# 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
#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
```

```
/* 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,
 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 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

/* 時間に使用する定数の宣言 */
#define WOVICLOCKSIZE 1000000.0
/* 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 1000001
#define STOPCLOCKNO 1000002

/* 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 tag_SleepMSec(double sleepPerSec, long ms); /* ミリ秒待ち関数 */
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)
{
#ifndef USE_BCC
	return woviClock;
#else
	if(clock() / CLOCKS_PER_SEC < WOVICLOCKSIZE)
	{
		woviClock = clock() / CLOCKS_PER_SEC;
		return woviClock;
	}
	else
	{
```

```
        return woviClock;
    }
#endif
}

/* ミリ秒待ち関数 */
void tag_SleepMSec(double sleepPerSec, long ms)
{
    double cnt;
    double set;
    cnt = 0.0;
    set = ((double) ms) / 1000;
    while(cnt < set)
    {
        cnt += 1.0 / sleepPerSec;
    }
    return;
}

/* ミリ秒待ち関数 */
void SleepMSec(long ms)
{
    double start;
    double set;
    double end;
    start = getClock();
    set = ((double) ms) / 1000;
    end = start;
    while(end < start + set)
```

```
	{
		end = getClock();
#ifndef USE_BCC
		if(woviClock >= WOVICLOCKSIZE)
		{
			PrintF(Panel, "¥n");
			PrintF(Panel, "OVERWOVI");
			woviClock -= WOVICLOCKSIZE;
			tag_SleepMSec(SLEEP_PER_SEC, ms);
			break;
		}
#else
		if(clock() / CLOCKS_PER_SEC >= WOVICLOCKSIZE)
		{
			if(woviClock >= WOVICLOCKSIZE)
			{
				woviClock -= WOVICLOCKSIZE;
			}
			tag_SleepMSec(SLEEP_PER_SEC, ms);
			break;
		}
#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;
#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;
        }
    #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 < List->preClock)
			{
				woviClockCompare -= WOVICLOCKSIZE;
			}
			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
	if(clock() / CLOCKS_PER_SEC < WOVICLOCKSIZE)
	{
		woviClock = clock() / CLOCKS_PER_SEC;
	}
	else
	{
		woviClock += 1.0 / threadPerSec;
	}
#endif
	if(woviClock >= WOVICLOCKSIZE)
	{
		PrintF(Panel, "¥n");
		PrintF(Panel, "OVERWOVI");
```

```
            woviClock -= WOVICLOCKSIZE;
        }
        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
```

```
/* main.h */

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

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

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 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)
        {
            delete_(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
    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 main.obj
$(TARGET): $(OBJS)
	$(LL) $(LFLAGS) $(LIB) \
	$(OBJS) c0x32.obj,$(TARGET),,cw32.lib import32.lib
main.obj : main.c main.h Timer.h Panel.h C.h
	$(CC) $(CFLAGS) -c main.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
.DATA.L     int_error
.DATA.L     int_error
.DATA.L     int_error

.DATA.L     int_error
.DATA.L     int_error
.DATA.L     int_error
.DATA.L     int_error

.DATA.L     int_error
.DATA.L     int_error
.DATA.L     int_error
.DATA.L     int_error

IRQ0:  .DATA.L     int_error
IRQ1:  .DATA.L     int_error
IRQ2:  .DATA.L     int_error
IRQ3:  .DATA.L     int_error
IRQ4:  .DATA.L     int_error
IRQ5:  .DATA.L     usb_interrupt
```

```
	.SECTION	ITU0,CODE,LOCATE=H'000060
	.DATA.L		int_error
	.DATA.L		int_error
	.DATA.L		_ITU_OVI_0
	.DATA.L		int_error

	.DATA.L		int_error
	.DATA.L		int_error
	.DATA.L		int_error
	.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'8000, er0		; 転送元(8000)
	mov.l	#H'0FFEF10, 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 usbtest
PRINT usbtest
INPUT start,main,Timer,Panel,sci,lcd,usb
LIB c:\h8\akic\c38hab
START R(0FFEF10),P(200),D(8000),C(9000)
ROM (D,R)
EXIT
```

```
@rem build.bat
set CurrentDir="D:\C_Thread\Thread_Work"
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% 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 start.asm >> error.txt
l38h -subcommand=usbtest.sub >> error.txt
c38h usbtest >> error.txt
del %CurrentDir%\*.obj >> error.txt
del %CurrentDir%\usbtest.abs >> error.txt
error.txt
exit
```

# 出力ファイル

```
 Start         Length     Name                   Class
0001:00401000 00000C064H _TEXT                  CODE
0002:0040E000 000002F4CH _DATA                  DATA
0003:00410F4C 000000960H _BSS                   BSS
0004:00000000 0000000A4H _TLS                   TLS
```

PROGRAM NAME =

```
1                   1       .CPU 300HA
2 000000            2       .SECTION    V,CODE,LOCATE=H'000000
3                   3       .IMPORT _main
4                   4       .IMPORT _usb_int
5                   5       .IMPORT _InterruptITU0
6                   6
7 000000 00000000   7       .DATA.L     _start      ;リセットベクトル
8 000004 00000000   8       .DATA.L     int_error
9 000008 00000000   9       .DATA.L     int_error
10 00000C 00000000  10      .DATA.L     int_error
11                  11
12 000010 00000000  12      .DATA.L     int_error
13 000014 00000000  13      .DATA.L     int_error
14 000018 00000000  14      .DATA.L     int_error
15 00001C 00000000  15      .DATA.L     int_error
16                  16
17 000020 00000000  17      .DATA.L     int_error
18 000024 00000000  18      .DATA.L     int_error
19 000028 00000000  19      .DATA.L     int_error
20 00002C 00000000  20      .DATA.L     int_error
21                  21
22 000030 00000000  22  IRQ0:  .DATA.L     int_error
23 000034 00000000  23  IRQ1:  .DATA.L     int_error
24 000038 00000000  24  IRQ2:  .DATA.L     int_error
```

```
25 00003C 00000000          25    IRQ3:   .DATA.L       int_error
26 000040 00000000          26    IRQ4:   .DATA.L       int_error
27 000044 00000000          27    IRQ5:   .DATA.L       usb_interrupt
28                          28
29 000060                   29          .SECTION      ITU0,CODE,LOCATE=H'000060
30 000060 00000000          30          .DATA.L       int_error
31 000064 00000000          31          .DATA.L       int_error
32 000068 00000000          32          .DATA.L       _ITU_OVI_0
33 00006C 00000000          33          .DATA.L       int_error
34                          34
35 000070 00000000          35          .DATA.L       int_error
36 000074 00000000          36          .DATA.L       int_error
37 000078 00000000          37          .DATA.L       int_error
38 00007C 00000000          38          .DATA.L       int_error
39                          39
40                          40    ;------------------------------------------------------
41 000000                   41          .SECTION      P,CODE,ALIGN=2
42 000000                   42    _start:
43 000000 7A0700FFFF10      43          mov.l   #H'0FFFF10,er7
44                          44
45                          45    ;初期化付きデータを使用する場合、RAMに転送する
46 000006 7A0000008000      46          mov.l   #H'8000, er0              ; 転送元(8000)
47 00000C 7A0100FFEF10      47          mov.l   #H'0FFEF10, er1             ; 転送先
48 000012 7A0200000000      48          mov.l   #DATA_END, er2              ; 転送終了
49 000018                   49    init_loop:
50 000018 1F92              50          cmp.l   er1, er2
51 00001A 58700008          51          beq     init_end
52 00001E 6C0B              52          mov.b   @er0+, r3l
53 000020 689B              53          mov.b   r3l, @er1
```

```
  54 000022 0B71            54          inc.l   #1, er1
  55 000024 40F2            55          bra     init_loop
  56 000026                 56     init_end:
  57 000026 5E000000        57          jsr     @_main
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***     05/02/17 00:05:56
PAGE    2
PROGRAM NAME =


  58                        58
  59                        59     ; 割り込み未使用
  60 00002A                 60     int_error:
  61 00002A 5670            61          rte
  62                        62
  63 00002C                 63     usb_interrupt:
  64 00002C 01006DF0        64          push.l  er0
  65 000030 01006DF1        65          push.l  er1
  66 000034 01006DF2        66          push.l  er2
  67 000038 01006DF3        67          push.l  er3
  68 00003C 01006DF4        68          push.l  er4
  69 000040 01006DF5        69          push.l  er5
  70 000044 01006DF6        70          push.l  er6
  71 000048 5E000000        71          jsr     @_usb_int
  72 00004C 01006D76        72          pop.l   er6
  73 000050 01006D75        73          pop.l   er5
  74 000054 01006D74        74          pop.l   er4
  75 000058 01006D73        75          pop.l   er3
  76 00005C 01006D72        76          pop.l   er2
  77 000060 01006D71        77          pop.l   er1
  78 000064 01006D70        78          pop.l   er0
```

```
 79 000068 5670            79          rte
 80                  80
 81 00006A              81    _ITU_OVI_0:
 82 00006A 01006DF0      82          push.l  er0
 83 00006E 01006DF1      83          push.l  er1
 84 000072 01006DF2      84          push.l  er2
 85 000076 01006DF3      85          push.l  er3
 86 00007A 01006DF4      86          push.l  er4
 87 00007E 01006DF5      87          push.l  er5
 88 000082 01006DF6      88          push.l  er6
 89 000086 5E000000      89          jsr @_InterruptITU0
 90 00008A 01006D76      90          pop.l   er6
 91 00008E 01006D75      91          pop.l   er5
 92 000092 01006D74      92          pop.l   er4
 93 000096 01006D73      93          pop.l   er3
 94 00009A 01006D72      94          pop.l   er2
 95 00009E 01006D71      95          pop.l   er1
 96 0000A2 01006D70      96          pop.l   er0
 97 0000A6 5670            97          rte
 98                  98
 99                  99    ;--------------------------------------------------------
100                  100   ;割り込み許可、禁止ルーチン
101                  101
102                  102         .EXPORT _EnableInterrupt,_DisableInterrupt
103 0000A8             103   _EnableInterrupt:
104 0000A8 063F          104         andc.b  #H'3f,ccr
105 0000AA 5470          105         rts
106 0000AC             106   _DisableInterrupt:
107 0000AC 04C0          107         orc.b   #H'c0,ccr
```

```
 108 0000AE 5470            108          rts
 109                        109
 110                        110    ;--------------------------------------------------------
 111 000000                 111          .SECTION      D,DATA
 112                        112
 113                        113
 114 000000                 114          .SECTION      B,DATA
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***     05/02/17 00:05:56
PAGE    3
PROGRAM NAME =

 115 000000 00000002        115    DATA_END:     .RES.W        1
 116                        116
 117                        117          .END
 *****TOTAL ERRORS      0
 *****TOTAL WARNINGS    0
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***     05/02/17 00:05:56
PAGE    4

*** CROSS REFERENCE LIST

NAME                    SECTION  ATTR VALUE          SEQUENCE

B                       B      SCT  00000000    114*
D                       D      SCT  00000000    111*
DATA_END                       B          00000000    48   115*
IRQ0                    V          00000030    22*
IRQ1                    V          00000034    23*
```

```
IRQ2                V       00000038   24*
IRQ3                V       0000003C   25*
IRQ4                V       00000040   26*
IRQ5                V       00000044   27*
ITU0                ITU0    SCT  00000060   29*
P                   P       SCT  00000000   41*
V                   V       SCT  00000000    2*
_DisableInterrupt           P       EXPT 000000AC   102  106*
_EnableInterrupt            P       EXPT 000000A8   102  103*
_ITU_OVI_0                  P          0000006A    32   81*
_InterruptITU0                    IMPT 00000000     5   89
_main                             IMPT 00000000     3   57
_start                      P          00000000     7   42*
_usb_int                          IMPT 00000000     4   71
init_end                    P          00000026    51   56*
init_loop                   P          00000018    49*  55
int_error                   P          0000002A     8    9   10   12   13   14   15   17   18   19   20   22
                                         23   24   25   26   30   31   33   35   36   37   38   60*
usb_interrupt               P          0000002C    27   63*
*** H8/300H ASSEMBLER (Evaluation software) Ver.1.0 ***    05/02/17 00:05:56
PAGE    5

*** SECTION DATA LIST

SECTION                   ATTRIBUTE    SIZE         START

V                         ABS-CODE    0000048        000000
ITU0                      ABS-CODE    0000020        000060
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=usbtest.sub

LINK SUBCOMMANDS

```
OUTPUT usbtest
PRINT usbtest
INPUT start,main,Timer,Panel,sci,lcd,usb
LIB c:\h8\akic\c38hab
START R(0FFEF10),P(200),D(8000),C(9000)
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'00000047   H'00000048
                                start                    start

* TOTAL ADDRESS *          H'00000000   -   H'00000047   H'00000048

ATTRIBUTE :  CODE  NOSHR

ITU0                H'00000060   -   H'0000007F   H'00000020
                                start                    start

* TOTAL ADDRESS *          H'00000060   -   H'0000007F   H'00000020

ATTRIBUTE :  CODE  NOSHR

P                   H'00000200   -   H'000002AF   H'000000B0
                                start                    start
                    H'000002B0   -   H'00001115   H'00000E66
                                main                     main
                    H'00001116   -   H'00001799   H'00000684
                                Timer                    Timer
                    H'0000179A   -   H'00001895   H'000000FC
```

```
                    Panel                   Panel
H'00001896  -  H'00001967  H'000000D2
                    sci                     sci
H'00001968  -  H'00001BA5  H'0000023E
                    lcd                     lcd
H'00001BA6  -  H'00002697  H'00000AF2
                    usb                     usb
H'00002698  -  H'000026D1  H'0000003A
                    rand                    rand
H'000026D2  -  H'0000272F  H'0000005E
                    sprintf                 sprintf
H'00002730  -  H'00002759  H'0000002A
                    strcmp                  strcmp
H'0000275A  -  H'00002775  H'0000001C
                    strlen                  strlen
H'00002776  -  H'000027A5  H'00000030
                    vsprintf                vsprintf
H'000027A6  -  H'00002A7F  H'000002DA
                    addd3                   addd3
H'00002A80  -  H'00002CB1  H'00000232
                    divd3                   divd3
H'00002CB2  -  H'00002CBD  H'0000000C
                    eqd3                    eqd3


            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'00002CBE  -  H'00002CCD  H'00000010
                    ged3                    ged3
H'00002CCE  -  H'00002CDD  H'00000010
                    ltd3                    ltd3
H'00002CDE  -  H'00002D23  H'00000046
                    ltod3                   ltod3
H'00002D24  -  H'00002D41  H'0000001E
                    mv83                    mv83
H'00002D42  -  H'00002D4F  H'0000000E
                    ned3                    ned3
H'00002D50  -  H'00002D71  H'00000022
                    spregld3                spregld3
H'00002D72  -  H'00002D99  H'00000028
                    spregsv3                spregsv3
H'00002D9A  -  H'00004B47  H'00001DAE
                    _fmtout                 _fmtout
H'00004B48  -  H'00004C03  H'000000BC
                    cmpd3                   cmpd3
H'00004C04  -  H'00004C29  H'00000026
                    divl3                   divl3
```

```
                H'00004C2A   -   H'00004C49   H'00000020
                                 mull3                mull3
                H'00004C4A   -   H'0000502F   H'000003E6
                                 _dti                 _dti
                H'00005030   -   H'000051D3   H'000001A4
                                 _its                 _its
                H'000051D4   -   H'0000522D   H'0000005A
                                 memcpy               memcpy
                H'0000522E   -   H'00005269   H'0000003C
                                 divul3               divul3
                H'0000526A   -   H'00005291   H'00000028
                                 _allzero             _allzero
                H'00005292   -   H'00005389   H'000000F8
                                 _calcnpw             _calcnpw
                H'0000538A   -   H'0000542D   H'000000A4
                                 _log10               _log10
                H'0000542E   -   H'000054A5   H'00000078
                                 _lsfts               _lsfts
                H'000054A6   -   H'000054D3   H'0000002E
                                 _pow5                _pow5
                H'000054D4   -   H'0000554D   H'0000007A
                                 _rsfts               _rsfts
                H'0000554E   -   H'000055F9   H'000000AC
                                 _sub                 _sub
                H'000055FA   -   H'0000569D   H'000000A4
                                 _unpack              _unpack



                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'0000569E   -   H'000056DB   H'0000003E
                                 memcmp               memcmp
                H'000056DC   -   H'00005763   H'00000088
                                 _mult64              _mult64
                H'00005764   -   H'000058C5   H'00000162
                                 _power               _power
                H'000058C6   -   H'000059AF   H'000000EA
                                 _rnd                 _rnd
                H'000059B0   -   H'00005A4B   H'0000009C
                                 _setsbit             _setsbit
                H'00005A4C   -   H'00005B51   H'00000106
                                 frexp                frexp
                H'00005B52   -   H'00005C87   H'00000136
                                 modf                 modf
                H'00005C88   -   H'00005CA9   H'00000022
                                 dslc3                dslc3
                H'00005CAA   -   H'00005CCB   H'00000022
```

```
                        dsruc3                  dsruc3
                H'00005CCC  -  H'00005D45  H'0000007A
                        dtol3                   dtol3
                H'00005D46  -  H'00005D7B  H'00000036
                        itod3                   itod3
                H'00005D7C  -  H'00006069  H'000002EE
                        muld3                   muld3
                H'0000606A  -  H'000060BB  H'00000052
                        _duchek                 _duchek
                H'000060BC  -  H'0000610D  H'00000052
                        _lsft                   _lsft
                H'0000610E  -  H'0000629B  H'0000018E
                        _mult                   _mult
                H'0000629C  -  H'00006337  H'0000009C
                        _pow10                  _pow10
                H'00006338  -  H'0000637F  H'00000048
                        _add                    _add
                H'00006380  -  H'000063AF  H'00000030
                        memset                  memset

* TOTAL ADDRESS *        H'00000200  -  H'000063AF  H'000061B0


ATTRIBUTE :  DATA  NOSHR ROM

D               H'00008000  -  H'00008000  H'00000000
                        start                   start
                H'00008000  -  H'0000800F  H'00000010
                        usb                     usb



                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'00008000  -  H'0000800F  H'00000010


ATTRIBUTE :  DATA  NOSHR

C               H'00009000  -  H'0000927B  H'0000027C
                        main                    main
                H'0000927C  -  H'000092D5  H'0000005A
                        Timer                   Timer
                H'000092D6  -  H'000092E0  H'0000000B
                        Panel                   Panel
                H'000092E2  -  H'0000939B  H'000000BA
                        usb                     usb
                H'0000939C  -  H'000093A3  H'00000008
                        _fmtout                 _fmtout
                H'000093A4  -  H'000094A3  H'00000100
                        _ctype                  _ctype
```

```
                    H'000094A4   -   H'0000952B   H'00000088
                                     _its                    _its
                    H'0000952C   -   H'00009533   H'00000008
                                     _log10                  _log10
                    H'00009534   -   H'00009613   H'000000E0
                                     _pow5                   _pow5
                    H'00009614   -   H'00009717   H'00000104
                                     _power                  _power
                    H'00009718   -   H'0000971F   H'00000008
                                     frexp                   frexp
                    H'00009720   -   H'00009727   H'00000008
                                     modf                    modf

* TOTAL ADDRESS *           H'00009000   -   H'00009727   H'00000728


ATTRIBUTE :  DATA  NOSHR RAM

R                   H'00FFEF10   -   H'00FFEF10   H'00000000
                                     start                   start
                    H'00FFEF10   -   H'00FFEF1F   H'00000010
                                     usb                     usb

* TOTAL ADDRESS *           H'00FFEF10   -   H'00FFEF1F   H'00000010


ATTRIBUTE :  DATA  NOSHR

B                   H'00FFEF20   -   H'00FFEF21   H'00000002
                                     start                   start
                    H'00FFEF22   -   H'00FFEF89   H'00000068
                                     main                    main



                    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

B                   H'00FFEF8A   -   H'00FFF153   H'000001CA
                                     Timer                   Timer
                    H'00FFF154   -   H'00FFF1D3   H'00000080
                                     Panel                   Panel
                    H'00FFF1D4   -   H'00FFF223   H'00000050
                                     sci                     sci
                    H'00FFF224   -   H'00FFF263   H'00000040
                                     lcd                     lcd
                    H'00FFF264   -   H'00FFF537   H'000002D4
                                     usb                     usb
                    H'00FFF538   -   H'00FFF573   H'0000003C
```

```
                    _fmtout                 _fmtout
H'00FFF574   -   H'00FFF577   H'00000004
                    _rnext                  _rnext
H'00FFF578   -   H'00FFF579   H'00000002
                    _errno                  _errno

* TOTAL ADDRESS *          H'00FFEF20   -   H'00FFF579   H'0000065A


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

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

        SYMBOL  NAME                ADDR          TYPE

        $ADDD$3                   H'00002870       DAT
        $CMPD$3                   H'00004B48       DAT
        $DIVD$3                   H'00002B1A       DAT
        $DIVL$3                   H'00004C04       DAT
        $DIVUL$3                  H'0000522E       DAT
        $DSLC$3                   H'00005C88       DAT
        $DSRUC$3                  H'00005CAA       DAT
        $DTOL$3                   H'00005CCC       DAT
        $EQD$3                    H'00002CB2       DAT
        $GED$3                    H'00002CBE       DAT
        $ITOD$3                   H'00005D46       DAT
        $LTD$3                    H'00002CCE       DAT
        $LTOD$3                   H'00002CDE       DAT
        $MULD$3                   H'00005E32       DAT
        $MULL$3                   H'00004C2A       DAT
        $MV8$3                    H'00002D24       DAT
        $NED$3                    H'00002D42       DAT
        $SUBD$3                   H'00002840       DAT
        $sp_regld$3               H'00002D50       DAT
        $sp_regsv$3               H'00002D72       DAT
        _Clear                  H'0000179A       ENT
        _ClearLCD                 H'00001A86       ENT
        _Cnt                    H'00FFEF22       DAT
        _Destroy                  H'00000F80       ENT
        _DisableInterrupt         H'000002AC       DAT
        _DispUSBPort              H'00001CB6       ENT
        _EnableInterrupt          H'000002A8       DAT
        _GetSCI                   H'000018C6       ENT
        _GetSW                    H'000010C4       ENT
        _H8init                   H'000010EA       ENT
        _Init                   H'00000D76       ENT
        _InitITU                  H'000016D4       ENT
        _InitLCD                  H'00001A16       ENT
        _InitSCI                  H'00001896       ENT
        _InitUSB                  H'00001BBC       ENT
        _InterruptITU0            H'0000170A       ENT
        _LCDOut4                  H'000019BE       ENT
        _LocateLCD                H'00001AC6       ENT
        _PrintF                   H'000017C2       ENT
        _PrintLCD                 H'00001AEA       ENT
        _PrintSCI                 H'000018E6       ENT
```

| | | |
|---|---|---|
| _PutLCD | H'00001A9A | ENT |
| _PutSCI | H'000018B6 | ENT |
| _Repaint | H'000004DA | ENT |
| _Run | H'00000546 | ENT |
| _ScanSCI | H'000018D6 | ENT |
| _SetLED | H'00001076 | ENT |
| _SleepMSec | H'000011AC | ENT |
| _Start | H'00001498 | ENT |
| _Stop | H'000014C6 | ENT |
| _Thread_Start | H'0000155E | ENT |



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

| SYMBOL NAME | ADDR | TYPE |
|---|---|---|
| _Thread_Toggle | H'0000159C | ENT |
| _Thread_checkAllDelete | H'000014E4 | ENT |
| _Thread_checkStayAnother | H'000014FE | ENT |
| _Thread_getThread | H'00001524 | ENT |
| __add | H'00006338 | ENT |
| __allzero | H'0000526A | ENT |
| __calcnpw | H'00005292 | ENT |
| __ctype | H'000093A4 | DAT |
| __dti | H'00004C4A | ENT |
| __duchek | H'0000606A | ENT |
| __errno | H'00FFF578 | DAT |
| __fmtout | H'00002D9A | ENT |
| __its | H'00005030 | ENT |
| __log10 | H'0000538A | ENT |
| __lsft | H'000060BC | ENT |
| __lsfts | H'0000542E | ENT |
| __mult | H'0000610E | ENT |
| __mult64 | H'000056DC | ENT |
| __pow10 | H'0000629C | ENT |
| __pow5 | H'000054A6 | ENT |
| __power | H'00005764 | ENT |
| __rnd | H'000058C6 | ENT |
| __rnext | H'00FFF574 | DAT |
| __rsfts | H'000054D4 | ENT |
| __setsbit | H'000059B0 | ENT |
| __sub | H'0000554E | ENT |
| __unpack | H'000055FA | ENT |
| _countUpNextRun | H'000012EC | ENT |
| _delete_ | H'00001464 | ENT |
| _frexp | H'00005A4C | ENT |
| _getClock | H'00001116 | ENT |
| _get_inbufflen | H'00002670 | ENT |
| _get_outbufflen | H'00002684 | ENT |
| _i_cnt | H'00FFEF26 | DAT |
| _initWOVI | H'0000177E | ENT |
| _init_usbbuff | H'000024D4 | ENT |
| _j_cnt | H'00FFEF28 | DAT |
| _main | H'000002B0 | ENT |

| | | |
|---|---|---|
| _memcmp | H'0000569E | ENT |
| _memcpy | H'000051D4 | ENT |
| _memset | H'00006380 | ENT |
| _modf | H'00005B52 | ENT |
| _new_Thread | H'0000130C | ENT |
| _nextRun | H'000012A8 | ENT |
| _rand | H'00002698 | ENT |
| _read_buff | H'00002612 | ENT |
| _read_outbuff | H'000025B4 | ENT |
| _sprintf | H'000026D2 | ENT |
| _strcmp | H'00002730 | ENT |
| _strlen | H'0000275A | ENT |
| _tag_SleepMSec | H'00001128 | ENT |



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

| SYMBOL NAME | ADDR | TYPE |
|---|---|---|
| _th | H'00FFEF2A | DAT |
| _th1 | H'00FFEF32 | DAT |
| _th101 | H'00FFEFCE | DAT |
| _th102 | H'00FFEFEC | DAT |
| _th111 | H'00FFF00A | DAT |
| _th112 | H'00FFF028 | DAT |
| _th113 | H'00FFF046 | DAT |
| _th114 | H'00FFF064 | DAT |
| _th119 | H'00FFF082 | DAT |
| _th120 | H'00FFF0A0 | DAT |
| _th121 | H'00FFF0BE | DAT |
| _th122 | H'00FFF0DC | DAT |
| _th123 | H'00FFF0FA | DAT |
| _th130 | H'00FFF118 | DAT |
| _th131 | H'00FFF136 | DAT |
| _th19 | H'00FFEF42 | DAT |
| _th20 | H'00FFEF46 | DAT |
| _usb_int | H'00001D1A | ENT |
| _vsprintf | H'00002776 | ENT |
| _wovi | H'00001732 | ENT |
| _woviClock | H'00FFEF8A | DAT |
| _woviInit | H'000016A2 | ENT |
| _woviRun | H'000015E0 | ENT |
| _woviThreadFirst | H'00FFEF92 | DAT |
| _woviThreadLast | H'00FFEFB0 | DAT |
| _write_buff | H'0000254E | ENT |
| _write_inbuff | H'000024F2 | ENT |

S00E000075736274657374204D4F54D8
S1130000000002000000022A0000022A0000022A67
S11300100000022A0000022A0000022A0000022A2D
S11300200000022A0000022A0000022A0000022A1D
S10B00300000022A0000022A6D
S11300380000022A0000022A0000022A0000022C03
S11300600000022A0000022A0000026A0000022A9D
S11300700000022A0000022A0000022A0000022ACD
S11302007A0700FFFF107A00000080007A0100FFE8
S1130210EF107A0200FFEF201F92587000086C0B5A
S1130220689B0B7140F25E0002B0567001006DF0E6
S113023001006DF101006DF201006DF301006DF439
S113024001006DF501006DF65E001D1A01006D766B
S113025001006D7501006D7401006D7301006D7215
S113026001006D7101006D70567001006DF00100A9
S11302706DF101006DF201006DF301006DF40100F9
S11302806DF501006DF65E00170A01006D76010041
S11302906D7501006D7401006D7301006D720100D5
S11302A06D7101006D705670063F547004C0547038
S11302B05E002D727A370000000A790B00011933B2
S11302C00FF47A0600FFEF4A19550B5579257FFF86
S11302D04DF85C000E145E0018965E001A160D3879
S11302E00D305C000D900D380DB05C000D880D389D
S11302F0790000025C000D7E0D38790000035C007C
S11303000D745E001BBC7FF472507FF570505E006D
S113031002A80D3228F117506DF07A00000090000A
S113032001006DF05E0018E60B970B877A00000062
S1130330900F01006DF05E001AEA0B9718886EC8E3
S113034000036EC800026EC8000168C85E00177E15
S11303507A00000007D05E0011AC7900001E5E0039
S1130360130C0F857900001F5E00130C01006FF062
S113037000040FD05E00149801006F7000045E004B
S1130380149801006B200000927801006DF00100C9
S10E03906B200000927401006DF05E12
S113039B0017320B970B975E0014FE792000014672
S11303ABD65E00179A0FD05E0014647A010000909A
S11303BB1B0D305E0017C27A00000007D05E0011E0
S11303CBAC5E00179A19550D505C000CEC0D0D0D1E
S11303DB5117F10AC16819D90117D1660147540D99
S11303EBB80D505C000C840D5009B06DF07A000011
S11303FB00902C01006DF00FE05E0026D20B970BE3
S113040B8701006DF67A000000903101006DF05EFC
S113041B0018E60B970B970FE05E00275A09B00DF8
S113042B010FE05E00254E01006DF65E001AEA0B2C
S113043B9740080D380D505C000C300DD017F50F9D
S113044BC10AD168980B557925000458D0FF785E03
S113045B0018D60D0047165E0018C617D00D055EA3
S111046B001A9A68ED0DB10FE05E00254E5E9B
S11304790026700D004738790100400FE05E002621
S1130489120D057A010000903501006DF15E001A25
S1130499EA0B9701006DF65E001AEA0B9701006DEE
S11304A9F65E0018E60B970D510FE05E00254E0D21
S11304B928790000035C000BB40D20795000010D6D
S11304C90219550B55792527104DF85A0003D054B5
S11304D9705E002D727A0500FFEF2219665E001720
S11304E99A19EE400E7A01000090370D605E0017ED
S11304F9C20B5E69501D0E4DEC7A01000090390D57

S1130509605E0017C27A010000903D0D605E00171E
S10A0519C219EE400E7A0146
S1130520000090370D605E0017C20B5E6F50000233
S11305301D0E4DEA7A010000903F0D605E0017C268
S11305405E002D5054705E002D727A370000000A51
S11305507A03000000017A04000007D019550F86C2
S1130560188886EF800036EF800026EF8000168F850
S113057069607920000146365C00FF5E7A0000FF67
S1130580EF2269010B5169815E00269817F0790209
S1130590000901D053207918000A790000645280C1
S11305A017F00F810FE05E0012A85A000D6A696010
S11305B07920000246365C00FF207A0000FFEF241A
S11305C069010B5169815E00269817F079020009D1
S11305D001D053207918000A79000064528017F083
S11305E00F810FE05E0012A85A000D6A696079203E
S11305F0000B586000B601006F6000127A20000003
S1130600000146205E00179A7A01000090430D50C6
S10906105E0017C27A012F
S10A06160000076C0FE05E1A
S113061D0012EC5A000D6A01006F6000127A20007F
S113062D00000246265E00179A7A01000090540DD1
S113063D505E0017C27A010000905B0D505E0017EB
S113064DC20FE05E0014C65A000D6A01006F600010
S113065D127A2000000003461E5E00179A7A0100ED
S113066D0090660D505E0017C27A01000005DC0F85
S113067DE05E0012EC402401006F6000127A20004E
S113068D00000446165E00179A7A01000090770D5C
S10F069D505E0017C20FE05E0014C65A46
S11306A9000D6A69607920000C586000A801006F89
S11306B96000127A200000000146205E00179A7A32
S11306C901000090880D505E0017C27A0100000DE9
S11306D9480FE05E0012EC5A000D6A01006F6000DA
S11306E9127A200000000246205E00179A7A010060
S11306F90090990D505E0017C27A0100000D480F52
S1130709E05E0012EC5A000D6A01006F6000127A74
S11307192000000003461E5E00179A7A010000902C
S1130729AA0D505E0017C27A0100000D480FE05E62
S10E07390012EC401C5E00179A7A01CE
S1130744000090BB0D505E0017C20FE05E0014C69C
S11307540FE05E0014645A000D6A69607920000D8D
S1130764586000A801006F6000127A2000000001A5
S113077446205E00179A7A01000090C30D505E0074
S1130784417C27A01000013EC0FE05E0012EC5A006A
S11307940D6A01006F6000127A20000000024620F7
S11307A45E00179A7A01000090D40D505E0017C2C0
S11307B47A01000013EC0FE05E0012EC5A000D6A9C
S11307C401006F6000127A2000000003461E5E00E1
S10707D4179A7A01F2
S11307D8000090E50D505E0017C27A01000013EC8B
S11307E80FE05E0012EC401C5E00179A7A010000CD
S11307F890F60D505E0017C20FE05E0014C60FE0BE
S11308085E0014645A000D6A69607920000E58600E
S1130818800E401006F6000127A2000000001462006
S11308285E00179A7A01000090FE0D505E0017C211
S11308387A01000017700FE05E0012EC5A000D6A8F
S113084801006F6000127A200000000246405E003B
S1130858179A7A010000910F0D505E0017C27A01B2

S10A0868000017700FE05EB2
S113086F0012EC7A01000091160D505E0017C27949
S113087F00000B5E00155E0F867A01000005DC5E3B
S113088F0012EC5A000D6A01006F6000127A20000B
S113089F000003461E5E00179A7A01000091210D96
S11308AF505E0017C27A01000017700FE05E00124E
S11308BFEC403801006F6000127A200000000446FC
S11308CF2A7900000B5E0015240F8247060FA05EE6
S11308DF0014647A01000091320D505E0017C20FAD
S11308EFE05E0014C60FE05E0014645A000D6A69DF
S11308FF607920001346440FB10FE05E0012A81970
S113090FDD0DD05C0007AE0DD117F10FF20A92681F
S113091F2ADA0117D2660247160DD0791000145E3A
S113092F00159C7A01000003E80FE05E0012A80B8C
S113093F5D792D00044DCA5A000D6A696079200054
S113094F1446187A01000091390D505E0017C20F3B
S113095FC10FE05E0012EC5A000D6A696079200046
S113096F1546187A010000913B0D505E0017C20F18
S113097FC10FE05E0012EC5A000D6A696079200026
S113098F1646187A010000913D0D505E0017C20FF5
S113099FC10FE05E0012EC5A000D6A696079200006
S10F09AF1746187A010000913F0D505EBE
S11309BB0017C20FC10FE05E0012EC5A000D6A69FB
S11309CB607920001E465E01006F60001246240111
S11309DB006F6000120B7001006FE0001228D6E865
S11309EB0E38D67A01000003E80FE05E0012A85A16
S11309FB000D6A01006F6000127A2000000001589D
S1130A0B60035C01006F6000121B7001006FE0005C
S1130A1B127FD670007A01000003E80FE05E00122C
S1130A2BA85A000D6A69607920001F5860033001D2
S1130A3B006F600012460C1A910FE05E0012EC5A25
S1130A4B000D6A01006F6000127A2000000001465E
S1130A5B247A010000903D0D505E0017C27A01000D
S1130A6B0091410D505E0017C20FC10FE05E0012E3
S1130A7BEC5A000D6A01006F6000127A200000002F
S1130A8B02463019DD0DD00B505E00130C0DD2173F
S10E0A9BF210321032010078A06BA0B3
S1130AA600FFEF2A0B5D792D00024DDE0FB10FE03B
S1130AB65E0012EC5A000D6A01006F6000127A2084
S1130AC60000000346566B2000FFEF227920000D3D
S1130AD64D1A790000016BA000FFEF2601006F603D
S1130AE600120B7001006FE0001240246B2000FF20
S1130AF6EF247920000D4D18790000026BA000FF4A
S1130B06EF2601006F6000120B7001006FE0001208
S1130B160FB10FE05E0012A85A000D6A01006F6064
S1130B2600127A2000000004465A5E00179A6B20D2
S1130B3600FFEF2679200001460E7A01000091524C
S1130B460D505E0017C240186B2000FFEF26792078
S1100B560002460C7A01000091630D505E11
S1130B630017C201006B2000FFEF2A5E001464012B
S1130B73006B2000FFEF2E5E0014640FC10FE05ED5
S1130B830012EC5A000D6A01006F6000127A200014
S1130B93000005461C5E00179A7A010000901B0DA6
S1130BA3505E0017C20FC10FE05E0012EC5A000D36
S1130BB36A01006F6000127A2000000006461C5E83
S1130BC300179A7A01000091740D505E0017C20F4B
S10B0BD3C10FE05E0012EC5AB1

S1130BDB000D6A01006F6000127A200000000746C7
S1130BEB365E00179A19DD0DD07910000B5E0013DA
S1130BFB0C0DD217F210321032010078A06BA0004B
S1130C0BFFEF320B5D792D00044DDC0FB10FE05E6E
S1130C1B0012EC5A000D6A01006F6000127A20007B
S1130C2B00000846245E0014FE79200002460E01E4
S1130C3B006F6000120B7001006FE000120FB10F19
S1130C4BE05E0012A85A000D6A01006F6000127A71
S1130C5B2000000009460C0FC10FE05E0012EC5A96
S1130C6B000D6A01006F6000127A200000000A4633
S1130C7B1C5E00179A7A010000901B0D505E001743
S1130C8BC20FC10FE05E0012EC5A000D6A01006F38
S10F0C9B6000127A200000000B461C5E73
S1130CA700179A7A01000091850D505E0017C20F55
S1130CB7C10FE05E0012EC5A000D6A01006F60007D
S1130CC7127A200000000C4632790000135E0013ED
S1130CD70C01006BA000FFEF425E0014987900003F
S1130CE7145E00130C01006BA000FFEF465E0014B7
S1130CF7980FB10FE05E0012EC406801006F6000CF
S1130D07127A200000000D462E5E0014FE792000A3
S1130D1703461A01006B2000FFEF425E00146401D3
S1130D27006F6000120B7001006FE000120FB10F2C
S1130D37E05E0012A8402C01006F6000127A2000C9
S1130D4700000E460A0FC10FE05E0012EC401401CB
S1130D57006F6000127A200000000F46060FE05E66
S10D0D670014647A170000000A5E0E
S1130D71002D5054705E002D720F86790400097A9C
S1130D810500FFEF22696079200001462619006 9F9
S1130D91D05E00269817F001D053407918000A79E4
S1130DA100001E528017F00F810FE05E0012A85A57
S1130DB1000F7A6960792000024628190 06FD0007C
S1130DC1025E00269817F001D053407918000A7982
S1130DD100001E528017F00F810FE05E0012A85A27
S1130DE1000F7A696C792C00034D36792C00084E7B
S1130DF13069601B5017F010300A85190069D05E05
S1130E0100269817F001D053407918000A790000A1
S1130E11C8528017F00F810FE05E0012A85A000F2D
S1130E217A7A04000017C2195569607920000B46CC
S1130E311A7A01000091960D505D407A0100000578
S1130E41DC0FE05E0012EC5A000F7A696079200032
S1130E510C461A7A010000919E0D505D407A010003
S1090E61000BB80FE05E78
S1130E670012EC5A000F7A69607920000D46247A44
S1130E77010000903D0D505D407A01000091A80DDF
S1130E87505D407A01000011940FE05E0012EC5AA6
S1130E97000F7A69607920000E461A7A01000091E3
S1130EA7B00D505D407A01000017700FE05E00122D
S1130EB7EC5A000F7A7A03000007D06960792000A3
S1130EC71446245E00179A7A01000091BA0D505D0B
S1130ED7407A010000903D0D505D400FB10FE05E79
S1130EE70012EC5A000F7A69607920001546225EDA
S1080EF700179A7A01C7
S1130EFC000091CB0D505D407A010000903D0D50E8
S1130F0C5D400FB10FE05E0012EC40626960792026
S1130F1C001646225E00179A7A01000091DC0D50F0
S1130F2C5D407A010000903D0D505D400FB10FE024
S1130F3C5E0012EC4038696079 20001746225E008F

S1130F4C179A7A01000091ED0D505D407A01000073
S1130F5C903D0D505D400FB10FE05E0012EC400E62
S1130F6C69667926001E4606F8FF38D438D65E002B
S1130F7C2D5054705E002D727A04000017C219664E
S1130F8C0F8569507920000B46105E00179A7A0181
S1130F9C000091FE0D605D40404469507920000CC7
S1070FAC460C7A0171
S1130FB0000092090D605D4040306950792000DBA
S1130FC0460C7A01000092130D605D40401C69508D
S1130FD07920000E46147A010000903D0D605D40BB
S1130FE07A010000921D0D605D4069507920001464
S1130FF0461A5E00179A7A010000922E0D605D403A
S11310007A010000903D0D605D4040646950792095
S11310100015461A5E00179A7A010000923F0D6090
S11310205D407A010000903D0D605D404042695093
S113103079200016461A5E00179A7A010000925032
S11310400D605D407A010000903D0D605D404020E1
S1111050695579250017461 85E00179A7A0134
S113105E000092620D605D407A010000903D0D60CC
S113106E5D405E002D5054706DF66DF50D067909D9
S113107E000128D617500D950CE91A094B041015CB
S113108E40F866050D8846121A0E4B04101940F8E7
S113109E0D9029D6148939D640141A0E4B04101903
S11310AE40F8795900FF0D9029D6168939D60D507F
S11310BE6D756D7654706DF60D0628D31750790144
S11310CE00011A0E4B04101140F866104704191153
S11310DE4004790100010D106D765470F80638EC5A
S11310EEF8FF38C038C1188838D8F8FF38C81888C0
S11310FE38DBF8FF38C9F8DF38D0F8FF38CDF8F011
S10B110E38D1F8FF38D4547006
S1139000435055204D4F444520253032580A000C1B
S113901052656164792133303532004E455854200E
S1139020202020202020202020202020007377257559
S11390300025730A000C0020003C313E000A003C6E
S1139040323E003C313E3173742020202020200A
S1139050202020003C313E326E64003C313E53748C
S11390606F70202020003C313E337264202020208A
S113907020202020202020003C313E53746F7020209C
S113908020202020202020003C323E3173742020F9
S113909020202020202020003C323E326E6420FD
S11390A020202020202020202020003C323E337264E8
S11390B0202020202020202020202020003C323E5374FA
S11390C06F70003C333E3173742020202020202019
S11390D0202020003C333E326E64202020202020BC
S11390E020202020003C333E3372642020202020A7
S10990F0202020202000D7
S11390F63C333E53746F70003C343E31737420200E
S113910620202020202020202020003C343E326E6400A4
S11391163C313E53746172742020003C343E3372FA
S113912664202020202020202020202020003C343E5391
S1139136746F00300031003200330054687265 6189
S113914664205265616479204 74F2100474F414CA3
S1139156213C313E574F4E202020202000474F41CF
S11391664C213C323E574F4E202020202000436F97
S1139176756E74557020202020202020202020005456
S11391866F67676C652020202020202020202020200088
S11391963C313E496E6974003C323E496E69742027

S11391A620003C333E496E6974003C343E496E6987
S11391B6742020003C303E496E6974202020202014
S11391C620202020003C313E496E69742020202057
S11391D62020202020003C323E496E697420202046
S10991E6202020202020C0
S11391EC003C333E496E697420202020202020202F
S11391FC20003C313E44657374726F79003C323EFF
S113920C44657374726F003C333E44657374726FC0
S113921C003C343E44657374726F79202020202007
S113922C20003C303E44657374726F7920202020FB
S113923C2020003C313E44657374726F79202020EA
S113924C202020003C323E44657374726F792020D9
S113925C2020202020003C333E44657374726F79C8
S113926C2020202020202020415312D00000000099
S11311167A0000FFEF8A01006F7100045E002D2440
S113112654705E002D727A37000000180F867A001D
S1131136000092967A01000000100AF15E002D2449
S11311460FE10FF05E002CDE0F817A020000929E03
S11311567A00000000080AF05E002B1A40247A0188
S1131166000092A67A02000000300AF20FF05E0039
S11311762B1A7A01000000100AF10F820F905E000D
S113118628707A00000000100AF07A0100000008B7
S11311960AF15E002CCE0D0046C47A170000001833
S11311A65E002D5054705E002D727A3700000020C9
S11311B619660F857A02000000180AF201006DF223
S11311C65C00FF4C0B970FD10FF05E002CDE0F81F6
S11311D67A020000929E7A00000000100AF05E0078
S11311E62B1A7A00000000180AF07A0100000008A2
S11311F60AF15E002D2440747A02000000080AF208
S109120601006DF25C0023
S109120CFF080B977A00B6
S113121200FFEF8A7A01000092AE5E002CBE0D0041
S1131222474E7A010000927C0D605E0017C27A017C
S11312320000927E0D605E0017C27A0100FFEF8A02
S11312427A02000092AE0F905E00284001006B20EC
S1131252000092BA01006DF001006B20000092B60B
S113126201006DF00FD05C00FEBC0B970B97402A78
S11312727A01000000180AF17A02000000100AF253
S11312820FF05E0028700F817A00000000080AF058
S11312925E002CCE0D005860FF627A17000000201A
S11312A25E002D5054705E002D721B971B970F86A4
S11312B20F957A02000000020AE201006DF25C005F
S11312C2FE520B970FD10FF05E002CDE0F817A02D4
S11312D20000929E7A000000000A0AE05E002B1AC8
S11312E20B970B975E002D5054705E002D720F8684
S11312F20F950FE055B001006F6000120B700100F3
S11313026FE000125E002D5054705E002D720D05C9
S11313127A0400FFEF92400601006F44001A0100B5
S11313226F40001A01006F01001A46EC7925000193
S1131332460A7A0600FFEFCE5A0013EA7925000225
S1131342460A7A0600FFEFEC5A0013EA7925000BEE
S1131352460A7A0600FFF00A5A0013EA7925000CBE
S1131362460A7A0600FFF0285A0013EA7925000D8F
S107137246087A06A6
S113137600FFF046406E7925000E46087A0600FF08
S1131386F06440607925001346087A0600FFF08270
S11313964052792500144 6087A0600FFF0A040441F

S11313A67925001546087A0600FFF0BE40367925F2
S11313B6001646087A0600FFF0DC40287925001758
S11313C646087A0600FFF0FA401A7925001E4608F9
S11313D67A0600FFF118400C7925001F46067A06A7
S11313E600FFF1360FE6461C7A010000927C1900D5
S11313F65E0017C27A010000928719005E0017C2C9
S11314061A80405401006FE4001601006F40001A71
S113141601006FE0001A01006F86001601006FC617
S1131426001A7A00000092BE7A01000000020AE167
S10414365E54
S1131437002D247A00000092967A010000000A0A20
S1131447E15E002D2469E51A8001006FE000120FA9
S1131457E05E000D760FE05E002D50547001006DC5
S1131467F60F865E000F8001006F60001601006FA4
S113147761001A01006F81001A01006F60001A01F1
S1131487006F61001601006F81001601006D76542D
S1131497705E002D720F867A0200FFEF8A01006DDE
S11314A7F25C00FC6A0B977A0000FFEF8A7A01006F
S11314B70000020AE15E002D245E002D50547001E6
S11314C7006DF60F867A00000092C67A01000000CD
S11314D7020AE15E002D2401006D76547001006B52
S11314E72000FFEFAC01006F01001A4604190054F6
S11314F77079000001547001006B2100FFEFAC010C
S1131507006F11001A1988400C01006F11001A0DA2
S1131517800B500D080F9146F00D8054706DF50D3B
S10815270501006B202B
S113152C00FFEFAC01006F01001A472001006B2193
S113153C00FFEFAC69101D5046040F9040100100E2
S113154C6F11001A01006F10001A46E81A806D75AE
S113155C54705E002D720D0555BE0F86460E0D5050
S113156C5C00FD9C0F865C00FF22401C7A0000008F
S113157C00020AE07A01000092C65E002CB20D0054
S113158C47060FE05C00FF040FE05E002D50547023
S113159C5E002D720D0555800F86460E0D505C00B6
S11315ACFD5E0F865C00FEE440247A00000000021E
S11315BC0AE07A01000092C65E002CB20D004708C7
S11315CC0FE05C00FEC640060FE05C00FE8A5E0086
S11315DC2D5054705E002D721B971B977A0500FFDC
S11315ECEF9201006F56001A5A00168E01006F65B8
S11315FC001A7A00000000020AE07A01000092BE91
S113160C5E002D420D0047787A00000000020AE0CC
S113161C7A01000092C65E002D420D0047627A01EA
S109162C000000020AE1C8
S10E16327A020000000A0AE20FF05EDB
S113163D0028707A0000FFEF8A7A01000000020A89
S113164DE15E002CCE0D00470E0FF17A02000092E1
S113165DAE0F905E0028407A0000FFEF8A0FF15E17
S113166D002CBE0D0047187A0000FFEF8A7A0100A7
S113167D0000020AE15E002D240FE05E0005460F17
S113168DD601006F60001A5860FF600B970B975ED1
S113169D002D5054701A8001006BA000FFEFA87A43
S11316AD0000FFEFB001006BA000FFEFAC7A00006C
S10A16BDFFEF9201006BA097
S11316C400FFEFC61A8001006BA000FFEFCA54703D
S11316D4F80138601888386138621888386338909C
S11316E4F8FF389118883864F8883865F804386640
S11316F479009E576B80FF6819006B80FF6A19009D

S11317046B80FF6C547001006DF27F677220790067
S11317149E576B80FF687A0100FFEF8A7A0200000C
S113172492CE0F905E00287001006D725470010018
S11317346DF27A0000FFEF8A7A01000092AE5E0038
S11317442CBE0D00472A7A010000927C19005E002A
S113175417C27A010000927E19005E0017C27A0153
S113176400FFEF8A7A02000092AE0F905E002840D9
S10F17745C00FE6801006D7254707A0086
S1131780000092967A0100FFEF8A5E002D245C0030
S10D1790FF425E0002A85A0016A2F1
S113927C0A004F564552574F56490063616C6C6F49
S113928C63206661696C656400000000000000000E7
S113929C0000408F4000000000003FF00000000081
S11392AC0000412E84800000000040CF40000000ED
S11392BC0000412E848200000000412E84840000B3
S10D92CC00003F50624DD2F1A9FCEF
S113179A7A00000092D601006DF07A0000FFF1543E
S11317AA5E0026D20B977A0000FFF15401006DF018
S11317BA5E001AEA0B9754705E002D727A0600FFD8
S11317CAF1940D040F950C44586000BAAC00470E0F
S11317DAAC01587000B0AC02476E5A0018907A01F7
S11317EA000092D80FD05E0027300D00461E7A0003
S11317FA000092DE01006DF07A00000092DB010026
S113180A6DF00FD05E0026D20B970B9701006DF592
S113181A7A00000092DB01006DF00FE05E0026D231
S113182A0B970B9701006DF67A00000092DB01001B
S106183A6DF05EED
S113183D0018E60B970B9701006DF55E001AEA0B86
S113184D974040403E01006DF57A00000092DB01A8
S113185D006DF00FE05E0026D20B970B9701006D24
S113186DF67A00000092DB01006DF05E0018E60BC6
S113187D970B970FE05E00275A0B500D010FE05E9B
S10C188D00254E5E002D5054703D
S10E92D60C000A00002573000A0C00C6
S1131896188838BA38B8F85038B919000B50792077
S11318A601184DF8F83038BA28BCF88038BC5470A3
S11318B60C8820BC737047FA38BBE07F30BC547089
S11318C620BC736047FA29BDE0BF30BC0C98547046
S11318D67EBC736047067900000154701900547 08A
S11318E65E002D727A0600FFF1D47A05000000042B
S11318F67A00000000180AF00AD07308470E7A002F
S1131906000000180AF00AD00B70400A7A000000A3
S113191600180AF00AD00F85189968E901006DF0DE
S113192601006F71001C0FE05E0027760B971955B7
S11319360D5017F00AE0680947220D5017F00AE028
S11319466808A80A4606F80D5C00FF640D5017F0F8
S11319560AE068085C00FF580B5540D45E002D5022
S10519665470B8
S11319681911400C19880B58792806824DF80B5128
S11319781D014DF0547001006DF50D090D8D28D62C
S113198817500D010D99470C0D19796900607949B4
S1131998001040060D19796900600DD50D90148D5E
S11319A80CD8C88038D63DD639D67900000455B04E
S11319B801006D7554706DF56DF40D090D8528D60C
S11319C817500D010D99470C0D1979690060794974
S11319D8001040060D19796900600D541194119493
S11319E811941194EC0F0D90148CED0F148D0CC8F9

S11319F8C88038D63CD60CD8C88038D63DD639D618
S1131A08790000045C00FF586D746D755470010013
S1131A186DF619EE7900000F5C00FF44196679082A
S1131A2800030DE05C00FF4E0B56792600034DEED4
S1131A38790800020DE05C00FF3C1966790800286C
S1131A480D605C00FF70790800100D605C00FF6694
S1091A587908000E0D6089
S1131A5E5C00FF5C790800060D605C00FF5279089C
S1131A6E00010D605C00FF48790800020D605C0008
S1131A7EFF3E01006D7654707908000119005C0079
S1131A8EFF2E7908000219005A0019BE6DF60C8E4E
S1131A9EAE0C460455E2401CAE0A460455DA401419
S1131AAEAE0D460455D2400C17D60D6879000001D1
S1131ABE5C00FEFC6D76547001006DF60D0E0D8606
S1131ACE0D6079080040528009E0791000800D08FE
S1131ADE19005C00FEDA01006D7654705E002D7203
S1131AEE790C000119447A0600FFF2247A050000EE
S1131AFE00047A00000000180AF00AD07308470E9B
S1131B0E7A00000000180AF00AD00B70400A7A001F
S1131B1E000000180AF00AD00F85189968E9010031
S1131B2E6DF001006F71001C0FE05E0027760B97BE
S1131B3E19550D5017F00AE0680947560D5017F066
S1091B4E0AE06808A80A82
S1131B54460A0DC80D405C00FF68403C0D5017F069
S1131B640AE06808A80D460C790800020D405C00E1
S1131B74FE4840240D5017F00AE06808A80C4606F6
S1131B845C00FEFE40120D5017F00AE0680817D0FF
S1131B940D080DC05C00FE220B5540A05E002D50C5
S1051BA4547078
S1131BA61911400C19880B58792806824DF80B51E8
S1131BB61D014DF054705C0009145C0000B455041B
S1131BC65A001CA001006DF618886AA800FFF327C7
S1131BD66AA800FFF32918886AA800FFF3286AA8F1
S1131BE600FFF2657908008079000045C000808AC
S1131BF6790800107900002O5C0007FC7906000FC5
S1131C06790800100D605C0007EE0D687900000B83
S1131C165C0007E40D687900000D5C0007DA7908BB
S1131C260050790000095C0007CE790800D67900D8
S1131C3600075C0007C219660D605C000710790E89
S1131C4600010DE05C0006E60DE05C0007000D6890
S1131C567900002B5C0007A00D687900002F5C005B
S1131C6607960DE8790000275C00078C01006D7666
S1131C7654707908000519005C00077C790000643C
S1131C865C00FF1C790800C419005C00076A790828
S1091C96000379000001C8
S1131C9C5A0023FE79080002790000055C00075204
S1131CAC790800CC19005A0023FE5E002D727A04C9
S1131CBC000093947A000000936301006DF05E00C2
S1131CCC18E60B9719EE19660DE5790D001052D530
S1131CDC0D6009505C00070C17506DF001006DF49A
S1131CEC5E0018E60B970B870B56792600104DE018
S1131CFC7A000000939A01006DF05E0018E60B97D2
S1131D0C0B5E792E00044DBE5E002D505470010005
S1071D1C6DF67A06DD
S1131D2000FFF264790000065C0006C468E87C608A
S1131D3073404736790000OE5C0006B40C8E73487E
S1131D4047045C0002C8735E47085C0002C25A0085

S1131D501E0E730E47085C0002B85A001E0E731E57
S1131D6047045C0002AE5A001E0E7C60736047227B
S1131D707900000C5C0006780C8E730847085C0041
S1131D8000925A001E0E731E587000825C0001C63A
S1131D90407C7C607320471E7900000A5C0006507B
S1131DA00C8E730847065C00020A4062731E475E8E
S1131DB05C00023E40587C60731047527900000873
S1131DC05C00062C0C8E7368470A5C00FDFC5C000B
S1131DD0FECE403A734E471A790800C079000009D5
S1131DE05C00061A79080003790000055C00060E02
S1131DF0401C737E4718790800050790000095C0085
S1131E0005FC79080002790000055C0005F001007B
S1081E106D7654705EC5
S1131E15002D7219DD790000265C0005CE0C8E734A
S1131E2568587001047A0600FFF267790000086DAF
S1131E35F00FE179080026790000255C0005CC0B3D
S1131E45870DD05C0004E80DD05C00050279050020
S1131E55200D505C0004C06868E8605860009C6E03
S1131E65680001473CA801471EA8034772A8054718
S1131E753EA8064726A8084716A809475CA80A47A7
S1131E8526A80B4760406C5C0001865A001F1C6A3C
S1131E952800FFF26617500D08400E5C000224402F
S1131EA5765C00035440700DD879000021402479F5
S1131EB50800400D505C0005406E6E0002CE806A3E
S1131EC5AE00FFF2656A2800FFF26517500D087929
S1131ED50000045C000522403E5C00038E40385C34
S1131EE500018840326E680002472C0D505C0004E7
S1131EF50E40240D505C000406401C6868E860A889
S1081F052046080D5009
S1131F0A5C0003F6400C686EEE60AE400D505C0058
S1131F1A03E87A0000FFF3287D0070000DD05C000F
S1131F2A045440226A2800FFF329470818886AA83C
S1131F3A00FFF3290DD05C00041079080001790031
S10A1F4A00275C0004AE5EFA
S1131F51002D5054705E002D727900002E5C000438
S1131F618E0C8EE8C017504626790000406DF07A3A
S1131F710500FFF2A70FD17908002E7900002D5C2F
S1131F8100048C0B870D060D010FD05C00056279EF
S1131F910000015C00039C790800017900002F5CBB
S1131FA100045A5E002D505470790000365C000421
S1131FB13E54706DF6790000225C0004320C8E737E
S1131FC168472A7358472619005C0003866A28006C
S1131FD1FFF329470C5C0001C819005C0003A04012
S1131FE10C79080001790000275C0004106D765418
S1131FF1706DF67900002A5C0003F40C8E73684758
S11320010873584 7045C0004826D7654705470540D
S113201170547054705E002D727A0600FFF2676887
S113202168E803A80246426E680003E80747186E92
S1132031690003E9071751177178106A28000092A4
S1082041E217505C00F2
S113204602D46E680003E8071750790100011A08E5
S11320564B04101140F817117A0000FFF327680AA2
S1132066169A688A5E002D5054705E002D727A06A9
S113207600FFF2676868E803A80246406E6800033B
S1132086E80747186E690003E907175117717810B7
S11320966A28000092E217505C0002626E68000331
S11320A6E8071750790100011A084B04101140F88C

S11320B67A0000FFF327680A149A688A5E002D5097
S11120C654707A0000FFF3287D0072006A2830
S11320D400FFF26AA801470AA8024716A803472289
S11320E454706A29000092EA17517A00000092EAB8
S11320F440686A29000092FE17517A00000092FC9E
S113210440586A2800FFF269470EA801471AA8023B
S11321144726A803473254706A290000932317D132
S11321247A000000932340326A290000932717D1D1
S11321347A000000932740226A290000933917D1BB
S10D21447A000000933940126A2963
S113214E0000934117D17A00000093415502547059
S113215E5E002D720F840D187A0600FFF32A6A288B
S113216E00FFF26E17500C8018886A2900FFF26D7B
S113217E1751091069E01D804F0269E801006BA435
S113218E00FFF32CF8016AA800FFF32955065E0041
S113219E2D5054705E002D727A0600FFF32A7905D6
S11321AE000869604744696079200008 4E0269653A
S11321BE7A0400FFF32C6DF50100694179080022C2
S11321CE790000215C0002780B870D056961190106
S11321DE69E117F0010069410A81010069C1696073
S11321EE460818886AA800FFF3295E002D50547024
S10421FE5E7F
S11321FF002D726A2800FFF267E80317500D0519C7
S113220FEE790600210D0047067925000146100FD0
S113221FE05C0001DA0DE80D605C0001D2403C790F
S113222F250002462E6A2800FFF26AE80717500DB1
S113223F05790100011A084B04101140F86A2800B0
S113224FFFF3271750660147067908000140060D73
S113225FE840020DE80D605C0001945E002D5054C0
S113226F706DF66A2800FFF2696AA800FFF26646EE
S113227F3C19660D68790000285C0001720D6879BE
S113228F00002C5C0001680D68790000305C0001D0
S113229F5E0D68790000345C0001540D687900000D
S11322AF385C00014A0D687900003C404018886A89
S11322BFA800FFF327790600010D605C0000867903
S11322CF080011790000285C000124790800037 9C4
S11322DF00002B5C0001180D60554A790800127934
S10822EF00002C5C005F
S11322F401080D687900002F5C0000FE6D765470B0
S11323046DF60D065C0000E4C88017500D080D60DF
S11323145C0000E66D7654706DF60D065C0000CC2F
S1132324E87F17500D080D605C0000CE6D76547085
S113233401006DF60D0617F610367908000878606B
S11323446B2000FFEF185C0000B001006D76547041
S11323545E002D720D0617F610367A0500FFEF1096
S11323640AE569505C00008417500D06703E0D6841
S113237469505C0000845E002D5054705E002D7221
S11323841B971B977A0500FFF3280D0EFE017900B6
S113239400010DE11A094B04101040F86859175154
S11323A466104704702E4002722E701E17560DE0FD
S11323B417F0103001006FF000040D6801006F7115
S11323C4000478106B2000FFEF10010069F1552A17
S11323D4790000011B5E4B04101040F868591589FD
S10923E468D90B970B976B
S10423EA5E91
S11323EB002D5054706AA8004000036A2800400077
S11323FB0154700D016AA9004000030D806AA80007

S113240B40000154705E002D727A03004000030FED
S113241B840F9568BC19EE196640180DC055C6E8B4
S113242B0F471868BC6A280040000168D80B750B6E
S113243B5E0B566F7000181D064DE00DE05E002D10
S113244B5054705E002D720D0C0D830F956F74003D
S113245B1819EE1966401E0D30558AE81F471A0DE1
S113246BC06AA80040000368586AA8004000010B2B
S113247B750B5E0B561D464DDE0DE05E002D505465
S113248B7001006DF619EE7A0000FFF2E779010097
S113249B405C0001140D06790000015C00FEAA0DDF
S11324AB6647166DF67A0100FFF2E77908002A7981
S11324BB000029558E0B870D0E790000015C00FE81
S11324CBB40DE001006D76547019006BA000FFF39F
S10624DB326BA0BE
S11324DE00FFF33019006BA000FFF3366BA000FF73
S11324EEF33454705E002D727A0400FFF3327A05D2
S11324FE00FFF3300F830D1E7FF572501999403094
S113250E6940792001004C2C695017F06839780026
S113251E6AA900FFF33869500B5017F079010100D7
S113252E01D0531069D869400B5069C00B730B5916
S10E253E1DE94DCC7FF570500D905E41
S1132549002D5054705E002D72790E01007A04003B
S1132559FFF3367A0500FFF3340F830D127FF5720B
S113256950199940346940792001004C3269501758
S1132579F001D053E00D8017F0683978006AA9009B
S1132589FFF43869500B5017F001D053E069D8694B
S1132599400B5069C00B730B590D201D094DC67FA4
S11325A9F570500D905E002D5054705E002D727AB7
S11325B90500FFF3360F840D1E7FF572501966402F
S11325C9381DE64C386B2000FFF334695119107933
S11325D910010017F07901010001D053100D801784
S11325E9F00D6117F10AC178006A2800FFF4386811
S11325F99869501B5069D00B5669504EC47FF570CA
S1132609500D605E002D5054705E002D727A0500E6
S1132619FFF3320F840D1E7FF57250196640381D82
S1132629E64C386B2000FFF330695119107910011A
S10826390017F0790118
S113263E010001D053100D8017F00D6117F10AC17F
S113264E78006A2800FFF338689869501B5069D0E8
S110265E0B5669504EC47FF570500D605E41
S113266B002D5054706B2000FFF33217F0790101EA
S113267B0001D053100D8054706B2000FFF33617FD
S110268BF07901010001D053100D8054704F
S11392E220282C3034383C201201000100000008F1
S11392F2FEFF10000100010102010902270001012 2
S113930200C0640904000003000000030705810292
S11393124000FF070502024000FF070583024000E9
S1139322FF040309041203550053004200200054B2
S113933200450053005400080331002E0030002280
S113934203550053004200200054004500530054CB
S11393520020005000520 04F004700520041004DD0
S10493620007
S11380000023002B0033003B0027002F0037003FE5
S113936330302030312030322030332030342030 3D
S113937335203036203037203038203039203041 03
S11393832030422030432030442030452030460AE9
S10C939300253032582000 0A00C5

S113269801006DF67A0600FFF574010069607A019E
S11326A841C64E6D5E004C2A7A1000003039010095
S11326B869E06960198817707A01000080005E007C
S10D26C84C040D1001006D765470F0
S11326D25E002D720F857A06000000047A00000066
S11326E200180AF00AE07308470E7A000000001887
S11326F20AF00AE00B70400A7A00000000180AF0A0
S11327020AE00F8601006DF001006F70001C0100EA
S11327126DF001006DF51A91790000015E002D9AAA
S11127227A170000000C0D065E002D50547057
S11327305E002D720F860F9540040B760B7568694A
S113274068581C8946040C9946F068681750685D00
S10D2750175519505E002D50547008
S113275A5E002D720F851AE640020B760FD00B75B9
S10F276A680946F60FE05E002D50547025
S11327765E002D720F850F9601006F710018010020
S11327866DF101006DF601006DF51A9179000001F6
S11327965E002D9A7A170000000C5E002D505470CF
S11327A60FC446120FD5460E792107FF46120FB303
S11327B64604156E4A0A1AC47A050000000818660C
S11327C65A002A3A0D9946120FC4461A0FD54616D1
S11327D60FB3461E16E6580002660FA246360FB31F
S11327E646325800025A0FA246140FB34610580039
S11327F6024E0CE60D190FA40FB55A002A461035E2
S11328061234792C00104C0C1B5910351234792CC8
S113281600104DF410331232792A00104C0C1B5160
S1132826103312327 92A00104DF4580000CA1AC424
S11328361AD5186619995800020601006DF20100AF
S11328466DF301006F23000401006DF3010069239A
S1132856795B800001006DF30FF2550E0B970B9712
S113286601006D7301006D7254706DF601006DF514
S113287601006DF401006DF301006DF201006DF1CD
S113288601006DF00100691401006925103410354B
S10928961FD44218451097
S113289C01006F14000401006F2500041FD44406CB
S11328AC0F940FA10FC26816682E0FA001006914B4
S11328BC01006F150004010069020100 6F0300049D
S11328CC796C000F796A000F691911191119111913
S11328DC1119796907FF6901111111111111111E5
S11328EC796107FF792907FF5870FEAE0D115870F7
S11328FCFECC10351234103512341035123410331B
S113290C12321033123210331232794C0080794A5E
S113291C00800D901910474C792000364E3E7920DB
S113292C00204D0E793000200FB34702700A0FA31D
S113293C1AA2792000104D14793000100D380DB304
S113294C0D2B0DA219AA0D884702700B0C884F147E
S113295C113213334402700B1B5040F01AA27A034A
S113296C000000010C6815E84B2A0AB544020B74ED
S113297C0AA4792C010058500080113413354402F9
S107298C700D0B5963
S1132990792907FF5860006E1AC41AD5580000A69B
S11329A01FA446061FB55870FE8A1AB544087A3428
S11329B00000000145061AA4450440121AA4173565
S11329C017347A150000000144020B740CE60FC49F
S11329D046080FD41AD57919FFE00DCC460C0D4CDF
S11329E00DD40D5D19557919FFF0792C01004508B7
S11329F0113413350B5940F2792C00804C081035F3

S1132A0012341B5940F20D994E10113413350D99A0
S1132A104A08113413350B5940F4732D471C0CD855
S1132A20E80B47167A150000000844020B74792C52
S1132A3001004D06113413350B591134133511347C
S1132A40133511341335796C000F1019101910193F
S1132A501019649C101C1006131C01006D700100FA
S1132A60698401006F85000401006D7101006D72BE
S1132A7001006D7301006D7401006D756D76547006
S1132A8001F06401586000807924O7FF5860006CEE
S1132A905800007401F064235860006C40520F85A5
S1132AA001F06415460E0D44464601F0642346408A
S1132AB05800005410311230792800104C0C1B5C64
S1132AC010311230792800104DF4580000BA0FA5C8
S1132AD001F06435472610331232792A00104C0C6A
S1132AE01B5410331232792A00104DF45800009E03
S1132AF01AC4100E131C1AD55800018E790E07FF45
S1132B001AC41AD5580001621AC418EE790E07FFC9
S1132B1019DD790500085800015001006DF6010028
S1132B206DF501006DF401006DF301006DF201001C
S1132B306DF101006DF0681E6826156E691C111C8D
S1132B40111C111C111C796C07FF69241114111439
S1132B5011141114796407FF01006D10010069114C
S1132B607968000F01006F2300040100692279 6A6C
S1092B70000F792C07FFA2
S1132B765870FF06792407FF5870FF120DCC587062
S1132B86FF160D445870FF40194C791C03FF0DCEF8
S1132B96792EFFCC58D0FF5279480010794A00109D
S1132BA61AC47A0500000100103312321031123 0B4
S1132BB6441C0AB144087A100000000145040AA027
S1132BC640040AA004011235123444E0401C1AB131
S1132BD644087A300000000145041AA040041AA0F4
S1132BE604011235DD01123444C2730D46080AB1DD
S1132BF644020B700AA001F064014702700D792CA0
S1132C0600804C06103512341B5E792E07FF58C020
S1132C16FEE40DEE4E2E113413354402700D0DEE07
S1132C264A040B5E40F0732D47180CD8E80B471285
S1132C367A1500000008440A0B74792C00804D02B3
S1132C460B5E4020732D471C0CD8E80B47167A15EC
S1132C560000000844020B74792C01004D06113460
S1092C6613350B5E11346F
S1132C6C133511341335113413357 96C000F101ED1
S1132C7C101E101E101E64EC101C100E131C0100F1
S1132C8C6D700100698401006F85000401006D7192
S1132C9C01006D7201006D7301006D7401006D759F
S1092CAC01006D76547077
S10F2CB25E004B48A80147021900547053
S1132CBE5E004B480C884E0419005470F801547092
S1132CCE5E004B480C884604F8015470190054708A
S1132CDE01006DF201006DF11AA20F9147224A060F
S1132CEE7902080017B17912041F1B52103144FAEE
S1132CFE10311212103112121031121210311212 2F
S1132D0E698201006F8100026F8A000601006D71F6
S1092D1E01006D72547008
S1132D2401006DF201006902010069920 1006F0262
S1112D34000401006F92000401006D725470E0
S1112D425E004B481A084704790000015470E4
S1132D5001006F76001401006D7201006FF2001024

S1132D6001006D7201006D7301006D7401006D75DA
S1052D7054709A
S1132D7201006DF501006DF401006DF301006DF2C8
S1132D8201006F72001001006DF201006FF6001472
S10B2D9201006F72000454708C
S1132D9A5E002D727A37000000B27A0300FFF54C09
S1132DAA7A0400FFF53C0D0201006FF100AA01004D
S1132DBA6F7600CE01006F7500D20D00466C0100DC
S1132DCA6F7000CA460E790004526BA000FFF578B3
S1132DDA5A002E6C01006F7000CA6E080010E807D3
S1132DEA17D0460C790005146BA000FFF5784072E2
S1132DFA01006F7000CA6E0800107328461073082A
S1132E0A470C790105166BA100FFF57840540100C0
S1132E1A6F7000CA6E0800107368464601006F702F
S1132E2A00AA01006BA000FFF5700D2017F0010046
S1132E3A6BA000FFF53801006F7000CA01006BA098
S1132E4A00FFF53E1A8001006BA000FFF5421888C7
S1132E5A68C8F8306EF800885C001CBA0D00587018
S1132E6A0A1C7900FFFF5A0038AA6868A825586027
S1132E7A09CC0B76188868C80FF001006BA000FF15
S1072E8AF5500FF0FD
S1132E8E01006BA000FFF5541A8001006BA000FF38
S1092E9EF55801006BA0D2
S1132EA400FFF55C1A8001006BA000FFF5600100D0
S1132EB46BA000FFF5641A8001006BA000FFF568A6
S1132EC440306868A8204718A8234720A82B470A3E
S1132ED4A82D461C7D40701040167D407030401074
S1132EE47C407330460A7D40704040047D4070202E
S1132EF40B766868A82D47CAA82B47C6A82047C2E3
S1132F04A82347BEF9206AA900FFF5466869A930DA
S1132F14460AF9306AA900FFF5460B761A800100C8
S1132F246BA000FFF5486868A82A586000800FD09A
S1132F340BF001006FF000A07308470A01006F70E3
S1132F4400A00B70400601006F7000A00F851BF0FA
S1132F5401006FF000967308470A01006F70009632
S1132F641B70400601006F700096690017F00100A2
S1132F746BA000FFF5484C167D40701001006B20D8
S1132F8400FFF54817B001006BA000FFF5480100EE
S1092F946B2000FFF5486D
S1132F9A7A20000002004F0E7D4070007900051868
S1132FAA6BA000FFF5780B76686817D0796000FF8D
S1132FBA17F078006A28000093A4732858700086D3
S1132FCA1A8001006BA000FFF5484064686817D0B7
S1132FDA7930003017F001006FF000A07A01000089
S1102FEA02001A810F907A010000000A5EB8
S1132FF7004C0401006B2100FFF5481F904D24018D
S1133007006B2000FFF5487A010000000A5E004CC0
S11330172A01006F7100A00A9001006BA000FFF561
S113302748400E7D407000790005186BA000FFF53E
S1133037780B76686817D0796000FF17F078006A15
S113304728000093A4732846867A00FFFFFFFF0139
S11330570069B06868A82E586001000B766868A8F5
S11330672A466C0FD00BF001006FF00096730847E8
S11330770A01006F7000960B70400601006F700025
S1133087960F851BF001006FF000A07308470A0134
S1133097006F7000A01B70400601006F7000A069ED
S11330A70017F0010069B04C0A7A00FFFFFFFF0128

S11330B70069B0010069307A20000002004F0E7DDD
S11330C7407000790005186BA000FFF5780B766850
S11330D76817D0796000FF17F078006A280000931B
S10830E7A473284776E5
S11330EC1A80010069B04058686817D079300030F5
S11330FC17F001006FF000A07A01000002001A81A2
S113310C0F907A010000000A5E004C040100693143
S113311C1F904D1C010069307A010000000A5E000B
S113312C4C2A01006F7100A00A90010069B0400E97
S113313C7D407000790005186BA000FFF5780B76C5
S113314C686817D0796000FF17F078006A280000D0
S113315C93A4732846927A000000002001006BA010
S113316C00FFF56C6868A8684708A86C4704A84C6E
S113317C4610686817D017F001006BA000FFF56CC0
S113318C0B766868A825587004A4A84558700212D9
S113319CA8475870020CA8584742A863587002E419
S11331ACA8644738A865587001F8A866587001F2EE
S11331BCA867587001ECA8694722A86E58700430B0
S11331CCA86F4718A87058700398A873587002F822
S10931DCA8754708A8785E
S10D31E247045A00367601006B2003
S11331EC00FFF56C7A200000006C46440FD00B9066
S11331FC01006FF000AA7308470A01006F7000AA60
S113320C0B70400601006F7000AA0F851B90010024
S113321C6FF000967308470A01006F7000961B70DD
S113322C400601006F700096010069025A00337E5C
S113323C01006B2000FFF56C7A20000000685860D9
S113324C009A6868A875470CA8584708A8784704DB
S113325CA86F46420FD00BF001006FF000AA730861
S113326C470A01006F7000AA0B70400601006F70D3
S113327C00AA0F851BF001006FF000967308470A34
S113328C01006F7000961B70400601006F70009672
S113329C6900177040400FD00BF001006FF00096DF
S11332AC7308470A01006F7000960B70400601000B
S11332BC6F7000960F851BF001006FF000AA730866
S11332CC470A01006F7000AA1B70400601006F7063
S10932DC00AA690017F0CF
S11332E20F825A00337E6868A875470CA8584708AE
S11332F2A8784704A86F46420FD00BF001006FF085
S11333020096730847 0A01006F7000960B7040061F
S113331201006F7000960F851BF001006FF000AA89
S11333227308470A01006F7000AA1B704006010070
S11333326F7000AA6900177040400FD00BF00100B4
S11333426FF000AA7308470A01006F7000AA0B709E
S113335240060 1006F7000AA0F851BF001006FF099
S11333620096730847 0A01006F7000961B704006AF
S113337201006F700096690017F00F820100693037
S11333827A20FFFFFFFF460A7A00000000010100D6
S113339269B0686817D06DF07A01000000020AF183
S11333A20FA05C00050E0B875A0035EA01006B2063
S11333B200FFF56C7A200000004C46420FD00B90C0
S11333C20B9001006FF000AA7308470A01006F70A7
S10933D200AA0B70400687
S11333D801006F7000AA0F851B901B9001006FF00E
S11333E800967308470A01006F7000961B70404AE5
S11333F801006F70009640420FD00B900B900100B4
S11334086FF000AA7308470A01006F7000AA0B70D7

S1133418400601006F7000AA0F851B901B900100E6
S11334286FF000967308470A01006F7100961B71CD
S1133438400601006F7100960F907A01000000802A
S11334480AF15E002D24010069307A20FFFFFFFF97
S1133458460A7A0000000006010069B0686917D1BE
S113346801006F70008401006DF001006F7000842B
S113347801006DF07A00000000080AF05C0007867E
S11334880B970B975A0035EA0FD00BF001006FF03A
S113349800AA7308470A01006F7000AA0B70400660
S11334A801006F7000AA0F851BF001006FF00096F2
S11334B87308470A01006F7000961B7040060100ED
S10934C86F7000966E0810
S11334CE00015A00363C0FD00B9001006FF000AA9A
S11334DE7308470A01006F7000AA0B7040060100C3
S11334EE6F7000AA0F851B9001006FF00096730892
S11334FE470A01006F7000961B70400601006F7043
S113350E0096010069000F825E00275A01006FF0DA
S113351E00AA010069317A21FFFFFFFF4712010064
S113352E69311F814C0A0100693001006FF000AA56
S113353E0FF001006BA000FFF5541A8001006BA081
S113354E00FFF56001006B2000FFF54801006F716D
S110355E00AA1A9001006BA000FFF5685A47
S113356B0036840FD00B9001006FF0009673084761
S113357B0A01006F7000960B70400601006F70001C
S113358B960F851B9001006FF000AA7308470A0181
S113359B006F7000AA1B70400601006F7000AA0138
S11335AB0069007A6000FFFFFF01006FF0009601D6
S11335BB0069307A20FFFFFFFF460A7A0000000004
S11335CB01010069B0790000786DF07A0100000009
S11335DB020AF101006F7000985C0002CE0B870F9B
S11335EBF00F825E00275A01006FF000AA5A0036D3
S11335FB840FD00B9001006FF000A07308470A01F2
S113360B006F7000A00B70400601006F7000A00FDD
S113361B851B900F81730847060F901B7040020F99
S113362B90010069006B2100FFF5446981404AF862
S113363B2568F80FF00F827A000000000101006F7C
S113364BF000AA0FF101006BA100FFF5541A9101D1
S106365B006BA15D
S113365E00FFF56001006B2100FFF5481A810100A0
S113366E6BA100FFF568400E790005186BA000FFF3
S113367EF5787D4070006868A86E587001B86A28A6
S113368E00FFF53C7308586001AC7C40731046365E
S113369E01006B2000FFF5684F2C40207A010000DB
S11336AE00017A0000FFF5465C0013B87A0000FFB4
S11336BEF568010069011B710100698101006B202E
S11336CE00FFF5684ED601006B2000FFF55C462423
S11336DE01006B2000FFF560461A01006B2000FF0E
S11336EEF564461001006F7100AA0FA05C001374FD
S11336FE5A00380801006B2000FFF5500FA11A90F5
S113370E01006FF0009C01006FF000A04F3001002C
S113371E6F71009C0FA05C00134A40227A000000D8
S113372E00880AF07A01000000015C0013367A006B
S113373E00FFF55C010069011B7101006981010045
S113374E6B2000FFF55C4ED401006B2000FFF55497
S113375E01006B2100FFF5501A9001006FF0009CE1
S113376E01006F7100A00A8101006FF100A00F80AC
S113377E4F3601006F71009C01006B2000FFF55066

S113378E5C0012E040227A00000000880AF07A0101
S113379E000000015C0012CC7A0000FFF56001000E
S11337AE69011B710100698101006B2000FFF56047
S10937BE4ED401006F71FF
S11137C400AA01006F7000A01A8101006B20A3
S11337D200FFF5545C00129840227A000000008832
S11337E20AF07A01000000015C0012847A0000FFF3
S11337F2F564010069011B710100698101006B20FD
S113380200FFF5644ED47C407310473601006B20F1
S113381200FFF5684F2C40207A01000000017A0076
S113382200FFF5465C0012487A0000FFF5680100CC
S113383269011B710100698101006B2000FFF568BA
S11338424ED60B76404001006FF600A6400E0100F3
S11338526F7000A60B7001006FF000A601006F707D
S113386200A668086EF8009547086E780095A825AB
S113387246DC01006F7100A61AE10FE05C0011F053
S113388201006F7600A6686847145C0012900D0071
S1133892460C6A2800FFF53C73085870F5D45C00A7
S11338A2125E6B2000FFF5447A17000000B25E003F
S11338B22D5054705E002D727A37000000247A0571
S10938C2000000040AF5FA
S11338C801006FF0001801006FF1002001006FF193
S11338D8001C18AA689A6F72003C0C22463CAA582E
S11338E8472CAA644712AA69470EAA6F4712AA75FA
S11338F84706AA78471840227A000000000A4006C3
S11339087A000000000801006FF00014400C7A00F0
S1133918000000001001006FF000146F70003C792064
S113392800644706792000694612010006F70001889
S11339384C0A01006F74001817B4400601006F7435
S113394800181AE61AB30FC4467401006B2000FF6F
S1133958F54C476AF83068D87A06000000014062DF
S11339680FD00AE00F82010069F00FC001006F71E8
S113397800145E00522E0100697068890FA0680860
S1133988A8094E0C0FD00AE0680989306889401EDF
S11339986F70003C79200078460A0FD00AE0680966
S11339A8895740080FD00AE06809893768890FC02A
S10939B801006F71001411
S11139BE5E00522E0F840B760FC4469E6A28BD
S11339CC00FFF53C732847626F70003C0C00465AAD
S11339DCA858471EA86F4706A8784716404C010005
S11339EC6F70001847440FE00B760AD0F9306889E2
S11339FC403801006F700018473001006F700020D1
S1133A0C0B7001006FF000201B70F9306889010006
S1133A1C6F7000200B7001006FF000201B706E792B
S1133A2C003D68897A03000000020FB00AE00F839F
S1133A3C01006FF600146F70003C79200064470698
S1133A4C79200069467201006B2000FFF54C460893
S1133A5C01006F700018476001006F7000184D4231
S1133A6C6A2800FFF53C733847160B7301006F701F
S1133A7C00200B7001006FF000201B70F92B4036F7
S1133A8C6A2800FFF53C7348472E0B7301006F70D7
S1133A9C00200B7001006FF000201B70F920688967
S1133AAC40160B7301006F7000200B7001006FF058
S1093ABC00201B70F92D30
S1133AC268897A0600FFF55401006F70001C680CC8
S1133AD2AC2B4708AC2D4704AC20460E01006F7097
S1133AE2001C0B70010069E040466A2800FFF53CA8

S1133AF2732847326F70003C0C004634A858470ABB
S1133B02A86F4714A8784702402601006F70001C73
S1133B120BF0010069E0401801006F70001C0B708C
S1133B22010069E0400A01006F70001C010069E0B6
S1133B327A0400FFF54C010069401FB04F140100E5
S1133B426946010069441AB401006BA400FFF560E1
S1133B52400C0FB61A8001006BA000FFF5607A04D7
S1133B6200FFF56801006F70001C680BAB2B470860
S1133B72AB2D4704AB20463801006B2000FFF5480C
S1133B821FE04F2C6A2800FFF546A83046220100A9
S10D3B926B2000FFF5481AE07A01EA
S1133B9C00FFF560010069120A82010069921A8024
S1133BAC010069C0401E01006B2000FFF5481FE0B7
S1133BBC4F0C01006B2000FFF5481AE040021A80FD
S1133BCC010069C001006F7600141B76401A0100D6
S1133BDC6F7000200B7001006FF000201B700FE161
S1133BEC1B760AD1681968890FE64CE201006F70E5
S1133BFC0020189968897A17000000245E002D5064
S1133C0C54705E002D727A37000000647A0300FF53
S1133C1CF53C7A04000000040AF47A0500FFF54C25
S1133C2C0F866FF1005001006FF6005CF83068C826
S1133C3C7A000000007C0AF07A010000939C5E007D
S1133C4C2D420D00587000B87A000000002F0AF0C6
S1133C5C01006DF07A000000002E0AF001006DF0F7
S1133C6C01006F70008801006DF001006F70008817
S1133C7C01006DF07A01000000320AF17A000000B5
S1093C8C00115E004C4A2A
S1133C927A17000000100D024752188868E80100E5
S1133CA2695001006B2100FFF5481F904F06010088
S1133CB26950400801006B2000FFF54801006BA02A
S1133CC200FFF5687A0600FFF5460D207920FFFF15
S1133CD2470C79200001460CF82B5A0049FAF82DBB
S1043CE25A84
S1133CE30049FAF82A68E85A0049FC7A0000000000
S1133CF31101006DF00FC10B717A00000000260A59
S1133D03F05E0050300B97400CF8016EF8002F184B
S1133D13886EC800017A000000000101006FF00003
S1133D2360400E01006F7000600B7001006FF000C4
S1133D336001006F7000600AC06808A83047E4019F
S1133D43006F7000607A20000000014F7A01006F5A
S1133D537000600AC05E00275A0F8201006F700073
S1133D63601B7001006F71002A0A8101006FF1006B
S1133D732A7A000000000101006FF000584030016F
S1133D83006F7000600AC001006F7100580AC168B8
S1133D9308689801006F7000580B7001006FF00002
S1133DA35801006F7000600B7001006FF000600139
S1133DB3006F7000580FA11F904FC401006F700074
S1133DC3580AC0189968890FC00B7001006FF0007F
S1083DD3445E00275AC5
S1133DD801006FF000601A8001006FF000580100C5
S1133DE86F7000607A01000000025E004C0401005D
S1133DF86FF0004001006F70004401006F710060B4
S1133E080A901B7001006FF0003C404E01006F7078
S1133E18004401006F7100580A9001006FF0004CD4
S1133E2868086EF8005701006F71003C01006F725B
S1133E3800581AA101006FF1004801006F72004C8D
S1133E48681968A901006F710048689801006F71CB

S1133E5800580B7101006FF1005801006F70005892
S1133E6801006F7100401F904DA201006F70006048
S1133E78461AF8306EC800017A00000000010100FC
S1133E886FF000601A8001006FF0002A6F70005015
S1133E9879200067470679200047463C7D307050FB
S1133EA801006F70002A01006F7100600A901B7097
S1133EB87A20FFFFFFFC4D0C01006B2100FFF54C3E
S1093EC81F904F0C6F7008
S1133ECE00501BD06FF000504008790000666FF071
S1133EDE00506E78002FA80146246838E8186EF853
S1133EEE0057A8084706A810470A401A0FE00B769A
S1133EFEF92B40100FE00B76F920688940080FE08C
S1133F0E0B76F92D68896F70005079200066586022
S1133F1E077201006F70002A58D001861A80401A6A
S1133F2E0FE00B7601006F7100580AC10B7168190F
S1133F3E688901006F7000580B7001006FF0005814
S1133F4E01006F7100601F904DD61A800F82010021
S1133F5E6BA600FFF55401006F70002A01006BA0E1
S1133F6E00FFF5607C307350460A01006B2000FFA2
S1133F7EF54C4E067C307320470E0FE00B76F92E70
S1133F8E68890FA00B700F827C307350470C7C3006
S1133F9E735047247C307320471E01006BA600FF2D
S1133FAEF5580100695001006BA000FFF564010094
S1093FBE69500FA10A8106
S1133FC40F9201006F70002A0FA10A9001006F7114
S1133FD400600A9001006FF0006001006F71005CE3
S1133FE46819A92B4708A92D4704A92046520100A3
S1133FF46B2000FFF54801006F7100601F904F4074
S11340046A2800FFF546A830463601006F70005C4D
S10940140B7001006BA01C
S113401A00FFF55001006B2000FFF54801006F71A6
S113402A00601A9001006BA000FFF55C1A80010082
S113403A6BA000FFF5685A0049F801006B2000FFE6
S113404AF54801006F7100601F904F4C01006B200F
S113405A00FFF54801006F7100601A9001006BA020
S113406A00FFF5686E78002FA80146186E78002FB6
S113407AA801586009786E780057A8084706A8105F
S113408A5860096A7A0000FFF568010069011B712B
S113409A010069815A0049F81A8001006BA000FFE8
S11340AAF5685A0049F801006F70006001006F71EA
S11340BA002A0A8101006FF1002A010069520AA14C
S11340CA01006FF100521F814C120F914D0E010036
S11340DA6F7100520B710FC05C00092201006F70EF
S11340EA002A58F0028A6848A83046147A00000069
S11340FA000101006FF0004C01006FF00052402CE8
S109410A1A8001006FF0B2
S1134110004C1A8001006FF0005201006F70002AFA
S11341200B7001006FF0002A01006F7000600B70CC
S113413001006FF000601A8001006FF00058403AF0
S11341400FE00B7601006F71004C0AC16819688992
S113415001006F70002A1B7001006FF0002A01003C
S11341606F7000580B7001006FF0005801006F7002
S1134170004C0B7001006FF0004C01006F70002ABF
S11341804EBE0FE00B76F92E688940240FE00B76C4
S113419001006F71004C0B7101006FF1004C1B713A
S11341A00AC168196889010069501B70010069D050
S11341B001006F7000580B7001006FF00058010090

S11341C06F70004C01006F7100521A9001006F7103
S11341D000601F904C0A01006B2000FFF54C4EACB1
S11341E07C307350470C7C307350472E7C307320E7
S11341F047280100695001006F7100580A810100CE
S10742006FF10058FF
S107420401006BA6A1
S113420800FFF5540100695001006BA000FFF56041
S113421840226E68FFFFA830461A40101B76010043
S11342286F7000581B7001006FF000586E68FFFF35
S1134238A83047E87C307320464A6E68FFFFA82EF3
S1134248464201006B2000FFF56047067C3073503F
S113425847321B7601006F70005801006B2100FF85
S1134268F5601A901B7001006FF000581A80010066
S11342786BA000FFF56001006F70005C01006BA08C
S113428800FFF55401006F70005C6808A82B47080D
S1134298A82D4704A820465001006B2000FFF548CD
S11342A801006F7100581F904F3E6A2800FFF546C2
S11342B8A830463401006F70005C0B7001006BA0DE
S11342C800FFF55001006B2000FFF54801006F71F6
S11342D800581A9001006BA000FFF55C1A800100DA
S11342E86BA000FFF568406201006B2000FFF548F2
S10942F801006F71005884
S10B42FE1F904F4601006B20E5
S113430600FFF54801006F7100581A9001006BA079
S113431600FFF5686E78002FA80146146E78002F0B
S1134326A80146286E780057A8084704A810461C1B
S11343367A0000FFF568010069011B7101006981BC
S1134346400A1A8001006BA000FFF56801006B208C
S113435600FFF55401006F71005C1F9058600692D0
S113436601006B2000FFF55001006BA000FFF55420
S11343765A0049F86848A83046147A00000000013C
S113438601006FF0004C01006FF00052402A1A80C2
S113439601006FF0004C01006FF0005201006F70D6
S11343A6002A0B7001006FF0002A01006F70006095
S11343B60B7001006FF0006001006F70002A4F144C
S11343C67A000000000101006FF0004C0FE00B764D
S11343D6684940060FE00B76F93068897A000000D9
S11343E6000101006FF000580FE10B76FA2E689A70
S10943F601006F70005886
S11343FC0B7001006FF0005801006F70002A58C059
S113440C009E01006BA600FFF55401006F70002A9B
S113441C17B0010069511F814F2001006F70002AF2
S113442C17B001006BA000FFF56001006F70002A4C
S113443C01006F7100581A81401801006950010086
S113444C6BA000FFF5600100695001006F7100580B
S113445C0A8101006FF1005801006F70002A0100FE
S113446C69510A81010069D140360FE00B760100D6
S113447C6F71004C0AC168196889010069501B707F
S113448C010069D001006F7000580B7001006FF0D0
S113449C005801006F70004C0B7001006FF0004C62
S11344AC01006F70004C01006F7100521A900100F3
S11344BC6F7100601F904C0A01006B2000FFF54CDC
S11344CC4EA87C307350470C7C30735047347C308F
S11344DC7320472E010069504F4A01006BA600FF61
S10944ECF55801006950C0
S10744F201006BA0B7
S11344F600FFF5640100695001006F7100580A81DD

S113450601006FF1005840226E68FFFFA830461A7B
S113451640101B7601006F7000581B7001006FF08E
S113452600586E68FFFFA83047E87C30732046705A
S11345366E68FFFFA82E466801006B2000FFF5603A
S1134546460A01006B2000FFF56447067C30735072
S1134556474E1B7601006F70005801006B2100FF68
S1134566F5601A9001006B2100FFF5641A901B7029
S113457601006FF000581A8001006BA000FFF5647C
S113458601006BA000FFF56001006F70005C010085
S11345966BA000FFF5541A8001006BA000FFF558CD
S11345A601006F70005C6808A82B4708A82D470414
S11345B6A820465001006B2000FFF54801006F71EB
S11345C600581F904F3E6A2800FFF546A830463430
S11345D601006F70005C0B7001006BA000FFF550CB
S10745E601006B2042
S11345EA00FFF54801006F7100581A9001006BA093
S11345FA00FFF55C1A8001006BA000FFF5684062BA
S113460A01006B2000FFF54801006F7100581F90ED
S113461A4F4601006B2000FFF54801006F710058F7
S113462A1A9001006BA000FFF5686E78002FA801AD
S113463A46146E78002FA80146286E780057A808FA
S113464A4704A810461C7A0000FFF56801006901B7
S113465A1B7101006981400A1A8001006BA000FFE7
S113466AF56801006B2000FFF55401006F71005CCF
S113467A1F90461001006B2000FFF55001006BA04C
S113468A00FFF5545A0049F8010069500B70010004
S113469A6F7100601F904C0C010069510BF10FC040
S11346AA5C00035A6848A830460E7A0000000001ED
S11346BA01006FF0004840241A8001006FF000489F
S11346CA01006F70002A1B7001006FF0002A0100BD
S11346DA6F7000600B7001006FF0006001006F7073
S10946EA006001006F7186
S11346F000481A9001006F72002A0A8201006FF2CB
S1134700002A0FE00B760AC1681968897A00000055
S1134710000101006FF000580FE10B76FA2E689A42
S113472001006F7000580B7001006FF0005801001A
S11347306F700048402E0FE00B7601006F71004C44
S11347400AC168196889010069501B70010069D0AA
S113475001006F7000580B7001006FF000580100EA
S11347606F70004C0B7001006FF0004C01006F7113
S113477000601F904E0A01006B2000FFF54C4EB6FF
S11347807C307350470C7C307350472E7C30732041
S1134790472801006BA600FFF55401006950010092
S10547A06BA009
S11347A200FFF5600100695001006F7100580A8132
S11347B201006FF1005840226E68FFFFA830461ACD
S11347C240101B7601006F7000581B7001006FF0E0
S11347D200586E68FFFFA83047E87C307320464AD2
S11347E26E68FFFFA82E464201006B2000FFF560B2
S11347F247067C30735047321B7601006F700058B6
S113480201006B2100FFF5601A901B7001006FF02D
S1134812005801006F70005C01006BA000FFF554AB
S11348221A8001006BA000FFF5606F7000507920C1
S1134832006546080FE00B76F96540060FE00B763C
S1134842F945688901006F7000580B7001006FF021
S1134852005801006F70002A4D0A0FE00B76F92B06
S1134862688940160FE00B76F92D688901006F7095

S1134872002A17B001006FF0002A01006F70005880
S11348820B7001006FF0005801006F70002A7A204C
S1094892000000644D0E5E
S11348980BF601006F7000580B800B7040140FE08B
S11348A80B76F9306889F83068E801006F700058B2
S11348B80BF001006FF000580FE00B7001006FF070
S11348C8003040360FE01B76010069F001006F707D
S11348D8002A7A010000000A5E004C0489300100B6
S11348E86970688901006F70002A7A010000000A64
S11348F85E004C0401006FF0002A01006F70002A6B
S11349084EC201006F76003001006F70005C6808CA
S1134918A82B4708A82D4704A820465001006B2060
S113492800FFF54801006F7100581F904F3E6A2839
S113493800FFF546A830463401006F70005C0B7029
S113494801006BA000FFF55001006B2000FFF54844
S113495801006F7100581A9001006BA000FFF55C0D
S11349681A8001006BA000FFF568406201006B200C
S113497800FFF54801006F7100581F904F46010072
S10549886B209F
S113498A00FFF54801006F7100581A9001006BA0EF
S113499A00FFF5686E78002FA80146146E78002F81
S11349AAA80146286E780057A8084704A810461C91
S11349BA7A0000FFF568010069011B710100698132
S11349CA400A1A8001006BA000FFF56801006B2002
S11349DA00FFF55401006F71005C1F90461001003F
S11349EA6B2000FFF55001006BA000FFF5541888F7
S11349FA68E87A17000000645E002D5054705E0068
S1134A0A2D720F850F960FD00AE06808A8344F520B
S1134A1A0FD00AE0F93068891B760FD00AE06809DB
S1134A2A8901688940200FD00AE06808A8394F1421
S1134A3A0FD10AE1681888F668986E18FFFF880193
S1134A4A6E98FFFF1B760FE64EDC6E580001A839FD
S1134A5A4F106E58000188F66ED800016858880115
S1134A6A68D85E002D5054705E002D721B977A032E
S1094A7A00FFF53E7A0681
S1134A8000FFF542010069F00F9501006B2100FF63
S1134A90F5387A21000000001462601006DF50F81EB
S1134AA0010069305E0051D40B97010069300AD0D0
S1134AB0010069B0010069600AD0010069E0403A71
S1134AC019444030010069700B70010069F01B70DC
S10F4AD0680817D00100693101006B2257
S1134ADC00FFF5705D207920FFFF4712010069602C
S1134AEC0B70010069E00B5417F41FD44DCA0B97DC
S1134AFC5E002D50547001006B2000FFF5387A20B6
S1134B0C00000001460C01006B2000FFF53E1899D4
S1134B1C6889547001006B2000FFF5387A2000007F
S1134B2C000146041900547001006B2000FFF53E90
S10F4B3C6E090010E94017D10D105470F1
S10B939C0000000000000000C6
S1134B4801006DF501006DF401006DF301006DF2D4
S1134B5801006DF101006D120100691301006F017D
S1134B680004010069000D840D8C796C7FF0792CA9
S1134B787FF047540DAC796C7FF0792C7FF0475662
S1134B880D8C65AC4B5E1AB144087A300000000105
S1134B9845141AA0451001F06410475A0C444B0AF7
S1134BA879000002400C0C444BF6190040047900CC
S1134BB8FFFF01006D7101006D7201006D7301004B

S1134BC86D7401006D7554700F85796D000F01F0D8
S1134BD8641546DA409E0FA5796D000F01F0643520
S1134BE846CC409C01F064207A607FFFFFFF46AC0F
S10F4BF801F0643146A67900000140B6CC
S1134C046DF20D820C2A4A0217B00D994A04D28020
S1134C1417B15E00522E0C224A0217B00CAA4A02A4
S1094C2417B16D7254701C
S1134C2A01006DF301006DF20D8252120D935203CE
S1134C3A52100928093801006D7201006D7354700E
S11393A4202020202020202020606060606020202076
S11393B4202020202020202020202020202020202020A6
S11393C448101010101010101010101010101010105E
S11393D484848484848484848484101010101010FE
S11393E41081818181818101010101010101010157
S11393F401010101010101010101010110101010100B
S113940410828282828282020202020202020202227
S113941402020202020202020202021010101020CF
S11394240000000000000000000000000000000035
S11394340000000000000000000000000000000025
S11394440000000000000000000000000000000015
S11394540000000000000000000000000000000005
S11394640000000000000000000000000000000F5
S11394740000000000000000000000000000000E5
S11394840000000000000000000000000000000D5
S10994940000000000CF
S10D949A0000000000000000C5
S1134C4A5E002D727A370000003C7A040000000CE3
S1134C5A0AF47A05000000140AF501006FF000381F
S1134C6A01006FF1003401006F73005C7A020000E7
S1134C7A000801006FF2002C19226FF2002A7A064B
S1134C8A000000540AF6686AEA7F46620FE00B7076
S1134C9A7A01000000075E00526A0D0047501A802D
S1134CAA401A01006F70003401006F7100380A90D6
S1134CBA1899688901006F7000380B7001006FF052
S1134CCA00387A20000000084DD81A80010069B024
S1134CDA7C607370470A01006F700060F9FF400837
S1134CEA01006F700060F90168895A0050227A0046
S1134CFA0000001C0AF001006DF07A010000002890
S1134D0A0AF101006F70005C01006DF001006F7021
S1134D1A005C01006DF001006F70006C5E0055FAD3
S1134D2A7A170000000C01006F7000606808A8FF82
S1074D3A460C7A00A6
S1134D3E000000540AF07D0072706F700024792019
S1134D4E07FF464A6E680001A8F0463A0FE00BF0E3
S1134D5E7A01000000065E00526A0D00472801002A
S1134D6E6F7000606808A8014608790000015A00B8
S1134D7E502401006F7000606808A8FF4610790088
S1134D8EFFFF5A005024790000025A0050246F701E
S1134D9E0024793003FF6FF000244D1E01006F7065
S1134DAE005801006DF001006F70005801006DF0A6
S1134DBE5E00538A0B970B97401E01006F700058CD
S1134DCE01006DF001006F70005801006DF05E0080
S1134DDE538A0B970B971B500D0618886EF80023FA
S1134DEE0B560D6017F001006F7100381A90010019
S1134DFE69B0010069F10FD1010069705E0054A61C
S1134E0E01006F70003801006F71002C5E004C04BE
S1134E1E01006FF000301AE67A000000000801006E

S1094E2E6F7100301A90C1
S1134E340F82401401006F7000300AE00AD00FD1D2
S1134E440AE1680868980B760FA01F864DE6400AAE
S1134E540FD00AE0189968890B767A2600000008B7
S1134E644DEE6F7000326F78002E52806F71003AEE
S1134E7419016DF17A01000000080FD05E00542E71
S1134E840B8701006F7600381B760FC10FE05E00BD
S1134E9454A60FE001006F71002C5E004C04010066
S1134EA46FF000301AE67A000000000801006F7109
S1134EB400301A900F82401401006F7000300AE032
S1134EC40AC00FC10AE1680868980B760FA01F8611
S1134ED44DE6400A0FC00AE0189968890B767A26D2
S1134EE4000000084DEE6F7000326F78002E528080
S1134EF46F71003A19011B516DF17A01000000082A
S1134F040FC05E00542E0B8701006F700034010044
S1134F146DF00100693101006DF17A010000002494
S1094F240AF16F70002C7E
S1134F2A5E0052920B970B977A06000000040AF66A
S1134F3A7A000000000801006DF001006F7100386B
S1134F4A0FE05E0051D40B977A00000000080100BD
S10A4F5A6DF00FD10FE05EC3
S1134F6100569E0B970D004F12790000016FF00060
S1134F712A010069300B705A00501A7A00000000B0
S1134F810801006DF001006F7100380FE05E005100
S1134F91D40B977A000000000801006DF00FD10FC8
S1134FA1E05E00569E0B970D004632010069300BFF
S1134FB170010069B001006F70003401006DF001F0
S1134FC100693301006DF37A01000000240AF16FD7
S1134FD170002C5E0052920B970B9740447A0000AD
S1134FE100000801006DF001006F7100380FE05EF1
S1134FF10051D40B977A000000000801006DF00FF7
S1135001C10FE05E00569E0B970D004C146F7000AC
S11350112A460E010069301B70010069B05A004F26
S11250210C19007A170000003C5E002D505470EC
S11350305E002D727A370000002C0FF30F860100FB
S11350406FF100107A000000000101006FF00018FA
S113505001006F7500440A95188868D87A0000002B
S1135060000801006DF00FE17A000000000C0AF067
S11350705E0051D40B9701006F7000445A0051C079
S11350800FA00B707A01000000055E004C040B704A
S113509010307A06000000081A867A000000000823
S11350A01AE001006DF00FA11031103110317A11A7
S11350B0000094A40AE10FB00AE05E0051D40B97FC
S11350C07A000000000801006FF000281A80010038
S11350D06FF0002001006FF0001C7A04000000084C
S11350E01AE401006FF400145A00518201006F703A
S11350F0001401006DF00FB10AE17A000000000C0A
S11351000AF00AE05E00569E0B970D004D3C01002D
S11351106DF40FB00AE001006DF07A010000001099
S10951200AF10AE11A8006
S11351265E00554E0B970B9717F001006FF00018B2
S113513601006F70002801006F7100200A810100D1
S11351466FF1002001006F7000287A010000000251
S11351565E004C0401006FF000284730790000011F
S11351666DF00FB00AE00FC15E0054D40B87010047
S11351766F70001C0B7001006FF0001C01006F7054
S1135186001C7A200000000458D0FF5A01006F70FB

S11351960018461C6E78002388306CD84004F8301B
S11351A668D81B7501006F7000101F8544F040120C
S11351B66E78002388306CD80FA01B700F8258C0FE
S11151C6FEB87A170000002C5E002D505470C6
S11394A4000000000000000800000000000000505D
S11394B4000000000000032000000000000001F4023
S11394C400000000000013880000000000000C35009B
S11394D400000000007A12000000000004C4B4007D
S11394E4000000002FAF080000000001DCD650008C
S11394F400000012A05F2000000000BA43B7400040
S113950400000746A5288000000048C273950000A8
S11395140002D79883D20000001C6BF526340000A8
S10B9524011C37937E080000CF
S11351D45E002D720F850F9401006F7600181FC5B2
S11351E447401FC544200FD30FC21AC440120FB047
S11351F40B730FA10B710F921B71681968890B74E0
S11352041FE445EA401C0FD30AE30AE40FC21AC49D
S1135214400C0FA01B700F8268086CB80B741FE45A
S10D522445F00FD05E002D505470CA
S113522E01006DF20D9946100D82177253120DA8DF
S113523E53100D810D28401E0F920D8117717908A1
S113524E0010121012311AA144020AA11B5846F281
S10F525E12101710177001006D725470CD
S113526A5E002D720F840F951AE6400E0FC00AE0F6
S113527A680947041900400A0B761FD64DEE7900D8
S10B528A00015E002D50547079
S11352925E002D727A370000000C7A0500000001CF
S11352A20AF50D0C0F967904000801006DF57A01D9
S11352B20000000E0AF101006F70002817B05E00B3
S11352C257640B9701006DF66DFC7A000000001025
S11352D20AF00FD15E0056DC0B970B87790C003F67
S11352E26F70000A190C0DC017F001D053400D0E58
S11352F27900004219C017F001006DF07A01000035
S113530200090FD05E0059B00B97790600084014CC
S11353120D6019E017F00AD00D6117F10AD1680880
S113532268981B561DE64CE8400C0D6017F00AD036
S1135332189968891B560D664CF00DE05240190C02
S11353426DFC7A01000000090FD05E0054D40B8774
S11353527900003F6FF0000A7A01000000400FD08D
S11353625E0058C67A000000000801006DF00FD1FC
S113537201006F70002C5E0051D40B977A17000066
S1095382000C5E002D503B
S105538854705C
S113538A01006DF27A370000001A7A000000001853
S113539A0AF001006F71002601006DF101006F71BF
S11353AA002601006DF17A01000000080AF10100EC
S11353BA6DF15E005A4C7A170000000C6F710018E9
S11353CA0FF05E005D460F817A020000952C7A0089
S11353DA000000080AF05E005E327A000000001046
S11353EA0AF001006F71000C01006DF101006F7189
S11353FA000C01006DF17A01000000080AF10100B6
S113540A6DF15E005B527A170000000C7A0000000F
S113541A00100AF05E005CCC7A170000001A010043
S107542A6D725470D8
S10B952C3FD34395810624DDC2
S113542E5E002D721B971B87010069F00F946F7935
S113543E001E790D0008199D4754790100FF0DD206

S113544E1A0A4B04101140F80C9B18DD1B740FC67F
S113545E4028010069740AE468450CB816850FC02C
S113546E0D915E005C88684814D868C80DD01A0880
S113547E4B04110540F80C5D1B760FE64CD40CDD86
S113548E4706790100014002191117F10F900B879E
S10B549E0B975E002D505470C2
S11354A65E002D720F860F957A000000000801003A
S11354B66DF01036103610367A01000095340AE185
S11154C60FD05E0051D40B975E002D50547032
S1139534000000000000000100000000000000051E
S1139544000000000000001900000000000000007D7E
S113955400000000000002710000000000000C3550
S11395640000000000003D09000000000001312D4F
S1139574000000000005F5E100000000001DCD65BA
S113958400000000009502F90000000002E90EDD6E
S1139594000000000E8D4A510000000048C273957C
S11395A4000000016BCC41E9000000071AFD498D5E
S11395B40000002386F26FC1000000B1A2BC2EC5D7
S11395C4000003782DACE9D900001158E460913D03
S11395D4000056BC75E2D6310001B1AE4D6E2EF5D6
S11395E4000878678326EAC9002A5A058FC295EDD5
S11395F400D3C21BCECCEDA10422CA8B0A00A4253E
S113960414ADF4B7320334B96765C793FA10079DF1
S11354D45E002D727A370000000A0F83010069F120
S11354E46F790022790C0008199C4752FAFF0DC00A
S11354F41A084B04110A40F80CA218CC1AE64026E9
S11355040FB50AE568540C2816840FD00D915E007C
S11355145CAA685814C868D80DC01A084B04100450
S113552440F80C4C0B76010069701F864DD20CCCED
S1135534470679010001400219 1117F10F907A17F8
S10D55440000000A5E002D505470B1
S113554E5E002D727A370000000C01006FF000042C
S113555E0F9601006F7500280AD61B7601006F7334
S113556E00240AD31B737A02000000011AC44044BC
S113557E6868175068391751191017F01AC00100CF
S113558E69F04C14010069717A11000001000100E9
S113559E69F17900000140021900 17F00F84010030
S11355AE697047041A800F826E78000368E81B75D2
S11355BE1B761B730FD546B87A2400000001460EE6
S11355CE01006F70000446067900FFFF40120FA022
S11355DE7A2000000001460419004004790000001FE
S10F55EE7A170000000C5E002D50547072
S11355FA5E002D727A03000000070F820F950100E7
S113560A6F7600207A04000000180AF46941796170
S113561A800079218000460679 01FFFF4004790161
S113562A00010FA06889694079607FF01190119099
S113563A1190119069D07A000000000701006DF003
S113564A0B740FC10FE05E0051D40B97790000036E
S113565A6DF00FB10FE05E00542E0B8769504F06B1
S113566A7D607070402869500B5069D07D607270FC
S113567A4016790000016DF00FB10FE05E00542E61
S113568A0B8769501B5069D07C60737047E45E00D6
S107569A2D505470C8
S113569E5E002D720F860F9301006F7400180FE5D5
S11356AE0FC44604190040201AE6400A6C586C39A0
S11356BE1C9846060B761FC645F21B756858175085
S11156CE1B73683B175319305E002D50547048

S11356DC5E002D727A37000000100FF60F850F94C1
S11356EC69506F710028091069D001006DF6010033
S11356FC6F71002E0FC05E00610E0B977C60737090
S113570C470869500B5069D0400C7A010000001017
S113571C0FE05E0060BC7A000000004201006DF0F7
S113572C7A01000000100FE05E0059B00B976E6811
S113573C0008E8E06EE800087A00000000090100A8
S113574C6DF00FE10FC05E0051D40B977A17000078
S10B575C00105E002D50547093
S11357645E002D727A37000000187A0500000009E4
S11357740AF50F840F920FC44D087A030000000149
S113578440147A03FFFFFFFF0FC07A01FFFFFFFFFF
S11357945E004C2A0F840FC07A010000001B5E00D8
S11357A44C040F860FC07A010000001B5E004C04FA
S11357B40F947A2300000001462E7A0000000008AB
S11357C401006DF00FE11031103110317A11000036
S11357D496140FD05E0051D40B970FE0103078006D
S11357E46B21000096E440320FC446021B767A0014
S11357F40000000801006DF00FE110311031103189
S11358047A110000967C0FD05E0051D40B970FE001
S1135814103078006B21000096FE6FF100120FC464
S113582447747A23FFFFFFFF460A7A000000001B38
S11358341AC00F8401006F70003001006DF00FA1D6
S11358440FC05E00629C0B970FE646087A230000A4
S107585400014762A3
S113585801006F70003001006DF00FA069006DF05A
S11358687A00000000180AF00FD15E0056DC0B978F
S11358780B877A01000000400FD05E0058C67920DC
S11358880001460E6F7000120B506FF000127D502E
S113589870700FA06F71001269817A000000000810
S11358A801006DF00FD101006F7000345E0051D418
S11158B80B977A17000000185E002D505470F5
S1139614800000000000000000CECB8F27F4200F3A17
S1139624A70C3C40A64E6C5286F0AC99B4E8DAFD24
S1139634DA01EE641A708DEAB01AE745B101E9E480
S11396448E41ADE9FBEBC27DE5D3EF282A242E81BD
S1139654B9A74A0637CE2EE195F83D0A1FB69CD921
S1139664F24A01A73CF2DCD0C3B8358109E84F07BD
S11396749E19DB92B4E31BA99E74D1B791E07E4893
S1139684C428D05AA4751E4CF2D56790AB41C2A42A
S1139694964E858C91BA2655BA121A4650E4DDECDF
S11396A4E65829B3046B0AFA8E938662882AF53F37
S11396B4B080392CC4349DEDDA7F5BF5909668490C
S11396C4873E4F75E2224E68A76C582338ED262354
S11396D4CF42894A5DCE35EB8049A4AC0C5811AE18
S11396E40000005900B3010D016601C0021A0273A0
S11396F402CD0327038003DA0434FFA6FF4CFEF2F2
S1099704FE99FE3FFDE5A6
S111970AFD8CFD32FCD8FC7FFC25FBCBFB72F3
S11358C65E002D727A370000000C0F840F957A065E
S11358D6000000081AB30FD00FE15E004C040AC0A3
S11358E60F820FD00FE15E004C04790000801A0985
S11358F64B04119040F86EF8000511086EF8000B82
S113590611086EF800041B75010069F50FD00FE14D
S11359165E004C040AC00F85010069700FE15E004A
S11359264C04790600801A094B04119640F80FA01F
S113593668086E790005169847280FA068066E78E2

S1135946000B166846086E780004166847087A0343
S113595600000001400C685816E847067A03000069
S113596600017A2300000001463240140CE8175068
S1135976171068591689 68D9100E46041B75FE015F
S1135986685816E847041FC544E21FC5450868580A
S113599614E868D84006790000014002190 07A1716
S10D59A60000000C5E002D50547049
S11359B05E002D721B971B970F82010069F17A031A
S11359C00000000801006F7000200FB15E004C045E
S11359D00FA60A8601006F7000200FB15E004C0411
S11359E0790400801A094B04119440F8686816C8BA
S11359F046500CCD1855400414D5110D0CDD46F856
S1135A00686816584708686814C868E840340FA0E7
S1135A10010069710A90010069F001006F700020B4
S1135A200FB15E004C040FA50A85400C6868470857
S1135A30685814C868D8400A0B76010069701F863D
S10F5A4045EA0B970B975E002D50547045
S1135A4C5E002D720F857A040000000701006F7051
S1135A5C002001006DF001006F70002001006DF05B
S1135A6C5E00606A0B970B970D0647487926000179
S1135A7C460A790004B86BA000FFF578792600027A
S1135A8C460A7900044C6BA000FFF57819667A007E
S1135A9C0000001C0AF00D6117F10A90F9FF6889E8
S1135AAC0B56792600084DE67A000000001C0AF01C
S1135ABC5A005B427A000000001C0AF07A010000D5
S1135ACC97185E002CB20D00470C190069D07A00B0
S1135ADC0000971840607A060000001C0AF6696300
S1135AEC1113111311131113796307FF793303FE88
S1135AFC69D36950791003FE462869500B5069D05D
S1135B0C69607960800F69E00FE30B73400E0FC17E
S1135B1C0FB05E0060BC69501B5069D0696073485C
S1135B2C47EC69607960800F79403FE069E07A0067
S1095B3C0000001C0AF04A
S1135B4201006F7100185E002D245E002D50547009
S10B9718000000000000000046
S1135B525E002D720F8601006F70002001006DF050
S1135B6201006F70002001006DF05E00606A0B9708
S1135B720B970D05474079250001460A790004B8C1
S1135B826BA000FFF57879250002460A7900044CE0
S1135B926BA000FFF57819667A000000001C0AF07A
S1135BA20D6117F10A90F9FF68890B5679260008EF
S1135BB24DE65A005C707A050000001C0AF569552F
S1135BC2111511151115115115796507FF793503FFA4
S1135BD20D5C4C0E7A00000097200FE15E002D242D
S1135BE240607A000000001C0AF00FE15E002D24E1
S1135BF2792C00344C4C0FE50B950BF579090034E5
S1135C0219C90D9017F07901001001D053100D0935
S1135C12400A0FD01BF5191169811B590D994EF2D8
S1135C227900003419C017F07901001001D0531024
S1135C327909FFFF1B584B04101940F8695066900D
S1075C4269D07A01A7
S1135C460000001C0AF10FE20F905E0028400FE0EF
S1135C567A01000097205E002CCE0D00470C7A00D7
S1135C660000001C0AF07D0070707A000000001C22
S1135C760AF001006F7100185E002D245E002D509E
S1055C86547055
S10B97200000000000000003E

S1135C8801006DF20D1A4F14792A00084D04FA0029
S1135C984008680A100A1B5A4EFA688A01006D7296
S1055CA8547033
S1135CAA01006DF20D1A4F14792A00084D04FA0007
S1135CBA4008680A110A1B5A4EFA688A01006D7273
S1055CCA547011
S1135CCC01006DF201006DF101006F010004010090
S1135CDC69000D82111211121112112112796207FF50
S1135CEC0D8A7968000F793203FF4B4A79220053EE
S1135CFC4E44794800107912FFEC4B227922002094
S1135D0C4C0C0D224F1E103112301B5240F40F90CD
S1135D1C7912FFE00D224F0C10301B5240F611305C
S1135D2C0B524BFA796A8000470217B001006D7170
S10D5D3C01006D7254701A8040F2EA
S1135D4601006DF119990D11471A4A0679090800E0
S1135D5617917919040F1B59101144FA1031103198
S1135D661031103101006981 1A9101006F8100041D
S1095D7601006D71547081
S1135D7C0F8501F064155860009A792407FF4714C6
S1135D8C0D44586000820FA501F06435586000780B
S1135D9C580000800FA501F0643558600076406808
S1135DAC0FA501F06435466C0DCC465C0F8501F0F4
S1135DBC64154654405E0F8501F06415473C103161
S1135DCC1230792800104C0C1B5C103112307928DE
S1135DDC00104DF4580000C40FA501F06435471AA8
S1135DEC10331232792A00104C0C1B54103312321C
S1135DFC792A00104DF4580000A81AA21AB3687D32
S1135E0C100D131A58000228790107FF1AA21AB3AE
S1135E1C580001FA790107FF1AA27A03000000085F
S1135E2C68FA580001E801006DF601006DF50100F8
S1135E3C6DF401006DF301006DF201006DF10100D1
S1135E4C6DF07A3700000018691D692565D569F571
S1135E5C691C111C111C111C111C796C07FF692482
S1095E6C111411141114BE
S1135E721114796407FF01006D100100691179683B
S1135E82000F01006F23000401006922796A000FE9
S1135E92792C07FF5870FEE2792407FF5870FF0A36
S1135EA20DCC5870FF1A0D445870FF36094C793CDB
S1135EB203FF792C07FF58C0FF58792CFFCB58D02A
S1135EC2FF426FFC000279480010794A001001007A
S1135ED26FF000040F9601006FF2000801006FF3E8
S1135EE2000C0D1552356FF500160D34529452B154
S1135EF20A946511990009D44406791C000199009A
S1135F026FF400140DC50D1D18990D0452340AC502
S1135F1299000DE452B40AC599000D6452240AC5CE
S1135F2299006FF500120DD50D1D18990D84523489
S1135F320AC50D0452B40AC599000DE452240AC5D8
S1135F4299000D6452A40AC599006FF500100DD58E
S1135F520D1D18990DB352830AB50D0452240AC5B7
S1095F6299000DE452A4B6
S1135F680AC59900528209D24404791A0001091A10
S1135F786F74000852040AC26F7400085284094AF5
S1135F880D5B6F73001001006F7400126F7D0016B4
S1135F9819556F710002113213331334133503 50DA0E1
S1135FA879600100471211321333133413350B513F
S1135FB8792107FF5870FE5401F064454702700BBE
S1135FC80D114E2E113213334402700B0D114A0476

S1135FD80B5140F0732B47180CB8E80B47127A1390
S1135FE800000008440A0B72792A00804D020B5105
S1135FF84020732B471C0CB8E80B47167A13000094
S1136008000844020B72792A01004D06113213333A
S11360180B51113213331132133311321333796A9B
S1136028000F10111011101110111011641A101A68784A
S11360381008131A7A170000001801006D70010088
S1136048698201006F83000401006D7101006D72A4
S107605801006D7360
S111605C01006D7401006D7501006D765470C6
S113606A01006DF57A01000000080AF16818E87F5B
S113607AA87F460A0B716818E8F0A8F047041900CC
S113608A402A6818F01050006898460A0B711AD50E
S113609A400E6818470679000000140100B710B7512
S11360AA7A25000000064DEA7900000201006D75A9
S10560BA54701D
S11360BC5E002D720F840F931AD51B730FB64028F5
S11360CC0FC30AE30FB26838E880175017700F83B9
S11360DC0FA06809100968890FD547080FC00AE09B
S11360EC7D0070000FB51B760FE64CD40FD5470619
S11360FC790100014002191117F10F905E002D5028
S105610C5470CA
S113610E5E002D727A37000000387A050000000415
S113611E0AF50F840F967A000000001001006DF04F
S113612E191101006F7000545E0063800B970FE32B
S113613E0B930BF37A160000000701006FF6003481
S113614E790600030FC00B900BF001006FF00028CF
S113615E7A140000000701006FF4002C5A00628AC3
S113616E6838460C01006F7000346809587000FAE5
S113617E7A000000001001006DF019110FD05E00BF
S113618E63800B9701006F70002801006FF00030E1
S113619E01006F74002C790E0003407C01006F70B8
S11361AE00301A916809FA0810311A0A4EFA1A8049
S11361BE684801F064101A916839FA0810311A0A06
S11361CE4EFA01006F720034010069F01A806828DC
S11361DE01F06401010069705E004C2A01006FF04A
S11361EE00247A000000000401006DF07A01000023
S10961FE00280AF10DE286
S1136204096217F210320AD20FA05E0063380B97AB
S11362141B5E01006F7000301BF001006FF0003053
S11362241BF40DEE4C807926000346247A0000000B
S1136234000A01006DF00D6417F40FC110310AD187
S113624401006F7000540AC00AC05E0051D440229A
S11362547A000000000A01006DF00D6417F40FC109
S113626410310AD101006F7000540AC00AC05E00E5
S113627463380B971B561BF301006F7000341BF03C
S113628401006FF000340D6658C0FEDE7A1700007B
S10B629400385E002D50547028
S113629C5E002D727A370000000C0F860F940D6090
S11362AC7910003F69C00FF10FE05E0054A67A002D
S11362BC0000000801006DF0191101006F70002837
S11362CC5E0063800B971A800F8219550FF64010EE
S11362DC0B760FA00B700F8269407930000869C0F0
S11362EC686847ECFB804004110B0B55686816B8C3
S11362FC47F66DF57A03000000080FA01A830FB15F
S113630C0FE05E00542E0B876940195069C00100E1
S113631C6DF30FE101006F7000285E0051D40B97F1

S10F632C7A170000000C5E002D50547026
S11363385E002D720F860F9401006F7500180AD640
S11363481B760AD41B740FC319CC402268681750F4
S1136358683917510910 09C00D0468E817F4790062
S1136368010001D053040D4C1B751B761B730FD50D
S10B637846DA5E002D5054705B
S11363805E002D721B870F8469F101006F73001A81
S11363900FC51AE6400C0FD00B756E7900016889A2
S11363A00B761FB645F00FC00B875E002D5054705F
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 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 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 usbtest
: PRINT usbtest
: INPUT start,main,Timer,Panel,sci,lcd,usb
: LIB c:\h8\akic\c38hab
: START R(0FFEF10),P(200),D(8000),C(9000)
: 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。

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