```
// USB_CNC_Machine リモート PIO 端末プログラムのメイン関数およびユーザアプリ部
// 2017.03.16 編集作成　by Takehiko Inoue
// 2018.01.16:revised X,Y,Z 軸駆動 steps by T.Inoue

/***********　メイン関数 ***************************/
#pragma code
void main(void)
{

        /* IO ピン初期設定　*/
        ANSEL = 0x00;                           // 0b10000000 RC3,RC2,RC1,RC0,RA4,-,-,-全てデジタルに設定
        ANSELH =0x00;                           // 0b00000000 -,-,-,-,RB5,RB4,RC7,RC6 デジタル
        TRISA = 0xFF;                           // 0b00111000 -,-,RA5,RA4,(RA3),-,-,- 入力
        TRISB = 0x30;                           // 0b00110000 RB7,RB6 出力、RB5,RB4 入力-,-,-,-
        LATB = 0x00;                            // 出力初期化
        LATC = 0x00;                            // 出力初期化
        TRISC = 0x00;                           // RC0,1,2,RC3-7 出力
        /* ADC 初期化 */
        ADCON0 = 0;                                     // 停止
        ADCON1 = 0;                                     // VDD-Vss
        ADCON2 = 0;                                     // FOSC/2
        /* USB 関連 **/
        USBDeviceInit();                        // USB 初期化
        USBGenericInHandle = 0;
        USBGenericOutHandle = 0;
        blinkStatusValid = TRUE;                // USB 目印 LED 有効化
        /** USB アタッチ許可と割り込み許可 */
```

```
        USBDeviceAttach();

        /*********** メインループ ***************/
    while(1)
    {
                /** USB 目印 LED 点滅 **/
                if(blinkStatusValid)
                        BlinkUSBStatus();               // LED 点滅実行

                /*** USB 接続中なら送受信実行 ***/
                if((USBDeviceState >= CONFIGURED_STATE)&&(USBSuspendControl!=1))
                        ProcessIO();                    // ユーザーアプリ実行
        }
}
/********************************************************
 * ユーザーアプリ、入出力処理関数
 * USB からのコマンドにより機能実行
 ********************************************************/
void ProcessIO(void)
{
        int i=0;
        int j=10;                                       //Delay100TCYx(j)(10)=0.1ms の遅延
        int k=9;                                        //Delay1KTCYx(k)(9)=0.9ms の遅延
        int n=15;                                               //Delay1KTCYx(n)(15)=1.5ms の遅延

        /***** データ受信処理 ******/
        if(!USBHandleBusy(USBGenericOutHandle))         // 受信完了か？
```

```
{
        /** 受信データ取り出し **/
        blinkStatusValid = FALSE;                       // USB 目印 LED 中止
        counter = 0;                                            // 送信バイト数リセット
        INPacket[0] = OUTPacket[0];                     // エコーバック
        INPacket[1] = OUTPacket[1];
        /******** コマンドの処理 ********/
        switch(OUTPacket[0])                            // コマンドコードチェック
        {
                /** 接続確認 OK 応答 **/

                case CHECK:
                        INPacket[2] = 'O';
                        INPacket[3] = 'K';
                        counter=0x04;                                   // 送信バイト数 4
                        break;

                /** 出力ピン出力要求と状態応答の場合 ***/

                case POUT:
                        if(!mPI_1)
                        {
                                LimitStop();
                                break;
                        }
                        else{
                        if(OUTPacket[1] == 0x32){               // DO アドレス 1PO2_X 軸回転方向
```

```
        if(OUTPacket[2] == 0x30)        // Off フ制御 cw(green)(+)
        {       mPO_2_Off();                                    // フォトカプラ削除で反転
                Delay1KTCYx(n);
        }
        else if(OUTPacket[2] == 0x31)   // On 制御 ccw(red)(-)
        {       mPO_2_On();                                             // フォトカプラ削除で反転
                Delay1KTCYx(n);
        }

        counter = 0x04;
        INPacket[2] = mPIO_2 + 0x30;
}
if(OUTPacket[1] == 0x34){                       // DO アドレス 1PO4_Y 軸回転方向
        if(OUTPacket[2] == 0x30)        // Off 制御 cw(green)(+)
        {       mPO_4_Off();                                    // フォトカプラ削除で反転
                Delay1KTCYx(n);
        }
        else if(OUTPacket[2] == 0x31)   // On 制御 ccw(red)(-)
        {       mPO_4_On();                                             // フォトカプラ削除で反転
                Delay1KTCYx(n);
        }

        counter = 0x04;
        INPacket[2] = mPIO_4 + 0x30;
}
if(OUTPacket[1] == 0x36){                       // DO アドレス 1PO6_Z 軸回転方向
        if(OUTPacket[2] == 0x30)        // Off 制御 cw(green)down(-)
```

```
        {       mPO_6_Off();                            // フォトカプラ削除で反転
                Delay1KTCYx(n);
        }
        else if(OUTPacket[2] == 0x31)   // On 制御 ccw(red)up(+)
        {       mPO_6_On();                                     // フォトカプラ削除で反転
                Delay1KTCYx(n);
        }

        counter = 0x04;
        INPacket[2] = mPIO_6 + 0x30;
}
if(OUTPacket[1] == 0x37){               // DO アドレス 1PO7_MillMotor_on-off
        if(OUTPacket[2] == 0x31){               // On 御制 run(red)
                mPO_7_On();
                Delay1KTCYx(n);
        }
        else if(OUTPacket[2] == 0x30){  // Off 制御 stop(green)
                mPO_7_Off();
                Delay1KTCYx(n);
        }

        counter = 0x04;
        INPacket[2] = mPIO_7 + 0x30;
}
if(OUTPacket[1] == 0x31){               // X 軸駆動
        if(OUTPacket[2] == 0x30)        // On 制御(run)か？
        {
```

```
                for(i=0;i<10;i++){                  // 10 パルスの発生（1/40 回転＝1.0 x 1/40＝0.025ｍｍ）
                //mPO_1_On();                       // run clock オン制御
                Delay100TCYx(j);
                mPO_1_Off();
                Delay1KTCYx(k);
                mPO_1_On();
                }
        }
        else if(OUTPacket[2] == 0x31)     // Off 制御(stop)か？
        {       mPO_1_Off();
                Delay1KTCYx(n);
        }                                                       // run clock オフ制御
        counter = 0x04;                                 // 送信バイト数 4

        INPacket[2] = mPIO_1 + 0x30;      // 状態応答セット
}

if(OUTPacket[1] == 0x33){                 //Y 軸駆動
        if(OUTPacket[2] == 0x30)          // On 制御か？
        {
                for(i=0;i<10;i++){                  // 10 パルスの発生（1/40 回転＝1.0 x 1/40＝0.025ｍｍ）
                //mPO_3_On();                               // On 制御
                Delay100TCYx(j);
                mPO_3_Off();
                Delay1KTCYx(k);
                mPO_3_On();
                }
```

```
            }
            else if(OUTPacket[2] == 0x31)
            {           mPO_3_Off();
                        Delay1KTCYx(n);
            }
            counter = 0x04;
            INPacket[2] = mPIO_3 + 0x30;
}
if(OUTPacket[1] == 0x35){                   //Z 軸駆動
            if(OUTPacket[2] == 0x30)        // On 制御か？
            {
                        for(i=0;i<10;i++){              // 10 パルスの発生（1/40 回転＝1.0 x 1/40＝0.025ｍｍ）
                        //mPO_5_On();                                   // On 制御
                        Delay100TCYx(j);
                        mPO_5_Off();
                        Delay1KTCYx(k);
                        mPO_5_On();
                        }
            }
            else if(OUTPacket[2] == 0x31)
            {           mPO_5_Off();
                        Delay1KTCYx(n);
            }
            counter = 0x04;
            INPacket[2] = mPIO_5 + 0x30;
}
}
```

```
        break;

/**** 一括転送 **************/
case ALL:
        /* DI 状態セット */
        if(mPI_1){                                          // LimitSwitch 状態 OK か？
                INPacket[2] = 0x30;                         // OK-On 応答セット
        }
        else{
                INPacket[2] = 0x31;                         // Stop-off 応答セット
                LimitStop();
        }
        counter = 4;

        if(mPI_2)
                INPacket[3] = 0x31;
        else
                INPacket[3] = 0x30;
        counter = 4;

        break;


/** リセットデバイス　***/
case RESET:
        Reset();
```

```
                        break;
                /*** 不明 ****/
                default:
                        Nop();
                        break;
        }

        /**** USB 送受信実行　*****/
        if(counter != 0)                                                        // 送信データありか？
        {
                if(!USBHandleBusy(USBGenericInHandle))                  // ビジーチェック
                {       /* 送信実行　*/
                        USBGenericInHandle = USBGenWrite(USBGEN_EP_NUM,(BYTE*)&INPacket,64);
                }
        }
        /* 次の受信実行 */
        USBGenericOutHandle = USBGenRead(USBGEN_EP_NUM,(BYTE*)&OUTPacket,USBGEN_EP_SIZE);
    }
}
```