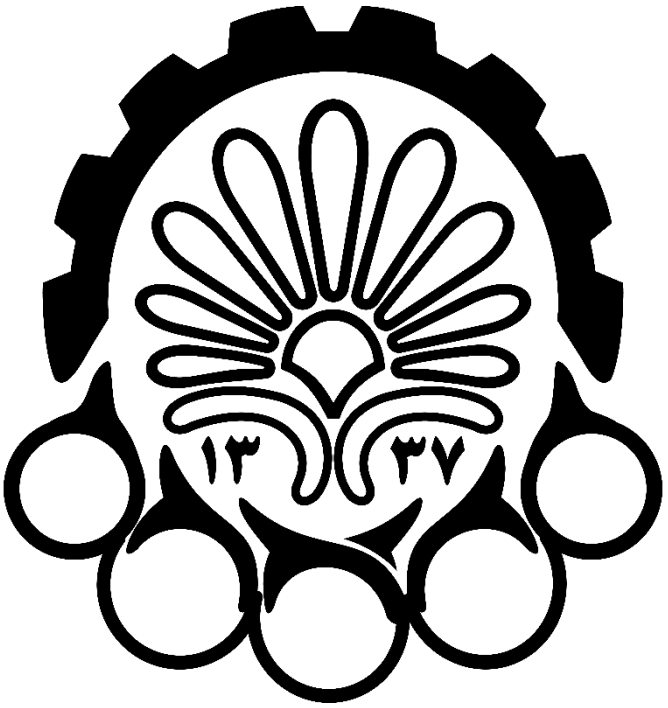


دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

گزارش تمرین سری سوم آزمایشگاه سیستم های ریزپردازنده‌ای و مدارهای واسطه

سهیل داودی ۹۵۲۳۰۴۱

محمد جواد رنجبر ۹۵۲۳۰۴۸

بخش 1 : LCD

void send_4bit(unsigned char data)

برای ساختار یافته کردن کد در ابتدا این تابع را تعریف می کنیم که مقدار هرکدام از پین های داده LCD را با توجه داده ورودی تنظیم می کنیم.

void lcd_write(unsigned char data)

بدلیل اینکه در مد ۴ بیت میخواهیم LCD را راه بیندازیم باید داده ورودی را به رو دو بخش LSB و MSB تقسیم می کنیم و در دو مرحله این دو رو به LCD ارسال کنیم.

برای فرستادن به LCD ابتدا باید پایه enable را یک کنیم, سپس با استفاده از تابع check_LSB_MSB در ابتدا پین های LCD را ریست کرده و بخش MSB را ارسال کرده و پایه enable را ریست می کنیم. و بار دیگر همین روند را برای فرستادن بخش LSB نیز انجام می دهیم.

برای تمیزکردن کد و ست و ریست کردن پایه enable, تابع blink_En را می نویسیم که در ابتدا پایه enable را ۱ و سپس ۰ میکند.

: void lcd_command(unsigned char command)

همانطورکه می دانیم برای فرستادن دستور به LCD باید مقدار پایه rs صفر باشد.

پس در این تابع ابتدا پایه rs را صفرکرده و سپس با استفاده از تابع lcd_write , دستور را به LCD می فرستیم.

: void lcd_data(unsigned char data)

همانطورکه می دانیم برای فرستادن داده به LCD باید مقدار پایه rs یک باشد.

پس در این تابع ابتدا پایه rs را یک کرده و سپس با استفاده از تابع lcd_write , دیتا را به LCD می فرستیم.

void LCD_Init(void)

در این تابع میخواهیم LCD را راه اندازی کنیم که باید دستورات زیر را برای ان ارسال کنیم.

0x28(Function set) : نشان می دهدکه در مد ۴ بیتی کار می کنیم, LCD دو خط دارد و هر خانه ی آن 8*5 پیکسل می باشد.

0x06(Entery mode set) : نشان دهنده جهت حرکت کرسر می باشدکه به سمت راست حرکت می کند و با چاپ هرکاراکتر ادرس DDRAM افزایش پیدا میکند و همچنین صحفه شیفت پیدا نمیکند.

0x0C(Display control) : این دستور برای روشن شدن LCD و خاموش بودن cursor و چشمک نزدن آن استفاده می شود.

همچنین از تابع lcd_clear هم استفاده می کنیم تا lcd پاک شود وکرسر به خانه اول بازگردد.

void LCD_PutChar(unsigned char data)

برای این که هنگامی که خط اول پر می شود به خط دوم برویم, یک کانتر تعریف می کنیم که اگرکانترکمتر از 16 بود فقط کافی است ورودی را چاپ کنیم.

اگر این کانتر برابر 16 بود یعنی خط اول پر شده است, پس از تابع LCD_SetCursor استفاده می کنیم وکرسر را به خط دوم منتقل می کنیم.

حالت بعدی اگر این کانتر برابر 32 باشد یعنی LCD پر شده است, پس باید به خط اول برگردیم و همچنین این کانتر را نیز صفرکنیم.

void LCD_PutString(char *str)

همانطورکه می دانیم هر رشته از حروف در اخر خود یک '\0' دارندکه نشان دهنده این است که رشته به انتها رسیده است.

پس با استفاده از یک حلقه پوینتر ورودی را دی رفرنس کرده و با استفاده از تابع LCD_PutChar آن را روی LCD چاپ می کنیم.

void LCD_SetCursor(unsigned char row, unsigned char col)

با توجه به دیتاشیت آدرس اولین خانه سطر اول از 0x00 شروع شده و ادرس اولین خانه سطر دوم نیز از 0x40.

حال می توان از دستورالعمل set ddram address استفاده کرد و ادرس کرسر را تغیر داد.

در اینجا یک سوییچ روی row قرار دادیم که اگر row برابر ۱ بود باید 0x80 را با Col-1 جمع می کنیم تا به خانه مورد نظر در سطر اول برسیم و همچنین برای سطر دوم نیز با عدد 0XC0 جمع کنیم.

همچنین همانطورکه قبلا گفته شده است, کانتر را باید تغییر دهیم تا شمارش کاراکتر های چاپ شده به هم نخورد.

: void LCD_CreateChar(uint8_t Location,unsigned char data[])

در ابتدا باید با استفاده از دستور set CGRAM address , ادرس خانه ای که میخواهیم کاراکتر جدید در آن نوشته شود را مشخص کنیم.

دقت شودکه فقط ۸ کاراکتر جدید می توانیم اضافه کنیم که ادرس آن ها می تواند از 0x00 تا 0x07 باشد.

سپس با استفاده از یک حلقه for اطلاعات مربوط به هر سطر را بصورت دیتا به LCD می فرستیم.

با توجه به دیتاشیت با اجراکردن CGRAM data write , کانتر آدرس رم افزایش یک واحدی خواهد داشت و لازم نیست در هر مرحله آدرس CGRAM را افزایش دهیم و سپس داده را بفرستیم.

: void LCD_PutCustom(uint8_t Location)

برای اینکه بعد از ذخیره کردن کاراکتر جدیدی بتوانیم آن را نمایش دهیم باید از دستور set ddram address استفاده کنیم که این دستور در تابع LCD_SetCursor, به کارگرفته شده است .

برای ذخیره کردن مکان کرسر در هر لحظه نیز تو متغییر row_curosr و col_cursor را تعریف می کنیم که مقدار آن ها با توجه به اینکه چندکاراکتر در LCD نوشته شده است, تغییر می کند.

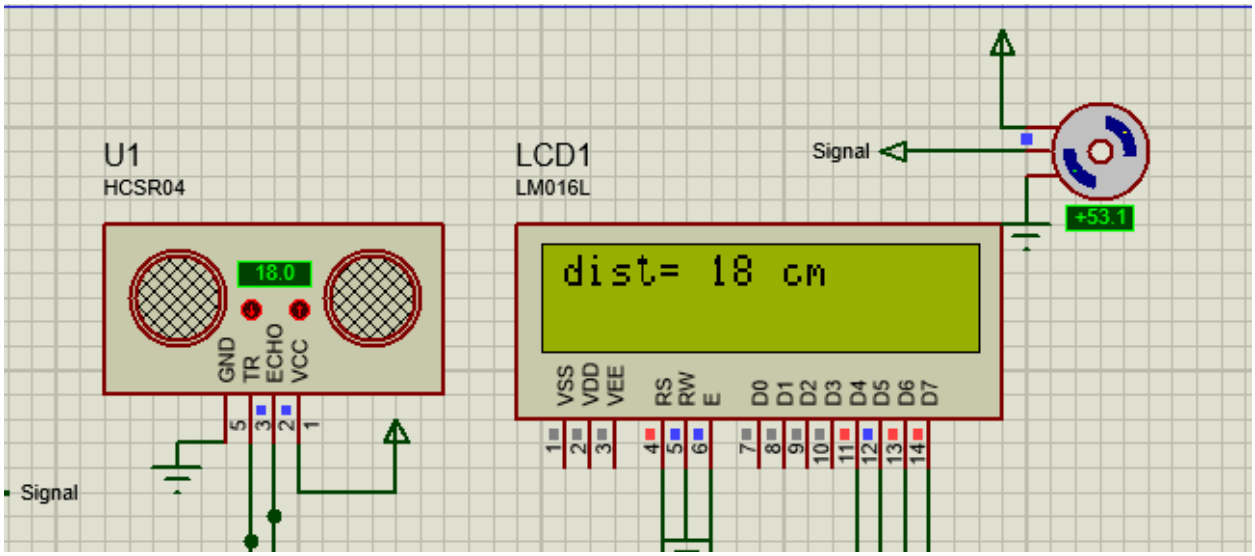
بخش Ultera Sonic و Servo :

void HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef *htim)

در این فانکشن تفاوت زمان بین روی یک لبه بالارونده و یک لبه پایین رونده را محاسبه میکنیم و در نتیجه فاصله را به دست میاوریم . در صورتی که فاصله بین ۳ تا ۳۰ ثانتی متر باشد زاویه موتور را به صورتی تنظیم میکنیم تا به مانع برخورد نکند.

در قسمت main

در while فاصله را بر روی صفحه نمایش نشان میدهیم.



سوال ۱ :حداقل ۶۰ میلی ثانیه برای اندازه‌گیری فاصله نیاز است که فرکانس تقریبا برابر ۱۷ هرتز میشود

سوال ۲ : ۵۰ هرتز