طراحی اسیلسکوپ ساده در محیط شبیه‌سازی proteus

دراین قسمت به طور مختصر به بیان قسمت های مختلف سیستم خواهیم پرداخت.

قسمت sampling_unit

شامل ادوات ADC، Timer و پورت USART برای انتقال داده در درون میکروکنترولر می‌باشد، همچنین شامل بخش های خارجی صفحه کلید و یک ال سی دی برای نمایش داده هاست،

قسمت view_unit

شامل ادوات پورت USART برای ارتباط با بخش نمونه‌برداری به همراه یک قطعه glcd برای نمایش وضعیت کنونی شکل موج می‌باشد.

هم اکنون به بررسی جزعی قسمت های مختلف و شیوه‌ی راه اندازی آن‌ها می‌پردازیم.

-قسمت نمونه گیریADC برای تبدیل سیگنال آنالوگ به دیجیتال:

برای راه اندازی این مبدل، در ابتدا پورت های A0,A1 را در رجیستر مودر به روی حالتanalog تنظیم می نماییم، سپس در رجیستر دوم قطعات جانبیAPB2ENR به فعالسازی کلاک برای قطعه ی ADC اقدام می کنیم، در نهایت در رجیستر دوم کنترلی برای این قطعه، حالت عملیاتی آن را به حالت تک نمونه گیری قرار داده و از حالت زخیره نیرو خارج می نماییم و در پایان آن را فعال می کنیم.

```
ADC initiation //
                                                    RCC->AHB1ENR |= 7;
// enable port A,B,C clock
                                ;GPIOA->MODER  = 0x00000000
                                                  GPIOA->MODER |= 0x33;
// set port A0,A1 to analog
//                                                  RCC->APB2ENR |= (1<<8);
enable A2D clock
//                                                  ADC1->CR2 |= (1<<10);
enable: EOC set at end of conversion | single convert mode
// enable ADC without power saving    ADC1->CR2 &= 0b1111111111111110;
time
                                                    ADC1->CR2 |= 1;
//ENABLE ADC
```

-قسمت تایمر Timer پردازنده

در این قسمت بعد از غیرفعال کردن وقفه های میکروکنترولر، در رجیستر اول قطعات جانبی، تایمر های مورد نظر را فعال می نماییم، بعد از قرار دادن مقدار های لازم برای تنظیم دوره‌ی این تایمر ها (که همان رجیستر های ARR,PSC می‌باشند) در رجیستر کنترلی شماره‌ی اول این قسمت، حالت شمارنده را فعال می نماییم، و همچنین در رجیسترDIER تنظیمات را به گونه ای در نظر می گیریم که بعد از هر مرتبه شمارش، یک وقفه به سیستم زده شود و هندلر تعریف شده آن را هندل نماید. در پایان وقفه های را دوباره فعال می کنیم.

```
//                                                  disable_irq();__
disable interrupts
                                                    RCC->APB1ENR |= 1;
// enable timer 2
//                                                  TIM2->PSC = 16 - 1;
set period of timer2 = 10us
```

```
;TIM2->ARR = 10 - 1
                                                    TIM2->CR1 = 1;
// enable timer2 CounterEnable(CEN)
                                                    TIM2->DIER |= 1;
// send interrupt after update event
// set interrupt                                NVIC_EnableIRQ(TIM2_IRQn);
function for the timer2
;()enable_irq___
```

-قسمت پورت USART برای ارتباط بین دو میکروکنترولر

در این قسمت بعد از فعال کردن امکان USART1 در رجیستر اول و دوم قطعات جانبی میکروکنترولر، تابع های جایگزین شماره ی ۷ را برای پین های A9,A10 در نظر می گیریم، طبق دفتر چه مرجع این تابع های جایگزین همان امکان فرستنده و گیرنده برای USART1 خواهند بود، بعد از تنظیم کردن مقدار baud rate و فرکانس مورد نظر (که در این جا همان 16khz) می باشد، ساختار ارسال داده را مشخص خواهیم کرد، در اینجا ما ۸ بیت را برای داده، ۱ بیت را برای بیت شروع کننده، و ۱ بیت را برای بیت خاتمه دهنده در نظر گرفته ایم، بعد از قرار دادن این تنظیمات، پورت USART1 را فعال می نماییم.

```
USART initiation //
;RCC->AHB1ENR |= 1
//                                          RCC->APB2ENR |= (1<<4);
enable USART1
//                                           GPIOA->AFR[1] = 0x770;
alt7 for USART1 make A9 tx , A10 rx
// enable alternate                          GPIOA->MODER |= 0x280000;
function for PA9 and PA10
// 9600                                  USART1->BRR = 0x008B;
baud @ 16 MHz
// enable                                 USART1->CR1 = 0x000C;
Tx,Rx, 8-bit data by default 1 start bit
stop bit // 1                             USART1->CR2 = 0x0000;
// no flow control                            USART1->CR3 = 0x0000;
// enable USART1                              USART1->CR1 |= 0x2000;
```

-صفحه کلید:

به صورت Busy Waiting چک خواهد شد و در صورت فشارده شدن دکمه توسط کاربر امکان مورد نظر برای او اجرا خواهد شد و در عین حال LCD هم به روز رسانی خواهد شد. بعد از تنظیم کردن چهار پورت Aبه عنوان خروجی برای تحریک هر سطر و قرار دادن سه پورت دیگر از A برای دریافت حالت از صفحه کلید، می توانیم کد آن را نوشته و از ان استفاده نمایم، فقط باید به طور دوره ای آن را چک بنماییم.

قسمتی از تابع فشرده شدن کلید که در آن در هر مرتبه، یک سطر تحریک شده و ستون های آن به ترتیب چک خواهند شد تا کلید فشرده شده احتمالی یافته شود.

```
;GPIOA->ODR = 0xEFFF
;(۱۵)delayMs
۱ //
;(۵)delayMs
```

```
if((GPIOB->IDR | 0xEFFF) == 0xEFFF){return 1;}
۲ //
;(۵)delayMs
if((GPIOB->IDR | 0xDFFF) == 0xDFFF){return 2;}
۳ //
;(۵)delayMs
if((GPIOB->IDR | 0xBFFF) == 0xBFFF){return 3;}

;GPIOA->ODR = 0xDFFF
;(۱۵)delayMs
۴ //
;(۵)delayMs
if((GPIOB->IDR | 0xEFFF) == 0xEFFF){return 4;}
۵ //
;(۵)delayMs
if((GPIOB->IDR | 0xDFFF) == 0xDFFF){return 5;}
۶ //
;(۵)delayMs
if((GPIOB->IDR | 0xBFFF) == 0xBFFF){return 6;}
```

-راه اندازیLCD برای نمایش داده ها

بعد از انتخاب پورت های درست به عنوان ورودی داده و دستور برای واحد نمایشLCD به فعالسازی آن می پردازیم و طبق مراحل شرح داده شده در دیتاشیت این قطعه، به ترتیب ابتدا دنباله ی فعالسازی را به آن می دهیم، سپس حال دو خط با ۱۶ کاراکتر در هر خط را برای آن ست می کنیم، و جهت گیری نشانه نوشتن را برای آن تنظیم خواهیم کرد، همچین صفحه ی نمایش را پاک خواهیم کرد و نشانه را به خانه ی ابتدایی صفحه انتقال خواهیم داد. در نهایت صفحه را روشت کردن و نشانه را در حالت چشمک زن قرار می دهیم.

```
LCD initiation //
//GPIOC init make D0..D7 output                GPIOC->MODER = 0x00005555;
// set pin output mode for LCD                 GPIOB->MODER |= 0x00000015;

// turn off EN and R/W                         GPIOB->BSRR = 0x00C00000;
// initialization sequence                             delayMs(300);
;LCD_command(0x30)
;(۱۰۰)delayMs
;LCD_command(0x30)
;(۱۰)delayMs
;LCD_command(0x30)
```

```
;(۱۰)delayMs
                                                    LCD_command(0x38);
// set 8-bit data, 2-line, 5x7 font
;(۱۰)delayMs
                                                    LCD_command(0x06);
// move cursor right after each char
;(۱۰)delayMs
                                                    LCD_command(0x01);
// clear screen, move cursor to home
;(۱۰)delayMs
                                                    LCD_command(0x0F);
// turn on display, cursor blinking
;(۱۰)delayMs
```

-راه اندازی واحد GLCD

برای این کار، بعد از انتخاب پورت های مناسب برای ورودی داده و دستور برای GLCD بعد از دادن دنباله ی فعالسازی به آن طبق گفته ی دفترچه راهنما، دستگاه آماده استفاده خواهد بود، برای آن دو بافر را در نظر گرفته و از آن ها برای نمایش شکل موج استفاده خواهیم کرد.

```
//Enable PORTC Clock                                RCC->AHB1ENR |=1<<2;
//PA0-PA7                                           GPIOA->MODER |=0x5555;
"Become OUTPUT "DATA PORT
"GPIOC->MODER |=0x0155;       //PC2-PC6 Become OUTPUT "CTRL PORT
//RESET DATA PORT                                   GPIOA->BSRR  |=0xFF0000;
//RESET CTRL PORT                                   GPIOC->BSRR  |=0x1F0000;


                                ;GPIOC->ODR &= !0x8
;(۳۰)delayMs
;GPIOC->ODR &= 0xEF
;(۳۰)delayMs
;GPIOC->ODR |= 0x10
;(۳۰)delayMs
;command_out(0x3F)
;(۳۰)delayMs
;command_out(0x40)
;(۳۰)delayMs
;command_out(0xBA)
;(۳۰)delayMs
```

بعد از فعالسازی قطعات فوق، کافی است که الگوریتم مورد نظر را در جایگاه درست برای قطعات قرار داده و زمان بندی را به طور مناسب رعایت کنیم تا بتوانیم نتیجه ی مطلوب را حاصل نماییم. برای یافتن کد مورد نظر می توانید به کد های قرار داده شده در فایل ارسالی مراجعه کنید. همه به زبان سی هستند و ارزش سخت افزاری نخواهند داشت.