```
void   SetMask(unsigned);
void   SetCamera(int);
void   SetPIBorEGA(int);
void   SetBlank(int);
void   FillPibRect(unsigned,int,int,int,int,int);
void   WaitEven(void);
void   WaitOdd(void);
int    InitPIB(void);
int    TestRow(int);
int    WaitVsync(void);
void   FillPibRow(unsigned,int,int,int,int);
void    FGetPibRow(char far *,int,int,int);
void    FPutPibRow(char far *,int,int,int);
void    GetPibRow(char far *,int,int,`int);
void    PutPibRow(char far *,int,int,int);
unsigned GetPixel(unsigned *,int,int);
void PutPixel(unsigned *,int,int,int);
void SaveData(unsigned,unsigned,int);
void SetBright(int);
```

Biểu đồ màu cho các màu tính ra bằng vỉ mạch PIB được giới thiệu ở cuối cuốn sách này. Nếu bạn có phần kiểm tra vật lý của "độ sáng" (ví dụ như College Physic) bạn sẽ có thể so sánh biểu đồ màu của kiểm tra vật lý với kết quả rút ra từ chương trình 11.1. Nếu hệ thống có khả năng hiện nhiều màu hơn bạn sẽ có một sơ đồ dày đặc hơn.

Bây giờ ta sẽ so sánh vỉ mạch PIB với vỉ mạch VGA. Để làm như việc này bạn sẽ phải tính ra một bảng màu có cùng một số mức của đỏ, lục, lam. Nếu các màu chính có $2^6 = 64$ mức trên VGA, bạn có thể chọn các mức 0, 12, 24, 36, 48, và 60 cho tất cả các màu đỏ, lục, lam để xác lập ra bảng cân bằng màu. Vì vậy mà ta có tổng cộng $6 \times 6 \times 6 = 216$ màu. Bây giờ đưa 216 màu này vào vỉ mạch VGA. Khi tính 32,680 màu cho chuyển đổi 15 bit/ điểm, bạn nhóm 6 màu lại với nhau, chẳng hạn, $(R/6) \times 6$,...v.v.., ở đây *R,G,B* là giá trị nguyên. Nếu được biểu diễn ảnh 5 bit màu, bạn cần nhân kết quả với 2 để chuyển sang dạng 6 bit/màu như trường hợp tương thích với VGA. Biết *R,G,B* chúng ta có thể tính địa chỉ của bảng màu dùng

$$Address = (B/6+G+R\times6) / 2$$

Mã chương trình cho hiện sơ đồ biểu đồ trên VGA như ví dụ trên cho ở chương trình 11.2

Chương trình 11.2 "CHROMVGA.C". Displaying the chromaticity diagram on a VGA card.

```
/* Program for displaying chromaticity
diagram using a VGA.  The program emulates
a video card capable of 15 bits/pixel.*/

#include <stdio.h>
#include <dos.h>
#include <conio.h>
#include <math.h>
#include <io.h>

void main()
 {
 int xp,yp,l,R,G,B;
 unsigned i;
 float x,y,X,Y,Z,D;
 unsigned char a[648],b[648],display_mode,active_page;
 int color;
 char far *farptr;
 union REGS reg;
 struct SREGS sreg;

clrscr();
        /* Generate color palette.  We assume 6 bits
        per color for VGA, and therefore the range
        will extend from 0 to 63per color. */
l=0 ;
for(R=0;R<64;R+=12)
for(G=0;G<64;G+=12)
for(B=0;B<64;B+=12)
  {
   a[1]=R;
   a[++l]=G;
   a [++l]=B;
   l++;
  }
/* Get current mode. */
reg.h.ah=0x0F;
int86(0x10,&reg,&reg);
```

```
display_mode=reg.h.al;
active_page=reg.h.bh;
/* Setting display mode to 32Ox2OO. */
reg.h.ah=0;
reg.h.al=0x13;
int86(0x10,&reg,&reg);
/* Read color palette.*/
reg.h.ah=0x10;
reg.h.al=0x17;
reg.x.bx=0;
reg.x.cx=216;
sreg.es=FP_SEG(b);
reg.x.dx=FP_OFF(b);
int86x(0x10,&reg,&reg,&sreg);
/* Setting new color palette. */
reg.h.ah=0x10;
reg.h.al=0x12;
reg.x.bx=0;
reg.x.cx=216;
sreg.es=FP_SEG(a);
reg.x.dx=FP_OFF(a);
int86x(0x10,&reg,&reg,&sreg);
for(i=1;i<32768;i++)
 {
  R=(0x001F&i);
  B=(0x03E0 & i) >> 5;
  G=(0x7C00 & i) >> 10;
  X=2.769*R+1.7518*G+1.1300*B;
  Y=R+4.5907*G+0.0601*B;
  Z=0.0565*G+5.5943*B;
  D=X+Y+Z;
        x=X/D; y=Y/D;
  xp=(int)(300.0*x+0.5);
  yp=(int)(190.0-y*190+0.5);
        /* Group every 6 colors in red, green and
        blue,i.e. (R/6)*6, and then multiply by 2
        to spread the range from 0 to 63 (6-bits.) */
  R=(R/6)*12;
  G=(G/6)*12;
  B=(B/6)*12;
  color=(B/6+G+R*6)/2; /* position in palette. */
  reg.h.ah=0x0C;
```

```
  reg.h.al=(char)color;
  reg.h.bh=0;
  reg.x.cx=xp;
  reg.x.dx=yp;
  int86(0x10,&reg,&reg);
     }
getch();
/* Restore previous color palette. */
reg.h.ah=0x10;
reg.h.al=0x12;
reg.x.bx=0;
reg.x.cx=216;
sreg.es=FP_SEG(b);
reg.x.dx=FP_OFF(b);
int86x(0x10,&reg,&reg,&sreg);
/* Restore previous mode and page. */
        reg.h.ah=0x00;
        reg.h.al=display_mode;
        int86(0x10,&reg,&reg);
        reg.h.ah=0x05;
        reg.h.al=active_page;
        int86(0x10,&reg,&reg);
        }
```


11.5 Hiển thị ảnh màu trên hệ thống màu 15 bit và 8 bit

Phần sắp tới chúng ta sẽ xây dựng chương trình cho hiển thị ảnh và sửa lại màu, độ sáng ..., trên vỉ mạch PIB. Chương trình cũng bao gồm nén và chứa ảnh trên một file. Sau đó là một chương trình cho hiển thị ảnh màu bằng vỉ mạch VGA.

Chương trình 11.3 "DISPPIB.C". To display, freeze, and save an image on the PlB board.

```
/* Program for displaying images in live mode
through the PIB board.  Through the program you
can adjust colors, brightness,etc..You are also
given the choices to freeze and save the image
to a file. */

#include <custom.h>
```

```
#include <stdio.h>
#include <io.h>
#include  <conio.h>
#include <graphics.h>
#include  <stdlib.h>

#define U_ARROW 0x48
#define D_ARROW 0x50

void main()
  {
  int ind,k1,k2,k3,k4,ind2,i;
  char a;
  char buff[1024],file_name[30];
  FILE *fptro;

clrscr();
InitPIB();
SetScreen(0);
SetInDispMode();
SetCamera(0);
SetInterlace(0);
SetLiveMode();
k1=140; k2=126; k3=162; k4=44;
SetHue(k1);
SetContrast(k2);
SetSaturation(k3);
SetBright(k4);
textattr(WHITE+(BLUE<<4));
cputs("Do you wish to adjust colors? y or n-->");
ind=1;
while (ind)
   {
   a=getche();
   switch(a)
         {
         case 'Y':
         case 'y':
gotoxy(1,2);
cputs("Adjusting colors:");
cputs(" To adjust colors press H or C or S or B,");
gotoxy(1,3);
```

```
cputs("for Hue, Contrast, Saturation, and Brightness,");
cputs(" respectively.");
gotoxy(1,4);
cputs("You can also use the up and down arrow keys.");
gotoxy(1,5);
cputs(" Then press + to increase or - to decrease");
cputs(" value.");
gotoxy(1,6);
cputs( " To exit adjustment press ESC\n");
gotoxy(30,8);
textattr(WHITE+(RED<<4));
cprintf("Hue %4d",k1);
gotoxy(30,9);
cprintf("Contrast    %4d",k2);
gotoxy(30,10);
cprintf ( "Sat urati on       %4d",k3);
gotoxy(30,11);
cprintf ( "Brightness        %4d",k4);
a=getch();
ind2=1;
i=-1;
  while(ind2)
    {
   if((a==D_ARROW)||(a==U_ARROW))
     {
    switch(a)
        {
        case D_ARROW:
         i++;
        if(i>3) i=0;
        break;
        case U_ARROW:
        i--;
        if(i<0) i=3;
          }
        switch(i)
           {
          case 0:
                a='H'; break;
          case 1:
                a='C'; break;
          case 2:
```

```
                a='S'; break;
            case 3:
                a='B';
             }
           }
textattr(WHITE+(GREEN<<4));
gotoxy(1,10);
cprintf("Key pressed %c\n",a);
switch(a)
  {
   case 'h':
   case 'H':
            gotoxy(30,8);
            cprintf("Hue     %4d",k1);
     while(((a=(char)getch())=='+')||(a=='-'))
         {
         switch(a)
         {
         case '+':
                ++k1 ;
                if(k1>255) k1=0;
                break;
         case '-':
                --k1;
                if(k1<0) k1=255;
                }
                SetHue(k1);
                gotoxy(30,14);
                cprintf("Key pressed %c",a);
                gotoxy(30,8);
                cprintf("Hue         %4d",k1);
         }
         textattr(WHITE+(RED<<4));
         gotoxy(30,8);
         cprintf("Hue           %4d",k1);
         break;
 case 'c':
 case 'C':
         gotoxy(30,9);
         cprintf("Contrast     %4d",k2);
    while(((a=(char)getch())-='+')||(a-='-'))
      {
```

```
      switch(a)
       {
         case '+':
          ++k2 ;
          if(k2>255) k2=0;
          break;
         case '-':
          --k2;
          if(k2<0) k2=255;
          }
                SetContrast(k2);
         gotoxy(30,14);
         cprintf("Key pressed %c",a);
         gotoxy(30,9);
         cprintf("Contrast     %4d",k2);
       }
    textattr(WHITE+(RED<<4));
    gotoxy(30,9);
    cprintf("Contrast     %4d",k2);
    break;
case 's':
case 'S':
         gotoxy(30,10);
       cprintf("Saturation %4d",k3);
   while(((a=(char)getch())=='+')||(a=='-'))
      {
      switch(a)
         {
         case '+':
           ++k3;
           if(k3>255) k3=0;
           break;
         case '-':
            --k3;
            if(k3<0) k3=255;
             }
             SetSaturation(k3);
             gotoxy(30,14);
             cprintf("Key pressed %c",a);
             gotoxy(30,10);
             cprintf("Saturation %4d",k3);
           }
```

```
        textattr(WHITE+(RED<<4));
        gotoxy(30,10);
        cprintf("Saturation %4d",k3);
        break;
   case 'b':
   case 'B':
        gotoxy(30,11);
    cprintf("Brightness %4d",k4);
   while(((a=(char)getch())=='+')||(a=='-'))
      {
        switch(a)
         {
         case '+':
                ++k4 ;
                if(k4>255) k4=0;
                break;
         case '-':
                --k4;
                if(k4<0) k4=255;
                }
                SetBright(k4);
                gotoxy(30,14);
                cprintf("Key pressed %c",a);
                gotoxy(30,11);
                cprintf("Brightness %4d",k4);
           }
        textattr(WHITE+(RED<<4));
        gotoxy(30,11);
        cprintf("Brightness %4d",k4);
        break;
    case (char)27:
                ind2=0;
                break;
    default:
                a=getch();
                break;
                 }
                 }
    ind=0;
    break;
case 'N':
case 'n':
```

```
  ind=0;
  break ;
default:
  gotoxy(40,1);
  break;
  }
  }
textattr((BLACK<<4));
clrscr();
textattr(WHITE+(BLUE<<4));
gotoxy(1,1);
cputs("Do you wish to freeze image (y or n)-->");
while(((a=getch())!='y')&&(a!='n'));
gotoxy(40,1);
cprintf("%c",a);
if(a=='n')
        {
        textattr(LIGHTGRAY+(BLACK<<4));
        clrscr();
        exit(1);
        }
CaptureFrame();
SetInDispMode();
gotoxy(1,2);
cputs( "DO You wish to save image on disk (y or n)-->");
while(((a=getch())!='y')&&(a!='n'));
gotoxy(46,2);
cprintf( "%c", a);
if(a=='n')
   {
   textattr(LIGHTGRAY+(BLACK<<4));
   clrscr();
   exit(1);
   }
gotoxy(1,3);
cprintf("Enter name of fi I e to save image to ->");
scanf( "%s" , file_name);
fptro=fopen( file_name,"wb");
gotoxy(70,25);
textattr(WHITE+(GREEN<<4)+BLINK);
cputs( "WAIT");
for(i=0;i<256;i++)
```