BA4DD
S11353B400FFE64EF8016AA800FFE64B55065E00BF
S11353C4603454705E0060567A0600FFE64C79053B
S11353D400086960474469607920000084E026965E2
S11353E47A0400FFE64E6DF5010069417908002255
S11353F4790000215C0002780B870D0569611901AE
S113540469E117F0010069410A81010069C169601A
S1135414460818886AA800FFE64B5E00603454709F
S10454245E26
S11354250060566A2800FFE589E80317500D051942
S1135435EE790600210D004706792500014610 0F78
S1135445E05C0001DA0DE80D605C0001D2403C79B7
S1135455250002462E6A2800FFE58CE80717500D44
S113546505790100011A084B04101140F86A280058
S1135475FFE6491750660147067908000140060D06
S1135485E840020DE80D605C0001945E0060345451
S1135495706DF66A2800FFE58B6AA800FFE588466C
S11354A53C19660D68790000285C0001720D687966
S11354B500002C5C0001680D68790000305C000178
S11354C55E0D68790000345C0001540D68790000B5
S11354D5385C00014A0D687900003C404018886A31
S11354E5A800FFE649790600010D605C0000867996
S11354F5080011790000285C00012479080003796C
S113550500002B5C0001180D60554A7908001279DB
S108551500002C5C0006
S113551A01080D687900002F5C0000FE6D76547057
S113552A6DF60D065C0000E4C88017500D080D6087
S113553A5C0000E66D7654706DF60D065C0000CCD7
S113554AE87F17500D080D605C0000CE6D7654702D
S113555A01006DF60D0617F6103679080008786013
S113556A6B2000FFE0085C0000B001006D76547008
S113557A5E0060560D0617F610367A0500FFE00046
S113558A0AE569505C00008417500D06703E0D68E9
S113559A69505C0000845E00603454705E0060569B
S11355AA1B971B977A0500FFE64A0D0EFE01790049
S11355BA00010DE11A094B04101040F868591751FC
S11355CA66104704702E4002722E701E17560DE0A5
S11355DA17F0103001006FF000040D6801006F71BD
S11355EA000478106B2000FFE000010069F1552ADE
S11355FA790000011B5E4B04101040F868591589A5
S109560A68D90B970B9712
S10456105E38
S113561100603454706AA8004000036A2800400007
S11356210154700D016AA9004000030D806AA800AE
S1135631400000154705E0060567A03004000030F7E
S1135641840F9568BC19EE196640180DC055C6E85C
S11356510F471868BC6A280040000168D80B750B16
S11356615E0B566F7000181D064DE00DE05E006085
S1135671345 4705E0060560D0C0D830F956F7400EA
S11356811819EE1966401E0D30558AE81F471A0D89
S1135691C06AA80040000368586AA8004000010BD3
S11356A1750B5E0B561D464DDE0DE05E00603454F6

S11356B17001006DF619EE7A0000FFE60979010029
S11356C1405C0001140D06790000015C00FEAA0D87
S11356D16647166DF67A0100FFE6097908002A7913
S11356E1000029558E0B870D0E790000015C00FE29
S11356F1B40DE001006D76547019006BA000FFE654
S1065701546BA043
S113570400FFE65219006BA000FFE6586BA000FFF0
S1135714E65654705E0060567A0400FFE6547A0538
S113572400FFE6520F830D1E7FF572501999403026
S1135734694079200 1004C2C695017F068397800CE
S11357446AA900FFE65A69500B5017F0790101006A
S113575401D0531069D869400B5069C00B730B59BE
S10E57641DE94DCC7FF570500D905EE9
S113576F00603454705E006056790E01007A0400B5
S113577FFFE6587A0500FFE6560F830D127FF57289
S113578F501999403469407920010 04C3269501700
S113579FF001D053E00D8017F0683978006AA90043
S11357AFFFE75A69500B5017F001D053E069D869DE
S11357BF400B5069C00B730B590D201D094DC67F4C
S11357CFF570500D905E00603454705E0060567A31
S11357DF0500FFE6580F840D1E7FF57250196640C2
S11357EF381DE64C386B2000FFE6566951191079C6
S11357FF10010017F07901010001D053100D80172C
S113580FF00D6117F10AC178006A2800FFE75A68A3
S113581F9869501B5069D00B5669504EC47FF57071
S113582F500D605E00603454705E0060567A050060
S113583FFFE6540F840D1E7FF57250196640381D15
S113584FE64C386B2000FFE65269511910791001AD
S108585F0017F07901C0
S1135864010001D053100D8017F00D6117F10AC127
S113587478006A2800FFE65A689869501B5069D07B
S11058840B5669504EC47FF570500D605EE9
S1135891006 03454706B2000FFE65417F079010166
S11358A10001D053100D8054706B2000FFE6581790
S11058B1F07901010001D053100D805470F7
S1139F6620282C3034383C20120100010000000860
S1139F76FEFF10000100010102010902270001 0191
S1139F8600C0640904000003000000030705810202
S1139F964000FF070502024000FF07058302400059
S1139FA6FF040309041203550053004200200054 22
S1139FB60045005300540008033 1002E00300022F0
S1139FC603550053004200200054004500530054 3B
S1139FD6002000500052004F004700520041004D40
S1049FE60077
S11399C00023002B0033003B0027002F0037003F0C
S1139FE730302030312030322030332030342030AD
S1139FF735203036203037203038203039203041 73
S113A00720304220304320304420304520304 60A58
S10CA01700253032582000 0A0034
S11358BE01006DF67A0600FFE896010069607A0131
S11358CE41C64E6D5E007F0E7A1000003039010026
S11358DE69E06960198817707A01000080005E0024
S10D58EE7EE80D1001006D76547082
S11358F85E0060560F857A06000000047A000000F7
S113590800180AF00AE07308470E7A00000000182E
S11359180AF00AE00B70400A7A00000000180AF047
S11359280AE00F8601006DF001006F70001C010092

S11359386DF001006DF51A91790000015E00607E3B
S11159487A170000000C0D065E0060345470E8
S11359565E0060560F860F9540040B760B756869DB
S113596668581C8946040C9946F068681750685DA8
S10D5976175519505E006034547099
S11359805E0060560F840FC60F950FE00B766C59BF
S10F5990688946F60FC05E006034547056
S113599C5E0060560F851AE640020B760FD00B752E
S10F59AC680946F60FE05E00603454709A
S11359B85E0060560F850F9601006F710018010095
S11359C86DF101006DF601006DF51A917900000182
S11359D85E00607E7A170000000C5E00603454702D
S11359E80FC446120FD5460E792107FF46120FB38F
S11359F84604156E4A0A1AC47A0500000008186698
S1135A085A005C7C0D9946120FC4461A0FD54616E8
S1135A180FB3461E16E6580002660FA246360FB3AA
S1135A2846325800025A0FA246140FB346105800C4
S1135A38024E0CE60D190FA40FB55A005C881035F9
S1135A481234792C00104C0C1B5910351234792C54
S1135A5800104DF410331232792A00104C0C1B51EC
S1135A6810331232792A00104DF4580000CA1AC4B0
S1135A781AD5186619995800020601006DF201003B
S1135A886DF301006F23000401006DF30100692326
S1135A98795B800001006DF30FF2550E0B970B979E
S1135AA801006D7301006D7254706DF601006DF5A0
S1135AB801006DF401006DF301006DF201006DF159
S1135AC801006DF00100691401006925103410 35D7
S1095AD81FD44218451023
S1135ADE01006F14000401006F2500041FD4440657
S1135AEE0F940FA10FC26816682E0FA00100691440
S1135AFE01006F150004010069020 1006F03000429
S1135B0E796C000F796A000F69191119111911199E
S1135B1E1119796907FF6901111111111111111170
S1135B2E796107FF792907FF5870FEAE0D11587082
S1135B3EFECC10351234103512341035123410 33A6
S1135B4E12321033123210331232794C0080794AEA
S1135B5E00800D901910474C792000364E3E792067
S1135B6E00204D0E793000200FB34702700A0FA3A9
S1135B7E1AA2792000104D14793000100D380DB390
S1135B8E0D2B0DA219AA0D884702700B0C884F140A
S1135B9E113213334402700B1B5040F01AA27A03D6
S1135BAE000000010C6815E84B2A0AB544020B7479
S1135BBE0AA4792C0100585000801 1341335440285
S1075BCE700D0B59EF
S1135BD2792907FF5860006E1AC41AD5580000A627
S1135BE21FA446061FB55870FE8A1AB544087A34B4
S1135BF20000000145061AA4450440121AA41735F1
S1135C0217347A150000000144020B740CE60FC42A
S1135C1246080FD41AD57919FFE00DCC460C0D4C6A
S1135C220DD40D5D19557919FFF0792C0100450842
S1135C32113413350B5940F2792C00804C0810357E
S1135C4212341B5940F20D994E10113413350D992C
S1135C524A08113413350B5940F4732D471C0CD8E1
S1135C62E80B47167A150000000844020B74792CDE
S1135C7201004D06113413350B5911341335113408
S1135C8213351134133579 6C000F101910191019CB
S1135C921019649C101C1006131C01006D70010086

S1135CA2698401006F85000401006D7101006D724A
S1135CB201006D7301006D7401006D756D76547092
S1135CC201F06401586000080792407FF5860006C7A
S1135CD25800007401F064235860006C40520F8531
S1135CE201F06415460E0D44464601F06423464016
S1135CF258000054103112307928001O4C0C1B5CF0
S1135D021031123079280010 4DF4580000BA0FA553
S1135D1201F06435472610331232792A00104C0CF5
S1135D221B5410331232792A00104DF45800009E8E
S1135D321AC4100E131C1AD55800018E790E07FFD0
S1135D421AC41AD5580001621AC418EE790E07FF55
S1135D5219DD790500085800015001006DF60100B4
S1135D626DF501006DF401006DF301006DF20100A8
S1135D726DF101006DF0681E6826156E691C111C19
S1135D82111C111C111C796C07FF692411141114C5
S1135D9211141114796407FF01006D1001006911D8
S1135DA27968000F01006F23000401006922796AF8
S1095DB2000F792C07FF2E
S1135DB85870FF06792407FF5870FF120DCC5870EE
S1135DC8FF160D445870FF40194C791C03FF0DCE84
S1135DD8792EFFCC58D0FF5279480010794A001029
S1135DE81AC47A0500000100103312321031123040
S1135DF8441C0AB144087A100000000145040AA0B3
S1135E0840040AA004011235123444E0401C1AB1BC
S1135E1844087A300000000145041AA040041AA07F
S1135E2804011235DD01123444C2730D46080AB168
S1135E3844020B700AA001F064014702700D792C2B
S1135E4800804C06103512341B5E792E07FF58C0AC
S1135E58FEE40DEE4E2E113413354402700D0DEE93
S1135E684A040B5E40F0732D47180CD8E80B471211
S1135E787A1500000008440A0B74792C00804D023F
S1135E880B5E4020732D471C0CD8E80B47167A1578
S1135E980000000844020B74792C01004D061134EC
S1095EA813350B5E1134FB
S1135EAE133511341335114134133596C000F101E5D
S1135EBE101E101E101E64EC101C100E131C01007D
S1135ECE6D700100698401006F85000401006D711E
S1135EDE01006D7201006D7301006D7401006D752B
S1095EEE01006D76547003
S1135EF401006DF201006DF101006F010004010066
S1135F0469000D82111211121112112112796207FF25
S1135F140D8A7968000F793203FF4B4A79220053C3
S1135F244E44794800107912FFEC4B227922002069
S1135F344C0C0D224F1E103112301B5240F40F90A3
S1135F447912FFE00D224F0C10301B5240F6113032
S1135F540B524BFA796A8000470217B001006D7146
S10D5F6401006D7254701A8040F2C0
S10F5F6E5E007E2CA8014702190054704D
S1135F7A5E007E2C0C884E0419005470F80154708C
S1135F8A5E007E2C0C884604F80154701900547084
S1135F9A01006DF201006DF11AA20F9147224A0620
S1135FAA7902080017B17912041F1B52103144FAFF
S1135FBA1031121210311212103112121031121240
S1135FCA698201006F8100026F8A000601006D7108
S1095FDA01006D7254701A
S1135FE001006DF201006902010069920100 6F0274
S1115FF0000401006F92000401006D725470F2

S1135FFE6DF301006DF201006DF101006DF06C0B9C
S113600E689B0B011B7246F601006D7001006D71EA
S10B601E01006D726D735470F3
S11160265E007E2C1A084704790000015470B6
S113603401006F76001401006D7201006FF200100D
S113604401006D7201006D7301006D7401006D75C3
S1056054547083
S113605601006DF501006DF401006DF301006DF2B1
S113606601006F72001001006DF201006FF600145B
S10B607601006F720004547075
S113607E5E0060567A37000000B27A0300FFE86EC6
S113608E7A0400FFE85E0D0201006FF100AA010021
S113609E6F7600CE01006F7500D20D00466C0100C5
S11360AE6F7000CA460E790004526BA000FFE89A87
S11360BE5A00615001006F7000CA6E080010E807A5
S11360CE17D0460C790005146BA000FFE89A4072B6
S11360DE01006F7000CA6E08001073284610730813
S11360EE470C790105166BA100FFE89A4054010095
S11360FE6F7000CA6E0800107368464601006F7019
S113610E00AA01006BA000FFE8920D2017F001001A
S113611E6BA000FFE85A01006F7000CA01006BA06C
S113612E00FFE8601A8001006BA000FFE864188886
S113613E68C8F8306EF800885C001CBA0D00587001
S113614E0A1C7900FFFF5A006B8E6868A8255860F9
S113615E09CC0B76188868C80FF001006BA000FFFE
S107616EE8720FF0D1
S113617201006BA000FFE8761A8001006BA000FF0C
S1096182E87A01006BA0A6
S113618800FFE87E1A8001006BA000FFE88201008F
S11361986BA000FFE8861A8001006BA000FFE88A65
S11361A840306868A8204718A8234720A82B470A27
S11361B8A82D461C7D40701040167D40703040105D
S11361C87C407330460A7D40704040047D40702017
S11361D80B766868A82D47CAA82B47C6A82047C2CC
S11361E8A82347BEF9206AA900FFE8686869A930AF
S11361F8460AF9306AA900FFE8680B761A8001009D
S11362086BA000FFE86A6868A82A586000800FD06E
S11362180BF001006FF000A07308470A01006F70CC
S113622800A00B70400601006F7000A00F851BF0E3
S113623801006FF000967308470A01006F7000961B
S11362481B70400601006F700096690017F001008B
S11362586BA000FFE86A4C167D40701001006B20AC
S113626800FFE86A17B001006BA000FFE86A0100AD
S10962786B2000FFE86A41
S113627E7A20000002004F0E7D4070007900051851
S113628E6BA000FFE89A0B76686817D0796000FF61
S113629E17F078006A280000A0287328587000862B
S11362AE1A8001006BA000FFE86A4064686817D08B
S11362BE7930003017F001006FF000A07A01000072
S11062CE02001A810F907A010000000A5EA1
S11362DB007EE801006B2100FFE86A1F904D24014B
S11362EB006B2000FFE86A7A010000000A5E007F62
S11362FB0E01006F7100A00A9001006BA000FFE874
S113630B6A400E7D407000790005186BA000FFE812
S113631B9A0B76686817D0796000FF17F078006ADC
S113632B280000A028732846867A00FFFFFFFF0191
S113633B0069B06868A82E586001000B766868A8DE

S113634B2A466C0FD00BF001006FF00096730847D1
S113635B0A01006F7000960B70400601006F70000E
S113636B960F851BF001006FF000A07308470A011D
S113637B006F7000A01B70400601006F7000A069D6
S113638B0017F0010069B04C0A7A00FFFFFFFF0111
S113639B0069B0010069307A20000002004F0E7DC6
S11363AB407000790005186BA000FFE89A0B766824
S11363BB6817D0796000FF17F078006A280000A0F7
S10863CB28732847764A
S11363D01A80010069B04058686817D079300030DE
S11363E017F001006FF000A07A01000002001A818B
S11363F00F907A010000000A5E007EE80100693117
S11364001F904D1C010069307A010000000A5E00F4
S11364107F0E01006F7100A00A90010069B0400E69
S11364207D407000790005186BA000FFE89A0B7699
S1136430686817D0796000FF17F078006A280000B9
S1136440A028732846927A000000002001006BA068
S113645000FFE88E6868A8684708A86C4704A84C42
S11364604610686817D017F001006BA000FFE88E94
S11364700B766868A825587004A4A84558700212C2
S1136480A8475870020CA8584742A863587002E402
S1136490A8644738A865587001F8A866587001F2D7
S11364A0A867587001ECA8694722A86E5870043099
S11364B0A86F4718A87058700398A873587002F80B
S10964C0A8754708A87847
S10D64C647045A00695A01006B20D5
S11364D000FFE88E7A200000006C46440FD00B903A
S11364E001006FF000AA7308470A01006F7000AA49
S11364F00B70400601006F7000AA0F851B9001000E
S11365006FF000967308470A01006F7000961B70C6
S1136510400601006F700096010069025A0066622E
S113652001006B2000FFE88E7A20000000685860AD
S1136530009A6868A875470CA8584708A8784704C4
S1136540A86F46420FD00BF001006FF000AA73084A
S1136550470A01006F7000AA0B70400601006F70BC
S113656000AA0F851BF001006FF000967308470A1D
S113657001006F7000961B70400601006F7000965B
S11365806900177040400FD00BF001006FF00096C8
S11365907308470A01006F7000960B7040060100F4
S11365A06F7000960F851BF001006FF000AA73084F
S11365B0470A01006F7000AA1B70400601006F704C
S10965C000AA690017F0B8
S11365C60F825A0066626868A875470CA858470880
S11365D6A8784704A86F46420FD00BF001006FF06E
S11365E600967308470A01006F7000960B70400609
S11365F601006F7000960F851BF001006FF000AA73
S11366067308470A01006F7000AA1B704006010059
S11366166F7000AA6900177040400FD00BF001009D
S11366266FF000AA7308470A01006F7000AA0B7087
S1136636400601006F7000AA0F851BF001006FF082
S11366460096730847 0A01006F7000961B70400698
S113665601006F700096690017F00F820100693020
S11366667A20FFFFFFFF460A7A00000000010100BF
S113667669B0686817D06DF07A01000000020AF16C
S11366860FA05C00050E0B875A0068CE01006B2035
S113669600FFE88E7A200000004C46420FD00B9094
S11366A60B9001006FF000AA7308470A01006F7090

S10966B600AA0B70400670
S11366BC01006F7000AA0F851B901B9001006FF0F7
S11366CC00967308470A01006F7000961B70404ACE
S11366DC01006F70009640420FD00B900B9001009D
S11366EC6FF000AA7308470A01006F7000AA0B70C1
S11366FC400601006F7000AA0F851B901B900100D0
S113670C6FF000967308470A01006F7100961B71B6
S113671C400601006F7100960F907A010000008013
S113672C0AF15E005FE0010069307A20FFFFFFFF92
S113673C460A7A0000000006010069B0686917D1A7
S113674C01006F70008401006DF001006F70008414
S113675C01006DF07A00000000080AF05C00078667
S113676C0B970B975A0068CE0FD00BF001006FF00C
S113677C00AA7308470A01006F7000AA0B70400649
S113678C01006F7000AA0F851BF001006FF00096DB
S113679C7308470A01006F7000961B7040060100D6
S10967AC6F7000966E08F9
S11367B200015A0069200FD00B9001006FF000AA6C
S11367C27308470A01006F7000AA0B7040060100AC
S11367D26F7000AA0F851B9001006FF0009673087B
S11367E2470A01006F7000961B70400601006F702C
S11367F20096010069000F825E00599C01006FF050
S113680200AA010069317A21FFFFFFFF471201004D
S113681269311F814C0A0100693001006FF000AA3F
S11368220FF001006BA000FFE8761A8001006BA055
S113683200FFE88201006B2000FFE86A01006F712C
S110684200AA1A9001006BA000FFE88A5A1B
S113684F0069680FD00B9001006FF0009673084733
S113685F0A01006F7000960B70400601006F700005
S113686F960F851B9001006FF000AA7308470A016A
S113687F006F7000AA1B70400601006F7000AA0121
S113688F0069007A6000FFFFFF01006FF0009601BF
S113689F0069307A20FFFFFFFF460A7A00000000ED
S11368AF01010069B0790000786DF07A01000000F2
S11368BF020AF101006F7000985C0002CE0B870F84
S11368CFF00F825E00599C01006FF000AA5A006915
S11368DF680FD00B9001006FF000A07308470A01F7
S11368EF006F7000A00B70400601006F7000A00FC7
S11368FF851B900F81730847060F901B7040020F83
S113690F90010069006B2100FFE8666981404AF836
S113691F2568F80FF00F827A000000000101006F65
S113692FF000AA0FF101006BA100FFE8761A9101A5
S106693F006BA146
S113694200FFE88201006B2100FFE86A1A8101005F
S11369526BA100FFE88A400E790005186BA000FFC7
S1136962E89A7D4070006868A86E587001B86A287A
S113697200FFE85E7308586001AC7C407310463632
S113698201006B2000FFE88A4F2C40207A010000AF
S113699200017A0000FFE8685C0013B87A0000FF88
S11369A2E88A010069011B710100698101006B2002
S11369B200FFE88A4ED601006B2000FFE87E4624E2
S11369C201006B2000FFE882461A01006B2000FFE2
S11369D2E886461001006F7100AA0FA05C001374D1
S11369E25A006AEC01006B2000FFE8720FA11A90B3
S11369F201006FF0009C01006FF000A04F30010016
S1136A026F71009C0FA05C00134A40227A000000C1
S1136A1200880AF07A01000000015C0013367A0054

S1136A2200FFE87E010069011B7101006981010019
S1136A326B2000FFE87E4ED401006B2000FFE87656
S1136A4201006B2100FFE8721A9001006FF0009CB5
S1136A5201006F7100A00A8101006FF100A00F8095
S1136A624F3601006F71009C01006B2000FFE8723A
S1136A725C0012E040227A00000000880AF07A01EA
S1136A82000000015C0012CC7A0000FFE8820100E2
S1136A9269011B710100698101006B2000FFE8821B
S1096AA24ED401006F71E8
S1116AA800AA01006F7000A01A8101006B208C
S1136AB600FFE8765C00129840227A000000008806
S1136AC60AF07A01000000015C0012847A0000FFDC
S1136AD6E886010069011B710100698101006B20D1
S1136AE600FFE8864ED47C407310473601006B20C6
S1136AF600FFE88A4F2C40207A01000000017A004B
S1136B0600FFE8685C0012487A0000FFE88A01008B
S1136B1669011B710100698101006B2000FFE88A8E
S1136B264ED60B76404001006FF600A6400E0100DC
S1136B366F7000A60B7001006FF000A601006F7066
S1136B4600A668086EF8009547086E780095A82594
S1136B5646DC01006F7100A61AE10FE05C0011F03C
S1136B6601006F7600A6686847145C0012900D005A
S1136B76460C6A2800FFE85E73085870F5D45C007B
S1136B86125E6B2000FFE8667A17000000B25E0013
S1136B96603454705E0060567A37000000247A052C
S1096BA6000000040AF5E3
S1136BAC01006FF0001801006FF1002001006FF17C
S1136BBC001C18AA689A6F72003C0C22463CAA5817
S1136BCC472CAA644712AA69470EAA6F4712AA75E3
S1136BDC4706AA78471840227A000000000A4006AC
S1136BEC7A000000000801006FF00014400C7A00DA
S1136BFC0000001001006FF000146F70003C79204E
S1136C0C006447067920006946120 1006F70001872
S1136C1C4C0A01006F74001817B4400601006F741E
S1136C2C00181AE61AB30FC4467401006B2000FF58
S1136C3CE86E476AF83068D87A06000000014062B3
S1136C4C0FD00AE00F82010069F00FC001006F71D1
S1136C5C00145E0085120100697068890FA0680832
S1136C6CA8094E0C0FD00AE0680989306889401EC8
S1136C7C6F70003C79200078460A0FD00AE068094F
S1136C8C895740080FD00AE06809893768890FC013
S1096C9C01006F710014FA
S1116CA25E0085120F840B760FC4469E6A288F
S1136CB000FFE85E732847626F70003C0C00465A81
S1136CC0A858471EA86F4706A8784716404C0100EE
S1136CD06F70001847440FE00B760AD0F9306889CB
S1136CE0403801006F700018473001006F700020BA
S1136CF00B7001006FF000201B70F93068890100F0
S1136D006F7000200B7001006FF000201B706E7914
S1136D10003D68897A03000000020FB00AE00F8388
S1136D2001006FF600146F70003C79200064470681
S1136D3079200069467201006B2000FFE86E460867
S1136D4001006F700018476001006F7000184D421A
S1136D506A2800FFE85E733847160B7301006F70F3
S1136D6000200B7001006FF000201B70F92B4036E0
S1136D706A2800FFE85E7348472E0B7301006F70AB
S1136D8000200B7001006FF000201B70F920688950

S1136D9040160B7301006F7000200B7001006FF041
S1096DA000201B70F92D19
S1136DA668897A0600FFE87601006F70001C680C9C
S1136DB6AC2B4708AC2D4704AC20460E01006F7080
S1136DC6001C0B70010069E040466A2800FFE85E7C
S1136DD6732847326F70003C0C004634A858470AA4
S1136DE6A86F4714A8784702402601006F70001C5D
S1136DF60BF0010069E0401801006F70001C0B7076
S1136E06010069E0400A01006F70001C010069E09F
S1136E167A0400FFE86E010069401FB04F140100B9
S1136E266946010069441AB401006BA400FFE882B5
S1136E36400C0FB61A8001006BA000FFE8827A04AB
S1136E4600FFE88A01006F70001C680BAB2B470834
S1136E56AB2D4704AB20463801006B2000FFE86AE0
S1136E661FE04F2C6A2800FFE868A830462201007D
S10D6E766B2000FFE86A1AE07A01BE
S1136E8000FFE882010069120A82010069921A80F8
S1136E90010069C0401E01006B2000FFE86A1FE08B
S1136EA04F0C01006B2000FFE86A1AE040021A80D1
S1136EB0010069C001006F7600141B76401A0100BF
S1136EC06F7000200B7001006FF000201B700FE14A
S1136ED01B760AD1681968890FE64CE201006F70CE
S1136EE00020189968897A17000000245E00603436
S1136EF054705E0060567A37000000647A0300FF26
S1136F00E85E7A04000000040AF47A0500FFE86EE4
S1136F100F866FF1005001006FF6005CF83068C80F
S1136F207A000000007C0AF07A010000A0205E00D5
S1136F3060260D00587000B87A000000002F0AF098
S1136F4001006DF07A000000002E0AF001006DF0E0
S1136F5001006F70008801006DF001006F70008800
S1136F6001006DF07A01000000320AF17A0000009E
S1096F7000115E007F2EFC
S1136F767A17000000100D024752188868E80100CE
S1136F86695001006B2100FFE86A1F904F0601005C
S1136F966950400801006B2000FFE86A01006BA0FE
S1136FA600FFE88A7A0600FFE8680D207920FFFFD4
S1136FB6470C79200001460CF82B5A007CDEF82D8D
S1046FC65A6D
S1136FC7007CDEF82A68E85A007CE07A00000000BB
S1136FD71101006DF00FC10B717A00000000260A42
S1136FE7F05E0083140B97400CF8016EF8002F181E
S1136FF7886EC800017A000000000101006FF000ED
S113700760400E01006F7000600B7001006FF000AD
S11370176001006F7000600AC06808A83047E40188
S1137027006F7000607A20000000014F7A01006F43
S11370377000600AC05E00599C0F8201006F7000E8
S1137047601B7001006F71002A0A8101006FF10054
S11370572A7A000000000101006FF0005840300158
S1137067006F7000600AC001006F7100580AC168A1
S113707708689801006F7000580B7001006FF000EB
S11370875801006F7000600B7001006FF000600122
S1137097006F7000580FA11F904FC401006F70005D
S11370A7580AC0189968890FC00B7001006FF00068
S10870B7445E00599C3A
S11370BC01006FF000601A8001006FF000580100AE
S11370CC6F7000607A01000000025E007EE8010030
S11370DC6FF0004001006F70004401006F7100609D

S11370EC0A901B7001006FF0003C404E01006F7062
S11370FC004401006F7100580A9001006FF0004CBE
S113710C68086EF8005701006F71003C01006F7244
S113711C00581AA101006FF1004801006F72004C76
S113712C681968A901006F710048689801006F71B4
S113713C00580B7101006FF1005801006F7000587B
S113714C01006F7100401F904DA201006F70006031
S113715C461AF8306EC800017A00000000010100E5
S113716C6FF000601A8001006FF0002A6F700050FE
S113717C7920006747067920004746 3C7D307050E4
S113718C01006F70002A01006F7100600A901B7080
S113719C7A20FFFFFFFC4D0C01006B2100FFE86E12
S10971AC1F904F0C6F70F1
S11371B200501BD06FF000504008790000666FF05A
S11371C200506E78002FA80146246838E8186EF83C
S11371D20057A8084706A810470A401A0FE00B7683
S11371E2F92B40100FE00B76F920688940080FE075
S11371F20B76F92D68896F7000507920006658600C
S1137202077201006F70002A58D001861A80401A53
S11372120FE00B7601006F7100580AC10B716819F8
S1137222688901006F7000580B7001006FF00058FD
S113723201006F7100601F904DD61A800F8201000A
S11372426BA600FFE87601006F70002A01006BA0B5
S113725200FFE8827C307350460A01006B2000FF76
S1137262E86E4E067C307320470E0FE00B76F92E44
S113727268890FA00B700F827C307350470C7C30EF
S1137282735047247C307320471E01006BA600FF16
S1137292E87A0100695001006BA000FFE886010053
S10972A269500FA10A81EF
S11372A80F9201006F70002A0FA10A9001006F71FD
S11372B800600A9001006FF0006001006F71005CCC
S11372C86819A92B4708A92D4704A920465201008C
S11372D86B2000FFE86A01006F7100601F904F4048
S11372E86A2800FFE868A830463601006F70005C22
S10972F80B7001006BA006
S11372FE00FFE87201006B2000FFE86A01006F7166
S113730E00601A9001006BA000FFE87E1A80010056
S113731E6BA000FFE88A5A007CDC01006B2000FFA3
S113732EE86A01006F7100601F904F4C01006B20E3
S113733E00FFE86A01006F7100601A9001006BA0F4
S113734E00FFE88A6E78002FA80146186E78002F8A
S113735EA801586009786E780057A8084706A81048
S113736E5860096A7A0000FFE88A010069011B71FF
S113737E010069815A007CDC1A8001006BA000FFBA
S113738EE88A5A007CDC01006F70006001006F71A7
S113739E002A0A8101006FF1002A010069520AA135
S11373AE01006FF100521F814C120F914D0E01001F
S11373BE6F7100520B710FC05C00092201006F70D8
S11373CE002A58F0028A6848A83046147A00000052
S11373DE000101006FF0004C01006FF00052402CD1
S10973EE1A8001006FF09C
S11373F4004C1A8001006FF0005201006F70002AE4
S11374040B7001006FF0002A01006F7000600B70B5
S113741401006FF000601A8001006FF00058403AD9
S11374240FE00B7601006F71004C0AC1681968897B
S113743401006F70002A1B7001006FF0002A010025
S11374446F7000580B7001006FF0005801006F70EB

S1137454004C0B7001006FF0004C01006F70002AA8
S11374644EBE0FE00B76F92E688940240FE00B76AD
S113747401006F71004C0B7101006FF1004C1B7123
S11374840AC168196889010069501B70010069D039
S113749401006F7000580B7001006FF00058010079
S11374A46F70004C01006F7100521A9001006F71EC
S11374B400601F904C0A01006B2000FFE86E4EAC85
S11374C47C307350470C7C307350472E7C307320D0
S11374D4472801006950010006F7100580A810100B7
S10774E46FF10058E9
S10774E801006BA68B
S11374EC00FFE8760100695001006BA000FFE88201
S11374FC40226E68FFFFA830461A40101B7601002D
S113750C6F7000581B7001006FF000586E68FFFF1E
S113751CA83047E87C307320464A6E68FFFFA82EDC
S113752C464201006B2000FFE88247067C30735013
S113753C47321B7601006F70005801006B2100FF6E
S113754CE8821A901B7001006FF000581A8001003A
S113755C6BA000FFE88201006F70005C01006BA060
S113756C00FFE87601006F70005C6808A82B4708E1
S113757CA82D4704A820465001006B2000FFE86AA1
S113758C01006F7100581F904F3E6A2800FFE86896
S113759CA830463401006F70005C0B7001006BA0C7
S11375AC00FFE87201006B2000FFE86A01006F71B5
S11375BC00581A9001006BA000FFE87E1A800100AE
S11375CC6BA000FFE88A406201006B2000FFE86AB1
S10975DC01006F7100586D
S10B75E21F904F4601006B20CE
S11375EA00FFE86A01006F7100581A9001006BA04E
S11375FA00FFE88A6E78002FA80146146E78002FE0
S113760AA80146286E780057A8084704A810461C04
S113761A7A0000FFE88A010069011B710100698190
S113762A400A1A8001006BA000FFE88A01006B2060
S113763A00FFE87601006F71005C1F9058600692A4
S113764A01006B2000FFE87201006BA000FFE876DF
S113765A5A007CDC6848A83046147A00000000010E
S113766A01006FF0004C01006FF00052402A1A80AB
S113767A01006FF0004C01006FF0005201006F70BF
S113768A002A0B7001006FF0002A01006F7000607E
S113769A0B7001006FF0006001006F70002A4F1435
S11376AA7A000000000101006FF0004C0FE00B7636
S11376BA684940060FE00B76F93068897A000000C2
S11376CA000101006FF000580FE10B76FA2E689A59
S10976DA01006F7000586F
S11376E00B7001006FF0005801006F70002A58C042
S11376F0009E01006BA600FFE87601006F70002A70
S113770017B0010069511F814F2001006F70002ADB
S113771017B001006BA000FFE88201006F70002A20
S113772001006F7100581A8140180100695001006F
S11377306BA000FFE8820100695001006F710058DF
S11377400A8101006FF1005801006F70002A0100E7
S113775069510A81010069D140360FE00B760100BF
S11377606F71004C0AC168196889010069501B7068
S1137770010069D001006F7000580B7001006FF0B9
S113778000 5801006F70004C0B7001006FF0004C4B
S113779001006F70004C01006F7100521A900100DC
S11377A06F7100601F904C0A01006B2000FFE86EB0

S11377B04EA87C307350470C7C30735047347C3078
S11377C07320472E010069504F4A01006BA600FF4A
S10977D0E87A0100695094
S10777D601006BA0A0
S11377DA00FFE8860100695001006F7100580A81B1
S11377EA01006FF1005840226E68FFFFA830461A65
S11377FA40101B7601006F7000581B7001006FF078
S113780A00586E68FFFFA83047E87C307320467043
S113781A6E68FFFFA82E466801006B2000FFE8820E
S113782A460A01006B2000FFE88647067C30735046
S113783A474E1B7601006F70005801006B2100FF51
S113784AE8821A9001006B2100FFE8861A901B70E8
S113785A01006FF000581A8001006BA000FFE88650
S113786A01006BA000FFE88201006F70005C010059
S113787A6BA000FFE8761A8001006BA000FFE87A8C
S113788A01006F70005C6808A82B4708A82D4704FD
S113789AA820465001006B2000FFE86A01006F71BF
S11378AA00581F904F3E6A2800FFE868A830463404
S11378BA01006F70005C0B7001006BA000FFE8729F
S10778CA01006B202B
S11378CE00FFE86A01006F7100581A9001006BA067
S11378DE00FFE87E1A8001006BA000FFE88A406279
S11378EE01006B2000FFE86A01006F7100581F90C2
S11378FE4F4601006B2000FFE86A01006F710058CC
S113790E1A9001006BA000FFE88A6E78002FA80181
S113791E46146E78002FA80146286E780057A808E3
S113792E4704A810461C7A0000FFE88A010069018B
S113793E1B7101006981400A1A8001006BA000FFD0
S113794EE88A01006B2000FFE87601006F71005C8E
S113795E1F90461001006B2000FFE87201006BA020
S113796E00FFE8765A007CDC010069500B700100C1
S113797E6F7100601F904C0C010069510BF10FC029
S113798E5C00035A6848A830460E7A0000000001D6
S113799E01006FF0004840241A8001006FF0004888
S11379AE01006F70002A1B7001006FF0002A0100A6
S11379BE6F7000600B7001006FF0006001006F705C
S10979CE006001006F716F
S11379D400481A9001006F72002A0A8201006FF2B4
S11379E4002A0FE00B760AC1681968897A0000003F
S11379F4000101006FF000580FE10B76FA2E689A2C
S1137A0401006F7000580B7001006FF00058010003
S1137A146F700048402E0FE00B7601006F71004C2D
S1137A240AC168196889010069501B70010069D093
S1137A3401006F7000580B7001006FF000580100D3
S1137A446F70004C0B7001006FF0004C01006F71FC
S1137A5400601F904E0A01006B2000FFE86E4EB6D3
S1137A647C307350470C7C307350472E7C3073202A
S1137A74472801006BA600FFE87601006950010066
S1057A846BA0F2
S1137A8600FFE8820100695001006F7100580A8106
S1137A9601006FF1005840226E68FFFFA830461AB6
S1137AA640101B7601006F7000581B7001006FF0C9
S1137AB600586E68FFFFA83047E87C307320464ABB
S1137AC66E68FFFFA82E464201006B2000FFE88286
S1137AD647067C30735047321B7601006F7000589F
S1137AE601006B2100FFE8821A901B7001006FF002
S1137AF6005801006F70005C01006BA000FFE87680

S1137B061A8001006BA000FFE8826F700050792095
S1137B16006546080FE00B76F96540060FE00B7625
S1137B26F945688901006F7000580B7001006FF00A
S1137B36005801006F70002A4D0A0FE00B76F92BEF
S1137B46688940160FE00B76F92D688901006F707E
S1137B56002A17B001006FF0002A01006F70005869
S1137B660B7001006FF0005801006F70002A7A2035
S1097B76000000644D0E47
S1137B7C0BF601006F7000580B800B7040140FE074
S1137B8C0B76F9306889F83068E801006F7000589B
S1137B9C0BF001006FF000580FE00B7001006FF059
S1137BAC003040360FE01B76010069F001006F7066
S1137BBC002A7A010000000A5E007EE88930010089
S1137BCC6970688901006F70002A7A010000000A4D
S1137BDC5E007EE801006FF0002A01006F70002A3E
S1137BEC4EC201006F76003001006F70005C6808B4
S1137BFCA82B4708A82D4704A820465001006B204A
S1137C0C00FFE86A01006F7100581F904F3E6A280D
S1137C1C00FFE868A830463401006F70005C0B70FD
S1137C2C01006BA000FFE87201006B2000FFE86A03
S1137C3C01006F7100581A9001006BA000FFE87EE1
S1137C4C1A8001006BA000FFE88A406201006B20E0
S1137C5C00FFE86A01006F7100581F904F46010046
S1057C6C6B2088
S1137C6E00FFE86A01006F7100581A9001006BA0C3
S1137C7E00FFE88A6E78002FA80146146E78002F55
S1137C8EA80146286E780057A8084704A810461C7A
S1137C9E7A0000FFE88A010069011B710100698106
S1137CAE400A1A8001006BA000FFE88A01006B20D6
S1137CBE00FFE87601006F71005C1F904610010013
S1137CCE6B2000FFE87201006BA000FFE8761888B6
S1137CDE68E87A17000000645E00603454705E003A
S1137CEE60560F850F960FD00AE06808A8344F52DE
S1137CFE0FD00AE0F93068891B760FD00AE06809C5
S1137D0E8901688940200FD00AE06808A8394F140A
S1137D1E0FD10AE1681888F668986E18FFFF88017C
S1137D2E6E98FFFF1B760FE64EDC6E580001A839E6
S1137D3E4F106E58000188F66ED8000168588801FE
S1137D4E68D85E00603454705E0060561B977A03E9
S1097D5E00FFE8607A0655
S1137D6400FFE864010069F00F9501006B2100FF37
S1137D74E85A7A21000000001462601006DF50F81BF
S1137D84010069305E0084B80B97010069300AD0A2
S1137D94010069B0010069600AD0010069E0403A5A
S1137DA419444030010069700B70010069F01B70C5
S10F7DB4680817D00100693101006B2240
S1137DC000FFE8925D207920FFFF47120100696000
S1137DD00B70010069E00B5417F41FD44DCA0B97C5
S1137DE05E006034547001006B2000FFE85A7A2073
S1137DF000000001460C01006B2000FFE8601899A9
S1137E006889547001006B2000FFE85A7A20000053
S1137E1000014604190054700 1006B2000FFE86064
S10F7E206E090010E94017D10D105470DA
S10BA020000000000000000035
S1137E2C01006DF501006DF401006DF301006DF2BD
S1137E3C01006DF101006D120100691301006F0166
S1137E4C0004010069000D840D8C796C7FF0792C92

S1137E5C7FF047540DAC796C7FF0792C7FF047564B
S1137E6C0D8C65AC4B5E1AB144087A3000000001EE
S1137E7C45141AA0451001F06410475A0C444B0AE0
S1137E8C79000002400C0C444BF6190040047900B5
S1137E9CFFFF01006D7101006D7201006D73010034
S1137EAC6D7401006D7554700F85796D000F01F0C1
S1137EBC641546DA409E0FA5796D000F01F0643509
S1137ECC46CC409C01F064207A607FFFFFFF46ACF8
S10F7EDC01F0643146A67900000140B6B5
S1137EE86DF20D820C2A4A0217B00D994A04D2800A
S1137EF817B15E0085120C224A0217B00CAA4A0277
S1097F0817B16D72547005
S1137F0E01006DF301006DF20D8252120D935203B7
S1137F1E52100928093801006D7201006D735470F7
S113A028202020202020202020606060602020E5
S113A038202020202020202020202020202020202015
S113A048481010101010101010101010101010CD
S113A058848484848484848484101010101010106D
S113A068108181818181810101010101010101C6
S113A078010101010101010101010110101010107A
S113A088108282828282820202020202020202297
S113A098020202020202020202020210101010203F
S113A0A8000000000000000000000000000000A5
S113A0B8000000000000000000000000000000095
S113A0C8000000000000000000000000000000085
S113A0D8000000000000000000000000000000075
S113A0E8000000000000000000000000000000065
S113A0F8000000000000000000000000000000055
S113A1080000000000000000000000000000000044
S109A1180000000000003E
S10DA11E0000000000000000000034
S1137F2E5E0060567A370000003C7A040000000CB5
S1137F3E0AF47A05000000140AF501006FF0003808
S1137F4E01006FF1003401006F73005C7A020000D0
S1137F5E000801006FF2002C19226FF2002A7A0634
S1137F6E000000540AF6686AEA7F46620FE00B705F
S1137F7E7A01000000075E00854E0D0047501A80FF
S1137F8E401A01006F70003401006F7100380A90BF
S1137F9E1899688901006F7000380B7001006FF03B
S1137FAE00387A20000000084DD81A80010069B00D
S1137FBE7C607370470A01006F700060F9FF400820
S1137FCE01006F700060F90168895A0083067A0018
S1137FDE0000001C0AF001006DF07A010000002879
S1137FEE0AF101006F70005C01006DF001006F700B
S1137FFE005C01006DF001006F70006C5E0088DEA6
S113800E7A170000000C01006F7000606808A8FF6B
S107801E460C7A008F
S1138022000000540AF07D0072706F700024792002
S1138032070FF464A6E680001A8F0463A0FE00BF0CC
S11380427A01000000065E00854E0D0047280100FC
S11380526F7000606808A8014608790000015A00A1
S1138062830801006F7000606808A8FF461079005A
S1138072FFFF5A008308790000025A0083086F70D9
S1138082002479300FF6FF000244D1E01006F704E
S1138092005801006DF001006F70005801006DF08F
S11380A25E00866E0B970B97401E01006F7000589F
S11380B201006DF001006F70005801006DF05E0069

S11380C2866E0B970B971B500D0618886EF80023CC
S11380D20B560D6017F001006F7100381A90010002
S11380E269B0010069F10FD1010069705E00878AEE
S11380F201006F70003801006F71002C5E007EE892
S113810201006FF000301AE67A0000000008010057
S10981126F7100301A90AA
S11381180F82401401006F7000300AE00AD00FD1BB
S11381280AE1680868980B760FA01F864DE6400A97
S11381380FD00AE0189968890B767A2600000008A0
S11381484DEE6F7000326F78002E52806F71003AD7
S113815819016DF17A01000000080FD05E00871243
S11381680B8701006F7600381B760FC10FE05E00A6
S1138178878A0FE001006F71002C5E007EE8010022
S11381886FF000301AE67A000000000801006F71F2
S113819800301A900F82401401006F7000300AE01B
S11381A80AC00FC10AE1680868980B760FA01F86FA
S11381B84DE6400A0FC00AE0189968890B767A26BB
S11381C8000000084DEE6F7000326F78002E528069
S11381D86F71003A19011B516DF17A010000000813
S11381E80FC05E0087120B8701006F700034010017
S11381F86DF00100693101006DF17A01000000247E
S10982080AF16F70002C67
S113820E5E0085760B970B977A06000000040AF63C
S113821E7A000000000801006DF001006F71003854
S113822E0FE05E0084B80B977A000000000801008F
S10A823E6DF00FD10FE05EAC
S11382450089820B970D004F12790000016FF00032
S11382552A010069300B705A0082FE7A0000000083
S11382650801006DF001006F7100380FE05E0084B6
S1138275B80B977A000000000801006DF00FD10FCD
S1138285E05E0089820B970D004632010069300BD1
S113829570010069B001006F70003401006DF001D9
S11382A500693301006DF37A01000000240AF16FC0
S11382B570002C5E0085760B970B9740447A00007F
S11382C500000801006DF001006F7100380FE05EDA
S11382D50084B80B977A000000000801006DF00FC9
S11382E5C10FE05E0089820B970D004C146F70007F
S11382F52A460E010069301B70010069B05A0081DE
S1128305F019007A170000003C5E0060345470DA
S11383145E0060567A370000002C0FF30F860100CD
S11383246FF100107A000000000101006FF00018E3
S113833401006F7500440A95188868D87A00000014
S1138344000801006DF00FE17A000000000C0AF050
S11383545E0084B80B9701006F7000445A0084A434
S11383640FA00B707A01000000055E007EE80B701D
S113837410307A06000000081A867A00000000080C
S11383841AE001006DF00FA11031103110317A1190
S11383940000A1280AE10FB00AE05E0084B80B973D
S11383A47A000000000801006FF000281A80010021
S11383B46FF0002001006FF0001C7A040000000835
S11383C41AE401006FF400145A00846601006F700C
S11383D4001401006DF00FB10AE17A000000000CF3
S11383E40AF00AE05E0089820B970D004D3C010000
S11383F46DF40FB00AE001006DF07A010000001083
S10984040AF10AE11A80EF
S113840A5E0088320B970B9717F001006FF0001884
S113841A01006F70002801006F7100200A810100BA

S113842A6FF1002001006F7000287A01000000023A
S113843A5E007EE801006FF00028473079000001F2
S113844A6DF00FB00AE00FC15E0087B80B87010019
S113845A6F70001C0B7001006FF0001C01006F703D
S113846A001C7A200000000458D0FF5A01006F70E4
S113847A0018461C6E78002388306CD84004F83004
S113848A68D81B7501006F7000101F8544F04012F5
S113849A6E78002388306CD80FA01B700F8258C0E7
S11184AAFEB87A170000002C5E006034547098
S113A1280000000000000000800000000000000050CC
S113A1380000000000000320000000000000001F4092
S113A1480000000000001388000000000000C35000A
S113A15800000000007A12000000000004C4B400EC
S113A168000000002FAF080000000001DCD65000FB
S113A17800000012A05F2000000000BA43B74000AF
S113A18800000746A5288000000048C27395000018
S113A1980002D79883D20000001C6BF52634000018
S10BA1A8011C37937E0800003F
S11384B85E0060560F850F9401006F7600181FC584
S11384C847401FC544200FD30FC21AC440120FB030
S11384D80B730FA10B710F921B71681968890B74C9
S11384E81FE445EA401C0FD30AE30AE40FC21AC487
S11384F8400C0FA01B700F8268086CB80B741FE444
S10D850845F00FD05E00603454709C
S113851201006DF20D9946100D82177253120DA8C8
S113852253100D810D28401E0F920D81177179088A
S1138532001012101231 1AA144020AA11B5846F26A
S10F8542121017101770010 06D725470B6
S113854E5E0060560F840F951AE6400E0FC00AE0C8
S113855E680947041900400A0B761FD64DEE7900C1
S10B856E00015E00603454704B
S11385765E0060567A370000000C7A0500000001A1
S11385860AF50D0C0F967904000801006DF57A01C2
S11385960000000E0AF101006F70002817B05E009C
S11385A68A480B9701006DF66DFC7A0000000010F7
S11385B60AF00FD15E0089C00B970B87790C003F39
S11385C66F70000A190C0DC017F001D053400D0E41
S11385D67900004219C017F001006DF07A0100001E
S11385E600090FD05E008C940B977906000840149F
S11385F60D6019E017F00AD00D6117F10AD168086A
S113860668981B561DE64CE8400C0D6017F00AD01F
S1138616189968891B560D664CF00DE05240190CEB
S11386266DFC7A01000000090FD05E0087B80B8746
S11386367900003F6FF0000A7A01000000400FD076
S11386465E008BAA7A000000000801006DF00FD1CE
S113865601006F70002C5E0084B80B977A17000038
S1098666000C5E0060340D
S105866C547045
S113866E01006DF27A370000001A7A00000000183C
S113867E0AF001006F71002601006DF101006F71A8
S113868E002601006DF17A01000000080AF10100D5
S113869E6DF15E008D307A170000000C6F710018BB
S11386AE0FF05E008FB00F817A020000A1B07A0046
S11386BE000000080AF05E00909C7A000000001093
S11386CE0AF001006F71000C01006DF101006F7172
S11386DE000C01006DF17A01000000080AF101009F
S11386EE6DF15E008E367A170000000C7A000000E2

S11386FE00100AF05E005EF47A170000001A010003
S107870E6D725470C1
S10BA1B03FD34395810624DD32
S11387125E0060561B971B87010069F00F946F7907
S1138722001E790D0008199D4754790100FF0DD2EF
S11387321A0A4B04101140F80C9B18DD1B740FC668
S1138742402801006974 0AE468450CB816850FC015
S11387520D915E008F6C684814D868C80DD01A0852
S11387624B04110540F80C5D1B760FE64CD40CDD6F
S1138772470679010001400219 1117F10F900B8787
S10B87820B975E006034547094
S113878A5E0060560F860F957A000000000801000C
S113879A6DF01036103610367A010000A1B80AE1DE
S11187AA0FD05E0084B80B975E0060345470ED
S113A1B8000000000000000010000000000000058E
S113A1C80000000000000001900000000000000 7DEE
S113A1D8000000000000027100000000000000C35C0
S113A1E80000000000003D09000000000001312DBF
S113A1F8000000000005F5E100000000001DCD652A
S113A208000000000009502F90000000002E90EDDDD
S113A218000000000E8D4A510000000048C27395EB
S113A228000000016BCC41E9000000071AFD498DCD
S113A2380000002386F26FC1000000B1A2BC2EC546
S113A248000003782DACE9D900001158E460913D72
S113A258000056BC75E2D6310001B1AE4D6E2EF545
S113A268000878678326EAC9002A5A058FC295ED44
S113A27800D3C21BCECCEDA10422CA8B0A00A425AD
S113A28814ADF4B7320334B96765C793FA10079D61
S11387B85E0060567A370000000A0F83010069F1F2
S11387C86F790022790C0008199C4752FAFF0DC0F3
S11387D81A084B04110A40F80CA218CC1AE64026D2
S11387E80FB50AE568540C2816840FD00D915E0066
S11387F88F8E685814C868D80DC01A084B04100423
S113880840F80C4C0B76010069701F864DD20CCCD6
S1138818470679010001400219 1117F10F907A17E1
S10D88280000000A5E006034547083
S11388325E0060567A370000000C01006FF00004FE
S11388420F9601006F7500280AD61B7601006F731D
S113885200240AD31B737A02000000011AC44044A5
S1138862686817506839175119 1017F01AC00100B8
S113887269F04C14010069717A11000001000100D2
S113888269F17900000140021900 17F00F84010019
S1138892697047041A800F826E78000368E81B75BB
S11388A21B761B730FD546B87A2400000001460ECF
S11388B201006F70000446067900FFFF40120FA00B
S11388C27A2000000001460419004004 79000001E7
S10F88D27A170000000C5E006034547044
S11388DE5E0060567A03000000070F820F950100B9
S11388EE6F7600207A04000000180AF4694179615A
S11388FE800079218000460679 01FFFF400479014B
S113890E00010FA06889694079607FF01190119082
S113891E1190119069D07A000000000701006DF0EC
S113892E0B740FC10FE05E0084B80B977900000340
S113893E6DF00FB10FE05E0087120B8769504F0683
S113894E7D607070402869500B5069D07D607270E5
S113895E4016790000016DF00FB10FE05E00871233
S113896E0B8769501B5069D07C60737047E45E00BF

S107897E603454709A
S11389825E0060560F860F9301006F7400180FE5A7
S11389920FC44604190040201AE6400A6C586C3989
S11389A21C9846060B761FC645F21B75685817506E
S11189B21B73683B175319305E00603454701A
S11389C05E0060567A37000000100FF60F850F9493
S11389D069506F710028091069D001006DF601001C
S11389E06F71002E0FC05E0093780B977C607370DD
S11389F0470869500B5069D0400C7A010000001001
S1138A000FE05E0093267A000000004201006DF043
S1138A107A01000000100FE05E008C940B976E68E3
S1138A200008E8E06EE800087A0000000009010091
S1138A306DF00FE10FC05E0084B80B977A1700004A
S10B8A4000105E006034547065
S1138A485E0060567A37000000187A0500000009B6
S1138A580AF50F840F920FC44D087A030000000132
S1138A6840147A03FFFFFFFF0FC07A01FFFFFFFFE8
S1138A785E007F0E0F840FC07A010000001B5E00AA
S1138A887EE80F860FC07A010000001B5E007EE8B7
S1138A980F947A2300000001462E7A000000000894
S1138AA801006DF00FE11031103110317A1100001F
S1138AB8A2980FD05E0084B80B970FE010307800AF
S1138AC86B210000A36840320FC446021B767A006C
S1138AD80000000801006DF00FE110311031103172
S1138AE87A110000A3000FD05E0084B80B970FE043
S1138AF8103078006B210000A3826FF100120FC4BD
S1138B0847747A23FFFFFFFF460A7A000000001B21
S1138B181AC00F8401006F70003001006DF00FA1BF
S1138B280FC05E0095060B970FE646087A230000F0
S1078B38000147628C
S1138B3C01006F70003001006DF00FA069006DF043
S1138B4C7A00000000180AF00FD15E0089C00B9761
S1138B5C0B877A01000000400FD05E008BAA7920AE
S1138B6C0001460E6F7000120B506FF000127D5017
S1138B7C70700FA06F71001269817A0000000008F9
S1138B8C01006DF00FD101006F7000345E0084B8EA
S1118B9C0B977A17000000185E0060345470C7
S113A29880000000000000000CECB8F27F4200F3A87
S113A2A8A70C3C40A64E6C5286F0AC99B4E8DAFD94
S113A2B8DA01EE641A708DEAB01AE745B101E9E4F0
S113A2C88E41ADE9FBEBC27DE5D3EF282A242E812D
S113A2D8B9A74A0637CE2EE195F83D0A1FB69CD991
S113A2E8F24A01A73CF2DCD0C3B8358109E84F072D
S113A2F89E19DB92B4E31BA99E74D1B791E07E4803
S113A308C428D05AA4751E4CF2D56790AB41C2A499
S113A318964E858C91BA2655BA121A4650E4DDEC4E
S113A328E65829B3046B0AFA8E938662882AF53FA6
S113A338B080392CC4349DEDDA7F5BF5909668497B
S113A348873E4F75E2224E68A76C582338ED2623C3
S113A358CF42894A5DCE35EB8049A4AC0C5811AE87
S113A3680000005900B3010D016601C0021A02730F
S113A37802CD0327038003DA0434FFA6FF4CFEF261
S109A388FE99FE3FFDE516
S111A38EFD8CFD32FCD8FC7FFC25FBCBFB7263
S1138BAA5E0060567A370000000C0F840F957A0630
S1138BBA000000081AB30FD00FE15E007EE80AC076
S1138BCA0F820FD00FE15E007EE8790000801A0958

S1138BDA4B04119040F86EF8000511086EF8000B6B
S1138BEA11086EF800041B75010069F50FD00FE137
S1138BFA5E007EE80AC00F85010069700FE15E001E
S1138C0A7EE8790600801A094B04119640F80FA0F2
S1138C1A68086E790005169847280FA068066E78CB
S1138C2A000B166846086E780004166847087A032C
S1138C3A00000001400C685816E847067A03000052
S1138C4A00017A2300000001463240140CE8175051
S1138C5A1710685916896 8D9100E46041B75FE0148
S1138C6A685816E847041FC544E21FC545086858F3
S1138C7A14E868D84006790000014 00219007A17FF
S10D8C8A0000000C5E00603454701B
S1138C945E0060561B971B970F82010069F17A03EC
S1138CA40000000801006F7000200FB15E007EE831
S1138CB40FA60A8601006F7000200FB15E007EE8E4
S1138CC4790400801A094B04119440F8686816C8A3
S1138CD446500CCD1855400414D5110D0CDD46F83F
S1138CE4686816584708686814C868E840340FA0D1
S1138CF4010069710A90010069F001006F7000209E
S1138D040FB15E007EE80FA50A85400C686847082A
S1138D14685814C868D8400A0B76010069701F8626
S10F8D2445EA0B970B975E006034547017
S1138D305E0060560F857A040000000701006F7023
S1138D40002001006DF001006F70002001006DF044
S1138D505E0092D40B970B970D06474879260001C6
S1138D60460A790004B86BA000FFE89A792600024E
S1138D70460A7900044C6BA000FFE89A19667A0052
S1138D800000001C0AF00D6117F10A90F9FF6889D1
S1138D900B56792600084DE67A000000001C0AF005
S1138DA05A008E267A000000001C0AF07A010000A7
S1138DB0A39C5E005F6E0D00470C190069D07A001A
S1138DC00000A39C40607A060000001C0AF6696359
S1138DD01113111131113111379630 7FF793303FE71
S1138DE069D36950791003FE462869500B5069D046
S1138DF069607960800F69E00FE30B73400E0FC168
S1138E000FB05E00932669501B5069D069607348A8
S1138E1047EC69607960800F79403FE069E07A0050
S1098E200000001C0AF033
S1138E2601006F7100185E005FE05E0060345470ED
S10BA39C0000000000000000B6
S1138E365E0060560F8601006F70002001006DF022
S1138E4601006F70002001006DF05E0092D40B9755
S1138E560B970D05474079250001460A790004B8AA
S1138E666BA000FFE89A79250002460A7900044CB4
S1138E766BA000FFE89A19667A000000001C0AF04E
S1138E860D6117F10A90F9FF68890B5679260008D8
S1138E964DE65A008F547A050000001C0AF5695501
S1138EA61115111151115115796507FF793503FF8D
S1138EB60D5C4C0E7A000000A3A40FE15E005FE098
S1138EC640607A000000001C0AF00FE15E005FE0DC
S1138ED6792C00344C4C0FE50B950BF579090034CE
S1138EE619C90D9017F07901001001D053100D091F
S1138EF6400A0FD01BF5191169811B590D994EF2C2
S1138F067900003419C017F07901001001D053100D
S1138F167909FFFF1B584B04101940F869506690F6
S1078F2669D07A0190
S1138F2A0000001C0AF10FE20F905E005A820FE064

S1138F3A7A010000A3A45E005F8A0D00470C7A0041
S1138F4A0000001C0AF07D0070707A000000001C0B
S1138F5A0AF001006F7100185E005FE05E00603482
S1058F6A54703E
S10BA3A40000000000000000AE
S1138F6C01006DF20D1A4F14792A00084D04FA0012
S1138F7C4008680A100A1B5A4EFA688A01006D727F
S1058F8C54701C
S1138F8E01006DF20D1A4F14792A00084D04FA00F0
S1138F9E4008680A110A1B5A4EFA688A01006D725C
S1058FAE5470FA
S1138FB001006DF119990D11471A4A067909080044
S1138FC017917919040F1B59101144FA10311031FC
S1138FD010311031010069811A9101006F81000481
S1098FE001006D715470E5
S1138FE60F8501F064155860009A792407FF47142A
S1138FF60D44586000820FA501F06435586000786F
S1139006580000800FA501F064355860007640686B
S11390160FA501F06435466C0DCC465C0F8501F057
S1139026641546544O5E0F8501F06415473C1031C4
S1139036123079280O104C0C1B5C10311230792841
S113904600104DF4580000C40FA501F06435471A0B
S1139056103312327920A00104C0C1B54103312327F
S1139066792A00104DF4580000A81AA21AB3687D95
S1139076100D131A58000228790107FF1AA21AB312
S1139086580001FA790107FF1AA27A0300000008C3
S113909668FA580001E801006DF601006DF501005C
S11390A66DF401006DF301006DF201006DF1010035
S11390B66DF07A3700000018691D692565D569F5D5
S11390C6691C111C111C111C111C796C07FF6924E6
S10990D611141114111422
S11390DC1114796407FF01006D100100691179689F
S11390EC000F01006F23000401006922796A000F4D
S11390FC792C07FF5870FEE2792407FF5870FF0A9A
S113910C0DCC5870FF1A0D445870FF36094C793C3E
S113911C03FF792C07FF58C0FF58792CFFCB58D08D
S113912CFF426FFC000279480010794A00100100DD
S113913C6FF000040F9601006FF2000801006FF34B
S113914C000C0D1552356FF500160D34529452B1B7
S113915C0A946511990009D44406791C00019900FD
S113916C6FF400140DC50D1D18990D0452340AC566
S113917C99000DE452B40AC599000D6452240AC532
S113918C99006FF500120DD50D1D18990D845234ED
S113919C0AC50D0452B40AC599000DE452240AC53C
S11391AC99000D6452A40AC599006FF500100DD5F2
S11391BC0D1D18990DB352830AB50D0452240AC51B
S10991CC99000DE452A41A
S11391D20AC59900528209D24404791A0001091A74
S11391E26F74000852040AC26F7400085284094A59
S11391F20D5B6F73001001006F7400126F7D001618
S113920219556F710002113213331334133503DA044
S1139212796001004712113213331334133503B51A2
S1139222792107FF5870FE5401F064454702700B21
S11392320D114E2E113213334402700B0D114A04D9
S11392420B5140F0732B47180CB8E80B47127A13F3
S1139252000000084O0A0B72792A00804D020B5168
S113926240207328471C0CB8E80B47167A130000F7

S1139272000844020B72792A01004D06113213339E
S11392820B5111321333113213331321333796AFF
S1139292000F10111011101110111011641A101A6878AE
S11392A21008131A7A170000001801006D700100EC
S11392B2698201006F83000401006D7101006D7208
S10792C201006D73C4
S11192C601006D7401006D7501006D7654702A
S11392D401006DF57A01000000080AF16818E87FBF
S11392E4A87F460A0B716818E8F0A8F04704190030
S11392F4402A6818F01050006898460A0B711AD572
S1139304400E6818470679000000140100B710B7575
S11393147A25000000064DEA7900000201006D750C
S1059324547080
S11393265E0060560F840F931AD51B730FB6402841
S11393360FC30AE30FB26838E880175017700F831C
S11393460FA06809100968890FD547080FC00AE0FE
S11393567D0070000FB51B760FE64CD40FD547067C
S1139366790100014002191117F10F905E00603474
S105937654702E
S11393785E0060567A37000000387A050000000462
S11393880AF50F840F967A000000001001006DF0B3
S1139398191101006F7000545E0095EA0B970FE3F3
S11393A80B930BF37A160000000701006FF60034E5
S11393B8790600030FC00B900BF001006FF0002833
S11393C87A140000000701006FF4002C5A0094F48B
S11393D86838460C01006F7000346809587000FA49
S11393E87A000000001001006DF019110FD05E0023
S11393F895EA0B9701006F70002801006FF00030A9
S113940801006F74002C790E0003407C01006F701B
S113941800301A916809FA0810311A0A4EFA1A80AC
S1139428684801F064101A916839FA0810311A0A69
S11394384EFA01006F720034010069F01A8068283F
S113944801F06401010069705E007F0E01006FF096
S113945800247A000000000401006DF07A01000086
S109946800280AF10DE2E9
S113946E096217F210320AD20FA05E0095A20B9773
S113947E1B5E01006F7000301BF001006FF00030B7
S113948E1BF40DEE4C807926000346247A0000006F
S113949E000A01006DF00D6417F40FC110310AD1EB
S11394AE01006F7000540AC00AC05E0084B84022E7
S11394BE7A000000000A01006DF00D6417F40FC16D
S11394CE10310AD101006F7000540AC00AC05E0049
S11394DE95A20B971B561BF301006F7000341BF004
S11394EE01006FF000340D6658C0FEDE7A170000DF
S10B94FE00385E006034547075
S11395065E0060567A370000000C0F860F940D60DC
S11395167910003F69C00FF10FE05E00878A7A0079
S1139526000000801006DF0191101006F7000289A
S11395365E0095EA0B971A800F8219550FF64010B5
S11395460B760FA00B700F82694079300000869C053
S1139556686847ECFB804004110B0B55686816B826
S113956647F66DF57A03000000080FA01A830FB1C2
S11395760FE05E0087120B876940195069C001002E
S11395866DF30FE101006F7000285E0084B80B973E
S10F95967A170000000C5E006034547073
S11395A25E0060560F860F9401006F7500180AD68D
S11395B21B760AD41B740FC319CC40226868175058

S11395C268391751091009C00D0468E817F47900C6
S11395D2010001D053040D4C1B751B761B730FD571
S10B95E246DA5E0060345470A8
S11395EA5E0060561B870F8469F101006F73001ACE
S11395FA0FC51AE6400C0FD00B756E790001688906
S113960A0B761FB645F00FC00B875E0060345470AB
S9030000FD

H8/300H LINKAGE EDITOR (Evaluation software) Ver.1.0

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'000012CB	H'0000101C	main	main
		H'000012CC - H'0000178B	H'000004C0	EV_Simulator	EV_Simulator
		H'0000178C - H'00001AF9	H'0000036E	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'00004965   H'00000564
                                 Timer                       Timer
                H'00004966   -   H'00004ABB   H'00000156
                                 Panel                       Panel
                H'00004ABC   -   H'00004B8D   H'000000D2
                                 sci                         sci
                H'00004B8E   -   H'00004DCB   H'0000023E
                                 lcd                         lcd
                H'00004DCC   -   H'000058BD   H'00000AF2
                                 usb                         usb
                H'000058BE   -   H'000058F7   H'0000003A
                                 rand                        rand
                H'000058F8   -   H'00005955   H'0000005E
                                 sprintf                     sprintf
                H'00005956   -   H'0000597F   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'00005980   -   H'0000599B   H'0000001C
                                 strcpy                      strcpy
                H'0000599C   -   H'000059B7   H'0000001C
                                 strlen                      strlen
                H'000059B8   -   H'000059E7   H'00000030
                                 vsprintf                    vsprintf
                H'000059E8   -   H'00005CC1   H'000002DA
                                 addd3                       addd3
                H'00005CC2   -   H'00005EF3   H'00000232
                                 divd3                       divd3
                H'00005EF4   -   H'00005F6D   H'0000007A
                                 dtol3                       dtol3
                H'00005F6E   -   H'00005F79   H'0000000C
                                 eqd3                        eqd3
                H'00005F7A   -   H'00005F89   H'00000010
                                 ged3                        ged3
                H'00005F8A   -   H'00005F99   H'00000010
                                 ltd3                        ltd3
                H'00005F9A   -   H'00005FDF   H'00000046
```

```
                    ltod3                   ltod3
H'00005FE0  -  H'00005FFD  H'0000001E
                    mv83                    mv83
H'00005FFE  -  H'00006025  H'00000028
                    mvn3                    mvn3
H'00006026  -  H'00006033  H'0000000E
                    ned3                    ned3
H'00006034  -  H'00006055  H'00000022
                    spregld3                spregld3
H'00006056  -  H'0000607D  H'00000028
                    spregsv3                spregsv3
H'0000607E  -  H'00007E2B  H'00001DAE
                    _fmtout                 _fmtout
H'00007E2C  -  H'00007EE7  H'000000BC
                    cmpd3                   cmpd3
H'00007EE8  -  H'00007F0D  H'00000026
                    divl3                   divl3
H'00007F0E  -  H'00007F2D  H'00000020
                    mull3                   mull3
H'00007F2E  -  H'00008313  H'000003E6
                    _dti                    _dti
H'00008314  -  H'000084B7  H'000001A4
                    _its                    _its
H'000084B8  -  H'00008511  H'0000005A
                    memcpy                  memcpy
H'00008512  -  H'0000854D  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
                    UNIT NAME                MODULE NAME


ATTRIBUTE :  CODE  NOSHR

P                   H'0000854E  -  H'00008575  H'00000028
                    _allzero                _allzero
H'00008576  -  H'0000866D  H'000000F8
                    _calcnpw                _calcnpw
H'0000866E  -  H'00008711  H'000000A4
                    _log10                  _log10
H'00008712  -  H'00008789  H'00000078
                    _lsfts                  _lsfts
H'0000878A  -  H'000087B7  H'0000002E
                    _pow5                   _pow5
H'000087B8  -  H'00008831  H'0000007A
                    _rsfts                  _rsfts
H'00008832  -  H'000088DD  H'000000AC
                    _sub                    _sub
H'000088DE  -  H'00008981  H'000000A4
                    _unpack                 _unpack
```

H'00008982 - H'000089BF H'0000003E
memcmp memcmp
H'000089C0 - H'00008A47 H'00000088
_mult64 _mult64
H'00008A48 - H'00008BA9 H'00000162
_power _power
H'00008BAA - H'00008C93 H'000000EA
_rnd _rnd
H'00008C94 - H'00008D2F H'0000009C
_setsbit _setsbit
H'00008D30 - H'00008E35 H'00000106
frexp frexp
H'00008E36 - H'00008F6B H'00000136
modf modf
H'00008F6C - H'00008F8D H'00000022
dslc3 dslc3
H'00008F8E - H'00008FAF H'00000022
dsruc3 dsruc3
H'00008FB0 - H'00008FE5 H'00000036
itod3 itod3
H'00008FE6 - H'000092D3 H'000002EE
muld3 muld3
H'000092D4 - H'00009325 H'00000052
_duchek _duchek
H'00009326 - H'00009377 H'00000052
_lsft _lsft
H'00009378 - H'00009505 H'0000018E
_mult _mult
H'00009506 - H'000095A1 H'0000009C
_pow10 _pow10

*** LINKAGE EDITOR LINK MAP LIST ***

SECTION NAME START - END LENGTH
UNIT NAME MODULE NAME

ATTRIBUTE : CODE NOSHR

P H'000095A2 - H'000095E9 H'00000048
_add _add
H'000095EA - H'00009619 H'00000030
memset memset

* TOTAL ADDRESS * H'00000200 - H'00009619 H'0000941A

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'00009C8D   H'0000028E
                                main                    main
                    H'00009C8E   -   H'00009CBC   H'0000002F
                                EV_Simulator            EV_Simulator
                    H'00009CBE   -   H'00009CC8   H'0000000B
                                EV_Puls                 EV_Puls
                    H'00009CCA   -   H'00009CF1   H'00000028
                                EV_Controller           EV_Controller
                    H'00009CF2   -   H'00009D57   H'00000066
                                EV_Input                EV_Input
                    H'00009D58   -   H'00009E17   H'000000C0
                                EV_Display              EV_Display
                    H'00009E18   -   H'00009E63   H'0000004C
                                EV_OpenClose            EV_OpenClose
                    H'00009E64   -   H'00009EA6   H'00000043
                                EV_UpDown               EV_UpDown
                    H'00009EA8   -   H'00009F19   H'00000072
                                EV_File                 EV_File
                    H'00009F1A   -   H'00009F21   H'00000008
                                EV_Time                 EV_Time
                    H'00009F22   -   H'00009F59   H'00000038
                                Timer                   Timer
                    H'00009F5A   -   H'00009F64   H'0000000B
                                Panel                   Panel
                    H'00009F66   -   H'0000A01F   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'0000A020   -   H'0000A027   H'00000008
                                _fmtout                 _fmtout
                    H'0000A028   -   H'0000A127   H'00000100
                                _ctype                  _ctype
                    H'0000A128   -   H'0000A1AF   H'00000088
                                _its                    _its
                    H'0000A1B0   -   H'0000A1B7   H'00000008
                                _log10                  _log10
                    H'0000A1B8   -   H'0000A297   H'000000E0
                                _pow5                   _pow5
```

H'0000A298 - H'0000A39B H'00000104
_power _power
H'0000A39C - H'0000A3A3 H'00000008
frexp frexp
H'0000A3A4 - H'0000A3AB H'00000008
modf modf

* TOTAL ADDRESS * H'00009A00 - H'0000A3AB H'000009AC

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'00FFE21F H'0000020E
main main
H'00FFE220 - H'00FFE233 H'00000014
EV_File EV_File
H'00FFE234 - H'00FFE475 H'00000242
Timer Timer
H'00FFE476 - H'00FFE4F5 H'00000080
Panel Panel
H'00FFE4F6 - H'00FFE545 H'00000050
sci sci
H'00FFE546 - H'00FFE585 H'00000040
lcd lcd

*** LINKAGE EDITOR LINK MAP LIST ***

SECTION NAME START - END LENGTH
UNIT NAME MODULE NAME

ATTRIBUTE : DATA NOSHR

B H'00FFE586 - H'00FFE859 H'000002D4
usb usb
H'00FFE85A - H'00FFE895 H'0000003C
_fmtout _fmtout
H'00FFE896 - H'00FFE899 H'00000004
_rnext _rnext
H'00FFE89A - H'00FFE89B H'00000002

_errno _errno

* TOTAL ADDRESS * H'00FFE010 - H'00FFE89B H'0000088C

H8/300H LINKAGE EDITOR (Evaluation software) Ver.1.0 PAGE : 1

*** LINKAGE EDITOR EXTERNALLY DEFINED SYMBOLS LIST ***

SYMBOL NAME	ADDR	TYPE
$ADDD$3	H'00005AB2	DAT
$CMPD$3	H'00007E2C	DAT
$DIVD$3	H'00005D5C	DAT
$DIVL$3	H'00007EE8	DAT
$DIVUL$3	H'00008512	DAT
$DSLC$3	H'00008F6C	DAT
$DSRUC$3	H'00008F8E	DAT
$DTOL$3	H'00005EF4	DAT
EQD3	H'00005F6E	DAT
GED3	H'00005F7A	DAT
$ITOD$3	H'00008FB0	DAT
LTD3	H'00005F8A	DAT
$LTOD$3	H'00005F9A	DAT
$MULD$3	H'0000909C	DAT
$MULL$3	H'00007F0E	DAT
$MV8$3	H'00005FE0	DAT
MVN3	H'00005FFE	DAT
NED3	H'00006026	DAT
$SUBD$3	H'00005A82	DAT
sp_regld3	H'00006034	DAT
sp_regsv3	H'00006056	DAT
_Checkfmove	H'000043C4	ENT
_Clear	H'00004966	ENT
_ClearLCD	H'00004CAC	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'00001136	ENT
_DisableInterrupt	H'000002AC	DAT
_Disp	H'00002E04	ENT
_DispInput	H'00002DB4	ENT
_DispUSBPort	H'00004EDC	ENT
_Door	H'00002E06	ENT
_Down	H'00003E6E	ENT
_DownMotor	H'00003942	ENT
_EV_AddressDataSet	H'000018E4	ENT
_EV_AddressSet	H'00001800	ENT
_EV_Controller	H'00001AFA	ENT
_EV_DataSet	H'00001872	ENT
_EV_EnableSet	H'000017E6	ENT
_EV_File	H'00003F0A	ENT
_EV_Input	H'000024E2	ENT
_EV_Puls	H'0000199E	ENT

_EV_Set	H'0000178C	ENT
_EV_Simulator	H'000012CC	ENT
_EV_Time	H'000042AA	ENT
_EnableInterrupt	H'000002A8	DAT
_GetChar	H'00002476	ENT
_GetCurrentTime	H'0000430C	ENT

*** LINKAGE EDITOR EXTERNALLY DEFINED SYMBOLS LIST ***

SYMBOL NAME	ADDR	TYPE
_GetPermit	H'000043A6	ENT
_GetSCI	H'00004AEC	ENT
_GetSW	H'0000127A	ENT
_H8init	H'000012A0	ENT
_Init	H'00000EA2	ENT
_InitITU	H'000048EA	ENT
_InitLCD	H'00004C3C	ENT
_InitSCI	H'00004ABC	ENT
_InitUSB	H'00004DE2	ENT
_InterruptITU0	H'00004920	ENT
_LCDOut4	H'00004BE4	ENT
_Limit_Read	H'00004266	ENT
_LocateLCD	H'00004CEC	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'000013B6	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'0000498E	ENT
_PrintLCD	H'00004D10	ENT
_PrintSCI	H'00004B0C	ENT
_PutLCD	H'00004CC0	ENT
_PutSCI	H'00004ADC	ENT
_Read	H'00003F9E	ENT

_ReadString	H'00003FF8	ENT
_Repaint	H'000004DC	ENT
_Run	H'00000548	ENT
_ScanSCI	H'00004AFC	ENT
_SetCurrentTime	H'000042EE	ENT
_SetLED	H'0000122C	ENT
_SetPermit	H'00004372	ENT
_SleepMSec	H'00004414	ENT
_Start	H'000046D6	ENT

*** LINKAGE EDITOR EXTERNALLY DEFINED SYMBOLS LIST ***

SYMBOL NAME	ADDR	TYPE
_Stop	H'00004704	ENT
_Thread_Start	H'0000479C	ENT
_Thread_Toggle	H'000047DA	ENT
_Thread_checkAllDelete	H'00004722	ENT
_Thread_checkStayAnother	H'0000473C	ENT
_Thread_getThread	H'00004762	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'000095A2	ENT
__allzero	H'0000854E	ENT
__calcnpw	H'00008576	ENT
__ctype	H'0000A028	DAT
__dti	H'00007F2E	ENT
__duchek	H'000092D4	ENT
__errno	H'00FFE89A	DAT
__fmtout	H'0000607E	ENT
__its	H'00008314	ENT
__log10	H'0000866E	ENT
__lsft	H'00009326	ENT
__lsfts	H'00008712	ENT
__mult	H'00009378	ENT
__mult64	H'000089C0	ENT
__pow10	H'00009506	ENT
__pow5	H'0000878A	ENT
__power	H'00008A48	ENT
__rnd	H'00008BAA	ENT
__rnext	H'00FFE896	DAT
__rsfts	H'000087B8	ENT
__setsbit	H'00008C94	ENT
__sub	H'00008832	ENT
__unpack	H'000088DE	ENT
_cntrl	H'00FFE08C	DAT
_countUpNextRun	H'000044EA	ENT

delete	H'000046A2	ENT
_frexp	H'00008D30	ENT
_getClock	H'00004402	ENT
_get_inbufflen	H'00005896	ENT
_get_outbufflen	H'000058AA	ENT
_i_cnt	H'00FFE016	DAT
_in	H'00FFE04E	DAT
_initWOVI	H'0000494C	ENT
_init_usbbuff	H'000056FA	ENT
_j_cnt	H'00FFE018	DAT
_main	H'000002B0	ENT
_memcmp	H'00008982	ENT

*** LINKAGE EDITOR EXTERNALLY DEFINED SYMBOLS LIST ***

SYMBOL NAME	ADDR	TYPE
_memcpy	H'000084B8	ENT
_memset	H'000095EA	ENT
_modf	H'00008E36	ENT
_new_EV_Status	H'00003EB6	ENT
_new_Thread	H'0000450A	ENT
_nextRun	H'000044A6	ENT
_puls	H'00FFE184	DAT
_rand	H'000058BE	ENT
_read_buff	H'00005838	ENT
_read_outbuff	H'000057DA	ENT
_s	H'00FFE04A	DAT
_simu	H'00FFE198	DAT
_sprintf	H'000058F8	ENT
_status	H'00FFE220	DAT
_strcmp	H'00005956	ENT
_strcpy	H'00005980	ENT
_strlen	H'0000599C	ENT
_th	H'00FFE01A	DAT
_th1	H'00FFE022	DAT
_th101	H'00FFE278	DAT
_th102	H'00FFE296	DAT
_th111	H'00FFE2B4	DAT
_th112	H'00FFE2D2	DAT
_th113	H'00FFE2F0	DAT
_th114	H'00FFE30E	DAT
_th119	H'00FFE32C	DAT
_th120	H'00FFE34A	DAT
_th121	H'00FFE368	DAT
_th122	H'00FFE386	DAT
_th123	H'00FFE3A4	DAT
_th130	H'00FFE3C2	DAT
_th131	H'00FFE3E0	DAT
_th141	H'00FFE3FE	DAT
_th142	H'00FFE41C	DAT
_th143	H'00FFE43A	DAT
_th144	H'00FFE458	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
_usb_int	H'00004F40	ENT
_vsprintf	H'000059B8	ENT
_wovi	H'00004948	ENT
_woviClock	H'00FFE234	DAT
_woviInit	H'000048B8	ENT
_woviRun	H'0000481E	ENT
_woviThreadFirst	H'00FFE23C	DAT
_woviThreadLast	H'00FFE25A	DAT
_write_buff	H'00005774	ENT

*** LINKAGE EDITOR EXTERNALLY DEFINED SYMBOLS LIST ***

SYMBOL NAME	ADDR	TYPE
_write_inbuff	H'00005718	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:
警告 W8057 Timer.c 398: パラメータ 'threadPerSec' は一度も使用されない(関数 wovi )
        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 146: 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 293: '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(0FFE000), 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