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。

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.

```
/************************************************************************/
/*      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 */
      }
}
```

```
/* 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
/* タイマ割り込み用 */
#ifndef USE_BCC
#define TSTR (*(volatile unsigned char *)0xFFFF60)
#define TSNC (*(volatile unsigned char *)0xFFFF61)
#define TMDR (*(volatile unsigned char *)0xFFFF62)
#define TFCR (*(volatile unsigned char *)0xFFFF63)
#define TOER (*(volatile unsigned char *)0xFFFF90)
#define TOCR (*(volatile unsigned char *)0xFFFF91)
#define TCR0  (*(volatile unsigned char *)0xFFFF64)
#define TIOR0 (*(volatile unsigned char *)0xFFFF65)
#define TIER0 (*(volatile unsigned char *)0xFFFF66)
#define TSR0  (*(volatile unsigned char *)0xFFFF67)
#define TCNT0 (*(volatile unsigned int *)0xFFFF68)
#define GRA0 (*(volatile unsigned char *)0xFFFF6A)
#define GRB0 (*(volatile unsigned char *)0xFFFF6C)
#endif
/* 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)
{
	TSTR = 0x01; /* timer 0 enable */
	TSNC = 0;
	TMDR = 0x0;
	TFCR = 0x0;
	TOER = 0x0;
	TOCR = 0xff;
	TCR0 = 0x00; /* 分周なし */
	TIOR0 = 0x88;
	TIER0 = 0x04; /* オーバーフロー割り込み許可 */
	TCNT0 = 0xffff - 25000; /* 1 msec interval */
	GRA0 = 0;
	GRB0 = 0;
```

```
}

/* タイマ割り込み用 */
void InterruptITU0(void)
{
	TSR0 &= 0xfb;
	TCNT0 = 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

#ifndef USE_BCC
#define PB_DDR (*(volatile unsigned char *)0xFFFFD4) /* PB DDR Address*/
#define PB_DR (*(volatile unsigned char *)0xFFFFD6) /* PB DR  Address*/
#define GRA0 (*(volatile unsigned char *)0xFFFF6A)
#define GRB0 (*(volatile unsigned char *)0xFFFF6C)
#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;
		GRA0 = 0;
		GRB0 = 0;
	}
#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
```

```
-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
```

# 付録

```
 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 ***     01/27/17 19:11:48
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 ***     01/27/17 19:11:48
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 ***     01/27/17 19:11:48
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 ***    01/27/17 19:11:48
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'0000111D   H'00000E6E
                                main                     main
                    H'0000111E   -   H'0000179D   H'00000680
                                Timer                    Timer
                    H'0000179E   -   H'00001899   H'000000FC
```

```
                    Panel                   Panel
               H'0000189A   -   H'0000196B   H'000000D2
                    sci                     sci
               H'0000196C   -   H'00001BA9   H'0000023E
                    lcd                     lcd
               H'00001BAA   -   H'0000269B   H'00000AF2
                    usb                     usb
               H'0000269C   -   H'000026D5   H'0000003A
                    rand                    rand
               H'000026D6   -   H'00002733   H'0000005E
                    sprintf                 sprintf
               H'00002734   -   H'0000275D   H'0000002A
                    strcmp                  strcmp
               H'0000275E   -   H'00002779   H'0000001C
                    strlen                  strlen
               H'0000277A   -   H'000027A9   H'00000030
                    vsprintf                vsprintf
               H'000027AA   -   H'00002A83   H'000002DA
                    addd3                   addd3
               H'00002A84   -   H'00002CB5   H'00000232
                    divd3                   divd3
               H'00002CB6   -   H'00002CC1   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'00002CC2   -   H'00002CD1   H'00000010
                         ged3                    ged3
               H'00002CD2   -   H'00002CE1   H'00000010
                         ltd3                    ltd3
               H'00002CE2   -   H'00002D27   H'00000046
                         ltod3                   ltod3
               H'00002D28   -   H'00002D45   H'0000001E
                         mv83                    mv83
               H'00002D46   -   H'00002D53   H'0000000E
                         ned3                    ned3
               H'00002D54   -   H'00002D75   H'00000022
                         spregld3                spregld3
               H'00002D76   -   H'00002D9D   H'00000028
                         spregsv3                spregsv3
               H'00002D9E   -   H'00004B4B   H'00001DAE
                         _fmtout                 _fmtout
               H'00004B4C   -   H'00004C07   H'000000BC
                         cmpd3                   cmpd3
               H'00004C08   -   H'00004C2D   H'00000026
                         divl3                   divl3
```

H'00004C2E - H'00004C4D H'00000020
mull3 mull3
H'00004C4E - H'00005033 H'000003E6
_dti _dti
H'00005034 - H'000051D7 H'000001A4
_its _its
H'000051D8 - H'00005231 H'0000005A
memcpy memcpy
H'00005232 - H'0000526D H'0000003C
divul3 divul3
H'0000526E - H'00005295 H'00000028
_allzero _allzero
H'00005296 - H'0000538D H'000000F8
_calcnpw _calcnpw
H'0000538E - H'00005431 H'000000A4
_log10 _log10
H'00005432 - H'000054A9 H'00000078
_lsfts _lsfts
H'000054AA - H'000054D7 H'0000002E
_pow5 _pow5
H'000054D8 - H'00005551 H'0000007A
_rsfts _rsfts
H'00005552 - H'000055FD H'000000AC
_sub _sub
H'000055FE - H'000056A1 H'000000A4
_unpack _unpack



*** LINKAGE EDITOR LINK MAP LIST ***

SECTION NAME START - END LENGTH
UNIT NAME MODULE NAME

ATTRIBUTE : CODE NOSHR

P H'000056A2 - H'000056DF H'0000003E
memcmp memcmp
H'000056E0 - H'00005767 H'00000088
_mult64 _mult64
H'00005768 - H'000058C9 H'00000162
_power _power
H'000058CA - H'000059B3 H'000000EA
_rnd _rnd
H'000059B4 - H'00005A4F H'0000009C
_setsbit _setsbit
H'00005A50 - H'00005B55 H'00000106
frexp frexp
H'00005B56 - H'00005C8B H'00000136
modf modf
H'00005C8C - H'00005CAD H'00000022
dslc3 dslc3
H'00005CAE - H'00005CCF H'00000022

```
                        dsruc3                  dsruc3
                H'00005CD0   -   H'00005D49   H'0000007A
                        dtol3                   dtol3
                H'00005D4A   -   H'00005D7F   H'00000036
                        itod3                   itod3
                H'00005D80   -   H'0000606D   H'000002EE
                        muld3                   muld3
                H'0000606E   -   H'000060BF   H'00000052
                        _duchek                 _duchek
                H'000060C0   -   H'00006111   H'00000052
                        _lsft                   _lsft
                H'00006112   -   H'0000629F   H'0000018E
                        _mult                   _mult
                H'000062A0   -   H'0000633B   H'0000009C
                        _pow10                  _pow10
                H'0000633C   -   H'00006383   H'00000048
                        _add                    _add
                H'00006384   -   H'000063B3   H'00000030
                        memset                  memset

* TOTAL ADDRESS *        H'00000200   -   H'000063B3   H'000061B4


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'00002874       DAT
        $CMPD$3                   H'00004B4C       DAT
        $DIVD$3                   H'00002B1E       DAT
        $DIVL$3                   H'00004C08       DAT
        $DIVUL$3                  H'00005232       DAT
        $DSLC$3                   H'00005C8C       DAT
        $DSRUC$3                  H'00005CAE       DAT
        $DTOL$3                   H'00005CD0       DAT
        $EQD$3                    H'00002CB6       DAT
        $GED$3                    H'00002CC2       DAT
        $ITOD$3                   H'00005D4A       DAT
        $LTD$3                    H'00002CD2       DAT
        $LTOD$3                   H'00002CE2       DAT
        $MULD$3                   H'00005E36       DAT
        $MULL$3                   H'00004C2E       DAT
        $MV8$3                    H'00002D28       DAT
        $NED$3                    H'00002D46       DAT
        $SUBD$3                   H'00002844       DAT
        $sp_regld$3               H'00002D54       DAT
        $sp_regsv$3               H'00002D76       DAT
        _Clear                  H'0000179E       ENT
        _ClearLCD                 H'00001A8A       ENT
        _Cnt                    H'00FFEF22       DAT
        _Destroy                  H'00000F88       ENT
        _DisableInterrupt           H'000002AC       DAT
        _DispUSBPort                H'00001CBA       ENT
        _EnableInterrupt            H'000002A8       DAT
        _GetSCI                   H'000018CA       ENT
        _GetSW                    H'000010CC       ENT
        _H8init                   H'000010F2       ENT
        _Init                   H'00000D76       ENT
        _InitITU                  H'000016DC       ENT
        _InitLCD                  H'00001A1A       ENT
        _InitSCI                  H'0000189A       ENT
        _InitUSB                  H'00001BC0       ENT
        _InterruptITU0              H'0000170E       ENT
        _LCDOut4                  H'000019C2       ENT
        _LocateLCD                  H'00001ACA       ENT
        _PrintF                   H'000017C6       ENT
        _PrintLCD                   H'00001AEE       ENT
        _PrintSCI                 H'000018EA       ENT
```

| | | |
|---|---|---|
| _PutLCD | H'00001A9E | ENT |
| _PutSCI | H'000018BA | ENT |
| _Repaint | H'000004DA | ENT |
| _Run | H'00000546 | ENT |
| _ScanSCI | H'000018DA | ENT |
| _SetLED | H'0000107E | ENT |
| _SleepMSec | H'000011B4 | ENT |
| _Start | H'000014A0 | ENT |
| _Stop | H'000014CE | ENT |
| _Thread_Start | H'00001566 | ENT |



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

| SYMBOL NAME | ADDR | TYPE |
|---|---|---|
| _Thread_Toggle | H'000015A4 | ENT |
| _Thread_checkAllDelete | H'000014EC | ENT |
| _Thread_checkStayAnother | H'00001506 | ENT |
| _Thread_getThread | H'0000152C | ENT |
| __add | H'0000633C | ENT |
| __allzero | H'0000526E | ENT |
| __calcnpw | H'00005296 | ENT |
| __ctype | H'000093A4 | DAT |
| __dti | H'00004C4E | ENT |
| __duchek | H'0000606E | ENT |
| __errno | H'00FFF578 | DAT |
| __fmtout | H'00002D9E | ENT |
| __its | H'00005034 | ENT |
| __log10 | H'0000538E | ENT |
| __lsft | H'000060C0 | ENT |
| __lsfts | H'00005432 | ENT |
| __mult | H'00006112 | ENT |
| __mult64 | H'000056E0 | ENT |
| __pow10 | H'000062A0 | ENT |
| __pow5 | H'000054AA | ENT |
| __power | H'00005768 | ENT |
| __rnd | H'000058CA | ENT |
| __rnext | H'00FFF574 | DAT |
| __rsfts | H'000054D8 | ENT |
| __setsbit | H'000059B4 | ENT |
| __sub | H'00005552 | ENT |
| __unpack | H'000055FE | ENT |
| _countUpNextRun | H'000012F4 | ENT |
| _delete_ | H'0000146C | ENT |
| _frexp | H'00005A50 | ENT |
| _getClock | H'0000111E | ENT |
| _get_inbufflen | H'00002674 | ENT |
| _get_outbufflen | H'00002688 | ENT |
| _i_cnt | H'00FFEF26 | DAT |
| _initWOVI | H'00001782 | ENT |
| _init_usbbuff | H'000024D8 | ENT |
| _j_cnt | H'00FFEF28 | DAT |
| _main | H'000002B0 | ENT |

| | | |
|---|---|---|
| _memcmp | H'000056A2 | ENT |
| _memcpy | H'000051D8 | ENT |
| _memset | H'00006384 | ENT |
| _modf | H'00005B56 | ENT |
| _new_Thread | H'00001314 | ENT |
| _nextRun | H'000012B0 | ENT |
| _rand | H'0000269C | ENT |
| _read_buff | H'00002616 | ENT |
| _read_outbuff | H'000025B8 | ENT |
| _sprintf | H'000026D6 | ENT |
| _strcmp | H'00002734 | ENT |
| _strlen | H'0000275E | ENT |
| _tag_SleepMSec | H'00001130 | 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'00001D1E | ENT |
| _vsprintf | H'0000277A | ENT |
| _wovi | H'00001736 | ENT |
| _woviClock | H'00FFEF8A | DAT |
| _woviInit | H'000016AA | ENT |
| _woviRun | H'000015E8 | ENT |
| _woviThreadFirst | H'00FFEF92 | DAT |
| _woviThreadLast | H'00FFEFB0 | DAT |
| _write_buff | H'00002552 | ENT |
| _write_inbuff | H'000024F6 | ENT |

S00E0000757362746573742 04D4F54D8
S113000000002000000022A0000022A0000022A67
S11300100000022A0000022A0000022A0000022A2D
S11300200000022A0000022A0000022A0000022A1D
S10B00300000022A0000022A6D
S11300380000022A0000022A0000022A0000022C03
S11300600000022A0000022A0000026A0000022A9D
S11300700000022A0000022A0000022A0000022ACD
S11302007A0700FFFF107A00000080007A0100FFE8
S1130210EF107A0200FFEF201F92587000086C0B5A
S1130220689B0B7140F25E0002B0567001006DF0E6
S113023001006DF101006DF201006DF301006DF439
S113024001006DF501006DF65E001D1E01006D7667
S113025001006D7501006D7401006D7301006D7215
S113026001006D7101006D70567001006DF00100A9
S11302706DF101006DF201006DF301006DF40100F9
S11302806DF501006DF65E00170E01006D7601003D
S11302906D7501006D7401006D7301006D720100D5
S11302A06D7101006D705670063F547004C0547038
S11302B05E002D767A370000000A790B00011933AE
S11302C00FF47A0600FFEF4A19550B5579257FFF86
S11302D04DF85C000E1C5E00189A5E001A1A0D3869
S11302E00D305C000D980D380DB05C000D900D388D
S11302F0790000025C000D860D38790000035C0074
S11303000D7C5E001BC07FF472507FF570505E0061
S113031002A80D3228F117506DF07A00000090000A
S113032001006DF05E0018EA0B970B877A0000005E
S1130330900F01006DF05E001AEE0B9718886EC8DF
S113034000036EC800026EC8000168C85E00178211
S11303507A00000007D05E0011B47900001E5E0031
S113036013140F857900001F5E00131401006FF052
S113037000040FD05E0014A001006F7000045E0043
S113038014A001006B200000927801006DF00100C1
S10E03906B200000927401006DF05E12
S113039B0017360B970B975E001506792000014665
S11303ABD65E00179E0FD05E00146C7A010000908E
S11303BB1B0D305E0017C67A00000007D05E0011DC
S11303CBB45E00179E19550D505C000CF40D0D0D0A
S11303DB5117F10AC16819D90117D1660147540D99
S11303EBB80D505C000C8C0D5009B06DF07A000009
S11303FB00902C01006DF00FE05E0026D60B970BDF
S113040B8701006DF67A000000903101006DF05EFC
S113041B0018EA0B970B970FE05E00275E09B00DF0
S113042B010FE05E00255201006DF65E001AEE0B24
S113043B9740080D380D505C000C380DD017F50F95
S113044BC10AD168980B557925000458D0FF785E03
S113045B0018DA0D0047165E0018CA17D00D055E9B
S111046B001A9E68ED0DB10FE05E0025525E93
S11304790026740D004738790100400FE05E00261D
S1130489160D057A010000903501006DF15E001A21
S1130499EE0B9701006DF65E001AEE0B9701006DE6
S11304A9F65E0018EA0B970D510FE05E0025520D19
S11304B928790000035C000BBC0D20795000010D65
S11304C90219550B55792527104DF85A0003D054B5
S11304D9705E002D767A0500FFEF2219665E00171C
S11304E99E19EE400E7A01000090370D605E0017E9
S11304F9C60B5E69501D0E4DEC7A01000090390D53

S1130509605E0017C67A010000903D0D605E00171A
S10A0519C619EE400E7A0142
S1130520000090370D605E0017C60B5E6F5000022F
S11305301D0E4DEA7A010000903F0D605E0017C664
S11305405E002D5454705E002D767A370000000A49
S11305507A03000000017A04000007D019550F86C2
S1130560188886EF800036EF800026EF8000168F850
S113057069607920000146365C00FF5E7A0000FF67
S1130580EF2269010B5169815E00269C17F0790205
S1130590000901D053207918000A790000645280C1
S11305A017F00F810FE05E0012B05A000D6A696008
S11305B07920000246365C00FF207A0000FFEF241A
S11305C069010B5169815E00269C17F079020009CD
S11305D001D053207918000A79000064528017F083
S11305E00F810FE05E0012B05A000D6A6960792036
S11305F0000B586000B601006F6000127A20000003
S1130600000146205E00179E7A01000090430D50C2
S10906105E0017C67A012B
S10A06160000076C0FE05E1A
S113061D0012F45A000D6A01006F6000127A200077
S113062D00000246265E00179E7A01000090540DCD
S113063D505E0017C67A010000905B0D505E0017E7
S113064DC60FE05E0014CE5A000D6A01006F600004
S113065D127A2000000003461E5E00179E7A0100E9
S113066D0090660D505E0017C67A01000005DC0F81
S113067DE05E0012F4402401006F6000127A200046
S113068D00000446165E00179E7A01000090770D58
S10F069D505E0017C60FE05E0014CE5A3A
S11306A9000D6A69607920000C586000A801006F89
S11306B96000127A200000000146205E00179E7A2E
S11306C901000090880D505E0017C67A0100000DE5
S11306D9480FE05E0012F45A000D6A01006F6000D2
S11306E9127A200000000246205E00179E7A01005C
S11306F90090990D505E0017C67A0100000D480F4E
S1130709E05E0012F45A000D6A01006F6000127A6C
S11307192000000003461E5E00179E7A0100009028
S1130729AA0D505E0017C67A0100000D480FE05E5E
S10E07390012F4401C5E00179E7A01C2
S1130744000090BB0D505E0017C60FE05E0014CE90
S11307540FE05E00146C5A000D6A69607920000D85
S1130764586000A801006F6000127A2000000001A5
S113077446205E00179E7A01000090C30D505E0070
S1130778417C67A01000013EC0FE05E0012F45A005E
S11307940D6A01006F6000127A20000000024620F7
S11307A45E00179E7A01000090D40D505E0017C6B8
S11307B47A01000013EC0FE05E0012F45A000D6A94
S11307C401006F6000127A2000000003461E5E00E1
S10707D4179E7A01EE
S11307D8000090E50D505E0017C67A01000013EC87
S11307E80FE05E0012F4401C5E00179E7A010000C1
S11307F890F60D505E0017C60FE05E0014CE0FE0B2
S11308085E00146C5A000D6A69607920000E586006
S1130818800E401006F6000127A2000000001462006
S11308285E00179E7A01000090FE0D505E0017C609
S11308387A01000017700FE05E0012F45A000D6A87
S1130848 01006F6000127A200000000246405E003B
S1130858179E7A010000910F0D505E0017C67A01AA

S10A0868000017700FE05EB2
S113086F0012F47A01000091160D505E0017C6793D
S113087F00000B5E0015660F867A01000005DC5E33
S113088F0012F45A000D6A01006F6000127A200003
S113089F000003461E5E00179E7A01000091210D92
S11308AF505E0017C67A01000017700FE05E00124A
S11308BFF4403801006F6000127A200000000446F4
S11308CF2A7900000B5E00152C0F8247060FA05EDE
S11308DF00146C7A01000091320D505E0017C60FA1
S11308EFE05E0014CE0FE05E00146C5A000D6A69CF
S11308FF607920001346440FB10FE05E0012B01968
S113090FDD0DD05C0007B60DD117F10FF20A926817
S113091F2ADA0117D2660247160DD0791000145E3A
S113092F0015A47A01000003E80FE05E0012B00B7C
S113093F5D792D00044DCA5A000D6A696079200054
S113094F1446187A01000091390D505E0017C60F37
S113095FC10FE05E0012F45A000D6A69607920003E
S113096F1546187A010000913B0D505E0017C60F14
S113097FC10FE05E0012F45A000D6A69607920001E
S113098F1646187A010000913D0D505E0017C60FF1
S113099FC10FE05E0012F45A000D6A6960792000FE
S10F09AF1746187A010000913F0D505EBE
S11309BB0017C60FC10FE05E0012F45A000D6A69EF
S11309CB607920001E465E01006F60001246240111
S11309DB006F6000120B7001006FE0001228D6E865
S11309EB0E38D67A01000003E80FE05E0012B05A0E
S11309FB000D6A01006F6000127A2000000001589D
S1130A0B60035C01006F6000121B7001006FE0005C
S1130A1B127FD670007A01000003E80FE05E00122C
S1130A2BB05A000D6A69607920001F5860033001CA
S1130A3B006F600012460C1A910FE05E0012F45A1D
S1130A4B000D6A01006F6000127A2000000001465E
S1130A5B247A010000903D0D505E0017C67A010009
S1130A6B0091410D505E0017C60FC10FE05E0012DF
S1130A7BF45A000D6A01006F6000127A2000000027
S1130A8B02463019DD0DD00B505E0013140DD21737
S10E0A9BF210321032010078A06BA0B3
S1130AA600FFEF2A0B5D792D00024DDE0FB10FE03B
S1130AB65E0012F45A000D6A01006F6000127A207C
S1130AC60000000346566B2000FFEF227920000D3D
S1130AD64D1A790000016BA000FFEF2601006F603D
S1130AE600120B7001006FE0001240246B2000FF20
S1130AF6EF247920000D4D18790000026BA000FF4A
S1130B06EF2601006F6000120B7001006FE0001208
S1130B160FB10FE05E0012B05A000D6A01006F605C
S1130B2600127A2000000004465A5E00179E6B20CE
S1130B3600FFEF2679200001460E7A01000091524C
S1130B460D505E0017C640186B2000FFEF26792074
S1100B560002460C7A01000091630D505E11
S1130B630017C601006B2000FFEF2A5E00146C011F
S1130B73006B2000FFEF2E5E00146C0FC10FE05ECD
S1130B830012F45A000D6A01006F6000127A20000C
S1130B93000005461C5E00179E7A010000901B0DA2
S1130BA3505E0017C60FC10FE05E0012F45A000D2A
S1130BB36A01006F6000127A2000000006461C5E83
S1130BC300179E7A01000091740D505E0017C60F43
S10B0BD3C10FE05E0012F45AA9

S1130BDB000D6A01006F6000127A200000000746C7
S1130BEB365E00179E19DD0DD07910000B5E0013D6
S1130BFB140DD217F210321032010078A06BA00043
S1130C0BFFEF320B5D792D00044DDC0FB10FE05E6E
S1130C1B0012F45A000D6A01006F6000127A200073
S1130C2B00000846245E00150679200002460E01DB
S1130C3B006F6000120B7001006FE000120FB10F19
S1130C4BE05E0012B05A000D6A01006F6000127A69
S1130C5B2000000009460C0FC10FE05E0012F45A8E
S1130C6B000D6A01006F6000127A200000000A4633
S1130C7B1C5E00179E7A010000901B0D505E00173F
S1130C8BC60FC10FE05E0012F45A000D6A01006F2C
S10F0C9B6000127A200000000B461C5E73
S1130CA700179E7A01000091850D505E0017C60F4D
S1130CB7C10FE05E0012F45A000D6A01006F600075
S1130CC7127A200000000C4632790000135E0013ED
S1130CD71401006BA000FFEF425E0014A07900002F
S1130CE7145E00131401006BA000FFEF465E0014AF
S1130CF7A00FB10FE05E0012F4406801006F6000BF
S1130D07127A200000000D462E5E0015067920009A
S1130D1703461A01006B2000FFEF425E00146C01CB
S1130D27006F6000120B7001006FE000120FB10F2C
S1130D37E05E0012B0402C01006F6000127A2000C1
S1130D4700000E460A0FC10FE05E0012F4401401C3
S1130D57006F6000127A200000000F46060FE05E66
S10D0D6700146C7A170000000A5E06
S1130D71002D5454705E002D760F86790400097A94
S1130D810500FFEF22696079200001462619006 9F9
S1130D91D05E00269C17F001D053407918000A79E0
S1130DA100001E528017F00F810FE05E0012B05A4F
S1130DB1000F8269607920000246281900 6FD00074
S1130DC1025E00269C17F001D053407918000A797E
S1130DD100001E528017F00F810FE05E0012B05A1F
S1130DE1000F82696C792C00034D36792C00084E73
S1130DF13069601B5017F010300A85190069D05E05
S1130E0100269C17F001D053407918000A7900009D
S1130E11C8528017F00F810FE05E0012B05A000F25
S1130E21827A04000017C6195569607920000B46C0
S1130E311A7A01000091960D505D407A0100000578
S1130E41DC0FE05E0012F45A000F82696079200022
S1130E510C461A7A010000919E0D505D407A010003
S1090E61000BB80FE05E78
S1130E670012F45A000F8269607920000D46247A34
S1130E77010000903D0D505D407A01000091A80DDF
S1130E87505D407A01000011940FE05E0012F45A9E
S1130E97000F8269607920000E461A7A01000091DB
S1130EA7B00D505D407A01000017700FE05E00122D
S1130EB7F45A000F827A03000007D0696079200093
S1130EC71446245E00179E7A01000091BA0D505D07
S1130ED7407A010000903D0D505D400FB10FE05E79
S1130EE70012F45A000F826960792000154 6225ECA
S1080EF700179E7A01C3
S1130EFC000091CB0D505D407A010000903D0D50E8
S1130F0C5D400FB10FE05E0012F4406A6960792016
S1130F1C001646225E00179E7A01000091DC0D50EC
S1130F2C5D407A010000903D0D505D400FB10FE024
S1130F3C5E0012F4404069607920001746225E007F

S1130F4C179E7A01000091ED0D505D407A0100006F
S1130F5C903D0D505D400FB10FE05E0012F4401652
S1130F6C69667926001E460EF8FF38D438D61888E1
S1130F7C386A1888386C5E002D5454705E002D76D8
S1130F8C7A04000017C619660F8569507920000B87
S1130F9C46105E00179E7A01000091FE0D605D40C5
S10F0FAC404469507920000C460C7A0187
S1130FB8000092090D605D40403069507920000DB2
S1130FC8460C7A01000092130D605D40401C695085
S1130FD87920000E46147A010000903D0D605D40B3
S1130FE87A010000921D0D605D406950792000145C
S1130FF8461A5E00179E7A010000922E0D605D402E
S11310087A010000903D0D605D404064695079208D
S11310180015461A5E00179E7A010000923F0D6084
S11310285D407A010000903D0D605D40404269508B
S113103879200016461A5E00179E7A010000925026
S11310480D605D407A010000903D0D605D404020D9
S1111058695579250017461 85E00179E7A0128
S1131066000092620D605D407A010000903D0D60C4
S11310765D405E002D5454706DF66DF50D067909CD
S1131086000128D617500D950CE91A094B041015C3
S113109640F866050D8846121A0E4B04101940F8DF
S11310A60D9029D6148939D640141A0E4B041019FB
S11310B640F8795900FF0D9029D6168939D60D5077
S11310C66D756D7654706DF60D0628D3175079013C
S11310D600011A0E4B04101140F86610470419114B
S11310E64004790100010D106D765470F80638EC52
S11310F6F8FF38C038C1188838D8F8FF38C81888B8
S113110638DBF8FF38C9F8DF38D0F8FF38CDF8F008
S10B111638D1F8FF38D45470FE
S1139000435055204D4F444520253032580A000C1B
S113901052656164792133303532004E455854200E
S1139020202020202020202020202020007377257559
S11390300025730A000C0020003C313E000A003C6E
S1139040323E003C313E3173742020202020200A
S1139050202020003C313E326E64003C313E53748C
S11390606F70202020003C313E337264202020208A
S113907020202020202020003C313E53746F7020209C
S113908020202020202020003C323E3173742020F9
S113909020202020202020003C323E326E6420FD
S11390A020202020202020202020003C323E337264E8
S11390B020202020202020202020003C323E5374FA
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
S113111E7A0000FFEF8A01006F7100045E002D2834
S113112E54705E002D767A37000000180F867A0011
S113113E000092967A01000000100AF15E002D283D
S113114E0FE10FF05E002CE20F817A020000929EF7
S113115E7A00000000080AF05E002B1E40247A017C
S113116E000092A67A02000000300AF20FF05E0031
S113117E2B1E7A01000000100AF10F820F905E0001
S113118E28747A00000000100AF07A0100000008AB
S113119E0AF15E002CD20D0046C47A170000001827
S11311AE5E002D5454705E002D767A3700000020B9
S11311BE19660F857A02000000180AF201006DF21B
S11311CE5C00FF4C0B970FD10FF05E002CE20F81EA
S11311DE7A020000929E7A00000000100AF05E0070
S11311EE2B1E7A00000000180AF07A010000000896
S11311FE0AF15E002D2840747A02000000080AF2FC
S109120E01006DF25C001B
S1091214FF080B977A00AE
S113121A00FFEF8A7A01000092AE5E002CC20D0035
S113122A474E7A010000927C0D605E0017C67A0170
S113123A0000927E0D605E0017C67A0100FFEF8AF6
S113124A7A02000092AE0F905E00284401006B20E0
S113125A000092BA01006DF001006B20000092B603
S113126A01006DF00FD05C00FEBC0B970B97402A70
S113127A7A01000000180AF17A02000000100AF24B
S113128A0FF05E0028740F817A00000000080AF04C
S113129A5E002CD20D005860FF627A17000000200E
S11312AA5E002D5454705E002D761B971B970F8694
S11312BA0F957A02000000020AE201006DF25C0057
S11312CAFE520B970FD10FF05E002CE20F817A02C8
S11312DA0000929E7A000000000A0AE05E002B1EBC
S11312EA0B970B975E002D5454705E002D760F8674
S11312FA0F950FE055B001006F6000120B700100EB
S113130A6FE000125E002D5454705E002D760D05B9
S113131A7A0400FFEF92400601006F44001A0100AD
S113132A6F40001A01006F01001A46EC792500018B
S113133A460A7A0600FFEFCE5A0013F27925000215
S113134A460A7A0600FFEFEC5A0013F27925000BDE
S113135A460A7A0600FFF00A5A0013F27925000CAE
S113136A460A7A0600FFF0285A0013F27925000D7F
S107137A46087A069E
S113137E00FFF046406E7925000E46087A0600FF00
S113138EF0644060792500134 6087A0600FFF08268
S113139E40527925001446087A0600FFF0A0404417

S11313AE7925001546087A0600FFF0BE40367925EA
S11313BE001646087A0600FFF0DC40287925001750
S11313CE46087A0600FFF0FA401A7925001E4608F1
S11313DE7A0600FFF118400C7925001F46067A069F
S11313EE00FFF1360FE6461C7A010000927C1900CD
S11313FE5E0017C67A010000928719005E0017C6B9
S113140E1A80405401006FE4001601006F40001A69
S113141E01006FE0001A01006F86001601006FC60F
S113142E001A7A00000092BE7A01000000020AE15F
S104143E5E4C
S113143F002D287A00000092967A010000000A0A14
S113144FE15E002D2869E51A8001006FE000120F9D
S113145FE05E000D760FE05E002D54547001006DB9
S113146FF60F865E000F8801006F60001601006F94
S113147F61001A01006F81001A01006F60001A01E9
S113148F006F61001601006F81001601006D765425
S113149F705E002D760F867A0200FFEF8A01006DD2
S11314AFF25C00FC6A0B977A0000FFEF8A7A010067
S11314BF0000020AE15E002D285E002D54547001D6
S11314CF006DF60F867A00000092C67A01000000C5
S11314DF020AE15E002D2801006D76547001006B46
S11314EF2000FFEFAC01006F01001A4604190054EE
S11314FF70790000001547001006B2100FFEFAC0104
S113150F006F11001A1988400C01006F11001A0D9A
S113151F800B500D080F9146F00D8054706DF50D33
S108152F0501006B2023
S113153400FFEFAC01006F01001A472001006B218B
S113154400FFEFAC69101D5046040F9040100100DA
S11315546F11001A01006F10001A46E81A806D75A6
S113156454705E002D760D0555BE0F86460E0D5044
S11315745C00FD9C0F865C00FF22401C7A00000087
S113158400020AE07A01000092C65E002CB60D0048
S113159447060FE05C00FF040FE05E002D54547017
S11315A45E002D760D0555800F86460E0D505C00AA
S11315B4FD5E0F865C00FEE440247A000000000216
S11315C40AE07A01000092C65E002CB60D004708BB
S11315D40FE05C00FEC640060FE05C00FE8A5E007E
S11315E42D5454705E002D761B971B977A0500FFCC
S11315F4EF9201006F56001A5A00169601006F65A8
S1131604001A7A00000000020AE07A01000092BE88
S11316145E002D460D0047787A00000000020AE0C0
S11316247A01000092C65E002D460D0047627A01DE
S1091634000000020AE1C0
S10E163A7A020000000A0AE20FF05ED3
S11316450028747A0000FFEF8A7A01000000020A7D
S1131655E15E002CD20D00470E0FF17A02000092D5
S1131665AE0F905E0028447A0000FFEF8A0FF15E0B
S1131675002CC20D0047187A0000FFEF8A7A01009B
S11316850000020AE15E002D280FE05E0005460F0B
S1131695D601006F60001A5860FF600B970B975EC9
S11316A5002D5454701A8001006BA000FFEFA87A37
S11316B50000FFEFB001006BA000FFEFAC7A000064
S10A16C5FFEF9201006BA08F
S11316CC00FFEFC61A8001006BA000FFEFCA547035
S11316DCF80138601888386138621888386338909 4
S11316ECF8FF389118883864F8883865F804386638
S11316FC79009E576B80FF681888386A1888386C95

S113170C547001006DF27F67722079009E576B80D5
S113171CFF687A0100FFEF8A7A02000092CE0F90E5
S113172C5E00287401006D72547001006DF27A0032
S113173C00FFEF8A7A01000092AE5E002CC20D000E
S113174C472A7A010000927C19005E0017C67A01C1
S113175C0000927E19005E0017C67A0100FFEF8A23
S113176C7A02000092AE0F905E0028445C00FE6C7F
S10B177C01006D7254707A0044
S1131784000092967A0100FFEF8A5E002D285C0028
S10D1794FF465E0002A85A0016AAE1
S113927C0A004F564552574F56490063616C6C6F49
S113928C63206661696C656400000000000000000E7
S113929C0000408F4000000000003FF00000000081
S11392AC0000412E84800000000040CF40000000ED
S11392BC0000412E848200000000412E84840000B3
S10D92CC00003F50624DD2F1A9FCEF
S113179E7A00000092D601006DF07A0000FFF1543A
S11317AE5E0026D60B977A0000FFF15401006DF010
S11317BE5E001AEE0B9754705E002D767A0600FFCC
S11317CEF1940D040F950C44586000BAAC00470E0B
S11317DEAC01587000B0AC02476E5A0018947A01EF
S11317EE000092D80FD05E0027340D00461E7A00FB
S11317FE000092DE01006DF07A00000092DB010022
S113180E6DF00FD05E0026D60B970B9701006DF58A
S113181E7A00000092DB01006DF00FE05E0026D629
S113182E0B970B9701006DF67A00000092DB010017
S106183E6DF05EE9
S11318410018EA0B970B9701006DF55E001AEE0B7A
S1131851974040403E01006DF57A00000092DB01A4
S1131861006DF00FE05E0026D60B970B9701006D1C
S1131871F67A00000092DB01006DF05E0018EA0BBE
S1131881970B970FE05E00275E0B500D010FE05E93
S10C18910025525E002D54547031
S10E92D60C000A00002573000A0C00C6
S113189A188838BA38B8F85038B919000B50792073
S11318AA01184DF8F83038BA28BCF88038BC54709F
S11318BA0C8820BC737047FA38BBE07F30BC547085
S11318CA20BC736047FA29BDE0BF30BC0C98547042
S11318DA7EBC736047067900000154701900547086
S11318EA5E002D767A0600FFF1D47A050000000423
S11318FA7A00000000180AF00AD07308470E7A002B
S113190A000000180AF00AD00B70400A7A0000009F
S113191A00180AF00AD00F85189968E901006DF0DA
S113192A01006F71001C0FE05E00277A0B971955AF
S113193A0D5017F00AE0680947220D5017F00AE024
S113194A6808A80A4606F80D5C00FF640D5017F0F4
S113195A0AE068085C00FF580B5540D45E002D541A
S105196A5470B4
S113196C1911400C19880B58792806824DF80B5124
S113197C1D014DF0547001006DF50D090D8D28D628
S113198C17500D010D99470C0D19796900607949B0
S113199C001040060D19796900600DD50D90148D5A
S11319AC0CD8C88038D63DD639D67900000455B04A
S11319BC01006D7554706DF56DF40D090D8528D608
S11319CC17500D010D99470C0D1979690060794970
S11319DC001040060D19796900600D54119411948F
S11319EC11941194EC0F0D90148CED0F148D0CC8F5

S11319FCC88038D63CD60CD8C88038D63DD639D614
S1131A0C790000045C00FF586D746D75547001000F
S1131A1C6DF619EE7900000F5C00FF441966790826
S1131A2C00030DE05C00FF4E0B56792600034DEED0
S1131A3C790800020DE05C00FF3C19667908002868
S1131A4C0D605C00FF70790800100D605C00FF6690
S1091A5C7908000E0D6085
S1131A625C00FF5C790800060D605C00FF52790898
S1131A7200010D605C00FF48790800020D605C0004
S1131A82FF3E01006D7654707908000119005C0075
S1131A92FF2E7908000219005A0019C26DF60C8E46
S1131AA2AE0C460455E2401CAE0A460455DA401415
S1131AB2AE0D460455D2400C17D60D6879000001CD
S1131AC25C00FEFC6D76547001006DF60D0E0D8602
S1131AD20D6079080040528009E0791000800D08FA
S1131AE219005C00FEDA01006D7654705E002D76FB
S1131AF2790C000119447A0600FFF2247A050000EA
S1131B0200047A00000000180AF00AD07308470E96
S1131B127A00000000180AF00AD00B70400A7A001B
S1131B22000000180AF00AD00F85189968E901002D
S1131B326DF001006F71001C0FE05E00277A0B97B6
S1131B4219550D5017F00AE0680947560D5017F062
S1091B520AE06808A80A7E
S1131B58460A0DC80D405C00FF68403C0D5017F065
S1131B680AE06808A80D460C790800020D405C00DD
S1131B78FE4840240D5017F00AE06808A80C4606F2
S1131B885C00FEFE40120D5017F00AE0680817D0FB
S1131B980D080DC05C00FE220B5540A05E002D54BD
S1051BA8547074
S1131BAA1911400C19880B58792806824DF80B51E4
S1131BBA1D014DF054705C0009145C0000B4550417
S1131BCA5A001CA401006DF618886AA800FFF327BF
S1131BDA6AA800FFF32918886AA800FFF3286AA8ED
S1131BEA00FFF2657908008079000045C000808A8
S1131BFA7908001079000205C0007FC7906000FC1
S1131C0A790800100D605C0007EE0D687900000B7F
S1131C1A5C0007E40D687900000D5C0007DA7908B7
S1131C2A0050790000095C0007CE790800D67900D4
S1131C3A00075C0007C219660D605C000710790E85
S1131C4A00010DE05C0006E60DE05C0007000D688C
S1131C5A7900002B5C0007A00D687900002F5C0057
S1131C6A07960DE8790000275C00078C01006D7662
S1131C7A54707908000519005C00077C7900006438
S1131C8A5C00FF1C790800C419005C00076A790824
S1091C9A000379000001C4
S1131CA05A00240279080002790000055C000752FB
S1131CB0790800CC19005A0024025E002D767A04BC
S1131CC0000093947A000000936301006DF05E00BE
S1131CD018EA0B9719EE19660DE5790D001052D528
S1131CE00D6009505C00070C17506DF001006DF496
S1131CF05E0018EA0B970B870B56792600104DE010
S1131D007A000000939A01006DF05E0018EA0B97C9
S1131D100B5E792E00044DBE5E002D5454700100FD
S1071D206DF67A06D9
S1131D2400FFF264790000065C0006C468E87C6086
S1131D34734047367900000E5C0006B40C8E73487A
S1131D4447045C0002C8735E47085C0002C25A0081

S1131D541E12730E47085C0002B85A001E12731E4B
S1131D6447045C0002AE5A001E127C607360472273
S1131D747900000C5C0006780C8E730847085C003D
S1131D8400925A001E12731E587000825C0001C632
S1131D94407C7C607320471E7900000A5C00065077
S1131DA40C8E730847065C00020A4062731E475E8A
S1131DB45C00023E40587C6073104752790000086F
S1131DC45C00062C0C8E7368470A5C00FDFC5C0007
S1131DD4FECE403A734E471A790800C079000009D1
S1131DE45C00061A79080003790000055C00060EFE
S1131DF4401C737E4718790800050790000095C0081
S1131E0405FC79080002790000055C0005F0010077
S1081E146D7654705EC1
S1131E19002D7619DD790000265C0005CE0C8E7342
S1131E2968587001047A0600FFF267790000086DAB
S1131E39F00FE179080026790000255C0005CC0B39
S1131E49870DD05C0004E80DD05C0005027905001C
S1131E59200D505C0004C06868E8605860009C6EFF
S1131E69680001473CA801471EA8034772A8054714
S1131E793EA8064726A8084716A809475CA80A47A3
S1131E8926A80B4760406C5C0001865A001F206A34
S1131E992800FFF26617500D08400E5C000224402B
S1131EA9765C00035440700DD879000021402479F1
S1131EB90800400D505C0005406E6E0002CE806A3A
S1131EC9AE00FFF2656A2800FFF26517500D087925
S1131ED90000045C000522403E5C00038E40385C30
S1131EE900018840326E680002472C0D505C0004E3
S1131EF90E40240D505C000406401C6868E860A885
S1081F092046080D5005
S1131F0E5C0003F6400C686EEE60AE400D505C0054
S1131F1E03E87A0000FFF3287D0070000DD05C000B
S1131F2E045440226A2800FFF329470818886AA838
S1131F3E00FFF3290DD05C00041079080000179002D
S10A1F4E00275C0004AE5EF6
S1131F55002D5454705E002D767900002E5C00042C
S1131F658E0C8EE8C017504626790000406DF07A36
S1131F750500FFF2A70FD17908002E7900002D5C2B
S1131F8500048C0B870D060D010FD05C00056279EB
S1131F950000015C00039C790800017900002F5CB7
S1131FA500045A5E002D545470790000365C000419
S1131FB53E54706DF6790000225C0004320C8E737A
S1131FC568472A7358472619005C0003866A280068
S1131FD5FFF329470C5C0001C819005C0003A0400E
S1131FE50C79080001790000275C0004106D765414
S1131FF5706DF67900002A5C0003F40C8E73684754
S1132005087358470 45C0004826D76547054705409
S1132015705470547 05E002D767A0600FFF267687F
S113202568E803A80246426E680003E80747186E8E
S1132035690003E9071751177178106A28000092A0
S1082045E217505C00EE
S113204A02D46E680003E8071750790100011A08E1
S113205A4B04101140F817117A0000FFF327680A9E
S113206A169A688A5E002D5454705E002D767A069D
S113207A00FFF2676868E803A80246406E68000337
S113208AE80747186E690003E907175117717810B3
S113209A6A28000092E217505C0002626E6800032D
S11320AAE8071750790100011A084B04101140F888

S11320BA7A0000FFF327680A149A688A5E002D548F
S11120CA54707A0000FFF3287D0072006A282C
S11320D800FFF26AA801470AA8024716A803472285
S11320E854706A29000092EA17517A00000092EAB4
S11320F840686A29000092FE17517A00000092FC9A
S113210840586A2800FFF269470EA801471AA80237
S11321184726A803473254706A290000932317D12E
S11321287A000000932340326A290000932717D1CD
S11321387A000000932740226A290000933917D1B7
S10D21487A000000933940126A295F
S11321520000934117D17A00000093415502547055
S11321625E002D760F840D187A0600FFF32A6A2883
S113217200FFF26E17500C8018886A2900FFF26D77
S11321821751091069E01D804F0269E801006BA431
S113219200FFF32CF8016AA800FFF32955065E003D
S11321A22D5454705E002D767A0600FFF32A7905CA
S11321B200086960474469607920000084E02696536
S11321C27A0400FFF32C6DF50100694179080022BE
S11321D2790000215C0002780B870D056961190102
S11321E269E117F0010069410A81010069C169606F
S11321F2460818886AA800FFF3295E002D5454701C
S10422025E7A
S1132203002D766A2800FFF267E80317500D0519BE
S1132213EE790600210D0047067925000146100FCC
S1132223E05C0001DA0DE80D605C0001D2403C790B
S1132233250002462E6A2800FFF26AE80717500DAD
S113224305790100011A084B04101140F86A2800AC
S1132253FFF32717506601470679080000140060D6F
S1132263E840020DE80D605C0001945E002D5454B8
S1132273706DF66A2800FFF2696AA800FFF26646EA
S11322833C19660D68790000285C0001720D6879BA
S113229300002C5C0001680D68790000305C0001CC
S11322A35E0D68790000345C0001540D6879000009
S11322B3385C00014A0D687900003C404018886A85
S11322C3A800FFF327790600010D605C00008679FF
S11322D3080011790000285C000124790800037 9C0
S11322E300002B5C0001180D60554A790800127930
S10822F300002C5C005B
S11322F801080D687900002F5C0000FE6D765470AC
S11323086DF60D065C0000E4C88017500D080D60DB
S11323185C0000E66D7654706DF60D065C0000CC2B
S1132328E87F17500D080D605C0000CE6D76547081
S113233801006DF60D0617F6103679080008786067
S11323486B2000FFEF185C0000B001006D7654703D
S11323585E002D760D0617F610367A0500FFEF108E
S11323680AE569505C00008417500D06703E0D683D
S113237869505C0000845E002D5454705E002D7615
S11323881B971B977A0500FFF3280D0EFE017900B2
S113239800010DE11A094B04101040F86859175150
S11323A866104704702E4002722E701E17560DE0F9
S11323B817F0103001006FF000040D6801006F7111
S11323C8000478106B2000FFEF10010069F1552A13
S11323D8790000011B5E4B04101040F868591589F9
S10923E868D90B970B9767
S10423EE5E8D
S11323EF002D5454706AA8004000036A280040006F
S11323FF0154700D016AA9004000030D806AA80003

S113240F40000154705E002D767A03004000030FE5
S113241F840F9568BC19EE196640180DC055C6E8B0
S113242F0F471868BC6A280040000168D80B750B6A
S113243F5E0B566F7000181D064DE00DE05E002D0C
S113244F5454705E002D760D0C0D830F956F740031
S113245F1819EE1966401E0D30558AE81F471A0DDD
S113246FC06AA80040000368586AA8004000010B27
S113247F750B5E0B561D464DDE0DE05E002D54545D
S113248F7001006DF619EE7A0000FFF2E779010093
S113249F405C0001140D06790000015C00FEAA0DDB
S11324AF6647166DF67A0100FFF2E77908002A797D
S11324BF000029558E0B870D0E790000015C00FE7D
S11324CFB40DE001006D76547019006BA000FFF39B
S10624DF326BA0BA
S11324E200FFF33019006BA000FFF3366BA000FF6F
S11324F2F33454705E002D767A0400FFF3327A05CA
S113250200FFF3300F830D1E7FF57250199940308F
S1132512694079200100 4C2C695017F06839780022
S11325226AA900FFF33869500B5017F079010100D3
S113253201D0531069D869400B5069C00B730B5912
S10E25421DE94DCC7FF570500D905E3D
S113254D002D5454705E002D76790E01007A04002F
S113255DFFF3367A0500FFF3340F830D127FF57207
S113256D501999403469407920010 04C3269501754
S113257DF001D053E00D8017F0683978006AA90097
S113258DFFF43869500B5017F001D053E069D86947
S113259D400B5069C00B730B590D201D094DC67FA0
S11325ADF570500D905E002D5454705E002D767AAB
S11325BD0500FFF3360F840D1E7FF572501966402B
S11325CD381DE64C386B2000FFF33469511910792F
S11325DD10010017F07901010001D053100D801780
S11325EDF00D6117F10AC178006A2800FFF438680D
S11325FD9869501B5069D00B5669504EC47FF570C6
S113260D500D605E002D5454705E002D767A0500DA
S113261DFFF3320F840D1E7FF57250196640381D7E
S113262DE64C386B2000FFF3306951191079100116
S108263D0017F0790114
S1132642010001D053100D8017F00D6117F10AC17B
S113265278006A2800FFF338689869501B5069D0E4
S11026620B5669504EC47FF570500D605E3D
S113266F002D5454706B2000FFF33217F0790101E2
S113267F0001D053100D8054706B2000FFF33617F9
S110268FF07901010001D053100D8054704B
S11392E220282C3034383C201201000100000008F1
S11392F2FEFF10000100010102010902270001012 2
S113930200C0640904000003000000030705810292
S11393124000FF070502024000FF070583024000E9
S1139322FF040309041203550053004200200054B2
S11393320045005300540008033100 2E0030002280
S11393420355005300420020005400450053 0054CB
S11393520020005000 52004F004700520041004DD0
S10493620007
S11380000023002B0033003B0027002F0037003FE5
S11393633030203031203032203033203034203 03D
S11393733520303620303720303820303920304103
S11393832030422030432030442030452030460AE9
S10C939300253032582000 0A00C5

S113269C01006DF67A0600FFF574010069607A019A
S11326AC41C64E6D5E004C2E7A100000303901008D
S11326BC69E06960198817707A01000080005E0078
S10D26CC4C080D1001006D765470E8
S11326D65E002D760F857A06000000047A0000005E
S11326E600180AF00AE07308470E7A000000001883
S11326F60AF00AE00B70400A7A00000000180AF09C
S11327060AE00F8601006DF001006F70001C0100E6
S11327166DF001006DF51A91790000015E002D9EA2
S11127267A170000000C0D065E002D5454704F
S11327345E002D760F860F9540040B760B75686942
S113274468581C8946040C9946F068681750685DFC
S10D2754175519505E002D54547000
S113275E5E002D760F851AE640020B760FD00B75B1
S10F276E680946F60FE05E002D5454701D
S113277A5E002D760F850F9601006F710018010018
S113278A6DF101006DF601006DF51A9179000001F2
S113279A5E002D9E7A170000000C5E002D545470C3
S11327AA0FC446120FD5460E792107FF46120FB3FF
S11327BA4604156E4A0A1AC47A0500000008186608
S11327CA5A002A3E0D9946120FC4461A0FD54616C9
S11327DA0FB3461E16E6580002660FA246360FB31B
S11327EA46325800025A0FA246140FB34610580035
S11327FA024E0CE60D190FA40FB55A002A4A1035DA
S113280A1234792C00104C0C1B5910351234792CC4
S113281A00104DF410331232792A00104C0C1B515C
S113282A10331232792A00104DF4580000CA1AC420
S113283A1AD5186619995800020601006DF20100AB
S113284A6DF301006F23000401006DF30100692396
S113285A795B800001006DF30FF2550E0B970B970E
S113286A01006D7301006D7254706DF601006DF510
S113287A01006DF401006DF301006DF201006DF1C9
S113288A01006DF00100691401006925103410354 7
S109289A1FD44218451093
S11328A001006F14000401006F2500041FD44406C7
S11328B00F940FA10FC26816682E0FA001006914B0
S11328C001006F150004010069020100 6F03000499
S11328D0796C000F796A000F69191119111911190F
S11328E01119796907FF6901111111111111111E1
S11328F0796107FF792907FF5870FEAE0D115870F3
S1132900FECC10351234103512341035123410331 6
S113291012321033123210331232794C0080794A5A
S113292000800D901910474C792000364E3E7920D7
S113293000204D0E793000200FB34702700A0FA319
S11329401AA2792000104D14793000100D380DB300
S11329500D2B0DA219AA0D884702700B0C884F147A
S1132960113213334402700B1B5040F01AA27A0346
S1132970000000010C6815E84B2A0AB544020B74E9
S11329800AA4792C010058500080113413354402F5
S1072990700D0B595F
S1132994792907FF5860006E1AC41AD5580000A697
S11329A41FA446061FB55870FE8A1AB544087A3424
S11329B40000000145061AA4450440121AA4173561
S11329C417347A150000000144020B740CE60FC49B
S11329D446080FD41AD57919FFE00DCC460C0D4CDB
S11329E40DD40D5D19557919FFF0792C01004508B3
S11329F4113413350B5940F2792C00804C081035EF

S1132A0412341B5940F20D994E10113413350D999C
S1132A144A08113413350B5940F4732D471C0CD851
S1132A24E80B47167A150000000844020B74792C4E
S1132A3401004D06113413350B5911341335113478
S1132A44133511341335796C000F1019101910193B
S1132A541019649C101C1006131C01006D700100F6
S1132A64698401006F85000401006D7101006D72BA
S1132A7401006D7301006D7401006D756D76547002
S1132A8401F06401586000807924O7FF5860006CEA
S1132A945800007401F064235860006C40520F85A1
S1132AA401F06415460E0D44464601F06423464086
S1132AB458000054103112307928001O4C0C1B5C60
S1132AC41031123079280010 4DF4580000BA0FA5C4
S1132AD401F06435472610331232792A00104C0C66
S1132AE41B5410331232792A00104DF45800009EFF
S1132AF41AC4100E131C1AD55800018E790E07FF41
S1132B041AC41AD5580001621AC418EE790E07FFC5
S1132B1419DD790500085800015001006DF6010024
S1132B246DF501006DF401006DF301006DF2010018
S1132B346DF101006DF0681E6826156E691C111C89
S1132B44111C111C111C796C07FF69241114111435
S1132B5411141114796407FF01006D100100691148
S1132B647968000F01006F2300040100692279 6A68
S1092B74000F792C07FF9E
S1132B7A5870FF06792407FF5870FF120DCC58705E
S1132B8AFF160D445870FF40194C791C03FF0DCEF4
S1132B9A792EFFCC58D0FF5279480010794A001099
S1132BAA1AC47A0500000100103312321031123 0B0
S1132BBA441C0AB144087A100000000145040AA023
S1132BCA40040AA004011235123444E0401C1AB12D
S1132BDA44087A300000000145041AA040041AA0F0
S1132BEA04011235DD01123444C2730D46080AB1D9
S1132BFA44020B700AA001F064014702700D792C9C
S1132C0A00804C06103512341B5E792E07FF58C01C
S1132C1AFEE40DEE4E2E113413354402700D0DEE03
S1132C2A4A040B5E40F0732D47180CD8E80B471281
S1132C3A7A1500000008440A0B74792C00804D02AF
S1132C4A0B5E4020732D471C0CD8E80B47167A15E8
S1132C5A0000000844020B74792C01004D0611345C
S1092C6A13350B5E11346B
S1132C701335113413351134133 5796C000F101ECD
S1132C80101E101E101E64EC101C100E131C0100ED
S1132C906D700100698401006F85000401006D718E
S1132CA001006D7201006D7301006D7401006D759B
S1092CB001006D76547073
S10F2CB65E004B4CA8014702190054704B
S1132CC25E004B4C0C884E0419005470F80154708A
S1132CD25E004B4C0C884604F80154701900547082
S1132CE201006DF201006DF11AA20F9147224A060B
S1132CF27902080017B17912041F1B52103144FAEA
S1132D021031121210311212103112121031121 22A
S1132D12698201006F8100026F8A000601006D71F2
S1092D2201006D72547004
S1132D2801006DF201006902010069920100 6F025E
S1112D38000401006F92000401006D725470DC
S1112D465E004B4C1A084704790000015470DC
S1132D5401006F76001401006D7201006FF2001020

S1132D6401006D7201006D7301006D7401006D75D6
S1052D74547096
S1132D7601006DF501006DF401006DF301006DF2C4
S1132D8601006F72001001006DF201006FF600146E
S10B2D9601006F720004547088
S1132D9E5E002D767A37000000B27A0300FFF54C01
S1132DAE7A0400FFF53C0D0201006FF100AA010049
S1132DBE6F7600CE01006F7500D20D00466C0100D8
S1132DCE6F7000CA460E790004526BA000FFF578AF
S1132DDE5A002E7001006F7000CA6E080010E807CB
S1132DEE17D0460C790005146BA000FFF5784072DE
S1132DFE01006F7000CA6E08001073284610730826
S1132E0E470C790105166BA100FFF57840540100BC
S1132E1E6F7000CA6E0800107368464601006F702B
S1132E2E00AA01006BA000FFF5700D2017F0010042
S1132E3E6BA000FFF53801006F7000CA01006BA094
S1132E4E00FFF53E1A8001006BA000FFF5421888C3
S1132E5E68C8F8306EF800885C001CBA0D00587014
S1132E6E0A1C7900FFFF5A0038AE6868A82558601F
S1132E7E09CC0B76188868C80FF001006BA000FF11
S1072E8EF5500FF0F9
S1132E9201006BA000FFF5541A8001006BA000FF34
S1092EA2F55801006BA0CE
S1132EA800FFF55C1A8001006BA000FFF5600100CC
S1132EB86BA000FFF5641A8001006BA000FFF568A2
S1132EC840306868A8204718A8234720A82B470A3A
S1132ED8A82D461C7D40701040167D407030401070
S1132EE87C407330460A7D40704040047D4070202A
S1132EF80B766868A82D47CAA82B47C6A82047C2DF
S1132F08A82347BEF9206AA900FFF5466869A930D6
S1132F18460AF9306AA900FFF5460B761A800100C4
S1132F286BA000FFF5486868A82A586000800FD096
S1132F380BF001006FF000A07308470A01006F70DF
S1132F4800A00B70400601006F7000A00F851BF0F6
S1132F5801006FF000967308470A01006F7000962E
S1132F681B70400601006F700096690017F001009E
S1132F786BA000FFF5484C167D40701001006B20D4
S1132F8800FFF54817B001006BA000FFF5480100EA
S1092F986B2000FFF54869
S1132F9E7A20000002004F0E7D407000700051864
S1132FAE6BA000FFF5780B76686817D0796000FF89
S1132FBE17F078006A28000093A4732858700086CF
S1132FCE1A8001006BA000FFF5484064686817D0B3
S1132FDE7930003017F001006FF000A07A01000085
S1102FEE02001A810F907A010000000A5EB4
S1132FFB004C0801006B2100FFF5481F904D240185
S113300B006B2000FFF5487A010000000A5E004CBC
S113301B2E01006F7100A00A9001006BA000FFF559
S113302B48400E7D407000790005186BA000FFF53A
S113303B780B76686817D0796000FF17F078006A11
S113304B28000093A4732846867A00FFFFFFFF0135
S113305B0069B06868A82E586001000B766868A8F1
S113306B2A466C0FD00BF001006FF00096730847E4
S113307B0A01006F7000960B70400601006F700021
S113308B960F851BF001006FF000A07308470A0130
S113309B006F7000A01B70400601006F7000A069E9
S11330AB0017F0010069B04C0A7A00FFFFFFFF0124

S11330BB0069B0010069307A20000002004F0E7DD9
S11330CB407000790005186BA000FFF5780B76684C
S11330DB6817D0796000FF17F078006A2800009317
S10830EBA473284776E1
S11330F01A80010069B04058686817D079300030F1
S113310017F001006FF000A07A01000002001A819D
S11331100F907A010000000A5E004C08010069313B
S11331201F904D1C010069307A010000000A5E0007
S11331304C2E01006F7100A00A90010069B0400E8F
S11331407D407000790005186BA000FFF5780B76C1
S1133150686817D0796000FF17F078006A280000CC
S113316093A4732846927A000000002001006BA00C
S113317000FFF56C6868A8684708A86C4704A84C6A
S11331804610686817D017F001006BA000FFF56CBC
S11331900B766868A825587004A4A84558700212D5
S11331A0A8475870020CA8584742A863587002E415
S11331B0A8644738A865587001F8A866587001F2EA
S11331C0A867587001ECA8694722A86E58700430AC
S11331D0A86F4718A87058700398A873587002F81E
S10931E0A8754708A8785A
S10D31E647045A00367A01006B20FB
S11331F000FFF56C7A200000006C46440FD00B9062
S113320001006FF000AA7308470A01006F7000AA5B
S11332100B70400601006F7000AA0F851B90010020
S11332206FF000967308470A01006F7000961B70D9
S1133230400601006F700096010069025A00338254
S113324001006B2000FFF56C7A20000000685860D5
S1133250009A6868A875470CA8584708A8784704D7
S1133260A86F46420FD00BF001006FF000AA73085D
S1133270470A01006F7000AA0B70400601006F70CF
S113328000AA0F851BF001006FF000967308470A30
S113329001006F7000961B70400601006F7000966E
S11332A06900177040400FD00BF001006FF00096DB
S11332B07308470A01006F7000960B704006010007
S11332C06F7000960F851BF001006FF000AA730862
S11332D0470A01006F7000AA1B70400601006F705F
S10932E000AA690017F0CB
S11332E60F825A0033826868A875470CA8584708A6
S11332F6A8784704A86F46420FD00BF001006FF081
S11333060096730847 0A01006F7000960B7040061B
S113331601006F7000960F851BF001006FF000AA85
S11333267308470A01006F7000AA1B70400601006C
S11333366F7000AA6900177040400FD00BF00100B0
S11333466FF000AA7308470A01006F7000AA0B709A
S1133356400601006F7000AA0F851BF001006FF095
S11333600967308470A01006F7000961B704006AB
S113337601006F700096690017F00F820100693033
S11333867A20FFFFFFFF460A7A00000000010100D2
S113339669B0686817D06DF07A01000000020AF17F
S11333A60FA05C00050E0B875A0035EE01006B205B
S11333B600FFF56C7A200000004C46420FD00B90BC
S11333C60B9001006FF000AA7308470A01006F70A3
S10933D600AA0B70400683
S11333DC01006F7000AA0F851B901B9001006FF00A
S11333EC00967308470A01006F7000961B70404AE1
S11333FC01006F70009640420FD00B900B900100B0
S113340C6FF000AA7308470A01006F7000AA0B70D3

S113341C400601006F7000AA0F851B901B900100E2
S113342C6FF000967308470A01006F7100961B71C9
S113343C400601006F7100960F907A010000008026
S113344C0AF15E002D28010069307A20FFFFFFFF8F
S113345C460A7A0000000006010069B0686917D1BA
S113346C01006F70008401006DF001006F70008427
S113347C01006DF07A00000000080AF05C0007867A
S113348C0B970B975A0035EE0FD00BF001006FF032
S113349C00AA7308470A01006F7000AA0B7040065C
S11334AC01006F7000AA0F851BF001006FF00096EE
S11334BC7308470A01006F7000961B7040060100E9
S10934CC6F7000966E080C
S11334D200015A0036400FD00B9001006FF000AA92
S11334E27308470A01006F7000AA0B7040060100BF
S11334F26F7000AA0F851B9001006FF0009673088E
S1133502470A01006F7000961B70400601006F703E
S1133512009601006900 0F825E00275E01006FF0D2
S113352200AA010069317A21FFFFFFFF4712010060
S113353269311F814C0A0100693001006FF000AA52
S11335420FF001006BA000FFF5541A8001006BA07D
S113355200FFF56001006B2000FFF54801006F7169
S110356200AA1A9001006BA000FFF5685A43
S113356F0036880FD00B9001006FF0009673084759
S113357F0A01006F7000960B70400601006F700018
S113358F960F851B9001006FF000AA7308470A017D
S113359F006F7000AA1B70400601006F7000AA0134
S11335AF0069007A6000FFFFFF01006FF0009601D2
S11335BF0069307A20FFFFFFFF460A7A0000000000
S11335CF01010069B0790000786DF07A0100000005
S11335DF020AF101006F7000985C0002CE0B870F97
S11335EFF00F825E00275E01006FF000AA5A0036CB
S11335FF880FD00B9001006FF000A07308470A01EA
S113360F006F7000A00B70400601006F7000A00FD9
S113361F851B900F81730847060F901B7040020F95
S113362F90010069006B2100FFF5446981404AF85E
S113363F2568F80FF00F827A000000000101006F78
S113364FF000AA0FF101006BA100FFF5541A9101CD
S106365F006BA159
S113366200FFF56001006B2100FFF5481A8101009C
S11336726BA100FFF568400E790005186BA000FFEF
S1133682F5787D4070006868A86E587001B86A28A2
S113369200FFF53C7308586001AC7C40731046365A
S11336A201006B2000FFF5684F2C40207A010000D7
S11336B200017A0000FFF5465C0013B87A0000FFB0
S11336C2F568010069011B710100698101006B202A
S11336D200FFF5684ED601006B2000FFF55C46241F
S11336E201006B2000FFF560461A01006B2000FF0A
S11336F2F564461001006F7100AA0FA05C001374F9
S11337025A00380C01006B2000FFF5500FA11A90EC
S113371201006FF0009C01006FF000A04F30010028
S11337226F71009C0FA05C00134A40227A000000D4
S113373200880AF07A01000000015C0013367A0067
S113374200FFF55C010069011B7101006981010041
S11337526B2000FFF55C4ED401006B2000FFF55493
S113376201006B2100FFF5501A9001006FF0009CDD
S113377201006F7100A00A8101006FF100A00F80A8
S11337824F3601006F71009C01006B2000FFF55062

S11337925C0012E040227A00000000880AF07A01FD
S11337A2000000015C0012CC7A0000FFF56001000A
S11337B269011B710100698101006B2000FFF56043
S10937C24ED401006F71FB
S11137C800AA01006F7000A01A8101006B209F
S11337D600FFF5545C00129840227A00000000882E
S11337E60AF07A01000000015C0012847A0000FFEF
S11337F6F564010069011B710100698101006B20F9
S113380600FFF5644ED47C407310473601006B20ED
S113381600FFF5684F2C40207A01000000017A0072
S113382600FFF5465C0012487A0000FFF5680100C8
S113383669011B710100698101006B2000FFF568B6
S11338464ED60B76404001006FF600A6400E0100EF
S11338566F7000A60B7001006FF000A601006F7079
S113386600A668086EF8009547086E780095A825A7
S113387646DC01006F7100A61AE10FE05C0011F04F
S113388601006F7600A6686847145C0012900D006D
S1133896460C6A2800FFF53C73085870F5D45C00A3
S11338A6125E6B2000FFF5447A17000000B25E003B
S11338B62D5454705E002D767A37000000247A0565
S10938C6000000040AF5F6
S11338CC01006FF0001801006FF1002001006FF18F
S11338DC001C18AA689A6F72003C0C22463CAA582A
S11338EC472CAA644712AA69470EAA6F4712AA75F6
S11338FC4706AA78471840227A000000000A4006BF
S113390C7A000000000801006FF00014400C7A00EC
S113391C0000001001006FF000146F70003C792060
S113392C00644706792000694612010 06F70001885
S113393C4C0A01006F74001817B4400601006F7431
S113394C00181AE61AB30FC4467401006B2000FF6B
S113395CF54C476AF83068D87A06000000014062DB
S113396C0FD00AE00F82010069F00FC001006F71E4
S113397C00145E0052320100697068890FA0680858
S113398CA8094E0C0FD00AE0680989306889401EDB
S113399C6F70003C79200078460A0FD00AE0680962
S11339AC895740080FD00AE06809893768890FC026
S10939BC01006F7100140D
S11139C25E0052320F840B760FC4469E6A28B5
S11339D000FFF53C732847626F70003C0C00465AA9
S11339E0A858471EA86F4706A8784716404C010001
S11339F06F70001847440FE00B760AD0F9306889DE
S1133A00403801006F700018473001006F700020CC
S1133A100B7001006FF000201B70F9306889010002
S1133A206F7000200B7001006FF000201B706E7927
S1133A30003D68897A03000000020FB00AE00F839B
S1133A4001006FF600146F70003C79200064470694
S1133A5079200069467201006B2000FFF54C46088F
S1133A6001006F700018476001006F7000184D422D
S1133A706A2800FFF53C733847160B7301006F701B
S1133A8000200B7001006FF000201B70F92B4036F3
S1133A906A2800FFF53C7348472E0B7301006F70D3
S1133AA000200B7001006FF000201B70F920688963
S1133AB040160B7301006F7000200B7001006FF054
S1093AC000201B70F92D2C
S1133AC668897A0600FFF55401006F70001C680CC4
S1133AD6AC2B4708AC2D4704AC20460E01006F7093
S1133AE6001C0B70010069E040466A2800FFF53CA4

S1133AF6732847326F70003C0C004634A858470AB7
S1133B06A86F4714A8784702402601006F70001C6F
S1133B160BF0010069E0401801006F70001C0B7088
S1133B26010069E0400A01006F70001C010069E0B2
S1133B367A0400FFF54C010069401FB04F140100E1
S1133B466946010069441AB401006BA400FFF560DD
S1133B56400C0FB61A8001006BA000FFF5607A04D3
S1133B6600FFF56801006F70001C680BAB2B47085C
S1133B76AB2D4704AB20463801006B2000FFF54808
S1133B861FE04F2C6A2800FFF546A83046220100A5
S10D3B966B2000FFF5481AE07A01E6
S1133BA000FFF560010069120A82010069921A8020
S1133BB0010069C0401E01006B2000FFF5481FE0B3
S1133BC04F0C01006B2000FFF5481AE040021A80F9
S1133BD0010069C001006F7600141B76401A0100D2
S1133BE06F7000200B7001006FF000201B700FE15D
S1133BF01B760AD1681968890FE64CE201006F70E1
S1133C000020189968897A17000000245E002D545B
S1133C1054705E002D767A37000000647A0300FF4B
S1133C20F53C7A04000000040AF47A0500FFF54C21
S1133C300F866FF1005001006FF6005CF83068C822
S1133C407A000000007C0AF07A010000939C5E0079
S1133C502D460D00587000B87A000000002F0AF0BE
S1133C6001006DF07A000000002E0AF001006DF0F3
S1133C7001006F70008801006DF001006F70008813
S1133C8001006DF07A01000000320AF17A000000B1
S1093C9000115E004C4E22
S1133C967A17000000100D024752188868E80100E1
S1133CA6695001006B2100FFF5481F904F06010084
S1133CB66950400801006B2000FFF54801006BA026
S1133CC600FFF5687A0600FFF5460D207920FFFF11
S1133CD6470C79200001460CF82B5A0049FEF82DB3
S1043CE65A80
S1133CE70049FEF82A68E85A004A007A00000000F3
S1133CF71101006DF00FC10B717A00000000260A55
S1133D07F05E0050340B97400CF8016EF8002F1843
S1133D17886EC800017A000000000101006FF000FF
S1133D2760400E01006F7000600B7001006FF000C0
S1133D376001006F7000600AC06808A83047E4019B
S1133D47006F7000607A20000000014F7A01006F56
S1133D577000600AC05E00275E0F8201006F70006B
S1133D67601B7001006F71002A0A8101006FF10067
S1133D772A7A000000000101006FF000584030016B
S1133D87006F7000600AC001006F7100580AC168B4
S1133D9708689801006F7000580B7001006FF000FE
S1133DA75801006F7000600B7001006FF000600135
S1133DB7006F7000580FA11F904FC401006F700070
S1133DC7580AC0189968890FC00B7001006FF0007B
S1083DD7445E00275EBD
S1133DDC01006FF000601A8001006FF000580100C1
S1133DEC6F7000607A01000000025E004C08010055
S1133DFC6FF0004001006F70004401006F710060B0
S1133E0C0A901B7001006FF0003C404E01006F7074
S1133E1C004401006F7100580A9001006FF0004CD0
S1133E2C68086EF8005701006F71003C01006F7257
S1133E3C00581AA101006FF1004801006F72004C89
S1133E4C681968A901006F710048689801006F71C7

S1133E5C00580B7101006FF1005801006F7000588E
S1133E6C01006F7100401F904DA201006F70006044
S1133E7C461AF8306EC800017A00000000010100F8
S1133E8C6FF000601A8001006FF0002A6F70005011
S1133E9C792000674706792000474 63C7D307050F7
S1133EAC01006F70002A01006F7100600A901B7093
S1133EBC7A20FFFFFFFC4D0C01006B2100FFF54C3A
S1093ECC1F904F0C6F7004
S1133ED200501BD06FF00050400879000066 6FF06D
S1133EE200506E78002FA80146246838E8186EF84F
S1133EF20057A8084706A810470A401A0FE00B7696
S1133F02F92B40100FE00B76F920688940080FE087
S1133F120B76F92D68896F7000507920006658601E
S1133F22077201006F70002A58D001861A80401A66
S1133F320FE00B7601006F7100580AC10B7168190B
S1133F42688901006F7000580B7001006FF0005810
S1133F5201006F7100601F904DD61A800F8201001D
S1133F626BA600FFF55401006F70002A01006BA0DD
S1133F7200FFF5607C307350460A01006B2000FF9E
S1133F82F54C4E067C307320470E0FE00B76F92E6C
S1133F9268890FA00B700F827C307350470C7C3002
S1133FA2735047247C307320471E01006BA600FF29
S1133FB2F5580100695001006BA000FFF564010090
S1093FC269500FA10A8102
S1133FC80F9201006F70002A0FA10A9001006F7110
S1133FD800600A9001006FF0006001006F71005CDF
S1133FE86819A92B4708A92D4704A920465201009F
S1133FF86B2000FFF54801006F7100601F904F4070
S11340086A2800FFF546A830463601006F70005C49
S10940180B7001006BA018
S113401E00FFF55001006B2000FFF54801006F71A2
S113402E00601A9001006BA000FFF55C1A8001007E
S113403E6BA000FFF5685A0049FC01006B2000FFDE
S113404EF54801006F7100601F904F4C01006B200B
S113405E00FFF54801006F7100601A9001006BA01C
S113406E00FFF5686E78002FA80146186E78002FB2
S113407EA801586009786E780057A8084706A8105B
S113408E5860096A7A0000FFF568010069011B7127
S113409E010069815A0049FC1A8001006BA000FFE0
S11340AEF5685A0049FC01006F70006001006F71E2
S11340BE002A0A8101006FF1002A010069520AA148
S11340CE01006FF100521F814C120F914D0E010032
S11340DE6F7100520B710FC05C00092201006F70EB
S11340EE002A58F0028A6848A83046147A00000065
S11340FE000101006FF0004C01006FF00052402CE4
S109410E1A8001006FF0AE
S1134114004C1A8001006FF0005201006F70002AF6
S11341240B7001006FF0002A01006F7000600B70C8
S113413401006FF000601A8001006FF00058403AEC
S11341440FE00B7601006F71004C0AC1681968898E
S113415401006F70002A1B7001006FF0002A010038
S11341646F7000580B7001006FF0005801006F70FE
S1134174004C0B7001006FF0004C01006F70002ABB
S11341844EBE0FE00B76F92E688940240FE00B76C0
S113419401006F71004C0B7101006FF1004C1B7136
S11341A40AC1681968 89010069501B70010069D04C
S11341B401006F7000580B7001006FF0005801008C

S11341C46F70004C01006F7100521A9001006F71FF
S11341D400601F904C0A01006B2000FFF54C4EACAD
S11341E47C307350470C7C307350472E7C307320E3
S11341F447280100695001006F7100580A810100CA
S10742046FF10058FB
S107420801006BA69D
S113420C00FFF5540100695001006BA000FFF5603D
S113421C40226E68FFFFA830461A40101B7601003F
S113422C6F7000581B7001006FF000586E68FFFF31
S113423CA83047E87C307320464A6E68FFFFA82EEF
S113424C464201006B2000FFF56047067C3073503B
S113425C47321B7601006F70005801006B2100FF81
S113426CF5601A901B7001006FF000581A80010062
S113427C6BA000FFF56001006F70005C01006BA088
S113428C00FFF55401006F70005C6808A82B470809
S113429CA82D4704A820465001006B2000FFF548C9
S11342AC01006F7100581F904F3E6A2800FFF546BE
S11342BCA830463401006F70005C0B7001006BA0DA
S11342CC00FFF55001006B2000FFF54801006F71F2
S11342DC00581A9001006BA000FFF55C1A800100D6
S11342EC6BA000FFF568406201006B2000FFF548EE
S10942FC01006F71005880
S10B43021F904F4601006B20E0
S113430A00FFF54801006F7100581A9001006BA075
S113431A00FFF5686E78002FA80146146E78002F07
S113432AA80146286E780057A8084704A810461C17
S113433A7A0000FFF568010069011B7101006981B8
S113434A400A1A8001006BA000FFF56801006B2088
S113435A00FFF55401006F71005C1F9058600692CC
S113436A01006B2000FFF55001006BA000FFF5541C
S113437A5A0049FC6848A83046147A000000000134
S113438A01006FF0004C01006FF00052402A1A80BE
S113439A01006FF0004C01006FF0005201006F70D2
S11343AA002A0B7001006FF0002A01006F70006091
S11343BA0B7001006FF0006001006F70002A4F1448
S11343CA7A000000000101006FF0004C0FE00B7649
S11343DA684940060FE00B76F93068897A000000D5
S11343EA000101006FF000580FE10B76FA2E689A6C
S10943FA01006F70005882
S11344000B7001006FF0005801006F70002A58C054
S1134410009E01006BA600FFF55401006F70002A97
S113442017B0010069511F814F2001006F70002AEE
S113443017B001006BA000FFF56001006F70002A48
S1134440001006F7100581A81401801006950010082
S11344506BA000FFF5600100695001006F71005807
S11344600A8101006FF1005801006F70002A0100FA
S113447069510A81010069D140360FE00B760100D2
S11344806F71004C0AC168196889010069501B707B
S1134490010069D001006F7000580B7001006FF0CC
S11344A0005801006F70004C0B7001006FF0004C5E
S11344B001006F70004C01006F7100521A900100EF
S11344C06F7100601F904C0A01006B2000FFF54CD8
S11344D04EA87C307350470C7C30735047347C308B
S11344E07320472E010069504F4A01006BA600FF5D
S10944F0F55801006950BC
S10744F601006BA0B3
S11344FA00FFF5640100695001006F7100580A81D9

S113450A01006FF1005840226E68FFFFA830461A77
S113451A40101B7601006F7000581B7001006FF08A
S113452A00586E68FFFFA83047E87C307320467056
S113453A6E68FFFFA82E466801006B2000FFF56036
S113454A460A01006B2000FFF56447067C3073506E
S113455A474E1B7601006F70005801006B2100FF64
S113456AF5601A9001006B2100FFF5641A901B7025
S113457A01006FF000581A8001006BA000FFF56478
S113458A01006BA000FFF56001006F70005C010081
S113459A6BA000FFF5541A8001006BA000FFF558C9
S11345AA01006F70005C6808A82B4708A82D470410
S11345BAA820465001006B2000FFF54801006F71E7
S11345CA00581F904F3E6A2800FFF546A83046342C
S11345DA01006F70005C0B7001006BA000FFF550C7
S10745EA01006B203E
S11345EE00FFF54801006F7100581A9001006BA08F
S11345FE00FFF55C1A8001006BA000FFF5684062B6
S113460E01006B2000FFF54801006F7100581F90E9
S113461E4F4601006B2000FFF54801006F710058F3
S113462E1A9001006BA000FFF5686E78002FA801A9
S113463E46146E78002FA80146286E780057A808F6
S113464E4704A810461C7A0000FFF56801006901B3
S113465E1B7101006981400A1A8001006BA000FFE3
S113466EF56801006B2000FFF55401006F71005CCB
S113467E1F90461001006B2000FFF55001006BA048
S113468E00FFF5545A0049FC010069500B700100FC
S113469E6F7100601F904C0C010069510BF10FC03C
S11346AE5C00035A6848A830460E7A0000000001E9
S11346BE01006FF0004840241A8001006FF000489B
S11346CE01006F70002A1B7001006FF0002A0100B9
S11346DE6F7000600B7001006FF0006001006F706F
S10946EE006001006F7182
S11346F400481A9001006F72002A0A8201006FF2C7
S1134704002A0FE00B760AC1681968897A00000051
S1134714000101006FF000580FE10B76FA2E689A3E
S113472401006F7000580B7001006FF00058010016
S11347346F700048402E0FE00B7601006F71004C40
S11347440AC168196889010069501B70010069D0A6
S113475401006F7000580B7001006FF000580100E6
S11347646F70004C0B7001006FF0004C01006F710F
S113477400601F904E0A01006B2000FFF54C4EB6FB
S11347847C307350470C7C307350472E7C3073203D
S1134794472801006BA600FFF5540100695001008E
S10547A46BA005
S11347A600FFF5600100695001006F7100580A812E
S11347B601006FF1005840226E68FFFFA830461AC9
S11347C640101B7601006F7000581B7001006FF0DC
S11347D600586E68FFFFA83047E87C307320464ACE
S11347E66E68FFFFA82E464201006B2000FFF560AE
S11347F647067C30735047321B7601006F700058B2
S113480601006B2100FFF5601A901B7001006FF029
S1134816005801006F70005C01006BA000FFF554A7
S11348261A8001006BA000FFF5606F7000507920BD
S1134836006546080FE00B76F96540060FE00B7638
S1134846F945688901006F7000580B7001006FF01D
S1134856005801006F70002A4D0A0FE00B76F92B02
S1134866688940160FE00B76F92D688901006F7091

S1134876002A17B001006FF0002A01006F7000587C
S11348860B7001006FF0005801006F70002A7A2048
S1094896000000644D0E5A
S113489C0BF601006F7000580B800B7040140FE087
S11348AC0B76F9306889F83068E801006F700058AE
S11348BC0BF001006FF000580FE00B7001006FF06C
S11348CC003040360FE01B76010069F001006F7079
S11348DC002A7A010000000A5E004C0889300100AE
S11348EC6970688901006F70002A7A010000000A60
S11348FC5E004C0801006FF0002A01006F70002A63
S113490C4EC201006F76003001006F70005C6808C6
S113491CA82B4708A82D4704A820465001006B205C
S113492C00FFF54801006F7100581F904F3E6A2835
S113493C00FFF546A830463401006F70005C0B7025
S113494C01006BA000FFF55001006B2000FFF54840
S113495C01006F7100581A9001006BA000FFF55C09
S113496C1A8001006BA000FFF568406201006B2008
S113497C00FFF54801006F7100581F904F4601006E
S105498C6B209B
S113498E00FFF54801006F7100581A9001006BA0EB
S113499E00FFF5686E78002FA80146146E78002F7D
S11349AEA80146286E780057A8084704A810461C8D
S11349BE7A0000FFF568010069011B71010069812E
S11349CE400A1A8001006BA000FFF56801006B20FE
S11349DE00FFF55401006F71005C1F90461001003B
S11349EE6B2000FFF55001006BA000FFF5541888F3
S11349FE68E87A17000000645E002D5454705E0060
S1134A0E2D760F850F960FD00AE06808A8344F5203
S1134A1E0FD00AE0F93068891B760FD00AE06809D7
S1134A2E8901688940200FD00AE06808A8394F141D
S1134A3E0FD10AE1681888F668986E18FFFF88018F
S1134A4E6E98FFFF1B760FE64EDC6E580001A839F9
S1134A5E4F106E58000188F66ED800016858880111
S1134A6E68D85E002D5454705E002D761B977A0322
S1094A7E00FFF53E7A067D
S1134A8400FFF542010069F00F9501006B2100FF5F
S1134A94F5387A21000000001462601006DF50F81E7
S1134AA4010069305E0051D80B97010069300AD0C8
S1134AB4010069B0010069600AD0010069E0403A6D
S1134AC419444030010069700B70010069F01B70D8
S10F4AD4680817D00100693101006B2253
S1134AE000FFF5705D207920FFFF47120100696028
S1134AF00B70010069E00B5417F41FD44DCA0B97D8
S1134B005E002D54547001006B2000FFF5387A20AD
S1134B1000000001460C01006B2000FFF53E1899D0
S1134B206889547001006B2000FFF5387A2000007B
S1134B3000014604190054700 1006B2000FFF53E8C
S10F4B406E090010E94017D10D105470ED
S10B939C0000000000000000C6
S1134B4C01006DF501006DF401006DF301006DF2D0
S1134B5C01006DF101006D120100691301006F0179
S1134B6C0004010069000D840D8C796C7FF0792CA5
S1134B7C7FF047540DAC796C7FF0792C7FF047565E
S1134B8C0D8C65AC4B5E1AB144087A300000000101
S1134B9C45141AA0451001F06410475A0C444B0AF3
S1134BAC79000002400C0C444BF6190040047900C8
S1134BBCFFFF01006D7101006D7201006D73010047

S1134BCC6D7401006D7554700F85796D000F01F0D4
S1134BDC641546DA409E0FA5796D000F01F064351C
S1134BEC46CC409C01F064207A607FFFFFFF46AC0B
S10F4BFC01F0643146A67900000140B6C8
S1134C086DF20D820C2A4A0217B00D994A04D2801C
S1134C1817B15E0052320C224A0217B00CAA4A029C
S1094C2817B16D72547018
S1134C2E01006DF301006DF20D8252120D935203CA
S1134C3E52100928093801006D7201006D7354700A
S11393A4202020202020202020606060606020202076
S11393B42020202020202020202020202020202020A6
S11393C448101010101010101010101010101010105E
S11393D484848484848484848484101010101010FE
S11393E41081818181818101010101010101010157
S11393F401010101010101010101010110101010100B
S113940410828282828282020202020202020227
S11394140202020202020202020202101010102CF
S113942400000000000000000000000000000035
S113943400000000000000000000000000000025
S113944400000000000000000000000000000015
S113945400000000000000000000000000000005
S113946400000000000000000000000000000F5
S113947400000000000000000000000000000E5
S113948400000000000000000000000000000D5
S10994940000000000CF
S10D949A0000000000000000C5
S1134C4E5E002D767A370000003C7A040000000CDB
S1134C5E0AF47A05000000140AF501006FF000381B
S1134C6E01006FF1003401006F73005C7A020000E3
S1134C7E000801006FF2002C19226FF2002A7A0647
S1134C8E000000540AF6686AEA7F46620FE00B7072
S1134C9E7A01000000075E00526E0D0047501A8025
S1134CAE401A01006F70003401006F7100380A90D2
S1134CBE1899688901006F7000380B7001006FF04E
S1134CCE00387A20000000084DD81A80010069B020
S1134CDE7C607370470A01006F700060F9FF400833
S1134CEE01006F700060F90168895A0050267A003E
S1134CFE0000001C0AF001006DF07A01000000288C
S1134D0E0AF101006F70005C01006DF001006F701D
S1134D1E005C01006DF001006F70006C5E0055FECB
S1134D2E7A170000000C01006F7000606808A8FF7E
S1074D3E460C7A00A2
S1134D42000000540AF07D0072706F700024792015
S1134D5207FF464A6E680001A8F0463A0FE00BF0DF
S1134D627A01000000065E00526E0D004728010022
S1134D726F7000606808A8014608790000015A00B4
S1134D82502801006F7000606808A8FF4610790080
S1134D92FFFF5A005028790000025A0050286F7012
S1134DA20024793003FF6FF000244D1E01006F7061
S1134DB2005801006DF001006F70005801006DF0A2
S1134DC25E00538E0B970B97401E01006F700058C5
S1134DD201006DF001006F70005801006DF05E007C
S1134DE2538E0B970B971B500D0618886EF80023F2
S1134DF20B560D6017F001006F7100381A90010015
S1134E0269B0010069F10FD1010069705E0054AA13
S1134E1201006F70003801006F71002C5E004C08B6
S1134E2201006FF000301AE67A000000000801006A

S1094E326F7100301A90BD
S1134E380F82401401006F7000300AE00AD00FD1CE
S1134E480AE1680868980B760FA01F864DE6400AAA
S1134E580FD00AE0189968890B767A2600000008B3
S1134E684DEE6F7000326F78002E52806F71003AEA
S1134E7819016DF17A01000000080FD05E00543269
S1134E880B8701006F7600381B760FC10FE05E00B9
S1134E9854AA0FE001006F71002C5E004C0801005A
S1134EA86FF000301AE67A000000000801006F7105
S1134EB800301A900F82401401006F7000300AE02E
S1134EC80AC00FC10AE1680868980B760FA01F860D
S1134ED84DE6400A0FC00AE0189968890B767A26CE
S1134EE8000000084DEE6F7000326F78002E52807C
S1134EF86F71003A19011B516DF17A010000000826
S1134F080FC05E0054320B8701006F70003401003C
S1134F186DF00100693101006DF17A010000002490
S1094F280AF16F70002C7A
S1134F2E5E0052960B970B977A06000000040AF662
S1134F3E7A000000000801006DF001006F71003867
S1134F4E0FE05E0051D80B977A00000000080100B5
S10A4F5E6DF00FD10FE05EBF
S1134F650056A20B970D004F12790000016FF00058
S1134F752A010069300B705A00501E7A00000000A8
S1134F850801006DF001006F7100380FE05E0051FC
S1134F95D80B977A000000000801006DF00FD10FC0
S1134FA5E05E0056A20B970D004632010069300BF7
S1134FB570010069B001006F70003401006DF001EC
S1134FC500693301006DF37A01000000240AF16FD3
S1134FD570002C5E0052960B970B9740447A0000A5
S1134FE500000801006DF001006F7100380FE05EED
S1134FF50051D80B977A000000000801006DF00FEF
S1135005C10FE05E0056A20B970D004C146F7000A4
S11350152A460E010069301B70010069B05A004F22
S11250251019007A170000003C5E002D545470E0
S11350345E002D767A370000002C0FF30F860100F3
S11350446FF100107A000000000101006FF00018F6
S113505401006F7500440A95188868D87A00000027
S1135064000801006DF00FE17A000000000C0AF063
S11350745E0051D80B9701006F7000445A0051C46D
S11350840FA00B707A01000000055E004C080B7042
S113509410307A06000000081A867A00000000081F
S11350A41AE001006DF00FA11031103110317A11A3
S11350B4000094A40AE10FB00AE05E0051D80B97F4
S11350C47A000000000801006FF000281A80010034
S11350D46FF0002001006FF0001C7A040000000848
S11350E41AE401006FF400145A00518601006F7032
S11350F4001401006DF00FB10AE17A000000000C06
S11351040AF00AE05E0056A20B970D004D3C010025
S11351146DF40FB00AE001006DF07A010000001095
S10951240AF10AE11A8002
S113512A5E0055520B970B9717F001006FF00018AA
S113513A01006F70002801006F7100200A810100CD
S113514A6FF1002001006F7000287A01000000024D
S113515A5E004C0801006FF0002847307900000117
S113516A6DF00FB00AE00FC15E0054D80B8701003F
S113517A6F70001C0B7001006FF0001C01006F7050
S113518A001C7A200000000458D0FF5A01006F70F7

S113519A0018461C6E78002388306CD84004F83017
S11351AA68D81B7501006F7000101F8544F0401208
S11351BA6E78002388306CD80FA01B700F8258C0FA
S11151CAFEB87A170000002C5E002D545470BE
S11394A4000000000000000800000000000000505D
S11394B4000000000000032000000000000001F4023
S11394C400000000000013880000000000000C35009B
S11394D400000000007A12000000000004C4B4007D
S11394E4000000002FAF080000000001DCD650008C
S11394F400000012A05F2000000000BA43B7400040
S113950400000746A5288000000048C273950000A8
S11395140002D79883D20000001C6BF526340000A8
S10B9524011C37937E080000CF
S11351D85E002D760F850F9401006F7600181FC5AA
S11351E847401FC544200FD30FC21AC440120FB043
S11351F80B730FA10B710F921B71681968890B74DC
S11352081FE445EA401C0FD30AE30AE40FC21AC499
S1135218400C0FA01B700F8268086CB80B741FE456
S10D522845F00FD05E002D545470C2
S113523201006DF20D9946100D82177253120DA8DB
S113524253100D810D28401E0F920D81177179089D
S1135252001012101231 1AA144020AA11B5846F27D
S10F526212101710177001006D725470C9
S113526E5E002D760F840F951AE6400E0FC00AE0EE
S113527E680947041900400A0B761FD64DEE7900D4
S10B528E00015E002D54547071
S11352965E002D767A370000000C7A0500000001C7
S11352A60AF50D0C0F967904000801006DF57A01D5
S11352B60000000E0AF101006F70002817B05E00AF
S11352C657680B9701006DF66DFC7A00000000101D
S11352D60AF00FD15E0056E00B970B87790C003F5F
S11352E66F70000A190C0DC017F001D053400D0E54
S11352F67900004219C017F001006DF07A01000031
S113530600090FD05E0059B40B97790600084014C4
S11353160D6019E017F00AD00D6117F10AD168087C
S113532668981B561DE64CE8400C0D6017F00AD032
S1135336189968891B560D664CF00DE05240190CFE
S11353466DFC7A01000000090FD05E0054D80B876C
S11353567900003F6FF0000A7A01000000400FD089
S11353665E0058CA7A000000000801006DF00FD1F4
S113537601006F70002C5E0051D80B977A1700005E
S1095386000C5E002D5433
S105538C547058
S113538E01006DF27A370000001A7A00000000184F
S113539E0AF001006F71002601006DF101006F71BB
S11353AE002601006DF17A01000000080AF10100E8
S11353BE6DF15E005A507A170000000C6F710018E1
S11353CE0FF05E005D4A0F817A020000952C7A0081
S11353DE000000080AF05E005E367A00000000103E
S11353EE0AF001006F71000C01006DF101006F7185
S11353FE000C01006DF17A01000000080AF10100B2
S113540E6DF15E005B567A170000000C7A00000007
S113541E00100AF05E005CD07A170000001A01003B
S107542E6D725470D4
S10B952C3FD34395810624DDC2
S11354325E002D761B971B87010069F00F946F792D
S1135442001E790D0008199D4754790100FF0DD202

S11354521A0A4B04101140F80C9B18DD1B740FC67B
S1135462402801006974 0AE468450CB816850FC028
S11354720D915E005C8C684814D868C80DD01A0878
S11354824B04110540F80C5D1B760FE64CD40CDD82
S1135492470679010001 4002191117F10F900B879A
S10B54A20B975E002D545470BA
S11354AA5E002D760F860F957A0000000008010032
S11354BA6DF01036103610367A01000095340AE181
S11154CA0FD05E0051D80B975E002D54547026
S1139534000000000000000100000000000000051E
S1139544000000000000001900000000000000007D7E
S1139554000000000000027100000000000000C3550
S113956400000000000003D09000000000001312D4F
S113957400000000000 5F5E100000000001DCD65BA
S1139584000000000 9502F90000000002E90EDD6E
S113959400000000 0E8D4A510000000048C273957C
S11395A4000000016BCC41E9000000071AFD498D5E
S11395B40000002386F26FC1000000B1A2BC2EC5D7
S11395C4000003782DACE9D900001158E460913D03
S11395D4000056BC75E2D6310001B1AE4D6E2EF5D6
S11395E4000878678326EAC9002A5A058FC295EDD5
S11395F400D3C21BCECCEDA10422CA8B0A00A4253E
S113960414ADF4B7320334B96765C793FA10079DF1
S11354D85E002D767A370000000A0F83010069F118
S11354E86F790022790C0008199C4752FAFF0DC006
S11354F81A084B04110A40F80CA218CC1AE64026E5
S11355080FB50AE568540C2816840FD00D915E0078
S11355185CAE685814C868D80DC01A084B04100448
S113552840F80C4C0B76010069701F864DD20CCCE9
S1135538470679010001 4002191117F10F907A17F4
S10D55480000000A5E002D545470A9
S11355525E002D767A370000000C01006FF0000424
S11355620F9601006F7500280AD61B7601006F7330
S113557200240AD31B737A02000000011AC44044B8
S1135582686817506839 1751191017F01AC00100CB
S113559269F04C14010069717A11000001000100E5
S11355A269F17900000 14002190017F00F8401002C
S11355B2697047041A800F826E78000368E81B75CE
S11355C21B761B730FD546B87A2400000001460EE2
S11355D201006F70000446067900FFFF40120FA01E
S11355E27A2000000001460419004004790000 01FA
S10F55F27A170000000C5E002D5454706A
S11355FE5E002D767A03000000070F820F950100DF
S113560E6F7600207A04000000180AF4694179616C
S113561E8000792180004606790 1FFFF400479015D
S113562E00010FA06889694079607FF01190119095
S113563E1190119069D07A000000000701006DF0FF
S113564E0B740FC10FE05E0051D80B977900000366
S113565E6DF00FB10FE05E0054320B8769504F06A9
S113566E7D607070402869500B5069D07D607270F8
S113567E4016790000016DF00FB10FE05E00543259
S113568E0B8769501B5069D07C60737047E45E00D2
S107569E2D545470C0
S11356A25E002D760F860F9301006F7400180FE5CD
S11356B20FC44604190040201AE6400A6C586C399C
S11356C21C9846060B761FC645F21B756858175081
S11156D21B73683B175319305E002D54547040

S11356E05E002D767A37000000100FF60F850F94B9
S11356F069506F710028091069D001006DF601002F
S11357006F71002E0FC05E0061120B977C60737087
S1135710470869500B5069D0400C7A010000001013
S11357200FE05E0060C07A000000004201006DF0EF
S11357307A01000000100FE05E0059B40B976E6809
S11357400008E8E06EE800087A00000000090100A4
S11357506DF00FE10FC05E0051D80B977A17000070
S10B576000105E002D5454708B
S11357685E002D767A37000000187A0500000009DC
S11357780AF50F840F920FC44D087A030000000145
S113578840147A03FFFFFFFF0FC07A01FFFFFFFFFB
S11357985E004C2E0F840FC07A010000001B5E00D0
S11357A84C080F860FC07A010000001B5E004C08EE
S11357B80F947A2300000001462E7A0000000008A7
S11357C801006DF00FE11031103110317A11000032
S11357D896140FD05E0051D80B970FE01030780065
S11357E86B21000096E440320FC446021B767A0010
S11357F80000000801006DF00FE110311031103185
S11358087A110000967C0FD05E0051D80B970FE0F9
S1135818103078006B21000096FE6FF100120FC460
S1135820847747A23FFFFFFFF460A7A000000001B34
S11358381AC00F8401006F70003001006DF00FA1D2
S11358480FC05E0062A00B970FE646087A2300009C
S1075858000147629F
S113585C01006F70003001006DF00FA069006DF056
S113586C7A00000000180AF00FD15E0056E00B9787
S113587C0B877A01000000400FD05E0058CA7920D4
S113588C0001460E6F7000120B506FF000127D502A
S113589C70700FA06F71001269817A00000000080C
S11358AC01006DF00FD101006F7000345E0051D810
S11158BC0B977A17000000185E002D545470ED
S1139614800000000000000CECB8F27F4200F3A17
S1139624A70C3C40A64E6C5286F0AC99B4E8DAFD24
S1139634DA01EE641A708DEAB01AE745B101E9E480
S11396448E41ADE9FBEBC27DE5D3EF282A242E81BD
S1139654B9A74A0637CE2EE195F83D0A1FB69CD921
S1139664F24A01A73CF2DCD0C3B8358109E84F07BD
S11396749E19DB92B4E31BA99E74D1B791E07E4893
S1139684C428D05AA4751E4CF2D56790AB41C2A42A
S1139694964E858C91BA2655BA121A4650E4DDECDF
S11396A4E65829B3046B0AFA8E938662882AF53F37
S11396B4B080392CC4349DEDDA7F5BF5909668490C
S11396C4873E4F75E2224E68A76C582338ED262354
S11396D4CF42894A5DCE35EB8049A4AC0C5811AE18
S11396E40000005900B3010D016601C0021A0273A0
S11396F402CD0327038003DA0434FFA6FF4CFEF2F2
S1099704FE99FE3FFDE5A6
S111970AFD8CFD32FCD8FC7FFC25FBCBFB72F3
S11358CA5E002D767A370000000C0F840F957A0656
S11358DA000000081AB30FD00FE15E004C080AC09B
S11358EA0F820FD00FE15E004C08790000801A097D
S11358FA4B04119040F86EF8000511086EF8000B7E
S113590A11086EF800041B75010069F50FD00FE149
S113591A5E004C080AC00F85010069700FE15E0042
S113592A4C08790600801A094B04119640F80FA017
S113593A68086E790005169847280FA068066E78DE

S113594A000B166846086E780004166847087A033F
S113595A00000001400C685816E847067A03000065
S113596A00017A2300000001463240140CE8175064
S113597A17106859168968D9100E46041B75FE015B
S113598A685816E847041FC544E21FC54508685806
S113599A14E868D8400679000001400219007A1712
S10D59AA0000000C5E002D54547041
S11359B45E002D761B971B970F82010069F17A0312
S11359C40000000801006F7000200FB15E004C0856
S11359D40FA60A8601006F7000200FB15E004C0809
S11359E4790400801A094B04119440F8686816C8B6
S11359F446500CCD1855400414D5110D0CDD46F852
S1135A0468681658470868681 4C868E840340FA0E3
S1135A14010069710A90010069F001006F700020B0
S1135A240FB15E004C080FA50A85400C686847084F
S1135A34685814C868D8400A0B76010069701F8639
S10F5A4445EA0B970B975E002D5454703D
S1135A505E002D760F857A040000000701006F7049
S1135A60002001006DF001006F70002001006DF057
S1135A705E00606E0B970B970D0647487926000171
S1135A80460A790004B86BA000FFF5787926000276
S1135A90460A7900044C6BA000FFF57819667A007A
S1135AA00000001C0AF00D6117F10A90F9FF6889E4
S1135AB00B56792600084DE67A000000001C0AF018
S1135AC05A005B467A000000001C0AF07A010000CD
S1135AD097185E002CB60D00470C190069D07A00A8
S1135AE00000971840607A060000001C0AF66963FC
S1135AF01113111131113113796307FF793303FE84
S1135B0069D36950791003FE462869500B5069D058
S1135B1069607960800F69E00FE30B73400E0FC17A
S1135B200FB05E0060C069501B5069D06960734854
S1135B3047EC69607960800F79403FE069E07A0063
S1095B400000001C0AF046
S1135B4601006F7100185E002D285E002D545470FD
S10B9718000000000000000046
S1135B565E002D760F8601006F70002001006DF048
S1135B6601006F70002001006DF05E00606E0B9700
S1135B760B970D05474079250001460A790004B8BD
S1135B866BA000FFF57879250002460A7900044CDC
S1135B966BA000FFF57819667A000000001C0AF076
S1135BA60D6117F10A90F9FF68890B5679260008EB
S1135BB64DE65A005C747A050000001C0AF5695527
S1135BC61115111511151115796507FF793503FFA0
S1135BD60D5C4C0E7A00000097200FE15E002D2825
S1135BE640607A000000001C0AF00FE15E002D28D9
S1135BF6792C00344C4C0FE50B950BF579090034E1
S1135C0619C90D9017F07901001001D053100D0931
S1135C16400A0FD01BF5191169811B590D994EF2D4
S1135C267900003419C017F07901001001D0531020
S1135C367909FFFF1B584B04101940F86950669009
S1075C4669D07A01A3
S1135C4A0000001C0AF10FE20F905E0028440FE0E7
S1135C5A7A01000097205E002CD20D00470C7A00CF
S1135C6A0000001C0AF07D0070707A000000001C1E
S1135C7A0AF001006F7100185E002D285E002D5492
S1055C8A547051
S10B9720000000000000003E

S1135C8C01006DF20D1A4F14792A00084D04FA0025
S1135C9C4008680A100A1B5A4EFA688A01006D7292
S1055CAC54702F
S1135CAE01006DF20D1A4F14792A00084D04FA0003
S1135CBE4008680A110A1B5A4EFA688A01006D726F
S1055CCE54700D
S1135CD001006DF201006DF101006F01000401008C
S1135CE069000D82111211121112112112796207FF4C
S1135CF00D8A7968000F793203FF4B4A79220053EA
S1135D004E44794800107912FFEC4B22792200208F
S1135D104C0C0D224F1E103112301B5240F40F90C9
S1135D207912FFE00D224F0C10301B5240F6113058
S1135D300B524BFA796A8000470217B001006D716C
S10D5D4001006D7254701A8040F2E6
S1135D4A01006DF119990D11471A4A0679090800DC
S1135D5A17917919040F1B59101144FA1031103194
S1135D6A10311031010069811A9101006F81000419
S1095D7A01006D7154707D
S1135D800F8501F064155860009A792407FF4714C2
S1135D900D44586000820FA501F064355860007807
S1135DA0580000800FA501F0643558600076406804
S1135DB00FA501F06435466C0DCC465C0F8501F0F0
S1135DC064154654405E0F8501F06415473C10315D
S1135DD01230792800104C0C1B5C103112307928DA
S1135DE000104DF4580000C40FA501F06435471AA4
S1135DF010331232792A00104C0C1B541033123218
S1135E00792A00104DF4580000A81AA21AB3687D2D
S1135E10100D131A58000228790107FF1AA21AB3AA
S1135E20580001FA790107FF1AA27A03000000085B
S1135E3068FA580001E801006DF601006DF50100F4
S1135E406DF401006DF301006DF201006DF10100CD
S1135E506DF07A3700000018691D692565D569F56D
S1135E60691C111C111C111C111C796C07FF69247E
S1095E70111411141114BA
S1135E761114796407FF01006D1001006911796837
S1135E86000F01006F23000401006922796A000FE5
S1135E96792C07FF5870FEE2792407FF5870FF0A32
S1135EA60DCC5870FF1A0D445870FF36094C793CD7
S1135EB603FF792C07FF58C0FF58792CFFCB58D026
S1135EC6FF426FFC000279480010794A0010010076
S1135ED66FF000040F9601006FF2000801006FF3E4
S1135EE6000C0D1552356FF500160D34529452B150
S1135EF60A946511990009D44406791C0001990096
S1135F066FF400140DC50D1D18990D0452340AC5FE
S1135F1699000DE452B40AC599000D6452240AC5CA
S1135F2699006FF500120DD50D1D18990D84523485
S1135F360AC50D0452B40AC599000DE452240AC5D4
S1135F4699000D6452A40AC599006FF500100DD58A
S1135F560D1D18990DB352830AB50D0452240AC5B3
S1095F6699000DE452A4B2
S1135F6C0AC59900528209D24404791A0001091A0C
S1135F7C6F74000852040AC26F7400085284094AF1
S1135F8C0D5B6F73001001006F7400126F7D0016B0
S1135F9C19556F710002113213331333413350DA0DD
S1135FAC796001004712113213331331341335 0B513B
S1135FBC792107FF5870FE5401F064454702700BBA
S1135FCC0D114E2E113213334402700B0D114A0472

S1135FDC0B5140F0732B47180CB8E80B47127A138C
S1135FEC00000008440A0B72792A00804D020B5101
S1135FFC4020732B471C0CB8E80B47167A13000090
S113600C000844020B72792A01004D061132133336
S113601C0B51113213331132133311321333796A97
S113602C000F10111011110111011641A101A687846
S113603C1008131A7A170000001801006D70010084
S113604C698201006F83000401006D7101006D72A0
S107605C01006D735C
S111606001006D7401006D7501006D765470C2
S113606E01006DF57A01000000080AF16818E87F57
S113607EA87F460A0B716818E8F0A8F047041900C8
S113608E402A6818F01050006898460A0B711AD50A
S113609E400E6818470679000000140100B710B750E
S11360AE7A25000000064DEA7900000201006D75A5
S10560BE547019
S11360C05E002D760F840F931AD51B730FB64028ED
S11360D00FC30AE30FB26838E880175017700F83B5
S11360E00FA06809100968890FD547080FC00AE097
S11360F07D0070000FB51B760FE64CD40FD5470615
S1136100790100014002191117F10F905E002D541F
S10561105470C6
S11361125E002D767A37000000387A05000000040D
S11361220AF50F840F967A000000001001006DF04B
S1136132191101006F7000545E0063840B970FE323
S11361420B930BF37A160000000701006FF600347D
S1136152790600030FC00B900BF001006FF00028CB
S11361627A140000000701006FF4002C5A00628EBB
S1136172683846 0C01006F7000346809587000FAE1
S11361827A000000001001006DF019110FD05E00BB
S113619263840B9701006F70002801006FF00030D9
S11361A201006F74002C790E0003407C01006F70B4
S11361B200301A916809FA0810311A0A4EFA1A8045
S11361C2684801F064101A916839FA0810311A0A02
S11361D24EFA01006F720034010069F01A806828D8
S11361E201F06401010069705E004C2E01006FF042
S11361F200247A000000000401006DF07A0100001F
S109620200280AF10DE281
S1136208096217F210320AD20FA05E00633C0B97A3
S11362181B5E01006F7000301BF001006FF000304F
S11362281BF40DEE4C807926000346247A00000007
S1136238000A01006DF00D6417F40FC110310AD183
S113624801006F7000540AC00AC05E0051D8402292
S11362587A000000000A01006DF00D6417F40FC105
S113626810310AD101006F7000540AC00AC05E00E1
S1136278633C0B971B561BF301006F7000341BF034
S113628801006FF000340D6658C0FEDE7A17000077
S10B629800385E002D54547020
S11362A05E002D767A370000000C0F860F940D6088
S11362B07910003F69C00FF10FE05E0054AA7A0025
S11362C00000000801006DF0191101006F70002833
S11362D05E0063840B971A800F8219550FF64010E6
S11362E00B760FA00B700F8269407930000869C0EC
S11362F0686847ECFB804004110B0B55686816B8BF
S113630047F66DF57A03000000080FA01A830FB15A
S11363100FE05E0054320B876940195069C00100D9
S11363206DF30FE101006F7000285E0051D80B97E9

S10F63307A170000000C5E002D5454701E
S113633C5E002D760F860F9401006F7500180AD638
S113634C1B760AD41B740FC319CC402268681750F0
S113635C68391751091009C00D0468E817F479005E
S113636C010001D053040D4C1B751B761B730FD509
S10B637C46DA5E002D54547053
S11363845E002D761B870F8469F101006F73001A79
S11363940FC51AE6400C0FD00B756E79000168899E
S11363A40B761FB645F00FC00B875E002D54547057
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
```

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