```
typedef unsigned char *STRPTR;
typedef STRPTR *APTR;
typedef unsigned char UBYTE;
typedef unsigned short BOOL;
#define DEBUG 0
#define FALSE (0)
#define TRUE (1==1)
/* === System Macros ================================================== */
#define SetBit(v,f)     ((v)|=(bits[(f)]))
#define ClearBit(v,f)       ((v)&=~(bits[(f)]))
#define ToggleBit(v,f)  ((v)^=(bits[(f)]))
#define BitIsSet(v,f)       ((short int)(((v)&(bits[(f)]))!=0))
#define BitIsClear(v,f) ((short int)(((v)&(bits[(f)]))==0))
#define CLOCK   write_pin(4,1);Delay(1);write_pin(4,0)
#define OLD_CLOCK   Write_pin(4,1);Delay(1);Write_pin(4,0)
#define ADD(z) write_pin(9,z);CLOCK
long DDR=0xBFE301;
long DR=0xBFE101;
long DDR1=0xBFD200;
long DR1=0xBFD000;
unsigned char reg;
int bits[16]={1,2,4,8,16,32,64,128,256,512,1024,2048,4096,8198,16396,32792};
int cp=255;

#define GR_CODE 36568
#define DTP_CODE /*46586*/19172
#define DEMO_CODE 45914
#define OLD_CODE 18949

delay()
{
    int i;
    for(i=0;i<200;i++);
}

write_pin(input,state)
int input,state;
{
    int i;
    Write_pin(13,1);
    Write_pin(input,state);
    Write_pin(13,0);
    Delay(1);
}
Write_pin(input,state)
int input,state;
{
    if(input<10)
    {
        reg=*(unsigned char *)DR;
        write_mode();
        if(state==0)
        {
            if(BitIsSet(reg,input-2))
                ClearBit(reg,(input-2)); /* set bit to 0 */
        }
        else
        {
            if(BitIsClear(reg,input-2))
                SetBit(reg,(input-2)); /* set bit to 0 */

        }
        write_mode();
        poke(DR,reg);
    }
    else
    {
        reg=*(unsigned char *)DR1;
        write_mode();
        if(state==0)
        {
            if(BitIsSet(reg,input-11))
                ClearBit(reg,(input-11)); /* set bit to 0 */
```

```
        }
        else
        {
            if(BitIsClear(reg,input-11))
                SetBit(reg,(input-11)); /* set bit to 0 */
        }
        write_mode();
        poke(DR1,reg);
    }
    /*if(read_pin(input)!=state) printf("Par. port error! %ld.%ld\n",input,state);*/
}
read_pin(input)
int input;
{
    if(input<10)
    {
        reg=*(unsigned char *)DR;
        write_mode();
        if(BitIsSet(reg,input-2))
            return(TRUE);
        else
            return(FALSE);
    }
    else
    {
        reg=*(unsigned char *)DR1;
        write_mode();
        if(BitIsSet(reg,input-11))
            return(TRUE);
        else
            return(FALSE);
    }
}
static write_mode()
{
    poke(DDR1,255);
    poke(DDR,cp);
}
static read_mode()
{
    poke(DDR1,0);
    poke(DDR,0);
}

static get_plug(a1,a2,a3,a4,a5,a6)
int a1,a2,a3,a4,a5,a6;
{
    unsigned short code=0xffff;
    int i;
    Disable();
/* init plug */
    code=0xffff;
    write_pin(12,1);
    write_pin(13,0);
    write_pin(8,0);
    write_pin(4,0);
    write_pin(2,1);
    write_pin(3,1);
    write_pin(9,1);
    write_pin(8,1); /* chip select */
    CLOCK;
    ADD(1);
    ADD(0);


    ADD(a1);
    ADD(a2);
    ADD(a3);
    ADD(a4);
    ADD(a5);
    ADD(a6);
    for(i=0;i<16;i++)
    {
```

```
        write_pin(4,1);write_pin(4,0);
        write_pin(12,1);
        reg=*(unsigned char *)DR1;
        if(BitIsClear(reg,1))
        {
            SetBit(code,15-i);
        }
        else
        {
            ClearBit(code,15-i);
        }
    }
    Write_pin(11,0);
    Write_pin(12,0);
    Write_pin(13,0);
    write_mode();
    Enable();
    return(code);
}

ReadPlug(n)
int n;
{
    unsigned short code,i;
    unsigned short result=FALSE;
    if(n==0)
    {
        code=get_plug(1,1,1,1,1,1);
#if DEBUG
        printf("Address 0 is %lx\n",code);
#endif
        code=(unsigned short)get_plug(1,1,1,1,1,0);
        if(code==0x6770)
        {
#if DEBUG
            printf("RCI!\n");
#endif
            result=TRUE;
        }
        else
        if(code==0x499b)
        {
#if DEBUG
            printf("DTP!\n");
#endif
            result=TRUE;
        }
        else
        if(code==0x67ad)
        {
#if DEBUG
            printf("RCI!\n");
#endif
            result=TRUE;
        }
        else
        {
            result=FALSE;
        }
#if DEBUG
        printf("code=%lx\n",code);
#endif

        code=(unsigned short)get_plug(1,1,1,1,0,1);
#if DEBUG
        printf("sernum=%ld\n",code);
#endif
        return(result);
    }
    code=(unsigned short)get_plug(1,1,1,1,0,1);
    return(code);
}
#define PROTECTION 1
```

```
CheckProtection()
{
    int i;
#if PROTECTION

    if(ReadPlug(0))
    {
        return(TRUE);
    }
    else
    {
        return(FALSE);
    }
#else
    printf("UnProtected Version! for HarmonySoft use only!\n");
    return(TRUE);
#endif
}
static Poke(a,b)
APTR a;
unsigned char b;
{
    CopyMem(&b,a,1);
}
/* 6A7C */
```