発明の巻

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。

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 */
#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 */
#include <conio.h> /* kbhit(), getche() */
#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);
```

```
/* 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;
}
```

```
/* 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)
    {
        double woviClockCompare;
        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))
            {
                woviClockCompare = List->preClock + List->setClock;
                if(woviClock >= woviClockCompare)
                {
                    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 */
void woviInit(void)
{
        woviThreadFirst.previous = NULL;
        woviThreadFirst.next = &woviThreadLast;
        woviThreadLast.previous = &woviThreadFirst;
        woviThreadLast.next = NULL;
        return;
}

#ifndef USE_BCC
/* タイマ割り込み用 */
void InitITU(void)
{
        ITU.TSTR = 0x01; /* timer 0 enable */
        ITU.TSNC = 0;
        ITU.TMDR = 0x0;
        ITU.TFCR = 0x0;
        ITU.TOER = 0x0;
        ITU.TOCR = 0xff;
        ITU0.TCR = 0x00; /* 分周なし */
        ITU0.TIOR = 0x88;
```

```
	ITU0.TIER = 0x04; /* オーバーフロー割り込み許可 */
	ITU0.TCNT = 0xffff - 25000; /* 1 msec interval */
	ITU0.GRA = 0;
	ITU0.GRB = 0;
}

/* タイマ割り込み用 */
void InterruptITU0(void)
{
	ITU0.TSR &= 0xfb;
	ITU0.TCNT = 0xffff - 25000;/* 1 msec interval */
	woviClock += 0.001;
	return;
}
#endif

/* タイマ関数 */
void wovi(double threadPerSec)
{
#ifdef USE_BCC
	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)
```

```
{
	woviClock = 0.0;
	/* タイマ割り込み用 */
#ifndef USE_BCC
	InitITU();
	EnableInterrupt();
#endif
	woviInit();
	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{
	int TimeTemp;
	int *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"

/* 時間を表す外部変数宣言 */
extern double woviClock;

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

```
#ifndef USE_BCC
	*This->p_TimeTemp = woviClock;
#else
	*This->p_TimeTemp = clock();
#endif

	/* 戻る */
	return;
}

int GetCurrentTime(struct EV_Time *This)
{
	This->p_TimeTemp = &This->TimeTemp;
#ifndef USE_BCC
	return ((woviClock - *This->p_TimeTemp)/SLEEP_PER_SEC);
#else
	return ((clock() - *This->p_TimeTemp)/CLOCKS_PER_SEC);
#endif
}

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
    Ret = (char) getchar();
#elif USE_MSVS2005
    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())
        {
            key = (char) getche();
        }
        if(key == 'r')
        {
            Cnt.cnt[0]++;
        }
        nextRun(This, (((rand() % 9) + 10) * 30));
        while(kbhit())
        {
            key = (char) getche();
            if(key == NULL)
            {
                break;
            }
        }
#endif
    }
    else if(This->ID == 2)
    {
```

```
            Repaint();
#ifndef USE_BCC
            Cnt.cnt[1]++;
            nextRun(This, (((rand() % 9) + 10) * 100));
#else
            if(kbhit())
            {
                key = (char) getche();
            }
            if(key == 'l')
            {
                Cnt.cnt[1]++;
            }
            nextRun(This, (((rand() % 9) + 10) * 30));
            while(kbhit())
            {
                key = (char) getche();
                if(key == NULL)
                {
                    break;
                }
            }
#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)
    {
        nextRun(This, 1);
#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';
        if(kbhit())
        {
            key = (char) getche();
        }
```

```
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 */
#ifndef USE_BCC
            countUpNextRun(This, 0);
#else
            printf("¥nThread Ready GO! で開始して競馬のコースが8コースありますが、");
            printf("¥n<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。");
            printf("¥nゴールまで80歩です。");
            /* 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
set CurrentDir="%~dp0"
set bccDir="C:\borland\bcc55\Bin"
set akih8asmDir="c:\h8\akiasm"
set akih8cDir="c:\h8\akic"
set akih8usbDir="c:\h8\usb"
C:
set path=%bccDir%;%path%
D:
cd %CurrentDir%
del error.txt
make -f makefile.mak >> error.txt
make -f makefile.mak clean >> error.txt
C:
set path=%akih8cDir%;%akih8asmDir%;%path%
D:
cd %CurrentDir%
cc38h -cpu=300ha -include=%akih8cDir%,%akih8usbDir% Panel.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir%,%akih8usbDir% Timer.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir%,%akih8usbDir% EV_Time.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir%,%akih8usbDir% EV_File.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir%,%akih8usbDir% EV_UpDown.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir%,%akih8usbDir% EV_OpenClose.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir%,%akih8usbDir% EV_Input.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir%,%akih8usbDir% EV_Controller.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir%,%akih8usbDir% EV_Simulator.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir%,%akih8usbDir% main.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir%,%akih8usbDir% usb.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir%,%akih8usbDir% sci.c >> error.txt
cc38h -cpu=300ha -include=%akih8cDir%,%akih8usbDir% 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
```

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 00000F664H _TEXT                  CODE
 0002:00411000 000003670H _DATA                  DATA
 0003:00414670 000000AFCH _BSS                   BSS
 0004:00000000 0000000A4H _TLS                   TLS
```

```
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***     07/08/17 20:22:49
PAGE    1
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 ***     07/08/17 20:22:49
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 ***    07/08/17 20:22:49
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 ***     07/08/17 20:22:49
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 ***     07/08/17 20:22:49
 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*
```

*** 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 *** 07/08/17 20:22:49

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	

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'00003F03   H'00000162
                                 EV_Time                 EV_Time
                H'00003F04   -   H'00004457   H'00000554
                                 Timer                   Timer
                H'00004458   -   H'000045AD   H'00000156
                                 Panel                   Panel
                H'000045AE   -   H'0000467F   H'000000D2
                                 sci                     sci
                H'00004680   -   H'000048BD   H'0000023E
                                 lcd                     lcd
                H'000048BE   -   H'000053AF   H'00000AF2
                                 usb                     usb
                H'000053B0   -   H'000053E9   H'0000003A
                                 rand                    rand
                H'000053EA   -   H'00005447   H'0000005E
                                 sprintf                 sprintf
                H'00005448   -   H'00005471   H'0000002A
                                 strcmp                  strcmp
                H'00005472   -   H'0000548D   H'0000001C
                                 strcpy                  strcpy
                H'0000548E   -   H'000054A9   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'000054AA   -   H'000054D9   H'00000030
                                 vsprintf                vsprintf
                H'000054DA   -   H'000057B3   H'000002DA
                                 addd3                   addd3
                H'000057B4   -   H'000059E5   H'00000232
                                 divd3                   divd3
                H'000059E6   -   H'00005A5F   H'0000007A
                                 dtol3                   dtol3
                H'00005A60   -   H'00005A6B   H'0000000C
                                 eqd3                    eqd3
                H'00005A6C   -   H'00005A7B   H'00000010
                                 ged3                    ged3
                H'00005A7C   -   H'00005AB1   H'00000036
                                 itod3                   itod3
                H'00005AB2   -   H'00005AC1   H'00000010
                                 ltd3                    ltd3
                H'00005AC2   -   H'00005B07   H'00000046
                                 ltod3                   ltod3
                H'00005B08   -   H'00005B25   H'0000001E
```

```
                    mv83                    mv83
                H'00005B26  -  H'00005B4D  H'00000028
                    mvn3                    mvn3
                H'00005B4E  -  H'00005B5B  H'0000000E
                    ned3                   ned3
                H'00005B5C  -  H'00005B7D  H'00000022
                    spregld3                spregld3
                H'00005B7E  -  H'00005BA5  H'00000028
                    spregsv3                spregsv3
                H'00005BA6  -  H'00007953  H'00001DAE
                    _fmtout                 _fmtout
                H'00007954  -  H'00007A0F  H'000000BC
                    cmpd3                   cmpd3
                H'00007A10  -  H'00007A35  H'00000026
                    divl3                  divl3
                H'00007A36  -  H'00007A55  H'00000020
                    mull3                   mull3
                H'00007A56  -  H'00007E3B  H'000003E6
                    _dti                   _dti
                H'00007E3C  -  H'00007FDF  H'000001A4
                    _its                   _its
                H'00007FE0  -  H'00008039  H'0000005A
                    memcpy                  memcpy
                H'0000803A  -  H'00008075  H'0000003C
                    divul3                  divul3
                H'00008076  -  H'0000809D  H'00000028
                    _allzero                _allzero


                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'0000809E  -  H'00008195  H'000000F8
                    _calcnpw                _calcnpw
                H'00008196  -  H'00008239  H'000000A4
                    _log10                  _log10
                H'0000823A  -  H'000082B1  H'00000078
                    _lsfts                 _lsfts
                H'000082B2  -  H'000082DF  H'0000002E
                    _pow5                   _pow5
                H'000082E0  -  H'00008359  H'0000007A
                    _rsfts                 _rsfts
                H'0000835A  -  H'00008405  H'000000AC
                    _sub                   _sub
                H'00008406  -  H'000084A9  H'000000A4
                    _unpack                 _unpack
                H'000084AA  -  H'000084E7  H'0000003E
                    memcmp                  memcmp
```

```
                H'000084E8   -   H'0000856F   H'00000088
                                 _mult64                 _mult64
                H'00008570   -   H'000086D1   H'00000162
                                 _power                  _power
                H'000086D2   -   H'000087BB   H'000000EA
                                 _rnd                    _rnd
                H'000087BC   -   H'00008857   H'0000009C
                                 _setsbit                _setsbit
                H'00008858   -   H'0000895D   H'00000106
                                 frexp                   frexp
                H'0000895E   -   H'00008A93   H'00000136
                                 modf                    modf
                H'00008A94   -   H'00008AB5   H'00000022
                                 dslc3                   dslc3
                H'00008AB6   -   H'00008AD7   H'00000022
                                 dsruc3                  dsruc3
                H'00008AD8   -   H'00008DC5   H'000002EE
                                 muld3                   muld3
                H'00008DC6   -   H'00008E17   H'00000052
                                 _duchek                 _duchek
                H'00008E18   -   H'00008E69   H'00000052
                                 _lsft                   _lsft
                H'00008E6A   -   H'00008FF7   H'0000018E
                                 _mult                   _mult
                H'00008FF8   -   H'00009093   H'0000009C
                                 _pow10                  _pow10
                H'00009094   -   H'000090DB   H'00000048
                                 _add                    _add
                H'000090DC   -   H'0000910B   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'0000910B   H'00008F0C


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'00FFE1C7   H'000001B6
                                    main                    main
                    H'00FFE1C8   -   H'00FFE1D9   H'00000012
                                    EV_File                 EV_File
                    H'00FFE1DA   -   H'00FFE3FD   H'00000224
                                    Timer                   Timer
                    H'00FFE3FE   -   H'00FFE47D   H'00000080
                                    Panel                   Panel
                    H'00FFE47E   -   H'00FFE4CD   H'00000050
                                    sci                     sci
                    H'00FFE4CE   -   H'00FFE50D   H'00000040
                                    lcd                     lcd
                    H'00FFE50E   -   H'00FFE7E1   H'000002D4
                                    usb                     usb
                    H'00FFE7E2   -   H'00FFE81D   H'0000003C
                                    _fmtout                 _fmtout
                    H'00FFE81E   -   H'00FFE821   H'00000004
                                    _rnext                  _rnext
                    H'00FFE822   -   H'00FFE823   H'00000002
                                    _errno                  _errno

* TOTAL ADDRESS *           H'00FFE010   -   H'00FFE823   H'00000814


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

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

                    SYMBOL  NAME                ADDR            TYPE

                    $ADDD$3                     H'000055A4      DAT
                    $CMPD$3                     H'00007954      DAT
                    $DIVD$3                     H'0000584E      DAT
                    $DIVL$3                     H'00007A10      DAT
                    $DIVUL$3                    H'0000803A      DAT
                    $DSLC$3                     H'00008A94      DAT
                    $DSRUC$3                    H'00008AB6      DAT
                    $DTOL$3                     H'000059E6      DAT
                    $EQD$3                      H'00005A60      DAT
                    $GED$3                      H'00005A6C      DAT
                    $ITOD$3                     H'00005A7C      DAT
                    $LTD$3                      H'00005AB2      DAT
                    $LTOD$3                     H'00005AC2      DAT
                    $MULD$3                     H'00008B8E      DAT
```

$MULL$3	H'00007A36	DAT
$MV8$3	H'00005B08	DAT
MVN3	H'00005B26	DAT
NED3	H'00005B4E	DAT
$SUBD$3	H'00005574	DAT
sp_regld3	H'00005B5C	DAT
sp_regsv3	H'00005B7E	DAT
_Checkfmove	H'00003EC6	ENT
_Clear	H'00004458	ENT
_ClearLCD	H'0000479E	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'000049CE	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'00003DFA	ENT
_GetPermit	H'00003EA8	ENT
_GetSCI	H'000045DE	ENT
_GetSW	H'0000121E	ENT
_H8init	H'00001244	ENT
_Init	H'00000E6C	ENT
_InitITU	H'000043DC	ENT

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

SYMBOL NAME	ADDR	TYPE
_InitLCD	H'0000472E	ENT
_InitSCI	H'000045AE	ENT
_InitUSB	H'000048D4	ENT
_InterruptITU0	H'00004412	ENT
_LCDOut4	H'000046D6	ENT
_Limit_Read	H'00003D5E	ENT
_LocateLCD	H'000047DE	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'00004480 ENT
_PrintLCD H'00004802 ENT
_PrintSCI H'000045FE ENT
_PutLCD H'000047B2 ENT
_PutSCI H'000045CE ENT
_Read H'00003AB4 ENT
_ReadString H'00003AF8 ENT
_Repaint H'000004E4 ENT
_Run H'00000550 ENT
_ScanSCI H'000045EE ENT
_SetCurrentTime H'00003DDE ENT
_SetLED H'000011D0 ENT
_SetPermit H'00003E74 ENT
_SleepMSec H'00003F16 ENT
_Start H'000041C8 ENT
_Stop H'000041F6 ENT
_Thread_Start H'0000428E ENT
_Thread_Toggle H'000042CC ENT
_Thread_checkAllDelete H'00004214 ENT
_Thread_checkStayAnother H'0000422E ENT
_Thread_getThread H'00004254 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'00003E52 ENT
_Wait_ms H'00003EE8 ENT
_Write H'00003A3E ENT
_WriteString H'00003A7C ENT
__add H'00009094 ENT

__allzero	H'00008076	ENT
__calcnpw	H'0000809E	ENT
__ctype	H'00009876	DAT
__dti	H'00007A56	ENT
__duchek	H'00008DC6	ENT
__errno	H'00FFE822	DAT
__fmtout	H'00005BA6	ENT
__its	H'00007E3C	ENT
__log10	H'00008196	ENT
__lsft	H'00008E18	ENT
__lsfts	H'0000823A	ENT
__mult	H'00008E6A	ENT
__mult64	H'000084E8	ENT
__pow10	H'00008FF8	ENT
__pow5	H'000082B2	ENT
__power	H'00008570	ENT
__rnd	H'000086D2	ENT
__rnext	H'00FFE81E	DAT
__rsfts	H'000082E0	ENT
__setsbit	H'000087BC	ENT
__sub	H'0000835A	ENT
__unpack	H'00008406	ENT
_cntrl	H'00FFE07A	DAT
_countUpNextRun	H'00003FEC	ENT
delete	H'00004194	ENT
_frexp	H'00008858	ENT
_getClock	H'00003F04	ENT
_get_inbufflen	H'00005388	ENT
_get_outbufflen	H'0000539C	ENT
_i_cnt	H'00FFE016	DAT
_in	H'00FFE04A	DAT
_initWOVI	H'0000443E	ENT
_init_usbbuff	H'000051EC	ENT
_j_cnt	H'00FFE018	DAT
_main	H'000002B0	ENT
_memcmp	H'000084AA	ENT
_memcpy	H'00007FE0	ENT
_memset	H'000090DC	ENT
_modf	H'0000895E	ENT
_new_EV_Status	H'000039E6	ENT
_new_Thread	H'0000400C	ENT
_nextRun	H'00003FA8	ENT
_rand	H'000053B0	ENT

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

SYMBOL NAME	ADDR	TYPE
_read_buff	H'0000532A	ENT
_read_outbuff	H'000052CC	ENT
_s	H'00FFE046	DAT
_simu	H'00FFE146	DAT
_sprintf	H'000053EA	ENT

_status	H'00FFE1C8	DAT
_strcmp	H'00005448	ENT
_strcpy	H'00005472	ENT
_strlen	H'0000548E	ENT
_th	H'00FFE01A	DAT
_th1	H'00FFE022	DAT
_th101	H'00FFE21E	DAT
_th102	H'00FFE23C	DAT
_th111	H'00FFE25A	DAT
_th112	H'00FFE278	DAT
_th113	H'00FFE296	DAT
_th114	H'00FFE2B4	DAT
_th119	H'00FFE2D2	DAT
_th120	H'00FFE2F0	DAT
_th121	H'00FFE30E	DAT
_th122	H'00FFE32C	DAT
_th123	H'00FFE34A	DAT
_th130	H'00FFE368	DAT
_th131	H'00FFE386	DAT
_th141	H'00FFE3A4	DAT
_th142	H'00FFE3C2	DAT
_th143	H'00FFE3E0	DAT
_th19	H'00FFE032	DAT
_th20	H'00FFE036	DAT
_th41	H'00FFE03A	DAT
_th42	H'00FFE03E	DAT
_th43	H'00FFE042	DAT
_usb_int	H'00004A32	ENT
_vsprintf	H'000054AA	ENT
_wovi	H'0000443A	ENT
_woviClock	H'00FFE1DA	DAT
_woviInit	H'000043AA	ENT
_woviRun	H'00004310	ENT
_woviThreadFirst	H'00FFE1E2	DAT
_woviThreadLast	H'00FFE200	DAT
_write_buff	H'00005266	ENT
_write_inbuff	H'0000520A	ENT

S00E0000635F7468726561644D4F54C8
S1130000000002000000022A0000022A0000022A67
S1130010000022A0000022A0000022A0000022A2D
S1130020000022A0000022A0000022A0000022A1D
S10B00300000022A0000022A6D
S1130038000022A0000022A0000022A0000022C03
S1130048000022A0000022A0000022A0000022AF5
S1130058000022A0000022A0000022A0000022AE5
S10B00680000026A0000022AF5
S1130070000022A0000022A0000022A0000022ACD
S1130080000022A0000022A0000022A0000022ABD
S1130090000022A0000022A0000022A0000022AAD
S10B00A00000022A0000022AFD
S11300A80000022A0000022A0000022A0000022A95
S11300B80000022A0000022A0000022A0000022A85
S11300C80000022A0000022A0000022A0000022A75
S10B00D80000022A0000022AC5
S11300E00000022A0000022A0000022A0000022A5D
S10700F00000022ADD
S11302007A0700FFFF107A00000091C07A0100FF17
S1130210E0007A0200FFE0101F92587000086C0B98
S1130220689B0B7140F25E0002B0567001006DF0E6
S113023001006DF101006DF201006DF301006DF439
S113024001006DF501006DF65E004A3201006D7626
S113025001006D7501006D7401006D7301006D7215
S113026001006D7101006D70567001006DF00100A9
S11302706DF101006DF201006DF301006DF40100F9
S11302806DF501006DF65E00441201006D7601000C
S11302906D7501006D7401006D7301006D720100D5
S11302A06D7101006D705670063F547004C0547038
S11302B05E005B7E7A370000000A790B0001193378
S11302C00FF47A0600FFE18819550B5579257FFF56
S11302D04DF85C000F6E5E0045AE5E00472E0D3894
S11302E00D305C000EEA0D380DB05C000EE20D38E7
S11302F0790000025C000ED80D38790000035C0021
S11303000ECE5E0048D47FF472507FF570505E00CD
S113031002A80D3228F117506DF07A000000920008
S113032001006DF05E0045FE0B970B877A0000001D
S1130330920F01006DF05E0048020B9718886EC89B
S113034000036EC800026EC8000168C87A0000FF8F
S1130350E0465E0039E65E00443E7A00000007D0C6
S11303605E003F167900001E5E00400C0F85790089
S1130370001F5E00400C01006FF000040FD05E0010
S113038041C801006F7000045E0041C801006B208A
S11303900000948A01006DF001006B200000948638
S11303A001006DF05E00443A0B970B975E00422EFE
S11303B07920000146D65E0044580FD05E00419478
S11303C07A010000921B0D305E0044807A00000029
S11303D007D05E003F165E00445819550D505C006F
S11303E00E3C0D0D0D5117F10AC16819D90117D132
S11303F0660147540DB80D505C000DD40D5009B083
S11304006DF07A000000922C01006DF00FE05E00A9
S113041053EA0B970B8701006DF67A0000009231C7
S113042001006DF05E0045FE0B970B970FE05E0039
S1130430548E09B00D010FE05E00526601006DF6A7
S11304405E0048020B9740080D380D505C000D808C
S11304500DD017F50FC10AD168980B557925000403

S108046058D0FF785E97
S11304650045EE0D0047165E0045DE17D00D055E0F
S11304750047B268ED0DB10FE05E0052665E0053B2
S1130485880D004738790100400FE05E00532A0DBF
S1130495057A010000923501006DF15E0048020BFB
S11304A59701006DF65E0048020B9701006DF65E3D
S11304B50045FE0B970D510FE05E0052660D28793E
S11304C50000035C000D040D20795000010D021995
S11304D5550B55792527104DF85A0003DA54705EEC
S11304E5005B7E7A0500FFE01219665E0044581929
S11304F5EE400E7A01000092370D605E0044800BDA
S11305055E69501D0E4DEC7A01000092390D605E57
S11305150044807A010000923D0D605E004480191D
S1130525EE400E7A01000092370D605E0044800BA9
S11305355E6F5000021D0E4DEA7A010000923F0DD9
S1130545605E0044805E005B5C54705E005B7E7A97
S1130555370000000A7A03000000017A040000074F
S1130565D019550F8618886EF800036EF800026ED1
S1130575F8000168F869607920000146365C00FFE0
S11305855E7A0000FFE01269010B5169815E005339
S1130595B017F07902000901D053207918000A79C0
S11305A5000064528017F00F810FE05E003FA85AE8
S11305B5000E2269607920000246365C00FF207A2E
S11305C50000FFE01469010B5169815E0053B01708
S11305D5F07902000901D053207918000A79000047
S11305E564528017F00F810FE05E003FA85A000E9A
S11305F52269607920000B586000B601006F600026
S1060605127A2043
S10A06080000000146205E23
S113060F0044587A01000092430D505E0044807AF3
S113061F010000076C0FE05E003FEC5A000E220151
S113062F006F6000127A200000000246265E00442D
S113063F587A01000092540D505E0044807A0100F5
S113064F00925B0D505E0044800FE05E0041F65A4E
S113065F000E2201006F6000127A20000000034693
S113066F1E5E0044587A01000092660D505E0044EE
S113067F807A01000005DC0FE05E003FEC402401AF
S113068F006F6000127A200000000446165E0044DB
S106069F587A0182
S11306A2000092770D505E0044800FE05E0041F639
S11306B25A000E2269607920000C586000A80100DC
S11306C26F6000127A200000000146205E00445849
S11306D27A01000092880D505E0044807A01000086
S11306E20D480FE05E003FEC5A000E2201006F60DE
S11306F200127A200000000246205E0044587A016C
S1130702000092990D505E0044807A0100000D486A
S11307120FE05E003FEC5A000E2201006F600012F0
S11307227A2000000003461E5E0044587A0100004E
S108073292AA0D505EC8
S11307370044807A0100000D480FE05E003FEC4063
S11307471C5E0044587A01000092BB0D505E0044C2
S1130757800FE05E0041F60FE05E0041945A000E01
S113076722269607920000D586000A801006F6000BE
S1130777127A200000000146205E0044587A0100E7
S11307870092C30D505E0044807A01000013EC0F02
S1130797E05E003FEC5A000E2201006F6000127A00
S11307A7200000000246205E0044587A01000092B0

S11307B7D40D505E0044807A01000013EC0FE05E15
S10707C7003FEC5AA6
S11307CB000E2201006F6000127A20000000034626
S11307DB1E5E0044587A01000092E50D505E004402
S11307EB807A01000013EC0FE05E003FEC401C5ECF
S11307FB0044587A01000092F60D505E0044800FBE
S113080BE05E0041F60FE05E0041945A000E226950
S113081B607920000E586000EA01006F6000127AC5
S113082B200000000146205E0044587A010000922C
S113083BFE0D505E0044807A01000017700FE05EDE
S113084B003FEC5A000E2201006F6000127A200069
S10E085B000002463E5E0044587A0194
S11308660000930F0D505E0044807A01000017705C
S11308760FE05E003FEC7A01000093160D505E0018
S1130886448079000000B5E00428E7A01000005DC8D
S11308965E003FEC5A000E2201006F6000127A20C0
S11308A600000003461E5E0044587A0100009321AF
S11308B60D505E0044807A01000017700FE05E0061
S11308C63FEC404001006F6000127A2000000004F4
S11308D646327900000B5E00425401006FF00004BB
S11308E6470A01006F7000045E0041947A0100001C
S11308F693320D505E0044800FE05E0041F60FE038
S10809065E0041945A5C
S113090B000E2269607920001346440FB10FE05E9D
S113091B003FA819DD0DD05C0008F80DD117F10FBE
S113092BF20A92682ADA0117D2660247160DD079BA
S113093B1000145E0042CC7A01000003E80FE05E66
S113094B003FA80B5D792D00044DCA5A000E226996
S113095B607920001446187A01000093390D505E1C
S113096B0044800FC10FE05E003FEC5A000E22697A
S113097B607920001546187A010000933B0D505EF9
S113098B0044800FC10FE05E003FEC5A000E22695A
S113099B607920001646187A010000933D0D505ED6
S10F09AB0044800FC10FE05E003FEC5AD7
S11309B7000E2269607920001746187A0100009318
S11309C73F0D505E0044800FC10FE05E003FEC5ABD
S11309D7000E2269607920001E465E01006F6000E9
S11309E712462401006F6000120B7001006FE000D4
S11309F71228D6E80E38D67A01000003E80FE05E26
S1130A07003FA85A000E2201006F6000127A2000EF
S1130A17000001586004040 1006F6000121B70019D
S1130A27006FE000127FD670007A01000003E80F21
S1130A37E05E003FA85A000E2269607920001F5824
S1130A476003B801006F600012460C1A910FE05E55
S1130A57003FEC5A000E2201006F6000127A20005B
S1130A6700000146247A010000923D0D505E0044C8
S1130A77807A01000093410D505E0044800FC10F3F
S1090A87E05E003FEC5AA3
S1130A8D000E2201006F6000127A20000000024662
S1130A9D3019DD0DD00B505E00400C0DD217F21046
S1130AAD321032010078A06BA000FFE01A0B5D79C4
S1130ABD2D00024DDE0FB10FE05E003FEC5A000E2C
S1130ACD0001006F6000127A200000000346566B90
S1130ADD2000FFE0127920000D4D1A790000016B03
S1130AEDA000FFE01601006F6000120B7001006F94
S1130AFDE0001240246B2000FFE0147920000D4D1F
S1130B0D18790000026BA000FFE01601006F600072

S1130B1D120B7001006FE000120FB10FE05E003F8A
S1130B2DA85A000E0001006F6000127A2000000029
S1130B3D04465A5E0044586B2000FFE016792000EE
S1130B4D01460E7A01000093520D505E0044804021
S1060B5D186B20EF
S1130B6000FFE01679200002460C7A01000093632F
S1130B700D505E00448001006B2000FFE01A5E0010
S1130B80419401006B2000FFE01E5E0041940FC101
S1130B900FE05E003FEC5A000E0001006F60001290
S1130BA07A2000000005461C5E0044587A010000CC
S1130BB0921B0D505E0044800FC10FE05E003FECBE
S1130BC05A000E0001006F6000127A200000000638
S10B0BD0461C5E0044587A0143
S1130BD8000093740D505E0044800FC10FE05E0067
S1130BE83FEC5A000E0001006F6000127A200000EB
S1130BF8000746365E00445819DD0DD07910000B06
S1130C085E00400C0DD217F210321032010078A0AA
S1130C186BA000FFE0220B5D792D00044DDC0FB1C2
S1130C280FE05E003FEC5A000E0001006F600012F7
S1130C387A200000000846245E00422E7920000234
S1130C48460E01006F6000120B7001006FE0001286
S1130C580FB10FE05E003FA85A000E0001006F605D
S1130C6800127A2000000009460C0FC10FE05E0055
S1130C783FEC5A000E0001006F6000127A2000005A
S1130C88000A461C5E0044587A010000921B0D506E
S1040C985EFA
S1130C990044800FC10FE05E003FEC5A000E0001D3
S1130CA9006F6000127A200000000B461C5E0044AE
S1130CB9587A01000093850D505E0044800FC10FDF
S1130CC9E05E003FEC5A000E0001006F6000127AEB
S1130CD9200000000C4634790000135E00400C012B
S1130CE9006BA000FFE0325E0041C8790000145E8A
S1130CF900400C01006BA000FFE0365E0041C80F05
S1130D09B10FE05E003FEC5A000E0001006F600076
S1130D19127A200000000D46305E00422E79200031
S10A0D2903461A01006B20D1
S1130D3000FFE0325E00419401006F6000120B700F
S1130D4001006FE000120FB10FE05E003FA85A00F0
S1130D500E0001006F6000127A200000000E460CA6
S1130D600FC10FE05E003FEC5A000E0001006F6000
S1130D7000127A200000000F461A5E0044587A01E0
S1130D800000921B0D505E0044800FC10FE05E0017
S1130D903FEC406C01006F6000127A2000000010ED
S1130DA0461A5E0044587A01000093960D505E0087
S1130DB044800FC10FE05E003FEC404401006F60D0
S1130DC000127A2000000011463679000295E00E7
S1130DD0400C01006BA000FFE03A7900002A5E009E
S1090DE0400C01006BA0B2
S1130DE600FFE03E7900002B5E00400C01006BA083
S1130DF600FFE0420FE05E00419440206960792 0E5
S1130E06002946187A01000000640FE05E003FA83F
S1130E160FE17A0000FFE04A5E002612696079203E
S1130E26002A461A7A01000000640FE05E003FA81C
S1130E360FE17A0000FFE07A5E001FB2401C696092
S1130E467920002B46140FC10FE05E003FA80FE187
S1130E567A0000FFE1465E00135A7A170000000A83
S1130E665E005B5C54705E005B7E0F86790400094E

S1130E767A0500FFE0126960792000014626190011
S1130E8669D05E0053B017F001D053407918000AB9
S1130E967900001E528017F00F810FE05E003FA815
S1040EA65AEE
S1130EA70010D4696079200002462819006FD0002A
S1130EB7025E0053B017F001D053407918000A7946
S1130EC700001E528017F00F810FE05E003FA85A03
S1130ED70010D4696C792C00034D36792C00084E29
S1130EE73069601B5017F010300A85190069D05E0E
S1130EF70053B017F001D053407918000A79000066
S1130F07C8528017F00F810FE05E003FA85A001008
S1130F17D47A0400004480195569607920000B4690
S1130F271A7A01000093A70D505D407A010000056E
S1130F37DC0FE05E003FEC5A0010D46960792000B3
S1130F470C461A7A01000093AF0D505D407A0100F9
S1130F57000BB80FE05E003FEC5A0010D4696079CC
S1130F6720000D46247A010000923D0D505D407A22
S1040F770175
S1130F78000093B90D505D407A01000011940FE011
S1130F885E003FEC5A0010D469607920000E461ABF
S1130F987A01000093C10D505D407A01000017707B
S1130FA80FE05E003FEC5A0010D47A03000007D02C
S1130FB869607920001446245E0044587A010000D1
S1130FC893CB0D505D407A010000923D0D505D407A
S1130FD80FB10FE05E003FEC5A0010D4696079202E
S1130FE8001546245E0044587A01000093DC0D5036
S1130FF85D407A010000923D0D505D400FB10FE056
S10810085E003FEC5AFD
S113100D0010D4696079200001646245E0044587A96
S113101D01000093ED0D505D407A010000923D0DEE
S113102D505D400FB10FE05E003FEC5A0010D469E4
S113103D607920001746225E0044587A0100009320
S113104DFE0D505D407A010000923D0D505D400F45
S113105DB10FE05E003FEC406E69607920001E46E3
S113106D08F8FF38D438D6405E69607920002946E8
S113107D187A01000000640FE05E003FA87A0000BB
S113108DFFE04A5E002538403E69607920002A461C
S113109D1A7A01000000640FE05E003FA80FE17AA9
S11310AD0000FFE07A5E001E94401C696079200009
S10B10BD2B46140FB10FE05E96
S11310C5003FA80FE17A0000FFE1465E0012705E63
S11310D5005B5C54705E005B7E7A040000448019FB
S11310E5660F8569507920000B46105E0044587AD7
S11310F5010000940F0D605D404044695079200064
S11311050C460C7A010000941A0D605D404030696D
S1131115507920000D460C7A01000094240D605D82
S113112540401C69507920000E46147A0100009254
S11311353D0D605D407A010000942E0D605D4069B0
S1131145507920014461A5E0044587A0100009431
S11311553F0D605D407A010000923D0D605D4040AA
S1121165646950792000154 61A5E0044587A01D8
S1131174000094500D605D407A010000923D0D60C3
S11311845D4040426950792000164 61A5E00445877
S11311947A01000094610D605D407A010000923D84
S11311A40D605D4040206955792500174 6185E009F
S11311B444587A01000094730D605D407A01000085
S11311C4923D0D605D405E005B5C54706DF66DF5A1

S11311D40D067909000128D617500D950CE91A0953
S11311E44B04101540F866050D8846121A0E4B047D
S11311F4101940F80D9029D6148939D640141A0EC3
S11312044B04101940F8795900FF0D9029D616891B
S113121439D60D506D756D7654706DF60D0628D361
S1131224175079010011A0E4B04101140F866108F
S11312344704191140047901000 10D106D765470AF
S1131244F80638ECF8FF38C038C1188838D8F8FFE6
S113125438C8188838DBF8FF38C9F8DF38D0F8FF06
S109126438CDF8F038D18B
S109126AF8FF38D45470B4
S1139200435055204D4F444520253032580A000C19
S113921052656164792133303532004E455854200C
S113922020202020202020202020007377257557
S11392300025730A000C0020003C313E000A003C6C
S1139240323E003C313E3173742020202020202008
S1139250202020003C313E326E64003C313E53748A
S11392606F70202020003C313E3372642020202088
S1139270202020202020003C313E53746F7020209A
S11392802020202020202003C323E3173742020F7
S11392902020202020202020003C323E326E6420FB
S11392A020202020202020202003C323E337264E6
S11392B0202020202020202020202003C323E5374F8
S11392C06F70003C333E3173742020202020202017
S11392D0202020003C333E326E64202020202020BA
S11392E020202020003C333E3372642020202020A5
S10992F0202020202000D5
S11392F63C333E53746F70003C343E31737420200C
S1139306202020202020202020003C343E326E6400A2
S11393163C313E53746172742020003C343E3372F8
S113932664202020202020202020202020003C343E538F
S1139336746F00300031003200330054687265 6187
S113934664205265616479204 74F2100474F414CA1
S1139356213C313E574F4E202020202000474F41CD
S11393664C213C323E574F4E202020202000436F95
S1139376756E74557020202020202020202020005454
S11393866F67676C65202020202020202020202020200086
S113939645562020202020202020202020202020202069
S11393A6003C313E496E6974003C323E496E697435
S11393B62020003C333E496E6974003C343E496EBE
S11393C669742020003C303E496E697420202020B9
S11393D62020202020003C313E496E697420202045
S10993E6202020202020BE
S11393EC003C323E496E6974202020202020202020202E
S11393FC20003C333E496E697420202020202020201D
S113940C2020003C313E44657374726F79003C320A
S113941C3E44657374726F003C333E4465737472DF
S113942C6F003C343E44657374726F7920202020A6
S113943C2020003C303E44657374726F79202020E9
S113944C202020003C313E44657374726F792020D8
S113945C20202020003C323E44657374726F7920C7
S113946C202020202020003C333E44657374726F0F
S113947C792020202020202020000415312D000000E
S105948C0000DB
S11312705E005B7E0F860F957A00000000200AE077
S11312800FD15E003DA27A00000000200AE05E005C
S11312903DDE01006FE600027A00000000060AE06E

S11312A001006FE000087A000000000C0AE0010072
S11312B06FE0000E7A00000000120AE001006FE008
S11312C0001C18886EE8001A7A0500003A347A0088
S11312D0000000320AE05D507A00000000360AE0A8
S11312E05D507A000000003A0AE05D507A00000089
S11312F0003E0AE05D5001006F60000201006DF0E6
S11313007A000000003A0AE07A010000948E5E0041
S11313103AB40B9779200001473A790000096DF040
S113132001006F60001C01006DF07A000000003EB8
S11313300AE07A01000094995E003AF80B970B8754
S1131340792000001470E7A01000000120AE1686863
S11313505C00042A5E005B5C54705E005B7E0F865B
S10913600F9501006F6010
S1131366000E01006DF07A00000000360AE07A01F3
S1131376000094A45E003AB40B976E68000CA87143
S1131386460E5E0044580FD05E0041945A0017285B
S113139601006F60000201006DF07A000000003A60
S11313A60AE07A010000948E5E003AB40B97790046
S11313B600096DF001006F60001C01006DF07A00FA
S11013C60000003E0AE07A01000094995EE9
S11313D3003AF80B970B8701006F60000801006D5B
S11313E3F07A00000000320AE07A01000094B15E53
S11313F3003AB40B976E680006A873461CF84E6D4B
S1131403F07A00000000360AE07A01000094A45E3B
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
S1131585 12A86E4606F8796EE800125A0016FC6832
S1131595 68A86F46366E680016A879587001586EAC
S11315A5680017A879460AF86E6EE800175A001600
S11315B5FC6E680018A86E470E6E680019A86E4683
S11315C506F8796EE800195A0016FC6868A84F46B4
S11315D52E6E680016A8795870011C6E680017A84E
S11315E579587001126E680018A86E4608F8796E6E
S11315F5E8001840066E680019A86E5A0016FC68C4
S1131605 68A86846466E680016A879460AF86E6E9D

S1131615E800165A0016FC6E680017A879460AF802
S11316256E6EE800175A0016FC6E680018A86E4621
S108163508F8796EE8DE
S113163A0018400E6E680019A86E4606F8796EE81F
S113164A00195A0016FC6868A86346346E680019C4
S113165AA8795870009C6E680018A879460AF86E33
S113166A6EE800185A0016FC6E680017A86E470E3B
S113167A6E680016A86E4606F8796EE80016407280
S113168A6868A84346286E680019A87947646E6893
S113169A0018A879475C6E680017A86E4608F8799F
S11316AA6EE8001740066E680016A86E4044686824
S11316BAA874463E6E680019A8794608F86E6EE863
S11316CA0019402E6E680018A8794608F86E6EE86D
S11316DA0018401E6E680017A86E4608F8796EE86F
S11316EA0017400E6E680016A86E4606F8796EE873
S11316FA00167A00000000120AE001006DF07A0079
S113170A0000003E0AE07A01000094995E003A7CE8
S113171A0B977A01000000120AE1686855565E00C9
S108172A5B5C54705EDE
S113172F005B7E7A0500004480790600027A01008F
S113173F0094BD0D605D507A01000094EC0D605D67
S113174F507A010000950D0D605D507A01000095F0
S113175F390D605D507A01000095670D605D507A19
S113176F01000095740D605D505E005B5C54700169
S113177F006DF6790600030C80A0754704A06A4636
S113178F106E180003A87946086E180002A879474F
S113179F14A06B46426E180003A86E463A6E1800EB
S11317AF02A879463219EE7A010000957E0D605E2C
S11317BF0044800B5E792E00044DEC790E00047A01
S11317CF01000095900D605E0044800B5E792E0042
S11317DF0C4DEC5A001E8AA0554704A06A46106EA2
S11317EF180003A86E46086E180002A879473AA09E
S11317FF644704A06B461E6E180003A86E46166E50
S113180F180002A86E460E6E180001A86E460668F1
S109181F18A86E4714A097
S113182573465A6E180003A86E46526E180002A836
S111183579464A19EE7A01000095900D605E27
S11318430044800B5E792E00024DEC790E00027A80
S1131853010000957E0D605E0044800B5E792E00CF
S1131863064DEC790E00067A01000095900D605E3B
S11318730044800B5E792E000C4DEC5A001E8AA0A7
S11318837346686E180003A86E46606E180002A8BC
S11318936E46586E180001A86E46506818A86E4627
S11318A34A19EE7A01000095900D605E0044800BA7
S11318B35E792E00044DEC790E00047A0100009545
S11318C37E0D605E0044800B5E792E00084DEC793B
S11318D30E00087A01000095900D605E0044800BB2
S11318E35E792E000C4DEC5A001E8AA0754704A0A6
S11318F36A461E6E180003A86E46166E180002A8E9
S11319036E460E6E180001A86E46066818A86E4749
S113191328A0444704A06B460E6E180001A879461D
S1131923066818A86E4712A07346586E180001A8DC
S10919337946506818A874
S11319396E464A19EE7A01000095900D605E0044E7
S1121949800B5E792E00064DEC790E00067A01B5
S11319580000957E0D605E0044800B5E792E000AC0
S11319684DEC790E000A7A01000095900D605E0037

S113197844800B5E792E000C4DEC5A001E8AA06A37
S1131988460E6E180001A87946066818A86E471611
S1131998A0644704A06B46406E180001A879463836
S11319A86818A879463219EE7A01000095900D60FF
S11319B85E0044800B5E792E00084DEC790E00081A
S11319C87A010000957E0D605E0044800B5E792EDF
S11319D8000C4DEC5A001E8A6E180003A879586053
S11319E8024E6E180002A87958600244A07346108C
S11319F86E180007A87946086E180006A879472CC0
S1131A08A06F4704A06846106E180007A879460817
S1131A186E180006A8794714A07446426E1800078A
S1131A28A86E463A6E180006A879463219EE7A016E
S1131A38000095A20D605E0044800B5E792E0004C1
S1091A484DEC790E0004D1
S1131A4E7A01000095900D605E0044800B5E792E46
S1131A5E000C4DEC5A001E8AA04F4704A068461096
S1131A6E6E180007A86E46086E180006A879473C44
S1131A7EA0634704A07446206E180007A86E46188C
S1131A8E6E180006A86E46106E180005A86E46085E
S1131A9E6E180004A86E4714A07346426E18000712
S1131AAEA86E463A6E180006A879463219EE7A01E8
S1131ABE000095B40D605E0044800B5E792E000429
S1131ACE4DEC790E00047A01000095900D605E00D6
S1131ADE44800B5E792E000C4DEC5A001E8AA073C7
S1131AEE46526E180007A86E464A6E180006A86E78
S1131AFE46426E180005A86E463A6E180004A86E8C
S1131B0E463219EE7A01000095C60D605E004480E0
S1131B1E0B5E792E00044DEC790E00047A01000061
S1131B2E95900D605E0044800B5E792E000C4DEC9B
S1131B3E5A001C34A06F4704A06846206E18000795
S1131B4EA86E46186E180006A86E46106E1800058D
S1131B5EA86E46086E180004A86E472CA0434704CF
S1131B6EA07446106E180005A87946086E18000476
S1131B7EA86E4714A07346406E180005A879463820
S1131B8E6E180004A86E463019EE7A01000095D83F
S1131B9E0D605E0044800B5E792E00044DEC790ED1
S1071BAE00047A01B1
S1131BB2000095900D605E0044800B5E792E000C50
S1131BC24DEC406EA07346106E180005A8794608C6
S1131BD26E180004A879472CA06846106E180005F9
S1131BE2A87946086E180004A86E4718A063470434
S1131BF2A074463E6E180005A87946366E18000496
S1121C02A879462E19EE7A010000957E0D605EDB
S1131C110044800B5E792E00044DEC790E00047AAA
S1131C2101000095900D605E0044800B5E792E00EB
S1131C310C4DEC5A001E8A6E180001A879586002F7
S1131C41486818A87958600240A07346106E1800BE
S1131C5107A87946086E180006A879472CA06F4794
S1131C6104A06846106E180007A87946086E18008C
S1131C7106A8794714A07446426E180007A86E4659
S1131C813A6E180006A879463219EE7A01000095DA
S1131C91900D605E0044800B5E792E00084DEC7957
S1131CA10E00087A01000095A20D605E0044800BCE
S1131CB15E792E000C4DEC5A001E8AA04F4704A0FA
S1131CC16846106E180007A86E46086E180006A82D
S1131CD179473CA0634704A07446206E180007A807
S1131CE16E46186E180006A86E46106E180005A8F9

S1131CF16E46086E180004A86E4714A07346426E20
S1091D01180007A86E465E
S1131D073A6E180006A879463219EE7A0100009553
S1131D17900D605E0044800B5E792E00084DEC79D0
S1131D270E00087A01000095B40D605E0044800B35
S1131D375E792E000C4DEC5A001E8AA07346526E34
S1131D47180007A86E464A6E180006A86E46426E2C
S1131D57180005A86E463A6E180004A86E46321995
S10D1D67EE7A01000095900D605E16
S1131D710044800B5E792E00084DEC790E00087A41
S1131D8101000095C60D605E0044800B5E792E0054
S1131D910C4DEC5A001E8AA06F4704A06846206EC2
S1131DA1180007A86E46186E180006A86E46106E36
S1131DB1180005A86E46086E180004A86E472CA0EB
S1131DC1434704A07446106E180005A87946086EAF
S1131DD1180004A86E4714A07346406E180005A8A6
S1131DE17946386E180004A86E463019EE7A010060
S1131DF10095900D605E0044800B5E792E00084DC6
S1131E01EC790E00087A01000095D80D605E00445C
S1131E11800B5E792E000C4DEC406EA07346106E64
S1131E21180005A87946086E180004A879472CA064
S1131E316846106E180005A87946086E180004A8B4
S1131E416E4718A0634704A074463E6E180005A8A8
S1131E517946366E180004A879462E19EE7A0100E8
S1091E610095900D605E88
S1131E670044800B5E792E00084DEC790E00087A4A
S1131E77010000957E0D605E0044800B5E792E00A5
S1101E870C4DEC5C00F8A001006D7654706A
S113948E4D6F746F722E74787400004C696D69742D
S113949E2E7478740000436F6D6D616E642E747854
S11394AE74000053616666574792E74787400000A33
S11394BE5550203D202775272C20444F574E203DD5
S11394CE202764272C204F50454E203D20276F2701
S11394DE2C20434C4F5345203D20276327000A453C
S11394EE4D455247454E4359203D202773272C2087
S11394FE5245434F56455259203D20277227000AA5
S113950E31737420466C6F6F722043414C4C203D77
S113951E202779272C20326E6420466C6F6F7220C1
S113952E43414C4C203D20275927000A31737420A8
S113953E466C6F6F7220434C4F5345203D20276876
S113954E272C20326E6420466C6F6F7220434C4F73
S113955E5345203D20274827000A5155495445209D
S113956E3D20277127000A434F4D4D414E443E0087
S109957E0A20202020302A
S1139584303030303030302020202000A2020209A
S1139594202020202020202020202020000A30EA
S11395A4303030202020202020202030303030300064
S11395B40A203030303020202020202030303030303A
S11395C420000A20203030303020202020303030305A
S11395D4302020000A2020203030303020203030304A
S10995E4303020202000BE
S1131E945E005B7E0F860F9501006FE60024010050
S1131EA46F6000245E00318E7A00000000280AE08F
S1131EB45E0032027A00000000300AE05E003464FF
S1131EC47A00000000380AE05E0036C67A0000009B
S1131ED4004C0AE001006FE000705E0029167A00EE
S1131EE4000000740AE05E00298A7A000000007C86

S1131EF40AE05E002BEC7A00000000840AE05E0036
S1131F042E4E7A00000000A20AE00FD15E003DA22B
S1131F147A00000000C00AE05E003A347A00000050
S1131F2400C40AE05E003A347A00000000C80AE004
S1131F345E003A347A00000000B40AE001006FE066
S1131F4400B67A00000000BA0AE001006FE000BCAA
S1131F547A00000000C80AE0F9735E003CD87A00F6
S1131F64000000380AE001006F6100245E0036E0DF
S1131F747A00000000840AE001006F6100705E00D3
S1071F842E767A0038
S1131F88000000C40AE0F94E5E003BC0792000015E
S1131F9847127A00000000C40AE0F9635E003B3090
S1131FA8792000015E005B5C54705E005B7EFC631D
S1131FB8F54EFD730F860F9301006F6000B60100A5
S1131FC86DF07A00000000C00AE07A01000095EA8B
S1041FD85EA7
S1131FD9003AB40B9779200001587004E07A0000A5
S1131FE90000C40AE001006F6100BC5E003B747924
S1131FF9200001587004C66E6800BAA848587003D7
S113200976A8594756A863587002DCA8645870012A
S11320198CA86858700406A86F5870026EA8714797
S11320291AA87258700496A87358700490A8754733
S113203954A8795870013C5A0024C67A000000005C
S1132049C80AE00CD95E003CD80FB05E0041945A2F
S11320590024C65A0024C65A0024C601006F600032
S11320692401006F00000C69007920000146160164
S1132079006F60007001006F00001469007920008F
S113208901587004380 1006F60002401006F0000DB
S1132099146900792000014 61A01006F600070017C
S11320A9006F000014690079200001587002 4C5A2E
S11320B90022A801006F60007001006F00000C6925
S10920C90079200001462E
S11320CF4E7A00000000380AE07A37000000140F40
S11320DFF17A02000000145E005B267A0000000014
S11320EF280AE07A37000000080FF17A0200000097
S11320FF085E005B2601006F6100B601006F600090
S113210F245E00394A7A170000001C5A0024C601C6
S113211F006F60007001006F00000C6901464A7A7E
S113212F00000000840AE07A370000001E0FF17AE6
S113213F020000001E5E005B267A000000007C0A8E
S113214FE07A37000000080FF17A02000000085E02
S113215F005B2601006F6100B601006F6000705EC7
S10D216F0031407A17000000265AE1
S11321790024C601006F60002401006F0000146988
S1132189007920000146160 1006F60007001006F9D
S113219900001469007920000158700320 01006FC1
S11321A960002401006F00000C69007920000146DA
S11321B91A01006F60007001006F00001469007953
S11321C9200001587001345A0022A801006F6000F1
S11321D97001006F00000C690079200001464E7AF6
S11321E900000000380AE07A37000000140FF17A82
S11321F902000000145E005B267A00000000300A2A
S1132209E07A37000000080FF17A02000000085E47
S1132219005B2601006F6100B601006F6000245E58
S11322290039987A170000001C5A0024C601006F70
S1132239600070 01006F00000C6901464A7A0000D2
S1132249000840AE07A370000001E0FF17A0200C9

S113225900001E5E005B267A000000007C0AE07A1B
S108226937000000082E
S113226E0FF17A02000000085E005B2601006F6129
S113227E00B601006F6000705E0031407A170000F7
S113228E00265A0024C601006F60007001006F0023
S113229E00146900792000014 75C7A000000008475
S11322AE0AE07A370000001E0FF17A020000001ECA
S11322BE5E005B267A00000000740AE07A370000A5
S11322CE00080FF17A02000000085E005B26010091
S11322DE6F6100B601006F6000705E0030F25A004D
S11322EE237601006F60007001006F00000C69001F
S11322FE79200001462E7A00000000C80AE00CD9AE
S113230E5E003CD87A00000000C40AE00C595E005F
S113231E3BC07A00000000C40AE00CC95E003B30EB
S104232E5A51
S113232F0024C67A00000000840AE07A3700000018
S113233F1E0FF17A020000001E5E005B267A00007A
S113234F00007C0AE07A37000000080FF17A0200E0
S113235F0000085E005B2601006F6100B601006F8D
S113236F6000705E0031407A17000000265A002487
S113237FC601006F60002401006F0000146900792B
S113238F2000015860013001006F60007001006F81
S113239F00000C690079200001462E7A000000002E
S11323AFC80AE00CD95E003CD87A00000000C40ACA
S11323BFE00C595E003BC07A00000000C40AE00C39
S11323CFC95E003B305A0024C67A00000000840A1D
S11323DFE07A370000001E0FF17A020000001E5E44
S11323EF005B267A000000007C0AE07A37000000C9
S11323FF080FF17A02000000085E005B2601006FF0
S113240F6100B601006F6000705E0031407A170003
S107241F0000265A36
S11324230024C601006F60002401006F00000C69E3
S1132433007920000158600 8A01006F6000700179
S1132443006F00000C690079200001462C7A00001C
S11324530000C80AE00CD95E003CD87A00000000F3
S1132463C40AE00C595E003BC07A00000000C40AB2
S1132473E00CC95E003B30404A7A00000000840A46
S1132483E07A370000001E0FF17A020000001E5E9F
S1132493005B267A000000007C0AE07A3700000024
S11324A3080FF17A02000000085E005B2601006F4B
S10D24B36100B601006F6000705E67
S11224BD0031407A17000000265E005B5C54700C
S10F95EA5361666574792E747874000078
S11324CC5E005B7E1B970FF40C8D18886EC800039F
S11324DC6EC800026EC8000168C819660D605E0004
S11324EC121E0D0E0D6117F10AC16819D90117D10E
S11324FC660147260D600C004620A800470EA80174
S113250C470EA802470EA803470E400EFD75400A5E
S113251CFD644006FD6F4002FD630B5679260004F3
S113252C4DBA0CD80B975E005B5C54705E005B7EFF
S113253C0F8618886EE800066EE800077A00000024
S113254C00080AE001006FE0000A7A000000000EA8
S113255C0AE001006FE0001818886EE8001601000D
S113256C6FE600027A0500003A347A000000001C82
S113257C0AE05D507A00000000200AE05D507A000A
S113258C000000240AE05D507A000000002C0AE0F1
S113259C5D5001006F60000201006DF07A000000D5

S11325AC001C0AE07A01000095F65E003AB40B9722
S10925BC79200001474AEB
S11325C201006F60000A01006DF07A000000002C28
S11325D20AE07A01000096025E003AB40B97792072
S11325E200014726790000096DF001006F600018B1
S11325F201006DF07A00000000280AE07A01000071
S1132602960D5E003AF80B970B875E005B5C547085
S11326125E005B7EFD4E0F860F946E6800065C00C3
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
S11328AE0000963D0DD05E00448040566868A868CF
S11328BE461AF85968E86DF07A000000001C0AE029
S11328CE7A01000096185E003A3E0B877A000000EC
S11328DE00200AE06E6900065E003BC07A0000002D
S11328EE00200AE00CD95E003B3040167A0000004F

S11328FE00200AE06E6900065E003BC00FC05E005A
S10B290E41945E005B5C547010
S11395F65361666574792E74787400004D6F746FC9
S1139606722E74787400004C696D69742E747874C4
S113961600005361666574792E747874000A4120DC
S11396264261736B65742069736E2774203173749A
S113963620466C6F6F72000A41204261736B65743A
S11396462069736E277420326E6420466C6F6F72C6
S10496560010
S113291601006DF60F86190069E06FE000026FE0B3
S113292600046FE0000601006FE600087A0000006D
S113293600020AE001006FE0000C7A0000000004C8
S11329460AE001006FE000107A00000000060AE0CA
S113295601006FE0001419006FE000187A00000010
S113296600180AE001006FE0001A19006FE0001E6C
S11329767A000000001E0AE001006FE0002001005B
S11329866D76547001006DF60F865E003A347A0058
S1132996000000040AE05E003A3401006D765470CC
S11329A65E005B7E0F850F947A06000000180AF618
S11329B601006F60001469007920000146520100 8E
S11329C66F60001A6901462801006F66001A7900D4
S11329D6000169E07A00000000040AD0F9735E0082
S11329E63CD87A0100009658790000015E004480C5
S11329F66848A859586001E8F87268C86DF07A010A
S1092A060000965D0FD0F5
S1132A0C5E003A3E0B875A002BE6FB6801006F60B1
S1132A1C001069007920000146 6E01006F60001AF6
S1132A2C1911698101006F6000206901462C0100B6
S1132A3C6F6600207900000169E07A000000000451
S1132A4C0AD0F96F5E003CD87A01000096697900D0
S1132A5C00015E0044805A002BE66848A859586070
S1132A6C01787A00000000040AD00CB95E003CD84F
S1132A7CF87268C86DF07A010000965D0FD05E00A5
S1132A8C3A3E0B875A002BE601006F600008690180
S1132A9C466E01006F60001A1911698101006F60A5
S1132AAC00206901462C01006F66002079000001AB
S1102ABC69E07A00000000040AD0F94F5EC3
S1132AC9003CD87A010000966E790000015E00444B
S1132AD9805A002BE66848A859586001007A00001B
S1132AE90000040AD00CB95E003CD8F87268C86DBE
S1132AF9F07A010000965D0FD05E003A3E0B875ACB
S1132B09002BE601006F60000C6901466C01006F40
S1132B1960001A1911698101006F6000206901467B
S1132B292C01006F6600207900000169E07A00003A
S1132B390000040AD0F96F5E003CD87A01000096C0
S1132B4969790000015E0044805A002BE66848A8B1
S1132B5959586000887A00000000040AD00CB95E55
S1132B69003CD8F87268C86DF07A010000965D0FD1
S1132B79D05E003A3E0B87406401006F60000C6928
S1132B8900792000014656 01006F60001A19116986
S1132B998101006F6000206901462801006F66000A
S1132BA9207900000169E07A00000000040AD00CD2
S1082BB9B95E003CD8E9
S1052BBE7A0197
S1132BC00000967A790000015E0044806848A859A5
S1132BD04614F87268C86DF07A010000965D0FD054
S1132BE05E003A3E0B875E005B5C547001006DF63D

S1132BF00F865E003A347A00000000040AE05E00AB
S1132C003A3401006D7654705E005B7E0F850F943D
S1132C107A06000000180AF601006F60000C6900D4
S1132C2079200001465201006F60001A69014628AD
S1132C3001006F66001A7900000169E07A00000064
S1132C4000040AD0F9735E003CD87A01000096585C
S1132C50790000015E0044806848A859586001E883
S1132C60F87268C86DF07A010000965D0FD05E00BF
S1132C703A3E0B875A002E48FB7401006F60000830
S1132C80690079200001466E01006F60001A191176
S1132C90698101006F6000206901462C01006F66A5
S1132CA000207900000169E07A00000000040AD0E6
S1092CB0F9635E003CD84D
S1122CB67A01000096857900000015E0044805A80
S1132CC5002E486848A859586001787A000000002A
S1132CD5040AD00CB95E003CD8F87268C86DF07A66
S1132CE5010000965D0FD05E003A3E0B875A002E19
S1132CF54801006F6000106901466E01006F6000B6
S1132D051A1911698101006F6000206901462C01C0
S1132D15006F6600207900000169E07A0000000079
S1132D25040AD0F9435E003CD87A010000968B79FA
S1132D350000015E0044805A002E486848A859588F
S1132D456001007A00000000040AD00CB95E003C63
S1132D55D8F87268C86DF07A010000965D0FD05EF1
S1132D65003A3E0B875A002E4801006F6000146934
S1132D7501466C01006F60001A1911698101006F2A
S1132D856000206901462C01006F66002079000070
S1132D950169E07A00000000040AD0F9635E003C93
S1132DA5D87A01000096857900000015E0044805AB7
S1132DB5002E486848A859586000887A000000002A
S1132DC5040AD00CB95E003CD8F87268C86DF07A75
S1132DD5010000965D0FD05E003A3E0B874064010B
S1132DE5006F60001469007920000146560100 6FE9
S1132DF560001A1911698101006F6000206901469D
S1132E052801006F6600207900000169E07A00005F
S1132E150000040AD00CB95E003CD87A0100009684
S1132E2598790000015E0044806848A8594614F863
S1132E357268C86DF07A010000965D0FD05E003AA6
S1132E453E0B875E005B5C547001006DF60F867A5E
S1132E5500000000040AE001006FE000067A0000AC
S1132E6500000E0AE001006FE0001801006D7654C2
S1132E75705E005B7E0F860F957A000000000E0AD8
S1132E85E001006FE000187A000000000A0AE00183
S1132E95006F6100185E003D5E6E680012A87946FA
S1092EA506790000014064
S1132EAB0219006FE0001C7A0400003EC66F60003D
S1132EBB1C6DF001006F51002001006F50000C5D81
S1132ECB400B876E680013A8794606790000014012
S1132EDB0219006FE0001C6DF001006F510020011F
S1132EEB006F5000085D400B876E680014A879468D
S1132EFB0679000001400219006FE0001C6DF00120
S1132F0B006F51002001006F5000105D400B876E66
S1132F1B680015A879460679000001400219006F75
S1132F2BE0001C6F66001C6DF601006F5100200161
S1132F3B006F5000145D400B875E005B5C54705E4A
S1132F4B005B7E0F860F957A000000000E0AE001EE
S1132F5B006FE000187A000000000A0AE001006F1E

S1132F6B6100185E003D5E6E680012A87946067913
S1132F7B000001400219006FE0001C7A0400003EC0
S1132F8BC66F60001C6DF001006F51002001006FD4
S1082F9B50000C5D4035
S1132FA00B876E680013A879460679000000140027A
S1132FB019006FE0001C6DF001006F51002001004B
S1132FC06F5000085D400B876E680014A8794606B1
S1132FD079000001400219006FE0001C6DF0010050
S1132FE06F51002001006F5000105D400B876E6829
S1132FF00015A879460679000001400219006FE028
S1133000001C6F66001C6DF601006F51002001006B
S11230106F5000145D400B875E005B5C54705E75
S113301F005B7E0F860F957A000000000E0AE00119
S113302F006FE000187A000000000A0AE001006F49
S113303F6100185E003D5E6E680012A8794606793E
S113304F000001400219006FE0001C7A0400003EEB
S113305FC66F60001C6DF001006F51002001006FFF
S113306F50000C5D400B876E680013A879460679F4
S113307F000001400219006FE0001C6DF001006FAA
S113308F51002001006F5000085D400B876E6800F0
S113309F14A879460679000001400219006FE00079
S11330AF1C6DF001006F51002001006F5000105D87
S11330BF400B876E680015A879460679000001401A
S11330CF0219006FE0001C6F66001C6DF601006FA4
S11330DF51002001006F5000145D400B875E005BB1
S11330EF5C54705E005B7E0F860F9501006F60006E
S11330FF14690146360FE07A37000000240FF17A86
S109310F02000000245E33
S1133115005B267A000000003C0AF00FD15C00F842
S1133125807A17000000247A00000000200AF00FBF
S1133135E15C00FE105E005B5C54705E005B7E0F1D
S1133145860F9501006F60000C690146360FE07A22
S113315537000000240FF17A02000000245E005BB3
S1133165267A000000003C0AF00FD15C00FA947A3D
S113317517000000247A00000000200AF00FE15C2C
S10C318500FE965E005B5C5470D1
S113965853544F50005361666574792E74787400BF
S1139668004F50454E004F50454E2053706565647A
S113967879004F50454E20537461727400434C4F28
S1139688534500434C4F5345205370656564790037
S10F9698434C4F5345205374617274001F
S113318E01006DF60F86190069E06FE000026FE033
S113319E00046FE0000601006FE600087A000000ED
S11331AE00020AE001006FE0000C7A000000000448
S11331BE0AE001006FE000107A00000000060AE04A
S11331CE01006FE0001419006FE000187A00000090
S11331DE00180AE001006FE0001A19006FE0001EEC
S11331EE7A000000001E0AE001006FE000200100DB
S11331FE6D76547001006DF60F865E003A347A00D8
S113320E000000040AE05E003A3401006D7654704B
S113321E5E005B7E0F850F947A06000000180AF697
S113322E01006F60001469007920000146520100 0D
S113323E6F60001A6901462801006F66001A790053
S113324E000169E07A00000000040AD0F9735E0001
S113325E3CD87A01000096A4790000015E004480F8
S113326E6848A859586001E8F87268C86DF07A0189
S109327E000096A90FD029

S11332845E003A3E0B875A00345EFB6A01006F60AE
S1133294001069007920000 1466E01006F60001A76
S11332A41911698101006F6000206901462C010036
S11332B46F6600207900000169E07A0000000004D1
S11332C40AD0F9755E003CD87A01000096B57900FE
S11332D400015E0044805A00345E6848A85958606F
S11332E401787A00000000040AD00CB95E003CD8CF
S11332F4F87268C86DF07A01000096A90FD05E00D9
S11333043A3E0B875A00345E01006F60000869017E
S1133314466E01006F60001A1911698101006F6024
S113332400206901462C01006F660020790000012A
S110333469E07A00000000040AD0F9555E3C
S1133341003CD87A01000096B8790000015E004480
S1133351805A00345E6848A859586001007A000019
S11333610000040AD00CB95E003CD8F87268C86D3D
S1133371F07A01000096A90FD05E003A3E0B875AFE
S113338100345E01006F60000C6901466C01006F3F
S113339160001A1911698101006F600020690146FB
S11333A12C01006F6600207900000169E07A0000BA
S11333B10000040AD0F9755E003CD87A010000963A
S11333C1B5790000015E0044805A00345E6848A864
S11333D159586000887A00000000040AD00CB95ED5
S11333E1003CD8F87268C86DF07A01000096A90F05
S11333F1D05E003A3E0B87406401006F60000C69A8
S1133401007920000 1465601006F60001A19116905
S11334118101006F60002069014628 01006F660089
S1133421207900000169E07A00000000040AD00C51
S1083431B95E003CD868
S10534367A0116
S1133438000096C2790000015E0044806848A859DC
S11334484614F87268C86DF07A01000096A90FD087
S11334585E003A3E0B875E005B5C547001006DF6BC
S11334680F865E003A347A00000000040AE05E002A
S11334783A3401006D7654705E005B7E0F850F94BD
S11334887A06000000180AF601006F60000C690054
S1133498792000 01465201006F60001A690146282D
S11334A801006F66001A7900000169E07A000000E4
S11334B800040AD0F9735E003CD87A01000096A490
S11334C8790000015E0044806848A859586001E803
S11334D8F87268C86DF07A01000096A90FD05E00F3
S11334E83A3E0B875A0036C0FB6B01006F60000839
S11334F8690079200001466E01006F60001A1911F6
S1133508698101006F6000206901462C01006F6624
S11335180020790000 0169E07A00000000040AD065
S1093528F9645E003CD8CB
S112352E7A01000096CB790000015E0044805AB9
S113353D0036C06848A859586001787A0000000029
S113354D040AD00CB95E003CD8F87268C86DF07AE5
S113355D01000096A90FD05E003A3E0B875A003644
S113356DC001006F6000106901466E01006F6000BD
S113357D1A1911698101006F6000206901462C0140
S113358D006F6600207900000169E07A00000000F9
S113359D040AD0F9445E003CD87A01000096D07934
S11335AD0000015E0044805A0036C06848A859588F
S11335BD6001007A00000000040AD00CB95E003CE3
S11335CDD8F87268C86DF07A01000096A90FD05E25
S11335DD003A3E0B875A0036C001006F6000146934

S11335ED01466C01006F60001A1911698101006FAA
S11335FD6000206901462C01006F660020790000F0
S113360D0169E07A00000000040AD0F9645E003C11
S113361DD87A01000096CB790000015E0044805AF0
S113362D0036C06848A859586000887A0000000029
S113363D040AD00CB95E003CD8F87268C86DF07AF4
S113364D01000096A90FD05E003A3E0B874064013E
S113365D006F60001469007920000146560 1006F68
S113366D60001A1911698101006F6000206901461C
S113367D2801006F6600207900000169E07A0000DF
S113368D0000040AD00CB95E003CD87A0100009604
S113369DDC790000015E0044806848A8594614F89F
S11336AD7268C86DF07A01000096A90FD05E003ADA
S11336BD3E0B875E005B5C547001006DF60F867ADE
S11336CD00000000040AE001006FE0000E01006D30
S11336DD7654705E005B7E0F860F957A00000000B6
S11336ED040AE001006FE0000E01006F61000E0F90
S11336FDE05E003D5E6E680004A879460679000021
S113370D01400219006FE000127A0400003EC66FFB
S108371D6000126DF0D5
S113372201006F51002001006F50000C5D400B87B8
S11337326E680005A879460679000001400219006 7
S11337426FE000126DF001006F51002001006F5015
S113375200085D400B876E680006A879460679006B
S1133762000 1400219006FE000126DF001006F5179
S1133772002001006F5000105D400B876E68000748
S1133782A879460679000001400219006FE0001291
S11337926F6600126DF601006F51002001006F5039
S11337A200145D400B875E005B5C54705E005B7EC1
S11337B20F860F957A00000000040AE001006FE013
S11337C2000E01006F61000E0FE05E003D5E6E6849
S11337D20004A879460679000001400219006FE04F
S11337E200127A0400003EC66F6000126DF0010001
S11337F26F51002001006F50000C5D400B876E6813
S11338020005A879460679000001400219006FE01D
S107381200126DF040
S113381601006F51002001006F5000085D400B87C7
S11338266E680006A879460679000001400219007 1
S11338366FE000126DF001006F51002001006F5020
S113384600105D400B876E680007A879460679006D
S1133856000 1400219006FE000126F6600126DF658
S113386601006F51002001006F5000145D400B876B
S10A38765E005B5C54705E11
S113387D005B7E0F860F957A00000000040AE001BD
S113388D006FE0000E01006F61000E0FE05E003D62
S113389D5E6E680004A879460679000001400219 9E
S11338AD006FE000127A0400003EC66F6000126DD7
S11338BDF001006F51002001006F50000C5D400BB3
S11338CD876E680005A879460679000001400219 44
S11338DD006FE000126DF001006F51002001006FC9
S11338ED5000085D400B876E680006A8794606797F
S11338FD000001400219006FE000126DF001006F2E
S113390D51002001006F5000105D400B876E680061
S113391D07A879460679000001400219006FE000FF
S113392D126F6600126DF601006F51002001006FDA
S113393D5000145D400B875E005B5C54705E005B52
S113394D7E0F860F9501006F600014690146360FD7

S113395DE07A37000000240FF17A02000000245EA4
S108396D005B267A0057
S11339720000003C0AF00FD15C00F8A07A170000A7
S113398200247A00000000200AF00FE15C00FE1C14
S11339925E005B5C54705E005B7E0F860F950100D8
S11339A26F60000C690146360FE07A37000000248D
S11339B20FF17A02000000245E005B267A00000009
S11339C2003C0AF00FD15C00FAB47A17000000241D
S11339D27A00000000200AF00FE15C00FE9C5E000A
S10739E25B5C547063
S11396A453544F50005361666574792E7478740073
S11396B400555000555020537065656479005502A
S11396C420537461727400444F574E00444F574EF5
S11396D420537065656479004 44F574E2053746179
S10696E47274009A
S11339E601006DF60F867A0000FFE1C8010069E069
S11339F67A0600FFE1C8F87268E87A00000000065C
S1133A060AE001006FE000027A01000096E8010077
S1133A166F6000025E005472F8736EE8000FF84E92
S1133A266EE800106EE8001101006D7654700F8188
S1133A361A800100699054705E005B7E0F946E7D60
S1133A4600197A0000FFE1C86849A9434716A94D42
S1133A56470CA9504714A9534614688D40106E8D20
S1133A66000F400A6E8D001040046E8D0011190080
S1133A765E005B5C547001006DF60F966869A94C95
S1133A8646247A0600FFE1C87A00000000060AE031
S1133A9601006FE0000201006F71000801006F6012
S1133AA600025E005472190001006D7654705E00C8
S1133AB65B7E0F9401006F7500187A0000FFE1C862
S1133AC66849A9434716A94D470CA9504716A95358
S1093AD646186808400ACF
S1133ADC6E08000F40046E08001068D840066E088C
S1133AEC001168D819005E005B5C547001006DF620
S1133AFC0F966869A94C46247A0600FFE1C87A0040
S1133B0C000000060AE001006FE0000201006F6193
S1133B1C000201006F7000085E005472190001006E
S1133B2C6D7654705E005B7E0F850C9E6DF67A018C
S1133B3C000096F20FD05C00FEF80B870C00461CBD
S1133B4CA8004714A80146147A01000097057900D0
S1133B5C00015E00448040041966400479060001AC
S10C3B6C0D605E005B5C54705EA9
S1133B75005B7E0F850F9601006DF67A01000097B5
S1133B85140FD05C00FF280B970C004624A80147AF
S1133B9506A8024716401A7A01000097217900000A
S1133BA5015E00448079060001400879060002406 1
S1133BB50219660D605E005B5C54705E005B7E0FF0
S1133BC5850C9E6DF67A01000097140FD05C00FEFC
S1133BD5680B870C00461CA8004714A80146147AF5
S1133BE501000097057900000 15E00448040041937
S1133BF5664004790600010D605E005B5C54705EEF
S1133C05005B7E0F850C9E6DF67A01000097300FE1
S1133C15D05C00FE240B870C00461CA8004714A8A3
S1133C250146147A01000097057900000 15E0044FE
S1133C358040041966400479060001 0D605E005B4F
S1133C455C54705E005B7E0F850F9601006DF67AFE
S1043C55016A
S1133C56000097440FD05C00FE540B970C004624DB

S1133C66A8014706A8024716401A7A0100009721C1
S1133C76790000015E0044807906000014008790658
S1133C860002400219660D605E005B5C54705E00C4
S1133C965B7E0F850C9E6DF67A01000097440FD06C
S1133CA65C00FD940B870C00461CA8004714A80172
S1133CB646147A0100009705790000015E004480EE
S1133CC6400419664004790600010D605E005B5CE2
S1133CD654705E005B7E0F850C9E6DF67A010000C4
S1133CE697520FD05C00FD500B870C004616A800B8
S1133CF64712A801460E7A01000970579000001D4
S1133D065E0044805E005B5C54705E005B7E1B87D6
S1133D160F8518886EF800017A06000000010AF67E
S1133D2601006DF67A01000097520FD05C00FD7E0C
S1133D360B970C004618A800470EA8014710A802C7
S1093D46470C400A40088F
S1103D4C40066E780001400218880B875E68
S1133D59005B5C54705E005B7E0F850F96790000F3
S1133D69096DF001006DF67A010000975D0FD05CD3
S1133D7900FD7C0B970B870C004618A8014706A882
S1133D89024710400E7A0100009721790000015E75
S10C3D990044805E005B5C547081
S11396E879796E6E79796E6E00005065726D697462
S11396F8436F6D6D616E642E74787400000A57723F
S11397086974696E67204572726F7200436F6D6D7D
S1139718616E642E74787400000A52656164696E20
S113972867204572726F72005065726D6974547563
S1139738726E4F70656E2E74787400005475726E75
S11397484F70656E2E74787400004D6F746F722EAF
S113975874787400004C696D69742E747874000011
S1133DA25E005B7E0F860F95190069E001006FE6E6
S1133DB200020FE055267A00000000060AE0010027
S1133DC26FE0000819006FE0000619006FE0000CB5
S1133DD201006FE5000E5E005B5C54700F81010011
S1133DE26F9100027A0000FFE1DA5E0059E60100FA
S1133DF26F110002699054705E005B7E7A37000097
S1133E0200180F8601006FE6000201006F610002D5
S1133E1269117A00000000100AF05E005A7C7A01F0
S1133E2200FFE1DA0F827A00000000080AF05E0068
S1133E3255740F817A02000097680FF05E00584EA6
S1133E425E0059E67A17000000185E005B5C54704E
S1133E525E005B7E0F850D16790E03E852E617F6B8
S1133E620FE101006F50000E5E003FA85E005B5C35
S1133E72547001006DF60F867A00000000060AE016
S1133E8201006FE0000879210001460A7900000170
S1093E926FE00006400A88
S1133E980D11460619006FE0000601006D76547097
S1133EA801006DF60F867A00000000060AE00100A3
S1133EB86FE000086F60000601006D7654705E00C5
S1133EC85B7E0F860F9569606F7100181D10470A96
S1133ED8190069D06F70001869E05E005B5C54706C
S1133EE85E005B7E0F850D1617F60FE101006F501C
S10F3EF8000E5E003FA85E005B5C54708F
S10B976840CF400000000000A7
S1133F047A0000FFE1DA01006F7100045E005B08D0
S1133F1454705E005B7E7A37000000200F867A02BD
S1133F24000000180AF201006DF255D40B970FE15B
S1133F340FF05E005AC20F817A02000097807A0064

S1133F44000000100AF05E00584E7A0000000018CA
S1133F540AF07A01000000080AF15E005B084010D1
S1133F647A02000000080AF201006DF255920B97E1
S1133F747A01000000180AF17A02000000100AF224
S1133F840FF05E0055A40F817A00000000080AF0C8
S1133F945E005AB20D0046C87A17000000205E0086
S1133FA45B5C54705E005B7E1B971B970F860F95BB
S1133FB47A02000000020AE201006DF25C00FF4095
S1133FC40B970FD10FF05E005AC20F817A020000E3
S1133FD497807A000000000A0AE05E00584E0B97AF
S1133FE40B975E005B5C54705E005B7E0F860F95DF
S1073FF40FE055B0D2
S1133FF801006F6000120B7001006FE000125E0099
S11340085B5C54705E005B7E0D057A0400FFE1E2A1
S1134018400601006F44001A01006F40001A0100B6
S11140286F01001A46EC79250001460A7A065C
S113403600FFE21E5A00411A79250002460A7A0653
S113404600FFE23C5A00411A7925000B460A7A061C
S113405600FFE25A5A00411A7925000C460A7A06ED
S113406600FFE2785A00411A7925000D460A7A06BE
S113407600FFE2965A00411A7925000E460A7A068F
S113408600FFE2B45A00411A79250013460A7A065C
S108409600FFE2D25A15
S113409B00411A7925001446087A0600FFE2F04026
S11340AB6E7925001546087A0600FFE30E4060790A
S11340BB25001646087A0600FFE32C4052792500AB
S11340CB1746087A0600FFE34A40447925001E464B
S11340DB087A0600FFE36840367925001F46087A05
S11340EB0600FFE3864028792500294 6087A060057
S11340FBFFE3A4401A7925002A46087A0600FFE35A
S113410BC2400C7925002B46067A0600FFE3E00F2D
S113411BE6461C7A010000977019005E0044807A12
S113412B010000977219005E0044801A804054010D
S113413B006FE4001601006F40001A01006FE000EE
S112414B1A01006F86001601006FC6001A7A0072
S113415A000097887A01000000020AE15E005B080A
S113416A7A00000097907A010000000A0AE15E00D3
S113417A5B0869E51A8001006FE000120FE05E0038
S113418A0E6C0FE05E005B5C547001006DF60F86E7
S113419A5E0010DA01006F60001601006F61001AF9
S11341AA01006F81001A01006F60001A01006F613C
S11341BA001601006F81001601006D7654705E00CF
S11341CA5B7E0F867A0200FFE1DA01006DF25C0082
S11341DAFD280B977A0000FFE1DA7A01000000025A
S11341EA0AE15E005B085E005B5C547001006DF6D9
S11341FA0F867A00000097987A01000000020AE10C
S113420A5E005B0801006D76547001006B2000FFAD
S113421AE1FC01006F01001A460419005470790089
S113422A0001547001006B2100FFE1FC01006F11D2
S113423A001A1988400C01006F11001A0D800B50E7
S109424A0D080F9146F080
S10F42500D8054706DF50D0501006B200E
S113425C00FFE1FC01006F01001A472001006B21F4
S113426C00FFE1FC69101D5046040F904010010043
S113427C6F11001A01006F10001A46E81A806D7551
S113428C54705E005B7E0D0555BE0F86460E0D50B9
S113429C5C00FD6C0F865C00FF22401C7A00000062

S11342AC00020AE07A01000097985E005A600D0044
S11342BC47060FE05C00FF040FE05E005B5C54708C
S11342CC5E005B7E0D0555800F86460E0D505C001F
S11342DCFD2E0F865C00FEE440247A0000000002F1
S11342EC0AE07A01000097985E005A600D004708B7
S11342FC0FE05C00FEC640060FE05C00FE8A5E0029
S113430C5B5C54705E005B7E1B971B977A0500FF0A
S113431CE1E201006F56001A407201006F65001A4A
S113432C7A00000000020AE07A01000097885E0020
S113433C5B4E0D0047547A00000000020AE07A013C
S108434C000097985EDC
S1134351005B4E0D00473E7A01000000020AE17A3C
S1134361020000000A0AE20FF05E0055A47A000081
S1134371FFE1DA0FF15E005A6C0D0047187A000075
S1134381FFE1DA7A01000000020AE15E005B080F37
S1134391E05E0005500FD601006F60001A46860BE0
S11343A1970B975E005B5C54701A8001006BA00051
S11343B1FFE1F87A0000FFE20001006BA000FFE1DA
S11343C1FC7A0000FFE1E201006BA000FFE2161A94
S11343D18001006BA000FFE21A5470F801386018E5
S11343E1883861386218838633890F8FF38911833
S11343F1883864F8883865F804386679009E576B05
S113440180FF6819006B80FF6A19006B80FF6C5491
S11344117001006DF27F67722079009E576B80FFF8
S10C4421687A0100FFE1DA7A0276
S113442A000097A00F905E0055A401006D725470AE
S113443A5A0043107A00000097907A0100FFE1DAEC
S111444A5E005B08558C5E0002A85A0043AA70
S11397700A0063616C6C6F63206661696C656400E9
S1139780408F400000000000BFF000000000000018
S113979000000000000000C00000000000000006
S10B97A03F50624DD2F1A9FC18
S11344587A00000097A801006DF07A0000FFE3FEE0
S11344685E0053EA0B977A0000FFE3FE01006DF04C
S11344785E0048020B9754705E005B7E7A0500FF6E
S1134488E43E0D040F960C4458600114AC004768D1
S1134498AC014710AC0258700106AC03587000C257
S11344A85A0045A855AA7A01000097AA0FE05E00B2
S11344B854480D00461E7A00000097B001006DF0C5
S11344C87A00000097AD01006DF00FE05E0053EA3B
S11344D80B970B9701006DF67A00000097AD01006A
S11344E86DF00FD05E0053EA0B970B9701006DF543
S10D44F87A00000097AD40507A01EE
S1134502000097AA0FE05E0054480D00461E7A0091
S1134512000097B001006DF07A00000097AD010032
S11345226DF00FE05E0053EA0B970B9701006DF6F7
S11345327A00000097AD01006DF00FD05E0053EAE0
S11345420B970B9701006DF57A00000097AD010000
S11345526DF05E0045FE0B970B9701006DF65E0052
S1134562480 20B974040403E01006DF67A0000007E
S113457297AD01006DF00FD05E0053EA0B970B97D6
S113458201006DF57A00000097AD01006DF05E0049
S113459245FE0B970B970FD05E00548E0B500D0107
S10A45A20FD05E0052665EBC
S10845A9005B5C54708F
S10E97A80C000A00002573000A0C00EF
S11345AE188838BA38B8F85038B919000B50792032

S11345BE01184DF8F83038BA28BCF88038BC54705E
S11345CE0C8820BC737047FA38BBE07F30BC547044
S11345DE20BC736047FA29BDE0BF30BC0C98547001
S11345EE7EBC736047067900000154701900547045
S11345FE5E005B7E7A0600FFE47E7A05000000040F
S113460E7A00000000180AF00AD07308470E7A00E9
S113461E000000180AF00AD00B70400A7A0000005E
S113462E00180AF00AD00F85189968E901006DF099
S113463E01006F71001C0FE05E0054AA0B97195511
S113464E0D5017F00AE0680947220D5017F00AE0E3
S113465E6808A80A4606F80D5C00FF640D5017F0B3
S113466E0AE068085C00FF580B5540D45E005B5CA3
S105467E547073
S11346801911400C19880B58792806824DF80B51E3
S11346901D014DF0547001006DF50D090D8D28D6E7
S11346A017500D010D99470C0D197969006079496F
S11346B0001040060D19796900600DD50D90148D19
S11346C00CD8C88038D63DD639D67900000455B009
S11346D001006D7554706DF56DF40D090D8528D6C7
S11346E017500D010D99470C0D197969006079492F
S11346F0001040060D19796900600D54119411944E
S113470011941194EC0F0D90148CED0F148D0CC8B3
S1134710C88038D63CD60CD8C88038D63DD639D6D2
S1134720790000045C00FF586D746D7554700100CE
S11347306DF619EE7900000F5C00FF4419667908E5
S113474000030DE05C00FF4E0B56792600034DEE8F
S1134750790800020DE05C00FF3C19667908002827
S11347600D605C00FF70790800100D605C00FF664F
S10947707908000E0D6044
S11347765C00FF5C790800060D605C00FF52790857
S113478600010D605C00FF48790800020D605C00C3
S1134796FF3E01006D7654707908000119005C0034
S11347A6FF2E7908000219005A0046D66DF60C8EC4
S11347B6AE0C460455E2401CAE0A460455DA4014D4
S11347C6AE0D460455D2400C17D60D68790000018C
S11347D65C00FEFC6D76547001006DF60D0E0D86C1
S11347E60D6079080040528009E0791000800D08B9
S11347F619005C00FEDA01006D7654705E005B7E84
S1134806790C000119447A0600FFE4CE7A0500000C
S113481600047A00000000180AF00AD07308470E55
S11348267A00000000180AF00AD00B70400A7A00DA
S1134836000000180AF00AD00F85189968E90100EC
S11348466DF001006F71001C0FE05E0054AA0B9718
S113485619550D5017F00AE0680947560D5017F021
S10948660AE06808A80A3D
S113486C460A0DC80D405C00FF68403C0D5017F024
S113487C0AE06808A80D460C790800020D405C009C
S113488CFE4840240D5017F00AE06808A80C4606B1
S113489C5C00FEFE40120D5017F00AE0680817D0BA
S11348AC0D080DC05C00FE220B5540A05E005B5C46
S10548BC547033
S11348BE1911400C19880B58792806824DF80B51A3
S11348CE1D014DF054705C0009145C0000B45504D6
S11348DE5A0049B801006DF618886AA800FFE5D1A1
S11348EE6AA800FFE5D318886AA800FFE5D26AA874
S11348FE00FFE50F79080080790000045C000808CA
S113490E7908001079000020 5C0007FC7906000F7F

S113491E790800100D605C0007EE0D687900000B3E
S113492E5C0007E40D687900000D5C0007DA790876
S113493E0050790000095C0007CE790800D6790093
S113494E00075C0007C219660D605C000710790E44
S113495E00010DE05C0006E60DE05C0007000D684B
S113496E7900002B5C0007A00D687900002F5C0016
S113497E07960DE8790000275C00078C01006D7621
S113498E54707908000519005C00077C79000064F7
S113499E5C00FF1C790800C419005C00076A7908E3
S10949AE00037900000183
S11349B45A0051167908000279000055C00075279
S11349C4790800CC19005A0051165E005B7E7A0404
S11349D4000098667A000000983501006DF05E00CF
S11349E445FE0B9719EE19660DE5790D001052D5A6
S11349F40D6009505C00070C17506DF001006DF455
S1134A045E0045FE0B970B870B56792600104DE08D
S1134A147A000000986C01006DF05E0045FE0B9770
S1134A240B5E792E00044DBE5E005B5C5470010086
S1074A346DF67A0698
S1134A3800FFE50E790000065C0006C468E87C60A8
S1134A4873404736790000E5C0006B40C8E734839
S1134A5847045C0002C8735E47085C0002C25A0040
S1134A684B26730E47085C0002B85A004B26731E88
S1134A7847045C0002AE5A004B267C6073604722F1
S1134A887900000C5C0006780C8E730847085C00FC
S1134A9800925A004B26731E587000825C0001C6B0
S1134AA8407C7C607320471E7900000A5C00065036
S1134AB80C8E730847065C00020A4062731E475E49
S1134AC85C00023E40587C6073104752790000082E
S1134AD85C00062C0C8E7368470A5C00FDFC5C00C6
S1134AE8FECE403A734E471A790800C07900000990
S1134AF85C00061A79080003790000055C00060EBD
S1134B08401C737E4718790800507900000095C003F
S1134B1805FC79080002790000055C0005F0010036
S1084B286D7654705E80
S1134B2D005B7E19DD790000265C0005CE0C8E73CB
S1134B3D68587001047A0600FFE511790000086DCD
S1134B4DF00FE1790800267900002 55C0005CC0BF8
S1134B5D870DD05C0004E80DD05C000502790500DB
S1134B6D200D505C0004C06868E8605860009C6EBE
S1134B7D680001473CA801471EA8034772A80547D3
S1134B8D3EA8064726A8084716A809475CA80A4762
S1134B9D26A80B4760406C5C0001865A004C346AB2
S1134BAD2800FFE51017500D08400E5C000224404D
S1134BBD765C00035440700DD879000021402479B0
S1134BCD0800400D505C0005406E6E0002CE806AF9
S1134BDDAE00FFE50F6A2800FFE50F17500D0879AA
S1134BED0000045C000522403E5C00038E40385CEF
S1134BFD00018840326E680002472C0D505C0004A2
S1134C0D0E40240D505C000406401C6868E860A843
S1084C1D2046080D50C4
S1134C225C0003F6400C686EEE60AE400D505C0013
S1134C3203E87A0000FFE5D27D0070000DD05C002E
S1134C42045440226A2800FFE5D3470818886AA85B
S1134C5200FFE5D30DD05C00041079080001790050
S10A4C6200275C0004AE5EB5
S1134C69005B5C54705E005B7E7900002E5C00047F

S1134C798E0C8EE8C017504626790000406DF07AF5
S1134C890500FFE5510FD17908002E7900002D5C4D
S1134C9900048C0B870D060D010FD05C00056279AA
S1134CA90000015C00039C790800017900002F5C76
S1134CB900045A5E005B5C5470790000365C0004A2
S1134CC93E54706DF6790000225C0004320C8E7339
S1134CD968472A7358472619005C0003866A280027
S1134CE9FFE5D3470C5C0001C819005C0003A04031
S1134CF90C79080001790000275C0004106D7654D3
S1134D09706DF67900002A5C0003F40C8E73684712
S1134D1908735847045C0004826D765470547054C8
S1134D2970547054705E005B7E7A0600FFE511686B
S1134D3968E803A80246426E680003E80747186E4D
S1134D49690003E9071751177178106A280000975A
S1084D59B417505C00DB
S1134D5E02D46E680003E8071750790100011A08A0
S1134D6E4B04101140F817117A0000FFE5D1680AC1
S1134D7E169A688A5E005B5C54705E005B7E7A06F0
S1134D8E00FFE5116868E803A80246406E68000359
S1134D9EE80747186E690003E90717511771781072
S1134DAE6A28000097B417505C0002626E68000315
S1134DBEE8071750790100011A084B04101140F847
S1134DCE7A0000FFE5D1680A149A688A5E005B5C7C
S1114DDE54707A0000FFE5D27D0072006A284F
S1134DEC00FFE514A801470AA8024716A8034722A7
S1134DFC54706A29000097BC17517A00000097BCC5
S1134E0C40686A29000097D017517A00000097CEAA
S1134E1C40586A2800FFE513470EA801471AA80259
S1134E2C4726A803473254706A29000097F517D117
S1134E3C7A00000097F540326A29000097F917D1E0
S1134E4C7A00000097F940226A290000980B17D1C9
S10D4E5C7A000000980B40126A2947
S1134E660000981317D17A00000098135502547066
S1134E765E005B7E0F840D187A0600FFE5D46A2870
S1134E8600FFE51817500C8018886A2900FFE517FC
S1134E961751091069E01D804F0269E801006BA4F0
S1134EA600FFE5D6F8016AA800FFE5D355065E00C4
S1134EB65B5C54705E005B7E7A0600FFE5D4790581
S1134EC60008696047446960792000084E026965F5
S1134ED67A0400FFE5D66DF50100694179080022E1
S1134EE6790000215C0002780B870D0569611901C1
S1134EF669E117F0010069410A81010069C169602E
S1134F06460818886AA800FFE5D35E005B5C547008
S1044F165E39
S1134F17005B7E6A2800FFE511E80317500D0519AA
S1134F27EE790600210D0047067925000146100F8B
S1134F37E05C0001DA0DE80D605C0001D2403C79CA
S1134F47250002462E6A2800FFE514E80717500DCF
S1134F5705790100011A084B04101140F86A28006B
S1134F67FFE5D11750660147067908000140060D92
S1134F77E840020DE80D605C0001945E005B5C5441
S1134F87706DF66A2800FFE5136AA800FFE510466F
S1134F973C19660D68790000285C0001720D687979
S1134FA700002C5C0001680D68790000305C00018B
S1134FB75E0D68790000345C0001540D68790000C8
S1134FC7385C00014A0D687900003C404018886A44
S1134FD7A800FFE5D1790600010D605C0000867922

S1134FE7080011790000285C00012479080003797F
S1134FF700002B5C0001180D60554A7908001279EF
S108500700002C5C0019
S113500C01080D687900002F5C0000FE6D7654706A
S113501C6DF60D065C0000E4C88017500D080D609A
S113502C5C0000E66D7654706DF60D065C0000CCEA
S113503CE87F17500D080D605C0000CE6D76547040
S113504C01006DF60D0617F6103679080008786026
S113505C6B2000FFE0085C0000B001006D7654701B
S113506C5E005B7E0D0617F610367A0500FFE00036
S113507C0AE569505C00008417500D06703E0D68FC
S113508C69505C0000845E005B5C54705E005B7E68
S113509C1B971B977A0500FFE5D20D0EFE017900D5
S11350AC00010DE11A094B04101040F8685917510F
S11350BC66104704702E4002722E701E17560DE0B8
S11350CC17F0103001006FF000040D6801006F71D0
S11350DC000478106B2000FFE000010069F1552AF1
S11350EC790000011B5E4B04101040F868591589B8
S10950FC68D90B970B9726
S10451025E4B
S1135103005B5C54706AA8004000036A28004000F7
S11351130154700D016AA9004000030D806AA800C1
S113512340000154705E005B7E7A03004000030F6E
S1135133840F9568BC19EE196640180DC055C6E86F
S11351430F471868BC6A280040000168D80B750B29
S11351535E0B566F7000181D064DE00DE05E005B9D
S11351635C54705E005B7E0D0C0D830F956F7400B2
S11351731819EE1966401E0D30558AE81F471A0D9C
S1135183C06AA80040000368586AA8004000010BE6
S1135193750B5E0B561D464DDE0DE05E005B5C54E6
S11351A37001006DF619EE7A0000FFE591790100B5
S11351B3405C0001140D06790000015C00FEAA0D9A
S11351C36647166DF67A0100FFE5917908002A799F
S11351D3000029558E0B870D0E790000015C00FE3C
S11351E3B40DE001006D76547019006BA000FFE568
S10651F3DC6BA0CF
S11351F600FFE5DA19006BA000FFE5E06BA000FFF6
S1135206E5DE54705E005B7E7A0400FFE5DC7A051A
S113521600FFE5DA0F830D1E7FF5725019994030B2
S11352266940792001004C2C695017F068397800E1
S11352366AA900FFE5E269500B5017F079010100F6
S113524601D0531069D869400B5069C00B730B59D1
S10E52561DE94DCC7FF570500D905EFC
S1135261005B5C54705E005B7E790E01007A040082
S1135271FFE5E07A0500FFE5DE0F830D127FF5728E
S113528150199940346940792001004C3269501713
S1135291F001D053E00D8017F0683978006AA90056
S11352A1FFE6E269500B5017F001D053E069D8696A
S11352B1400B5069C00B730B590D201D094DC67F5F
S11352C1F570500D905E005B5C54705E005B7E7AFE
S11352D10500FFE5E00F840D1E7FF572501966404E
S11352E1381DE64C386B2000FFE5DE695119107952
S11352F110010017F07901010001D053100D80173F
S1135301F00D6117F10AC178006A2800FFE6E2682F
S11353119869501B5069D00B5669504EC47FF57084
S1135321500D605E005B5C54705E005B7E7A05002D
S1135331FFE5DC0F840D1E7FF57250196640381DA1

S1135341E64C386B2000FFE5DA6951191079100139
S10853510017F07901D3
S1135356010001D053100D8017F00D6117F10AC13A
S113536678006A2800FFE5E2689869501B5069D007
S11053760B5669504EC47FF570500D605EFC
S1135383005B5C54706B2000FFE5DC17F0790101CF
S11353930001D053100D8054706B2000FFE5E0171C
S11053A3F07901010001D053100D8054700A
S11397B420282C3034383C2012010001000000081A
S11397C4FEFF10000100010102010902270001014B
S11397D400C06409040000030000000307058102BC
S11397E44000FF070502024000FF07058302400013
S11397F4FF040309041203550053004200200054DC
S113980400450053005400080331002E00300022A9
S113981403550053004200200054004500530054F4
S113982400200050005200 4F004700520041004DF9
S10498340030
S11391C00023002B0033003B0027002F0037003F14
S113983530302030312030322030332030342030 66
S113984535203036203037203038203039203041 2C
S113985520304220304320304420304520304 60A12
S10C9865002530325820000A00EE
S11353B001006DF67A0600FFE81E010069607A01BC
S11353C041C64E6D5E007A367A1000003039010016
S11353D069E06960198817707A01000080005E0037
S10D53E07A100D1001006D76547071
S11353EA5E005B7E0F857A06000000047A000000E7
S11353FA00180AF00AE07308470E7A000000001842
S113540A0AF00AE00B70400A7A00000000180AF05A
S113541A0AE00F8601006DF001006F70001C0100A5
S113542A6DF001006DF51A91790000015E005BA62B
S111543A7A170000000C0D065E005B5C5470D8
S11354485E005B7E0F860F9540040B760B756869CB
S113545868581C8946040C9946F068681750685DBB
S10D5468175519505E005B5C547089
S11354725E005B7E0F840FC60F950FE00B766C59AF
S10F5482688946F60FC05E005B5C547046
S113548E5E005B7E0F851AE640020B760FD00B751E
S10F549E680946F60FE05E005B5C54708A
S11354AA5E005B7E0F850F9601006F710018010085
S11354BA6DF101006DF601006DF51A917900000195
S11354CA5E005BA67A170000000C5E005B5C5470FA
S11354DA0FC446120FD5460E792107FF46120FB3A2
S11354EA4604156E4A0A1AC47A05000000081866AB
S11354FA5A00576E0D9946120FC4461A0FD546160F
S113550A0FB3461E16E6580002660FA246360FB3BD
S113551A46325800025A0FA246140FB346105800D7
S113552A024E0CE60D190FA40FB55A00577A10351F
S113553A1234792C00104C0C1B5910351234792C67
S113554A00104DF410331232792A00104C0C1B51FF
S113555A10331232792A00104DF4580000CA1AC4C3
S113556A1AD5186619995800020601006DF201004E
S113557A6DF301006F23000401006DF30100692339
S113558A795B800001006DF30FF2550E0B970B97B1
S113559A01006D7301006D7254706DF601006DF5B3
S11355AA01006DF401006DF301006DF201006DF16C
S11355BA01006DF0010069140100692510341035EA

S10955CA1FD44218451036
S11355D001006F14000401006F2500041FD444066A
S11355E00F940FA10FC26816682E0FA00100691453
S11355F001006F150004010069020100 6F0300043C
S1135600796C000F796A000F6919111911191119B1
S11356101119796907FF6901111111111111111183
S1135620796107FF792907FF5870FEAE0D11587095
S1135630FECC10351234103512341035123410 33B9
S113564012321033123210331232794C0080794AFD
S113565000800D901910474C792000364E3E79207A
S113566000204D0E793000200FB34702700A0FA3BC
S11356701AA2792000104D14793000100D380DB3A3
S11356800D2B0DA219AA0D884702700B0C884F141D
S1135690113213334402700B1B5040F01AA27A03E9
S11356A0000000010C6815E84B2A0AB544020B748C
S11356B00AA4792C010058500080 11341335440298
S10756C0700D0B5902
S11356C4792907FF5860006E1AC41AD5580000A63A
S11356D41FA446061FB55870FE8A1AB544087A34C7
S11356E40000000145061AA4450440121AA4173504
S11356F417347A150000000144020B740CE60FC43E
S113570446080FD41AD57919FFE00DCC460C0D4C7D
S11357140DD40D5D19557919FFF0792C0100450855
S1135724113413350B5940F2792C00804C08103591
S113573412341B5940F20D994E10113413350D993F
S11357444A08113413350B5940F4732D471C0CD8F4
S1135754E80B47167A150000000844020B74792CF1
S113576401004D06113413350B591134133511341B
S1135774133511341335796C000F101910191019DE
S11357841019649C101C1006131C01006D70010099
S1135794698401006F85000401006D7101006D725D
S11357A401006D7301006D7401006D756D765470A5
S11357B401F06401586000 80792407FF5860006C8D
S11357C45800007401F064235860006C40520F8544
S11357D401F06415460E0D44464601F06423464029
S11357E458000054103112307928001 04C0C1B5C03
S11357F41031123079280010 4DF4580000BA0FA567
S113580401F06435472610331232792A00104C0C08
S11358141B5410331232792A00104DF45800009EA1
S11358241AC4100E131C1AD55800018E790E07FFE3
S11358341AC41AD5580001621AC418EE790E07FF68
S113584419DD790500085800015001006DF60100C7
S11358546DF501006DF401006DF301006DF20100BB
S11358646DF101006DF0681E6826156E691C111C2C
S1135874111C111C111C796C07FF692411141114D8
S11358411141114796407FF01006D1001006911EB
S11358947968000F01006F2300040 1006922796A0B
S10958A4000F792C07FF41
S11358AA5870FF06792407FF5870FF120DCC587001
S11358BAFF160D445870FF40194C791C03FF0DCE97
S11358CA792EFFCC58D0FF5279480010794A00103C
S11358DA1AC47A0500000100103312321031123053
S11358EA441C0AB144087A100000000145040AA0C6
S11358FA40040AA004011235123444E0401C1AB1D0
S113590A44087A300000000145041AA040041AA092
S113591A04011235DD01123444C2730D46080AB17B
S113592A44020B700AA001F064014702700D792C3E

S113593A00804C06103512341B5E792E07FF58C0BF
S113594AFEE40DEE4E2E113413354402700D0DEEA6
S113595A4A040B5E40F0732D47180CD8E80B471224
S113596A7A1500000008440A0B74792C00804D0252
S113597A0B5E4020732D471C0CD8E80B47167A158B
S113598A0000000844020B74792C01004D061134FF
S109599A13350B5E11340E
S11359A013351134133511341335796C000F101E70
S11359B0101E101E101E64EC101C100E131C010090
S11359C06D700100698401006F85000401006D7131
S11359D001006D7201006D7301006D7401006D753E
S10959E001006D76547016
S11359E601006DF201006DF101006F010004010079
S11359F669000D82111211121112112112796207FF39
S1135A060D8A7968000F793203FF4B4A79220053D6
S1135A164E44794800107912FFEC4B22792200207C
S1135A264C0C0D224F1E103112301B5240F40F90B6
S1135A367912FFE00D224F0C10301B5240F6113045
S1135A460B524BFA796A8000470217B001006D7159
S10D5A5601006D7254701A8040F2D3
S10F5A605E007954A8014702190054703D
S1135A6C5E0079540C884E0419005470F80154707C
S1135A7C01006DF119990D11471A4A0679090800AD
S1135A8C17917919040F1B59101144FA1031103165
S1135A9C10311031010069811A9101006F810004EA
S1095AAC01006D7154704E
S1135AB25E0079540C884604F8015470190054703E
S1135AC201006DF201006DF11AA20F9147224A06FD
S1135AD27902080017B17912041F1B52103144FADC
S1135AE21031121210311212103112121031121210 3D
S1135AF2698201006F8100026F8A000601006D71E5
S1095B0201006D725470F6
S1135B0801006DF201006902010069920100699201006F0250
S1115B18000401006F92000401006D725470CE
S1135B266DF301006DF201006DF101006DF06C0B78
S1135B36689B0B011B7246F601006D7001006D71C7
S10B5B4601006D726D735470D0
S1115B4E5E0079541A08470479000001547070
S1135B5C01006F76001401006D7201006FF20010EA
S1135B6C01006D7201006D7301006D7401006D75A0
S1055B7C547060
S1135B7E01006DF501006DF401006DF301006DF28E
S1135B8E01006F72001001006DF201006FF6001438
S10B5B9E01006F720004547052
S1135BA65E005B7E7A37000000B27A0300FFE7F6F9
S1135BB67A0400FFE7E60D0201006FF100AA010077
S1135BC66F7600CE01006F7500D20D00466C0100A2
S1135BD66F7000CA460E790004526BA000FFE822DC
S1135BE65A005C7801006F7000CA6E080010E8075F
S1135BF617D0460C790005146BA000FFE82240720B
S1135C0601006F7000CA6E0800107328461073 08EF
S1135C16470C790105166BA100FFE82240540100E9
S1135C266F7000CA6E0800107368464601006F70F5
S1135C3600AA01006BA000FFE81A0D2017F001006F
S1135C466BA000FFE7E201006F7000CA01006BA0C2
S1135C5600FFE7E81A8001006BA000FFE7EC188855
S1135C6668C8F8306EF800885C001CBA0D005870DE

S1135C760A1C7900FFFF5A0066B66868A8255860B3
S1135C8609CC0B76188868C80FF001006BA000FFDB
S1075C96E7FA0FF027
S1135C9A01006BA000FFE7FE1A8001006BA000FF62
S1095CAAE80201006BA0FB
S1135CB000FFE8061A8001006BA000FFE80A01005C
S1135CC06BA000FFE80E1A8001006BA000FFE81232
S1135CD040306868A8204718A8234720A82B470A04
S1135CE0A82D461C7D40701040167D40703040103A
S1135CF07C407330460A7D40704040047D407020F4
S1135D000B766868A82D47CAA82B47C6A82047C2A8
S1135D10A82347BEF9206AA900FFE7F06869A93004
S1135D20460AF9306AA900FFE7F00B761A800100F2
S1135D306BA000FFE7F26868A82A586000800FD0C4
S1135D400BF001006FF000A07308470A01006F70A9
S1135D5000A00B70400601006F7000A00F851BF0C0
S1135D6001006FF000967308470A01006F700096F8
S1135D701B70400601006F700096690017F0010068
S1135D806BA000FFE7F24C167D40701001006B2002
S1135D9000FFE7F217B001006BA000FFE7F201007C
S1095DA06B2000FFE7F297
S1135DA67A20000002004F0E7D407000790005182E
S1135DB66BA000FFE8220B76686817D0796000FFB6
S1135DC617F078006A28000098767328587000086C2
S1135DD61A8001006BA000FFE7F24064686817D0E1
S1135DE67930003017F001006FF000A07A0100004F
S1105DF602001A810F907A010000000A5E7E
S1135E03007A1001006B2100FFE7F21F904D24017C
S1135E13006B2000FFE7F27A010000000A5E007ABC
S1135E233601006F7100A00A9001006BA000FFE729
S1135E33F2400E7D407000790005186BA000FFE867
S1135E43220B76686817D0796000FF17F078006A31
S1135E5328000098767328468 67A00FFFFFFFF0128
S1135E630069B06868A82E586001000B766868A8BB
S1135E732A466C0FD00BF001006FF00096730847AE
S1135E830A01006F7000960B70400601006F7000EB
S1135E93960F851BF001006FF000A07308470A01FA
S1135EA3006F7000A01B70400601006F7000A069B3
S1135EB30017F0010069B04C0A7A00FFFFFFFF01EE
S1135EC30069B0010069307A20000002004F0E7DA3
S1135ED3407000790005186BA000FFE8220B766879
S1135EE36817D0796000FF17F078006A28000098DC
S1085EF37673284776D9
S1135EF81A80010069B04058686817D079300030BB
S1135F0817F001006FF000A07A01000002001A8167
S1135F180F907A010000000A5E007A1001006931CF
S1135F281F904D1C010069307A010000000A5E00D1
S1135F387A3601006F7100A00A90010069B0400E23
S1135F487D407000790005186BA000FFE8220B76EE
S1135F58686817D0796000FF17F078006A28000096
S1135F689876732846927A000000002001006BA0FF
S1135F7800FFE8166868A8684708A86C4704A84C97
S1135F884610686817D017F001006BA000FFE816E9
S1135F980B766868A825587004A4A845587002129F
S1135FA8A8475870020CA8584742A863587002E4DF
S1135FB8A8644738A865587001F8A866587001F2B4
S1135FC8A867587001ECA8694722A86E5870043076

S1135FD8A86F4718A87058700398A873587002F8E8
S1095FE8A8754708A87824
S10D5FEE47045A00648201006B208F
S1135FF800FFE8167A200000006C46440FD00B908F
S113600801006FF000AA7308470A01006F7000AA25
S11360180B70400601006F7000AA0F851B900100EA
S11360286FF000967308470A01006F7000961B70A3
S1136038400601006F700096010069025A00618AE8
S113604801006B2000FFE8167A2000000068586002
S1136058009A6868A875470CA8584708A8784704A1
S1136068A86F46420FD00BF001006FF000AA730827
S1136078470A01006F7000AA0B70400601006F7099
S113608800AA0F851BF001006FF000967308470AFA
S113609801006F7000961B70400601006F70009638
S11360A86900177040400FD00BF001006FF00096A5
S11360B87308470A01006F7000960B7040060100D1
S11360C86F7000960F851BF001006FF000AA73082C
S11360D8470A01006F7000AA1B70400601006F7029
S10960E800AA690017F095
S11360EE0F825A00618A6868A875470CA85847083A
S11360FEA8784704A86F46420FD00BF001006FF04B
S113610E00967308470A01006F7000960B704006E5
S113611E01006F7000960F851BF001006FF000AA4F
S113612E7308470A01006F7000AA1B704006010036
S113613E6F7000AA6900177040400FD00BF001007A
S113614E6FF000AA7308470A01006F7000AA0B7064
S113615E400601006F7000AA0F851BF001006FF05F
S113616E00967308470A01006F7000961B70400675
S113617E01006F700096690017F00F8201006930FD
S113618E7A20FFFFFFFF460A7A000000000101009C
S113619E69B0686817D06DF07A01000000020AF149
S11361AE0FA05C00050E0B875A0063F601006B20EF
S11361BE00FFE8167A200000004C46420FD00B90E9
S11361CE0B9001006FF000AA7308470A01006F706D
S10961DE00AA0B7040064D
S11361E401006F7000AA0F851B901B9001006FF0D4
S11361F400967308470A01006F7000961B70404AAB
S113620401006F70009640420FD00B900B90010079
S11362146FF000AA7308470A01006F7000AA0B709D
S1136224400601006F7000AA0F851B901B900100AC
S11362346FF000967308470A01006F7100961B7193
S1136244400601006F7100960F907A0100000080F0
S11362540AF15E005B08010069307A20FFFFFFFF4B
S1136264460A7A0000000006010069B0686917D184
S113627401006F70008401006DF001006F700084F1
S113628401006DF07A00000000080AF05C00078644
S11362940B970B975A0063F60FD00BF001006FF0C6
S11362A400AA7308470A01006F7000AA0B70400626
S11362B401006F7000AA0F851BF001006FF00096B8
S11362C47308470A01006F7000961B7040060100B3
S10962D46F7000966E08D6
S11362DA00015A0064480FD00B9001006FF000AA26
S11362EA7308470A01006F7000AA0B704006010089
S11362FA6F7000AA0F851B9001006FF00096730858
S113630A470A01006F7000961B70400601006F7008
S113631A0096010069000F825E00548E01006FF03F
S113632A00AA010069317A21FFFFFFFF471201002A

S113633A69311F814C0A0100693001006FF000AA1C
S113634A0FF001006BA000FFE7FE1A8001006BA0AB
S113635A00FFE80A01006B2000FFE7F201006F71FA
S110636A00AA1A9001006BA000FFE8125A70
S11363770064900FD00B9001006FF00096730847ED
S11363870A01006F7000960B70400601006F7000E2
S1136397960F851B9001006FF000AA7308470A0147
S11363A7006F7000AA1B70400601006F7000AA01FE
S11363B70069007A6000FFFFFF01006FF00096019C
S11363C70069307A20FFFFFFFF460A7A00000000CA
S11363D701010069B0790000786DF07A01000000CF
S11363E7020AF101006F7000985C0002CE0B870F61
S11363F7F00F825E00548E01006FF000AA5A00640A
S1136407900FD00B9001006FF000A07308470A01AB
S1136417006F7000A00B70400601006F7000A00FA3
S1136427851B900F81730847060F901B7040020F5F
S113643790010069006B2100FFE7EE6981404AF88C
S11364472568F80FF00F827A000000000101006F42
S1136457F000AA0FF101006BA100FFE7FE1A9101FB
S1066467006BA123
S113646A00FFE80A01006B2100FFE7F21A8101002D
S113647A6BA100FFE812400E790005186BA000FF1C
S113648AE8227D4070006868A86E587001B86A28CF
S113649A00FFE7E67308586001AC7C407310463688
S11364AA01006B2000FFE8124F2C40207A01000004
S11364BA00017A0000FFE7F05C0013B87A0000FFDE
S11364CAE812010069011B710100698101006B2057
S11364DA00FFE8124ED601006B2000FFE8064624AF
S11364EA01006B2000FFE80A461A01006B2000FF37
S11364FAE80E461001006F7100AA0FA05C00137426
S113650A5A00661401006B2000FFE7FA0FA11A90E4
S113651A01006FF0009C01006FF000A04F300100F2
S113652A6F71009C0FA05C00134A40227A0000009E
S113653A00880AF07A01000000015C0013367A0031
S113654A00FFE806010069011B710100698101006E
S113655A6B2000FFE8064ED401006B2000FFE7FE24
S113656A01006B2100FFE7FA1A9001006FF0009C0B
S113657A01006F7100A00A8101006FF100A00F8072
S113658A4F3601006F71009C01006B2000FFE7FA90
S113659A5C0012E040227A00000000880AF07A01C7
S11365AA000000015C0012CC7A0000FFE80A010037
S11365BA69011B710100698101006B2000FFE80A70
S10965CA4ED401006F71C5
S11165D000AA01006F7000A01A8101006B2069
S11365DE00FFE7FE5C00129840227A00000000885C
S11365EE0AF07A01000000015C0012847A0000FFB9
S11365FEE80E010069011B710100698101006B2026
S113660E00FFE80E4ED47C407310473601006B201A
S113661E00FFE8124F2C40207A01000000017A009F
S113662E00FFE7F05C0012487A0000FFE812010059
S113663E69011B710100698101006B2000FFE812E3
S113664E4ED60B76404001006FF600A6400E0100B9
S113665E6F7000A60B7001006FF000A601006F7043
S113666E00A668086EF8009547086E780095A82571
S113667E46DC01006F7100A61AE10FE05C0011F019
S113668E01006F7600A6686847145C0012900D0037
S113669E460C6A2800FFE7E673085870F5D45C00D1

S11366AE125E6B2000FFE7EE7A17000000B25E0069
S11366BE5B5C54705E005B7E7A37000000247A05C3
S10966CE000000040AF5C0
S11366D401006FF0001801006FF1002001006FF159
S11366E4001C18AA689A6F72003C0C22463CAA58F4
S11366F4472CAA644712AA69470EAA6F4712AA75C0
S11367044706AA78471840227A000000000A400688
S11367147A000000000801006FF00014400C7A00B6
S1136724000000 1001006FF000146F70003C79202A
S113673400644706792000 69461201006F7000184F
S11367444C0A01006F74001817B4400601006F74FB
S113675400181AE61AB30FC4467401006B2000FF35
S1136764E7F6476AF83068D87A0600000001406209
S11367740FD00AE00F82010069F00FC001006F71AE
S113678400145E00803A0100697068890FA06808EC
S1136794A8094E0C0FD00AE0680989306889401EA5
S11367A46F70003C79200078460A0FD00AE068092C
S11367B4895740080FD00AE06809893768890FC0F0
S10967C401006F710014D7
S11167CA5E00803A0F840B760FC4469E6A2849
S11367D800FFE7E6732847626F70003C0C00465AD7
S11367E8A858471EA86F4706A8784716404C0100CB
S11367F86F70001847440FE00B760AD0F9306889A8
S1136808403801006F700018473001006F70002096
S11368180B7001006FF000201B70F93068890100CC
S11368286F7000200B7001006FF000201B706E79F1
S1136838003D68897A03000000020FB00AE00F8365
S113684801006FF600146F70003C7920006447065E
S1136858792000 69467201006B2000FFE7F64608BD
S113686801006F700018476001006F7000184D42F7
S11368786A2800FFE7E6733847160B7301006F7049
S113688800200B7001006FF000201B70F92B4036BD
S11368986A2800FFE7E67348472E0B7301006F7001
S11368A800200B7001006FF000201B70F92068892D
S11368B840160B7301006F7000200B7001006FF01E
S10968C800201B70F92DF6
S11368CE68897A0600FFE7FE01006F70001C680CF2
S11368DEAC2B4708AC2D4704AC20460E01006F705D
S11368EE001C0B70010069E040466A2800FFE7E6D2
S11368FE732847326F70003C0C004634A858470A81
S113690EA86F4714A8784702402601006F70001C39
S113691E0BF0010069E0401801006F70001C0B7052
S113692E010069E0400A01006F70001C010069E07C
S113693E7A0400FFE7F6010069401FB04F1401000F
S113694E6946010069441AB401006BA400FFE80A0A
S113695E400C0FB61A8001006BA000FFE80A7A0400
S113696E00FFE81201006F70001C680BAB2B470889
S113697EAB2D4704AB20463801006B2000FFE7F236
S113698E1FE04F2C6A2800FFE7F0A83046220100D3
S10D699E6B2000FFE7F21AE07A0114
S11369A800FFE80A010069120A82010069921A804D
S11369B8010069C0401E01006B2000FFE7F21FE0E1
S11369C84F0C01006B2000FFE7F21AE040021A8027
S11369D8010069C001006F7600141B76401A01009C
S11369E86F7000200B7001006FF000201B700FE127
S11369F81B760AD1681968890FE64CE201006F70AB
S1136A080020189968897A17000000245E005B5CEF

S1136A1854705E005B7E7A37000000647A0300FFDF
S1136A28E7E67A04000000040AF47A0500FFE7F6B3
S1136A380F866FF1005001006FF6005CF83068C8EC
S1136A487A000000007C0AF07A010000986E5E006C
S1136A585B4E0D00587000B87A000000002F0AF052
S1136A6801006DF07A000000002E0AF001006DF0BD
S1136A7801006F70008801006DF001006F700088DD
S1136A8801006DF07A01000000320AF17A0000007B
S1096A9800115E007A56B6
S1136A9E7A17000000100D024752188868E80100AB
S1136AAE695001006B2100FFE7F21F904F060100B2
S1136ABE6950400801006B2000FFE7F201006BA054
S1136ACE00FFE8127A0600FFE7F00D207920FFFFA2
S1136ADE470C79200001460CF82B5A007806F82D46
S1046AEE5A4A
S1136AEF007806F82A68E85A0078087A0000000050
S1136AFF1101006DF00FC10B717A00000000260A1F
S1136B0FF05E007E3C0B97400CF8016EF8002F18D7
S1136B1F886EC800017A000000000101006FF000C9
S1136B2F60400E01006F7000600B7001006FF0008A
S1136B3F6001006F7000600AC06808A83047E40165
S1136B4F006F7000607A20000000014F7A01006F20
S1136B5F7000600AC05E00548E0F8201006F7000D8
S1136B6F601B7001006F71002A0A8101006FF10031
S1136B7F2A7A000000000101006FF0005840300135
S1136B8F006F7000600AC001006F7100580AC1687E
S1136B9F08689801006F7000580B7001006FF000C8
S1136BAF5801006F7000600B7001006FF0006001FF
S1136BBF006F7000580FA11F904FC401006F70003A
S1136BCF580AC0189968890FC00B7001006FF00045
S1086BDF445E00548E2A
S1136BE401006FF000601A8001006FF0005801008B
S1136BF46F7000607A01000000025E007A100100E9
S1136C046FF0004001006F70004401006F71006079
S1136C140A901B7001006FF0003C404E01006F703E
S1136C24004401006F7100580A9001006FF0004C9A
S1136C3468086EF8005701006F71003C01006F7221
S1136C4400581AA101006FF1004801006F72004C53
S1136C54681968A901006F710048689801006F7191
S1136C6400580B7101006FF1005801006F70005858
S1136C7401006F7100401F904DA201006F7000600E
S1136C84461AF8306EC800017A00000000010100C2
S1136C946FF000601A8001006FF0002A6F700050DB
S1136CA47920006747067920004746 3C7D307050C1
S1136CB401006F70002A01006F7100600A901B705D
S1136CC47A20FFFFFFFC4D0C01006B2100FFE7F668
S1096CD41F904F0C6F70CE
S1136CDA00501BD06FF000504008790000666FF037
S1136CEA00506E78002FA80146246838E8186EF819
S1136CFA0057A8084706A810470A401A0FE00B7660
S1136D0AF92B40100FE00B76F920688940080FE051
S1136D1A0B76F92D68896F700050792000665860E8
S1136D2A077201006F70002A58D001861A80401A30
S1136D3A0FE00B7601006F7100580AC10B716819D5
S1136D4A688901006F7000580B7001006FF00058DA
S1136D5A01006F7100601F904DD61A800F820100E7
S1136D6A6BA600FFE7FE01006F70002A01006BA00B

S1136D7A00FFE80A7C307350460A01006B2000FFCB
S1136D8AE7F64E067C307320470E0FE00B76F92E9A
S1136D9A68890FA00B700F827C307350470C7C30CC
S1136DAA735047247C307320471E01006BA600FFF3
S1136DBAE8020100695001006BA000FFE80E010020
S1096DCA69500FA10A81CC
S1136DD00F9201006F70002A0FA10A9001006F71DA
S1136DE000600A9001006FF0006001006F71005CA9
S1136DF06819A92B4708A92D4704A9204652010069
S1136E006B2000FFE7F201006F7100601F904F409D
S1136E106A2800FFE7F0A830463601006F70005C77
S1096E200B7001006BA0E2
S1136E2600FFE7FA01006B2000FFE7F201006F7134
S1136E3600601A9001006BA000FFE8061A800100AB
S1136E466BA000FFE8125A00780401006B2000FFD4
S1136E56E7F201006F7100601F904F4C01006B2039
S1136E6600FFE7F201006F7100601A9001006BA04A
S1136E7600FFE8126E78002FA80146186E78002FDF
S1136E86A801586009786E780057A8084706A81025
S1136E965860096A7A0000FFE812010069011B7154
S1136EA6010069815A0078041A8001006BA000FF73
S1136EB6E8125A00780401006F70006001006F71D8
S1136EC6002A0A8101006FF1002A010069520AA112
S1136ED601006FF100521F814C120F914D0E0100FC
S1136EE66F7100520B710FC05C00092201006F70B5
S1136EF6002A58F0028A6848A83046147A0000002F
S1136F06000101006FF0004C01006FF00052402CAD
S1096F161A8001006FF078
S1136F1C004C1A8001006FF0005201006F70002AC0
S1136F2C0B7001006FF0002A01006F7000600B7092
S1136F3C01006FF000601A8001006FF00058403AB6
S1136F4C0FE00B7601006F71004C0AC16819688958
S1136F5C01006F70002A1B7001006FF0002A010002
S1136F6C6F7000580B7001006FF0005801006F70C8
S1136F7C004C0B7001006FF0004C01006F70002A85
S1136F8C4EBE0FE00B76F92E688940240FE00B768A
S1136F9C01006F71004C0B7101006FF1004C1B7100
S1136FAC0AC168196889010069501B70010069D016
S1136FBC01006F7000580B7001006FF00058010056
S1136FCC6F70004C01006F7100521A9001006F71C9
S1136FDC00601F904C0A01006B2000FFE7F64EACDB
S1136FEC7C307350470C7C307350472E7C307320AD
S1136FFC47280100695001006F7100580A81010094
S107700C6FF10058C5
S107701001006BA667
S113701400FFE7FE0100695001006BA000FFE80ACE
S113702440226E68FFFFA830461A40101B76010009
S11370346F7000581B7001006FF000586E68FFFFFB
S1137044A83047E87C307320464A6E68FFFFA82EB9
S1137054464201006B2000FFE80A47067C30735068
S113706447321B7601006F70005801006B2100FF4B
S1137074E80A1A901B7001006FF000581A8001008F
S11370846BA000FFE80A01006F70005C01006BA0B5
S113709400FFE7FE01006F70005C6808A82B470837
S11370A4A82D4704A820465001006B2000FFE7F2F7
S11370B401006F7100581F904F3E6A2800FFE7F0EC
S11370C4A830463401006F70005C0B7001006BA0A4

S11370D400FFE7FA01006B2000FFE7F201006F7184
S11370E400581A9001006BA000FFE8061A80010003
S11370F46BA000FFE812406201006B2000FFE7F27F
S109710401006F71005849
S10B710A1F904F4601006B20AA
S113711200FFE7F201006F7100581A9001006BA0A3
S113712200FFE8126E78002FA80146146E78002F34
S1137132A80146286E780057A8084704A810461CE1
S11371427A0000FFE812010069011B7101006981E5
S1137152400A1A8001006BA000FFE81201006B20B5
S113716200FFE7FE01006F71005C1F9058600692FA
S113717201006B2000FFE7FA01006BA000FFE7FEAE
S11371825A0078046848A83046147A0000000001C7
S113719201006FF0004C01006FF00052402A1A8088
S11371A201006FF0004C01006FF0005201006F709C
S11371B2002A0B7001006FF0002A01006F7000605B
S11371C20B7001006FF0006001006F70002A4F1412
S11371D27A000000000101006FF0004C0FE00B7613
S11371E2684940060FE00B76F93068897A0000009F
S11371F2000101006FF000580FE10B76FA2E689A36
S109720201006F7000584B
S11372080B7001006FF0005801006F70002A58C01E
S1137218009E01006BA600FFE7FE01006F70002AC5
S113722817B0010069511F814F2001006F70002AB8
S113723817B001006BA000FFE80A01006F70002A75
S113724801006F7100581A8140180100695001004C
S11372586BA000FFE80A0100695001006F71005834
S11372680A8101006FF1005801006F70002A0100C4
S113727869510A81010069D140360FE00B7601009C
S11372886F71004C0AC168196889010069501B7045
S1137298010069D001006F7000580B7001006FF096
S11372A8005801006F70004C0B7001006FF0004C28
S11372B801006F70004C01006F7100521A900100B9
S11372C86F7100601F904C0A01006B2000FFE7F606
S11372D84EA87C307350470C7C30735047347C3055
S11372E87320472E010069504F4A01006BA600FF27
S10972F8E80201006950E9
S10772FE01006BA07D
S113730200FFE80E0100695001006F7100580A8105
S113731201006FF1005840226E68FFFFA830461A41
S113732240101B7601006F7000581B7001006FF054
S113733200586E68FFFFA83047E87C307320467020
S11373426E68FFFFA82E466801006B2000FFE80A63
S1137352460A01006B2000FFE80E47067C3073509B
S1137362474E1B7601006F70005801006B2100FF2E
S1137372E80A1A9001006B2100FFE80E1A901B70B5
S113738201006FF000581A8001006BA000FFE80EA5
S113739201006BA000FFE80A01006F70005C0100AE
S11373A26BA000FFE7FE1A8001006BA000FFE8025A
S11373B201006F70005C6808A82B4708A82D4704DA
S11373C2A820465001006B2000FFE7F201006F7115
S11373D200581F904F3E6A2800FFE7F0A83046345A
S11373E201006F70005C0B7001006BA000FFE7FAF5
S10773F201006B2008
S11373F600FFE7F201006F7100581A9001006BA0BD
S113740600FFE8061A8001006BA000FFE812406245
S113741601006B2000FFE7F201006F7100581F9017

S11374264F4601006B2000FFE7F201006F71005821
S11374361A9001006BA000FFE8126E78002FA801D6
S113744646146E78002FA80146286E780057A808C0
S11374564704A810461C7A0000FFE81201006901E0
S11374661B7101006981400A1A8001006BA000FFAD
S1137476E81201006B2000FFE7FE01006F71005C5C
S11374861F90461001006B2000FFE7FA01006BA076
S113749600FFE7FE5A007804010069500B700100F3
S11374A66F7100601F904C0C010069510BF10FC006
S11374B65C00035A6848A830460E7A0000000001B3
S11374C601006FF0004840241A8001006FF0004865
S11374D601006F70002A1B7001006FF0002A010083
S11374E66F7000600B7001006FF0006001006F7039
S10974F6006001006F714C
S11374FC00481A9001006F72002A0A8201006FF291
S113750C002A0FE00B760AC1681968897A0000001B
S113751C000101006FF000580FE10B76FA2E689A08
S113752C01006F7000580B7001006FF000580100E0
S113753C6F700048402E0FE00B7601006F71004C0A
S113754C0AC168196889010069501B70010069D070
S113755C01006F7000580B7001006FF000580100B0
S113756C6F70004C0B7001006FF0004C01006F71D9
S113757C00601F904E0A01006B2000FFE7F64EB629
S113758C7C307350470C7C307350472E7C30732007
S113759C472801006BA600FFE7FE010069500100BC
S10575AC6BA0CF
S11375AE00FFE80A0100695001006F7100580A815B
S11375BE01006FF1005840226E68FFFFA830461A93
S11375CE40101B7601006F7000581B7001006FF0A6
S11375DE00586E68FFFFA83047E87C307320464A98
S11375EE6E68FFFFA82E464201006B2000FFE80ADB
S11375FE47067C30735047321B7601006F7000587C
S113760E01006B2100FFE80A1A901B7001006FF056
S113761E005801006F70005C01006BA000FFE7FED5
S113762E1A8001006BA000FFE80A6F7000507920EA
S113763E006546080FE00B76F96540060FE00B7602
S113764EF945688901006F7000580B7001006FF0E7
S113765E005801006F70002A4D0A0FE00B76F92BCC
S113766E688940160FE00B76F92D688901006F705B
S113767E002A17B001006FF0002A01006F70005846
S113768E0B7001006FF0005801006F70002A7A2012
S109769E000000644D0E24
S11376A40BF601006F7000580B800B7040140FE051
S11376B40B76F9306889F83068E801006F70005878
S11376C40BF001006FF000580FE00B7001006FF036
S11376D4003040360FE01B76010069F001006F7043
S11376E4002A7A010000000A5E007A108930010042
S11376F46970688901006F70002A7A010000000A2A
S11377045E007A1001006FF0002A01006F70002AF6
S11377144EC201006F76003001006F70005C680890
S1137724A82B4708A82D4704A820465001006B2026
S113773400FFE7F201006F7100581F904F3E6A2863
S113774400FFE7F0A830463401006F70005C0B7053
S113775401006BA000FFE7FA01006B2000FFE7F2D2
S113776401006F7100581A9001006BA000FFE80636
S11377741A8001006BA000FFE812406201006B2035
S113778400FFE7F201006F7100581F904F4601009C

S10577946B2065
S113779600FFE7F201006F7100581A9001006BA019
S11377A600FFE8126E78002FA80146146E78002FAA
S11377B6A80146286E780057A8084704A810461C57
S11377C67A0000FFE812010069011B71010069815B
S11377D6400A1A8001006BA000FFE81201006B202B
S11377E600FFE7FE01006F71005C1F904610010069
S11377F66B2000FFE7FA01006BA000FFE7FE188885
S113780668E87A17000000645E005B5C54705E00F3
S11378165B7E0F850F960FD00AE06808A8344F5297
S11378260FD00AE0F93068891B760FD00AE06809A1
S1137836890168894020 0FD00AE06808A8394F14E7
S11378460FD10AE1681888F668986E18FFFF880159
S11378566E98FFFF1B760FE64EDC6E580001A839C3
S11378664F106E58000188F66ED8000168588801DB
S113787668D85E005B5C54705E005B7E1B977A0380
S109788600FFE7E87A06AB
S113788C00FFE7EC010069F00F9501006B2100FF8D
S113789CE7E27A2100000001462601006DF50F8115
S11378AC010069305E007FE00B97010069300AD05C
S11378BC010069B0010069600AD0010069E0403A37
S11378CC19444030010069700B70010069F01B70A2
S10F78DC680817D00100693101006B221D
S11378E800FFE81A5D207920FFFF47120100696055
S11378F80B70010069E00B5417F41FD44DCA0B97A2
S11379085E005B5C547001006B2000FFE7E27A20A5
S1137918000000001460C01006B2000FFE7E81899FE
S11379286889547001006B2000FFE7E27A200000A9
S1137938000146041900547001006B2000FFE7E8BA
S10F79486E090010E94017D10D105470B7
S10B986E0000000000000000EF
S113795401006DF501006DF401006DF301006DF29A
S113796401006DF101006D120100691301006F0143
S1137974000401006900 0D840D8C796C7FF0792C6F
S11379847FF047540DAC796C7FF0792C7FF0475628
S11379940D8C65AC4B5E1AB144087A3000000001CB
S11379A445141AA0451001F06410475A0C444B0ABD
S11379B479000002400C0C444BF619004004790092
S11379C4FFFF01006D7101006D7201006D73010011
S11379D46D7401006D7554700F85796D000F01F09E
S11379E4641546DA409E0FA5796D000F01F06435E6
S11379F446CC409C01F064207A607FFFFFFF46ACD5
S10F7A0401F0643146A67900000140B691
S1137A106DF20D820C2A4A0217B00D994A04D280E6
S1137A2017B15E00803A0C224A0217B00CAA4A0230
S1097A3017B16D725470E2
S1137A3601006DF301006DF20D8252120D93520394
S1137A4652100928093801006D7201006D735470D4
S113987620202020202020202060606060 6020209F
S113988620202020202020202020202020202020CF
S11398964810101010101010101010101010101087
S11398A6848484848484848484841010101010101027
S11398B6108181818181810101010101010101010180
S11398C601010101010101010101101010101034
S11398D610828282828202020202020202020251
S11398E602020202020202020202021010101020F9
S11398F60000000000000000000000000000005F

S113990600000000000000000000000000004E
S113991600000000000000000000000000003E
S113992600000000000000000000000000002E
S113993600000000000000000000000000001E
S113994600000000000000000000000000000E
S113995600000000000000000000000000000FE
S109996600000000000F8
S10D996C0000000000000000000EE
S1137A565E005B7E7A370000003C7A040000000C6F
S1137A660AF47A05000000140AF501006FF00038E5
S1137A7601006FF1003401006F73005C7A020000AD
S1137A86000801006FF2002C19226FF2002A7A0611
S1137A96000000540AF6686AEA7F46620FE00B703C
S1137AA67A01000000075E0080760D0047501A80B9
S1137AB6401A01006F70003401006F7100380A909C
S1137AC61899688901006F7000380B7001006FF018
S1137AD600387A20000000084DD81A80010069B0EA
S1137AE67C607370470A01006F700060F9FF4008FD
S1137AF601006F700060F90168895A007E2E7A00D2
S1137B060000001C0AF001006DF07A010000002855
S1137B160AF101006F70005C01006DF001006F70E7
S1137B26005C01006DF001006F70006C5E0084065E
S1137B367A170000000C01006F7000606808A8FF48
S1077B46460C7A006C
S1137B4A000000540AF07D0072706F7000247920DF
S1137B5A07FF464A6E680001A8F0463A0FE00BF0A9
S1137B6A7A01000000065E0080760D0047280100B6
S1137B7A6F7000606808A8014608790000015A007E
S1137B8A7E3001006F7000606808A8FF4610790014
S1137B9AFFFF5A007E30790000025A007E306F7070
S1137BAA0024793003FF6FF000244D1E01006F702B
S1137BBA005801006DF001006F70005801006DF06C
S1137BCA5E0081960B970B97401E01006F70005859
S1137BDA01006DF001006F70005801006DF05E0046
S1137BEA81960B970B971B500D0618886EF8002386
S1137BFA0B560D6017F001006F7100381A900100DF
S1137C0A69B0010069F10FD1010069705E0082B2A7
S1137C1A01006F70003801006F71002C5E007A104A
S1137C2A01006FF000301AE67A0000000008010034
S1097C3A6F7100301A9087
S1137C400F82401401006F7000300AE00AD00FD198
S1137C500AE1680868980B760FA01F864DE6400A74
S1137C600FD00AE0189968890B767A26000000087D
S1137C704DEE6F7000326F78002E52806F71003AB4
S1137C8019016DF17A01000000080FD05E00823AFD
S1137C900B8701006F7600381B760FC10FE05E0083
S1137CA082B20FE001006F71002C5E007A100100B8
S1137CB06FF000301AE67A000000000801006F71CF
S1137CC000301A900F82401401006F7000300AE0F8
S1137CD00AC00FC10AE1680868980B760FA01F86D7
S1137CE04DE6400A0FC00AE0189968890B767A2698
S1137CF0000000084DEE6F7000326F78002E528046
S1137D006F71003A19011B516DF17A0100000008EF
S1137D100FC05E00823A0B8701006F7000340100D0
S1137D206DF00100693101006DF17A01000000245A
S1097D300AF16F70002C44
S1137D365E00809E0B970B977A06000000040AF6F6

S1137D467A000000000801006DF001006F71003831
S1137D560FE05E007FE00B977A0000000008010049
S10A7D666DF00FD10FE05E89
S1137D6D0084AA0B970D004F12790000016FF000EC
S1137D7D2A010069300B705A007E267A000000003C
S1137D8D0801006DF001006F7100380FE05E007F98
S1137D9DE00B977A000000000801006DF00FD10F82
S1137DADE05E0084AA0B970D00463201006930 0B8B
S1137DBD70010069B001006F70003401006DF001B6
S1137DCD00693301006DF37A01000000240AF16F9D
S1137DDD70002C5E00809E0B970B9740447A000039
S1137DED00000801006DF001006F7100380FE05EB7
S1137DFD007FE00B977A000000000801006DF00F83
S1137E0DC10FE05E0084AA0B970D004C146F700038
S1137E1D2A460E010069301B70010069B05A007DBE
S1127E2D1819007A170000003C5E005B5C54706C
S1137E3C5E005B7E7A370000002C0FF30F86010087
S1137E4C6FF100107A000000000101006FF00018C0
S1137E5C01006F7500440A95188868D87A000000F1
S1137E6C000801006DF00FE17A000000000C0AF02D
S1137E7C5E007FE00B9701006F7000445A007FCCCB
S1137E8C0FA00B707A01000000055E007A100B70D6
S1137E9C10307A06000000081A867A0000000008E9
S1137EAC1AE001006DF00FA11031103110317A116D
S1137EBC000099760AE10FB00AE05E007FE00B97B1
S1137ECC7A000000000801006FF000281A800100FE
S1137EDC6FF0002001006FF0001C7A040000000812
S1137EEC1AE401006FF400145A007F8E01006F70C6
S1137EFC001401006DF00FB10AE17A000000000CD0
S1137F0C0AF00AE05E0084AA0B970D004D3C0100B9
S1137F1C6DF40FB00AE001006DF07A01000000105F
S1097F2C0AF10AE11A80CC
S1137F325E00835A0B970B9717F001006FF000183E
S1137F4201006F70002801006F7100200A81010097
S1137F526FF1002001006F7000287A010000000217
S1137F625E007A1001006FF00028473079000001AB
S1137F726DF00FB00AE00FC15E0082E00B870100D3
S1137F826F70001C0B7001006FF0001C01006F701A
S1137F92001C7A200000000458D0FF5A01006F70C1
S1137FA20018461C6E78002388306CD84004F830E1
S1137FB268D81B7501006F7000101F8544F04012D2
S1137FC26E78002388306CD80FA01B700F8258C0C4
S1117FD2FEB87A170000002C5E005B5C547052
S113997600000000000000080000000000000005086
S113998600000000000003200000000000001F404C
S11399960000000000013880000000000C3500C4
S11399A600000000007A12000000000004C4B400A6
S11399B6000000002FAF080000000001DCD65000B5
S11399C600000012A05F2000000000BA43B7400069
S11399D600000746A5288000000048C273950000D2
S11399E60002D79883D20000001C6BF526340000D2
S10B99F6011C37937E080000F9
S1137FE05E005B7E0F850F9401006F7600181FC53E
S1137FF047401FC544200FD30FC21AC440120FB00D
S11380000B730FA10B710F921B71681968890B74A5
S11380101FE445EA401C0FD30AE30AE40FC21AC463
S1138020400C0FA01B700F8268086CB80B741FE420

S10D803045F00FD05E005B5C547056
S113803A01006DF20D9946100D82177253120DA8A5
S113804A53100D810D28401E0F920D811771790867
S113805A0010121012311AA144020AA11B5846F247
S10F806A12101710177001006D72547093
S11380765E005B7E0F840F951AE6400E0FC00AE082
S1138086680947041900400A0B761FD64DEE79009E
S10B809600015E005B5C547005
S113809E5E005B7E7A370000000C7A05000000015B
S11380AE0AF50D0C0F967904000801006DF57A019F
S11380BE0000000E0AF101006F70002817B05E0079
S11380CE85700B9701006DF66DFC7A0000000010B1
S11380DE0AF00FD15E0084E80B970B87790C003FF3
S11380EE6F70000A190C0DC017F001D053400D0E1E
S11380FE7900004219C017F001006DF07A010000FB
S113810E00090FD05E0087BC0B9779060008401458
S113811E0D6019E017F00AD00D6117F10AD1680846
S113812E68981B561DE64CE8400C0D6017F00AD0FC
S113813E189968891B560D664CF00DE05240190CC8
S113814E6DFC7A01000000090FD05E0082E00B8700
S113815E7900003F6FF0000A7A01000000400FD053
S113816E5E0086D27A000000000801006DF00FD188
S113817E01006F70002C5E007FE00B977A170000F2
S109818E000C5E005B5CC7
S1058194547022
S113819601006DF27A370000001A7A000000001819
S11381A60AF001006F71002601006DF101006F7185
S11381B6002601006DF17A01000000080AF10100B2
S11381C66DF15E0088587A170000000C6F71001875
S11381D60FF05E005A7C0F817A02000099FE7A0046
S11381E6000000080AF05E008B8E7A000000001083
S11381F60AF001006F71000C01006DF101006F714F
S1138206000C01006DF17A01000000080AF101007B
S11382166DF15E00895E7A170000000C7A0000009B
S113822600100AF05E0059E67A170000001A0100F2
S10782366D7254709E
S10B99FE3FD34395810624DDEC
S113823A5E005B7E1B971B87010069F00F946F79C1
S113824A001E790D0008199D4754790100FF0DD2CC
S113825A1A0A4B04101140F80C9B18DD1B740FC645
S113826A4028010069740AE468450CB816850FC0F2
S113827A0D915E008A94684814D868C80DD01A080C
S113828A4B04110540F80C5D1B760FE64CD40CDD4C
S113829A4706790100014002191117F10F900B8764
S10B82AA0B975E005B5C54704E
S11382B25E005B7E0F860F957A00000000080100C6
S11382C26DF01036103610367A0100009A060AE174
S11182D20FD05E007FE00B975E005B5C547084
S1139A060000000000000001000000000000000547
S1139A160000000000000019000000000000007DA7
S1139A260000000000000027100000000000000C3579
S1139A360000000000003D09000000000001312D78
S1139A46000000000005F5E100000000001DCD65E3
S1139A5600000000009502F90000000002E90EDD97
S1139A66000000000E8D4A510000000048C27395A5
S1139A76000000016BCC41E9000000071AFD498D87
S1139A860000002386F26FC1000000B1A2BC2EC500

S1139A96000003782DACE9D900001158E460913D2C
S1139AA6000056BC75E2D6310001B1AE4D6E2EF5FF
S1139AB6000878678326EAC9002A5A058FC295EDFE
S1139AC600D3C21BCECCEDA10422CA8B0A00A42567
S1139AD614ADF4B7320334B96765C793FA10079D1B
S11382E05E005B7E7A370000000A0F83010069F1AC
S11382F06F790022790C0008199C4752FAFF0DC0D0
S11383001A084B04110A40F80CA218CC1AE64026AE
S11383100FB50AE568540C2816840FD00D915E0042
S11383208AB6685814C868D80DC01A084B041004DC
S113833040F80C4C0B76010069701F864DD20CCCB3
S1138340470679010001400219111 7F10F907A17BE
S10D83500000000A5E005B5C54703D
S113835A5E005B7E7A370000000C01006FF00004B8
S113836A0F9601006F7500280AD61B7601006F73FA
S113837A00240AD31B737A02000000011AC4404482
S113838A6868175068391751191017F01AC0010095
S113839A69F04C14010069717A11000001000100AF
S11383AA69F1790000014002190017F00F840100F6
S11383BA697047041A800F826E78000368E81B7598
S11383CA1B761B730FD546B87A2400000001460EAC
S11383DA01006F70000446067900FFFF40120FA0E8
S11383EA7A200000000146041900400479000001C4
S10F83FA7A170000000C5E005B5C5470FE
S11384065E005B7E7A03000000070F820F95010072
S11384166F7600207A04000000180AF46941796136
S1138426800079218000460679 01FFFF4004790127
S113843600010FA06889694079607FF0119011905F
S11384461190119069D07A000000000701006DF0C9
S11384560B740FC10FE05E007FE00B9779000003FA
S11384666DF00FB10FE05E00823A0B8769504F063D
S11384767D607070402869500B5069D07D607270C2
S1138486401679000001 6DF00FB10FE05E00823AED
S11384960B8769501B5069D07C60737047E45E009C
S10784A65B5C547054
S11384AA5E005B7E0F860F9301006F7400180FE561
S11384BA0FC44604190040201AE6400A6C586C3966
S11384CA1C9846060B761FC645F21B75685817504B
S11184DA1B73683B175319305E005B5C5470D4
S11384E85E005B7E7A37000000100FF60F850F944D
S11384F869506F710028091069D001006DF60100F9
S11385086F71002E0FC05E008E6A0B977C607370CC
S1138518470869500B5069D0400C7A0100000010DD
S11385280FE05E008E187A000000004201006DF033
S11385387A01000000100FE05E0087BC0B976E689D
S11385480008E8E06EE800087A000000000901006E
S11385586DF00FE10FC05E007FE00B977A17000004
S10B856800105E005B5C54701F
S11385705E005B7E7A37000000187A050000000970
S11385800AF50F840F920FC44D087A03000000010F
S113859040147A03FFFFFFFF0FC07A01FFFFFFFFC5
S11385A05E007A360F840FC07A010000001B5E0064
S11385B07A100F860FC07A010000001B5E007A104C
S11385C00F947A2300000001462E7A000000000871
S11385D001006DF00FE11031103110317A110000FC
S11385E09AE60FD05E007FE00B970FE01030780023
S11385F06B2100009BB640320FC446021B767A0003

S1138600000000801006DF00FE11031103110314E
S11386107A1100009B4E0FD05E007FE00B970FE0B6
S1138620103078006B2100009BD06FF100120FC453
S113863047747A23FFFFFFFF460A7A000000001BFE
S11386401AC00F8401006F70003001006DF00FA19C
S11386500FC05E008FF80B970FE646087A230000E1
S10786600001476269
S113866401006F70003001006DF00FA069006DF020
S11386747A00000000180AF00FD15E0084E80B971B
S11386840B877A01000000400FD05E0086D2792068
S11386940001460E6F7000120B506FF000127D50F4
S11386A470700FA06F71001269817A0000000008D6
S11386B401006DF00FD101006F7000345E007FE0A4
S11186C40B977A17000000185E005B5C547081
S1139AE68000000000000000CECB8F27F4200F3A41
S1139AF6A70C3C40A64E6C5286F0AC99B4E8DAFD4E
S1139B06DA01EE641A708DEAB01AE745B101E9E4A9
S1139B168E41ADE9FBEBC27DE5D3EF282A242E81E6
S1139B26B9A74A0637CE2EE195F83D0A1FB69CD94A
S1139B36F24A01A73CF2DCD0C3B8358109E84F07E6
S1139B469E19DB92B4E31BA99E74D1B791E07E48BC
S1139B56C428D05AA4751E4CF2D56790AB41C2A453
S1139B66964E858C91BA2655BA121A4650E4DDEC08
S1139B76E65829B3046B0AFA8E938662882AF53F60
S1139B86B080392CC4349DEDDA7F5BF59096684935
S1139B96873E4F75E2224E68A76C582338ED26237D
S1139BA6CF42894A5DCE35EB8049A4AC0C5811AE41
S1139BB60000005900B3010D016601C0021A0273C9
S1139BC602CD0327038003DA0434FFA6FF4CFEF21B
S1099BD6FE99FE3FFDE5D0
S1119BDCFD8CFD32FCD8FC7FFC25FBCBFB721D
S11386D25E005B7E7A370000000C0F840F957A06EA
S11386E2000000081AB30FD00FE15E007A100AC02F
S11386F20F820FD00FE15E007A10790000801A0911
S11387024B04119040F86EF8000511086EF8000B47
S113871211086EF800041B75010069F50FD00FE113
S11387225E007A100AC00F85010069700FE15E00D6
S11387327A10790600801A094B04119640F80FA0AB
S113874268086E790005169847280FA068066E78A8
S1138752000B166846086E780004166847087A0309
S11387620000001400C685816E847067A0300002F
S113877200017A2300000001463240140CE817502E
S1138782171068591689 68D9100E46041B75FE0125
S1138792685816E847041FC544E21FC545086858D0
S11387A214E868D8400679000001400219007A17DC
S10D87B20000000C5E005B5C5470D5
S11387BC5E005B7E1B971B970F82010069F17A03A6
S11387CC0000000801006F7000200FB15E007A10EA
S11387DC0FA60A8601006F7000200FB15E007A109D
S11387EC790400801A094B04119440F8686816C880
S11387FC46500CCD1855400414D5110D0CDD46F81C
S113880C6868165847086868 14C868E840340FA0AD
S113881C010069710A90010069F001006F7000207A
S113882C0FB15E007A100FA50A85400C68684708E3
S113883C685814C868D8400A0B76010069701F8603
S10F884C45EA0B970B975E005B5C5470D1
S11388585E005B7E0F857A040000000701006F70DD

S1138868002001006DF001006F70002001006DF021
S11388785E008DC60B970B970D06474879260001B6
S1138888460A790004B86BA000FFE82279260002A3
S1138898460A7900044C6BA000FFE82219667A00A7
S11388A80000001C0AF00D6117F10A90F9FF6889AE
S11388B80B56792600084DE67A000000001C0AF0E2
S11388C85A00894E7A000000001C0AF07A01000061
S11388D89BEA5E005A600D00470C190069D07A00C4
S11388E800009BEA40607A060000001C0AF66963F0
S11388F81113111311131113796307FF793303FE4E
S113890869D36950791003FE462869500B5069D022
S1138918696079608 00F69E00FE30B73400E0FC144
S11389280FB05E008E1869501B5069D06960734898
S113893847EC69607960800F79403FE069E07A002D
S10989480000001C0AF010
S113894E01006F7100185E005B085E005B5C547083
S10B9BEA000000000000000070
S113895E5E005B7E0F8601006F70002001006DF0DC
S113896E01006F70002001006DF05E008DC60B9745
S113897E0B970D05474079250001460A790004B887
S113898E6BA000FFE82279250002460A7900044C09
S113899E6BA000FFE82219667A000000001C0AF0A3
S11389AE0D6117F10A90F9FF68890B5679260008B5
S11389BE4DE65A008A7C7A050000001C0AF56955BB
S11389CE11151115111511115796507FF793503FF6A
S11389DE0D5C4C0E7A0000009BF20FE15E005B080B
S11389EE40607A000000001C0AF00FE15E005B0895
S11389FE792C00344C4C0FE50B950BF579090034AB
S1138A0E19C90D9017F07901001001D053100D09FB
S1138A1E400A0FD01BF5191169811B590D994EF29E
S1138A2E7900003419C017F07901001001D05310EA
S1138A3E7909FFFF1B584B04101940F869506690D3
S1078A4E69D07A016D
S1138A520000001C0AF10FE20F905E0055740FE054
S1138A627A0100009BF25E005AB20D00470C7A00B5
S1138A720000001C0AF07D0070707A000000001CE8
S1138A820AF001006F7100185E005B085E005B5C18
S1058A9254701B
S10B9BF2000000000000000068
S1138A9401006DF20D1A4F14792A00084D04FA00EF
S1138AA44008680A100A1B5A4EFA688A01006D725C
S1058AB45470F9
S1138AB601006DF20D1A4F14792A00084D04FA00CD
S1138AC64008680A110A1B5A4EFA688A01006D7239
S1058AD65470D7
S1138AD80F8501F064155860009A792407FF47143D
S1138AE80D44586000820FA501F064355860007882
S1138AF8580000800FA501F0643558600076406 87F
S1138B080FA501F06435466C0DCC465C0F8501F06A
S1138B1864154654405E0F8501F06415473C1031D7
S1138B281230792800104C0C1B5C10311230792854
S1138B3800104DF4580000C40FA501F06435471A1E
S1138B4810331232792A00104C0C1B541033123292
S1138B58792A00104DF4580000A81AA21AB3687DA8
S1138B68100D131A58000228790107FF1AA21AB325
S1138B78580001FA790107FF1AA27A0300000008D6
S1138B8868FA580001E801006DF601006DF501006F

S1138B986DF401006DF301006DF201006DF1010048
S1138BA86DF07A3700000018691D692565D569F5E8
S1138BB8691C111C111C111C111C796C07FF6924F9
S1098BC811141114111435
S1138BCE1114796407FF01006D10010069117968B2
S1138BDE000F01006F23000401006922796A000F60
S1138BEE792C07FF5870FEE2792407FF5870FF0AAD
S1138BFE0DCC5870FF1A0D445870FF36094C793C52
S1138C0E03FF792C07FF58C0FF58792CFFCB58D0A0
S1138C1EFF426FFC000279480010794A00100100F0
S1138C2E6FF000040F9601006FF2000801006FF35E
S1138C3E000C0D1552356FF500160D34529452B1CA
S1138C4E0A946511990009D44406791C0001990010
S1138C5E6FF400140DC50D1D18990D0452340AC579
S1138C6E99000DE452B40AC599000D6452240AC545
S1138C7E99006FF500120DD50D1D18990D84523400
S1138C8E0AC50D0452B40AC599000DE452240AC54F
S1138C9E99000D6452A40AC599006FF500100DD505
S1138CAE0D1D18990DB352830AB50D0452240AC52E
S1098CBE99000DE452A42D
S1138CC40AC59900528209D24404791A0001091A87
S1138CD46F74000852040AC26F7400085284094A6C
S1138CE40D5B6F73001001006F7400126F7D00162B
S1138CF419556F710002113213331331341335 0DA058
S1138D047960010047121132133313341335 0B51B5
S1138D14792107FF5870FE5401F064454702700B34
S1138D240D114E2E113213334402700B0D114A04EC
S1138D340B5140F0732B47180CB8E80B47127A1306
S1138D4400000008440A0B72792A00804D020B517B
S1138D544020732B471C0CB8E80B47167A1300000A
S1138D64000844020B72792A01004D0611321333B1
S1138D740B5111321333113213331321333796A12
S1138D84000F10111011101110111011641A101A6878C1
S1138D941008131A7A170000001801006D700100FF
S1138DA4698201006F83000401006D7101006D721B
S1078DB401006D73D7
S1118DB801006D7401006D7501006D7654703D
S1138DC601006DF57A01000000080AF16818E87FD2
S1138DD6A87F460A0B716818E8F0A8F04704190043
S1138DE6402A6818F01050006898460A0B711AD585
S1138DF6400E6818470679000001401 00B710B7589
S1138E067A25000000064DEA7900000201006D751F
S1058E16547093
S1138E185E005B7E0F840F931AD51B730FB6402831
S1138E280FC30AE30FB26838E880175017700F832F
S1138E380FA06809100968890FD547080FC00AE011
S1138E487D0070000FB51B760FE64CD40FD547068F
S1138E5879010001400219111 7F10F905E005B5C64
S1058E68547041
S1138E6A5E005B7E7A37000000387A050000000452
S1138E7A0AF50F840F967A000000001001006DF0C6
S1138E8A191101006F7000545E0090DC0B970FE319
S1138E9A0B930BF37A160000000701006FF60034F8
S1138EAA790600030FC00B900BF001006FF0002846
S1138EBA7A140000000701006FF4002C5A008FE6B1
S1138ECA6838460C01006F7000346809587000FA5C
S1138EDA7A000000001001006DF019110FD05E0036

S1138EEA90DC0B9701006F70002801006FF00030CF
S1138EFA01006F74002C790E0003407C01006F702F
S1138F0A00301A916809FA0810311A0A4EFA1A80BF
S1138F1A684801F064101A916839FA0810311A0A7C
S1138F2A4EFA01006F720034010069F01A80682852
S1138F3A01F06401010069705E007A3601006FF086
S1138F4A00247A000000000401006DF07A01000099
S1098F5A00280AF10DE2FC
S1138F60096217F210320AD20FA05E0090940B9799
S1138F701B5E01006F7000301BF001006FF00030CA
S1138F801BF40DEE4C807926000346247A00000082
S1138F90000A01006DF00D6417F40FC110310AD1FE
S1138FA001006F7000540AC00AC05E007FE04022D7
S1138FB07A000000000A01006DF00D6417F40FC180
S1138FC010310AD101006F7000540AC00AC05E005C
S1138FD090940B971B561BF301006F7000341BF02A
S1138FE001006FF000340D6658C0FEDE7A170000F2
S10B8FF000385E005B5C547065
S1138FF85E005B7E7A370000000C0F860F940D60CD
S11390087910003F69C00FF10FE05E0082B27A0069
S1139018000000801006DF0191101006F700028AD
S11390285E0090DC0B971A800F8219550FF64010DB
S11390380B760FA00B700F8269407930000869C066
S1139048686847ECFB804004110B0B55686816B839
S113905847F66DF57A03000000080FA01A830FB1D5
S11390680FE05E00823A0B876940195069C001001E
S11390786DF30FE101006F7000285E007FE00B972E
S10F90887A170000000C5E005B5C547063
S11390945E005B7E0F860F9401006F7500180AD67D
S11390A41B760AD41B740FC319CC4022686817506B
S11390B468391751091009C00D0468E817F47900D9
S11390C4010001D053040D4C1B751B761B730FD584
S10B90D446DA5E005B5C547098
S11390DC5E005B7E1B870F8469F101006F73001ABE
S11390EC0FC51AE6400C0FD00B756E790001688919
S11390FC0B761FB645F00FC00B875E005B5C54709C
S9030000FD

```
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:

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、

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コースありますが、
<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。
ゴールまで80歩です。

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

==========

makefile.mak build.bat は複数のファイルを1個のプロジェクトとして

コンパイルするためのファイルです。

著作者:

しのみや　ひでみね

篠宮　　英峰