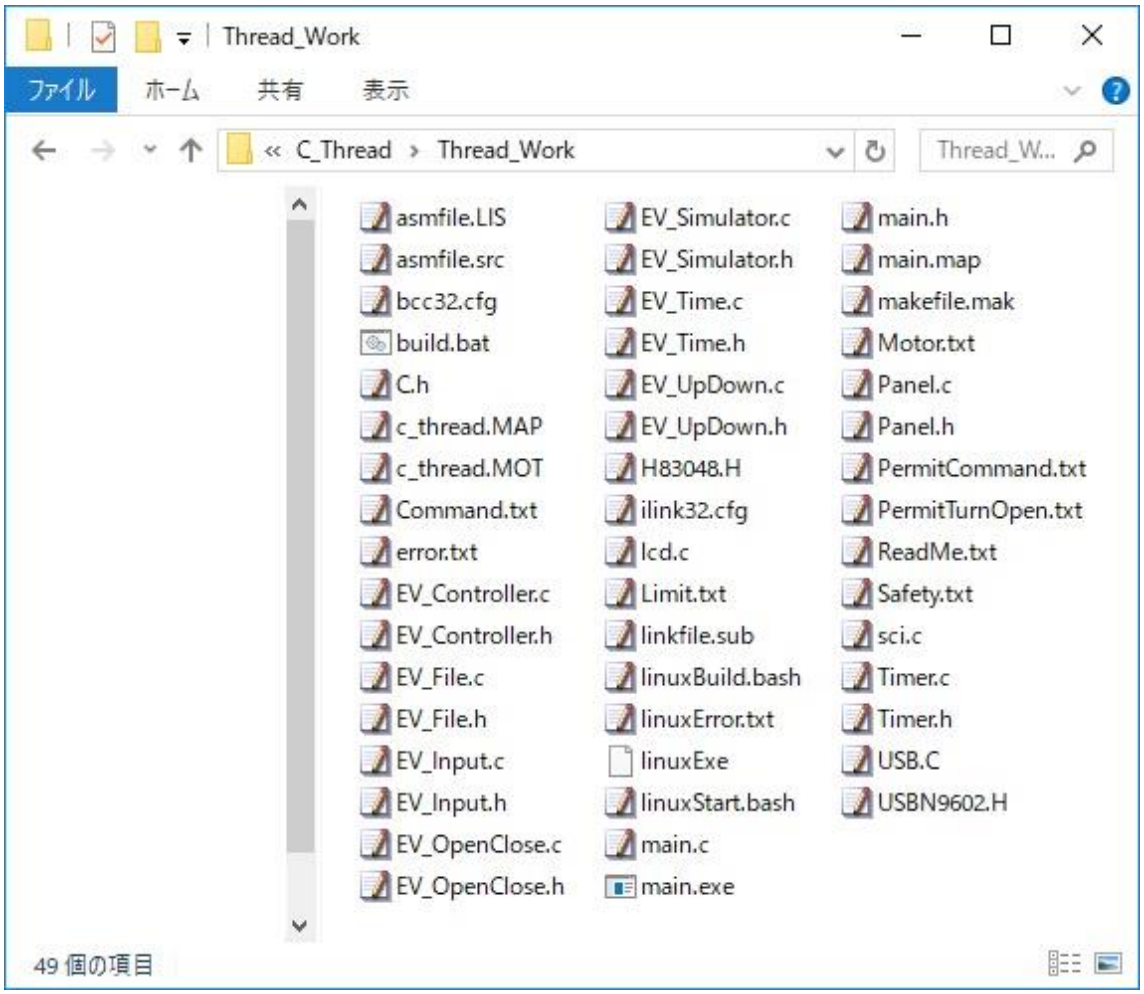
発明の巻

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 3 月 11 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。

==========

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

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

もしも、 AKI-H8 3052F USB開発セット に ACアダプター が

入っていなかったら(私は未確認です)、 ご自宅の最寄りの

電子パーツ専門店の人 に 電流・電圧 を聞いて、 ACアダプター も、

購入してください

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

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 1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 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、 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 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 80 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であることを意味します

yynnnynn は籠が下階停止状態で扉が閉低速区域であることを意味します

yynnnnnn は籠が下階停止状態で扉が中間高速区域であることを意味します

yynnnnyn は籠が下階停止状態で扉が開低速区域であることを意味します

yynnnnyy は籠が下階停止状態で扉が開停止状態であることを意味します

nynnyynn は籠が下階低速区域で扉が閉停止状態であることを意味します

nnnnyynn は籠が中間高速区域で扉が閉停止状態であることを意味します

nnynyynn は籠が上階低速区域で扉が閉停止状態であることを意味します

nnyyyynn は籠が上階停止状態で扉が閉停止状態であることを意味します

nnyynynn は籠が上階停止状態で扉が閉低速区域であることを意味します

nnyynnnn は籠が上階停止状態で扉が中間高速区域であることを意味します

nnyynnyn は籠が上階停止状態で扉が開低速区域であることを意味します

nnyynnyy は籠が上階停止状態で扉が開停止状態であることを意味します

==========

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;
#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;
    }
#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_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

#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

/*==========================================================================
 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(ClsPnl, "¥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(ClsPnl, "¥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(ClsPnl, "¥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(ClsPnl, "¥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_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_UpDown_h
#define EV_UpDown_h
#include "EV_UpDown.h"
#endif

#ifndef EV_OpenClose_h
```

```
#define EV_OpenClose_h
#include "EV_OpenClose.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]);

/*==========================================================================
 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;
}

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])
{
    int    i;
#ifdef USE_BCC
    /* 画面クリア */
    CLEAR;
#endif
    if      ((((ch == 'u') || (ch == 'j')) && (str[3] == 'y') && (str[2] == 'y'))
        ||  ((ch == 'k') && (str[3] == 'n') && (str[2] == 'y')))
    {
        for(i = 0; i < 4; i++)
        {
                PrintF(Monitor, "¥n    00000000    ");
        }
        for(i = 4; i < 12; i++)
        {
                PrintF(Monitor, "¥n                ");
        }
    }
```

```
    else if((((ch == 'U') ¦¦ (ch == 'j')) && (str[3] == 'n') && (str[2] == 'y'))
        ¦¦  (((ch == 'd') ¦¦ (ch == 'k')) && (str[3] == 'n') && (str[2] == 'n') && (str[1] == 'n') && (str[0] ==
'n'))
        ¦¦  ((ch == 's') && (str[3] == 'n') && (str[2] == '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 == 's') && (str[3] == 'n') && (str[2] == 'n') && (str[1] == 'n') && (str[0] == 'n'))
    {
        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 == 'u') ¦¦ (ch == 'j')) && (str[3] == 'n') && (str[2] == 'n') && (str[1] == 'n') && (str[0] ==
'n'))
        ¦¦  (((ch == 'D') ¦¦ (ch == 'k')) && (str[1] == 'y') && (str[0] == 'n'))
        ¦¦  ((ch == 's') && (str[1] == 'y') && (str[0] == 'n')))
    {
        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 == 'j') && (str[1] == 'y') && (str[0] == 'n'))
        ¦¦  (((ch == 'd') ¦¦ (ch == 'k')) && (str[1] == 'y') && (str[0] == 'y')))
    {
        for(i = 0; i < 8; i++)
        {
                PrintF(Monitor, "¥n              ");
        }
```

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

```
        ¦¦ (((ch == 'C') ¦¦ (ch == 't')) && (str[5] == 'y') && (str[4] == 'n'))
        ¦¦ ((ch == 's') && (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 == 's') && (str[5] == 'y') && (str[4] == 'y'))
        ¦¦ ((ch == 'h') && (str[5] == 'y') && (str[4] == 'n'))
        ¦¦ (((ch == 'c') ¦¦ (ch == 't')) && (str[5] == 'y') && (str[4] == 'y')))
    {
        for(i = 0; i < 8; i++)
        {
            PrintF(Monitor, "¥n              ");
        }
        for(i = 8; i < 12; i++)
        {
            PrintF(Monitor, "¥n    00000000    ");
        }
    }
}
/* 入力指示 */
DispInput();
```

```
        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_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_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;
#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;

/*==========================================================================
 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_checkStayAnother() == 1)
        {
            break;
        }
    }
#endif
    Clear();
#ifdef USE_THREAD
    /* LEDTEST */
    delete_(th30);
#endif
    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 80 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);

            delete_(This);
        }
    }
    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, 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, 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_Input.obj
EV_Controller.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_Controller.h EV_Input.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_OpenClose.h EV_UpDown.h EV_File.h EV_Time.h
Timer.h Panel.h C.h
	$(CC) $(CFLAGS) -c EV_Simulator.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_OpenClose.h EV_UpDown.h EV_File.h EV_Time.h Timer.h Panel.h
C.h
	$(CC) $(CFLAGS) -c EV_Input.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_Controller, EV_Input, 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_Input.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% EV_Controller.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_Controller.c EV_Input.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

yynnyynn

q

C

N

S

出力ファイル

```
 Start         Length     Name                   Class
0001:00401000 00000FED0H _TEXT                  CODE
0002:00411000 000003754H _DATA                  DATA
0003:00414754 000000B50H _BSS                   BSS
0004:00000000 0000000A4H _TLS                   TLS
```

```
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***     03/11/18 15:11:13
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 ***     03/11/18 15:11:13
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 ***     03/11/18 15:11:13
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 ***     03/11/18 15:11:13
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 ***     03/11/18 15:11:13
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 *** 03/11/18 15:11:13

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 ***    03/11/18 15:11:13
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
S113024001006DF501006DF65E00527401006D76DC
S113025001006D7501006D7401006D7301006D7215
S113026001006D7101006D70567001006DF00100A9
S11302706DF101006DF201006DF301006DF40100F9
S11302806DF501006DF65E004C5401006D760100C2
S11302906D7501006D7401006D7301006D720100D5
S11302A06D7101006D705670063F547004C0547038
S11302B05E00638A7A370000000A790B0001193364
S11302C00FF47A0600FFE1C819550B5579257FFF16
S11302D04DF85C000F6A5E004DF05E004F700D3804
S11302E00D305C000EE60D380DB05C000EDE0D38EF
S11302F0790000025C000ED40D38790000035C0025
S11303000ECA5E0051167FF472507FF570505E0086
S113031002A80D3228F117506DF07A0000009A0000
S113032001006DF05E004E400B970B877A000000D2
S11303309A0F01006DF05E0050440B9718886EC849
S113034000036EC800026EC8000168C87A0000FF8F
S1130350E0465E0041FA5E004C807A00000007D060
S11303605E0047587900001E5E00484E0F857900F5
S1130370001F5E00484E01006FF000040FD05E00C6
S11303804A0A01006F7000045E004A0A01006B20F4
S113039000009C8A01006DF001006B2000009C8628
S11303A001006DF05E004C7C0B970B975E004A706A
S11303B07920000146D65E004C9A0FD05E0049D6E4
S11303C07A0100009A1B0D305E004CC27A000000D7
S11303D007D05E0047585E004C9A19550D505C00DB
S11303E00E380D0D0D5117F10AC16819D90117D136
S11303F0660147540DB80D505C000DD00D5009B087
S11304006DF07A0000009A2C01006DF00FE05E00A1
S11304105C2C0B970B8701006DF67A0000009A3174
S113042001006DF05E004E400B970B970FE05E00EE
S11304305CD009B00D010FE05E005AA801006DF613
S11304405E0050440B9740080D380D505C000D7C46
S11304500DD017F50FC10AD168980B557925000403

S108046058D0FF785E97
S1130465004E300D0047165E004E2017D00D055E79
S1130475004FF468ED0DB10FE05E005AA85E005B16
S1130485CA0D004738790100400FE05E005B6C0D33
S1130495057A0100009A3501006DF15E0050440BA9
S11304A59701006DF65E0050440B9701006DF65EF3
S11304B5004E400B970D510FE05E005AA80D2879A9
S11304C50000035C000D000D20795000010D021999
S11304D5550B55792527104DF85A0003DA54705EEC
S11304E500638A7A0500FFE01219665E004C9A19CB
S11304F5EE400E7A0100009A370D605E004CC20B88
S11305055E69501D0E4DEC7A0100009A390D605E4F
S1130515004CC27A0100009A3D0D605E004CC21981
S1130525EE400E7A0100009A370D605E004CC20B57
S11305355E6F5000021D0E4DEA7A0100009A3F0DD1
S1130545605E004CC25E00636854705E00638A7A25
S1130555370000000A7A03000000017A040000074F
S1130565D019550F8618886EF800036EF800026ED1
S1130575F8000168F869607920000146365C00FFE0
S11305855E7A0000FFE01269010B5169815E005B31
S1130595F217F07902000901D053207918000A797E
S11305A5000064528017F00F810FE05E0047EA5A9E
S11305B5000E5C69607920000246365C00FF207AF4
S11305C50000FFE01469010B5169815E005BF217BE
S11305D5F07902000901D053207918000A79000047
S11305E564528017F00F810FE05E0047EA5A000E50
S11305F55C69607920000B586000B601006F6000EC
S1060605127A2043
S10A06080000000146205E23
S113060F004C9A7A0100009A430D505E004CC27A57
S113061F010000076C0FE05E00482E5A000E5C01CC
S113062F006F6000127A200000000246265E004C25
S113063F9A7A0100009A540D505E004CC27A010061
S113064F009A5B0D505E004CC20FE05E004A385AB1
S113065F000E5C01006F6000127A20000000034659
S113066F1E5E004C9A7A0100009A660D505E004C94
S113067FC27A01000005DC0FE05E00482E40240122
S113068F006F6000127A200000000446165E004CD3
S106069F9A7A0140
S11306A200009A770D505E004CC20FE05E004A389C
S11306B25A000E5C69607920000C586000A80100A2
S11306C26F6000127A200000000146205E004C9AFF
S11306D27A0100009A880D505E004CC27A01000034
S11306E20D480FE05E00482E5A000E5C01006F6059
S11306F200127A200000000246205E004C9A7A0122
S113070200009A990D505E004CC27A0100000D4818
S11307120FE05E00482E5A000E5C01006F6000126B
S11307227A2000000003461E5E004C9A7A01000004
S10807329AAA0D505EC0
S1130737004CC27A0100000D480FE05E00482E40CE
S11307471C5E004C9A7A0100009ABB0D505E004C68
S1130757C20FE05E004A380FE05E0049D65A000E2A
S11307675C69607920000D586000A801006F600084
S1130777127A200000000146205E004C9A7A01009D
S1130787009AC30D505E004CC27A01000013EC0FB0
S1130797E05E00482E5A000E5C01006F6000127A7B
S11307A7200000000246205E004C9A7A0100009A5E

S11307B7D40D505E004CC27A01000013EC0FE05ECB
S10707C700482E5A5B
S11307CB000E5C01006F6000127A200000000346EC
S11307DB1E5E004C9A7A0100009AE50D505E004CA8
S11307EBC27A01000013EC0FE05E00482E401C5E42
S11307FB004C9A7A0100009AF60D505E004CC20F22
S113080BE05E004A380FE05E0049D65A000E5C6981
S113081B607920000E586000E401006F6000127ACB
S113082B200000000146205E004C9A7A0100009ADA
S113083BFE0D505E004CC27A01000017700FE05E94
S113084B00482E5A000E5C01006F6000127A2000E4
S10E085B00000246405E004C9A7A0148
S113086600009B0F0D505E004CC27A01000017700A
S11308760FE05E00482E7A0100009B160D505E00C5
S11308864CC27900000B5E004AD00F867A01000045
S113089605DC5E00482E5A000E5C01006F600012F4
S11308A67A2000000003461E5E004C9A7A0100007F
S11308B69B210D505E004CC27A01000017700FE0B9
S11308C65E00482E403801006F6000127A20000057
S11308D60004462A7900000B5E004A960F824706FB
S11308E60FA05E0049D67A0100009B320D505E00D0
S11208F64CC20FE05E004A380FE05E0049D65A4D
S1130905000E5C69607920001346440FB10FE05E69
S11309150047EA19DD0DD05C0008FA0DD117F10F78
S1130925F20A92682ADA0117D2660247160DD079C0
S1130935100014 5E004B0E7A01000003E80FE05E21
S11309450047EA0B5D792D00044DCA5A000E5C6918
S1130955607920001446187A0100009B390D505E1A
S1130965004CC20FC10FE05E00482E5A000E5C69B1
S1130975607920001546187A0100009B3B0D505EF7
S1130985004CC20FC10FE05E00482E5A000E5C6991
S1130995607920001646187A0100009B3D0D505ED4
S10F09A5004CC20FC10FE05E00482E5A48
S11309B1000E5C69607920001746187A0100009BDC
S11309C13F0D505E004CC20FC10FE05E00482E5A2E
S11309D1000E5C69607920001E465E01006F6000B5
S11309E112462401006F6000120B7001006FE000DA
S11309F11228D6E80E38D67A01000003E80FE05E2C
S1130A010047EA5A000E5C01006F6000127A200071
S1130A11000001586004 4401006F6000121B700163
S1130A21006FE000127FD670007A01000003E80F27
S1130A31E05E0047EA5A000E5C69607920001F58A6
S1130A416003B801006F600012460C1A910FE05E5B
S1130A5100482E5A000E5C01006F6000127A2000DC
S1130A6100000146247A0100009A3D0D505E004CBE
S1130A71C27A0100009B410D505E004CC20FC10FB1
S1090A81E05E00482E5A5E
S1130A87000E5C01006F6000127A2000000002462E
S1130A973019DD0DD00B505E00484E0DD217F21002
S1130AA7321032010078A06BA000FFE01A0B5D79CA
S1130AB72D00024DDE0FB10FE05E00482E5A000DE8
S1130AC7FA01006F6000127A200000000346566B9C
S1130AD72000FFE0127920000D4D1A790000016B09
S1130AE7A000FFE01601006F6000120B7001006F9A
S1130AF7E0001240246B2000FFE0147920000D4D25
S1130B0718790000026BA000FFE01601006F600078
S1130B17120B7001006FE000120FB10FE05E004788

S1130B27EA5A000DFA01006F6000127A20000000F4
S1130B3704465A5E004C9A6B2000FFE016792000AA
S1130B4701460E7A0100009B520D505E004CC240D5
S1060B57186B20F5
S1130B5A00FFE01679200002460C7A0100009B632D
S1130B6A0D505E004CC201006B2000FFE01A5E00CC
S1130B7A49D601006B2000FFE01E5E0049D60FC173
S1130B8A0FE05E00482E5A000DFA01006F60001252
S1130B9A7A2000000005461C5E004C9A7A01000088
S1130BAA9A1B0D505E004CC20FC10FE05E00482E27
S1130BBA5A000DFA01006F6000127A200000000645
S10B0BCA461C5E004C9A7A01FF
S1130BD200009B740D505E004CC20FC10FE05E001B
S1130BE2482E5A000DFA01006F6000127A200000AD
S1130BF2000746365E004C9A19DD0DD07910000BC2
S1130C025E00484E0DD217F210321032010078A066
S1130C126BA000FFE0220B5D792D00044DDC0FB1C8
S1130C220FE05E00482E5A000DFA01006F600012B9
S1130C327A200000000846245E004A7079200002F0
S1130C42460E01006F6000120B7001006FE000128C
S1130C520FB10FE05E0047EA5A000DFA01006F6020
S1130C6200127A2000000009460C0FC10FE05E005B
S1130C72482E5A000DFA01006F6000127A2000001C
S1130C82000A461C5E004C9A7A0100009A1B0D5022
S1040C925E00
S1130C93004CC20FC10FE05E00482E5A000DFA014B
S1130CA3006F6000127A200000000B461C5E004CAC
S1130CB39A7A0100009B850D505E004CC20FC10F51
S1130CC3E05E00482E5A000DFA01006F6000127AAD
S1130CD3200000000C4634790000135E00484E01E7
S1130CE3006BA000FFE0325E004A0A790000145E45
S1130CF300484E01006BA000FFE0365E004A0A0F76
S1130D03B10FE05E00482E5A000DFA01006F600038
S1130D13127A200000000D46305E004A70792000ED
S10A0D2303461A01006B20D7
S1130D2A00FFE0325E0049D601006F6000120B70CB
S1130D3A01006FE000120FB10FE05E0047EA5A00AC
S1130D4A0DFA01006F6000127A200000000E460CB3
S1130D5A0FC10FE05E00482E5A000DFA01006F60C2
S1130D6A00127A200000000F461A5E004C9A7A019C
S1130D7A00009A1B0D505E004CC20FC10FE05E00CB
S1130D8A482E406C01006F6000127A2000000010A8
S1130D9A461A5E004C9A7A0100009B960D505E003B
S1130DAA4CC20FC10FE05E00482E404401006F6041
S1130DBA00127A200000001146367900002 95E00ED
S1130DCA484E01006BA000FFE03A7900002A5E005A
S1090DDA484E01006BA06E
S1130DE000FFE03E7900002B5E00484E01006BA03F
S1130DF000FFE0420FE05E0049D640606960792061
S1130E000029461A7A01000000640FE05E0047EAF9
S1130E100FE17A0000FFE04A5E0029A4403E6960CA
S1130E207920002A461A7A01000000640FE05E0070
S1130E3047EA0FE17A0000FFE0885E002030401CA3
S1130E4069607920002B46140FC10FE05E0047EA6A
S1130E500FE17A0000FFE1805E0013567A1700006D
S1130E60000A5E00636854705E00638A0F8679042B
S1130E7000097A0500FFE012696079200001462627

S1130E80190069D05E005BF217F001D05340791866
S1130E90000A7900001E528017F00F810FE05E00F8
S1060EA047EA5AC1
S1130EA30010D0696079200002462819006FD00032
S1130EB3025E005BF217F001D053407918000A7900
S1130EC300001E528017F00F810FE05E0047EA5ABD
S1130ED30010D0696C792C00034D36792C00084E31
S1130EE33069601B5017F010300A85190069D05E12
S1130EF3005BF217F001D053407918000A79000020
S1130F03C8528017F00F810FE05E0047EA5A0010C2
S1130F13D07A0400004CC2195569607920000B464E
S1130F231A7A0100009BA70D505D407A010000056A
S1130F33DC0FE05E00482E5A0010D0696079200070
S1130F430C461A7A0100009BAF0D505D407A0100F5
S1130F53000BB80FE05E00482E5A0010D069607989
S1130F6320000D46247A0100009A3D0D505D407A1E
S1040F730179
S1130F7400009BB90D505D407A01000011940FE00D
S1130F845E00482E5A0010D069607920000E461A7C
S1130F947A0100009BC10D505D407A010000177077
S1130FA40FE05E00482E5A0010D07A03000007D0E9
S1130FB4696079200001446245E004C9A7A0100008B
S1130FC49BCB0D505D407A0100009A3D0D505D406E
S1130FD40FB10FE05E00482E5A0010D069607920EB
S1130FE4001546245E004C9A7A0100009BDC0D50E8
S1130FF45D407A0100009A3D0D505D400FB10FE052
S10810045E00482E5AB6
S11310090010D0696079200001646245E004C9A7A54
S11310190100009BED0D505D407A0100009A3D0DE2
S1131029505D400FB10FE05E00482E5A0010D069A1
S113103960792000174 6225E004C9A7A0100009BD2
S1131049FE0D505D407A0100009A3D0D505D400F41
S1131059B10FE05E00482E406E69607920001E469C
S113106908F8FF38D438D6405E69607920002946EC
S1131079187A01000000640FE05E0047EA7A000075
S1131089FFE04A5E002878403E69607920002A46DD
S11310991A7A01000000640FE05E0047EA0FE17A63
S11310A90000FFE0885E001E90401C696079200003
S10B10B92B46140FB10FE05E9A
S11310C10047EA0FE17A0000FFE1805E00126C5EE7
S11310D100636854705E00638A7A0400004CC2198D
S11310E1660F8569507920000B46105E004C9A7A91
S11310F10100009C0F0D605D404044695079200060
S11311010C460C7A0100009C1A0D605D4040306969
S1131111507920000D460C7A0100009C240D605D7E
S113112140401C69507920000E46147A0100009A50
S11311313D0D605D407A0100009C2E0D605D4069AC
S1131141507920001 4461A5E004C9A7A0100009CE3
S11311513F0D605D407A0100009A3D0D605D4040A6
S112116164695079200015461A5E004C9A7A0192
S113117000009C500D605D407A0100009A3D0D60B7
S11311805D4040426950792000 16461A5E004C9A31
S11311907A0100009C610D605D407A0100009A3D78
S11311A00D605D4040206955792500 1746185E00A3
S11311B04C9A7A0100009C730D605D407A01000037
S11311C09A3D0D605D405E00636854706DF66DF589
S11311D00D067909000128D617500D950CE91A0957

S11311E04B04101540F866050D8846121A0E4B0481
S11311F0101940F80D9029D6148939D640141A0EC7
S11312004B04101940F8795900FF0D9029D616891F
S113121039D60D506D756D7654706DF60D0628D365
S1131220175079010011A0E4B04101140F8661093
S113123047041911400479010010D106D765470B3
S1131240F80638ECF8FF38C038C1188838D8F8FFEA
S113125038C8188838DBF8FF38C9F8DF38D0F8FF0A
S109126038CDF8F038D18F
S1091266F8FF38D45470B8
S1139A00435055204D4F444520253032580A000C11
S1139A1052656164792133303532004E4558542004
S1139A2020202020202020202020202000737725754F
S1139A300025730A000C0020003C313E000A003C64
S1139A40323E003C313E3173742020202020202000
S1139A50202020003C313E326E64003C313E537482
S1139A606F70202020003C313E3372642020202080
S1139A70202020202020003C313E53746F70202092
S1139A802020202020202020003C323E3173742020EF
S1139A902020202020202020003C323E326E6420F3
S1139AA02020202020202020003C323E337264DE
S1139AB020202020202020202020003C323E5374F0
S1139AC06F70003C333E3173742020202020200F
S1139AD0202020003C333E326E64202020202020B2
S1139AE020202020003C333E33726420202020209D
S1099AF0202020202000CD
S1139AF63C333E53746F70003C343E317374202004
S1139B062020202020202020003C343E326E64009A
S1139B163C313E53746172742020003C343E3372F0
S1139B2664202020202020202020003C343E5387
S1139B36746F00300031003200330054687265617F
S1139B4664205265616479204 74F2100474F414C99
S1139B56213C313E574F4E202020202000474F41C5
S1139B664C213C323E574F4E202020202000436F8D
S1139B76756E74557020202020202020202000544C
S1139B866F67676C652020202020202020202020007E
S1139B9648656C6C6F2045562020202020202020200D
S1139BA6003C313E496E6974003C323E496E69742D
S1139BB62020003C333E496E6974003C343E496EB6
S1139BC669742020003C303E496E697420202020B1
S1139BD62020202020003C313E496E69742020203D
S1099BE6202020202020B6
S1139BEC003C323E496E6974202020202020202026
S1139BFC20003C333E496E6974202020202020202015
S1139C0C2020003C313E44657374726F79003C3202
S1139C1C3E44657374726F003C333E4465737472D7
S1139C2C6F003C343E44657374726F79202020209E
S1139C3C2020003C303E44657374726F79202020E1
S1139C4C202020003C313E44657374726F792020D0
S1139C5C20202020003C323E44657374726F7920BF
S1139C6C202020202020003C333E44657374726F07
S1139C7C79202020202020202000000415312D0000006
S1059C8C0000D3
S113126C5E00638A0F860F957A00000000200AE067
S113127C0FD15E0045EE7A00000000200AE05E000C
S113128C463201006FE600027A00000000060AE015
S113129C01006FE000087A000000000C0AE0010076

S11312AC6FE0000E7A00000000120AE001006FE00C
S11312BC001C18886EE8001A7A050000424E7A006A
S11312CC000000380AE05D507A000000003C0AE0A0
S11312DC5D507A00000000400AE05D507A00000087
S11312EC00440AE05D5001006F60000201006DF0E4
S11312FC7A00000000400AE07A0100009C8E5E0038
S113130C42E20B9779200001473A790000096DF00E
S113131C01006F60001C01006DF07A0000000044B6
S113132C0AE07A0100009C995E00433C0B970B8703
S113133C79200001470E7A01000000120AE1686867
S113134C5C00042A5E00636854705E00638A0F8637
S109135C0F9501006F6014
S1131362000E01006DF07A000000003C0AE07A01F1
S113137200009CA45E0042E20B976E68000CA87109
S1131382460E5E004C9A0FD05E0049D65A001724CF
S113139201006F60000201006DF07A00000000405E
S11313A20AE07A0100009C8E5E0042E20B9779000C
S11313B200096DF001006F60001C01006DF07A00FE
S11013C2000000440AE07A0100009C995EDF
S11313CF00433C0B970B8701006F60000801006D12
S11313DFF07A00000000380AE07A0100009CB15E49
S11313EF0042E20B976E680006A873461CF84E6D19
S11313FFF07A000000003C0AE07A0100009CA45E32
S113140F0042580B875A0017186868A87546366E3E
S113141F680012A879587002D06E680013A8794635
S113142F0AF86E6EE800135A0016F86E680014A8D7
S113143F6E470E6E680015A86E4606F8796EE800C3
S113144F155A0016F86868A855462E6E680012A83C
S113145F79587002946E680013A8795870028A6ED7
S113146F680014A86E4608F8796EE8001440066EFB
S113147F680015A86E5A0016F86868A86A46466E83
S113148F680012A879460AF86E6EE800125A001621
S113149FF86E680013A879460AF86E6EE800135ABF
S11314AF0016F86E680014A86E4608F8796EE80007
S10814BF14400E6E68ED
S11314C40015A86E4606F8796EE800155A0016F85A
S11314D46868A86446366E680015A87958700214C3
S11314E46E680014A879460AF86E6EE800145A0070
S11314F416F86E680013A86E470E6E680012A86E85
S11315044606F8796EE800125A0016F86868A8448B
S1131514462E6E680015A879587001D86E680014B9
S1131524A879587001CE6E680013A86E4608F8793E
S11015346EE8001340066E680012A86E5AA0
S11315410016F86868A86B46466E680015A87946C8
S11315510AF86E6EE800155A0016F86E680014A8B2
S113156179460AF86E6EE800145A0016F86E6800A0
S113157113A86E4608F8796EE80013400E6E6800F2
S113158112A86E4606F8796EE800125A0016F8683A
S113159168A86F46366E680016A879587001586EB0
S11315A1680017A879460AF86E6EE800175A001604
S11315B1F86E680018A86E470E6E680019A86E468B
S11315C106F8796EE800195A0016F86868A84F46BC
S11315D12E6E680016A8795870011C6E680017A852
S11315E179587001126E680018A86E4608F8796E72
S11315F1E8001840066E680019A86E5A0016F868CC
S113160168A86846466E680016A879460AF86E6EA1
S1131611E800165A0016F86E680017A879460AF80A

S11316216E6EE800175A0016F86E680018A86E4629
S108163108F8796EE8E2
S11316360018400E6E680019A86E4606F8796EE823
S113164600195A0016F86868A86346346E680019CC
S1131656A8795870009C6E680018A879460AF86E37
S11316666EE800185A0016F86E680017A86E470E43
S11316766E680016A86E4606F8796EE80016407284
S11316866868A84346286E680019A87947646E6897
S11316960018A879475C6E680017A86E4608F879A3
S11316A66EE8001740066E680016A86E4044686828
S11316B6A874463E6E680019A8794608F86E6EE867
S11316C60019402E6E680018A8794608F86E6EE871
S11316D60018401E6E680017A86E4608F8796EE873
S11316E60017400E6E680016A86E4606F8796EE877
S11316F600167A00000000120AE001006DF07A007D
S1131706000000440AE07A0100009C995E0042AAA8
S11317160B977A01000000120AE1686855565E00CD
S1081726636854705ECE
S113172B00638A7A0500004CC2790600027A010035
S113173B009CBD0D605D507A0100009CEC0D605D5B
S113174B507A0100009D0D0D605D507A0100009DE4
S113175B390D605D507A0100009D670D605D507A15
S113176B0100009D730D605D505E00636854700152
S113177B006DF6790600030C80A0754704A06A463A
S113178B106E180003A87946086E180002A8794753
S113179B14A06B46426E180003A86E463A6E1800EF
S11317AB02A879463219EE7A0100009D7D0D605E29
S11317BB004CC20B5E792E00044DEC790E00047ABB
S11317CB0100009D8F0D605E004CC20B5E792E00F5
S11317DB0C4DEC5A001E86A0554704A06A46106EAA
S11317EB180003A86E46086E180002A879473AA0A2
S11317FB644704A06B461E6E180003A86E46166E54
S113180B180002A86E460E6E180001A86E460668F5
S109181B18A86E4714A09B
S113182173465A6E180003A86E46526E180002A83A
S111183179464A19EE7A0100009D8F0D605E24
S113183F004CC20B5E792E00024DEC790E00027A3A
S113184F0100009D7D0D605E004CC20B5E792E0082
S113185F064DEC790E00067A0100009D8F0D605E38
S113186F004CC20B5E792E000C4DEC5A001E86A065
S113187F7346686E180003A86E46606E180002A8C0
S113188F6E46586E180001A86E46506818A86E462B
S113189F4A19EE7A0100009D8F0D605E004CC20B5A
S11318AF5E792E00044DEC790E00047A0100009D41
S11318BF7D0D605E004CC20B5E792E00084DEC79F6
S11318CF0E00087A0100009D8F0D605E004CC20B65
S11318DF5E792E000C4DEC5A001E86A0754704A0AE
S11318EF6A461E6E180003A86E46166E180002A8ED
S11318FF6E460E6E180001A86E46066818A86E474E
S113190F28A0444704A06B460E6E180001A8794621
S113191F066818A86E4712A07346586E180001A8E0
S109192F7946506818A878
S11319356E464A19EE7A0100009D8F0D605E004CDC
S1121945C20B5E792E00064DEC790E00067A0177
S113195400009D7D0D605E004CC20B5E792E000A73
S11319644DEC790E000A7A0100009D8F0D605E0034
S11319744CC20B5E792E000C4DEC5A001E86A06AF5

S1131984460E6E180001A87946066818A86E471615
S1131994A0644704A06B46406E180001A87946383A
S11319A46818A879463219EE7A0100009D8F0D60FC
S11319B45E004CC20B5E792E00084DEC790E0008D4
S11319C47A0100009D7D0D605E004CC20B5E792E92
S11319D4000C4DEC5A001E866E180003A87958605B
S11319E4024E6E180002A87958600244A073461090
S11319F46E180007A87946086E180006A879472CC4
S1131A04A06F4704A06846106E180007A87946081B
S1131A146E180006A8794714A07446426E1800078E
S1131A24A86E463A6E180006A879463219EE7A0172
S1131A3400009DA10D605E004CC20B5E792E000474
S1091A444DEC790E0004D5
S1131A4A7A0100009D8F0D605E004CC20B5E792EF9
S1131A5A000C4DEC5A001E86A04F4704A06846109E
S1131A6A6E180007A86E46086E180006A879473C48
S1131A7AA0634704A07446206E180007A86E461890
S1131A8A6E180006A86E46106E180005A86E460862
S1131A9A6E180004A86E4714A07346426E18000716
S1131AAAA86E463A6E180006A879463219EE7A01EC
S1131ABA00009DB30D605E004CC20B5E792E0004DC
S1131ACA4DEC790E00047A0100009D8F0D605E00D3
S1131ADA4CC20B5E792E000C4DEC5A001E86A07385
S1131AEA46526E180007A86E464A6E180006A86E7C
S1131AFA46426E180005A86E463A6E180004A86E90
S1131B0A463219EE7A0100009DC50D605E004CC293
S1131B1A0B5E792E00044DEC790E00047A01000065
S1131B2A9D8F0D605E004CC20B5E792E000C4DEC4E
S1131B3A5A001C30A06F4704A06846206E1800079D
S1131B4AA86E46186E180006A86E46106E18000591
S1131B5AA86E46086E180004A86E472CA0434704D3
S1131B6AA07446106E180005A87946086E1800047A
S1131B7AA86E4714A07346406E180005A879463824
S1131B8A6E180004A86E463019EE7A0100009DD73C
S1131B9A0D605E004CC20B5E792E00044DEC790E8B
S1071BAA00047A01B5
S1131BAE00009D8F0D605E004CC20B5E792E000C03
S1131BBE4DEC406EA07346106E180005A8794608CA
S1131BCE6E180004A879472CA06846106E180005FD
S1131BDEA87946086E180004A86E4718A063470438
S1131BEEA074463E6E180005A87946366E1800049A
S1121BFEA879462E19EE7A0100009D7D0D605ED9
S1131C0D004CC20B5E792E00044DEC790E00047A64
S1131C1D0100009D8F0D605E004CC20B5E792E009E
S1131C2D0C4DEC5A001E866E180001A879586002FF
S1131C3D486818A87958600240A07346106E1800C2
S1131C4D07A87946086E180006A879472CA06F4798
S1131C5D04A06846106E180007A87946086E180090
S1131C6D06A8794714A07446426E180007A86E465D
S1131C7D3A6E180006A879463219EE7A0100009DD6
S1131C8D8F0D605E004CC20B5E792E00084DEC7912
S1131C9D0E00087A0100009DA10D605E004CC20B81
S1131CAD5E792E000C4DEC5A001E86A04F4704A002
S1131CBD6846106E180007A86E46086E180006A831
S1131CCD79473CA0634704A07446206E180007A80B
S1131CDD6E46186E180006A86E46106E180005A8FD
S1131CED6E46086E180004A86E4714A07346426E24

S1091CFD180007A86E4663
S1131D033A6E180006A879463219EE7A0100009D4F
S1131D138F0D605E004CC20B5E792E00084DEC798B
S1131D230E00087A0100009DB30D605E004CC20BE8
S1131D335E792E000C4DEC5A001E86A07346526E3C
S1131D43180007A86E464A6E180006A86E46426E30
S1131D53180005A86E463A6E180004A86E46321999
S10D1D63EE7A0100009D8F0D605E13
S1131D6D004CC20B5E792E00084DEC790E00087AFB
S1131D7D0100009DC50D605E004CC20B5E792E0007
S1131D8D0C4DEC5A001E86A06F4704A06846206ECA
S1131D9D180007A86E46186E180006A86E46106E3A
S1131DAD180005A86E46086E180004A86E472CA0EF
S1131DBD434704A07446106E180005A87946086EB3
S1131DCD180004A86E4714A07346406E180005A8AA
S1131DDD7946386E180004A86E463019EE7A010064
S1131DED009D8F0D605E004CC20B5E792E00084D79
S1131DFDEC790E00087A0100009DD70D605E004C52
S1131E0DC20B5E792E000C4DEC406EA07346106E26
S1131E1D180005A87946086E180004A879472CA068
S1131E2D6846106E180005A87946086E180004A8B8
S1131E3D6E4718A0634704A074463E6E180005A8AC
S1131E4D7946366E180004A879462E19EE7A0100EC
S1091E5D009D8F0D605E85
S1131E63004CC20B5E792E00084DEC790E00087A04
S1131E730100009D7D0D605E004CC20B5E792E0058
S1101E830C4DEC5C00F8A001006D7654706E
S1139C8E4D6F746F722E74787400004C696D697425
S1139C9E2E7478740000436F6D6D616E642E74784C
S1139CAE7400005361666574792E74787400000A2B
S1139CBE5550203D202775272C20444F574E203DCD
S1139CCE202764272C204F50454E203D20276F27F9
S1139CDE2C20434C4F5345203D20276327000A4534
S1139CEE4D455247454E4359203D202773272C207F
S1139CFE5245434F56455259203D20277227000A9D
S1139D0E31737420466C6F6F722043414C4C203D6F
S1139D1E202779272C20326E6420466C6F6F7220B9
S1139D2E43414C4C203D20275927000A31737420A0
S1139D3E466C6F6F7220434C4F5345203D2027686E
S1139D4E272C20326E6420466C6F6F7220434C4F6B
S1139D5E5345203D20274827000A51554954203D9D
S1139D6E20277127000A434F4D4D414E443E000AB2
S1099D7E202020203030FC
S1139D84303030303030302020202000A20202020A2
S1139D94202020202020202020202020000A3030D2
S1139DA4303020202020202020202030303030000A82
S1139DB42030303030202020202020303030302010C
S1139DC4000A2020303030302020202030303030042
S1139DD42020000A20202030303030202030303042
S1089DE43020202000E7
S1131E905E00638A0F860F9501006FE60024010040
S1131EA06F6000245E0039B27A00000000280AE067
S1131EB05E003A267A00000000300AE05E003C86AD
S1131EC07A00000000380AE05E003EE67A00000077
S1131ED0004C0AE001006FE000705E00314A7A00B6
S1131EE0000000740AE05E0031BE7A000000007C4E
S1131EF00AE05E00341E7A00000000840AE05E00FF

S1131F00367E7A00000000A20AE00FD15E0045EEA3
S1131F107A050000424E7A00000000E00AE05D50BE
S1131F207A00000000E40AE05D507A00000000E857
S1131F300AE05D507A00000000EC0AE05D507A0090
S1131F40000000F00AE05D507A00000000F40AE0AF
S1131F505D507A00000000BA0AE001006FE000BCA7
S1131F607A00000000C00AE001006FE000CA7A00B6
S1131F70000000CE0AE001006FE000D07A0000000C
S1071F8000D40AE09C
S1131F8401006FE000D67A00000000DA0AE00100E5
S1131F946FE000DC7A00000000F40AE0F9735E00ED
S1131FA445247A00000000380AE001006F61002430
S1131FB45E003F007A00000000840AE001006F61C4
S1131FC400705E0036A6790000096DF001006F60B1
S1131FD400CA01006DF07A00000000E40AE07A010F
S1131FE400009DEA5E00433C0B970B877A000000D8
S1131FF400E80AE0F94E5E00445079200001 4726C8
S11320047A00000000EC0AE0F9635E0043C0792023
S1132014000147127A00000000F00AE0F94E5E0066
S113202444E0792000015E00636854705E00638AB3
S1132034790D0001F463FC4E19550F860F930100CB
S11320446F6000BC01006DF07A00000000E00AE05C
S10520547A010C
S113205600009DF55E0042E20B977920000158705F
S1132066079E790000096DF001006F6000CA010048
S11320766DF07A00000000E40AE07A0100009DEAB0
S11320865E00433C0B970B877A00000000E80AE0EA
S113209601006F6100D05E0044047920000158708E
S11320A6075E7A00000000EC0AE001006F6100D6CB
S11320B65E00437479200001587007447A000000DB
S11320C600F00AE001006F6100DC5E0044947920B1
S11320D600015870072A6E6800CEA848587005108C
S11320E6A8594762A86358700458A8645870022216
S11320F6A868587005C4A86F58700398A871471844
S1132106A8725870067EA8734726A875475EA879F5
S1132116587001D85A00278A7A00000000F40AE0B2
S1132126F9735E0045240FB05E0049D65A00278A2C
S11321367A00000000A20AE00D515E0046B65A007E
S1092146278A5A00278AD4
S113214C01006F60002401006F00000C690079200E
S113215C0001461201006F60007001006F00000C5B
S113216C690158700 61801006F60002401006F00AC
S113217C0014690079200001464E01006F60007065
S113218C01006F000014690079200001 5860034AB4
S113219C7A00000000EC0AE00C495E0043C07A00B0
S11321AC000000A20AE00DD15E0046B67A000000E2
S11321BC00E80AE00CC95E0044505E004C9A7A01B8
S11321CC00009E015A00273801006F600070010067
S11321DC6F00000C69007920000146767A0000003C
S11321EC00A20AE00D515E0046B67A00000000A280
S11321FC0AE05E0046327A00000000F00AE00CC9E7
S113220C5E0044E07A00000000380AE07A370000F0
S113221C00140FF17A02000000145E0063327A009E
S113222C000000280AE07A37000000080FF17A0258
S113223C000000085E00633201006F6100BC010006
S113224C6F6000245E00416A7A170000001C5A007C
S113225C278A01006F60007001006F00000C690198

S113226C46667A00000000A20AE00D515E0046B6F5
S113227C7A00000000A20AE05E0046327A000000F9
S113228C00840AE07A370000001E0FF17A02000086
S113229C001E5E0063327A000000007C0AE07A378D
S11322AC000000080FF17A02000000085E006332A0
S10922BC01006F6100BC8C
S11322C201006F6000705E00396A7A170000002611
S11322D2401A7A00000000A20AE00D515E0046B6E1
S11322E27A00000000A20AE05E0046325A00278A02
S11322F201006F60002401006F000014690079205F
S1132302000146120100 6F60007001006F00000CB3
S1132312690158700 47201006F60002401006F00AC
S1132322000C690079200001465001006F600070C3
S113233201006F000014690079200001463 87A0019
S1132342000000EC0AE00C495E0043C07A00000082
S10A235200A20AE00DD15EB9
S11323590046B67A00000000E80AE00CC95E0044B2
S1132369505E004C9A7A0100009E015A0027385AA0
S11323790024E601006F60007001006F00000C6922
S1132389007920000146767A00000000A20AE00DD8
S1132399515E0046B67A00000000A20AE05E0046DC
S11323A9327A00000000F00AE00CC95E0044E07ACA
S11323B900000000380AE07A37000000140FF17AB0
S11323C902000000145E0063327A00000000300A44
S11323D9E07A37000000080FF17A02000000085E76
S11323E900633201006F6100BC01006F6000245E6D
S11323F90041B27A170000001C5A00278A01006FB6
S1132409600070 01006F00000C690146667A0000E4
S11324190000A20AE00D515E0046B67A00000000F2
S1132429A20AE05E0046327A00000000840AE07ADC
S1132439370000001E0FF17A020000001E5E0063E0
S1062449327A00E1
S113244C0000007C0AE07A37000000080FF17A02E2
S113245C000000085E00633201006F6100BC0100E4
S113246C6F6000705E00396A7A1700000026401A0C
S10E247C7A00000000A20AE00D515E90
S11324870046B67A00000000A20AE05E0046325A10
S113249700278A01006F60007001006F0000146954
S11324A7007920000146387A00000000EC0AE00CAE
S11324B7495E0043C07A00000000A20AE00DD15E26
S11324C70046B67A00000000E80AE00CC95E004443
S11324D7505E004C9A7A0100009E015A0027387A11
S11324E700000000A20AE00D515E0046B67A000024
S11324F70000A20AE05E0046327A00000000840A68
S1132507E07A370000001E0FF17A020000001E5E1A
S11325170063327A00000000740AE07A3700000093
S1132527080FF17A02000000085E00633201006FB2
S11325376100BC01006F6000705E0039225A0026FB
S1132547B67A00000000A20AE00D515E0046B67A93
S113255700000000A20AE05E00463201006F60003F
S11325677001006F00000C69007920000146387A7A
S108257700000000F06C
S113257C0AE00CC95E0044E07A00000000E80AE0BF
S106258C0CC95E16
S113258F0044507A00000000EC0AE00C495E00435F
S113259FC05E004C9A7A0100009E015A0027387AD8
S11325AF00000000840AE07A370000001E0FF17A62

S11325BF020000001E5E0063327A000000007C0AF6
S11325CFE07A37000000080FF17A02000000085E7E
S11325DF00633201006F6100BC01006F6000705E29
S11325EF00396A5A0026B601006F60002401006F9C
S11325FF000014690079200001586001 7E7A000001
S113260F0000A20AE00D515E0046B67A00000000FA
S113261FA20AE05E00463201006F60007001006F96
S113262F00000C69007920000146387A0000000091
S113263FE80AE00CC95E0044507A00000000F00A7B
S113264FE00CC95E0044E07A00000000EC0AE00CE5
S113265F495E0043C05E004C9A7A0100009E015A06
S113266F0027387A00000000840AE07A3700000060
S108267F1E0FF17A02B9
S11326840000001E5E0063327A000000007C0AE052
S11226947A37000000080FF17A02000000085E99
S11326A300633201006F6100BC01006F6000705E64
S11326B300396A7A17000000265A00278A01006F3F
S11326C360002401006F00000C69007920000158A9
S11326D36000B47A00000000A20AE00D515E0046D8
S11326E3B67A00000000A20AE05E00463201006FE2
S11326F360007001006F00000C690079200001463F
S11327033C7A00000000E80AE00CC95E0044507AFA
S113271300000000F00AE00CC95E0044E07A000008
S11327230000EC0AE00C495E0043C05E004C9A7A59
S11327330100009E010D505E004CC2404A7A000026
S11327430000840AE07A370000001E0FF17A0200CA
S113275300001E5E0063327A000000007C0AE07A08
S1132763370000000080FF17A02000000085E0063DF
S11327733201006F6100BC01006F6000705E0039BD
S11327836A7A17000000267A00000000A20AE05EBE
S108279300465079200F
S1132798000A4D6A01006F60007001006F000014A9
S11327A869007920000146567A00000000A20AE079
S11327B85E0046EA7920000146447A00000000A240
S11327C80AE00D515E0046B67A00000000A20AE056
S11327D85E0046327A00000000F00AE0F96F5E00FE
S11027E844E07A00000000EC0AE00CC95E3A
S11327F50043C07A00000000E80AE00C495E00448B
S10A2805505E00636854708C
S1139DEA4C696D69742E7478740000 5361666574E6
S1139DFA792E747874000048656C6C6F2045562080
S10B9E0A20202020202020006D
S113280C5E00638A1B970FF40C8D18886EC8000347
S113281C6EC800026EC8000168C819660D605E00C0
S113282C121A0D0E0D6117F10AC16819D90117D1CE
S113283C660147260D600C004620A800470EA80130
S113284C470EA802470EA803470E400EFD75400A1B
S113285CFD644006FD634002FD710B5679260004AE
S113286C4DBA0CD80B975E00636854705E00638A94
S113287C0F8618886EE800067A00000000120AE042
S113288C01006FE000147A00000000180AE0010058
S113289C6FE0002218886EE8002001006FE600024A
S11328AC7A00000000070AE001006FE000087A00DC
S11328BC0000000C0AE001006FE0000E7A05000036
S11328CC424E7A00000000260AE05D507A000000B8
S11328DC002A0AE05D507A000000002E0AE05D50E9
S11328EC7A00000000320AE05D507A000000003AE2

S10728FC0AE05D503E
S113290001006F60000201006DF07A0000000026F4
S11329100AE07A0100009E125E0042E20B977920E2
S11329200001477A7A000000002E0AE001006F617F
S113293000085E0043747920000147627A000000BA
S113294000320AE001006F61000E5E0044947920BA
S11329500001474A01006F60001401006DF07A0026
S11329600000003A0AE07A0100009E1E5E0042E287
S11329700B977920000147267900000096DF00100CB
S11329806F60002201006DF07A00000000360AE05B
S11329907A0100009E295E00433C0B970B875E0083
S11329A0636854705E00638AFC74F568FD4E0F869D
S11329B00F937A01000000060AE17A000000002A62
S11329C00AE05E0044047920000158700776 6E68BF
S11329D000065C00FE366EE8000601006F60000230
S11329E001006DF07A00000000260AE07A01000081
S10929F09E125E0042E2AC
S11329F60B977920000158700744 7A000000002ED7
S1132A060AE001006F6100085E004374792000014B
S1132A165870072A7A00000000320AE001006F614D
S1062A26000E5E3E
S1132A2900449479200001587007100 1006F600079
S1132A391401006DF07A000000003A0AE07A0100FF
S1132A49009E1E5E0042E20B977920000158700632
S1132A59EA790000096DF001006F60002201006D41
S1132A69F07A00000000360AE07A0100009E295E30
S1132A7900433C0B970B87790D00016E680006A88C
S1132A89715870008EA872474CA873586000AAF851
S1132A99736DF07A00000000260AE07A0100009EB7
S1132AA9345E0042580B877A000000003A0AE0F9C5
S1132AB9735E004524F84E6EE800067A00000000B4
S1132AC92A0AE0F94E5E0044507A000000002E0AFB
S1132AD9E0F94E40360C586DF07A00000000260AE2
S1132AE9E07A0100009E345E0042580B87F84E6E6F
S1132AF9E800067A000000002A0AE0F94E5E004465
S1132B09507A000000002E0AE0F9635E0043C05AC0
S1082B190031447A00C5
S1132B1E0000002A0AE06E6900065E0044507A0146
S1102B2E00009E3F0DD05E004CC20FB05E54
S1132B3B0049D65A0031446E680007A863586004F5
S1132B4BBCF46F6E680006A8485870041CA859584B
S1132B5B7002F2A86358700080A86458700194A89F
S1132B6B6858700376A86F4710A875587000C8A8EB
S1132B7B795870023A5A0031446E68001CA87947A1
S1132B8B0E7A00000000320AE00C495E0044E06854
S1132B9B68A868586003B0F85968E86DF07A0000CC
S1132BAB0000260AE07A0100009E345E0042580BB7
S1132BBB876E680012A873461C6868A85946166E80
S1132BCB68001FA86E460E7A000000003A0AE00C5C
S1132BDB595E0045245A002F526E68001CA8794792
S1132BEB0E7A00000000320AE00C495E0044E068F4
S1132BFB68A868463EF85968E86DF07A0000000053
S1132C0B260AE07A0100009E345E0042580B876E61
S1132C1B680012A873461C6868A85946166E6800AC
S1082C2B1CA86E460E1B
S1132C307A000000003A0AE00CC95E0045245A00FD
S1132C402F526E68001CA879470E7A0000000032EC

S1132C500AE00C495E0044E06868A8685860009A7E
S1132C60F85968E86DF07A00000000260AE07A015E
S1132C7000009E345E0042580B876E680012A873F2
S1132C8046266868A85946206E680018A879461831
S1132C906E68001CA86E46107A000000003A0AE035
S1062CA00CC95EFB
S1132CA300452440526E680012A873461E6868A844
S1132CB35946186E68001BA86E46107A0000000080
S1132CC33A0AE0F96A5E004524402C6E680012A8B4
S1132CD37346246868A859461E6E68001BA8794684
S1132CE3166E68001FA86E460E7A000000003A0AAB
S1132CF3E00C595E0045245A002F526E68001CA84D
S1132D0379470E7A00000000320AE00C495E004462
S1132D13E06868A8685860009AF85968E86DF07A23
S1132D2300000000260AE07A0100009E345E0042A0
S1132D33580B876E680012A87346266868A859461D
S1132D43206E68001BA87946186E68001CA86E469F
S1132D53107A000000003A0AE00CC95E00452440E3
S1132D63526E680012A873461E6868A85946186E07
S1132D73680018A86E46107A000000003A0AE0F9CA
S1132D836B5E004524402C6E680012A873462468CA
S1082D9368A859461E6B
S1132D986E680018A87946166E68001FA86E460E5E
S1132DA87A000000003A0AE00C595E0045245A00F4
S1132DB82F527A03000000180AE36838A8794712EB
S1132DC86E380004A879470A7A0100009E445A0025
S1132DD82F826868A868466CF85968E86DF07A002D
S1132DE8000000260AE07A0100009E345E00425883
S1132DF80B876E680012A873461E6868A8594618A0
S1132E086E680018A86E46107A000000003A0AE0BF
S1132E18F96B5E004524402C6E680012A8734624A3
S1132E286868A859461E6E680018A87946166E6821
S1132E38001FA86E460E7A000000003A0AE00C59FB
S1082E485E0045245A61
S1132E4D002F527A03000000180AE36E380003A81E
S1132E5D7947126E380004A879470A7A0100009E5B
S1132E6D5E5A002F826868A868466CF85968E86D49
S1132E7DF07A00000000260AE07A0100009E345E1D
S1132E8D0042580B876E680012A873461E6868A827
S1132E9D5946186E68001BA86E46107A0000000094
S1132EAD3A0AE0F96A5E004524402C6E680012A8C8
S1132EBD7346246868A859461E6E68001BA8794698
S1132ECD166E68001FA86E460E7A000000003A0ABF
S1132EDDE00C595E004524406C6E680018A87947D4
S1132EED0A7A0100009E445A002F826E68001CA8C6
S1132EFD79470E7A00000000320AE00C495E004467
S1132F0DE06868A868463EF85968E86DF07A0000F5
S1132F1D0000260AE07A0100009E345E0042580B41
S1132F2D876E680012A873461C6868A85946166E0A
S1092F3D68001CA86E46AB
S1132F430E7A000000003A0AE00CC95E0045247AB9
S1132F53000000002A0AE06E6900065E0044507A0E
S1132F63000000002E0AE00CD95E0043C05A003172
S10E2F73446E68001BA87947107A0128
S1132F7E00009E5E0DD05E004CC25A0031446E6856
S1132F8E001CA879470E7A00000000320AE00C49B3
S1132F9E5E0044E06868A868463EF85968E86DF03C

S1132FAE7A00000000260AE07A0100009E345E00DB
S1132FBE42580B876E680012A873461C6868A8599E
S1132FCE46166E68001CA86E460E7A000000003A84
S1132FDE0AE00CC95E0045247A000000002A0AE0CC
S1132FEE6E6900065E0044507A000000002E0AE06F
S1132FFE0CD95E0043C05A0031446E68000CA86FB2
S113300E586001327A03000000180AE36838A87981
S113301E5860008C6E380007A879587000826E686D
S113302E0006A864470CA86F4708A87947045A00FE
S113303E31446868A8684646F85968E86DF07A0026
S113304E000000260AE07A0100009E345E0042581A
S113305E0B876E680012A87346246868A859461E2B
S109306E6E680018A8794A
S113307446166E68001FA86E460E7A000000003ADA
S11330840AE00C595E0045247A000000002A0AE095
S11330946E6900065E0044507A00000000320AE0C4
S10A30A40CD95E0044E05A61
S11330AB0031447A03000000180AE36E380003A8CA
S11330BB79586000846E380007A879477C6E6800E6
S11330CB06A859470AA86F4706A8754702406A68BE
S11330DB68A8684646F85968E86DF07A0000000066
S11330EB260AE07A0100009E345E0042580B876E7D
S11330FB680012A87346246868A859461E6E6800B8
S113310B1BA87946166E68001FA86E460E7A000040
S113311B00003A0AE00C595E0045247A00000000D7
S113312B2A0AE06E6900065E0044507A0000000034
S112313B320AE00CD95E0044E05E006368547012
S1139E125361666574792E74787400004D6F746FA4
S1139E22722E74787400004C696D69742E747874A0
S1139E3200005361666574792E7478740051554934
S1139E4254000A41204261736B65742069736E2763
S1139E52742031737420466C6F6F72000A41204282
S1139E6261736B65742069736E277420326E64208C
S1099E72466C6F6F7200E5
S113314A01006DF60F86190069E06FE000026FE077
S113315A00046FE0000601006FE600087A00000031
S113316A00020AE001006FE0000C7A00000000048C
S113317A0AE001006FE000107A00000000060AE08E
S113318A01006FE0001419006FE000187A000000D4
S113319A00180AE001006FE0001A19006FE0001E30
S11331AA7A000000001E0AE001006FE0002001001F
S11331BA6D76547001006DF60F865E00424E7A00FA
S11331CA000000040AE05E00424E01006D7654706E
S11331DA5E00638A0F850F9601006F740018010061
S11331EA6F6000146900792000001465201006F6084
S11331FA001A6901462801006F66001A7900000166
S113320A69E07A00000000040AD0F9735E004524DD
S113321A7A0100009E78790000015E004CC268487A
S113322AA859586001E8F87268C86DF07A0100007D
S108323A9E7D0FD05E34
S113323F0042580B875A003418FB6801006F600077
S113324F10690079200001466E01006F60001A19A2
S113325F11698101006F6000206901462C01006F25
S113326F6600207900000169E07A00000000040A7B
S113327FD0F96F5E0045247A0100009E8979000022
S113328F015E004CC25A0034186848A859586001AF
S113329F787A00000000040AD00CB95E004524F8C8

S11332AF7268C86DF07A0100009E7D0FD05E0042F8
S11332BF580B875A00341801006F600008690146E4
S11332CF6E01006F60001A1911698101006F6000B0
S11332DF206901462C01006F660020790000016907
S10F32EFE07A00000000040AD0F94F5EF2
S11332FB0045247A0100009E8E790000015E004C8C
S113330BC25A0034186848A859586001007A000063
S113331B0000040AD00CB95E004524F87268C86D2E
S113332BF07A0100009E7D0FD05E0042580B875A46
S113333B00341801006F60000C6901466C01006FCB
S113334B60001A1911698101006F60002069014641
S113335B2C01006F6600207900000169E07A000000
S113336B0000040AD0F96F5E0045247A0100009E29
S113337B89790000015E004CC25A0034186848A8D2
S113338B59586000887A00000000040AD00CB95E1B
S113339B004524F87268C86DF07A0100009E7D0F1A
S11333ABD05E0042580B87406401006F60000C69CC
S11333BB0079200001465601006F60001A1911694C
S11333CB8101006F60002069014628 01006F6600D0
S11333DB207900000169E07A00000000040AD00C98
S10833EBB95E0045245A
S10533F07A015D
S11333F200009E9A790000015E004CC26848A859F9
S11334024614F87268C86DF07A0100009E7D0FD0F1
S11334125E0042580B875E006368547001006DF6CC
S11334220F865E00424E7A00000000040AE05E004E
S1133432424E01006D7654705E00638A0F850F96CB
S113344201006F74001801006F60000C690079209D
S1133452000146520 1006F60001A6901462801000B
S11334626F66001A7900000169E07A000000000427
S11334720AD0F9735E0045247A0100009E78790030
S113348200015E004CC26848A859586001E8F8720E
S113349268C86DF07A0100009E7D0FD05E0042582D
S11334A20B875A003678FB7401006F6000086900CD
S11334B279200001466E01006F60001A19116981BB
S11334C201006F6000206901462C01006F66002035
S11334D27900000169E07A00000000040AD0F96370
S10934E25E0045247A019F
S11034E800009EA5790000015E004CC25A51
S11334F50036786848A859586001787A00000000BA
S1133505040AD00CB95E004524F87268C86DF07AD8
S11335150100009E7D0FD05E0042580B875A00368E
S11335257801006F6000106901466E01006F60004D
S11335351A1911698101006F6000206901462C0188
S1133545006F6600207900000169E07A0000000041
S1133555040AD0F9435E0045247A0100009EAB7945
S11335650000015E004CC25A0036786848A85958D5
S11335756001007A00000000040AD00CB95E004522
S113358524F87268C86DF07A0100009E7D0FD05E45
S11335950042580B875A00367801006F60001469A2
S11335A501466C01006F60001A1911698101006FF2
S11335B56000206901462C01006F66002079000038
S11335C50169E07A00000000040AD0F9635E004552
S11335D5247A0100009EA5790000015E004CC25AC1
S11335E50036786848A859586000887A00000000BA
S11335F5040AD00CB95E004524F87268C86DF07AE8
S11336050100009E7D0FD05E0042580B8740640188

S1133615006F60001469007920000146560 1006FB0
S113362560001A1911698101006F60002069014664
S11336352801006F6600207900000169E07A000027
S11336450000040AD00CB95E0045247A0100009EEF
S1133655B8790000015E004CC26848A8594614F8C1
S11336657268C86DF07A0100009E7D0FD05E00423E
S1133675580B875E006368547001006DF60F867AF8
S113368500000000040AE001006FE000067A000074
S113369500000E0AE001006FE0001801006D76548A
S11336A5705E00638A0F860F957A000000000E0A8C
S11336B5E001006FE000187A000000000A0AE0014B
S11336C5006F6100185E0045AA6E680012A879466E
S10936D50679000001402C
S11336DB0219006FE0001C7A04000047086F6000BA
S11336EB1C6DF001006F51002001006F50000C5D49
S11336FB400B876E680013A87946067900000140DA
S113370B0219006FE0001C6DF001006F51002001E6
S113371B006F5000085D400B876E680014A8794654
S113372B0679000001400219006FE0001C6DF001E7
S113373B006F51002001006F5000105D400B876E2E
S113374B680015A879460679000001400219006F3D
S113375BE0001C6F66001C6DF601006F5100200129
S113376B006F5000145D400B875E00636854705EFE
S113377B00638A0F860F957A000000000E0AE001A2
S113378B006FE000187A000000000A0AE001006FE6
S113379B6100185E0045AA6E680012A87946067987
S11337AB000001400219006FE0001C7A040000477F
S11337BB086F60001C6DF001006F51002001006F5A
S10837CB50000C5D40FD
S11337D00B876E680013A879460679000001400242
S11337E019006FE0001C6DF001006F510020010013
S11337F06F5000085D400B876E680014A879460679
S1133800790000001400219006FE0001C6DF0010017
S11338106F51002001006F5000105D400B876E68F0
S11338200015A879460679000001400219006FE0EF
S1133830001C6F66001C6DF601006F510020010033
S11238406F5000145D400B875E00636854705E29
S113384F00638A0F860F957A000000000E0AE001CD
S113385F006FE000187A000000000A0AE001006F11
S113386F6100185E0045AA6E680012A879460679B2
S113387F000001400219006FE0001C7A04000047AA
S113388F086F60001C6DF001006F51002001006F85
S113389F50000C5D400B876E680013A879460679BC
S11338AF000001400219006FE0001C6DF001006F72
S11338BF51002001006F5000085D400B876E6800B8
S11338CF14A879460679000001400219006FE00041
S11338DF1C6DF001006F51002001006F5000105D4F
S11338EF400B876E680015A87946067900000140E2
S11338FF0219006FE0001C6F66001C6DF601006F6C
S113390F51002001006F5000145D400B875E006370
S113391F6854705E00638A0F860F9501006F600015
S113392F14690146307A00000000200AF00FE15CB1
S106393F00FE384C
S113394201006DF57A000000001C0AF00FE15C0033
S1133952F8860B977A00000000200AF00FE15C0062
S1133962FE165E00636854705E00638A0F860F95CD
S113397201006F60000C690146307A0000000020EC

S11339820AF00FE15C00FEC401006DF57A0000004D
S1133992001C0AF00FE15C00FA9E0B977A0000000C
S11339A200200AF00FE15C00FEA25E00636854701F
S1139E7853544F50005361666574792E7478740097
S1139E88004F50454E004F50454E20537065656452
S1139E9879004F50454E20537461727400434C4F00
S1139EA8534500434C4F5345205370656564790000F
S10F9EB8434C4F53452053746172740 0F7
S11339B201006DF60F86190069E06FE000026FE007
S11339C200046FE0000601006FE600087A000000C1
S11339D200020AE001006FE0000C7A00000000041C
S11339E20AE001006FE000107A00000000060AE01E
S11339F201006FE0001419006FE000187A00000064
S1133A0200180AE001006FE0001A19006FE0001EBF
S1133A127A000000001E0AE001006FE000200100AE
S1133A226D76547001006DF60F865E00424E7A0089
S1133A32000000040AE05E00424E01006D765470FD
S1133A425E00638A0F850F9601006F7400180100F0
S1133A526F6000146900792000014652010 06F6013
S1133A62001A6901462801006F66001A79000001F5
S1133A7269E07A00000000040AD0F9735E0045246D
S1133A827A0100009EC4790000015E004CC26848BE
S1133A92A859586001E8F87268C86DF07A0100000D
S1083AA29EC90FD05E78
S1133AA70042580B875A003C80FB6A01006F600095
S1133AB710690079200001466E01006F60001A1932
S1133AC711698101006F6000206901462C01006FB5
S1133AD76600207900000169E07A00000000040A0B
S1133AE7D0F9755E0045247A0100009ED579000060
S1133AF7015E004CC25A003C806848A859586001CF
S1133B07787A00000000040AD00CB95E004524F857
S1133B177268C86DF07A0100009EC90FD05E00423B
S1133B27580B875A003C8001006F60000869014603
S1133B376E01006F60001A1911698101006F60003F
S1133B47206901462C01006F660020790000016996
S10F3B57E07A00000000040AD0F9555E7B
S1133B630045247A0100009ED8790000015E004CD1
S1133B73C25A003C806848A859586001007A000083
S1133B830000040AD00CB95E004524F87268C86DBE
S1133B93F07A0100009EC90FD05E0042580B875A8A
S1133BA3003C8001006F60000C6901466C01006FEB
S1133BB360001A1911698101006F600020690146D1
S1133BC32C01006F6600207900000169E07A000090
S1133BD30000040AD0F9755E0045247A0100009EB3
S1133BE3D5790000015E004CC25A003C806848A8A6
S1133BF359586000887A00000000040AD00CB95EAB
S1133C03004524F87268C86DF07A0100009EC90F5D
S1133C13D05E0042580B87406401006F60000C695B
S1133C2300792000014656 01006F60001A191169DB
S1133C338101006F60002069014628 01006F66005F
S1133C43207900000169E07A00000000040AD00C27
S1083C53B95E004524E9
S1053C587A01EC
S1133C5A00009EE2790000015E004CC26848A85940
S1133C6A4614F87268C86DF07A0100009EC90FD035
S1133C7A5E0042580B875E006368547001006DF65C
S1133C8A0F865E00424E7A00000000040AE05E00DE

S1133C9A424E01006D7654705E00638A0F850F965B
S1133CAA01006F74001801006F60000C690079202D
S1133CBA0001465201006F60001A6901462801009B
S1133CCA6F66001A7900000169E07A0000000004B7
S1133CDA0AD0F9735E0045247A0100009EC4790074
S1133CEA00015E004CC26848A859586001E8F8729E
S1133CFA68C86DF07A0100009EC90FD05E00425871
S1133D0A0B875A003EE0FB6B01006F6000086900F5
S1133D1A79200001466E01006F60001A191169814A
S1133D2A01006F6000206901462C01006F660020C4
S1133D3A7900000169E07A00000000040AD0F964FE
S1093D4A5E0045247A012E
S1103D5000009EEB790000015E004CC25A9A
S1133D5D003EE06848A859586001787A00000000D9
S1133D6D040AD00CB95E004524F87268C86DF07A68
S1133D7D0100009EC90FD05E0042580B875A003ECA
S1133D8DE001006F6000106901466E01006F600075
S1133D9D1A1911698101006F6000206901462C0118
S1133DAD006F6600207900000169E07A00000000D1
S1133DBD040AD0F9445E0045247A0100009EF0798F
S1133DCD0000015E004CC25A003EE06848A85958F5
S1133DDD6001007A00000000040AD00CB95E0045B2
S1133DED24F87268C86DF07A0100009EC90FD05E89
S1133DFD0042580B875A003EE001006F60001469C2
S1133E0D01466C01006F60001A1911698101006F81
S1133E1D6000206901462C01006F660020790000C7
S1133E2D0169E07A00000000040AD0F9645E0045E0
S1133E3D247A0100009EEB790000015E004CC25A0A
S1133E4D003EE06848A859586000887A00000000D9
S1133E5D040AD00CB95E004524F87268C86DF07A77
S1133E6D0100009EC90FD05E0042580B87406401CC
S1133E7D006F60001469007920000146560100 6F40
S1133E8D60001A1911698101006F600020690146F4
S1133E9D2801006F6600207900000169E07A0000B7
S1133EAD0000040AD00CB95E0045247A0100009E7F
S1133EBDFC790000015E004CC26848A8594614F80D
S1133ECD7268C86DF07A0100009EC90FD05E004282
S1133EDD580B875E006368547001006DF60F867A88
S1133EED00000000040AE001006FE0000E01006D08
S1133EFD7654705E00638A0F860F957A000000007A
S1133F0D040AE001006FE0000E01006F61000E0F67
S1133F1DE05E0045AA6E680004A8794606790000A4
S1133F2D01400219006FE000127A04000047086F88
S1083F3D6000126DF0AD
S1133F4201006F51002001006F50000C5D400B8790
S1133F526E680005A879460679000001400219003F
S1133F626FE000126DF001006F51002001006F50ED
S1133F7200085D400B876E680006A8794606790043
S1133F820001400219006FE000126DF001006F5151
S1133F92002001006F5000105D400B876E68000720
S1133FA2A879460679000001400219006FE0001269
S1133FB26F6600126DF601006F51002001006F5011
S1133FC200145D400B875E00636854705E00638A71
S1133FD20F860F957A00000000040AE001006FE0EB
S1133FE2000E01006F61000E0FE05E0045AA6E68CD
S1133FF20004A879460679000001400219006FE027
S113400200127A04000047086F6000126DF001008D

S11340126F51002001006F50000C5D400B876E68EA
S11340220005A879460679000001400219006FE0F5
S107403200126DF018
S113403601006F51002001006F5000085D400B879F
S11340466E680006A879460679000001400219004 9
S11340566FE000126DF001006F51002001006F50F8
S113406600105D400B876E680007A8794606790045
S1134076000 1400219006FE000126F6600126DF630
S113408601006F51002001006F5000145D400B8743
S10A40965E00636854705ED5
S113409D00638A0F860F957A00000000040AE00181
S11340AD006FE0000E01006F61000E0FE05E004532
S11340BDAA6E680004A879460679000001400219 2A
S11340CD006FE000127A04000047086F6000126D64
S11340DDF001006F51002001006F50000C5D400B8B
S11340ED876E680005A879460679000001400219 1C
S11340FD006FE000126DF001006F51002001006FA1
S113410D5000085D400B876E680006A87946067956
S113411D000001400219006FE000126DF001006F05
S113412D51002001006F5000105D400B876E680039
S113413D07A879460679000001400219006FE000D7
S113414D126F6600126DF601006F51002001006FB2
S113415D5000145D400B875E00636854705E00630E
S113416D8A0F860F9501006F600014690146307A3E
S113417D00000000200AF00FE15C00FE4401006D19
S106418DF57A00BD
S1134190000000 1C0AF00FE15C00F8A60B977A0000
S11341A0000000200AF00FE15C00FE225E0063685D
S11341B054705E00638A0F860F9501006F60000CD8
S11341C0690146307A00000000200AF00FE15C002C
S11341D0FECA01006DF57A000000001C0AF00FE131
S11341E05C00FABE0B977A00000000200AF00FE192
S10D41F05C00FEA85E0063685470D3
S1139EC453544F50005361666574792E747874004B
S1139ED400555000555020537065656479005550 2
S1139EE42053746172740044 4F574E00444F574ECD
S1139EF4205370656564790044 4F574E2053746151
S1069F0472740071
S11341FA01006DF60F867A0000FFE208010069E00C
S113420A7A0600FFE208F87268E87A0000000006FE
S113421A0AE001006FE000027A0100009F08010032
S113422A6F6000025E005CB4F8736EE8000FF84E2C
S113423A6EE800106EE80011F84E6EE800120100F5
S113424A6D7654700F811A800100699054705E0074
S113425A638A0F946E7D00197A0600FFE2086849A3
S113426AA9434716A94D470CA9504714A9534628F1
S113427A68ED40246EED000F401E6EED00104018ED
S113428A6E480006A8434706A8544708400A6EED3D
S113429A001140046EED001219005E006368547049
S11342AA01006DF60F966869A94C46247A0600FF49
S11342BAE2087A00000000060AE001006FE000024B
S11342CA01006F71000801006F6000025E005CB4B8
S11342DA190001006D7654705E00638A0F94010021
S10942EA6F7500187A064F
S11342F000FFE2086849A9434716A94D470CA95096
S11343004716A953462E686E400A6E6E000F40048E
S11343106E6E001068DE401C6E480006A843470618

S1134320A854470A400E6E6E001168DE40066E6E9A
S1134330001268DE19005E006368547001006DF6B8
S11343400F966869A94C46247A0600FFE2087A00B2
S1134350000000060AE001006FE0000201006F6147
S1134360000201006F7000085E005CB419000100D8
S11343706D7654705E00638A0F850F9601006DF6AB
S11343807A0100009F120FD05C00FF560B970C00C0
S11343904624A8014706A8024716401A7A010000DE
S11343A09F25790000015E004CC279060001400898
S11343B079060002400219660D605E00636854705E
S10443C05E9B
S11343C100638A0F850C9E6DF67A0100009F120F20
S11343D1D05C00FE820B870C00461CA8004714A882
S11343E10146147A0100009F34790000015E004CFC
S11343F1C2400419664004790600010D605E006342
S1134401685470 5E00638A0F850F9601006DF67A1A
S1134411010000 9F430FD05C00FEC60B970C0046C2
S113442124A8014706A8024716401A7A0100009FF3
S11344312579000 0015E004CC2790600014008792C
S1134441060002400219660D605E00636854705EE7
S113445100638A0F850C9E6DF67A0100009F430F5E
S1134461D05C00FDF20B870C00461CA8004714A882
S1134471014 6147A0100009F34790000015E004C6B
S1134481C2400419664004790600010D605E0063B1
S1134491685470 5E00638A0F850F9601006DF67A8A
S10444A10116
S11344A200009F500FD05C00FE360B970C00462491
S11344B2A8014706A8024716401A7A0100009F2561
S11344C2790000015E004CC27906000140087906BA
S11344D20002400219660D605E00636854705E005C
S11344E2638A0F850C9E6DF67A0100009F500FD0F0
S11344F25C00FD620B870C00461CA8004714A80150
S113450246147A0100009F34790000015E004CC218
S1134512400419664004790600010D605E00636879
S113452254705E00638A0F850C9E6DF67A0100005B
S11345329F640FD05C00FD1E0B870C004616A8007B
S11345424712A801460E7A0100009F347900000148
S11345525E004CC25E00636854705E00638A1B8710
S11345620F8518886EF800017A06000000010AF62A
S113457201006DF67A0100009F640FD05C00FD60BC
S11345820B970C004618A800470EA8014710A80273
S1094592470C400A40083B
S110459840066E780001400218880B875E14
S11345A500636854705E00638A0F850F9679000077
S11345B5096DF001006DF67A0100009F6F0FD05C65
S11345C500FD740B970B870C004618A8014706A836
S11345D5024710400E7A0100009F25790000015E15
S10C45E5004CC25E0063685470CF
S1139F0879796E6E79796E6E00005065726D697439
S1139F18436F6D6D616E642E74787400000A526528
S1139F286164696E67204572726F72000A577269BD
S1139F3874696E67204572726F7200436F6D6D614D
S1139F486E642E74787400005065726D697454756C
S1139F58726E4F70656E2E74787400004D6F746F57
S1139F68722E74787400004C696D69742E74787459
S1059F780000E4
S11345EE5E00638A0F860F957A0000009F7A0FE1B3

S11345FE5E00631401006FE600080FE055267A0093
S113460E0000000C0AE001006FE0000E19006FE0DD
S113461E000C19006FE0001201006FE500145E003C
S113462E636854705E00638A0F8601006FE60008AC
S113463E0FE201006DF25E0047460B975E00636862
S113464E54705E00638A7A37000000100F860100F3
S113465E6FE600087A02000000080AF201006DF20C
S113466E5E0047460B9701006F6600080FA10FE22D
S113467E0FF05E005DB65E0062287A170000001030
S113468E5E00636854705E00638A0F850D16790EA3
S113469E03E852E617F60FE101006F5000145E00B7
S11346AE47EA5E006368547001006DF60F867A0068
S11346BE0000000C0AE001006FE0000E79210001FA
S11346CE460A790000016FE0000C400A0D11460600
S10946DE19006FE0000C5F
S11346E401006D76547001006DF60F867A000000A8
S11346F4000C0AE001006FE0000E6F60000C010083
S11347046D7654705E00638A0F860F9569606F71CE
S113471400181D10470A190069D06F70001869E06A
S11347245E00636854705E00638A0F850D1617F686
S11347340FE101006F5000145E0047EA5E006368F6
S10547445470AC
S10B9F7A0000000000000000DC
S11347467A0000FFE21C01006F7100045E0063142F
S113475654705E00638A7A37000000200F867A025F
S1134766000000180AF201006DF255D40B970FE111
S11347760FF05E0062CE0F817A0200009F927A00EC
S1134786000000100AF05E0060907A000000001836
S11347960AF07A01000000080AF15E006314401073
S11347A67A02000000080AF201006DF255920B9797
S11347B67A01000000180AF17A02000000100AF2DA
S11347C60FF05E005DE60F817A00000000080AF034
S11347D65E0062BE0D0046C87A17000000205E0028
S11347E6636854705E00638A1B971B970F860F9549
S11347F67A02000000020AE201006DF25C00FF404B
S11348060B970FD10FF05E0062CE0F817A02000084
S11348169F927A000000000A0AE05E0060900B9700
S11348260B975E00636854705E00638A0F860F956C
S10748360FE055B087
S113483A01006F6000120B7001006FE000125E004E
S113484A636854705E00638A0D057A0400FFE224EC
S113485A400601006F44001A01006F40001A01006C
S111486A6F01001A46EC79250001460A7A0612
S113487800FFE2605A00495C79250002460A7A067D
S113488800FFE27E5A00495C7925000B460A7A0646
S113489800FFE29C5A00495C7925000C460A7A0617
S11348A800FFE2BA5A00495C7925000D460A7A06E8
S11348B800FFE2D85A00495C7925000E460A7A06B9
S11348C800FFE2F65A00495C79250013460A7A0686
S10848D800FFE3145A88
S11348DD00495C7925001446087A0600FFE332404F
S11348ED6E7925001546087A0600FFE3504060797E
S11348FD25001646087A0600FFE36E40527925001F
S113490D1746087A0600FFE38C40447925001E46BE
S113491D087A0600FFE3AA40367925001F46087A78
S113492D0600FFE3C840287925002946087A0600CA
S113493DFFE3E6401A7925002A46087A0600FFE4CC

S113494D04400C7925002B46067A0600FFE4220F5E
S113495DE6461C7A0100009F8219005E004CC27A64
S113496D0100009F8419005E004CC21A804054015F
S113497D006FE4001601006F40001A01006FE000A4
S112498D1A01006F86001601006FC6001A7A0028
S113499C00009F9A7A01000000020AE15E00631492
S11349AC7A0000009FA27A010000000A0AE15E006F
S11349BC631469E51A8001006FE000120FE05E00DA
S11349CC0E680FE05E006368547001006DF60F868D
S11349DC5E0010D601006F60001601006F61001AB3
S11349EC01006F81001A01006F60001A01006F61F2
S11349FC001601006F81001601006D7654705E0085
S1134A0C638A0F867A0200FFE21C01006DF25C00E0
S1134A1CFD280B977A0000FFE21C7A0100000002CC
S1134A2C0AE15E0063145E006368547001006DF666
S1134A3C0F867A0000009FAA7A01000000020AE1A7
S1134A4C5E00631401006D76547001006B2000FF4F
S1134A5CE23E01006F01001A4604190054707900FC
S1134A6C0001547001006B2100FFE23E01006F1145
S1134A7C001A1988400C01006F11001A0D800B509D
S1094A8C0D080F9146F036
S10F4A920D8054706DF50D0501006B20C4
S1134A9E00FFE23E01006F01001A472001006B2167
S1134AAE00FFE23E69101D5046040F9040100100B6
S1134ABE6F11001A01006F10001A46E81A806D7507
S1134ACE54705E00638A0D0555BE0F86460E0D505B
S1134ADE5C00FD6C0F865C00FF22401C7A00000018
S1134AEE00020AE07A0100009FAA5E0062A20D0096
S1134AFE47060FE05C00FF040FE05E00636854702E
S1134B0E5E00638A0D0555800F86460E0D505C00C0
S1134B1EFD2E0F865C00FEE440247A0000000002A6
S1134B2E0AE07A0100009FAA5E0062A20D00470808
S1134B3E0FE05C00FEC640060FE05C00FE8A5E00DE
S1134B4E636854705E00638A1B971B977A0500FF98
S1134B5EE22401006F56001A407201006F65001ABD
S1134B6E7A00000000020AE07A0100009F9A5E00BC
S1134B7E635A0D0047547A00000000020AE07A01DE
S1084B8E00009FAA5E78
S1134B9300635A0D00473E7A01000000020AE17ADE
S1134BA3020000000A0AE20FF05E005DE60F817A5D
S1134BB30000FFE21C5E0062AE0D0047187A00009E
S1134BC3FFE21C7A01000000020AE15E0063140F96
S1134BD3E05E0005500FD601006F60001A46860B96
S1134BE3970B975E00636854701A8001006BA000F3
S1134BF3FFE23A7A0000FFE24201006BA000FFE20A
S1134C033E7A0000FFE22401006BA000FFE2581A82
S1134C138001006BA000FFE25C5470F80138601858
S1134C2388386138621888838633890F8FF389118E8
S1134C33883864F8883865F804386679009E576BBA
S1134C4380FF6819006B80FF6A19006B80FF6C5447
S1134C537001006DF27F67722079009E576B80FFAE
S10C4C63687A0100FFE21C7A02E9
S1134C6C00009FB20F905E005DE601006D72547000
S1134C7C5A004B527A0000009FA27A0100FFE21CFB
S1114C8C5E006314558C5E0002A85A004BECC8
S1139F820A0063616C6C6F63206661696C656400CF
S1139F92408F400000000000BFF0000000000000FE

S1139FA2000000000000000C000000000000000EC
S10B9FB23F50624DD2F1A9FCFE
S1134C9A7A0000009FBA01006DF07A0000FFE44039
S1134CAA5E005C2C0B977A0000FFE44001006DF074
S1134CBA5E0050440B9754705E00638A7A0500FFC6
S1134CCAE4800D040F960C4458600114AC00476845
S1134CDAAC014710AC0258700106AC03587000C20D
S1134CEA5A004DEA55AA7A0100009FBC0FE05E0004
S1134CFA5C8A0D00461E7A0000009FC201006DF017
S1134D0A7A0000009FBF01006DF00FE05E005C2C8B
S1134D1A0B970B9701006DF67A0000009FBF010005
S1134D2A6DF00FD05E005C2C0B970B9701006DF5AD
S10D4D3A7A0000009FBF40507A0189
S1134D4400009FBC0FE05E005C8A0D00461E7A00E3
S1134D5400009FC201006DF07A0000009FBF0100B4
S1134D646DF00FE05E005C2C0B970B9701006DF662
S1134D747A0000009FBF01006DF00FD05E005C2C31
S1134D840B970B9701006DF57A0000009FBF01009C
S1134D946DF05E004E400B970B9701006DF65E00BD
S1134DA450440B974040403E01006DF67A000000EA
S1134DB49FBF01006DF00FD05E005C2C0B970B9727
S1134DC401006DF57A0000009FBF01006DF05E00E5
S1134DD44E400B970B970FD05E005CD00B500D0128
S10A4DE40FD05E005AA85E28
S1084DEB006368547031
S10E9FBA0C000A00002573000A0C00D5
S1134DF0188838BA38B8F85038B919000B507920E8
S1134E0001184DF8F83038BA28BCF88038BC547013
S1134E100C8820BC737047FA38BBE07F30BC5470F9
S1134E2020BC736047FA29BDE0BF30BC0C985470B6
S1134E307EBC736047067900000154701900547 0FA
S1134E405E00638A7A0600FFE4C07A05000000046E
S1134E507A00000000180AF00AD07308470E7A009F
S1134E60000000180AF00AD00B70400A7A00000014
S1134E7000180AF00AD00F85189968E901006DF04F
S1134E8001006F71001C0FE05E005CEC0B9719557D
S1134E900D5017F00AE0680947220D5017F00AE099
S1134EA06808A80A4606F80D5C00FF640D5017F069
S1134EB00AE068085C00FF580B5540D45E00636845
S1054EC0547029
S1134EC21911400C19880B58792806824DF80B5199
S1134ED21D014DF0547001006DF50D090D8D28D69D
S1134EE217500D010D99470C0D1979690060794925
S1134EF2001040060D19796900600DD50D90148DCF
S1134F020CD8C88038D63DD639D67900000455B0BE
S1134F1201006D7554706DF56DF40D090D8528D67C
S1134F2217500D010D99470C0D19796900607949E4
S1134F32001040060D19796900600D541194119403
S1134F4211941194EC0F0D90148CED0F148D0CC869
S1134F52C88038D63CD60CD8C88038D63DD639D688
S1134F62790000045C00FF586D746D755470010084
S1134F726DF619EE7900000F5C00FF44196679089B
S1134F8200030DE05C00FF4E0B56792600034DEE45
S1134F92790800020DE05C00FF3C196679080028DD
S1134FA20D605C00FF70790800100D605C00FF6605
S1094FB27908000E0D60FA
S1134FB85C00FF5C790800060D605C00FF5279080D

S1134FC800010D605C00FF48790800020D605C0079
S1134FD8FF3E01006D7654707908000119005C00EA
S1134FE8FF2E7908000219005A004F186DF60C8E2F
S1134FF8AE0C460455E2401CAE0A460455DA40148A
S1135008AE0D460455D2400C17D60D687900000141
S11350185C00FEFC6D76547001006DF60D0E0D8676
S11350280D6079080040528009E0791000800D086E
S113503819005C00FEDA01006D7654705E00638A25
S1135048790C000119447A0600FFE5107A0500007F
S113505800047A00000000180AF00AD07308470E0B
S11350687A00000000180AF00AD00B70400A7A0090
S1135078000000180AF00AD00F85189968E90100A2
S11350886DF001006F71001C0FE05E005CEC0B9784
S113509819550D5017F00AE0680947560D5017F0D7
S10950A80AE06808A80AF3
S11350AE460A0DC80D405C00FF68403C0D5017F0DA
S11350BE0AE06808A80D460C790800020D405C0052
S11350CEFE4840240D5017F00AE06808A80C460667
S11350DE5C00FEFE40120D5017F00AE0680817D070
S11350EE0D080DC05C00FE220B5540A05E006368E8
S10550FE5470E9
S11351001911400C19880B58792806824DF80B5158
S11351101D014DF054705C0009145C0000B455048B
S11351205A0051FA01006DF618886AA800FFE613C9
S11351306AA800FFE61518886AA800FFE6146AA8A3
S113514000FFE5517908008079000045C0008083D
S1135150790800107900002O5C0007FC7906000F35
S1135160790800100D605C0007EE0D687900000BF4
S11351705C0007E40D687900000D5C0007DA79082C
S1135180005079000095C0007CE790800D6790049
S113519000075C0007C219660D605C000710790EFA
S11351A000010DE05C0006E60DE05C0007000D6801
S11351B07900002B5C0007A00D687900002F5C00CC
S11351C007960DE8790000275C00078C01006D76D7
S11351D0547079080005190O5C00077C79000064AD
S11351E05C00FF1C790800C419005C00076A790899
S10951F000037900000139
S11351F65A0059587908000279000055C000752E5
S1135206790800CC19005A0059585E00638A7A045B
S11352160000A0787A000000A04701006DF05E0050
S11352264E400B9719EE19660DE5790D001052D510
S11352360D6009505C00070C17506DF001006DF40A
S11352465E004E400B970B870B56792600104DE0F8
S11352567A000000A07E01006DF05E004E400B97C1
S11352660B5E792E00044DBE5E0063685470010028
S10752766DF67A064E
S113527A00FFE550790000065C0006C468E87C601C
S113528A734047367900000E5C0006B40C8E7348EF
S113529A47045C0002C8735E47085C0002C25A00F6
S11352AA5368730E47085C0002B85A005368731EAA
S11352BA47045C0002AE5A0053687C60736047225D
S11352CA7900000C5C0006780C8E730847085C00B2
S11352DA00925A005368731E587000825C0001C61C
S11352EA407C7C607320471E7900000A5C000650EC
S11352FA0C8E730847065C00020A4062731E475EFF
S113530A5C00023E40587C607310475279000008E3
S113531A5C00062C0C8E7368470A5C00FDFC5C007B

S113532AFECE403A734E471A790800C07900000945
S113533A5C00061A79080003790000055C00060E72
S113534A401C737E4718790800050790000095C00F5
S113535A05FC79080002790000055C0005F00100EC
S108536A6D7654705E36
S113536F00638A19DD790000265C0005CE0C8E736D
S113537F68587001047A0600FFE553790000086D41
S113538FF00FE179080026790000255C0005CC0BAE
S113539F870DD05C0004E80DD05C00050279050091
S11353AF200D505C0004C06868E8605860009C6E74
S11353BF680001473CA801471EA8034772A8054789
S11353CF3EA8064726A8084716A809475CA80A4718
S11353DF26A80B4760406C5C0001865A0054766A1E
S11353EF2800FFE55217500D08400E5C00022440C1
S11353FF765C00035440700DD8790000214024796 6
S113540F0800400D505C0005406E6E0002CE806AAE
S113541FAE00FFE5516A2800FFE55117500D0879DB
S113542F0000045C000522403E5C00038E40385CA4
S113543F00018840326E680002472C0D505C000457
S113544F0E40240D505C000406401C6868E860A8F9
S108545F2046080D507A
S11354645C0003F6400C686EEE60AE400D505C00C9
S113547403E87A0000FFE6147D0070000DD05C00A1
S1135484045440226A2800FFE615470818886AA8CE
S113549400FFE6150DD05C0004107908000179 00C3
S10A54A400275C0004AE5E6B
S11354AB00636854705E00638A7900002E5C00040D
S11354BB8E0C8EE8C017504626790000406DF07AAB
S11354CB0500FFE5930FD17908002E7900002D5CC1
S11354DB00048C0B870D060D010FD05C0005627960
S11354EB0000015C00039C790800017900002F5C2C
S11354FB00045A5E0063685470790000365C000444
S113550B3E54706DF6790000225C0004320C8E73EE
S113551B68472A7358472619005C0003866A2800DC
S113552BFFE615470C5C0001C819005C0003A040A3
S113553B0C79080001790000275C0004106D765488
S113554B706DF67900002A5C0003F40C8E736847C8
S113555B08735847045C0004826D7654705470547E
S113556B70547054705E00638A7A0600FFE55368CB
S113557B68E803A80246426E680003E80747186E03
S113558B690003E9071751177178106A2800009F08
S108559BC617505C007F
S11355A002D46E680003E8071750790100011A0856
S11355B04B04101140F817117A0000FFE613680A34
S11355C0169A688A5E00636854705E00638A7A067E
S11355D000FFE5536868E803A80246406E680003CD
S11355E0E80747186E690003E90717511771781028
S11355F06A2800009FC617505C0002626E680003B1
S1135600E8071750790100011A084B04101140F8FC
S11356107A0000FFE613680A149A688A5E006368DA
S111562054707A0000FFE6147D0072006A28C1
S113562E00FFE556A801470AA8024716A80347221A
S113563E54706A2900009FCE17517A0000009FCE46
S113564E40686A2900009FE217517A0000009FE02C
S113565E40586A2800FFE555470EA801471AA802CD
S113566E4726A803473254706A290000A00717D1B2
S113567E7A000000A00740326A290000A00B17D160

S113568E7A000000A00B40226A290000A01D17D14A
S10D569E7A000000A01D40126A29E3
S11356A80000A02517D17A000000A02555025470E8
S11356B85E00638A0F840D187A0600FFE6166A28CF
S11356C800FFE55A17500C8018886A2900FFE5592E
S11356D81751091069E01D804F0269E801006BA4A6
S11356E800FFE618F8016AA800FFE61555065E00F4
S11356F8636854705E00638A7A0600FFE6167905CC
S1135708000869604744696079200008 4E026965AA
S11357187A0400FFE6186DF5010069417908002253
S1135728790000215C0002780B870D056961190176
S113573869E117F0010069410A81010069C16960E3
S1135748460818886AA800FFE6155E006368547067
S10457585EEF
S113575900638A6A2800FFE553E80317500D05190A
S1135769EE790600210D004706792500014 6100F41
S1135779E05C0001DA0DE80D605C0001D2403C7980
S1135789250002462E6A2800FFE556E80717500D43
S11357905790100011A084B04101140F86A280021
S11357A9FFE613175066014706790800014 0060D05
S11357B9E840020DE80D605C0001945E00636854E3
S11357C9706DF66A2800FFE5556AA800FFE55246A1
S11357D93C19660D68790000285C0001720D68792F
S11357E900002C5C0001680D68790000305C000141
S11357F95E0D68790000345C0001540D687900007E
S1135809385C00014A0D687900003C404018886AF9
S1135819A800FFE613790600010D605C0000867994
S11358290800011790000285C000124790800037934
S113583900002B5C0001180D60554A7908001279A4
S108584900002C5C00CF
S113584E01080D687900002F5C0000FE6D76547020
S113585E6DF60D065C0000E4C88017500D080D6050
S113586E5C0000E66D7654706DF60D065C0000CCA0
S113587EE87F17500D080D605C0000CE6D765470F6
S113588E01006DF60D0617F61036790800087860DC
S113589E6B2000FFE0085C0000B001006D765470D1
S11358AE5E00638A0D0617F610367A0500FFE000D8
S11358BE0AE569505C00008417500D06703E0D68B2
S11358CE69505C0000845E00636854705E00638AF6
S11358DE1B971B977A0500FFE6140D0EFE01790048
S11358EE00010DE11A094B04101040F868591751C5
S11358FE66104704702E4002722E701E17560DE06E
S113590E17F0103001006FF000040D6801006F7185
S113591E000478106B2000FFE000010069F1552AA6
S113592E790000011B5E4B04101040F8685915896D
S109593E68D90B970B97DB
S10459445E01
S1135945006368547 06AA8004000036A2800400099
S113595501547 00D016AA9004000030D806AA80077
S1135965400001547 05E00638A7A03004000030F10
S1135975840F9568BC19EE196640180DC055C6E825
S11359850F471868BC6A280040000168D80B750BDF
S11359955E0B566F7000181D064DE00DE05E00634B
S11359A56854705E00638A0D0C0D830F956F740048
S11359B51819EE1966401E0D30558AE81F471A0D52
S11359C5C06AA80040000368586AA8004000010B9C
S11359D5750B5E0B561D464DDE0DE05E0063685488

S11359E57001006DF619EE7A0000FFE5D379010029
S11359F5405C0001140D06790000015C00FEAA0D50
S1135A056647166DF67A0100FFE5D37908002A7912
S1135A15000029558E0B870D0E790000015C00FEF1
S1135A25B40DE001006D76547019006BA000FFE61C
S1065A351E6BA042
S1135A3800FFE61C19006BA000FFE6226BA000FF25
S1135A48E62054705E00638A7A0400FFE61E7A0536
S1135A5800FFE61C0F830D1E7FF572501999403025
S1135A686940792001004C2C695017F06839780097
S1135A786AA900FFE62469500B5017F07901010069
S1135A8801D0531069D869400B5069C00B730B5987
S10E5A981DE94DCC7FF570500D905EB2
S1135AA300636854705E00638A790E01007A040010
S1135AB3FFE6227A0500FFE6200F830D127FF572BE
S1135AC3501999403469407920010 04C32695017C9
S1135AD3F001D053E00D8017F0683978006AA9000C
S1135AE3FFE72469500B5017F001D053E069D869DD
S1135AF3400B5069C00B730B590D201D094DC67F15
S1135B03F570500D905E00636854705E00638A7A8B
S1135B130500FFE6220F840D1E7FF57250196640C0
S1135B23381DE64C386B2000FFE6206951191079C4
S1135B3310010017F07901010001D053100D8017F4
S1135B43F00D6117F10AC178006A2800FFE72468A2
S1135B539869501B5069D00B5669504EC47FF5703A
S1135B63500D605E00636854705E00638A7A0500BB
S1135B73FFE61E0F840D1E7FF57250196640381D14
S1135B83E64C386B2000FFE61C69511910791001AC
S1085B930017F0790189
S1135B98010001D053100D8017F00D6117F10AC1F0
S1135BA878006A2800FFE624689869501B5069D07A
S1105BB80B5669504EC47FF570500D605EB2
S1135BC500636854706B2000FFE61E17F07901012E
S1135BD50001D053100D8054706B2000FFE622178F
S1105BE5F07901010001D053100D805470C0
S1139FC620282C3034383C20120100010000000800
S1139FD6FEFF10000100010102010902270 0010131
S1139FE600C06409040000030000000307058102A2
S1139FF64000FF070502024000FF070583024000F9
S113A006FF040309041203550053004200200054C1
S113A01600450053005400080331002E003000228F
S113A026035500530042002000540045 00530054DA
S113A03600200050005 2004F004700520041004DDF
S104A0460016
S11399C00023002B0033003B0027002F0037003F0C
S113A047303020303120303220303320303420304C
S113A057352030362030372030382030392030 4112
S113A06720304220304320304420304520304 60AF8
S10CA07700253032582000 0A00D4
S1135BF201006DF67A0600FFE860010069607A0130
S1135C0241C64E6D5E0082427A10000030390100B7
S1135C1269E06960198817707A01000080005E00EC
S10D5C22821C0D1001006D76547012
S1135C2C5E00638A0F857A06000000047A00000088
S1135C3C00180AF00AE07308470E7A0000000018F7
S1135C4C0AF00AE00B70400A7A00000000180AF010
S1135C5C0AE00F8601006DF001006F70001C01005B

S1135C6C6DF001006DF51A91790000015E0063B2CD
S1115C7C7A170000000C0D065E00636854707A
S1135C8A5E00638A0F860F9540040B760B7568696D
S1135C9A68581C8946040C9946F068681750685D71
S10D5CAA175519505E00636854702B
S1135CB45E00638A0F840FC60F950FE00B766C5951
S10F5CC4688946F60FC05E0063685470E8
S1135CD05E00638A0F851AE640020B760FD00B75C0
S10F5CE0680946F60FE05E00636854702C
S1135CEC5E00638A0F850F9601006F710018010027
S1135CFC6DF101006DF601006DF51A91790000014B
S1135D0C5E0063B27A170000000C5E006368547087
S1135D1C0FC446120FD5460E792107FF46120FB357
S1135D2C4604156E4A0A1AC47A0500000008186660
S1135D3C5A005FB00D9946120FC4461A0FD546167A
S1135D4C0FB3461E16E6580002660FA246360FB373
S1135D5C46325800025A0FA246140FB3461058008D
S1135D6C024E0CE60D190FA40FB55A005FBC10358B
S1135D7C1234792C00104C0C1B5910351234792C1D
S1135D8C00104DF410331232792A00104C0C1B51B5
S1135D9C10331232792A00104DF4580000CA1AC479
S1135DAC1AD5186619995800020601006DF2010004
S1135DBC6DF301006F23000401006DF301006923EF
S1135DCC795B800001006DF30FF2550E0B970B9767
S1135DDC01006D7301006D7254706DF601006DF569
S1135DEC01006DF401006DF301006DF201006DF122
S1135DFC01006DF0010069140100692510341035A0
S1095E0C1FD442184510EB
S1135E1201006F14000401006F2500041FD444061F
S1135E220F940FA10FC26816682E0FA00100691408
S1135E3201006F150004010069020100 6F030004F1
S1135E42796C000F796A000F691911191119111967
S1135E521119796907FF6901111111111111111139
S1135E62796107FF792907FF5870FEAE0D1158704B
S1135E72FECC10351234103512341035123410336F
S1135E8212321033123210331232794C0080794AB3
S1135E9200800D901910474C792000364E3E792030
S1135EA200204D0E793000200FB34702700A0FA372
S1135EB21AA2792000104D14793000100D380DB359
S1135EC20D2B0DA219AA0D884702700B0C884F14D3
S1135ED2113213334402700B1B5040F01AA27A039F
S1135EE2000000010C6815E84B2A0AB544020B7442
S1135EF20AA4792C0100585000801134133544024E
S1075F02700D0B59B7
S1135F06792907FF5860006E1AC41AD5580000A6EF
S1135F161FA446061FB55870FE8A1AB544087A347C
S1135F260000000145061AA4450440121AA41735B9
S1135F3617347A150000000144020B740CE60FC4F3
S1135F4646080FD41AD57919FFE00DCC460C0D4C33
S1135F560DD40D5D19557919FFF0792C010045080B
S1135F66113413350B5940F2792C00804C08103547
S1135F7612341B5940F20D994E10113413350D99F5
S1135F864A08113413350B5940F4732D471C0CD8AA
S1135F96E80B47167A150000000844020B74792CA7
S1135FA601004D06113413350B59113413351134D1
S1135FB6133511341335796C000F10191019101994
S1135FC61019649C101C1006131C01006D7001004F

S1135FD6698401006F85000401006D7101006D7213
S1135FE601006D7301006D7401006D756D7654705B
S1135FF601F06401586000807924O7FF5860006C43
S11360065800007401F064235860006C40520F85F9
S113601601F06415460E0D44464601F064234640DE
S1136026580000541031123079280010 4C0C1B5CB8
S113603610311230792800104DF4580000BA0FA51C
S113604601F06435472610331232792A00104C0CBE
S11360561B5410331232792A00104DF45800009E57
S11360661AC4100E131C1AD55800018E790E07FF99
S11360761AC41AD5580001621AC418EE790E07FF1E
S113608619DD790500085800015001006DF601007D
S11360966DF501006DF401006DF301006DF2010071
S11360A66DF101006DF0681E6826156E691C111CE2
S11360B6111C111C111C796C07FF6924111411148E
S11360C611141114796407FF01006D1001006911A1
S11360D67968000F01006F23000401006922796AC1
S10960E6000F792C07FFF7
S11360EC5870FF06792407FF5870FF120DCC5870B7
S11360FCFF160D445870FF40194C791C03FF0DCE4D
S113610C792EFFCC58D0FF5279480010794A0010F1
S113611C1AC47A0500000100103312321031123008
S113612C441C0AB144087A100000000145040AA07B
S113613C40040AA004011235123444E0401C1AB185
S113614C44087A300000000145041AA040041AA048
S113615C04011235DD01123444C2730D46080AB131
S113616C44020B700AA001F064014702700D792CF4
S113617C00804C06103512341B5E792E07FF58C075
S113618CFEE40DEE4E2E113413354402700D0DEE5C
S113619C4A040B5E40F0732D47180CD8E80B4712DA
S11361AC7A1500000008440A0B74792C00804D0208
S11361BC0B5E4020732D471C0CD8E80B47167A1541
S11361CC0000000844020B74792C01004D061134B5
S10961DC13350B5E1134C4
S11361E213351134133511341335796C000F101E26
S11361F2101E101E101E64EC101C100E131C010046
S11362026D700100698401006F85000401006D71E6
S113621201006D7201006D7301006D7401006D75F3
S109622201006D765470CB
S113622801006DF201006DF101006F01000401002E
S113623869000D82111211121112112112796207FFEE
S11362480D8A7968000F793203FF4B4A792200538C
S11362584E44794800107912FFEC4B227922002032
S11362684C0C0D224F1E103112301B5240F40F906C
S11362787912FFE00D224F0C10301B5240F61130FB
S11362880B524BFA796A8000470217B001006D710F
S10D629801006D7254701A8040F289
S10F62A25E008160A801470219005470DF
S11362AE5E0081600C884E0419005470F80154701E
S11362BE5E0081600C884604F80154701900547016
S11362CE01006DF201006DF11AA20F9147224A06E9
S11362DE7902080017B17912041F1B52103144FAC8
S11362EE10311212103112121031121210311212 09
S11362FE698201006F8100026F8A000601006D71D1
S109630E01006D725470E2
S113631401006DF2010069020100699201006F023C
S1116324000401006F92000401006D725470BA

S11363326DF301006DF201006DF101006DF06C0B64
S1136342689B0B011B7246F601006D7001006D71B3
S10B635201006D726D735470BC
S111635A5E0081601A08470479000001547048
S113636801006F76001401006D7201006FF20010D6
S113637801006D7201006D7301006D7401006D758C
S105638854704C
S113638A01006DF501006DF401006DF301006DF27A
S113639A01006F72001001006DF201006FF6001424
S10B63AA01006F72000454703E
S11363B25E00638A7A37000000B27A0300FFE8388E
S11363C27A0400FFE8280D0201006FF100AA010020
S11363D26F7600CE01006F7500D20D00466C01008E
S11363E26F7000CA460E790004526BA000FFE86486
S11363F25A00648401006F7000CA6E080010E80737
S113640217D0460C790005146BA000FFE8644072B4
S113641201006F7000CA6E080010732846107308DB
S1136422470C790105166BA100FFE8644054010093
S11364326F7000CA6E0800107368464601006F70E1
S113644200AA01006BA000FFE85C0D2017F0010019
S11364526BA000FFE82401006F7000CA01006BA06B
S113646200FFE82A1A8001006BA000FFE82E1888BB
S113647268C8F8306EF800885C001CBA0D005870CA
S11364820A1C7900FFFF5A006EC26868A82558608B
S113649209CC0B76188868C80FF001006BA000FFC7
S10764A2E83C0FF0D0
S11364A601006BA000FFE8401A8001006BA000FF0B
S10964B6E84401006BA0A5
S11364BC00FFE8481A8001006BA000FFE84C0100C4
S11364CC6BA000FFE8501A8001006BA000FFE8549A
S11364DC40306868A8204718A8234720A82B470AF0
S11364ECA82D461C7D40701040167D407030401026
S11364FC7C407330460A7D40704040047D407020E0
S113650C0B766868A82D47CAA82B47C6A82047C294
S113651CA82347BEF9206AA900FFE8326869A930AD
S113652C460AF9306AA900FFE8320B761A8001009B
S113653C6BA000FFE8346868A82A586000800FD06D
S113654C0BF001006FF000A07308470A01006F7095
S113655C00A00B70400601006F7000A00F851BF0AC
S113656C01006FF000967308470A01006F700096E4
S113657C1B70400601006F700096690017F0010054
S113658C6BA000FFE8344C167D40701001006B20AB
S113659C00FFE83417B001006BA000FFE8340100E2
S10965AC6B2000FFE83440
S11365B27A20000002004F0E7D407000790005181A
S11365C26BA000FFE8640B76686817D0796000FF60
S11365D217F078006A280000A08873285870008694
S11365E21A8001006BA000FFE8344064686817D08A
S11365F27930003017F001006FF000A07A0100003B
S110660202001A810F907A010000000A5E69
S113660F00821C01006B2100FFE8341F904D240111
S113661F006B2000FFE8347A010000000A5E00825D
S113662F4201006F7100A00A9001006BA000FFE808
S113663F34400E7D407000790005186BA000FFE811
S113664F640B76686817D0796000FF17F078006ADB
S113665F280000A088732846867A00FFFFFFFF01FA
S113666F0069B06868A82E586001000B766868A8A7

S113667F2A466C0FD00BF001006FF000967308479A
S113668F0A01006F7000960B70400601006F7000D7
S113669F960F851BF001006FF000A07308470A01E6
S11366AF006F7000A01B70400601006F7000A0699F
S11366BF0017F0010069B04C0A7A00FFFFFFFF01DA
S11366CF0069B0010069307A20000002004F0E7D8F
S11366DF407000790005186BA000FFE8640B766823
S11366EF6817D0796000FF17F078006A280000A0C0
S10866FF8873284776B3
S11367041A80010069B04058686817D079300030A6
S113671417F001006FF000A07A01000002001A8153
S11367240F907A010000000A5E00821C01006931A7
S11367341F904D1C010069307A010000000A5E00BD
S1136744824201006F7100A00A90010069B0400EFB
S11367547D407000790005186BA000FFE8640B7698
S1136764686817D0796000FF17F078006A28000082
S1136774A088732846927A000000002001006BA0D1
S113678400FFE8586868A8684708A86C4704A84C41
S1136794461068 6817D017F001006BA000FFE85893
S11367A40B766868A825587004A4A845587002128B
S11367B4A8475870020CA8584742A863587002E4CB
S11367C4A8644738A865587001F8A866587001F2A0
S11367D4A867587001ECA8694722A86E5870043062
S11367E4A86F4718A87058700398A873587002F8D4
S10967F4A8754708A87810
S10D67FA47045A006C8E01006B2067
S113680400FFE8587A200000006C46440FD00B9038
S113681401006FF000AA7308470A01006F7000AA11
S11368240B70400601006F7000AA0F851B900100D6
S11368346FF000967308470A01006F7000961B708F
S1136844400601006F700096010069025A006996C0
S113685401006B2000FFE8587A20000000685860AC
S1136864009A6868A875470CA8584708A87847048D
S1136874A86F46420FD00BF001006FF000AA730813
S1136884470A01006F7000AA0B70400601006F7085
S113689400AA0F851BF001006FF000967308470AE6
S11368A401006F7000961B70400601006F70009624
S11368B46900177040400FD00BF001006FF0009691
S11368C47308470A01006F7000960B7040060100BD
S11368D46F7000960F851BF001006FF000AA730818
S11368E4470A01006F7000AA1B70400601006F7015
S10968F400AA690017F081
S11368FA0F825A0069966868A875470CA858470812
S113690AA8784704A86F46420FD00BF001006FF036
S113691A00967308470A01006F7000960B704006D1
S113692A01006F7000960F851BF001006FF000AA3B
S113693A7308470A01006F7000AA1B704006010022
S113694A6F7000AA6900177040400FD00BF0010066
S113695A6FF000AA7308470A01006F7000AA0B7050
S113696A400601006F7000AA0F851BF001006FF04B
S113697A00967308470A01006F7000961B70400661
S113698A01006F700096690017F00F8201006930E9
S113699A7A20FFFFFFFF460A7A0000000001010088
S11369AA69B0686817D06DF07A01000000020AF135
S11369BA0FA05C00050E0B875A006C0201006B20C6
S11369CA00FFE8587A200000004C46420FD00B9093
S11369DA0B9001006FF000AA7308470A01006F7059

S10969EA00AA0B70400639
S11369F001006F7000AA0F851B901B9001006FF0C0
S1136A0000967308470A01006F7000961B70404A96
S1136A1001006F70009640420FD00B900B90010065
S1136A206FF000AA7308470A01006F7000AA0B7089
S1136A30400601006F7000AA0F851B901B90010098
S1136A406FF000967308470A01006F7100961B717F
S1136A50400601006F7100960F907A0100000080DC
S1136A600AF15E006314010069307A20FFFFFFFF23
S1136A70460A7A0000000006010069B0686917D170
S1136A8001006F70008401006DF001006F700084DD
S1136A9001006DF07A00000000080AF05C00078630
S1136AA00B970B975A006C020FD00BF001006FF09D
S1136AB000AA7308470A01006F7000AA0B70400612
S1136AC001006F7000AA0F851BF001006FF00096A4
S1136AD07308470A01006F7000961B70400601009F
S1096AE06F7000966E08C2
S1136AE600015A006C540FD00B9001006FF000AAFE
S1136AF67308470A01006F7000AA0B704006010075
S1136B066F7000AA0F851B9001006FF00096730843
S1136B16470A01006F7000961B70400601006F70F4
S1136B260096010069000F825E005CD001006FF0E1
S1136B3600AA010069317A21FFFFFFFF4712010016
S1136B4669311F814C0A0100693001006FF000AA08
S1136B560FF001006BA000FFE8401A8001006BA054
S1136B6600FFE84C01006B2000FFE83401006F7161
S1106B7600AA1A9001006BA000FFE8545A1A
S1136B83006C9C0FD00B9001006FF00096730847C5
S1136B930A01006F7000960B70400601006F7000CE
S1136BA3960F851B9001006FF000AA7308470A0133
S1136BB3006F7000AA1B70400601006F7000AA01EA
S1136BC30069007A6000FFFFFF01006FF000960188
S1136BD30069307A20FFFFFFFF460A7A00000000B6
S1136BE301010069B0790000786DF07A01000000BB
S1136BF3020AF101006F7000985C0002CE0B870F4D
S1136C03F00F825E005CD001006FF000AA5A006CA3
S1136C139C0FD00B9001006FF000A07308470A018B
S1136C23006F7000A00B70400601006F7000A00F8F
S1136C33851B900F81730847060F901B7040020F4B
S1136C4390010069006B2100FFE8306981404AF835
S1136C532568F80FF00F827A000000000101006F2E
S1136C63F000AA0FF101006BA100FFE8401A9101A4
S1066C73006BA10F
S1136C7600FFE84C01006B2100FFE8341A81010094
S1136C866BA100FFE854400E790005186BA000FFC6
S1136C96E8647D4070006868A86E587001B86A2879
S1136CA600FFE8287308586001AC7C407310463631
S1136CB601006B2000FFE8544F2C40207A010000AE
S1136CC600017A0000FFE8325C0013B87A0000FF87
S1136CD6E854010069011B710100698101006B2001
S1136CE600FFE8544ED601006B2000FFE848462417
S1136CF601006B2000FFE84C461A01006B2000FFE1
S1136D06E850461001006F7100AA0FA05C001374CF
S1136D165A006E2001006B2000FFE83C0FA11A9079
S1136D2601006FF0009C01006FF000A04F300100DE
S1136D366F71009C0FA05C00134A40227A0000008A
S1136D4600880AF07A01000000015C0013367A001D

S1136D5600FFE848010069011B7101006981010018
S1136D666B2000FFE8484ED401006B2000FFE8408B
S1136D7601006B2100FFE83C1A9001006FF0009CB4
S1136D8601006F7100A00A8101006FF100A00F805E
S1136D964F3601006F71009C01006B2000FFE83C39
S1136DA65C0012E040227A00000000880AF07A01B3
S1136DB6000000015C0012CC7A0000FFE84C0100E1
S1136DC669011B710100698101006B2000FFE84C1A
S1096DD64ED401006F71B1
S1116DDC00AA01006F7000A01A8101006B2055
S1136DEA00FFE8405C00129840227A000000008805
S1136DFA0AF07A01000000015C0012847A0000FFA5
S1136E0AE850010069011B710100698101006B20CF
S1136E1A00FFE8504ED47C407310473601006B20C4
S1136E2A00FFE8544F2C40207A01000000017A0049
S1136E3A00FFE8325C0012487A0000FFE8540100C0
S1136E4A69011B710100698101006B2000FFE8548D
S1136E5A4ED60B76404001006FF600A6400E0100A5
S1136E6A6F7000A60B7001006FF000A601006F702F
S1136E7A00A668086EF8009547086E780095A8255D
S1136E8A46DC01006F7100A61AE10FE05C0011F005
S1136E9A01006F7600A6686847145C0012900D0023
S1136EAA460C6A2800FFE82873085870F5D45C007A
S1136EBA125E6B2000FFE8307A17000000B25E0012
S1136ECA636854705E00638A7A37000000247A0587
S1096EDA000000040AF5AC
S1136EE001006FF0001801006FF1002001006FF145
S1136EF0001C18AA689A6F72003C0C22463CAA58E0
S1136F00472CAA644712AA69470EAA6F4712AA75AB
S1136F104706AA78471840227A000000000A400674
S1136F207A000000000801006FF00014400C7A00A2
S1136F300000001001006FF000146F70003C792016
S1136F4000644706792000694612010006F7000183B
S1136F504C0A01006F74001817B4400601006F74E7
S1136F6000181AE61AB30FC4467401006B2000FF21
S1136F70E838476AF83068D87A06000000014062B2
S1136F800FD00AE00F82010069F00FC001006F719A
S1136F9000145E008846010069706889 0FA06808C4
S1136FA0A8094E0C0FD00AE0680989306889401E91
S1136FB06F70003C79200078460A0FD00AE0680918
S1136FC0895740080FD00AE06809893768890FC0DC
S1096FD001006F710014C3
S1116FD65E0088460F840B760FC4469E6A2821
S1136FE400FFE828732847626F70003C0C00465A80
S1136FF4A858471EA86F4706A8784716404C0100B7
S11370046F70001847440FE00B760AD0F930688993
S1137014403801006F700018473001006F70002082
S11370240B7001006FF000201B70F93068890100B8
S11370346F7000200B7001006FF000201B706E79DD
S1137044003D68897A03000000020FB00AE00F8351
S113705401006FF600146F70003C7920006447064A
S1137064792000694672010006B2000FFE838460866
S113707401006F700018476001006F7000184D42E3
S11370846A2800FFE828733847160B7301006F70F2
S113709400200B7001006FF000201B70F92B4036A9
S11370A46A2800FFE8287348472E0B7301006F70AA
S11370B400200B7001006FF000201B70F920688919

S11370C440160B7301006F7000200B7001006FF00A
S10970D400201B70F92DE2
S11370DA68897A0600FFE84001006F70001C680C9B
S11370EAAC2B4708AC2D4704AC20460E01006F7049
S11370FA001C0B70010069E040466A2800FFE8287B
S113710A732847326F70003C0C004634A858470A6C
S113711AA86F4714A8784702402601006F70001C25
S113712A0BF0010069E0401801006F70001C0B703E
S113713A010069E0400A01006F70001C010069E068
S113714A7A0400FFE838010069401FB04F140100B8
S113715A6946010069441AB401006BA400FFE84CB4
S113716A400C0FB61A8001006BA000FFE84C7A04AA
S113717A00FFE85401006F70001C680BAB2B470833
S113718AAB2D4704AB20463801006B2000FFE834DF
S113719A1FE04F2C6A2800FFE832A830462201007C
S10D71AA6B2000FFE8341AE07A01BD
S11371B400FFE84C010069120A82010069921A80F7
S11371C4010069C0401E01006B2000FFE8341FE08A
S11371D44F0C01006B2000FFE8341AE040021A80D0
S11371E4010069C001006F7600141B76401A010088
S11371F46F7000200B7001006FF000201B700FE113
S11372041B760AD1681968890FE64CE201006F7096
S1137214002018996889 7A17000000245E006368C7
S113722454705E00638A7A37000000647A0300FFB7
S1137234E8287A04000000040AF47A0500FFE83819
S11372440F866FF1005001006FF6005CF83068C8D8
S11372547A000000007C0AF07A010000A0805E003E
S1137264635A0D00587000B87A000000002F0AF02A
S113727401006DF07A000000002E0AF001006DF0A9
S113728401006F70008801006DF001006F700088C9
S113729401006DF07A01000000320AF17A00000067
S10972A400115E0082628E
S11372AA7A17000000100D024752188868E8010097
S11372BA695001006B2100FFE8341F904F0601005B
S11372CA6950400801006B2000FFE83401006BA0FD
S11372DA00FFE8547A0600FFE8320D207920FFFF09
S11372EA470C79200001460CF82B5A008012F82D1E
S10472FA5A36
S11372FB008012F82A68E85A0080147A0000000014
S113730B1101006DF00FC10B717A00000000260A0A
S113731BF05E0086480B97400CF8016EF8002F18AF
S113732B886EC800017A000000000101006FF000B5
S113733B60400E01006F7000600B7001006FF00076
S113734B6001006F7000600AC06808A83047E40151
S113735B006F7000607A20000000014F7A01006F0C
S113736B7000600AC05E005CD00F8201006F70007A
S113737B601B7001006F71002A0A8101006FF1001D
S113738B2A7A000000000101006FF0005840300121
S113739B006F7000600AC001006F7100580AC1686A
S11373AB08689801006F7000580B7001006FF000B4
S11373BB5801006F7000600B7001006FF0006001EB
S11373CB006F7000580FA11F904FC401006F700026
S11373DB580AC0189968890FC00B7001006FF00031
S10873EB445E005CD0CC
S11373F001006FF000601A8001006FF00058010077
S11374006F7000607A01000000025E00821C0100C0
S11374106FF0004001006F70004401006F71006065

S11374200A901B7001006FF0003C404E01006F702A
S1137430004401006F7100580A9001006FF0004C86
S113744068086EF8005701006F71003C01006F720D
S113745000581AA101006FF1004801006F72004C3F
S1137460681968A901006F710048689801006F717D
S113747000580B7101006FF1005801006F70005844
S113748001006F7100401F904DA201006F700060FA
S1137490461AF8306EC800017A00000000010100AE
S11374A06FF000601A8001006FF0002A6F700050C7
S11374B07920006747067920004746 3C7D307050AD
S11374C001006F70002A01006F7100600A901B7049
S11374D07A20FFFFFFFC4D0C01006B2100FFE83811
S10974E01F904F0C6F70BA
S11374E600501BD06FF000504008790000666FF023
S11374F600506E78002FA80146246838E8186EF805
S11375060057A8084706A810470A401A0FE00B764B
S1137516F92B40100FE00B76F920688940080FE03D
S11375260B76F92D68896F70005079200066586 0D4
S1137536077201006F70002A58D001861A80401A1C
S11375460FE00B7601006F7100580AC10B716819C1
S1137556688901006F7000580B7001006FF00058C6
S113756601006F7100601F904DD61A800F820100D3
S11375766BA600FFE84001006F70002A01006BA0B4
S113758600FFE84C7C307350460A01006B2000FF75
S1137596E8384E067C307320470E0FE00B76F92E43
S11375A668890FA00B700F827C307350470C7C30B8
S11375B6735047247C307320471E01006BA600FFDF
S11375C6E8440100695001006BA000FFE850010088
S10975D669500FA10A81B8
S11375DC0F9201006F70002A0FA10A9001006F71C6
S11375EC00600A9001006FF0006001006F71005C95
S11375FC6819A92B4708A92D4704A9204652010055
S113760C6B2000FFE83401006F7100601F904F4046
S113761C6A2800FFE832A830463601006F70005C20
S109762C0B7001006BA0CE
S113763200FFE83C01006B2000FFE83401006F719A
S113764200601A9001006BA000FFE8481A80010055
S11376526BA000FFE8545A00801001006B2000FF6A
S1137662E83401006F7100601F904F4C01006B20E2
S113767200FFE83401006F7100601A9001006BA0F3
S113768200FFE8546E78002FA80146186E78002F89
S1137692A801586009786E780057A8084706A81011
S11376A25860096A7A0000FFE854010069011B71FE
S11376B2010069815A0080101A8001006BA000FF4B
S11376C2E8545A00801001006F70006001006F716E
S11376D2002A0A8101006FF1002A010069520AA1FE
S11376E201006FF100521F814C120F914D0E0100E8
S11376F26F7100520B710FC05C00092201006F70A1
S1137702002A58F0028A6848A83046147A0000001A
S1137712000101006FF0004C01006FF00052402C99
S10977221A8001006FF064
S1137728004C1A8001006FF0005201006F70002AAC
S11377380B7001006FF0002A01006F7000600B707E
S113774801006FF000601A8001006FF00058403AA2
S11377580FE00B7601006F71004C0AC16819688944
S113776801006F70002A1B7001006FF0002A0100EE
S11377786F7000580B7001006FF0005801006F70B4

S1137788004C0B7001006FF0004C01006F70002A71
S11377984EBE0FE00B76F92E688940240FE00B7676
S11377A801006F71004C0B7101006FF1004C1B71EC
S11377B80AC168196889010069501B70010069D002
S11377C801006F7000580B7001006FF00058010042
S11377D86F70004C01006F7100521A9001006F71B5
S11377E800601F904C0A01006B2000FFE8384EAC84
S11377F87C307350470C7C307350472E7C30732099
S1137808472801006950010 06F7100580A8101007F
S10778186FF10058B1
S107781C01006BA653
S113782000FFE8400100695001006BA000FFE84C35
S113783040226E68FFFFA830461A40101B760100F5
S11378406F7000581B7001006FF000586E68FFFFE7
S1137850A83047E87C307320464A6E68FFFFA82EA5
S1137860464201006B2000FFE84C47067C30735012
S113787047321B7601006F70005801006B2100FF37
S1137880E84C1A901B7001006FF000581A80010039
S11378906BA000FFE84C01006F70005C01006BA05F
S11378A000FFE84001006F70005C6808A82B4708E0
S11378B0A82D4704A820465001006B2000FFE834A0
S11378C001006F7100581F904F3E6A2800FFE83295
S11378D0A830463401006F70005C0B7001006BA090
S11378E000FFE83C01006B2000FFE83401006F71EA
S11378F000581A9001006BA000FFE8481A800100AD
S11379006BA000FFE854406201006B2000FFE834E5
S109791001006F71005835
S10B79161F904F4601006B2096
S113791E00FFE83401006F7100581A9001006BA04C
S113792E00FFE8546E78002FA80146146E78002FDE
S113793EA80146286E780057A8084704A810461CCD
S113794E7A0000FFE854010069011B71010069818F
S113795E400A1A8001006BA000FFE85401006B205F
S113796E00FFE84001006F71005C1F9058600692A3
S113797E01006B2000FFE83C01006BA000FFE84014
S113798E5A0080106848A83046147A00000000019F
S113799E01006FF0004C01006FF00052402A1A8074
S11379AE01006FF0004C01006FF0005201006F7088
S11379BE002A0B7001006FF0002A01006F70006047
S11379CE0B7001006FF0006001006F70002A4F14FE
S11379DE7A000000000101006FF0004C0FE00B76FF
S11379EE684940060FE00B76F93068897A0000008B
S11379FE000101006FF000580FE10B76FA2E689A22
S1097A0E01006F70005837
S1137A140B7001006FF0005801006F70002A58C00A
S1137A24009E01006BA600FFE84001006F70002A6E
S1137A3417B0010069511F814F2001006F70002AA4
S1137A4417B001006BA000FFE84C01006F70002A1F
S1137A5401006F7100581A81401801006950010038
S1137A646BA000FFE84C0100695001006F710058DE
S1137A740A8101006FF1005801006F70002A0100B0
S1137A8469510A81010069D140360FE00B76010088
S1137A946F71004C0AC168196889010069501B7031
S1137AA4010069D001006F7000580B7001006FF082
S1137AB4005801006F70004C0B7001006FF0004C14
S1137AC401006F70004C01006F7100521A900100A5
S1137AD46F7100601F904C0A01006B2000FFE838AF

S1137AE44EA87C307350470C7C30735047347C3041
S1137AF47320472E010069504F4A01006BA600FF13
S1097B04E8440100695092
S1077B0A01006BA068
S1137B0E00FFE8500100695001006F7100580A81AF
S1137B1E01006FF1005840226E68FFFFA830461A2D
S1137B2E40101B7601006F7000581B7001006FF040
S1137B3E00586E68FFFFA83047E87C30732046700C
S1137B4E6E68FFFFA82E466801006B2000FFE84C0D
S1137B5E460A01006B2000FFE85047067C30735045
S1137B6E474E1B7601006F70005801006B2100FF1A
S1137B7EE84C1A9001006B2100FFE8501A901B701D
S1137B8E01006FF000581A8001006BA000FFE8504F
S1137B9E01006BA000FFE84C01006F70005C010058
S1137BAE6BA000FFE8401A8001006BA000FFE844C1
S1137BBE01006F70005C6808A82B4708A82D4704C6
S1137BCEA820465001006B2000FFE83401006F71BE
S1137BDE00581F904F3E6A2800FFE832A830463403
S1137BEE01006F70005C0B7001006BA000FFE83C9E
S1077BFE01006B20F4
S1137C0200FFE83401006F7100581A9001006BA065
S1137C1200FFE8481A8001006BA000FFE8544062AD
S1137C2201006B2000FFE83401006F7100581F90C0
S1137C324F4601006B2000FFE83401006F710058CA
S1137C421A9001006BA000FFE8546E78002FA80180
S1137C5246146E78002FA80146286E780057A808AC
S1137C624704A810461C7A0000FFE854010069018A
S1137C721B7101006981400A1A8001006BA000FF99
S1137C82E85401006B2000FFE84001006F71005CC3
S1137C921F90461001006B2000FFE83C01006BA01F
S1137CA200FFE8405A008010010069500B70010088
S1137CB26F7100601F904C0C010069510BF10FC0F2
S1137CC25C00035A6848A830460E7A00000000019F
S1137CD201006FF0004840241A8001006FF0004851
S1137CE201006F70002A1B7001006FF0002A01006F
S1137CF26F7000600B7001006FF0006001006F7025
S1097D02006001006F7137
S1137D0800481A9001006F72002A0A8201006FF27C
S1137D18002A0FE00B760AC1681968897A00000007
S1137D28000101006FF000580FE10B76FA2E689AF4
S1137D3801006F7000580B7001006FF000580100CC
S1137D486F700048402E0FE00B7601006F71004CF6
S1137D580AC168196889010069501B70010069D05C
S1137D6801006F7000580B7001006FF0005801009C
S1137D786F70004C0B7001006FF0004C01006F71C5
S1137D8800601F904E0A01006B2000FFE8384EB6D2
S1137D987C307350470C7C307350472E7C307320F3
S1137DA8472801006BA600FFE84001006950010065
S1057DB86BA0BB
S1137DBA00FFE84C0100695001006F7100580A8105
S1137DCA01006FF1005840226E68FFFFA830461A7F
S1137DDA40101B7601006F7000581B7001006FF092
S1137DEA00586E68FFFFA83047E87C307320464A84
S1137DFA6E68FFFFA82E464201006B2000FFE84C85
S1137E0A47067C30735047321B7601006F70005867
S1137E1A01006B2100FFE84C1A901B7001006FF000
S1137E2A005801006F70005C01006BA000FFE8407E

S1137E3A1A8001006BA000FFE84C6F700050792094
S1137E4A006546080FE00B76F96540060FE00B76EE
S1137E5AF945688901006F7000580B7001006FF0D3
S1137E6A005801006F70002A4D0A0FE00B76F92BB8
S1137E7A688940160FE00B76F92D688901006F7047
S1137E8A002A17B001006FF0002A01006F70005832
S1137E9A0B7001006FF0005801006F70002A7A20FE
S1097EAA000000644D0E10
S1137EB00BF601006F7000580B800B7040140FE03D
S1137EC00B76F9306889F83068E801006F70005864
S1137ED00BF001006FF000580FE00B7001006FF022
S1137EE0003040360FE01B76010069F001006F702F
S1137EF0002A7A010000000A5E00821C893001001A
S1137F006970688901006F70002A7A010000000A15
S1137F105E00821C01006FF0002A01006F70002ACE
S1137F204EC201006F76003001006F70005C68087C
S1137F30A82B4708A82D4704A820465001006B2012
S1137F4000FFE83401006F7100581F904F3E6A280C
S1137F5000FFE832A830463401006F70005C0B70FC
S1137F6001006BA000FFE83C01006B2000FFE83438
S1137F7001006F7100581A9001006BA000FFE848E0
S1137F801A8001006BA000FFE854406201006B20DF
S1137F9000FFE83401006F7100581F904F46010045
S1057FA06B2051
S1137FA200FFE83401006F7100581A9001006BA0C2
S1137FB200FFE8546E78002FA80146146E78002F54
S1137FC2A80146286E780057A8084704A810461C43
S1137FD27A0000FFE854010069011B710100698105
S1137FE2400A1A8001006BA000FFE85401006B20D5
S1137FF200FFE84001006F71005C1F904610010012
S11380026B2000FFE83C01006BA000FFE8401888EA
S113801268E87A17000000645E00636854705E00CB
S1138022638A0F850F960FD00AE06808A8344F526F
S11380320FD00AE0F93068891B760FD00AE068098D
S1138042890168894020 0FD00AE06808A8394F14D3
S11380520FD10AE1681888F668986E18FFFF880145
S11380626E98FFFF1B760FE64EDC6E580001A839AF
S11380724F106E58000188F66ED8000168588801C7
S113808268D85E00636854705E00638A1B977A0344
S109809200FFE82A7A0654
S113809800FFE82E010069F00F9501006B2100FF36
S11380A8E8247A2100000001462601006DF50F81BE
S11380B8010069305E0087EC0B97010069300AD034
S11380C8010069B0010069600AD0010069E0403A23
S11380D819444030010069700B70010069F01B708E
S10F80E8680817D00100693101006B2209
S11380F400FFE85C5D207920FFFF471201006960FF
S11381040B70010069E00B5417F41FD44DCA0B978D
S11381145E006368547001006B2000FFE8247A203A
S1138124000000001460C01006B2000FFE82A1899A7
S11381346889547001006B2000FFE8247A20000052
S1138144000146041900547001006B2000FFE82A63
S10F81546E090010E94017D10D105470A3
S10BA0800000000000000D5
S113816001006DF501006DF401006DF301006DF286
S113817001006DF101006D120100691301006F012F
S1138180004010069000D840D8C796C7FF0792C5B

S11381907FF047540DAC796C7FF0792C7FF0475614
S11381A00D8C65AC4B5E1AB144087A3000000001B7
S11381B045141AA0451001F06410475A0C444B0AA9
S11381C079000002400C0C444BF61900400479007E
S11381D0FFFF01006D7101006D7201006D730100FD
S11381E06D7401006D7554700F85796D000F01F08A
S11381F0641546DA409E0FA5796D000F01F06435D2
S113820046CC409C01F064207A607FFFFFFF46ACC0
S10F821001F0643146A67900000140B67D
S113821C6DF20D820C2A4A0217B00D994A04D280D2
S113822C17B15E0088460C224A0217B00CAA4A0208
S109823C17B16D725470CE
S113824201006DF301006DF20D8252120D93520380
S1138252521009280938 01006D7201006D735470C0
S113A088202020202020202060606060202085
S113A098202020202020202020202020202020B5
S113A0A848101010101010101010101010106D
S113A0B884848484848484848484101010101100D
S113A0C810818181818181010101010101010166
S113A0D80101010101010101010101101010101A
S113A0E810828282828282020202020202020237
S113A0F802020202020202020202021010101020DF
S113A10800000000000000000000000000000044
S113A11800000000000000000000000000000034
S113A12800000000000000000000000000000024
S113A13800000000000000000000000000000014
S113A14800000000000000000000000000000004
S113A158000000000000000000000000000000F4
S113A168000000000000000000000000000000E4
S109A178000000000000DE
S10DA17E0000000000000000000D4
S11382625E00638A7A370000003C7A040000000C47
S11382720AF47A05000000140AF501006FF00038D1
S113828201006FF1003401006F73005C7A02000099
S1138292000801006FF2002C19226FF2002A7A06FD
S11382A2000000540AF6686AEA7F46620FE00B7028
S11382B27A01000000075E0088820D0047501A8091
S11382C2401A01006F70003401006F7100380A9088
S11382D21899688901006F7000380B7001006FF004
S11382E200387A20000000084DD81A80010069B0D6
S11382F27C607370470A01006F700060F9FF4008E9
S113830201006F700060F90168895A00863A7A00A9
S11383120000001C0AF001006DF07A010000002841
S11383220AF101006F70005C01006DF001006F70D3
S1138332005C01006DF001006F70006C5E008C1236
S11383427A170000000C01006F7000606808A8FF34
S1078352460C7A0058
S1138356000000540AF07D0072706F7000247920CB
S113836607FF464A6E680001A8F0463A0FE00BF095
S11383767A01000000065E0088820D00472801008E
S11383866F7000606808A8014608790000015A006A
S1138396863C01006F7000606808A8FF46107900EC
S11383A6FFFF5A00863C790000025A00863C6F7034
S11383B60024793003FF6FF000244D1E01006F7017
S11383C6005801006DF001006F70005801006DF058
S11383D65E0089A20B970B97401E01006F70005831
S11383E601006DF001006F7000 5801006DF05E0032

S11383F689A20B970B971B500D0618886EF800235E
S11384060B560D6017F001006F7100381A900100CA
S113841669B0010069F10FD1010069705E008ABE7F
S113842601006F70003801006F71002C5E00821C22
S113843601006FF000301AE67A0000000008010020
S10984466F7100301A9073
S113844C0F82401401006F7000300AE00AD00FD184
S113845C0AE1680868980B760FA01F864DE6400A60
S113846C0FD00AE0189968890B767A260000000869
S113847C4DEE6F7000326F78002E52806F71003AA0
S113848C19016DF17A01000000080FD05E008A46D5
S113849C0B8701006F7600381B760FC10FE05E006F
S11384AC8ABE0FE001006F71002C5E00821C01007C
S11384BC6FF000301AE67A000000000801006F71BB
S11384CC00301A900F82401401006F7000300AE0E4
S11384DC0AC00FC10AE1680868980B760FA01F86C3
S11384EC4DE6400A0FC00AE0189968890B767A2684
S11384FC000000084DEE6F7000326F78002E528032
S113850C6F71003A19011B516DF17A0100000008DB
S113851C0FC05E008A460B8701006F7000340100A8
S113852C6DF00100693101006DF17A010000002446
S109853C0AF16F70002C30
S11385425E0088AA0B970B977A06000000040AF6CE
S11385527A000000000801006DF001006F7100381D
S11385620FE05E0087EC0B977A0000000008010021
S10A85726DF00FD10FE05E75
S1138579008CB60B970D004F12790000016FF000C4
S11385892A010069300B705A0086327A0000000014
S11385990801006DF001006F7100380FE05E00877C
S11385A9EC0B977A000000000801006DF00FD10F62
S11385B9E05E008CB60B970D004632010069300B63
S11385C970010069B001006F70003401006DF001A2
S11385D900693301006DF37A01000000240AF16F89
S11385E970002C5E0088AA0B970B9740447A000011
S11385F900000801006DF001006F7100380FE05EA3
S11386090087EC0B977A000000000801006DF00F5A
S1138619C10FE05E008CB60B970D004C146F700010
S11386292A460E010069301B70010069B05A0085A2
S11286392419007A170000003C5E006368547038
S11386485E00638A7A370000002C0FF30F8601005F
S11386586FF100107A000000000101006FF00018AC
S113866801006F7500440A95188868D87A000000DD
S1138678000801006DF00FE17A000000000C0AF019
S11386885E0087EC0B9701006F7000445A0087D88F
S11386980FA00B707A01000000055E00821C0B70AE
S11386A810307A06000000081A867A0000000008D5
S11386B81AE001006DF00FA11031103110317A1159
S11386C80000A1880AE10FB00AE05E0087EC0B976F
S11386D87A000000000801006FF000281A800100EA
S11386E86FF0002001006FF0001C7A0400000008FE
S11386F81AE401006FF400145A00879A01006F709E
S1138708001401006DF00FB10AE17A000000000CBB
S11387180AF00AE05E008CB60B970D004D3C010091
S11387286DF40FB00AE001006DF07A01000000104B
S10987380AF10AE11A80B8
S113873E5E008B660B970B9717F001006FF0001816
S113874E01006F70002801006F7100200A81010083

S113875E6FF1002001006F7000287A010000000203
S113876E5E00821C01006FF0002847307900000183
S113877E6DF00FB00AE00FC15E008AEC0B870100AB
S113878E6F70001C0B7001006FF0001C01006F7006
S113879E001C7A200000000458D0FF5A01006F70AD
S11387AE0018461C6E78002388306CD84004F830CD
S11387BE68D81B7501006F7000101F8544F04012BE
S11387CE6E78002388306CD80FA01B700F8258C0B0
S11187DEFEB87A170000002C5E00636854702A
S113A188000000000000000800000000000000506C
S113A198000000000000003200000000000001F4032
S113A1A8000000000001388000000000000C3500AA
S113A1B800000000007A12000000000004C4B4008C
S113A1C8000000002FAF080000000001DCD650009B
S113A1D800000012A05F2000000000BA43B740004F
S113A1E800000746A5288000000048C273950000B8
S113A1F80002D79883D20000001C6BF526340000B8
S10BA208011C37937E080000DE
S11387EC5E00638A0F850F9401006F7600181FC516
S11387FC47401FC544200FD30FC21AC440120FB0F9
S113880C0B730FA10B710F921B71681968890B7491
S113881C1FE445EA401C0FD30AE30AE40FC21AC44F
S113882C400C0FA01B700F8268086CB80B741FE40C
S10D883C45F00FD05E00636854702E
S113884601006DF20D9946100D82177253120DA891
S113885653100D810D28401E0F920D811771790853
S1138866001012101231 1AA144020AA11B5846F233
S10F8876121017101770010 06D7254707F
S11388825E00638A0F840F951AE6400E0FC00AE05A
S1138892680947041900400A0B761FD64DEE79008A
S10B88A200015E0063685470DD
S11388AA5E00638A7A370000000C7A050000000133
S11388BA0AF50D0C0F967904000801006DF57A018B
S11388CA0000000E0AF101006F70002817B05E0065
S11388DA8D7C0B9701006DF66DFC7A000000001089
S11388EA0AF00FD15E008CF40B970B87790C003FCB
S11388FA6F70000A190C0DC017F001D053400D0E0A
S113890A7900004219C017F001006DF07A010000E6
S113891A00090FD05E008FC80B9779060008401430
S113892A0D6019E017F00AD00D6117F10AD1680832
S113893A68981B561DE64CE8400C0D6017F00AD0E8
S113894A189968891B560D664CF00DE05240190CB4
S113895A6DFC7A01000000090FD05E008AEC0B87D8
S113896A7900003F6FF0000A7A01000000400FD03F
S113897A5E008EDE7A000000000801006DF00FD160
S113898A01006F70002C5E0087EC0B977A170000CA
S109899A000C5E0063689F
S10589A054700E
S11389A201006DF27A370000001A7A000000001805
S11389B20AF001006F71002601006DF101006F7171
S11389C2002601006DF17A01000000080AF101009E
S11389D26DF15E0090647A170000000C6F7100184D
S11389E20FF05E0092E40F817A020000A2107A0077
S11389F2000000080AF05E0093D07A000000001025
S1138A020AF001006F71000C01006DF101006F713A
S1138A12000C01006DF17A01000000080AF1010067
S1138A226DF15E00916A7A170000000C7A00000073

S1138A3200100AF05E0062287A170000001A010093
S1078A426D7254708A
S10BA2103FD34395810624DDD1
S1138A465E00638A1B971B87010069F00F946F7999
S1138A56001E790D0008199D4754790100FF0DD2B8
S1138A661A0A4B04101140F80C9B18DD1B740FC631
S1138A764028010069740AE468450CB816850FC0DE
S1138A860D915E0092A0684814D868C80DD01A08E4
S1138A964B04110540F80C5D1B760FE64CD40CDD38
S1138AA64706790100014002191117F10F900B8750
S10B8AB60B975E006368547026
S1138ABE5E00638A0F860F957A000000000801009E
S1138ACE6DF01036103610367A010000A2180AE146
S1118ADE0FD05E0087EC0B975E006368547048
S113A2180000000000000010000000000000052D
S113A2280000000000000019000000000000007D8D
S113A2380000000000000027100000000000000C355F
S113A2480000000000003D09000000000001312D5E
S113A2580000000000005F5E100000000001DCD65C9
S113A2680000000009502F90000000002E90EDD7D
S113A278000000000E8D4A510000000048C273958B
S113A288000000016BCC41E9000000071AFD498D6D
S113A2980000002386F26FC1000000B1A2BC2EC5E6
S113A2A8000003782DACE9D900001158E460913D12
S113A2B8000056BC75E2D6310001B1AE4D6E2EF5E5
S113A2C8000878678326EAC9002A5A058FC295EDE4
S113A2D800D3C21BCECCEDA10422CA8B0A00A4254D
S113A2E814ADF4B7320334B96765C793FA10079D01
S1138AEC5E00638A7A370000000A0F83010069F184
S1138AFC6F790022790C0008199C4752FAFF0DC0BC
S1138B0C1A084B04110A40F80CA218CC1AE640269A
S1138B1C0FB50AE568540C2816840FD00D915E002E
S1138B2C92C2685814C868D80DC01A084B041004B4
S1138B3C40F80C4C0B76010069701F864DD20CCC9F
S1138B4C4706790100014002191117F10F907A17AA
S10D8B5C0000000A5E006368547015
S1138B665E00638A7A370000000C01006FF0000490
S1138B760F9601006F7500280AD61B7601006F73E6
S1138B8600240AD31B737A02000000011AC440446E
S1138B9668681750683917511910 17F01AC0010081
S1138BA669F04C14010069717A110000010001009B
S1138BB669F17900000140021900 17F00F840100E2
S1138BC6697047041A800F826E78000368E81B7584
S1138BD61B761B730FD546B87A2400000001460E98
S1138BE601006F70000446067900FFFF40120FA0D4
S1138BF67A2000000001460419004004790000 01B0
S10F8C067A170000000C5E0063685470D5
S1138C125E00638A7A03000000070F820F9501004A
S1138C226F7600207A04000000180AF46941796122
S1138C32800079218000460679 01FFFF4004790113
S1138C4200010FA06889694079607FF0119011904B
S1138C521190119069D07A000000000701006DF0B5
S1138C620B740FC10FE05E0087EC0B9779000003D2
S1138C726DF00FB10FE05E008A460B8769504F0615
S1138C827D607070402869500B5069D07D607270AE
S1138C924016790000016DF00FB10FE05E008A46C5
S1138CA20B8769501B5069D07C60737047E45E0088

S1078CB2636854702C
S1138CB65E00638A0F860F9301006F7400180FE539
S1138CC60FC44604190040201AE6400A6C586C3952
S1138CD61C9846060B761FC645F21B756858175037
S1118CE61B73683B175319305E0063685470AC
S1138CF45E00638A7A37000000100FF60F850F9425
S1138D0469506F710028091069D001006DF60100E4
S1138D146F71002E0FC05E0096AC0B977C6073706E
S1138D24470869500B5069D0400C7A0100000010C9
S1138D340FE05E00965A7A000000004201006DF0D5
S1138D447A01000000100FE05E008FC80B976E6875
S1138D540008E8E06EE800087A000000000901005A
S1138D646DF00FE10FC05E0087EC0B977A170000DC
S10B8D7400105E0063685470F7
S1138D7C5E00638A7A37000000187A050000000948
S1138D8C0AF50F840F920FC44D087A0300000001FB
S1138D9C40147A03FFFFFFFF0FC07A01FFFFFFFFB1
S1138DAC5E0082420F840FC07A010000001B5E003C
S1138DBC821C0F860FC07A010000001B5E00821C10
S1138DCC0F947A2300000001462E7A00000000085D
S1138DDC01006DF00FE11031103110317A110000E8
S1138DECA2F80FD05E0087EC0B970FE010307800E1
S1138DFC6B210000A3C840320FC446021B767A00D5
S1138E0C0000000801006DF00FE11031103110313A
S1138E1C7A110000A3600FD05E0087EC0B970FE074
S1138E2C103078006B210000A3E26FF100120FC425
S1138E3C47747A23FFFFFFFF460A7A000000001BEA
S1138E4C1AC00F8401006F70003001006DF00FA188
S1138E5C0FC05E00983A0B970FE646087A23000082
S1078E6C0001476255
S1138E7001006F70003001006DF00FA069006DF00C
S1138E807A00000000180AF00FD15E008CF40B97F3
S1138E900B877A01000000400FD05E008EDE792040
S1138EA00001460E6F7000120B506FF000127D50E0
S1138EB070700FA06F71001269817A0000000008C2
S1138EC001006DF00FD101006F7000345E0087EC7C
S1118ED00B977A17000000185E006368547059
S113A2F88000000000000000CECB8F27F4200F3A27
S113A308A70C3C40A64E6C5286F0AC99B4E8DAFD33
S113A318DA01EE641A708DEAB01AE745B101E9E48F
S113A3288E41ADE9FBEBC27DE5D3EF282A242E81CC
S113A338B9A74A0637CE2EE195F83D0A1FB69CD930
S113A348F24A01A73CF2DCD0C3B8358109E84F07CC
S113A3589E19DB92B4E31BA99E74D1B791E07E48A2
S113A368C428D05AA4751E4CF2D56790AB41C2A439
S113A378964E858C91BA2655BA121A4650E4DDECEE
S113A388E65829B3046B0AFA8E938662882AF53F46
S113A398B080392CC4349DEDDA7F5BF5909668491B
S113A3A8873E4F75E2224E68A76C582338ED262363
S113A3B8CF42894A5DCE35EB8049A4AC0C5811AE27
S113A3C80000005900B3010D016601C0021A0273AF
S113A3D802CD0327038003DA0434FFA6FF4CFEF201
S109A3E8FE99FE3FFDE5B6
S111A3EEFD8CFD32FCD8FC7FFC25FBCBFB7203
S1138EDE5E00638A7A370000000C0F840F957A06C2
S1138EEE000000081AB30FD00FE15E00821C0AC007
S1138EFE0F820FD00FE15E00821C790000801A09E9

S1138F0E4B04119040F86EF8000511086EF8000B33
S1138F1E11086EF800041B75010069F50FD00FE1FF
S1138F2E5E00821C0AC00F85010069700FE15E00AE
S1138F3E821C790600801A094B04119640F80FA083
S1138F4E68086E790005169847280FA068066E7894
S1138F5E000B166846086E780004166847087A03F5
S1138F6E00000001400C685816E847067A0300001B
S1138F7E00017A2300000001463240140CE817501A
S1138F8E17106859168968D9100E46041B75FE0111
S1138F9E685816E847041FC544E21FC545086858BC
S1138FAE14E868D84006790000014002190007A17C8
S10D8FBE0000000C5E0063685470AD
S1138FC85E00638A1B971B970F82010069F17A037E
S1138FD80000000801006F7000200FB15E00821CC2
S1138FE80FA60A8601006F7000200FB15E00821C75
S1138FF8790400801A094B04119440F8686816C86C
S113900846500CCD1855400414D5110D0CDD46F807
S1139018686816584708686814C868E840340FA099
S1139028010069710A90010069F001006F70002066
S11390380FB15E00821C0FA50A85400C68684708BB
S1139048685814C868D8400A0B76010069701F86EF
S10F905845EA0B970B975E0063685470A9
S11390645E00638A0F857A040000000701006F70B5
S1139074002001006DF001006F70002001006DF00D
S11390845E0096080B970B970D0647487926000157
S1139094460A790004B86BA000FFE864792600024D
S11390A4460A7900044C6BA000FFE86419667A0051
S11390B40000001C0AF00D6117F10A90F9FF68899A
S11390C40B56792600084DE67A000000001C0AF0CE
S11390D45A00915A7A000000001C0AF07A01000039
S11390E4A3FC5E0062A20D00470C190069D07A004C
S11390F40000A3FC40607A060000001C0AF66963C2
S113910411131113111311131796307FF793303FE39
S113911469D36950791003FE462869500B5069D00E
S1139124696079608 00F69E00FE30B73400E0FC130
S11391340FB05E00965A69501B5069D0696073483A
S113914447EC69607960800F79403FE069E07A0019
S10991540000001C0AF0FC
S113915A01006F7100185E0063145E006368547047
S10BA3FC000000000000000056
S113916A5E00638A0F8601006F70002001006DF0B4
S113917A01006F70002001006DF05E0096080B97E6
S113918A0B970D05474079250001460A790004B873
S113919A6BA000FFE86479250002460A7900044CB3
S11391AA6BA000FFE86419667A000000001C0AF04D
S11391BA0D6117F10A90F9FF68890B5679260008A1
S11391CA4DE65A0092887A050000001C0AF5695593
S11391DA11151115111511115796507FF793503FF56
S11391EA0D5C4C0E7A000000A4040FE15E006314C8
S11391FA40607A000000001C0AF00FE15E0063146D
S113920A792C00344C4C0FE50B950BF57909003496
S113921A19C90D9017F07901001001D053100D09E7
S113922A400A0FD01BF5191169811B590D994EF28A
S113923A7900003419C017F07901001001D05310D6
S113924A7909FFFF1B584B04101940F869506690BF
S107925A69D07A0159
S113925E0000001C0AF10FE20F905E005DB60FE0F6

S113926E7A010000A4045E0062BE0D00470C7A0072
S113927E0000001C0AF07D0070707A000000001CD4
S113928E0AF001006F7100185E0063145E006368DC
S105929E547007
S10BA404000000000000000004D
S11392A001006DF20D1A4F14792A00084D04FA00DB
S11392B04008680A100A1B5A4EFA688A01006D7248
S10592C05470E5
S11392C201006DF20D1A4F14792A00084D04FA00B9
S11392D24008680A110A1B5A4EFA688A01006D7225
S10592E25470C3
S11392E401006DF119990D11471A4A06790908000D
S11392F417917919040F1B59101144FA10311031C5
S113930410311031010069811A9101006F81000449
S109931401006D715470AD
S113931A0F8501F064155860009A792407FF4714F2
S113932A0D44586000820FA501F064355860007837
S113933A580000800FA501F0643558600076406834
S113934A0FA501F06435466C0DCC465C0F8501F020
S113935A64154654405E0F8501F06415473C10318D
S113936A1230792800104C0C1B5C1031123079280A
S113937A00104DF4580000C40FA501F06435471AD4
S113938A10331232792A00104C0C1B541033123248
S113939A792A00104DF4580000A81AA21AB3687D5E
S11393AA100D131A58000228790107FF1AA21AB3DB
S11393BA580001FA790107FF1AA27A03000000088C
S11393CA68FA580001E801006DF601006DF5010025
S11393DA6DF401006DF301006DF201006DF10100FE
S11393EA6DF07A3700000018691D692565D569F59E
S11393FA691C111C111C111C111C796C07FF6924AF
S109940A111411141114EA
S11394101114796407FF01006D1001006911796867
S1139420000F01006F230004010069227 96A000F15
S1139430792C07FF5870FEE2792407FF5870FF0A62
S11394400DCC5870FF1A0D445870FF36094C793C07
S1139450003FF792C07FF58C0FF58792CFFCB58D056
S1139460FF426FFC000279480010794A00100100A6
S11394706FF000040F9601006FF2000801006FF314
S1139480000C0D1552356FF500160D34529452B180
S11394900A946511990009D44406791C00019900C6
S11394A06FF400140DC50D1D18990D0452340AC52F
S11394B099000DE452B40AC599000D6452240AC5FB
S11394C099006FF500120DD50D1D18990D845234B6
S11394D00AC50D0452B40AC599000DE452240AC505
S11394E099000D6452A40AC599006FF500100DD5BB
S11394F00D1D18990DB352830AB50D0452240AC5E4
S109950099000DE452A4E2
S11395060AC59900528209D24404791A0001091A3C
S11395166F74000852040AC26F7400085284094A21
S11395260D5B6F73001001006F7400126F7D0016E0
S1139536195 56F710002113213331334133 50DA00D
S11395467960010047121132133331334133 50B516B
S1139556792107FF5870FE5401F064454702700BEA
S11395660D114E2E113213334402700B0D114A04A2
S11395760B5140F0732B47180CB8E80B47127A13BC
S1139586000000008440A0B72792A00804D020B5131
S1139596402 0732B471C0CB8E80B47167A130000C0

S11395A6000844020B72792A01004D061132133367
S11395B60B5111321333113213331132133796AC8
S11395C6000F10111011110111011641A101A687877
S11395D61008131A7A170000001801006D700100B5
S11395E6698201006F83000401006D7101006D72D1
S10795F601006D738D
S11195FA01006D7401006D7501006D765470F3
S113960801006DF57A01000000080AF16818E87F87
S1139618A87F460A0B716818E8F0A8F047041900F8
S1139628402A6818F01050006898460A0B711AD53A
S1139638400E6818470679000001401 00B710B753E
S11396487A25000000064DEA7900000201006D75D5
S1059658547049
S113965A5E00638A0F840F931AD51B730FB64028D3
S113966A0FC30AE30FB26838E880175017700F83E5
S113967A0FA06809100968890FD547080FC00AE0C7
S113968A7D0070000FB51B760FE64CD40FD5470645
S113969A790100014002191117F10F905E00636806
S10596AA5470F7
S11396AC5E00638A7A37000000387A0500000004F4
S11396BC0AF50F840F967A000000001001006DF07C
S11396CC191101006F7000545E00991E0B970FE384
S11396DC0B930BF37A160000000701006FF60034AE
S11396EC790600030FC00B900BF001006FF00028FC
S11396FC7A140000000701006FF4002C5A0098281C
S113970C6838460C01006F7000346809587000FA11
S113971C7A000000001001006DF019110FD05E00EB
S113972C991E0B9701006F70002801006FF0003039
S113973C01006F74002C790E0003407C01006F70E4
S113974C00301A916809FA0810311A0A4EFA1A8075
S113975C684801F064101A916839FA0810311A0A32
S113976C4EFA01006F720034010069F01A80682808
S113977C01F06401010069705E00824201006FF028
S113978C00247A000000000401006DF07A0100004F
S109979C00280AF10DE2B2
S11397A2096217F210320AD20FA05E0098D60B9705
S11397B21B5E01006F7000301BF001006FF0003080
S11397C21BF40DEE4C807926000346247A00000038
S11397D2000A01006DF00D6417F40FC110310AD1B4
S11397E201006F7000540AC00AC05E0087EC402279
S11397F27A000000000A01006DF00D6417F40FC136
S113980210310AD101006F7000540AC00AC05E0011
S113981298D60B971B561BF301006F7000341BF095
S113982201006FF000340D6658C0FEDE7A170000A7
S10B983200385E006368547006
S113983A5E00638A7A370000000C0F860F940D606E
S113984A7910003F69C00FF10FE05E008ABE7A000B
S113985A0000000801006DF0191101006F70002863
S113986A5E00991E0B971A800F8219550FF6401046
S113987A0B760FA00B700F8269407930000869C01C
S113988A686847ECFB804004110B0B55686816B8EF
S113989A47F66DF57A03000000080FA01A830FB18B
S11398AA0FE05E008A460B876940195069C00100C0
S11398BA6DF30FE101006F7000285E0087EC0B97D0
S10F98CA7A170000000C5E006368547005
S11398D65E00638A0F860F9401006F7500180AD61F
S11398E61B760AD41B740FC319CC40226868175021

S11398F668391751091009C00D0468E817F479008F
S1139906010001D053040D4C1B751B761B730FD539
S10B991646DA5E006368547039
S113991E5E00638A1B870F8469F101006F73001A5F
S113992E0FC51AE6400C0FD00B756E7900016889CE
S113993E0B761FB645F00FC00B875E00636854703D
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_Controller, EV_Input, 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'0000126B	H'00000FBC	main	main
		H'0000126C	-	H'00001E8F	H'00000C24	EV_Simulator	EV_Simulator
		H'00001E90	-	H'0000280B	H'0000097C	EV_Controller	EV_Controller
		H'0000280C	-	H'00003149	H'0000093E	EV_Input	EV_Input
		H'0000314A	-	H'000039B1	H'00000868	EV_OpenClose	EV_OpenClose
		H'000039B2	-	H'000041F9	H'00000848	EV_UpDown	EV_UpDown

```
                H'000041FA   -   H'000045ED   H'000003F4
                                 EV_File                 EV_File
                H'000045EE   -   H'00004745   H'00000158
                                 EV_Time                 EV_Time
                H'00004746   -   H'00004C99   H'00000554
                                 Timer                   Timer
                H'00004C9A   -   H'00004DEF   H'00000156
                                 Panel                   Panel
                H'00004DF0   -   H'00004EC1   H'000000D2
                                 sci                     sci
                H'00004EC2   -   H'000050FF   H'0000023E
                                 lcd                     lcd
                H'00005100   -   H'00005BF1   H'00000AF2
                                 usb                     usb
                H'00005BF2   -   H'00005C2B   H'0000003A
                                 rand                    rand
                H'00005C2C   -   H'00005C89   H'0000005E
                                 sprintf                 sprintf
                H'00005C8A   -   H'00005CB3   H'0000002A
                                 strcmp                  strcmp
                H'00005CB4   -   H'00005CCF   H'0000001C
                                 strcpy                  strcpy
                H'00005CD0   -   H'00005CEB   H'0000001C
                                 strlen                  strlen



                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'00005CEC   -   H'00005D1B   H'00000030
                                 vsprintf                vsprintf
                H'00005D1C   -   H'00005FF5   H'000002DA
                                 addd3                   addd3
                H'00005FF6   -   H'00006227   H'00000232
                                 divd3                   divd3
                H'00006228   -   H'000062A1   H'0000007A
                                 dtol3                   dtol3
                H'000062A2   -   H'000062AD   H'0000000C
                                 eqd3                    eqd3
                H'000062AE   -   H'000062BD   H'00000010
                                 ged3                    ged3
                H'000062BE   -   H'000062CD   H'00000010
                                 ltd3                    ltd3
                H'000062CE   -   H'00006313   H'00000046
                                 ltod3                   ltod3
                H'00006314   -   H'00006331   H'0000001E
                                 mv83                    mv83
                H'00006332   -   H'00006359   H'00000028
```

```
                    mvn3                    mvn3
               H'0000635A   -   H'00006367   H'0000000E
                    ned3                    ned3
               H'00006368   -   H'00006389   H'00000022
                    spregld3                spregld3
               H'0000638A   -   H'000063B1   H'00000028
                    spregsv3                spregsv3
               H'000063B2   -   H'0000815F   H'00001DAE
                    _fmtout                 _fmtout
               H'00008160   -   H'0000821B   H'000000BC
                    cmpd3                   cmpd3
               H'0000821C   -   H'00008241   H'00000026
                    divl3                   divl3
               H'00008242   -   H'00008261   H'00000020
                    mull3                   mull3
               H'00008262   -   H'00008647   H'000003E6
                    _dti                    _dti
               H'00008648   -   H'000087EB   H'000001A4
                    _its                    _its
               H'000087EC   -   H'00008845   H'0000005A
                    memcpy                  memcpy
               H'00008846   -   H'00008881   H'0000003C
                    divul3                  divul3
               H'00008882   -   H'000088A9   H'00000028
                    _allzero                _allzero
               H'000088AA   -   H'000089A1   H'000000F8
                    _calcnpw                _calcnpw


               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'000089A2   -   H'00008A45   H'000000A4
                    _log10                  _log10
               H'00008A46   -   H'00008ABD   H'00000078
                    _lsfts                  _lsfts
               H'00008ABE   -   H'00008AEB   H'0000002E
                    _pow5                   _pow5
               H'00008AEC   -   H'00008B65   H'0000007A
                    _rsfts                  _rsfts
               H'00008B66   -   H'00008C11   H'000000AC
                    _sub                    _sub
               H'00008C12   -   H'00008CB5   H'000000A4
                    _unpack                 _unpack
               H'00008CB6   -   H'00008CF3   H'0000003E
                    memcmp                  memcmp
               H'00008CF4   -   H'00008D7B   H'00000088
                    _mult64                 _mult64
```

```
                H'00008D7C   -   H'00008EDD   H'00000162
                                 _power                  _power
                H'00008EDE   -   H'00008FC7   H'000000EA
                                 _rnd                    _rnd
                H'00008FC8   -   H'00009063   H'0000009C
                                 _setsbit                _setsbit
                H'00009064   -   H'00009169   H'00000106
                                 frexp                   frexp
                H'0000916A   -   H'0000929F   H'00000136
                                 modf                    modf
                H'000092A0   -   H'000092C1   H'00000022
                                 dslc3                   dslc3
                H'000092C2   -   H'000092E3   H'00000022
                                 dsruc3                  dsruc3
                H'000092E4   -   H'00009319   H'00000036
                                 itod3                   itod3
                H'0000931A   -   H'00009607   H'000002EE
                                 muld3                   muld3
                H'00009608   -   H'00009659   H'00000052
                                 _duchek                 _duchek
                H'0000965A   -   H'000096AB   H'00000052
                                 _lsft                   _lsft
                H'000096AC   -   H'00009839   H'0000018E
                                 _mult                   _mult
                H'0000983A   -   H'000098D5   H'0000009C
                                 _pow10                  _pow10
                H'000098D6   -   H'0000991D   H'00000048
                                 _add                    _add
                H'0000991E   -   H'0000994D   H'00000030
                                 memset                  memset


                H8/300H LINKAGE EDITOR (Evaluation software) Ver.1.0   PAGE :    4

                   ***    LINKAGE EDITOR LINK MAP LIST     ***


SECTION   NAME              START    -    END      LENGTH
                                UNIT NAME                MODULE NAME

* TOTAL ADDRESS *           H'00000200   -   H'0000994D   H'0000974E


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'00009DE8  H'0000015B
                    EV_Simulator            EV_Simulator
          H'00009DEA  -  H'00009E11  H'00000028
                    EV_Controller           EV_Controller
          H'00009E12  -  H'00009E77  H'00000066
                    EV_Input                EV_Input
          H'00009E78  -  H'00009EC3  H'0000004C
                    EV_OpenClose            EV_OpenClose
          H'00009EC4  -  H'00009F06  H'00000043
                    EV_UpDown               EV_UpDown
          H'00009F08  -  H'00009F79  H'00000072
                    EV_File                 EV_File
          H'00009F7A  -  H'00009F81  H'00000008
                    EV_Time                 EV_Time
          H'00009F82  -  H'00009FB9  H'00000038
                    Timer                   Timer
          H'00009FBA  -  H'00009FC4  H'0000000B
                    Panel                   Panel
          H'00009FC6  -  H'0000A07F  H'000000BA
                    usb                     usb
          H'0000A080  -  H'0000A087  H'00000008
                    _fmtout                 _fmtout
          H'0000A088  -  H'0000A187  H'00000100
                    _ctype                  _ctype
          H'0000A188  -  H'0000A20F  H'00000088
                    _its                    _its
          H'0000A210  -  H'0000A217  H'00000008
                    _log10                  _log10
          H'0000A218  -  H'0000A2F7  H'000000E0
                    _pow5                   _pow5
          H'0000A2F8  -  H'0000A3FB  H'00000104
                    _power                  _power


          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'0000A3FC  -  H'0000A403  H'00000008
                    frexp                   frexp
          H'0000A404  -  H'0000A40B  H'00000008
                    modf                    modf

* TOTAL ADDRESS *          H'00009A00  -  H'0000A40B  H'00000A0C


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'00FFE207   H'000001F6
                                    main                    main
                    H'00FFE208   -   H'00FFE21B   H'00000014
                                    EV_File                 EV_File
                    H'00FFE21C   -   H'00FFE43F   H'00000224
                                    Timer                   Timer
                    H'00FFE440   -   H'00FFE4BF   H'00000080
                                    Panel                   Panel
                    H'00FFE4C0   -   H'00FFE50F   H'00000050
                                    sci                     sci
                    H'00FFE510   -   H'00FFE54F   H'00000040
                                    lcd                     lcd
                    H'00FFE550   -   H'00FFE823   H'000002D4
                                    usb                     usb
                    H'00FFE824   -   H'00FFE85F   H'0000003C
                                    _fmtout                 _fmtout
                    H'00FFE860   -   H'00FFE863   H'00000004
                                    _rnext                  _rnext
                    H'00FFE864   -   H'00FFE865   H'00000002
                                    _errno                  _errno

* TOTAL ADDRESS *           H'00FFE010   -   H'00FFE865   H'00000856


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

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

                    SYMBOL  NAME                ADDR            TYPE

                    $ADDD$3                     H'00005DE6      DAT
                    $CMPD$3                     H'00008160      DAT
                    $DIVD$3                     H'00006090      DAT
                    $DIVL$3                     H'0000821C      DAT
                    $DIVUL$3                    H'00008846      DAT
                    $DSLC$3                     H'000092A0      DAT
                    $DSRUC$3                    H'000092C2      DAT
                    $DTOL$3                     H'00006228      DAT
                    $EQD$3                      H'000062A2      DAT
                    $GED$3                      H'000062AE      DAT
                    $ITOD$3                     H'000092E4      DAT
                    $LTD$3                      H'000062BE      DAT
                    $LTOD$3                     H'000062CE      DAT
                    $MULD$3                     H'000093D0      DAT
```

$MULL$3 H'00008242 DAT
$MV8$3 H'00006314 DAT
MVN3 H'00006332 DAT
NED3 H'0000635A DAT
$SUBD$3 H'00005DB6 DAT
sp_regld3 H'00006368 DAT
sp_regsv3 H'0000638A DAT
_Checkfmove H'00004708 ENT
_Clear H'00004C9A ENT
_ClearLCD H'00004FE0 ENT
_Close H'0000396A ENT
_CloseMotor H'0000341E ENT
_Cnt H'00FFE012 DAT
_Command_Read H'00004404 ENT
_Command_Write H'00004450 ENT
_Destroy H'000010D6 ENT
_DisableInterrupt H'000002AC DAT
_Disp H'0000177A ENT
_DispInput H'0000172A ENT
_DispUSBPort H'00005210 ENT
_Door H'0000314A ENT
_Down H'000041B2 ENT
_DownMotor H'00003C86 ENT
_EV_Controller H'00001E90 ENT
_EV_File H'0000424E ENT
_EV_Input H'00002878 ENT
_EV_Simulator H'0000126C ENT
_EV_Time H'000045EE ENT
_EnableInterrupt H'000002A8 DAT
_GetChar H'0000280C ENT
_GetCurrentTime H'00004650 ENT
_GetPermit H'000046EA ENT
_GetSCI H'00004E20 ENT
_GetSW H'0000121A ENT
_H8init H'00001240 ENT
_Init H'00000E68 ENT
_InitITU H'00004C1E ENT

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

SYMBOL NAME ADDR TYPE

_InitLCD H'00004F70 ENT
_InitSCI H'00004DF0 ENT
_InitUSB H'00005116 ENT
_InterruptITU0 H'00004C54 ENT
_LCDOut4 H'00004F18 ENT
_Limit_Read H'000045AA ENT
_LocateLCD H'00005020 ENT
_Motor_Read H'0000455C ENT
_Motor_Write H'00004524 ENT
_OnCloseMotor H'0000343A ENT
_OnController H'00002030 ENT

```
_OnDownMotor                H'00003CA2      ENT
_OnInitWaitDoorChangeLog    H'000036A6      ENT
_OnInitWaitPositionChangeLog H'00003F00     ENT
_OnInput                    H'000029A4      ENT
_OnOpenMotor                H'000031DA      ENT
_OnSimulator                H'00001356      ENT
_OnUpMotor                  H'00003A42      ENT
_OnWaitCloseDoorChangeLog   H'0000384E      ENT
_OnWaitDownPositionChangeLog H'0000409C     ENT
_OnWaitOpenDoorChangeLog    H'0000377A      ENT
_OnWaitUpPositionChangeLog  H'00003FCE      ENT
_Open                       H'00003922      ENT
_OpenMotor                  H'000031BE      ENT
_PermitCommand_Read         H'00004374      ENT
_PermitCommand_Write        H'000043C0      ENT
_PermitTurnOpen_Read        H'00004494      ENT
_PermitTurnOpen_Write       H'000044E0      ENT
_Position                   H'000039B2      ENT
_PrintF                     H'00004CC2      ENT
_PrintLCD                   H'00005044      ENT
_PrintSCI                   H'00004E40      ENT
_PutLCD                     H'00004FF4      ENT
_PutSCI                     H'00004E10      ENT
_Read                       H'000042E2      ENT
_ReadString                 H'0000433C      ENT
_Repaint                    H'000004E4      ENT
_Run                        H'00000550      ENT
_ScanSCI                    H'00004E30      ENT
_SetCurrentTime             H'00004632      ENT
_SetLED                     H'000011CC      ENT
_SetPermit                  H'000046B6      ENT
_SleepMSec                  H'00004758      ENT
_Start                      H'00004A0A      ENT
_Stop                       H'00004A38      ENT
_Thread_Start               H'00004AD0      ENT
_Thread_Toggle              H'00004B0E      ENT
_Thread_checkAllDelete      H'00004A56      ENT
_Thread_checkStayAnother    H'00004A70      ENT
_Thread_getThread           H'00004A96      ENT
_Up                         H'0000416A      ENT
```


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

```
SYMBOL  NAME                ADDR            TYPE

_UpMotor                    H'00003A26      ENT
_WaitDoorChangeLog          H'0000367E      ENT
_WaitPositionChangeLog      H'00003EE6      ENT
_WaitSecond                 H'00004694      ENT
_Wait_ms                    H'0000472A      ENT
_Write                      H'00004258      ENT
_WriteString                H'000042AA      ENT
__add                       H'000098D6      ENT
```

__allzero	H'00008882	ENT
__calcnpw	H'000088AA	ENT
__ctype	H'0000A088	DAT
__dti	H'00008262	ENT
__duchek	H'00009608	ENT
__errno	H'00FFE864	DAT
__fmtout	H'000063B2	ENT
__its	H'00008648	ENT
__log10	H'000089A2	ENT
__lsft	H'0000965A	ENT
__lsfts	H'00008A46	ENT
__mult	H'000096AC	ENT
__mult64	H'00008CF4	ENT
__pow10	H'0000983A	ENT
__pow5	H'00008ABE	ENT
__power	H'00008D7C	ENT
__rnd	H'00008EDE	ENT
__rnext	H'00FFE860	DAT
__rsfts	H'00008AEC	ENT
__setsbit	H'00008FC8	ENT
__sub	H'00008B66	ENT
__unpack	H'00008C12	ENT
_cntrl	H'00FFE088	DAT
_countUpNextRun	H'0000482E	ENT
delete	H'000049D6	ENT
_frexp	H'00009064	ENT
_getClock	H'00004746	ENT
_get_inbufflen	H'00005BCA	ENT
_get_outbufflen	H'00005BDE	ENT
_i_cnt	H'00FFE016	DAT
_in	H'00FFE04A	DAT
_initWOVI	H'00004C80	ENT
_init_usbbuff	H'00005A2E	ENT
_j_cnt	H'00FFE018	DAT
_main	H'000002B0	ENT
_memcmp	H'00008CB6	ENT
_memcpy	H'000087EC	ENT
_memset	H'0000991E	ENT
_modf	H'0000916A	ENT
_new_EV_Status	H'000041FA	ENT
_new_Thread	H'0000484E	ENT
_nextRun	H'000047EA	ENT
_rand	H'00005BF2	ENT

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

SYMBOL NAME	ADDR	TYPE
_read_buff	H'00005B6C	ENT
_read_outbuff	H'00005B0E	ENT
_s	H'00FFE046	DAT
_simu	H'00FFE180	DAT
_sprintf	H'00005C2C	ENT

_status	H'00FFE208	DAT
_strcmp	H'00005C8A	ENT
_strcpy	H'00005CB4	ENT
_strlen	H'00005CD0	ENT
_th	H'00FFE01A	DAT
_th1	H'00FFE022	DAT
_th101	H'00FFE260	DAT
_th102	H'00FFE27E	DAT
_th111	H'00FFE29C	DAT
_th112	H'00FFE2BA	DAT
_th113	H'00FFE2D8	DAT
_th114	H'00FFE2F6	DAT
_th119	H'00FFE314	DAT
_th120	H'00FFE332	DAT
_th121	H'00FFE350	DAT
_th122	H'00FFE36E	DAT
_th123	H'00FFE38C	DAT
_th130	H'00FFE3AA	DAT
_th131	H'00FFE3C8	DAT
_th141	H'00FFE3E6	DAT
_th142	H'00FFE404	DAT
_th143	H'00FFE422	DAT
_th19	H'00FFE032	DAT
_th20	H'00FFE036	DAT
_th41	H'00FFE03A	DAT
_th42	H'00FFE03E	DAT
_th43	H'00FFE042	DAT
_usb_int	H'00005274	ENT
_vsprintf	H'00005CEC	ENT
_wovi	H'00004C7C	ENT
_woviClock	H'00FFE21C	DAT
_woviInit	H'00004BEC	ENT
_woviRun	H'00004B52	ENT
_woviThreadFirst	H'00FFE224	DAT
_woviThreadLast	H'00FFE242	DAT
_write_buff	H'00005AA8	ENT
_write_inbuff	H'00005A4C	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 393: 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_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_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 291: '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_Input.obj EV_Controller.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_Controller, EV_Input, 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