How to mix or separate three color channels

> ***QUESTION***
>
> I need to send a picture to 3 separate monochrome monitors in an embedded application I am writing. I could simply allow only one monitor to work at a time but thought it might be possible to drive the RGB outputs seperately. This could be achived by creating 3 pictures, one with red shades, one with green shades and one with blue shades and blending them together. I could do this by mixing the pictures pixel by pixel but suspect this would be extremely slow.

But it's the only way (all other ways I can think of go back to the same). Use scanline and it is not that "extremely slow":

```
{ ... }
var
  Ptr1, Ptr2, Ptr3: ^Byte;
  PtrMix: ^Byte;
  X, Y: Integer;
begin
  for Y := 0 to Height-1 do
  begin
    Ptr1 := RedBitmap.ScanLine[Y];
    Ptr2 := GreenBitmap.ScanLine[Y];
    Ptr3 := BlueBitmap.ScanLine[Y];
    PtrMix := MixBitmap.ScanLine[Y];
    for X := 0 to Width-1 do
    begin
      PtrMix^ := Ptr1^;
      Inc (Ptr1);
      Inc (PtrMix);
      PtrMix^ := Ptr2^;
      Inc (Ptr2);
      Inc (PtrMix);
      PtrMix^ := Ptr3^;
      Inc (Ptr3);
      Inc (PtrMix);
    end;
  end;
end;
```

Make sure *MixBitmap* has 24 bit format, *RedBitmap*, *GreenBitmap* and *BlueBitmap* have 8 bit format (or change the code for other formats). All bitmaps should have the same size (or use the smallest size for *WidthM* and *Height*).

Separating 3 color channels (instead of mixing) is very similar. Simply change `PtrMix^ := Ptr1^;` to `Ptr1^ := PtrMix^;` (and for the other channels, too).

Original resource:	The Delphi Pool
Author:	Jens Gruschel
Added:	2009-11-06
Last updated:	2009-11-06