دانشکده مهندسی پزشکی

دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

به نام خدا

گزارش پروژه میکروپروسسور

نام و نام خانوادگی اعضای گروه :

رضاکریم زاده

9333084

آرمین فروزانفر

9333023

استاد: مهندس نگین غلامی‌پور

پاییز96

معرفی پروژه:

خیلی ساده میخوایم یک ساز بسازیم که با تولید فرکانس های هر پرده بتوان آهنگ نواخت .برای این کار از نمایشگر رنگی بورد و صفحه لمسی آن میبایست استفاده شود.

ما برای این کار در صدد برآمدیم تا یک پیانوی دیجیتال بسازیم که حدودا یک اکتاو صدا را برای ما تولید کند پس با برآورد امکانات لازم برای ساخت این ساز می‌توان دریافت که امکانات زیر را لازم داریم:

- راه اندازی صفحه‌ی تاچ و دریافت اطلاعات برای یافتن نقطه‌ای که لمس شده است و پخش نت مربوط به آن.
- طراحی صفحه ای که یک اکتاو از کلاویه های پیانو را به کاربر نمایش دهد.
- داشتن نت های مربوط به هر کلاویه و پخش آنها در زمان تاچ کردن
- استفاده از مبدل دیجیتال به آنالوگ برای تبدیل مقادیر گسسته‌ی صدای هر نوت به ولتاژ آنالوگ
- استفاده از یک تقویت کننده صوتی تا بتوان خروجی میکروکنترلر را به اسپیکر وصل کرد.

بنابراین با طی مراحل بالا به جواب مطلوب می‌رسیم.

صفحه‌ی شروع به کار:

```
int main(void)
{
	delay_init();
	NVIC_Configuration();
	LCD_Init();
	LCD_Clear(White);
	KEY_Init();
	POINT_COLOR=RED;
	LCD_ShowString(60,190,200,16,16, "Micro Lab Project");
	LCD_ShowString(70,210,200,16,16, "electrical Piano");
	tp_dev.init();
```

```
    delay_ms(1000);
    initTim2();
    intro_page();
}
```

در این قسمت از کد پس از اینکه میکرو شروع به کار کرد وارد این بخش می‌شود که بدنه‌ی اصلی کد است و قسمت‌های مختلف میکرو را راه اندازی میکند. در ابتدا یک تابع با نام delay_init() را فراخوانی می‌کند که یک تابع آماده است که فایل های آن به پروژه اضافه شده است و می‌تواند تاخیر های مختلفی را ایجاد کند.

تابع بعدی مربوط به بردار وقفه است که بعدا به طور کامل توضیح داده خواهد شد.

تابع LCD_Init() که باز هم یک تابع آماده است مربوط به راه اندازی و استفاده از ال سی دی تاچ میکرو است که با فراخوانی آن این امکان راه اندازی می‌شود.

تابع بعدی LCD_ShowString() است که مختصات محل چاپ یک رشته و سایز فونت و اندازه‌ی حروف و رشته‌ی مورد نظر را می‌گیرد و روی صفحه نمایش میکرو نمایش می‌دهد. که در اینجا ما عنوان پروژه را به نمایش گذاشته‌ایم و با یک تاخیر یک ثانیه ای آنرا قابل رویت کرده ایم.

تابع بعدی;tp_dev.init() تاچ ال سی دی را فعال می‌کند و می توان برای یافتن نقطه ی تاچ از توابع درون این کتابخانه استفاده کرد.

پس از آن با تابعی برای راه اندازی تایمر2 به کار برده شده که بعدا توضیح می‌دهیم.

درنهایت با فراخوانی تابع intro_page() وارد صفحه‌ی آغازین می شویم که در زیر توضیح داده می‌شود.

```
Void intro_page(void)
{
    LCD_Clear(WHITE);
    LCD_DrawRectangle(80,320,160,360);
    LCD_Fill(80,320,160,360,BLUE);
    LCD_Show_Image(0,20,200,130,*loginpic);
```

```
BACK_COLOR = White;
POINT_COLOR = Red;
LCD_ShowString(20,160,200,16,16, " Piano");
LCD_ShowString(20,180,200,16,16, " Represented By : ");
LCD_ShowString(20,200,200,16,16, " Reza Karimzade ");
LCD_ShowString(20,220,200,16,16, " Armin Froozanfar ");
LCD_ShowString(20,240,200,16,16, " many tanx to:");
LCD_ShowString(20,260,200,16,16, " Negin Gholamipour ");
BACK_COLOR = BLUE;
POINT_COLOR = Yellow;
LCD_ShowString(100,330,100,16,16,"Begin");
while(1)
{
        key=KEY_Scan(0);//physical key scan
        tp_dev.scan(0);//Touch scan
        if(tp_dev.sta&TP_PRES_DOWN)
        {
                if(tp_dev.x<lcddev.width&&tp_dev.y<lcddev.height)
                {

if((tp_dev.x>80)&&(tp_dev.x<160)&&(tp_dev.y>320)&&(tp_dev.y<360))
                                main_page();
                }
        }
        if (key==KEY_2) main_page();
```

```
            if(key==KEY_1)      intro_page();
            delay_ms(10);
        }
}
```

در این تابع در بالای صفحه ی میکرو یک تصویر از آرم دانشگاه و دانشکده قرار می‌دهد که ماتریس آن ساخته شده است و به صورت کتابخانه اضافه شده و با دستور LCD_Show_Image روی ال سی دی نمایش داده می‌شود سپس نام اعضا و عنوان پروژه و یک باکس در قسمت پایین که به صورت دکمه عمل می‌کند اضافه شده که در صورت لمس آن وارد صفحه‌ی اصلی برای نواختن ساز می‌شویم بنابراین در یک حلقه‌ی بینهایت کد گیر می‌کند تا اینکه آن قسمتی که مشخص کرده ایم لمس شود یعنی سه تا دستورهای if صرفا با دریافت مختصات لمس شده توسط تاچ و گرفتن آن با تابع tp_dev.scan(0) چک می‌کند که آیا محل مورد نظر ما لمس شده است یا نه.

طراحی صفحه‌ی کلاویه‌ها:

این قسمت صفحه‌ی اصلی ماست و در این قسمت با لمس هر کلاویه نت مربوط به آن کلاویه باید نواخته شود پس لازم است ابتدا برای کاربر یک صفحه‌ی نمایش نشان دهد تا کاربر نت مورد نظرش را انتخاب کند برای این‌کار از دستورات زیر بهره می‌جوییم.

```
void main_page(void)
{
        RCC -> APB1ENR |= 0x1 << 29;
        RCC -> APB2ENR |= 0x207F << 2;
        DAC -> CR |= 0x10000;
        BACK_COLOR = White;
        POINT_COLOR = Black;
        LCD_Clear(WHITE);
        LCD_DrawLine(0,52,239,52);
        LCD_DrawLine(0,104,239,104);
```

```
LCD_DrawLine(0,156,239,156);
LCD_DrawLine(0,208,239,208);
LCD_DrawLine(0,260,239,260);
LCD_DrawLine(0,312,239,312);
LCD_DrawLine(0,364,239,364);
LCD_Fill(100,40,239,64,Black);
LCD_Fill(100,92,239,116,Black);
LCD_Fill(100,196,239,220,Black);
LCD_Fill(100,248,239,272,Black);
LCD_Fill(100,300,239,324,Black);

        while(1)
{
        if(stop==1)
        {
                stop=0;
                make_sound(pressed);
        }
}
}
```

در این قسمت از کد ابتدا کلاک مربوط به مبدل دیجیتال به آنالوگ فعال می‌شود سپس خود مبدل فعال می-شود.

در ادامه در صفحه‌ی ال سی دی میکرو با دستورات LCD_DrawLine و LCD_Fill یک صفحه به فرم زیر طراحی می‌کنیم:

در انتهای این قسمت کد در یک حلقه‌ی بی‌نهایت گیر می‌کند و هر بار که پرچم stop برابر 1 شد نوت را با تابع make_sound پخش می‌کند.

```
void make_sound(int keypressed)
{
	zarib =.1 ;
	i=0;
	while(i<40617 && stop==0)
	{
		DAC -> DHR8R2 = (uint8_t)(zarib*array[i]);
		tmp=1;
		for(k=1;k<=keypressed ; k++ )
		{
		 tmp=1.06*tmp;
		}
		tmp=floor(22*tmp);
		delay_us(tmp);
		i++;
```

```
        }
}
```

به دلیل آنکه حافظه‌ی میکرو محدود است و در حدود 1 مگابایت است و حجم مربوط به هر نت تقریبا نیم مگابایت است بنابراین نمی‌توان برای هر کلاویه یک نت درون میکرو آپلود کرد بنابراین لازم است یک نت پایه داشته باشیم و نت های دیگر را از روی آن بسازیم ما برای این کار از نت پایه ی دو استفاده کردیم که ماتریسی با حدود 40000 درایه شد سپس نت های دیگر را از روی فرمول زیر از روی این نت ساختیم.

$$f(n) = \left(\sqrt[12]{2}\right)^{n-49} \times 440\,\mathrm{Hz}$$

این فرمول بیان می‌کند که فرکانس نت بعدی با ضرب فرکانس فعلی در ریشه‌ی 12 عدد 2 بدست می‌آید.

بنابراین اگر هر کدام از کلید ها زده شود با توجه به عدد خورده شده فرکانس نت پایه را با یک تاخیر که تاخیر نت پایه 22 میکروثانیه است تاخیر نت جدید را می‌سازد و هر درایه از آنرا با این تاخیر پخش می‌کند بنابراین با یک نت می‌توانیم سایر نت ها را بسازیم.

حال لازم است که هر لحظه چک کنیم آیا در صفحه‌ی نمایش کلیدی لمس شده است یا خیر و با گزارش شکاره کلید لمس شده نت آنرا پخش کنیم بنابراین نیاز داریم تا اینتراپت تایمر2 را راه اندازی کنیم تا بعد از هر اینتراپت این کار را بررسی کند بنابراین توابعی که در بالا به آنها آشاره ای نشد در اینجا کاربردشان نمود پیدا می‌کند.

```
void initTim2(void)
{
   RCC->APB1ENR |= RCC_APB1Periph_TIM2;
  TIM2->CR1 =  TIM_CR1_URS;   //TIM_CR1_DIR |
   TIM2->CR2 = 0;
   TIM2->CNT = 0;
   TIM2->PSC = 2500; // Clock prescaler
   TIM2->ARR = 7200; // Auto reload value
```

```
    TIM2->SR = 0; // Clean interrups & events flag
    TIM2->DIER = TIM_DIER_UIE; // Enable update interrupts
    // NVIC_SetPriority & NVIC_EnableIRQ defined in core_cm3.h
    NVIC_SetPriority (TIM2_IRQn, (1<<__NVIC_PRIO_BITS) - 1); // set Priority for Cortex-M0 System Interrupts
    NVIC_EnableIRQ (TIM2_IRQn)
    TIM2->EGR = TIM_EGR_UG;
    TIM2->CR1 |= TIM_CR1_CEN; // Enable timer
}
```

عملکرد کلی این تابع به این شکل است که ابتدا کلاک مربوط به شمارنده‌ی 2 را فعال می کند سپس خود تایمر را فعال می‌کند سپس prescaler و مقدار شمارش مشخص می‌شود که با توجه به مقادیر داده شده هر 250 میلی ثانیه یک اینتراپت تایمر رخ خواهد داد.
خطوط بعدی مربوط به فعال سازی بردار وقفه و فعال کردن تایمر است که به صورت کامنت هر خط توضیح داده شده است.

حال با فعال سازی وقفه بعد از overflow شدن تایمر سراغ تابع وقفه می‌رود:

```
void TIM2_IRQHandler(void)
{
    if(TIM2->SR & TIM_SR_UIF)
    {
          tp_dev.scan(0);//Touch scan
    if(tp_dev.sta&TP_PRES_DOWN)
    {
      if(tp_dev.x<lcddev.width&&tp_dev.y<lcddev.height)
      {
        stop=1;
        if((tp_dev.x>100)&&(tp_dev.y>40)&&(tp_dev.y<64))
          pressed=12;
        else if((tp_dev.x>100)&&(tp_dev.y>92)&&(tp_dev.y<116))
          pressed=10;
        else if((tp_dev.x>100)&&(tp_dev.y>196)&&(tp_dev.y<220))
          pressed=7;
        else if((tp_dev.x>100)&&(tp_dev.y>248)&&(tp_dev.y<272))
          pressed=5;
        else if((tp_dev.x>100)&&(tp_dev.y>300)&&(tp_dev.y<324))
          pressed=3;
        else if((tp_dev.y>0)&&(tp_dev.y<52))
          pressed=13;
        else if((tp_dev.y>52)&&(tp_dev.y<104))
          pressed=11;
        else if((tp_dev.y>104)&&(tp_dev.y<156))
          pressed=9;
        else if((tp_dev.y>156)&&(tp_dev.y<208))
          pressed=8;
        else if((tp_dev.y>208)&&(tp_dev.y<260))
          pressed=6;
        else if((tp_dev.y>260)&&(tp_dev.y<312))
          pressed=4;
        else if((tp_dev.y>312)&&(tp_dev.y<364))
          pressed=2;
        else if((tp_dev.y>364))
          pressed=1;
      }
    }
    }
    GPIOF->ODR^=(1<<6);
    TIM2->SR = 0;
}
```

در این تابع ابتدا اگر کلیدی زده شده بود فلگ stop را برابر 1 می‌کند تا نت بتواند اجرا شود سپس بررسی می-کند که کدام نقطه زده شده وشماره‌ی نت مربوط را مشخص می کندتا نت در تابع make_sound نواخته شود.