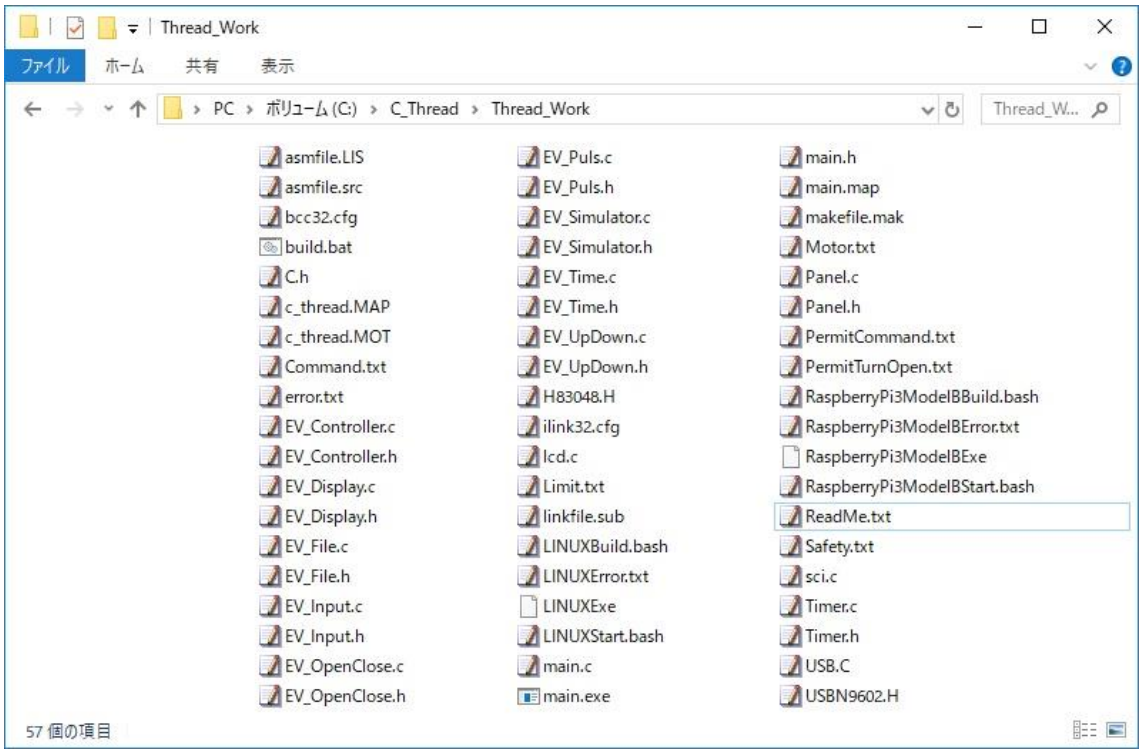
発明の巻

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

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

セスが 1 個しか入らないでしょうから。C , Assembly , Batch Shell Script 使用。2018 年 4 月 22 日版プログラムです。初期デバッグ対応します。日本語で、日本国内で、初期サポート対応可能です。改造は自己責任でお願いします。324,000 円の中身は初期サポート代です。サンプルコードは開発終了品です。購入後、ご自由に、使用・改造・配布、O.K. です。

お問い合わせ先： info@hidemine.ciao.jp



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

タイトル(title):

C 言語の疑似スレッド

サブタイトル(subtitle):

C 言語の偽物のスレッド

The thread at the imitation of the C language.

似ているが独創的な別物

The resembling but original singleton.

C 言語によるスレッドという概念の模倣

Copying a concept,

the thread, by the C language.

解説

C言語のプロジェクト Thread について、

このプロジェクトは予告なくデバッグ目的で更新されることがあります。

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

==========

このプロジェクトは、3通りの方法で コンパイル 実行 できます。

1つ目の方法：
Windows版 Borland BCC C/C++ のダウンロードとインストール
bcc コンパイラー で検索します
C++Builder のホームページを開きます
C++ Compiler 5.5 / Turbo Debugger (日本語) (コンパイラのみで軽い)
(テキストエディタと組み合わせて使用します)(フリーソフト)
(ユーザー登録が必要)(メールアドレスが必要)をダウンロードして
解凍します
freecommandlinetools2.exeをインストールします
freeturbodebugger.exeをインストールします

2つ目の方法：
Windows版 AKI-H8 3052F USB開発セット の購入
メーカー： ルネサスエレクトロニクスさん
販売者： 秋月電子通商さん
通販コード： K-00182

商品名： AKI-H8 3052F USB開発セット

商品価格： 税込7,000円 (2018年3月6日現在)

AKI-H8 3052F USB開発セット で検索します

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

AC/DCコンバータ(ACアダプター) と、 RS-232Cケーブル と、延長USBコード を、ご自宅の最寄りの電子パーツ専門店 で注文購入します

AC/DCコンバータ の 電流電圧 は お店の人 に聞いてください

RS-232C は、パソコンに端子がある必要があり、また、パソコンと AKI-H8基板 の両方の コネクタ の オス端子・メス端子 を確認して購入してください

延長USBコード も オス端子・メス端子 を確認してください

カラー短絡ソケット6mm 青 2228CG-BU で検索します

カラー短絡ソケット6mm 青 2228CG-BU のような 短絡コネクタ を、ご自宅の最寄りの電子パーツ専門店 で 2個 注文購入します

(100円 か 200円 支払うと何個かまとめて購入できます)

USB開発セット のマニュアルに従い、 usbフォルダ の main.c を コンパイルして、 H8WriteTurbo をインストールして、 AKI-H8基板 への書き込み時に、 短絡ソケット(短絡コネクタ) を使用して、マニュアルに従い モード を調節して、 usbフォルダ の usbtest.MOT を AKI-H8基板 へ書き込み、走らせてみます

押しボタンを押すと、sw1 sw2 sw3 sw4 などと 液晶パネル に表示されれば大丈夫です。

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

C_Threadフォルダ の中の、 Thread_Workフォルダ の中の、 c_thread.MOT を、 モード を調節しながら、 AKI-H8基板 へ書き込み、走らせてみます

液晶パネル ・ 押しボタン ・ LED などが機能していれば、成功です

3つ目の方法：

LINUX Raspbian GCC の利用

私は、 Rasberry Pi 3 Model B (ラズベリーパイ) 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

==========

main.c の 関数 Run の ID==31 を見てください。

Thread Ready GO! で開始して競馬のコースが8コースあります。

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

ゴールまで14歩です。

There are 14 cells to a GOAL.

<1>コースは'r'ボタンが鞭で<2>コースは'l'ボタンが鞭です。

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

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

スレッドを使用しています。

Thread *th[8]; でオブジェクト宣言しています。

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

この2行は Java で次と同じ意味です。

Thread th[] = new Thread[8];

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

void Repaint(void)

```
{
	…
}
void Run(Thread *This)
{
	…
}
void Init(Thread *This)
{
	…
}
void Destroy(Thread *This)
{
	…
}
```

はそれぞれ Java で次と同じ意味です。

```
public void paint(Graphics g)
{
	…
}
public void run()
{
	…
}
public void init()
{
	…
}
public void destroy()
```

{

　　…

}

delete_(th[i]); でオブジェクトを消去しています。

この1行は C++ で次と同じ意味です。

delete th[i];

スレッド20が走り始めたら、

0以外の数字キーを押してみてください。

その数字に20を加えた番号のスレッドが、キーを押す度毎に、

起動・消去を繰り返します。

20を含めて、全部スレッドが消去されると、終了です。

これらのスレッドに関する仕様は Timer.c に記述しました。

==========

2階建エレベーターEVについて

使用方法

EV_Simulator にエレベーターが表示されます

EV_Controller にエレベーターの動作が表示されます

EV_Input の使用方法

u キーを押すとエレベーターが2階に上昇して扉が開きます

d キーを押すとエレベーターが1階に下降して扉が開きます

o キーを押すと扉が開きます

c キーを押すと扉が閉じます

s キーを押すと籠が非常停止します

r キーを押すと籠が非常停止から復帰します

Y キーを押すとエレベーターが2階に上昇して扉が開きます

H キーを押すと2階で扉が閉じます

y キーを押すとエレベーターが1階に下降して扉が開きます

h キーを押すと1階で扉が閉じます

籠が無い階で H h キーを押しても籠は動作しません

籠が無い階で Y y キーを押したとき籠の扉が開いていると、

籠は動作しません

開いた状態の扉は一定時間後自動で閉じます

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

閉まりかけの時に開く動作をするキーを押すと扉が反転して開きます

動作説明

全体の動作説明

モーターの情報は Motor.txt にあります

エレベーター塔内のリミットスイッチの情報は Limit.txt にあります

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

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

エレベーターの画面表示もしています

EV_Controller はエレベータを制御していて Command.txt Limit.txt
を採取して PermitCommand.txt Motor.txt に書き込んでいます

EV_Controllerの動作説明

エレベーターには現在位置情報(Limit.txt)があります
最簡形の2階建ての場合通常系には5個の位置状態があります
下の階の停止状態
下の階の低速区域
中間の高速区域
上の階の低速区域
上の階の停止状態
5個の区域の境界に合計4個のセンサーがあります
4個のセンサーがエレベーターの現在位置を取得しています
4個のセンサーからの信号はメンバ変数(Positionクラスの
*p_UnderSlow *p_UnderStop *p_UpperSlow *p_UpperStop)
に読み込みます
昇りのメソッド(UpMotorクラスのOnUpMotor)と降りのメソッド
(DownMotorクラスのOnDownMotor)で使って
モーターに出力(Motor.txtに出力)します

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

後はインスタンス(Position P UpMotor UPMT DownMotor DNMT
Door DR OpenMotor OPMT CloseMotor CLMT)
を宣言して仕様に合わせてメソッドを呼び出すだけでO.K.です

終了方法

エレベーターが通常停止しているときに q キーを押します

メンテナンス

異常終了した場合、終了後、Thread_Work フォルダの次のファイルを

チェックしてください

Safety.txt

Safety.txt を開いて r にして上書き保存してください

r は通常動作を意味します

s は非常停止を意味します

h は復帰を意味します

Y はスターデルタのスター起動を意味します

Command.txt

Command.txt を開いて q にして上書き保存してください

q は終了を意味します

u は上昇を意味します

d は下降を意味します

o は開を意味します

c は閉を意味します

Y は上階呼びを意味します

y は下階呼びを意味します

H は上階閉を意味します

h は下階閉を意味します

N は信号無しを意味します

PermitCommand.txt

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

N は命令入力禁止を意味します

c は命令入力許可を意味します

PermitTurnOpen.txt

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

N は反転開信号入力禁止を意味します

o は反転開信号入力許可を意味します

Motor.txt

Motor.txt を開いて s にして上書き保存してください

s はモーター停止を意味します

j はモーター上昇回転開始を意味します

u はモーター低速上昇回転を意味します

U はモーター高速上昇回転を意味します

k はモーター下降回転開始を意味します

d はモーター低速下降回転を意味します

D はモーター高速下降回転を意味します

h はモーター開回転開始を意味します

o はモーター低速開回転を意味します

O はモーター高速開回転を意味します

t はモーター閉回転開始を意味します

c はモーター低速閉回転を意味します

C はモーター高速閉回転を意味します

Limit.txt

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

yynnyynn は籠が下階停止状態で扉が閉停止状態を意味します

yynnnynn は籠が下階停止状態で扉が閉低速区域を意味します

yynnnnnn は籠が下階停止状態で扉が中間高速区域を意味します

yynnnnyn は籠が下階停止状態で扉が開低速区域を意味します

yynnnnyy は籠が下階停止状態で扉が開停止状態を意味します

nynnyynn は籠が下階低速区域で扉が閉停止状態を意味します

nnnnyynn は籠が中間高速区域で扉が閉停止状態を意味します

nnynyynn は籠が上階低速区域で扉が閉停止状態を意味します

nnyyyynn は籠が上階停止状態で扉が閉停止状態を意味します

nnyynynn は籠が上階停止状態で扉が閉低速区域を意味します

nnyynnnn は籠が上階停止状態で扉が中間高速区域を意味します

nnyynnyn は籠が上階停止状態で扉が開低速区域を意味します

nnyynnyy は籠が上階停止状態で扉が開停止状態を意味します

==========

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

```
/* Panel.h */

#ifndef USE_BCC
/*=============================================================
                        外部参照
=============================================================*/
/* asmfile.src 内に定義 */
extern  void EnableInterrupt(void);
extern  void DisableInterrupt(void);
/* lcd.c */
extern  void InitLCD(void);
extern  void PrintLCD(char *fmt,...);
extern  void PutLCD(char c);
/* usb.c */
extern  void InitUSB(void);
extern  void DispUSBPort(void);
/* バッファ処理 */
extern  int get_inbufflen(void);
extern  int write_buff(char *p,int size);
extern  int read_buff(char *p,int size);
/* sci.c */
extern  void InitSCI(void);
extern  void PrintSCI(char *fmt, ...);
extern  int ScanSCI(void);
extern  char GetSCI(void);
/* main.c内定義 */
void H8init(void);
int SetLED(int no,int onoff);
```

```
int GetSW(int no);
#endif
/* 表示を表す列挙体宣言 */
enum PrintF
{
 Panel,
 ClsPnl,
 InputCommand,
 Monitor
};
/* 画面クリア */
void Clear(void);
/* 表示を表す関数のプロトタイプ宣言 */
void PrintF(int mode, char *str);
#ifdef USE_LINUX
int kbhit(void);
#endif
```

```
/* Panel.c */

#include "C.h"
#include "Panel.h"

/* 画面クリア */
void Clear(void)
{
#ifndef USE_BCC
 static char buff[64];
 sprintf(buff,"¥f");
 PrintLCD(buff);
#else
 CLEAR;
#endif
}

/* 表示のための関数 */
void PrintF(int mode, char *str)
{
#ifndef USE_BCC
 static char buff[64];
#endif
 switch(mode)
 {
 case ClsPnl:
#ifndef USE_BCC
  Clear();
```

```
  if(strcmp(str, "¥n¥0") == 0)
  {
      sprintf(str, "%s", "¥n¥f");
  }
  sprintf(buff,"%s", str);
  PrintSCI("%s",buff);
  PrintLCD(str);
#endif
  break;
 case Panel:
#ifndef USE_BCC
  if(strcmp(str, "¥n¥0") == 0)
  {
      sprintf(str, "%s", "¥n¥f");
  }
  sprintf(buff,"%s", str);
  PrintSCI("%s",buff);
  PrintLCD(str);
#else
  printf("%s", str);
#endif
  break;
 case InputCommand:
#ifdef USE_BCC
  printf("%s", str);
#endif
  break;
 case Monitor:
#ifndef USE_BCC
```

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

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

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

    ch = getchar();
```

```
    tcsetattr(STDIN_FILENO, TCSANOW, &oldt);
    fcntl(STDIN_FILENO, F_SETFL, oldf);

    if (ch != EOF) {
        ungetc(ch, stdin);
        return 1;
    }

    return 0;
}
#endif
```

```
/* Timer.h */

#ifndef Panel_h
#define Panel_h
#include "Panel.h"
#endif

/* 疑似スレッド定義 */
#ifdef USE_THREAD
/* 疑似スレッドに使用する定数の宣言 */
#define INITCLOCKNO -1
#define STOPCLOCKNO -2

/* 構造体宣言 */
typedef struct tag_Thread
{
	/* 疑似スレッドID */
	int ID;
	/* 指定開始時 */
	double preClock;
	/* woviClockがpreClockからsetClock秒増えたらRunを呼ぶ */
	double setClock;
	/* Runが呼ばれた回数を調べるために使用(countUpNextRunが呼ばれた回数) */
	long count;
	/* List機能 */
	struct tag_Thread *previous;
	struct tag_Thread *next;
}Thread;
```

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

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

```
/* Timer.c */

#include "C.h"
#include "Timer.h"

/* 時間を表す外部変数宣言 */
double woviClock;

/* 疑似スレッド定義 */
#ifdef USE_THREAD
/* wovi用疑似インスタンス宣言 */
Thread woviThreadFirst;
Thread woviThreadLast;
#endif

/* 時刻取得 */
double getClock(void)
{
	return woviClock;
}

/* ミリ秒待ち関数 */
void SleepMSec(long ms)
{
#ifndef USE_BCC
	double start;
	double set;
	double end;
```

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

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

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

#ifndef USE_BCC
Thread th101;
Thread th102;
Thread th111;
Thread th112;
Thread th113;
Thread th114;
Thread th119;
Thread th120;
Thread th121;
Thread th122;
Thread th123;
Thread th130;
Thread th131;
Thread th141;
Thread th142;
Thread th143;
Thread th144;
#endif

/* スレッドのコンストラクタの代用 */
Thread *new_Thread(int id)
```

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

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

```
    else if(id == 31)
    {
        new_List = &th131;
    }
    else if(id == 41)
    {
        new_List = &th141;
    }
    else if(id == 42)
    {
        new_List = &th142;
    }
    else if(id == 43)
    {
        new_List = &th143;
    }
    else if(id == 44)
    {
        new_List = &th144;
    }
#endif
#ifdef USE_BCC
    new_List = (Thread *)calloc(1, sizeof(Thread));
#endif
    if(new_List == NULL)
    {
        PrintF(Panel, "¥n");
        PrintF(Panel, "calloc failed");
        return NULL;
```

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

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

/* スレッドのvoid start(void)の代用 */
void Start(Thread *This)
{
```

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

/* スレッドのvoid stop(void)の代用 */
void Stop(Thread *This)
{
	This->preClock = STOPCLOCKNO;
	return;
}

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

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

```
    {
        checkthread = checkthread->next;
        i = i + 1;
    }
    return i;
}

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

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

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

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

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

void Thread_Toggle(int id)
{
    Thread *th;

    th = Thread_getThread(id);
    if(th == NULL)
    {
```

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

    /* タイマ関数 */
    /* Runを呼ぶタイミング */
    void woviRun(void)
    {
        Thread *List;
        Thread *next_List;
        List = &woviThreadFirst;
        List = List->next;
        while(List->next != NULL)
        {
            next_List = List->next;
            if((List->preClock != INITCLOCKNO) && (List->preClock != STOPCLOCKNO))
            {
                if(woviClock >= List->preClock + List->setClock)
                {
```

```
                    List->preClock = woviClock;
                    /* スレッドのvoid run(void)の代用 */
                    Run(List);
                }
            }
            List = next_List;
        }
        return;
}

/* タイマ初期化関数 */
/* 関数main の冒頭で、 */
/* スレッド構造体リスト の 両端 を初期化します。 */
void woviInit(void)
{
        woviThreadFirst.previous = NULL;
        woviThreadFirst.next = &woviThreadLast;
        woviThreadLast.previous = &woviThreadFirst;
        woviThreadLast.next = NULL;
        return;
}

#ifndef USE_BCC
/* タイマ割り込み用 */
/* 関数main の冒頭で、 */
/* タイマ割り込み の専用設定をして、 */
/* タイマ0割り込み を立ち上げます。 */
void InitITU(void)
{
```

```
    ITU.TSTR = 0x01; /* timer 0 enable */
    ITU.TSNC = 0;
    ITU.TMDR = 0x0;
    ITU.TFCR = 0x0;
    ITU.TOER = 0x0;
    ITU.TOCR = 0xff;
    ITU0.TCR = 0x00; /* 分周なし */
    ITU0.TIOR = 0x88;
    ITU0.TIER = 0x04; /* オーバーフロー割り込み許可 */
    /* AKI-H8 3052F USB の演算速度は 25MHz なので、 */
    /* 25Kカウント すると、 1ms です。 */
    ITU0.TCNT = 0xffff - 25000; /* 1 msec interval */
    ITU0.GRA = 0;
    ITU0.GRB = 0;
}

/* タイマ割り込み用 */
/* 周りの関数が 関数main から呼び出されているのに、 */
/* この関数だけは、 asmfile.src の タイマ0割り込み から */
/* 直接呼び出されています。 */
void InterruptITU0(void)
{
    ITU0.TSR &= 0xfb;
    /* AKI-H8 3052F USB の演算速度は 25MHz なので、 */
    /* 25Kカウント すると、 1ms です。 */
    ITU0.TCNT = 0xffff - 25000; /* 1 msec interval */
    /* woviClock はシステムリセット時からの秒数時計です。 */
    woviClock += 0.001;
```

```
	return;
}
#endif

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

/* タイマ初期化関数 */
void initWOVI(void)
{
```

```
	/* 関数main の冒頭で、 */
	/* 秒数時計woviClock を */
	/* 0.0秒 で初期化します。 */
	woviClock = 0.0;
	/* タイマ割り込み用 */
#ifndef USE_BCC
	/* タイマ0割り込み を立ち上げます。 */
	InitITU();
	/* asmfile.src の 割り込み許可ラベル を呼びます。 */
	EnableInterrupt();
#endif
	/* スレッド構造体リスト の 両端 を初期化します。 */
	wovilnit();
	return;
}
#endif

#ifdef USE_BCC
/* 現在日時表示 */
void PrintCurrentTime(void)
{
	time_t timer;
	struct tm *t_st;

	/* 現在時刻の取得 */
	time(&timer);

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

```
printf("%d",t_st->tm_year+1900);
if(t_st->tm_mon+1 < 10)
{
	printf("0%d",t_st->tm_mon+1);
}
else
{
	printf("%d",t_st->tm_mon+1);
}
if(t_st->tm_mday < 10)
{
	printf("0%d",t_st->tm_mday);
}
else
{
	printf("%d",t_st->tm_mday);
}
printf(" ");
if(t_st->tm_hour < 10)
{
	printf("0%d",t_st->tm_hour);
}
else
{
	printf("%d",t_st->tm_hour);
}
if(t_st->tm_min < 10)
{
```

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

        return;
    }
    #endif
```

```
/* EV_Time.h */

#ifndef Panel_h
#define Panel_h
#include "Panel.h"
#endif

#ifndef Timer_h
#define Timer_h
#include "Timer.h"
#endif

#define OPENTIMEOUT 10

/*==============================================================
 時間を表す構造体
==============================================================*/
struct EV_Time{
	double TimeTemp;
	double *p_TimeTemp;
	int Permit;
	int *p_Permit;
	int tmpTimeSafety;
	Thread *th;
};

/*==============================================================
 時間を表すプロトタイプ宣言
```

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

```
/* EV_Time.c */

#include "C.h"
#include "EV_Time.h"

/*==========================================================================
  時間を表す関数
==========================================================================*/
void EV_Time(struct EV_Time *This, Thread *th)
{
	This->TimeTemp = 0;
	This->p_TimeTemp = &This->TimeTemp;
	SetCurrentTime(This);
	This->p_Permit = &This->Permit;
	This->Permit = OFF;
	This->tmpTimeSafety = 0;
	This->th = th;

	/* 戻る */
	return;
}

void SetCurrentTime(struct EV_Time *This)
{
	This->p_TimeTemp = &This->TimeTemp;
	*This->p_TimeTemp = getClock();

	/* 戻る */
```

```
	return;
}

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

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

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

	/* 戻る */
	return;
}

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

```
	return This->Permit;
}

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

	/* 戻る */
	return;
}

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

	nextRun(This->th, num);
	/* 戻る */
	return;
}
```

```
/* EV_File.h */

#ifndef Panel_h
#define Panel_h
#include "Panel.h"
#endif

/*============================================================
  ファイルを表す構造体
============================================================*/
#ifdef NOTUSE_FILES
typedef struct tag_Handle_EV_Status
{
	char safety;
	char *p_limit;
	char limit[9];
	char motor;
	char command;
	char permitcommand;
	char permitturnopen;
}Handle_EV_Status;

typedef struct tag_EV_Status
{
	Handle_EV_Status *p_status;
}EV_Status;
#endif
```

```
/* ファイルストリーム */
struct EV_File
{
	FILE *fp;
};

/*==========================================================
  ファイルを表すプロトタイプ宣言
==========================================================*/
#ifdef NOTUSE_FILES
void new_EV_Status(EV_Status *This);
#endif

void EV_File(struct EV_File *This);
int Write(struct EV_File *This, char *filename, char ch);
int WriteString(struct EV_File *This, char *filename, char *str);
int Read(struct EV_File *This, char *filename, char *p_ch);
int ReadString(struct EV_File *This, char *filename, char *str, int strlength);
int PermitCommand_Read(struct EV_File *This, char *p_PermitCommand);
int PermitCommand_Write(struct EV_File *This, char PermitCommand);
int Command_Read(struct EV_File *This, char *p_Command);
int Command_Write(struct EV_File *This, char Command);
int PermitTurnOpen_Read(struct EV_File *This, char *p_PermitTurnOpen);
int PermitTurnOpen_Write(struct EV_File *This, char PermitTurnOpen);
void Motor_Write(struct EV_File *This, char Motor);
char Motor_Read(struct EV_File *This);
void Limit_Read(struct EV_File *This, char *str);
```

```
/* EV_File.c */

#include "C.h"
#include "EV_File.h"

/*==========================================================================
  ファイル不使用時大域オブジェクト宣言
==========================================================================*/
#ifdef NOTUSE_FILES
Handle_EV_Status status;
#endif

/*==========================================================================
  ファイルを表す関数
==========================================================================*/
#ifdef NOTUSE_FILES
void new_EV_Status(EV_Status *This)
{
	This->p_status = &status;
	status.safety = 'r';
	status.p_limit = &status.limit[0];
	strcpy(status.p_limit, "yynnyynn¥0");
	status.motor = 's';
	status.command = 'N';
	status.permitcommand = 'N';
	status.permitturnopen = 'N';
}
#endif
```

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

	/* 戻る */
	return;
}

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

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

```
}
#endif

#ifdef NOTUSE_FILES
int WriteString(struct EV_File *This, char *filename, char *str)
{
	switch(filename[0])
	{
	case 'L':
		status.p_limit = &status.limit[0];
		strcpy(status.p_limit, str);
		break;
	default:
		break;
	}
	return OK;
}
#else
/* 文字列書き込み */
int WriteString(struct EV_File *This, char *filename, char *str)
{
	int Ret = OK;
	if((This->fp = fopen(filename, "w")) == NULL){
		Ret = NG;
	}
	/* ¥nは追記されない */
	else if(fputs(str, This->fp) >= 0){
		fclose(This->fp);
		/* 書き込み成功 */
```

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

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

```
            case 'C':
                *p_ch = status.permitcommand;
                break;
            case 'T':
                *p_ch = status.permitturnopen;
                break;
            default:
                break;
            }
            break;
        default:
            break;
        }
        return OK;
}
#else
int Read(struct EV_File *This, char *filename, char *p_ch)
{
        int Ret = OK;
        if((This->fp = fopen(filename, "r")) == NULL){
            Ret = NG;
        }
        else if((*p_ch = fgetc(This->fp)) == EOF){
            fclose(This->fp);
            Ret = ONE_MORE_TIME;
        }
        else if(*p_ch == '¥n'){
            fclose(This->fp);
            Ret = ONE_MORE_TIME;
```

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

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

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

int PermitCommand_Read(struct EV_File *This, char *p_PermitCommand)
{
	int Ret = OK;
	switch(Read(This, "PermitCommand.txt¥0", p_PermitCommand)){
```

```
	case NG:
		PrintF(ClsPnl, "¥nReading Error");
		Ret = NG;
		break;
	case ONE_MORE_TIME:
		Ret = ONE_MORE_TIME;
		break;
	default:
		Ret = OK;
		break;
	}
	return Ret;
}

int PermitCommand_Write(struct EV_File *This, char PermitCommand)
{
	int Ret = OK;
	switch(Write(This, "PermitCommand.txt¥0", PermitCommand)){
	case NG:
		PrintF(ClsPnl, "¥nWriting Error");
		Ret = NG;
		break;
	case OK:
		Ret = OK;
		break;
	default:
		Ret = NG;
		break;
```

```
	}
	return Ret;
}

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

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

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

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

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

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

```
	}

	/* 戻る */
	return;
}

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

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

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

	/* 戻る */
	return;
}
```

```
/* EV_UpDown.h */

#ifndef Panel_h
#define Panel_h
#include "Panel.h"
#endif

#ifndef Timer_h
#define Timer_h
#include "Timer.h"
#endif

#ifndef EV_Time_h
#define EV_Time_h
#include "EV_Time.h"
#endif

#ifndef EV_File_h
#define EV_File_h
#include "EV_File.h"
#endif

/*==========================================================================
  上昇下降を表す構造体
==========================================================================*/
struct Position
{
    int m_UNSL;
```

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

/* 上昇 */
struct UpMotor
{
    struct EV_File SF;
    struct EV_File MF;
};

/* 下降 */
struct DownMotor
{
    struct EV_File SF;
```

```
	struct EV_File MF;
};

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

/*==========================================================
  上昇下降を表すプロトタイプ宣言
==========================================================*/
void Position(struct Position *This);
/* 上昇 */
void UpMotor(struct UpMotor *This);
void OnUpMotor(struct UpMotor *This, struct Position *P, char *p_Safety);
/* 下降 */
void DownMotor(struct DownMotor *This);
void OnDownMotor(struct DownMotor *This, struct Position *P, char *p_Safety);
/* 経過時間 */
void WaitPositionChangeLog(struct WaitPositionChangeLog *This);
void OnInitWaitPositionChangeLog(struct WaitPositionChangeLog *This, struct Position *P);
void OnWaitUpPositionChangeLog(struct WaitPositionChangeLog *This, struct Position *P);
void OnWaitDownPositionChangeLog(struct WaitPositionChangeLog *This, struct Position *P);
/* 上昇 */
```

```
void Up(struct Position *P, struct UpMotor UPMT, struct WaitPositionChangeLog WPCL, char
*p_Safety);
/* 下降 */
void Down(struct Position *P, struct DownMotor DNMT, struct WaitPositionChangeLog WPCL, char
*p_Safety);
```

```
/* EV_UpDown.c */

#include "C.h"
#include "EV_UpDown.h"

/*==========================================================================
  上昇下降を表す関数
==========================================================================*/
/*
 * Position
 */
void Position(struct Position *This)
{
	/* 初期値 */
	This->m_UNSL = OFF;
	This->m_UNST = OFF;
	This->m_UPSL = OFF;
	This->m_UPST = OFF;
	This->p_UnderSlow = &This->m_UNSL;
	This->p_UnderStop = &This->m_UNST;
	This->p_UpperSlow = &This->m_UPSL;
	This->p_UpperStop = &This->m_UPST;
	/* Sleep用 */
	This->fstop = OFF;
	This->p_fstop = &This->fstop;
	This->fmove = OFF;
	This->p_fmove = &This->fmove;
```

```
	/* 戻る */
	return;
}

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

	/* 戻る */
	return;
}

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

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

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

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

    /* 戻る */
    return;
}

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

    /* 戻る */
    return;
}

/*
 * OnDownMotor
 */
/* 下降 */
```

```
void OnDownMotor(struct DownMotor *This, struct Position *P, char *p_Safety)
{
	if(*P->p_UnderStop == ON){
		/* Sleep用 */
		if(*P->p_fstop == OFF){
			*P->p_fstop = ON;
			/* 現在実行中の命令を外部に報告 */
			Motor_Write(&This->MF, 's');
			PrintF(ClsPnl, "STOP");
		}
		if(*p_Safety == 'Y'){
			*p_Safety = 'r';
			Write(&This->SF, "Safety.txt¥0", 'r');
		}
	}
	else if(*P->p_UnderSlow == ON){
		/* Sleep用 */
		*P->p_fstop = OFF;
		if(*P->p_fmove == OFF){
			*P->p_fmove = ON;
			/* 現在実行中の命令を外部に報告 */
			Motor_Write(&This->MF, 'd');
			PrintF(ClsPnl, "DOWN");
		}
		else if(*p_Safety == 'Y'){
			Motor_Write(&This->MF, 'k');
			*p_Safety = 'r';
			Write(&This->SF, "Safety.txt¥0", 'r');
		}
```

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

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

        /* 戻る */
        return;
    }

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

```
	/* 戻る */
	return;
}

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

	/* 戻る */
	return;
}

/*
```

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

	/* 戻る */
	return;
}

/*
 * OnWaitDownPositionChangeLog
 */
/* エレベーターの位置仮想ログ */
void OnWaitDownPositionChangeLog(struct WaitPositionChangeLog *This, struct Position *P)
{
	/* リミットスイッチの読み込み */
	This->p_strLimit = &This->strLimit[0];
```

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

	/* 戻る */
	return;
}

/*
 * Up
 */
/* 上昇 */
void Up(struct Position *P, struct UpMotor UPMT, struct WaitPositionChangeLog WPCL, char
*p_Safety)
{
	/* 到着まで */
	if(*P->p_UpperStop == OFF){
		/* エレベーターの位置仮想ログ */
		OnWaitUpPositionChangeLog(&WPCL, P);

		/* 上昇 */
		OnUpMotor(&UPMT, P, p_Safety);
```

```
        /* エレベーターの位置仮想ログ */
        OnWaitUpPositionChangeLog(&WPCL, P);
    }

    /* 戻る */
    return;
}

/*
 * Down
 */
/* 下降 */
void Down(struct Position *P, struct DownMotor DNMT, struct WaitPositionChangeLog WPCL, char
*p_Safety)
{
    /* 到着まで */
    if(*P->p_UnderStop == OFF){
        /* エレベーターの位置仮想ログ */
        OnWaitDownPositionChangeLog(&WPCL, P);

        /* 下降 */
        OnDownMotor(&DNMT, P, p_Safety);

        /* エレベーターの位置仮想ログ */
        OnWaitDownPositionChangeLog(&WPCL, P);
    }

    /* 戻る */
    return;
```

```
}
```

```
/* EV_OpenClose.h */

#ifndef Panel_h
#define Panel_h
#include "Panel.h"
#endif

#ifndef Timer_h
#define Timer_h
#include "Timer.h"
#endif

#ifndef EV_Time_h
#define EV_Time_h
#include "EV_Time.h"
#endif

#ifndef EV_File_h
#define EV_File_h
#include "EV_File.h"
#endif

#ifndef EV_UpDown_h
#define EV_UpDown_h
#include "EV_UpDown.h"
#endif

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

```
 開閉を表す構造体
==========================================================================*/
struct Door
{
	int m_CLSL;
	int m_CLST;
	int m_OPSL;
	int m_OPST;
	/* 閉減速位置 */
	int *p_CloserSlow;
	/* 閉停止位置 */
	int *p_CloserStop;
	/* 開減速位置 */
	int *p_OpennerSlow;
	/* 開停止位置 */
	int *p_OpennerStop;
	/* Sleep用 */
	int fstop;
	int *p_fstop;
	int fmove;
	int *p_fmove;
};

/* 開 */
struct OpenMotor
{
	struct EV_File SF;
	struct EV_File MF;
};
```

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

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

/*==========================================================
  開閉を表すプロトタイプ宣言
==========================================================*/
void Door(struct Door *This);
/* 開 */
void OpenMotor(struct OpenMotor *This);
void OnOpenMotor(struct OpenMotor *This, struct Door *DR, char *p_Safety);
/* 閉 */
```

```
void CloseMotor(struct CloseMotor *This);
void OnCloseMotor(struct CloseMotor *This, struct Door *DR, char *p_Safety);
/* 経過時間 */
void WaitDoorChangeLog(struct WaitDoorChangeLog *This);
void OnInitWaitDoorChangeLog(struct WaitDoorChangeLog *This, struct Door *DR);
void OnWaitOpenDoorChangeLog(struct WaitDoorChangeLog *This, struct Door *DR);
void OnWaitCloseDoorChangeLog(struct WaitDoorChangeLog *This, struct Door *DR);
/* 開 */
void Open(struct Door *DR, struct OpenMotor OPMT, struct WaitDoorChangeLog WDCL, char
*p_Safety);
/* 閉 */
void Close(struct Door *DR, struct CloseMotor CLMT, struct WaitDoorChangeLog WDCL, char
*p_Safety);
```

```
/* EV_OpenClose.c */

#include "C.h"
#include "EV_OpenClose.h"

/*==========================================================================
 開閉を表す関数
==========================================================================*/
/*
 * Door
 */
void Door(struct Door *This)
{
	/* 初期値 */
	This->m_CLSL = OFF;
	This->m_CLST = OFF;
	This->m_OPSL = OFF;
	This->m_OPST = OFF;
	This->p_CloserSlow = &This->m_CLSL;
	This->p_CloserStop = &This->m_CLST;
	This->p_OpennerSlow = &This->m_OPSL;
	This->p_OpennerStop = &This->m_OPST;
	/* Sleep用 */
	This->fstop = OFF;
	This->p_fstop = &This->fstop;
	This->fmove = OFF;
	This->p_fmove = &This->fmove;
```

```
	/* 戻る */
	return;
}

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

	/* 戻る */
	return;
}

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

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

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

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

    /* 戻る */
    return;
}

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

    /* 戻る */
    return;
}

/*
 * OnCloseMotor
 */
/* 閉 */
```

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

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

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

        /* 戻る */
        return;
    }

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

```
	/* 戻る */
	return;
}

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

	/* 戻る */
	return;
}
```

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

	/* 戻る */
	return;
}

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

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

	/* 戻る */
	return;
}

/*
 * Open
 */
/* 開 */
void Open(struct Door *DR, struct OpenMotor OPMT, struct WaitDoorChangeLog WDCL, char
*p_Safety)
{
	/* 到着まで */
	if(*DR->p_OpennerStop == OFF)
	{
		/* エレベーターの位置仮想ログ */
		OnWaitOpenDoorChangeLog(&WDCL, DR);

		/* 開 */
```

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

        /* エレベーターの位置仮想ログ */
        OnWaitOpenDoorChangeLog(&WDCL, DR);
    }

    /* 戻る */
    return;
}

/*
 * Close
 */
/* 閉 */
void Close(struct Door *DR, struct CloseMotor CLMT, struct WaitDoorChangeLog WDCL, char
*p_Safety)
{
    /* 到着まで */
    if(*DR->p_CloserStop == OFF)
    {
        /* エレベーターの位置仮想ログ */
        OnWaitCloseDoorChangeLog(&WDCL, DR);

        /* 閉 */
        OnCloseMotor(&CLMT, DR, p_Safety);

        /* エレベーターの位置仮想ログ */
        OnWaitCloseDoorChangeLog(&WDCL, DR);
    }
```

```
	/* 戻る */
	return;
}
```

```
/* EV_Display.h */

#ifndef Panel_h
#define Panel_h
#include "Panel.h"
#endif

#ifndef Timer_h
#define Timer_h
#include "Timer.h"
#endif

#ifndef EV_Time_h
#define EV_Time_h
#include "EV_Time.h"
#endif

#ifndef EV_File_h
#define EV_File_h
#include "EV_File.h"
#endif

/*==========================================================================
  シミュレータを表す関数のプロトタイプ宣言
==========================================================================*/
void DispInput(void);
void Disp(char ch, char str[9]);
```

```
/* EV_Display.c */

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

void DispInput(void)
{
	/* 入力指示 */
	PrintF(InputCommand, "¥nUP = 'u', DOWN = 'd', OPEN = 'o', CLOSE = 'c'");
	PrintF(InputCommand, "¥nEMERGENCY = 's', RECOVERY = 'r'");
	PrintF(InputCommand, "¥n1st Floor CALL = 'y', 2nd Floor CALL = 'Y'");
	PrintF(InputCommand, "¥n1st Floor CLOSE = 'h', 2nd Floor CLOSE = 'H'");
	PrintF(InputCommand, "¥nQUIT = 'q'");
	PrintF(InputCommand, "¥nCOMMAND>");
}

/*
 * 表示関数
 */
void Disp(char ch, char str[9])
{
#ifdef USE_BCC
	int	i;
	/* 画面クリア */
	CLEAR;
	if((((ch == 'o') ¦¦ (ch == 'h')) && (str[3] == 'y') && (str[7] == 'y'))
		¦¦ ((ch == 't') && (str[3] == 'y') && (str[7] == 'n') && (str[6] == 'y'))
		¦¦ ((ch == 's') && (str[3] == 'y') && (str[7] == 'y')))
```

```
    {
        for(i = 0; i < 4; i++)
        {
            PrintF(Monitor, "¥n0000       0000");
        }
        for(i = 4; i < 12; i++)
        {
            PrintF(Monitor, "¥n                ");
        }
    }
    else if((((ch == 'O') ¦¦ (ch == 'h')) && (str[3] == 'y') && (str[7] == 'n') && (str[6] == 'y'))
        ¦¦  (((ch == 'c') ¦¦ (ch == 't')) && (str[3] == 'y') && (str[7] == 'n') && (str[6] == 'n') && (str[5] ==
'n') && (str[4] == 'n'))
        ¦¦  ((ch == 's') && (str[3] == 'y') && (str[7] == 'n') && (str[6] == 'y')))
    {
        for(i = 0; i < 4; i++)
        {
            PrintF(Monitor, "¥n 0000     0000 ");
        }
        for(i = 4; i < 12; i++)
        {
            PrintF(Monitor, "¥n                ");
        }
    }
    else if((ch == 's') && (str[3] == 'y') && (str[7] == 'n') && (str[6] == 'n') && (str[5] == 'n') && (str[4]
== 'n'))
    {
        for(i = 0; i < 4; i++)
        {
```

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

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

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

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

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

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

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

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

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

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

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

```
        ¦¦  ((ch == 's') && (str[0] == 'y') && (str[7] == 'y')))
    {
        for(i = 0; i < 8; i++)
        {
            PrintF(Monitor, "¥n              ");
        }
        for(i = 8; i < 12; i++)
        {
            PrintF(Monitor, "¥n0000      0000");
        }
    }
    else if((((ch == 'O') ¦¦ (ch == 'h')) && (str[0] == 'y') && (str[7] == 'n') && (str[6] == 'y'))
        ¦¦  (((ch == 'c') ¦¦ (ch == 't')) && (str[0] == 'y') && (str[7] == 'n') && (str[6] == 'n') && (str[5] ==
'n') && (str[4] == 'n'))
        ¦¦  ((ch == 's') && (str[0] == 'y') && (str[7] == 'n') && (str[6] == 'y')))
    {
        for(i = 0; i < 8; i++)
        {
            PrintF(Monitor, "¥n              ");
        }
        for(i = 8; i < 12; i++)
        {
            PrintF(Monitor, "¥n 0000     0000 ");
        }
    }
    else if((ch == 's') && (str[0] == 'y') && (str[7] == 'n') && (str[6] == 'n') && (str[5] == 'n') && (str[4]
== 'n'))
    {
        for(i = 0; i < 8; i++)
```

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

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

```
/* EV_Input.h */

#ifndef Panel_h
#define Panel_h
#include "Panel.h"
#endif

#ifndef Timer_h
#define Timer_h
#include "Timer.h"
#endif

#ifndef EV_Time_h
#define EV_Time_h
#include "EV_Time.h"
#endif

#ifndef EV_File_h
#define EV_File_h
#include "EV_File.h"
#endif

/*==========================================================================
  入力を表す関数のプロトタイプ宣言
==========================================================================*/
char GetChar(char Ret);

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

```
 入力を表す構造体
==========================================================================*/
struct EV_Input
{
    /* 安全 */
    char Safety;
    char *p_Safety;
    char Command;
    char PermitCommand;
    char *p_PermitCommand;
    char PermitTurnOpen;
    char *p_PermitTurnOpen;
    char ch;
    char *p_ch;
    char str[9];
    char *p_str;

    /* ファイルストリーム */
    struct EV_File SF;
    struct EV_File CF;
    struct EV_File PCF;
    struct EV_File PTOF;
    struct EV_File LF;
    struct EV_File MF;
};

/*==========================================================================
 入力を表すコンストラクタとメソッドのプロトタイプ宣言
==========================================================================*/
```

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

```
/* EV_Input.c */

#include "C.h"
#include "EV_Input.h"

/*==========================================================================
  入力を表す関数
==========================================================================*/
char GetChar(char Ret)
{
#ifndef USE_BCC
	char sw[4];
	int i;
	int j;
	/* スイッチワーク初期化 */
	sw[0] = sw[1] = sw[2] = sw[3] = 0;
	/* ボタンが押された時にLCD,SCI,USBにメッセージを送る */
	for(j=0;j<4;j++)
	{
		i = GetSW(j);
		if( ((sw[j]^1) & i) ) /* sw = off->onで条件成立 */
		{
			switch(j){
			case 0:
				Ret = 'u'; /* sw0 Up 2階で開く */
				break;
			case 1:
				Ret = 'd'; /* sw1 Down 1階で開く */
```

```
                    break;
                case 2:
                    Ret = 'c'; /* sw2 Close 閉じる */
                    break;
                case 3:
                    Ret = 'q'; /* sw3 Quit 終了 */
                    break;
                default:
                    break;
                }
            }
        }
    #else
    #ifdef USE_LINUX
        if(kbhit())
        {
            Ret = (char) getchar();
        }
    #else
    #ifdef USE_MSVS2005
        if(_kbhit())
        {
            Ret = (char) _getche();
        }
    #else
        if(kbhit())
        {
            Ret = (char) getche();
        }
```

```
#endif
#endif
#endif
	return Ret;
}

/*==========================================================================
 入力を表すコンストラクタとメソッド
==========================================================================*/
void EV_Input(struct EV_Input *This)
{
	/* 初期化 */
	This->Command = '¥0';
	This->p_ch = &This->ch;
	This->p_str = &This->str[0];
	This->str[8] = '¥0';
	/* 安全初期化 */
	This->p_Safety = &This->Safety;
	/* 入力許可初期化 */
	This->p_PermitCommand = &This->PermitCommand;
	/* 反転開許可初期化 */
	This->p_PermitTurnOpen = &This->PermitTurnOpen;

	EV_File(&This->SF);
	EV_File(&This->CF);
	EV_File(&This->PCF);
	EV_File(&This->PTOF);
	EV_File(&This->MF);
```

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

void OnInput(struct EV_Input *This, Thread *th)
{
    if(Command_Read(&This->CF, &This->Command) == NG) return;
    This->Command = GetChar(This->Command);

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

    switch(This->Command){
```

```
case 's':
    /* 命令入力 */
    Write(&This->SF, "Safety.txt", 's');
    Motor_Write(&This->MF, 's');
    This->Command = 'N';
    Command_Write(&This->CF, 'N');
    PermitCommand_Write(&This->PCF, 'N');
    break;
case 'r':
    /* 命令入力 */
    Write(&This->SF, "Safety.txt", 'h');
    This->Command = 'N';
    Command_Write(&This->CF, 'N');
    PermitCommand_Write(&This->PCF, 'c');
    break;
case 'q':
    /* 命令入力 */
    Command_Write(&This->CF, This->Command);
    PrintF(ClsPnl, "QUIT");
    delete_(th);
    break;
default:
    if(This->PermitCommand == 'c'){
        switch(This->Command){
        case 'o':
            if(This->str[4] != 'y'){
                PermitTurnOpen_Write(&This->PTOF, 'o');
            }
```

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

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

```
                    This->Safety = 'Y';
                    Write(&This->SF, "Safety.txt", 'Y');
                    if((This->ch == 's') && (This->Safety == 'Y') && (This->str[3] == 'y') && (This->str[4] == 'n')){
                        Motor_Write(&This->MF, 't');
                    }
                    else if((This->ch == 's') && (This->Safety == 'Y') && (This->str[0] == 'n')){
                        Motor_Write(&This->MF, 'k');
                    }
                    else if((This->ch == 's') && (This->Safety == 'Y') && (This->str[0] == 'y') && (This->str[7] == 'n')){
                        Motor_Write(&This->MF, 'h');
                    }
                }
                /* 命令入力 */
                Command_Write(&This->CF, This->Command);
                PermitCommand_Write(&This->PCF, 'N');
                break;
            case 'y':
                if((This->str[0] != 'y') && (This->str[4] != 'y')){
                    PrintF(Panel, "¥nA Basket isn't 1st Floor");
                }
                else{
                    if(This->Safety == 'h'){
                        /* 命令入力 */
                        This->Safety = 'Y';
                        Write(&This->SF, "Safety.txt", 'Y');
                        if((This->ch == 's') && (This->Safety == 'Y') && (This->str[0] == 'n')){
                            Motor_Write(&This->MF, 'k');
```

```
                    }
                    else if((This->ch == 's') && (This->Safety == 'Y') && (This->str[0] == 'y') &&
(This->str[7] == 'n')){
                        Motor_Write(&This->MF, 'h');
                    }
                }
                /* 命令入力 */
                Command_Write(&This->CF, This->Command);
                PermitCommand_Write(&This->PCF, 'N');
            }
            break;
        case 'Y':
            if((This->str[3] != 'y') && (This->str[4] != 'y')){
                PrintF(Panel, "¥nA Basket isn't 2nd Floor");
            }
            else{
                if(This->Safety == 'h'){
                    /* 命令入力 */
                    This->Safety = 'Y';
                    Write(&This->SF, "Safety.txt", 'Y');
                    if((This->ch == 's') && (This->Safety == 'Y') && (This->str[3] == 'n')){
                        Motor_Write(&This->MF, 'j');
                    }
                    else if((This->ch == 's') && (This->Safety == 'Y') && (This->str[3] == 'y') &&
(This->str[7] == 'n')){
                        Motor_Write(&This->MF, 'h');
                    }
                }
```

```
            /* 命令入力 */
            Command_Write(&This->CF, This->Command);
            PermitCommand_Write(&This->PCF, 'N');
        }
        break;
    case 'h':
        if(This->str[0] != 'y'){
            PrintF(Panel, "¥nA Basket isn't 1st Floor");
        }
        else{
            if(This->str[4] != 'y'){
                PermitTurnOpen_Write(&This->PTOF, 'o');
            }
            if(This->Safety == 'h'){
                /* 命令入力 */
                This->Safety = 'Y';
                Write(&This->SF, "Safety.txt", 'Y');
                if((This->ch == 's') && (This->Safety == 'Y') && (This->str[4] == 'n')){
                    Motor_Write(&This->MF, 't');
                }
            }
            /* 命令入力 */
            Command_Write(&This->CF, This->Command);
            PermitCommand_Write(&This->PCF, 'N');
        }
        break;
    case 'H':
        if(This->str[3] != 'y'){
            PrintF(Panel, "¥nA Basket isn't 2nd Floor");
```

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

```
                    /* 命令入力 */
                    This->Safety = 'Y';
                    Write(&This->SF, "Safety.txt", 'Y');
                    if((This->ch == 's') && (This->Safety == 'Y') && (This->str[0] == 'y') && (This->str[7] == 'n')){
                        Motor_Write(&This->MF, 'h');
                    }
                }
                /* 命令入力 */
                Command_Write(&This->CF, This->Command);
                PermitTurnOpen_Write(&This->PTOF, 'N');
                break;
            default:
                break;
            }
        }
        else if((This->str[3] == 'y') && (This->str[7] != 'y')){
            switch(This->Command){
            case 'o':
            case 'u':
            case 'Y':
                if(This->Safety == 'h'){
                    /* 命令入力 */
                    This->Safety = 'Y';
                    Write(&This->SF, "Safety.txt", 'Y');
                    if((This->ch == 's') && (This->Safety == 'Y') && (This->str[3] == 'y') && (This->str[7] == 'n')){
                        Motor_Write(&This->MF, 'h');
                    }
```

```
                    }
                    /* 命令入力 */
                    Command_Write(&This->CF, This->Command);
                    PermitTurnOpen_Write(&This->PTOF, 'N');
                    break;
                default:
                    break;
                }
            }
        }
        break;
    }
    return;
}
```

```
/* EV_Controller.h */

#ifndef Panel_h
#define Panel_h
#include "Panel.h"
#endif

#ifndef Timer_h
#define Timer_h
#include "Timer.h"
#endif

#ifndef EV_Time_h
#define EV_Time_h
#include "EV_Time.h"
#endif

#ifndef EV_File_h
#define EV_File_h
#include "EV_File.h"
#endif

#ifndef EV_UpDown_h
#define EV_UpDown_h
#include "EV_UpDown.h"
#endif

#ifndef EV_OpenClose_h
```

```
#define EV_OpenClose_h
#include "EV_OpenClose.h"
#endif

/*==========================================================================
 制御盤を表す構造体宣言
==========================================================================*/
struct EV_Controller
{
    /* エレベーターの現在位置オブジェクトの宣言 */
    struct Position P;
    struct Position *p_P;

    /* 上昇オブジェクトの宣言 */
    struct UpMotor UPMT;
    /* 下降オブジェクトの宣言 */
    struct DownMotor DNMT;

    /* エレベーターの位置仮想ログ */
    struct WaitPositionChangeLog WPCL;

    /* エレベーターの現在位置オブジェクトの宣言 */
    struct Door DR;
    struct Door *p_DR;

    /* 開オブジェクトの宣言 */
    struct OpenMotor OPMT;
    /* 閉オブジェクトの宣言 */
    struct CloseMotor CLMT;
```

```
    /* エレベーターの位置仮想ログ */
    struct WaitDoorChangeLog WDCL;

    /* 経過時間テンポラリ */
    struct EV_Time T;

    /* 安全 */
    char Safety;
    char *p_Safety;
    /* Limit */
    char str[9];
    char *p_str;
    /* 命令 */
    char Command;
    char *p_Command;
    char PermitCommand;
    char *p_PermitCommand;
    char PermitTurnOpen;
    char *p_PermitTurnOpen;

    /* ファイルストリーム */
    struct EV_File SF;
    struct EV_File LF;
    struct EV_File CF;
    struct EV_File PCF;
    struct EV_File PTOF;
    struct EV_File MF;
};
```

```
/*==========================================================================
  制御を表すコンストラクタとメソッドのプロトタイプ宣言
==========================================================================*/
void EV_Controller(struct EV_Controller *This, Thread *th);
void OnController(struct EV_Controller *This, Thread *th);
```

```
/* EV_Controller.c */

#include "C.h"
#include "EV_Controller.h"

/*==========================================================================
  制御関数
==========================================================================*/
void EV_Controller(struct EV_Controller *This, Thread *th)
{
    /* 初期化 */
    This->p_P = &This->P;
    Position(This->p_P);
    UpMotor(&This->UPMT);
    DownMotor(&This->DNMT);
    WaitPositionChangeLog(&This->WPCL);
    This->p_DR = &This->DR;
    Door(This->p_DR);
    OpenMotor(&This->OPMT);
    CloseMotor(&This->CLMT);
    WaitDoorChangeLog(&This->WDCL);
    EV_Time(&This->T, th);
    EV_File(&This->SF);
    EV_File(&This->LF);
    EV_File(&This->CF);
    EV_File(&This->PCF);
    EV_File(&This->PTOF);
    EV_File(&This->MF);
```

```
    /* 安全初期化 */
    This->p_Safety = &This->Safety;
    /* Limit初期化 */
    This->p_str = &This->str[0];
    /* 命令初期化 */
    This->p_Command = &This->Command;
    This->p_PermitCommand = &This->PermitCommand;
    This->p_PermitTurnOpen = &This->PermitTurnOpen;

    /* モーター停止命令 */
    Motor_Write(&This->MF, 's');

    /* エレベーターの位置仮想ログ */
    /* 初期値 */
    OnInitWaitPositionChangeLog(&This->WPCL, This->p_P);

    /* エレベーターの位置仮想ログ */
    /* 初期値 */
    OnInitWaitDoorChangeLog(&This->WDCL, This->p_DR);

    /* リミットスイッチの前状態読み込み */
    ReadString(&This->LF, "Limit.txt¥0", This->p_str, 9);
    /* 命令初期化 */
    if(Command_Write(&This->CF, 'N') == NG) return;
    if(PermitCommand_Write(&This->PCF, 'c') == NG) return;
    if(PermitTurnOpen_Write(&This->PTOF, 'N') == NG) return;
}
```

```
/*
 * 主制御関数
 */
void OnController(struct EV_Controller *This, Thread *th)
{
	/* 安全入力 */
	if(Read(&This->SF, "Safety.txt¥0", This->p_Safety) == NG) return;
	/* リミットスイッチの前状態読み込み */
	ReadString(&This->LF, "Limit.txt¥0", This->p_str, 9);
	/* 命令入力 */
	if(Command_Read(&This->CF, This->p_Command) == NG) return;
	/* 入力許可 */
	if(PermitCommand_Read(&This->PCF, This->p_PermitCommand) == NG) return;
	/* 反転開許可 */
	if(PermitTurnOpen_Read(&This->PTOF, This->p_PermitTurnOpen) == NG) return;

	switch(This->Command){
	/* 終了命令ならば */
	case 'q':
		Motor_Write(&This->MF, 's');
		delete_(th);
		break;
	/* 非常停止命令ならば */
	case 's':
		SetPermit(&This->T, OFF);
		break;
	/* 復帰命令ならば */
	case 'r':
		break;
```

```
/* 上階呼命令ならば */
case 'Y':
    if((*This->p_P->p_UnderStop == ON) && (*This->p_DR->p_CloserStop == OFF)){
        break;
    }
/* 上昇命令ならば */
case 'u':
    if(*This->p_P->p_UpperStop == ON){
        /* 開完了時 */
        if(*This->p_DR->p_OpennerStop == ON){
            PermitCommand_Write(&This->PCF, 'c');
            SetPermit(&This->T, ON);
            Command_Write(&This->CF, 'N');
            Clear();
            PrintF(Panel, "Hello EV       ");
            break;
        }
        else{
            SetPermit(&This->T, OFF);
            SetCurrentTime(&This->T);
            /* 開 */
            Open(This->p_DR,This->OPMT,This->WDCL, This->p_Safety);
        }
    }
    else if(*This->p_DR->p_CloserStop == ON){
        /* 閉完了時 */
        SetPermit(&This->T, OFF);
        SetCurrentTime(&This->T);
        PermitTurnOpen_Write(&This->PTOF, 'N');
```

```
            /* 上昇 */
            Up(This->p_P,This->UPMT,This->WPCL, This->p_Safety);
        }
        else if(*This->p_DR->p_CloserStop == OFF){
            SetPermit(&This->T, OFF);
            SetCurrentTime(&This->T);
            /* 閉 */
            Close(This->p_DR,This->CLMT,This->WDCL, This->p_Safety);
        }
        else{
            SetPermit(&This->T, OFF);
            SetCurrentTime(&This->T);
        }
        break;
    /* 下階呼命令ならば */
    case 'y':
        if((*This->p_P->p_UpperStop == ON) && (*This->p_DR->p_CloserStop == OFF)){
            break;
        }
    /* 下降命令ならば */
    case 'd':
        if(*This->p_P->p_UnderStop == ON){
            /* 開完了時 */
            if(*This->p_DR->p_OpennerStop == ON){
                PermitCommand_Write(&This->PCF, 'c');
                SetPermit(&This->T, ON);
                Command_Write(&This->CF, 'N');
                Clear();
```

```
            PrintF(Panel, "Hello EV       ");
            break;
        }
        else{
            SetPermit(&This->T, OFF);
            SetCurrentTime(&This->T);
            /* 開 */
            Open(This->p_DR,This->OPMT,This->WDCL, This->p_Safety);
        }
    }
    else if(*This->p_DR->p_CloserStop == ON){
        /* 閉完了時 */
        SetPermit(&This->T, OFF);
        SetCurrentTime(&This->T);
        PermitTurnOpen_Write(&This->PTOF, 'N');
        /* 下降 */
        Down(This->p_P,This->DNMT,This->WPCL, This->p_Safety);
    }
    else if(*This->p_DR->p_CloserStop == OFF){
        SetPermit(&This->T, OFF);
        SetCurrentTime(&This->T);
        /* 閉 */
        Close(This->p_DR,This->CLMT,This->WDCL, This->p_Safety);
    }
    else{
        SetPermit(&This->T, OFF);
        SetCurrentTime(&This->T);
    }
    break;
```

```
/* 開命令ならば */
case 'o':
	/* 開完了時 */
	if(*This->p_DR->p_OpennerStop == ON){
		PermitCommand_Write(&This->PCF, 'c');
		SetPermit(&This->T, ON);
		Command_Write(&This->CF, 'N');
		Clear();
		PrintF(Panel, "Hello EV       ");
		break;
	}
	else{
		SetPermit(&This->T, OFF);
		SetCurrentTime(&This->T);
		/* 開 */
		Open(This->p_DR,This->OPMT,This->WDCL, This->p_Safety);
	}
	break;
/* 閉命令ならば */
case 'c':
	SetPermit(&This->T, OFF);
	SetCurrentTime(&This->T);
	/* 閉完了時 */
	if(*This->p_DR->p_CloserStop == ON){
		PermitTurnOpen_Write(&This->PTOF, 'N');
		Command_Write(&This->CF, 'N');
		PermitCommand_Write(&This->PCF, 'c');
		Clear();
		PrintF(Panel, "Hello EV       ");
```

```
            break;
        }
        else{
            /* 閉 */
            Close(This->p_DR,This->CLMT,This->WDCL, This->p_Safety);
        }
        break;
    /* 閉命令ならば */
    case 'H':
        if(*This->p_P->p_UpperStop == ON){
            SetPermit(&This->T, OFF);
            SetCurrentTime(&This->T);
            /* 閉完了時 */
            if(*This->p_DR->p_CloserStop == ON){
                Command_Write(&This->CF, 'N');
                PermitTurnOpen_Write(&This->PTOF, 'N');
                PermitCommand_Write(&This->PCF, 'c');
                Clear();
                PrintF(Panel, "Hello EV       ");
                break;
            }
            else{
                /* 閉 */
                Close(This->p_DR,This->CLMT,This->WDCL, This->p_Safety);
            }
        }

        break;
    /* 閉命令ならば */
```

```
    case 'h':
        if(*This->p_P->p_UnderStop == ON){
            SetPermit(&This->T, OFF);
            SetCurrentTime(&This->T);
            /* 閉完了時 */
            if(*This->p_DR->p_CloserStop == ON){
                Command_Write(&This->CF, 'N');
                PermitTurnOpen_Write(&This->PTOF, 'N');
                PermitCommand_Write(&This->PCF, 'c');
                Clear();
                PrintF(Panel, "Hello EV       ");
                break;
            }
            else{
                /* 閉 */
                Close(This->p_DR,This->CLMT,This->WDCL, This->p_Safety);
            }
        }
        break;
    default:
        break;
    }
    if((GetCurrentTime(&This->T) >= OPENTIMEOUT) && (*This->p_DR->p_OpennerStop == ON) &&
(GetPermit(&This->T) == ON)){
        SetPermit(&This->T, OFF);
        SetCurrentTime(&This->T);
        PermitTurnOpen_Write(&This->PTOF, 'o');
        PermitCommand_Write(&This->PCF, 'N');
```

```
            /* 閉 */
            Command_Write(&This->CF, 'c');
        }
        return;
    }
```

```
/* EV_Puls.h */

#ifndef Panel_h
#define Panel_h
#include "Panel.h"
#endif

#ifndef Timer_h
#define Timer_h
#include "Timer.h"
#endif

#ifndef EV_Time_h
#define EV_Time_h
#include "EV_Time.h"
#endif

#ifndef EV_File_h
#define EV_File_h
#include "EV_File.h"
#endif

/* Raspberry Pi 3 Model B I/O */
#ifdef USE_RASPBIAN
#define GPIO16 16
#define GPIO17 17
#define GPIO18 18
#define GPIO19 19
```

```
#define GPIO20 20
#endif

/*==========================================================
 送信を表す構造体
==========================================================*/
struct EV_Puls
{
	char ch;
	char *p_ch;
	char Command;
	char *p_Command;
	/* ファイルストリーム */
	struct EV_File CF;
	struct EV_File MF;
};

/*==========================================================
 送信を表す関数のプロトタイプ宣言
==========================================================*/
void EV_Set(int addressDataSet, int dataSet, int addressClockSet,
 int clockSet);
void EV_EnableSet(void);
int EV_AddressSet(Thread *th, int base, int address);
int EV_DataSet(Thread *th, int base, int data);
void EV_AddressDataSet(struct EV_Puls *puls, Thread *th,
 int address_1, int address_2, int address_3, int address_4,
 int data_1, int data_2, int data_3, int data_4);
```

```
/*==========================================================================
  送信を表すコンストラクタとメソッドのプロトタイプ宣言
==========================================================================*/
void EV_Puls(struct EV_Puls *This, Thread *th);
void OnPuls(struct EV_Puls *This, Thread *th);
```

```
/* EV_Puls.c */

#include "C.h"
#include "EV_Puls.h"

void EV_Set(int addressDataSet, int dataSet, int addressClockSet,
 int clockSet){
#ifndef USE_BCC
	/* address data set */
	if(addressDataSet == 0){
		PB.DR &= 0xfe;
	}
	else if(addressDataSet == 1){
		PB.DR |= 0x01;
	}
	/* data set */
	if(dataSet == 0){
		PB.DR &= 0xfd;
	}
	else if(dataSet == 1){
		PB.DR |= 0x02;
	}
	/* address clock set */
	if(addressClockSet == 0){
		PB.DR &= 0xfb;
	}
	else if(addressClockSet == 1){
		PB.DR |= 0x04;
```

```
    }
    /* clock set */
    if(clockSet == 0){
        PB.DR &= 0xf7;
    }
    else if(clockSet == 1){
        PB.DR ¦= 0x08;
    }
    /* disable set */
    PB.DR &= 0xef;
#endif
#ifdef USE_RASPBIAN
    /* address data set */
    digitalWrite(GPIO16, addressDataSet);
    /* data set */
    digitalWrite(GPIO17, dataSet);
    /* address clock set */
    digitalWrite(GPIO18, addressClockSet);
    /* clock set */
    digitalWrite(GPIO19, clockSet);
    /* disable set */
    digitalWrite(GPIO20, 0);
#endif
    return;
}

void EV_EnableSet(void){
#ifndef USE_BCC
    /* address data set */
```

```
        PB.DR &= 0xfe;
        /* data set */
        PB.DR &= 0xfd;
        /* address clock set */
        PB.DR &= 0xfb;
        /* clock set */
        PB.DR &= 0xf7;
        /* enable set */
        PB.DR ¦= 0x10;
    #endif
    #ifdef USE_RASPBIAN
        /* address data set */
        digitalWrite(GPIO16, 0);
        /* data set */
        digitalWrite(GPIO17, 0);
        /* address clock set */
        digitalWrite(GPIO18, 0);
        /* clock set */
        digitalWrite(GPIO19, 0);
        /* enable set */
        digitalWrite(GPIO20, 1);
    #endif
        return;
    }

    int EV_AddressSet(Thread *th, int base, int address){
        int Ret;
        Ret = NG;
        if(th->count == base){
```

```
        EV_Set(address, 0, 0, 0);
        th->count++;
        Ret = OK;
    }
    else if(th->count == base + 1){
        EV_Set(address, 0, 1, 0);
        th->count++;
        Ret = OK;
    }
    return Ret;
}

int EV_DataSet(Thread *th, int base, int data){
    int Ret;
    Ret = NG;
    if(th->count == base){
        EV_Set(0, data, 0, 0);
        th->count++;
        Ret = OK;
    }
    else if(th->count == base + 1){
        EV_Set(0, data, 0, 1);
        th->count++;
        Ret = OK;
    }
    return Ret;
}

void EV_AddressDataSet(struct EV_Puls *puls, Thread *th,
```

```
 int address_1, int address_2, int address_3, int address_4,
 int data_1, int data_2, int data_3, int data_4){
      if(EV_AddressSet(th, 0, address_1) == OK);
      else if(EV_AddressSet(th, 2, address_2) == OK);
      else if(EV_AddressSet(th, 4, address_3) == OK);
      else if(EV_AddressSet(th, 6, address_4) == OK);
      else if(EV_DataSet(th, 8, data_1) == OK);
      else if(EV_DataSet(th, 10, data_2) == OK);
      else if(EV_DataSet(th, 12, data_3) == OK);
      else if(EV_DataSet(th, 14, data_4) == OK);
      else{
           EV_EnableSet();
           th->count = 0;
           switch(puls->Command){
           case 'q':
#ifndef USE_BCC
                /* address data set */
                PB.DR &= 0xfe;
                /* data set */
                PB.DR &= 0xfd;
                /* address clock set */
                PB.DR &= 0xfb;
                /* clock set */
                PB.DR &= 0xf7;
                /* disable set */
                PB.DR &= 0xef;
#endif
#ifdef USE_RASPBIAN
```

```
                /* address data set */
                digitalWrite(GPIO16, 0);
                /* data set */
                digitalWrite(GPIO17, 0);
                /* address clock set */
                digitalWrite(GPIO18, 0);
                /* clock set */
                digitalWrite(GPIO19, 0);
                /* disable set */
                digitalWrite(GPIO20, 0);
#endif
                delete_(th);
                break;
            default:
                break;
            }
        }
        return;
}

void EV_Puls(struct EV_Puls *This, Thread *th){
        This->p_ch = &This->ch;
        This->p_Command = &This->Command;
        EV_File(&This->MF);
        EV_File(&This->CF);
#ifndef USE_BCC
        PB.DDR = 0xff; /* bit7..0 out */
        PB.DR ¦= 0xff;
#endif
```

```
#ifdef USE_RASPBIAN
	if (wiringPiSetupGpio() == -1) exit(NG);
	pinMode(GPIO16, OUTPUT);
	pinMode(GPIO17, OUTPUT);
	pinMode(GPIO18, OUTPUT);
	pinMode(GPIO19, OUTPUT);
	pinMode(GPIO20, OUTPUT);
#endif
	return;
}

void OnPuls(struct EV_Puls *This, Thread *th){
	if(th->count == 0){
		/* 命令入力 */
		Command_Read(&This->CF, This->p_Command);
		/* モーター命令解読 */
		Read(&This->MF, "Motor.txt¥0", This->p_ch);
	}
	switch(This->ch){
	case 's':
		EV_AddressDataSet(This, th, 0, 1, 1, 1, 0, 0, 0, 0);
		break;
	case 'j':
		EV_AddressDataSet(This, th, 0, 1, 1, 1, 0, 0, 0, 1);
		break;
	case 'u':
		EV_AddressDataSet(This, th, 0, 1, 1, 1, 0, 0, 1, 0);
		break;
	case 'U':
```

```
        EV_AddressDataSet(This, th, 0, 1, 1, 1, 0, 0, 1, 1);
        break;
    case 'k':
        EV_AddressDataSet(This, th, 0, 1, 1, 1, 0, 1, 0, 0);
        break;
    case 'd':
        EV_AddressDataSet(This, th, 0, 1, 1, 1, 0, 1, 0, 1);
        break;
    case 'D':
        EV_AddressDataSet(This, th, 0, 1, 1, 1, 0, 1, 1, 0);
        break;
    case 'h':
        EV_AddressDataSet(This, th, 0, 1, 1, 1, 0, 1, 1, 1);
        break;
    case 'o':
        EV_AddressDataSet(This, th, 0, 1, 1, 1, 1, 0, 0, 0);
        break;
    case 'O':
        EV_AddressDataSet(This, th, 0, 1, 1, 1, 1, 0, 0, 1);
        break;
    case 't':
        EV_AddressDataSet(This, th, 0, 1, 1, 1, 1, 0, 1, 0);
        break;
    case 'c':
        EV_AddressDataSet(This, th, 0, 1, 1, 1, 1, 0, 1, 1);
        break;
    case 'C':
        EV_AddressDataSet(This, th, 0, 1, 1, 1, 1, 1, 0, 0);
        break;
```

```
        default:
            break;
        }
        return;
    }
```

```
/* EV_Simulator.h */

#ifndef Panel_h
#define Panel_h
#include "Panel.h"
#endif

#ifndef Timer_h
#define Timer_h
#include "Timer.h"
#endif

#ifndef EV_Time_h
#define EV_Time_h
#include "EV_Time.h"
#endif

#ifndef EV_File_h
#define EV_File_h
#include "EV_File.h"
#endif

#ifndef EV_Display_h
#define EV_Display_h
#include "EV_Display.h"
#endif

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

```
 シミュレータを表す構造体
==========================================================================*/
struct EV_Simulator
{
	char ch;
	char *p_ch;
	char ch2;
	char *p_ch2;
	char ch3;
	char *p_ch3;
	char str[9];
	char *p_str;

	/* 時間管理 */
	struct EV_Time T;

	/* ファイルストリーム */
	struct EV_File SF;
	struct EV_File CF;
	struct EV_File MF;
	struct EV_File LF;
};

/*==========================================================================
 シミュレータを表すコンストラクタとメソッドのプロトタイプ宣言
==========================================================================*/
void EV_Simulator(struct EV_Simulator *This, Thread *th);
void OnSimulator(struct EV_Simulator *This, Thread *th);
```

```
/* EV_Simulator.c */

#include "C.h"
#include "EV_Simulator.h"

/*==========================================================================
  シミュレータ関数
==========================================================================*/
void EV_Simulator(struct EV_Simulator *This, Thread *th)
{
	/* 初期化 */
	EV_Time(&This->T, th);
	SetCurrentTime(&This->T);
	This->p_ch = &This->ch;
	This->p_ch2 = &This->ch2;
	This->p_ch3 = &This->ch3;
	This->p_str = &This->str[0];
	This->str[8] = '¥0';
	EV_File(&This->SF);
	EV_File(&This->CF);
	EV_File(&This->MF);
	EV_File(&This->LF);

	/* モーター命令解読 */
	if(Read(&This->MF, "Motor.txt¥0", This->p_ch) == NG) return;
	/* リミットスイッチの前状態読み込み */
	if(ReadString(&This->LF, "Limit.txt¥0", This->p_str, 9) == NG) return;
	/* 籠表示 */
```

```
	Disp(This->ch, This->str);
}

void OnSimulator(struct EV_Simulator *This, Thread *th)
{
	/* 終了条件 */
	Read(&This->CF, "Command.txt¥0", This->p_ch3);
	if(This->ch3 == 'q'){
		Clear();
		delete_(th);
		return;
	}

	/* モーター命令解読 */
	Read(&This->MF, "Motor.txt¥0", This->p_ch);

	/* リミットスイッチの前状態読み込み */
	ReadString(&This->LF, "Limit.txt¥0", This->p_str, 9);

	/* 停止条件 */
	Read(&This->SF, "Safety.txt¥0", This->p_ch2);
	if(This->ch2 == 's'){
		Write(&This->CF, "Command.txt¥0", 'N');
		/* 籠表示 */
		Disp(This->ch, This->str);
		return;
	}

	/* リミットスイッチの新状態作成 */
```

```
if(This->ch == 'u'){
        if(This->str[0] == 'y');
        else if(This->str[1] == 'y') This->str[1] = 'n';
        else if(This->str[2] == 'n');
        else if(This->str[3] == 'n') This->str[3] = 'y';
}
else if(This->ch == 'U'){
        if(This->str[0] == 'y');
        else if(This->str[1] == 'y');
        else if(This->str[2] == 'n') This->str[2] = 'y';
        else if(This->str[3] == 'n');
}
else if(This->ch == 'j'){
        if(This->str[0] == 'y') This->str[0] = 'n';
        else if(This->str[1] == 'y') This->str[1] = 'n';
        else if(This->str[2] == 'n') This->str[2] = 'y';
        else if(This->str[3] == 'n') This->str[3] = 'y';
}
else if(This->ch == 'd'){
        if(This->str[3] == 'y');
        else if(This->str[2] == 'y') This->str[2] = 'n';
        else if(This->str[1] == 'n');
        else if(This->str[0] == 'n') This->str[0] = 'y';
}
else if(This->ch == 'D'){
        if(This->str[3] == 'y');
        else if(This->str[2] == 'y');
        else if(This->str[1] == 'n') This->str[1] = 'y';
        else if(This->str[0] == 'n');
```

```
}
else if(This->ch == 'k'){
	if(This->str[3] == 'y') This->str[3] = 'n';
	else if(This->str[2] == 'y') This->str[2] = 'n';
	else if(This->str[1] == 'n') This->str[1] = 'y';
	else if(This->str[0] == 'n') This->str[0] = 'y';
}
else if(This->ch == 'o'){
	if(This->str[4] == 'y');
	else if(This->str[5] == 'y') This->str[5] = 'n';
	else if(This->str[6] == 'n');
	else if(This->str[7] == 'n') This->str[7] = 'y';
}
else if(This->ch == 'O'){
	if(This->str[4] == 'y');
	else if(This->str[5] == 'y');
	else if(This->str[6] == 'n') This->str[6] = 'y';
	else if(This->str[7] == 'n');
}
else if(This->ch == 'h'){
	if(This->str[4] == 'y') This->str[4] = 'n';
	else if(This->str[5] == 'y') This->str[5] = 'n';
	else if(This->str[6] == 'n') This->str[6] = 'y';
	else if(This->str[7] == 'n') This->str[7] = 'y';
}
else if(This->ch == 'c'){
	if(This->str[7] == 'y');
	else if(This->str[6] == 'y') This->str[6] = 'n';
	else if(This->str[5] == 'n');
```

```
        else if(This->str[4] == 'n') This->str[4] = 'y';
    }
    else if(This->ch == 'C'){
        if(This->str[7] == 'y');
        else if(This->str[6] == 'y');
        else if(This->str[5] == 'n') This->str[5] = 'y';
        else if(This->str[4] == 'n');
    }
    else if(This->ch == 't'){
        if(This->str[7] == 'y') This->str[7] = 'n';
        else if(This->str[6] == 'y') This->str[6] = 'n';
        else if(This->str[5] == 'n') This->str[5] = 'y';
        else if(This->str[4] == 'n') This->str[4] = 'y';
    }

    /* リミットスイッチの新状態書き込み */
    WriteString(&This->LF, "Limit.txt¥0", This->str);

    /* 籠表示 */
    Disp(This->ch, This->str);

    return;
}
```

```
/* main.h */

#ifndef Panel_h
#define Panel_h
#include "Panel.h"
#endif

#ifndef Timer_h
#define Timer_h
#include "Timer.h"
#endif

#ifndef EV_Time_h
#define EV_Time_h
#include "EV_Time.h"
#endif

#ifndef EV_File_h
#define EV_File_h
#include "EV_File.h"
#endif

#ifndef EV_UpDown_h
#define EV_UpDown_h
#include "EV_UpDown.h"
#endif

#ifndef EV_OpenClose_h
#define EV_OpenClose_h
#include "EV_OpenClose.h"
#endif

#ifndef EV_Display_h
#define EV_Display_h
#include "EV_Display.h"
#endif

#ifndef EV_Input_h
#define EV_Input_h
#include "EV_Input.h"
#endif

#ifndef EV_Controller_h
#define EV_Controller_h
#include "EV_Controller.h"
#endif

#ifndef EV_Puls_h
#define EV_Puls_h
#include "EV_Puls.h"
#endif

#ifndef EV_Simulator_h
#define EV_Simulator_h
#include "EV_Simulator.h"
#endif
```

```
#ifdef USE_THREAD
typedef struct tag_Count
{
#ifndef USE_BCC
	int cnt[2];
#else
	int cnt[8];
#endif
}Count;
#endif
```

```
/* main.c */

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

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

```
/*==========================================================================
 入力オブジェクト宣言
==========================================================================*/
struct EV_Input in;

/*==========================================================================
 制御オブジェクト宣言
==========================================================================*/
struct EV_Controller cntrl;

/*==========================================================================
 送信オブジェクト宣言
==========================================================================*/
struct EV_Puls puls;

/*==========================================================================
 シミュレータオブジェクト宣言
==========================================================================*/
struct EV_Simulator simu;

void main(void)
{
#ifndef USE_BCC
    char sw[4];
    int i;
    int j;
    int f;
    int cnt;
    static char buff[64];
```

```
#endif
#ifdef USE_THREAD
    Thread *th30;
    Thread *th31;
#endif
#ifndef USE_BCC
    for(i=0;i<0x7fff;i++) {}
    H8init(); /* H8 レジスタ初期化 */
    InitSCI(); /* SCI1初期化(serial) */
    InitLCD(); /* LCD初期化 */

    /* LED OFF */
    SetLED(0,0);
    SetLED(1,0);
    SetLED(2,0);
    SetLED(3,0);

    /*-----------------------------------------------*/
    /* USB初期化 */
    InitUSB();
    INTC.ISCR &= (-1^0x20);  /* IRQ_5 ｾﾝｽｺﾝﾄﾛｰﾙ Active Low */
    INTC.IER ¦= 0x20; /* IRQ5 Enable */
    /*-----------------------------------------------*/
    EnableInterrupt(); /* 割り込み許可 ccr */

    f = 0;
    PrintSCI("CPU MODE %02X¥n",MDCR); /* MODE 6 */
    PrintLCD("¥fReady!3052"); /* '¥f'はLCDクリアに利用 */
```

```
    /* スイッチワーク初期化 */
    sw[0] = sw[1] = sw[2] = sw[3] = 0;
#else
    printf("¥nHello BCC");
#endif
#ifdef NOTUSE_FILES
    /* ファイル初期化 */
    new_EV_Status(&s);
#endif
#ifdef USE_THREAD
    /* タイマー初期化 */
    initWOVI();
    /* 2秒待機 */
    SleepMSec(2000);
    /* LEDTEST */
    th30 = new_Thread(30);
    th31 = new_Thread(31);
    Start(th30);
    Start(th31);
    for(;;)
    {
        /* タイマー呼び出し */
        wovi(5000000.0);
        if(Thread_checkAllDelete() == OK)
        {
            break;
        }
    }
#endif
```

```
    Clear();
    PrintF(Panel, "NEXT          ");
    /* 2秒待機 */
    SleepMSec(2000);
    Clear();
#ifndef USE_BCC
    for(;;)
    {
        /*-----------------------------------------------*/
        /* ボタンが押された時にLCD,SCI,USBにメッセージを送る */
        for(j=0;j<4;j++)
        {
            i = GetSW(j);
            if( ((sw[j]^1) & i) ) /* sw = off->onで条件成立 */
            {
                SetLED(j,1); /* LED押した瞬間点灯 */
                sprintf(buff,"sw%u",j+1);
                PrintSCI("%s¥n",buff);
                /* NULL(0x00)まで送信 */
                write_buff(buff,strlen(buff)+1);
                PrintLCD(buff);
            }
            else    SetLED(j,0);
            sw[j] = i;
        }

        /*-----------------------------------------------*/
        /* HOSTからのシリアル入力をLCD,USBに送る */
```

```
        if( ScanSCI() ) /* SCIに受信データあり? */
        {
            i = GetSCI(); /* シリアル入力 */
            PutLCD(i); /* LCD出力 */
            buff[0] = i;
            write_buff(buff,1); /* USB出力 */
        }

        /*-----------------------------------------------*/
        /* USBからデータを受信した場合、そのままHOSTへリダイレクトする  */
        if( get_inbufflen() ) /* 受信データあり? */
        {
            /* データ取得(buffサイズは64byteまで) */
            cnt = read_buff(buff,64);
            PrintLCD("¥f"); /* LCDクリア */
            PrintLCD(buff); /* LCDへ表示 */
            PrintSCI(buff); /* シリアル出力 */
            write_buff(buff,cnt); /* USBへリダイレクト */
        }

        /*-----------------------------------------------*/
        /* 動作確認のため点滅 */
        SetLED(3,f);
        f ^= 1;
        for(i=0;i<10000;i++) {} /* 適当にウエイト */
    }
#else
    printf("END");
    /* 5秒待機 */
```

```
	SleepMSec(5000);
	return;
#endif
}

#ifdef USE_THREAD
/*
 * 擬似スレッドの擬似メソッド関数
 */
 /* public void paint(Graphics g)の代用 */
void Repaint(void)
{
#ifndef USE_BCC
	int i;
#else
	int i,j;
#endif
	Clear();
#ifndef USE_BCC
	for(i = 0; i < Cnt.cnt[0]; i++)
	{
		PrintF(Panel, " ");
	}
	PrintF(Panel, "<1>");
	PrintF(Panel, "¥n");
	for(i = 0; i < Cnt.cnt[1]; i++)
	{
		PrintF(Panel, " ");
	}
```

```
        PrintF(Panel, "<2>");
#else
        for(i = 0; i < 8; i++)
        {
                for(j = 0; j < Cnt.cnt[i]; j++)
                {
                        printf(" ");
                }
                printf("<%d>", (i + 1));
                for(j = 0; j < 13 - Cnt.cnt[i]; j++)
                {
                        printf(" ");
                }
                printf("¦");
                printf("¥n");
        }
#endif
        return;
}

/*
 * 疑似スレッドの疑似メソッド関数
 */
/* スレッドのpublic void run()の代用 */
void Run(Thread *This)
{
        int i;
        Thread *th11;
#ifdef USE_BCC
```

```
    char key = '¥0';
#else
    int j;
    char sw[4];
    /* スイッチワーク初期化 */
    sw[0] = sw[1] = sw[2] = sw[3] = 0;
#endif
    if(This->ID == 1)
    {
        Repaint();
#ifndef USE_BCC
        Cnt.cnt[0]++;
        nextRun(This, (((rand() % 9) + 10) * 100));
#else
        if(kbhit())
        {
#ifdef USE_LINUX
            key = (char) getchar();
#else
            key = (char) getche();
#endif
        }
        if(key == 'r')
        {
            Cnt.cnt[0]++;
        }
        nextRun(This, (((rand() % 9) + 10) * 30));
        while(kbhit())
```

```
            {
#ifdef USE_LINUX
                key = (char) getchar();
#else
                key = (char) getche();
#endif
            }
#endif
        }
        else if(This->ID == 2)
        {
            Repaint();
#ifndef USE_BCC
            Cnt.cnt[1]++;
            nextRun(This, (((rand() % 9) + 10) * 100));
#else
            if(kbhit())
            {
#ifdef USE_LINUX
                key = (char) getchar();
#else
                key = (char) getche();
#endif
            }
            if(key == 'l')
            {
                Cnt.cnt[1]++;
            }
            nextRun(This, (((rand() % 9) + 10) * 30));
```

```
            while(kbhit())
            {
#ifdef USE_LINUX
                key = (char) getchar();
#else
                key = (char) getche();
#endif
            }
#endif
        }
#ifdef USE_BCC
        else if(((This->ID) >= 3) && ((This->ID) <= 8))
        {
                Repaint();
                Cnt.cnt[(This->ID) - 1]++;
                nextRun(This, (((rand() % 9) + 10) * 40));
        }
#endif
        else if(This->ID == 11)
        {
            if(This->count == 1)
            {
                Clear();
                PrintF(Panel, "<1>1st         ");
                countUpNextRun(This, (1900 * 1));
            }
            else if(This->count == 2)
            {
                Clear();
```

```
            PrintF(Panel, "<1>2nd");
            PrintF(Panel, "<1>Stop   ");
            Stop(This);
        }
        else if(This->count == 3)
        {
            Clear();
            PrintF(Panel, "<1>3rd        ");
            countUpNextRun(This, (1500 * 1));
        }
        else if(This->count == 4)
        {
            Clear();
            PrintF(Panel, "<1>Stop       ");
            Stop(This);
        }
    }
    else if(This->ID == 12)
    {
        if(This->count == 1)
        {
            Clear();
            PrintF(Panel, "<2>1st        ");
            countUpNextRun(This, (1700 * 2));
        }
        else if(This->count == 2)
        {
            Clear();
            PrintF(Panel, "<2>2nd        ");
```

```
            countUpNextRun(This, (1700 * 2));
        }
        else if(This->count == 3)
        {
            Clear();
            PrintF(Panel, "<2>3rd         ");
            countUpNextRun(This, (1700 * 2));
        }
        else
        {
            Clear();
            PrintF(Panel, "<2>Stop");
            Stop(This);
            delete_(This);
        }
    }
    else if(This->ID == 13)
    {
        if(This->count == 1)
        {
            Clear();
            PrintF(Panel, "<3>1st         ");
            countUpNextRun(This, (1700 * 3));
        }
        else if(This->count == 2)
        {
            Clear();
            PrintF(Panel, "<3>2nd         ");
```

```
            countUpNextRun(This, (1700 * 3));
        }
        else if(This->count == 3)
        {
            Clear();
            PrintF(Panel, "<3>3rd         ");
            countUpNextRun(This, (1700 * 3));
        }
        else
        {
            Clear();
            PrintF(Panel, "<3>Stop");
            Stop(This);
            delete_(This);
        }
    }
    else if(This->ID == 14)
    {
        if(This->count == 1)
        {
            Clear();
            PrintF(Panel, "<4>1st         ");
            countUpNextRun(This, (1500 * 4));
        }
        else if(This->count == 2)
        {
            Clear();
            PrintF(Panel, "<4>2nd");
            countUpNextRun(This, (1500 * 4));
```

```
            PrintF(Panel, "<1>Start  ");
            th11 = Thread_Start(11);
            countUpNextRun(th11, (1500 * 1));
        }
        else if(This->count == 3)
        {
            Clear();
            PrintF(Panel, "<4>3rd         ");
            countUpNextRun(This, (1500 * 4));
        }
        else if(This->count == 4)
        {
            th11 = Thread_getThread(11);
            if(th11 != NULL)
            {
                delete_(th11);
            }

            PrintF(Panel, "<4>Sto");
            Stop(This);
            delete_(This);
        }
    }
    else if(This->ID == 19)
    {
#ifdef USE_LINUX
        nextRun(This, 4000);
#else
```

```
        nextRun(This, 1);
#endif
#ifndef USE_BCC
        /* ボタンが押された時にLCD,SCI,USBにメッセージを送る */
        for(j=0;j<4;j++)
        {
            i = GetSW(j);
            if( ((sw[j]^1) & i) ) /* sw = off->onで条件成立 */
            {
                Thread_Toggle(j + 20);
                nextRun(This, 1000);
            }
        }
#else
        key = '¥0';
        key = GetChar(key);

        if(key == '1')
        {
            Thread_Toggle(21);
        }
        else if(key == '2')
        {
            Thread_Toggle(22);
        }
        else if(key == '3')
        {
            Thread_Toggle(23);
        }
```

```
else if(key == '4')
{
	Thread_Toggle(24);
}
else if(key == '5')
{
	Thread_Toggle(25);
}
else if(key == '6')
{
	Thread_Toggle(26);
}
else if(key == '7')
{
	Thread_Toggle(27);
}
else if(key == '8')
{
	Thread_Toggle(28);
}
else if(key == '9')
{
	Thread_Toggle(29);
}
else if(key == '0')
{
	Thread_Toggle(20);
}
```

```
#endif
    }
    else if(This->ID == 20)
    {
        PrintF(Panel, "0");
        countUpNextRun(This, 2000);
    }
    else if(This->ID == 21)
    {
        PrintF(Panel, "1");
        countUpNextRun(This, 2000);
    }
    else if(This->ID == 22)
    {
        PrintF(Panel, "2");
        countUpNextRun(This, 2000);
    }
    else if(This->ID == 23)
    {
        PrintF(Panel, "3");
        countUpNextRun(This, 2000);
    }
#ifdef USE_BCC
    else if((This->ID >= 24) && (This->ID <= 29))
    {
        printf("%d", This->ID - 20);
        nextRun(This, 2000);
    }
#endif
```

```
#ifndef USE_BCC
    else if(This->ID == 30)
    {
        if(This->count == 0)
        {
            This->count++;
            PB.DR &= 0x0e;
            nextRun(This, 1000);
        }
        else if(This->count == 1)
        {
            This->count--;
            PB.DR ¦= 0x01;
            nextRun(This, 1000);
        }
    }
#endif
    else if(This->ID == 31)
    {
        if(This->count == 0)
        {
            /* 第1部分 */
#ifdef USE_BCC
            printf("¥nThread Ready GO! There are 8 cources on a race.");
            printf("¥nThere are 14 cells to a GOAL.");
            printf("¥nFor the <1> cource, You click a 'R' button.");
            printf("¥nFor the <2> cource, You click a 'L' button.");
#endif
#ifndef USE_BCC
```

```
            countUpNextRun(This, 0);
#else
            /* 5秒待機 */
            countUpNextRun(This, 5000);
#endif
        }
        else if(This->count == 1)
        {
            /* 擬似スレッド開始 */
            PrintF(Panel, "¥n");
            PrintF(Panel, "Thread Ready GO!");
            /* 2秒待機 */
            countUpNextRun(This, 2000);
        }
        else if(This->count == 2)
        {
#ifndef USE_BCC
            /* 擬似スレッドの擬似インスタンス初期化 */
            for(i = 0; i < 2; i++)
            {
                th[i] = new_Thread(i + 1);
            }
#else
            /* 擬似スレッドの擬似インスタンス初期化 */
            for(i = 0; i < 8; i++)
            {
                th[i] = new_Thread(i + 1);
            }
#endif
```

```
            countUpNextRun(This, 1);
        }
        else if(This->count == 3)
        {
#ifndef USE_BCC
            if(Cnt.cnt[0] >= 13)
            {
                i_cnt = 1;
                This->count++;
            }
            else if(Cnt.cnt[1] >= 13)
            {
                i_cnt = 2;
                This->count++;
            }
#else
            i_cnt = Cnt.cnt[0];
            j_cnt = 0;
            for(i = 1; i < 8; i++)
            {
                if(i_cnt < Cnt.cnt[i])
                {
                    i_cnt = Cnt.cnt[i];
                    j_cnt = i;
                }
            }
            if(i_cnt >= 13) This->count++;
#endif
```

```
            nextRun(This, 1);
        }
        else if(This->count == 4)
        {
            Clear();
            if(i_cnt == 1)
            {
                PrintF(Panel, "GOAL!<1>WON    ");
            }
            else if(i_cnt == 2)
            {
                PrintF(Panel, "GOAL!<2>WON    ");
            }
#ifdef USE_BCC
            else
            {
                printf("GOAL!¥n<%d>WON", (j_cnt + 1));
            }
#endif
#ifndef USE_BCC
            delete_(th[0]);
            delete_(th[1]);
#else
            for(i = 0; i < 8; i++)
            {
                delete_(th[i]);
            }
#endif
#ifdef USE_LINUX
```

```
            This->count = 15;
#else
            This->count++;
#endif
            /* 2秒待機 */
            nextRun(This, 2000);
        }
        else if(This->count == 5)
        {
            Clear();
            PrintF(Panel, "NEXT          ");
            /* 2秒待機 */
            countUpNextRun(This, 2000);
        }
        else if(This->count == 6)
        {
            Clear();
            /* 第2部分 */
            PrintF(Panel, "CountUp       ");
            /* 2秒待機 */
            countUpNextRun(This, 2000);
        }
        else if(This->count == 7)
        {
            /* 疑似スレッド開始 */
            Clear();
            /* 疑似スレッドの疑似インスタンス初期化 */
            for(i = 0; i < 4; i++)
            {
```

```
            th1[i] = new_Thread(i + 11);
        }
        countUpNextRun(This, 1);
    }
    else if(This->count == 8)
    {
        if(Thread_checkStayAnother() == 2)
        {
            This->count++;
        }
        nextRun(This, 1);
    }
    else if(This->count == 9)
    {
        /* 2秒待機 */
        countUpNextRun(This, 2000);
    }
    else if(This->count == 10)
    {
        Clear();
        PrintF(Panel, "NEXT          ");
        /* 2秒待機 */
        countUpNextRun(This, 2000);
    }
    else if(This->count == 11)
    {
        Clear();
        /* 第3部分 */
        PrintF(Panel, "Toggle        ");
```

```
        /* 2秒待機 */
        countUpNextRun(This, 2000);
    }
    else if(This->count == 12)
    {
        th19 = new_Thread(19);
        Start(th19);
        th20 = new_Thread(20);
        Start(th20);
        countUpNextRun(This, 1);
    }
    else if(This->count == 13)
    {
        if(Thread_checkStayAnother() == 3)
        {
            delete_(th19);
            This->count++;
        }
        nextRun(This, 1);
    }
    else if(This->count == 14)
    {
        countUpNextRun(This, 2000);
    }
    else if(This->count == 15)
    {
        Clear();
        PrintF(Panel, "NEXT          ");
```

```
        /* 2秒待機 */
        countUpNextRun(This, 2000);
    }
    else if(This->count == 16)
    {
        Clear();
        /* 第4部分 */
        PrintF(Panel, "Hello EV       ");
        /* 2秒待機 */
        countUpNextRun(This, 2000);
    }
    else if(This->count == 17)
    {
        /* 擬似スレッドの擬似インスタンス初期化 */
        th41 = new_Thread(41);
        th42 = new_Thread(42);
        th43 = new_Thread(43);
        th44 = new_Thread(44);

        delete_(This);

        /* LEDTEST */
        delete_(Thread_getThread(30));
    }
}
else if(This->ID == 41)
{
    nextRun(This, 100);
    OnInput(&in, This);
```

```
    }
    else if(This->ID == 42)
    {
        nextRun(This, 100);
        OnController(&cntrl, This);
    }
    else if(This->ID == 43)
    {
        nextRun(This, 110);
        OnPuls(&puls, This);
    }
    else if(This->ID == 44)
    {
        nextRun(This, 2000);
        OnSimulator(&simu, This);
    }
    return;
}

/* スレッドのコンストラクタのpublic void init()の代用 */
void Init(Thread *This)
{
    if(This->ID == 1)
    {
        Cnt.cnt[0] = 0;
        nextRun(This, (((rand() % 9) + 10) * 30));
    }
    else if(This->ID == 2)
    {
```

```
        Cnt.cnt[1] = 0;
        nextRun(This, (((rand() % 9) + 10) * 30));
    }
    else if((This->ID >= 3) && (This->ID <= 8))
    {
        Cnt.cnt[(This->ID) - 1] = 0;
        nextRun(This, (((rand() % 9) + 10) * 200));
    }
    else if(This->ID == 11)
    {
        PrintF(Panel, "<1>Init");
        countUpNextRun(This, (1500 * 1));
    }
    else if(This->ID == 12)
    {
        PrintF(Panel, "<2>Init  ");
        countUpNextRun(This, (1500 * 2));
    }
    else if(This->ID == 13)
    {
        PrintF(Panel, "¥n");
        PrintF(Panel, "<3>Init");
        countUpNextRun(This, (1500 * 3));
    }
    else if(This->ID == 14)
    {
        PrintF(Panel, "<4>Init  ");
        countUpNextRun(This, (1500 * 4));
    }
```

```
else if(This->ID == 20)
{
	Clear();
	PrintF(Panel, "<0>Init	");
	PrintF(Panel, "¥n");
	countUpNextRun(This, 2000);
}
else if(This->ID == 21)
{
	Clear();
	PrintF(Panel, "<1>Init	");
	PrintF(Panel, "¥n");
	countUpNextRun(This, 2000);
}
else if(This->ID == 22)
{
	Clear();
	PrintF(Panel, "<2>Init	");
	PrintF(Panel, "¥n");
	countUpNextRun(This, 2000);
}
else if(This->ID == 23)
{
	Clear();
	PrintF(Panel, "<3>Init	");
	PrintF(Panel, "¥n");
	countUpNextRun(This, 2000);
}
```

```
#ifdef USE_BCC
    else if((This->ID >= 24) && (This->ID <= 29))
    {
        printf("¥n<%d>Init¥n", This->ID - 20);
        nextRun(This,2000);
    }
#endif
#ifndef USE_BCC
    else if(This->ID == 30)
    {
        PB.DDR = 0xff; /* bit7..0 out */
        PB.DR ¦= 0xff;
    }
#endif
    else if(This->ID == 41)
    {
        nextRun(This, 100);
        EV_Input(&in);
    }
    else if(This->ID == 42)
    {
        nextRun(This, 100);
        EV_Controller(&cntrl, This);
    }
    else if(This->ID == 43)
    {
        nextRun(This, 110);
        EV_Puls(&puls, This);
    }
```

```
    else if(This->ID == 44)
    {
        nextRun(This, 2000);
        EV_Simulator(&simu, This);
    }
    return;
}

/* スレッドのデストラクタの代用 */
void Destroy(Thread *This)
{
    if(This->ID == 11)
    {
        Clear();
        PrintF(Panel, "<1>Destroy");
    }
    else if(This->ID == 12)
    {
        PrintF(Panel, "<2>Destro");
    }
    else if(This->ID == 13)
    {
        PrintF(Panel, "<3>Destro");
    }
    else if(This->ID == 14)
    {
        PrintF(Panel, "¥n");
        PrintF(Panel, "<4>Destroy     ");
    }
```

```
    if(This->ID == 20)
    {
        Clear();
        PrintF(Panel, "<0>Destroy     ");
        PrintF(Panel, "¥n");
    }
    else if(This->ID == 21)
    {
        Clear();
        PrintF(Panel, "<1>Destroy     ");
        PrintF(Panel, "¥n");
    }
    else if(This->ID == 22)
    {
        Clear();
        PrintF(Panel, "<2>Destroy     ");
        PrintF(Panel, "¥n");
    }
    else if(This->ID == 23)
    {
        Clear();
        PrintF(Panel, "<3>Destroy     ");
        PrintF(Panel, "¥n");
    }
#ifdef USE_BCC
    else if((This->ID >= 24) && (This->ID <= 29))
    {
        printf("¥n<%d>Destroy¥n", This->ID - 20);
    }
```

```
#endif
	return;
}
#endif

#ifndef USE_BCC
/*=====================================================================
                                LEDコントロール
---------------------------------------------------------------------
	int SetLED(int no,int onoff)

	int		no		LEDナンバー 0～3
	int		onoff		0=OFF,1=ON
	戻り値			以前のLEDの状態	(0=OFF,else=ON)

	LEDをコントロールします。
=====================================================================*/
int SetLED(int no,int onoff)
{
	int f;
	f = PB.DR&(1<<no);
	if( onoff == 0 ) PB.DR ¦= (1<<no); /* off (1) */
	else PB.DR &= 0xff^(1<<no); /* on (0) */
	return( f );
}

/*=====================================================================
                                SW状態取得
```

```
----------------------------------------------------------------
    int GetSW(int no)

    int         no              SWナンバー 0～3
    戻り値                      SWの状態(0=OFF,else=ON)

    SWの状態を取得します。
================================================================*/
int GetSW(int no)
{
    return( ((PA.DR&(1<<no))?0:1) );
}

/*================================================================
                                H8初期化
----------------------------------------------------------------
    BUSモードや、ポートの初期化

    P1    bit1  BUS       USB A0
    P3          BUS       USB D7..0
    P6 bit4     BUS       USB RD
    P6 bit5     BUS       USB WR
    P8 bit2     BUS       USB CS
    P9 bit5     BUS       USB INT(IRQ5)
    P9 bit3     BUS       RS232C
    P9 bit1     BUS       RS232C

    PA bit0..3  IN        SW0..3
    PB bit0..3  OUT       LED0..3  LCD DB4..7
```

```
    PB bit4     OUT             LCD RS
    PB bit7     OUT             LCD E
==========================================================================*/
void H8init()
{
    BSC.ABWCR = 0x06; /* 8bit BUS MODE */

    P1.DDR = 0xff; /* all OUT */
    P2.DDR = 0xff; /* all OUT */
    P2.PCR = 0x00; /* Pull up off */
    P5.DDR = 0xff; /* all OUT */
    P5.PCR = 0x00; /* Pull up off */
    P6.DDR = 0xff; /* all OUT */
    P9.DDR = 0xdf; /* Bit5 IN */
    P8.DDR = 0xff; /* all OUT */
    PA.DDR = 0xf0; /* bit7..4 out , bit3..0 in */
    PB.DDR = 0xff; /* bit7..0 out */
}
#endif
```

実行環境

```
-I"c:\borland\Bcc55\include"
-L"c:\borland\Bcc55\lib"
```

-L"c:\borland\Bcc55\Lib"

```
# makefile.mak
CC = bcc32
LL = ilink32
INCLUDE = -I"C:\borland\bcc55\Include"
LIB = -L"C:\borland\bcc55\Lib"
CFLAGS = -O2 -w -tWC -D"USE_BCC"
LFLAGS = /Tpe
TARGET = main.exe
OBJS = Panel.obj Timer.obj EV_Time.obj EV_File.obj EV_UpDown.obj EV_OpenClose.obj EV_Display.obj
EV_Input.obj EV_Controller.obj EV_Puls.obj EV_Simulator.obj main.obj
$(TARGET): $(OBJS)
	$(LL) $(LFLAGS) $(LIB) \
	$(OBJS) c0x32.obj,$(TARGET),,cw32.lib import32.lib
main.obj : main.c main.h EV_Simulator.h EV_Puls.h EV_Controller.h EV_Input.h EV_Display.h EV_OpenClose.h
EV_UpDown.h EV_File.h EV_Time.h Timer.h Panel.h C.h
	$(CC) $(CFLAGS) -c main.c
EV_Simulator.obj : EV_Simulator.c EV_Simulator.h EV_Display.h EV_File.h EV_Time.h Timer.h Panel.h C.h
	$(CC) $(CFLAGS) -c EV_Simulator.c
EV_Puls.obj : EV_Puls.c EV_Puls.h EV_File.h EV_Time.h Timer.h Panel.h C.h
	$(CC) $(CFLAGS) -c EV_Puls.c
EV_Controller.obj : EV_Controller.c EV_Controller.h EV_OpenClose.h EV_UpDown.h EV_File.h EV_Time.h
Timer.h Panel.h C.h
	$(CC) $(CFLAGS) -c EV_Controller.c
EV_Input.obj : EV_Input.c EV_Input.h EV_File.h EV_Time.h Timer.h Panel.h C.h
	$(CC) $(CFLAGS) -c EV_Input.c
EV_Display.obj : EV_Display.c EV_Display.h EV_File.h EV_Time.h Timer.h Panel.h C.h
	$(CC) $(CFLAGS) -c EV_Display.c
EV_OpenClose.obj : EV_OpenClose.c EV_OpenClose.h EV_UpDown.h EV_File.h EV_Time.h Timer.h Panel.h
C.h
	$(CC) $(CFLAGS) -c EV_OpenClose.c
EV_UpDown.obj : EV_UpDown.c EV_UpDown.h EV_File.h EV_Time.h Timer.h Panel.h C.h
	$(CC) $(CFLAGS) -c EV_UpDown.c
EV_File.obj : EV_File.c EV_File.h Panel.h C.h
	$(CC) $(CFLAGS) -c EV_File.c
EV_Time.obj : EV_Time.c EV_Time.h Timer.h Panel.h C.h
	$(CC) $(CFLAGS) -c EV_Time.c
Timer.obj : Timer.c Timer.h Panel.h C.h
	$(CC) $(CFLAGS) -c Timer.c
Panel.obj : Panel.c Panel.h C.h
	$(CC) $(CFLAGS) -c Panel.c
clean:
	del *.obj
	del main.tds
	del main.ilc
	del main.ild
	del main.ilf
	del main.ils
```

```
; asmfile.src

 .CPU 300HA
 .SECTION V,CODE,LOCATE=H'000000

; C言語の関数 を参照
 .IMPORT _main ; C言語の関数main を参照
 .IMPORT _usb_int ; C言語の関数usb_int を参照
 .IMPORT _InterruptITU0 ; C言語の関数InterruptITU0 を参照

;------------------------------------------------------------
; リセットベクタ の 転送先ラベル が _start になっています
; リセットベクタ
 .DATA.L _start

;------------------------------------------------------------
; リセットベクタ に続く1番から60番までの 割り込みベクタ
; について、使用しない 割り込みベクタ は ラベルint_error
; に転送されます
; 割り込みベクタ
; 1 Reserved
_INT_Reserved1: .DATA.L int_error
; 2 Reserved
_INT_Reserved2: .DATA.L int_error
; 3 Reserved
_INT_Reserved3: .DATA.L int_error
; 4 Reserved
_INT_Reserved4: .DATA.L int_error
```

```
; 5 Reserved
_INT_Reserved5: .DATA.L int_error
; 6 Reserved
_INT_Reserved6: .DATA.L int_error
; 7 NMI
_INT_NMI: .DATA.L int_error
; 8 TRAP
_INT_TRAP1: .DATA.L int_error
; 9 TRAP
_INT_TRAP2: .DATA.L int_error
; 10 TRAP
_INT_TRAP3: .DATA.L int_error
; 11 TRAP
_INT_TRAP4: .DATA.L int_error
; 12 IRQ0
IRQ0: .DATA.L int_error
; 13 IRQ1
IRQ1: .DATA.L int_error
; 14 IRQ2
IRQ2: .DATA.L int_error
; 15 IRQ3
IRQ3: .DATA.L int_error
; 16 IRQ4
IRQ4: .DATA.L int_error
; 17 IRQ5
IRQ5: .DATA.L usb_interrupt ; USB割り込み
; 18 Reserved
_INT_Reserved18: .DATA.L int_error
; 19 Reserved
```

```
_INT_Reserved19: .DATA.L int_error
; 20 WOVI
_INT_WOVI: .DATA.L int_error
; 21 CMI
_INT_CMI: .DATA.L int_error
; 22 Reserved
_INT_Reserved22: .DATA.L int_error
; 23 Reserved
_INT_Reserved23: .DATA.L int_error
; 24 IMIA0
_INT_IMIA0: .DATA.L int_error
; 25 IMIB0
_INT_IMIB0: .DATA.L int_error
; タイマ0割り込み は、 ラベル_ITU_OVI_0 に転送されます
; 26 OVI0
_INT_OVI0: .DATA.L _ITU_OVI_0 ; タイマ0割り込み
; 27 Reserved
_INT_Reserved27: .DATA.L int_error
; 28 IMIA1
_INT_IMIA1: .DATA.L int_error
; 29 IMIB1
_INT_IMIB1: .DATA.L int_error
; 30 OVI1
_INT_OVI1: .DATA.L int_error
; 31 Reserved
_INT_Reserved31: .DATA.L int_error
; 32 IMIA2
_INT_IMIA2: .DATA.L int_error
; 33 IMIB2
```

```
_INT_IMIB2: .DATA.L int_error
; 34 OVI2
_INT_OVI2: .DATA.L int_error
; 35 Reserved
_INT_Reserved35: .DATA.L int_error
; 36 IMIA3
_INT_IMIA3: .DATA.L int_error
; 37 IMIB3
_INT_IMIB3: .DATA.L int_error
; 38 OVI3
_INT_OVI3: .DATA.L int_error
; 39 Reserved
_INT_Reserved39: .DATA.L int_error
; 40 IMIA4
_INT_IMIA4: .DATA.L int_error
; 41 IMIB4
_INT_IMIB4: .DATA.L int_error
; 42 OVI4
_INT_OVI4: .DATA.L int_error
; 43 Reserved
_INT_Reserved43: .DATA.L int_error
; 44 DEND0A
_INT_DEND0A: .DATA.L int_error
; 45 DEND0B
_INT_DEND0B: .DATA.L int_error
; 46 DEND1A
_INT_DEND1A: .DATA.L int_error
; 47 DEND1B
_INT_DEND1B: .DATA.L int_error
```

```
; 48 Reserved
_INT_Reserved48: .DATA.L int_error
; 49 Reserved
_INT_Reserved49: .DATA.L int_error
; 50 Reserved
_INT_Reserved50: .DATA.L int_error
; 51 Reserved
_INT_Reserved51: .DATA.L int_error
; 52 ERI0
_INT_ERI0: .DATA.L int_error
; 53 RXI0
_INT_RXI0: .DATA.L int_error
; 54 TXI0
_INT_TXI0: .DATA.L int_error
; 55 TEI0
_INT_TEI0: .DATA.L int_error
; 56 ERI1
_INT_ERI1: .DATA.L int_error
; 57 RXI1
_INT_RXI1: .DATA.L int_error
; 58 TXI1
_INT_TXI1: .DATA.L int_error
; 59 TEI1
_INT_TEI1: .DATA.L int_error
; 60 ADI
_INT_ADI: .DATA.L int_error

;---------------------------------------------------------
```

```
 .SECTION P,CODE,ALIGN=2
; _start の ラベル から処理を開始
; リセットベクタ の転送先
_start:
 mov.l #H'0FFFF10,er7
; 初期化付きデータを使用する場合、RAMに転送する
; c_thread.MAP のメモリアドレス使用状況を見る
; メモリアドレスが重複するとコンパイルエラーになる
; linkfile.sub も D(99C0), C(9A00) 等必要があれば合わせる
 mov.l #H'99C0, er0 ; 転送元(99C0)
 mov.l #H'0FFE000, er1 ; 転送先
 mov.l #DATA_END, er2 ; 転送終了
init_loop:
 cmp.l er1, er2
 beq init_end
 mov.b @er0+, r3l
 mov.b r3l, @er1
 inc.l #1, er1
 bra init_loop
init_end:
; C言語の関数main を呼び出しています
; C言語の関数main は、 void main(void); という形で、
; main.c に記述があります
 jsr @_main
; 割り込み未使用
int_error:
; rte (returnと同じ意味) で終了
 rte
```

```
;---------------------------------------------------------
; USB割り込み からの転送先
usb_interrupt:
; スタック 退避
 push.l er0
 push.l er1
 push.l er2
 push.l er3
 push.l er4
 push.l er5
 push.l er6
; C言語の関数usb_int を呼び出しています
 jsr @_usb_int
; スタック 戻
 pop.l er6
 pop.l er5
 pop.l er4
 pop.l er3
 pop.l er2
 pop.l er1
 pop.l er0
; 終了
 rte

;---------------------------------------------------------
; タイマ0割り込み からの転送先
_ITU_OVI_0:
; スタック 退避
 push.l er0
```

```
 push.l er1
 push.l er2
 push.l er3
 push.l er4
 push.l er5
 push.l er6
; C言語の関数InterruptITU0 を呼び出しています
; C言語の関数InterruptITU0 は void InterruptITU0(void);
; という形で、 Timer.h Timer.c に記述があります
 jsr @_InterruptITU0
; スタック 戻
 pop.l er6
 pop.l er5
 pop.l er4
 pop.l er3
 pop.l er2
 pop.l er1
 pop.l er0
; 終了
 rte

;---------------------------------------------------------
; C言語から
; _EnableInterrupt (割り込み許可)
; _DisableInterrupt (割り込み禁止)
; を呼び出せるようにしています
; C言語の Panel.h に 外部参照プロトタイプ宣言 があります
; extern void EnableInterrupt(void);
; extern void DisableInterrupt(void);
```

```
; C言語からの呼び出し名は、
; EnableInterrupt();
; DisableInterrupt();
; です
; 割り込み許可、禁止ルーチン
 .EXPORT _EnableInterrupt,_DisableInterrupt
_EnableInterrupt:
 andc.b #H'3f,ccr
 rts
_DisableInterrupt:
 orc.b #H'c0,ccr
 rts

;------------------------------------------------------
 .SECTION D,DATA

 .SECTION B,DATA
DATA_END: .RES.W 1

 .END
```

```
OUTPUT c_thread
PRINT c_thread
INPUT asmfile, main, EV_Simulator, EV_Puls, EV_Controller, EV_Input, EV_Display, EV_OpenClose,
EV_UpDown, EV_File, EV_Time, Timer, Panel, sci, lcd, usb
LIB c:\h8\akic\c38hab
START R(0FFE000), P(200), D(99C0), C(9A00)
ROM (D, R)
EXIT
```

```
@rem build.bat
C:
set bccDir="C:\borland\bcc55\Bin"
set akih8asmDir="c:\h8\akiasm"
set akih8cDir="c:\h8\akic"
set path=%bccDir%;%path%
set path=%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.exe -cpu=300ha -include=%akih8cDir% usb.c >> error.txt
cc38h.exe -cpu=300ha -include=%akih8cDir% sci.c >> error.txt
cc38h.exe -cpu=300ha -include=%akih8cDir% lcd.c >> error.txt
cc38h.exe -cpu=300ha -include=%akih8cDir% Panel.c >> error.txt
cc38h.exe -cpu=300ha -include=%akih8cDir% Timer.c >> error.txt
cc38h.exe -cpu=300ha -include=%akih8cDir% EV_Time.c >> error.txt
cc38h.exe -cpu=300ha -include=%akih8cDir% EV_File.c >> error.txt
cc38h.exe -cpu=300ha -include=%akih8cDir% EV_UpDown.c >> error.txt
cc38h.exe -cpu=300ha -include=%akih8cDir% EV_OpenClose.c >> error.txt
cc38h.exe -cpu=300ha -include=%akih8cDir% EV_Display.c >> error.txt
cc38h.exe -cpu=300ha -include=%akih8cDir% EV_Input.c >> error.txt
cc38h.exe -cpu=300ha -include=%akih8cDir% EV_Controller.c >> error.txt
cc38h.exe -cpu=300ha -include=%akih8cDir% EV_Puls.c >> error.txt
cc38h.exe -cpu=300ha -include=%akih8cDir% EV_Simulator.c >> error.txt
cc38h.exe -cpu=300ha -include=%akih8cDir% main.c >> error.txt
a38h.exe asmfile.src >> error.txt
l38h.exe -subcommand=linkfile.sub >> error.txt
c38h.exe c_thread.abs >> error.txt
del %CurrentDir%*.obj >> error.txt
del %CurrentDir%c_thread.abs >> error.txt
error.txt
exit
```

```
#!/bin/bash
cd `dirname $0`
find $PWD -name LINUXBuild.bash
rm ./LINUXError.txt
gcc -D"USE_LINUX" -o LINUXExe main.c EV_Simulator.c EV_Puls.c EV_Controller.c EV_Input.c
EV_Display.c EV_OpenClose.c EV_UpDown.c EV_File.c EV_Time.c Timer.c Panel.c &>>./LINUXError.txt
exit
```

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

```
#!/bin/bash
cd `dirname $0`
find $PWD -name RaspberryPi3ModelBBuild.bash
rm ./RaspberryPi3ModelBError.txt
gcc -D"USE_RASPBIAN" -o RaspberryPi3ModelBExe main.c EV_Simulator.c EV_Puls.c EV_Controller.c
EV_Input.c EV_Display.c EV_OpenClose.c EV_UpDown.c EV_File.c EV_Time.c Timer.c Panel.c -
I/usr/local/include -L/usr/local/lib -lwiringPi &>>./RaspberryPi3ModelBError.txt
exit
```

```
#!/bin/bash
cd `dirname $0`
find $PWD -name RaspberryPi3ModelBStart.bash
./RaspberryPi3ModelBExe
exit
```

実行環境状態ファイル

r

yynnyynn

q

C

N

S

出力ファイル

```
 Start         Length     Name                   Class
 0001:00401000 000010C0CH _TEXT                  CODE
 0002:00412000 0000039A4H _DATA                  DATA
 0003:004159A4 000000B6CH _BSS                   BSS
 0004:00000000 0000000A4H _TLS                   TLS
```

PROGRAM NAME =

```
 1                1  ; asmfile.src
 2                2
 3                3   .CPU 300HA
 4 000000         4   .SECTION V,CODE,LOCATE=H'000000
 5                5
 6                6  ; C言語の関数 を参照
 7                7   .IMPORT _main ; C言語の関数main を参照
 8                8   .IMPORT _usb_int ; C言語の関数usb_int を参照
 9                9   .IMPORT _InterruptITU0 ; C言語の関数InterruptITU0 を参照
10               10
11               11   ;------------------------------------------------------
12               12   ; リセットベクタ の 転送先ラベル が _start になっています
13               13   ; リセットベクタ
14 000000 00000000      14   .DATA.L _start
15               15
16               16   ;------------------------------------------------------
17               17   ; リセットベクタ に続く1番から60番までの 割り込みベクタ
18               18   ; について、使用しない 割り込みベクタ は ラベルint_error
19               19   ; に転送されます
20               20   ; 割り込みベクタ
21               21   ; 1 Reserved
22 000004 00000000      22   _INT_Reserved1: .DATA.L int_error
23               23   ; 2 Reserved
24 000008 00000000      24   _INT_Reserved2: .DATA.L int_error
```

```
25                      25    ; 3 Reserved
26 00000C 00000000      26    _INT_Reserved3: .DATA.L int_error
27                      27    ; 4 Reserved
28 000010 00000000      28    _INT_Reserved4: .DATA.L int_error
29                      29    ; 5 Reserved
30 000014 00000000      30    _INT_Reserved5: .DATA.L int_error
31                      31    ; 6 Reserved
32 000018 00000000      32    _INT_Reserved6: .DATA.L int_error
33                      33    ; 7 NMI
34 00001C 00000000      34    _INT_NMI: .DATA.L int_error
35                      35    ; 8 TRAP
36 000020 00000000      36    _INT_TRAP1: .DATA.L int_error
37                      37    ; 9 TRAP
38 000024 00000000      38    _INT_TRAP2: .DATA.L int_error
39                      39    ; 10 TRAP
40 000028 00000000      40    _INT_TRAP3: .DATA.L int_error
41                      41    ; 11 TRAP
42 00002C 00000000      42    _INT_TRAP4: .DATA.L int_error
43                      43    ; 12 IRQ0
44 000030 00000000      44    IRQ0: .DATA.L int_error
45                      45    ; 13 IRQ1
46 000034 00000000      46    IRQ1: .DATA.L int_error
47                      47    ; 14 IRQ2
48 000038 00000000      48    IRQ2: .DATA.L int_error
49                      49    ; 15 IRQ3
50 00003C 00000000      50    IRQ3: .DATA.L int_error
51                      51    ; 16 IRQ4
52 000040 00000000      52    IRQ4: .DATA.L int_error
53                      53    ; 17 IRQ5
```

```
 54 000044 00000000          54    IRQ5: .DATA.L usb_interrupt ; USB割り込み
 55                          55    ; 18 Reserved
 56 000048 00000000          56    _INT_Reserved18: .DATA.L int_error
 57                          57    ; 19 Reserved
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***     04/22/18 20:24:55
 PAGE    2
 PROGRAM NAME =


 58 00004C 00000000          58    _INT_Reserved19: .DATA.L int_error
 59                          59    ; 20 WOVI
 60 000050 00000000          60    _INT_WOVI: .DATA.L int_error
 61                          61    ; 21 CMI
 62 000054 00000000          62    _INT_CMI: .DATA.L int_error
 63                          63    ; 22 Reserved
 64 000058 00000000          64    _INT_Reserved22: .DATA.L int_error
 65                          65    ; 23 Reserved
 66 00005C 00000000          66    _INT_Reserved23: .DATA.L int_error
 67                          67    ; 24 IMIA0
 68 000060 00000000          68    _INT_IMIA0: .DATA.L int_error
 69                          69    ; 25 IMIB0
 70 000064 00000000          70    _INT_IMIB0: .DATA.L int_error
 71                          71    ; タイマ0割り込み は、 ラベル_ITU_OVI_0 に転送されます
 72                          72    ; 26 OVI0
 73 000068 00000000          73    _INT_OVI0: .DATA.L _ITU_OVI_0 ; タイマ0割り込み
 74                          74    ; 27 Reserved
 75 00006C 00000000          75    _INT_Reserved27: .DATA.L int_error
 76                          76    ; 28 IMIA1
 77 000070 00000000          77    _INT_IMIA1: .DATA.L int_error
 78                          78    ; 29 IMIB1
```

```
 79 000074 00000000          79    _INT_IMIB1: .DATA.L int_error
 80                          80    ; 30 OVI1
 81 000078 00000000          81    _INT_OVI1: .DATA.L int_error
 82                          82    ; 31 Reserved
 83 00007C 00000000          83    _INT_Reserved31: .DATA.L int_error
 84                          84    ; 32 IMIA2
 85 000080 00000000          85    _INT_IMIA2: .DATA.L int_error
 86                          86    ; 33 IMIB2
 87 000084 00000000          87    _INT_IMIB2: .DATA.L int_error
 88                          88    ; 34 OVI2
 89 000088 00000000          89    _INT_OVI2: .DATA.L int_error
 90                          90    ; 35 Reserved
 91 00008C 00000000          91    _INT_Reserved35: .DATA.L int_error
 92                          92    ; 36 IMIA3
 93 000090 00000000          93    _INT_IMIA3: .DATA.L int_error
 94                          94    ; 37 IMIB3
 95 000094 00000000          95    _INT_IMIB3: .DATA.L int_error
 96                          96    ; 38 OVI3
 97 000098 00000000          97    _INT_OVI3: .DATA.L int_error
 98                          98    ; 39 Reserved
 99 00009C 00000000          99    _INT_Reserved39: .DATA.L int_error
100                         100    ; 40 IMIA4
101 0000A0 00000000         101    _INT_IMIA4: .DATA.L int_error
102                         102    ; 41 IMIB4
103 0000A4 00000000         103    _INT_IMIB4: .DATA.L int_error
104                         104    ; 42 OVI4
105 0000A8 00000000         105    _INT_OVI4: .DATA.L int_error
106                         106    ; 43 Reserved
107 0000AC 00000000         107    _INT_Reserved43: .DATA.L int_error
```

```
 108                    108    ; 44 DEND0A
 109 0000B0 00000000        109    _INT_DEND0A: .DATA.L int_error
 110                    110    ; 45 DEND0B
 111 0000B4 00000000        111    _INT_DEND0B: .DATA.L int_error
 112                    112    ; 46 DEND1A
 113 0000B8 00000000        113    _INT_DEND1A: .DATA.L int_error
 114                    114    ; 47 DEND1B
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***     04/22/18 20:24:55
PAGE    3
PROGRAM NAME =

 115 0000BC 00000000        115    _INT_DEND1B: .DATA.L int_error
 116                    116    ; 48 Reserved
 117 0000C0 00000000        117    _INT_Reserved48: .DATA.L int_error
 118                    118    ; 49 Reserved
 119 0000C4 00000000        119    _INT_Reserved49: .DATA.L int_error
 120                    120    ; 50 Reserved
 121 0000C8 00000000        121    _INT_Reserved50: .DATA.L int_error
 122                    122    ; 51 Reserved
 123 0000CC 00000000        123    _INT_Reserved51: .DATA.L int_error
 124                    124    ; 52 ERI0
 125 0000D0 00000000        125    _INT_ERI0: .DATA.L int_error
 126                    126    ; 53 RXI0
 127 0000D4 00000000        127    _INT_RXI0: .DATA.L int_error
 128                    128    ; 54 TXI0
 129 0000D8 00000000        129    _INT_TXI0: .DATA.L int_error
 130                    130    ; 55 TEI0
 131 0000DC 00000000        131    _INT_TEI0: .DATA.L int_error
```

```
132                     132    ; 56 ERI1
133 0000E0 00000000         133    _INT_ERI1: .DATA.L int_error
134                     134    ; 57 RXI1
135 0000E4 00000000         135    _INT_RXI1: .DATA.L int_error
136                     136    ; 58 TXI1
137 0000E8 00000000         137    _INT_TXI1: .DATA.L int_error
138                     138    ; 59 TEI1
139 0000EC 00000000         139    _INT_TEI1: .DATA.L int_error
140                     140    ; 60 ADI
141 0000F0 00000000         141    _INT_ADI: .DATA.L int_error
142                     142
143                     143    ;------------------------------------------------------
144 000000                  144    .SECTION P,CODE,ALIGN=2
145                     145    ; _start の ラベル から処理を開始
146                     146    ; リセットベクタ の転送先
147 000000                  147    _start:
148 000000 7A0700FFFF10        148    mov.l #H'0FFFF10,er7
149                     149    ; 初期化付きデータを使用する場合、RAMに転送する
150                     150    ; c_thread.MAP のメモリアドレス使用状況を見る
151                     151    ; メモリアドレスが重複するとコンパイルエラーになる
152                     152    ; linkfile.sub も D(99C0), C(9A00) 等必要があれば合わせる
153 000006 7A00000099C0      153    mov.l #H'99C0, er0 ; 転送元(99C0)
154 00000C 7A0100FFE000        154    mov.l #H'0FFE000, er1 ; 転送先
155 000012 7A0200000000        155    mov.l #DATA_END, er2 ; 転送終了
156 000018                  156    init_loop:
157 000018 1F92             157    cmp.l er1, er2
158 00001A 58700008         158    beq init_end
159 00001E 6C0B             159    mov.b @er0+, r3l
160 000020 689B             160    mov.b r3l, @er1
```

```
 161 000022 0B71            161     inc.l #1, er1
 162 000024 40F2            162     bra init_loop
 163 000026                 163     init_end:
 164                        164     ; C言語の関数main を呼び出しています
 165                        165     ; C言語の関数main は、 void main(void); という形で、
 166                        166     ; main.c に記述があります
 167 000026 5E000000        167     jsr @_main
 168                        168     ; 割り込み未使用
 169 00002A                 169     int_error:
 170                        170     ; rte (returnと同じ意味) で終了
 171 00002A 5670            171     rte
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***     04/22/18 20:24:55
PAGE    4
PROGRAM NAME =


 172                        172
 173                        173     ;---------------------------------------------------------
 174                        174     ; USB割り込み からの転送先
 175 00002C                 175     usb_interrupt:
 176                        176     ; スタック 退避
 177 00002C 01006DF0        177     push.l er0
 178 000030 01006DF1        178     push.l er1
 179 000034 01006DF2        179     push.l er2
 180 000038 01006DF3        180     push.l er3
 181 00003C 01006DF4        181     push.l er4
 182 000040 01006DF5        182     push.l er5
 183 000044 01006DF6        183     push.l er6
 184                        184     ; C言語の関数usb_int を呼び出しています
 185 000048 5E000000        185     jsr @_usb_int
```

```
186                     186    ;スタック 戻
187 00004C 01006D76         187     pop.l er6
188 000050 01006D75         188     pop.l er5
189 000054 01006D74         189     pop.l er4
190 000058 01006D73         190     pop.l er3
191 00005C 01006D72         191     pop.l er2
192 000060 01006D71         192     pop.l er1
193 000064 01006D70         193     pop.l er0
194                     194    ;終了
195 000068 5670             195     rte
196                     196
197                     197    ;-------------------------------------------------------
198                     198    ;タイマ0割り込み からの転送先
199 00006A                  199    _ITU_OVI_0:
200                     200    ;スタック 退避
201 00006A 01006DF0         201     push.l er0
202 00006E 01006DF1         202     push.l er1
203 000072 01006DF2         203     push.l er2
204 000076 01006DF3         204     push.l er3
205 00007A 01006DF4         205     push.l er4
206 00007E 01006DF5         206     push.l er5
207 000082 01006DF6         207     push.l er6
208                     208    ;C言語の関数InterruptITU0 を呼び出しています
209                     209    ;C言語の関数InterruptITU0 は void InterruptITU0(void);
210                     210    ;という形で、 Timer.h Timer.c に記述があります
211 000086 5E000000         211     jsr @_InterruptITU0
212                     212    ;スタック 戻
213 00008A 01006D76         213     pop.l er6
214 00008E 01006D75         214     pop.l er5
```

```
 215 000092 01006D74          215     pop.l er4
 216 000096 01006D73          216     pop.l er3
 217 00009A 01006D72          217     pop.l er2
 218 00009E 01006D71          218     pop.l er1
 219 0000A2 01006D70          219     pop.l er0
 220                  220    ; 終了
 221 0000A6 5670              221     rte
 222                  222
 223                  223    ;--------------------------------------------------------
 224                  224    ; C言語から
 225                  225    ; _EnableInterrupt (割り込み許可)
 226                  226    ; _DisableInterrupt (割り込み禁止)
 227                  227    ; を呼び出せるようにしています
 228                  228    ; C言語の Panel.h に 外部参照プロトタイプ宣言 があります
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***     04/22/18 20:24:55
PAGE    5
PROGRAM NAME =


 229                  229    ; extern void EnableInterrupt(void);
 230                  230    ; extern void DisableInterrupt(void);
 231                  231    ; C言語からの呼び出し名は、
 232                  232    ; EnableInterrupt();
 233                  233    ; DisableInterrupt();
 234                  234    ; です
 235                  235    ; 割り込み許可、禁止ルーチン
 236                  236     .EXPORT _EnableInterrupt,_DisableInterrupt
 237 0000A8             237     _EnableInterrupt:
 238 0000A8 063F              238     andc.b #H'3f,ccr
```

```
 239 0000AA 5470            239     rts
 240 0000AC                 240    _DisableInterrupt:
 241 0000AC 04C0            241     orc.b #H'c0,ccr
 242 0000AE 5470            242     rts
 243                        243
 244                        244    ;--------------------------------------------------------
 245 000000                 245     .SECTION D,DATA
 246                        246
 247 000000                 247     .SECTION B,DATA
 248 000000 00000002        248     DATA_END: .RES.W 1
 249                        249
 250                        250     .END
 *****TOTAL ERRORS      0
 *****TOTAL WARNINGS    0
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***     04/22/18 20:24:55
PAGE    6

*** CROSS REFERENCE LIST

NAME                    SECTION  ATTR VALUE          SEQUENCE

B                       B       SCT  00000000    247*
D                       D       SCT  00000000    245*
DATA_END                        B          00000000    155   248*
IRQ0                     V          00000030    44*
IRQ1                     V          00000034    46*
IRQ2                     V          00000038    48*
IRQ3                     V          0000003C    50*
IRQ4                     V          00000040    52*
```

```
IRQ5                    V       00000044    54*
P                       P       SCT  00000000   144*
V                       V       SCT  00000000     4*
_DisableInterrupt       P       EXPT 000000AC   236   240*
_EnableInterrupt        P       EXPT 000000A8   236   237*
_INT_ADI                V       000000F0   141*
_INT_CMI                V       00000054    62*
_INT_DEND0A             V       000000B0   109*
_INT_DEND0B             V       000000B4   111*
_INT_DEND1A             V       000000B8   113*
_INT_DEND1B             V       000000BC   115*
_INT_ERI0               V       000000D0   125*
_INT_ERI1               V       000000E0   133*
_INT_IMIA0              V       00000060    68*
_INT_IMIA1              V       00000070    77*
_INT_IMIA2              V       00000080    85*
_INT_IMIA3              V       00000090    93*
_INT_IMIA4              V       000000A0   101*
_INT_IMIB0              V       00000064    70*
_INT_IMIB1              V       00000074    79*
_INT_IMIB2              V       00000084    87*
_INT_IMIB3              V       00000094    95*
_INT_IMIB4              V       000000A4   103*
_INT_NMI                V       0000001C    34*
_INT_OVI0               V       00000068    73*
_INT_OVI1               V       00000078    81*
_INT_OVI2               V       00000088    89*
_INT_OVI3               V       00000098    97*
_INT_OVI4               V       000000A8   105*
```

```
_INT_RXI0               V       000000D4  127*
_INT_RXI1               V       000000E4  135*
_INT_Reserved1          V       00000004  22*
_INT_Reserved18         V       00000048  56*
_INT_Reserved19         V       0000004C  58*
_INT_Reserved2          V       00000008  24*
_INT_Reserved22         V       00000058  64*
_INT_Reserved23         V       0000005C  66*
_INT_Reserved27         V       0000006C  75*
_INT_Reserved3          V       0000000C  26*
_INT_Reserved31         V       0000007C  83*
_INT_Reserved35         V       0000008C  91*
_INT_Reserved39         V       0000009C  99*
_INT_Reserved4          V       00000010  28*
_INT_Reserved43         V       000000AC  107*
_INT_Reserved48         V       000000C0  117*
_INT_Reserved49         V       000000C4  119*
```


*** CROSS REFERENCE LIST

```
NAME                SECTION  ATTR VALUE     SEQUENCE

_INT_Reserved5          V       00000014  30*
_INT_Reserved50         V       000000C8  121*
_INT_Reserved51         V       000000CC  123*
_INT_Reserved6          V       00000018  32*
_INT_TEI0               V       000000DC  131*
```

```
_INT_TEI1               V       000000EC   139*
_INT_TRAP1              V       00000020    36*
_INT_TRAP2              V       00000024    38*
_INT_TRAP3              V       00000028    40*
_INT_TRAP4              V       0000002C    42*
_INT_TXI0               V       000000D8   129*
_INT_TXI1               V       000000E8   137*
_INT_WOVI               V       00000050    60*
_ITU_OVI_0              P       0000006A    73  199*
_InterruptITU0                IMPT 00000000     9  211
_main                         IMPT 00000000     7  167
_start                  P       00000000    14  147*
_usb_int                      IMPT 00000000     8  185
init_end                P       00000026   158  163*
init_loop               P       00000018   156*  162
int_error               P       0000002A    22   24   26   28   30   32   34   36   38   40   42   44
                                    46   48   50   52   56   58   60   62   64   66   68   70
                                    75   77   79   81   83   85   87   89   91   93   95   97
                                    99  101  103  105  107  109  111  113  115  117  119  121
                                   123  125  127  129  131  133  135  137  139  141  169*
usb_interrupt           P       0000002C    54  175*
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***     04/22/18 20:24:55
PAGE    8

*** SECTION DATA LIST

SECTION                 ATTRIBUTE   SIZE        START
```

V ABS-CODE 00000F4 000000

P REL-CODE 00000B0

D REL-DATA 0000000

B REL-DATA 0000002

S00E0000635F7468726561644D4F54C8
S1130000000002000000022A0000022A0000022A67
S1130010000022A0000022A0000022A0000022A2D
S1130020000022A0000022A0000022A0000022A1D
S10B00300000022A0000022A6D
S1130038000022A0000022A0000022A0000022C03
S1130048000022A0000022A0000022A0000022AF5
S1130058000022A0000022A0000022A0000022AE5
S10B00680000026A0000022AF5
S1130070000022A0000022A0000022A0000022ACD
S1130080000022A0000022A0000022A0000022ABD
S1130090000022A0000022A0000022A0000022AAD
S10B00A00000022A0000022AFD
S11300A80000022A0000022A0000022A0000022A95
S11300B80000022A0000022A0000022A0000022A85
S11300C80000022A0000022A0000022A0000022A75
S10B00D80000022A0000022AC5
S11300E00000022A0000022A0000022A0000022A5D
S10700F00000022ADD
S11302007A0700FFFF107A00000099C07A0100FF0F
S1130210E0007A0200FFE0101F92587000086C0B98
S1130220689B0B7140F25E0002B0567001006DF0E6
S113023001006DF101006DF201006DF301006DF439
S113024001006DF501006DF65E004F5C01006D76F7
S113025001006D7501006D7401006D7301006D7215
S113026001006D7101006D70567001006DF00100A9
S11302706DF101006DF201006DF301006DF40100F9
S11302806DF501006DF65E00493C01006D760100DD
S11302906D7501006D7401006D7301006D720100D5
S11302A06D7101006D705670063F547004C0547038
S11302B05E0060727A370000000A790B000119337F
S11302C00FF47A0600FFE1E019550B5579257FFFFE
S11302D04DF85C000FD85E004AD85E004C580D38CC
S11302E00D305C000F540D380DB05C000F4C0D3811
S11302F0790000025C000F420D38790000035C00B6
S11303000F385E004DFE7FF472507FF570505E0033
S113031002A80D3228F117506DF07A0000009A0000
S113032001006DF05E004B280B970B877A000000ED
S11303309A0F01006DF05E004D2C0B9718886EC864
S113034000036EC800026EC8000168C87A0000FF8F
S1130350E04A5E003ED25E0049687A00000007D0A2
S11303605E0044307900001E5E00452601006FF0F8
S1130370000047900001F5E0045260F8501006F70A1
S113038000045E0046F20FD05E0046F201006B20CF
S113039000009C8A01006DF001006B2000009C8628
S11303A001006DF05E0049640B970B975E00473EBA
S11303B00D0046D85E0049827A0100009A1B0D3079
S11303C05E0049AA7A00000007D05E0044305E0058
S11303D0498219550D505C000EAE0D0D0D5117F1EC
S11303E00AC16819D90117D1660147540DB80D50D8
S11303F05C000E460D5009B06DF07A0000009A2C97
S113040001006DF00FE05E0059140B970B8701009C
S11304106DF67A0000009A3101006DF05E004B2802
S11304200B970B970FE05E0059B809B00D010FE071
S11304305E00579001006DF65E004D2C0B9740084F
S11304400D380D505C000DF20DD017F50FC10AD118
S113045068980B557925000458D0FF785E004B1837

S10804600D0047165ECC
S1130465004B0817D00D055E004CDC68ED0DB10F90
S1130475E05E0057905E0058B20D004738790100E1
S1130485400FE05E0058540D057A0100009A3501CE
S1130495006DF15E004D2C0B9701006DF65E004D6E
S11304A52C0B9701006DF65E004B280B970D510F32
S11304B5E05E0057900D28790000035C000D760D72
S11304C520795000010D0219550B55792527104D3B
S11304D5F85A0003D254705E0060727A0500FFE09B
S11304E51219665E00498219EE400E7A0100009AE0
S11304F5370D605E0049AA0B5E69501D0E4DEC7AFF
S11005050100009A390D605E0049AA7A01D9
S113051200009A3D0D605E0049AA19EE400E7A0171
S113052200009A370D605E0049AA0B5E6F5000020D
S11305321D0E4DEA7A0100009A3F0D605E0049AA42
S11305425E00605054705E0060727A370000000AE9
S11305527A03000000017A04000007D019550F86C0
S113056218886EF800036EF800026EF8000168F84E
S1130572696079200001463 65C00FF5E7A0000FF65
S1130582E01269010B5169815E0058DA17F07902B2
S1130592000901D053207918000A790000645280BF
S11305A217F00F810FE05E0044C25A000EA4696087
S11305B27920000246365C00FF207A0000FFE01437
S11305C269010B5169815E0058DA17F0790200095B
S11305D201D053207918000A79000064528017F081
S11305E20F810FE05E0044C25A000EA469607920B5
S11305F2000B586000B601006F6000127A20000001
S1080602000146205E2B
S10806070049827A01A5
S113060C00009A430D505E0049AA7A010000076C62
S113061C0FE05E0045065A000EA401006F60001245
S113062C7A200000000246265E0049827A0100000F
S113063C9A540D505E0049AA7A0100009A5B0D5042
S113064C5E0049AA0FE05E0047205A000EA4010089
S113065C6F6000127A2000000003461E5E00498280
S113066C7A0100009A660D505E0049AA7A010000D7
S113067C05DC0FE05E004506402401006F600012AC
S111068C7A200000000446165E0049827A01BF
S113069A00009A770D505E0049AA0FE05E004720DA
S11306AA5A000EA469607920000C586000A8010062
S11306BA6F6000127A200000000146205E00498222
S11306CA7A0100009A880D505E0049AA7A01000057
S11306DA0D480FE05E0045065A000EA401006F6044
S11306EA00127A200000000246205E0049827A0145
S11306FA00009A990D505E0049AA7A0100000D483C
S113070A0FE05E0045065A000EA401006F60001256
S113071A7A2000000003461E5E0049827A01000027
S108072A9AAA0D505EC8
S113072F0049AA7A0100000D480FE05E004506401C
S113073F1C5E0049827A0100009ABB0D505E00498E
S113074FAA0FE05E0047200FE05E0046BE5A000E80
S113075FA469607920000D586000A801006F600044
S113076F127A200000000146205E0049827A0100C0
S113077F009AC30D505E0049AA7A01000013EC0FD3
S113078FE05E0045065A000EA401006F6000127A66
S113079F200000000246205E0049827A0100009A81
S11307AFD40D505E0049AA7A01000013EC0FE05EEE

S10707BF0045065A8E
S11307C3000EA401006F6000127A200000000346AC
S11307D31E5E0049827A0100009AE50D505E0049CE
S11307E3AA7A01000013EC0FE05E004506401C5E8D
S11307F30049827A0100009AF60D505E0049AA0F60
S1130803E05E0047200FE05E0046BE5A000EA46977
S1130813607920000E586000E401006F6000127AD3
S113082320000000146205E0049827A0100009AFD
S1130833FE0D505E0049AA7A01000017700FE05EB7
S11308430045065A000EA401006F6000127A2000CF
S10E085300000246405E0049827A016B
S113085E00009B0F0D505E0049AA7A01000017702D
S113086E0FE05E0045067A0100009B160D505E00F8
S113087E49AA7900000B5E0047B80F867A01000083
S113088E05DC5E0045065A000EA401006F600012DF
S113089E7A2000000003461E5E0049827A010000A2
S11308AE9B210D505E0049AA7A01000017700FE0DC
S11308BE5E004506403801006F6000127A2000008A
S11308CE0004462A7900000B5E00477E0F8247061E
S11308DE0FA05E0046BE7A0100009B320D505E00F3
S11208EE49AA0FE05E0047200FE05E0046BE5AA6
S11308FD000EA469607920001346440FB10FE05E2A
S113090D0044C219DD0DD05C0009700DD117F10F34
S113091DF20A92682ADA0117D2660247160DD079C8
S113092D1000145E0047F67A01000003E80FE05E45
S113093D0044C20B5D792D00044DCA5A000EA46903
S113094D607920001446187A0100009B390D505E22
S113095D0049AA0FC10FE05E0045065A000EA469B7
S113096D607920001546187A0100009B3B0D505EFF
S113097D0049AA0FC10FE05E0045065A000EA46997
S113098D607920001646187A0100009B3D0D505EDC
S10F099D0049AA0FC10FE05E0045065A96
S11309A9000EA469607920001746187A0100009B9C
S11309B93F0D505E0049AA0FC10FE05E0045065A7C
S11309C9000EA469607920001E465E01006F600075
S11309D912462401006F6000120B7001006FE000E2
S11309E91228D6E80E38D67A01000003E80FE05E34
S11309F90044C25A000EA401006F6000127A20005D
S1130A0900000158600494010 06F6000121B70011B
S1130A19006FE000127FD670007A01000003E80F2F
S1130A29E05E0044C25A000EA469607920001F5891
S1130A396003E601006F600012460C1A910FE05E35
S1130A490045065A000EA401006F6000127A2000C7
S1130A5900000146247A0100009A3D0D505E0049C9
S1130A69AA7A0100009B410D505E0049AA0FC10FEC
S1090A79E05E0045065A91
S1130A7F000EA401006F6000127A200000000246EE
S1130A8F3019DD0DD00B505E0045260DD217F21035
S1130A9F321032010078A06BA000FFE01A0B5D79D2
S1130AAF2D00024DDE0FB10FE05E0045065A000E1A
S1130ABF1E01006F6000127A200000000346566B80
S1130ACF2000FFE0127920000D4D1A790000016B11
S1130ADFA000FFE01601006F6000120B7001006FA2
S1130AEFE0001240246B2000FFE0147920000D4D2D
S1130AFF18790000026BA000FFE01601006F600081
S1130B0F120B7001006FE000120FB10FE05E004493
S1130B1FC25A000E1E01006F6000127A20000000FF

S1130B2F0446685E0049826B2000FFE016792000BF
S1130B3F01460E7A0100009B520D505E0049AA40F8
S1060B4F186B20FD
S1130B5200FFE01679200002460C7A0100009B6335
S1130B620D505E0049AA01006B2000FFE01A5E00EF
S1130B7246BE01006B2000FFE01E5E0046BE010080
S1130B826F6000120B7001006FE000120FC10FE0E3
S1130B925E0044C25A000E1E01006F6000127A20EA
S1130BA200000005461C5E0049827A0100009A1B80
S1130BB20D505E0049AA0FC10FE05E0045065A00C0
S1130BC20E1E01006F6000127A200000006461C10
S1090BD25E0049827A0176
S1130BD800009B740D505E0049AA0FC10FE05E0030
S1130BE845065A000E1E01006F6000127A200000AD
S1130BF8000746365E00498219DD0DD07910000BD7
S1130C085E0045260DD217F210321032010078A08B
S1130C186BA000FFE0220B5D792D00044DDC0FB1C2
S1130C280FE05E0045065A000E1E01006F600012B9
S1130C387A200000000846245E0047587920000205
S1130C48460E01006F6000120B7001006FE0001286
S1130C580FB10FE05E0044C25A000E1E01006F6020
S1130C6800127A2000000009460C0FC10FE05E0055
S1130C7845065A000E1E01006F6000127A2000001C
S1130C88000A461C5E0049827A0100009A1B0D5037
S1040C985EFA
S1130C990049AA0FC10FE05E0045065A000E1E0166
S1130CA9006F6000127A200000000B461C5E0049A9
S1130CB9827A0100009B850D505E0049AA0FC10F7E
S1130CC9E05E0045065A000E1E01006F6000127AAD
S1130CD9200000000C4634790000135E004526010C
S1130CE9006BA000FFE0325E0046F2790000145E5B
S1130CF900452601006BA000FFE0365E0046F20FB7
S1130D09B10FE05E0045065A000E1E01006F600038
S1130D19127A200000000D46305E00475879200002
S10A0D2903461A01006B20D1
S1130D3000FFE0325E0046BE01006F6000120B70E0
S1130D4001006FE000120FB10FE05E0044C25A00D1
S1130D500E1E01006F6000127A200000000E460C88
S1130D600FC10FE05E0045065A000E1E01006F60C2
S1130D7000127A200000000F461C5E0049827A01AF
S1130D8000009A1B0D505E0049AA0FC10FE05E00E0
S1130D9045065A000E1E01006F6000127A20000003
S1130DA00010461A5E0049827A0100009B960D509E
S1130DB05E0049AA0FC10FE05E00450640600100D6
S1130DC06F6000127A2000000114652790000295A
S10B0DD05E00452601006BA043
S1130DD800FFE03A7900002A5E00452601006BA077
S1130DE800FFE03E7900002B5E00452601006BA062
S1130DF800FFE0427900002C5E00452601006BA04D
S1130E0800FFE0460FE05E0046BE7900001E5E006C
S1130E18477E5E0046BE5A000EA469607920002909
S1130E28461A7A01000000640FE05E0044C20FE135
S1130E387A0000FFE04E5E00262A40606960792050
S1130E48002A461A7A01000000640FE05E0044C2DB
S10C0E580FE17A0000FFE08C5E5B
S1130E61001CB6403E69607920002B461A7A0100C6
S1130E7100006E0FE05E0044C20FE17A0000FFE163

S1130E81845E0019F8401C69607920002C46140F18
S1130E91C10FE05E0044C20FE17A0000FFE1985EFA
S1130EA10013C47A170000000A5E00605054705E9C
S1130EB10060720F86790400097A0500FFE0126968
S1130EC160792000014626190069D05E0058DA17BF
S1130ED1F001D053407918000A7900001E5280179F
S1130EE1F00F810FE05E0044C25A00113E69607940
S1130EF1200002462819006FD000025E0058DA175D
S1130F01F001D053407918000A7900001E5280176E
S1130F11F00F810FE05E0044C25A00113E696C7903
S1130F212C00034D36792C00084E3069601B501795
S1130F31F010300A85190069D05E0058DA17F00104
S1130F41D053407918000A790000C8528017F00F76
S1070F51810FE05ECB
S1070F550044C25A35
S1130F5900113E7A04000049AA19556960792000F5
S1130F690B461A7A0100009BA70D505D407A0100D8
S1130F790005DC0FE05E0045065A00113E69607901
S1130F8920000C461A7A0100009BAF0D505D407A90
S1130F990100000BB80FE05E0045065A00113E69D7
S1130FA9607920000D46247A0100009A3D0D505DB9
S1130FB9407A0100009BB90D505D407A0100001190
S1130FC9940FE05E0045065A00113E6960792000DE
S1130FD90E461A7A0100009BC10D505D407A01004B
S1130FE90017700FE05E0045065A00113E7A0300B0
S10F0FF90007D069607920001446245ED4
S11310050049827A0100009BCB0D505D407A0100B7
S1131015009A3D0D505D400FB10FE05E0045065A45
S113102500113E69607920001546245E0049827AE5
S11310350100009BDC0D505D407A0100009A3D0DD7
S1131045505D400FB10FE05E0045065A00113E6941
S1131055607920001646245E0049827A0100009BD0
S1131065ED0D505D407A0100009A3D0D505D400F36
S1131075B10FE05E0045065A00113E696079200014
S10710851746245E85
S11310890049827A0100009BFE0D505D407A010000
S1131099009A3D0D505D400FB10FE05E0045065AC1
S11310A900113E69607920001E460AF8FF38D438DA
S11310B9D65A00113E69607920002946187A010041
S11310C90000640FE05E0044C27A0000FFE04E5E58
S11310D90024FE406069607920002A461A7A0100DB
S11310E90000640FE05E0044C20FE17A0000FFE0F4
S11310F98C5E001B16403E69607920002B461A7AE4
S1131109010000006E0FE05E0044C20FE17A0000A7
S1131119FFE1845E0019BA401C69607920002C46FE
S1101129140FB10FE05E0044C20FE17A0025
S113113600FFE1985E0012DA5E00605054705E00B4
S113114660727A04000049AA19660F8569507920EE
S1131156000B46105E0049827A0100009C0F0D6069
S11311665D40404469507920000C460C7A0100002A
S11311769C1A0D605D40403069507920000D460C85
S11311867A0100009C240D605D40401C6950792063
S1131196000E46147A0100009A3D0D605D407A0107
S11311A600009C2E0D605D40695079200014461A9C
S11311B65E0049827A0100009C3F0D605D407A0122
S11311C600009A3D0D605D4040646950792000152A
S10B11D6461A5E0049827A010A

S11311DE00009C500D605D407A0100009A3D0D6049
S11311EE5D4040426950792000164 61A5E004982DE
S11311FE7A0100009C610D605D407A0100009A3D0A
S113120E0D605D4040206955792500174 6185E0034
S113121E49827A0100009C730D605D407A010000E3
S113122E9A3D0D605D405E00605054706DF66DF535
S113123E0D067909000128D617500D950CE91A09E8
S113124E4B04101540F866050D8846121A0E4B0412
S113125E101940F80D9029D6148939D640141A0E58
S113126E4B04101940F8795900FF0D9029D61689B1
S113127E39D60D506D756D7654706DF60D0628D3F7
S113128E1750790100011A0E4B04101140F8661025
S113129E4704191140047901000 10D106D76547045
S11312AEF80638ECF8FF38C038C1188838D8F8FF7C
S11312BE38C8188838DBF8FF38C9F8DF38D0F8FF9C
S10912CE38CDF8F038D121
S10912D4F8FF38D454704A
S1139A00435055204D4F444520253032580A000C11
S1139A1052656164792133303532004E4558542004
S1139A202020202020202020202020000737725754F
S1139A300025730A000C0020003C313E000A003C64
S1139A40323E003C313E3173742020202020202000
S1139A50202020003C313E326E64003C313E537482
S1139A606F70202020003C313E3372642020202080
S1139A70202020202020003C313E53746F70202092
S1139A802020202020202020003C323E3173742020EF
S1139A902020202020202020003C323E326E6420F3
S1139AA0202020202020202020003C323E337264DE
S1139AB02020202020202020202020003C323E5374F0
S1139AC06F70003C333E3173742020202020200F
S1139AD0202020003C333E326E64202020202020B2
S1139AE020202020003C333E33726420202020209D
S1099AF0202020202000CD
S1139AF63C333E53746F70003C343E317374202004
S1139B062020202020202020003C343E326E64009A
S1139B163C313E53746172742020003C343E3372F0
S1139B2664202020202020202020202020003C343E5387
S1139B36746F00300031003200330054687265617F
S1139B4664205265616479204 74F2100474F414C99
S1139B56213C313E574F4E202020202000474F41C5
S1139B664C213C323E574F4E202020202000436F8D
S1139B76756E74557020202020202020202020000544C
S1139B866F67676C652020202020202020202020020007E
S1139B9648656C6C6F2045562020202020202020200D
S1139BA6003C313E496E6974003C323E496E69742D
S1139BB62020003C333E496E6974003C343E496EB6
S1139BC669742020003C303E496E697420202020B1
S1139BD62020202020003C313E496E69742020203D
S1099BE6202020202020B6
S1139BEC003C323E496E6974202020202020202026
S1139BFC20003C333E496E697420202020202020 2015
S1139C0C2020003C313E44657374726F79003C3202
S1139C1C3E44657374726F003C333E4465737472D7
S1139C2C6F003C343E44657374726F79202020209E
S1139C3C2020003C303E44657374726F79202020E1
S1139C4C202020003C313E44657374726F792020D0
S1139C5C20202020003C323E44657374726F7920BF

S1139C6C202020202020003C333E44657374726F07
S1139C7C79202020202020200000415312D0000006
S1059C8C0000D3
S11312DA5E0060720F860F957A00000000200AE014
S11312EA0FD15E0042C67A00000000200AE05E00C9
S11312FA430A01006FE600027A00000000060AE0D2
S113130A01006FE000087A000000000C0AE0010007
S113131A6FE0000E7A00000000120AE001006FE09D
S113132A001C18886EE8001A7A0500003F267A0026
S113133A000000380AE05D507A000000003C0AE031
S113134A5D507A00000000400AE05D507A00000018
S113135A00440AE05D5001006F60000201006DF075
S113136A7A00000000400AE07A0100009C8E5E00C9
S113137A3FBA0B9779200001473A790000096DF0CB
S113138A01006F60001C01006DF07A000000004448
S113139A0AE07A0100009C995E0040140B970B87C0
S11313AA79200001470E7A01000000120AE16868F9
S11313BA5E002E205E00605054705E0060720F86DD
S10913CA0F9501006F60A6
S11313D0000E01006DF07A000000003C0AE07A0183
S11313E000009CA45E003FBA0B976E68000CA871C6
S11313F0460E5E0049820FD05E0046BE5A00179427
S113140001006F60000201006DF07A0000000040EF
S11314100AE07A0100009C8E5E003FBA0B977900C8
S113142000096DF001006F60001C01006DF07A008F
S10B1430000000440AE07A0108
S113143800009C995E0040140B970B8701006F60B6
S11314480008 01006DF07A00000000380AE07A0114
S113145800009CB15E003FBA0B976E680006A87344
S1131468461CF84E6DF07A000000003C0AE07A0151
S113147800009CA45E003F300B875A0017866868FB
S1131488A87546366E680012A879587002D06E683F
S11314980013A879460AF86E6EE800135A00176617
S11314A86E680014A86E470E6E680015A86E46068F
S11314B8F8796EE800155A0017666868A855462E2D
S11314C86E680012A879587002946E680013A879A0
S11314D85870028A6E680014A86E4608F8796EE898
S11314E8001440066E680015A86E5A0017666868EF
S11314F8A86A46466E680012A879460AF86E6EE82E
S113150800125A0017666E680013A879460AF86E27
S11315186EE800135A0017666E680014A86E460832
S1091528F8796EE80014DF
S113152E400E6E680015A86E4606F8796EE8001533
S113153E5A0017666868A86446366E680015A8795F
S113154E587002146E680014A879460AF86E6EE895
S113155E00145A0017666E680013A86E470E6E6865
S110156E0012A86E4606F8796EE800125AC6
S113157B0017666868A844462E6E680015A879584C
S113158B7001D86E680014A879587001CE6E68008C
S113159B13A86E4608F8796EE8001340066E6800D0
S11315AB12A86E5A0017666868A86B46466E6800E9
S11315BB15A879460AF86E6EE800155A0017666E81
S11315CB680014A879460AF86E6EE800145A0017DF
S11315DB666E680013A86E4608F8796EE800134030
S11315EB0E6E680012A86E4606F8796EE800125A62
S11315FB0017666868A86F46366E680016A8795898
S113160B7001586E680017A879460AF86E6EE800E9

S113161B175A0017666E680018A86E470E6E68009F
S113162B19A86E4606F8796EE800195A0017666812
S113163B68A84F462E6E680016A8795870011C6E69
S113164B680017A879587001126E680018A86E46C7
S113165B08F8796EE8001840066E680019A86E5AF0
S109166B0017666868A881
S11316716846466E680016A879460AF86E6EE80059
S1131681165A0017666E680017A879460AF86E6E37
S1131691E800175A0017666E680018A86E4608F826
S11316A1796EE80018400E6E680019A86E4606F8B8
S11316B1796EE800195A0017666868A86346346EA4
S11316C1680019A8795870009C6E680018A87946BB
S11316D10AF86E6EE800185A0017666E680017A8BC
S11316E16E470E6E680016A86E4606F8796EE8001E
S11316F11640726868A84346286E680019A879479E
S1131701646E680018A879475C6E680017A86E4676
S113171108F8796EE8001740066E680016A86E4057
S1131721446868A874463E6E680019A8794608F8AB
S11317316E6EE80019402E6E680018A8794608F805
S11317416E6EE80018401E6E680017A86E4608F812
S1131751796EE80017400E6E680016A86E4606F80B
S1081761796EE800169B
S11317667A00000000120AE001006DF07A00000022
S10E177600440AE07A0100009C995E29
S1131781003F820B977A01000000120AE168685E4C
S10C1791002E205E00605054702C
S1139C8E4D6F746F722E74787400004C696D697425
S1139C9E2E7478740000436F6D6D616E642E74784C
S1129CAE7400005361666574792E747874000036
S113179A0D0046067FD67200400A792000014604EE
S11317AA7FD670000D8846067FD67210400A7928C4
S11317BA000146047FD670100D1146067FD67220AB
S11317CA400A7921000146047FD670200D99460606
S11317DA7FD67230400A7929000146047FD67030D9
S11317EA7FD6724054707FD672007FD672107FD62E
S11317FA72207FD672307FD6704054705E0060725A
S113180A0F850D1C0D94790E000119660DC017F092
S113181A01006F5100121F81461E0D690D610D688B
S113182A0D405C00FF6A01006F5000120B7001004B
S113183A6FD000120D6E402E0B5C0DC017F0010025
S113184A6F5100121F81461E0D69790100010D684F
S113185A0D405C00FF3A01006F5000120B7001004B
S113186A6FD000120D6E0DE05E00605054705E0082
S113187A60720F850D1C0D94790E000119660DC057
S109188A17F001006F518D
S113189000121F81461E0D690D610D480D605C002D
S11318A0FEF801006F5000120B7001006FD00012A0
S11318B00D6E402E0B5C0DC017F001006F5100122E
S11318C01F81461E790900010D610D480D605C0002
S11318D0FEC801006F5000120B7001006FD00012A0
S11318E00D6E0DE05E00605054705E0060720F85F7
S11318F00F966F79001819110FE05C00FF080D00B7
S1131900587000B06F79001A790100020FE05C0093
S1131910FEF40D005870009C6F79001C79010004DF
S11319200FE05C00FEE00D00587000886F79001E28
S1131930790100060FE05C00FECC0D0047766F795D
S113194000207901000 80FE05C00FF2C0D004764C4

S11319506F7900227901000A0FE05C00FF1A0D0085
S113196047526F7900247901000C0FE05C00FF08F7
S11319700D0047406F7900267901000E0FE05C00EF
S1091980FEF60D00472EE8
S11319865C00FE661A8001006FE000126E580006C6
S1131996A871461A7FD672007FD672107FD6722040
S11319A67FD672307FD672400FE05E0046BE5E0081
S11319B66050547001006DF60F8601006FE6000259
S11319C67A00000000060AE001006FE000087A00D2
S11319D6000000100AE05E003F267A000000000CBB
S11319E60AE05E003F26F8FF38D438D601006D764C
S11319F654705E0060720F850F9401006F400012F1
S1131A0646307A000000000C0AD001006F5100082E
S1131A165E0040DC01006F50000201006DF07A00A9
S1131A26000000100AD07A0100009CBE5E003FBA97
S1131A360B9719EE790600016858A843587000AC55
S1131A46A8444758A84F4770A8554738A863477C0A
S1131A56A8644740A868474CA86A471CA86B472C4C
S1131A66A86F474AA873470CA8744756A875470E2C
S1091A765A001B106DFE77
S1131A7C40026DF66DFE400A6DFE6DF640046DF688
S1131A8C6DF66DFE401E6DFE6DFE6DF640166DF629
S1131A9C6DFE6DF6400E6DFE6DF66DF640066DF641
S1131AAC6DF66DF66DFE40266DFE6DFE6DFE6DF6EC
S1131ABC401C6DF66DFE6DFE6DF640126DFE6DF6FF
S1131ACC6DFE6DF640086DF66DF66DFE6DF66DF6FA
S1131ADC6DF66DF66DFE0FC10FD05C00FE007A172C
S1131AEC00000010401E6DFE6DFE6DF66DF66DF67A
S1131AFC6DF66DF66DFE0FC10FD05C00FDE07A172D
S10D1B0C000000105E0060505470EA
S10E9CBE4D6F746F722E7478740000F9
S1131B165E0060720F860F9501006FE600240100D8
S1131B266F6000245E00368A7A00000000280AE00F
S1131B365E0036FE7A00000000300AE05E00395E81
S1131B467A00000000380AE05E003BBE7A0000001F
S1131B56004C0AE001006FE000705E002E227A005E
S1131B660000000740AE05E002E967A000000007CF6
S1131B760AE05E0030F67A00000000840AE05E00A8
S1131B8633567A00000000A20AE00FD15E0042C677
S1131B967A0500003F267A00000000E00AE05D5067
S1131BA67A00000000E40AE05D507A00000000E8D5
S1131BB60AE05D507A00000000EC0AE05D507A000E
S1131BC6000000F00AE05D507A00000000F40AE02D
S1131BD65D507A00000000BA0AE001006FE000BC25
S1131BE67A00000000C00AE001006FE000CA7A0034
S1131BF6000000CE0AE001006FE000D07A0000008A
S1071C0600D40AE019
S1131C0A01006FE000D67A00000000DA0AE0010062
S1131C1A6FE000DC7A00000000F40AE0F9735E006A
S1131C2A41FC7A00000000380AE001006F610024D9
S1131C3A5E003BD87A00000000840AE001006F616D
S1131C4A00705E00337E790000096DF001006F6059
S1131C5A00CA01006DF07A00000000E40AE07A018C
S1131C6A00009CCA5E0040140B970B877A000000A1
S1131C7A00E80AE0F94E5E00412879200001472670
S1131C8A7A00000000EC0AE0F9635E0040987920CC
S1131C9A000147127A00000000F00AE0F94E5E00E4

S1131CAA41B8792000015E00605054705E00607292
S1131CBA790D0001F463FC4E19550F860F93010049
S1131CCA6F6000BC01006DF07A00000000E00AE0DA
S1051CDA7A018A
S1131CDC00009CD55E003FBA0B9779200001587029
S1131CEC079E790000096DF001006F6000CA0100C6
S1131CFC6DF07A00000000E40AE07A0100009CCA4F
S1131D0C5E0040140B970B877A00000000E80AE092
S1131D1C01006F6100D05E0040DC79200001587037
S1131D2C075E7A00000000EC0AE001006F6100D648
S1131D3C5E00404C79200001587007447A00000083
S1131D4C00F00AE001006F6100DC5E00416C792059
S1131D5C00015870072A6E6800CEA8485870051009
S1131D6CA8594762A86358700458A8645870022293
S1131D7CA868587005C4A86F58700398A8714718C1
S1131D8CA8725870067EA8734726A875475EA87973
S1131D9C587001D85A0024107A00000000F40AE0AD
S1131DACF9735E0041FC0FB05E0046BE5A0024106E
S1131DBC7A00000000A20AE00D515E00438E5A0027
S1091DCC24105A0024104C
S1131DD201006F60002401006F00000C690079208C
S1131DE20001461201006F60007001006F00000CD9
S1131DF2690158700061801006F60002401006F002A
S1131E020014690079200001464E01006F600070E2
S1131E1201006F000014690079200001586003 4A31
S1131E227A00000000EC0AE00C495E0040987A0058
S1131E32000000A20AE00DD15E00438E7A0000008A
S1131E4200E80AE00CC95E0041285E0049827A017B
S1131E5200009CE15A0023BE01006F600070010084
S1131E626F00000C69007920000146767A000000B9
S1131E7200A20AE00D515E00438E7A00000000A228
S1131E820AE05E00430A7A00000000F00AE00CC98F
S1131E925E0041B87A00000000380AE07A37000099
S1131EA200140FF17A02000000145E00601A7A0037
S1131EB2000000280AE07A37000000080FF17A02D6
S1131EC2000000085E00601A01006F6100BC01009F
S1131ED26F6000245E003E427A170000001C5A0025
S1131EE2241001006F60007001006F00000C690193
S1131EF246667A00000000A20AE00D515E00438E9E
S1131F027A00000000A20AE05E00430A7A000000A1
S1131F1200840AE07A370000001E0FF17A02000003
S1131F22001E5E00601A7A000000007C0AE07A3725
S1131F32000000080FF17A02000000085E00601A38
S1091F4201006F6100BC09
S1131F4801006F6000705E0036427A1700000026B9
S1131F58401A7A00000000A20AE00D515E00438E89
S1131F687A00000000A20AE05E00430A5A00241027
S1131F7801006F60002401006F00001469007920DC
S1131F880001461201006F60007001006F00000C31
S1131F9869015870047201006F60002401006F002A
S1131FA8000C690079200001465001006F60007041
S1131FB801006F000014690079200000146387A0097
S1131FC8000000EC0AE00C495E0040987A0000002B
S10A1FD800A20AE00DD15E37
S1131FDF00438E7A00000000E80AE00CC95E00415E
S1131FEF285E0049827A0100009CE15A0023BE5A01
S1131FFF00216C01006F60007001006F00000C691D

S113200F007920000146767A00000000A20AE00D55
S113201F515E00438E7A00000000A20AE05E004387
S113202F0A7A00000000F00AE00CC95E0041B87A9A
S113203F00000000380AE07A37000000140FF17A2D
S113204F02000000145E00601A7A00000000300ADC
S113205FE07A37000000080FF17A02000000085EF3
S113206F00601A01006F6100BC01006F6000245E05
S113207F003E8A7A170000001C5A00241001006FDB
S113208F60007001006F00000C690146667A000062
S113209F0000A20AE00D515E00438E7A000000009B
S11320AFA20AE05E00430A7A00000000840AE07A85
S11320BF370000001E0FF17A020000001E5E006061
S10620CF1A7A0077
S11320D20000007C0AE07A37000000080FF17A0260
S11320E2000000085E00601A01006F6100BC01007D
S11320F26F6000705E0036427A1700000026401AB5
S10E21027A00000000A20AE00D515E0D
S113210D00438E7A00000000A20AE05E00430A5AE3
S113211D00241001006F60007001006F000014694E
S113212D007920000146387A00000000EC0AE00C2B
S113213D495E0040987A00000000A20AE00DD15ECE
S113214D00438E7A00000000E80AE00CC95E0041EE
S113215D285E0049827A0100009CE15A0023BE7A71
S113216D00000000A20AE00D515E00438E7A0000CC
S113217D0000A20AE05E00430A7A00000000840A10
S113218DE07A370000001E0FF17A020000001E5E98
S113219D00601A7A00000000740AE07A370000002C
S11321AD080FF17A02000000085E00601A01006F4B
S11321BD6100BC01006F6000705E0035FA5A0023A8
S11321CD3C7A00000000A20AE00D515E00438E7AB6
S11321DD00000000A20AE05E00430A01006F6000E8
S11321ED7001006F00000C69007920000146387AF8
S10821FD00000000F0EA
S11322020AE00CC95E0041B87A00000000E80AE067
S10622120CC95E93
S11322150041287A00000000EC0AE00C495E00400A
S1132225985E0049827A0100009CE15A0023BE7A38
S113223500000000840AE07A370000001E0FF17ADF
S113224502000000 1E5E00601A7A000000007C0A8E
S1132255E07A37000000080FF17A02000000085EFB
S113226500601A01006F6100BC01006F6000705EC1
S11322750036425A00233C01006F60002401006FC1
S1132285000014690079200001586001 7E7A00007E
S11322950000A20AE00D515E00438E7A00000000A3
S11322A5A20AE05E00430A01006F60007001006F3F
S11322B500000C69007920000146387A000000000F
S11322C5E80AE00CC95E0041287A00000000F00A24
S11322D5E00CC95E0041B87A00000000EC0AE00C8E
S11322E5495E0040985E0049827A0100009CE15AEC
S11322F50023BE7A00000000840AE07A370000005C
S10823051E0FF17A0236
S113230A0000001E5E00601A7A000000007C0AE0EA
S112231A7A37000000080FF17A02000000085E16
S113232900601A01006F6100BC01006F6000705EFC
S11323390036427A17000000265A00241001006F64
S1132349600024 01006F00000C690079200001582 6
S11323596000B47A00000000A20AE00D515E004358

S11323698E7A00000000A20AE05E00430A01006FB2
S1132379600007001006F00000C69007920000146BC
S11323893C7A00000000E80AE00CC95E0041287AA3
S113239900000000F00AE00CC95E0041B87A0000B1
S11323A90000EC0AE00C495E0040985E0049827A1D
S11323B90100009CE10D505E0049AA404A7A0000E1
S11323C90000840AE07A370000001E0FF17A020048
S11323D900001E5E00601A7A000000007C0AE07AA1
S11323E937000000080FF17A02000000085E006060
S11323F91A01006F6100BC01006F6000705E003656
S1132409427A17000000267A00000000A20AE05E63
S10824190043287920B7
S113241E000A4D6A01006F60007001006F00001426
S113242E69007920000146567A00000000A20AE0F6
S113243E5E0043C27920000146447A00000000A2E8
S113244E0AE00D515E00438E7A00000000A20AE0FE
S113245E5E00430A7A00000000F00AE0F96F5E00A6
S110246E41B87A00000000EC0AE00CC95EE2
S113247B0040987A00000000E80AE00C495E004136
S10A248B285E00605054704D
S1139CCA4C696D69742E7478740000536166657407
S1139CDA792E747874000048656C6C6F20455620A1
S10B9CEA20202020202020008F
S11324925E0060721B970FF40C8D18886EC80003E0
S11324A26EC800026EC8000168C819660D605E003E
S11324B212880D0E0D6117F10AC16819D90117D1DE
S11324C2660147260D600C004620A800470EA801AE
S11324D2470EA802470EA803470E400EFD75400A99
S11324E2FD644006FD634002FD710B56792600042C
S11324F24DBA0CD80B975E00605054705E00607248
S11325020F8618886EE800067A00000000120AE0BF
S113251201006FE000147A00000000180AE00100D5
S11325226FE0002218886EE8002001006FE60002C7
S11325327A00000000070AE001006FE000087A0059
S11325420000000C0AE001006FE0000E7A050000B3
S11325523F267A00000000260AE05D507A00000060
S1132562002A0AE05D507A000000002E0AE05D5066
S11325727A00000000320AE05D507A000000003A5F
S10725820AE05D50BB
S113258601006F60000201006DF07A000000002672
S11325960AE07A0100009CF25E003FBA0B977920AD
S11325A60001477A7A000000002E0AE001006F61FD
S11325B600085E00404C7920000147627A00000063
S11325C600320AE001006F61000E5E00416C792063
S11325D60001474A01006F60001401006DF07A00A4
S11325E60000003A0AE07A0100009CFE5E003FBA52
S11325F60B977920000147267900000096DF0010049
S11326066F60002201006DF07A00000000360AE0D8
S11326167A0100009D095E0040140B970B875E004C
S1132626605054705E006072FC74F568FD4E0F8650
S11326360F937A01000000060AE17A000000002ADF
S11326460AE05E0040DC79200001587007766E6868
S113265600065C00FE366EE8000601006F600002AD
S113266601006DF07A00000000260AE07A010000FE
S10926769CF25E003FBA76
S113267C0B977920000158700744 7A000000002E54
S113268C0AE001006F6100085E00404C79200001F4

S113269C5870072A7A00000000320AE001006F61CB
S10626AC000E5EBC
S11326AF00416C7920000158700710010006F600022
S11326BF1401006DF07A000000003A0AE07A01007D
S11326CF009CFE5E003FBA0B9779200001587006FD
S11326DFEA790000096DF001006F60002201006DBF
S11326EFF07A00000000360AE07A0100009D095ECF
S11326FF0040140B970B876E680006A87158700083
S113270F8EA872474CA873586000ACF8736DF07ABB
S113271F00000000260AE07A0100009D145E003FCE
S113272F300B877A000000003A0AE0F9735E00412C
S113273FFCF84E6EE800067A000000002A0AE0F962
S113274F4E5E0041287A000000002E0AE0F94E4049
S113275F360C586DF07A00000000260AE07A01006B
S113276F009D145E003F300B87F84E6EE800067A2B
S113277F000000002A0AE0F94E5E0041287A0000AB
S113278F00002E0AE0F9635E0040985A002DCA7AC2
S108279F000000002A08
S11327A40AE06E6900065E0041287A0100009D1F5D
S10E27B4790000015E0049AA0FB05E2F
S11327BF0046BE5A002DCA6E680007A8635860040E
S11327CFBEF46F19DD6E680006A8485870041CA884
S11327DF59587002F2A86358700080A864587001AA
S11327EF94A86858700376A86F4710A8755870009F
S11327FFC8A8795870023A5A002DCA6E68001CA8EF
S113280F79470E7A00000000320AE00C495E00415E
S113281FB86868A868586003B0F85968E86DF07A2B
S113282F00000000260AE07A0100009D145E003FBD
S113283F300B876E680012A873461C6868A8594648
S113284F166E68001FA86E460E7A000000003A0A43
S113285FE00C595E0041FC5A002BD86E68001CA88F
S113286F79470E7A00000000320AE00C495E0041FE
S113287FB86868A868463EF85968E86DF07A0000B2
S113288F0000260AE07A0100009D145E003F300B22
S113289F876E680012A873461C6868A85946166E9F
S10928AF68001CA86E4640
S11328B50E7A000000003A0AE00CC95E0041FC5A9A
S11328C5002BD86E68001CA879470E7A000000001B
S11328D5320AE00C495E0041B86868A86858600090
S11328E59AF85968E86DF07A00000000260AE07A44
S11328F50100009D145E003F300B876E680012A82F
S1132905734626686 8A85946206E680018A8794654
S1132915186E68001CA86E46107A000000003A0A7B
S1072925E00CC95E98
S11329290041FC40526E680012A873461E6868A8ED
S1132939594618 6E68001BA86E46107A00000000FD
S11329493A0AE0F96A5E0041FC402C6E680012A85D
S1132959734624686 8A859461E6E68001BA8794601
S1132969166E68001FA86E460E7A000000003A0A28
S1132979E00C595E0041FC5A002BD86E68001CA874
S113298979470E7A00000000320AE00C495E0041E3
S1132999B86868A8685860009AF85968E86DF07AC9
S11329A900000000260AE07A0100009D145E003F42
S11329B9300B876E680012A87346266868A85946C3
S11329C9206E68001BA87946186E68001CA86E461D
S11329D9107A000000003A0AE00CC95E0041FC408D
S11329E9526E680012A873461E6868A85946186E85

S11329F9680018A86E46107A000000003A0AE0F948
S1132A096B5E0041FC402C6E680012A87346246873
S1082A1968A859461EE8
S1132A1E6E680018A87946166E68001FA86E460EDB
S1132A2E7A000000003A0AE00C595E0041FC5A009D
S1132A3E2BD87A03000000180AE36838A8794712E6
S1132A4E6E380004A879470A7A0100009D245A00C3
S1132A5E2C086868A868466CF85968E86DF07A0027
S1132A6E000000260AE07A0100009D145E003F304C
S1132A7E0B876E680012A873461E6868A85946181D
S1132A8E6E680018A86E46107A000000003A0AE03D
S1132A9EF96B5E0041FC402C6E680012A87346244D
S1132AAE6868A859461E6E680018A87946166E689F
S1132ABE001FA86E460E7A000000003A0AE00C5979
S1082ACE5E0041FC5A0B
S1132AD3002BD87A03000000180AE36E380003A81A
S1132AE37947126E380004A879470A7A0100009DDA
S1132AF33E5A002C086868A868466CF85968E86D64
S1132B03F07A00000000260AE07A0100009D145EBB
S1132B13003F300B876E680012A873461E6868A8CF
S1132B235946186E68001BA86E46107A0000000011
S1132B333A0AE0F96A5E0041FC402C6E680012A871
S1132B437346246868A859461E6E68001BA8794615
S1132B53166E68001FA86E460E7A000000003A0A3C
S1132B63E00C595E0041FC406C6E680018A879477D
S1132B730A7A0100009D245A002C086E68001CA8E1
S1132B8379470E7A00000000320AE00C495E0041E7
S1132B93B86868A868463EF85968E86DF07A00009B
S1132BA30000260AE07A0100009D145E003F300B0B
S1132BB3876E680012A873461C6868A85946166E88
S1092BC368001CA86E4629
S1132BC90E7A000000003A0AE00CC95E0041FC7A63
S1132BD9000000002A0AE06E6900065E0041287AB7
S1132BE9000000002E0AE00CD95E0040985A002D1F
S10E2BF9CA6E68001BA87947107A0120
S1132C0400009D3E0DD05E0049AA5A002DCA6E688D
S1132C14001CA879470E7A00000000320AE00C4930
S1132C245E0041B86868A868463EF85968E86DF0E4
S1132C347A00000000260AE07A0100009D145E0079
S1132C443F300B876E680012A873461C6868A85946
S1132C5446166E68001CA86E460E7A000000003A01
S1132C640AE00CC95E0041FC7A000000002A0AE075
S1132C746E6900065E0041287A000000002E0AE017
S1132C840CD95E0040985A002DCA6E68000CA86FD8
S1132C94586001327A03000000180AE36838A879FF
S1132CA45860008C6E380007A879587000826E68EB
S1132CB40006A864470CA86F4708A87947045A007C
S1132CC42DCA6868A8684646F85968E86DF07A0022
S1132CD4000000260AE07A0100009D145E003F30E4
S1132CE40B876E680012A87346246868A859461EA9
S1092CF46E680018A879C8
S1132CFA46166E68001FA86E460E7A000000003A58
S1132D0A0AE00C595E0041FC7A000000002A0AE03E
S1132D1A6E6900065E0041287A00000000320AE06C
S10A2D2A0CD95E0041B85A09
S1132D31002DCA7A03000000180AE36E380003A8C5
S1132D4179586000846E380007A879477C6E680063

S1132D5106A859470AA86F4706A8754702406A683B
S1132D6168A8684646F85968E86DF07A00000000E3
S1132D71260AE07A0100009D145E003F300B876E46
S1132D81680012A87346246868A859461E6E680035
S1132D911BA87946166E68001FA86E460E7A0000BE
S1132DA100003A0AE00C595E0041FC7A0000000081
S1132DB12A0AE06E6900065E0041287A00000000DD
S1122DC1320AE00CD95E0041B85E0060505470D6
S1139CF25361666574792E74787400004D6F746FC6
S1139D02722E74787400004C696D69742E747874C1
S1139D1200005361666574792E7478740051554955
S1139D2254000A41204261736B65742069736E2784
S1139D32742031737420466C6F6F72000A412042A3
S1139D4261736B65742069736E277420326E6420AD
S1099D52466C6F6F720006
S1132DD05E0060727A05000049AA790600027A0152
S1132DE000009D580D605D507A0100009D870D60C5
S1132DF05D507A0100009DA80D605D507A010000CE
S1132E009DD40D605D507A0100009E020D605D50FF
S1132E107A0100009E0E0D605D505E00605054709C
S1052E2040AEBF
S1139D580A5550203D202775272C20444F574E2065
S1139D683D202764272C204F50454E203D20276F48
S1139D78272C20434C4F5345203D20276327000AB7
S1139D88454D455247454E4359203D202773272CBF
S1139D98205245434F56455259203D2027722700EC
S1139DA80A31737420466C6F6F722043414C4C2008
S1139DB83D202779272C20326E6420466C6F6F7202
S1139DC82043414C4C203D20275927000A31737406
S1139DD820466C6F6F7220434C4F5345203D20271C
S1139DE868272C20326E6420466C6F6F7220434CB8
S1139DF84F5345203D20274827000A5155495420F1
S1139E083D20277127000A434F4D4D414E443E00E4
S1132E2201006DF60F86190069E06FE000026FE0A2
S1132E3200046FE0000601006FE600087A0000005C
S1132E4200020AE001006FE0000C7A0000000004B7
S1132E520AE001006FE000107A00000000060AE0B9
S1132E6201006FE0001419006FE000187A000000FF
S1132E7200180AE001006FE0001A19006FE0001E5B
S1132E827A000000001E0AE001006FE0002001004A
S1132E926D76547001006DF60F865E003F267A0050
S1132EA2000000040AE05E003F2601006D765470C4
S1132EB25E0060720F850F9601006F7400180100A7
S1132EC26F60001469007920000146520100 6F60AF
S1132ED2001A6901462801006F66001A7900000191
S1132EE269E07A00000000040AD0F9735E0041FC35
S1132EF27A0100009E18790000015E0049AA684821
S1132F02A859586001E8F87268C86DF07A010000A8
S1082F129E1D0FD05EBF
S1132F17003F300B875A0030F0FB6801006F6000F9
S1132F2710690079200001466E01006F60001A19CD
S1132F3711698101006F6000206901462C01006F50
S1132F47660020790000 0169E07A00000000040AA6
S1132F57D0F96F5E0041FC7A0100009E29790000D9
S1132F67015E0049AA5A0030F06848A85958600121
S1132F77787A00000000040AD00CB95E0041FCF81F
S1132F877268C86DF07A0100009E1D0FD05E003F86

S1132F97300B875A0030F001006F60000869014663
S1132FA76E01006F60001A1911698101006F6000DB
S1132FB7206901462C01006F660020790000016932
S10F2FC7E07A00000000040AD0F94F5E1D
S1132FD30041FC7A0100009E2E790000015E004946
S1132FE3AA5A0030F06848A859586001007A0000D3
S1132FF30000040AD00CB95E0041FCF87268C86D86
S1133003F07A0100009E1D0FD05E003F300B875AFC
S11330130030F001006F60000C6901466C01006F22
S113302360001A1911698101006F6000206901466C
S11330332C01006F6600207900000169E07A00002B
S11330430000040AD0F96F5E0041FC7A0100009E80
S1133053297900000015E0049AA5A0030F06848A8A4
S1133063595860000887A00000000040AD00CB95E46
S11330730041FCF87268C86DF07A0100009E1D0FD1
S1133083D05E003F300B87406401006F60000C6922
S113309300792000014656010 06F60001A19116977
S11330A38101006F60002069014628010 06F6600FB
S11330B3207900000169E07A00000000040AD00CC3
S10830C3B95E0041FCB1
S10530C87A0188
S11330CA00009E3A790000015E0049AA6848A8599F
S11330DA4614F87268C86DF07A0100009E1D0FD07D
S11330EA5E003F300B875E006050547001006DF63E
S11330FA0F865E003F267A00000000040AE05E00A5
S113310A3F2601006D7654705E0060720F850F963C
S113311A01006F74001801006F60000C69007920C8
S113312A0001465201006F60001A69014628010036
S113313A6F66001A7900000169E07A000000000452
S113314A0AD0F9735E0041FC7A0100009E187900E7
S113315A00015E0049AA6848A859586001E8F87254
S113316A68C86DF07A0100009E1D0FD05E003F30E3
S113317A0B875A003350FB7401006F600008690023
S113318A79200001466E01006F60001A19116981E6
S113319A01006F6000206901462C01006F66002060
S11331AA7900000169E07A00000000040AD0F9639B
S10931BA5E0041FC7A01F6
S11031C000009E45790000015E0049AA5AF7
S11331CD0033506848A859586001787A0000000010
S11331DD040AD00CB95E0041FCF87268C86DF07A30
S11331ED0100009E1D0FD05E003F300B875A003348
S11331FD5001006F6000106901466E01006F6000A1
S113320D1A1911698101006F6000206901462C01B3
S113321D006F6600207900000169E07A000000006C
S113322D040AD0F9435E0041FC7A0100009E4B79FC
S113323D0000015E0049AA5A0033506848A8595846
S113324D6001007A00000000040AD00CB95E004151
S113325DFCF87268C86DF07A0100009E1D0FD05EF8
S113326D003F300B875A00335001006F6000146923
S113327D01466C01006F60001A1911698101006F1D
S113328D6000206901462C01006F66002079000063
S113329D0169E07A00000000040AD0F9635E004181
S11332ADFC7A0100009E45790000015E0049AA5A8F
S11332BD0033506848A859586000887A0000000010
S11332CD040AD00CB95E0041FCF87268C86DF07A3F
S11332DD0100009E1D0FD05E003F300B874064013F
S11332ED006F60001469007920000146560 1006FDC

S11332FD60001A1911698101006F60002069014690
S113330D2801006F6600207900000169E07A000052
S113331D0000040AD00CB95E0041FC7A0100009E46
S113332D58790000015E0049AA6848A8594614F867
S113333D7268C86DF07A0100009E1D0FD05E003FCC
S113334D300B875E006050547001006DF60F867A66
S113335D00000000040AE001006FE000067A00009F
S113336D00000E0AE001006FE0001801006D7654B5
S113337D705E0060720F860F957A000000000E0AD2
S113338DE001006FE000187A000000000A0AE00176
S113339D006F6100185E0042826E680012A87946C4
S10933AD06790000014057
S11333B30219006FE0001C7A04000043E06F600011
S11333C31C6DF001006F51002001006F50000C5D74
S11333D3400B876E680013A8794606790000014005
S11333E30219006FE0001C6DF001006F5100200112
S11333F3006F5000085D400B876E680014A8794680
S1133403067900000140021 9006FE0001C6DF00112
S1133413006F51002001006F5000105D400B876E59
S1133423680015A879460679000001400219006F68
S1133433E0001C6F66001C6DF601006F5100200154
S1133443006F5000145D400B875E00605054705E44
S1133453006 0720F860F957A000000000E0AE001E8
S1133463006FE000187A000000000A0AE001006F11
S1133473610 185E0042826E680012A879460679DD
S113348300000140021 9006FE0001C7A04000043AE
S1133493E06F60001C6DF001006F51002001006FAD
S10834A350000C5D4028
S11334A80B876E680013A8794606790000014002 6D
S11334B819006FE0001C6DF001006F51002001003E
S11334C86F5000085D400B876E680014A8794606A4
S11334D879000001400219006FE0001C6DF0010043
S11334E86F51002001006F5000105D400B876E681C
S11334F80015A879460679000001400219006FE01B
S1133508001C6F66001C6DF601006F51002001005E
S11235186F5000145D400B875E00605054705E6F
S1133527006 0720F860F957A000000000E0AE00113
S1133537006FE000187A000000000A0AE001006F3C
S1133547610 185E0042826E680012A87946067908
S113355700000140021 9006FE0001C7A04000043D9
S1133567E06F60001C6DF001006F51002001006FD8
S113357750000C5D400B876E680013A879460679E7
S113358700000140021 9006FE0001C6DF001006F9D
S1133597510 2001006F5000085D400B876E6800E3
S11335A714A879460679000001400219006FE0006C
S11335B71C6DF001006F51002001006F5000105D7A
S11335C7400B876E680015A879460679000001400D
S11335D70219006FE0001C6F66001C6DF601006F97
S11335E751002001006F5000145D400B875E00609F
S11335F75054705E0060720F860F9501006F600074
S1133607146901463 7A00000000200AF00FE15CDC
S106361700FE3877
S113361A01006DF57A000000001C0AF00FE15C005E
S113362AF8860B977A00000000200AF00FE15C008D
S113363AFE165E00605054705E0060720F860F952E
S113364A01006F60000C690146307A000000002017
S113365A0AF00FE15C00FEC401006DF57A00000078

S113366A001C0AF00FE15C00FA9E0B977A00000037
S113367A00200AF00FE15C00FEA25E006050547065
S1139E1853544F50005361666574792E74787400F7
S1139E28004F50454E004F50454E205370656564B2
S1139E3879004F50454E20537461727400434C4F60
S1139E48534500434C4F5345205370656564790 06F
S10F9E58434C4F53452053746172740057
S113368A01006DF60F86190069E06FE000026FE032
S113369A00046FE0000601006FE600087A000000EC
S11336AA00020AE001006FE0000C7A000000000447
S11336BA0AE001006FE000107A00000000060AE049
S11336CA01006FE0001419006FE000187A0000008F
S11336DA00180AE001006FE0001A19006FE0001EEB
S11336EA7A000000001E0AE001006FE000200100DA
S11336FA6D76547001006DF60F865E003F267A00E0
S113370A000000040AE05E003F2601006D76547053
S113371A5E0060720F850F9601006F740018010036
S113372A6F60001469007920000146520 1006F603E
S113373A001A6901462801006F66001A7900000120
S113374A69E07A00000000040AD0F9735E0041FCC4
S113375A7A0100009E64790000015E0049AA684864
S113376AA859586001E8F87268C86DF07A01000038
S108377A9E690FD05E03
S113377F003F300B875A003958FB6A01006F600016
S113378F10690079200001466E01006F60001A195D
S113379F11698101006F6000206901462C01006FE0
S11337AF6600207900000169E07A00000000040A36
S11337BFD0F9755E0041FC7A0100009E7579000017
S11337CF015E0049AA5A0039586848A85958600140
S11337DF787A00000000040AD00CB95E0041FCF8AF
S11337EF7268C86DF07A0100009E690FD05E003FCA
S11337FF300B875A00395801006F60000869014682
S113380F6E01006F60001A1911698101006F60006A
S113381F206901462C01006F6600207900000169C1
S10F382FE07A00000000040AD0F9555EA6
S113383B0041FC7A0100009E78790000015E00498B
S113384BAA5A0039586848A859586001007A0000F1
S113385B0000040AD00CB95E0041FCF87268C86D15
S113386BF07A0100009E690FD05E003F300B875A40
S113387B00395801006F60000C6901466C01006F41
S113388B60001A1911698101006F600020690146FC
S113389B2C01006F6600207900000169E07A0000BB
S11338AB0000040AD0F9755E0041FC7A0100009E0A
S11338BB75790000015E0049AA5A0039586848A877
S11338CB59586000887A00000000040AD00CB95ED6
S11338DB0041FCF87268C86DF07A0100009E690F15
S11338EBD05E003F300B87406401006F60000C69B2
S11338FB0079200001465601006F60001A19116907
S113390B8101006F60002069014628 01006F66008A
S113391B207900000169E07A00000000040AD00C52
S108392BB95E0041FC40
S10539307A0117
S113393200009E82790000015E0049AA6848A859E6
S11339424614F87268C86DF07A0100009E690FD0C0
S11339525E003F300B875E006050547001006DF6CD
S11339620F865E003F267A00000000040AE05E0034
S11339723F2601006D7654705E0060720F850F96CC

S113398201006F74001801006F60000C6900792058
S1133992000146520100 6F60001A690146280100C6
S11339A26F66001A7900000169E07A0000000004E2
S11339B20AD0F9735E0041FC7A0100009E6479002B
S11339C200015E0049AA6848A859586001E8F872E4
S11339D268C86DF07A0100009E690FD05E003F3027
S11339E20B875A003BB8FB6B01006F60000869004C
S11339F279200001466E01006F60001A1911698176
S1133A0201006F6000206901462C01006F660020EF
S1133A127900000169E07A00000000040AD0F96429
S1093A225E0041FC7A0185
S1103A2800009E8B790000015E0049AA5A40
S1133A35003BB86848A859586001787A000000002F
S1133A45040AD00CB95E0041FCF87268C86DF07ABF
S1133A550100009E690FD05E003F300B875A003B83
S1133A65B801006F6000106901466E01006F6000C8
S1133A751A1911698101006F6000206901462C0143
S1133A85006F6600207900000169E07A00000000FC
S1133A95040AD0F9445E0041FC7A0100009E907946
S1133AA50000015E0049AA5A003BB86848A8595866
S1133AB56001007A00000000040AD00CB95E0041E1
S1133AC5FCF87268C86DF07A0100009E690FD05E3C
S1133AD5003F300B875A003BB801006F6000146943
S1133AE501466C01006F60001A1911698101006FAD
S1133AF56000206901462C01006F660020790000F3
S1133B050169E07A00000000040AD0F9645E00410F
S1133B15FC7A0100009E8B790000015E0049AA5AD8
S1133B25003BB86848A859586000887A000000002F
S1133B35040AD00CB95E0041FCF87268C86DF07ACE
S1133B450100009E690FD05E003F300B8740640182
S1133B55006F60001469007920000146560100 6F6B
S1133B6560001A1911698101006F6000206901461F
S1133B752801006F6600207900000169E07A0000E2
S1133B850000040AD00CB95E0041FC7A0100009ED6
S1133B959C790000015E0049AA6848A8594614F8B3
S1133BA57268C86DF07A0100009E690FD05E003F10
S1133BB5300B875E006050547001006DF60F867AF6
S1133BC500000000040AE001006FE0000E01006D33
S1133BD57654705E0060720F860F957A00000000C0
S1133BE5040AE001006FE0000E01006F61000E0F93
S1133BF5E05E0042826E680004A8794606790000FB
S1133C0501400219006FE000127A04000043E06FDF
S1083C156000126DF0D8
S1133C1A01006F51002001006F50000C5D400B87BB
S1133C2A6E680005A879460679000001400219006A
S1133C3A6FE000126DF001006F51002001006F5018
S1133C4A00085D400B876E680006A879460679006E
S1133C5A0001400219006FE000126DF001006F517C
S1133C6A002001006F5000105D400B876E6800074B
S1133C7AA879460679000001400219006FE0001294
S1133C8A6F6600126DF601006F51002001006F503C
S1133C9A00145D400B875E00605054705E006072D2
S1133CAA0F860F957A00000000040AE001006FE016
S1133CBA000E01006F61000E0FE05E0042826E6823
S1133CCA0004A879460679000001400219006FE052
S1133CDA00127A04000043E06F6000126DF00100E5
S1133CEA6F51002001006F50000C5D400B876E6816

S1133CFA0005A879460679000001400219006FE021
S1073D0A00126DF043
S1133D0E01006F51002001006F5000085D400B87CA
S1133D1E6E680006A879460679000001400219007 4
S1133D2E6FE000126DF001006F51002001006F5023
S1133D3E00105D400B876E680007A8794606790070
S1133D4E0001400219006FE000126F6600126DF65B
S1133D5E01006F51002001006F5000145D400B876E
S10A3D6E5E00605054705E1B
S1133D750060720F860F957A00000000040AE001C7
S1133D85006FE0000E01006F61000E0FE05E004260
S1133D95826E680004A879460679000001400219 7D
S1133DA5006FE000127A04000043E06F6000126DBB
S1133DB5F001006F51002001006F50000C5D400BB6
S1133DC5876E680005A879460679000001400219 47
S1133DD5006FE000126DF001006F51002001006FCC
S1133DE55000085D400B876E680006A87946067982
S1133DF5000001400219006FE000126DF001006F31
S1133E0551002001006F5000105D400B876E680064
S1133E1507A879460679000001400219006FE00002
S1133E25126F6600126DF601006F51002001006FDD
S1133E355000145D400B875E00605054705E006057
S1133E45720F860F9501006F600014690146307A81
S1133E5500000000200AF00FE15C00FE4401006D44
S1063E65F57A00E8
S1133E680000001C0AF00FE15C00F8A60B977A002B
S1133E78000000200AF00FE15C00FE225E006050A3
S1133E8854705E0060720F860F9501006F60000C1E
S1133E98690146307A00000000200AF00FE15C0057
S1133EA8FECA01006DF57A000000001C0AF00FE15C
S1133EB85C00FABE0B977A00000000200AF00FE1BD
S10D3EC85C00FEA85E006050547019
S1139E6453544F50005361666574792E74787400AB
S1139E74005550005550205370656564790055506 2
S1139E842053746172740044 4F574E00444F574E2D
S1139E9420537065656479004 4 4F574E20537461B1
S1069EA4727400D2
S1133ED201006DF60F867A0000FFE220010069E01F
S1133EE27A0600FFE220F87268E87A000000000612
S1133EF20AE001006FE000027A0100009EA80100BF
S1133F026F6000025E00599CF8736EE8000FF84E72
S1133F126EE800106EE80011F84E6EE80012010020
S1133F226D7654700F811A800100699054705E009F
S1133F3260720F946E7D00197A0600FFE2206849D1
S1133F42A9434716A94D470CA9504714A95346281C
S1133F5268ED40246EED000F401E6EED0010401818
S1133F626E480006A8434706A8544708400A6EED68
S1133F72001140046EED001219005E00605054708F
S1133F8201006DF60F966869A94C46247A0600FF74
S1133F92E2207A00000000060AE001006FE000025E
S1133FA201006F71000801006F6000025E00599CFE
S1133FB2190001006D7654705E0060720F94010067
S1093FC26F7500187A067A
S1133FC800FFE2206849A9434716A94D470CA950A9
S1133FD84716A953462E686E400A6E6E000F4004BA
S1133FE86E6E001068DE401C6E480006A843470644
S1133FF8A854470A400E6E6E001168DE40066E6EC6

S1134008001268DE19005E006050547001006DF6FE
S11340180F966869A94C46247A0600FFE2207A00C5
S1134028000000060AE001006FE0000201006F6172
S1134038000201006F7000085E00599C190001001E
S11340486D7654705E0060720F850F9601006DF6F1
S11340587A0100009EB20FD05C00FF560B970C004C
S11340684624A8014706A8024716401A7A01000009
S11340789EC5790000015E0049AA7906000140083F
S1134088790600024002196 60D605E0060505470A4
S10440985EC6
S11340990060720F850C9E6DF67A0100009EB20FC7
S11340A9D05C00FE820B870C00461CA8004714A8AD
S11340B90146147A0100009ED4790000015E00498B
S11340C9AA400419664004790600010D605E006088
S11340D95054705E0060720F850F9601006DF67A79
S11340E90100009EE30FD05C00FEC60B970C00464F
S11340F924A8014706A8024716401A7A0100009E20
S1134109C5790000015E0049AA7906000140879D2
S1134119060002400219660D605E00605054705E2D
S11341290060720F850C9E6DF67A0100009EE30F05
S1134139D05C00FDF20B870C00461CA8004714A8AD
S11341490146147A0100009ED4790000015E0049FA
S1134159AA400419664004790600010D605E0060F7
S11341695054705E0060720F850F9601006DF67AE8
S10441790141
S113417A00009EF00FD05C00FE360B970C0046241D
S113418AA8014706A8024716401A7A0100009EC5ED
S113419A790000015E0049AA790600014008790600
S11341AA0002400219660D605E00605054705E00A2
S11341BA60720F850C9E6DF67A0100009EF00FD097
S11341CA5C00FD620B870C00461CA8004714A8017B
S11341DA46147A0100009ED4790000015E0049AAC0
S11341EA400419664004790600010D605E006050C0
S11341FA54705E0060720F850C9E6DF67A010000A2
S113420A9F040FD05C00FD1E0B870C004616A80006
S113421A4712A801460E7A0100009ED479000001D4
S113422A5E0049AA5E00605054705E0060721B878C
S113423A0F8518886EF800017A06000000010AF655
S113424A01006DF67A0100009F040FD05C00FD6047
S113425A0B970C004618A800470EA8014710A8029E
S109426A470C400A400866
S110427040066E780001400218880B875E3F
S113427D00605054705E0060720F850F96790000D8
S113428D096DF001006DF67A0100009F0F0FD05CF0
S113429D00FD740B970B870C004618A8014706A861
S11342AD024710400E7A0100009EC5790000015EA1
S10C42BD0049AA5E006050547030
S1139EA879796E6E79796E6E00005065726D69749A
S1139EB8436F6D6D616E642E74787400000A526589
S1139EC86164696E67204572726F72000A5772691E
S1139ED874696E67204572726F7200436F6D6D61AE
S1139EE86E642E74787400005065726D69745475CD
S1139EF8726E4F70656E2E74787400004D6F746FB8
S1139F08722E74787400004C696D69742E747874B9
S1059F18000044
S11342C65E0060720F860F957A0000009F1A0FE159
S11342D65E005FFC01006FE600080FE055267A00DA

S11342E60000000C0AE001006FE0000E19006FE009
S11342F6000C19006FE0001201006FE500145E0068
S1134306605054705E0060720F8601006FE600080D
S11343160FE201006DF25E00441E0B975E006050D3
S113432654705E0060727A37000000100F86010039
S11343366FE600087A02000000080AF201006DF237
S11343465E00441E0B9701006F6600080FA10FE283
S11343560FF05E005A9E5E005F107A170000001091
S11343665E00605054705E0060720F850D16790E04
S113437603E852E617F60FE101006F5000145E00E2
S113438644C25E006050547001006DF60F867A00D9
S11343960000000C0AE001006FE0000E7921000125
S11343A6460A790000016FE0000C400A0D1146062B
S10943B619006FE0000C8A
S11343BC01006D76547001006DF60F867A000000D3
S11343CC000C0AE001006FE0000E6F60000C0100AE
S11343DC6D7654705E0060720F860F9569606F7115
S11343EC00181D10470A190069D06F70001869E096
S11343FC5E00605054705E0060720F850D1617F6E8
S113440C0FE101006F5000145E0044C25E00605067
S105441C5470D7
S10B9F1A00000000000000003C
S113441E7A0000FFE23401006F7100045E005FFC5E
S113442E54705E0060727A37000000200F867A02A5
S113443E000000180AF201006DF255D40B970FE13C
S113444E0FF05E005FB60F817A0200009F327A0092
S113445E000000100AF05E005D787A00000000187C
S113446E0AF07A01000000080AF15E005FFC4010BA
S113447E7A02000000080AF201006DF255920B97C2
S113448E7A01000000180AF17A02000000100AF205
S113449E0FF05E005ACE0F817A00000000080AF07A
S11344AE5E005FA60D0046C87A17000000205E006E
S11344BE605054705E0060721B971B970F860F95AA
S11344CE7A02000000020AE201006DF25C00FF4076
S11344DE0B970FD10FF05E005FB60F817A020000CB
S11344EE9F327A000000000A0AE05E005D780B97A7
S11344FE0B975E00605054705E0060720F860F95CE
S107450E0FE055B0B2
S113451201006F6000120B7001006FE000125E0079
S1134522605054705E0060720D057A0400FFE23C35
S1134532400601006F44001A01006F40001A010097
S11145426F01001A46EC79250001460A7A063D
S113455000FFE2785A004644792500024 60A7A06AB
S113456000FFE2965A0046447925000B460A7A0674
S113457000FFE2B45A0046447925000C460A7A0645
S113458000FFE2D25A0046447925000D460A7A0616
S113459000FFE2F05A0046447925000E460A7A06E7
S11345A000FFE30E5A00464479250013460A7A06B3
S10845B000FFE32C5A9B
S11345B500464479250014460A7A0600FFE34A5A61
S11345C50046447925001546087A0600FFE368404E
S11345D56E7925001646087A0600FFE38640607962
S11345E525001746087A0600FFE3A4405279250003
S11345F51E46087A0600FFE3C240447925001F469C
S1134605087A0600FFE3E040367925002946087A53
S11346150600FFE3FE40287925002A46087A0600AE
S1134625FFE41C401A7925002B46087A0600FFE4AF

S11346353A400C7925002C46067A0600FFE4580F0C
S1134645E6461C7A0100009F2219005E0049AA7AFA
S10446550160
S113465600009F2419005E0049AA1A8040540100F5
S11346666FE4001601006F40001A01006FE0001AA4
S113467601006F86001601006FC6001A7A0000005B
S11346869F3A7A01000000020AE15E005FFC7A00AD
S113469600009F427A010000000A0AE15E005FFC07
S11346A669E51A8001006FE000120FE05E000EB0AC
S11346B60FE05E006050547001006DF60F865E00D9
S11346C6114401006F60001601006F61001A0100BA
S11346D66F81001A01006F60001A01006F610016F6
S11346E601006F81001601006D7654705E006072E2
S11346F60F867A0200FFE23401006DF25C00FD18BA
S11347060B977A0000FFE2347A01000000020AE107
S11347165E005FFC5E006050547001006DF60F860C
S11347267A0000009F4A7A01000000020AE15E0057
S11347365FFC01006D76547001006B2000FFE256AA
S109474601006F01001ADF
S113474C46041900547079000001547001006B2168
S113475C00FFE25601006F11001A1988400C01008A
S113476C6F11001A0D800B500D080F9146F00D8040
S113477C54706DF50D0501006B2000FFE25601002E
S113478C6F01001A472001006B2100FFE2566910EC
S113479C1D5046040F90401001006F11001A0100C8
S11347AC6F10001A46E81A806D7554705E006072C3
S11347BC0D0555BE0F86460E0D505C00FD5C0F8635
S11347CC5C00FF22401C7A00000000020AE07A0120
S11347DC00009F4A5E005F8A0D0047060FE05C00F5
S11347ECFF040FE05E00605054705E0060720D05B4
S11347FC55800F86460E0D505C00FD1E0F865C0027
S113480CFEE440247A00000000020AE07A01000072
S113481C9F4A5E005F8A0D0047080FE05C00FEC6EE
S113482C40060FE05C00FE8A5E00605054705E0030
S113483C60721B971B977A0500FFE23C01006F56D1
S107484C001A407299
S113485001006F65001A7A00000000020AE07A0185
S113486000009F3A5E0060420D0047547A0000004A
S109487000020AE07A01D8
S113487600009F4A5E0060420D00473E7A01000039
S113488600020AE17A020000000A0AE20FF05E0063
S11348965ACE0F817A0000FFE2345E005F960D0068
S11348A647187A0000FFE2347A01000000020AE1A9
S11348B65E005FFC0FE05E0005480FD601006F60E7
S11348C6001A46860B970B975E00605054701A8049
S11348D601006BA000FFE2527A0000FFE25A0100DA
S11348E66BA000FFE2567A0000FFE23C01006BA0DA
S10D48F600FFE2701A8001006BA0BE
S113490000FFE2745470F801386018883861386227
S1134910188838633890F8FF389118883864F88815
S11349203865F804386679009E576B80FF68190074
S11349306B80FF6A19006B80FF6C547001006DF28D
S11349407F67722079009E576B80FF687A0100FFB2
S1134950E2347A0200009F520F905E005ACE0100AB
S11349606D7254705A00483A7A0000009F427A01EF
S113497000FFE2345E005FFC558C5E0002A85A0023
S105498048D416

S1139F220A0063616C6C6F63206661696C6564002F
S1139F32408F400000000000BFF00000000000005E
S1139F4200000000000000000C0000000000000004C
S10B9F523F50624DD2F1A9FC5E
S11349827A0000009F5A01006DF07A0000FFE4767E
S11349925E0059140B977A0000FFE47601006DF074
S11349A25E004D2C0B9754705E0060727A0500FF17
S11349B2E4B60D040F960C4458600114AC0047682A
S11349C2AC014710AC0258700106AC03587000C228
S11349D25A004AD255AA7A0100009F5C0FE05E009A
S11349E259720D00461E7A0000009F6201006DF0AD
S11349F27A0000009F5F01006DF00FE05E00591422
S1134A020B970B9701006DF67A0000009F5F010080
S1134A126DF00FD05E0059140B970B9701006DF5E3
S10D4A227A0000009F5F40507A0104
S1134A2C00009F5C0FE05E0059720D00461E7A0079
S1134A3C00009F6201006DF07A0000009F5F01008F
S1134A4C6DF00FE05E0059140B970B9701006DF698
S1134A5C7A0000009F5F01006DF00FD05E005914C7
S1134A6C0B970B9701006DF57A0000009F5F010017
S1134A7C6DF05E004B280B970B9701006DF65E00F3
S1134A8C4D2C0B974040403E01006DF67A00000020
S1134A9C9F5F01006DF00FD05E0059140B970B97BD
S1134AAC01006DF57A0000009F5F01006DF05E0060
S1134ABC4B280B970B970FD05E0059B80B500D0179
S10A4ACC0FD05E0057905E5E
S1084AD3006050547067
S10E9F5A0C000A00002573000A0C0035
S1134AD8188838BA38B8F85038B919000B50792003
S1134AE801184DF8F83038BA28BCF88038BC54702F
S1134AF80C8820BC737047FA38BBE07F30BC547015
S1134B0820BC736047FA29BDE0BF30BC0C985470D1
S1134B187EBC736047067900000154701900547015
S1134B285E0060727A0600FFE4F67A05000000046E
S1134B387A00000000180AF00AD07308470E7A00BA
S1134B48000000180AF00AD00B70400A7A0000002F
S1134B5800180AF00AD00F85189968E901006DF06A
S1134B6801006F71001C0FE05E0059D40B971955B3
S1134B780D5017F00AE0680947220D5017F00AE0B4
S1134B886808A80A4606F80D5C00FF640D5017F084
S1134B980AE068085C00FF580B5540D45E0060507B
S1054BA8547044
S1134BAA1911400C19880B58792806824DF80B51B4
S1134BBA1D014DF0547001006DF50D090D8D28D6B8
S1134BCA17500D010D99470C0D1979690060794940
S1134BDA001040060D19796900600DD50D90148DEA
S1134BEA0CD8C88038D63DD639D67900000455B0DA
S1134BFA01006D7554706DF56DF40D090D8528D698
S1134C0A17500D010D99470C0D19796900607949FF
S1134C1A001040060D19796900600D54119411941E
S1134C2A11941194EC0F0D90148CED0F148D0CC884
S1134C3AC88038D63CD60CD8C88038D63DD639D6A3
S1134C4A790000045C00FF586D746D7554001009F
S1134C5A6DF619EE7900000F5C00FF4419667908B6
S1134C6A00030DE05C00FF4E0B56792600034DEE60
S1134C7A790800020DE05C00FF3C196679080028F8
S1134C8A0D605C00FF70790800100D605C00FF6620

S1094C9A7908000E0D6015
S1134CA05C00FF5C790800060D605C00FF52790828
S1134CB000010D605C00FF48790800020D605C0094
S1134CC0FF3E01006D7654707908000119005C0005
S1134CD0FF2E7908000219005A004C006DF60C8E65
S1134CE0AE0C460455E2401CAE0A460455DA4014A5
S1134CF0AE0D460455D2400C17D60D68790000015D
S1134D005C00FEFC6D76547001006DF60D0E0D8691
S1134D100D6079080040528009E0791000800D0889
S1134D2019005C00FEDA01006D7654705E0060725B
S1134D30790C000119447A0600FFE5467A05000064
S1134D4000047A00000000180AF00AD07308470E26
S1134D507A00000000180AF00AD00B70400A7A00AB
S1134D60000000180AF00AD00F85189968E90100BD
S1134D706DF001006F71001C0FE05E0059D40B97BA
S1134D8019550D5017F00AE0680947560D5017F0F2
S1094D900AE06808A80A0E
S1134D96460A0DC80D405C00FF68403C0D5017F0F5
S1134DA60AE06808A80D460C790800020D405C006D
S1134DB6FE4840240D5017F00AE06808A80C460682
S1134DC65C00FEFE40120D5017F00AE0680817D08B
S1134DD60D080DC05C00FE220B5540A05E0060501E
S1054DE6547004
S1134DE81911400C19880B58792806824DF80B5174
S1134DF81D014DF054705C0009145C0000B45504A7
S1134E085A004EE201006DF618886AA800FFE649C9
S1134E186AA800FFE64B18886AA800FFE64A6AA852
S1134E2800FFE5877908008079000004 5C00080822
S1134E3879080010790000205C0007FC7906000F50
S1134E48790800100D605C0007EE0D687900000B0F
S1134E585C0007E40D687900000D5C0007DA790847
S1134E680050790000095C0007CE790800D6790064
S1134E7800075C0007C219660D605C000710790E15
S1134E8800010DE05C0006E60DE05C0007000D681C
S1134E987900002B5C0007A00D687900002F5C00E7
S1134EA807960DE8790000275C00078C01006D76F2
S1134EB8547079080005 19005C00077C79000064C8
S1134EC85C00FF1C790800C419005C00076A7908B4
S1094ED800037900000154
S1134EDE5A00564079080002790000055C0007521B
S1134EEE790800CC19005A0056405E0060727A04AD
S1134EFE0000A0187A0000009FE701006DF05E002D
S1134F0E4B280B9719EE19660DE5790D001052D546
S1134F1E0D6009505C00070C17506DF001006DF425
S1134F2E5E004B280B970B870B56792600104DE02E
S1134F3E7A000000A01E01006DF05E004B280B9757
S1134F4E0B5E792E00044DBE5E006050547001005E
S1074F5E6DF67A0669
S1134F6200FFE586790000065C0006C468E87C6001
S1134F727340473679000005C0006B40C8E73480A
S1134F8247045C0002C8735E47085C0002C25A0011
S1134F925050730E47085C0002B85A005050731EFB
S1134FA247045C0002AE5A0050507C607360472293
S1134FB27900000C5C0006780C8E730847085C00CD
S1134FC200925A005050731E587000825C0001C652
S1134FD2407C7C607320471E7900000A5C00065007
S1134FE20C8E730847065C00020A4062731E475E1A

S1134FF25C00023E40587C607310475279000008FF
S11350025C00062C0C8E7368470A5C00FDFC5C0096
S1135012FECE403A734E471A790800C07900000960
S11350225C00061A7908000379000055C00060E8D
S1135032401C737E4718790800507900000095C0010
S113504205FC7908000279000055C0005F0010007
S10850526D7654705E51
S113505700607219DD790000265C0005CE0C8E73A3
S1135067685870010 47A0600FFE589790000086D26
S1135077F00FE1790800267900002 55C0005CC0BC9
S1135087870DD05C0004E80DD05C000502790500AC
S1135097200D505C0004C06868E8605860009C6E8F
S11350A7680001473CA801471EA8034772A80547A4
S11350B73EA8064726A8084716A809475CA80A4733
S11350C726A80B4760406C5C0001865A00515E6A54
S11350D72800FFE58817500D08400E5C00022440A6
S11350E7765C00035440700DD87900002140247981
S11350F70800400D505C0005406E6E0002CE806ACA
S1135107AE00FFE5876A2800FFE58717500D08798A
S113511700000 45C000522403E5C00038E40385CBF
S1135127000188403 26E680002472C0D505C000472
S11351370E40240D505C000406401C6868E860A814
S10851472046080D5095
S113514C5C0003F6400C686EEE60AE400D505C00E4
S113515C03E87A0000FFE64A7D0070000DD05C0086
S113516C045440226A2800FFE64B470818886AA8B3
S113517C00FFE64B0DD05C000410790800017900A8
S10A518C00275C0004AE5E86
S113519300605054705E0060727900002E5C00045E
S11351A38E0C8EE8C017504626790000406DF07AC6
S11351B30500FFE5C90FD17908002E7900002D5CA6
S11351C300048C0B870D060D010FD05C000562797B
S11351D30000015C00039C790800017900002F5C47
S11351E300045A5E0060505470790000365C00047A
S11351F33E54706DF6790000225C0004320C8E730A
S113520368472A7358472619005C0003866A2800F7
S1135213FFE64B470C5C0001C819005C0003A04088
S11352230C79080001790000275C0004106D7654A3
S1135233706DF67900002A5C0003F40C8E736847E3
S113524308735847045C0004826D76547054705499
S1135253705470547 05E0060727A0600FFE58968CB
S113526368E803A80246426E680003E80747186E1E
S1135273690003E9071751177178106A2800009F23
S10852836617505C00FA
S113528802D46E680003E8071750790100011A0871
S11352984B04101140F817117A0000FFE649680A19
S11352A8169A688A5E00605054705E0060727A06CF
S11352B800FFE5896868E803A80246406E680003B2
S11352C8E80747186E690003E90717511771781043
S11352D86A2800009F6617505C0002626E6800032C
S11352E8E8071750790100011A084B04101140F818
S11352F87A0000FFE649680A149A688A5E006050DB
S111530854707A0000FFE64A7D0072006A28A6
S113531600FFE58CA801470AA8024716A8034722FF
S113532654706A2900009F6E17517A0000009F6E21
S113533640686A2900009F8217517A0000009F8007
S113534640586A2800FFE58B470EA801471AA802B2

S11353564726A803473254706A2900009FA717D12E
S11353667A0000009FA740326A2900009FAB17D13D
S11353767A0000009FAB40226A2900009FBD17D127
S10D53867A0000009FBD40126A295F
S113539000009FC517D17A0000009FC555025470C5
S11353A05E0060720F840D187A0600FFE64C6A28CF
S11353B000FFE59017500C8018886A2900FFE58FDD
S11353C01751091069E01D804F0269E801006BA4C1
S11353D000FFE64EF8016AA800FFE64B55065E00A3
S11353E0605054705E0060727A0600FFE64C7905E7
S11353F000086960474469607920000084E026965C6
S11354007A0400FFE64E6DF5010069417908002238
S1135410790000215C0002780B870D056961190191
S113542069E117F0010069410A81010069C16960FE
S1135430460818886AA800FFE64B5E006050547067
S10454405E0A
S11354410060726A2800FFE589E80317500D05190A
S1135451EE790600210D0047067925000146100F5C
S1135461E05C0001DA0DE80D605C0001D2403C799B
S1135471250002462E6A2800FFE58CE80717500D28
S11354810579010011A084B04101140F86A28003C
S1135491FFE6491750660147067908000140060DEA
S11354A1E840020DE80D605C0001945E0060505419
S11354B1706DF66A2800FFE58B6AA800FFE5884650
S11354C13C19660D68790000285C0001720D68794A
S11354D100002C5C0001680D68790000305C00015C
S11354E15E0D68790000345C0001540D6879000099
S11354F1385C00014A0D687900003C404018886A15
S1135501A800FFE649790600010D605C0000867979
S11355110800117900000285C00012479080003794F
S113552100002B5C0001180D60554A7908001279BF
S108553100002C5C00EA
S113553601080D687900002F5C0000FE6D7654703B
S11355466DF60D065C0000E4C88017500D080D606B
S11355565C0000E66D7654706DF60D065C0000CCBB
S1135566E87F17500D080D605C0000CE6D76547011
S113557601006DF60D0617F61036790800087860F7
S11355866B2000FFE0085C0000B001006D765470EC
S11355965E0060720D0617F610367A0500FFE0000E
S11355A60AE569505C00008417500D06703E0D68CD
S11355B669505C0000845E00605054705E00607247
S11355C61B971B977A0500FFE64A0D0EFE0179002D
S11355D600010DE11A094B04101040F868591751E0
S11355E666104704702E4002722E701E17560DE089
S11355F617F0103001006FF000040D6801006F71A1
S1135606000478106B2000FFE000010069F1552AC1
S1135616790000011B5E4B04101040F86859158988
S109562668D90B970B97F6
S104562C5E1C
S113562D00605054706AA8004000036A28004000CF
S113563D0154700D016AA9004000030D806AA80092
S113564D40000154705E0060727A03004000030F46
S113565D840F9568BC19EE196640180DC055C6E840
S113566D0F471868BC6A280040000168D80B750BFA
S113567D5E0B566F7000181D064DE00DE05E006069
S113568D5054705E0060720D0C0D830F956F740096
S113569D1819EE1966401E0D30558AE81F471A0D6D

S11356ADC06AA80040000368586AA8004000010BB7
S11356BD750B5E0B561D464DDE0DE05E00605054BE
S11356CD7001006DF619EE7A0000FFE6097901000D
S11356DD405C0001140D06790000015C00FEAA0D6B
S11356ED6647166DF67A0100FFE6097908002A79F7
S11356FD000029558E0B870D0E790000015C00FE0D
S113570DB40DE001006D76547019006BA000FFE637
S106571D546BA027
S113572000FFE65219006BA000FFE6586BA000FFD4
S1135730E65654705E0060727A0400FFE6547A0500
S113574000FFE6520F830D1E7FF57250199940300A
S1135750694079200100 4C2C695017F068397800B2
S11357606AA900FFE65A69500B5017F0790101004E
S113577001D0531069D869400B5069C00B730B59A2
S10E57801DE94DCC7FF570500D905ECD
S113578B00605054705E006072790E01007A040061
S113579BFFE6587A0500FFE6560F830D127FF5726D
S11357AB501999403469407920010 04C32695017E4
S11357BBF001D053E00D8017F0683978006AA90027
S11357CBFFE75A69500B5017F001D053E069D869C2
S11357DB400B5069C00B730B590D201D094DC67F30
S11357EBF570500D905E00605054705E0060727ADD
S11357FB0500FFE6580F840D1E7FF57250196640A6
S113580B381DE64C386B2000FFE6566951191079A9
S113581B10010017F07901010001D053100D80170F
S113582BF00D6117F10AC178006A2800FFE75A6887
S113583B9869501B5069D00B5669504EC47FF57055
S113584B500D605E00605054705E0060727A05000C
S113585BFFE6540F840D1E7FF5725019664038 1DF9
S113586BE64C386B2000FFE6526951191079100191
S108587B0017F07901A4
S1135880010001D053100D8017F00D6117F10AC10B
S113589078006A2800FFE65A689869501B5069D05F
S11058A00B5669504EC47FF570500D605ECD
S11358AD00605054706B2000FFE65417F07901012E
S11358BD0001D053100D8054706B2000FFE6581774
S11058CDF07901010001D053100D805470DB
S1139F6620282C3034383C20120100010000000860
S1139F76FEFF10000100010102010902270001 0191
S1139F8600C0640904000003000000030705810202
S1139F964000FF070502024000FF07058302400059
S1139FA6FF040309041203550053004200200054 22
S1139FB60045005300540008033 1002E00300022F0
S1139FC603550053004200200054004500 5300543B
S1139FD600200050005 2004F004700520041004D40
S1049FE60077
S11399C00023002B0033003B0027002F0037003F0C
S1139FE730302030312030322030332030342030AD
S1139FF735203036203037203038203039203041 73
S113A00720304220304320304420304520304 60A58
S10CA01700253032582000 0A0034
S11358DA01006DF67A0600FFE896010069607A0115
S11358EA41C64E6D5E007F2A7A10000030390100EE
S11358FA69E06960198817707A01000080005E0008
S10D590A7F040D1001006D76547048
S11359145E0060720F857A06000000047A000000BE
S113592400180AF00AE07308470E7A000000001812

S11359340AF00AE00B70400A7A00000000180AF02B
S11359440AE00F8601006DF001006F70001C010076
S11359546DF001006DF51A91790000015E00609A03
S11159647A170000000C0D065E0060505470B0
S11359725E0060720F860F9540040B760B756869A3
S113598268581C8946040C9946F068681750685D8C
S10D5992175519505E006050547061
S113599C5E0060720F840FC60F950FE00B766C5987
S10F59AC688946F60FC05E00605054701E
S11359B85E0060720F851AE640020B760FD00B75F6
S10F59C8680946F60FE05E006050547062
S11359D45E0060720F850F9601006F71001801005D
S11359E46DF101006DF601006DF51A917900000166
S11359F45E00609A7A170000000C5E0060505470D9
S1135A040FC446120FD5460E792107FF46120FB372
S1135A144604156E4A0A1AC47A050000000818667B
S1135A245A005C980D9946120FC4461A0FD54616B0
S1135A340FB3461E16E6580002660FA246360FB38E
S1135A4446325800025A0FA246140FB346105800A8
S1135A54024E0CE60D190FA40FB55A005CA41035C1
S1135A641234792C00104C0C1B5910351234792C38
S1135A7400104DF410331232792A00104C0C1B51D0
S1135A8410331232792A00104DF4580000CA1AC494
S1135A941AD5186619995800020601006DF201001F
S1135AA46DF301006F23000401006DF3010069230A
S1135AB4795B800001006DF30FF2550E0B970B9782
S1135AC401006D7301006D7254706DF601006DF584
S1135AD401006DF401006DF301006DF201006DF13D
S1135AE401006DF00100691401006925103410 35BB
S1095AF41FD44218451007
S1135AFA01006F14000401006F2500041FD444063B
S1135B0A0F940FA10FC26816682E0FA00100691423
S1135B1A01006F150004010069020 1006F0300040C
S1135B2A796C000F796A000F691911191119111982
S1135B3A1119796907FF6901111111111111111154
S1135B4A796107FF792907FF5870FEAE0D11587066
S1135B5AFECC10351234103512341035123410338A
S1135B6A12321033123210331232794C0080794ACE
S1135B7A00800D901910474C792000364E3E79204B
S1135B8A00204D0E793000200FB34702700A0FA38D
S1135B9A1AA2792000104D14793000100D380DB374
S1135BAA0D2B0DA219AA0D884702700B0C884F14EE
S1135BBA113213334402700B1B5040F01AA27A03BA
S1135BCA000000010C6815E84B2A0AB544020B745D
S1135BDA0AA4792C0100585000801 1341335440269
S1075BEA700D0B59D3
S1135BEE792907FF5860006E1AC41AD5580000A60B
S1135BFE1FA446061FB55870FE8A1AB544087A3498
S1135C0E0000000145061AA4450440121AA41735D4
S1135C1E17347A150000000144020B740CE60FC40E
S1135C2E46080FD41AD57919FFE00DCC460C0D4C4E
S1135C3E0DD40D5D19557919FFF0792C0100450826
S1135C4E113413350B5940F2792C00804C08103562
S1135C5E12341B5940F20D994E10113413350D9910
S1135C6E4A08113413350B5940F4732D471C0CD8C5
S1135C7EE80B47167A150000000844020B74792CC2
S1135C8E01004D06113413350B59113413351134EC

S1135C9E1335113413357796C000F101910191019AF
S1135CAE1019649C101C1006131C01006D7001006A
S1135CBE698401006F85000401006D7101006D722E
S1135CCE01006D7301006D7401006D756D76547076
S1135CDE01F06401586000807924O7FF5860006C5E
S1135CEE5800007401F064235860006C40520F8515
S1135CFE01F06415460E0D44464601F064234640FA
S1135D0E5800005410311230792800104C0C1B5CD3
S1135D1E10311230792800104DF4580000BA0FA537
S1135D2E01F06435472610331232792A00104C0CD9
S1135D3E1B5410331232792A00104DF45800009E72
S1135D4E1AC4100E131C1AD55800018E790E07FFB4
S1135D5E1AC41AD5580001621AC418EE790E07FF39
S1135D6E19DD790500085800015001006DF6010098
S1135D7E6DF501006DF401006DF301006DF201008C
S1135D8E6DF101006DF0681E6826156E691C111CFD
S1135D9E111C111C111C796C07FF692411141114A9
S1135DAE11141114796407FF01006D1001006911BC
S1135DBE7968000F01006F23000401006922796ADC
S1095DCE000F792C07FF12
S1135DD45870FF06792407FF5870FF120DCC5870D2
S1135DE4FF160D445870FF40194C791C03FF0DCE68
S1135DF4792EFFCC58D0FF5279480010794A00100D
S1135E041AC47A0500000100103312321031123023
S1135E14441C0AB144087A100000000145040AA096
S1135E2440040AA004011235123444E0401C1AB1A0
S1135E3444087A300000000145041AA040041AA063
S1135E4404011235DD01123444C2730D46080AB14C
S1135E5444020B700AA001F064014702700D792C0F
S1135E6400804C06103512341B5E792E07FF58C090
S1135E74FEE40DEE4E2E113413354402700D0DEE77
S1135E844A040B5E40F0732D47180CD8E80B4712F5
S1135E947A1500000008440A0B74792C00804D0223
S1135EA40B5E4020732D471C0CD8E80B47167A155C
S1135EB40000000844020B74792C01004D061134D0
S1095EC413350B5E1134DF
S1135ECA133511341335113413357796C000F101E41
S1135EDA101E101E101E64EC101C100E131C010061
S1135EEA6D700100698401006F85000401006D7102
S1135EFA01006D7201006D7301006D7401006D750F
S1095F0A01006D765470E6
S1135F1001006DF201006DF101006F010004010049
S1135F2069000D8211121112111211121112796207FF09
S1135F300D8A7968000F793203FF4B4A79220053A7
S1135F404E44794800107912FFEC4B22792200204D
S1135F504C0C0D224F1E103112301B5240F40F9087
S1135F607912FFE00D224F0C10301B5240F6113016
S1135F700B524BFA796A8000470217B001006D712A
S10D5F8001006D7254701A8040F2A4
S10F5F8A5E007E48A80147021900547015
S1135F965E007E480C884E0419005470F801547054
S1135FA65E007E480C884604F8015470190054704C
S1135FB601006DF201006DF11AA20F9147224A0604
S1135FC67902080017B17912041F1B52103144FAE3
S1135FD6103112121031121210311212103112121224
S1135FE6698201006F8100026F8A000601006D71EC
S1095FF601006D725470FE

S1135FFC01006DF2010069020100699201006F0258
S111600C000401006F92000401006D725470D5
S113601A6DF301006DF201006DF101006DF06C0B7F
S113602A689B0B011B7246F601006D7001006D71CE
S10B603A01006D726D735470D7
S11160425E007E481A0847047900000154707E
S113605001006F76001401006D7201006FF20010F1
S113606001006D7201006D7301006D7401006D75A7
S1056070547067
S113607201006DF501006DF401006DF301006DF295
S113608201006F72001001006DF201006FF600143F
S10B609201006F720004547059
S113609A5E0060727A37000000B27A0300FFE86E8E
S11360AA7A0400FFE85E0D0201006FF100AA010005
S11360BA6F7600CE01006F7500D20D00466C0100A9
S11360CA6F7000CA460E790004526BA000FFE89A6B
S11360DA5A00616C01006F7000CA6E080010E8076D
S11360EA17D0460C790005146BA000FFE89A40729A
S11360FA01006F7000CA6E080010732846107308F7
S113610A470C790105166BA100FFE89A4054010078
S113611A6F7000CA6E0800107368464601006F70FC
S113612A00AA01006BA000FFE8920D2017F00100FE
S113613A6BA000FFE85A01006F7000CA01006BA050
S113614A00FFE8601A8001006BA000FFE86418886A
S113615A68C8F8306EF800885C001CBA0D005870E5
S113616A0A1C7900FFFF5A006BAA6868A8255860C1
S113617A09CC0B76188868C80FF001006BA000FFE2
S107618AE8720FF0B5
S113618E01006BA000FFE8761A8001006BA000FFF0
S109619EE87A01006BA08A
S11361A400FFE87E1A8001006BA000FFE882010073
S11361B46BA000FFE8861A8001006BA000FFE88A49
S11361C440306868A8204718A8234720A82B470A0B
S11361D4A82D461C7D40701040167D407030401041
S11361E47C407330460A7D40704040047D407020FB
S11361F40B766868A82D47CAA82B47C6A82047C2B0
S1136204A82347BEF9206AA900FFE8686869A93092
S1136214460AF9306AA900FFE8680B761A80010080
S11362246BA000FFE86A6868A82A586000800FD052
S11362340BF001006FF000A07308470A01006F70B0
S113624400A00B70400601006F7000A00F851BF0C7
S113625401006FF000967308470A01006F700096FF
S11362641B70400601006F700096690017F001006F
S11362746BA000FFE86A4C167D40701001006B2090
S113628400FFE86A17B001006BA000FFE86A010091
S10962946B2000FFE86A25
S113629A7A20000002004F0E7D4070007900051835
S11362AA6BA000FFE89A0B76686817D0796000FF45
S11362BA17F078006A280000A0287328587000860F
S11362CA1A8001006BA000FFE86A4064686817D06F
S11362DA7930003017F001006FF000A07A01000056
S11062EA02001A810F907A010000000A5E85
S11362F7007F0401006B2100FFE86A1F904D240112
S1136307006B2000FFE86A7A010000000A5E007F45
S11363172A01006F7100A00A9001006BA000FFE83B
S11363276A400E7D407000790005186BA000FFE8F6
S11363379A0B76686817D0796000FF17F078006AC0

S1136347280000A028732846867A00FFFFFFF0175
S11363570069B06868A82E586001000B766868A8C2
S11363672A466C0FD00BF001006FF00096730847B5
S11363770A01006F7000960B70400601006F7000F2
S1136387960F851BF001006FF000A07308470A0101
S1136397006F7000A01B70400601006F7000A069BA
S11363A70017F0010069B04C0A7A00FFFFFFFF01F5
S11363B70069B0010069307A20000002004F0E7DAA
S11363C7407000790005186BA000FFE89A0B766808
S11363D76817D0796000FF17F078006A280000A0DB
S10863E728732847762E
S11363EC1A80010069B04058686817D079300030C2
S11363FC17F001006FF000A07A01000002001A816F
S113640C0F907A010000000A5E007F0401006931DD
S113641C1F904D1C010069307A010000000A5E00D8
S113642C7F2A01006F7100A00A90010069B0400E31
S113643C7D407000790005186BA000FFE89A0B767D
S113644C686817D0796000FF17F078006A2800009D
S113645CA028732846927A000000002001006BA04C
S113646C00FFE88E6868A8684708A86C4704A84C26
S113647C4610686817D017F001006BA000FFE88E78
S113648C0B766868A825587004A4A84558700212A6
S113649CA8475870020CA8584742A863587002E4E6
S11364ACA8644738A865587001F8A866587001F2BB
S11364BCA867587001ECA8694722A86E587004307D
S11364CCA86F4718A87058700398A873587002F8EF
S10964DCA8754708A8782B
S10D64E247045A00697601006B209D
S11364EC00FFE88E7A200000006C46440FD00B901E
S11364FC01006FF000AA7308470A01006F7000AA2D
S113650C0B70400601006F7000AA0F851B900100F1
S113651C6FF000967308470A01006F7000961B70AA
S113652C400601006F700096010069025A00667EF6
S113653C01006B2000FFE88E7A2000000068586091
S113654C009A6868A875470CA8584708A8784704A8
S113655CA86F46420FD00BF001006FF000AA73082E
S113656C470A01006F7000AA0B70400601006F70A0
S113657C00AA0F851BF001006FF000967308470A01
S113658C01006F7000961B70400601006F7000963F
S113659C6900177040400FD00BF001006FF00096AC
S11365AC7308470A01006F7000960B7040060100D8
S11365BC6F7000960F851BF001006FF000AA730833
S11365CC470A01006F7000AA1B70400601006F7030
S10965DC00AA690017F09C
S11365E20F825A00667E6868A875470CA858470848
S11365F2A8784704A86F46420FD00BF001006FF052
S113660200967308470A01006F7000960B704006EC
S113661201006F7000960F851BF001006FF000AA56
S11366227308470A01006F7000AA1B70400601003D
S11366326F7000AA6900177040400FD00BF0010081
S11366426FF000AA7308470A01006F7000AA0B706B
S1136652400601006F7000AA0F851BF001006FF066
S113666200967308470A01006F7000961B7040067C
S113667201006F700096690017F00F820100693004
S11366827A20FFFFFFFF460A7A00000000010100A3
S113669269B0686817D06DF07A01000000020AF150
S11366A20FA05C00050E0B875A0068EA01006B20FD

S11366B200FFE88E7A200000004C46420FD00B9078
S11366C20B9001006FF000AA7308470A01006F7074
S10966D200AA0B70400654
S11366D801006F7000AA0F851B901B9001006FF0DB
S11366E800967308470A01006F7000961B70404AB2
S11366F801006F70009640420FD00B900B90010081
S11367086FF000AA7308470A01006F7000AA0B70A4
S1136718400601006F7000AA0F851B901B900100B3
S11367286FF000967308470A01006F7100961B719A
S1136738400601006F7100960F907A0100000080F7
S11367480AF15E005FFC010069307A20FFFFFFFF5A
S1136758460A7A0000000006010069B0686917D18B
S113676801006F70008401006DF001006F700084F8
S113677801006DF07A00000000080AF05C0007864B
S11367880B970B975A0068EA0FD00BF001006FF0D4
S113679800AA7308470A01006F7000AA0B7040062D
S11367A801006F7000AA0F851BF001006FF00096BF
S11367B87308470A01006F7000961B7040060100BA
S10967C86F7000966E08DD
S11367CE00015A00693C0FD00B9001006FF000AA34
S11367DE7308470A01006F7000AA0B704006010090
S11367EE6F7000AA0F851B9001006FF0009673085F
S11367FE470A01006F7000961B70400601006F7010
S113680E0096010069000F825E0059B801006FF017
S113681E00AA010069317A21FFFFFFFF4712010031
S113682E69311F814C0A0100693001006FF000AA23
S113683E0FF001006BA000FFE8761A8001006BA039
S113684E00FFE88201006B2000FFE86A01006F7110
S110685E00AA1A9001006BA000FFE88A5AFF
S113686B0069840FD00B9001006FF00096730847FB
S113687B0A01006F7000960B70400601006F7000E9
S113688B960F851B9001006FF000AA7308470A014E
S113689B006F7000AA1B70400601006F7000AA0105
S11368AB0069007A6000FFFFFF01006FF0009601A3
S11368BB0069307A20FFFFFFFF460A7A00000000D1
S11368CB01010069B0790000786DF07A01000000D6
S11368DB020AF101006F7000985C0002CE0B870F68
S11368EBF00F825E0059B801006FF000AA5A0069DD
S11368FB840FD00B9001006FF000A07308470A01BF
S113690B006F7000A00B70400601006F7000A00FAA
S113691B851B900F81730847060F901B7040020F66
S113692B90010069006B2100FFE8666981404AF81A
S113693B2568F80FF00F827A000000000101006F49
S113694BF000AA0FF101006BA100FFE8761A910189
S106695B006BA12A
S113695E00FFE88201006B2100FFE86A1A81010043
S113696E6BA100FFE88A400E790005186BA000FFAB
S113697EE89A7D4070006868A86E587001B86A285E
S113698E00FFE85E7308586001AC7C407310463616
S113699E01006B2000FFE88A4F2C40207A01000093
S11369AE00017A0000FFE8685C0013B87A0000FF6C
S11369BEE88A010069011B710100698101006B20E6
S11369CE00FFE88A4ED601006B2000FFE87E4624C6
S11369DE01006B2000FFE882461A01006B2000FFC6
S11369EEE886461001006F7100AA0FA05C001374B5
S11369FE5A006B0801006B2000FFE8720FA11A907A
S1136A0E01006FF0009C01006FF000A04F300100F9

S1136A1E6F71009C0FA05C00134A40227A000000A5
S1136A2E00880AF07A01000000015C0013367A0038
S1136A3E00FFE87E010069011B71010069810100FD
S1136A4E6B2000FFE87E4ED401006B2000FFE8763A
S1136A5E01006B2100FFE8721A9001006FF0009C99
S1136A6E01006F7100A00A8101006FF100A00F8079
S1136A7E4F3601006F71009C01006B2000FFE8721E
S1136A8E5C0012E040227A00000000880AF07A01CE
S1136A9E000000015C0012CC7A0000FFE8820100C6
S1136AAE69011B710100698101006B2000FFE882FF
S1096ABE4ED401006F71CC
S1116AC400AA01006F7000A01A8101006B2070
S1136AD200FFE8765C00129840227A0000000088EA
S1136AE20AF07A01000000015C0012847A0000FFC0
S1136AF2E886010069011B710100698101006B20B5
S1136B0200FFE8864ED47C407310473601006B20A9
S1136B1200FFE88A4F2C40207A01000000017A002E
S1136B2200FFE8685C0012487A0000FFE88A01006F
S1136B3269011B710100698101006B2000FFE88A72
S1136B424ED60B76404001006FF600A6400E0100C0
S1136B526F7000A60B7001006FF000A601006F704A
S1136B6200A668086EF8009547086E780095A82578
S1136B7246DC01006F7100A61AE10FE05C0011F020
S1136B8201006F7600A6686847145C0012900D003E
S1136B92460C6A2800FFE85E73085870F5D45C005F
S1136BA2125E6B2000FFE8667A17000000B25E00F7
S1136BB2605054705E0060727A37000000247A05D8
S1096BC2000000040AF5C7
S1136BC801006FF0001801006FF1002001006FF160
S1136BD8001C18AA689A6F72003C0C22463CAA58FB
S1136BE8472CAA644712AA69470EAA6F4712AA75C7
S1136BF84706AA78471840227A000000000A400690
S1136C087A000000000801006FF00014400C7A00BD
S1136C180000001001006FF000146F70003C792031
S1136C2800644706792000694612010 06F70001856
S1136C384C0A01006F74001817B4400601006F7402
S1136C4800181AE61AB30FC4467401006B2000FF3C
S1136C58E86E476AF83068D87A0600000001406297
S1136C680FD00AE00F82010069F00FC001006F71B5
S1136C7800145E00852E0100697068890FA06808FA
S1136C88A8094E0C0FD00AE0680989306889401EAC
S1136C986F70003C79200078460A0FD00AE0680933
S1136CA8895740080FD00AE06809893768890FC0F7
S1096CB801006F710014DE
S1116CBE5E00852E0F840B760FC4469E6A2857
S1136CCC00FFE85E732847626F70003C0C00465A65
S1136CDCA858471EA86F4706A8784716404C0100D2
S1136CEC6F70001847440FE00B760AD0F9306889AF
S1136CFC403801006F700018473001006F7000209E
S1136D0C0B7001006FF000201B70F93068890100D3
S1136D1C6F7000200B7001006FF000201B706E79F8
S1136D2C003D68897A03000000020FB00AE00F836C
S1136D3C01006FF600146F70003C79200064470665
S1136D4C79200069467201006B2000FFE86E46084B
S1136D5C01006F700018476001006F7000184D42FE
S1136D6C6A2800FFE85E733847160B7301006F70D7
S1136D7C00200B7001006FF000201B70F92B4036C4

S1136D8C6A2800FFE85E7348472E0B7301006F708F
S1136D9C00200B7001006FF000201B70F920688934
S1136DAC40160B7301006F7000200B7001006FF025
S1096DBC00201B70F92DFD
S1136DC268897A0600FFE87601006F70001C680C80
S1136DD2AC2B4708AC2D4704AC20460E01006F7064
S1136DE2001C0B70010069E040466A2800FFE85E60
S1136DF2732847326F70003C0C004634A858470A88
S1136E02A86F4714A8784702402601006F70001C40
S1136E120BF0010069E0401801006F70001C0B7059
S1136E22010069E0400A01006F70001C010069E083
S1136E327A0400FFE86E010069401FB04F1401009D
S1136E426946010069441AB401006BA400FFE88299
S1136E52400C0FB61A8001006BA000FFE8827A048F
S1136E6200FFE88A01006F70001C680BAB2B470818
S1136E72AB2D4704AB20463801006B2000FFE86AC4
S1136E821FE04F2C6A2800FFE868A8304622010061
S10D6E926B2000FFE86A1AE07A01A2
S1136E9C00FFE882010069120A82010069921A80DC
S1136EAC010069C0401E01006B2000FFE86A1FE06F
S1136EBC4F0C01006B2000FFE86A1AE040021A80B5
S1136ECC010069C001006F7600141B76401A0100A3
S1136EDC6F7000200B7001006FF000201B700FE12E
S1136EEC1B760AD1681968890FE64CE201006F70B2
S1136EFC0020189968897A17000000245E006050FE
S1136F0C54705E0060727A37000000647A0300FFED
S1136F1CE85E7A04000000040AF47A0500FFE86EC8
S1136F2C0F866FF1005001006FF6005CF83068C8F3
S1136F3C7A000000007C0AF07A010000A0205E00B9
S1136F4C60420D00587000B87A000000002F0AF060
S1136F5C01006DF07A000000002E0AF001006DF0C4
S1136F6C01006F70008801006DF001006F700088E4
S1136F7C01006DF07A01000000320AF17A00000082
S1096F8C00115E007F4AC4
S1136F927A17000000100D024752188868E80100B2
S1136FA2695001006B2100FFE86A1F904F06010040
S1136FB26950400801006B2000FFE86A01006BA0E2
S1136FC200FFE88A7A0600FFE8680D207920FFFFB8
S1136FD2470C79200001460CF82B5A007CFAF82D55
S1046FE25A51
S1136FE3007CFAF82A68E85A007CFC7A0000000067
S1136FF31101006DF00FC10B717A00000000260A26
S1137003F05E0083300B97400CF8016EF8002F18E5
S1137013886EC800017A000000000101006FF000D0
S113702360400E01006F7000600B7001006FF00091
S11370336001006F7000600AC06808A83047E4016C
S1137043006F7000607A20000000014F7A01006F27
S11370537000600AC05E0059B80F8201006F7000B0
S1137063601B7001006F71002A0A8101006FF10038
S11370732A7A000000000101006FF000584030013C
S1137083006F7000600AC001006F7100580AC16885
S113709308689801006F7000580B7001006FF000CF
S11370A35801006F7000600B7001006FF000600106
S11370B3006F7000580FA11F904FC401006F700041
S11370C3580AC0189968890FC00B7001006FF0004C
S10870D3445E0059B802
S11370D801006FF000601A8001006FF00058010092

S11370E86F7000607A01000000025E007F040100F7
S11370F86FF0004001006F70004401006F71006081
S11371080A901B7001006FF0003C404E01006F7045
S1137118004401006F7100580A9001006FF0004CA1
S113712868086EF8005701006F71003C01006F7228
S113713800581AA101006FF1004801006F72004C5A
S1137148681968A901006F710048689801006F7198
S113715800580B7101006FF1005801006F7000585F
S113716801006F7100401F904DA201006F70006015
S1137178461AF8306EC800017A00000000010100C9
S11371886FF000601A8001006FF0002A6F700050E2
S1137198792000674706792000474 63C7D307050C8
S11371A801006F70002A01006F7100600A901B7064
S11371B87A20FFFFFFFC4D0C01006B2100FFE86EF6
S10971C81F904F0C6F70D5
S11371CE00501BD06FF000504008790000666FF03E
S11371DE00506E78002FA80146246838E8186EF820
S11371EE0057A8084706A810470A401A0FE00B7667
S11371FEF92B40100FE00B76F920688940080FE059
S113720E0B76F92D68896F700050792000665860EF
S113721E077201006F70002A58D001861A80401A37
S113722E0FE00B7601006F7100580AC10B716819DC
S113723E688901006F7000580B7001006FF00058E1
S113724E01006F7100601F904DD61A800F820100EE
S113725E6BA600FFE87601006F70002A01006BA099
S113726E00FFE8827C307350460A01006B2000FF5A
S113727EE86E4E067C307320470E0FE00B76F92E28
S113728E68890FA00B700F827C307350470C7C30D3
S113729E735047247C307320471E01006BA600FFFA
S11372AEE87A0100695001006BA000FFE886010037
S10972BE69500FA10A81D3
S11372C40F9201006F70002A0FA10A9001006F71E1
S11372D400600A9001006FF0006001006F71005CB0
S11372E46819A92B4708A92D4704A9204652010070
S11372F46B2000FFE86A01006F7100601F904F402C
S11373046A2800FFE868A830463601006F70005C05
S10973140B7001006BA0E9
S113731A00FFE87201006B2000FFE86A01006F7149
S113732A00601A9001006BA000FFE87E1A8001003A
S113733A6BA000FFE88A5A007CF801006B2000FF6B
S113734AE86A01006F7100601F904F4C01006B20C7
S113735A00FFE86A01006F7100601A9001006BA0D8
S113736A00FFE88A6E78002FA80146186E78002F6E
S113737AA801586009786E780057A8084706A8102C
S113738A5860096A7A0000FFE88A010069011B71E3
S113739A010069815A007CF81A8001006BA000FF82
S11373AAE88A5A007CF801006F70006001006F716F
S11373BA002A0A8101006FF1002A010069520AA119
S11373CA01006FF100521F814C120F914D0E010003
S11373DA6F7100520B710FC05C00092201006F70BC
S11373EA002A58F0028A6848A83046147A00000036
S11373FA000101006FF0004C01006FF00052402CB5
S109740A1A8001006FF07F
S1137410004C1A8001006FF0005201006F70002AC7
S11374200B7001006FF0002A01006F7000600B7099
S113743001006FF000601A8001006FF00058403ABD
S11374400FE00B7601006F71004C0AC1681968895F

S113745001006F70002A1B7001006FF0002A010009
S11374606F7000580B7001006FF0005801006F70CF
S1137470004C0B7001006FF0004C01006F70002A8C
S11374804EBE0FE00B76F92E688940240FE00B7691
S113749001006F71004C0B7101006FF1004C1B7107
S11374A00AC168196889010069501B70010069D01D
S11374B001006F7000580B7001006FF0005801005D
S11374C06F70004C01006F7100521A9001006F71D0
S11374D000601F904C0A01006B2000FFE86E4EAC69
S11374E07C307350470C7C307350472E7C307320B4
S11374F0472801006950010 06F7100580A8101009B
S10775006FF10058CC
S107750401006BA66E
S113750800FFE8760100695001006BA000FFE882E4
S113751840226E68FFFFA830461A40101B76010010
S11375286F7000581B7001006FF000586E68FFFF02
S1137538A83047E87C307320464A6E68FFFFA82EC0
S1137548464201006B2000FFE88247067C307350F7
S113755847321B7601006F70005801006B2100FF52
S1137568E8821A901B7001006FF000581A8001001E
S11375786BA000FFE88201006F70005C01006BA044
S113758800FFE87601006F70005C6808A82B4708C5
S1137598A82D4704A820465001006B2000FFE86A85
S11375A801006F7100581F904F3E6A2800FFE8687A
S11375B8A830463401006F70005C0B7001006BA0AB
S11375C800FFE87201006B2000FFE86A01006F7199
S11375D800581A9001006BA000FFE87E1A80010092
S11375E86BA000FFE88A406201006B2000FFE86A95
S10975F801006F71005851
S10B75FE1F904F4601006B20B2
S113760600FFE86A01006F7100581A9001006BA031
S113761600FFE88A6E78002FA80146146E78002FC3
S1137626A80146286E780057A8084704A810461CE8
S11376367A0000FFE88A010069011B710100698174
S1137646400A1A8001006BA000FFE88A01006B2044
S113765600FFE87601006F71005C1F905860069288
S113766601006B2000FFE87201006BA000FFE876C3
S11376765A007CF86848A83046147A0000000001D6
S113768601006FF0004C01006FF00052402A1A808F
S113769601006FF0004C01006FF0005201006F70A3
S11376A6002A0B7001006FF0002A01006F70006062
S11376B60B7001006FF0006001006F70002A4F1419
S11376C67A000000000101006FF0004C0FE00B761A
S11376D6684940060FE00B76F93068897A000000A6
S11376E6000101006FF000580FE10B76FA2E689A3D
S10976F601006F70005853
S11376FC0B7001006FF0005801006F70002A58C026
S113770C009E01006BA600FFE87601006F70002A53
S113771C17B0010069511F814F2001006F70002ABF
S113772C17B001006BA000FFE88201006F70002A04
S113773C01006F7100581A81401801006950010053
S113774C6BA000FFE8820100695001006F710058C3
S113775C0A8101006FF1005801006F70002A0100CB
S113776C69510A81010069D140360FE00B760100A3
S113777C6F71004C0AC168196889010069501B704C
S113778C010069D001006F7000580B7001006FF09D
S113779C005801006F70004C0B7001006FF0004C2F

S11377AC01006F70004C01006F7100521A900100C0
S11377BC6F7100601F904C0A01006B2000FFE86E94
S11377CC4EA87C307350470C7C30735047347C305C
S11377DC7320472E010069504F4A01006BA600FF2E
S10977ECE87A0100695078
S10777F201006BA084
S11377F600FFE88601006950010O6F7100580A8195
S113780601006FF1005840226E68FFFFA830461A48
S113781640101B7601006F7000581B7001006FF05B
S113782600586E68FFFFA83047E87C307320467027
S11378366E68FFFFA82E466801006B2000FFE882F2
S1137846460A01006B2000FFE88647067C3073502A
S1137856474E1B7601006F70005801006B2100FF35
S1137866E8821A9001006B2100FFE8861A901B70CC
S113787601006FF000581A8001006BA000FFE88634
S113788601006BA000FFE88201006F70005C01003D
S11378966BA000FFE8761A8001006BA000FFE87A70
S11378A601006F70005C6808A82B4708A82D4704E1
S11378B6A820465001006B2000FFE86A01006F71A3
S11378C600581F904F3E6A2800FFE868A8304634E8
S11378D601006F70005C0B7001006BA000FFE87283
S10778E601006B200F
S11378EA00FFE86A01006F7100581A9001006BA04B
S11378FA00FFE87E1A8001006BA000FFE88A40625D
S113790A01006B2000FFE86A01006F7100581F90A5
S113791A4F4601006B2000FFE86A01006F710058AF
S113792A1A9001006BA000FFE88A6E78002FA80165
S113793A46146E78002FA80146286E780057A808C7
S113794A4704A810461C7A0000FFE88A010069016F
S113795A1B7101006981400A1A8001006BA000FFB4
S113796AE88A01006B2000FFE87601006F71005C72
S113797A1F90461001006B2000FFE87201006BA004
S113798A00FFE8765A007CF8010069500B70010089
S113799A6F7100601F904C0C010069510BF10FC00D
S11379AA5C00035A6848A830460E7A0000000001BA
S11379BA01006FF0004840241A8001006FF000486C
S11379CA01006F70002A1B7001006FF0002A01008A
S11379DA6F7000600B7001006FF0006001006F7040
S10979EA006001006F7153
S11379F000481A9001006F72002A0A8201006FF298
S1137A00002A0FE00B760AC1681968897A00000022
S1137A10000101006FF000580FE10B76FA2E689A0F
S1137A2001006F7000580B7001006FF000580100E7
S1137A306F700048402E0FE00B7601006F71004C11
S1137A400AC1681968890100695O1B70010069D077
S1137A5001006F7000580B7001006FF000580100B7
S1137A606F70004C0B7001006FF0004C01006F71E0
S1137A7000601F904E0A01006B2000FFE86E4EB6B7
S1137A807C307350470C7C307350472E7C3073200E
S1137A90472801006BA600FFE8760100695001004A
S1057AA06BA0D6
S1137AA200FFE88201006950010O6F7100580A81EA
S1137AB201006FF1005840226E68FFFFA830461A9A
S1137AC240101B7601006F7000581B7001006FF0AD
S1137AD200586E68FFFFA83047E87C307320464A9F
S1137AE26E68FFFFA82E464201006B2000FFE8826A
S1137AF247067C30735047321B7601006F70005883

S1137B0201006B2100FFE8821A901B7001006FF0E5
S1137B12005801006F70005C01006BA000FFE87663
S1137B221A8001006BA000FFE8826F700050792079
S1137B32006546080FE00B76F96540060FE00B7609
S1137B42F945688901006F7000580B7001006FF0EE
S1137B52005801006F70002A4D0A0FE00B76F92BD3
S1137B62688940160FE00B76F92D688901006F7062
S1137B72002A17B001006FF0002A01006F7000584D
S1137B820B7001006FF0005801006F70002A7A2019
S1097B92000000644D0E2B
S1137B980BF601006F7000580B800B7040140FE058
S1137BA80B76F9306889F83068E801006F7000587F
S1137BB80BF001006FF000580FE00B7001006FF03D
S1137BC8003040360FE01B76010069F001006F704A
S1137BD8002A7A010000000A5E007F048930010050
S1137BE86970688901006F70002A7A010000000A31
S1137BF85E007F0401006FF0002A01006F70002A05
S1137C084EC201006F76003001006F70005C680897
S1137C18A82B4708A82D4704A820465001006B202D
S1137C2800FFE86A01006F7100581F904F3E6A28F1
S1137C3800FFE868A830463401006F70005C0B70E1
S1137C4801006BA000FFE87201006B2000FFE86AE7
S1137C5801006F7100581A9001006BA000FFE87EC5
S1137C681A8001006BA000FFE88A406201006B20C4
S1137C7800FFE86A01006F7100581F904F4601002A
S1057C886B206C
S1137C8A00FFE86A01006F7100581A9001006BA0A7
S1137C9A00FFE88A6E78002FA80146146E78002F39
S1137CAAA80146286E780057A8084704A810461C5E
S1137CBA7A0000FFE88A010069011B7101006981EA
S1137CCA400A1A8001006BA000FFE88A01006B20BA
S1137CDA00FFE87601006F71005C1F9046100100F7
S1137CEA6B2000FFE87201006BA000FFE87618889A
S1137CFA68E87A17000000645E00605054705E0002
S1137D0A60720F850F960FD00AE06808A8344F52A5
S1137D1A0FD00AE0F93068891B760FD00AE06809A8
S1137D2A8901688940200FD00AE06808A8394F14EE
S1137D3A0FD10AE1681888F668986E18FFFF880160
S1137D4A6E98FFFF1B760FE64EDC6E580001A839CA
S1137D5A4F106E58000188F66ED8000168588801E2
S1137D6A68D85E00605054705E0060721B977A0395
S1097D7A00FFE8607A0639
S1137D8000FFE864010069F00F9501006B2100FF1B
S1137D90E85A7A2100000001462601006DF50F81A3
S1137DA0010069305E0084D40B97010069300AD06A
S1137DB0010069B0010069600AD0010069E0403A3E
S1137DC019444030010069700B70010069F01B70A9
S10F7DD0680817D00100693101006B2224
S1137DDC00FFE8925D207920FFFF471201006960E4
S1137DEC0B70010069E00B5417F41FD44DCA0B97A9
S1137DFC5E006050547001006B2000FFE85A7A203B
S1137E0C00000001460C01006B2000FFE86018998C
S1137E1C6889547001006B2000FFE85A7A20000037
S1137E2C000146041900547001006B2000FFE86048
S10F7E3C6E090010E94017D10D105470BE
S10BA020000000000000000035
S1137E4801006DF501006DF401006DF301006DF2A1

S1137E5801006DF101006D120100691301006F014A
S1137E680004010069000D840D8C796C7FF0792C76
S1137E787FF047540DAC796C7FF0792C7FF047562F
S1137E880D8C65AC4B5E1AB144087A3000000001D2
S1137E9845141AA0451001F06410475A0C444B0AC4
S1137EA879000002400C0C444BF619004004790099
S1137EB8FFFF01006D7101006D7201006D73010018
S1137EC86D7401006D7554700F85796D000F01F0A5
S1137ED8641546DA409E0FA5796D000F01F06435ED
S1137EE846CC409C01F064207A607FFFFFFF46ACDC
S10F7EF801F0643146A67900000140B699
S1137F046DF20D820C2A4A0217B00D994A04D280ED
S1137F1417B15E00852E0C224A0217B00CAA4A023E
S1097F2417B16D725470E9
S1137F2A01006DF301006DF20D8252120D9352039B
S1137F3A52100928093801006D7201006D735470DB
S113A028202020202020202020606060602020E5
S113A038202020202020202020202020202015
S113A048481010101010101010101010101010CD
S113A058848484848484848484101010101010106D
S113A068108181818181810101010101010101C6
S113A078010101010101010101010110101010107A
S113A088108282828282020202020202020297
S113A098020202020202020202020210101010203F
S113A0A8000000000000000000000000000000A5
S113A0B8000000000000000000000000000000095
S113A0C8000000000000000000000000000000085
S113A0D8000000000000000000000000000000075
S113A0E8000000000000000000000000000000065
S113A0F8000000000000000000000000000000055
S113A1080000000000000000000000000000000044
S109A1180000000000003E
S10DA11E000000000000000000000034
S1137F4A5E0060727A370000003C7A040000000C7D
S1137F5A0AF47A05000000140AF501006FF00038EC
S1137F6A01006FF1003401006F73005C7A020000B4
S1137F7A000801006FF2002C19226FF2002A7A0618
S1137F8A000000540AF6686AEA7F46620FE00B7043
S1137F9A7A01000000075E00856A0D0047501A80C7
S1137FAA401A01006F70003401006F7100380A90A3
S1137FBA1899688901006F7000380B7001006FF01F
S1137FCA00387A20000000084DD81A80010069B0F1
S1137FDA7C607370470A01006F700060F9FF400804
S1137FEA01006F700060F90168895A0083227A00E0
S1137FFA0000001C0AF001006DF07A01000000285D
S113800A0AF101006F70005C01006DF001006F70EE
S113801A005C01006DF001006F70006C5E0088FA6D
S113802A7A170000000C01006F7000606808A8FF4F
S107803A460C7A0073
S113803E000000540AF07D0072706F7000247920E6
S113804E07FF464A6E680001A8F0463A0FE00BF0B0
S113805E7A01000000065E00856A0D0047280100C4
S113806E6F7000606808A8014608790000015A0085
S113807E832401006F7000606808A8FF4610790022
S113808EFFFF5A008324790000025A0083246F7085
S113809E0024793003FF6FF000244D1E01006F7032
S11380AE005801006DF001006F70005801006DF073

S11380BE5E00868A0B970B97401E01006F70005867
S11380CE01006DF001006F70005801006DF05E004D
S11380DE868A0B970B971B500D0618886EF8002394
S11380EE0B560D6017F001006F7100381A900100E6
S11380FE69B0010069F10FD1010069705E0087A6B6
S113810E01006F70003801006F71002C5E007F0458
S113811E01006FF000301AE67A000000000801003B
S109812E6F7100301A908E
S11381340F82401401006F7000300AE00AD00FD19F
S11381440AE1680868980B760FA01F864DE6400A7B
S11381540FD00AE0189968890B767A260000000884
S11381644DEE6F7000326F78002E52806F71003ABB
S113817419016DF17A01000000080FD05E00872E0B
S11381840B8701006F7600381B760FC10FE05E008A
S113819487A60FE001006F71002C5E007F040100CD
S11381A46FF000301AE67A000000000801006F71D6
S11381B400301A900F82401401006F7000300AE0FF
S11381C40AC00FC10AE1680868980B760FA01F86DE
S11381D44DE6400A0FC00AE0189968890B767A269F
S11381E4000000084DEE6F7000326F78002E52804D
S11381F46F71003A19011B516DF17A0100000008F7
S11382040FC05E00872E0B8701006F7000340100DE
S11382146DF00100693101006DF17A010000002461
S10982240AF16F70002C4B
S113822A5E0085920B970B977A06000000040AF604
S113823A7A000000000801006DF001006F71003838
S113824A0FE05E0084D40B977A0000000008010057
S10A825A6DF00FD10FE05E90
S113826100899E0B970D004F12790000016FF000FA
S11382712A010069300B705A00831A7A000000004A
S1138281080 1006DF001006F7100380FE05E00849A
S1138291D40B977A000000000801006DF00FD10F95
S11382A1E05E00899E0B970D004632010069300B99
S11382B170010069B001006F70003401006DF001BD
S11382C100693301006DF37A01000000240AF16FA4
S11382D170002C5E0085920B970B9740447A000047
S11382E100000801006DF001006F7100380FE05EBE
S11382F10084D40B977A000000000801006DF00F91
S1138301C10FE05E00899E0B970D004C146F700046
S11383112A460E010069301B70010069B05A0082C0
S11283210C19007A170000003C5E006050547086
S11383305E0060727A370000002C0FF30F86010095
S11383406FF100107A000000000101006FF00018C7
S113835001006F7500440A95188868D87A000000F8
S1138360000801006DF00FE17A000000000C0AF034
S11383705E0084D40B9701006F7000445A0084C0E0
S11383800FA00B707A01000000055E007F040B70E4
S113839010307A06000000081A867A0000000008F0
S11383A01AE001006DF00FA11031103110317A1174
S11383B00000A1280AE10FB00AE05E0084D40B9705
S11383C07A000000000801006FF000281A80010005
S11383D06FF0002001006FF0001C7A040000000819
S11383E01AE401006FF400145A00848201006F70D4
S11383F0001401006DF00FB10AE17A000000000CD7
S11384000AF00AE05E00899E0B970D004D3C0100C7
S11384106DF40FB00AE001006DF07A010000001066
S10984200AF10AE11A80D3

S11384265E00884E0B970B9717F001006FF000184C
S113843601006F70002801006F7100200A8101009E
S11384466FF1002001006F7000287A01000000021E
S11384565E007F0401006FF00028473079000001B9
S11384666DF00FB00AE00FC15E0087D40B870100E1
S11384766F70001C0B7001006FF0001C01006F7021
S1138486001C7A200000000458D0FF5A01006F70C8
S11384960018461C6E78002388306CD84004F830E8
S11384A668D81B7501006F7000101F8544F04012D9
S11384B66E78002388306CD80FA01B700F8258C0CB
S11184C6FEB87A170000002C5E006050547060
S113A1280000000000000000800000000000000050CC
S113A13800000000000003200000000000001F4092
S113A1480000000000001388000000000000C35000A
S113A15800000000007A12000000000004C4B400EC
S113A168000000002FAF080000000001DCD65000FB
S113A17800000012A05F2000000000BA43B74000AF
S113A18800000746A5288000000048C27395000018
S113A1980002D79883D20000001C6BF52634000018
S10BA1A8011C37937E0800003F
S11384D45E0060720F850F9401006F7600181FC54C
S11384E447401FC544200FD30FC21AC440120FB014
S11384F40B730FA10B710F921B71681968890B74AD
S11385041FE445EA401C0FD30AE30AE40FC21AC46A
S1138514400C0FA01B700F8268086CB80B741FE427
S10D852445F00FD05E006050547064
S113852E01006DF20D9946100D82177253120DA8AC
S113853E53100D810D28401E0F920D81177179086E
S113854E0010121012311AA144020AA11B5846F24E
S10F855E12101710177001006D7254709A
S113856A5E0060720F840F951AE6400E0FC00AE090
S113857A680947041900400A0B761FD64DEE7900A5
S10B858A00015E006050547013
S11385925E0060727A370000000C7A050000000169
S11385A20AF50D0C0F967904000801006DF57A01A6
S11385B20000000E0AF101006F70002817B05E0080
S11385C28A640B9701006DF66DFC7A0000000010BF
S11385D20AF00FD15E0089DC0B970B87790C003F01
S11385E26F70000A190C0DC017F001D053400D0E25
S11385F27900004219C017F001006DF07A01000002
S113860200090FD05E008CB00B9779060008401466
S11386120D6019E017F00AD00D6117F10AD168084D
S113862268981B561DE64CE8400C0D6017F00AD003
S1138632189968891B560D664CF00DE05240190CCF
S11386426DFC7A01000000090FD05E0087D40B870E
S11386527900003F6FF0000A7A01000000400FD05A
S11386625E008BC67A000000000801006DF00FD196
S113867201006F70002C5E0084D40B977A17000000
S1098682000C5E006050D5
S1058688547029
S113868A01006DF27A370000001A7A000000001820
S113869A0AF001006F71002601006DF101006F718C
S11386AA002601006DF17A01000000080AF10100B9
S11386BA6DF15E008D4C7A170000000C6F71001883
S11386CA0FF05E008FCC0F817A020000A1B07A000E
S11386DA000000080AF05E0090B87A00000000105B
S11386EA0AF001006F71000C01006DF101006F7156

S11386FA000C01006DF17A01000000080AF1010083
S113870A6DF15E008E527A170000000C7A000000A9
S113871A00100AF05E005F107A170000001A0100C9
S107872A6D725470A5
S10BA1B03FD34395810624DD32
S113872E5E0060721B971B87010069F00F946F79CF
S113873E001E790D0008199D4754790100FF0DD2D3
S113874E1A0A4B04101140F80C9B18DD1B740FC64C
S113875E4028010069740AE468450CB816850FC0F9
S113876E0D915E008F88684814D868C80DD01A081A
S113877E4B04110540F80C5D1B760FE64CD40CDD53
S113878E4706790100014002191117F10F900B876B
S10B879E0B975E00605054705C
S11387A65E0060720F860F957A00000000080100D4
S11387B66DF01036103610367A010000A1B80AE1C2
S11187C60FD05E0084D40B975E006050547099
S113A1B8000000000000000010000000000000058E
S113A1C80000000000000001900000000000000007DEE
S113A1D800000000000000271000000000000C35C0
S113A1E80000000000003D09000000000001312DBF
S113A1F8000000000005F5E100000000001DCD652A
S113A2080000000009502F90000000002E90EDDDD
S113A218000000000E8D4A510000000048C27395EB
S113A228000000016BCC41E9000000071AFD498DCD
S113A2380000002386F26FC1000000B1A2BC2EC546
S113A248000003782DACE9D900001158E460913D72
S113A258000056BC75E2D6310001B1AE4D6E2EF545
S113A268000878678326EAC9002A5A058FC295ED44
S113A27800D3C21BCECCEDA10422CA8B0A00A425AD
S113A28814ADF4B7320334B96765C793FA10079D61
S11387D45E0060727A370000000A0F83010069F1BA
S11387E46F790022790C0008199C4752FAFF0DC0D7
S11387F41A084B04110A40F80CA218CC1AE64026B6
S11388040FB50AE568540C2816840FD00D915E0049
S11388148FAA685814C868D80DC01A084B041004EA
S113882440F80C4C0B76010069701F864DD20CCCBA
S1138834470679010001400219 1117F10F907A17C5
S10D88440000000A5E00605054704B
S113884E5E0060727A370000000C01006FF00004C6
S113885E0F9601006F7500280AD61B7601006F7301
S113886E00240AD31B737A02000000011AC4404489
S113887E6868175068391751191017F01AC001009C
S113888E69F04C14010069717A11000001000100B6
S113889E69F17900000140021900 17F00F840100FD
S11388AE697047041A800F826E78000368E81B759F
S11388BE1B761B730FD546B87A2400000001460EB3
S11388CE01006F70000446067900FFFF40120FA0EF
S11388DE7A2000000001460419004004790000 01CB
S10F88EE7A170000000C5E00605054700C
S11388FA5E0060727A03000000070F820F95010081
S113890A6F7600207A04000000180AF4694179613D
S113891A8000792180004606790 1FFFF400479012E
S113892A00010FA06889694079607FF01190119066
S113893A1190119069D07A000000000701006DF0D0
S113894A0B740FC10FE05E0084D40B977900000308
S113895A6DF00FB10FE05E00872E0B8769504F064B
S113896A7D607070402869500B5069D07D607270C9

S113897A4016790000016DF00FB10FE05E00872EFB
S113898A0B8769501B5069D07C60737047E45E00A3
S107899A6050547062
S113899E5E0060720F860F9301006F7400180FE56F
S11389AE0FC44604190040201AE6400A6C586C396D
S11389BE1C9846060B761FC645F21B756858175052
S11189CE1B73683B175319305E0060505470E2
S11389DC5E0060727A37000000100FF60F850F945B
S11389EC69506F710028091069D001006DF6010000
S11389FC6F71002E0FC05E0093940B977C607370A5
S1138A0C470869500B5069D0400C7A0100000010E4
S1138A1C0FE05E0093427A000000004201006DF00B
S1138A2C7A01000000100FE05E008CB00B976E68AB
S1138A3C0008E8E06EE800087A0000000009010075
S1138A4C6DF00FE10FC05E0084D40B977A17000012
S10B8A5C00105E00605054702D
S1138A645E0060727A37000000187A05000000097E
S1138A740AF50F840F920FC44D087A030000000116
S1138A8440147A03FFFFFFFF0FC07A01FFFFFFFFCC
S1138A945E007F2A0F840FC07A010000001B5E0072
S1138AA47F040F860FC07A010000001B5E007F0461
S1138AB40F947A2300000001462E7A000000000878
S1138AC401006DF00FE11031103110317A11000003
S1138AD4A2980FD05E0084D40B970FE01030780077
S1138AE46B210000A36840320FC446021B767A0050
S1138AF40000000801006DF00FE110311031103156
S1138B047A110000A3000FD05E0084D40B970FE00A
S1138B14103078006B210000A3826FF100120FC4A0
S1138B2447747A23FFFFFFFF460A7A000000001B05
S1138B341AC00F8401006F70003001006DF00FA1A3
S1138B440FC05E0095220B970FE646087A230000B8
S1078B540001476270
S1138B5801006F70003001006DF00FA069006DF027
S1138B687A00000000180AF00FD15E0089DC0B9729
S1138B780B877A01000000400FD05E008BC6792076
S1138B880001460E6F7000120B506FF000127D50FB
S1138B9870700FA06F71001269817A0000000008DD
S1138BA801006DF00FD101006F7000345E0084D4B2
S1118BB80B977A17000000185E00605054708F
S113A29880000000000000000CECB8F27F4200F3A87
S113A2A8A70C3C40A64E6C5286F0AC99B4E8DAFD94
S113A2B8DA01EE641A708DEAB01AE745B101E9E4F0
S113A2C88E41ADE9FBEBC27DE5D3EF282A242E812D
S113A2D8B9A74A0637CE2EE195F83D0A1FB69CD991
S113A2E8F24A01A73CF2DCD0C3B8358109E84F072D
S113A2F89E19DB92B4E31BA99E74D1B791E07E4803
S113A308C428D05AA4751E4CF2D56790AB41C2A499
S113A318964E858C91BA2655BA121A4650E4DDEC4E
S113A328E65829B3046B0AFA8E938662882AF53FA6
S113A338B080392CC4349DEDDA7F5BF5909668497B
S113A348873E4F75E2224E68A76C582338ED2623C3
S113A358CF42894A5DCE35EB8049A4AC0C5811AE87
S113A3680000005900B3010D016601C0021A02730F
S113A37802CD0327038003DA0434FFA6FF4CFEF261
S109A388FE99FE3FFDE516
S111A38EFD8CFD32FCD8FC7FFC25FBCBFB7263
S1138BC65E0060727A370000000C0F840F957A06F8

S1138BD6000000081AB30FD00FE15E007F040AC03D
S1138BE60F820FD00FE15E007F04790000801A091F
S1138BF64B04119040F86EF8000511086EF8000B4F
S1138C0611086EF800041B75010069F50FD00FE11A
S1138C165E007F040AC00F85010069700FE15E00E4
S1138C267F04790600801A094B04119640F80FA0B9
S1138C3668086E790005169847280FA068066E78AF
S1138C46000B166846086E780004166847087A0310
S1138C5600000001400C685816E847067A03000036
S1138C6600017A2300000001463240140CE8175035
S1138C761710685916896 8D9100E46041B75FE012C
S1138C86685816E847041FC544E21FC545086858D7
S1138C9614E868D8400679000001400219007A17E3
S10D8CA60000000C5E0060505470E3
S1138CB05E0060721B971B970F82010069F17A03B4
S1138CC00000000801006F7000200FB15E007F04F8
S1138CD00FA60A8601006F7000200FB15E007F04AB
S1138CE0790400801A094B04119440F8686816C887
S1138CF046500CCD1855400414D5110D0CDD46F823
S1138D0068681658470868 6814C868E840340FA0B4
S1138D10010069710A90010069F001006F70002081
S1138D200FB15E007F040FA50A85400C68684708F1
S1138D30685814C868D8400A0B76010069701F860A
S10F8D4045EA0B970B975E0060505470DF
S1138D4C5E0060720F857A040000000701006F70EB
S1138D5C002001006DF001006F70002001006DF028
S1138D6C5E0092F00B970B970D064748792600018E
S1138D7C460A790004B86BA000FFE89A7926000232
S1138D8C460A7900044C6BA000FFE89A19667A0036
S1138D9C0000001C0AF00D6117F10A90F9FF6889B5
S1138DAC0B56792600084DE67A000000001C0AF0E9
S1138DBC5A008E427A000000001C0AF07A0100006F
S1138DCCA39C5E005F8A0D00470C190069D07A00E2
S1138DDC0000A39C40607A060000001C0AF669633D
S1138DEC1113111311131113796307FF793303FE55
S1138DFC69D36950791003FE462869500B5069D02A
S1138E0C69607960800F69E00FE30B73400E0FC14B
S1138E1C0FB05E00934269501B5069D06960734870
S1138E2C47EC69607960800F79403FE069E07A0034
S1098E3C0000001C0AF017
S1138E4201006F7100185E005FFC5E006050547099
S10BA39C0000000000000000B6
S1138E525E0060720F8601006F70002001006DF0EA
S1138E6201006F70002001006DF05E0092F00B971D
S1138E720B970D05474079250001460A790004B88E
S1138E826BA000FFE89A79250002460A7900044C98
S1138E926BA000FFE89A19667A000000001C0AF032
S1138EA20D6117F10A90F9FF68890B5679260008BC
S1138EB24DE65A008F707A050000001C0AF56955C9
S1138EC2111511151115115115796507FF793503FF71
S1138ED20D5C4C0E7A000000A3A40FE15E005FFC60
S1138EE240607A000000001C0AF00FE15E005FFCA4
S1138EF2792C00344C4C0FE50B950BF579090034B2
S1138F0219C90D9017F07901001001D053100D0902
S1138F12400A0FD01BF5191169811B590D994EF2A5
S1138F227900003419C017F07901001001D05310F1
S1138F327909FFFF1B584B04101940F869506690DA

S1078F4269D07A0174
S1138F460000001C0AF10FE20F905E005A9E0FE02C
S1138F567A010000A3A45E005FA60D00470C7A0009
S1138F660000001C0AF07D0070707A000000001CEF
S1138F760AF001006F7100185E005FFC5E0060502E
S1058F86547022
S10BA3A40000000000000000AE
S1138F8801006DF20D1A4F14792A00084D04FA00F6
S1138F984008680A100A1B5A4EFA688A01006D7263
S1058FA8547000
S1138FAA01006DF20D1A4F14792A00084D04FA00D4
S1138FBA4008680A110A1B5A4EFA688A01006D7240
S1058FCA5470DE
S1138FCC01006DF119990D11471A4A067909080028
S1138FDC17917919040F1B59101144FA10311031E0
S1138FEC10311031010069811A9101006F81000465
S1098FFC01006D715470C9
S11390020F8501F064155860009A792407FF47140D
S11390120D44586000820FA501F064355860007852
S1139022580000800FA501F064355860007640684F
S11390320FA501F06435466C0DCC465C0F8501F03B
S1139042641546544O5E0F8501F06415473C1031A8
S11390521230792800104C0C1B5C10311230792825
S113906200104DF4580000C40FA501F06435471AEF
S11390721033123279 2A00104C0C1B541033123263
S1139082792A00104DF4580000A81AA21AB3687D79
S1139092100D131A58000228790107FF1AA21AB3F6
S11390A2580001FA790107FF1AA27A0300000008A7
S11390B268FA580001E801006DF601006DF5010040
S11390C26DF401006DF301006DF201006DF1010019
S11390D26DF07A3700000018691D692565D569F5B9
S11390E2691C111C111C111C111C796C07FF6924CA
S10990F211141114111406
S11390F81114796407FF01006D1001006911796883
S1139108000F01006F23000401006922796A000F30
S1139118792C07FF5870FEE2792407FF5870FF0A7D
S11391280DCC5870FF1A0D445870FF36094C793C22
S113913803FF792C07FF58C0FF58792CFFCB58D071
S1139148FF426FFC000279480010794A00100100C1
S11391586FF000040F9601006FF2000801006FF32F
S1139168000C0D1552356FF500160D34529452B19B
S11391780A946511990009D44406791C00019900E1
S11391886FF400140DC50D1D18990D0452340AC54A
S113919899000DE452B40AC599000D6452240AC516
S11391A899006FF500120DD50D1D18990D845234D1
S11391B80AC50D0452B40AC599000DE452240AC520
S11391C899000D6452A40AC599006FF500100DD5D6
S11391D80D1D18990DB352830AB50D0452240AC5FF
S10991E899000DE452A4FE
S11391EE0AC59900528209D24404791A0001091A58
S11391FE6F74000852040AC26F7400085284094A3D
S113920E0D5B6F73001001006F7400126F7D0016FB
S113921E19556F710002113213333133413350DA028
S113922E7960010047121132133313341335 0B5186
S113923E792107FF5870FE5401F064454702700B05
S113924E0D114E2E113213334402700B0D114A04BD
S113925E0B5140F0732B47180CB8E80B47127A13D7

S113926E00000008440A0B72792A00804D020B514C
S113927E4020732B471C0CB8E80B47167A130000DB
S113928E000844020B72792A01004D061132133382
S113929E0B5111321333113213331132133796AE3
S11392AE000F10111011101110111011641A101A687892
S11392BE1008131A7A170000001801006D700100D0
S11392CE698201006F83000401006D7101006D72EC
S10792DE01006D73A8
S11192E201006D7401006D7501006D7654700E
S11392F001006DF57A01000000080AF16818E87FA3
S1139300A87F460A0B716818E8F0A8F04704190013
S1139310402A6818F01050006898460A0B711AD555
S1139320400E6818470679000001401 00B710B7559
S11393307A25000000064DEA7900000201006D75F0
S1059340547064
S11393425E0060720F840F931AD51B730FB6402809
S11393520FC30AE30FB26838E880175017700F8300
S11393620FA06809100968890FD547080FC00AE0E2
S11393727D0070000FB51B760FE64CD40FD5470660
S113938279010001400219111 7F10F905E0060503C
S1059392547012
S11393945E0060727A37000000387A05000000042A
S11393A40AF50F840F967A000000001001006DF097
S11393B4191101006F7000545E0096060B970FE3BA
S11393C40B930BF37A160000000701006FF60034C9
S11393D4790600030FC00B900BF001006FF0002817
S11393E47A140000000701006FF4002C5A00951052
S11393F46838460C01006F7000346809587000FA2D
S11394047A000000001001006DF019110FD05E0006
S113941496060B9701006F70002801006FF000306F
S113942401006F74002C790E0003407C01006F70FF
S113943400301A916809FA0810311A0A4EFA1A8090
S1139444684801F064101A916839FA0810311A0A4D
S11394544EFA01006F720034010069F01A80682823
S113946401F06401010069705E007F2A01006FF05E
S113947400247A000000000401006DF07A0100006A
S109948400280AF10DE2CD
S113948A096217F210320AD20FA05E0095BE0B973B
S113949A1B5E01006F7000301BF001006FF000309B
S11394AA1BF40DEE4C807926000346247A00000053
S11394BA000A01006DF00D6417F40FC110310AD1CF
S11394CA01006F7000540AC00AC05E0084D44022AF
S11394DA7A000000000A01006DF00D6417F40FC151
S11394EA10310AD101006F7000540AC00AC05E002D
S11394FA95BE0B971B561BF301006F7000341BF0CC
S113950A01006FF000340D6658C0FEDE7A170000C2
S10B951A00385E00605054703C
S11395225E0060727A370000000C0F860F940D60A4
S11395327910003F69C00FF10FE05E0087A67A0041
S1139542000000801006DF0191101006F7000287E
S11395525E0096060B971A800F8219550FF640107C
S11395620B760FA00B700F8269407930000869C037
S1139572686847ECFB804004110B0B55686816B80A
S113958247F66DF57A03000000080FA01A830FB1A6
S11395920FE05E00872E0B876940195069C00100F6
S11395A26DF30FE101006F7000285E0084D40B9706
S10F95B27A170000000C5E00605054703B

S11395BE5E0060720F860F9401006F7500180AD655
S11395CE1B760AD41B740FC319CC4022686817503C
S11395DE68391751091009C00D0468E817F47900AA
S11395EE010001D053040D4C1B751B761B730FD555
S10B95FE46DA5E006050547070
S11396065E0060721B870F8469F101006F73001A95
S11396160FC51AE6400C0FD00B756E7900016889E9
S11396260B761FB645F00FC00B875E006050547073
S9030000FD

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

LINK COMMAND LINE

LNK -subcommand=linkfile.sub

LINK SUBCOMMANDS

```
OUTPUT c_thread
PRINT c_thread
INPUT asmfile, main, EV_Simulator, EV_Puls, EV_Controller, EV_Input, EV_Display, EV_OpenClose,
EV_UpDown, EV_File, EV_Time, Timer, Panel, sci, lcd, usb
LIB c:\h8\akic\c38hab
START R(0FFE000), P(200), D(99C0), C(9A00)
ROM (D, R)
EXIT
```


*** LINKAGE EDITOR LINK MAP LIST ***

SECTION	NAME	START - END	LENGTH	UNIT NAME	MODULE NAME
ATTRIBUTE : CODE NOSHR					
V		H'00000000 - H'000000F3	H'000000F4	asmfile	asmfile
* TOTAL ADDRESS *		H'00000000 - H'000000F3	H'000000F4		
ATTRIBUTE : CODE NOSHR					
P		H'00000200 - H'000002AF	H'000000B0	asmfile	asmfile
		H'000002B0 - H'000012D9	H'0000102A	main	main
		H'000012DA - H'00001799	H'000004C0	EV_Simulator	EV_Simulator
		H'0000179A - H'00001B15	H'0000037C	EV_Puls	EV_Puls
		H'00001B16 - H'00002491	H'0000097C	EV_Controller	EV_Controller
		H'00002492 - H'00002DCF	H'0000093E	EV_Input	EV_Input
		H'00002DD0 - H'00002E21	H'00000052	EV_Display	EV_Display

```
                H'00002E22   -   H'00003689   H'00000868
                                 EV_OpenClose                EV_OpenClose
                H'0000368A   -   H'00003ED1   H'00000848
                                 EV_UpDown                   EV_UpDown
                H'00003ED2   -   H'000042C5   H'000003F4
                                 EV_File                     EV_File
                H'000042C6   -   H'0000441D   H'00000158
                                 EV_Time                     EV_Time
                H'0000441E   -   H'00004981   H'00000564
                                 Timer                       Timer
                H'00004982   -   H'00004AD7   H'00000156
                                 Panel                       Panel
                H'00004AD8   -   H'00004BA9   H'000000D2
                                 sci                         sci
                H'00004BAA   -   H'00004DE7   H'0000023E
                                 lcd                         lcd
                H'00004DE8   -   H'000058D9   H'00000AF2
                                 usb                         usb
                H'000058DA   -   H'00005913   H'0000003A
                                 rand                        rand
                H'00005914   -   H'00005971   H'0000005E
                                 sprintf                     sprintf
                H'00005972   -   H'0000599B   H'0000002A
                                 strcmp                      strcmp



                H8/300H LINKAGE EDITOR (Evaluation software) Ver.1.0   PAGE :    2

                   ***    LINKAGE EDITOR LINK MAP LIST     ***


SECTION   NAME                START     -     END       LENGTH
                                 UNIT NAME                   MODULE NAME


ATTRIBUTE :   CODE  NOSHR

P                  H'0000599C   -   H'000059B7   H'0000001C
                                 strcpy                      strcpy
                H'000059B8   -   H'000059D3   H'0000001C
                                 strlen                      strlen
                H'000059D4   -   H'00005A03   H'00000030
                                 vsprintf                    vsprintf
                H'00005A04   -   H'00005CDD   H'000002DA
                                 addd3                       addd3
                H'00005CDE   -   H'00005F0F   H'00000232
                                 divd3                       divd3
                H'00005F10   -   H'00005F89   H'0000007A
                                 dtol3                       dtol3
                H'00005F8A   -   H'00005F95   H'0000000C
                                 eqd3                        eqd3
                H'00005F96   -   H'00005FA5   H'00000010
                                 ged3                        ged3
                H'00005FA6   -   H'00005FB5   H'00000010
                                 ltd3                        ltd3
                H'00005FB6   -   H'00005FFB   H'00000046
```

```
                    ltod3                   ltod3
H'00005FFC  -  H'00006019  H'0000001E
                    mv83                    mv83
H'0000601A  -  H'00006041  H'00000028
                    mvn3                    mvn3
H'00006042  -  H'0000604F  H'0000000E
                    ned3                    ned3
H'00006050  -  H'00006071  H'00000022
                    spregld3                spregld3
H'00006072  -  H'00006099  H'00000028
                    spregsv3                spregsv3
H'0000609A  -  H'00007E47  H'00001DAE
                    _fmtout                 _fmtout
H'00007E48  -  H'00007F03  H'000000BC
                    cmpd3                   cmpd3
H'00007F04  -  H'00007F29  H'00000026
                    divl3                   divl3
H'00007F2A  -  H'00007F49  H'00000020
                    mull3                   mull3
H'00007F4A  -  H'0000832F  H'000003E6
                    _dti                    _dti
H'00008330  -  H'000084D3  H'000001A4
                    _its                    _its
H'000084D4  -  H'0000852D  H'0000005A
                    memcpy                  memcpy
H'0000852E  -  H'00008569  H'0000003C
                    divul3                  divul3


H8/300H LINKAGE EDITOR (Evaluation software) Ver.1.0   PAGE :    3

    ***    LINKAGE EDITOR LINK MAP LIST     ***


SECTION   NAME              START    -    END      LENGTH
                    UNIT NAME               MODULE NAME


ATTRIBUTE :  CODE  NOSHR

P                   H'0000856A  -  H'00008591  H'00000028
                    _allzero                _allzero
H'00008592  -  H'00008689  H'000000F8
                    _calcnpw                _calcnpw
H'0000868A  -  H'0000872D  H'000000A4
                    _log10                  _log10
H'0000872E  -  H'000087A5  H'00000078
                    _lsfts                  _lsfts
H'000087A6  -  H'000087D3  H'0000002E
                    _pow5                   _pow5
H'000087D4  -  H'0000884D  H'0000007A
                    _rsfts                  _rsfts
H'0000884E  -  H'000088F9  H'000000AC
                    _sub                    _sub
H'000088FA  -  H'0000899D  H'000000A4
                    _unpack                 _unpack
```

```
                H'0000899E   -   H'000089DB   H'0000003E
                                 memcmp                  memcmp
                H'000089DC   -   H'00008A63   H'00000088
                                 _mult64                 _mult64
                H'00008A64   -   H'00008BC5   H'00000162
                                 _power                  _power
                H'00008BC6   -   H'00008CAF   H'000000EA
                                 _rnd                    _rnd
                H'00008CB0   -   H'00008D4B   H'0000009C
                                 _setsbit                _setsbit
                H'00008D4C   -   H'00008E51   H'00000106
                                 frexp                   frexp
                H'00008E52   -   H'00008F87   H'00000136
                                 modf                    modf
                H'00008F88   -   H'00008FA9   H'00000022
                                 dslc3                   dslc3
                H'00008FAA   -   H'00008FCB   H'00000022
                                 dsruc3                  dsruc3
                H'00008FCC   -   H'00009001   H'00000036
                                 itod3                   itod3
                H'00009002   -   H'000092EF   H'000002EE
                                 muld3                   muld3
                H'000092F0   -   H'00009341   H'00000052
                                 _duchek                 _duchek
                H'00009342   -   H'00009393   H'00000052
                                 _lsft                   _lsft
                H'00009394   -   H'00009521   H'0000018E
                                 _mult                   _mult
                H'00009522   -   H'000095BD   H'0000009C
                                 _pow10                  _pow10


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

                   ***    LINKAGE EDITOR LINK MAP LIST     ***


SECTION   NAME              START    -    END      LENGTH
                                UNIT NAME                MODULE NAME


ATTRIBUTE :  CODE  NOSHR

P                 H'000095BE   -   H'00009605   H'00000048
                                 _add                    _add
                H'00009606   -   H'00009635   H'00000030
                                 memset                  memset

* TOTAL ADDRESS *          H'00000200   -   H'00009635   H'00009436


ATTRIBUTE :  DATA  NOSHR ROM

D                 H'000099C0   -   H'000099C0   H'00000000
                                 asmfile                 asmfile
                H'000099C0   -   H'000099CF   H'00000010
```

usb usb

* TOTAL ADDRESS * H'000099C0 - H'000099CF H'00000010

ATTRIBUTE : DATA NOSHR

C H'00009A00 - H'00009C8D H'0000028E
main main
H'00009C8E - H'00009CBC H'0000002F
EV_Simulator EV_Simulator
H'00009CBE - H'00009CC8 H'0000000B
EV_Puls EV_Puls
H'00009CCA - H'00009CF1 H'00000028
EV_Controller EV_Controller
H'00009CF2 - H'00009D57 H'00000066
EV_Input EV_Input
H'00009D58 - H'00009E17 H'000000C0
EV_Display EV_Display
H'00009E18 - H'00009E63 H'0000004C
EV_OpenClose EV_OpenClose
H'00009E64 - H'00009EA6 H'00000043
EV_UpDown EV_UpDown
H'00009EA8 - H'00009F19 H'00000072
EV_File EV_File
H'00009F1A - H'00009F21 H'00000008
EV_Time EV_Time
H'00009F22 - H'00009F59 H'00000038
Timer Timer
H'00009F5A - H'00009F64 H'0000000B
Panel Panel
H'00009F66 - H'0000A01F H'000000BA
usb usb

*** LINKAGE EDITOR LINK MAP LIST ***

SECTION NAME START - END LENGTH
UNIT NAME MODULE NAME

ATTRIBUTE : DATA NOSHR

C H'0000A020 - H'0000A027 H'00000008
_fmtout _fmtout
H'0000A028 - H'0000A127 H'00000100
_ctype _ctype
H'0000A128 - H'0000A1AF H'00000088
_its _its
H'0000A1B0 - H'0000A1B7 H'00000008
_log10 _log10
H'0000A1B8 - H'0000A297 H'000000E0
_pow5 _pow5

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

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


ATTRIBUTE :  DATA  NOSHR RAM

R                   H'00FFE000   -   H'00FFE000   H'00000000
                                    asmfile                 asmfile
                    H'00FFE000   -   H'00FFE00F   H'00000010
                                    usb                     usb

* TOTAL ADDRESS *           H'00FFE000   -   H'00FFE00F   H'00000010


ATTRIBUTE :  DATA  NOSHR

B                   H'00FFE010   -   H'00FFE011   H'00000002
                                    asmfile                 asmfile
                    H'00FFE012   -   H'00FFE21F   H'0000020E
                                    main                    main
                    H'00FFE220   -   H'00FFE233   H'00000014
                                    EV_File                 EV_File
                    H'00FFE234   -   H'00FFE475   H'00000242
                                    Timer                   Timer
                    H'00FFE476   -   H'00FFE4F5   H'00000080
                                    Panel                   Panel
                    H'00FFE4F6   -   H'00FFE545   H'00000050
                                    sci                     sci
                    H'00FFE546   -   H'00FFE585   H'00000040
                                    lcd                     lcd



                    H8/300H LINKAGE EDITOR (Evaluation software) Ver.1.0   PAGE :    6

                       ***    LINKAGE EDITOR LINK MAP LIST     ***


SECTION   NAME              START     -     END       LENGTH
                                UNIT NAME                   MODULE NAME


ATTRIBUTE :  DATA  NOSHR

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

_errno _errno

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

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

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

SYMBOL NAME	ADDR	TYPE
$ADDD$3	H'00005ACE	DAT
$CMPD$3	H'00007E48	DAT
$DIVD$3	H'00005D78	DAT
$DIVL$3	H'00007F04	DAT
$DIVUL$3	H'0000852E	DAT
$DSLC$3	H'00008F88	DAT
$DSRUC$3	H'00008FAA	DAT
$DTOL$3	H'00005F10	DAT
EQD3	H'00005F8A	DAT
GED3	H'00005F96	DAT
$ITOD$3	H'00008FCC	DAT
LTD3	H'00005FA6	DAT
$LTOD$3	H'00005FB6	DAT
$MULD$3	H'000090B8	DAT
$MULL$3	H'00007F2A	DAT
$MV8$3	H'00005FFC	DAT
MVN3	H'0000601A	DAT
NED3	H'00006042	DAT
$SUBD$3	H'00005A9E	DAT
sp_regld3	H'00006050	DAT
sp_regsv3	H'00006072	DAT
_Checkfmove	H'000043E0	ENT
_Clear	H'00004982	ENT
_ClearLCD	H'00004CC8	ENT
_Close	H'00003642	ENT
_CloseMotor	H'000030F6	ENT
_Cnt	H'00FFE012	DAT
_Command_Read	H'000040DC	ENT
_Command_Write	H'00004128	ENT
_Destroy	H'00001144	ENT
_DisableInterrupt	H'000002AC	DAT
_Disp	H'00002E20	ENT
_DispInput	H'00002DD0	ENT
_DispUSBPort	H'00004EF8	ENT
_Door	H'00002E22	ENT
_Down	H'00003E8A	ENT
_DownMotor	H'0000395E	ENT
_EV_AddressDataSet	H'000018EA	ENT
_EV_AddressSet	H'00001806	ENT
_EV_Controller	H'00001B16	ENT
_EV_DataSet	H'00001878	ENT
_EV_EnableSet	H'000017F0	ENT
_EV_File	H'00003F26	ENT
_EV_Input	H'000024FE	ENT
_EV_Puls	H'000019BA	ENT

_EV_Set	H'0000179A	ENT
_EV_Simulator	H'000012DA	ENT
_EV_Time	H'000042C6	ENT
_EnableInterrupt	H'000002A8	DAT
_GetChar	H'00002492	ENT
_GetCurrentTime	H'00004328	ENT

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

SYMBOL NAME	ADDR	TYPE
_GetPermit	H'000043C2	ENT
_GetSCI	H'00004B08	ENT
_GetSW	H'00001288	ENT
_H8init	H'000012AE	ENT
_Init	H'00000EB0	ENT
_InitITU	H'00004906	ENT
_InitLCD	H'00004C58	ENT
_InitSCI	H'00004AD8	ENT
_InitUSB	H'00004DFE	ENT
_InterruptITU0	H'0000493C	ENT
_LCDOut4	H'00004C00	ENT
_Limit_Read	H'00004282	ENT
_LocateLCD	H'00004D08	ENT
_Motor_Read	H'00004234	ENT
_Motor_Write	H'000041FC	ENT
_OnCloseMotor	H'00003112	ENT
_OnController	H'00001CB6	ENT
_OnDownMotor	H'0000397A	ENT
_OnInitWaitDoorChangeLog	H'0000337E	ENT
_OnInitWaitPositionChangeLog	H'00003BD8	ENT
_OnInput	H'0000262A	ENT
_OnOpenMotor	H'00002EB2	ENT
_OnPuls	H'000019F8	ENT
_OnSimulator	H'000013C4	ENT
_OnUpMotor	H'0000371A	ENT
_OnWaitCloseDoorChangeLog	H'00003526	ENT
_OnWaitDownPositionChangeLog	H'00003D74	ENT
_OnWaitOpenDoorChangeLog	H'00003452	ENT
_OnWaitUpPositionChangeLog	H'00003CA6	ENT
_Open	H'000035FA	ENT
_OpenMotor	H'00002E96	ENT
_PermitCommand_Read	H'0000404C	ENT
_PermitCommand_Write	H'00004098	ENT
_PermitTurnOpen_Read	H'0000416C	ENT
_PermitTurnOpen_Write	H'000041B8	ENT
_Position	H'0000368A	ENT
_PrintF	H'000049AA	ENT
_PrintLCD	H'00004D2C	ENT
_PrintSCI	H'00004B28	ENT
_PutLCD	H'00004CDC	ENT
_PutSCI	H'00004AF8	ENT
_Read	H'00003FBA	ENT

_ReadString	H'00004014	ENT
_Repaint	H'000004DC	ENT
_Run	H'00000548	ENT
_ScanSCI	H'00004B18	ENT
_SetCurrentTime	H'0000430A	ENT
_SetLED	H'0000123A	ENT
_SetPermit	H'0000438E	ENT
_SleepMSec	H'00004430	ENT
_Start	H'000046F2	ENT

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

SYMBOL NAME	ADDR	TYPE
_Stop	H'00004720	ENT
_Thread_Start	H'000047B8	ENT
_Thread_Toggle	H'000047F6	ENT
_Thread_checkAllDelete	H'0000473E	ENT
_Thread_checkStayAnother	H'00004758	ENT
_Thread_getThread	H'0000477E	ENT
_Up	H'00003E42	ENT
_UpMotor	H'000036FE	ENT
_WaitDoorChangeLog	H'00003356	ENT
_WaitPositionChangeLog	H'00003BBE	ENT
_WaitSecond	H'0000436C	ENT
_Wait_ms	H'00004402	ENT
_Write	H'00003F30	ENT
_WriteString	H'00003F82	ENT
__add	H'000095BE	ENT
__allzero	H'0000856A	ENT
__calcnpw	H'00008592	ENT
__ctype	H'0000A028	DAT
__dti	H'00007F4A	ENT
__duchek	H'000092F0	ENT
__errno	H'00FFE89A	DAT
__fmtout	H'0000609A	ENT
__its	H'00008330	ENT
__log10	H'0000868A	ENT
__lsft	H'00009342	ENT
__lsfts	H'0000872E	ENT
__mult	H'00009394	ENT
__mult64	H'000089DC	ENT
__pow10	H'00009522	ENT
__pow5	H'000087A6	ENT
__power	H'00008A64	ENT
__rnd	H'00008BC6	ENT
__rnext	H'00FFE896	DAT
__rsfts	H'000087D4	ENT
__setsbit	H'00008CB0	ENT
__sub	H'0000884E	ENT
__unpack	H'000088FA	ENT
_cntrl	H'00FFE08C	DAT
_countUpNextRun	H'00004506	ENT

delete	H'000046BE	ENT
_frexp	H'00008D4C	ENT
_getClock	H'0000441E	ENT
_get_inbufflen	H'000058B2	ENT
_get_outbufflen	H'000058C6	ENT
_i_cnt	H'00FFE016	DAT
_in	H'00FFE04E	DAT
_initWOVI	H'00004968	ENT
_init_usbbuff	H'00005716	ENT
_j_cnt	H'00FFE018	DAT
_main	H'000002B0	ENT
_memcmp	H'0000899E	ENT

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

SYMBOL NAME	ADDR	TYPE
_memcpy	H'000084D4	ENT
_memset	H'00009606	ENT
_modf	H'00008E52	ENT
_new_EV_Status	H'00003ED2	ENT
_new_Thread	H'00004526	ENT
_nextRun	H'000044C2	ENT
_puls	H'00FFE184	DAT
_rand	H'000058DA	ENT
_read_buff	H'00005854	ENT
_read_outbuff	H'000057F6	ENT
_s	H'00FFE04A	DAT
_simu	H'00FFE198	DAT
_sprintf	H'00005914	ENT
_status	H'00FFE220	DAT
_strcmp	H'00005972	ENT
_strcpy	H'0000599C	ENT
_strlen	H'000059B8	ENT
_th	H'00FFE01A	DAT
_th1	H'00FFE022	DAT
_th101	H'00FFE278	DAT
_th102	H'00FFE296	DAT
_th111	H'00FFE2B4	DAT
_th112	H'00FFE2D2	DAT
_th113	H'00FFE2F0	DAT
_th114	H'00FFE30E	DAT
_th119	H'00FFE32C	DAT
_th120	H'00FFE34A	DAT
_th121	H'00FFE368	DAT
_th122	H'00FFE386	DAT
_th123	H'00FFE3A4	DAT
_th130	H'00FFE3C2	DAT
_th131	H'00FFE3E0	DAT
_th141	H'00FFE3FE	DAT
_th142	H'00FFE41C	DAT
_th143	H'00FFE43A	DAT
_th144	H'00FFE458	DAT

_th19	H'00FFE032	DAT
_th20	H'00FFE036	DAT
_th41	H'00FFE03A	DAT
_th42	H'00FFE03E	DAT
_th43	H'00FFE042	DAT
_th44	H'00FFE046	DAT
_usb_int	H'00004F5C	ENT
_vsprintf	H'000059D4	ENT
_wovi	H'00004964	ENT
_woviClock	H'00FFE234	DAT
_woviInit	H'000048D4	ENT
_woviRun	H'0000483A	ENT
_woviThreadFirst	H'00FFE23C	DAT
_woviThreadLast	H'00FFE25A	DAT
_write_buff	H'00005790	ENT

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

SYMBOL NAME	ADDR	TYPE
_write_inbuff	H'00005734	ENT

MAKE Version 5.2 Copyright (c) 1987, 2000 Borland

	bcc32 -O2 -w -tWC -D"USE_BCC" -c Panel.c

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

Panel.c:

	bcc32 -O2 -w -tWC -D"USE_BCC" -c Timer.c

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

Timer.c:

警告 W8057 Timer.c 398: パラメータ 'threadPerSec' は一度も使用されない(関数 wovi)

	bcc32 -O2 -w -tWC -D"USE_BCC" -c EV_Time.c

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

EV_Time.c:

	bcc32 -O2 -w -tWC -D"USE_BCC" -c EV_File.c

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

EV_File.c:

警告 W8004 EV_File.c 74: 'Ret' に代入した値は使われていない(関数 Write)

警告 W8004 EV_File.c 108: 'Ret' に代入した値は使われていない(関数 WriteString)

警告 W8071 EV_File.c 166: 変換によって有効桁が失われる(関数 Read)

警告 W8004 EV_File.c 162: 'Ret' に代入した値は使われていない(関数 Read)

警告 W8004 EV_File.c 203: 'Ret' に代入した値は使われていない(関数 ReadString)

警告 W8004 EV_File.c 229: 'Ret' に代入した値は使われていない(関数 PermitCommand_Read)

警告 W8004 EV_File.c 247: 'Ret' に代入した値は使われていない(関数 PermitCommand_Write)

警告 W8004 EV_File.c 265: 'Ret' に代入した値は使われていない(関数 Command_Read)

警告 W8004 EV_File.c 283: 'Ret' に代入した値は使われていない(関数 Command_Write)

警告 W8004 EV_File.c 301: 'Ret' に代入した値は使われていない(関数 PermitTurnOpen_Read)

警告 W8004 EV_File.c 319: 'Ret' に代入した値は使われていない(関数 PermitTurnOpen_Write)

警告 W8066 EV_File.c 343: 実行されないコード(関数 Motor_Write)

警告 W8066 EV_File.c 364: 実行されないコード(関数 Motor_Read)

警告 W8066 EV_File.c 367: 実行されないコード(関数 Motor_Read)

警告 W8066 EV_File.c 382: 実行されないコード(関数 Limit_Read)

警告 W8066 EV_File.c 385: 実行されないコード(関数 Limit_Read)

bcc32 -O2 -w -tWC -D"USE_BCC" -c EV_UpDown.c

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

EV_UpDown.c:

bcc32 -O2 -w -tWC -D"USE_BCC" -c EV_OpenClose.c

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

EV_OpenClose.c:

bcc32 -O2 -w -tWC -D"USE_BCC" -c EV_Display.c

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

EV_Display.c:

bcc32 -O2 -w -tWC -D"USE_BCC" -c EV_Input.c

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

EV_Input.c:

bcc32 -O2 -w -tWC -D"USE_BCC" -c EV_Controller.c

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

EV_Controller.c:

bcc32 -O2 -w -tWC -D"USE_BCC" -c EV_Puls.c

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

EV_Puls.c:

警告 W8057 EV_Puls.c 53: パラメータ 'addressDataSet' は一度も使用されない(関数 EV_Set)

警告 W8057 EV_Puls.c 53: パラメータ 'dataSet' は一度も使用されない(関数 EV_Set)

警告 W8057 EV_Puls.c 53: パラメータ 'addressClockSet' は一度も使用されない(関数 EV_Set)

警告 W8057 EV_Puls.c 53: パラメータ 'clockSet' は一度も使用されない(関数 EV_Set)

警告 W8057 EV_Puls.c 182: パラメータ 'th' は一度も使用されない(関数 EV_Puls)

bcc32 -O2 -w -tWC -D"USE_BCC" -c EV_Simulator.c

Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland

EV_Simulator.c:

警告 W8019 EV_Simulator.c 68: コードは効果を持たない(関数 OnSimulator)

```
警告 W8019 EV_Simulator.c 86: コードは効果を持たない(関数 OnSimulator )
警告 W8019 EV_Simulator.c 104: コードは効果を持たない(関数 OnSimulator )
警告 W8019 EV_Simulator.c 122: コードは効果を持たない(関数 OnSimulator )
        bcc32 -O2 -w -tWC -D"USE_BCC" -c main.c
Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland
main.c:
警告 W8004 main.c 293: 'key' に代入した値は使われていない(関数 Run )
        ilink32 /Tpe -L"C:¥borland¥bcc55¥Lib"   Panel.obj Timer.obj EV_Time.obj EV_File.obj
EV_UpDown.obj EV_OpenClose.obj EV_Display.obj EV_Input.obj EV_Controller.obj EV_Puls.obj
EV_Simulator.obj main.obj c0x32.obj,main.exe,,cw32.lib import32.lib
Turbo Incremental Link 5.00 Copyright (c) 1997, 2000 Borland
MAKE Version 5.2  Copyright (c) 1987, 2000 Borland
        del *.obj
        del main.tds
        del main.ilc
        del main.ild
        del main.ilf
        del main.ils
H8/300H ASSEMBLER (Evaluation software) Ver.1.0
  *****TOTAL ERRORS      0
  *****TOTAL WARNINGS    0
H8/300H LINKAGE EDITOR (Evaluation software) Ver.1.0

: OUTPUT c_thread
: PRINT c_thread
: INPUT asmfile, main, EV_Simulator, EV_Puls, EV_Controller, EV_Input, EV_Display, EV_OpenClose,
EV_UpDown, EV_File, EV_Time, Timer, Panel, sci, lcd, usb
: LIB c:¥h8¥akic¥c38hab
: START R(0FFE000), P(200), D(99C0), C(9A00)
```

: ROM (D, R)

: EXIT

LINKAGE EDITOR COMPLETED

H8/300H OBJECT CONVERTER (Evaluation software) Ver.1.0

OBJECT CONVERTER COMPLETED

著作者:

しのみや　ひでみね

篠宮　　英峰