1. LT768_LCD

void SPIInit();
void SPI_CmdWrite(int cmd);
void SPI_DataWrite(int data);
void SPI_DataWrite_Pixel(int data);
int SPI_StatusRead(void);
int SPI_DataRead(void);
void SPI_RegisterWrite(unsigned char Cmd,unsigned char Data);
unsigned char SPI_RegisterRead(unsigned char Cmd);
void IICInit();
void IIC_CmdWrite(int cmd);
void IIC_DataWrite(int data);
void IIC_DataWrite_Pixel(int data);
int IIC_StatusRead(void);
int IIC_DataRead(void);
void IIC_RegisterWrite(unsigned char Cmd,unsigned char Data);
unsigned char IIC_RegisterRead(unsigned char Cmd);
void Parallel_Init(void);
void LCD_CmdWrite(u8 cmd);
void LCD_DataWrite(u8 data);
void LCD_DataWrite_Pixel(u16 data);
u8 LCD_StatusRead(void);
u16 LCD_DataRead(void);
void LCD_RegisterWrite(unsigned char Cmd,unsigned char Data);
unsigned char LCD_RegisterRead(unsigned char Cmd);

2. LT768

//**Staus**//
void Check_Mem_WR_FIFO_not_Full(void);
void Check_Mem_WR_FIFO_Empty(void);
void Check_Mem_RD_FIFO_not_Full(void);
void Check_Mem_RD_FIFO_not_Empty(void);
void Check_2D_Busy(void);
void Check_SDRAM_Ready(void);
unsigned char Power_Saving_Status(void);
void Check_Power_is_Normal(void);
void Check_Power_is_Saving(void);
void Check_NO_Interrupt(void);

```
void Check_Interrupt_Occur(void);

void Check_Busy_Draw(void);

//**[00h]**//
void LT768_SW_Reset(void);
//**[01h]**//
void Enable_PLL(void);
void LT768_Sleep(void);
void LT768_WakeUp(void);
void TFT_24bit(void);
void TFT_18bit(void);
void TFT_16bit(void);
void TFT_LVDS(void);
void Key_Scan_Enable(void);
void Key_Scan_Disable(void);
void LT768_I2CM_Enable(void);
void LT768_I2CM_Disable(void);
void Enable_SFlash_SPI(void);
void Disable_SFlash_SPI(void);
void Host_Bus_8bit(void);
void Host_Bus_16bit(void);
//**[02h]**//

void RGB_8b_8bpp(void);
void RGB_8b_16bpp(void);
void RGB_8b_24bpp(void);

void RGB_16b_8bpp(void);
void RGB_16b_16bpp(void);
void RGB_16b_24bpp_mode1(void);
void RGB_16b_24bpp_mode2(void);

void MemRead_Left_Right_Top_Down(void);
void MemRead_Right_Left_Top_Down(void);
void MemRead_Top_Down_Left_Right(void);
void MemRead_Down_Top_Left_Right(void);

void MemWrite_Left_Right_Top_Down(void);
void MemWrite_Right_Left_Top_Down(void);
void MemWrite_Top_Down_Left_Right(void);
void MemWrite_Down_Top_Left_Right(void);
//**[03h]**//
void Interrupt_Active_Low(void);
```

```
void Interrupt_Active_High(void);
void ExtInterrupt_Debounce(void);
void ExtInterrupt_Nodebounce(void);
void ExtInterrupt_Input_Low_Level_Trigger(void);
void ExtInterrupt_Input_High_Level_Trigger(void);
void ExtInterrupt_Input_Falling_Edge_Trigger(void);
void ExtInterrupt_Input_Rising_Edge_Trigger(void);
void LVDS_Format1(void);
void LVDS_Format2(void);
void Graphic_Mode(void);
void Text_Mode(void);
void Memory_Select_SDRAM(void);
void Memory_Select_Graphic_Cursor_RAM(void);
void Memory_Select_Color_Palette_RAM(void);
//**[05h]**//
//**[06h]**//
//**[07h]**//
//**[09h]**//
//**[0Ah]**//
//**[0Bh]**//
void Enable_Resume_Interrupt(void);
void Disable_Resume_Interrupt(void);
void Enable_ExtInterrupt_Input(void);
void Enable_ExtInterrupt_Inpur_Flag(void);
void Disable_ExtInterrupt_Input(void);
void Enable_I2CM_Interrupt(void);
void Disable_I2CM_Interrupt(void);
void Enable_Vsync_Interrupt(void);
void Disable_Vsync_Interrupt(void);
void Enable_KeyScan_Interrupt(void);
void Disable_KeyScan_Interrupt(void);
void Enable_DMA_Draw_BTE_Interrupt(void);
void Disable_DMA_Draw_BTE_Interrupt(void);
void Enable_PWM1_Interrupt(void);
void Disable_PWM1_Interrupt(void);
void Enable_PWM0_Interrupt(void);
void Disable_PWM0_Interrupt(void);
//**[0Ch]**//
unsigned char Read_Interrupt_status(void);
void Clear_Resume_Interrupt_Flag(void);
void Clear_ExtInterrupt_Input_Flag(void);
void Clear_I2CM_Interrupt_Flag(void);
void Clear_Vsync_Interrupt_Flag(void);
void Clear_KeyScan_Interrupt_Flag(void);
```

```
void Clear_DMA_Draw_BTE_Interrupt_Flag(void);
void Clear_PWM1_Interrupt_Flag(void);
void Clear_PWM0_Interrupt_Flag(void);
//**[0Dh]**//
void Mask_Resume_Interrupt_Flag(void);
void Mask_ExtInterrupt_Input_Flag(void);
void Mask_I2CM_Interrupt_Flag(void);
void Mask_Vsync_Interrupt_Flag(void);
void Mask_KeyScan_Interrupt_Flag(void);
void Mask_DMA_Draw_BTE_Interrupt_Flag(void);
void Mask_PWM1_Interrupt_Flag(void);
void Mask_PWM0_Interrupt_Flag(void);
//
void Enable_Resume_Interrupt_Flag(void);
void Enable_ExtInterrupt_Input_Flag(void);
void Enable_I2CM_Interrupt_Flag(void);
void Enable_Vsync_Interrupt_Flag(void);
void Enable_KeyScan_Interrupt_Flag(void);
void Enable_DMA_Draw_BTE_Interrupt_Flag(void);
void Enable_PWM1_Interrupt_Flag(void);
void Enable_PWM0_Interrupt_Flag(void);
//**[0Eh]**//
void Enable_GPIOF_PullUp(void);
void Enable_GPIOE_PullUp(void);
void Enable_GPIOD_PullUp(void);
void Enable_GPIOC_PullUp(void);
void Enable_XDB15_8_PullUp(void);
void Enable_XDB7_0_PullUp(void);
void Disable_GPIOF_PullUp(void);
void Disable_GPIOE_PullUp(void);
void Disable_GPIOD_PullUp(void);
void Disable_GPIOC_PullUp(void);
void Disable_XDB15_8_PullUp(void);
void Disable_XDB7_0_PullUp(void);

//**[0Fh]**//
void XPDAT18_Set_GPIO_D7(void);
void XPDAT18_Set_KOUT4(void);
void XPDAT17_Set_GPIO_D5(void);
void XPDAT17_Set_KOUT2(void);
void XPDAT16_Set_GPIO_D4(void);
void XPDAT16_Set_KOUT1(void);
void XPDAT9_Set_GPIO_D3(void);
void XPDAT9_Set_KOUT3(void);
```

```
void XPDAT8_Set_GPIO_D2(void);
void XPDAT8_Set_KIN3(void);
void XPDAT2_Set_GPIO_D6(void);
void XPDAT2_Set_KIN4(void);
void XPDAT1_Set_GPIO_D1(void);
void XPDAT1_Set_KIN2(void);
void XPDAT0_Set_GPIO_D0(void);
void XPDAT0_Set_KIN1(void);

//**[10h]**//
void Enable_PIP1(void);
void Disable_PIP1(void);
void Enable_PIP2(void);
void Disable_PIP2(void);
void Select_PIP1_Parameter(void);
void Select_PIP2_Parameter(void);
void Select_Main_Window_8bpp(void);
void Select_Main_Window_16bpp(void);
void Select_Main_Window_24bpp(void);
//**[11h]**//
void Select_PIP1_Window_8bpp(void);
void Select_PIP1_Window_16bpp(void);
void Select_PIP1_Window_24bpp(void);
void Select_PIP2_Window_8bpp(void);
void Select_PIP2_Window_16bpp(void);
void Select_PIP2_Window_24bpp(void);
//**[12h]**//
void PCLK_Rising(void);
void PCLK_Falling(void);
void Display_ON(void);
void Display_OFF(void);
void Color_Bar_ON(void);
void Color_Bar_OFF(void);
void HSCAN_L_to_R(void);
void HSCAN_R_to_L(void);
void VSCAN_T_to_B(void);
void VSCAN_B_to_T(void);
void PDATA_Set_RGB(void);
void PDATA_Set_RBG(void);
void PDATA_Set_GRB(void);
void PDATA_Set_GBR(void);
void PDATA_Set_BRG(void);
void PDATA_Set_BGR(void);
void PDATA_IDLE_STATE(void);
```

```
//**[13h]**//
void HSYNC_Low_Active(void);
void HSYNC_High_Active(void);
void VSYNC_Low_Active(void);
void VSYNC_High_Active(void);
void DE_Low_Active(void);
void DE_High_Active(void);
void Idle_DE_Low(void);
void Idle_DE_High(void);
void Idle_PCLK_Low(void);
void Idle_PCLK_High(void);
void Idle_PDAT_Low(void);
void Idle_PDAT_High(void);
void Idle_HSYNC_Low(void);
void Idle_HSYNC_High(void);
void Idle_VSYNC_Low(void);
void Idle_VSYNC_High(void);
//**[14h][15h][1Ah][1Bh]**//
void LCD_HorizontalWidth_VerticalHeight(unsigned short WX,unsigned short HY);
//**[16h][17h]**//
void LCD_Horizontal_Non_Display(unsigned short WX);
//**[18h]**//
void LCD_HSYNC_Start_Position(unsigned short WX);
//**[19h]**//
void LCD_HSYNC_Pulse_Width(unsigned short WX);
//**[1Ch][1Dh]**//
void LCD_Vertical_Non_Display(unsigned short HY);
//**[1Eh]**//
void LCD_VSYNC_Start_Position(unsigned short HY);
//**[1Fh]**//
void LCD_VSYNC_Pulse_Width(unsigned short HY);
//**[20h][21h][22h][23h]**//
void Main_Image_Start_Address(unsigned long Addr);
//**[24h][25h]**//
void Main_Image_Width(unsigned short WX);
//**[26h][27h][28h][29h]**//
void Main_Window_Start_XY(unsigned short WX,unsigned short HY);
//**[2Ah][2Bh][2Ch][2Dh]**//
void PIP_Display_Start_XY(unsigned short WX,unsigned short HY);
//**[2Eh][2Fh][30h][31h]**//
void PIP_Image_Start_Address(unsigned long Addr);
//**[32h][33h]**//
void PIP_Image_Width(unsigned short WX);
```

```
//**[34h][35h][36h][37h]**//
void PIP_Window_Image_Start_XY(unsigned short WX,unsigned short HY);
//**[38h][39h][3Ah][3Bh]**//
void PIP_Window_Width_Height(unsigned short WX,unsigned short HY);
//**[3C]**//
void Enable_Graphic_Cursor(void);
void Disable_Graphic_Cursor(void);
void Select_Graphic_Cursor_1(void);
void Select_Graphic_Cursor_2(void);
void Select_Graphic_Cursor_3(void);
void Select_Graphic_Cursor_4(void);
void Enable_Text_Cursor(void);
void Disable_Text_Cursor(void);
void Enable_Text_Cursor_Blinking(void);
void Disable_Text_Cursor_Blinking(void);
//**[3D]**//
void Blinking_Time_Frames(unsigned char temp);
//**[3E][3Fh]**//
void Text_Cursor_H_V(unsigned short WX,unsigned short HY);
//**[40h][41h][42h][43h]**//
void Graphic_Cursor_XY(unsigned short WX,unsigned short HY);
//**[44]**//
void Set_Graphic_Cursor_Color_1(unsigned char temp);
//**[45]**//
void Set_Graphic_Cursor_Color_2(unsigned char temp);
//**[50h][51h][52h][53h]**//
void Canvas_Image_Start_address(unsigned long Addr);
//**[54h][55h]**//
void Canvas_image_width(unsigned short WX);
//**[56h][57h][58h][59h]**//
void Active_Window_XY(unsigned short WX,unsigned short HY);
//**[5Ah][5Bh][5Ch][5Dh]**//
void Active_Window_WH(unsigned short WX,unsigned short HY);
//**[5E]**//
void Select_Write_Data_Position(void);
void Select_Read_Data_Position(void);
void Memory_XY_Mode(void);
void Memory_Linear_Mode(void);
void Memory_8bpp_Mode(void);
void Memory_16bpp_Mode(void);
void Memory_24bpp_Mode(void);
//**[5Fh][60h][61h][62h]**//
void Goto_Pixel_XY(unsigned short WX,unsigned short HY);
void Goto_Linear_Addr(unsigned long Addr);
```

```
//**[63h][64h][65h][66h]**//
void Goto_Text_XY(unsigned short WX,unsigned short HY);

//////////////////////////////////////////////////////////////////////
////**** [ Function : Draw ] ****////
//**[67h]**//
void Start_Line(void);
void Start_Triangle(void);
void Start_Triangle_Fill(void);
//**[68h]~[73h]**//
void Line_Start_XY(unsigned short WX,unsigned short HY);
void Line_End_XY(unsigned short WX,unsigned short HY);
void Triangle_Point1_XY(unsigned short WX,unsigned short HY);
void Triangle_Point2_XY(unsigned short WX,unsigned short HY);
void Triangle_Point3_XY (unsigned short WX,unsigned short HY);
void Square_Start_XY(unsigned short WX,unsigned short HY);
void Square_End_XY(unsigned short WX,unsigned short HY);
//**[76h]**//
void Start_Circle_or_Ellipse(void);
void Start_Circle_or_Ellipse_Fill(void);
void Start_Left_Down_Curve(void);
void Start_Left_Up_Curve(void);
void Start_Right_Up_Curve(void);
void Start_Right_Down_Curve(void);
void Start_Left_Down_Curve_Fill(void);
void Start_Left_Up_Curve_Fill(void);
void Start_Right_Up_Curve_Fill(void);
void Start_Right_Down_Curve_Fill(void);
void Start_Square(void);
void Start_Square_Fill(void);
void Start_Circle_Square(void);
void Start_Circle_Square_Fill(void);
//**[77h]~[7Eh]**//
void Circle_Center_XY(unsigned short WX,unsigned short HY);
void Ellipse_Center_XY(unsigned short WX,unsigned short HY);
void Circle_Radius_R(unsigned short WX);
void Ellipse_Radius_RxRy(unsigned short WX,unsigned short HY);
void Circle_Square_Radius_RxRy(unsigned short WX,unsigned short HY);

//////////////////////////////////////////////////////////////////////
////**** [ Function : PWM ] ****////
//**[84h]**//
void Set_PWM_Prescaler_1_to_256(unsigned short WX);
//**[85h]**//
```

```
void Select_PWM1_Clock_Divided_By_1(void);
void Select_PWM1_Clock_Divided_By_2(void);
void Select_PWM1_Clock_Divided_By_4(void);
void Select_PWM1_Clock_Divided_By_8(void);
void Select_PWM0_Clock_Divided_By_1(void);
void Select_PWM0_Clock_Divided_By_2(void);
void Select_PWM0_Clock_Divided_By_4(void);
void Select_PWM0_Clock_Divided_By_8(void);
//[85h].[bit3][bit2]
void Select_PWM1_is_ErrorFlag(void);
void Select_PWM1(void);
void Select_PWM1_is_Osc_Clock(void);
//[85h].[bit1][bit0]
void Select_PWM0_is_GPIO_C7(void);
void Select_PWM0(void);
void Select_PWM0_is_Core_Clock(void);
//**[86h]**//
//[86h]PWM1
void Enable_PWM1_Inverter(void);
void Disable_PWM1_Inverter(void);
void Auto_Reload_PWM1(void);
void One_Shot_PWM1(void);
void Start_PWM1(void);
void Stop_PWM1(void);
//[86h]PWM0
void Enable_PWM0_Dead_Zone(void);
void Disable_PWM0_Dead_Zone(void);
void Enable_PWM0_Inverter(void);
void Disable_PWM0_Inverter(void);
void Auto_Reload_PWM0(void);
void One_Shot_PWM0(void);
void Start_PWM0(void);
void Stop_PWM0(void);
//**[87h]**//
void Set_Timer0_Dead_Zone_Length(unsigned char temp);
//**[88h][89h]**//
void Set_Timer0_Compare_Buffer(unsigned short WX);
//**[8Ah][8Bh]**//
void Set_Timer0_Count_Buffer(unsigned short WX);
//**[8Ch][8Dh]**//
void Set_Timer1_Compare_Buffer(unsigned short WX);
//**[8Eh][8Fh]**//
void Set_Timer1_Count_Buffer(unsigned short WX);
```

```
//////////////////////////////////////////////////////////////////////
////**** [ Function : BTE ] ****////
//**[90h]**//

//[90h]=====================================================
void BTE_Enable(void);
void BTE_Disable(void);

void Check_BTE_Busy(void);

void Pattern_Format_8X8(void);
void Pattern_Format_16X16(void);

//[91h]=====================================================
void BTE_ROP_Code(unsigned char setx);
void BTE_Operation_Code(unsigned char setx);

//[92h]=====================================================
void BTE_S0_Color_8bpp(void);
void BTE_S0_Color_16bpp(void);
void BTE_S0_Color_24bpp(void);

void BTE_S1_Color_8bpp(void);
void BTE_S1_Color_16bpp(void);
void BTE_S1_Color_24bpp(void);
void BTE_S1_Color_Constant(void);
void BTE_S1_Color_8bit_Alpha(void);
void BTE_S1_Color_16bit_Alpha(void);

void BTE_Destination_Color_8bpp(void);
void BTE_Destination_Color_16bpp(void);
void BTE_Destination_Color_24bpp(void);

//[93h][94h][95h][96h]======================================
void BTE_S0_Memory_Start_Address(unsigned long Addr);

//[97h][98h]================================================
void BTE_S0_Image_Width(unsigned short WX);

//[99h][9Ah][9Bh][9Ch]======================================
void BTE_S0_Window_Start_XY(unsigned short WX,unsigned short HY);

//[9Dh][9Eh][9Fh][A0h]======================================
void BTE_S1_Memory_Start_Address(unsigned long Addr);
```

```
void S1_Constant_color_256(unsigned char temp);
void S1_Constant_color_65k(unsigned short temp);
void S1_Constant_color_16M(unsigned long temp);

//[A1h][A2h]=================================================
void BTE_S1_Image_Width(unsigned short WX);

//[A3h][A4h][A5h][A6h]=======================================
void BTE_S1_Window_Start_XY(unsigned short WX,unsigned short HY);

//[A7h][A8h][A9h][AAh]=======================================
void BTE_Destination_Memory_Start_Address(unsigned long Addr);

//[ABh][ACh]=================================================
void BTE_Destination_Image_Width(unsigned short WX);

//[ADh][AEh][AFh][B0h]=======================================
void BTE_Destination_Window_Start_XY(unsigned short WX,unsigned short HY);

//[B1h][B2h][B3h][B4h]=======================================
void BTE_Window_Size(unsigned short WX, unsigned short WY);

//[B5h]=======================================================
void BTE_Alpha_Blending_Effect(unsigned char temp);


//**[B5h]**//

//////////////////////////////////////////////////////////////////////////
////**** [ Function : Serial Flash ] ****////


//REG[B6h] Serial flash DMA Controller REG (DMA_CTRL)
void Start_SFI_DMA(void);
void Check_Busy_SFI_DMA(void);

//REG[B7h] Serial Flash/ROM Controller Register (SFL_CTRL)
void Select_SFI_0(void);
void Select_SFI_1(void);
void Select_SFI_Font_Mode(void);
void Select_SFI_DMA_Mode(void);
void Select_SFI_24bit_Address(void);
void Select_SFI_32bit_Address(void);
void Select_SFI_Waveform_Mode_0(void);
```

```
void Select_SFI_Waveform_Mode_3(void);
void Select_SFI_0_DummyRead(void);
void Select_SFI_8_DummyRead(void);
void Select_SFI_16_DummyRead(void);
void Select_SFI_24_DummyRead(void);
void Select_SFI_Single_Mode(void);
void Select_SFI_Dual_Mode0(void);
void Select_SFI_Dual_Mode1(void);

//REG[B8h] SPI master Tx /Rx FIFO Data Register (SPIDR)
unsigned char SPI_Master_FIFO_Data_Put(unsigned char Data);
unsigned char SPI_Master_FIFO_Data_Get(void);

//REG[B9h] SPI master Control Register (SPIMCR2)
void Mask_SPI_Master_Interrupt_Flag(void);
void Select_nSS_drive_on_xnsfcs0(void);
void Select_nSS_drive_on_xnsfcs1(void);
void nSS_Inactive(void);
void nSS_Active(void);
void OVFIRQEN_Enable(void);
void EMTIRQEN_Enable(void);
void Reset_CPOL(void);
void Set_CPOL(void);
void Reset_CPHA(void);
void Set_CPHA(void);

//REG[BAh] SPI master Status Register (SPIMSR)
unsigned char Tx_FIFO_Empty_Flag(void);
unsigned char Tx_FIFO_Full_Flag(void);
unsigned char Rx_FIFO_Empty_Flag(void);
unsigned char Rx_FIFO_full_flag(void);
unsigned char OVFI_Flag(void);
void Clear_OVFI_Flag(void);
unsigned char EMTI_Flag(void);
void Clear_EMTI_Flag(void);

//REG[BB] SPI Clock period (SPIDIV)
void SPI_Clock_Period(unsigned char temp);


//**[BCh][BDh][BEh][BFh]**//
void SFI_DMA_Source_Start_Address(unsigned long Addr);
//**[C0h][C1h][C2h][C3h]**//
void SFI_DMA_Destination_Start_Address(unsigned long Addr);
```

```
void SFI_DMA_Destination_Upper_Left_Corner(unsigned short WX,unsigned short HY);
//**[C4h][C5h]**//
void SFI_DMA_Destination_Width(unsigned short WX);
//**[C6h][C7h][C8h][C9h]**//
void SFI_DMA_Transfer_Number(unsigned long Addr);
void SFI_DMA_Transfer_Width_Height(unsigned short WX,unsigned short HY);
//**[CAh][CBh]**//
void SFI_DMA_Source_Width(unsigned short WX);

//////////////////////////////////////////////////////////////////////
////**** [ Function : Font ] ****////
//**[CCh]**//

void Font_Select_UserDefine_Mode(void);
void CGROM_Select_Internal_CGROM(void);
void CGROM_Select_Genitop_FontROM(void);
void Font_Select_8x16_16x16(void);
void Font_Select_12x24_24x24(void);
void Font_Select_16x32_32x32(void);
void Internal_CGROM_Select_ISOIEC8859_1(void);
void Internal_CGROM_Select_ISOIEC8859_2(void);
void Internal_CGROM_Select_ISOIEC8859_3(void);
void Internal_CGROM_Select_ISOIEC8859_4(void);
//**[CDh]**//
void Enable_Font_Alignment(void);
void Disable_Font_Alignment(void);
void Font_Background_select_Transparency(void);
void Font_Background_select_Color(void);
void Font_0_degree(void);
void Font_90_degree(void);
void Font_Width_X1(void);
void Font_Width_X2(void);
void Font_Width_X3(void);
void Font_Width_X4(void);
void Font_Height_X1(void);
void Font_Height_X2(void);
void Font_Height_X3(void);
void Font_Height_X4(void);
//**[CEh]**//
void GTFont_Select_GT21L16TW_GT21H16T1W(void);
void GTFont_Select_GT23L16U2W(void);
void GTFont_Select_GT23L24T3Y_GT23H24T3Y(void);
void GTFont_Select_GT23L24M1Z(void);
void GTFont_Select_GT23L32S4W_GT23H32S4W(void);
```

```
void GTFont_Select_GT20L24F6Y(void);
void GTFont_Select_GT21L24S1W(void);
void GTFont_Select_GT22L16A1Y(void);
//**[CFh]**//
void Set_GTFont_Decoder(unsigned char temp);
//**[D0h]**//
void Font_Line_Distance(unsigned char temp);
//**[D1h]**//
void Set_Font_to_Font_Width(unsigned char temp);
//**[D2h]~[D4h]**//
void Foreground_RGB(unsigned char RED,unsigned char GREEN,unsigned char BLUE);
void Foreground_color_256(unsigned char temp);
void Foreground_color_65k(unsigned short temp);
void Foreground_color_16M(unsigned long temp);
//**[D5h]~[D7h]**//
void Background_RGB(unsigned char RED,unsigned char GREEN,unsigned char BLUE);
void Background_color_256(unsigned char temp);
void Background_color_65k(unsigned short temp);
void Background_color_16M(unsigned long temp);
//**[DBh]~[DEh]**//
void CGRAM_Start_address(unsigned long Addr);
//**[DFh]**//
void Power_Normal_Mode(void);
void Power_Saving_Standby_Mode(void);
void Power_Saving_Suspend_Mode(void);
void Power_Saving_Sleep_Mode(void);

//////////////////////////////////////////////////////////////////////////
////**** [ Function : SDRAM ] ****////
//**[E0h]~[E4h]**//
// LT768_Select_SDRAM();


//////////////////////////////////////////////////////////////////////////
////**** [ Function : I2C ] ****////
//**[E5h]~[EAh]**//
void LT768_I2CM_Clock_Prescale(unsigned short WX);
//**[E7h]**//
void LT768_I2CM_Transmit_Data(unsigned char temp);
//**[E8h]**//
unsigned char LT768_I2CM_Receiver_Data(void);
//**[E9h]**//

void LT768_I2CM_Read_With_Ack(void);
```

```
void LT768_I2CM_Read_With_Nack(void);
void LT768_I2CM_Write_With_Start(void);
void LT768_I2CM_Write(void);
void LT768_I2CM_Stop(void);


//**[EAh]**//
unsigned char LT768_I2CM_Check_Slave_ACK(void);
unsigned char LT768_I2CM_Bus_Busy(void);
unsigned char LT768_I2CM_transmit_Progress(void);
unsigned char LT768_I2CM_Arbitration(void);



//////////////////////////////////////////////////////////////////////////
////**** [ Function : GPIO ] ****////
//[F0h][F1h]
void Set_GPIO_A_In_Out(unsigned char temp);
void Write_GPIO_A_7_0(unsigned char temp);
unsigned char Read_GPIO_A_7_0(void);
//[F2h]
void Write_GPIO_B_7_4(unsigned char temp);
unsigned char Read_GPIO_B_7_0(void);
//[F3h][F4h]
void Set_GPIO_C_In_Out(unsigned char temp);
void Write_GPIO_C_7_0(unsigned char temp);
unsigned char Read_GPIO_C_7_0(void);
//[F5h][F6h]
void Set_GPIO_D_In_Out(unsigned char temp);
void Write_GPIO_D_7_0(unsigned char temp);
unsigned char Read_GPIO_D_7_0(void);
//[F7h][F8h]
void Set_GPIO_E_In_Out(unsigned char temp);
void Write_GPIO_E_7_0(unsigned char temp);
unsigned char Read_GPIO_E_7_0(void);
//[F9h][FAh]
void Set_GPIO_F_In_Out(unsigned char temp);
void Write_GPIO_F_7_0(unsigned char temp);
unsigned char Read_GPIO_F_7_0(void);


//////////////////////////////////////////////////////////////////////////
////**** [ Function : Key ] ****////
//**[FBh]~[FFh]**//
```

```
//[FBh]
void Long_Key_enable(void);
void Key_Scan_Freg(unsigned char temp);    //set bit2~0

//[FCh]
void Key_Scan_Wakeup_Function_Enable(void);
void Long_Key_Timing_Adjustment(unsigned char setx);//set bit5~3
unsigned char Numbers_of_Key_Hit(void);

//[FDh][FEh][FFh]
unsigned char Read_Key_Strobe_Data_0(void);
unsigned char Read_Key_Strobe_Data_1(void);
unsigned char Read_Key_Strobe_Data_2(void);

void Show_String(char *str);
void Show_picture(unsigned long numbers,const unsigned short *data);

void PWM0_ON(void);       //开 PWM0

//void LT768_HW_Reset(void);
//void System_Check_Temp(void);
void Enable_SPI_Flash_DMA(unsigned char val);
```

3. LT768_Lib

```
void Set_PCLK(u8 val);
void Set_HSYNC_Active(u8 val);
void Set_VSYNC_Active(u8 val);
void Set_DE_Active(u8 val);
void LT768_HW_Reset(void);
void System_Check_Temp(void);
void LT768_PLL_Initial(void);
void LT768_SDRAM_initail(unsigned char mclk);
void Set_LCD_Panel(void);
void LT768_initial(void);
void LT768_Init(void);
void MPU8_8bpp_Memory_Write
(
unsigned short x //x of coordinate
,unsigned short y // y of coordinate
,unsigned short w //width
,unsigned short h //height
```

```
,const unsigned char *data //8bit data
);
            void MPU8_16bpp_Memory_Write
(
  unsigned short x // x of coordinate
,unsigned short y // y of coordinate
,unsigned short w // width
,unsigned short h // height
,const unsigned char *data // 8bit data
);
            void MPU8_24bpp_Memory_Write
(
unsigned short x // x of coordinate
,unsigned short y // y of coordinate
,unsigned short w // width
,unsigned short h // height
,const unsigned char *data // 8bit data
);
            void MPU16_16bpp_Memory_Write
(
  unsigned short x          //x of coordinate
,unsigned short y                  //y of coordinate
,unsigned short w                   //width
,unsigned short h                   //height
,const unsigned short *data      //16bit data
);
            void MPU16_24bpp_Mode1_Memory_Write
(
unsigned short x //x of coordinate
,unsigned short y //y of coordinate
,unsigned short w //width
,unsigned short h //height
,const unsigned short *data //16bit data
);
            void MPU16_24bpp_Mode2_Memory_Write
(
unsigned short x //x of coordinate
,unsigned short y //y of coordinate
,unsigned short w //width
,unsigned short h //height
,const unsigned short *data //16bit data
);
            void LT768_DrawLine
(
```

```
 unsigned short X1
,unsigned short Y1
,unsigned short X2
,unsigned short Y2
,unsigned long    LineColor
);
            void LT768_DrawLine_Width
(
 unsigned short X1
,unsigned short Y1
,unsigned short X2
,unsigned short Y2
,unsigned long    LineColor
,unsigned short Width
);
            void LT768_DrawCircle
(
 unsigned short XCenter
,unsigned short YCenter
,unsigned short R
,unsigned long CircleColor
);
            void LT768_DrawCircle_Fill
(
 unsigned short XCenter
,unsigned short YCenter
,unsigned short R
,unsigned long ForegroundColor
);
            void LT768_DrawCircle_Width
(
 unsigned short XCenter
,unsigned short YCenter
,unsigned short R
,unsigned long CircleColor
,unsigned long ForegroundColor
,unsigned short Width
);
            void LT768_DrawEllipse
(
 unsigned short XCenter
,unsigned short YCenter
,unsigned short X_R
,unsigned short Y_R
```

```
,unsigned long EllipseColor
);
            void LT768_DrawEllipse_Fill
(
  unsigned short XCenter
,unsigned short YCenter
,unsigned short X_R
,unsigned short Y_R
,unsigned long ForegroundColor
);
            void LT768_DrawEllipse_Width
(
  unsigned short XCenter
,unsigned short YCenter
,unsigned short X_R
,unsigned short Y_R
,unsigned long EllipseColor
,unsigned long ForegroundColor
,unsigned short Width
);
            void LT768_DrawSquare
(
  unsigned short X1
,unsigned short Y1
,unsigned short X2
,unsigned short Y2
,unsigned long SquareColor
);
            void LT768_DrawSquare_Fill
(
  unsigned short X1
,unsigned short Y1
,unsigned short X2
,unsigned short Y2
,unsigned long ForegroundColor
);
            void LT768_DrawSquare_Width
(
  unsigned short X1
,unsigned short Y1
,unsigned short X2
,unsigned short Y2
,unsigned long SquareColor
,unsigned long ForegroundColor
```

```
,unsigned short Width
);
        void LT768_DrawCircleSquare
(
  unsigned short X1
,unsigned short Y1
,unsigned short X2
,unsigned short Y2
,unsigned short X_R
,unsigned short Y_R
,unsigned long CircleSquareColor
);
        void LT768_DrawCircleSquare_Fill
(
  unsigned short X1
,unsigned short Y1
,unsigned short X2
,unsigned short Y2
,unsigned short X_R
,unsigned short Y_R
,unsigned long ForegroundColor
);
        void LT768_DrawCircleSquare_Width
(
  unsigned short X1
,unsigned short Y1
,unsigned short X2
,unsigned short Y2
,unsigned short X_R
,unsigned short Y_R
,unsigned long CircleSquareColor
,unsigned long ForegroundColor
,unsigned short Width
);
        void LT768_DrawTriangle
(
  unsigned short X1
,unsigned short Y1
,unsigned short X2
,unsigned short Y2
,unsigned short X3
,unsigned short Y3
,unsigned long TriangleColor
);
```

```
        void LT768_DrawTriangle_Fill
(
  unsigned short X1
,unsigned short Y1
,unsigned short X2
,unsigned short Y2
,unsigned short X3
,unsigned short Y3
,unsigned long ForegroundColor
);
        void LT768_DrawLeftUpCurve
(
  unsigned short XCenter
,unsigned short YCenter
,unsigned short X_R
,unsigned short Y_R
,unsigned long CurveColor
);
        void LT768_DrawLeftDownCurve
(
  unsigned short XCenter
,unsigned short YCenter
,unsigned short X_R
,unsigned short Y_R
,unsigned long CurveColor
);
        void LT768_DrawRightUpCurve
(
  unsigned short XCenter
,unsigned short YCenter
,unsigned short X_R
,unsigned short Y_R
,unsigned long CurveColor
);
        void LT768_DrawRightDownCurve
(
  unsigned short XCenter
,unsigned short YCenter
,unsigned short X_R
,unsigned short Y_R
,unsigned long CurveColor
);
        void LT768_SelectDrawCurve
(
```

```
 unsigned short XCenter
,unsigned short YCenter
,unsigned short X_R
,unsigned short Y_R
,unsigned long CurveColor
,unsigned short    Dir
);
            void LT768_DrawLeftUpCurve_Fill
(
 unsigned short XCenter
,unsigned short YCenter
,unsigned short X_R
,unsigned short Y_R
,unsigned long ForegroundColor
);
            void LT768_DrawLeftDownCurve_Fill
(
 unsigned short XCenter
,unsigned short YCenter
,unsigned short X_R
,unsigned short Y_R
,unsigned long ForegroundColor
);
            void LT768_DrawRightUpCurve_Fill
(
 unsigned short XCenter
,unsigned short YCenter
,unsigned short X_R
,unsigned short Y_R
,unsigned long ForegroundColor
);
            void LT768_DrawRightDownCurve_Fill
(
 unsigned short XCenter
,unsigned short YCenter
,unsigned short X_R
,unsigned short Y_R
,unsigned long ForegroundColor
);
            void LT768_SelectDrawCurve_Fill
(
 unsigned short XCenter
,unsigned short YCenter
,unsigned short X_R
```

```
,unsigned short Y_R
,unsigned long CurveColor
,unsigned short    Dir
);
            void LT768_DrawQuadrilateral
(
 unsigned short X1
,unsigned short Y1
,unsigned short X2
,unsigned short Y2
,unsigned short X3
,unsigned short Y3
,unsigned short X4
,unsigned short Y4
,unsigned long ForegroundColor
);
            void LT768_DrawQuadrilateral_Fill
(
 unsigned short X1
,unsigned short Y1
,unsigned short X2
,unsigned short Y2
,unsigned short X3
,unsigned short Y3
,unsigned short X4
,unsigned short Y4
,unsigned long ForegroundColor
);
            void LT768_DrawPentagon
(
 unsigned short X1
,unsigned short Y1
,unsigned short X2
,unsigned short Y2
,unsigned short X3
,unsigned short Y3
,unsigned short X4
,unsigned short Y4
,unsigned short X5
,unsigned short Y5
,unsigned long ForegroundColor
);
            void LT768_DrawPentagon_Fill
(
```

```
  unsigned short X1
,unsigned short Y1
,unsigned short X2
,unsigned short Y2
,unsigned short X3
,unsigned short Y3
,unsigned short X4
,unsigned short Y4
,unsigned short X5
,unsigned short Y5
,unsigned long ForegroundColor
);
            unsigned char LT768_DrawCylinder
(
  unsigned short XCenter
,unsigned short YCenter
,unsigned short X_R
,unsigned short Y_R
,unsigned short H
,unsigned long CylinderColor
,unsigned long ForegroundColor
);
            void LT768_DrawQuadrangular
(
  unsigned short X1
,unsigned short Y1
,unsigned short X2
,unsigned short Y2
,unsigned short X3
,unsigned short Y3
,unsigned short X4
,unsigned short Y4
,unsigned short X5
,unsigned short Y5
,unsigned short X6
,unsigned short Y6
,unsigned long QuadrangularColor
,unsigned long ForegroundColor
);
            void LT768_MakeTable
(
   unsigned short X1,
   unsigned short Y1,
   unsigned short W,
```

```
    unsigned short H,
    unsigned short Line,
    unsigned short Row,
    unsigned long    TableColor,
    unsigned long    ItemColor,
    unsigned long    ForegroundColor,
    unsigned short width1,
    unsigned short width2,
    unsigned char    mode
);
            void LT768_Color_Bar_ON(void);
            void LT768_Color_Bar_OFF(void);
            void LT768_DMA_24bit_Linear
(
  unsigned char SCS                  // Select SPI : SCS ： 0          SCS ： 1
,unsigned char Clk                   // SPI Clock = System Clock /{(Clk+1);*2}
,unsigned long flash_addr           // Data flash address
,unsigned long memory_addr          // Data transfers to SDRAM address
,unsigned long data_num             // Data size
);
            void LT768_DMA_32bit_Linear
(
  unsigned char SCS                  // Select SPI : SCS ： 0          SCS ： 1
,unsigned char Clk                   // SPI Clock = System Clock /{(Clk+1);*2}
,unsigned long flash_addr           // Data flash address
,unsigned long memory_addr          // Data transfers to SDRAM address
,unsigned long data_num             // Data size
);
            void LT768_DMA_24bit_Block
(
  unsigned char SCS                 // Select SPI : SCS ： 0          SCS ： 1
,unsigned char Clk                  // SPI Clock = System Clock /{(Clk+1);*2}
,unsigned short X1                  // Transfer to SDRAM address:X1
,unsigned short Y1                  // Transfer to SDRAM address:Y1
,unsigned short X_W                 // DMA data width
,unsigned short Y_H                 // DMA data height
,unsigned short P_W                 // Picture's width
,unsigned long Addr                 // Flash address
);
            void LT768_DMA_32bit_Block
(
  unsigned char SCS                 // Select SPI : SCS ： 0          SCS ： 1
,unsigned char Clk                  // SPI Clock = System Clock /{(Clk+1);*2}
,unsigned short X1                  // Transfer to SDRAM address:X1
```

```
,unsigned short Y1              // Transfer to SDRAM address:Y1
,unsigned short X_W             // DMA data width
,unsigned short Y_H             // DMA data height
,unsigned short P_W             // Picture's width
,unsigned long Addr             // Flash address
);
        void LT768_Select_Internal_Font_Init
(
 unsigned char Size             // Setting font size   16：16*16      24:24*24      32:32*32
,unsigned char XxN              // Width magnification:1~4
,unsigned char YxN              // Heigtht magnification:1~4
,unsigned char ChromaKey        // 0：not transparent    1：transparent
,unsigned char Alignment        // 0：font not align       1：font align
);
        void LT768_Print_Internal_Font_String
(
 unsigned short x               // Font X coordinates
,unsigned short y               // Font Y coordinates
,unsigned long FontColor
,unsigned long BackGroundColor
,char *c                        // The first address of the data buffer
);
        void LT768_Select_Outside_Font_Init
(
 unsigned char SCS              // Select SPI : SCS：0       SCS：1
,unsigned char Clk              // SPI Clock = System Clock /{(Clk+1);*2}
,unsigned long FlashAddr
,unsigned long MemoryAddr
,unsigned long Num              // Font data Num
,unsigned char Size             // Font size   16：16*16      24:24*24      32:32*32
,unsigned char XxN              // Width magnification:1~4
,unsigned char YxN              // Heigtht magnification:1~4
,unsigned char ChromaKey        // 0：not transparent    1：transparent
,unsigned char Alignment        // 0：font not align       1：font align
);
        void LT768_Print_Outside_Font_String
(
 unsigned short x               // Font X coordinates
,unsigned short y               // Font Y coordinates
,unsigned long FontColor
,unsigned long BackGroundColor
,unsigned char *c                       // The first address of the data buffer
);
        void LT768_Text_cursor_Init
```

```
(
 unsigned char On_Off_Blinking
,unsigned short Blinking_Time
,unsigned short X_W                    // Horizontal size
,unsigned short Y_W                    // Vertical size
);
        void LT768_Enable_Text_Cursor(void);
        void LT768_Disable_Text_Cursor(void);
        void LT768_Graphic_cursor_Init
(
 unsigned char Cursor_N                // Select cursor : 1:cursor 1   2:cursor 2
3:cursor 3   4:cursor 4
,unsigned char Color1
,unsigned char Color2
,unsigned short X_Pos                  // X position
,unsigned short Y_Pos                  // Y position
,unsigned char *Cursor_Buf
);
        void LT768_Set_Graphic_cursor_Pos
(
 unsigned char Cursor_N                // Select cursor : 1:cursor 1   2:cursor 2
3:cursor 3   4:cursor 4
,unsigned short X_Pos                  // X position
,unsigned short Y_Pos                  // Y position
);
        void LT768_Enable_Graphic_Cursor(void);
        void LT768_Disable_Graphic_Cursor(void);
        void LT768_PIP_Init
(
 unsigned char On_Off          // 0 : forbid PIP    1 : enable PIP    2 : Keep the original
state
,unsigned char Select_PIP     // 1 : use PIP1   2 : use PIP2
,unsigned long PAddr          // PIP begin address
,unsigned short XP            // PIP X position,must can be divided by 4
,unsigned short YP            // PIP Y position,must can be divided by 4
,unsigned long ImageWidth
,unsigned short X_Dis         // Display window X position
,unsigned short Y_Dis         // Display window Y position
,unsigned short X_W           // Display window width,must can be divided by 4
,unsigned short Y_H           // Display window length ，must can be divided by 4
);
        void LT768_Set_DisWindowPos
(
 unsigned char On_Off          // 0 : forbid PIP    1 : enable PIP    2 : Keep the original
```

```
state
,unsigned char Select_PIP         // 1 : use PIP1      2 : use PIP2
,unsigned short X_Dis              // Display window X position
,unsigned short Y_Dis              // Display window Y position
);
            void BTE_Solid_Fill
(
  unsigned long Des_Addr                // The destination address of the fill
,unsigned short Des_W                   // Destination address picture width
,unsigned short XDes                    // X position
,unsigned short YDes                    // Y position
,unsigned short color                   // Filled color
,unsigned short X_W                     // Filled length
,unsigned short Y_H                     // Filled width
);
            void LT768_BTE_Memory_Copy
(
  unsigned long S0_Addr         // S0 image memory start address
,unsigned short S0_W            // S0 image width
,unsigned short XS0             // S0 image top left X coordinates
,unsigned short YS0             // S0 image top left Y coordinates
,unsigned long S1_Addr          // S1 image memory start address
,unsigned short S1_W            // S1 image width
,unsigned short XS1             // S1 image top left X coordinates
,unsigned short YS1             // S1 image top left Y coordinates
,unsigned long Des_Addr         // The memory starting address of the target image
,unsigned short Des_W           // The width of the target image
,unsigned short XDes            // The top left X coordinates
,unsigned short YDes            // The top left Y coordinates
,unsigned int ROP_Code          // Grating operation mode
/*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             // active window's width
,unsigned short Y_H             // active window's heigth
);
            void LT768_BTE_Memory_Copy_Chroma_key
(
  unsigned long S0_Addr         // S0 image memory start address
,unsigned short S0_W            // S0 image width
,unsigned short XS0             // S0 image top left X coordinates
,unsigned short YS0             // S0 image top left Y coordinates
,unsigned long Des_Addr         // The memory starting address of the target image
,unsigned short Des_W           // The width of the target image
,unsigned short XDes            // The top left X coordinates
,unsigned short YDes            // The top left Y coordinates
,unsigned long Background_color
,unsigned short X_W             // active window's width
,unsigned short Y_H             // active window's heigth
);
            void LT768_BTE_Pattern_Fill
(
  unsigned char P_8x8_or_16x16          // 0 : use 8x8 Icon , 1 : use 16x16 Icon.
,unsigned long S0_Addr                  // S0 image memory start address
,unsigned short S0_W                    // S0 image width
,unsigned short XS0                     // S0 image top left X coordinates
,unsigned short YS0                     // S0 image top left Y coordinates
,unsigned long Des_Addr                 // The memory starting address of the target image
,unsigned short Des_W                   // The width of the target image
,unsigned short XDes                    // The top left X coordinates
,unsigned short YDes                    // The top left Y coordinates
,unsigned int ROP_Code                  // Grating operation mode
/*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             // active window's width
,unsigned short Y_H             // active window's heigth
);
            void LT768_BTE_Pattern_Fill_With_Chroma_key
(
  unsigned char P_8x8_or_16x16          // 0 : use 8x8 Icon , 1 : use 16x16 Icon.
,unsigned long S0_Addr                  // S0 image memory start address
,unsigned short S0_W                    // S0 image width
,unsigned short XS0                     // S0 image top left X coordinates
,unsigned short YS0                     // S0 image top left Y coordinates
,unsigned long Des_Addr                 // The memory starting address of the target image
,unsigned short Des_W                   // The width of the target image
,unsigned short XDes                    // The top left X coordinates
,unsigned short YDes                    // The top left Y coordinates
,unsigned int ROP_Code                  // Grating operation mode
/*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 long Background_color
,unsigned short X_W             // active window's width
,unsigned short Y_H             // active window's heigth
);
            void LT768_BTE_MCU_Write_MCU_16bit
(
  unsigned long S1_Addr         // S1 image memory start address
```

```
,unsigned short S1_W          // S1 image width
,unsigned short XS1           // S1 image top left X coordinates
,unsigned short YS1           // S1 image top left Y coordinates
,unsigned long Des_Addr       // The memory starting address of the target image
,unsigned short Des_W         // The width of the target image
,unsigned short XDes          // The top left X coordinates
,unsigned short YDes          // The top left Y coordinates
,unsigned int ROP_Code        // Grating operation mode
/*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                   // active window's width
,unsigned short Y_H                   // active window's heigth
,const unsigned short *data           // S0 data buffer first address
);
          void LT768_BTE_MCU_Write_Chroma_key_MCU_16bit
(
 unsigned long Des_Addr               // The memory starting address of the target
image
,unsigned short Des_W                 // The width of the target image
,unsigned short XDes                  // The top left X coordinates
,unsigned short YDes                  // The top left Y coordinates
,unsigned long Background_color
,unsigned short X_W                   // active window's width
,unsigned short Y_H                   // active window's heigth
,const unsigned short *data           // S0 data recive address
);
          void LT768_BTE_MCU_Write_ColorExpansion_MCU_16bit
(
 unsigned long Des_Addr               // The memory starting address of the target image
```

```
,unsigned short Des_W                    // The width of the target image
,unsigned short XDes                     // The top left X coordinates
,unsigned short YDes                     // The top left Y coordinates
,unsigned short X_W                      // active window's width
,unsigned short Y_H                      // active window's heigth
,unsigned long Foreground_color
/*Foreground_color : The source (1bit map picture); map data 1 translate to Foreground color by
color expansion*/
,unsigned long Background_color
/*Background_color : The source (1bit map picture); map data 0 translate to Background color by
color expansion*/
,const unsigned short *data              // Data buffer first address
);
        void LT768_BTE_MCU_Write_ColorExpansion_Chroma_key_MCU_16bit
(
  unsigned long Des_Addr                 // The memory starting address of the target image
,unsigned short Des_W                    // The width of the target image
,unsigned short XDes                     // The top left X coordinates
,unsigned short YDes                     // The top left Y coordinates
,unsigned short X_W                      // active window's width
,unsigned short Y_H                      // active window's heigth
,unsigned long Foreground_color
/*Foreground_color : The source (1bit map picture); map data 1 translate to Foreground color by
color expansion*/
,const unsigned short *data              // Data buffer first address
);
        void BTE_Alpha_Blending
(
  unsigned long S0_Addr      // S0 image memory start address
,unsigned short S0_W         // S0 image width
,unsigned short XS0          // S0 image top left X coordinates
,unsigned short YS0          // S0 image top left Y coordinates
,unsigned long S1_Addr       // S1 image memory start address
,unsigned short S1_W         // S1 image width
,unsigned short XS1          // S1 image top left X coordinates
,unsigned short YS1          // S1 image top left Y coordinates
,unsigned long Des_Addr      // The memory starting address of the target image
,unsigned short Des_W        // The width of the target image
,unsigned short XDes         // The top left X coordinates
,unsigned short YDes         // The top left Y coordinates
,unsigned short X_W          // active window's width
,unsigned short Y_H          // active window's heigth
,unsigned char alpha         // The level of transparency(0~31);
);
```

```
        void LT768_PWM0_Init
(
 unsigned char on_off                    // 0：disable PWM0      1：enbale PWM0
,unsigned char Clock_Divided             // PWM clock    0~3(1,1/2,1/4,1/8);
,unsigned char Prescalar                 // clock division    1~256
,unsigned short Count_Buffer             // Setting the output cycle of the PWM
,unsigned short Compare_Buffer           // Set up duty ratio
);
        void LT768_PWM0_Duty(unsigned short Compare_Buffer);
        void LT768_PWM1_Init
(
 unsigned char on_off                    // 0：disable PWM0      1：enbale PWM0
,unsigned char Clock_Divided             // PWM clock    0~3(1,1/2,1/4,1/8);
,unsigned char Prescalar                 // clock division    1~256
,unsigned short Count_Buffer             // Setting the output cycle of the PWM
,unsigned short Compare_Buffer           // Set up duty ratio
);
        void LT768_PWM1_Duty(unsigned short Compare_Buffer);
        void LT768_Standby(void);
        void LT768_Wkup_Standby(void);
        void LT768_Suspend(void);
        void LT768_Wkup_Suspend(void);
        void LT768_SleepMode(void);
        void LT768_Wkup_Sleep(void);
```

4. LT768_Demo

```
void StartUp_picture(void);
void Display_Levetop(void);
unsigned char Draw_Circle_Ellipse(void);
unsigned char Draw_Line_Curve(void);
unsigned short abs1(int val);
unsigned char Draw_Rectangle(void);
unsigned char Draw_Pentagon(void);
unsigned char Draw_Triangle(void);
u8 Draw_Polygon(void);
u8 Draw_Table(void);
u8 Draw_Pillar_Demo(void);
u8 DMA_Demo(void);
unsigned char Text_Demo(void);
unsigned char Outside_Font(void);
unsigned char PIP_Demo(void);
```

unsigned char App_Demo_BTE(void);
unsigned char App_Demo_Alpha_Blending(void);
unsigned char App_Demo_slide_frame_buffer(void);
void Show_Temperature(int val);
void Show_Progress1(int val);
void Show_Progress2(int val);
void Button_ON_OFF(int val);
void Show_Text(int index);
void TEST_DoubleTriangle(u16 x,u16 y,u16 h,u16 h1,u16 l,u16 a,u16 color1,u16 color2);
void Pointer_Show(u16 a);
void Controller_Demo(void);
unsigned char Cartoon_Show(void);
void Main_GUI(void);
unsigned char Select_Function(void);

5. LT768_KEY

void KEY_Init();
unsigned char Scan_Key(void);
void Waiting_Key(void);
unsigned char Scan_Key_delay(unsigned int t);
unsigned char Scan_FunctionKey(void);