



*Professional
LCD Controller Provider*



RA8876

RA8877

New Example

2019.12.24

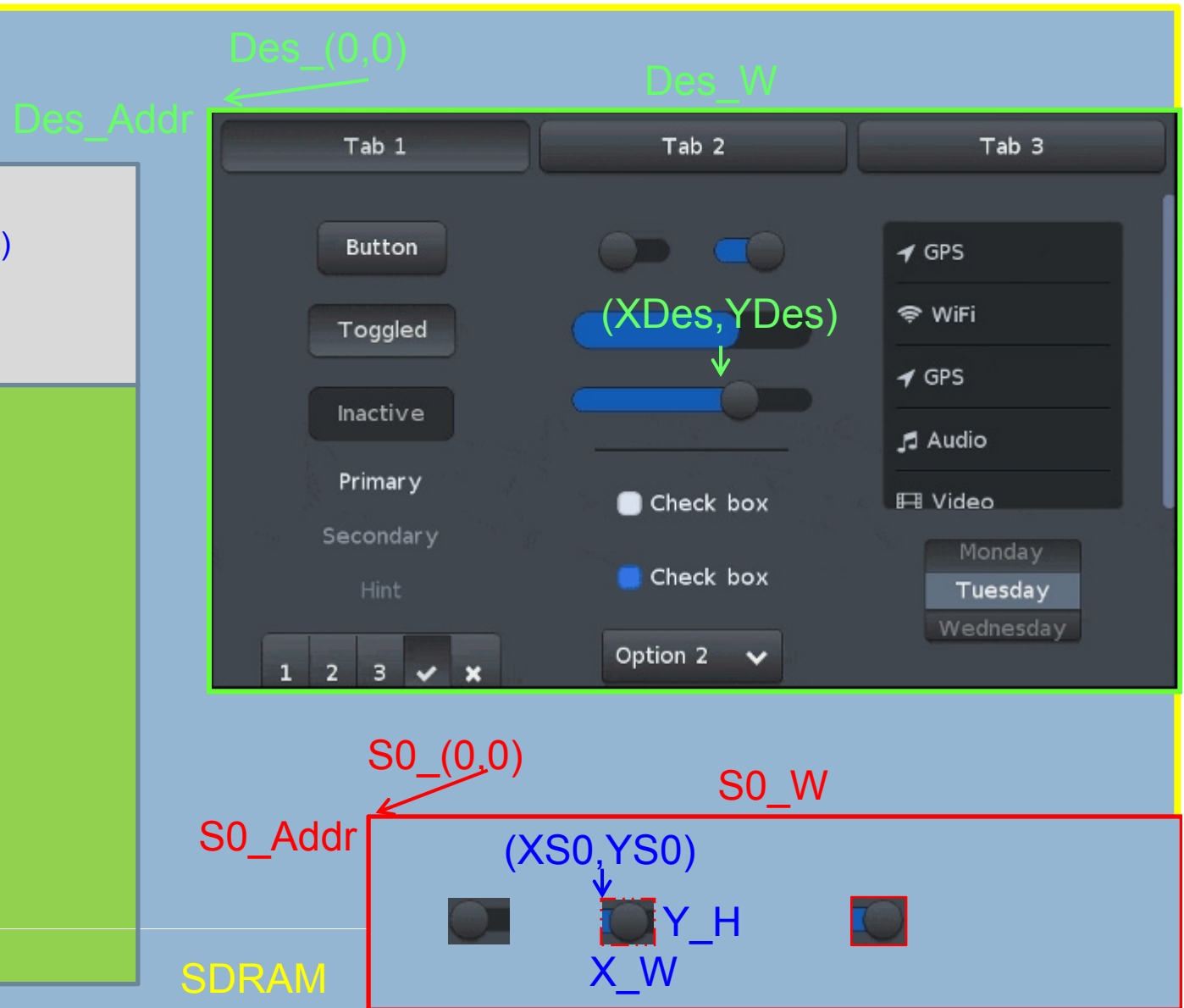


RA8876_API.c, using :

```
void BTE_Memory_Copy(S0_Addr,S0_W,XS0,YS0,S1_Addr,S1_W,XS1,YS1,Des_Addr,Des_W,XDes,YDes,ROP_Code,X_W,Y_H);
```

Copy with BTE Function

```
for( XDes=i; XDes<=j, XDes++)  
{  
    BTE_Memory_Copy(.....);  
}
```



```

void BTE_Memory_Copy
(
unsigned long S0_Addr //Start address of Source 0
,unsigned short S0_W //image width of Source 0 (recommend = canvas image width)
,unsigned short XS0 //coordinate X of Source 0
,unsigned short YS0 //coordinate Y of Source 0
,unsigned long S1_Addr //Start address of Source 1
,unsigned short S1_W //image width of Source 1 (recommend = canvas image width)
,unsigned short XS1 //coordinate X of Source 1
,unsigned short YS1 //coordinate Y of Source 1
,unsigned long Des_Addr //start address of Destination
,unsigned short Des_W //image width of Destination (recommend = canvas image width)
,unsigned short XDes //coordinate X of Destination
,unsigned short YDes //coordinate Y of Destination
,unsigned int ROP_Code
/*ROP_Code :
0000b          0(Blackness)
0001b          ~S0!E~S1 or ~(S0+S1)
0010b          ~S0!ES1
0011b          ~S0
0100b          S0!E~S1
0101b          ~S1
0110b          S0^S1
0111b          ~S0 + ~S1 or ~(S0 + S1)
1000b          S0!ES1
1001b          ~(S0^S1)
1010b          S1
1011b          ~S0+S1
1100b          S0
1101b          S0+~S1
1110b          S0+S1
1111b          1(whiteness)*/
,unsigned short X_W //X_W : Width of BTE Window
,unsigned short Y_H //Y_H : Length of BTE Window
)
{
    BTE_S0_Memory_Start_Address(S0_Addr);
    BTE_S0_Image_Width(S0_W);
    BTE_S0_Window_Start_XY(XS0,YS0);

    BTE_S1_Memory_Start_Address(S1_Addr);
    BTE_S1_Image_Width(S1_W);
    BTE_S1_Window_Start_XY(XS1,YS1);

    BTE_Destination_Memory_Start_Address(Des_Addr);
    BTE_Destination_Image_Width(Des_W);
    BTE_Destination_Window_Start_XY(XDes,YDes);

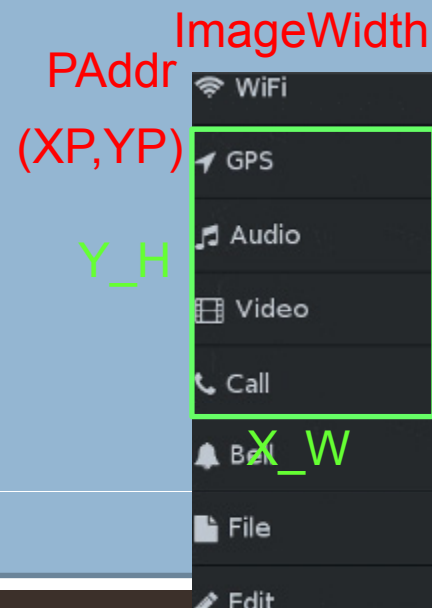
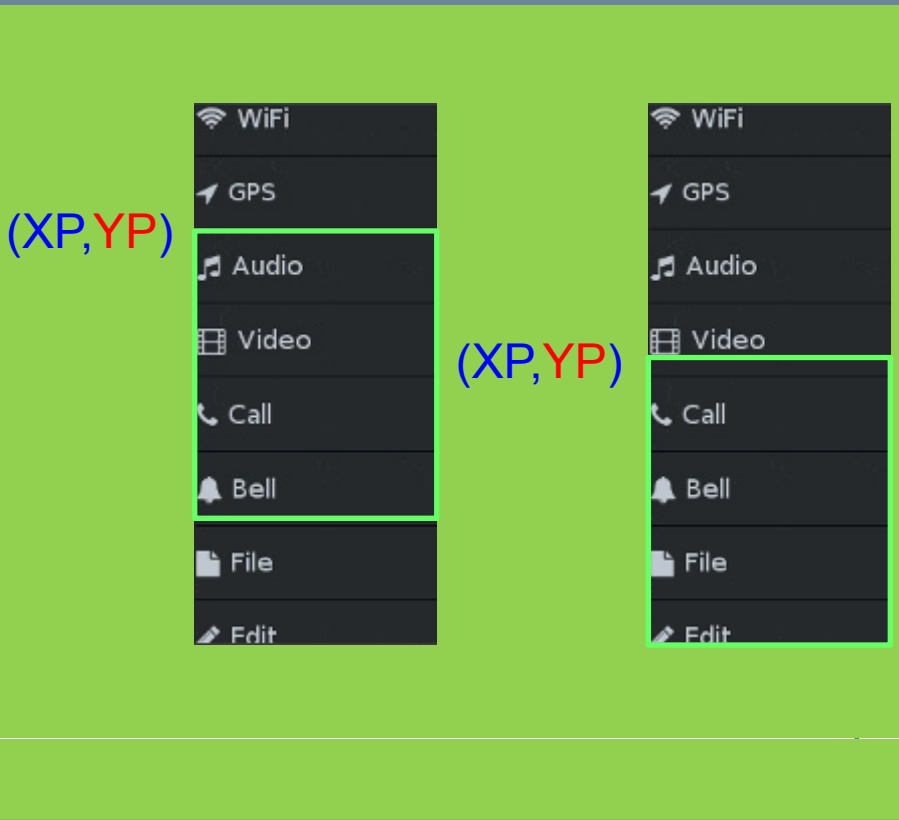
    BTE_ROP_Code(ROP_Code);
    BTE_Operation_Code(0x02); //BTE Operation: Memory copy (move) with ROP.
    BTE_Window_Size(X_W,Y_H);
    BTE_Enable();
    Check_BTE_Busy();
}

```

Using [BTE_Memory_Copy](#) (as on the previous page)
or [PIP Function](#) (as follows)
void [PIP](#)(On_Off, Select_PIP,PAddr,XP,YP,ImageWidth,X_Dis,Y_Dis,X_W,Y_H);

If using PIP function

```
for(YP=i; YP<=j, YP++)
{
    PIP(.....);
}
```



SDRAM

```

void PIP
(
    unsigned char On_Off // 0 : disable PIP, 1 : enable PIP, 2 : To maintain the original state
    ,unsigned char Select_PIP // 1 : use PIP1 , 2 : use PIP2
    ,unsigned long PAddr //start address of PIP
    ,unsigned short XP //coordinate X of PIP Window, It must be divided by 4.
    ,unsigned short YP //coordinate Y of PIP Window, It must be divided by 4.
    ,unsigned long ImageWidth //Image Width of PIP (recommend = canvas image width)
    ,unsigned short X_Dis //coordinate X of Display Window
    ,unsigned short Y_Dis //coordinate Y of Display Window
    ,unsigned short X_W //width of PIP and Display Window, It must be divided by 4.
    ,unsigned short Y_H //height of PIP and Display Window , It must be divided by 4.
)
{
    if(Select_PIP == 1 )
    {
        Select_PIP1_Parameter();
    }
    if(Select_PIP == 2 )
    {
        Select_PIP2_Parameter();
    }
    PIP_Display_Start_XY(X_Dis,Y_Dis);
    PIP_Image_Start_Address(PAddr);
    PIP_Image_Width(ImageWidth);
    PIP_Window_Image_Start_XY(XP,YP);
    PIP_Window_Width_Height(X_W,Y_H);

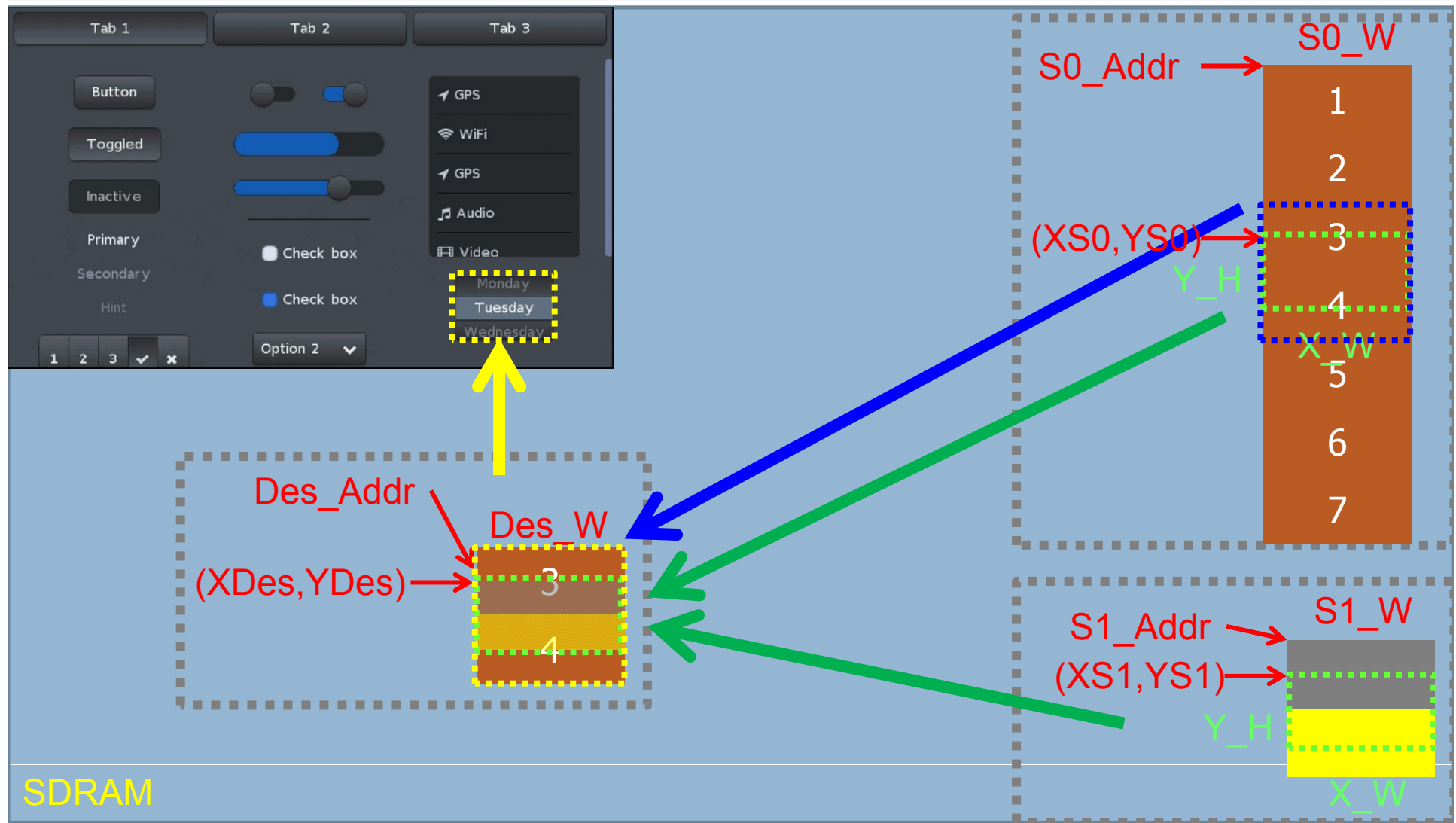
    if(On_Off == 0)
    {
        if(Select_PIP == 1 )
        {
            Disable_PIP1();
        }
        if(Select_PIP == 2 )
        {
            Disable_PIP2();
        }
    }
    if(On_Off == 1)
    {
        if(Select_PIP == 1 )
        {
            Enable_PIP1();
        }
        if(Select_PIP == 2 )
        {
            Enable_PIP2();
        }
    }
}

```

Step1: BTE_Memory_Copy();

Step2: BTE_Alpha_Blending(S0_Addr,S0_W,XS0,YS0,S1_Addr,S1_W,XS1,YS1,Des_Addr,Des_W, XDes,YDes,X_W,Y_H,alpha);

Step3: PIP(); or BTE_Memory_Copy();



```

void BTE_Alpha_Blending(unsigned long S0_Addr,unsigned short S0_W,unsigned short XS0,unsigned short YS0,unsigned long S1_Addr,unsigned short S1_W,unsigned short XS1,unsigned short
YS1,unsigned long Des_Addr,unsigned short Des_W, unsigned short XDes,unsigned short YDes,unsigned short X_W,unsigned short Y_H,unsigned char alpha)
{
/*
S0_Addr : Source 0 Start address
S0_W : Source 0 image width(recommend = canvas image width)
XS0 : Source 0 coordinate of X
YS0 : Source 0 coordinate of Y
S1_Addr : Source 1 Start address
S1_W : Source 1 image width(recommend = canvas image width)
XS1 : Source 1 coordinate of X
YS1 : Source 1 coordinate of Y
Des_Addr : Destination start address
Des_W : Destination image width(recommend = canvas image width)
XDes : Destination coordinate of X
YDes : Destination coordinate of Y
X_W : BTE Window Size of X
Y_H : BTE Window Size of Y
alpha : Alpha Blending effect 0 ~ 32, Destination data = (Source 0 * (1- alpha)) + (Source 1 * alpha)
*/

BTE_S0_Memory_Start_Address(S0_Addr);
BTE_S0_Image_Width(S0_W);
BTE_S0_Window_Start_XY(XS0,YS0);

BTE_S1_Memory_Start_Address(S1_Addr);
BTE_S1_Image_Width(S1_W);
BTE_S1_Window_Start_XY(XS1,YS1);

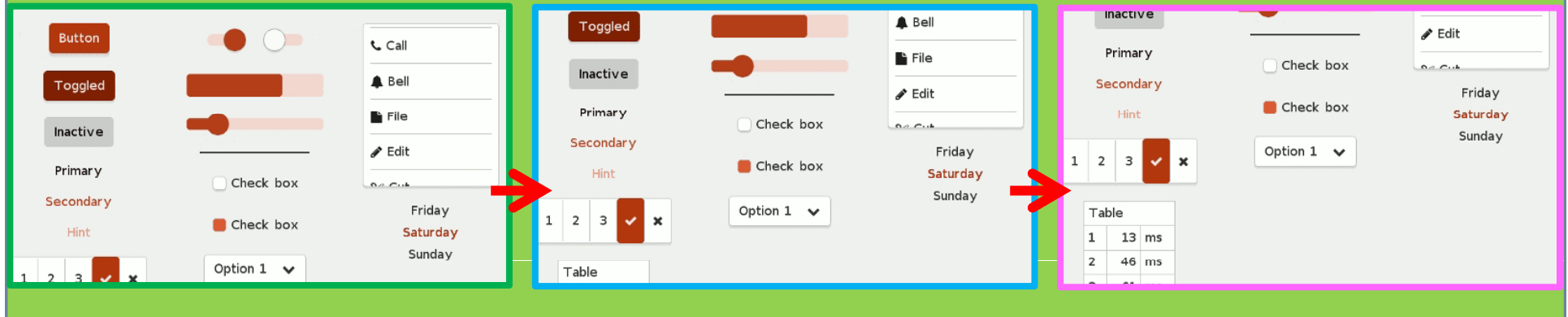
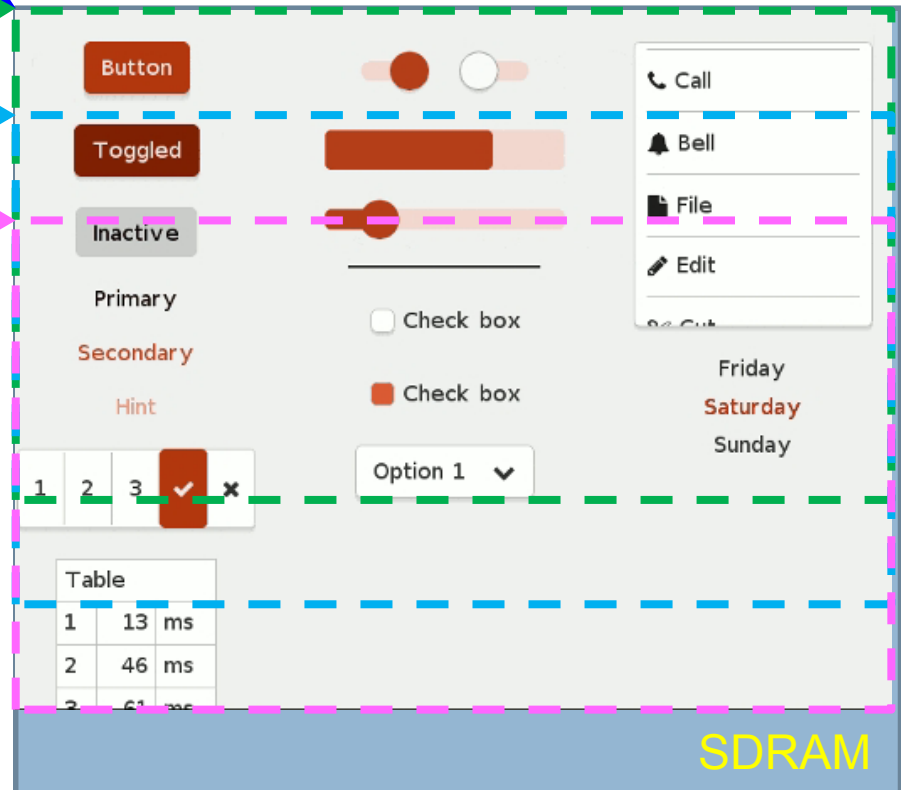
BTE_Destination_Memory_Start_Address(Des_Addr);
BTE_Destination_Image_Width(Des_W);
BTE_Destination_Window_Start_XY(XDes,YDes);

BTE_Window_Size(X_W,Y_H);
BTE_Operation_Code(0x0A); //BTE Operation: Memory write with opacity (w/o ROP)
BTE_Alpha_Blending_Effect(alpha);
BTE_Enable();
Check_BTE_Busy();
}

```

```
Main_Window_Start_XY(x=0,y=200);
```

image_width

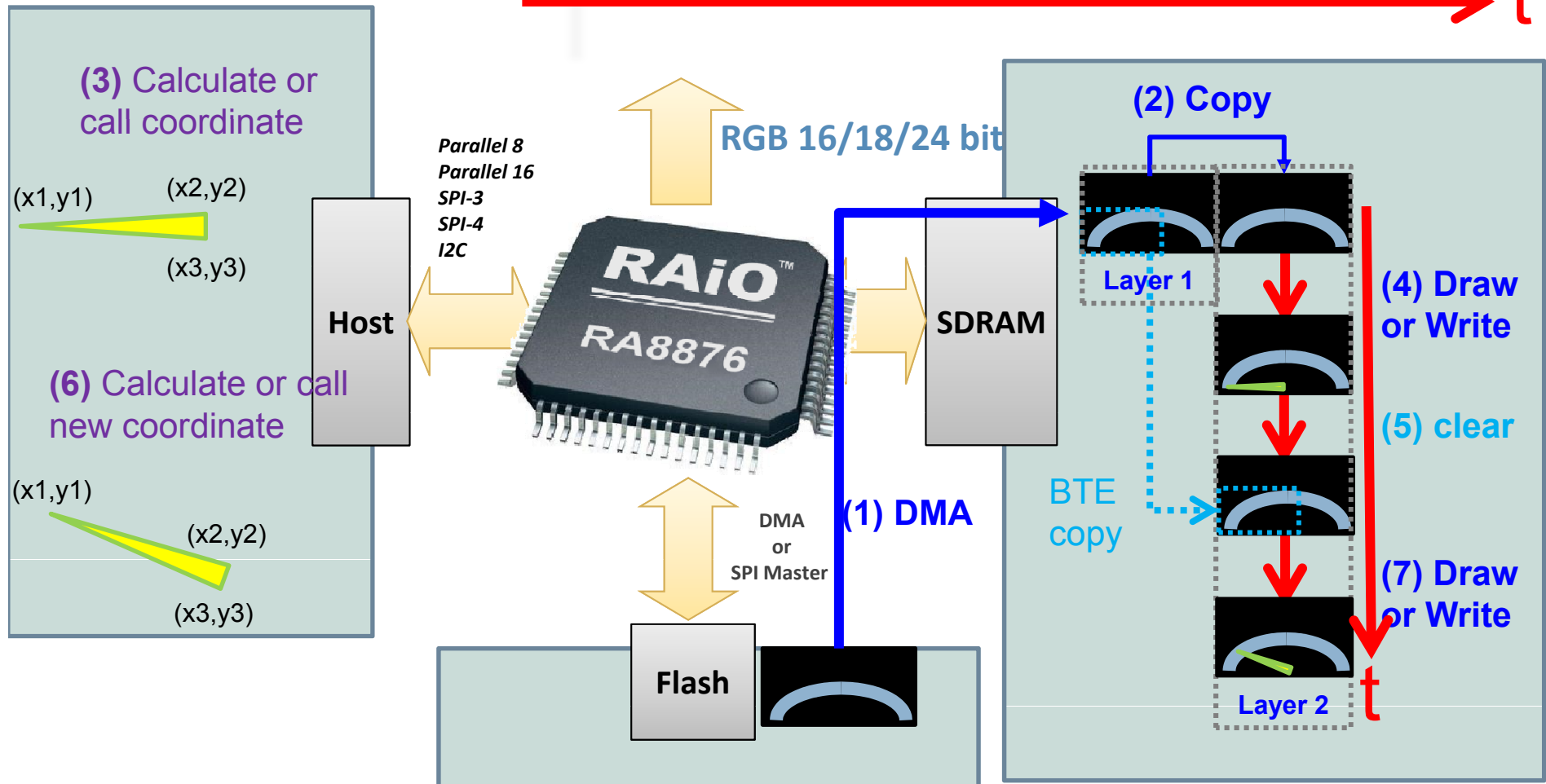
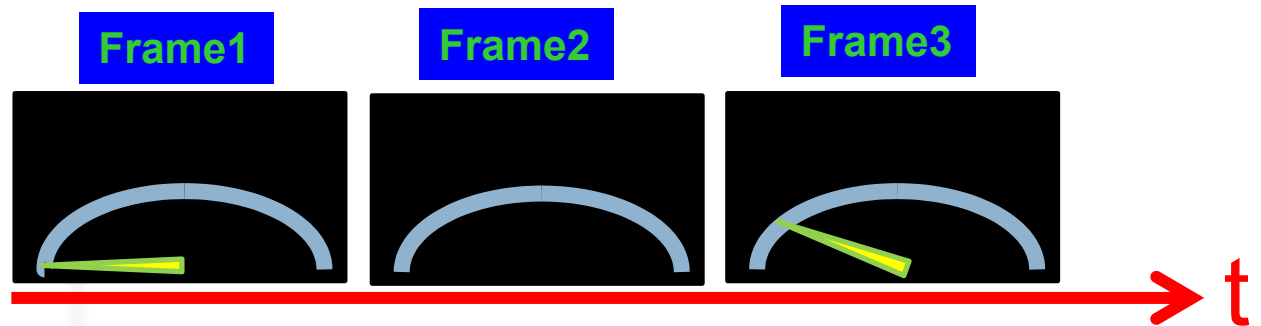


Example

2018.01.10

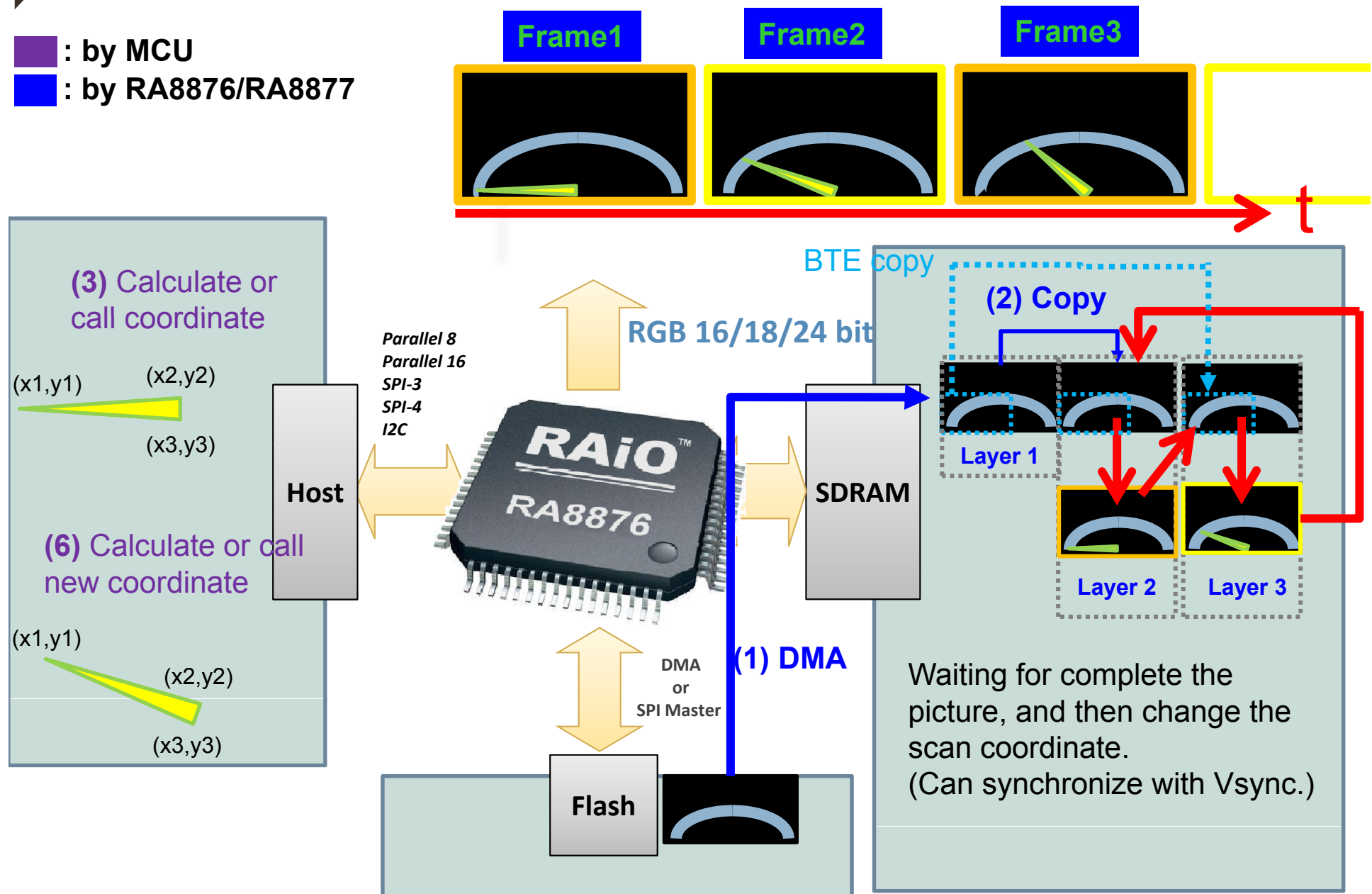
Schematic diagram 1

: by MCU
 : by RA8876/RA8877



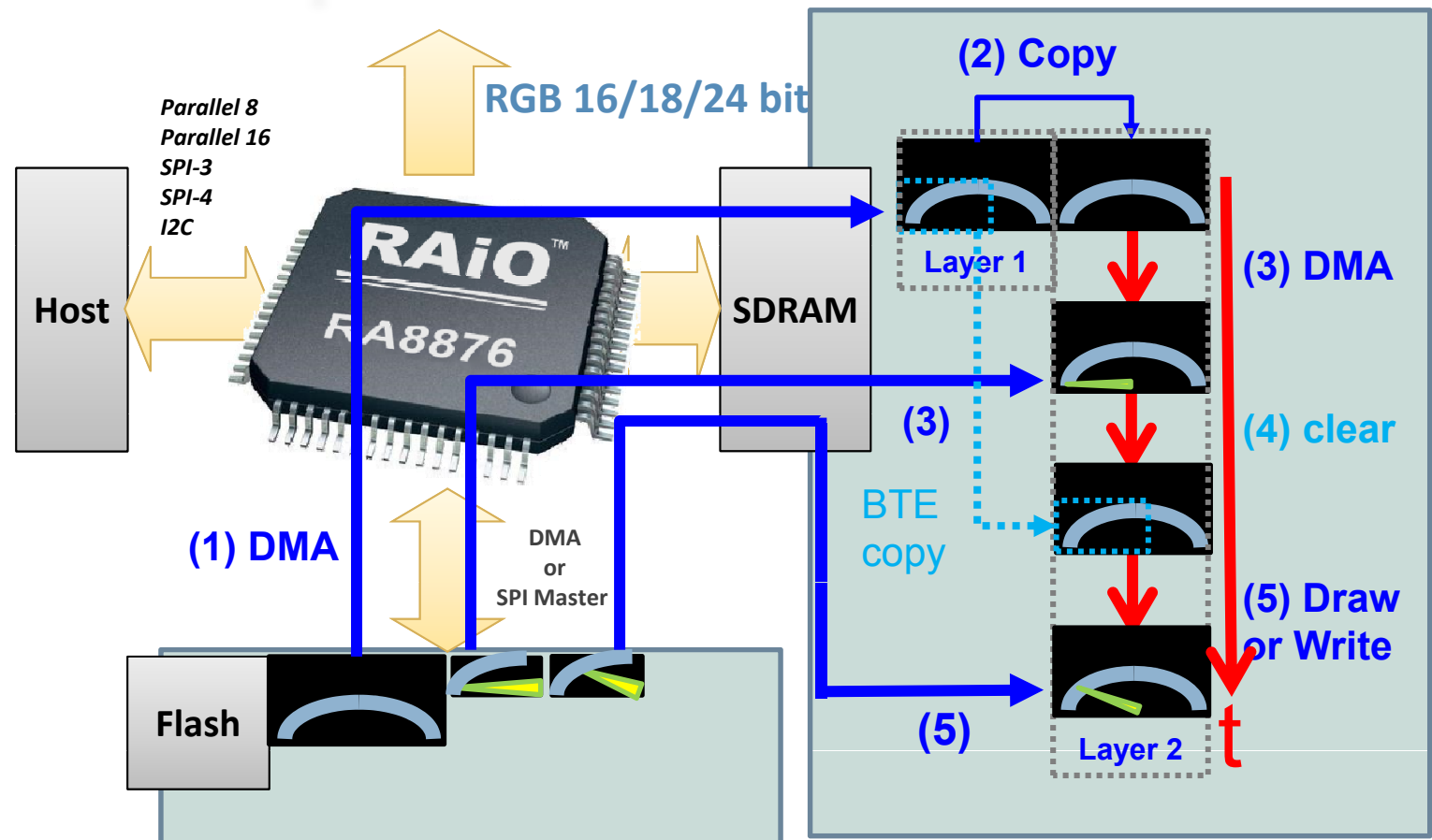
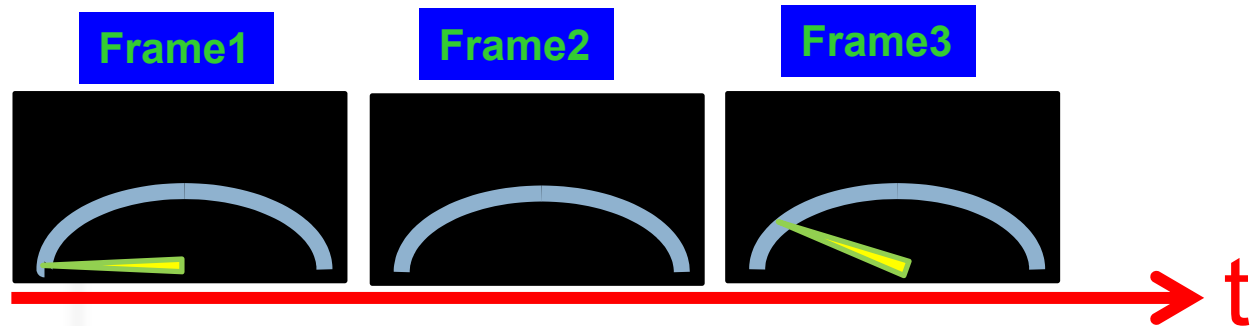
Schematic diagram 2

: by MCU
 : by RA8876/RA8877



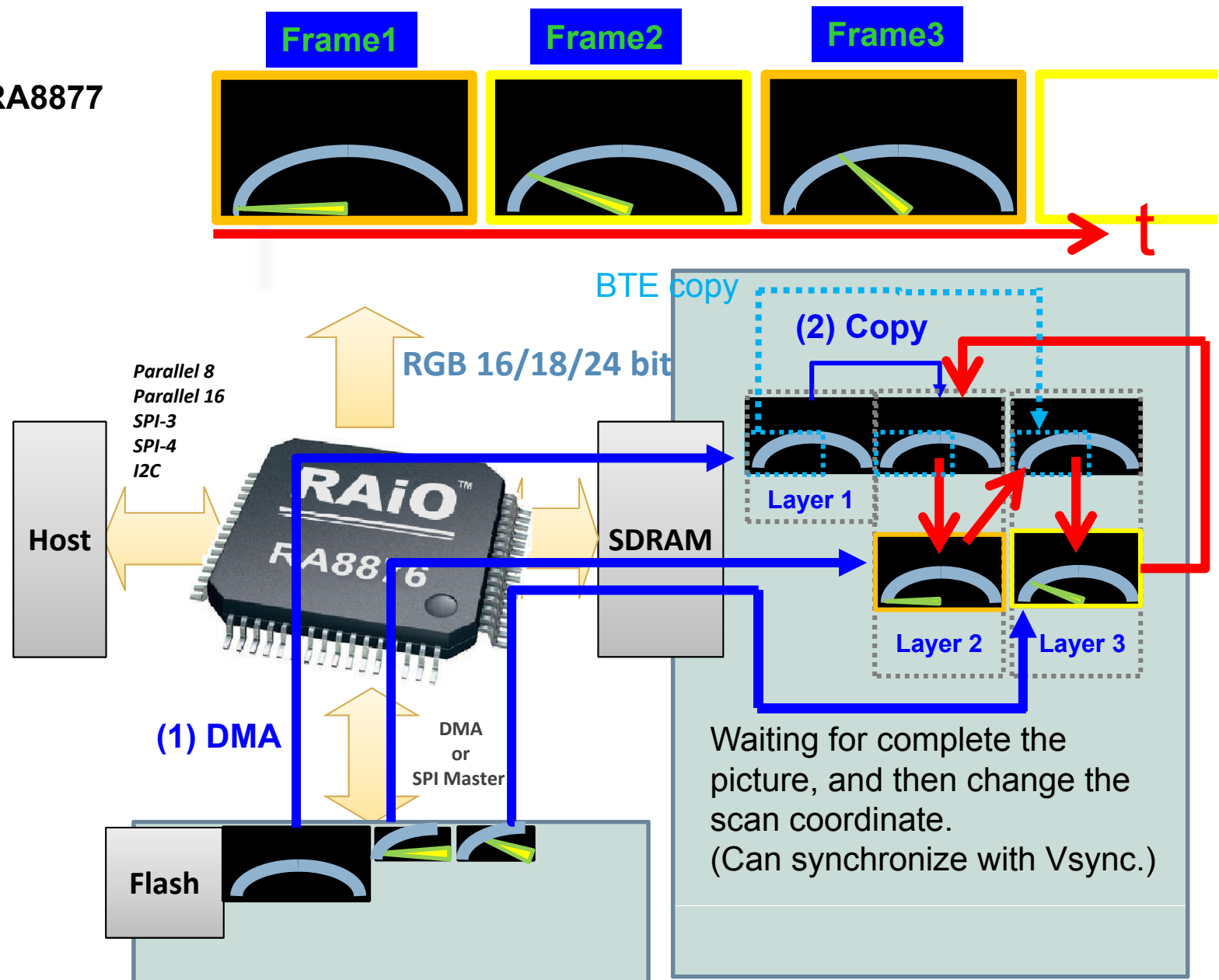
Schematic diagram 3

■ : by MCU
■ : by RA8876/RA8877



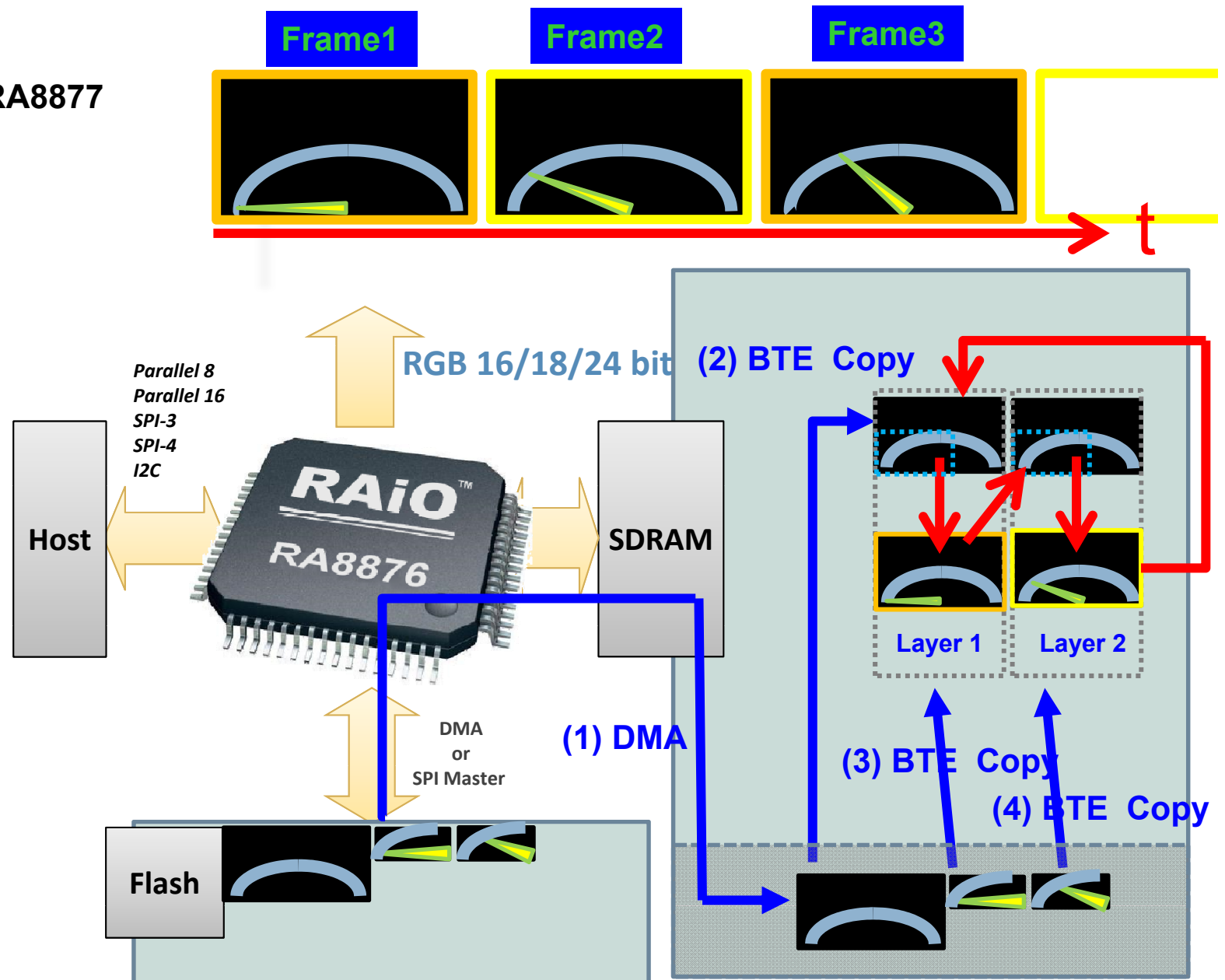
Schematic diagram 4

■ : by MCU
■ : by RA8876/RA8877



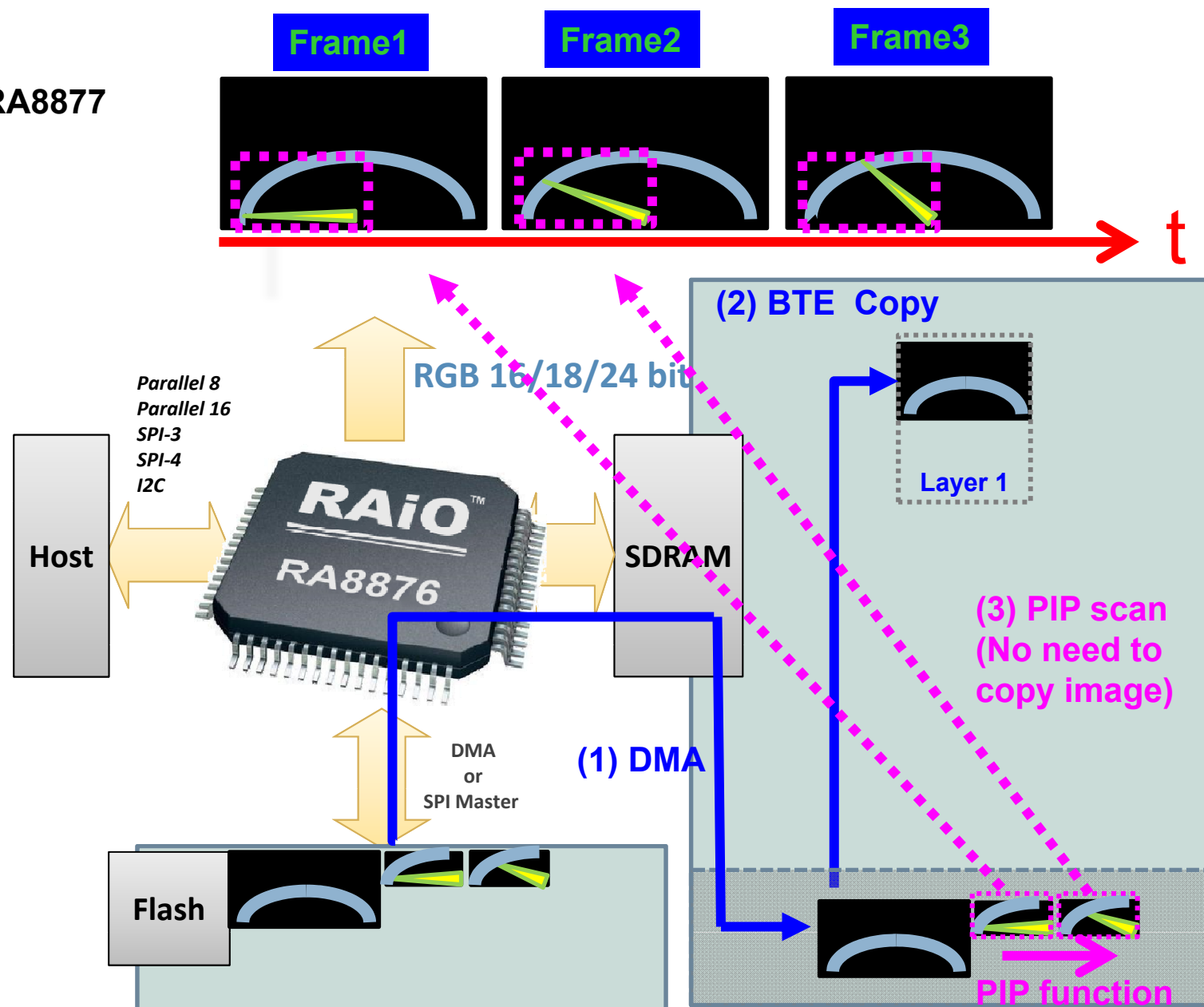
Schematic diagram 5

■ : by MCU
■ : by RA8876/RA8877



Schematic diagram 6

■ : by MCU
■ : by RA8876/RA8877



END