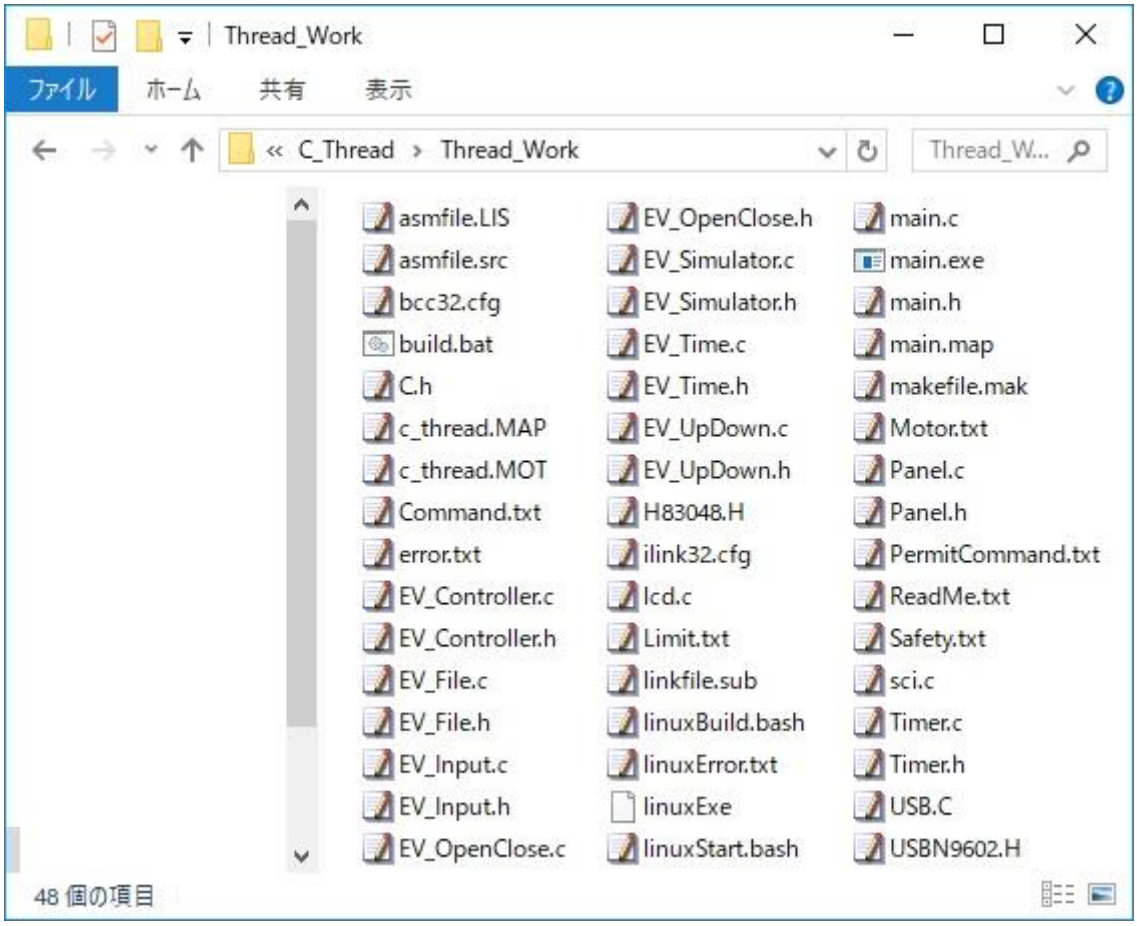
発明の巻

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。2017 年 12 月 13 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.

参照ライブラリ

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

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

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

struct st_itu0 {                        /* struct ITU0  */
      unsigned char     TCR;            /* TCR          */
      unsigned char     TIOR;           /* TIOR         */
      unsigned char     TIER;           /* TIER         */
      unsigned char     TSR;            /* TSR          */
      unsigned int      TCNT;           /* TCNT         */
      unsigned int      GRA;            /* GRA          */
      unsigned int      GRB;            /* GRB          */
};

struct st_itu3 {                        /* struct ITU3  */
      unsigned char     TCR;            /* TCR          */
      unsigned char     TIOR;           /* TIOR         */
      unsigned char     TIER;           /* TIER         */
      unsigned char     TSR;            /* TSR          */
      unsigned int      TCNT;           /* TCNT         */
      unsigned int      GRA;            /* GRA          */
      unsigned int      GRB;            /* GRB          */
      unsigned int      BRA;            /* BRA          */
      unsigned int      BRB;            /* BRB          */
      char              wk[2];          /*              */
};

struct st_tpc {                         /* struct TPC   */
      unsigned char     TPMR;           /* TPMR         */
      unsigned char     TPCR;           /* TPCR         */
```

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

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

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

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

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

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

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

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

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

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

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

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

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

struct st_ad {			/* struct A/D	*/
	unsigned int	DRA;		/* ADDRA		*/
	unsigned int	DRB;		/* ADDRB		*/
	unsigned int	DRC;		/* ADDRC		*/
	unsigned int	DRD;		/* ADDRD		*/
	unsigned char	CSR;		/* ADCSR		*/
	unsigned char	CR;		/* ADCR		*/
};

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

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

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

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

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


/* USB1.0リクエスト */
#define     USB_GET_STATUS          0
#define     USB_CLEAR_FEATURE       1
#define     USB_SET_FEATURE         3
#define     USB_SET_ADDRESS         5
#define     USB_GET_DESCRIPTOR      6
#define     USB_SET_DESCRIPTOR      7
#define     USB_GET_CONFIGURATION   8
#define     USB_SET_CONFIGURATION   9
#define     USB_GET_INTERFACE       10
#define     USB_SET_INTERFACE       11
#define     USB_SYNCH_FRAME         12

/* ディスクリプタ名 */
#define     USB_DEVICE              1
#define     USB_CONFIGURATION       2
```

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

/* HIDリクエスト */
#define     USB_GET_REPORT              0x01
#define     USB_GET_IDLE                0x02
#define     USB_GET_PROTOCOL            0x03
#define     USB_SET_REPORT              0x09
#define     USB_SET_IDLE                0x0A
#define     USB_SET_PROTOCOL            0x0B


/*============================================================
                        N9604 Register
============================================================*/
#define     USB_MCNTRL              0x00    /*Main control register   */
#define     USB_CCONF               0x01    /*Clk. config. register   */
#define     USB_TCR                 0x02    /*Xcvr config. register   */
#define     USB_RID                 0x03    /*Rev. ID     register   */
#define     USB_FAR                 0x04    /*Func address register   */
#define     USB_NFSR                0x05    /*Node func st register   */
#define     USB_MAEV                0x06    /*Main event   register   */
#define     USB_MAMSK               0x07    /*Main mask    register   */
#define     USB_ALTEV               0x08    /*Alt. event   register   */
#define     USB_ALTMSK              0x09    /*ALT  mask    register   */
```

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

```
/*

      USB N9604 コントロール

                                                    (C)2002 C.I.M

*/

#include      <stdio.h>
#include      <string.h>
#include      "usbn9602.h"
#include      "h83048.h"

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

static void RegisterSet();
static void ResetUSB();
static void WakeupUSB();

static void rx0();
static void rx1();
static void tx0();
static void tx1();
static void nako0();
static void nako1();
static void naki0();
static void naki1();

static void clrfeature();
```

```
static void setfeature();
static void getdescriptor();
static void send_desc_sub(void *ptr,int size);
static void send_desc();
static void getstatus();
static void setconfiguration();

static void SetStallUSB(int adr);
static void ClearStallUSB(int adr);
static void FlushRXC(int no);
static void FlushTXC(int no);
static void TxTogle(int no);

static void WriteUSB(int adr,int data);
static unsigned char ReadUSB(int adr);
static int ReadUSBBurst(int adr,int adr2,char *buff,int cnt);
static int WriteUSBBurst(int adr,int adr2,char *buff,int cnt);

/*---------------------------------*/
static int SendTX1();

/*---------------------------------*/
int get_inbufflen(void);
void init_usbbuff(void);
int write_inbuff(char *p,int size);
int get_outbufflen(void);
int write_buff(char *p,int size);
int read_outbuff(char *p,int size);
/*---------------------------------*/
```

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


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

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

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


/* コンフィグレーションディスクリプタ */
static const unsigned char cfg_desc[] = {
    0x09,                   /* length of this desc.                         */
    0x02,                   /* コンフィグレーション・ディスクリプタ         */
    9+9+7*3,                /* インターフェース／エンドポイントディスクリプタ等の合計長 CFG + IF +
EP*3 */
    0x00,                   /*                                              */
    0x01,                   /* インターフェース数 1                         */
    0x01,                   /* コンフィグレーションは 1                     */
    0x00,                   /* index of config. string                      */
    0xc0,                   /* attr.: self powered  D6=自己電源             */
    100,                    /*     ;max power (100 mA)                      */
/*},*/
/*static const unsigned char if_desc[] = {*/
    0x09,                   /* length of this desc.                         */
    0x04,                   /* INTERFACE descriptor                         */
    0x00,                   /* interface number                             */
    0x00,                   /* alternate setting                            */
    0x03,                   /* # of (non 0) endpoints                       */
    0x00,                   /* interface class                              */
```

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

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

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

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



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

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

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

```
            {
            }
        }
    }

    /*-----------------------------------------------------------*/
    /* USB初期化 */
    void InitUSB()
    {
        init_usbbuff();
        ResetUSB();
        RegisterSet();
        WakeupUSB();
    /*     PrintSCI(" REV = %d¥n",ReadUSB(USB_RID)); */
    /*     PrintSCI(" CLOCK = %02X¥n",ReadUSB(USB_CCONF)); */

    }

    static void RegisterSet()
    {
        STALLD = 0;
        senddesc = 0;
        DATA0_1 = 0;
        SETADDR = 0;

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

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

static void ResetUSB()
{
    WriteUSB(USB_MCNTRL,USB_SRST+USB_VGE);                /* USBリセット 3.3V供給 */
    wait(100);                                          /* 100msec */
    WriteUSB(USB_MCNTRL,USB_INT_L_P+USB_VGE);             /* 割込みはactive low push pull */
    WriteUSB(USB_CCONF,USB_CLKDIV-1);                     /* 48MHz/4 = 12MHz */
}

static void WakeupUSB()
{
    WriteUSB(USB_NFSR,USB_OPR_ST);                        /* 動作可にする */
    WriteUSB(USB_MCNTRL,USB_INT_L_P+USB_NAT+USB_VGE);     /* USBのﾉｰﾄﾞを動作可にする */
}

/* USBポートデータ表示 */
```

```
/* ※リードすると、ステータスが変わるレジスタもあるので注意 */
void DispUSBPort()
{
        int         i,j;
        PrintSCI("00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F¥n");
        for(j=0;j<4;j++)
        {
            for(i=0;i<16;i++)
                PrintSCI("%02X ",ReadUSB(i+j*16));
            PrintSCI("¥n");
        }
}

/*-----------------------------------------------------------*/
/* USB割り込み */
#ifdef  __GNUC__
void usb_int()  __attribute__ ((interrupt_handler));
#endif
void usb_int()
{
    unsigned char   nakevent,rxevent,txevent,altevent;
    char    reg;


    usbevent = ReadUSB(USB_MAEV);

    if( usbevent & USB_NAK )
    {
```

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

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

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

/*================================================================
                        RXイベントの処理
================================================================*/
/* RX0(system) */
/*
 リクエストコードの取得
        0 byte
            D7 ... データ方向 0=ホスト->デバイス , 1=デバイス->ホスト
            D6-D5 ... タイプ
                    0:標準 , 1:クラス , 2:ベンダ , 3:予約
            D4-D0 ... 受信側
                    0:デバイス , 1:インターフェイス , 2:エンドポイント , 3:その他
        1 byte
            特定のリクエスト
        2 byte
            value
```

```
        2 byte
            index
        2 byte
            length
*/
static void rx0()
{
    unsigned char   rxstat;
    rxstat = ReadUSB(USB_RXS0);
    if( rxstat & USB_SETUP_RX )
    {
        ReadUSBBurst(USB_RXD0,USB_RXS0,(char*)usbbuff,8);
        FlushRXC(0);
        FlushTXC(0);
        ClearStallUSB(USB_EPC0);
        if( (usbbuff[0] & 0x60) == 0 )
        {
            /* 標準リクエスト */
            switch( usbbuff[1] )
            {
                case    USB_CLEAR_FEATURE :
                    clrfeature();
                    break;
                case    USB_GET_CONFIGURATION :
                    WriteUSB(USB_TXD0,configno);
                    break;
                case    USB_GET_DESCRIPTOR :
                    getdescriptor();
```

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

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

```
/*-----------------------------------------------------------*/
/* RX1 受信 */
static void rx1()
{
    int             cnt;
    unsigned char   rxstat;
    rxstat = ReadUSB(USB_RXS1);             /* RX1ステータス取得 */
    /* SETUP,ERRORパケットでない */
    if( !(rxstat & (USB_SETUP_RX¦USB_RX_ERR)) )
    {
        /* ホストからデータの受信 */
        /* FIFOからデータ取得 */
        cnt = ReadUSBBurst(USB_RXD1,USB_RXS1,(char*)rx1buff,64);
        /* リングバッファに書き込み */
        write_inbuff((char*)rx1buff,cnt);
    }
    FlushRXC(1);                            /* バッファをフラッシュ */
    WriteUSB(USB_RXC1,USB_RX_EN);       /* 受信可に設定                 */
}

/*-----------------------------------------------------------*/
/* RX2 受信(not use) */
static void rx2()
{
    unsigned char   rxstat;
    rxstat = ReadUSB(USB_RXS2);
}

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

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

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

```
/* TX1送信終了 */
static void tx1()
{
    unsigned char   txstat;
    txstat = ReadUSB(USB_TXS1);
    if( (txstat & USB_ACK_STAT) && (txstat & USB_TX_DONE) )
    {
        /*
            送信終了後に次の送信データを送信するようにする
            送信データが無い場合、HOSTには0バイトで送る
        */
        SendTX1();
    }
    else
    {
        /*
            送ったサイズより小さい読み込みが行われた場合、こちらにくる場合がある
        */
    }
}

/*==========================================================================
                                NAKイベントの処理
==========================================================================*/
/*
    NAKイベントは、エラーが発生した場合など再送信する場合処理する
    NAK0しかENABLEにしていないので、それ以外は処理無し
*/
static void nako0()
```

```
{
}
static void nako1()
{
}
static void naki0()
{
}

static void naki1()
{
}



/*=============================================================
                        標準リクエストの処理
=============================================================*/
/* 選択機器 */
static void clrfeature()
{
    if( (usbbuff[0] & 3) == 2 )
    {
        /* エンドポイントデータ */
        if((usbbuff[3]&7) != 0 )
            ClearStallUSB(epctbl[usbbuff[3]&7]);
        STALLD &= -1 ^ (1<<(usbbuff[3]&7));
    }
}
static void setfeature()
```

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

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

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

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

```
    send_desc();
}

static void send_desc()
{
    int         sz;
    sz = 8;
    if( desc_size == 0 )    return;
    if( desc_size <= 8 )    sz = desc_size;
    sz = WriteUSBBurst(USB_TXD0,USB_TXS0,desc_ptr,sz);
    desc_size -= sz;
    desc_ptr += sz;
    if( desc_size == 0 )    senddesc = 0;
}
/*-------------------------------------------------------------*/
/* ステータス */
static void getstatus()
{
    int         data,ep;
    data = usbbuff[0]&3;
    if( (data == 0) || (data == 1) )       /* DEVICE,INTERFACE */
    {
        WriteUSB(USB_TXD0,0);
        WriteUSB(USB_TXD0,0);
    }
    else if( data== 2)                          /* エンドポイント */
    {
        ep = usbbuff[3]&7;
        /* epのSTALL状態を送信 */
```

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

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

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


        FlushRXC(1);
        /* EPC2をｱﾄﾞﾚｽ2としてｲﾈｰﾌﾞﾙ */
        WriteUSB(USB_EPC2,USB_EP_EN+02);
        /* RX1受信可 */
        WriteUSB(USB_RXC1,USB_RX_EN);
        /*
            USB_TX_LASTを立てると、READ時ストールしなくなる。
            ただし、データを送る前の最初のREADは0byteになります。
            host側は複数のリクエストを同時発行できないので、
            リクエストがストールするのはまずいの回避。
        */
    }
}

/*============================================================
                        汎用ルーチン
============================================================*/
/* STALLのセットとクリア */
static void SetStallUSB(int adr)
{
    WriteUSB(adr,ReadUSB(adr) ¦ 0x80);
}

static void ClearStallUSB(int adr)
{
    WriteUSB(adr,ReadUSB(adr)&0x7f);
}
```

```
/* FIFOのフラッシュ */
static void FlushRXC(int no)
{
	WriteUSB(rxcreg[no],USB_FLUSH);
}

static void FlushTXC(int no)
{
	int		d;
	d = ReadUSB(txcreg[no]);
	d ¦= USB_FLUSH;
	WriteUSB(txcreg[no],d);
}

/* 送信終了フラグセット */
/* reg = USB_TXC0～6	*/
static void TxTogle(int no)
{
	unsigned char	d;
	d = USB_TX_EN;
	if( DATA0_1 & (1<<no) ) d ¦= USB_TX_TOGL;
	else			d &= ˜USB_TX_TOGL;
	d ¦= USB_TX_LAST;
	WriteUSB(txcreg[no],d);
	DATA0_1 ^= (1<<no);
}
```

```
/*-----------------------------------------------------------*/
/* USBのアドレスから読み込み */
static unsigned char ReadUSB(int adr)
{
	USB9602R = (unsigned char)adr;
	return( USB9602D );
}
/* USBのアドレスへ書き込み */
static void WriteUSB(int adr,int data)
{
	USB9602R = (unsigned char)adr;
	USB9602D = (unsigned char)data;
}


/* バーストト転送 */
static int ReadUSBBurst(int adr,int adr2,char *buff,int cnt)
{
	int		i;
	int		rcnt;
	USB9602R = (unsigned char)adr;
	for(rcnt=0,i=0;i<cnt;i++)
	{
		if( (ReadUSB(adr2) & 0xf) == 0 )	break;

		USB9602R = (unsigned char)adr;
		*buff = USB9602D;
		buff++;
		rcnt++;
```

```
	}
	return(rcnt);
}

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

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

/*==========================================================================
				サンプルプログラム
==========================================================================*/
/* TX1送信ルーチン */
/*
	USBから一方的に送信できないため、今回は、
	TX1送信終了時にバッファ(outbuff)にあるデータを送信します。
	よって、HOSTからは定期的にREADを行う。
	それ以外のタイミングではEPC1のFIFOバッファのサイズに注意する。
*/
```

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

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

/*-------------------------------------------------------------*/
/*
 送受信バッファ
	inbuffがHOSTから送られてきたデータのバッファ
	outbuffがHOSTへ送るデータ用
	このサンプルではリングバッファを超えた分は捨てられます。
	今回は、256バイト確保しています。バッファがあふれないように
	メイン側で処理してください。
*/
#define	USBBUFFLEN	256		/* バッファのサイズ */
static int	inpos,inlen;		/* 入力バッファ位置、サイズ */
static int	outpos,outlen;		/* 出力バッファ位置、サイズ */
```

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

/*-------------------------------------------------------------*/
/*                              バッファの初期化                                */
/*-------------------------------------------------------------*/
void init_usbbuff()
{
      inpos = inlen = 0;
      outpos = outlen = 0;
}

/*-------------------------------------------------------------*/
/*                    HOSTからの受信バッファへ書き込み                    */
/*     char        *p          バッファポインタ                              */
/*     int              size   書き込みサイズ                                */
/*     戻り値                   書き込んだサイズ                              */
/*-------------------------------------------------------------*/
int write_inbuff(char *p,int size)
{
      int          i;
      INTC.IER &= (-1^0x20);                  /* IRQ5 Disable */
      for(i=0;i<size;i++)
      {
      if( inlen >= USBBUFFLEN )      break;
      inbuff[inpos] = *p;
      inpos = (inpos + 1)%USBBUFFLEN;
      inlen++;
```

```
        p++;
    }
        INTC.IER |= 0x20;                       /* IRQ5 Enable */
        return(i);
}
/*-------------------------------------------------------------*/
/*                      送信バッファへ書き込み                 */
/*    char      *p        バッファポインタ                     */
/*    int           size   書き込みサイズ                      */
/*    戻り値              書き込んだサイズ                     */
/*-------------------------------------------------------------*/
int write_buff(char *p,int size)
{
        int         i;
        INTC.IER &= (-1^0x20);                  /* IRQ5 Disable */
        for(i=0;i<size;i++)
        {
        if( outlen >= USBBUFFLEN )    break;
        outbuff[outpos%USBBUFFLEN] = *p;
        outpos = (outpos + 1)%USBBUFFLEN;
        outlen++;
        p++;
    }
        INTC.IER |= 0x20;                       /* IRQ5 Enable */
        return(i);
}

/*-------------------------------------------------------------*/
/*                      送信バッファから読み込み               */
```

```
/*     char        *p          バッファポインタ                          */
/*     int             size    バッファ最大サイズ                        */
/*     戻り値                  読み込んだサイズ                          */
/*--------------------------------------------------------------*/
int read_outbuff(char *p,int size)
{
    int    i;
    INTC.IER &= (-1^0x20);                  /* IRQ5 Disable */
    for(i=0;outlen>0;i++)
    {
        if( i >= size ) break;
        p[i] = outbuff[ (USBBUFFLEN+outpos-outlen)%USBBUFFLEN ];
        outlen--;
    }
    INTC.IER |= 0x20;                       /* IRQ5 Enable */
    return(i);
}
/*--------------------------------------------------------------*/
/*                      受信バッファから読み込み                */
/*     char        *p          バッファポインタ                          */
/*     int             size    バッファ最大サイズ                        */
/*     戻り値                  読み込んだサイズ                          */
/*--------------------------------------------------------------*/
int read_buff(char *p,int size)
{
    int    i;
    INTC.IER &= (-1^0x20);                  /* IRQ5 Disable */
    for(i=0;inlen>0;i++)
    {
```

```
        if( i >= size ) break;
        p[i] = inbuff[ (USBBUFFLEN+inpos-inlen)%USBBUFFLEN ];
        inlen--;
    }
    INTC.IER |= 0x20;                   /* IRQ5 Enable */
    return(i);
}
/*-------------------------------------------------------------*/
/*                      受信バッファのサイズ取得                    */
/*-------------------------------------------------------------*/
int get_inbufflen()
{
    return( inlen%USBBUFFLEN );
}

/*-------------------------------------------------------------*/
/*                      送信バッファのサイズ取得                    */
/*-------------------------------------------------------------*/
int get_outbufflen()
{
    return( outlen%USBBUFFLEN );
}
```

```
/*

                    SCI処理

                                                    (C)2002 C.I.M

*/

#include    <stdio.h>
#include    <stdarg.h>
#include    "h83048.h"


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


/*=================================================================
                                SCI初期化
-----------------------------------------------------------------
    9600bps ﾊﾟﾘﾃｨ無し STOP1
=================================================================*/
void InitSCI()
{
    int         i;
    SCI1.SCR = 0;
    SCI1.SMR = 0;                   /* ﾊﾟﾘﾃｨ無し STOP1                  */
    SCI1.BRR = 80;                  /* 9600bps     3052                */
    for(i=0;i<280;i++)    {}        /* wait                            */
    SCI1.SCR = 0x30;                /* TE = 1 , RE = 1                 */
    i = SCI1.SSR;
    SCI1.SSR = 0x80;                /* Clear Error Flag (TDRE=1)    */
```

```
}

/*================================================================
                        SCI出力
----------------------------------------------------------------
================================================================*/
void PutSCI(char c)
{
    unsigned char   i;
    while( 1 )
    {
        i = SCI1.SSR;
        if( i & 0x80 )      break;
    }
    SCI1.TDR = c;
    SCI1.SSR = i&0x7f;
}

/*================================================================
                        SCI入力
----------------------------------------------------------------
    データを受信するまで待ちつづけます。
================================================================*/
char GetSCI()
{
    unsigned char   i;
    char            c;
    while( 1 )
    {
```

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

/*=============================================================
                        SCI入力データチェック
-------------------------------------------------------------
    SCIにデータがあるかチェックします。
    戻り値          1 = データあり、0 = データなし
=============================================================*/
int ScanSCI()
{
    if( SCI1.SSR & 0x40 )   return(1);
    return(0);
}

/*=============================================================
                        SCI文字列出力
-------------------------------------------------------------
書式はprintf()と同じです。バッファは80文字分しか取っていないので、
必要ならば、増やしてください。
=============================================================*/
void PrintSCI(char *fmt, ...)
{
    int         i;
```

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

    for(i=0;;i++)
    {
        if( buff[i] == 0 )      break;
        /* 改行コードは2バイトにして送信 */
        if( buff[i] == '¥n' )   PutSCI('¥r');
        PutSCI(buff[i]);
    }
}
```

```
/*

        LCD処理

                                                (C)2002 C.I.M
*/



#include      <stdio.h>
#include      <stdarg.h>
#include      "h83048.h"

void ClearLCD();

/* PortB (write) b0..3 = LCD(LED) out , bit4 = LCD RS , bit7 = LCD E */
#define       LCD_RS              0x10
#define       LCD_E               0x80
#define       LCDMASK             0x60

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

```
/*====================================================================
                                LCD BYTE 出力
====================================================================*/
/*
    今回の基板は4bit接続なので、下位4bitのみ出力
*/
static void LCDOut8(short rs,short code)
{
    int         stat;
    int         pb;

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

    if( rs )    stat = (pb & LCDMASK) ¦ LCD_RS;
    else        stat = (pb & LCDMASK);
    PB.DR = code ¦ stat ¦ LCD_E;
    PB.DR = code ¦ stat;
    PB.DR = pb;                         /* 元のPBDに復帰 */
    wait(4);
}

/*====================================================================
                                LCD BYTE 出力(4bit)
====================================================================*/
void LCDOut4(int rs,int code)
{
    int         stat;
    int         pb;
    char    lb,hb;
```

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

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

    hb = ((code>>4)&0xf) ¦ stat;
    lb = (code&0xf) ¦ stat;

    PB.DR = hb ¦ LCD_E;
    PB.DR = hb;

    PB.DR = lb ¦ LCD_E;
    PB.DR = lb;

    PB.DR = pb;                         /* 元のPBDに復帰 */
    wait(4);
}

/*================================================================
                        LCD コントロール
----------------------------------------------------------------
    LCD初期化、表示、クリア
================================================================*/

void InitLCD()
{
    int         i;
    wait(15);
```

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


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

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

}


/*================================================================
                                LCD クリア
----------------------------------------------------------------
    LCD初期化、表示、クリア
================================================================*/
void ClearLCD()
{
    LCDOut4(0,0x01);      /* クリア */
    LCDOut4(0,0x02);      /* カーソルホーム */
}
```

```
/*================================================================
                        LCDキャラクタ表示
----------------------------------------------------------------
    '¥n','¥r','¥f'はLCDクリア処理を行います。
================================================================*/
void PutLCD(char c)
{
    if( c == '¥f' ) ClearLCD();
    else if( c == '¥n' )   ClearLCD();
    else if( c == '¥r' )   ClearLCD();
    else LCDOut4(1,c);
}

/*================================================================
                        LCDカーソル移動
----------------------------------------------------------------
    x = 0～15
    y = 0,1
================================================================*/
void LocateLCD(int x,int y)
{
    LCDOut4(0,0x80 + y*0x40 + x);
}

/*================================================================
                        LCD文字列表示
----------------------------------------------------------------
    パラメータはprintf()と同じです。64文字を超えないように設定してく
```

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

本文

```
/* C.h */

#define USE_THREAD
/* #define USE_BCC */
/* #define USE_LINUX */
#ifdef USE_LINUX
#define USE_BCC
#endif
#include <stdio.h> /* printf() */
#include <string.h> /* strcmp(), strlen() */
#include <stdlib.h> /* calloc(), free(), rand() */
#ifndef USE_BCC
#include "h83048.h"
#define SLEEP_PER_SEC 16000.0
#define NOTUSE_FILES
#else
#include <time.h> /* time_t, tm, time(), clock(), CLOCKS_PER_SEC */
#ifdef USE_LINUX
#include <termios.h> /* kbhit() */
#include <unistd.h> /* kbhit() */
#include <fcntl.h> /* kbhit() */
#else
#include <conio.h> /* kbhit(), getche() */
#endif
#define SLEEP_PER_SEC 100000000.0
#endif
#ifdef USE_LINUX
#define CLEAR system("clear")
#else
#define CLEAR system("cls")
#endif
#define OK 0
#define NG 1
#define ONE_MORE_TIME 2
#define ON 1
#define OFF 0
```

```
/* Panel.h */

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

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

```
/* Panel.c */

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

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

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

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

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

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

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

    ch = getchar();
```

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

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

    return 0;
}
#endif
```

```
/* Timer.h */

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

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

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

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

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

```
/* Timer.c */

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

/* 時間を表す外部変数宣言 */
double woviClock;

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

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

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

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

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

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

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

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

```
    Thread *List;
    Thread *new_List;
    List = &woviThreadFirst;
    while(List->next->next != NULL)
    {
        List = List->next;
    }
#ifndef USE_BCC
    if(id == 1)
    {
        new_List = &th101;
    }
    else if(id == 2)
    {
        new_List = &th102;
    }
    else if(id == 11)
    {
        new_List = &th111;
    }
    else if(id == 12)
    {
        new_List = &th112;
    }
    else if(id == 13)
    {
        new_List = &th113;
    }
    else if(id == 14)
```

```
{
	new_List = &th114;
}
else if(id == 19)
{
	new_List = &th119;
}
else if(id == 20)
{
	new_List = &th120;
}
else if(id == 21)
{
	new_List = &th121;
}
else if(id == 22)
{
	new_List = &th122;
}
else if(id == 23)
{
	new_List = &th123;
}
else if(id == 30)
{
	new_List = &th130;
}
else if(id == 31)
```

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

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

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

/* スレッドのvoid start(void)の代用 */
void Start(Thread *This)
{
	woviClock = getClock();
	This->preClock = woviClock;
	return;
}
```

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

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

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

```
}

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

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

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

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

void Thread_Toggle(int id)
{
	Thread *th;

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

```
            Start(th);
        }
        else
        {
            delete_(th);
        }
        return;
    }

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

```
            List = next_List;
        }
        return;
}

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

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

/*==============================================================
 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;
}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_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_Write(struct EV_File *This, char PermitTurnOpen);
int TurnOpen_Read(struct EV_File *This, char *p_chTurnOpen);
int TurnOpen_Write(struct EV_File *This, char TurnOpen);
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';
}
#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':
		status.permitcommand = ch;
		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(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':
		*p_ch = status.permitcommand;
		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_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_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;
}

int TurnOpen_Read(struct EV_File *This, char *p_chTurnOpen)
{
	int Ret = OK;
	switch(Read(This, "TurnOpen.txt¥0", p_chTurnOpen)){
	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 TurnOpen_Write(struct EV_File *This, char TurnOpen)
{
        int Ret = OK;
        switch(Write(This, "TurnOpen.txt¥0", TurnOpen)){
        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){
		/* 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){
		/* 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)
	{
		/* 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)
    {
        /* 閉 */
        OnCloseMotor(&CLMT, *DR, p_Safety);

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

    /* 戻る */
    return;
}
```

```
/* EV_Input.h */

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

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

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

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

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

#ifndef EV_OpenClose_h
```

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

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

/*==========================================================================
  入力を表す構造体
==========================================================================*/
struct EV_Input
{
    /* 安全 */
    char Safety;
    char *p_Safety;
    char Command;
    char PermitCommand;
    char 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 PF;
    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';
				break;
			case 1:
				Ret = 'd';
```

```
                break;
            case 2:
                Ret = 'o';
                break;
            case 3:
                Ret = 'c';
                break;
            default:
                break;
            }
        }
    }
#elif USE_LINUX
    if(kbhit())
    {
        Ret = (char) getchar();
    }
#elif USE_MSVS2005
    if(_kbhit())
    {
        Ret = (char) _getche();
    }
#else
    if(kbhit())
    {
        Ret = (char) getche();
    }
#endif
    return Ret;
```

```
}

/*==============================================================
  入力を表すコンストラクタとメソッド
==============================================================*/
void EV_Input(struct EV_Input *This)
{
	/* 初期化 */
	This->Command = '¥0';
	This->PermitCommand ='¥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;

	EV_File(&This->SF);
	EV_File(&This->CF);
	EV_File(&This->PF);
	EV_File(&This->MF);

	/* 安全入力 */
	if(Read(&This->SF, "Safety.txt¥0", This->p_Safety) == 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)
{
	This->Command = GetChar(This->Command);

	/* 安全入力 */
	if(Read(&This->SF, "Safety.txt¥0", This->p_Safety) == 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->PF, 'c');
		break;
	case 'r':
		/* 命令入力 */
		Write(&This->SF, "Safety.txt", 'h');
		This->Command = 'N';
		Command_Write(&This->CF, 'N');
		break;
	case 'o':
	case 'c':
	case 'u':
```

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

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

```
            PrintF(ClsPnl, "¥nA Basket isn't 2nd Floor");
        }
        else{
            if(This->Safety == 'h'){
                /* 命令入力 */
                This->Safety = 'Y';
                Write(&This->SF, "Safety.txt", 'Y');
            }
            /* 命令入力 */
            Command_Write(&This->CF, This->Command);
            PermitCommand_Write(&This->CF, 'N');
        }
        break;
    case 'q':
        /* 命令入力 */
        Command_Write(&This->CF, This->Command);
        delete_(th);
        break;
    default:
        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;

	/* 命令 */
	char Command;
	char *p_Command;

	/* ファイルストリーム */
	struct EV_File SF;
	struct EV_File CF;
	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->CF);
    EV_File(&This->MF);

    /* 安全初期化 */
    This->p_Safety = &This->Safety;
```

```
	/* 命令初期化 */
	This->p_Command = &This->Command;

	/* 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);

	/* 命令初期化 */
	if(Command_Write(&This->CF, 'N') == NG) return;

	/* ファイル初期化 */
	if(PermitCommand_Write(&This->CF, 'c') == 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;
```

```
/* 命令入力 */
if(Command_Read(&This->CF, This->p_Command) == 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':
		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_OpennerStop == ON)){
			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){
				Motor_Write(&This->MF, 's');
				Command_Write(&This->CF, 'N');
				PermitCommand_Write(&This->CF, 'c');
```

```
            }
            else{
                /* 開 */
                Open(This->p_DR,This->OPMT,This->WDCL, This->p_Safety);
            }
        }
        else if(*This->p_DR->p_CloserStop == ON){
            /* 閉完了時 */
            /* 上昇 */
            Up(This->p_P,This->UPMT,This->WPCL, This->p_Safety);
        }
        else if(*This->p_DR->p_CloserStop == OFF){
            /* 閉 */
            Close(This->p_DR,This->CLMT,This->WDCL, This->p_Safety);
        }
        break;
    /* 下階呼命令ならば */
    case   'y':
        if((*This->p_P->p_UpperStop == ON) && (*This->p_DR->p_OpennerStop == ON)){
            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){
                Motor_Write(&This->MF, 's');
                Command_Write(&This->CF, 'N');
                PermitCommand_Write(&This->CF, 'c');
```

```
            }
            else{
                /* 開 */
                Open(This->p_DR,This->OPMT,This->WDCL, This->p_Safety);
            }
        }
        else if(*This->p_DR->p_CloserStop == ON){
            /* 閉完了時 */
            /* 下降 */
            Down(This->p_P,This->DNMT,This->WPCL, This->p_Safety);
        }
        else if(*This->p_DR->p_CloserStop == OFF){
            /* 閉 */
            Close(This->p_DR,This->CLMT,This->WDCL, This->p_Safety);
        }
        break;
    /* 開命令ならば */
    case   'o':
        /* 開完了時 */
        if(*This->p_DR->p_OpennerStop == ON){
            Motor_Write(&This->MF, 's');
            Command_Write(&This->CF, 'N');
            PermitCommand_Write(&This->CF, 'c');
        }
        else{
            /* 開 */
            Open(This->p_DR,This->OPMT,This->WDCL, This->p_Safety);
        }
```

```
        break;
    /* 閉命令ならば */
    case   'c':
        /* 閉完了時 */
        if(*This->p_DR->p_CloserStop == ON){
            Motor_Write(&This->MF, 's');
            Command_Write(&This->CF, 'N');
            PermitCommand_Write(&This->CF, 'c');
        }
        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){
            /* 閉完了時 */
            if(*This->p_DR->p_CloserStop == ON){
                Motor_Write(&This->MF, 's');
                Command_Write(&This->CF, 'N');
                PermitCommand_Write(&This->CF, 'c');
            }
            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){
                /* 閉完了時 */
                if(*This->p_DR->p_CloserStop == ON){
                    Motor_Write(&This->MF, 's');
                    Command_Write(&This->CF, 'N');
                    PermitCommand_Write(&This->CF, 'c');
                }
                else{
                    /* 閉 */
                    Close(This->p_DR,This->CLMT,This->WDCL, This->p_Safety);
                }
            }

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

#ifndef EV_OpenClose_h
```

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

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

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

	/* 時間管理 */
	struct EV_Time T;

	/* ファイルストリーム */
	struct EV_File SF;
```

```
    struct EV_File CF;
    struct EV_File MF;
    struct EV_File LF;
};

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

```
/* EV_Simulator.c */

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

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

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

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

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

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

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

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

	/* リミットスイッチの新状態作成 */
```

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

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

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

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

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

    return;
}

void DispInput(void)
{
    /* 入力指示 */
    PrintF(InputCommand, "¥nUP = 'u', DOWN = 'd', OPEN = 'o', CLOSE = 'c'");
```

```
    PrintF(InputCommand, "¥nEMERGENCY = 's', RECOVERY = 'r'");
    PrintF(InputCommand, "¥n1st Floor CALL = 'y', 2nd Floor CALL = 'Y'");
    PrintF(InputCommand, "¥n1st Floor CLOSE = 'h', 2nd Floor CLOSE = 'H'");
    PrintF(InputCommand, "¥nQUITE = 'q'");
    PrintF(InputCommand, "¥nCOMMAND>");
}

/*
 * 表示関数
 */
void Disp(char ch, char str[9])
{
    int    i;
#ifdef USE_BCC
    /* 画面クリア */
    CLEAR;
#endif
    if      ((((ch == 'u') || (ch == 'j')) && (str[3] == 'y') && (str[2] == 'y'))
        ||  ((ch == 'k') && (str[3] == 'n') && (str[2] == 'y')))
    {
        for(i = 0; i < 4; i++)
        {
                PrintF(Monitor, "¥n    00000000    ");
        }
        for(i = 4; i < 12; i++)
        {
                PrintF(Monitor, "¥n                ");
        }
    }
```

```
    else if((((ch == 'U') ¦¦ (ch == 'j')) && (str[3] == 'n') && (str[2] == 'y'))
        ¦¦  (((ch == 'd') ¦¦ (ch == 'k')) && (str[3] == 'n') && (str[2] == 'n') && (str[1] == 'n') && (str[0] ==
'n'))
        ¦¦  ((ch == 's') && (str[3] == 'n') && (str[2] == 'y')))
    {
        for(i = 0; i < 2; i++)
        {
            PrintF(Monitor, "¥n            ");
        }
        for(i = 2; i < 6; i++)
        {
            PrintF(Monitor, "¥n    00000000    ");
        }
        for(i = 6; i < 12; i++)
        {
            PrintF(Monitor, "¥n            ");
        }
    }
    else if((ch == 's') && (str[3] == 'n') && (str[2] == 'n') && (str[1] == 'n') && (str[0] == 'n'))
    {
        for(i = 0; i < 4; i++)
        {
            PrintF(Monitor, "¥n            ");
        }
        for(i = 4; i < 8; i++)
        {
            PrintF(Monitor, "¥n    00000000    ");
        }
        for(i = 8; i < 12; i++)
```

```
        {
                PrintF(Monitor, "¥n              ");
        }
    }
    else if((((ch == 'u') ¦¦ (ch == 'j')) && (str[3] == 'n') && (str[2] == 'n') && (str[1] == 'n') && (str[0] ==
'n'))
        ¦¦  (((ch == 'D') ¦¦ (ch == 'k')) && (str[1] == 'y') && (str[0] == 'n'))
        ¦¦  ((ch == 's') && (str[1] == 'y') && (str[0] == 'n')))
    {
        for(i = 0; i < 6; i++)
        {
                PrintF(Monitor, "¥n              ");
        }
        for(i = 6; i < 10; i++)
        {
                PrintF(Monitor, "¥n    00000000    ");
        }
        for(i = 10; i < 12; i++)
        {
                PrintF(Monitor, "¥n              ");
        }
    }
    else if(((ch == 'j') && (str[1] == 'y') && (str[0] == 'n'))
        ¦¦  (((ch == 'd') ¦¦ (ch == 'k')) && (str[1] == 'y') && (str[0] == 'y')))
    {
        for(i = 0; i < 8; i++)
        {
                PrintF(Monitor, "¥n              ");
        }
```

```
        for(i = 8; i < 12; i++)
        {
                PrintF(Monitor, "¥n    00000000    ");
        }
    }
    else if((str[3] == 'y') && (str[2] == 'y'))
    {
        if    (((ch == 's') && (str[7] == 'y') && (str[6] == 'y'))
            ¦¦  (((ch == 'o') ¦¦ (ch == 'h')) && (str[7] == 'y') && (str[6] == 'y'))
            ¦¦  ((ch == 't') && (str[7] == 'n') && (str[6] == 'y')))
        {
            for(i = 0; i < 4; i++)
            {
                PrintF(Monitor, "¥n0000       0000");
            }
            for(i = 4; i < 12; i++)
            {
                PrintF(Monitor, "¥n              ");
            }
        }
        else if((((ch == 'O') ¦¦ (ch == 'h')) && (str[7] == 'n') && (str[6] == 'y'))
            ¦¦  (((ch == 'c') ¦¦ (ch == 't')) && (str[7] == 'n') && (str[6] == 'n') && (str[5] == 'n') && (str[4]
== 'n'))
            ¦¦  ((ch == 's') && (str[7] == 'n') && (str[6] == 'y')))
        {
            for(i = 0; i < 4; i++)
            {
                PrintF(Monitor, "¥n 0000     0000 ");
```

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

```
            }
        }
        else if(((ch == 's') && (str[5] == 'y') && (str[4] == 'y'))
            ¦¦  ((ch == 'h') && (str[5] == 'y') && (str[4] == 'n'))
            ¦¦  (((ch == 'c') ¦¦ (ch == 't')) && (str[5] == 'y') && (str[4] == 'y')))
        {
            for(i = 0; i < 4; i++)
            {
                PrintF(Monitor, "¥n    00000000    ");
            }
            for(i = 4; i < 12; i++)
            {
                PrintF(Monitor, "¥n                ");
            }
        }
    }
    else if((str[1] == 'y') && (str[0] == 'y'))
    {
        if    (((ch == 's') && (str[7] == 'y') && (str[6] == 'y'))
            ¦¦  (((ch == 'o') ¦¦ (ch == 'h')) && (str[7] == 'y') && (str[6] == 'y'))
            ¦¦  ((ch == 't') && (str[7] == 'n') && (str[6] == 'y')))
        {
            for(i = 0; i < 8; i++)
            {
                PrintF(Monitor, "¥n                ");
            }
            for(i = 8; i < 12; i++)
            {
                PrintF(Monitor, "¥n0000       0000");
```

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

```
        ¦¦ (((ch == 'C') ¦¦ (ch == 't')) && (str[5] == 'y') && (str[4] == 'n'))
        ¦¦ ((ch == 's') && (str[5] == 'y') && (str[4] == 'n')))
    {
        for(i = 0; i < 8; i++)
        {
            PrintF(Monitor, "¥n              ");
        }
        for(i = 8; i < 12; i++)
        {
            PrintF(Monitor, "¥n   0000  0000   ");
        }
    }
    else if(((ch == 's') && (str[5] == 'y') && (str[4] == 'y'))
        ¦¦ ((ch == 'h') && (str[5] == 'y') && (str[4] == 'n'))
        ¦¦ (((ch == 'c') ¦¦ (ch == 't')) && (str[5] == 'y') && (str[4] == 'y')))
    {
        for(i = 0; i < 8; i++)
        {
            PrintF(Monitor, "¥n              ");
        }
        for(i = 8; i < 12; i++)
        {
            PrintF(Monitor, "¥n    00000000    ");
        }
    }
}
/* 入力指示 */
DispInput();
```

```
        return;
    }
```

```
/* main.h */

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

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

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

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

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

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

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

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

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

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

```
/* main.c */

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

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

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

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

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

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

void main(void)
{
#ifndef USE_BCC
    char sw[4];
    int i;
    int j;
    int f;
    int cnt;
    static char buff[64];
#endif
#ifdef USE_THREAD
    Thread *th30;
    Thread *th31;
#endif
#ifndef USE_BCC
```

```
    for(i=0;i<0x7fff;i++) {}
    H8init(); /* H8 レジスタ初期化 */
    InitSCI(); /* SCI1初期化(serial) */
    InitLCD(); /* LCD初期化 */

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

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

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

    /* スイッチワーク初期化 */
    sw[0] = sw[1] = sw[2] = sw[3] = 0;
#else
    printf("¥nHello BCC");
#endif
#ifdef NOTUSE_FILES
```

```
    /* ファイル初期化 */
    new_EV_Status(&s);
#endif
#ifdef USE_THREAD
    /* タイマー初期化 */
    initWOVI();
    /* 2秒待機 */
    SleepMSec(2000);
    /* LEDTEST */
    th30 = new_Thread(30);
    th31 = new_Thread(31);
    Start(th30);
    Start(th31);
    for(;;)
    {
        /* タイマー呼び出し */
        wovi(5000000.0);
        if(Thread_checkStayAnother() == 1)
        {
            break;
        }
    }
#endif
    Clear();
#ifdef USE_THREAD
    /* LEDTEST */
    delete_(th30);
#endif
    PrintF(Panel, "NEXT          ");
```

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

        /*-----------------------------------------------*/
        /* HOSTからのシリアル入力をLCD,USBに送る */
        if( ScanSCI() ) /* SCIに受信データあり? */
        {
```

```
            i = GetSCI(); /* シリアル入力 */
            PutLCD(i); /* LCD出力 */
            buff[0] = i;
            write_buff(buff,1); /* USB出力 */
        }

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

        /*-----------------------------------------------*/
        /* 動作確認のため点滅 */
        SetLED(3,f);
        f ^= 1;
        for(i=0;i<10000;i++) {} /* 適当にウエイト */
    }
#else
    printf("END");
    /* 5秒待機 */
    SleepMSec(5000);
    return;
```

```
#endif
}

#ifdef USE_THREAD
/*
 * 擬似スレッドの擬似メソッド関数
 */
 /* public void paint(Graphics g)の代用 */
void Repaint(void)
{
#ifndef USE_BCC
    int i;
#else
    int i,j;
#endif
    Clear();
#ifndef USE_BCC
    for(i = 0; i < Cnt.cnt[0]; i++)
    {
        PrintF(Panel, " ");
    }
    PrintF(Panel, "<1>");
    PrintF(Panel, "¥n");
    for(i = 0; i < Cnt.cnt[1]; i++)
    {
        PrintF(Panel, " ");
    }
    PrintF(Panel, "<2>");
#else
```

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

/*
 * 疑似スレッドの疑似メソッド関数
 */
/* スレッドのpublic void run()の代用 */
void Run(Thread *This)
{
	int i;
	Thread *th11;
#ifdef USE_BCC
	char key = '¥0';
#else
	int j;
	char sw[4];
	/* スイッチワーク初期化 */
	sw[0] = sw[1] = sw[2] = sw[3] = 0;
#endif
```

```
        if(This->ID == 1)
        {
            Repaint();
#ifndef USE_BCC
            Cnt.cnt[0]++;
            nextRun(This, (((rand() % 9) + 10) * 100));
#else
            if(kbhit())
            {
#ifdef USE_LINUX
                key = (char) getchar();
#else
                key = (char) getche();
#endif
            }
            if(key == 'r')
            {
                Cnt.cnt[0]++;
            }
            nextRun(This, (((rand() % 9) + 10) * 30));
            while(kbhit())
            {
#ifdef USE_LINUX
                key = (char) getchar();
#else
                key = (char) getche();
#endif
            }
```

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

```
            }
#endif
        }
#ifdef USE_BCC
        else if(((This->ID) >= 3) && ((This->ID) <= 8))
        {
                Repaint();
                Cnt.cnt[(This->ID) - 1]++;
                nextRun(This, (((rand() % 9) + 10) * 200));
        }
#endif
        else if(This->ID == 11)
        {
            if(This->count == 1)
            {
                Clear();
                PrintF(Panel, "<1>1st         ");
                countUpNextRun(This, (1900 * 1));
            }
            else if(This->count == 2)
            {
                Clear();
                PrintF(Panel, "<1>2nd");
                PrintF(Panel, "<1>Stop   ");
                Stop(This);
            }
            else if(This->count == 3)
            {
                Clear();
```

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

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

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

            PrintF(Panel, "<1>Start  ");
            th11 = Thread_Start(11);
            countUpNextRun(th11, (1500 * 1));
        }
        else if(This->count == 3)
        {
```

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

            PrintF(Panel, "<4>Sto");
            Stop(This);
            delete_(This);
        }
    }
    else if(This->ID == 19)
    {
#ifdef USE_LINUX
        nextRun(This, 4000);
#else
        nextRun(This, 1);
#endif
#ifndef USE_BCC
        /* ボタンが押された時にLCD,SCI,USBにメッセージを送る */
        for(j=0;j<4;j++)
        {
            i = GetSW(j);
```

```
            if( ((sw[j]^1) & i) ) /* sw = off->onで条件成立 */
            {
                Thread_Toggle(j + 20);
                nextRun(This, 1000);
            }
        }
#else
        key = '¥0';
        key = GetChar(key);

        if(key == '1')
        {
            Thread_Toggle(21);
        }
        else if(key == '2')
        {
            Thread_Toggle(22);
        }
        else if(key == '3')
        {
            Thread_Toggle(23);
        }
        else if(key == '4')
        {
            Thread_Toggle(24);
        }
        else if(key == '5')
        {
            Thread_Toggle(25);
```

```
        }
        else if(key == '6')
        {
            Thread_Toggle(26);
        }
        else if(key == '7')
        {
            Thread_Toggle(27);
        }
        else if(key == '8')
        {
            Thread_Toggle(28);
        }
        else if(key == '9')
        {
            Thread_Toggle(29);
        }
        else if(key == '0')
        {
            Thread_Toggle(20);
        }
#endif
    }
    else if(This->ID == 20)
    {
        PrintF(Panel, "0");
        countUpNextRun(This, 2000);
    }
```

```
    else if(This->ID == 21)
    {
        PrintF(Panel, "1");
        countUpNextRun(This, 2000);
    }
    else if(This->ID == 22)
    {
        PrintF(Panel, "2");
        countUpNextRun(This, 2000);
    }
    else if(This->ID == 23)
    {
        PrintF(Panel, "3");
        countUpNextRun(This, 2000);
    }
#ifdef USE_BCC
    else if((This->ID >= 24) && (This->ID <= 29))
    {
        printf("%d", This->ID - 20);
        nextRun(This, 2000);
    }
#endif
#ifndef USE_BCC
    else if(This->ID == 30)
    {
        if(This->count == 0)
        {
            This->count++;
            PB.DR &= 0x0e;
```

```
            nextRun(This, 1000);
        }
        else if(This->count == 1)
        {
            This->count--;
            PB.DR ¦= 0x01;
            nextRun(This, 1000);
        }
    }
#endif
    else if(This->ID == 31)
    {
        if(This->count == 0)
        {
            /* 第1部分 */
#ifdef USE_BCC
            printf("¥nThread Ready GO! There are 8 cources on a race.");
            printf("¥nThere are 80 cells to a GOAL.");
            printf("¥nFor the <1> cource, You click a 'R' button.");
            printf("¥nFor the <2> cource, You click a 'L' button.");
#endif
#ifndef USE_BCC
            countUpNextRun(This, 0);
#else
            /* 5秒待機 */
            countUpNextRun(This, 5000);
#endif
        }
        else if(This->count == 1)
```

```
        {
            /* 擬似スレッド開始 */
            PrintF(Panel, "¥n");
            PrintF(Panel, "Thread Ready GO!");
            /* 2秒待機 */
            countUpNextRun(This, 2000);
        }
        else if(This->count == 2)
        {
#ifndef USE_BCC
            /* 擬似スレッドの擬似インスタンス初期化 */
            for(i = 0; i < 2; i++)
            {
                th[i] = new_Thread(i + 1);
            }
#else
            /* 擬似スレッドの擬似インスタンス初期化 */
            for(i = 0; i < 8; i++)
            {
                th[i] = new_Thread(i + 1);
            }
#endif
            countUpNextRun(This, 1);
        }
        else if(This->count == 3)
        {
#ifndef USE_BCC
            if(Cnt.cnt[0] >= 13)
            {
```

```
                i_cnt = 1;
                This->count++;
            }
            else if(Cnt.cnt[1] >= 13)
            {
                i_cnt = 2;
                This->count++;
            }
#else
            i_cnt = Cnt.cnt[0];
            j_cnt = 0;
            for(i = 1; i < 8; i++)
            {
                if(i_cnt < Cnt.cnt[i])
                {
                    i_cnt = Cnt.cnt[i];
                    j_cnt = i;
                }
            }
            if(i_cnt >= 77) This->count++;
#endif
            nextRun(This, 1);
        }
        else if(This->count == 4)
        {
            Clear();
            if(i_cnt == 1)
            {
```

```
                PrintF(Panel, "GOAL!<1>WON    ");
            }
            else if(i_cnt == 2)
            {
                PrintF(Panel, "GOAL!<2>WON    ");
            }
#ifdef USE_BCC
            else
            {
                printf("GOAL!¥n<%d>WON", (j_cnt + 1));
            }
#endif
#ifndef USE_BCC
            delete_(th[0]);
            delete_(th[1]);
#else
            for(i = 0; i < 8; i++)
            {
                delete_(th[i]);
            }
#endif
            /* 2秒待機 */
            countUpNextRun(This, 2000);
        }
        else if(This->count == 5)
        {
            Clear();
            PrintF(Panel, "NEXT          ");
            /* 2秒待機 */
```

```
        countUpNextRun(This, 2000);
    }
    else if(This->count == 6)
    {
        Clear();
        /* 第2部分 */
        PrintF(Panel, "CountUp        ");
        /* 2秒待機 */
        countUpNextRun(This, 2000);
    }
    else if(This->count == 7)
    {
        /* 疑似スレッド開始 */
        Clear();
        /* 疑似スレッドの疑似インスタンス初期化 */
        for(i = 0; i < 4; i++)
        {
            th1[i] = new_Thread(i + 11);
        }
        countUpNextRun(This, 1);
    }
    else if(This->count == 8)
    {
        if(Thread_checkStayAnother() == 2)
        {
            This->count++;
        }
        nextRun(This, 1);
    }
```

```
else if(This->count == 9)
{
	/* 2秒待機 */
	countUpNextRun(This, 2000);
}
else if(This->count == 10)
{
	Clear();
	PrintF(Panel, "NEXT          ");
	/* 2秒待機 */
	countUpNextRun(This, 2000);
}
else if(This->count == 11)
{
	Clear();
	/* 第3部分 */
	PrintF(Panel, "Toggle        ");
	/* 2秒待機 */
	countUpNextRun(This, 2000);
}
else if(This->count == 12)
{
	th19 = new_Thread(19);
	Start(th19);
	th20 = new_Thread(20);
	Start(th20);
	countUpNextRun(This, 1);
}
else if(This->count == 13)
```

```
{
	if(Thread_checkStayAnother() == 3)
	{
		delete_(th19);
		This->count++;
	}
	nextRun(This, 1);
}
else if(This->count == 14)
{
	countUpNextRun(This, 2000);
}
else if(This->count == 15)
{
	Clear();
	PrintF(Panel, "NEXT          ");
	/* 2秒待機 */
	countUpNextRun(This, 2000);
}
else if(This->count == 16)
{
	Clear();
	/* 第4部分 */
	PrintF(Panel, "EV            ");
	/* 2秒待機 */
	countUpNextRun(This, 2000);
}
else if(This->count == 17)
```

```
        {
            /* 擬似スレッドの擬似インスタンス初期化 */
            th41 = new_Thread(41);
            th42 = new_Thread(42);
            th43 = new_Thread(43);

            delete_(This);
        }
    }
    else if(This->ID == 41)
    {
        nextRun(This, 100);
        OnInput(&in, This);
    }
    if(This->ID == 42)
    {
        nextRun(This, 100);
        OnController(&cntrl, This);
    }
    else if(This->ID == 43)
    {
        nextRun(This, 2000);
        OnSimulator(&simu, This);
    }
    return;
}

/* スレッドのコンストラクタのpublic void init()の代用 */
void Init(Thread *This)
```

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

```
        PrintF(Panel, "<3>Init");
        countUpNextRun(This, (1500 * 3));
    }
    else if(This->ID == 14)
    {
        PrintF(Panel, "<4>Init  ");
        countUpNextRun(This, (1500 * 4));
    }
    else if(This->ID == 20)
    {
        Clear();
        PrintF(Panel, "<0>Init       ");
        PrintF(Panel, "¥n");
        countUpNextRun(This, 2000);
    }
    else if(This->ID == 21)
    {
        Clear();
        PrintF(Panel, "<1>Init       ");
        PrintF(Panel, "¥n");
        countUpNextRun(This, 2000);
    }
    else if(This->ID == 22)
    {
        Clear();
        PrintF(Panel, "<2>Init       ");
        PrintF(Panel, "¥n");
        countUpNextRun(This, 2000);
    }
```

```
    else if(This->ID == 23)
    {
        Clear();
        PrintF(Panel, "<3>Init        ");
        PrintF(Panel, "¥n");
        countUpNextRun(This, 2000);
    }
#ifdef USE_BCC
    else if((This->ID >= 24) && (This->ID <= 29))
    {
        printf("¥n<%d>Init¥n", This->ID - 20);
        nextRun(This,2000);
    }
#endif
#ifndef USE_BCC
    else if(This->ID == 30)
    {
        PB.DDR = 0xff; /* bit7..0 out  */
        PB.DR ¦= 0xff;
    }
#endif
    else if(This->ID == 41)
    {
        nextRun(This, 100);
        EV_Input(&in);
    }
    else if(This->ID == 42)
    {
```

```
        nextRun(This, 100);
        EV_Controller(&cntrl, This);
    }
    else if(This->ID == 43)
    {
        nextRun(This, 2000);
        EV_Simulator(&simu, This);
    }
    return;
}

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

```
        PrintF(Panel, "¥n");
        PrintF(Panel, "<4>Destroy     ");
    }
    if(This->ID == 20)
    {
        Clear();
        PrintF(Panel, "<0>Destroy     ");
        PrintF(Panel, "¥n");
    }
    else if(This->ID == 21)
    {
        Clear();
        PrintF(Panel, "<1>Destroy     ");
        PrintF(Panel, "¥n");
    }
    else if(This->ID == 22)
    {
        Clear();
        PrintF(Panel, "<2>Destroy     ");
        PrintF(Panel, "¥n");
    }
    else if(This->ID == 23)
    {
        Clear();
        PrintF(Panel, "<3>Destroy     ");
        PrintF(Panel, "¥n");
    }
#ifdef USE_BCC
    else if((This->ID >= 24) && (This->ID <= 29))
```

```
    {
        printf("¥n<%d>Destroy¥n", This->ID - 20);
    }
#endif
    return;
}
#endif

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

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

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

```
/*==========================================================================
                                    SW状態取得
---------------------------------------------------------------------
    int GetSW(int no)

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

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

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

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

```
	PA bit0..3	IN		SW0..3
	PB bit0..3	OUT		LED0..3  LCD DB4..7
	PB bit4		OUT			LCD RS
	PB bit7		OUT			LCD E
==========================================================*/
void H8init()
{
	BSC.ABWCR = 0x06; /* 8bit BUS MODE */

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

実行環境

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

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

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

```
        .CPU 300HA
        .SECTION        V,CODE,LOCATE=H'000000
        .IMPORT _main
        .IMPORT _usb_int
        .IMPORT _InterruptITU0

        .DATA.L         _start          ;リセットベクトル
;1 Reserved
_INT_Reserved1: .DATA.L         int_error
;2 Reserved
_INT_Reserved2: .DATA.L         int_error
;3 Reserved
_INT_Reserved3: .DATA.L         int_error
;4 Reserved
_INT_Reserved4: .DATA.L         int_error
;5 Reserved
_INT_Reserved5: .DATA.L         int_error
;6 Reserved
_INT_Reserved6: .DATA.L         int_error
;7 NMI
_INT_NMI:       .DATA.L         int_error
;8 TRAP
_INT_TRAP1:     .DATA.L         int_error
;9 TRAP
_INT_TRAP2:     .DATA.L         int_error
;10 TRAP
_INT_TRAP3:     .DATA.L         int_error
;11 TRAP
```

```
_INT_TRAP4:    .DATA.L       int_error
;12 IRQ0
IRQ0:  .DATA.L       int_error
;13 IRQ1
IRQ1:  .DATA.L       int_error
;14 IRQ2
IRQ2:  .DATA.L       int_error
;15 IRQ3
IRQ3:  .DATA.L       int_error
;16 IRQ4
IRQ4:  .DATA.L       int_error
;17 IRQ5
IRQ5:  .DATA.L       usb_interrupt        ;USB割り込み
;18 Reserved
_INT_Reserved18:      .DATA.L       int_error
;19 Reserved
_INT_Reserved19:      .DATA.L       int_error
;20 WOVI
_INT_WOVI:    .DATA.L       int_error
;21 CMI
_INT_CMI:     .DATA.L       int_error
;22 Reserved
_INT_Reserved22:      .DATA.L       int_error
;23 Reserved
_INT_Reserved23:      .DATA.L       int_error
;24 IMIA0
_INT_IMIA0:    .DATA.L       int_error
;25 IMIB0
_INT_IMIB0:    .DATA.L       int_error
```

```
;26 OVI0
_INT_OVI0:     .DATA.L       _ITU_OVI_0          ;タイマ0割り込み
;27 Reserved
_INT_Reserved27:       .DATA.L       int_error
;28 IMIA1
_INT_IMIA1:    .DATA.L       int_error
;29 IMIB1
_INT_IMIB1:    .DATA.L       int_error
;30 OVI1
_INT_OVI1:     .DATA.L       int_error
;31 Reserved
_INT_Reserved31:       .DATA.L       int_error
;32 IMIA2
_INT_IMIA2:    .DATA.L       int_error
;33 IMIB2
_INT_IMIB2:    .DATA.L       int_error
;34 OVI2
_INT_OVI2:     .DATA.L       int_error
;35 Reserved
_INT_Reserved35:       .DATA.L       int_error
;36 IMIA3
_INT_IMIA3:    .DATA.L       int_error
;37 IMIB3
_INT_IMIB3:    .DATA.L       int_error
;38 OVI3
_INT_OVI3:     .DATA.L       int_error
;39 Reserved
_INT_Reserved39:       .DATA.L       int_error
;40 IMIA4
```

```
_INT_IMIA4:     .DATA.L        int_error
;41 IMIB4
_INT_IMIB4:     .DATA.L        int_error
;42 OVI4
_INT_OVI4:      .DATA.L        int_error
;43 Reserved
_INT_Reserved43:        .DATA.L        int_error
;44 DEND0A
_INT_DEND0A:    .DATA.L        int_error
;45 DEND0B
_INT_DEND0B:    .DATA.L        int_error
;46 DEND1A
_INT_DEND1A:    .DATA.L        int_error
;47 DEND1B
_INT_DEND1B:    .DATA.L        int_error
;48 Reserved
_INT_Reserved48:        .DATA.L        int_error
;49 Reserved
_INT_Reserved49:        .DATA.L        int_error
;50 Reserved
_INT_Reserved50:        .DATA.L        int_error
;51 Reserved
_INT_Reserved51:        .DATA.L        int_error
;52 ERI0
_INT_ERI0:      .DATA.L        int_error
;53 RXI0
_INT_RXI0:      .DATA.L        int_error
;54 TXI0
_INT_TXI0:      .DATA.L        int_error
```

```
;55 TEI0
_INT_TEI0:      .DATA.L         int_error
;56 ERI1
_INT_ERI1:      .DATA.L         int_error
;57 RXI1
_INT_RXI1:      .DATA.L         int_error
;58 TXI1
_INT_TXI1:      .DATA.L         int_error
;59 TEI1
_INT_TEI1:      .DATA.L         int_error
;60 ADI
_INT_ADI:       .DATA.L         int_error

;------------------------------------------------------
        .SECTION        P,CODE,ALIGN=2
_start:
        mov.l   #H'0FFFF10,er7

;初期化付きデータを使用する場合、RAMに転送する
        mov.l   #H'91C0, er0                ; 転送元(91C0)
        mov.l   #H'0FFE000, er1             ; 転送先
        mov.l   #DATA_END, er2              ; 転送終了
init_loop:
        cmp.l   er1, er2
        beq     init_end
        mov.b   @er0+, r3l
        mov.b   r3l, @er1
        inc.l   #1, er1
```

```
        bra     init_loop
init_end:
        jsr     @_main

; 割り込み未使用
int_error:
        rte

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

_ITU_OVI_0:
        push.l  er0
        push.l  er1
```

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

;---------------------------------------------------------
; 割り込み許可、禁止ルーチン

        .EXPORT _EnableInterrupt,_DisableInterrupt
_EnableInterrupt:
        andc.b  #H'3f,ccr
        rts
_DisableInterrupt:
        orc.b   #H'c0,ccr
        rts

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

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

        .END
```

```
OUTPUT c_thread
PRINT c_thread
INPUT asmfile, main, EV_Simulator, EV_Controller, EV_Input, EV_OpenClose, EV_UpDown, EV_File,
EV_Time, Timer, Panel, sci, lcd, usb
LIB c:\h8\akic\c38hab
START R(0FFE000), P(200), D(91C0), C(9200)
ROM (D, R)
EXIT
```

```
@rem build.bat
C:
set bccDir="C:\borland\bcc55\Bin"
set akih8asmDir="c:\h8\akiasm"
set akih8cDir="c:\h8\akic"
set path=%bccDir%;%path%
set path=%akih8cDir%;%akih8asmDir%;%path%
set CurrentDir="%~dp0"
cd %CurrentDir%
del error.txt
make -f makefile.mak >> error.txt
make -f makefile.mak clean >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% Panel.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% Timer.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% EV_Time.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% EV_File.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% EV_UpDown.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% EV_OpenClose.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% EV_Input.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% EV_Controller.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% EV_Simulator.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% main.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% usb.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% sci.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir% lcd.c >> error.txt
a38h asmfile.src >> error.txt
l38h -subcommand=linkfile.sub >> error.txt
c38h c_thread >> error.txt
del %CurrentDir%*.obj >> error.txt
del %CurrentDir%c_thread.abs >> error.txt
error.txt
exit
```

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

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

実行環境状態ファイル

r

yynnyynn

q

C

S

出力ファイル

```
 Start         Length     Name                   Class
0001:00401000 00000F618H _TEXT                  CODE
0002:00411000 000003680H _DATA                  DATA
0003:00414680 000000B14H _BSS                   BSS
0004:00000000 0000000A4H _TLS                   TLS
```

PROGRAM NAME =

```
1                    1          .CPU 300HA
2 000000             2          .SECTION      V,CODE,LOCATE=H'000000
3                    3          .IMPORT _main
4                    4          .IMPORT _usb_int
5                    5          .IMPORT _InterruptITU0
6                    6
7 000000 00000000    7          .DATA.L       _start       ;リセットベクトル
8                    8     ;1 Reserved
9 000004 00000000    9     _INT_Reserved1: .DATA.L       int_error
10                   10    ;2 Reserved
11 000008 00000000   11    _INT_Reserved2: .DATA.L       int_error
12                   12    ;3 Reserved
13 00000C 00000000   13    _INT_Reserved3: .DATA.L       int_error
14                   14    ;4 Reserved
15 000010 00000000   15    _INT_Reserved4: .DATA.L       int_error
16                   16    ;5 Reserved
17 000014 00000000   17    _INT_Reserved5: .DATA.L       int_error
18                   18    ;6 Reserved
19 000018 00000000   19    _INT_Reserved6: .DATA.L       int_error
20                   20    ;7 NMI
21 00001C 00000000   21    _INT_NMI:     .DATA.L       int_error
22                   22    ;8 TRAP
23 000020 00000000   23    _INT_TRAP1:   .DATA.L       int_error
24                   24    ;9 TRAP
```

```
25 000024 00000000          25    _INT_TRAP2:    .DATA.L       int_error
26                          26    ;10 TRAP
27 000028 00000000          27    _INT_TRAP3:    .DATA.L       int_error
28                          28    ;11 TRAP
29 00002C 00000000          29    _INT_TRAP4:    .DATA.L       int_error
30                          30    ;12 IRQ0
31 000030 00000000          31    IRQ0:  .DATA.L       int_error
32                          32    ;13 IRQ1
33 000034 00000000          33    IRQ1:  .DATA.L       int_error
34                          34    ;14 IRQ2
35 000038 00000000          35    IRQ2:  .DATA.L       int_error
36                          36    ;15 IRQ3
37 00003C 00000000          37    IRQ3:  .DATA.L       int_error
38                          38    ;16 IRQ4
39 000040 00000000          39    IRQ4:  .DATA.L       int_error
40                          40    ;17 IRQ5
41 000044 00000000          41    IRQ5:  .DATA.L       usb_interrupt       ;USB割り込み
42                          42    ;18 Reserved
43 000048 00000000          43    _INT_Reserved18:      .DATA.L       int_error
44                          44    ;19 Reserved
45 00004C 00000000          45    _INT_Reserved19:      .DATA.L       int_error
46                          46    ;20 WOVI
47 000050 00000000          47    _INT_WOVI:     .DATA.L       int_error
48                          48    ;21 CMI
49 000054 00000000          49    _INT_CMI:      .DATA.L       int_error
50                          50    ;22 Reserved
51 000058 00000000          51    _INT_Reserved22:      .DATA.L       int_error
52                          52    ;23 Reserved
53 00005C 00000000          53    _INT_Reserved23:      .DATA.L       int_error
```

```
  54                    54    ;24 IMIA0
  55 000060 00000000          55    _INT_IMIA0:    .DATA.L       int_error
  56                    56    ;25 IMIB0
  57 000064 00000000          57    _INT_IMIB0:    .DATA.L       int_error
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***     12/13/17 18:41:00
PAGE    2
PROGRAM NAME =


  58                    58    ;26 OVI0
  59 000068 00000000          59    _INT_OVI0:     .DATA.L       _ITU_OVI_0          ;タイマ0割り込み
  60                    60    ;27 Reserved
  61 00006C 00000000          61    _INT_Reserved27:       .DATA.L       int_error
  62                    62    ;28 IMIA1
  63 000070 00000000          63    _INT_IMIA1:    .DATA.L       int_error
  64                    64    ;29 IMIB1
  65 000074 00000000          65    _INT_IMIB1:    .DATA.L       int_error
  66                    66    ;30 OVI1
  67 000078 00000000          67    _INT_OVI1:     .DATA.L       int_error
  68                    68    ;31 Reserved
  69 00007C 00000000          69    _INT_Reserved31:       .DATA.L       int_error
  70                    70    ;32 IMIA2
  71 000080 00000000          71    _INT_IMIA2:    .DATA.L       int_error
  72                    72    ;33 IMIB2
  73 000084 00000000          73    _INT_IMIB2:    .DATA.L       int_error
  74                    74    ;34 OVI2
  75 000088 00000000          75    _INT_OVI2:     .DATA.L       int_error
  76                    76    ;35 Reserved
  77 00008C 00000000          77    _INT_Reserved35:       .DATA.L       int_error
  78                    78    ;36 IMIA3
```

```
 79 000090 00000000          79    _INT_IMIA3:    .DATA.L       int_error
 80                          80    ;37 IMIB3
 81 000094 00000000          81    _INT_IMIB3:    .DATA.L       int_error
 82                          82    ;38 OVI3
 83 000098 00000000          83    _INT_OVI3:     .DATA.L       int_error
 84                          84    ;39 Reserved
 85 00009C 00000000          85    _INT_Reserved39:       .DATA.L       int_error
 86                          86    ;40 IMIA4
 87 0000A0 00000000          87    _INT_IMIA4:    .DATA.L       int_error
 88                          88    ;41 IMIB4
 89 0000A4 00000000          89    _INT_IMIB4:    .DATA.L       int_error
 90                          90    ;42 OVI4
 91 0000A8 00000000          91    _INT_OVI4:     .DATA.L       int_error
 92                          92    ;43 Reserved
 93 0000AC 00000000          93    _INT_Reserved43:       .DATA.L       int_error
 94                          94    ;44 DEND0A
 95 0000B0 00000000          95    _INT_DEND0A:   .DATA.L       int_error
 96                          96    ;45 DEND0B
 97 0000B4 00000000          97    _INT_DEND0B:   .DATA.L       int_error
 98                          98    ;46 DEND1A
 99 0000B8 00000000          99    _INT_DEND1A:   .DATA.L       int_error
100                         100    ;47 DEND1B
101 0000BC 00000000         101    _INT_DEND1B:   .DATA.L       int_error
102                         102    ;48 Reserved
103 0000C0 00000000         103    _INT_Reserved48:       .DATA.L       int_error
104                         104    ;49 Reserved
105 0000C4 00000000         105    _INT_Reserved49:       .DATA.L       int_error
106                         106    ;50 Reserved
107 0000C8 00000000         107    _INT_Reserved50:       .DATA.L       int_error
```

```
 108                    108    ;51 Reserved
 109 0000CC 00000000        109    _INT_Reserved51:      .DATA.L       int_error
 110                    110    ;52 ERI0
 111 0000D0 00000000        111    _INT_ERI0:    .DATA.L       int_error
 112                    112    ;53 RXI0
 113 0000D4 00000000        113    _INT_RXI0:    .DATA.L       int_error
 114                    114    ;54 TXI0
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***    12/13/17 18:41:00
PAGE    3
PROGRAM NAME =

 115 0000D8 00000000        115    _INT_TXI0:    .DATA.L       int_error
 116                    116    ;55 TEI0
 117 0000DC 00000000        117    _INT_TEI0:    .DATA.L       int_error
 118                    118    ;56 ERI1
 119 0000E0 00000000        119    _INT_ERI1:    .DATA.L       int_error
 120                    120    ;57 RXI1
 121 0000E4 00000000        121    _INT_RXI1:    .DATA.L       int_error
 122                    122    ;58 TXI1
 123 0000E8 00000000        123    _INT_TXI1:    .DATA.L       int_error
 124                    124    ;59 TEI1
 125 0000EC 00000000        125    _INT_TEI1:    .DATA.L       int_error
 126                    126    ;60 ADI
 127 0000F0 00000000        127    _INT_ADI:     .DATA.L       int_error
 128                    128
 129                    129    ;------------------------------------------------------
 130 000000                 130            .SECTION      P,CODE,ALIGN=2
 131 000000                 131    _start:
```

```
132 000000 7A0700FFFF10      132          mov.l   #H'0FFFF10,er7
133                          133
134                          134    ;初期化付きデータを使用する場合、RAMに転送する
135 000006 7A00000091C0      135          mov.l   #H'91C0, er0              ; 転送元(91C0)
136 00000C 7A0100FFE000      136          mov.l   #H'0FFE000, er1            ; 転送先
137 000012 7A0200000000      137          mov.l   #DATA_END, er2             ; 転送終了
138 000018                138    init_loop:
139 000018 1F92           139          cmp.l   er1, er2
140 00001A 58700008       140          beq     init_end
141 00001E 6C0B           141          mov.b   @er0+, r3l
142 000020 689B           142          mov.b   r3l, @er1
143 000022 0B71           143          inc.l   #1, er1
144 000024 40F2           144          bra     init_loop
145 000026                145    init_end:
146 000026 5E000000       146          jsr     @_main
147                          147
148                          148    ; 割り込み未使用
149 00002A                149    int_error:
150 00002A 5670           150          rte
151                          151
152 00002C                152    usb_interrupt:
153 00002C 01006DF0       153          push.l  er0
154 000030 01006DF1       154          push.l  er1
155 000034 01006DF2       155          push.l  er2
156 000038 01006DF3       156          push.l  er3
157 00003C 01006DF4       157          push.l  er4
158 000040 01006DF5       158          push.l  er5
159 000044 01006DF6       159          push.l  er6
160 000048 5E000000       160          jsr     @_usb_int
```

```
  161 00004C 01006D76          161          pop.l   er6
  162 000050 01006D75          162          pop.l   er5
  163 000054 01006D74          163          pop.l   er4
  164 000058 01006D73          164          pop.l   er3
  165 00005C 01006D72          165          pop.l   er2
  166 000060 01006D71          166          pop.l   er1
  167 000064 01006D70          167          pop.l   er0
  168 000068 5670            168          rte
  169                    169
  170 00006A                 170     _ITU_OVI_0:
  171 00006A 01006DF0          171          push.l  er0
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***     12/13/17 18:41:00
PAGE    4
PROGRAM NAME =


  172 00006E 01006DF1          172          push.l  er1
  173 000072 01006DF2          173          push.l  er2
  174 000076 01006DF3          174          push.l  er3
  175 00007A 01006DF4          175          push.l  er4
  176 00007E 01006DF5          176          push.l  er5
  177 000082 01006DF6          177          push.l  er6
  178 000086 5E000000          178          jsr @_InterruptITU0
  179 00008A 01006D76          179          pop.l   er6
  180 00008E 01006D75          180          pop.l   er5
  181 000092 01006D74          181          pop.l   er4
  182 000096 01006D73          182          pop.l   er3
  183 00009A 01006D72          183          pop.l   er2
  184 00009E 01006D71          184          pop.l   er1
  185 0000A2 01006D70          185          pop.l   er0
```

```
 186 0000A6 5670            186         rte
 187                  187
 188                  188   ;---------------------------------------------------------
 189                  189   ; 割り込み許可、禁止ルーチン
 190                  190
 191                  191         .EXPORT _EnableInterrupt,_DisableInterrupt
 192 0000A8              192   _EnableInterrupt:
 193 0000A8 063F          193         andc.b  #H'3f,ccr
 194 0000AA 5470          194         rts
 195 0000AC               195   _DisableInterrupt:
 196 0000AC 04C0          196         orc.b   #H'c0,ccr
 197 0000AE 5470          197         rts
 198                  198
 199                  199   ;---------------------------------------------------------
 200 000000               200         .SECTION      D,DATA
 201                  201
 202                  202
 203 000000               203         .SECTION      B,DATA
 204 000000 00000002        204   DATA_END:     .RES.W        1
 205                  205
 206                  206         .END
 *****TOTAL ERRORS     0
 *****TOTAL WARNINGS    0
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***    12/13/17 18:41:00
 PAGE    5

 *** CROSS REFERENCE LIST

 NAME                    SECTION  ATTR VALUE          SEQUENCE
```

```
B                       B       SCT  00000000    203*
D                       D       SCT  00000000    200*
DATA_END                        B          00000000    137   204*
IRQ0                    V          00000030    31*
IRQ1                    V          00000034    33*
IRQ2                    V          00000038    35*
IRQ3                    V          0000003C    37*
IRQ4                    V          00000040    39*
IRQ5                    V          00000044    41*
P                       P       SCT  00000000    130*
V                       V       SCT  00000000     2*
_DisableInterrupt               P       EXPT 000000AC    191   195*
_EnableInterrupt                P       EXPT 000000A8    191   192*
_INT_ADI                V          000000F0    127*
_INT_CMI                V          00000054     49*
_INT_DEND0A                V          000000B0     95*
_INT_DEND0B                V          000000B4     97*
_INT_DEND1A                V          000000B8     99*
_INT_DEND1B                V          000000BC    101*
_INT_ERI0               V          000000D0    111*
_INT_ERI1               V          000000E0    119*
_INT_IMIA0                 V          00000060     55*
_INT_IMIA1                 V          00000070     63*
_INT_IMIA2                 V          00000080     71*
_INT_IMIA3                 V          00000090     79*
_INT_IMIA4                 V          000000A0     87*
_INT_IMIB0                 V          00000064     57*
```

```
_INT_IMIB1              V       00000074    65*
_INT_IMIB2              V       00000084    73*
_INT_IMIB3              V       00000094    81*
_INT_IMIB4              V       000000A4    89*
_INT_NMI                V       0000001C    21*
_INT_OVI0               V       00000068    59*
_INT_OVI1               V       00000078    67*
_INT_OVI2               V       00000088    75*
_INT_OVI3               V       00000098    83*
_INT_OVI4               V       000000A8    91*
_INT_RXI0               V       000000D4   113*
_INT_RXI1               V       000000E4   121*
_INT_Reserved1          V       00000004    9*
_INT_Reserved18         V       00000048    43*
_INT_Reserved19         V       0000004C    45*
_INT_Reserved2          V       00000008    11*
_INT_Reserved22         V       00000058    51*
_INT_Reserved23         V       0000005C    53*
_INT_Reserved27         V       0000006C    61*
_INT_Reserved3          V       0000000C    13*
_INT_Reserved31         V       0000007C    69*
_INT_Reserved35         V       0000008C    77*
_INT_Reserved39         V       0000009C    85*
_INT_Reserved4          V       00000010    15*
_INT_Reserved43         V       000000AC    93*
_INT_Reserved48         V       000000C0   103*
_INT_Reserved49         V       000000C4   105*
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***     12/13/17 18:41:00
PAGE    6
```

*** CROSS REFERENCE LIST

NAME SECTION ATTR VALUE SEQUENCE

_INT_Reserved5 V 00000014 17*
_INT_Reserved50 V 000000C8 107*
_INT_Reserved51 V 000000CC 109*
_INT_Reserved6 V 00000018 19*
_INT_TEI0 V 000000DC 117*
_INT_TEI1 V 000000EC 125*
_INT_TRAP1 V 00000020 23*
_INT_TRAP2 V 00000024 25*
_INT_TRAP3 V 00000028 27*
_INT_TRAP4 V 0000002C 29*
_INT_TXI0 V 000000D8 115*
_INT_TXI1 V 000000E8 123*
_INT_WOVI V 00000050 47*
_ITU_OVI_0 P 0000006A 59 170*
_InterruptITU0 IMPT 00000000 5 178
_main IMPT 00000000 3 146
_start P 00000000 7 131*
_usb_int IMPT 00000000 4 160
init_end P 00000026 140 145*
init_loop P 00000018 138* 144
int_error P 0000002A 9 11 13 15 17 19 21 23 25 27 29 31
33 35 37 39 43 45 47 49 51 53 55 57
61 63 65 67 69 71 73 75 77 79 81 83
85 87 89 91 93 95 97 99 101 103 105 107

109 111 113 115 117 119 121 123 125 127 149*

usb_interrupt P 0000002C 41 152*

*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 *** 12/13/17 18:41:00

PAGE 7

*** SECTION DATA LIST

SECTION	ATTRIBUTE	SIZE	START
V	ABS-CODE	00000F4	000000
P	REL-CODE	00000B0	
D	REL-DATA	0000000	
B	REL-DATA	0000002	

S00E0000635F7468726561644D4F54C8
S1130000000002000000022A0000022A0000022A67
S1130010000022A0000022A0000022A0000022A2D
S1130020000022A0000022A0000022A0000022A1D
S10B00300000022A0000022A6D
S1130038000022A0000022A0000022A0000022C03
S1130048000022A0000022A0000022A0000022AF5
S1130058000022A0000022A0000022A0000022AE5
S10B00680000026A0000022AF5
S1130070000022A0000022A0000022A0000022ACD
S1130080000022A0000022A0000022A0000022ABD
S1130090000022A0000022A0000022A0000022AAD
S10B00A00000022A0000022AFD
S11300A80000022A0000022A0000022A0000022A95
S11300B80000022A0000022A0000022A0000022A85
S11300C80000022A0000022A0000022A0000022A75
S10B00D80000022A0000022AC5
S11300E00000022A0000022A0000022A0000022A5D
S10700F00000022ADD
S11302007A0700FFFF107A00000091C07A0100FF17
S1130210E0007A0200FFE0101F92587000086C0B98
S1130220689B0B7140F25E0002B0567001006DF0E6
S113023001006DF101006DF201006DF301006DF439
S113024001006DF501006DF65E004A2801006D7630
S113025001006D7501006D7401006D7301006D7215
S113026001006D7101006D70567001006DF00100A9
S11302706DF101006DF201006DF301006DF40100F9
S11302806DF501006DF65E00440801006D76010016
S11302906D7501006D7401006D7301006D720100D5
S11302A06D7101006D705670063F547004C0547038
S11302B05E005B3E7A370000000A790B00011933B8
S11302C00FF47A0600FFE19419550B5579257FFF4A
S11302D04DF85C000F6E5E0045A45E0047240D38A8
S11302E00D305C000EEA0D380DB05C000EE20D38E7
S11302F0790000025C000ED80D38790000035C0021
S11303000ECE5E0048CA7FF472507FF570505E00D7
S113031002A80D3228F117506DF07A000000920008
S113032001006DF05E0045F40B970B877A00000027
S1130330920F01006DF05E0047F80B9718886EC8A6
S113034000036EC800026EC8000168C87A0000FF8F
S1130350E0465E0039E65E0044347A00000007D0D0
S11303605E003F0C7900001E5E0040020F8579009D
S1130370001F5E00400201006FF000040FD05E001A
S113038041BE01006F7000045E0041BE01006B209E
S11303900000948A01006DF001006B200000948638
S11303A001006DF05E0044300B970B975E00422412
S11303B07920000146D65E00444E0FD05E00418A8C
S11303C07A010000921B0D305E0044767A00000033
S11303D007D05E003F0C5E00444E19550D505C0083
S11303E00E3C0D0D0D5117F10AC16819D90117D132
S11303F0660147540DB80D505C000DD40D5009B083
S11304006DF07A000000922C01006DF00FE05E00A9
S113041053E00B970B8701006DF67A0000009231D1
S113042001006DF05E0045F40B970B970FE05E0043
S1130430548409B00D010FE05E00525C01006DF6BB
S11304405E0047F80B9740080D380D505C000D8097
S11304500DD017F50FC10AD168980B557925000403

S108046058D0FF785E97
S11304650045E40D0047165E0045D417D00D055E23
S11304750047A868ED0DB10FE05E00525C5E0053C6
S11304857E0D004738790100400FE05E0053200DD3
S1130495057A010000923501006DF15E0047F80B06
S11304A59701006DF65E0047F80B9701006DF65E48
S11304B50045F40B970D510FE05E00525C0D287952
S11304C50000035C000D040D20795000010D021995
S11304D5550B55792527104DF85A0003DA54705EEC
S11304E5005B3E7A0500FFE01219665E00444E1973
S11304F5EE400E7A01000092370D605E0044760BE4
S11305055E69501D0E4DEC7A01000092390D605E57
S11305150044767A010000923D0D605E0044761931
S1130525EE400E7A01000092370D605E0044760BB3
S11305355E6F5000021D0E4DEA7A010000923F0DD9
S1130545605E0044765E005B1C54705E005B3E7A21
S1130555370000000A7A03000000017A040000074F
S1130565D019550F8618886EF800036EF800026ED1
S1130575F8000168F869607920000146365C00FFE0
S11305855E7A0000FFE01269010B5169815E005339
S1130595A617F07902000901D053207918000A79CA
S11305A5000064528017F00F810FE05E003F9E5AF2
S11305B5000E2269607920000246365C00FF207A2E
S11305C50000FFE01469010B5169815E0053A61712
S11305D5F07902000901D053207918000A79000047
S11305E564528017F00F810FE05E003F9E5A000EA4
S11305F52269607920000B586000B601006F600026
S1060605127A2043
S10A06080000000146205E23
S113060F00444E7A01000092430D505E0044767A07
S113061F010000076C0FE05E003FE25A000E22015B
S113062F006F6000127A200000000246265E00442D
S113063F4E7A01000092540D505E0044767A010009
S113064F00925B0D505E0044760FE05E0041EC5A62
S113065F000E2201006F6000127A20000000034693
S113066F1E5E00444E7A01000092660D505E0044F8
S113067F767A01000005DC0FE05E003FE2402401C3
S113068F006F6000127A200000000446165E0044DB
S106069F4E7A018C
S11306A2000092770D505E0044760FE05E0041EC4D
S11306B25A000E2269607920000C586000A80100DC
S11306C26F6000127A200000000146205E00444E53
S11306D27A01000092880D505E0044767A01000090
S11306E20D480FE05E003FE25A000E2201006F60E8
S11306F200127A200000000246205E00444E7A0176
S1130702000092990D505E0044767A0100000D4874
S11307120FE05E003FE25A000E2201006F600012FA
S11307227A2000000003461E5E00444E7A01000058
S108073292AA0D505EC8
S11307370044767A0100000D480FE05E003FE24077
S11307471C5E00444E7A01000092BB0D505E0044CC
S1130757760FE05E0041EC0FE05E00418A5A000E1F
S113076722696079200000D586000A801006F6000BE
S1130777127A200000000146205E00444E7A0100F1
S11307870092C30D505E0044767A01000013EC0F0C
S1130797E05E003FE25A000E2201006F6000127A0A
S11307A7200000000246205E00444E7A01000092BA

S11307B7D40D505E0044767A01000013EC0FE05E1F
S10707C7003FE25AB0
S11307CB000E2201006F6000127A20000000034626
S11307DB1E5E00444E7A01000092E50D505E00440C
S11307EB767A01000013EC0FE05E003FE2401C5EE3
S11307FB00444E7A01000092F60D505E0044760FD2
S113080BE05E0041EC0FE05E00418A5A000E226964
S113081B607920000E586000EA01006F6000127AC5
S113082B200000000146205E00444E7A0100009236
S113083BFE0D505E0044767A01000017700FE05EE8
S113084B003FE25A000E2201006F6000127A200073
S10E085B000002463E5E00444E7A019E
S11308660000930F0D505E0044767A010000177066
S11308760FE05E003FE27A01000093160D505E0022
S1130886447679000000B5E0042847A01000005DCA1
S11308965E003FE25A000E2201006F6000127A20CA
S11308A600000003461E5E00444E7A0100009321B9
S11308B60D505E0044767A01000017700FE05E006B
S11308C63FE2404001006F6000127A2000000004FE
S11308D646327900000B5E00424A01006FF00004C5
S11308E6470A01006F7000045E00418A7A01000026
S11308F693320D505E0044760FE05E0041EC0FE04C
S10809065E00418A5A66
S113090B000E2269607920001346440FB10FE05E9D
S113091B003F9E19DD0DD05C0008F80DD117F10FC8
S113092BF20A92682ADA0117D2660247160DD079BA
S113093B1000145E0042C27A01000003E80FE05E70
S113094B003F9E0B5D792D00044DCA5A000E2269A0
S113095B607920001446187A01000093390D505E1C
S113096B0044760FC10FE05E003FE25A000E22698E
S113097B607920001546187A010000933B0D505EF9
S113098B0044760FC10FE05E003FE25A000E22696E
S113099B607920001646187A010000933D0D505ED6
S10F09AB0044760FC10FE05E003FE25AEB
S11309B7000E2269607920001746187A0100009318
S11309C73F0D505E0044760FC10FE05E003FE25AD1
S11309D7000E2269607920001E465E01006F6000E9
S11309E712462401006F6000120B7001006FE000D4
S11309F71228D6E80E38D67A01000003E80FE05E26
S1130A07003F9E5A000E2201006F6000127A2000F9
S1130A17000001586004040 1006F6000121B70019D
S1130A27006FE000127FD670007A01000003E80F21
S1130A37E05E003F9E5A000E2269607920001F582E
S1130A476003B801006F600012460C1A910FE05E55
S1130A57003FE25A000E2201006F6000127A200065
S1130A6700000146247A010000923D0D505E0044C8
S1130A77767A01000093410D505E0044760FC10F53
S1090A87E05E003FE25AAD
S1130A8D000E2201006F6000127A20000000024662
S1130A9D3019DD0DD00B505E0040020DD217F21050
S1130AAD321032010078A06BA000FFE01A0B5D79C4
S1130ABD2D00024DDE0FB10FE05E003FE25A000E36
S1130ACD0001006F6000127A200000000346566B90
S1130ADD2000FFE0127920000D4D1A790000016B03
S1130AEDA000FFE01601006F6000120B7001006F94
S1130AFDE0001240246B2000FFE0147920000D4D1F
S1130B0D18790000026BA000FFE01601006F600072

S1130B1D120B7001006FE000120FB10FE05E003F8A
S1130B2D9E5A000E0001006F6000127A2000000033
S1130B3D04465A5E00444E6B2000FFE016792000F8
S1130B4D01460E7A01000093520D505E004476402B
S1060B5D186B20EF
S1130B6000FFE01679200002460C7A01000093632F
S1130B700D505E00447601006B2000FFE01A5E001A
S1130B80418A01006B2000FFE01E5E00418A0FC115
S1130B900FE05E003FE25A000E0001006F6000129A
S1130BA07A2000000005461C5E00444E7A010000D6
S1130BB0921B0D505E0044760FC10FE05E003FE2D2
S1130BC05A000E0001006F6000127A200000000638
S10B0BD0461C5E00444E7A014D
S1130BD8000093740D505E0044760FC10FE05E0071
S1130BE83FE25A000E0001006F6000127A200000F5
S1130BF8000746365E00444E19DD0DD07910000B10
S1130C085E0040020DD217F210321032010078A0B4
S1130C186BA000FFE0220B5D792D00044DDC0FB1C2
S1130C280FE05E003FE25A000E0001006F60001201
S1130C387A200000000846245E004224792000023E
S1130C48460E01006F6000120B7001006FE0001286
S1130C580FB10FE05E003F9E5A000E0001006F6067
S1130C6800127A2000000009460C0FC10FE05E0055
S1130C783FE25A000E0001006F6000127A20000064
S1130C88000A461C5E00444E7A010000921B0D5078
S1040C985EFA
S1130C990044760FC10FE05E003FE25A000E0001E7
S1130CA9006F6000127A200000000B461C5E0044AE
S1130CB94E7A01000093850D505E0044760FC10FF3
S1130CC9E05E003FE25A000E0001006F6000127AF5
S1130CD9200000000C4634790000135E0040020135
S1130CE9006BA000FFE0325E0041BE790000145E94
S1130CF900400201006BA000FFE0365E0041BE0F19
S1130D09B10FE05E003FE25A000E0001006F600080
S1130D19127A200000000D46305E0042247920003B
S10A0D2903461A01006B20D1
S1130D3000FFE0325E00418A01006F6000120B7019
S1130D4001006FE000120FB10FE05E003F9E5A00FA
S1130D500E0001006F6000127A200000000E460CA6
S1130D600FC10FE05E003FE25A000E0001006F600A
S1130D7000127A200000000F461A5E00444E7A01EA
S1130D800000921B0D505E0044760FC10FE05E0021
S1130D903FE2406C01006F6000127A2000000010F7
S1130DA0461A5E00444E7A01000093960D505E0091
S1130DB044760FC10FE05E003FE2404401006F60E4
S1130DC000127A200000001146367900002 95E00E7
S1130DD0400201006BA000FFE03A7900002A5E00A8
S1090DE0400201006BA0BC
S1130DE600FFE03E7900002B5E00400201006BA08D
S1130DF600FFE0420FE05E00418A402069607920EF
S1130E06002946187A01000000640FE05E003F9E49
S1130E160FE17A0000FFE04A5E002612696079203E
S1130E26002A461A7A01000000640FE05E003F9E26
S1130E360FE17A0000FFE07A5E001FB2401C696092
S1130E467920002B46140FC10FE05E003F9E0FE191
S1130E567A0000FFE14C5E00135A7A170000000A7D
S1130E665E005B1C54705E005B3E0F8679040009CE

S1130E767A0500FFE0126960792000014626190011
S1130E8669D05E0053A617F001D053407918000AC3
S1130E967900001E528017F00F810FE05E003F9E1F
S1040EA65AEE
S1130EA70010D4696079200002462819006FD0002A
S1130EB7025E0053A617F001D053407918000A7950
S1130EC700001E528017F00F810FE05E003F9E5A0D
S1130ED70010D4696C792C00034D36792C00084E29
S1130EE73069601B5017F010300A85190069D05E0E
S1130EF70053A617F001D053407918000A79000070
S1130F07C8528017F00F810FE05E003F9E5A001012
S1130F17D47A0400004476195569607920000B469A
S1130F271A7A01000093A70D505D407A010000056E
S1130F37DC0FE05E003FE25A0010D46960792000BD
S1130F470C461A7A01000093AF0D505D407A0100F9
S1130F57000BB80FE05E003FE25A0010D4696079D6
S1130F6720000D46247A010000923D0D505D407A22
S1040F770175
S1130F78000093B90D505D407A01000011940FE011
S1130F885E003FE25A0010D469607920000E461AC9
S1130F987A01000093C10D505D407A01000017707B
S1130FA80FE05E003FE25A0010D47A03000007D036
S1130FB869607920001446245E00444E7A010000DB
S1130FC893CB0D505D407A010000923D0D505D407A
S1130FD80FB10FE05E003FE25A0010D46960792038
S1130FE8001546245E00444E7A01000093DC0D5040
S1130FF85D407A010000923D0D505D400FB10FE056
S10810085E003FE25A07
S113100D0010D4696079200001646245E00444E7AA0
S113101D01000093ED0D505D407A010000923D0DEE
S113102D505D400FB10FE05E003FE25A0010D469EE
S113103D607920001746225E00444E7A010000932A
S113104DFE0D505D407A010000923D0D505D400F45
S113105DB10FE05E003FE2406E69607920001E46ED
S113106D08F8FF38D438D6405E69607920002946E8
S113107D187A01000000640FE05E003F9E7A0000C5
S113108DFFE04A5E002538403E69607920002A461C
S113109D1A7A01000000640FE05E003F9E0FE17AB3
S11310AD0000FFE07A5E001E94401C696079200009
S10B10BD2B46140FB10FE05E96
S11310C5003F9E0FE17A0000FFE14C5E0012705E67
S11310D5005B1C54705E005B3E7A04000044761985
S11310E5660F8569507920000B46105E00444E7AE1
S11310F5010000940F0D605D404044695079200064
S11311050C460C7A010000941A0D605D404030696D
S1131115507920000D460C7A01000094240D605D82
S113112540401C69507920000E46147A0100009254
S11311353D0D605D407A010000942E0D605D4069B0
S1131145507920014461A5E00444E7A010000943B
S11311553F0D605D407A010000923D0D605D4040AA
S1121165646950792000154 61A5E00444E7A01E2
S1131174000094500D605D407A010000923D0D60C3
S11311845D4040426950792000164 61A5E00444E81
S11311947A01000094610D605D407A010000923D84
S11311A40D605D4040206955792500174 6185E009F
S11311B4444E7A01000094730D605D407A0100008F
S11311C4923D0D605D405E005B1C54706DF66DF5E1

S11311D40D067909000128D617500D950CE91A0953
S11311E44B04101540F866050D8846121A0E4B047D
S11311F4101940F80D9029D6148939D640141A0EC3
S11312044B04101940F8795900FF0D9029D616891B
S113121439D60D506D756D7654706DF60D0628D361
S1131224175079010011A0E4B04101140F866108F
S11312344704191140047901000 10D106D765470AF
S1131244F80638ECF8FF38C038C1188838D8F8FFE6
S113125438C8188838DBF8FF38C9F8DF38D0F8FF06
S109126438CDF8F038D18B
S109126AF8FF38D45470B4
S1139200435055204D4F444520253032580A000C19
S113921052656164792133303532004E455854200C
S11392202020202020202020202020007377257557
S11392300025730A000C0020003C313E000A003C6C
S1139240323E003C313E3173742020202020202008
S1139250202020003C313E326E64003C313E53748A
S11392606F70202020003C313E3372642020202088
S1139270202020202020003C313E53746F7020209A
S11392802020202020202020003C323E3173742020F7
S11392902020202020202020003C323E326E6420FB
S11392A02020202020202020202020003C323E337264E6
S11392B020202020202020202020003C323E5374F8
S11392C06F70003C333E3173742020202020202017
S11392D0202020003C333E326E64202020202020BA
S11392E020202020003C333E3372642020202020A5
S10992F0202020202000D5
S11392F63C333E53746F70003C343E31737420200C
S1139306202020202020202020003C343E326E6400A2
S11393163C313E53746172742020003C343E3372F8
S113932664202020202020202020202020003C343E538F
S1139336746F00300031003200330054687265 6187
S113934664205265616479204 74F2100474F414CA1
S1139356213C313E574F4E202020202000474F41CD
S11393664C213C323E574F4E202020202000436F95
S1139376756E74557020202020202020202020005454
S11393866F67676C65202020202020202020202020086
S113939645562020202020202020202020202020202069
S11393A6003C313E496E6974003C323E496E697435
S11393B62020003C333E496E6974003C343E496EBE
S11393C669742020003C303E496E697420202020B9
S11393D62020202020003C313E496E697420202045
S10993E6202020202020BE
S11393EC003C323E496E69742020202020202020202E
S11393FC20003C333E496E6974202020202020202020 1D
S113940C2020003C313E44657374726F79003C320A
S113941C3E44657374726F003C333E4465737472DF
S113942C6F003C343E44657374726F7920202020A6
S113943C2020003C303E44657374726F79202020E9
S113944C202020003C313E44657374726F792020D8
S113945C20202020003C323E44657374726F7920C7
S113946C202020202020003C333E44657374726F0F
S113947C792020202020202020000415312D000000E
S105948C0000DB
S11312705E005B3E0F860F957A00000000200AE0B7
S11312800FD15E003DA27A00000000200AE05E005C
S11312903DE601006FE600027A00000000060AE066

S11312A001006FE000087A000000000C0AE0010072
S11312B06FE0000E7A00000000120AE001006FE008
S11312C0001C18886EE8001A7A0500003A347A0088
S11312D0000000380AE05D507A000000003C0AE09C
S11312E05D507A00000000400AE05D507A00000083
S11312F000440AE05D5001006F60000201006DF0E0
S11313007A00000000400AE07A010000948E5E003B
S11313103AB40B9779200001473A790000096DF040
S113132001006F60001C01006DF07A0000000044B2
S11313300AE07A01000094995E003AF80B970B8754
S1131340792000001470E7A01000000120AE1686863
S11313505C00042A5E005B1C54705E005B3E0F86DB
S10913600F9501006F6010
S1131366000E01006DF07A000000003C0AE07A01ED
S1131376000094A45E003AB40B976E68000CA87143
S1131386460E5E00444E0FD05E00418A5A0017286F
S113139601006F60000201006DF07A00000000405A
S11313A60AE07A010000948E5E003AB40B97790046
S11313B600096DF001006F60001C01006DF07A00FA
S11013C6000000440AE07A01000094995EE3
S11313D3003AF80B970B8701006F60000801006D5B
S11313E3F07A00000000380AE07A01000094B15E4D
S11313F3003AB40B976E680006A873461CF84E6D4B
S1131403F07A000000003C0AE07A01000094A45E35
S1131413003A3E0B875A00171C6868A87546366E58
S1131423680012A879587002D06E680013A8794631
S11314330AF86E6EE800135A0016FC6E680014A8CF
S11314436E470E6E680015A86E4606F8796EE800BF
S1131453155A0016FC6868A855462E6E680012A834
S11314637958700294 6E680013A8795870028A6ED3
S1131473680014A86E4608F8796EE8001440066EF7
S1131483680015A86E5A0016FC6868A86A46466E7B
S1131493680012A879460AF86E6EE800125A00161D
S11314A3FC6E680013A879460AF86E6EE800135AB7
S11314B30016FC6E680014A86E4608F8796EE800FF
S10814C314400E6E68E9
S11314C80015A86E4606F8796EE800155A0016FC52
S11314D86868A86446366E680015A87958700214BF
S11314E86E680014A879460AF86E6EE800145A006C
S11314F816FC6E680013A86E470E6E680012A86E7D
S11315084606F8796EE800125A0016FC6868A84483
S1131518462E6E680015A879587001D86E680014B5
S1131528A879587001CE6E680013A86E4608F8793A
S11015386EE8001340066E680012A86E5A9C
S11315450016FC6868A86B46466E680015A87946C0
S11315550AF86E6EE800155A0016FC6E680014A8AA
S113156579460AF86E6EE800145A0016FC6E680098
S1131577513A86E4608F8796EE80013400E6E6800EE
S1131588512A86E4606F8796EE800125A0016FC6832
S1131599568A86F46366E680016A879587001586EAC
S11315A5680017A879460AF86E6EE800175A001600
S11315B5FC6E680018A86E470E6E680019A86E4683
S11315C506F8796EE800195A0016FC6868A84F46B4
S11315D52E6E680016A8795870011C6E680017A84E
S11315E579587001126E680018A86E4608F8796E6E
S11315F5E8001840066E680019A86E5A0016FC68C4
S1131605 68A86846466E680016A879460AF86E6E9D

S1131615E800165A0016FC6E680017A879460AF802
S11316256E6EE800175A0016FC6E680018A86E4621
S108163508F8796EE8DE
S113163A0018400E6E680019A86E4606F8796EE81F
S113164A00195A0016FC6868A86346346E680019C4
S113165AA8795870009C6E680018A879460AF86E33
S113166A6EE800185A0016FC6E680017A86E470E3B
S113167A6E680016A86E4606F8796EE80016407280
S113168A6868A84346286E680019A87947646E6893
S113169A0018A879475C6E680017A86E4608F8799F
S11316AA6EE8001740066E680016A86E4044686824
S11316BAA874463E6E680019A8794608F86E6EE863
S11316CA0019402E6E680018A8794608F86E6EE86D
S11316DA0018401E6E680017A86E4608F8796EE86F
S11316EA0017400E6E680016A86E4606F8796EE873
S11316FA00167A00000000120AE001006DF07A0079
S113170A000000440AE07A01000094995E003A7CE2
S113171A0B977A01000000120AE1686855565E00C9
S108172A5B1C54705E1E
S113172F005B3E7A05000044767906000027A0100D9
S113173F0094BD0D605D507A01000094EC0D605D67
S113174F507A010000950D0D605D507A01000095F0
S113175F390D605D507A01000095670D605D507A19
S113176F01000095740D605D505E005B1C547001A9
S113177F006DF6790600030C80A0754704A06A4636
S113178F106E180003A87946086E180002A879474F
S113179F14A06B46426E180003A86E463A6E1800EB
S11317AF02A879463219EE7A010000957E0D605E2C
S11317BF0044760B5E792E00044DEC790E00047A0B
S11317CF01000095900D605E0044760B5E792E004C
S11317DF0C4DEC5A001E8AA0554704A06A46106EA2
S11317EF180003A86E46086E180002A879473AA09E
S11317FF644704A06B461E6E180003A86E46166E50
S113180F180002A86E460E6E180001A86E460668F1
S109181F18A86E4714A097
S113182573465A6E180003A86E46526E180002A836
S111183579464A19EE7A01000095900D605E27
S11318430044760B5E792E00024DEC790E00027A8A
S1131853010000957E0D605E0044760B5E792E00D9
S1131863064DEC790E00067A01000095900D605E3B
S11318730044760B5E792E000C4DEC5A001E8AA0B1
S11318837346686E180003A86E46606E180002A8BC
S11318936E46586E180001A86E46506818A86E4627
S11318A34A19EE7A01000095900D605E0044760BB1
S11318B35E792E00044DEC790E00047A0100009545
S11318C37E0D605E0044760B5E792E00084DEC7945
S11318D30E00087A01000095900D605E0044760BBC
S11318E35E792E000C4DEC5A001E8AA0754704A0A6
S11318F36A461E6E180003A86E46166E180002A8E9
S11319036E460E6E180001A86E46066818A86E4749
S113191328A0444704A06B460E6E180001A879461D
S1131923066818A86E4712A07346586E180001A8DC
S10919337946506818A874
S11319396E464A19EE7A01000095900D605E0044E7
S1121949760B5E792E00064DEC790E00067A01BF
S11319580000957E0D605E0044760B5E792E000ACA
S11319684DEC790E000A7A01000095900D605E0037

S113197844760B5E792E000C4DEC5A001E8AA06A41
S1131988460E6E180001A87946066818A86E471611
S1131998A0644704A06B46406E180001A879463836
S11319A86818A879463219EE7A01000095900D60FF
S11319B85E0044760B5E792E00084DEC790E000824
S11319C87A010000957E0D605E0044760B5E792EE9
S11319D8000C4DEC5A001E8A6E180003A879586053
S11319E8024E6E180002A87958600244A07346108C
S11319F86E180007A87946086E180006A879472CC0
S1131A08A06F4704A06846106E180007A879460817
S1131A186E180006A8794714A07446426E1800078A
S1131A28A86E463A6E180006A879463219EE7A016E
S1131A38000095A20D605E0044760B5E792E0004CB
S1091A484DEC790E0004D1
S1131A4E7A01000095900D605E0044760B5E792E50
S1131A5E000C4DEC5A001E8AA04F4704A068461096
S1131A6E6E180007A86E46086E180006A879473C44
S1131A7EA0634704A07446206E180007A86E46188C
S1131A8E6E180006A86E46106E180005A86E46085E
S1131A9E6E180004A86E4714A07346426E18000712
S1131AAEA86E463A6E180006A879463219EE7A01E8
S1131ABE000095B40D605E0044760B5E792E000433
S1131ACE4DEC790E00047A01000095900D605E00D6
S1131ADE44760B5E792E000C4DEC5A001E8AA073D1
S1131AEE46526E180007A86E464A6E180006A86E78
S1131AFE46426E180005A86E463A6E180004A86E8C
S1131B0E463219EE7A01000095C60D605E004476EA
S1131B1E0B5E792E00044DEC790E00047A01000061
S1131B2E95900D605E0044760B5E792E000C4DECA5
S1131B3E5A001C34A06F4704A06846206E18000795
S1131B4EA86E46186E180006A86E46106E1800058D
S1131B5EA86E46086E180004A86E472CA0434704CF
S1131B6EA07446106E180005A87946086E18000476
S1131B7EA86E4714A07346406E180005A879463820
S1131B8E6E180004A86E463019EE7A01000095D83F
S1131B9E0D605E0044760B5E792E00044DEC790EDB
S1071BAE00047A01B1
S1131BB2000095900D605E0044760B5E792E000C5A
S1131BC24DEC406EA07346106E180005A8794608C6
S1131BD26E180004A879472CA06846106E180005F9
S1131BE2A87946086E180004A86E4718A063470434
S1131BF2A074463E6E180005A87946366E18000496
S1121C02A879462E19EE7A010000957E0D605EDB
S1131C110044760B5E792E00044DEC790E00047AB4
S1131C2101000095900D605E0044760B5E792E00F5
S1131C310C4DEC5A001E8A6E180001A879586002F7
S1131C41486818A87958600240A07346106E1800BE
S1131C5107A87946086E180006A879472CA06F4794
S1131C6104A06846106E180007A87946086E18008C
S1131C7106A8794714A07446426E180007A86E4659
S1131C813A6E180006A879463219EE7A01000095DA
S1131C91900D605E0044760B5E792E00084DEC7961
S1131CA10E00087A01000095A20D605E0044760BD8
S1131CB15E792E000C4DEC5A001E8AA04F4704A0FA
S1131CC16846106E180007A86E46086E180006A82D
S1131CD179473CA0634704A07446206E180007A807
S1131CE16E46186E180006A86E46106E180005A8F9

S1131CF16E46086E180004A86E4714A07346426E20
S1091D01180007A86E465E
S1131D073A6E180006A879463219EE7A0100009553
S1131D17900D605E0044760B5E792E00084DEC79DA
S1131D270E00087A01000095B40D605E0044760B3F
S1131D375E792E000C4DEC5A001E8AA07346526E34
S1131D47180007A86E464A6E180006A86E46426E2C
S1131D57180005A86E463A6E180004A86E46321995
S10D1D67EE7A01000095900D605E16
S1131D710044760B5E792E00084DEC790E00087A4B
S1131D8101000095C60D605E0044760B5E792E005E
S1131D910C4DEC5A001E8AA06F4704A06846206EC2
S1131DA1180007A86E46186E180006A86E46106E36
S1131DB1180005A86E46086E180004A86E472CA0EB
S1131DC1434704A07446106E180005A87946086EAF
S1131DD1180004A86E4714A07346406E180005A8A6
S1131DE17946386E180004A86E463019EE7A010060
S1131DF10095900D605E0044760B5E792E00084DD0
S1131E01EC790E00087A01000095D80D605E00445C
S1131E11760B5E792E000C4DEC406EA07346106E6E
S1131E21180005A87946086E180004A879472CA064
S1131E316846106E180005A87946086E180004A8B4
S1131E416E4718A0634704A074463E6E180005A8A8
S1131E517946366E180004A879462E19EE7A0100E8
S1091E610095900D605E88
S1131E670044760B5E792E00084DEC790E00087A54
S1131E77010000957E0D605E0044760B5E792E00AF
S1101E870C4DEC5C00F8A001006D7654706A
S113948E4D6F746F722E74787400004C696D69742D
S113949E2E7478740000436F6D6D616E642E747854
S11394AE7400005361666574792E74787400000A33
S11394BE5550203D202775272C20444F574E203DD5
S11394CE202764272C204F50454E203D20276F2701
S11394DE2C20434C4F5345203D20276327000A453C
S11394EE4D455247454E4359203D202773272C2087
S11394FE5245434F56455259203D20277227000AA5
S113950E31737420466C6F6F722043414C4C203D77
S113951E202779272C20326E6420466C6F6F7220C1
S113952E43414C4C203D20275927000A31737420A8
S113953E466C6F6F7220434C4F5345203D20276876
S113954E272C20326E6420466C6F6F7220434C4F73
S113955E5345203D20274827000A5155495445209D
S113956E3D20277127000A434F4D4D414E443E0087
S109957E0A20202020302A
S1139584303030303030302020202000A2020209A
S1139594202020202020202020202020000A30EA
S11395A4303030202020202020202030303030300064
S11395B40A203030303020202020202030303030303A
S11395C420000A20203030303020202020303030305A
S11395D4302020000A2020203030303020203030304A
S10995E4303020202000BE
S1131E945E005B3E0F860F9501006FE60024010090
S1131EA46F6000245E00318E7A00000000280AE08F
S1131EB45E0032027A00000000300AE05E003464FF
S1131EC47A00000000380AE05E0036C67A0000009B
S1131ED4004C0AE001006FE000705E0029167A00EE
S1131EE4000000740AE05E00298A7A000000007C86

S1131EF40AE05E002BEC7A00000000840AE05E0036
S1131F042E4E7A00000000A20AE00FD15E003DA22B
S1131F147A00000000C60AE05E003A347A0000004A
S1131F2400CA0AE05E003A347A00000000CE0AE0F8
S1131F345E003A347A00000000BA0AE001006FE060
S1131F4400BC7A00000000C00AE001006FE000C298
S1131F547A00000000CE0AE0F9735E003CD87A00F0
S1131F64000000380AE001006F6100245E0036E0DF
S1131F747A00000000840AE001006F6100705E00D3
S1071F842E767A0038
S1131F88000000CA0AE0F94E5E003BC07920000158
S1131F9847127A00000000CA0AE0F9635E003B308A
S1131FA8792000015E005B1C54705E005B3EFC639D
S1131FB8F54EFD730F860F9301006F6000BC01009F
S1131FC86DF07A00000000C60AE07A01000095EA85
S1041FD85EA7
S1131FD9003AB40B9779200001587004E07A0000A5
S1131FE90000CA0AE001006F6100C25E003B747918
S1131FF9200001587004C66E6800C0A848587003D1
S113200976A8594756A863587002DCA8645870012A
S11320198CA86858700406A86F5870026EA8714797
S11320291AA87258700496A87358700490A8754733
S113203954A8795870013C5A0024C67A000000005C
S1132049CE0AE00CD95E003CD80FB05E00418A5A33
S11320590024C65A0024C65A0024C601006F600032
S11320692401006F00000C69007920000146160164
S1132079006F60007001006F00001469007920008F
S113208901587004380 1006F60002401006F0000DB
S1132099146900792000014 61A01006F600070017C
S11320A9006F000014690079200001587002 4C5A2E
S11320B90022A801006F60007001006F00000C6925
S10920C90079200001462E
S11320CF4E7A00000000380AE07A37000000140F40
S11320DFF17A02000000145E005AE67A0000000055
S11320EF280AE07A37000000080FF17A0200000097
S11320FF085E005AE601006F6100BC01006F6000CB
S113210F245E00394A7A170000001C5A0024C601C6
S113211F006F60007001006F00000C6901464A7A7E
S113212F00000000840AE07A370000001E0FF17AE6
S113213F020000001E5E005AE67A000000007C0ACF
S113214FE07A37000000080FF17A02000000085E02
S113215F005AE601006F6100BC01006F6000705E02
S10D216F0031407A17000000265AE1
S11321790024C601006F60002401006F0000146988
S1132189007920000146160 1006F60007001006F9D
S113219900001469007920000158700320 01006FC1
S11321A960002401006F00000C69007920000146DA
S11321B91A01006F60007001006F00001469007953
S11321C9200001587001345A0022A801006F6000F1
S11321D97001006F00000C690079200001464E7AF6
S11321E900000000380AE07A37000000140FF17A82
S11321F902000000145E005AE67A00000000300A6B
S1132209E07A37000000080FF17A02000000085E47
S1132219005AE601006F6100BC01006F6000245E93
S11322290039987A170000001C5A0024C601006F70
S1132239600070 01006F00000C6901464A7A0000D2
S1132249000840AE07A370000001E0FF17A0200C9

S113225900001E5E005AE67A000000007C0AE07A5C
S108226937000000082E
S113226E0FF17A02000000085E005AE601006F616A
S113227E00BC01006F6000705E0031407A170000F1
S113228E00265A0024C601006F60007001006F0023
S113229E0014690079200001475C7A000000008475
S11322AE0AE07A370000001E0FF17A020000001ECA
S11322BE5E005AE67A00000000740AE07A370000E6
S11322CE00080FF17A02000000085E005AE60100D2
S11322DE6F6100BC01006F6000705E0030F25A0047
S11322EE237601006F60007001006F00000C69001F
S11322FE79200001462E7A00000000CE0AE00CD9A8
S113230E5E003CD87A00000000CA0AE00C595E0059
S113231E3BC07A00000000CA0AE00CC95E003B30E5
S104232E5A51
S113232F0024C67A00000000840AE07A3700000018
S113233F1E0FF17A020000001E5E005AE67A0000BB
S113234F00007C0AE07A37000000080FF17A0200E0
S113235F0000085E005AE601006F6100BC01006FC8
S113236F6000705E0031407A17000000265A002487
S113237FC601006F60002401006F0000146900792B
S113238F2000015860013001006F60007001006F81
S113239F00000C690079200001462E7A000000002E
S11323AFCE0AE00CD95E003CD87A00000000CA0ABE
S11323BFE00C595E003BC07A00000000CA0AE00C33
S11323CFC95E003B305A0024C67A00000000840A1D
S11323DFE07A370000001E0FF17A020000001E5E44
S11323EF005AE67A000000007C0AE07A370000000A
S11323FF080FF17A02000000085E005AE601006F31
S113240F6100BC01006F6000705E0031407A1700FD
S107241F0000265A36
S11324230024C601006F60002401006F00000C69E3
S1132433007920000158600 8A01006F6000700179
S1132443006F00000C690079200001462C7A00001C
S11324530000CE0AE00CD95E003CD87A00000000ED
S1132463CA0AE00C595E003BC07A00000000CA0AA6
S1132473E00CC95E003B30404A7A00000000840A46
S1132483E07A370000001E0FF17A020000001E5E9F
S1132493005AE67A000000007C0AE07A3700000065
S11324A3080FF17A02000000085E005AE601006F8C
S10D24B36100BC01006F6000705E61
S11224BD0031407A17000000265E005B1C54704C
S10F95EA5361666574792E747874000078
S11324CC5E005B3E1B970FF40C8D18886EC80003DF
S11324DC6EC800026EC8000168C819660D605E0004
S11324EC121E0D0E0D6117F10AC16819D90117D10E
S11324FC660147260D600C004620A800470EA80174
S113250C470EA802470EA803470E400EFD75400A5E
S113251CFD644006FD6F4002FD630B5679260004F3
S113252C4DBA0CD80B975E005B1C54705E005B3E7F
S113253C0F8618886EE800066EE800077A00000024
S113254C00080AE001006FE0000A7A000000000EA8
S113255C0AE001006FE0001818886EE8001601000D
S113256C6FE600027A0500003A347A000000001C82
S113257C0AE05D507A00000000200AE05D507A000A
S113258C000000240AE05D507A000000002C0AE0F1
S113259C5D5001006F60000201006DF07A000000D5

S11325AC001C0AE07A01000095F65E003AB40B9722
S10925BC79200001474AEB
S11325C201006F60000A01006DF07A000000002C28
S11325D20AE07A01000096025E003AB40B97792072
S11325E200014726790000096DF001006F600018B1
S11325F201006DF07A00000000280AE07A01000071
S1132602960D5E003AF80B970B875E005B1C5470C5
S11326125E005B3EFD4E0F860F946E6800065C0003
S1132622FEA86EE8000601006F60000201006DF073
S11326327A000000001C0AE07A01000095F65E00B1
S11326423AB40B9779200001587002C201006F60FF
S1132652000A01006DF07A000000002C0AE07A0102
S1132662000096025E003AB40B977920000158707D
S1132672029C790000096DF001006F6000180100EF
S11326826DF07A00000000280AE07A010000960D3E
S10426925EE6
S1132693003AF80B970B87790D00016E680006A8C3
S11326A348587001FCA8595870014AA86358700030
S11326B3A8A864587000A2A86858700198A86F5816
S11326C3700096A8715870022EA872475CA87347CE
S11326D310A87558700082A879587000BA5A002957
S11326E310F8736DF07A000000001C0AE07A010011
S11326F30096185E003A3E0B877A000000002C0A0E
S1132703E0F9735E003CD8F84E6EE800067A0000E9
S11327130000200AE0F94E5E003BC07A000000008F
S1132723240AE0F9635A0028F4F8686DF07A00008C
S113273300001C0AE07A01000096185E003A3E0B83
S113274387F84E6EE800067A00000000200AE0F9DD
S11327534E5E003BC05A0029106868A868461AF801
S11327635968E86DF07A000000001C0AE07A010062
S11327730096185E003A3E0B877A00000000200A99
S1092783E06E6900065E32
S1132789003BC07A00000000200AE00CD95A002857
S1132799F47A040000000E0AE46848A87947126E27
S11127A9480004A879470A7A01000096235AD3
S11327B70028B26868A868461AF85968E86DF07A7D
S11327C7000000001C0AE07A01000096185E003A38
S11327D73E0B877A00000000200AE06E6900065E60
S11327E7003BC07A00000000200AE00CD95A0028F9
S11327F7F47A040000000E0AE46E480003A8794740
S1132807126E480004A879470A7A010000963D5AD8
S11328170028B26868A868461AF85968E86DF07A1C
S1132827000000001C0AE07A01000096185E003AD7
S11328373E0B877A00000000200AE06E6900065EFF
S1132847003BC07A00000000200AE00CD95A002898
S1132857F46E68000EA87947087A010000962340B2
S11328674A6868A868461AF85968E86DF07A00005C
S113287700001C0AE07A01000096185E003A3E0B3E
S1132887877A00000000200AE06E6900065E003BBD
S1132897C07A00000000200AE00CD940506E68009F
S10828A711A879470EA2
S10528AC7A01AC
S11328AE0000963D0DD05E00447640566868A868D9
S11328BE461AF85968E86DF07A000000001C0AE029
S11328CE7A01000096185E003A3E0B877A000000EC
S11328DE00200AE06E6900065E003BC07A0000002D
S11328EE00200AE00CD95E003B3040167A0000004F

S11328FE00200AE06E6900065E003BC00FC05E005A
S10B290E418A5E005B1C54705A
S11395F65361666574792E74787400004D6F746FC9
S1139606722E74787400004C696D69742E747874C4
S113961600005361666574792E747874000A4120DC
S11396264261736B65742069736E2774203173749A
S113963620466C6F6F72000A41204261736B65743A
S11396462069736E277420326E6420466C6F6F72C6
S10496560010
S113291601006DF60F86190069E06FE000026FE0B3
S113292600046FE0000601006FE600087A0000006D
S113293600020AE001006FE0000C7A0000000004C8
S11329460AE001006FE000107A00000000060AE0CA
S113295601006FE0001419006FE000187A00000010
S113296600180AE001006FE0001A19006FE0001E6C
S11329767A000000001E0AE001006FE0002001005B
S11329866D76547001006DF60F865E003A347A0058
S1132996000000040AE05E003A3401006D765470CC
S11329A65E005B3E0F850F947A06000000180AF658
S11329B601006F60001469007920000146520100 8E
S11329C66F60001A6901462801006F66001A7900D4
S11329D6000169E07A00000000040AD0F9735E0082
S11329E63CD87A0100009658790000015E004476CF
S11329F66848A859586001E8F87268C86DF07A010A
S1092A060000965D0FD0F5
S1132A0C5E003A3E0B875A002BE6FB6801006F60B1
S1132A1C001069007920000146 6E01006F60001AF6
S1132A2C1911698101006F6000206901462C0100B6
S1132A3C6F6600207900000169E07A000000000451
S1132A4C0AD0F96F5E003CD87A01000096697900D0
S1132A5C00015E0044765A002BE66848A85958607A
S1132A6C01787A00000000040AD00CB95E003CD84F
S1132A7CF87268C86DF07A010000965D0FD05E00A5
S1132A8C3A3E0B875A002BE601006F600008690180
S1132A9C466E01006F60001A1911698101006F60A5
S1132AAC00206901462C01006F66002079000001AB
S1102ABC69E07A00000000040AD0F94F5EC3
S1132AC9003CD87A010000966E790000015E00444B
S1132AD9765A002BE66848A859586001007A000025
S1132AE90000040AD00CB95E003CD8F87268C86DBE
S1132AF9F07A010000965D0FD05E003A3E0B875ACB
S1132B09002BE601006F60000C6901466C01006F40
S1132B1960001A1911698101006F6000206901467B
S1132B292C01006F6600207900000169E07A00003A
S1132B390000040AD0F96F5E003CD87A01000096C0
S1132B4969790000015E0044765A002BE66848A8BB
S1132B5959586000887A00000000040AD00CB95E55
S1132B69003CD8F87268C86DF07A010000965D0FD1
S1132B79D05E003A3E0B87406401006F60000C6928
S1132B8900792000014656 01006F60001A19116986
S1132B998101006F60002069014628 01006F66000A
S1132BA9207900000169E07A00000000040AD00CD2
S1082BB9B95E003CD8E9
S1052BBE7A0197
S1132BC00000967A790000015E0044766848A859AF
S1132BD04614F87268C86DF07A010000965D0FD054
S1132BE05E003A3E0B875E005B1C547001006DF67D

S1132BF00F865E003A347A00000000040AE05E00AB
S1132C003A3401006D7654705E005B3E0F850F947D
S1132C107A06000000180AF601006F60000C6900D4
S1132C2079200001465201006F60001A69014628AD
S1132C3001006F66001A7900000169E07A00000064
S1132C4000040AD0F9735E003CD87A01000096585C
S1132C50790000015E0044766848A859586001E88D
S1132C60F87268C86DF07A010000965D0FD05E00BF
S1132C703A3E0B875A002E48FB7401006F60000830
S1132C80690079200001466E01006F60001A191176
S1132C90698101006F6000206901462C01006F66A5
S1132CA000207900000169E07A00000000040AD0E6
S1092CB0F9635E003CD84D
S1122CB67A01000096857900000015E0044765A8A
S1132CC5002E486848A859586001787A000000002A
S1132CD5040AD00CB95E003CD8F87268C86DF07A66
S1132CE5010000965D0FD05E003A3E0B875A002E19
S1132CF54801006F6000106901466E01006F6000B6
S1132D051A1911698101006F6000206901462C01C0
S1132D15006F6600207900000169E07A0000000079
S1132D25040AD0F9435E003CD87A010000968B79FA
S1132D350000015E0044765A002E486848A8595899
S1132D456001007A00000000040AD00CB95E003C63
S1132D55D8F87268C86DF07A010000965D0FD05EF1
S1132D65003A3E0B875A002E4801006F6000146934
S1132D7501466C01006F60001A1911698101006F2A
S1132D856000206901462C01006F66002079000070
S1132D950169E07A00000000040AD0F9635E003C93
S1132DA5D87A0100009685790000015E0044765AC1
S1132DB5002E486848A859586000887A000000002A
S1132DC5040AD00CB95E003CD8F87268C86DF07A75
S1132DD5010000965D0FD05E003A3E0B874064010B
S1132DE5006F60001469007920000146560100 6FE9
S1132DF560001A1911698101006F6000206901469D
S1132E052801006F6600207900000169E07A00005F
S1132E150000040AD00CB95E003CD87A0100009684
S1132E2598790000015E0044766848A8594614F86D
S1132E357268C86DF07A010000965D0FD05E003AA6
S1132E453E0B875E005B1C547001006DF60F867A9E
S1132E5500000000040AE001006FE000067A0000AC
S1132E6500000E0AE001006FE0001801006D7654C2
S1132E75705E005B3E0F860F957A000000000E0A18
S1132E85E001006FE000187A000000000A0AE00183
S1132E95006F6100185E003D5E6E680012A87946FA
S1092EA506790000014064
S1132EAB0219006FE0001C7A0400003EBC6F600047
S1132EBB1C6DF001006F51002001006F50000C5D81
S1132ECB400B876E680013A8794606790000014012
S1132EDB0219006FE0001C6DF001006F510020011F
S1132EEB006F5000085D400B876E680014A879468D
S1132EFB0679000001400219006FE0001C6DF00120
S1132F0B006F51002001006F5000105D400B876E66
S1132F1B680015A879460679000001400219006F75
S1132F2BE0001C6F66001C6DF601006F5100200161
S1132F3B006F5000145D400B875E005B1C54705E8A
S1132F4B005B3E0F860F957A000000000E0AE0012E
S1132F5B006FE000187A000000000A0AE001006F1E

S1132F6B6100185E003D5E6E680012A87946067913
S1132F7B000001400219006FE0001C7A0400003EC0
S1132F8BBC6F60001C6DF001006F51002001006FDE
S1082F9B50000C5D4035
S1132FA00B876E680013A879460679000000140027A
S1132FB019006FE0001C6DF001006F51002001004B
S1132FC06F5000085D400B876E680014A8794606B1
S1132FD079000001400219006FE0001C6DF0010050
S1132FE06F51002001006F5000105D400B876E6829
S1132FF00015A879460679000001400219006FE028
S1133000001C6F66001C6DF601006F51002001006B
S11230106F5000145D400B875E005B1C54705EB5
S113301F005B3E0F860F957A000000000E0AE00159
S113302F006FE000187A000000000A0AE001006F49
S113303F6100185E003D5E6E680012A8794606793E
S113304F000001400219006FE0001C7A0400003EEB
S113305FBC6F60001C6DF001006F51002001006F09
S113306F50000C5D400B876E680013A879460679F4
S113307F000001400219006FE0001C6DF001006FAA
S113308F51002001006F5000085D400B876E6800F0
S113309F14A879460679000001400219006FE00079
S11330AF1C6DF001006F51002001006F5000105D87
S11330BF400B876E680015A879460679000001401A
S11330CF0219006FE0001C6F66001C6DF601006FA4
S11330DF51002001006F5000145D400B875E005BB1
S11330EF1C54705E005B3E0F860F9501006F6000EE
S11330FF14690146360FE07A37000000240FF17A86
S109310F02000000245E33
S1133115005AE67A000000003C0AF00FD15C00F883
S1133125807A17000000247A00000000200AF00FBF
S1133135E15C00FE105E005B1C54705E005B3E0F9D
S1133145860F9501006F60000C690146360FE07A22
S113315537000000240FF17A02000000245E005AB4
S1133165E67A000000003C0AF00FD15C00FA947A7D
S113317517000000247A00000000200AF00FE15C2C
S10C318500FE965E005B1C547011
S113965853544F50005361666574792E74787400BF
S1139668004F50454E004F50454E2053706565647A
S113967879004F50454E20537461727400434C4F28
S1139688534500434C4F5345205370656564790037
S10F9698434C4F5345205374617274001F
S113318E01006DF60F86190069E06FE000026FE033
S113319E00046FE0000601006FE600087A000000ED
S11331AE00020AE001006FE0000C7A000000000448
S11331BE0AE001006FE000107A00000000060AE04A
S11331CE01006FE0001419006FE000187A00000090
S11331DE00180AE001006FE0001A19006FE0001EEC
S11331EE7A000000001E0AE001006FE000200100DB
S11331FE6D76547001006DF60F865E003A347A00D8
S113320E000000040AE05E003A3401006D7654704B
S113321E5E005B3E0F850F947A06000000180AF6D7
S113322E01006F60001469007920000146520100 0D
S113323E6F60001A6901462801006F66001A790053
S113324E000169E07A00000000040AD0F9735E0001
S113325E3CD87A01000096A4790000015E00447602
S113326E6848A859586001E8F87268C86DF07A0189
S109327E000096A90FD029

S11332845E003A3E0B875A00345EFB6A01006F60AE
S1133294001069007920000 1466E01006F60001A76
S11332A41911698101006F6000206901462C010036
S11332B46F6600207900000169E07A0000000004D1
S11332C40AD0F9755E003CD87A01000096B57900FE
S11332D400015E0044765A00345E6848A859586079
S11332E401787A00000000040AD00CB95E003CD8CF
S11332F4F87268C86DF07A01000096A90FD05E00D9
S11333043A3E0B875A00345E01006F60000869017E
S1133314466E01006F60001A1911698101006F6024
S113332400206901462C01006F660020790000012A
S110333469E07A00000000040AD0F9555E3C
S1133341003CD87A01000096B8790000015E004480
S1133351765A00345E6848A859586001007A000023
S11333610000040AD00CB95E003CD8F87268C86D3D
S1133371F07A01000096A90FD05E003A3E0B875AFE
S113338100345E01006F60000C6901466C01006F3F
S113339160001A1911698101006F600020690146FB
S11333A12C01006F6600207900000169E07A0000BA
S11333B10000040AD0F9755E003CD87A010000963A
S11333C1B5790000015E0044765A00345E6848A86E
S11333D159586000887A00000000040AD00CB95ED5
S11333E1003CD8F87268C86DF07A01000096A90F05
S11333F1D05E003A3E0B87406401006F60000C69A8
S1133401007920000 1465601006F60001A19116905
S11334118101006F60002069014628 01006F660089
S1133421207900000169E07A00000000040AD00C51
S1083431B95E003CD868
S10534367A0116
S1133438000096C2790000015E0044766848A859E6
S11334484614F87268C86DF07A01000096A90FD087
S11334585E003A3E0B875E005B1C547001006DF6FC
S11334680F865E003A347A00000000040AE05E002A
S11334783A3401006D7654705E005B3E0F850F94FD
S11334887A06000000180AF601006F60000C690054
S1133498792000 01465201006F60001A690146282D
S11334A801006F66001A7900000169E07A000000E4
S11334B800040AD0F9735E003CD87A01000096A490
S11334C8790000015E0044766848A859586001E80D
S11334D8F87268C86DF07A01000096A90FD05E00F3
S11334E83A3E0B875A0036C0FB6B01006F60000839
S11334F8690079200001466E01006F60001A1911F6
S1133508698101006F6000206901462C01006F6624
S113351800207900000169E07A00000000040AD065
S1093528F9645E003CD8CB
S112352E7A01000096CB790000015E0044765AC3
S113353D0036C06848A859586001787A0000000029
S113354D040AD00CB95E003CD8F87268C86DF07AE5
S113355D01000096A90FD05E003A3E0B875A003644
S113356DC001006F6000106901466E01006F6000BD
S113357D1A1911698101006F6000206901462C0140
S113358D006F6600207900000169E07A00000000F9
S113359D040AD0F9445E003CD87A01000096D07934
S11335AD0000015E0044765A0036C06848A8595899
S11335BD6001007A00000000040AD00CB95E003CE3
S11335CDD8F87268C86DF07A01000096A90FD05E25
S11335DD003A3E0B875A0036C001006F6000146934

S11335ED01466C01006F60001A1911698101006FAA
S11335FD6000206901462C01006F660020790000F0
S113360D0169E07A00000000040AD0F9645E003C11
S113361DD87A01000096CB790000015E0044765AFA
S113362D0036C06848A859586000887A0000000029
S113363D040AD00CB95E003CD8F87268C86DF07AF4
S113364D01000096A90FD05E003A3E0B874064013E
S113365D006F60001469007920000146560 1006F68
S113366D60001A1911698101006F6000206901461C
S113367D2801006F6600207900000169E07A0000DF
S113368D0000040AD00CB95E003CD87A0100009604
S113369DDC790000015E0044766848A8594614F8A9
S11336AD7268C86DF07A01000096A90FD05E003ADA
S11336BD3E0B875E005B1C547001006DF60F867A1E
S11336CD00000000040AE001006FE0000E01006D30
S11336DD7654705E005B3E0F860F957A00000000F6
S11336ED040AE001006FE0000E01006F61000E0F90
S11336FDE05E003D5E6E680004A879460679000021
S113370D01400219006FE000127A0400003EBC6F05
S108371D6000126DF0D5
S113372201006F51002001006F50000C5D400B87B8
S11337326E680005A879460679000001400219006 7
S11337426FE000126DF001006F51002001006F5015
S113375200085D400B876E680006A879460679006B
S1133762000 1400219006FE000126DF001006F5179
S1133772002001006F5000105D400B876E68000748
S1133782A879460679000001400219006FE0001291
S11337926F6600126DF601006F51002001006F5039
S11337A200145D400B875E005B1C54705E005B3E41
S11337B20F860F957A00000000040AE001006FE013
S11337C2000E01006F61000E0FE05E003D5E6E6849
S11337D20004A879460679000001400219006FE04F
S11337E200127A0400003EBC6F6000126DF001000B
S11337F26F51002001006F50000C5D400B876E6813
S11338020005A879460679000001400219006FE01D
S107381200126DF040
S113381601006F51002001006F5000085D400B87C7
S11338266E680006A879460679000001400219007 1
S11338366FE000126DF001006F51002001006F5020
S113384600105D400B876E680007A879460679006D
S1133856000 1400219006FE000126F6600126DF658
S113386601006F51002001006F5000145D400B876B
S10A38765E005B1C54705E51
S113387D005B3E0F860F957A00000000040AE001FD
S113388D006FE0000E01006F61000E0FE05E003D62
S113389D5E6E680004A879460679000001400219 9E
S11338AD006FE000127A0400003EBC6F6000126DE1
S11338BDF001006F51002001006F50000C5D400BB3
S11338CD876E680005A879460679000001400219 44
S11338DD006FE000126DF001006F51002001006FC9
S11338ED5000085D400B876E680006A8794606797F
S11338FD000001400219006FE000126DF001006F2E
S113390D51002001006F5000105D400B876E680061
S113391D07A879460679000001400219006FE000FF
S113392D126F6600126DF601006F51002001006FDA
S113393D5000145D400B875E005B1C54705E005B92
S113394D3E0F860F9501006F600014690146360F17

S113395DE07A37000000240FF17A02000000245EA4
S108396D005AE67A0098
S11339720000003C0AF00FD15C00F8A07A170000A7
S113398200247A00000000200AF00FE15C00FE1C14
S11339925E005B1C54705E005B3E0F860F95010058
S11339A26F60000C690146360FE07A37000000248D
S11339B20FF17A02000000245E005AE67A0000004A
S11339C2003C0AF00FD15C00FAB47A17000000241D
S11339D27A00000000200AF00FE15C00FE9C5E000A
S10739E25B1C5470A3
S11396A453544F50005361666574792E7478740073
S11396B400555000555020537065656479005550 2A
S11396C420537461727400444F574E00444F574EF5
S11396D4205370656564790044F574E2053746179
S10696E47274009A
S11339E601006DF60F867A0000FFE1D4010069E05D
S11339F67A0600FFE1D4F87268E87A000000000650
S1133A060AE001006FE000027A01000096E8010077
S1133A166F6000025E005468F8736EE8000FF84E9C
S1133A266EE800106EE8001101006D7654700F8188
S1133A361A800100699054705E005B3E0F946E7DA0
S1133A4600197A0000FFE1D46849A9434716A94D36
S1133A56470CA9504714A9534614688D40106E8D20
S1133A66000F400A6E8D001040046E8D0011190080
S1133A765E005B1C547001006DF60F966869A94CD5
S1133A8646247A0600FFE1D47A00000000060AE025
S1133A9601006FE0000201006F71000801006F6012
S1133AA600025E005468190001006D7654705E00D2
S1133AB65B3E0F9401006F7500187A0000FFE1D496
S1133AC66849A9434716A94D470CA9504716A95358
S1093AD646186808400ACF
S1133ADC6E08000F40046E08001068D840066E088C
S1133AEC001168D819005E005B1C547001006DF660
S1133AFC0F966869A94C46247A0600FFE1D47A0034
S1133B0C000000060AE001006FE0000201006F6193
S1133B1C000201006F7000085E0054681900010078
S1133B2C6D7654705E005B3E0F850C9E6DF67A01CC
S1133B3C000096F20FD05C00FEF80B870C00461CBD
S1133B4CA8004714A80146147A01000097057900D0
S1133B5C00015E0044764004196640047906000 1B6
S10C3B6C0D605E005B1C54705EE9
S1133B75005B3E0F850F9601006DF67A01000097F5
S1133B85140FD05C00FF280B970C004624A80147AF
S1133B9506A8024716401A7A01000097217900000A
S1133BA5015E0044767906000140087906000 2406B
S1133BB50219660D605E005B1C54705E005B3E0F70
S1133BC5850C9E6DF67A01000097140FD05C00FEFC
S1133BD5680B870C00461CA8004714A80146147AF5
S1133BE501000097057900000 15E00447640041941
S1133BF5664004790600010D605E005B1C54705E2F
S1133C05005B3E0F850C9E6DF67A01000097300F21
S1133C15D05C00FE240B870C00461CA8004714A8A3
S1133C250146147A01000097057900000 15E0044FE
S1133C3576400419664004790600010D605E005B59
S1133C451C54705E005B3E0F850F9601006DF67A7E
S1043C55016A
S1133C56000097440FD05C00FE540B970C004624DB

S1133C66A8014706A8024716401A7A0100009721C1
S1133C76790000015E0044767906000014008790662
S1133C860002400219660D605E005B1C54705E0004
S1133C965B3E0F850C9E6DF67A01000097440FD0AC
S1133CA65C00FD940B870C00461CA8004714A80172
S1133CB646147A0100009705790000015E004476F8
S1133CC64004196640047906000010D605E005B1C22
S1133CD654705E005B3E0F850C9E6DF67A01000004
S1133CE697520FD05C00FD500B870C004616A800B8
S1133CF64712A801460E7A010000970579000001D4
S1133D065E0044765E005B1C54705E005B3E1B8760
S1133D160F8518886EF800017A06000000010AF67E
S1133D2601006DF67A01000097520FD05C00FD7E0C
S1133D360B970C004618A800470EA8014710A802C7
S1093D46470C400A40088F
S1103D4C40066E780001400218880B875E68
S1133D59005B1C54705E005B3E0F850F9679000073
S1133D69096DF001006DF67A010000975D0FD05CD3
S1133D7900FD7C0B970B870C004618A8014706A882
S1133D89024710400E7A0100009721790000015E75
S10C3D990044765E005B1C5470CB
S11396E879796E6E79796E6E00005065726D697462
S11396F8436F6D6D616E642E74787400000A57723F
S11397086974696E67204572726F7200436F6D6D7D
S1139718616E642E74787400000A52656164696E20
S113972867204572726F72005065726D6974547563
S1139738726E4F70656E2E747874000054757 26E75
S11397484F70656E2E74787400004D6F746F722EAF
S113975874787400004C696D69742E747874000011
S1133DA25E005B3E0F860F957A00000097680FE175
S1133DB25E005AC801006FE600080FE055267A003C
S1133DC20000000C0AE001006FE0000E19006FE032
S1133DD2000C19006FE0001201006FE500145E0091
S1133DE25B1C54705E005B3E0F8601006FE60008A9
S1133DF20FE201006DF25E003EFA0B975E005B1C60
S1133E0254705E005B3E7A37000000100F8601009B
S1133E126FE600087A02000000080AF201006DF260
S1133E225E003EFA0B9701006F6600080FA10FE2D6
S1133E320FF05E00556A5E0059DC7A17000000102D
S1133E425E005B1C54705E005B3E0F850D16790E9F
S1133E5203E852E617F60FE101006F5000145E000B
S1133E623F9E5E005B1C547001006DF60F867A0064
S1133E720000000C0AE001006FE0000E792100014E
S1133E82460A790000016FE0000C400A0D11460654
S1093E9219006FE0000CB3
S1133E9801006D76547001006DF60F867A000000FC
S1133EA8000C0AE001006FE0000E6F60000C0100D7
S1133EB86D7654705E005B3E0F860F9569606F7177
S1133EC800181D10470A190069D06F70001869E0BF
S1133ED85E005B1C54705E005B3E0F850D1617F683
S1133EE80FE101006F5000145E003F9E5E005B1CF3
S1053EF8547001
S10B9768000000000000000F6
S1133EFA7A0000FFE1E601006F7100045E005AC810
S1133F0A54705E005B3E7A37000000200F867A0207
S1133F1A000000180AF201006DF255D40B970FE165
S1133F2A0FF05E005A820F817A02000097807A00AE

S1133F3A000000100AF05E0058447A0000000018DE
S1133F4A0AF07A01000000080AF15E005AC840101C
S1133F5A7A02000000080AF201006DF255920B97EB
S1133F6A7A01000000180AF17A02000000100AF22E
S1133F7A0FF05E00559A0F817A00000000080AF0DC
S1133F8A5E005A720D0046C87A17000000205E00D0
S1133F9A5B1C54705E005B3E1B971B970F860F9545
S1133FAA7A02000000020AE201006DF25C00FF409F
S1133FBA0B970FD10FF05E005A820F817A0200002D
S1133FCA97807A000000000A0AE05E0058440B97C3
S1133FDA0B975E005B1C54705E005B3E0F860F9569
S1073FEA0FE055B0DC
S1133FEE01006F6000120B7001006FE000125E00A3
S1133FFE5B1C54705E005B3E0D057A0400FFE1EE20
S113400E400601006F44001A01006F40001A0100C0
S111401E6F01001A46EC79250001460A7A0666
S113402C00FFE22A5A00411079250002460A7A065B
S113403C00FFE2485A0041107925000B460A7A0624
S113404C00FFE2665A0041107925000C460A7A06F5
S113405C00FFE2845A0041107925000D460A7A06C6
S113406C00FFE2A25A0041107925000E460A7A0697
S113407C00FFE2C05A00411079250013460A7A0664
S108408C00FFE2DE5A13
S1134091004110792500144608 7A0600FFE2FC402E
S11340A16E7925001546087A0600FFE31A40607908
S11340B125001646087A0600FFE3384052792500A9
S11340C11746087A0600FFE35640447925001E4649
S11340D1087A0600FFE37440367925001F46087A03
S11340E10600FFE3924028792500294 6087A060055
S11340F1FFE3B0401A7925002A46087A0600FFE358
S1134101CE400C7925002B46067A0600FFE3EC0F1F
S1134111E6461C7A010000977019005E0044767A26
S113412101000097721 9005E0044761A8040540121
S1134131006FE4001601006F40001A01006FE000F8
S11241411A01006F86001601006FC6001A7A007C
S1134150000097887A01000000020AE15E005AC855
S11341607A00000097907A010000000A0AE15E00DD
S11341705AC869E51A8001006FE000120FE05E0083
S11341800E6C0FE05E005B1C547001006DF60F8631
S11341905E0010DA01006F60001601006F61001A03
S11341A001006F81001A01006F60001A01006F6146
S11341B0001601006F81001601006D7654705E00D9
S11341C05B3E0F867A0200FFE1E601006DF25C00C0
S11341D0FD280B977A0000FFE1E67A010000000258
S11341E00AE15E005AC85E005B1C547001006DF664
S11341F00F867A00000097987A01000000020AE116
S11342005E005AC801006D76547001006B2000FFF8
S1134210E20801006F01001A460419005470790086
S1134220000154700 1006B2100FFE20801006F11CF
S1134230001A1988400C01006F11001A0D800B50F1
S10942400D080F9146F08A
S10F42460D8054706DF50D0501006B2018
S113425200FFE20801006F01001A472001006B21F1
S113426200FFE20869101D5046040F904010010040
S11342726F11001A01006F10001A46E81A806D755B
S113428254705E005B3E0D0555BE0F86460E0D5003
S11342925C00FD6C0F865C00FF22401C7A0000006C

S11342A200020AE07A01000097985E005A560D0058
S11342B247060FE05C00FF040FE05E005B1C5470D6
S11342C25E005B3E0D0555800F86460E0D505C0069
S11342D2FD2E0F865C00FEE440247A0000000002FB
S11342E20AE07A01000097985E005A560D004708CB
S11342F20FE05C00FEC640060FE05C00FE8A5E0033
S11343025B1C54705E005B3E1B971B977A0500FF94
S1134312E1EE01006F56001A407201006F65001A48
S11343227A00000000020AE07A01000097885E002A
S11343325B0E0D0047547A00000000020AE07A0186
S1084342000097985EE6
S1134347005B0E0D00473E7A01000000020AE17A86
S1134357020000000A0AE20FF05E00559A0F817A05
S11343670000FFE1E65E005A620D0047187A00007D
S1134377FFE1E67A01000000020AE15E005AC80F76
S1134387E05E0005500FD601006F60001A46860BEA
S1134397970B975E005B1C54701A8001006BA0009B
S11343A7FFE2047A0000FFE20C01006BA000FFE2CA
S11343B7087A0000FFE1EE01006BA000FFE2221A7A
S11343C78001006BA000FFE2265470F801386018E3
S11343D7883861386218838633890F8FF3891183D
S11343E7883864F8883865F804386679009E576B0F
S11343F780FF6819006B80FF6A19006B80FF6C549C
S11344077001006DF27F67722079009E576B80FF02
S10C4417687A0100FFE1E67A0274
S1134420000097A00F905E00559A01006D725470C2
S11344305A0043067A00000097907A0100FFE1E6F4
S11144405E005AC8558C5E0002A85A0043A0C5
S11397700A0063616C6C6F63206661696C656400E9
S1139780408F400000000000BFF000000000000018
S113979000000000000000C00000000000000006
S10B97A03F50624DD2F1A9FC18
S113444E7A00000097A801006DF07A0000FFE40ADD
S113445E5E0053E00B977A0000FFE40A01006DF053
S113446E5E0047F80B9754705E005B3E7A0500FFC3
S113447EE44A0D040F960C4458600114AC004768CF
S113448EAC014710AC0258700106AC03587000C261
S113449E5A00459E55AA7A01000097AA0FE05E00C6
S11344AE543E0D00461E7A00000097B001006DF0D9
S11344BE7A00000097AD01006DF00FE05E0053E04F
S11344CE0B970B9701006DF67A00000097AD010074
S11344DE6DF00FD05E0053E00B970B9701006DF557
S10D44EE7A00000097AD40507A01F8
S11344F8000097AA0FE05E00543E0D00461E7A00A6
S1134508000097B001006DF07A00000097AD01003C
S11345186DF00FE05E0053E00B970B9701006DF60B
S11345287A00000097AD01006DF00FD05E0053E0F4
S11345380B970B9701006DF57A00000097AD01000A
S11345486DF05E0045F40B970B9701006DF65E0066
S113455847F80B974040403E01006DF67A00000093
S113456897AD01006DF00FD05E0053E00B970B97EA
S113457801006DF57A00000097AD01006DF05E0053
S113458845F40B970B970FD05E0054840B500D0125
S10A45980FD05E00525C5ED0
S108459F005B1C5470D9
S10E97A80C000A00002573000A0C00EF
S11345A4188838BA38B8F85038B919000B5079203C

S11345B401184DF8F83038BA28BCF88038BC547068
S11345C40C8820BC737047FA38BBE07F30BC54704E
S11345D420BC736047FA29BDE0BF30BC0C9854700B
S11345E47EBC736047067900000154701900547 04F
S11345F45E005B3E7A0600FFE48A7A05000000044D
S11346047A00000000180AF00AD07308470E7A00F3
S1134614000000180AF00AD00B70400A7A00000068
S113462400180AF00AD00F85189968E901006DF0A3
S113463401006F71001C0FE05E0054A00B97195525
S11346440D5017F00AE0680947220D5017F00AE0ED
S11346546808A80A4606F80D5C00FF640D5017F0BD
S11346640AE068085C00FF580B5540D45E005B1CED
S105467454707D
S11346761911400C19880B58792806824DF80B51ED
S11346861D014DF0547001006DF50D090D8D28D6F1
S113469617500D010D99470C0D1979690060794979
S11346A6001040060D19796900600DD50D90148D23
S11346B60CD8C88038D63DD639D67900000455B013
S11346C601006D7554706DF56DF40D090D8528D6D1
S11346D617500D010D99470C0D1979690060794939
S11346E6001040060D19796900600D541194119458
S11346F611941194EC0F0D90148CED0F148D0CC8BE
S1134706C88038D63CD60CD8C88038D63DD639D6DC
S1134716790000045C00FF586D746D7554700100D8
S11347266DF619EE7900000F5C00FF4419667908EF
S113473600030DE05C00FF4E0B56792600034DEE99
S1134746790800020DE05C00FF3C19667908002831
S11347560D605C00FF70790800100D605C00FF6659
S10947667908000E0D604E
S113476C5C00FF5C790800060D605C00FF52790861
S113477C00010D605C00FF48790800020D605C00CD
S113478CFF3E01006D7654707908000119005C003E
S113479CFF2E7908000219005A0046CC6DF60C8ED8
S11347ACAE0C460455E2401CAE0A460455DA4014DE
S11347BCAE0D460455D2400C17D60D687900000196
S11347CC5C00FEFC6D76547001006DF60D0E0D86CB
S11347DC0D6079080040528009E0791000800D08C3
S11347EC19005C00FEDA01006D7654705E005B3ECE
S11347FC790C000119447A0600FFE4DA7A0500000B
S113480C00047A00000000180AF00AD07308470E5F
S113481C7A00000000180AF00AD00B70400A7A00E4
S113482C000000180AF00AD00F85189968E90100F6
S113483C6DF001006F71001C0FE05E0054A00B972C
S113484C19550D5017F00AE0680947560D5017F02B
S109485C0AE06808A80A47
S1134862460A0DC80D405C00FF68403C0D5017F02E
S11348720AE06808A80D460C790800020D405C00A6
S1134882FE4840240D5017F00AE06808A80C4606BB
S11348925C00FEFE40120D5017F00AE0680817D0C4
S11348A20D080DC05C00FE220B5540A05E005B1C90
S10548B254703D
S11348B41911400C19880B58792806824DF80B51AD
S11348C41D014DF054705C0009145C0000B45504E0
S11348D45A0049AE01006DF618886AA800FFE5DDA9
S11348E46AA800FFE5DF18886AA800FFE5DE6AA866
S11348F400FFE51B79080080790000045C000808C8
S1134904790800107900000205C0007FC7906000F89

S1134914790800100D605C0007EE0D687900000B48
S11349245C0007E40D687900000D5C0007DA790880
S1134934005079000095C0007CE790800D679009D
S113494400075C0007C219660D605C000710790E4E
S113495400010DE05C0006E60DE05C0007000D6855
S11349647900002B5C0007A00D687900002F5C0020
S113497407960DE8790000275C00078C01006D762B
S1134984547079080005190O5C00077C7900006401
S11349945C00FF1C790800C419005C00076A7908ED
S10949A40003790000018D
S11349AA5A00510C7908000279000055C0007528D
S11349BA790800CC19005A00510C5E005B3E7A0458
S11349CA000098667A000000983501006DF05E00D9
S11349DA45F40B9719EE19660DE5790D001052D5BA
S11349EA0D6009505C00070C17506DF001006DF45F
S11349FA5E0045F40B970B870B56792600104DE0A2
S1134A0A7A000000986C01006DF05E0045F40B9784
S1134A1A0B5E792E00044DBE5E005B1C54700100D0
S1074A2A6DF67A06A2
S1134A2E00FFE51A790000065C0006C468E87C60A6
S1134A3E734047367900000E5C0006B40C8E734843
S1134A4E47045C0002C8735E47085C0002C25A004A
S1134A5E4B1C730E47085C0002B85A004B1C731EA6
S1134A6E47045C0002AE5A004B1C7C607360472205
S1134A7E7900000C5C0006780C8E730847085C0006
S1134A8E00925A004B1C731E587000825C0001C6C4
S1134A9E407C7C607320471E7900000A5C00065040
S1134AAE0C8E730847065C00020A4062731E475E53
S1134ABE5C00023E40587C60731047527900000838
S1134ACE5C00062C0C8E7368470A5C00FDFC5C00D0
S1134ADEFECE403A734E471A790800C0790000099A
S1134AEE5C00061A7908000379000055C00060EC7
S1134AFE401C737E4718790800050790000095C004A
S1134B0E05FC7908000279000055C0005F0010040
S1084B1E6D7654705E8A
S1134B23005B3E19DD790000265C0005CE0C8E7315
S1134B3368587001047A0600FFE51D790000086DCB
S1134B43F00FE179080026790000255C0005CC0B02
S1134B53870DD05C0004E80DD05C000502790500E5
S1134B63200D505C0004C06868E8605860009C6EC8
S1134B73680001473CA801471EA8034772A80547DD
S1134B833EA8064726A8084716A809475CA80A476C
S1134B9326A80B4760406C5C0001865A004C2A6AC6
S1134BA32800FFE51C17500D08400E5C000224404B
S1134BB3765C00035440700DD879000021402479BA
S1134BC30800400D505C0005406E6E0002CE806A03
S1134BD3AE00FFE51B6A2800FFE51B17500D08799C
S1134BE30000045C000522403E5C00038E40385CF9
S1134BF300018840326E680002472C0D505C0004AC
S1134C030E40240D505C000406401C6868E860A84D
S1084C132046080D50CE
S1134C185C0003F6400C686EEE60AE400D505C001D
S1134C2803E87A0000FFE5DE7D0070000DD05C002C
S1134C38045440226A2800FFE5DF470818886AA859
S1134C4800FFE5DF0DD05C0004107908000179004E
S10A4C5800275C0004AE5EBF
S1134C5F005B1C54705E005B3E7900002E5C000409

S1134C6F8E0C8EE8C017504626790000406DF07AFF
S1134C7F0500FFE55D0FD17908002E7900002D5C4B
S1134C8F00048C0B870D060D010FD05C00056279B4
S1134C9F0000015C00039C790800017900002F5C80
S1134CAF00045A5E005B1C5470790000365C0004EC
S1134CBF3E54706DF6790000225C0004320C8E7343
S1134CCF68472A7358472619005C0003866A280031
S1134CDFFFE5DF470C5C0001C819005C0003A0402F
S1134CEF0C79080001790000275C0004106D7654DD
S1134CFF706DF67900002A5C0003F40C8E7368471D
S1134D0F08735847045C0004826D765470547054D2
S1134D1F70547054705E005B3E7A0600FFE51D68A9
S1134D2F68E803A80246426E680003E80747186E57
S1134D3F690003E9071751177178106A2800009764
S1084D4FB417505C00E5
S1134D5402D46E680003E8071750790100011A08AA
S1134D644B04101140F817117A0000FFE5DD680ABF
S1134D74169A688A5E005B1C54705E005B3E7A067A
S1134D8400FFE51D6868E803A80246406E68000357
S1134D94E80747186E690003E9071751177178107C
S1134DA46A28000097B417505C0002626E6800031F
S1134DB4E8071750790100011A084B04101140F851
S1134DC47A0000FFE5DD680A149A688A5E005B1CBA
S1114DD454707A0000FFE5DE7D0072006A284D
S1134DE200FFE520A801470AA8024716A8034722A5
S1134DF254706A29000097BC17517A00000097BCCF
S1134E0240686A29000097D017517A00000097CEB4
S1134E1240586A2800FFE51F470EA801471AA80257
S1134E224726A803473254706A29000097F517D121
S1134E327A00000097F540326A29000097F917D1EA
S1134E427A00000097F940226A290000980B17D1D3
S10D4E527A000000980B40126A2951
S1134E5C0000981317D17A00000098135502547070
S1134E6C5E005B3E0F840D187A0600FFE5E06A28AE
S1134E7C00FFE52417500C8018886A2900FFE523EE
S1134E8C1751091069E01D804F0269E801006BA4FA
S1134E9C00FFE5E2F8016AA800FFE5DF55065E00B6
S1134EAC5B1C54705E005B3E7A0600FFE5E07905FF
S1134EBC000869604744696079200008 4E026965FF
S1134ECC7A0400FFE5E26DF50100694179080022DF
S1134EDC790000215C0002780B870D0569611901CB
S1134EEC69E117F0010069410A81010069C1696038
S1134EFC460818886AA800FFE5DF5E005B1C547047
S1044F0C5E43
S1134F0D005B3E6A2800FFE51DE80317500D0519E8
S1134F1DEE790600210D004706792500014610 0F95
S1134F2DE05C0001DA0DE80D605C0001D2403C79D4
S1134F3D250002462E6A2800FFE520E80717500DCD
S1134F4D05790100011A084B04101140F86A280075
S1134F5DFFE5DD17506601470679080000140060D90
S1134F6DE840020DE80D605C0001945E005B1C548B
S1134F7D706DF66A2800FFE51F6AA800FFE51C4661
S1134F8D3C19660D68790000285C0001720D687983
S1134F9D00002C5C0001680D68790000305C000195
S1134FAD5E0D68790000345C0001540D68790000D2
S1134FBD385C00014A0D687900003C404018886A4E
S1134FCDA800FFE5DD790600010D605C0000867920

S1134FDD080011790000285C000124790800037989
S1134FED00002B5C0001180D60554A7908001279F9
S1084FFD00002C5C0024
S113500201080D687900002F5C0000FE6D76547074
S11350126DF60D065C0000E4C88017500D080D60A4
S11350225C0000E66D7654706DF60D065C0000CCF4
S1135032E87F17500D080D605C0000CE6D7654704A
S113504201006DF60D0617F6103679080008786030
S11350526B2000FFE0085C0000B001006D76547025
S11350625E005B3E0D0617F610367A0500FFE00080
S11350720AE569505C00008417500D06703E0D6806
S113508269505C0000845E005B1C54705E005B3EF2
S11350921B971B977A0500FFE5DE0D0EFE017900D3
S11350A200010DE11A094B04101040F86859175119
S11350B266104704702E4002722E701E17560DE0C2
S11350C217F0103001006FF000040D6801006F71DA
S11350D2000478106B2000FFE000010069F1552AFB
S11350E2790000011B5E4B04101040F868591589C2
S10950F268D90B970B9730
S10450F85E56
S11350F9005B1C54706AA8004000036A2800400042
S11351090154700D016AA9004000030D806AA800CB
S113511940000154705E005B3E7A03004000030FB8
S1135129840F9568BC19EE196640180DC055C6E879
S11351390F471868BC6A280040000168D80B750B33
S11351495E0B566F7000181D064DE00DE05E005BA7
S11351591C54705E005B3E0D0C0D830F956F74003C
S11351691819EE1966401E0D30558AE81F471A0DA6
S1135179C06AA80040000368586AA8004000010BF0
S1135189750B5E0B561D464DDE0DE05E005B1C5430
S11351997001006DF619EE7A0000FFE59D790100B3
S11351A9405C0001140D06790000015C00FEAA0DA4
S11351B96647166DF67A0100FFE59D7908002A799D
S11351C9000029558E0B870D0E790000015C00FE46
S11351D9B40DE001006D76547019006BA000FFE572
S10651E9E86BA0CD
S11351EC00FFE5E619006BA000FFE5EC6BA000FFE8
S11351FCE5EA54705E005B3E7A0400FFE5E87A054D
S113520C00FFE5E60F830D1E7FF5725019994030B0
S113521C6940792001004C2C695017F068397800EB
S113522C6AA900FFE5EE69500B5017F079010100F4
S113523C01D0531069D869400B5069C00B730B59DB
S10E524C1DE94DCC7FF570500D905E06
S1135257005B1C54705E005B3E790E01007A04000C
S1135267FFE5EC7A0500FFE5EA0F830D127FF57280
S11352775019994034694079200 1004C326950171D
S1135287F001D053E00D8017F0683978006AA90060
S1135297FFE6EE69500B5017F001D053E069D86968
S11352A7400B5069C00B730B590D201D094DC67F69
S11352B7F570500D905E005B1C54705E005B3E7A88
S11352C70500FFE5EC0F840D1E7FF572501966404C
S11352D7381DE64C386B2000FFE5EA695119107950
S11352E710010017F07901010001D053100D801749
S11352F7F00D6117F10AC178006A2800FFE6EE682E
S11353079869501B5069D00B5669504EC47FF5708E
S1135317500D605E005B1C54705E005B3E7A0500B7
S1135327FFE5E80F840D1E7FF5725019664038 1D9F

S1135337E64C386B2000FFE5E66951191079100137
S10853470017F07901DD
S113534C010001D053100D8017F00D6117F10AC144
S113535C78006A2800FFE5EE689869501B5069D005
S110536C0B5669504EC47FF570500D605E06
S1135379005B1C54706B2000FFE5E817F07901010D
S11353890001D053100D8054706B2000FFE5EC171A
S1105399F07901010001D053100D80547014
S11397B420282C3034383C2012010001000000081A
S11397C4FEFF10000100010102010902270001014B
S11397D400C0640904000003000000307058102BC
S11397E44000FF070502024000FF07058302400013
S11397F4FF040309041203550053004200200054DC
S1139804004500530054000800331002E00300022A9
S1139814035500530042002000540045005300054F4
S113982400200050005200 4F004700520041004DF9
S10498340030
S11391C00023002B0033003B0027002F0037003F14
S1139835303020303120303220303320303420306 6
S1139845352030362030372030382030392030412C
S1139855203042203043203044203045203046 0A12
S10C986500253032582000 0A00EE
S11353A601006DF67A0600FFE82A010069607A01BA
S11353B641C64E6D5E0079F67A1000003039010061
S11353C669E06960198817707A01000080005E0041
S10D53D679D00D1001006D765470BC
S11353E05E005B3E0F857A06000000047A00000031
S11353F000180AF00AE07308470E7A00000000184C
S11354000AF00AE00B70400A7A00000000180AF064
S11354100AE00F8601006DF001006F70001C0100AF
S11354206DF001006DF51A91790000015E005B6675
S11154307A170000000C0D065E005B1C547022
S113543E5E005B3E0F860F9540040B760B75686915
S113544E68581C8946040C9946F068681750685DC5
S10D545E175519505E005B1C5470D3
S11354685E005B3E0F840FC60F950FE00B766C59F9
S10F5478688946F60FC05E005B1C547090
S11354845E005B3E0F851AE640020B760FD00B7568
S10F5494680946F60FE05E005B1C5470D4
S11354A05E005B3E0F850F9601006F7100180100CF
S11354B06DF101006DF601006DF51A91790000019F
S11354C05E005B667A170000000C5E005B1C547084
S11354D00FC446120FD5460E792107FF46120FB3AC
S11354E04604156E4A0A1AC47A05000000081866B5
S11354F05A0057640D9946120FC4461A0FD5461623
S11355000FB3461E16E6580002660FA246360FB3C7
S1135510463258000 25A0FA246140FB346105800E1
S1135520024E0CE60D190FA40FB55A005770103533
S11355301234792C00104C0C1B5910351234792C71
S113554000104DF410331232792A00104C0C1B5109
S1135550103312327 92A00104DF4580000CA1AC4CD
S11355601AD5186619995800020601006DF2010058
S11355706DF301006F23000401006DF30100692343
S1135580795B800001006DF30FF2550E0B970B97BB
S113559001006D7301006D7254706DF601006DF5BD
S11355A001006DF401006DF301006DF201006DF176
S11355B001006DF0010069140100692510341035F4

S10955C01FD44218451040
S11355C601006F14000401006F2500041FD4440674
S11355D60F940FA10FC26816682E0FA0010069145D
S11355E601006F150004010069020100 6F03000446
S11355F6796C000F796A000F6919111911191119BC
S11356061119796907FF690111111111111111118D
S1135616796107FF792907FF5870FEAE0D1158709F
S1135626FECC10351234103512341035123410 33C3
S113563612321033123210331232794C0080794A07
S113564600800D901910474C792000364E3E792084
S113565600204D0E793000200FB34702700A0FA3C6
S11356661AA2792000104D14793000100D380DB3AD
S11356760D2B0DA219AA0D884702700B0C884F1427
S1135686113213334402700B1B5040F01AA27A03F3
S1135696000000010C6815E84B2A0AB544020B7496
S11356A60AA4792C010058500080113413354402A2
S10756B6700D0B590C
S11356BA792907FF5860006E1AC41AD5580000A644
S11356CA1FA446061FB55870FE8A1AB544087A34D1
S11356DA0000000145061AA4450440121AA417350E
S11356EA17347A150000000144020B740CE60FC448
S11356FA46080FD41AD57919FFE00DCC460C0D4C88
S113570A0DD40D5D19557919FFF0792C010045085F
S113571A113413350B5940F2792C00804C0810359B
S113572A12341B5940F20D994E10113413350D9949
S113573A4A08113413350B5940F4732D471C0CD8FE
S113574AE80B47167A150000000844020B74792CFB
S113575A01004D06113413350B5911341335113425
S113576A133511341335796C000F101910191019E8
S113577A1019649C101C1006131C01006D700100A3
S113578A698401006F85000401006D7101006D7267
S113579A01006D7301006D7401006D756D765470AF
S11357AA01F06401586000 80792407FF5860006C97
S11357BA5800007401F064235860006C40520F854E
S11357CA01F06415460E0D44464601F06423464033
S11357DA58000054103112307928001 04C0C1B5C0D
S11357EA10311230792800104DF4580000BA0FA571
S11357FA01F0643547261033123279 2A00104C0C13
S113580A1B5410331232792A00104DF45800009EAB
S113581A1AC4100E131C1AD55800018E790E07FFED
S113582A1AC41AD5580001621AC418EE790E07FF72
S113583A19DD790500085800015001006DF60100D1
S113584A6DF501006DF401006DF301006DF20100C5
S113585A6DF101006DF0681E6826156E691C111C36
S113586A111C111C111C796C07FF692411141114E2
S113587A11141114796407FF01006D1001006911F5
S113588A7968000F01006F2300040100692279 6A15
S109589A000F792C07FF4B
S11358A05870FF06792407FF5870FF120DCC58700B
S11358B0FF160D445870FF40194C791C03FF0DCEA1
S11358C0792EFFCC58D0FF5279480010794A001046
S11358D01AC47A0500000100103312321031123 05D
S11358E0441C0AB144087A100000000145040AA0D0
S11358F040040AA004011235123444E0401C1AB1DA
S113590044087A300000000145041AA040041AA09C
S113591004011235DD01123444C2730D46080AB185
S113592044020B700AA001F064014702700D792C48

S113593000804C06103512341B5E792E07FF58C0C9
S1135940FEE40DEE4E2E113413354402700D0DEEB0
S11359504A040B5E40F0732D47180CD8E80B47122E
S11359607A1500000008440A0B74792C00804D025C
S11359700B5E4020732D471C0CD8E80B47167A1595
S1135980000000844020B74792C01004D06113409
S109599013350B5E113418
S113599613351134133511341335796C000F101E7A
S11359A6101E101E101E64EC101C100E131C01009A
S11359B66D700100698401006F85000401006D713B
S11359C601006D7201006D7301006D7401006D7548
S10959D601006D76547020
S11359DC01006DF201006DF101006F010004010083
S11359EC69000D82111211121112112112796207FF43
S11359FC0D8A7968000F793203FF4B4A79220053E1
S1135A0C4E44794800107912FFEC4B227922002086
S1135A1C4C0C0D224F1E103112301B5240F40F90C0
S1135A2C7912FFE00D224F0C10301B5240F611304F
S1135A3C0B524BFA796A8000470217B001006D7163
S10D5A4C01006D7254701A8040F2DD
S10F5A565E007914A80147021900547087
S1135A625E0079140C884E0419005470F8015470C6
S1135A725E0079140C884604F801547019005470BE
S1135A8201006DF201006DF11AA20F9147224A063D
S1135A927902080017B17912041F1B52103144FA1C
S1135AA21031121210311212103112121031121125D
S1135AB2698201006F8100026F8A000601006D7125
S1095AC201006D72547037
S1135AC801006DF201006902010069920100 6F0291
S1115AD8000401006F92000401006D7254700F
S1135AE66DF301006DF201006DF101006DF06C0BB9
S1135AF6689B0B011B7246F601006D7001006D7108
S10B5B0601006D726D73547010
S1115B0E5E0079141A084704790000015470F0
S1135B1C01006F76001401006D7201006FF200102A
S1135B2C01006D7201006D7301006D7401006D75E0
S1055B3C5470A0
S1135B3E01006DF501006DF401006DF301006DF2CE
S1135B4E01006F72001001006DF201006FF6001478
S10B5B5E01006F720004547092
S1135B665E005B3E7A37000000B27A0300FFE8026C
S1135B767A0400FFE7F20D0201006FF100AA0100AB
S1135B866F7600CE01006F7500D20D00466C0100E2
S1135B966F7000CA460E790004526BA000FFE82E10
S1135BA65A005C3801006F7000CA6E080010E807DF
S1135BB617D0460C790005146BA000FFE82E40723F
S1135BC601006F7000CA6E08001073284610730830
S1135BD6470C790105166BA100FFE82E405401001E
S1135BE66F7000CA6E0800107368464601006F7036
S1135BF600AA01006BA000FFE8260D2017F00100A4
S1135C066BA000FFE7EE01006F7000CA01006BA0F6
S1135C1600FFE7F41A8001006BA000FFE7F818887D
S1135C2668C8F8306EF800885C001CBA0D0058701E
S1135C360A1C7900FFFF5A0066766868A825586033
S1135C4609CC0B76188868C80FF001006BA000FF1B
S1075C56E8060FF05A
S1135C5A01006BA000FFE80A1A8001006BA000FF95

S1095C6AE80E01006BA02F
S1135C7000FFE8121A8001006BA000FFE816010084
S1135C806BA000FFE81A1A8001006BA000FFE81E5A
S1135C9040306868A8204718A8234720A82B470A44
S1135CA0A82D461C7D40701040167D40703040107A
S1135CB07C407330460A7D40704040047D40702034
S1135CC00B766868A82D47CAA82B47C6A82047C2E9
S1135CD0A82347BEF9206AA900FFE7FC6869A93039
S1135CE0460AF9306AA900FFE7FC0B761A80010027
S1135CF06BA000FFE7FE6868A82A586000800FD0F9
S1135D000BF001006FF000A07308470A01006F70E9
S1135D1000A00B70400601006F7000A00F851BF000
S1135D2001006FF000967308470A01006F70009638
S1135D301B70400601006F700096690017F00100A8
S1135D406BA000FFE7FE4C167D40701001006B2036
S1135D5000FFE7FE17B001006BA000FFE7FE0100A4
S1095D606B2000FFE7FECB
S1135D667A20000002004F0E7D407000790005186E
S1135D766BA000FFE82E0B76686817D0796000FFEA
S1135D8617F078006A280000987673285870008602
S1135D961A8001006BA000FFE7FE4064686817D015
S1135DA67930003017F001006FF000A07A0100008F
S1105DB602001A810F907A010000000A5EBE
S1135DC30079D001006B2100FFE7FE1F904D2401F2
S1135DD3006B2000FFE7FE7A010000000A5E0079F2
S1135DE3F601006F7100A00A9001006BA000FFE7AA
S1135DF3FE400E7D407000790005186BA000FFE89C
S1135E032E0B76686817D0796000FF17F078006A65
S1135E13280000987673284686 7A00FFFFFFFF0168
S1135E230069B06868A82E586001000B766868A8FB
S1135E332A466C0FD00BF001006FF00096730847EE
S1135E430A01006F7000960B70400601006F70002B
S1135E53960F851BF001006FF000A07308470A013A
S1135E63006F7000A01B70400601006F7000A069F3
S1135E730017F0010069B04C0A7A00FFFFFFFF012E
S1135E830069B0010069307A20000002004F0E7DE3
S1135E93407000790005186BA000FFE82E0B7668AD
S1135EA36817D0796000FF17F078006A280000981C
S1085EB3767328477619
S1135EB81A80010069B04058686817D079300030FB
S1135EC817F001006FF000A07A01000002001A81A8
S1135ED80F907A010000000A5E0079D00100693151
S1135EE81F904D1C010069307A010000000A5E0012
S1135EF879F601006F7100A00A90010069B0400EA5
S1135F087D407000790005186BA000FFE82E0B7622
S1135F18686817D0796000FF17F078006A280000D6
S1135F289876732846927A000000002001006BA03F
S1135F3800FFE8226868A8684708A86C4704A84CCB
S1135F484610686817D017F001006BA000FFE8221D
S1135F580B766868A825587004A4A84558700212DF
S1135F68A8475870020CA8584742A863587002E41F
S1135F78A8644738A865587001F8A866587001F2F4
S1135F88A867587001ECA8694722A86E58700430B6
S1135F98A86F4718A87058700398A873587002F828
S1095FA8A8754708A87864
S10D5FAE47045A00644201006B200F
S1135FB800FFE8227A200000006C46440FD00B90C3

S1135FC801006FF000AA7308470A01006F7000AA66
S1135FD80B70400601006F7000AA0F851B9001002B
S1135FE86FF000967308470A01006F7000961B70E4
S1135FF8400601006F700096010069025A00614A69
S113600801006B2000FFE8227A2000000068586036
S1136018009A6868A875470CA8584708A8784704E1
S1136028A86F46420FD00BF001006FF000AA730867
S1136038470A01006F7000AA0B70400601006F70D9
S113604800AA0F851BF001006FF000967308470A3A
S113605801006F7000961B70400601006F70009678
S11360686900177040400FD00BF001006FF00096E5
S11360787308470A01006F7000960B704006010011
S11360886F7000960F851BF001006FF000AA73086C
S1136098470A01006F7000AA1B70400601006F7069
S10960A800AA690017F0D5
S11360AE0F825A00614A6868A875470CA8584708BA
S11360BEA8784704A86F46420FD00BF001006FF08B
S11360CE00967308470A01006F7000960B70400626
S11360DE01006F7000960F851BF001006FF000AA90
S11360EE7308470A01006F7000AA1B704006010077
S11360FE6F7000AA6900177040400FD00BF00100BB
S113610E6FF000AA7308470A01006F7000AA0B70A4
S113611E400601006F7000AA0F851BF001006FF09F
S113612E00967308470A01006F7000961B704006B5
S113613E01006F700096690017F00F82010069303D
S113614E7A20FFFFFFFF460A7A00000000010100DC
S113615E69B0686817D06DF07A01000000020AF189
S113616E0FA05C00050E0B875A0063B601006B206F
S113617E00FFE8227A200000004C46420FD00B901D
S113618E0B9001006FF000AA7308470A01006F70AD
S109619E00AA0B7040068D
S11361A401006F7000AA0F851B901B9001006FF014
S11361B400967308470A01006F7000961B70404AEB
S11361C401006F70009640420FD00B900B900100BA
S11361D46FF000AA7308470A01006F7000AA0B70DE
S11361E4400601006F7000AA0F851B901B900100ED
S11361F46FF000967308470A01006F7100961B71D4
S1136204400601006F7100960F907A010000008030
S11362140AF15E005AC8010069307A20FFFFFFFFCC
S1136224460A7A0000000006010069B0686917D1C4
S113623401006F70008401006DF001006F70008431
S113624401006DF07A00000000080AF05C00078684
S11362540B970B975A0063B60FD00BF001006FF046
S113626400AA7308470A01006F7000AA0B70400666
S113627401006F7000AA0F851BF001006FF00096F8
S11362847308470A01006F7000961B7040060100F3
S10962946F7000966E0816
S113629A00015A0064080FD00B9001006FF000AAA6
S11362AA7308470A01006F7000AA0B7040060100C9
S11362BA6F7000AA0F851B9001006FF00096730898
S11362CA470A01006F7000961B70400601006F7049
S11362DA0096010069000F825E00548401006FF08A
S11362EA00AA010069317A21FFFFFFFF471201006B
S11362FA69311F814C0A0100693001006FF000AA5D
S113630A0FF001006BA000FFE80A1A8001006BA0DE
S113631A00FFE81601006B2000FFE7FE01006F7122
S110632A00AA1A9001006BA000FFE81E5AA4

S11363370064500FD00B9001006FF000967308476D
S11363470A01006F7000960B70400601006F700022
S1136357960F851B9001006FF000AA7308470A0187
S1136367006F7000AA1B70400601006F7000AA013E
S11363770069007A6000FFFFFF01006FF0009601DC
S11363870069307A20FFFFFFFF460A7A000000000A
S113639701010069B0790000786DF07A010000000F
S11363A7020AF101006F7000985C0002CE0B870FA1
S11363B7F00F825E00548401006FF000AA5A006454
S11363C7500FD00B9001006FF000A07308470A012C
S11363D7006F7000A00B70400601006F7000A00FE4
S11363E7851B900F81730847060F901B7040020FA0
S11363F790010069006B2100FFE7FA6981404AF8C1
S11364072568F80FF00F827A000000000101006F82
S1136417F000AA0FF101006BA100FFE80A1A91012E
S1066427006BA163
S113642A00FFE81601006B2100FFE7FE1A81010055
S113643A6BA100FFE81E400E790005186BA000FF50
S113644AE82E7D4070006868A86E587001B86A2803
S113645A00FFE7F27308586001AC7C4073104636BC
S113646A01006B2000FFE81E4F2C40207A01000038
S113647A00017A0000FFE7FC5C0013B87A0000FF12
S113648AE81E010069011B710100698101006B208B
S113649A00FFE81E4ED601006B2000FFE8124624D7
S11364AA01006B2000FFE816461A01006B2000FF6B
S11364BAE81A461001006F7100AA0FA05C0013745A
S11364CA5A0065D401006B2000FFE8060FA11A9059
S11364DA01006FF0009C01006FF000A04F30010033
S11364EA6F71009C0FA05C00134A40227A000000DF
S11364FA00880AF07A01000000015C0013367A0072
S113650A00FFE812010069011B71010069810100A2
S113651A6B2000FFE8124ED401006B2000FFE80A4B
S113652A01006B2100FFE8061A9001006FF0009C3E
S113653A01006F7100A00A8101006FF100A00F80B2
S113654A4F3601006F71009C01006B2000FFE806C3
S113655A5C0012E040227A00000000880AF07A0107
S113656A000000015C0012CC7A0000FFE81601006B
S113657A69011B710100698101006B2000FFE816A4
S109658A4ED401006F7105
S111659000AA01006F7000A01A8101006B20A9
S113659E00FFE80A5C00129840227A00000000888F
S11365AE0AF07A01000000015C0012847A0000FFF9
S11365BEE81A010069011B710100698101006B205A
S11365CE00FFE81A4ED47C407310473601006B204F
S11365DE00FFE81E4F2C40207A01000000017A00D4
S11365EE00FFE7FC5C0012487A0000FFE81E010082
S11365FE69011B710100698101006B2000FFE81E18
S113660E4ED60B76404001006FF600A6400E0100F9
S113661E6F7000A60B7001006FF000A601006F7083
S113662E00A668086EF8009547086E780095A825B1
S113663E46DC01006F7100A61AE10FE05C0011F059
S113664E01006F7600A6686847145C0012900D0077
S113665E460C6A2800FFE7F273085870F5D45C0005
S113666E125E6B2000FFE7FA7A17000000B25E009D
S113667E5B1C54705E005B3E7A37000000247A0583
S109668E000000040AF500
S113669401006FF0001801006FF1002001006FF199

S11366A4001C18AA689A6F72003C0C22463CAA5834
S11366B4472CAA644712AA69470EAA6F4712AA7500
S11366C44706AA78471840227A000000000A4006C9
S11366D47A000000000801006FF00014400C7A00F7
S11366E40000001001006FF000146F70003C79206B
S11366F400644706792000694612010006F70001890
S11367044C0A01006F74001817B4400601006F743B
S113671400181AE61AB30FC4467401006B2000FF75
S1136724E802476AF83068D87A060000000140623C
S11367340FD00AE00F82010069F00FC001006F71EE
S113674400145E007FFA0100697068890FA068086D
S1136754A8094E0C0FD00AE0680989306889401EE5
S11367646F70003C79200078460A0FD00AE068096C
S1136774895740080FD00AE06809893768890FC030
S109678401006F71001417
S111678A5E007FFA0F840B760FC4469E6A28CA
S113679800FFE7F2732847626F70003C0C00465A0B
S11367A8A858471EA86F4706A8784716404C01000B
S11367B86F70001847440FE00B760AD0F9306889E8
S11367C8403801006F700018473001006F700020D7
S11367D80B7001006FF000201B70F930688901000D
S11367E86F7000200B7001006FF000201B706E7932
S11367F8003D68897A03000000020FB00AE00F83A6
S113680801006FF600146F70003C7920006447069E
S1136818792000694672O1006B2000FFE8024608F0
S113682801006F700018476001006F7000184D4237
S11368386A2800FFE7F2733847160B7301006F707D
S113684800200B7001006FF000201B70F92B4036FD
S11368586A2800FFE7F27348472E0B7301006F7035
S113686800200B7001006FF000201B70F92068896D
S113687840160B7301006F7000200B7001006FF05E
S109688800201B70F92D36
S113688E68897A0600FFE80A01006F70001C680C25
S113689EAC2B4708AC2D4704AC20460E01006F709D
S11368AE001C0B70010069E040466A2800FFE7F206
S11368BE732847326F70003C0C004634A858470AC1
S11368CEA86F4714A8784702402601006F70001C7A
S11368DE0BF0010069E0401801006F70001C0B7093
S11368EE010069E0400A01006F70001C010069E0BD
S11368FE7A0400FFE802010069401FB04F14010043
S113690E6946010069441AB401006BA400FFE8163E
S113691E400C0FB61A8001006BA000FFE8167A0434
S113692E00FFE81E01006F70001C680BAB2B4708BD
S113693EAB2D4704AB20463801006B2000FFE7FE6A
S113694E1FE04F2C6A2800FFE7FCA8304622010007
S10D695E6B2000FFE7FE1AE07A0148
S113696800FFE816010069120A82010069921A8081
S1136978010069C0401E01006B2000FFE7FE1FE015
S11369884F0C01006B2000FFE7FE1AE040021A805B
S1136998010069C001006F7600141B76401A0100DC
S11369A86F7000200B7001006FF000201B700FE167
S11369B81B760AD1681968890FE64CE201006F70EB
S11369C80020189968897A17000000245E005B1C70
S11369D854705E005B3E7A37000000647A0300FF60
S11369E8E7F27A04000000040AF47A0500FFE802DB
S11369F80F866FF1005001006FF6005CF83068C82D
S1136A087A000000007C0AF07A010000986E5E00AC

S1136A185B0E0D00587000B87A000000002F0AF0D2
S1136A2801006DF07A000000002E0AF001006DF0FD
S1136A3801006F70008801006DF001006F7000881D
S1136A4801006DF07A01000000320AF17A000000BB
S1096A5800115E007A1636
S1136A5E7A17000000100D024752188868E80100EB
S1136A6E695001006B2100FFE7FE1F904F060100E6
S1136A7E6950400801006B2000FFE7FE01006BA088
S1136A8E00FFE81E7A0600FFE7FC0D207920FFFFCA
S1136A9E470C79200001460CF82B5A0077C6F82DC7
S1046AAE5A8A
S1136AAF0077C6F82A68E85A0077C87A0000000012
S1136ABF1101006DF00FC10B717A00000000260A5F
S1136ACFF05E007DFC0B97400CF8016EF8002F1859
S1136ADF886EC800017A000000000101006FF0000A
S1136AEF60400E01006F7000600B7001006FF000CB
S1136AFF6001006F7000600AC06808A83047E401A6
S1136B0F006F7000607A20000000014F7A01006F60
S1136B1F7000600AC05E0054840F8201006F700022
S1136B2F601B7001006F71002A0A8101006FF10071
S1136B3F2A7A000000000101006FF0005840300175
S1136B4F006F7000600AC001006F7100580AC168BE
S1136B5F08689801006F7000580B7001006FF00008
S1136B6F5801006F7000600B7001006FF00060013F
S1136B7F006F7000580FA11F904FC401006F70007A
S1136B8F580AC0189968890FC00B7001006FF00085
S1086B9F445E00548474
S1136BA401006FF000601A8001006FF000580100CB
S1136BB46F7000607A01000000025E0079D001006A
S1136BC46FF0004001006F70004401006F710060BA
S1136BD40A901B7001006FF0003C404E01006F707F
S1136BE4004401006F7100580A9001006FF0004CDB
S1136BF468086EF8005701006F71003C01006F7262
S1136C0400581AA101006FF1004801006F72004C93
S1136C14681968A901006F710048689801006F71D1
S1136C2400580B7101006FF1005801006F70005898
S1136C3401006F7100401F904DA201006F7000604E
S1136C44461AF8306EC800017A0000000001010002
S1136C546FF000601A8001006FF0002A6F7000501B
S1136C64792000674706792000474 63C7D30705001
S1136C7401006F70002A01006F7100600A901B709D
S1136C847A20FFFFFFFC4D0C01006B2100FFE8029B
S1096C941F904F0C6F700E
S1136C9A00501BD06FF000504008790000666FF077
S1136CAA00506E78002FA80146246838E8186EF859
S1136CBA0057A8084706A810470A401A0FE00B76A0
S1136CCAF92B40100FE00B76F920688940080FE092
S1136CDA0B76F92D68896F70005079200066586029
S1136CEA077201006F70002A58D001861A80401A71
S1136CFA0FE00B7601006F7100580AC10B71681916
S1136D0A688901006F7000580B7001006FF000581A
S1136D1A01006F7100601F904DD61A800F82010027
S1136D2A6BA600FFE80A01006F70002A01006BA03E
S1136D3A00FFE8167C307350460A01006B2000FFFF
S1136D4AE8024E067C307320470E0FE00B76F92ECD
S1136D5A68890FA00B700F827C307350470C7C300C
S1136D6A735047247C307320471E01006BA600FF33

S1136D7AE80E0100695001006BA000FFE81A010048
S1096D8A69500FA10A810C
S1136D900F9201006F70002A0FA10A9001006F711A
S1136DA000600A9001006FF0006001006F71005CE9
S1136DB06819A92B4708A92D4704A92046520100A9
S1136DC06B2000FFE7FE01006F7100601F904F40D2
S1136DD06A2800FFE7FCA830463601006F70005CAC
S1096DE00B7001006BA023
S1136DE600FFE80601006B2000FFE7FE01006F715C
S1136DF600601A9001006BA000FFE8121A800100E0
S1136E066BA000FFE81E5A0077C401006B2000FF49
S1136E16E7FE01006F7100601F904F4C01006B206D
S1136E2600FFE7FE01006F7100601A9001006BA07E
S1136E3600FFE81E6E78002FA80146186E78002F13
S1136E46A801586009786E780057A8084706A81065
S1136E565860096A7A0000FFE81E010069011B7188
S1136E66010069815A0077C41A8001006BA000FFF4
S1136E76E81E5A0077C401006F70006001006F714D
S1136E86002A0A8101006FF1002A010069520AA152
S1136E9601006FF100521F814C120F914D0E01003C
S1136EA66F7100520B710FC05C00092201006F70F5
S1136EB6002A58F0028A6848A83046147A0000006F
S1136EC6000101006FF0004C01006FF00052402CEE
S1096ED61A8001006FF0B9
S1136EDC004C1A8001006FF0005201006F70002A01
S1136EEC0B7001006FF0002A01006F7000600B70D3
S1136EFC01006FF000601A8001006FF00058403AF7
S1136F0C0FE00B7601006F71004C0AC16819688998
S1136F1C01006F70002A1B7001006FF0002A010042
S1136F2C6F7000580B7001006FF0005801006F7008
S1136F3C004C0B7001006FF0004C01006F70002AC5
S1136F4C4EBE0FE00B76F92E688940240FE00B76CA
S1136F5C01006F71004C0B7101006FF1004C1B7140
S1136F6C0AC168196889010069501B70010069D056
S1136F7C01006F7000580B7001006FF00058010096
S1136F8C6F70004C01006F7100521A9001006F7109
S1136F9C00601F904C0A01006B2000FFE8024EAC0E
S1136FAC7C307350470C7C307350472E7C307320ED
S1136FBC47280100695001006F7100580A810100D4
S1076FCC6FF1005806
S1076FD001006BA6A8
S1136FD400FFE80A0100695001006BA000FFE816F6
S1136FE440226E68FFFFA830461A40101B7601004A
S1136FF46F7000581B7001006FF000586E68FFFF3C
S1137004A83047E87C307320464A6E68FFFFA82EF9
S1137014464201006B2000FFE81647067C3073509C
S113702447321B7601006F70005801006B2100FF8B
S1137034E8161A901B7001006FF000581A800100C3
S11370446BA000FFE81601006F70005C01006BA0E9
S113705400FFE80A01006F70005C6808A82B47086A
S1137064A82D4704A820465001006B2000FFE7FE2B
S113707401006F7100581F904F3E6A2800FFE7FC20
S1137084A830463401006F70005C0B7001006BA0E4
S113709400FFE80601006B2000FFE7FE01006F71AB
S11370A400581A9001006BA000FFE8121A80010037
S11370B46BA000FFE81E406201006B2000FFE7FEA7
S10970C401006F7100588A

S10B70CA1F904F4601006B20EB
S11370D200FFE7FE01006F7100581A9001006BA0D8
S11370E200FFE81E6E78002FA80146146E78002F69
S11370F2A80146286E780057A8084704A810461C22
S11371027A0000FFE81E010069011B710100698119
S1137112400A1A8001006BA000FFE81E01006B20E9
S113712200FFE80A01006F71005C1F90586006922D
S113713201006B2000FFE80601006BA000FFE80AD4
S11371425A0077C46848A83046147A000000000148
S113715201006FF0004C01006FF00052402A1A80C8
S113716201006FF0004C01006FF0005201006F70DC
S1137172002A0B7001006FF0002A01006F7000609B
S11371820B7001006FF0006001006F70002A4F1452
S11371927A000000000101006FF0004C0FE00B7653
S11371A2684940060FE00B76F93068897A000000DF
S11371B2000101006FF000580FE10B76FA2E689A76
S10971C201006F7000588C
S11371C80B7001006FF0005801006F70002A58C05F
S11371D8009E01006BA600FFE80A01006F70002AF9
S11371E817B0010069511F814F2001006F70002AF9
S11371F817B001006BA000FFE81601006F70002AAA
S113720801006F7100581A8140180100695001008C
S11372186BA000FFE8160100695001006F71005868
S11372280A8101006FF1005801006F70002A010004
S113723869510A81010069D140360FE00B760100DC
S11372486F71004C0AC168196889010069501B7085
S1137258010069D001006F7000580B7001006FF0D6
S1137268005801006F70004C0B7001006FF0004C68
S113727801006F70004C01006F7100521A900100F9
S11372886F7100601F904C0A01006B2000FFE80239
S11372984EA87C307350470C7C30735047347C3095
S11372A87320472E010069504F4A01006BA600FF67
S10972B8E80E010069501D
S10772BE01006BA0BD
S11372C200FFE81A0100695001006F7100580A813A
S11372D201006FF1005840226E68FFFFA830461A82
S11372E240101B7601006F7000581B7001006FF095
S11372F200586E68FFFFA83047E87C307320467061
S11373026E68FFFFA82E466801006B2000FFE81697
S1137312460A01006B2000FFE81A47067C307350CF
S1137322474E1B7601006F70005801006B2100FF6E
S1137332E8161A9001006B2100FFE81A1A901B70DD
S113734201006FF000581A8001006BA000FFE81AD9
S113735201006BA000FFE81601006F70005C0100E2
S11373626BA000FFE80A1A8001006BA000FFE80E81
S113737201006F70005C6808A82B4708A82D47041A
S1137382A820465001006B2000FFE7FE01006F7149
S113739200581F904F3E6A2800FFE7FCA83046348E
S11373A201006F70005C0B7001006BA000FFE80628
S10773B201006B2048
S11373B600FFE7FE01006F7100581A9001006BA0F1
S11373C600FFE8121A8001006BA000FFE81E40626E
S11373D601006B2000FFE7FE01006F7100581F904C
S11373E64F4601006B2000FFE7FE01006F71005856
S11373F61A9001006BA000FFE81E6E78002FA8010B
S113740646146E78002FA80146286E780057A80800
S11374164704A810461C7A0000FFE81E0100690114

S11374261B7101006981400A1A8001006BA000FFED
S1137436E81E01006B2000FFE80A01006F71005C83
S11374461F90461001006B2000FFE80601006BA0A9
S113745600FFE80A5A0077C4010069500B70010067
S11374666F7100601F904C0C010069510BF10FC046
S11374765C00035A6848A830460E7A0000000001F3
S113748601006FF0004840241A8001006FF00048A5
S113749601006F70002A1B7001006FF0002A0100C3
S11374A66F7000600B7001006FF0006001006F7079
S10974B6006001006F718C
S11374BC00481A9001006F72002A0A8201006FF2D1
S11374CC002A0FE00B760AC1681968897A0000005C
S11374DC000101006FF000580FE10B76FA2E689A49
S11374EC01006F7000580B7001006FF00058010021
S11374FC6F700048402E0FE00B7601006F71004C4B
S113750C0AC168196889010069501B70010069D0B0
S113751C01006F7000580B7001006FF000580100F0
S113752C6F70004C0B7001006FF0004C01006F7119
S113753C00601F904E0A01006B2000FFE8024EB65C
S113754C7C307350470C7C307350472E7C30732047
S113755C472801006BA600FFE80A010069500100EF
S105756C6BA00F
S113756E00FFE8160100695001006F7100580A818F
S113757E01006FF1005840226E68FFFFA830461AD3
S113758E40101B7601006F7000581B7001006FF0E6
S113759E00586E68FFFFA83047E87C307320464AD8
S11375AE6E68FFFFA82E464201006B2000FFE8160F
S11375BE47067C30735047321B7601006F700058BC
S11375CE01006B2100FFE8161A901B7001006FF08B
S11375DE005801006F70005C01006BA000FFE80A09
S11375EE1A8001006BA000FFE8166F70005079201F
S11375FE006546080FE00B76F96540060FE00B7643
S113760EF945688901006F7000580B7001006FF027
S113761E005801006F70002A4D0A0FE00B76F92B0C
S113762E688940160FE00B76F92D688901006F709B
S113763E002A17B001006FF0002A01006F70005886
S113764E0B7001006FF0005801006F70002A7A2052
S109765E000000644D0E64
S11376640BF601006F7000580B800B7040140FE091
S11376740B76F9306889F83068E801006F700058B8
S11376840BF001006FF000580FE00B7001006FF076
S1137694003040360FE01B76010069F001006F7083
S11376A4002A7A010000000A5E0079D089300100C3
S11376B46970688901006F70002A7A010000000A6A
S11376C45E0079D001006FF0002A01006F70002A78
S11376D44EC201006F76003001006F70005C6808D1
S11376E4A82B4708A82D4704A820465001006B2067
S11376F400FFE7FE01006F7100581F904F3E6A2898
S113770400FFE7FCA830463401006F70005C0B7087
S113771401006BA000FFE80601006B2000FFE7FEF9
S113772401006F7100581A9001006BA000FFE8126A
S11377341A8001006BA000FFE81E406201006B2069
S113774400FFE7FE01006F7100581F904F460100D0
S10577546B20A5
S113775600FFE7FE01006F7100581A9001006BA04D
S113776600FFE81E6E78002FA80146146E78002FDE
S1137776A80146286E780057A8084704A810461C97

S11377867A0000FFE81E010069011B71010069818F
S1137796400A1A8001006BA000FFE81E01006B205F
S11377A600FFE80A01006F71005C1F90461001009C
S11377B66B2000FFE80601006BA000FFE80A1888AB
S11377C668E87A17000000645E005B1C54705E0074
S11377D65B3E0F850F960FD00AE06808A8344F5218
S11377E60FD00AE0F93068891B760FD00AE06809E2
S11377F68901688940200FD00AE06808A8394F1428
S11378060FD10AE1681888F668986E18FFFF880199
S11378166E98FFFF1B760FE64EDC6E580001A83903
S11378264F106E58000188F66ED80001685888011B
S113783668D85E005B1C54705E005B3E1B977A0340
S109784600FFE7F47A06DF
S113784C00FFE7F8010069F00F9501006B2100FFC1
S113785CE7EE7A2100000001462601006DF50F8149
S113786C010069305E007FA00B97010069300AD0DC
S113787C010069B0010069600AD0010069E0403A77
S113788C19444030010069700B70010069F01B70E2
S10F789C680817D00100693101006B225D
S11378A800FFE8265D207920FFFF47120100696089
S11378B80B70010069E00B5417F41FD44DCA0B97E2
S11378C85E005B1C547001006B2000FFE7EE7A201A
S11378D800000001460C01006B2000FFE7F4189933
S11378E86889547001006B2000FFE7EE7A200000DE
S11378F8000146041900547001006B2000FFE7F4EF
S10F79086E090010E94017D10D105470F7
S10B986E0000000000000000EF
S113791401006DF501006DF401006DF301006DF2DA
S113792401006DF101006D120100691301006F0183
S1137934000401006900 0D840D8C796C7FF0792CAF
S11379447FF047540DAC796C7FF0792C7FF0475668
S11379540D8C65AC4B5E1AB144087A30000000010B
S113796445141AA0451001F06410475A0C444B0AFD
S1137974790000024 00C0C444BF6190040047900D2
S1137984FFFF01006D7101006D7201006D73010051
S11379946D7401006D7554700F85796D000F01F0DE
S11379A4641546DA409E0FA5796D000F01F0643526
S11379B446CC409C01F064207A607FFFFFFF46AC15
S10F79C401F0643146A67900000140B6D2
S11379D06DF20D820C2A4A0217B00D994A04D28027
S11379E017B15E007FFA0C224A0217B00CAA4A02B2
S10979F017B16D72547023
S11379F601006DF301006DF20D8252120D935203D5
S1137A06521009280938 01006D7201006D73547014
S113987620202020202020202060606060602020 9F
S113988620202020202020202020202020202020CF
S11398964810101010101010101010101010101087
S11398A6848484848484848484841010101010101027
S11398B6108181818181810101010101010101010180
S11398C601010101010101010101011010101010 1034
S11398D610828282828282020202020202020251
S11398E602020202020202020202021010101020F9
S11398F60000000000000000000000000000005F
S1139906000000000000000000000000000000004E
S113991600000000000000000000000000000003E
S113992600000000000000000000000000000002E
S113993600000000000000000000000000000001E

S113994600000000000000000000000000000000E
S113995600000000000000000000000000000000FE
S1099966000000000F8
S10D996C0000000000000000000EE
S1137A165E005B3E7A370000003C7A040000000CEF
S1137A260AF47A05000000140AF501006FF0003825
S1137A3601006FF1003401006F73005C7A020000ED
S1137A46000801006FF2002C19226FF2002A7A0651
S1137A56000000540AF6686AEA7F46620FE00B707C
S1137A667A01000000075E0080360D0047501A8039
S1137A76401A01006F70003401006F7100380A90DC
S1137A861899688901006F7000380B7001006FF058
S1137A9600387A20000000084DD81A80010069B02A
S1137AA67C607370470A01006F700060F9FF40083D
S1137AB601006F700060F90168895A007DEE7A0053
S1137AC60000001C0AF001006DF07A010000002896
S1137AD60AF101006F70005C01006DF001006F7028
S1137AE6005C01006DF001006F70006C5E0083C6E0
S1137AF67A170000000C01006F7000606808A8FF89
S1077B06460C7A00AC
S1137B0A000000540AF07D0072706F70002479201F
S1137B1A07FF464A6E680001A8F0463A0FE00BF0E9
S1137B2A7A01000000065E0080360D004728010036
S1137B3A6F7000606808A8014608790000015A00BE
S1137B4A7DF001006F7000606808A8FF4610790095
S1137B5AFFFF5A007DF0790000025A007DF06F7032
S1137B6A0024793003FF6FF000244D1E01006F706B
S1137B7A005801006DF001006F70005801006DF0AC
S1137B8A5E0081560B970B97401E01006F700058D9
S1137B9A01006DF001006F70005801006DF05E0086
S1137BAA81560B970B971B500D0618886EF8002306
S1137BBA0B560D6017F001006F7100381A9001001F
S1137BCA69B0010069F10FD1010069705E00827228
S1137BDA01006F70003801006F71002C5E0079D0CC
S1137BEA01006FF000301AE67A0000000008010075
S1097BFA6F7100301A90C8
S1137C000F82401401006F7000300AE00AD00FD1D8
S1137C100AE1680868980B760FA01F864DE6400AB4
S1137C200FD00AE0189968890B767A2600000008BD
S1137C304DEE6F7000326F78002E52806F71003AF4
S1137C4019016DF17A01000000080FD05E0081FA7E
S1137C500B8701006F7600381B760FC10FE05E00C3
S1137C6082720FE001006F71002C5E0079D0010079
S1137C706FF000301AE67A000000000801006F710F
S1137C8000301A900F82401401006F7000300AE038
S1137C900AC00FC10AE1680868980B760FA01F8617
S1137CA04DE6400A0FC00AE0189968890B767A26D8
S1137CB0000000084DEE6F7000326F78002E528086
S1137CC06F71003A19011B516DF17A010000000830
S1137CD00FC05E0081FA0B8701006F700034010052
S1137CE06DF00100693101006DF17A01000000249B
S1097CF00AF16F70002C85
S1137CF65E00805E0B970B977A06000000040AF677
S1137D067A000000000801006DF001006F71003871
S1137D160FE05E007FA00B977A00000000080100C9
S10A7D266DF00FD10FE05EC9
S1137D2D00846A0B970D004F12790000016FF0006C

S1137D3D2A010069300B705A007DE67A00000000BD
S1137D4D0801006DF001006F7100380FE05E007FD8
S1137D5DA00B977A000000000801006DF00FD10F02
S1137D6DE05E00846A0B970D004632010069300B0B
S1137D7D70010069B001006F70003401006DF001F6
S1137D8D00693301006DF37A01000000240AF16FDD
S1137D9D70002C5E00805E0B970B9740447A0000B9
S1137DAD00000801006DF001006F7100380FE05EF7
S1137DBD007FA00B977A000000000801006DF00F03
S1137DCDC10FE05E00846A0B970D004C146F7000B9
S1137DDD2A460E010069301B70010069B05A007C00
S1127DEDD819007A170000003C5E005B1C54702D
S1137DFC5E005B3E7A370000002C0FF30F86010008
S1137E0C6FF100107A000000000101006FF0001800
S1137E1C01006F7500440A95188868D87A00000031
S1137E2C000801006DF00FE17A000000000C0AF06D
S1137E3C5E007FA00B9701006F7000445A007F8C8B
S1137E4C0FA00B707A01000000055E0079D00B7057
S1137E5C10307A06000000081A867A000000000829
S1137E6C1AE001006DF00FA11031103110317A11AD
S1137E7C000099760AE10FB00AE05E007FA00B9731
S1137E8C7A000000000801006FF000281A8001003E
S1137E9C6FF0002001006FF0001C7A040000000852
S1137EAC1AE401006FF400145A007F4E01006F7046
S1137EBC001401006DF00FB10AE17A000000000C10
S1137ECC0AF00AE05E00846A0B970D004D3C01003A
S1137EDC6DF40FB00AE001006DF07A0100000010A0
S1097EEC0AF10AE11A800D
S1137EF25E00831A0B970B9717F001006FF00018BF
S1137F0201006F70002801006F7100200A810100D7
S1137F126FF1002001006F7000287A010000000257
S1137F225E0079D001006FF000284730790000012C
S1137F326DF00FB00AE00FC15E0082A00B87010053
S1137F426F70001C0B7001006FF0001C01006F705A
S1137F52001C7A200000000458D0FF5A01006F7001
S1137F620018461C6E78002388306CD84004F83021
S1137F7268D81B7501006F7000101F8544F0401212
S1137F826E78002388306CD80FA01B700F8258C004
S1117F92FEB87A170000002C5E005B1C5470D2
S113997600000000000000800000000000005086
S113998600000000000003200000000000001F404C
S11399960000000000013880000000000C3500C4
S11399A600000000007A12000000000004C4B400A6
S11399B6000000002FAF080000000001DCD65000B5
S11399C600000012A05F2000000000BA43B7400069
S11399D600000746A5288000000048C273950000D2
S11399E60002D79883D20000001C6BF526340000D2
S10B99F6011C37937E080000F9
S1137FA05E005B3E0F850F9401006F7600181FC5BE
S1137FB047401FC544200FD30FC21AC440120FB04D
S1137FC00B730FA10B710F921B71681968890B74E6
S1137FD01FE445EA401C0FD30AE30AE40FC21AC4A4
S1137FE0400C0FA01B700F8268086CB80B741FE461
S10D7FF045F00FD05E005B1C5470D7
S1137FFA01006DF20D9946100D82177253120DA8E6
S113800A53100D810D28401E0F920D8117717908A7
S113801A0010121012311AA144020AA11B5846F287

S10F802A12101710177001006D725470D3
S11380365E005B3E0F840F951AE6400E0FC00AE002
S1138046680947041900400A0B761FD64DEE7900DE
S10B805600015E005B1C547085
S113805E5E005B3E7A370000000C7A0500000001DB
S113806E0AF50D0C0F967904000801006DF57A01DF
S113807E0000000E0AF101006F70002817B05E00B9
S113808E85300B9701006DF66DFC7A000000001031
S113809E0AF00FD15E0084A80B970B87790C003F73
S11380AE6F70000A190C0DC017F001D053400D0E5E
S11380BE7900004219C017F001006DF07A0100003B
S11380CE00090FD05E00877C0B97790600084014D9
S11380DE0D6019E017F00AD00D6117F10AD1680887
S11380EE68981B561DE64CE8400C0D6017F00AD03D
S11380FE189968891B560D664CF00DE05240190C09
S113810E6DFC7A01000000090FD05E0082A00B8780
S113811E7900003F6FF0000A7A01000000400FD093
S113812E5E0086927A000000000801006DF00FD108
S113813E01006F70002C5E007FA00B977A17000072
S109814E000C5E005B1C47
S1058154547062
S113815601006DF27A370000001A7A000000001859
S11381660AF001006F71002601006DF101006F71C5
S1138176002601006DF17A01000000080AF10100F2
S11381866DF15E0088187A170000000C6F710018F5
S11381960FF05E008A980F817A02000099FE7A003A
S11381A6000000080AF05E008B847A0000000010CD
S11381B60AF001006F71000C01006DF101006F718F
S11381C6000C01006DF17A01000000080AF10100BC
S11381D66DF15E00891E7A170000000C7A0000001C
S11381E600100AF05E0059DC7A170000001A01003D
S10781F66D725470DF
S10B99FE3FD34395810624DDEC
S11381FA5E005B3E1B971B87010069F00F946F7942
S113820A001E790D0008199D4754790100FF0DD20C
S113821A1A0A4B04101140F80C9B18DD1B740FC685
S113822A4028010069740AE468450CB816850FC032
S113823A0D915E008A54684814D868C80DD01A088C
S113824A4B04110540F80C5D1B760FE64CD40CDD8C
S113825A4706790100014002191117F10F900B87A4
S10B826A0B975E005B1C5470CE
S11382725E005B3E0F860F957A0000000008010046
S11382826DF01036103610367A0100009A060AE1B4
S11182920FD05E007FA00B975E005B1C547044
S1139A060000000000000001000000000000000547
S1139A160000000000000019000000000000007DA7
S1139A2600000000000000271000000000000C3579
S1139A360000000000003D09000000000001312D78
S1139A46000000000005F5E100000000001DCD65E3
S1139A5600000000009502F90000000002E90EDD97
S1139A66000000000E8D4A510000000048C27395A5
S1139A76000000016BCC41E9000000071AFD498D87
S1139A860000002386F26FC1000000B1A2BC2EC500
S1139A96000003782DACE9D900001158E460913D2C
S1139AA6000056BC75E2D6310001B1AE4D6E2EF5FF
S1139AB6000878678326EAC9002A5A058FC295EDFE
S1139AC600D3C21BCECCEDA10422CA8B0A00A42567

S1139AD614ADF4B7320334B96765C793FA10079D1B
S11382A05E005B3E7A370000000A0F83010069F12C
S11382B06F790022790C0008199C4752FAFF0DC010
S11382C01A084B04110A40F80CA218CC1AE64026EF
S11382D00FB50AE568540C2816840FD00D915E0083
S11382E08A76685814C868D80DC01A084B0410045D
S11382F040F80C4C0B76010069701F864DD20CCCF4
S1138300470679010001400219111 7F10F907A17FE
S10D83100000000A5E005B1C5470BD
S113831A5E005B3E7A370000000C01006FF0000438
S113832A0F9601006F7500280AD61B7601006F733A
S113833A00240AD31B737A02000000011AC44044C2
S113834A6868175068391751191017F01AC00100D5
S113835A69F04C14010069717A11000001000100EF
S113836A69F1790000014002190017F00F84010036
S113837A697047041A800F826E78000368E81B75D8
S113838A1B761B730FD546B87A2400000001460EEC
S113839A01006F70000446067900FFFF40120FA028
S11383AA7A200000000146041900400479000000104
S10F83BA7A170000000C5E005B1C54707E
S11383C65E005B3E7A03000000070F820F950100F3
S11383D66F7600207A04000000180AF46941796177
S11383E6800079218000460679 01FFFF4004790168
S11383F600010FA06889694079607FF011901190A0
S11384061190119069D07A000000000701006DF009
S11384160B740FC10FE05E007FA00B97790000037A
S11384266DF00FB10FE05E0081FA0B8769504F06BE
S11384367D607070402869500B5069D07D60727002
S1138446401679000001 6DF00FB10FE05E0081FA6E
S11384560B8769501B5069D07C60737047E45E00DC
S10784665B1C5470D4
S113846A5E005B3E0F860F9301006F7400180FE5E1
S113847A0FC44604190040201AE6400A6C586C39A6
S113848A1C9846060B761FC645F21B75685817508B
S111849A1B73683B175319305E005B1C547054
S11384A85E005B3E7A37000000100FF60F850F94CD
S11384B869506F710028091069D001006DF6010039
S11384C86F71002E0FC05E008E600B977C60737017
S11384D8470869500B5069D0400C7A01000000101E
S11384E80FE05E008E0E7A000000004201006DF07E
S11384F87A01000000100FE05E00877C0B976E681E
S11385080008E8E06EE800087A00000000090100AE
S11385186DF00FE10FC05E007FA00B977A17000084
S10B852800105E005B1C54709F
S11385305E005B3E7A37000000187A0500000009F0
S11385400AF50F840F920FC44D087A03000000014F
S113855040147A03FFFFFFFF0FC07A01FFFFFFFF05
S11385605E0079F60F840FC07A010000001B5E00E5
S113857079D00F860FC07A010000001B5E0079D00E
S11385800F947A2300000001462E7A0000000008B1
S113859001006DF00FE11031103110317A1100003C
S11385A09AE60FD05E007FA00B970FE010307800A3
S11385B06B2100009BB640320FC446021B767A0043
S11385C00000000801006DF00FE11031103110318F
S11385D07A1100009B4E0FD05E007FA00B970FE037
S11385E0103078006B2100009BD06FF100120FC494
S11385F047747A23FFFFFFFF460A7A000000001B3F

S11386001AC00F8401006F70003001006DF00FA1DC
S11386100FC05E008FEE0B970FE646087A2300002B
S107862000014762A9
S113862401006F70003001006DF00FA069006DF060
S11386347A00000000180AF00FD15E0084A80B979B
S11386440B877A01000000400FD05E0086927920E8
S11386540001460E6F7000120B506FF000127D5034
S113866470700FA06F71001269817A000000000816
S113867401006DF00FD101006F7000345E007FA024
S11186840B977A17000000185E005B1C547001
S1139AE68000000000000000CECB8F27F4200F3A41
S1139AF6A70C3C40A64E6C5286F0AC99B4E8DAFD4E
S1139B06DA01EE641A708DEAB01AE745B101E9E4A9
S1139B168E41ADE9FBEBC27DE5D3EF282A242E81E6
S1139B26B9A74A0637CE2EE195F83D0A1FB69CD94A
S1139B36F24A01A73CF2DCD0C3B8358109E84F07E6
S1139B469E19DB92B4E31BA99E74D1B791E07E48BC
S1139B56C428D05AA4751E4CF2D56790AB41C2A453
S1139B66964E858C91BA2655BA121A4650E4DDEC08
S1139B76E65829B3046B0AFA8E938662882AF53F60
S1139B86B080392CC4349DEDDA7F5BF59096684935
S1139B96873E4F75E2224E68A76C582338ED26237D
S1139BA6CF42894A5DCE35EB8049A4AC0C5811AE41
S1139BB60000005900B3010D016601C0021A0273C9
S1139BC602CD0327038003DA0434FFA6FF4CFEF21B
S1099BD6FE99FE3FFDE5D0
S1119BDCFD8CFD32FCD8FC7FFC25FBCBFB721D
S11386925E005B3E7A370000000C0F840F957A066A
S11386A2000000081AB30FD00FE15E0079D00AC0B0
S11386B20F820FD00FE15E0079D0790000801A0992
S11386C24B04119040F86EF8000511086EF8000B88
S11386D211086EF800041B75010069F50FD00FE154
S11386E25E0079D00AC00F85010069700FE15E0058
S11386F279D0790600801A094B04119640F80FA02D
S113870268086E790005169847280FA068066E78E8
S1138712000B166846086E780004166847087A0349
S113872200000001400C685816E847067A0300006F
S113873200017A2300000001463240140CE817506E
S11387421710685916896 8D9100E46041B75FE0165
S1138752685816E847041FC544E21FC54508685810
S113876214E868D84006790000014002190 07A171C
S10D87720000000C5E005B1C547055
S113877C5E005B3E1B971B970F82010069F17A0326
S113878C0000000801006F7000200FB15E0079D06B
S113879C0FA60A8601006F7000200FB15E0079D01E
S11387AC790400801A094B04119440F8686816C8C0
S11387BC46500CCD1855400414D5110D0CDD46F85C
S11387CC686816584708686814C868E840340FA0EE
S11387DC010069710A90010069F001006F700020BB
S11387EC0FB15E0079D00FA50A85400C6868470865
S11387FC685814C868D8400A0B76010069701F8644
S10F880C45EA0B970B975E005B1C547051
S11388185E005B3E0F857A040000000701006F705D
S1138828002001006DF001006F70002001006DF061
S11388385E008DBC0B970B970D0647487926000100
S1138848460A790004B86BA000FFE82E79260002D7
S1138858460A7900044C6BA000FFE82E19667A00DB

S11388680000001C0AF00D6117F10A90F9FF6889EE
S11388780B56792600084DE67A000000001C0AF022
S11388885A00890E7A000000001C0AF07A010000E1
S11388989BEA5E005A560D00470C190069D07A000E
S11388A800009BEA40607A060000001C0AF6696330
S11388B81113111131113113796307FF793303FE8E
S11388C869D36950791003FE462869500B5069D063
S11388D869607960800F69E00FE30B73400E0FC185
S11388E80FB05E008E0E69501B5069D069607348E3
S11388F847EC69607960800F79403FE069E07A006E
S10989080000001C0AF050
S113890E01006F7100185E005AC85E005B1C547044
S10B9BEA000000000000000070
S113891E5E005B3E0F8601006F70002001006DF05C
S113892E01006F70002001006DF05E008DBC0B978F
S113893E0B970D05474079250001460A790004B8C7
S113894E6BA000FFE82E79250002460A7900044C3D
S113895E6BA000FFE82E19667A000000001C0AF0D7
S113896E0D6117F10A90F9FF68890B5679260008F5
S113897E4DE65A008A3C7A050000001C0AF569553B
S113898E11151115115115115796507FF793503FFAA
S113899E0D5C4C0E7A0000009BF20FE15E005AC88C
S11389AE40607A000000001C0AF00FE15E005AC816
S11389BE792C00344C4C0FE50B950BF579090034EB
S11389CE19C90D9017F07901001001D053100D093C
S11389DE400A0FD01BF5191169811B590D994EF2DF
S11389EE7900003419C017F07901001001D053102B
S11389FE7909FFFF1B584B04101940F86950669014
S1078A0E69D07A01AD
S1138A120000001C0AF10FE20F905E00556A0FE09E
S1138A227A0100009BF25E005A720D00470C7A0035
S1138A320000001C0AF07D0070707A000000001C28
S1138A420AF001006F7100185E005AC85E005B1CD9
S1058A5254705B
S10B9BF2000000000000000068
S1138A5401006DF20D1A4F14792A00084D04FA002F
S1138A644008680A100A1B5A4EFA688A01006D729C
S1058A74547039
S1138A7601006DF20D1A4F14792A00084D04FA000D
S1138A864008680A110A1B5A4EFA688A01006D7279
S1058A96547017
S1138A9801006DF119990D11471A4A067909080061
S1138AA817917919040F1B59101144FA1031103119
S1138AB810311031010069811A9101006F8100049E
S1098AC801006D71547002
S1138ACE0F8501F064155860009A792407FF471447
S1138ADE0D44586000820FA501F06435586000788C
S1138AEE580000800FA501F0643558600076406889
S1138AFE0FA501F06435466C0DCC465C0F8501F075
S1138B0E64154654405E0F8501F06415473C1031E1
S1138B1E1230792800104C0C1B5C1031123079285E
S1138B2E00104DF4580000C40FA501F06435471A28
S1138B3E10331232792A00104C0C1B54103312329C
S1138B4E792A00104DF4580000A81AA21AB3687DB2
S1138B5E100D131A58000228790107FF1AA21AB32F
S1138B6E580001FA790107FF1AA27A0300000008E0
S1138B7E68FA580001E801006DF601006DF5010079

S1138B8E6DF401006DF301006DF201006DF1010052
S1138B9E6DF07A3700000018691D692565D569F5F2
S1138BAE691C111C111C111C111C796C07FF692403
S1098BBE1114111411143F
S1138BC41114796407FF01006D10010069117968BC
S1138BD4000F01006F23000401006922796A000F6A
S1138BE4792C07FF5870FEE2792407FF5870FF0AB7
S1138BF40DCC5870FF1A0D445870FF36094C793C5C
S1138C0403FF792C07FF58C0FF58792CFFCB58D0AA
S1138C14FF426FFC000279480010794A00100100FA
S1138C246FF000040F9601006FF2000801006FF368
S1138C34000C0D1552356FF500160D34529452B1D4
S1138C440A946511990009D44406791C000199001A
S1138C546FF400140DC50D1D18990D0452340AC583
S1138C6499000DE452B40AC599000D6452240AC54F
S1138C7499006FF500120DD50D1D18990D8452340A
S1138C840AC50D0452B40AC599000DE452240AC559
S1138C9499000D6452A40AC599006FF500100DD50F
S1138CA40D1D18990DB352830AB50D0452240AC538
S1098CB499000DE452A437
S1138CBA0AC59900528209D24404791A0001091A91
S1138CCA6F74000852040AC26F7400085284094A76
S1138CDA0D5B6F73001001006F7400126F7D001635
S1138CEA19556F710002113213331331334133 50DA062
S1138CFA79600100471211321333133413350B51C0
S1138D0A792107FF5870FE5401F064454702700B3E
S1138D1A0D114E2E113213334402700B0D114A04F6
S1138D2A0B5140F0732B47180CB8E80B47127A1310
S1138D3A00000008440A0B72792A00804D020B5185
S1138D4A4020732B471C0CB8E80B47167A13000014
S1138D5A000844020B72792A01004D0611321333BB
S1138D6A0B51113213331132133311321333796A1C
S1138D7A000F101110111011011641A101A6878CB
S1138D8A1008131A7A170000001801006D70010009
S1138D9A698201006F83000401006D7101006D7225
S1078DAA01006D73E1
S1118DAE01006D7401006D7501006D76547047
S1138DBC01006DF57A01000000080AF16818E87FDC
S1138DCCA87F460A0B716818E8F0A8F0470419004D
S1138DDC402A6818F01050006898460A0B711AD58F
S1138DEC400E6818470679000000140100B710B7593
S1138DFC7A25000000064DEA7900000201006D752A
S1058E0C54709D
S1138E0E5E005B3E0F840F931AD51B730FB640287B
S1138E1E0FC30AE30FB26838E880175017700F8339
S1138E2E0FA06809100968890FD547080FC00AE01B
S1138E3E7D0070000FB51B760FE64CD40FD5470699
S1138E4E790100014002191117F10F905E005B1CAE
S1058E5E54704B
S1138E605E005B3E7A37000000387A05000000049C
S1138E700AF50F840F967A000000001001006DF0D0
S1138E80191101006F7000545E0090D20B970FE32D
S1138E900B930BF37A160000000701006FF6003402
S1138EA0790600030FC00B900BF001006FF0002850
S1138EB07A140000000701006FF4002C5A008FDCC5
S1138EC06838460C01006F7000346809587000FA66
S1138ED07A000000001001006DF019110FD05E0040

S1138EE090D20B9701006F70002801006FF00030E3
S1138EF001006F74002C790E0003407C01006F7039
S1138F0000301A916809FA0810311A0A4EFA1A80C9
S1138F10684801F064101A916839FA0810311A0A86
S1138F204EFA01006F720034010069F01A8068285C
S1138F3001F06401010069705E0079F601006FF0D1
S1138F4000247A000000000401006DF07A010000A3
S1098F5000280AF10DE206
S1138F56096217F210320AD20FA05E00908A0B97AD
S1138F661B5E01006F7000301BF001006FF00030D4
S1138F761BF40DEE4C807926000346247A0000008C
S1138F86000A01006DF00D6417F40FC110310AD108
S1138F9601006F7000540AC00AC05E007FA0402221
S1138FA67A000000000A01006DF00D6417F40FC18A
S1138FB610310AD101006F7000540AC00AC05E0066
S1138FC6908A0B971B561BF301006F7000341BF03E
S1138FD601006FF000340D6658C0FEDE7A170000FC
S10B8FE600385E005B1C5470AF
S1138FEE5E005B3E7A370000000C0F860F940D6017
S1138FFE7910003F69C00FF10FE05E0082727A00B4
S113900E0000000801006DF0191101006F700028B7
S113901E5E0090D20B971A800F8219550FF64010EF
S113902E0B760FA00B700F8269407930000869C070
S113903E686847ECFB804004110B0B55686816B843
S113904E47F66DF57A03000000080FA01A830FB1DF
S113905E0FE05E0081FA0B876940195069C0010069
S113906E6DF30FE101006F7000285E007FA00B9778
S10F907E7A170000000C5E005B1C5470AD
S113908A5E005B3E0F860F9401006F7500180AD6C7
S113909A1B760AD41B740FC319CC40226868175075
S11390AA68391751091009C00D0468E817F47900E3
S11390BA010001D053040D4C1B751B761B730FD58E
S10B90CA46DA5E005B1C5470E2
S11390D25E005B3E1B870F8469F101006F73001A08
S11390E20FC51AE6400C0FD00B756E790001688923
S11390F20B761FB645F00FC00B875E005B1C5470E6
S9030000FD

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

LINK COMMAND LINE

LNK -subcommand=linkfile.sub

LINK SUBCOMMANDS

```
OUTPUT c_thread
PRINT c_thread
INPUT asmfile, main, EV_Simulator, EV_Controller, EV_Input, EV_OpenClose, EV_UpDown, EV_File,
EV_Time, Timer, Panel, sci, lcd, usb
LIB c:\h8\akic\c38hab
START R(0FFE000), P(200), D(91C0), C(9200)
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'0000126F   H'00000FC0
                                  main                       main
                       H'00001270   -   H'00001E93   H'00000C24
                                  EV_Simulator               EV_Simulator
                       H'00001E94   -   H'000024CB   H'00000638
                                  EV_Controller              EV_Controller
                       H'000024CC   -   H'00002915   H'0000044A
                                  EV_Input                   EV_Input
                       H'00002916   -   H'0000318D   H'00000878
                                  EV_OpenClose               EV_OpenClose
                       H'0000318E   -   H'000039E5   H'00000858
                                  EV_UpDown                  EV_UpDown
```

```
                H'000039E6   -   H'00003DA1   H'000003BC
                                 EV_File              EV_File
                H'00003DA2   -   H'00003EF9   H'00000158
                                 EV_Time              EV_Time
                H'00003EFA   -   H'0000444D   H'00000554
                                 Timer                Timer
                H'0000444E   -   H'000045A3   H'00000156
                                 Panel                Panel
                H'000045A4   -   H'00004675   H'000000D2
                                 sci                  sci
                H'00004676   -   H'000048B3   H'0000023E
                                 lcd                  lcd
                H'000048B4   -   H'000053A5   H'00000AF2
                                 usb                  usb
                H'000053A6   -   H'000053DF   H'0000003A
                                 rand                 rand
                H'000053E0   -   H'0000543D   H'0000005E
                                 sprintf              sprintf
                H'0000543E   -   H'00005467   H'0000002A
                                 strcmp               strcmp
                H'00005468   -   H'00005483   H'0000001C
                                 strcpy               strcpy
                H'00005484   -   H'0000549F   H'0000001C
                                 strlen               strlen


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

                   ***    LINKAGE EDITOR LINK MAP LIST     ***


SECTION   NAME              START    -    END      LENGTH
                                 UNIT NAME              MODULE NAME


ATTRIBUTE :   CODE  NOSHR

P                H'000054A0   -   H'000054CF   H'00000030
                                 vsprintf             vsprintf
                H'000054D0   -   H'000057A9   H'000002DA
                                 addd3                addd3
                H'000057AA   -   H'000059DB   H'00000232
                                 divd3                divd3
                H'000059DC   -   H'00005A55   H'0000007A
                                 dtol3                dtol3
                H'00005A56   -   H'00005A61   H'0000000C
                                 eqd3                 eqd3
                H'00005A62   -   H'00005A71   H'00000010
                                 ged3                 ged3
                H'00005A72   -   H'00005A81   H'00000010
                                 ltd3                 ltd3
                H'00005A82   -   H'00005AC7   H'00000046
                                 ltod3                ltod3
                H'00005AC8   -   H'00005AE5   H'0000001E
                                 mv83                 mv83
                H'00005AE6   -   H'00005B0D   H'00000028
```

```
                    mvn3                    mvn3
H'00005B0E  -  H'00005B1B  H'0000000E
                    ned3                    ned3
H'00005B1C  -  H'00005B3D  H'00000022
                    spregld3                spregld3
H'00005B3E  -  H'00005B65  H'00000028
                    spregsv3                spregsv3
H'00005B66  -  H'00007913  H'00001DAE
                    _fmtout                 _fmtout
H'00007914  -  H'000079CF  H'000000BC
                    cmpd3                   cmpd3
H'000079D0  -  H'000079F5  H'00000026
                    divl3                   divl3
H'000079F6  -  H'00007A15  H'00000020
                    mull3                   mull3
H'00007A16  -  H'00007DFB  H'000003E6
                    _dti                    _dti
H'00007DFC  -  H'00007F9F  H'000001A4
                    _its                    _its
H'00007FA0  -  H'00007FF9  H'0000005A
                    memcpy                  memcpy
H'00007FFA  -  H'00008035  H'0000003C
                    divul3                  divul3
H'00008036  -  H'0000805D  H'00000028
                    _allzero                _allzero
H'0000805E  -  H'00008155  H'000000F8
                    _calcnpw                _calcnpw


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

    ***    LINKAGE EDITOR LINK MAP LIST    ***


SECTION  NAME              START    -    END      LENGTH
                    UNIT NAME                MODULE NAME


ATTRIBUTE :  CODE  NOSHR

P                  H'00008156  -  H'000081F9  H'000000A4
                    _log10                  _log10
H'000081FA  -  H'00008271  H'00000078
                    _lsfts                  _lsfts
H'00008272  -  H'0000829F  H'0000002E
                    _pow5                   _pow5
H'000082A0  -  H'00008319  H'0000007A
                    _rsfts                  _rsfts
H'0000831A  -  H'000083C5  H'000000AC
                    _sub                    _sub
H'000083C6  -  H'00008469  H'000000A4
                    _unpack                 _unpack
H'0000846A  -  H'000084A7  H'0000003E
                    memcmp                  memcmp
H'000084A8  -  H'0000852F  H'00000088
                    _mult64                 _mult64
```

H'00008530 - H'00008691 H'00000162
_power _power
H'00008692 - H'0000877B H'000000EA
_rnd _rnd
H'0000877C - H'00008817 H'0000009C
_setsbit _setsbit
H'00008818 - H'0000891D H'00000106
frexp frexp
H'0000891E - H'00008A53 H'00000136
modf modf
H'00008A54 - H'00008A75 H'00000022
dslc3 dslc3
H'00008A76 - H'00008A97 H'00000022
dsruc3 dsruc3
H'00008A98 - H'00008ACD H'00000036
itod3 itod3
H'00008ACE - H'00008DBB H'000002EE
muld3 muld3
H'00008DBC - H'00008E0D H'00000052
_duchek _duchek
H'00008E0E - H'00008E5F H'00000052
_lsft _lsft
H'00008E60 - H'00008FED H'0000018E
_mult _mult
H'00008FEE - H'00009089 H'0000009C
_pow10 _pow10
H'0000908A - H'000090D1 H'00000048
_add _add
H'000090D2 - H'00009101 H'00000030
memset memset

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

*** LINKAGE EDITOR LINK MAP LIST ***

SECTION NAME START - END LENGTH
UNIT NAME MODULE NAME

* TOTAL ADDRESS * H'00000200 - H'00009101 H'00008F02

ATTRIBUTE : DATA NOSHR ROM

D H'000091C0 - H'000091C0 H'00000000
asmfile asmfile
H'000091C0 - H'000091CF H'00000010
usb usb

* TOTAL ADDRESS * H'000091C0 - H'000091CF H'00000010

ATTRIBUTE : DATA NOSHR

C H'00009200 - H'0000948D H'0000028E

```
                    main                    main
          H'0000948E  -  H'000095E9  H'0000015C
                    EV_Simulator            EV_Simulator
          H'000095EA  -  H'000095F5  H'0000000C
                    EV_Controller           EV_Controller
          H'000095F6  -  H'00009656  H'00000061
                    EV_Input                EV_Input
          H'00009658  -  H'000096A3  H'0000004C
                    EV_OpenClose            EV_OpenClose
          H'000096A4  -  H'000096E6  H'00000043
                    EV_UpDown               EV_UpDown
          H'000096E8  -  H'00009767  H'00000080
                    EV_File                 EV_File
          H'00009768  -  H'0000976F  H'00000008
                    EV_Time                 EV_Time
          H'00009770  -  H'000097A7  H'00000038
                    Timer                   Timer
          H'000097A8  -  H'000097B2  H'0000000B
                    Panel                   Panel
          H'000097B4  -  H'0000986D  H'000000BA
                    usb                     usb
          H'0000986E  -  H'00009875  H'00000008
                    _fmtout                 _fmtout
          H'00009876  -  H'00009975  H'00000100
                    _ctype                  _ctype
          H'00009976  -  H'000099FD  H'00000088
                    _its                    _its
          H'000099FE  -  H'00009A05  H'00000008
                    _log10                  _log10
          H'00009A06  -  H'00009AE5  H'000000E0
                    _pow5                   _pow5
          H'00009AE6  -  H'00009BE9  H'00000104
                    _power                  _power


          H8/300H LINKAGE EDITOR (Evaluation software) Ver.1.0   PAGE :    5

             ***    LINKAGE EDITOR LINK MAP LIST     ***


SECTION   NAME              START    -    END      LENGTH
                    UNIT NAME               MODULE NAME


ATTRIBUTE :  DATA  NOSHR

C                   H'00009BEA  -  H'00009BF1  H'00000008
                    frexp                   frexp
          H'00009BF2  -  H'00009BF9  H'00000008
                    modf                    modf

* TOTAL ADDRESS *          H'00009200  -  H'00009BF9  H'000009FA


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'00FFE1D3   H'000001C2
                                    main                    main
                    H'00FFE1D4   -   H'00FFE1E5   H'00000012
                                    EV_File                 EV_File
                    H'00FFE1E6   -   H'00FFE409   H'00000224
                                    Timer                   Timer
                    H'00FFE40A   -   H'00FFE489   H'00000080
                                    Panel                   Panel
                    H'00FFE48A   -   H'00FFE4D9   H'00000050
                                    sci                     sci
                    H'00FFE4DA   -   H'00FFE519   H'00000040
                                    lcd                     lcd
                    H'00FFE51A   -   H'00FFE7ED   H'000002D4
                                    usb                     usb
                    H'00FFE7EE   -   H'00FFE829   H'0000003C
                                    _fmtout                 _fmtout
                    H'00FFE82A   -   H'00FFE82D   H'00000004
                                    _rnext                  _rnext
                    H'00FFE82E   -   H'00FFE82F   H'00000002
                                    _errno                  _errno

* TOTAL ADDRESS *           H'00FFE010   -   H'00FFE82F   H'00000820


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

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

                    SYMBOL  NAME                ADDR            TYPE

                    $ADDD$3                     H'0000559A      DAT
                    $CMPD$3                     H'00007914      DAT
                    $DIVD$3                     H'00005844      DAT
                    $DIVL$3                     H'000079D0      DAT
                    $DIVUL$3                    H'00007FFA      DAT
                    $DSLC$3                     H'00008A54      DAT
                    $DSRUC$3                    H'00008A76      DAT
                    $DTOL$3                     H'000059DC      DAT
                    $EQD$3                      H'00005A56      DAT
                    $GED$3                      H'00005A62      DAT
                    $ITOD$3                     H'00008A98      DAT
                    $LTD$3                      H'00005A72      DAT
                    $LTOD$3                     H'00005A82      DAT
                    $MULD$3                     H'00008B84      DAT
```

$MULL$3	H'000079F6	DAT
$MV8$3	H'00005AC8	DAT
MVN3	H'00005AE6	DAT
NED3	H'00005B0E	DAT
$SUBD$3	H'0000556A	DAT
sp_regld3	H'00005B1C	DAT
sp_regsv3	H'00005B3E	DAT
_Checkfmove	H'00003EBC	ENT
_Clear	H'0000444E	ENT
_ClearLCD	H'00004794	ENT
_Close	H'00003140	ENT
_CloseMotor	H'00002BEC	ENT
_Cnt	H'00FFE012	DAT
_Command_Read	H'00003B74	ENT
_Command_Write	H'00003BC0	ENT
_Destroy	H'000010DA	ENT
_DisableInterrupt	H'000002AC	DAT
_Disp	H'0000177E	ENT
_DispInput	H'0000172E	ENT
_DispUSBPort	H'000049C4	ENT
_Door	H'00002916	ENT
_Down	H'00003998	ENT
_DownMotor	H'00003464	ENT
_EV_Controller	H'00001E94	ENT
_EV_File	H'00003A34	ENT
_EV_Input	H'00002538	ENT
_EV_Simulator	H'00001270	ENT
_EV_Time	H'00003DA2	ENT
_EnableInterrupt	H'000002A8	DAT
_GetChar	H'000024CC	ENT
_GetCurrentTime	H'00003E04	ENT
_GetPermit	H'00003E9E	ENT
_GetSCI	H'000045D4	ENT
_GetSW	H'0000121E	ENT
_H8init	H'00001244	ENT
_Init	H'00000E6C	ENT
_InitITU	H'000043D2	ENT

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

SYMBOL NAME	ADDR	TYPE
_InitLCD	H'00004724	ENT
_InitSCI	H'000045A4	ENT
_InitUSB	H'000048CA	ENT
_InterruptITU0	H'00004408	ENT
_LCDOut4	H'000046CC	ENT
_Limit_Read	H'00003D5E	ENT
_LocateLCD	H'000047D4	ENT
_Motor_Read	H'00003D10	ENT
_Motor_Write	H'00003CD8	ENT
_OnCloseMotor	H'00002C08	ENT
_OnController	H'00001FB2	ENT

_OnDownMotor H'00003480 ENT
_OnInitWaitDoorChangeLog H'00002E76 ENT
_OnInitWaitPositionChangeLog H'000036E0 ENT
_OnInput H'00002612 ENT
_OnOpenMotor H'000029A6 ENT
_OnSimulator H'0000135A ENT
_OnUpMotor H'0000321E ENT
_OnWaitCloseDoorChangeLog H'0000301E ENT
_OnWaitDownPositionChangeLog H'0000387C ENT
_OnWaitOpenDoorChangeLog H'00002F4A ENT
_OnWaitUpPositionChangeLog H'000037AE ENT
_Open H'000030F2 ENT
_OpenMotor H'0000298A ENT
_PermitCommand_Write H'00003B30 ENT
_PermitTurnOpen_Write H'00003C04 ENT
_Position H'0000318E ENT
_PrintF H'00004476 ENT
_PrintLCD H'000047F8 ENT
_PrintSCI H'000045F4 ENT
_PutLCD H'000047A8 ENT
_PutSCI H'000045C4 ENT
_Read H'00003AB4 ENT
_ReadString H'00003AF8 ENT
_Repaint H'000004E4 ENT
_Run H'00000550 ENT
_ScanSCI H'000045E4 ENT
_SetCurrentTime H'00003DE6 ENT
_SetLED H'000011D0 ENT
_SetPermit H'00003E6A ENT
_SleepMSec H'00003F0C ENT
_Start H'000041BE ENT
_Stop H'000041EC ENT
_Thread_Start H'00004284 ENT
_Thread_Toggle H'000042C2 ENT
_Thread_checkAllDelete H'0000420A ENT
_Thread_checkStayAnother H'00004224 ENT
_Thread_getThread H'0000424A ENT
_TurnOpen_Read H'00003C48 ENT
_TurnOpen_Write H'00003C94 ENT
_Up H'0000394A ENT

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

SYMBOL NAME ADDR TYPE

_UpMotor H'00003202 ENT
_WaitDoorChangeLog H'00002E4E ENT
_WaitPositionChangeLog H'000036C6 ENT
_WaitSecond H'00003E48 ENT
_Wait_ms H'00003EDE ENT
_Write H'00003A3E ENT
_WriteString H'00003A7C ENT
__add H'0000908A ENT

```
__allzero                H'00008036        ENT
__calcnpw                H'0000805E        ENT
__ctype                  H'00009876        DAT
__dti                    H'00007A16        ENT
__duchek                 H'00008DBC        ENT
__errno                  H'00FFE82E        DAT
__fmtout                 H'00005B66        ENT
__its                    H'00007DFC        ENT
__log10                  H'00008156        ENT
__lsft                   H'00008E0E        ENT
__lsfts                  H'000081FA        ENT
__mult                   H'00008E60        ENT
__mult64                 H'000084A8        ENT
__pow10                  H'00008FEE        ENT
__pow5                   H'00008272        ENT
__power                  H'00008530        ENT
__rnd                    H'00008692        ENT
__rnext                  H'00FFE82A        DAT
__rsfts                  H'000082A0        ENT
__setsbit                H'0000877C        ENT
__sub                    H'0000831A        ENT
__unpack                 H'000083C6        ENT
_cntrl                   H'00FFE07A        DAT
_countUpNextRun          H'00003FE2        ENT
_delete_                 H'0000418A        ENT
_frexp                   H'00008818        ENT
_getClock                H'00003EFA        ENT
_get_inbufflen           H'0000537E        ENT
_get_outbufflen          H'00005392        ENT
_i_cnt                   H'00FFE016        DAT
_in                      H'00FFE04A        DAT
_initWOVI                H'00004434        ENT
_init_usbbuff            H'000051E2        ENT
_j_cnt                   H'00FFE018        DAT
_main                    H'000002B0        ENT
_memcmp                  H'0000846A        ENT
_memcpy                  H'00007FA0        ENT
_memset                  H'000090D2        ENT
_modf                    H'0000891E        ENT
_new_EV_Status           H'000039E6        ENT
_new_Thread              H'00004002        ENT
_nextRun                 H'00003F9E        ENT
_rand                    H'000053A6        ENT
```


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

```
SYMBOL NAME              ADDR            TYPE

_read_buff               H'00005320      ENT
_read_outbuff            H'000052C2      ENT
_s                       H'00FFE046      DAT
_simu                    H'00FFE14C      DAT
_sprintf                 H'000053E0      ENT
```

_status	H'00FFE1D4	DAT
_strcmp	H'0000543E	ENT
_strcpy	H'00005468	ENT
_strlen	H'00005484	ENT
_th	H'00FFE01A	DAT
_th1	H'00FFE022	DAT
_th101	H'00FFE22A	DAT
_th102	H'00FFE248	DAT
_th111	H'00FFE266	DAT
_th112	H'00FFE284	DAT
_th113	H'00FFE2A2	DAT
_th114	H'00FFE2C0	DAT
_th119	H'00FFE2DE	DAT
_th120	H'00FFE2FC	DAT
_th121	H'00FFE31A	DAT
_th122	H'00FFE338	DAT
_th123	H'00FFE356	DAT
_th130	H'00FFE374	DAT
_th131	H'00FFE392	DAT
_th141	H'00FFE3B0	DAT
_th142	H'00FFE3CE	DAT
_th143	H'00FFE3EC	DAT
_th19	H'00FFE032	DAT
_th20	H'00FFE036	DAT
_th41	H'00FFE03A	DAT
_th42	H'00FFE03E	DAT
_th43	H'00FFE042	DAT
_usb_int	H'00004A28	ENT
_vsprintf	H'000054A0	ENT
_wovi	H'00004430	ENT
_woviClock	H'00FFE1E6	DAT
_woviInit	H'000043A0	ENT
_woviRun	H'00004306	ENT
_woviThreadFirst	H'00FFE1EE	DAT
_woviThreadLast	H'00FFE20C	DAT
_write_buff	H'0000525C	ENT
_write_inbuff	H'00005200	ENT

MAKE Version 5.2 Copyright (c) 1987, 2000 Borland

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

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

Panel.c:

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

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

Timer.c:

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

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

EV_Time.c:

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

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

EV_File.c:

警告 W8004 EV_File.c 63: 'Ret' に代入した値は使われていない(関数 Write)

警告 W8004 EV_File.c 97: 'Ret' に代入した値は使われていない(関数 WriteString)

警告 W8071 EV_File.c 145: 変換によって有効桁が失われる(関数 Read)

警告 W8004 EV_File.c 141: 'Ret' に代入した値は使われていない(関数 Read)

警告 W8004 EV_File.c 182: 'Ret' に代入した値は使われていない(関数 ReadString)

警告 W8004 EV_File.c 208: '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 226: '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 244: '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 262: 'Ret' に代入した値は使われていない(関数 PermitTurnOpen_Write)

警告 W8004 EV_File.c 280: 'Ret' に代入した値は使われていない(関数 TurnOpen_Read)

警告 W8004 EV_File.c 298: 'Ret' に代入した値は使われていない(関数 TurnOpen_Write)

警告 W8066 EV_File.c 322: 実行されないコード(関数 Motor_Write)

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

警告 W8066 EV_File.c 346: 実行されないコード(関数 Motor_Read)

警告 W8066 EV_File.c 361: 実行されないコード(関数 Limit_Read)

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

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

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

EV_UpDown.c:

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

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

EV_OpenClose.c:

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

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

EV_Input.c:

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

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

EV_Controller.c:

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

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

EV_Simulator.c:

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

警告 W8019 EV_Simulator.c 86: コードは効果を持たない(関数 OnSimulator)

警告 W8019 EV_Simulator.c 104: コードは効果を持たない(関数 OnSimulator)

警告 W8019 EV_Simulator.c 122: コードは効果を持たない(関数 OnSimulator)

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

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

main.c:

警告 W8004 main.c 286: 'key' に代入した値は使われていない(関数 Run)

ilink32 /Tpe -L"C:¥borland¥bcc55¥Lib" Panel.obj Timer.obj EV_Time.obj EV_File.obj EV_UpDown.obj EV_OpenClose.obj EV_Input.obj EV_Controller.obj EV_Simulator.obj main.obj c0x32.obj,main.exe,,cw32.lib import32.lib

Turbo Incremental Link 5.00 Copyright (c) 1997, 2000 Borland

MAKE Version 5.2 Copyright (c) 1987, 2000 Borland

```
        del *.obj
        del main.tds
        del main.ilc
        del main.ild
        del main.ilf
        del main.ils
H8/300H ASSEMBLER (Evaluation software) Ver.1.0
 *****TOTAL ERRORS      0
 *****TOTAL WARNINGS    0
H8/300H LINKAGE EDITOR (Evaluation software) Ver.1.0

: OUTPUT c_thread
: PRINT c_thread
: INPUT asmfile, main, EV_Simulator, EV_Controller, EV_Input, EV_OpenClose, EV_UpDown, EV_File,
EV_Time, Timer, Panel, sci, lcd, usb
: LIB c:¥h8¥akic¥c38hab
: START R(0FFE000), P(200), D(91C0), C(9200)
: ROM (D, R)
: EXIT

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

OBJECT CONVERTER COMPLETED
```

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。

==========

asmfile.src について。

リセットベクトル の 転送先ラベル が _start になっています。
ずっと下の方の _start の ラベル から処理を開始して、
 jsr @_main
で C言語の関数main を呼び出しています。

C言語の関数main は、 void main(void); という形で、
main.c に記述があります。

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

リセットベクトル に続く1番から60番までの 割り込みベクトル について、

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

;26 OVI0

_INT_OVI0: .DATA.L _ITU_OVI_0 ;タイマ0割り込み

で、 タイマ0割り込み は、 ラベル_ITU_OVI_0 に転送されます。

ラベル_ITU_OVI_0 から開始して、 スタック 退避をして、

jsr @_InterruptITU0

で、 C言語の関数InterruptITU0 を呼び出しています。

C言語の関数InterruptITU0 は void InterruptITU0(void); という形で、

Timer.h Timer.c に記述があります。

戻ってくると、再び スタック を戻して、 rte です。

ファイルの先頭に、

.IMPORT _main

.IMPORT _InterruptITU0

という記述があり、 C言語の関数 を参照しています。

.EXPORT _EnableInterrupt,_DisableInterrupt

_EnableInterrupt:

andc.b #H'3f,ccr

rts

_DisableInterrupt:

orc.b #H'c0,ccr

rts

で、C言語から

_EnableInterrupt (割り込み許可)

_DisableInterrupt (割り込み禁止)

を呼び出せるようにしています。

C言語からの呼び出し名は、

EnableInterrupt();

DisableInterrupt();

です。

==========

main.c の 関数 Run の ID==31 を見てください。

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

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

ゴールまで80歩です。

There are 80 cells to a GOAL.

<1>コースは'r'ボタンが鞭で<2>コースは'l'ボタンが鞭です。

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

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

スレッドを使用しています。

Thread *th[8]; でオブジェクト宣言しています。

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

この2行は Java で次と同じ意味です。

Thread th[] = new Thread[8];

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

void Repaint(void)

{

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

はそれぞれ Java で次と同じ意味です。

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

```
    ...
}
```

delete_(th[i]); でオブジェクトを消去しています。

この1行は C++ で次と同じ意味です。

delete th[i];

スレッド20が走り始めたら、

0以外の数字キーを押してみてください。

その数字に20を加えた番号のスレッドが、キーを押す度毎に、

起動・消去を繰り返します。

20を含めて、全部スレッドが消去されると、終了です。

これらのスレッドに関する仕様は Timer.c に記述しました。

==========

2階建エレベーターEVについて

使用方法

EV_Simulator にエレベーターが表示されます

EV_Controller にエレベーターの動作が表示されます

EV_Input の使用方法

u キーを押すとエレベーターが2階に上昇して扉が開きます

d キーを押すとエレベーターが1階に下降して扉が開きます

o キーを押すと扉が開きます

c キーを押すと扉が閉じます

s キーを押すと籠が非常停止します

r キーを押すと籠が非常停止から復帰します

Y キーを押すとエレベーターが2階に上昇して扉が開きます

H キーを押すと2階で扉が閉じます

y キーを押すとエレベーターが1階に下降して扉が開きます

h キーを押すと1階で扉が閉じます

籠が無い階で H h キーを押しても籠は動作しません

籠が無い階で Y y キーを押したとき籠の扉が開いていると籠は動作しません

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 N 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

Motor.txt

Motor.txt を開いて s にして上書き保存してください

s はモーター停止を意味します

j はモーター上昇回転開始を意味します

u はモーター低速上昇回転を意味します

U はモーター高速上昇回転を意味します

k はモーター下降回転開始を意味します

d はモーター低速下降回転を意味します

D はモーター高速下降回転を意味します

h はモーター開回転開始を意味します

o はモーター低速開回転を意味します

O はモーター高速開回転を意味します

t はモーター閉回転開始を意味します

c はモーター低速閉回転を意味します

C はモーター高速閉回転を意味します

Limit.txt

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

yynnyynn は籠が下階停止状態で扉が閉停止状態であることを意味します

yynnnynn は籠が下階停止状態で扉が閉低速区域であることを意味します

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

yynnnnyn は籠が下階停止状態で扉が開低速区域であることを意味します

yynnnnyy は籠が下階停止状態で扉が開停止状態であることを意味します

nynnyynn は籠が下階低速区域で扉が閉停止状態であることを意味します

nnnnyynn は籠が中間高速区域で扉が閉停止状態であることを意味します

nnynyynn は籠が上階低速区域で扉が閉停止状態であることを意味します

nnyyyynn は籠が上階停止状態で扉が閉停止状態であることを意味します

nnyynynn は籠が上階停止状態で扉が閉低速区域であることを意味します

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

nnyynnyn は籠が上階停止状態で扉が開低速区域であることを意味します

nnyynnyy は籠が上階停止状態で扉が開停止状態であることを意味します

==========

Linux CentOS6 の GCC を使用するときは、文字コードを utf8 にして

改行コードを <LF>のみ(UNIX) にして名前を付けて保存してください

makefile.mak build.bat asmfile.src linkfile.sub linuxBuild.bash は

複数のファイルを1個のプロジェクトとして

コンパイルするためのファイルです。

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

main.exe をダブルクリックすると、BCC実行ソフトが起動します。

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

linuxStart.bash をダブルクリックすると、

GCC実行ソフト linuxExe が起動します。

著作者:

しのみや　ひでみね

篠宮　　英峰