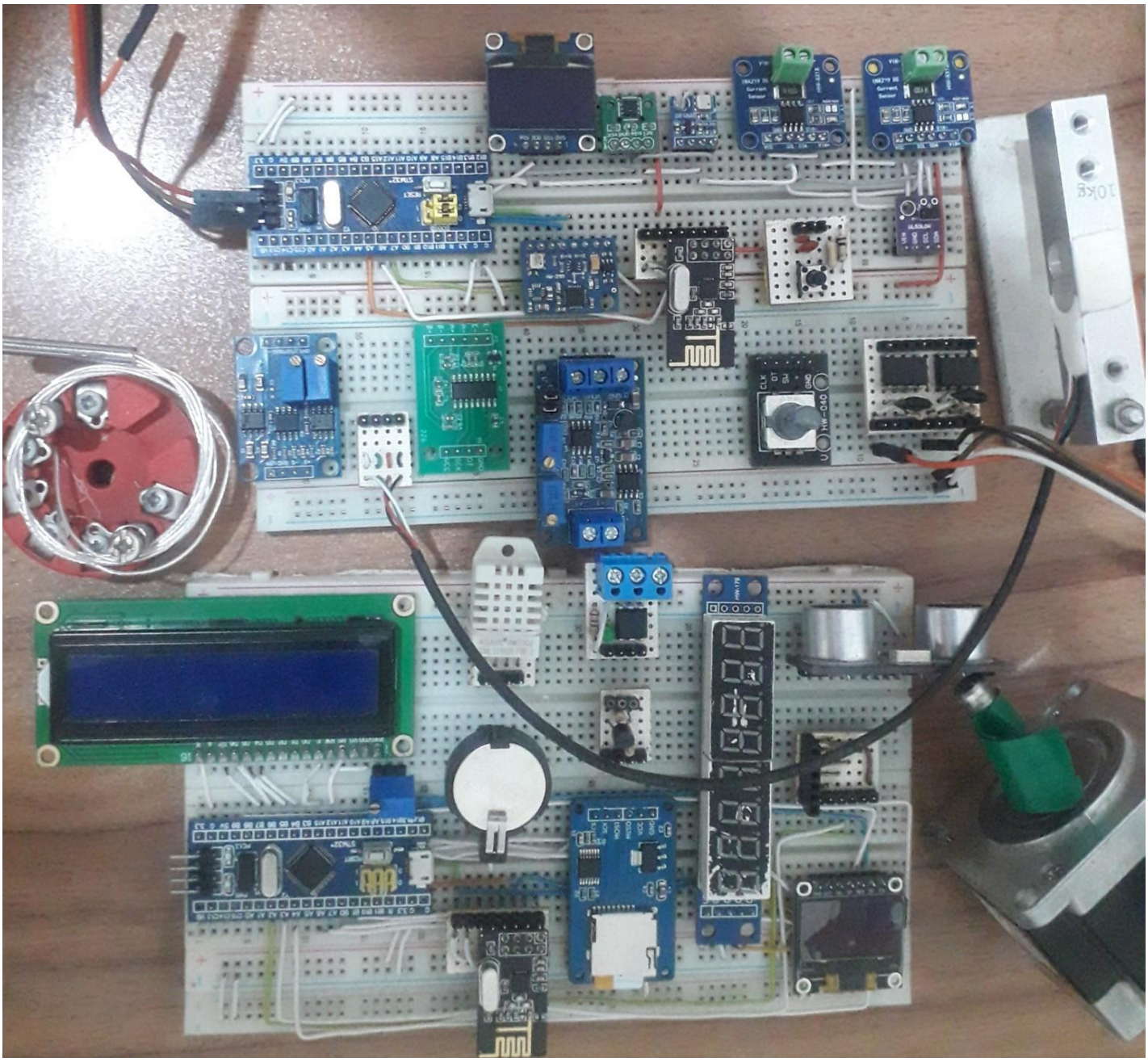
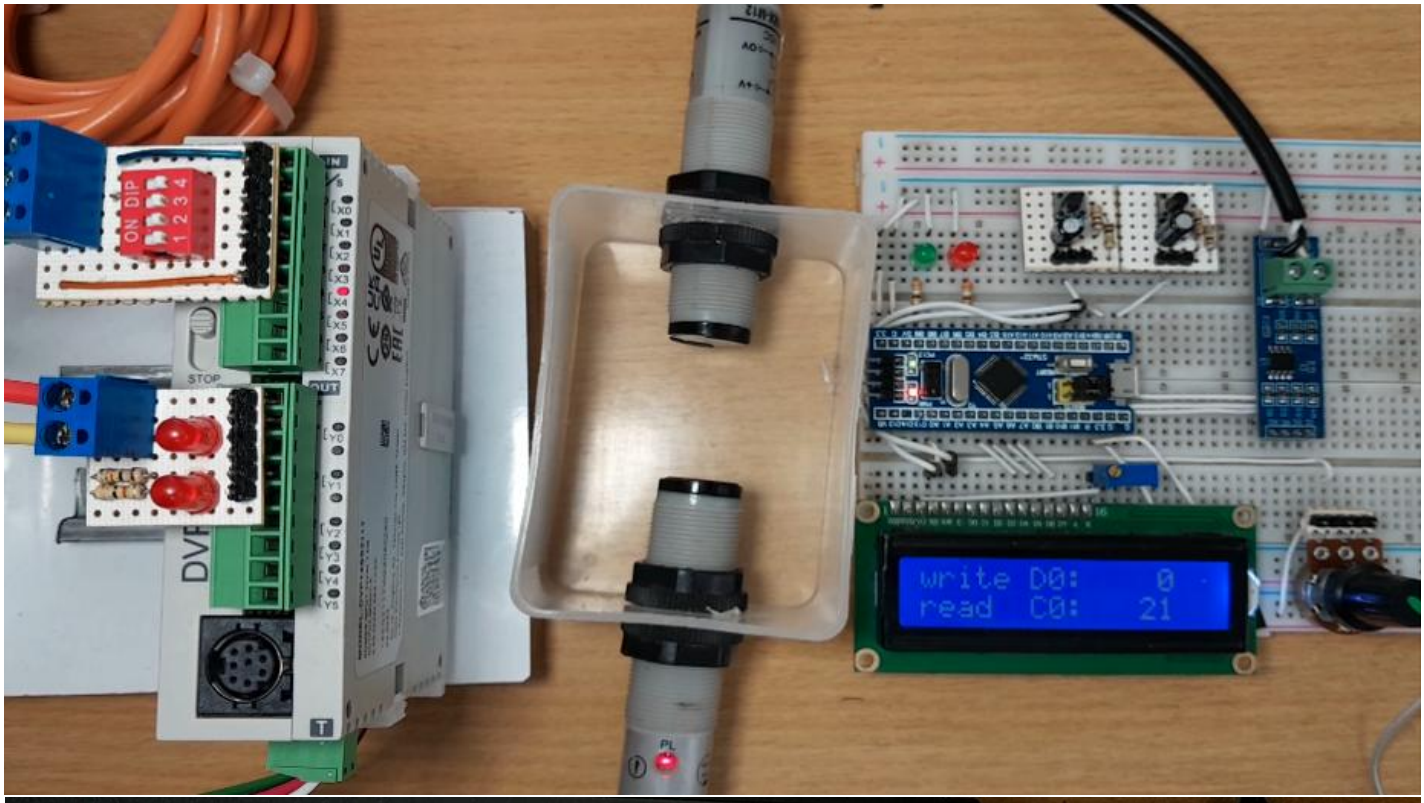
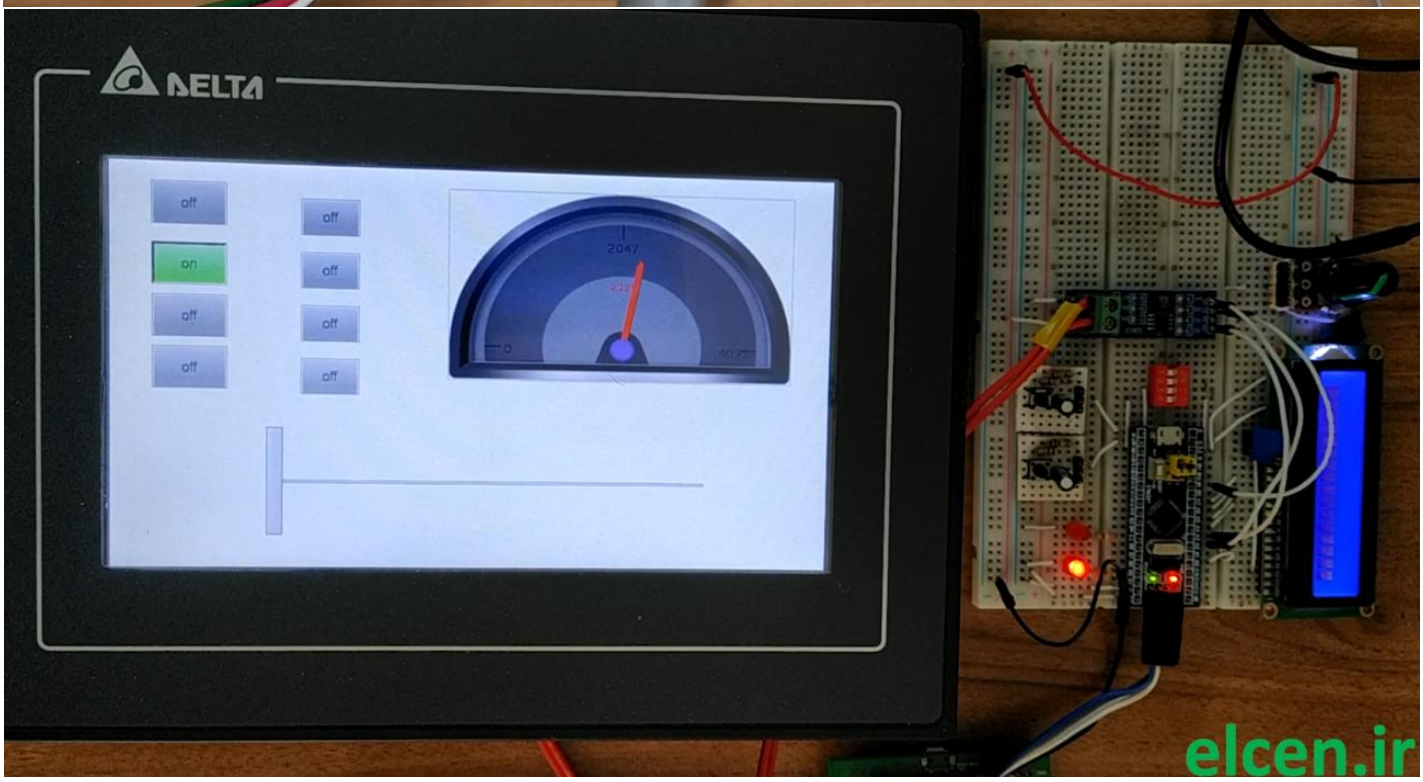
دوره آموزش پیشرفته STM32F103

elcen.ir




write D0: 0
read C0: 21


DELTA
off
on
off
off
off
off
off
off
off
2047
elcen.ir

پروژه محور :

در جریان نوشتن کتابخونه راه اندازی قطعات ، برنامه نویسی میکروکنترلر STM32 رو یادمیگیری. هم به روش رجیستری و هم با کتابخونه . HAL

آموزش استفاده از امکانات STM32 :

میکروکنترلرهای STM32 قدرت هایی دارند که میکروکنترلرهای ارزان تر ندارند، در این دوره انجام پروژه ها رو با استفاده از قابلیت های قدرتمند STM32 و با روش هایی که با AVR امکانش وجود نداره، یادمیگیرید.

برنامه نویسی پیشرفته:

- **برای زمانبندی کارها از تابع دیرکرد یا delay استفاده نمیکنیم. زمانبندی کارها با اینتراپت تایمر هست و تابع دیلی در هیچ کدام از پروژه ها وجود ندارد.**
- **در while(1) کاری مربوط به پریفرال ها انجام نمیشود و while(1) خالی است.**
- **پروژه ها با کمترین بار روی CPU و حداکثر بکارگیری از پریفرال ها انجام میشوند.**
- **شیوه های زمانبندی برنامه بدون بکارگیری RTOS رو یادمیگیرید و اینجوری وقتی همزمان چندین قطعه رو با هم راه اندازی میکنید مشکلی بوجود نمیاد. برتری این دوره این هست که بدون نیاز به RTOS میتونیم چندین قطعه رو همزمان راه اندازی کنیم و زمانبندی رو انجام بدیم.**

نام	محتوا	تایم	سایز	آپدیت	ایجاد				
آموزش پیشرفته STM32F103	67	54:51:26	24G 9G	05/3/23 21:35	02/9/15 07:53	شناسه	+ لایسنس	+ محتوا	حذف
1. تایمر و DMA									حذف
15. ADC و DMA									حذف
17. RTC									حذف
23. PWR									حذف
28. UART : ارتباط Labview و میکروکنترلر									حذف
33. مدباس و ساخت HMI برای PLC دلتا در Labview									حذف
39. UART : ارتباط مدباس بین PLC Delta DVP14SS2 و STM32									حذف
45. UART : ارتباط مدباس HMI دلتا و STM32F103									حذف
50. پریفرال I2C									حذف
65. پریفرال SPI									حذف

4G	15:13:21	13		
624M 282M	1:16:37	1		
2G 1G	4:51:46	5		
3G 1012M	3:33:00	4		
2G 963M	4:51:52	4		
2G	4:54:38	5		
2G 1G	5:52:03	5		
1G 492M	1:44:33	4		
7G 3G	11:32:45	14		
412M 197M	1:00:47	2		

- **ADC و SPI تکمیل نشده.**
- **USB در آینده اضافه میشه.**

فصل اول : تایمر و DMA

معرفی تایمرها

- کاربردهای پریفرال تایمر میکروکنترلرهایstm32
- TIMBASEو رجیسترهای شمارنده و ARR وPSC
- محاسبه ی ARR و PSC با تنظیم دوره تناوب کانتر و دوره تناوب کلاک کانتر
- رجیستر های CCR و کانال در حالت ورودی و خروجی
- کانتر بالارونده ، پایین رونده و edge aligned ، خروجی PWM1 وPWM2
- قابلیت پریلود و رجیستر سایه
- حالت تک پالس
- کنترل نرخ update event با رجیستر RCR
- حالت تک پالس با رجیستر RCR و تولید تعداد مشخص پالس
- انواع تایمرها درSTM32

اندازه گیری سرعت اجرای برنامه در میکروکنترلر STM32 با سه روش

- پروژه led چشمک زن با استفاده از اینتراپت آپدیت تایمر درSTM32
- ARRو PSC برای دقیق ترین تایمر ممکن درstm32f103c8
- اندازه گیری سرعت اجرای کد STM32 با یک تایمر، توابع شروع و توقف شمارنده و خوندن CNT
- معرفی و بررسی رجیسترهای راه اندازی شمارنده ۳۲ بیتی CYCCNT از واحد DWT
- اندازه گیری سرعت اجرای کد با شمارنده داخلCPU cortex-m3
- افزایش ظرفیت شمارنده با اتصال دو تایمر در حالت master-slave ، تنظیم تریگرهای ورودی و خروجی
- اندازه گیری سرعت اجرای کد با دو تایمر آبشاری در میکروکنترلر STM32

پین ETR و پالس شمار سخت افزاری با تایمر STM32

- پالس شمار نرم افزاری : شمارش و مقایسه ی تعداد پالس ها توسط CPU در اینتراپت خارجی
- پالس شمار سخت افزاری : کاربرد محاسبه تعداد دفعات عبور جسم از مقابل سنسورproximity
- شمارش پالس ها با قابلیت external clock و پین ETR تایمرSTM32
- مقایسه ی تعداد پالس ها و تولید اینتراپت توسط کانال خروجی در حالتfrozen

فاصله سنج التراسونیک با STM32 بدون دیلی | ارسال و دریافت پالس با تایمر

- دیتاشیت سنسور : SR-04 پینها، پالس تریگ و پالس اکو، زمان بندی
- تفاوت راه اندازی sr04 در stm32 و AVR ، استفاده از تایمر برای ارسال و دریافت پالس
- SR-04با stm32f103 بدون دیلی :
- ساخت پالس تریگ با کانال خروجی تایمر STM32 در حالتPWM
- اندازه گیری مدت زمان یک بودن پالس echo با دو کانال ورودیinput capture

lcd کاراکتری با STM32 بدون delay با تایمر و | DMA پیاده سازی سخت افزاری ارتباط پارالل

- دیتاشیت lcd کاراکتری : پین ها، چگونگی نمایش کاراکتر، حافظه ها ROM وRAM ، کامند ها، عملیات استارت.lcd
- کنترل زمان بندی بستر سخت افزاری ارتباط پارالل با تایمر stm32 و انتقال داده به باس پارالل با DMA میکروکنترلر stm32f103.
- کانال تایمر STM32 در حالت خروجی PWM برای پین E ارتباط پارالل
- تغییر وضعیت پین های باس پارالل در اینتراپت کانال خروجی تایمر میکروکنترلرSTM32
- DMAبه : GPIO تغییر وضعیت پین های باس پارالل توسط DMA با نوشتن در رجیستر BSRR
- آماده سازی داده ها برای نوشته شدن در رجیستر BSRR برای مد ۸ بیت و ۴ بیت ارتباط با lcd کاراکتری
- ارسال تعداد مشخص بایت با استفاده از قابلیت تک پالس و رجیستر RCR تایمر هایSTM32

سنسور دما و رطوبت DHT22 با STM32 بدون دیلی با تایمر و | DMA پیاده سازی سخت افزاری ارتباط ۱wire-

- دیتاشیت سنسور دما و رطوبت : DHT22پین ها، پروتکل ارتباطی
- چگونگی پیاده سازی بستر سخت افزاری ارتباط one wire توسط تایمرstm32f103

- کانال خروجی تایمر STM32 در حالت PWM برای ارسال پالس استارت و زمین کردن باس
- یک کردن خروجی PWM بعد از توقف تایمر
- دو کانال ورودی تایمر STM32 در حالت input capture برای اندازه گیری مدت زمان یک بودن ۴۰ پالس
- محاسبه ی ۴۰ بیت در ۴۰ اینتراپت لبه ی پایین رونده
- طراحی union برای نگهداری داده های سنسور
- تنظیم اینتراپت تایمر دوم برای استارت تایمر اصلی
- روش اول : استارت تایمر اول توسط تایمر دوم با قابلیت master/slave و بصورت سخت افزاری
- روش دوم : استفاده از کانال خروجی تایمر master برای زمین کردن باس one wire
- تنظیم دو کانال DMA برای انتقال اعداد رجیستر ها CCR1 و CCR2 به دو آرایه ۴۳ عضوی
- روش سوم: انتقال اعداد از رجیسترهای CCR1 و CCR2 به حافظه توسط DMA و محاسبه ی بیت ها در یک اینتراپت TC

سنسور دما DS18B20با STM32 بدون دیلی با تایمر و | DMA پیاده سازی سخت افزاری ارتباط ۱ wire-

- دیتاشیت سنسور دما : DS18B20 پینها، ترانزیستور C1815 برای زمین کردن باسone wire ، حافظه ها، بررسی transaction، کامند ROM وfunction ،
- زمان بندی نوشتن و خوندن بیتهای ۰ و ۱ از باس
- خوندن دما از DS18B20 با STM32 بوسیله ی دو تایمر و دو کانالDMA
- ارسال کامند ۸ بیتی با ارسال ۹ پالس PWM با کانال خروجی تایمر
- کانال DMA میکروکنترلر STM32 برای نوشتن ۹ مقدار عرض پالس PWM در رجیستر CCR3 و فرستادن یک بایت
- اندازه گیری عرض پالس ۷۲ بیت با کانال ورودی تایمر STM32 در حالتinput capture
- کانال DMA برای انتقال ۷۲ مقدار رجیستر CCR1 و خوندن ۷۲ بیت
- طراحی union برای ذخیره سازی بیتها
- اینتراپت DMA برای محاسبه ی بیت ها وCRC ، کتابخانه یCRC
- نوشتن تابع برای انجام ۷ عملیات با فاصله ها زمانی نامنظم با استفاده از قابلیت پریلود رجیستر ARR تایمرSTM32F103

انکودر با تایمرSTM32

- خروجی انکودرincremental ، روتاری انکودر، تشخیص سرعت و جهت با شمردن لبه با توجه به وضعیت کانال دیگر
- ورودی های TI1FP1 وTI2FP2 ، بررسی تفاوت سه حالت مختلف شمارش پالس های انکودر توسط تایمرSTM32

- شمارش پالس های روتاری انکودر و تشخیص جهت با تایمرSTM32

ورودی و خروجی PWM با تایمرSTM32

- فیلتر پایین گذر خازن مقاومتی ، فرکانس شکست
- پروژه خروجی آنالوگ با PWM در میکروکنترلر STM32
- اندازه گیری فرکانس و مدت زمان یک بودن PWM ورودی با دو کانال ، TI1FP1تریگر ورودی

کنترل سرعت و موقعیت استپر موتور با میکروکنترلر | STM32 ارسال پالس با تعداد و فرکانس مشخص

- درایور : DRV8825 تغذیه ها، پین های enable, reset, sleep, step, dir, مشخص کردن رزولوشن و محدود کردن دما با پتانسیومتر
- مدار ایزوله کردن پین های STEP و DIR با دو اپتوکوپلر N137۶
- درایور : TB6600 تغذیه و اتصال به موتور و میکروکنترلر ، محدود کردن جریان و مشخص کردن رزولوشن باDIP switch
- دیتاشیت استپرموتور
- رابطه ی سرعت زاویه ای با فرکانس PWM خروجی و مقدار رجیسترهای ARR وPSC
- مقایسه ی رزولوشن تولید سرعت زاویه ای در بازه های مختلف سرعت برای دو حالت ARR=1 وPSC=0
- کانال یک در حالت pwm برای تولید پالس step
- پروژه تنظیم سرعت و جهت استپر موتور با درایور ها DRV8825 و : TB6600 خروجی PWM با فرکانس متغیر
- کنترل موقعیت استپر موتور : مشخص کردن تعداد پالس های خروجی TIM3 با TIM1 با master/slave دو طرفه
- استفاده از TIM1 برای شمارش آپدیت های TIM3 و توقف TIM3 با قابلیتmaster/slave

فصل دوم : پریفرال ADC در میکروکنترلر STM32F1

نمونه برداری از تک کانال

- بررسی عملکرد پریفرال ADC در میکروکنترلر STM32F1 ، مبدل ADC ، کانال هایADC
- گروه regular و injected ، دیتا رجیسترهای DR و JDR ، صف کانال ها در رجیسترهای SQR و jSQR در ADC میکروکنترلر STM32F1
- تنظیم زمان نمونه برداری و بیت های SMP درADC
- پروژه اول : حالت کاری تک کانال ادامه دار و بررسی کاربردها ، single channel continuous conversion
- پروژه دوم : حالت کاری تک کانال تک تبدیل و بررسی کاربرد ها ، single channel single conversion
- پروژه ی سوم : تنظیم تریگر خارجی برای ADC ، زمان بندی تبدیل ADC با event تایمر ، بیت های EXTTRIG ، EXTSEL، SWSTART
- نوشتن توابع IRQHandler برای اینتراپت های ADC و مدیریت اینتراپت
- پروژه چهارم : تعیین وضعیت خروجی دیجیتال با توجه به مقدار ADC با استفاده از اینتراپتADC
- معرفی analog watchdog در میکروکنترلر STM32F1 و بیت های AWDSGH و AWDEN وAWDCH
- پروژه پنجم : مقایسه مقدار خروجی ADC با استفاده ازwatchdog event
- پروژه ششم : خوندن دما از سنسور داخلی میکروکنترلر با کانال ۱۶ و محاسبات دما ، استفاده از analog watchdog روی سنسور دمای داخلی
- پروژه هفتم : استفاده از DMA برای انتقال داده های ADC به حافظه
- نکته ADC : هنوز چند جلسه داره تا تکمیل شه

ADC هنوز تکمیل نشده ...

فصل سوم : پریفرال RTC در میکروکنترلر | STM32F1 ساعت ، تقویم وAlarm

اینتراپت ثانیه با RTC

- تنظیمات اولیه پریفرال RTC و اینتراپت ثانیه
- تنظیمات و فعالسازی کریستال خارجی LSI ، بیت های DBP ، LSEON، LSERDY , RTCSELوRTCEN
- ساختار پریفرال RTC درSTM32F1 ، معرفی رجیسترهای ALR , DIV , PRL , CNT و CR ، معرفی Backup domain و APB1 domain
- بیت RTOFF و نوشتن در رجیسترهای پریفرال RTC ، مکانیزم چک کردن همراه باtimeout
- نوشتن توابع برای ورود و خروج از حالت تنظیمات پریفرال RTC ، بیتCNF
- بیت RSF و مکانیزم چک کردن سنکرون بودنRTC
- نوشتن توابع IRQHandler و مدیریت اینتراپت هایRTC
- پروژه ساخت اینتراپت ثانیه با پریفرالRTC

تنظیم ساعت RTC و نمایش ساعت در LCD ، آپدیت تاریخ در اینتراپتOverflow

- نوشتن تابع دریافت ساعت از labview باUSART
- خوندن رجیستر CNT در پریفرال RTC و محاسبه ساعت ، کابرد یونیون برای ذخیره سازی مقدارCNT
- جداسازی تنظیمات اولیه و ثانویه RTC ، بردن و خارج کردن میکروکنترلر به حالت تنظیم اولیه RTC با پینGPIO
- نمایش ساعت در lcd کاراکتری در اینتراپت ثانیهRTC
- تست استفاده از باتری برای تامین برق بخش backup domain و پریفرالRTC
- تابع دریافت تقویم از labview با USART بصورت رجیستری
- آپدیت کردن تاریخ در انتهای هر روز در اینتراپت Overflow در پریفرال RTC میکروکنترلرSTM32F1

پریفرال RTC و کتابخانهHAL

- تنظیمات LSI و RTC در نرم افزارSTM32CubeIDE
- بررسی استراکچرRTC_HandleTypeDef
- بررسی تابع HAL_RTC_Init وHAL_RTC_MspInit
- فرمت نمایش BCD و Binary و تبدیل ها
- بررسی warning های کتابخانه HAL برای تنظیمات RTC درSTM32F1
- فعال کردن اینتراپت ثانیه RTC با کتابخانه HAL و پروژه اینتراپت ثانیه
- بررسی تابعHAL_RTCEX_RTCIRQHandler
- تابع HAL_RTC_SetTime بررسی
- جداسازی تنظیمات اولیه و ثانویه RTC در کتابخانه HAL ، دریافت زمان از labview با USART و تنظیم ساعت
- روش آپدیت تاریخ در کتابخانه HAL و بررسی تابعRTC_DateUpdate
- رفع مشکل از دست رفتن تاریخ در کتابخانه HAL ، معرفی رجیسترهای Backup و ذخیره تاریخ در رجیستر هایBackup
- نوشتن کد labview برای ارسال تاریخ به میکروکنترلر توسطUSART
- نوشتن کد دریافت تاریخ در میکروکنترلر از labview و تنظیم تاریخ

Alarm در RTC میکروکنترلر – STM32F1 رجیستری

- مدیریت و فعالسازی اینتراپت Alarm درEXTI17
- نوشتن تابع IRQHandler برای اینتراپتAlarm
- بررسی سیگنال RTC_Alarm و تفاوت استفاده از لبه بالارونده و لبه ی پایین رونده سیگنالRTC_Alarm
- مدیریت تعداد دلخواه Alarm در هر روز ، تشخیص Alarm با خوندن رجیستر CNT
- انجام پروژه تنظیم سه Alarm همراه با اینتراپت

فصل چهارم : پریفرال PWR و حالت های کم مصرف در میکروکنترلر STM32F1

اخطار افت ولتاژ

- اخطار افت ولتاژ PVD در میکروکنترلر STM32F1
- voltage domain ها در میکروکنترلر STM32F103 ، بازه ها و محدودیت ها
- معرفی POR وPDR
- معرفی PVD و سیگنال PVD output ، تنظیم حد ولتاژ با بیت های PLS ، فعالسازی با بیتPVDE
- نوشتن تابع IRQHandler برای اینتراپتPVD
- فعالسازی اینتراپت PVD درEXTI16
- بررسی تفاوت اینتراپت PVD در لبه ی بالارونده و پایین رونده سیگنالPVD Output
- تشخیص لبه بالارونده از پایین رونده با بیتPVDO
- پروژه اخطار در صورت افت ولتاژ در میکروکنترلر STM32F1

مدیریت مصرف توان در میکروکنترلر STM32F1 و حالت کم مصرفsleep

- بررسی مثال های مدیریت مصرف توان در حالت عادی با پایین آوردن کلاک و خاموش کردن کلاک پریفرال ها
- بررسی حالت های کم مصرف CPU و رگولاتور و بیت های SLEEPDEEP وPDDS
- تفاوت حالت های کم مصرف sleep ، stopو standby و تفاوت روشهای sleeponexit ، WFIو WFE برای ورود به حالت های کم مصرف
- وضعیت کلاک HCLK و FCLK و کلاک SysTick در حالت های کاری کم مصرف و غیرفعال کردنSysTick
- پروژه اول: حالت کاری کم مصرف sleep با : sleeponexit فقط اجرای روتین اینتراپت ها
- فعال کردن قابلیت دیباگ در حالت کم مصرف sleep با بیتDBG_SLEEP
- مروری بر inline assembly ، نکات استفاده از WFI ، استفاده از ISB وDSB
- روش های مختلف لود کردن کد در حالت های کم مصرف: دکمه ریست ، استفاده از بوت لودر ، استفاده ازst-link utility

- پروژه دوم: حالت کاری کم مصرف sleep با WFI درmain
- بررسی عمکرد WFE وSEV
- پروژه سوم: حالت کاری کم مصرف sleep با WFE ، تولید ایونت با اینتراپت EXTI ، با ایونت EXTI و با ایونت تایمر
- روش فعالسازی ایونت برای پریفرال ها بجر EXTI و عملکرد بیتSEVONPEND

حالت کم مصرف stop در میکروکنترلر STM32F1

- تعیین وضعیت بیت های SLEEPDEEP و PDDS و LPDS برای حالتstop
- نوشتن تابع تنظیمات مجدد کلاک بعد از خروج از حالتstop
- پروژه اول : رفتن به حالت stop باsleeponexit
- پروژه دوم : رفتن به حالت stop باWFI ، فراخوانی WFI در main و در اینتراپت ، بررسی اولیت اینتراپت برای فراخوانیWFI
- پروژه سوم: خروج از حالت stop با استفاده از ایونت پریفرالEXTI
- فعالسازی دیباگ در حالت stop با بیتDBG_STOP
- پروژه چهارم: خروج از حالت کم مصرف stop در ساعت مشخص باRTC Alarm

حالت کم مصرف standby در میکروکنترلر STM32F1

- فلگ WUF و صفر کردن آن با بیتCWUF
- فلگ SBF برای تشخیص حالت standby و صفر کردن باCSBF
- فعال کردن پین PA0 برای خروج از حالت standby با بیت EWUP و بررسی مدار مورد نیاز و بررسی اشکالاتpower on
- پروژه اول : خروج از حالت standby با استفاده از پینwake up
- خارج کردن اینتراپت SysTick از وضعیت pending با رجیسترICSR
- پروژه دوم: رفتن به حالت Standby در روتین اینتراپت
- پروژه سوم: خروج از حالت Standby در ساعت مشخص شده باRTC Alarm

فصل پنجم: پریفرال USART در میکروکنترلر STM32F1 و ارتباط با کامپیوتر

- معرفی ارتباط سنکرون و آسنکرون، معرفی پین های USART ، معرفی single و half duplex وfull duplex
- معرفی بسترسخت افزاری : USART طول کلمه، Baud rate، parityوstop bit
- فرستادن داده با USART به روش polling و با DMA در – STM32 رجیستری
- بیت های UE ، M، stopو TE ، محاسبه مقدار بیتهای Fraction و Mantissa با توجه به مقدار انتخابیbaud rate
- بررسی دیتارجیستر USART و رجیسترهای ارسال و دریافت داده و شیفت رجیسترها
- فلگ TXE و فلگ TC برای اطلاع از انتقال داده توسطUSART
- نوشتن تابع ارسال بایت ها با یوزارت به روش – polling رجیستری
- پروژه printf با یوزارت به روش polling ، پرینت پیغام ها در کامپیوتر در نرم افزارrealterm
- تنظیم رجیستر CPAR و بیتهای PL, MSIZE, PSIZE, MINC, PINC, CIRC, DIR برای انتقال داده ها با DMA به یوزارت
- نوشتن تابع استارت کردن عملیات DMA و رجیسترهای CMAP وCNDTR
- پروژه ی printf با یوزارت با DMA ، پرینت پیغام ها در نرم افزارRealterm
- بررسی کامل تمام ایونت ها و اینتراپت های یوزارتPE, FE, NE, ORE, IDLE, RXNE, TC, TXE, LBD, CTS :
- مدیریت اینتراپت های یوزارت و نوشتن تابع : IRQ_Handler صفر کردن فلگ ها و فراخوانی تابعCallback

بررسی کتابخانه HAL برای پریفرال USART و مقایسه توابع polling, interrupt و DMA

- تنظیمات یوزارت در نرم افزارSTM32CubeIDE
- تفاوت عملکر سه روش انتقال دیتا به دیتا رجیستر یوزارت polling : و interrupt وDMA
- بررسی عملکرد ورودی timeout در تابع HAL_UART_Transmit و اندازه گیری زمان اجرای این تابع، بررسی خروجی HAL_Timeout
- اندازه گیری زمان اجرای HAL_UART_Transmit_IT ، بررسی عملکرد تابع با قابلیت watchpoint در دیباگ و خروجی HAL_Busy
- بررسی نقش اینتراپت های DMA در کتابخانه HAL وقتی از DMA برای انتقال داده به یوزارت اسفاده میشه
- نقش تابع MSP و بررسی ماکرو LINK_DMA برای اجرای اینتراپت هایDMA

- استفاده و اندازه گیری زمان اجرای تابع HAL_UART_Transmit_DMA و خروجیHAL_Busy
- اضافه کردن قابلیت timeout به کد رجیستری ارسال داده با یوزات با استفاده ازSysTick
- برنامه ی قابلیت تشخیص busy در کد رجیستری برای ارسال داده با یوزارت
- بررسی خروجی HAL_Error در توابع یوزارت

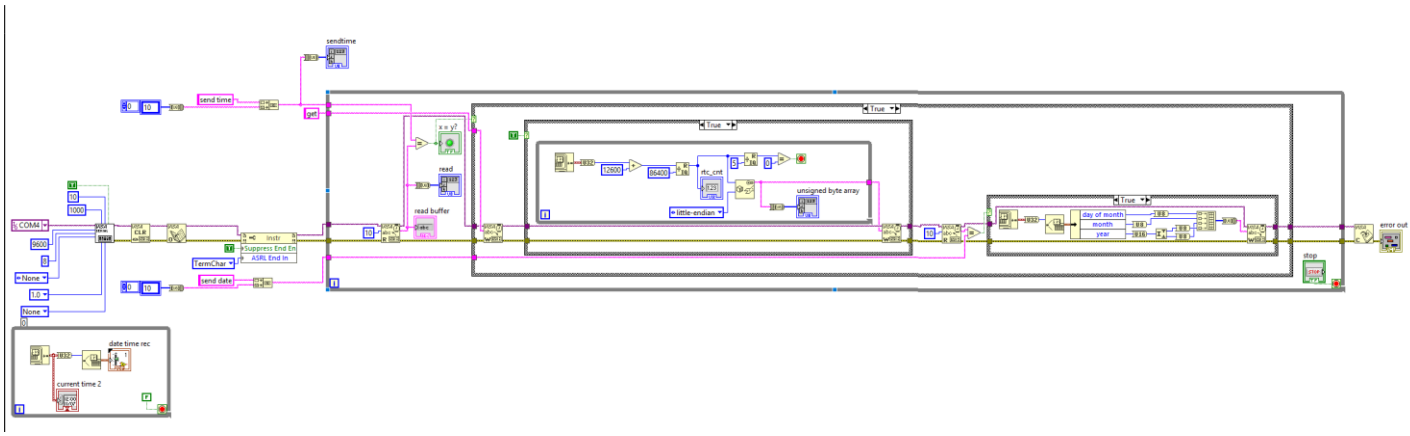
نوشتن برنامه های Labview برای ارتباط با میکروکنترلر STM32

- معرفی بخش های مختلف نرم افزارLabview ، نمایشگر و کنترل
- بلوک visa config و انجام تنظیمات بستر سخت افزاریUART
- بلوک های clear, open, read و close درlabview
- بررسی ۳ دلیل توقف بلوک read در labview و اخطار خروجی برای هرکدام
- شناسایی نوع اخطار های سریال در Labview با کلاستر اخطار ها درcase
- مشخص کردن کاراکتر termination سریال باVISA property node
- بررسی مثال اخطار timeout درlabview
- حلقه ی while در labview ، توقف و ایندکس حلقه
- چسباندن رشته ها به همدیگر و استفاده از شیفت رجیستر درlabview
- پروژه printf با USART درLabview
- پروژه ی ارسال بایت به Labview از میکروکنترلرSTM32

دریافت ساعت از Labview با استفاده از USART در میکروکنترلر STM32F1

- بلوک Get Date in second در Labview و بررسی زمان epoch در کامپیوتر
- محاسبه ی تعداد ثانیه های گذشته از ابتدای روز درLabview
- تبدیل عدد ۳۲ بیتی زمان به آرایه از چهار ۸ بیتی با بلوک های split number و کست کردن وbuild array
- بلوک flatten string و ارسال داده ها از Labview با بلوکsend
- نقش فلگ RXNE در دریافت داده های USART درSTM32
- استفاده و بررسی تابعHAL_UART_Recieve
- نوع union برای چهار بایت دریافتی زمان بوسیله یUSART
- سنکرون کردن ارتباط USART بین میکروکنترلر و Labview با استفاده ازtermination char

- اضافه کردن کاراکتر NULL به انتهای رشته درLabview
- تغییر کاراکتر termination char درlabview
- نوشتن تابع مدیریت ارتباط USART برای دریافت زمان از labview همراهtimeout
- تغییر تابع HAL_RTC_SetTime برای تنظیمRTC
- نوشتن تابع دریافت داده های USART بصورت رجیستری با روشpolling
- پروژه دریافت زمان از labview با کتابخونه HAL و رجیستری



فصل ششم: ارتباط مدباس در Labview با PLC دلتا DVP14SS2

این فصل آموزش نرم افزار Labview است.

مدباس در Labview و ساخت HMI برای PLC دلتا DVP14SS2 در کامپیوتر

- معرفی پینها و سیم کشی PLC دلتا DVP14SS2
- نوشتن برنامه ی PLC برای Ascii و RTU در برنامه ی WPL Soft
- معرفی RS485 و مبدل های RS485 به USB و RS485 به TTL
- معرفی رجیسترها و رله ها در حافظه ی PLC
- بررسی ساختار دستورات مدباس و تفاوت مدباس Ascii و RTU
- روش محاسبه ی chechsum در مدباس Ascii

نوشتن برنامه Labview برای تغییر خروجی های دیجیتال PLC با استفاده از Labview

- تنظیمات اولیه ارتباط سریال در Labview با بلوک configure port
- بلوک های open ، write ، read ، clear و close در labview
- بررسی کامند ۱۵ برای تغییر بیت ها و پیاده سازی در Labview
- معرفی آدرس ها در PLC
- ساخت دستور مدباس Ascii با بلوک concatenate string در Labview
- بلوک Build array و Boolean array to number در Labview
- تبدیل عدد به رشته ی hexadecimal در Labview و تبدیل رشته به عدد
- بلوک subsetstring و جدا کردن بخشی از یک رشته در Labview
- محاسبه ی LRC ارسالی و دریافتی در Labview
- بررسی پاسخ ارسالی از طرف PLC به کامپیوتر برای دستور 15
- تست برنامه Labview برای مدباس Ascii به روش اول
- بلوک format into string برای ساخت رشته
- استفاده از flat sequence در Labview برای اعمال ترتیب انجام کارها

- بلوک scan from string در Labview برای تفکیک داده های دریافتی از PLC
- بررسی و دریافت پیغام های خطا که توسط PLC به Labview ارسال میشود
- تشخیص نوع پیغام خطای ارسال شده توسط PLC در Labview با case structure
- بررسی اشتباه استفاده از local variable در Labview
- استفاده از کلاستر برای ذخیره سازی داده های دریافت شده از پنل
- بلوک های bundle by name و unbundle by name در Labview
- تست پروژه ی تغییر خروجی های دیجیتال PLC با Labview به روش دوم

نوشتن کتابخانه مدباس Ascii و RTU برای نوشتن و خوندن در کویل ها و رجیسترهای DVP14SS2 PLC

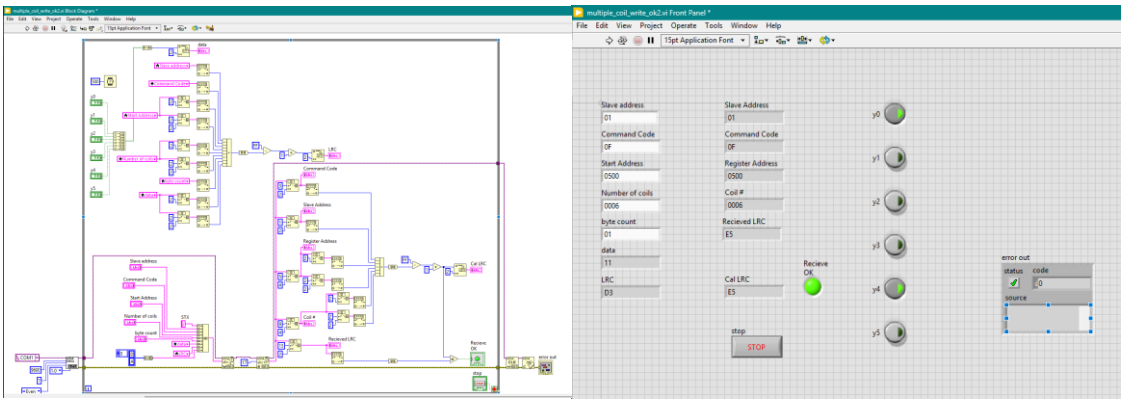
در جلسه ی قبلی برنامه ی Labview برای ارتباط مدباس Ascii با PLC دلتا DVP14SS2 نوشته شد، در این جلسه دستورات مختلف رو تست میکنیم.

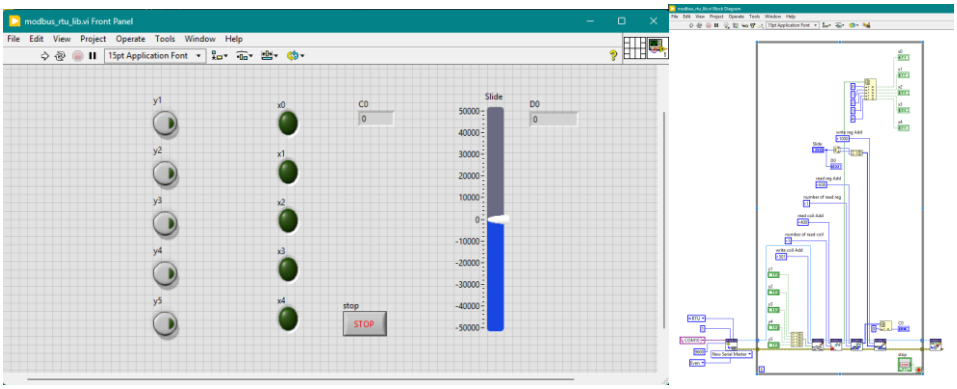
- بررسی ساختار دستور 02 برای خوندن ورودی های دیجیتال PLC
- تغییر برنامه ی Labview برای ارسال دستور 02
- بررسی نحوه ی شماره گذاری کویل ها در PLC
- بررسی ساختار پاسخ دریافتی به دستور 02
- خوندن ورودی های دیجیتال PLC در Labview
- بررسی ساختار دستور 06 برای نوشتن مقدار رجیستر ۱۶ بیتی
- رجیسترهای 32 و ۱۶ بیتی و تفاوت دستورات LD و DLD در WPLSoft
- تغییر مقدار رجیستر ۱۶ بیتی در PLC با استفاده از Labview
- ساختار دستور 16 برای تغییر چند رجیستر ۱۶ بیتی
- اصلاح برنامه ی Labview برای ارسال دستور 16
- تست تغییر مقدار رجیسترهای ۳۲ بیتی PLC با Labview
- ساختار دستور 03 برای خواندن مقدار رجیسترهای PLC و اصلاح برنامه Labview
- بررسی پاسخ PLC به دستور 03 و اصلاح برنامه ی Labview
- تست پروژه خوندن رجیستر PLC با استفاده از Labview

کتابخانه آماده مدباس Ascii و RTU در Labview

در دو جلسه ی قبل بدون استفاده از کتابخانه ی آماده تمامی کدهای مورد نیاز برای مدباس Ascii در Labview رو نوشتیم. در این جلسه کتابخونه ی آماده ای که وجود داره رو بررسی میکنیم.

- **آموزش نصب کتابخانه ی مدباس با** VI Package manager
- **بلوک** create Modbus instance
- **بلوک** write multiple coil **و تغییر خروجی های** PLC
- **تابع** shutdown
- **بلوک** Read discrete input **برای خوندن ورودی های دیجیتال** PLC
- **بلوک** Read holding register **برای خوندن رجیسترهای** PLC
- **تابع** write multiple register
- **تست برنامه مدباس** RTU **برای خوندن و نوشتن در بیت ها و رجیسترهای** PLC





فصل هفتم : ارتباط مدباس Ascii و RTU بین PLC دلتا و میکروکنترلر STM32F103

در این فصل کتابخوه ارتباط مدباس Ascii و RTU در میکروکنترلر STM32F103 برای ارتباط با PLC دلتا DVP14SS2 از صفر مینویسیم.

ارتباط مدباس Ascii بین PLC دلتا و میکروکنترلر STM32F103 به روش polling

- **ساخت استراکچر برای ذخیره سازی محتوای دستورات 02، 03، ۱۵ و ۱۶**
- **نوشتن تابع محاسبه ی LRC**
- **تابع sprint برای جایگذاری اعداد در رشته**
- **تعریف استراکچر برای دستورات مدباس RTU**
- **نوشتن تابع محاسبه CRC16 برای مدباس RTU**
- **نوشتن تابع سریعتر جایگرین تابع sprint**
- **تنظیمات UART و پین های مورد نیاز برای مبدل RS485 به TTL**
- **تعریف استراکچر برای ذخیره سازی پاسخ دستورات دریافت شده از PLC**
- **تابع spfintf برای تبدیل رشته به مجموعه ی اعداد**
- **نوشتن تابع سریعتر از sprint برای استخراج اعداد از رشته ی دریافتی**
- **ارسال دستورات خوندن و نوشتن بیت و رجیستر از میکروکنترلر به PLC با روش polling**

ارتباط مدباس Ascii بین PLC دلتا و میکروکنترلر STM32F103 با DMA

- **بررسی زمان مورد نیاز UART برای ارسال دیتا**
- **بررسی تفاوت داده های دریافتی با DMA و بدون DMA و نوشتن تابع حذف parity**
- **استفاده از اینتراپت DMA برای آغاز عملیات ارسال بعدی مدباس**
- **تشخیص مرحله ی عملیات در هر اینتراپت با enum**
- **ارسال و دریافت دستورات مدباس Ascii از STM32F103 به PLC با استفاده از UART و DMA**

ارتباط مدباس RTU بین PLC دلتا و میکروکنترلر STM32F103 با DMA

- اعمال تنظیمات ارتباط RTU در PLC با نرم افزار WPLSoft
- بررسی و تست زمانبندی مدباس RTU و اشتباهات رفرنس منوال PLC
- تغییر توابع و تعریف ها برای مدباس RTU
- اجرای مدباس RTU برای خوندن و نوشتن در بیت ها و رجیسترهای PLC توسط STM32F103
- ایجاد فاصله ی زمانی بین دریافت و ارسال بعدی با استفاده از اینتراپت تایمر در حالت تک پالس

درود، هر کدوم از این فصل ها چند ساعته و خیلی جزئیات داره ، جزئیات بقیه فصل ها رو ننوشتم.

فصل هشتم : ارتباط مدباس HMI دلتا و STM32F103

ارتباط مدباس RTU بین HMI دلتا DOP107 و میکروکنترلر STM32F103 با روش polling و DMA

- کتابخونه مدباس رو برای HMI دلتا هم مینویسیم

فصل نهم : I2C و DMA

- سنسور های شتاب سنج MPU6050 + دما و فشار هوای BMP085 + ولتاژ و جریان INA219 + دما و رطوبت AHT21 + نمایشگر OLED با I2C و DMA و مدیریت زمانبندی با تایمر

فصل هشتم : SPI و DMA

- ماژول NRF24L0 + SD Card + max7219 OLED SPI با SPI و DMA و مدیریت زمانبندی با تایمر

فصل هشتم : USB (هنوز اضافه نشده)