فاطمه سادات سیفی – 98243035

پارسا محمدپور – 98243050

سوالات تحلیلی:

سوال پروژه ای:

تنظیمات پین ها:

پین	اسم در پروتئوس	ورودی / خروجی میکرو (تایپ)	کاربرد
PA0	Temp	ورودی	کانال تبدیل کننده آنالوگ به دیجیتال برای دما سنج
PA1	Sig	ورودی	کانال تبدیل کننده آنالوگ به دیجیتال برای گرفتن سیگنال سینوسی
PA2	La	خروجی	ورودی a سون سگمنت سمت چپ
PA3	Lb	خروجی	ورودی b سون سگمنت سمت چپ
PA4	Lc	خروجی	ورودی c سون سگمنت سمت چپ
PA5	Ld	خروجی	ورودی d سون سگمنت سمت چپ
PA6	Le	خروجی	ورودی e سون سگمنت سمت چپ
PA7	Lf	خروجی	ورودی f سون سگمنت سمت چپ
PA8	Lg	خروجی	ورودی g سون سگمنت سمت چپ
PA9	RX	Alternate Function	ورودی RX عه VIRTUAL TERMINAL
PA10	RT	Alternate Function	ورودی TX عه VIRTUAL TERMINAL
PB0	Start	External interrupt ورودی(pull down)	ورودی برای دکمه start
PB1	Cooling	ورودی(pull down)	ورودی برای دکمه cooling
PB2	Reset	External interrupt ورودی(pull down)	ورودی برای دکمه reset
PB3	Green	خروجی	خروجی برای ثابت روشن نگه داشتن LED عه سبز

PB4	Red_pwm	کانال یک تایمر 2 (PWM)	کانال pwm برای چشمک زدن LED عه قرمز
PB5	Orange_pwm	کانال دو تایمر 2 (PWM)	کانال pwm برای چشمک زدن LED عه سبز
PB6	Red_simple	خروجی	خروجی برای ثابت روشن نگه داشتن LED عه قرمز
PB7	Orange_simple	خروجی	خروجی برای ثابت روشن نگه داشتن LED عه نارنجی
PB8	Ra	خروجی	ورودی a سون سگمنت سمت راست
PB9	Rb	خروجی	ورودی b سون سگمنت سمت راست
PB10	Rc	خروجی	ورودی c سون سگمنت سمت راست
PB12	Rd	خروجی	ورودی d سون سگمنت سمت راست
PB13	Re	خروجی	ورودی e سون سگمنت سمت راست
PB14	Rf	خروجی	ورودی f سون سگمنت سمت راست
PB15	Rg	خروجی	ورودی g سون سگمنت سمت راست
PC0	D0	خروجی	خروجی برای مبدل دیجیتال به آنالوگ(LSB کم ارزش ترین بیت)
PC1	D1	خروجی	خروجی برای مبدل دیجیتال به آنالوگ
PC2	D2	خروجی	خروجی برای مبدل دیجیتال به آنالوگ
PC3	D3	خروجی	خروجی برای مبدل دیجیتال به آنالوگ
PC4	D4	خروجی	خروجی برای مبدل دیجیتال به آنالوگ
PC5	D5	خروجی	خروجی برای مبدل دیجیتال به آنالوگ
PC6	D6	خروجی	خروجی برای مبدل دیجیتال به آنالوگ
PC7	D7	خروجی	خروجی برای مبدل دیجیتال به آنالوگ
PC8	D8	خروجی	خروجی برای مبدل دیجیتال به آنالوگ
PC9	D9	خروجی	خروجی برای مبدل دیجیتال به آنالوگ

PC10	D10	خروجی	خروجی برای مبدل دیجیتال به آنالوگ
PC11	D11	خروجی	خروجی برای مبدل دیجیتال به آنالوگ
PC12	D12	خروجی	خروجی برای مبدل دیجیتال به آنالوگ
PC13	D13	خروجی	خروجی برای مبدل دیجیتال به آنالوگ
PC14	D14	خروجی	خروجی برای مبدل دیجیتال به آنالوگ
PC15	D15	خروجی	خروجی برای مبدل دیجیتال به آنالوگ(MSB با ارزش ترین بیت)
VREF+	VDD	ورودی(ولتاژ منبع)	ولتاز منبع برای راه انداختن و فعال بودن و کار کردن عه کانال های تبدیل آنالوگ به دیجیتال
VSSA/VREF-	GND	ورودی(زمین)	ولتاز زمین برای راه انداختن و فعال بودن و کار کردن عه کانال های تبدیل آنالوگ به دیجیتال

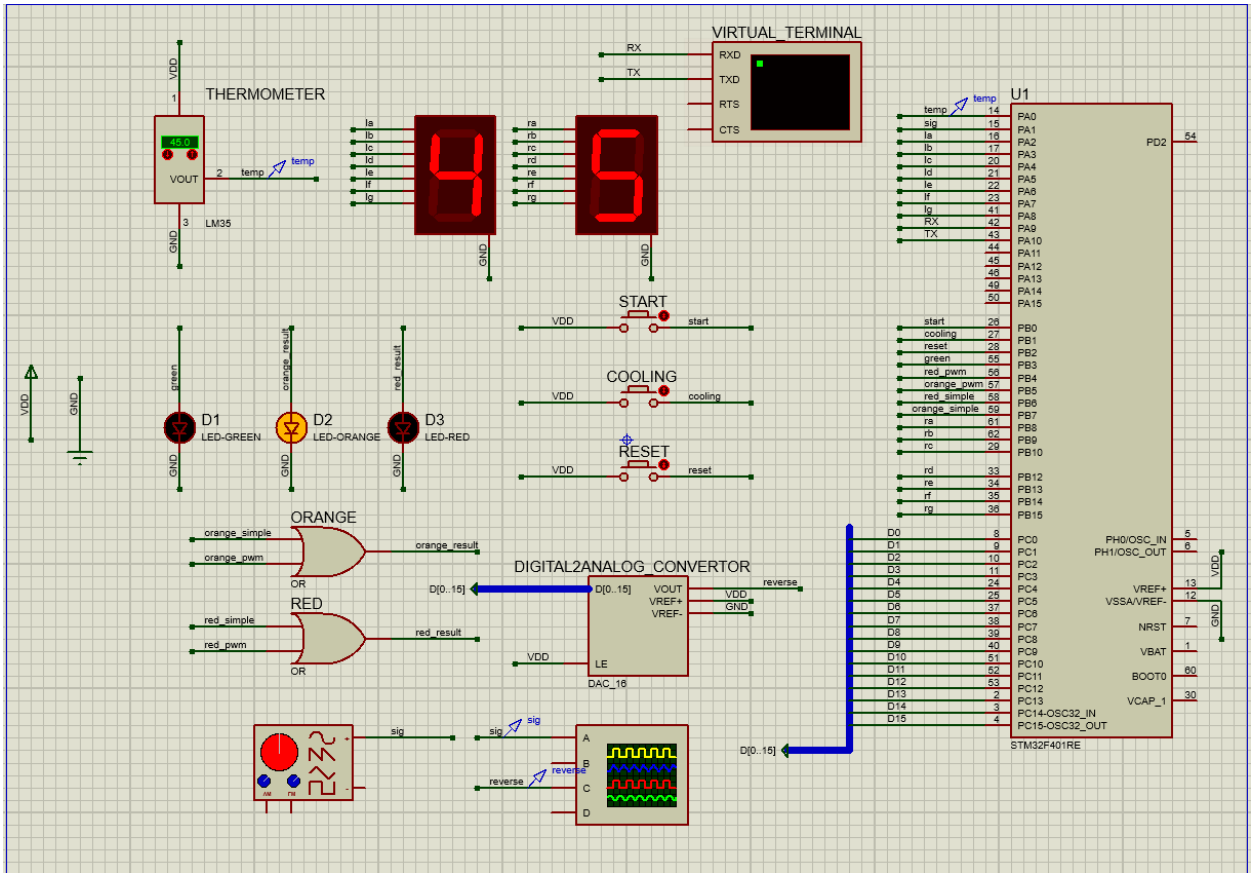
تنظیمات تایمر ها:

شماره تایمر	تنظیمات پیشرفته / عادی	کانال	نوع	استفاده
TIM2	عادی	-	ساختن interrupt	برای اندازه گیری دما هر ۲۰ میلی ثانیه یک بار
TIM3	پیشرفته	Channel1	(PWM)	برای چشمک زدن LED عه قرمز
TIM3	پیشرفته	Channel2	(PWM)	برای چشمک زدن LED عه سبز

پروتئوس:

در این قسمت به شماتیک مدار و توضیحات کار های انجام شده در پوتئوس می پردازیم.

۱- شماتیک مدار:



۲- اسم اعضا موجود در شماتیک مدار:

- **THEREMOMETER:**

دما سنج ما می باشد و با گرفتن ولتاژ منبع و ولتاژ زمین، با توجه به دمای موجود ، یک ولتاژ خروجی تولید می کند، که این ولتاژ در اصل تقریبا تقسیم بر 100 شده دمای این دماسنج می باشد. (مثلا اگر دما 45 درجه باشد، ولتاژ خروجی تولیدی آن نیز ...0.45 ولت می باشد)

- **VIRTUAL_TERMINAL :**

این ماژول در اصل برای لاگ گرفتن و چاپ کردن و نوشتن بر روی کامند لاین طور می باشد.(از آن برای دیباگ کردن و فهمیدن حالت هایی که در آن هستیم و یا دمای اندازه گیری شده و ... می باشد.)

- **DIGITAL2ANALOG_CONVERTOR :**

این ماژول در اصل همان مبدل دیجیتال به آنالوگ خارجی می باشد که با گرفتن 16 بیت ورودی به صورت BUS ، ولتاژ آنالوگ خروجی را تولید می کند.(دو ورودی دیگر برای ولتاژ منبع و ولتاژ زمین نیز دارد.)

- **OSCILLOSCOPE :**

این ماژول که دقیقا زیر مبدل دیجیتال به آنالوگ قرار دارد، برای نشان دادن شکل موج های مختلف استفاده می شود. که قابلیت نشان دادن تا ۴ نوع سیگنال به صورت همزمان را دارد. ما از کانال یک آن برای نشان دادن سیگنال شکل موج ورودی، و از کانال ۳ آن برای نشان دادن شکل موج سیگنال معکوس شده آن استفاده کردیم.

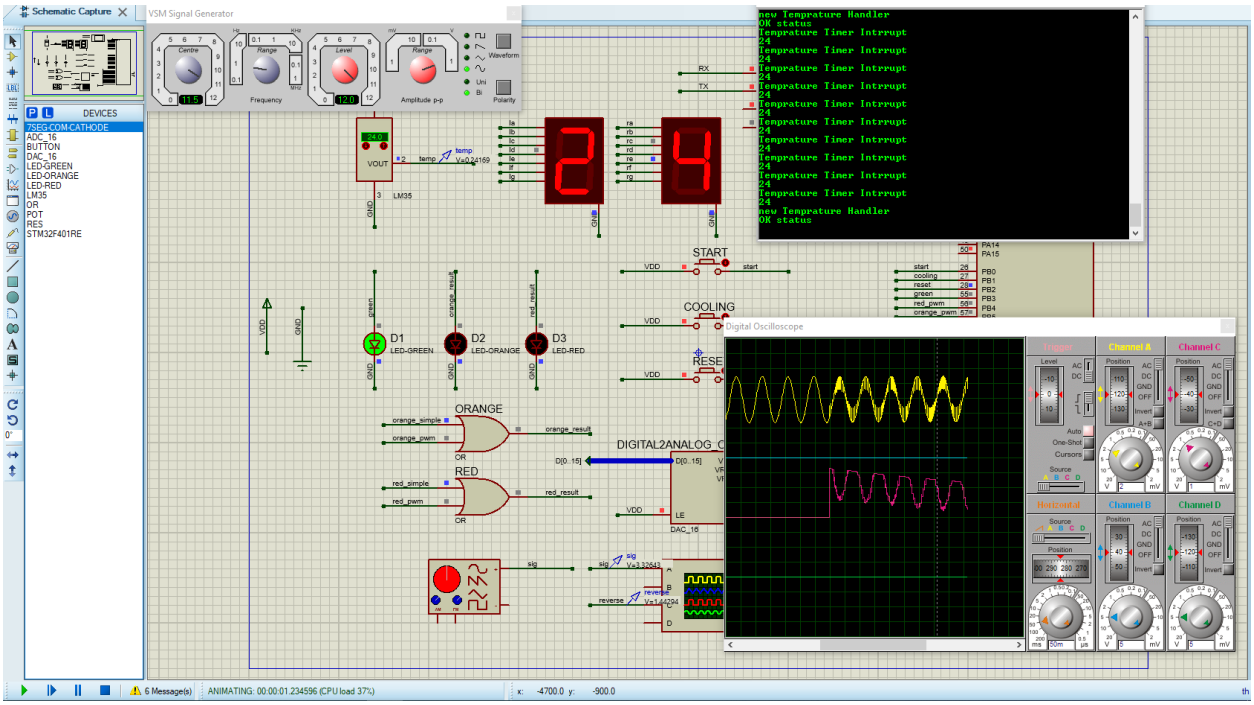
- **SIGNAL GENERATOE :**

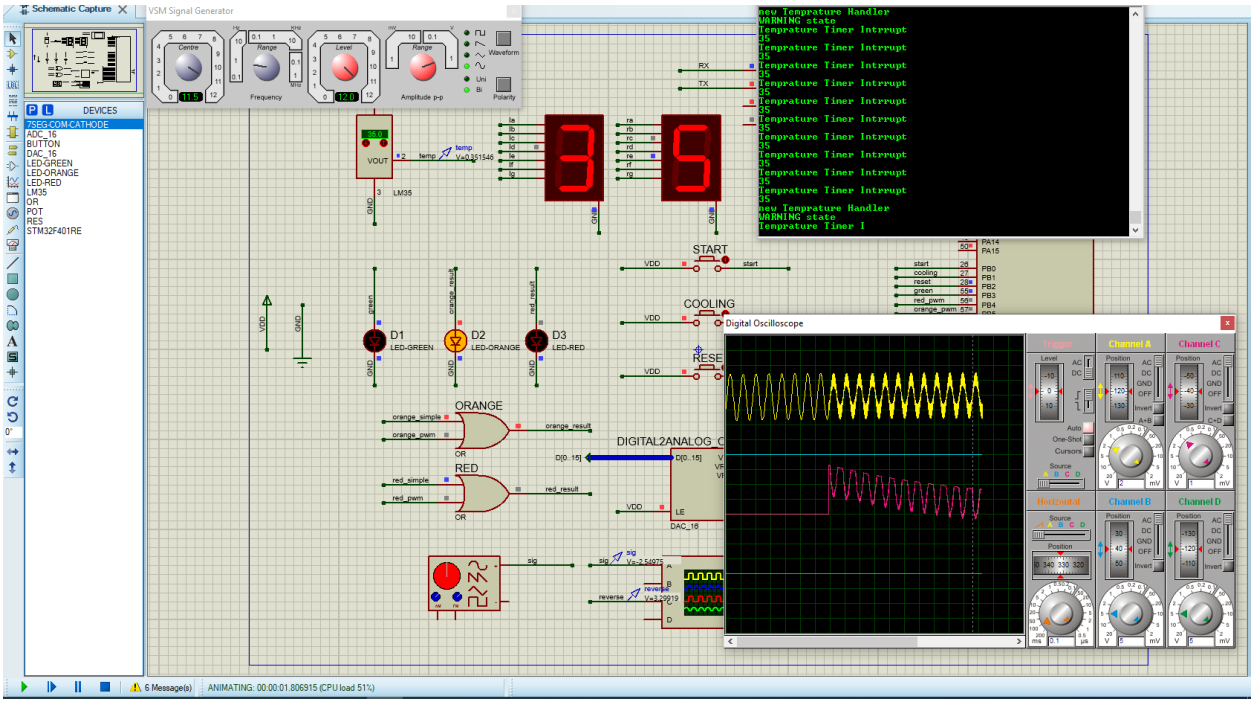
این ماژول دقیقا در سمت چپ OSCILLOSCOPE قرار دارد و کار آن تولید سیگنال های گوناگون می باشد.

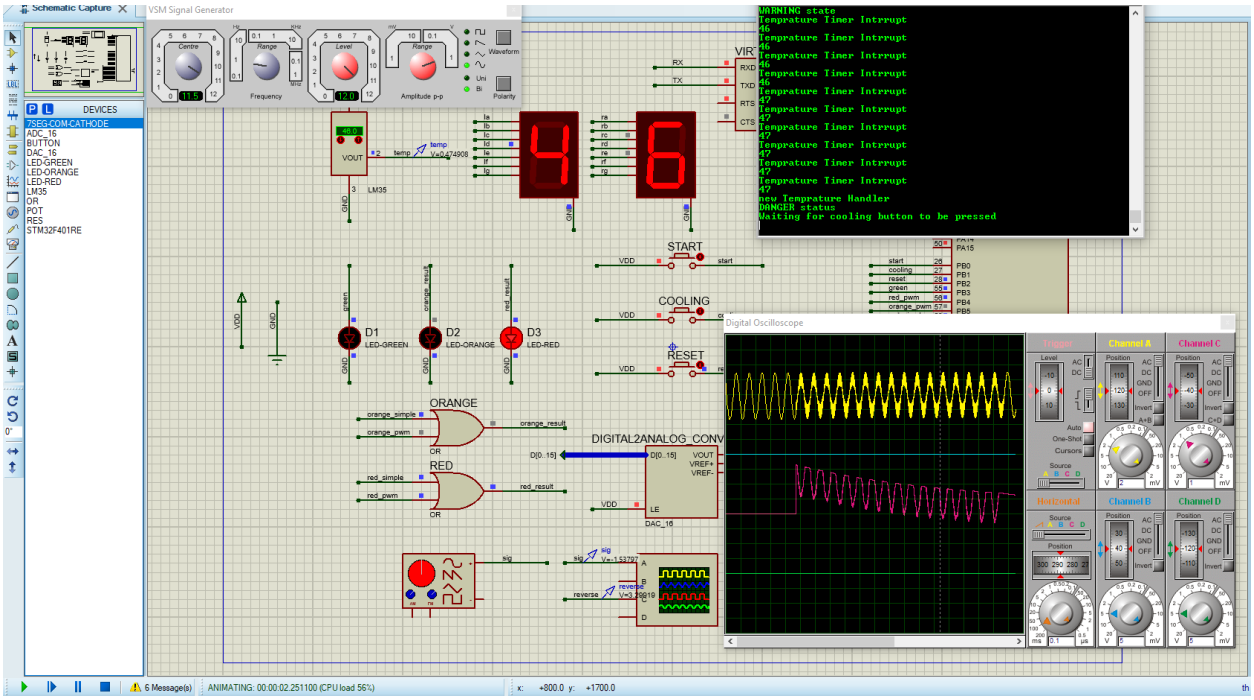
۳- نکاتی در رابطه با کار های جانبی صورت گرفته در پروتئوس:

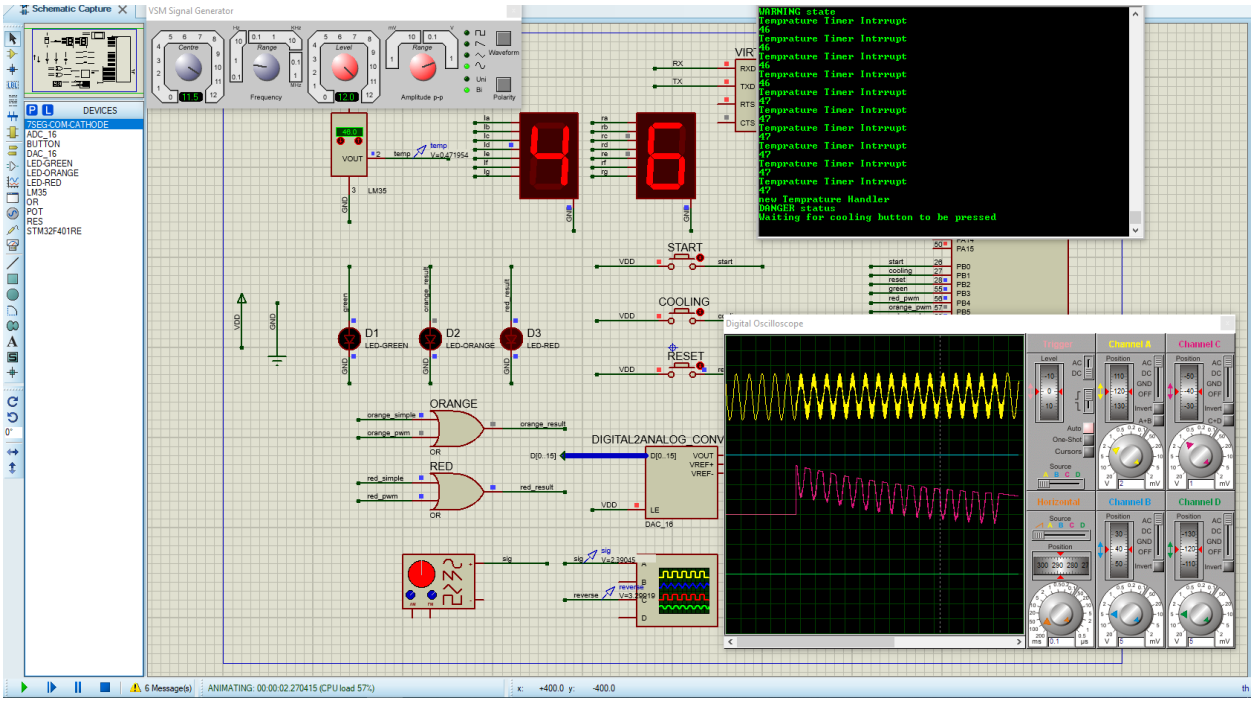
- در پروتئوس در قسمت Design در قسمت ...power rail رفتیم و مقدار vdd را برابر با 3.3 ولت قرار دادیم.
- همچنین برای قسمت virtual terminal قسمت baud آن را تغییر دادیم و با توجه به اینکه مقدار آن در stm32CubeMX برابر با 115200 بود، در پروتئوس هم همین مقدار را قرار دادیم.
- همانطور که در جدول GPIO ها ذکر شد، این سری برایفعال بودن کانال های تبدیل آنالوگ به دیجیتال خود میکرو، دو ورودی سمت راست میکرو برای ولتاژ های منبع و ولتاژ زمین را وصل کردیم.
- برای لاگ گرفتن و پیدا کردن مشکلات موجود در کد و یا نرم افزار، ماژول هایی برای اندازه گیری ولتاژ در جاهای مختلف قرار داده شده است.

۴- شکل های پروتئوس درحال اجرا شدن:



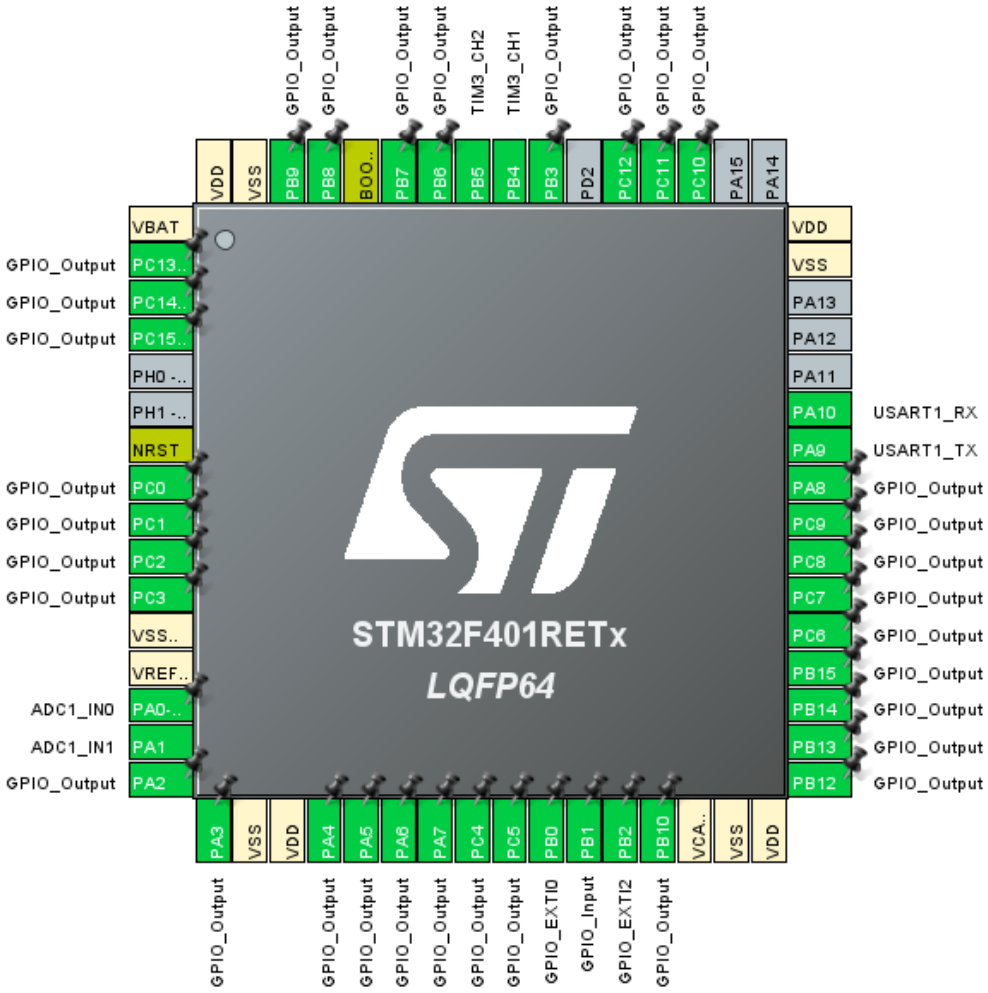



WARNING state
Temprature Timer Intrrupt
46
Temprature Timer Intrrupt
46
Temprature Timer Intrrupt
46
Temprature Timer Intrrupt
47
Temprature Timer Intrrupt
47
Temprature Timer Intrrupt
47
Temprature Timer Intrrupt
47
Temprature Timer Intrrupt
47
new Temprature Handler
DANGER status
Waiting for cooling button to be pressed
START
COOLING
RESET
D1
LED-GREEN
D2
LED-ORANGE
D3
LED-RED
ORANGE
RED
DIGITAL2ANALOG_CONV
DAC_16
LM35
Digital Oscilloscope
ANIMATING: 00:00:02.251100 (CPU load 56%)


WARNING state
Temprature Timer Intrrupt
46
Temprature Timer Intrrupt
46
Temprature Timer Intrrupt
46
Temprature Timer Intrrupt
47
Temprature Timer Intrrupt
47
Temprature Timer Intrrupt
47
Temprature Timer Intrrupt
47
Temprature Timer Intrrupt
47
new Temprature Handler
DANGER status
Waiting for cooling button to be pressed
START
COOLING
RESET
D1
LED-GREEN
D2
LED-ORANGE
D3
LED-RED
ORANGE
RED
DIGITAL2ANALOG_CONV
DAC_16
LM35
Digital Oscilloscope
ANIMATING: 00:00:02.270415 (CPU load 57%)

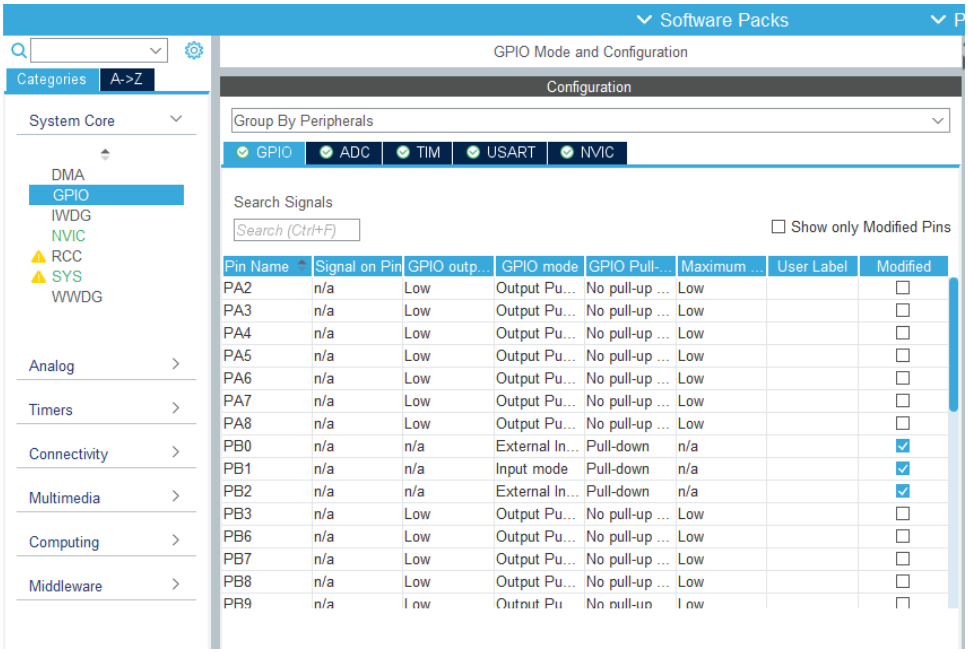
:Stm32CubeMx

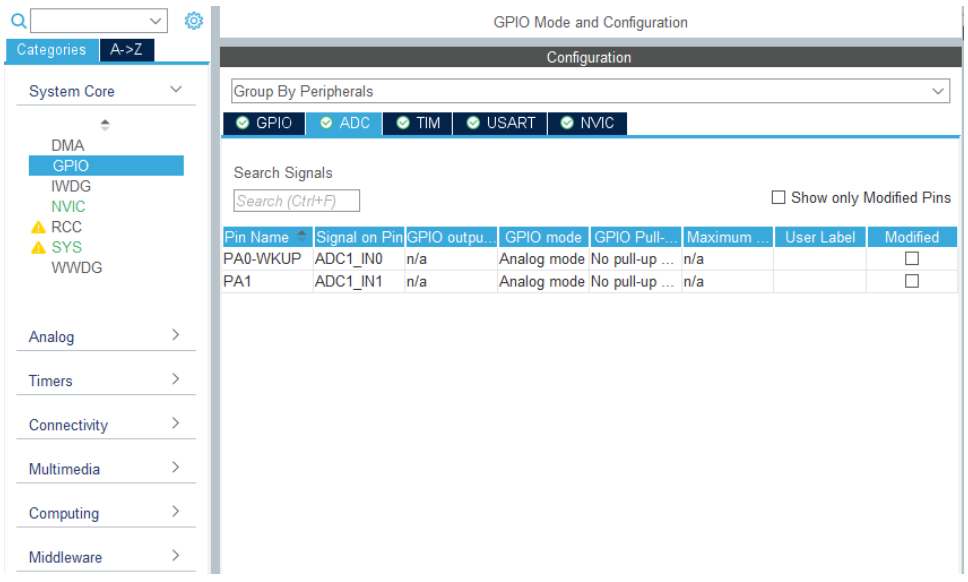
عکس قسمت های مختلف آن را قرار می دهیم ولی در ابتدا عکس کلی میکرو را قرار می دهیم:

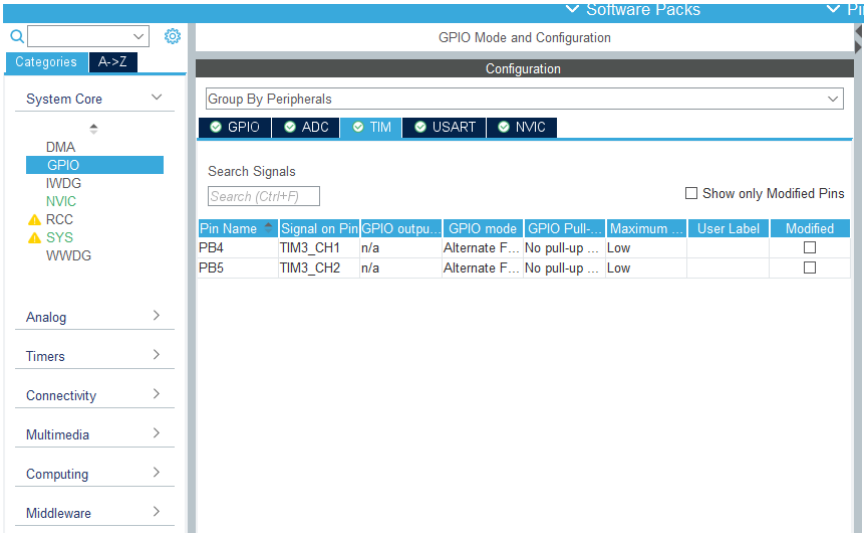


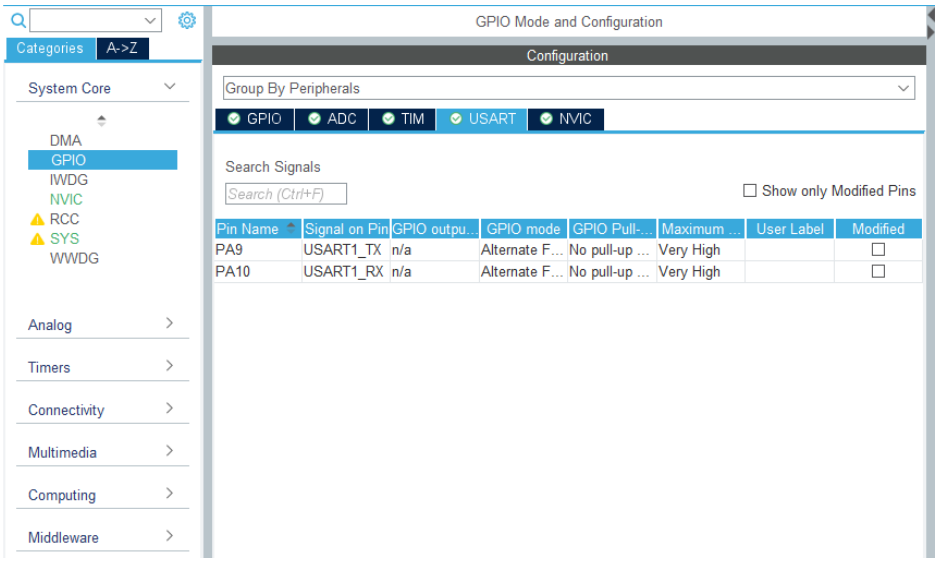
در این قسمت عکس از تنظیمات قسمت های مختلف نرم افزار stm32CubeMX را قرار می دهیم:

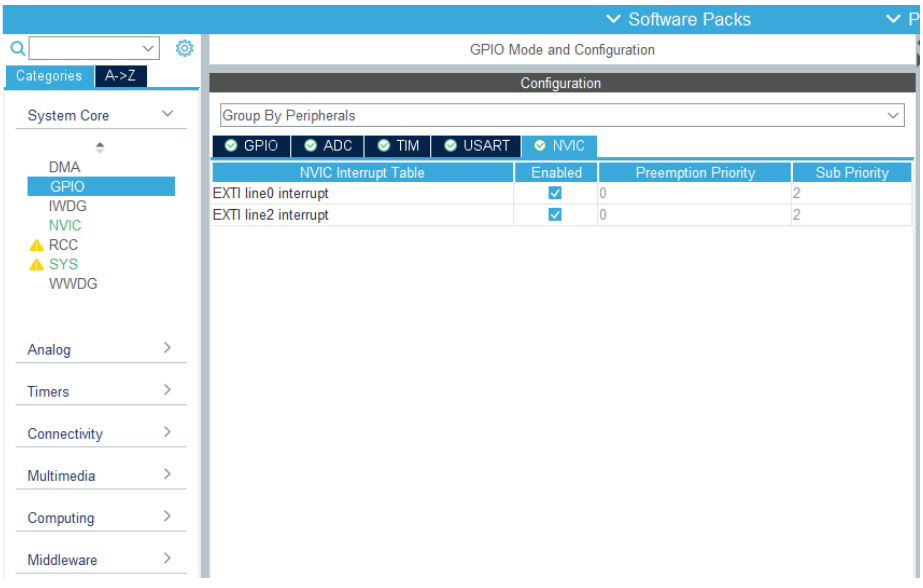
- **Pinout & Configuration :**
  - **GPIO:**



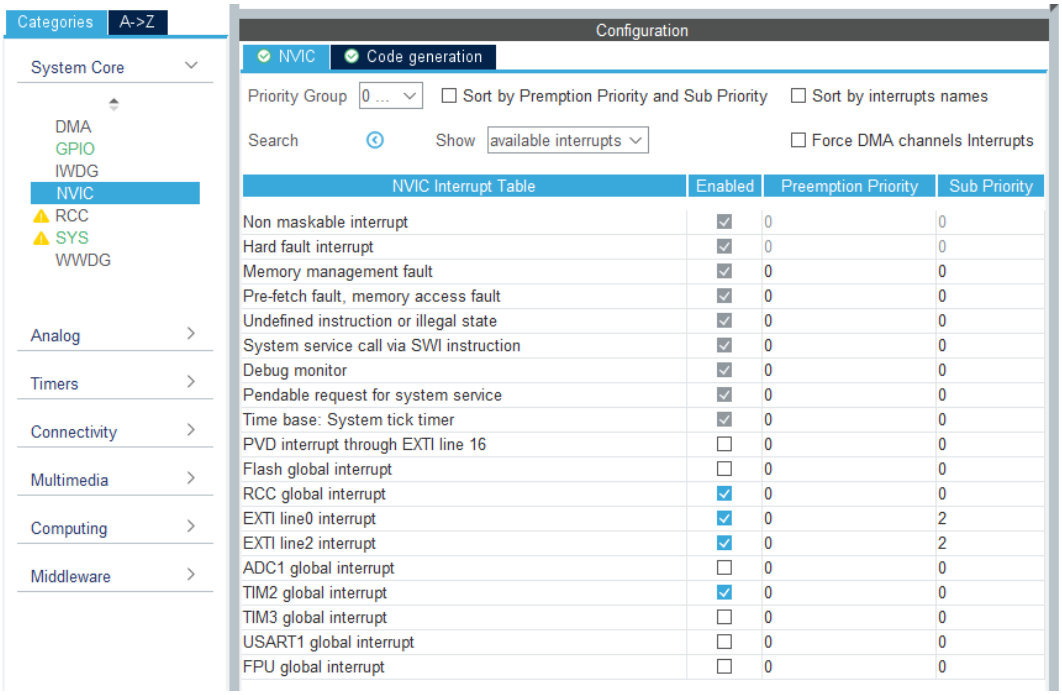



Software Packs
GPIO Mode and Configuration
Configuration
Categories
A->Z
System Core
DMA
GPIO
IWDG
NVIC
RCC
SYS
WWDG
Analog
Timers
Connectivity
Multimedia
Computing
Middleware
Group By Peripherals
GPIO
ADC
TIM
USART
NVIC
Search Signals
Search (Ctrl+F)
Show only Modified Pins
Pin Name
Signal on Pin
GPIO outpu...
GPIO mode
GPIO Pull-...
Maximum ...
User Label
Modified
PB4
TIM3_CH1
n/a
Alternate F...
No pull-up ...
Low
PB5
TIM3_CH2
n/a
Alternate F...
No pull-up ...
Low

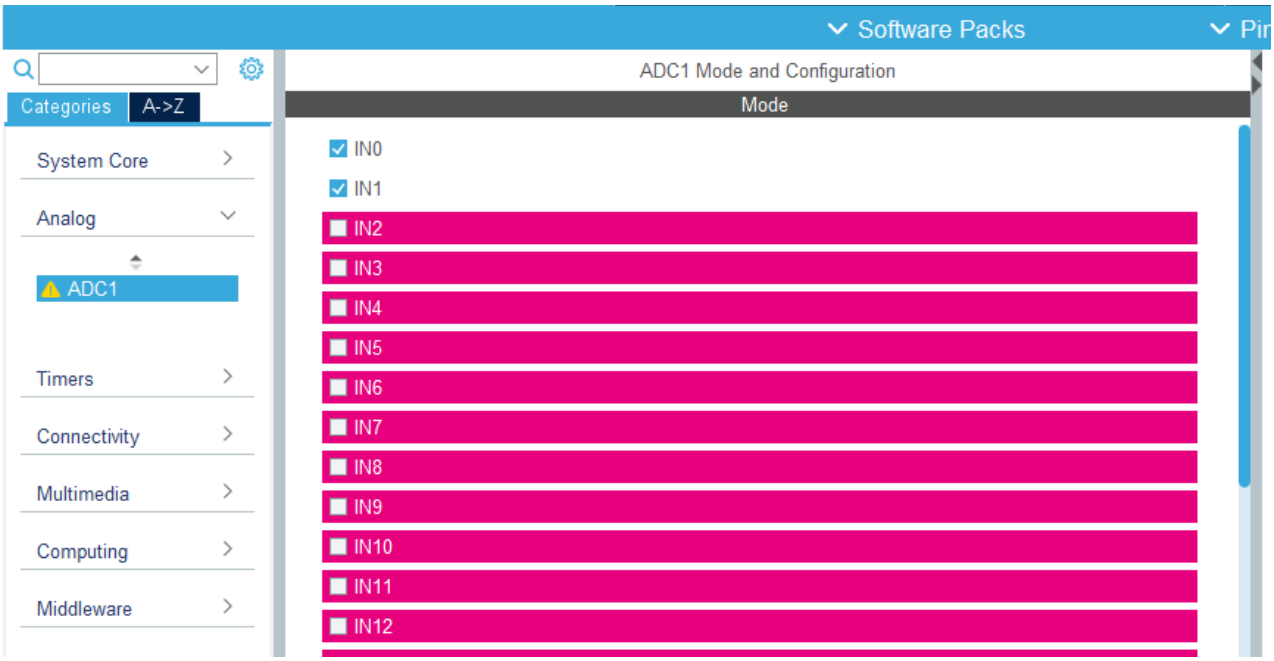

GPIO Mode and Configuration
Configuration
Categories
A->Z
System Core
DMA
GPIO
IWDG
NVIC
RCC
SYS
WWDG
Analog
Timers
Connectivity
Multimedia
Computing
Middleware
Group By Peripherals
GPIO
ADC
TIM
USART
NVIC
Search Signals
Search (Ctrl+F)
Show only Modified Pins
Pin Name
Signal on Pin
GPIO outpu...
GPIO mode
GPIO Pull-...
Maximum ...
User Label
Modified
PA9
USART1_TX
n/a
Alternate F...
No pull-up ...
Very High
PA10
USART1_RX
n/a
Alternate F...
No pull-up ...
Very High

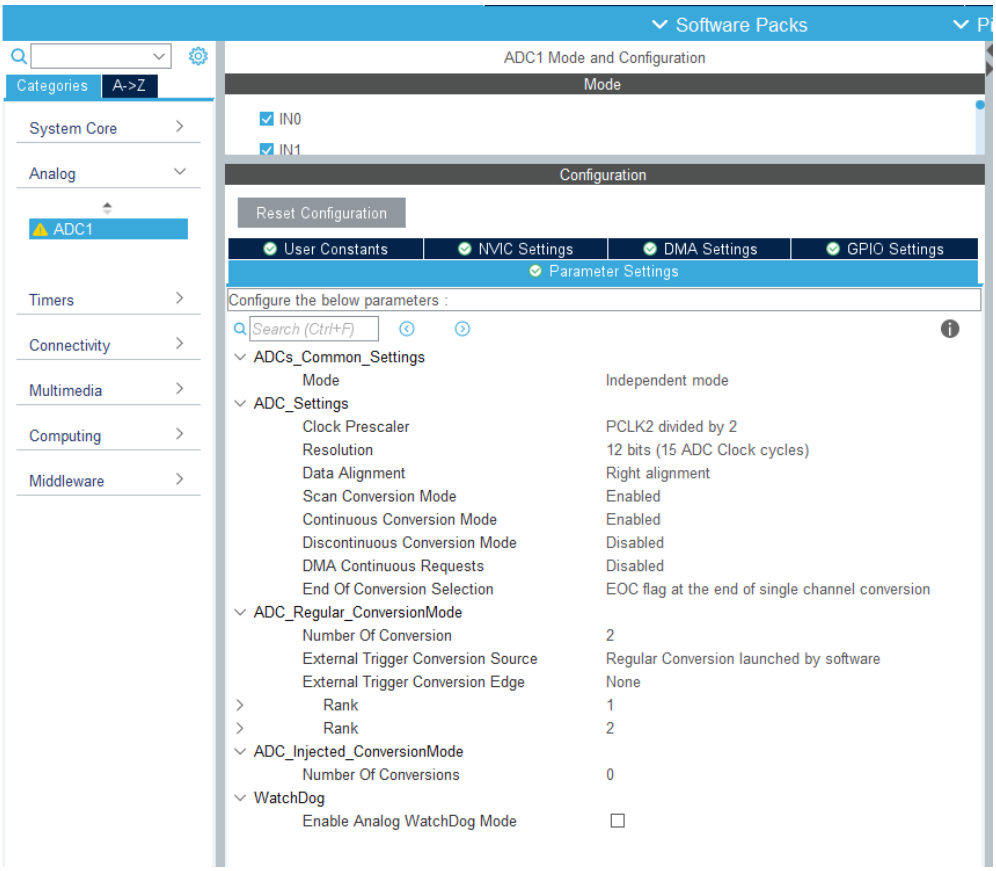

Software Packs
GPIO Mode and Configuration
Configuration
Categories
A->Z
System Core
DMA
GPIO
IWDG
NVIC
RCC
SYS
WWDG
Analog
Timers
Connectivity
Multimedia
Computing
Middleware
Group By Peripherals
GPIO
ADC
TIM
USART
NVIC
NVIC Interrupt Table
Enabled
Preemption Priority
Sub Priority
EXTI line0 interrupt
0
2
EXTI line2 interrupt
0
2

:NVIC ○

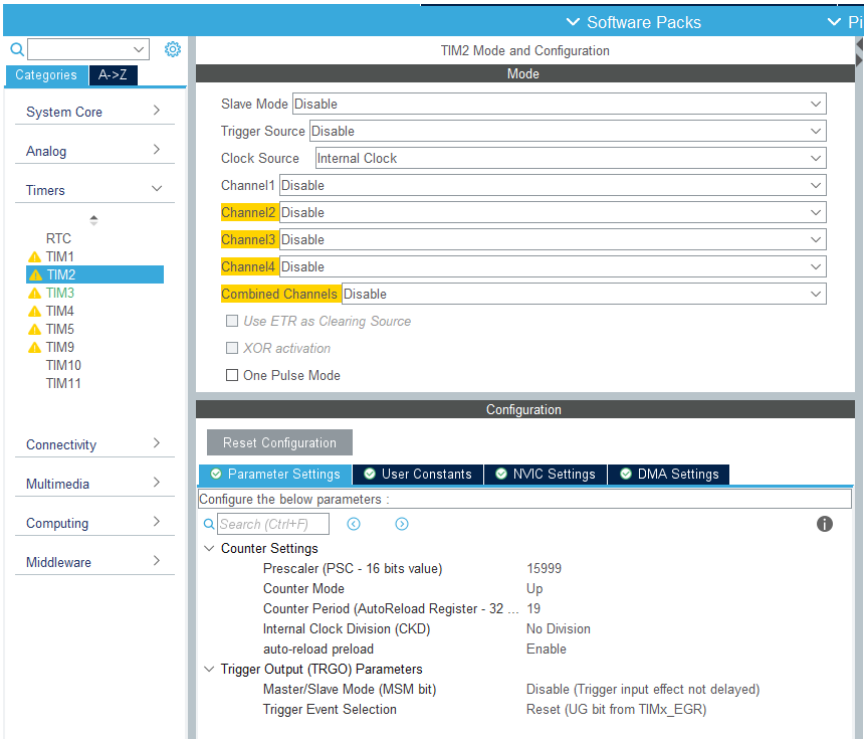

Categories
A->Z
System Core
DMA
GPIO
IWDG
NVIC
RCC
SYS
WWDG
Analog
Timers
Connectivity
Multimedia
Computing
Middleware
Configuration
NVIC
Code generation
Priority Group
0 ...
Sort by Premption Priority and Sub Priority
Sort by interrupts names
Search
Show
available interrupts
Force DMA channels Interrupts
NVIC Interrupt Table
Enabled
Preemption Priority
Sub Priority
Non maskable interrupt 0 0
Hard fault interrupt 0 0
Memory management fault 0 0
Pre-fetch fault, memory access fault 0 0
Undefined instruction or illegal state 0 0
System service call via SWI instruction 0 0
Debug monitor 0 0
Pendable request for system service 0 0
Time base: System tick timer 0 0
PVD interrupt through EXTI line 16 0 0
Flash global interrupt 0 0
RCC global interrupt 0 0
EXTI line0 interrupt 0 2
EXTI line2 interrupt 0 2
ADC1 global interrupt 0 0
TIM2 global interrupt 0 0
TIM3 global interrupt 0 0
USART1 global interrupt 0 0
FPU global interrupt 0 0

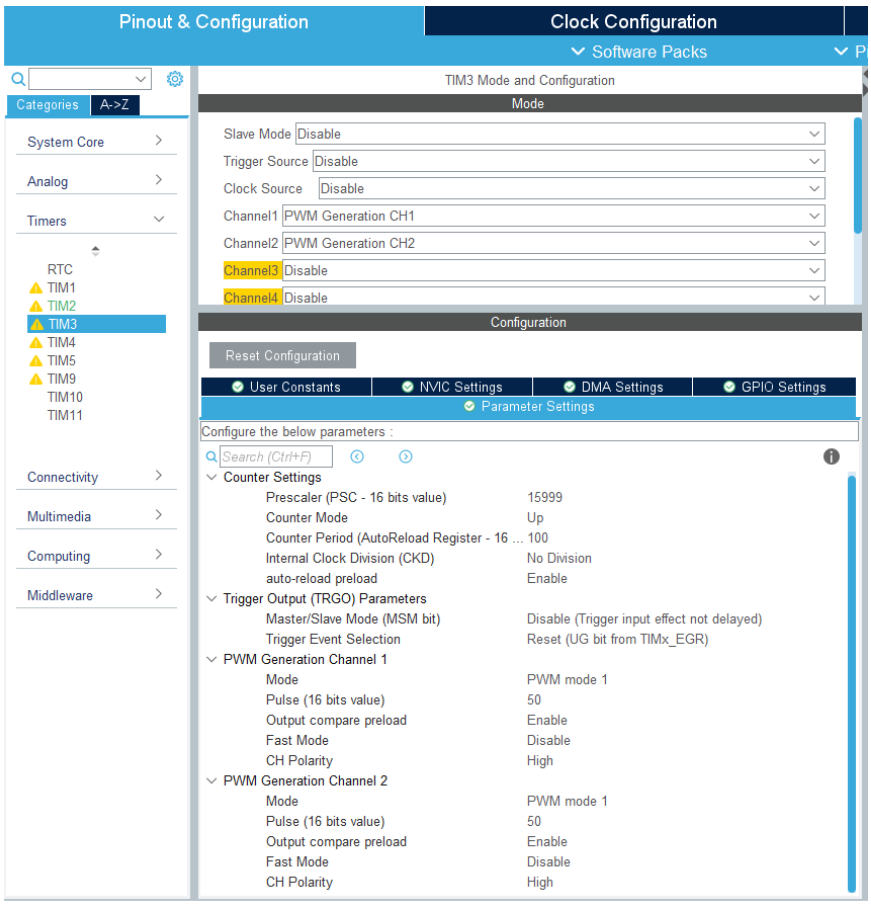
:Analog ○


Software Packs
Pin
Categories
A->Z
System Core
Analog
ADC1
Timers
Connectivity
Multimedia
Computing
Middleware
ADC1 Mode and Configuration
Mode
IN0
IN1
IN2
IN3
IN4
IN5
IN6
IN7
IN8
IN9
IN10
IN11
IN12

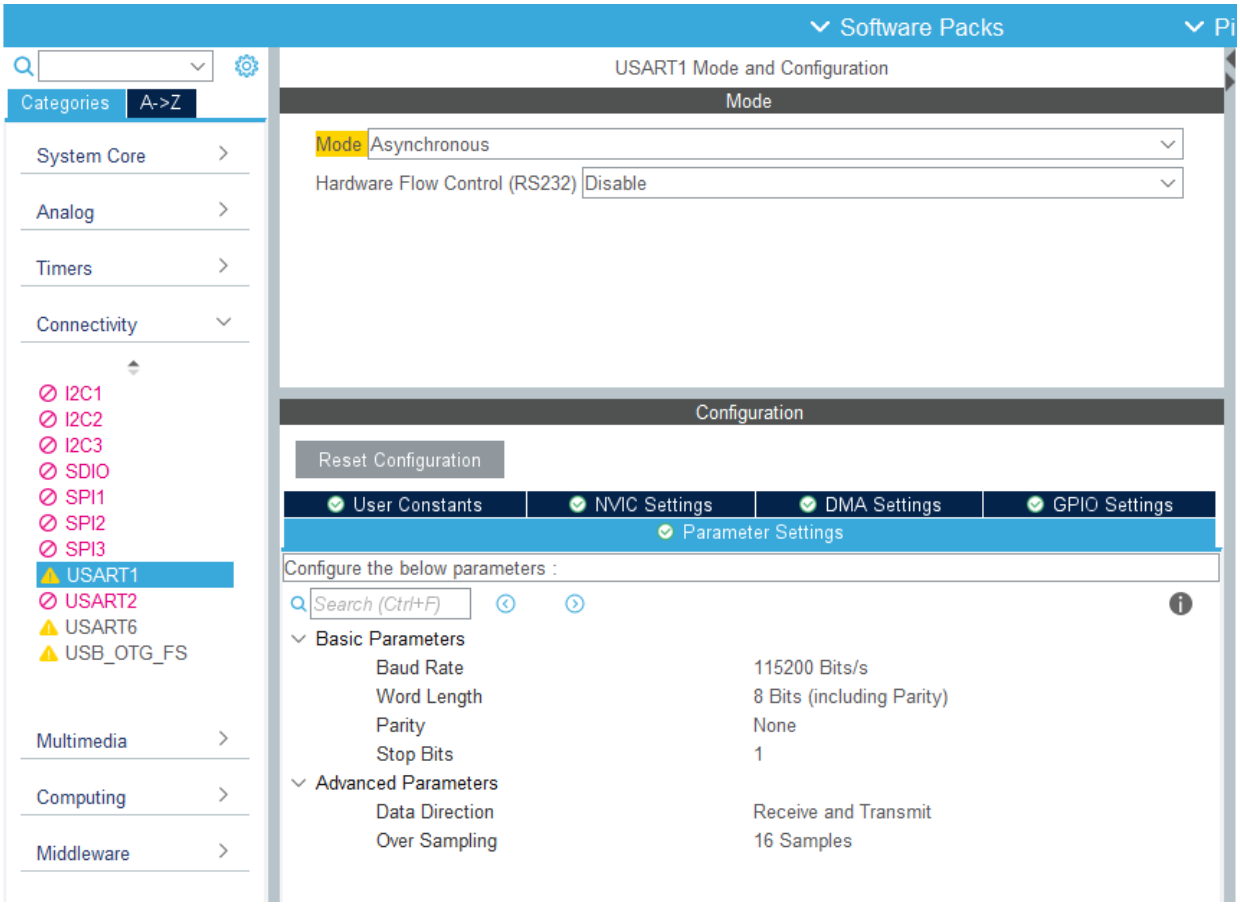

Software Packs
Categories
A->Z
System Core
Analog
ADC1
Timers
Connectivity
Multimedia
Computing
Middleware
ADC1 Mode and Configuration
Mode
IN0
IN1
Configuration
Reset Configuration
User Constants
NVIC Settings
DMA Settings
GPIO Settings
Parameter Settings
Configure the below parameters :
Search (Ctrl+F)
ADCs_Common_Settings
Mode
Independent mode
ADC_Settings
Clock Prescaler
PCLK2 divided by 2
Resolution
12 bits (15 ADC Clock cycles)
Data Alignment
Right alignment
Scan Conversion Mode
Enabled
Continuous Conversion Mode
Enabled
Discontinuous Conversion Mode
Disabled
DMA Continuous Requests
Disabled
End Of Conversion Selection
EOC flag at the end of single channel conversion
ADC_Regular_ConversionMode
Number Of Conversion
2
External Trigger Conversion Source
Regular Conversion launched by software
External Trigger Conversion Edge
None
Rank
1
Rank
2
ADC_Injected_ConversionMode
Number Of Conversions
0
WatchDog
Enable Analog WatchDog Mode

:Timers ○

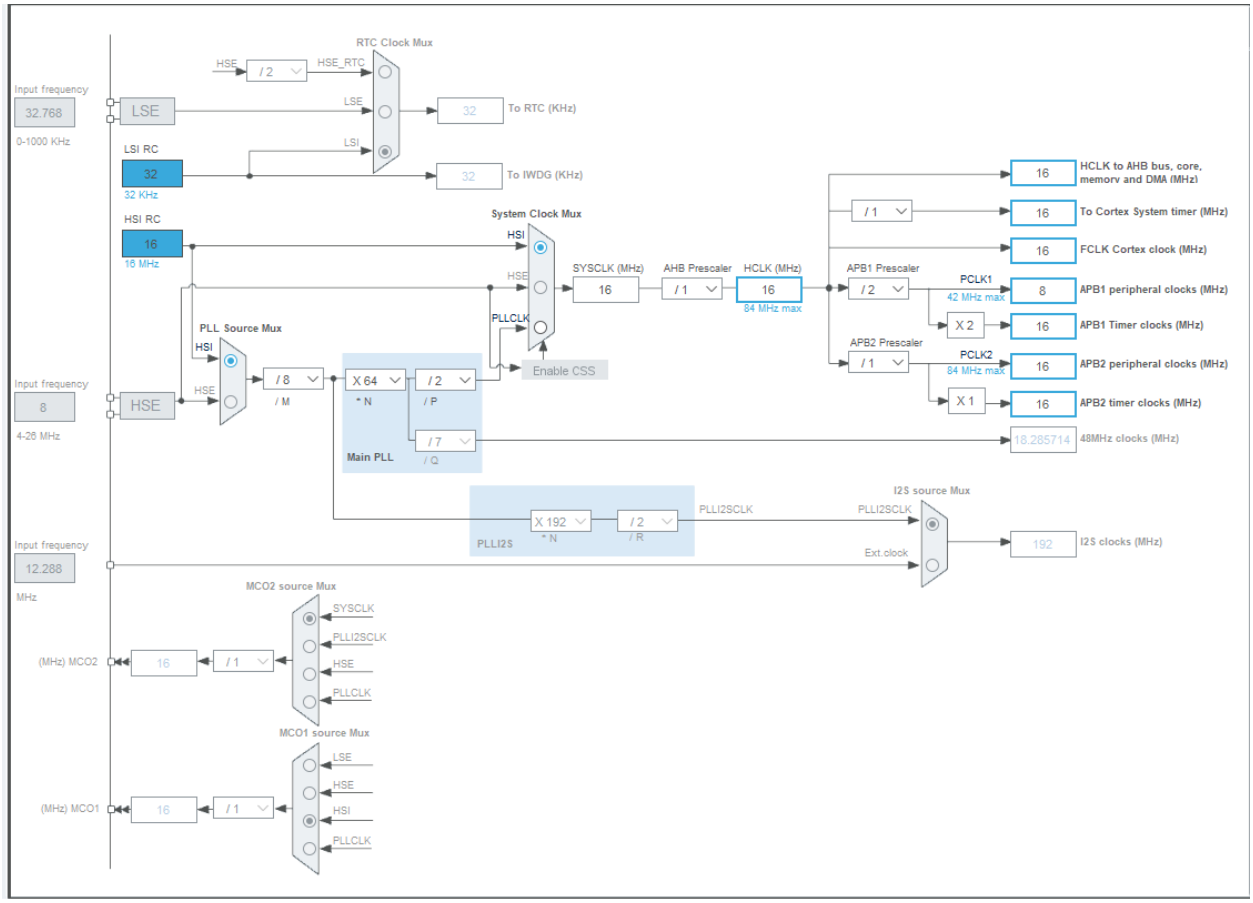

Software Packs
TIM2 Mode and Configuration
Mode
Slave Mode Disable
Trigger Source Disable
Clock Source Internal Clock
Channel1 Disable
Channel2 Disable
Channel3 Disable
Channel4 Disable
Combined Channels Disable
Use ETR as Clearing Source
XOR activation
One Pulse Mode
Configuration
Reset Configuration
Parameter Settings
User Constants
NVIC Settings
DMA Settings
Configure the below parameters :
Search (Ctrl+F)
Counter Settings
Prescaler (PSC - 16 bits value) 15999
Counter Mode Up
Counter Period (AutoReload Register - 32 ... 19
Internal Clock Division (CKD) No Division
auto-reload preload Enable
Trigger Output (TRGO) Parameters
Master/Slave Mode (MSM bit) Disable (Trigger input effect not delayed)
Trigger Event Selection Reset (UG bit from TIMx_EGR)
Categories
A->Z
System Core
Analog
Timers
RTC
TIM1
TIM2
TIM3
TIM4
TIM5
TIM9
TIM10
TIM11
Connectivity
Multimedia
Computing
Middleware


Pinout & Configuration
Clock Configuration
Software Packs
TIM3 Mode and Configuration
Mode
Slave Mode Disable
Trigger Source Disable
Clock Source Disable
Channel1 PWM Generation CH1
Channel2 PWM Generation CH2
Channel3 Disable
Channel4 Disable
Configuration
Reset Configuration
User Constants
NVIC Settings
DMA Settings
GPIO Settings
Parameter Settings
Configure the below parameters :
Search (Ctrl+F)
Counter Settings
Prescaler (PSC - 16 bits value) 15999
Counter Mode Up
Counter Period (AutoReload Register - 16 ... 100
Internal Clock Division (CKD) No Division
auto-reload preload Enable
Trigger Output (TRGO) Parameters
Master/Slave Mode (MSM bit) Disable (Trigger input effect not delayed)
Trigger Event Selection Reset (UG bit from TIMx_EGR)
PWM Generation Channel 1
Mode PWM mode 1
Pulse (16 bits value) 50
Output compare preload Enable
Fast Mode Disable
CH Polarity High
PWM Generation Channel 2
Mode PWM mode 1
Pulse (16 bits value) 50
Output compare preload Enable
Fast Mode Disable
CH Polarity High
Categories
A->Z
System Core
Analog
Timers
RTC
TIM1
TIM2
TIM3
TIM4
TIM5
TIM9
TIM10
TIM11
Connectivity
Multimedia
Computing
Middleware

:Connectivity ○



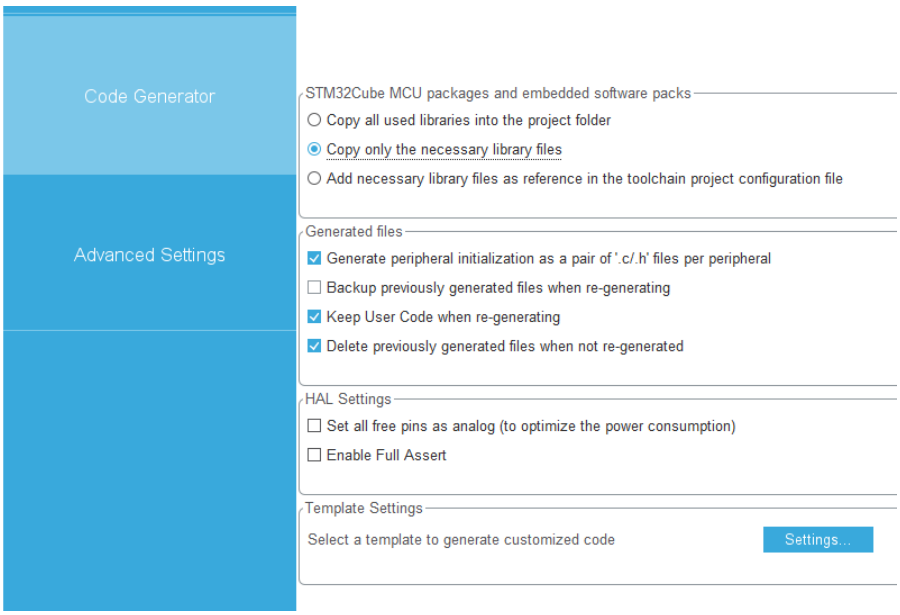
- :Clock Configuration



- **Project Manager:**
  - **Project:**

در این قسمت تنها Toolchain / IDE را برابر با MDK-ARM قرار می دهیم تا کد های قابل استفاده برای نرم ا فزار پروتئوس را برایمان تولید کند.

  - **Code Generator:**



در این قسمت گزینه generate peripherial initialization as pair .c / .h files per peripherial رو فعال می کنیم تا تنظیمات (تابع های مورد استفاده برای تنظیماتشان در واقع) قسمت های مختلف مثل GPIO ها، TIMER ها و ... در فایل های جدا قرار بگیرند تا خود کد اصلی مان خلوت تر شود.

  - **Advanced setting:**

در این قسمت تغییری ایجاد نکردیم.

- **Tools:**

در این قسمت تغییری ایجاد نکردیم.

Keil:

ابتدا به چند تا از موارد و مشکلاتی که به آنها برخوردیم و راه حل هایشان اشاره می کنیم:

۱- در ابتدا برای استفاده از VIRTUAL TERMINAL به دلیل تابع های مورد استفاده وقتی که build میکردیم، به ما error مربوط به ورژن کامپایلر میداد.
برای رفع این مشکل، در قسمت Options for target در قسمت target ورژن ARM Compiler در قسمت Code Generation را بالا بردیم. (در سیستم خودمان مقدار آن را v6.16 قرار دادیم.)

۲- برای راحتی استفاده از توابع مربوط به VIRTUAL TERMINAL از یک سری library هایی استفاده کردیم. این library ها عبارتند از stdio.h و string.h .

۳- در ورژن قبلی مورد استفاده پروتئوس (ورژن 8.13) میکرو قادر به خواندن مقدار آنالوگ نبود و برای فهمیدن این موضوع
این بسیار بسیار .... بسیار زیادی تلف شد! سپس مجبور به نصب ورژن قدیمی تر آن (ورژن 8.12) شدیم.

۴- در این ورژن پروتئوس، پروژه را از ابتدا ساختیم، اما به مشکلات زیادی برخرودیم که کار با این نرم افزار را عملا غیر ممکن می کرد. مشکلاتی مانند:
- کرش کردن مداوم و بی دلیل نرم افزار !
- اجرا نشدن درست و simulation های عجیب!
- طول کشیدن مدت زمان به نسبت زیاد برای لود شدن اجزا سیستم و شروع به اجرا شدن آن!(البته بعد از لود شدن و شروع به اجرا کردن سریعا از نرم افزار خارج میشد!)
- یکسان نبودن حاصل اجرای کد های یکسان در دو سیستم مختلف!
- کند شدن بسیار بسیار شدید نرم افزار و اجرای کند آن!(بعضا حتی قبل از اجرا شدن مقدار کمی نرم افزار کرش میکرد!)

به دلیل مشکلات ذکر شده کار ها و روش های مختلفی را امتحان کردیم. روشی که باعث پایداری به نسبت بیشتر و اجراب به نسبت بهتر نرم افزار شد(همچنان به شدت کند می باشد به طوری که حدودا ک میلی ثانیه نرم افزار پروتئوس، در حدود یک ثانیه واقعی طول می کشد!) این بود که در یک سیستم دیگر یک پروژه با ورژن 8.13 ساخته شد. سپس آن را بر روی این سیستم قرار دادیم و اجزا را روی آن قرار دادیم.

۵- در این ورژن از پروتئوس و کامپایلر کیل، فایل تولید شده در فراخوانی interrupt های تایمری مشکل داشت و تابع TIM2_IRQHandler(که به وسیله stm32CubeMX تولید شده بود)، تابع مورد نظر را فراخوانی نمیکرد!!!! و با پیاده سازی callback آن، این callback فراخوانی نمیشد!
برای رفع این مشکل، در قسمت Application/User/Core در فایل stm32f4xx_it.c خط های 245 تا 254 (که مربوط به پیاده سازیTIM2_IRQHandler می باشد) را کامنت کردیم. سپس در فایل اصلی مان به جای پیاده سازی callback ، خود این تابع را فراخوانی کردیم.

روال کلی کار:

در این کد، با ست کردن TIM2_IRQHandler، هر 20 میلی ثانیه یک بار در جاهایی که لازم است دما را اندازه گیری میکنیم و اگر برای مثال دهمین بار باشد، در کد آنها را مشخص میکنیم و عملیات های لازم را انجام می دهیم.

برای چشمک زن بودن LED ها از PWM استفاده می کنیم.

برای کلید های start و reset از interrupt ها استفاده می کنیم در حالی که برای دکمه cooling از روش polling ستفاده می کنیم.

حال به سراغ توضیح تیکه کد های مختلف می پردازیم:

- **Includeها:**
  - **"stdio.h":** از این library برای اینکه بتوانی در virtual terminal یک چیزی را چاپ کنیم (مخصوصا برای قسمتی که می خواستیم در تابع printNumber یک عدد را به آرایه (pointer و ...) تبدیل کنیم).
  - **"string.h":** از این library برای کار با string ها استفاده کردیم.

تیکه کد زیر مربوط به این قسمت می باشد:

```
#include "stdio.h"
#include "string.h"
```

- **Enum:**

برای اینکه بتوانی در این سوال، حالت های خواسته شده را پیاده سازی کنیم (مخصوصا که بعضی از آنها در خود صورت سوال ذکر شده اند)، یک enum تعریف کردیم که دارای حالت های زیر می باشد:

  - **BEFOR_START:** در این حالت ما منتظر هستیم تا کاربر دکمه start را فشار دهد تا کار سیستم آغاز شود.
  - **CHECKING:** در این حالت، یا کاربر دکمه start را فشار داده است که ما به این وضعیت آمده ایم یا اینکه پس از خاموش شدن سیستم کاربر دکمه reset را فشار داده است که به این وضعیت آمده ایم. در هر حال، این وضعیت به این معناست که ما باید LED عه نارنجی را به صورت چشمک زن روشن کنیم و دما را اندازه گیری کنیم و سپس به ادامه کار بپردازیم.

- **OK:** این وضعیت به این معنی می باشد که سیستم در حال کار است و آخرین باری که دما اندازه گیری شده است، دما کمتر از 35 درجه بوده است و باید از سیگنال سینوسی ورودی نمونه برداری شود و ... .

- **WARNING:** این وضعیت به این معنی است که سیستم در حال کار است و آخرین باری که در سیستم دمای محیط اندازه گیری شد، دمای محیط بین 35 درجه تا 45 درجه بوده است. سیستم باید به کار خود ادامه دهد، و از سیگنال سینوسی ورودی نمونه برداری کند و ... .

- **DANGER:** این وضعیت باید سیستم متوقف شود و 500 میلی ثانیه منتظر بماند و ببیند که آیا دکمه cooling فشار داده می شود یا خیر و در هر دو صورت کار ها ی لازم را انجام دهد(همچنین در همین حال LED قرمز هم باید به صورت چشمک زن باشد). در همین حال سیستم کار نمونه برداری از سیگنال ورودی را متوقف می کند.

- **COOLING:** این وضعیت به این معنی است که کاربر دکمه cooling را فشار داده است. در این حالت باید LED قرمز به صورت ثابت روشن باشد و 2 ثانیه منتظر بمانیم تا ببینیم که آیا دما پاین می آید یا خیر(بر خلاف صورت سوال برای متوجه شدن پیاده سازی این قسمت، ما دمای مورد نیاز برای ادامه کار سیستم در این حالت را همان زیر 46 بودن درنظر گرفتیم که پیاده سازی و تست آن ممکن باشد). اگر پایین آمد که سیستم به کار خود ادامه می دهد و نمونه برداری از شکل موج سیگنال ورودی را دوباره شرو میکند، و اگر پایین نیامده باشد، سیستم LED قرمز را به مدت 500 میلی ثانیه به صورت چشمک زن روشن می گذارد و سپس خاموش می شود تا کاربر دکمه restart را فشار دهد و ... .

- **TURN_OFF:** در این حالت سیستم خاموش است و هیچ کاری انجام نمی دهد.

تیکه کد زیر مربوط به این enum می باشد:

```
enum State{
    BEFOR_START,
    CHECKING,
    OK,
    WARNING,
    DANGER,
    COOLING,
    TURN_OFF
};
```

- **Define ها:**
  - **LED ها:** در ابتدای کد همانطور که مشخص شده است، یک سری define هایی انجام دادیم برای پورت LED ها و همچنین برای پین های LED ها که در هنگام استفاده از آ«ها در کد، مشکلی بوجود نیاید و خوانا تر باشد.
    تیکه کد زیر مربوط به این قسمت می باشد:

```
//////////////////////////////////////////////////// LED ////////////////////////////////////////////////////
#define LED_port GPIOB
#define green GPIO_PIN_3
#define orange GPIO_PIN_7
#define red GPIO_PIN_6
```

  - **سون سگمنت سمت چپ(مربوط به دهگان دما):** در این قسمت یک سری define هایی برای پورت و پین این سون سگمنت انجام دادیم تا در هنگام استفاده از آن راحت تر باشیم. تیکه کد زیر برای این قسمت می باشد:

```
//////////////////////////////////////////////////// left 7-Segment ////////////////////////////////////////////////////
#define left_port GPIOA

#define la GPIO_PIN_2
#define lb GPIO_PIN_3
#define lc GPIO_PIN_4
#define ld GPIO_PIN_5
#define le GPIO_PIN_6
#define lf GPIO_PIN_7
#define lg GPIO_PIN_8
```

  - **سون سگمنت سمت راستی(مربوط به یکان دما):** در این قسمت یک سری define هایی برای پورت و پین های این سون سگمنت انجام شده است. تیکه کد زیر مربوط به این قسمت می باشد:

```
//////////////////////////////////////////////////// right 7-Segment ////////////////////////////////////////////////////
#define right_port GPIOB

#define ra GPIO_PIN_8
#define rb GPIO_PIN_9
#define rc GPIO_PIN_10
#define rd GPIO_PIN_12
#define re GPIO_PIN_13
#define rf GPIO_PIN_14
#define rg GPIO_PIN_15
```

  - **MASK ها:** در این قسمت یک سری mask هایی تعریف کردیم تا در هنگام استفاده از آنها راحت تر باشیم. تیکه کد زیر برای این قسمت می باشد:

```
//////////////////////////////////////////////////// MASK ////////////////////////////////////////////////////
#define SET1(x)  (1ul << x)
#define SET0(x)  (~SET1(x))
```

- **متغییر ها:**

متغییر های موجود در این کد و استفاده از آنها به صورت زیر است:

- **Temperature:** از این متغییر برای این استفاده کردیم که هر وقت قرار بود که هر 20 میلی ثانیه یک بار دمای محیط خوانده شود، وقتی که مقدار آنالوگ را خواندیم و با انجام کار های مورد نیاز دمای محیط را محاسبه کردیم، حاصل را با این متغییر جمع کنیم و هر وقت که ۱۰ بار دما را اندازه گیری کردیم، این مقدار را تقسیم بر 10 می کنیم تا دمای محیط بدست بیاید.(در اصل تا قبل از این تقسیم بر 10 کردن، دمای محیط در هر سری که اندازه گیری شده است، به صورت تجمعی در این متغییر نگهداری می شود. بنابراین در بار دهم، با تقسیم بر 10 کردن آن، میانگین دمای محیط در این 10 سری بدست می آید.)

- **tempCounter:** این متغییر برای این است که هر وقت که دمای محیط را با استفاده از interrupt های تایمر پیدا کردیم(دمای کلی نه، صرفا هر بار که مقدار آنالوگ را از ورودی خواندیم و به دما تبدیل کردیم مقدار آن را) این متغییر را یکی زیاد می کنیم تا هر وقت این متغییر به 10 رسید، بدانیم که باید میانگین گیری کنیم و دمای (تا آن لحظه تجمعی را) بر 10 تقسیم کنیم.

- **Status:** این متغییر در اصل نشان دهنده وضعیت کلی سیستم می باشد و از جنس همان enum توضیح داده شده در ابتدای این قسمت می باشد.

- **externalSig:** این متغییر برای بدست آوردن مقدار آنالوگ می باشد، اما برای سیگنال ورودی است، نه برای دما. از این متغییر هنگامی که سیستم باید نمونه برداری کند، مقدار دیجیتالی که باید به DAC_16 داده شود تا برایمان مقدار آنالوگ را تولید کند، استفاده می شود.
  به صورت کلی، این متغییر مقدار معکوس شده سیگنال سینوسی ورودی می باشد.

- **توابع:**
  - **printString:**

این تابع به عنوان ورودی یک string دریافت کرده و با استفاده از کد های نوشته شده و توابع موجود برای کارکردن با virtual terminal، این string را بر روی ترمینال چاپ می کند. تیکه کد زیر مربوط به هیمن تابع است:

```
void printString(char* s){
    HAL_UART_Transmit(&huart1,(uint8_t*) s , strlen(s), HAL_MAX_DELAY);
}
```

  - **printNumber:**

این تابع به عنوان ورودی یک عدد دریافت کرده و با استفاده از کد های موجود برای تبدیل تبدیل یک عدد به string و توابع موجود برای کارکردن با virtual terminal این عدد را در ترمینال چاپ می کند. تیکه کد زیر مربوط به این قسمت می باشد:

```
void printNumber(uint32_t input){
  char number[10];
  sprintf(number, "%u", input);
  HAL_UART_Transmit(&huart1,(uint8_t*) number , 10, HAL_MAX_DELAY);
}
```

  - **newline:**

از این تابع برای این استفاده شده استکه هر وقت می خوساتیم در ترمینال به خط بعد برویم، به راحتی با استافده از این تابع این کار را انجام دهیم. تیکه کد زیر مربوط به این قسمت می باشد:

```
void newLine(void){
  char* s ="\n\r";
  HAL_UART_Transmit(&huart1,(uint8_t*) s , 2, HAL_MAX_DELAY);
}
```

  - **setLeftDigit:**

در این تابع با استفاده از define های صورت گرفته برای سون سگمنت سمت چپ (مربوط به رقم دهگان دما) و با استفاده زا ساز و کار های تعبیه شده در کتابخانه HAL برای نوشتن بر روی پین ها، با توجه به رقم دهگان دما به به عنوان ورودی به این تابع پاس داده می شود، هر یک از 7 بخش این سون سگمنت را تعیین کردیم که خاموش یا روشن باشد. عکس از کد این قسمت، به دلیل حجم زیاد آن، در اینجا آورده نشده است.

- **setRightDigit:**

دقیقا کار تابع بالا را می کند اما برای سون سگمنت سمت راست که برای رقم یکان دما می باشد. عکس از کد این قسمت هم بدلیل حجم بالای آن آورده نشده است.

- **set7Segment:**

این تابع هنگامی فراخوانی می شود که ما بخواهیم عدد دما را (متغییر temperature را) بر روب سون سگمنت ها نشان دهیم. برای این کار با توجه به اینکه سون سگمنت سمت چپی برای رقم دهگان دما می باشد، و سون سگمنت سمت راستی برای رقم یکان دما می باشد، پس باقی مانده متغییر temperature را (که در اصل می شود همان رقم یکان آن) به عنوان ورودی به تابع ست کننده سون سگمنت سمت راست (که در بالا به آن اشاره شد) می دهیم و حاصل تقسیم متغییر temperature به عدد 10 را (که در اصل همان رقم دهگان دما می باشد) به عنوان ورودی به تابع ست کننده سون سگمنت سمت چپ (که در بالا به آن اشاره شد)، می دهیم. اگر هم میخواستیم کلا سون سگمنت ها را خاتموش کنیم، به آنها عدد منفی یک را به عنوان ورودی می دهیم. تیکه کد زیر برای همین قسمت می باشد:

```
void set7Segment(int number){
  if(number == -1){
    setLeftDigit(-1);
    setRightDigit(-1);
    return;
  }
  setLeftDigit(number / 10);
  setRightDigit(number % 10);
}
```

- **getTemprature(مقدار خروجی این تابع، دمای دماسنج است):**

این تابع برای این است که مقدار آنالوگ دما (که دماسنج آن را به ما می دهد) را به عنوان ورودی دریافت کرده و مقدار آن را بخوانیم و به صورت یک عدد (از وری ولتاژ ورودی دما را به صورت یک عدد) برگردانیم.

برای این کار مراحل زیر را انجام میدهیم:

۱- ابتدا با کمک کد های موجود و ساخته شده بوسیله نرم افزار stm32CubeMX که در فولدر Application/User/Core در فایل adc.c قرار دارد، تنظیمات مربوط به ست کردن کانال صفر (کانال مورد استفاده برای خواندن مقدار دما) را انجام می دهیم.

۲- سپس با استافده از توابعی که کتابخانه HAL د راختیارمان قرار داده است برای خواندن مقدار ورودی، آن را می خوانیم.

۳- با توجه به اینکه با توجه به اینکه ولتاژ ورودی ما چه عددی بین 0 تا 3.3 است، این توابع کتابخانه HAL یک عدد بین 0 تا 4095 به ما برمیگرداند. پس ماباید این تناسب را برعکس کنیم تا بتوانی ولتاژ ورودی را پیدا کنیم.

۴- با توجه به اینکه دماسنج، ولتاژی که تولید می کند، همواره دمای محیط تقسیم بر 100 می باشد (یعنی مثلا اگر دمای محیط 27 درجه باشد، دماسنج مقدار 0.27 ولت را به ما می دهد) پس حاصل قسمت بالا را در 100 ضرب میکنیم تا دمای محیط را بدست آوریم.

تیکه کد مربوط به این قسمت، به صورت زیر می باشد:

```c
uint16_t getTemprature(void){
  ADC_ChannelConfTypeDef sConfig = {0};
  sConfig.Channel = ADC_CHANNEL_0;
  sConfig.Rank = 1;
  sConfig.SamplingTime = ADC_SAMPLETIME_3CYCLES;
  HAL_ADC_ConfigChannel(&hadc1, &sConfig);
  HAL_ADC_Start(&hadc1);
  HAL_ADC_PollForConversion(&hadc1, HAL_MAX_DELAY);
  uint16_t temp = HAL_ADC_GetValue(&hadc1);
  HAL_ADC_Stop(&hadc1);
  double res = (double)temp;
  res /= 4095;
  res *= 3.3;
  res *= 100;
  temp = res;
  return temp;
}
```

- **getExternalValue(خروج این تابع مقدار دیجیتال ورودی است):**

تقریبا کارکردی مشابه با تابع قسمت بالا دارد. این تابع برای بدست آوردن مقدار سیگنال سینوسی ورودی است.

در این تابع دیگر قسمت مربوط به تناسب تابع بالا را انجام نمیدهیم و صرفا مقدار دیجیتالی که میکرو به ما می دهد را بر میگردانیم.

```c
uint16_t getExternalValue(void){
  ADC_ChannelConfTypeDef sConfig = {0};
  sConfig.Channel = ADC_CHANNEL_1;
  sConfig.Rank = 1;
  sConfig.SamplingTime = ADC_SAMPLETIME_3CYCLES;
  HAL_ADC_ConfigChannel(&hadc1, &sConfig);
  HAL_ADC_Start(&hadc1);
  HAL_ADC_PollForConversion(&hadc1, HAL_MAX_DELAY);
  uint16_t temp = HAL_ADC_GetValue(&hadc1);
  HAL_ADC_Stop(&hadc1);
  return temp;
}
```

- **turnOffLEDs:**

خیلی از مواقع پیش می آمد که ما میخواستیم LED های روشن را خاموش کنیم، اما با توجه به اینکه از حالت های مختلف به حالت های مختلف رفتن، LED ای که روشن بود و باید خاموش میشد، متفاوت بود، یک تابع نوشتیم که هر وقت حالتمان (همام متغییر status مان) عوض میشد، این تابع صدا زده شود و تمام LED های موجود را خامو کند. تیکه کد زیر مربوط به این تابه و برای خاموش کردن تمامی LED هاست:

```c
void turnOffLEDs(void){
  HAL_GPIO_WritePin(LED_port, orange, 0);
  HAL_GPIO_WritePin(LED_port, green, 0);
  HAL_GPIO_WritePin(LED_port, red, 0);
}
```

- **stopBlinking:**

این تابع تقریبا مشابه با تابع بالا عمل می کند با این تفاوت که در این تابع، PWM های مربوط به چشمک زدن LED ها را قطع میکنیم. تیکه کد زیر مربوط به قطع کردن PWM (چشمک زدن LED ها) می باشد:

```
void stopBlinking(void){
  HAL_TIM_PWM_Stop(&htim3, TIM_CHANNEL_1);
  HAL_TIM_PWM_Stop(&htim3, TIM_CHANNEL_2);
```

- **setDAC :**

در این تابع عددی که مربوط به معکوس شده سیگنال است را تولید می کنیم و به اسیلوسکوپ می دهیم.

در این تابع از تابع getExternalValue استفاده می کنیم تا مقدار دیجیتال سیگنال سینوسی ورودی را داشته باشیم. برای تناسب در این قسمت به جای انجام کار های پیچیده، کار را راحت تر میکنیم، مستقیم عددی(همان عدد بین 0 تا 4095) که تابع مذکور به ما می دهد را از 4095 کم میکنیم. چرا؟ در این سوال از ما خواسته شده است که معکوس سیگنال سینوسی ورودی را تولید کنیم. برای این کار هم هر وقت که سیگنال ورودی ما بالا بود، ما باید سیگنالی کهتولید می کنیم، پایین باشد و برعکس.
برای این کار هم می توانستیم به ولتاژ ورودی عدد را تبدیل کنیم، سپس 3.3 را از آن کم کنیم و حاصل را دوباره به صورت یک عدد بین صفر تا 4095 تبدیل کنیم(البته با توجه به اینکه ما از مبدیل دیجیتال به آنالوگ 16 بیتی استفاده می کنیم، در این حالت دیگر بعد از بدست آوردن ولتاژی که باید تولید شود، به جای اینکه آن را با نسبت و تناسب به یک عدد بین 0 تا 4095 تبدیل کنیم، آن را به یک عدد بین 0 تا 2^{16} - 1 تبدیل میکردیم). خب به جای این کار، صرفا عددی که میکروبه ما می دهد، 4095 را از آن کم میکنیم و سپس باز هم به جای یک تناسب با اعداد بسیار بزرگ (در اصل اگر می خواستیم عدد ۱۶ بیتی معادل آن را بسازیم، خیلی کسر بزرگی می شد، باید تقسیم بر 2^{14} - 1 میکردیم و سپس ضرب در 2^{16} - 1 میکردیم)، صرفا این منهای یک ها را لحاظ نکردیم و عدد را 4 بیت به سمت چپ شیفت دادیم(انگار ضربدر 16 کردیم(اگر یک ها را در کسر گفته شده لحاظ نکنیم، باید عدد را در 16 ضرب میکردیم.))
حال باید این عدد
تیکه کد زیر مربوط به این قسمت می باشد:

```
void setDAC(void){
  uint16_t signal = getExternalValue();
  signal = 4095 - signal;
  // the ADC_16 input is 16 bit, but our number is 12 bit. So we should convert our number to 16 bit number.
  // and for that, we simplified this conversion and only multiply it by 4.(or in other way, shift it 4 bit to left)
  signal <<= 4;
  GPIOC -> ODR = signal;
}
```

- **newTempratureHandler:**

این تابع برای آن است که هروقت دما عوض می شود، بیایم و بررسی کنیم که وضعیت کنونی با توجه به دمای جدید چه باید باشد و اقدامات مورد نیاز را انجام دهیم.(توابع مورد نیاز را برای حر حالت فراخوانی کنیم)

تیکه کد زیر برای این قسمت می باشد:

```
void newTempratureHandler(void){
  printString("new Temprature Handler");
  newLine();
  if(temprature < 35){
    printString("OK status");
    newLine();
    OK_state();
  }else if(temprature < 46){
    printString("WARNING state");
    newLine();
    WARNING_state();
  }else {
    printString("DANGER status");
    newLine();
    DANGER_state();
  }
}
```

- **OK_state:**

در این تابع، اقدامات مورد نیاز برای حالتی را انجام می دهیم که دمایی که اندازه گیری شده است، در اینجا در حالت OK باشد و با توجه به آن، LED های متناظر را روشن میکنیم. (چرا یک if قرار دادیم ؟ برای اینکه اگر این if را قرار نمیدادیم و هر سری که دمای محیط اندازه گیری می شد(اگر سری قبلی وجود داشت و وضعیتمان مانند این سری OK بود)، ما اگر میامدیم و تابع turnOffLED را صدا می کردیم، LED عه سبز یک بار خاموش و سپس روشن می شد، که باعث می شد این چراغ چشمک زن به نظر برسد. پس با قرار دادن یک if بررسی کردیم که اگر وضعیت قبلیمان هم OK بود، هیچ کاریب صورت نگیرد.)

تیکه کد مربوط به این تیکه به صورت زیر می باشد:

```
void OK_state(void){
  if(status != OK){
    turnOffLEDs();
    stopBlinking();
    HAL_GPIO_WritePin(LED_port, green, 1);
    status = OK;
  }
  //else => if we come to this part, it means our status was OK befor too. So we don't need to change it.
}
```

- **WARNING_state:**

مشابه با تابع بالاست، اما دیگر به آن شرط نیازی نداریم.

تیکه کد مربوط به این قسمت به صورت زیر می باشد:

```
void WARNING_state(void){
  turnOffLEDs();
  stopBlinking();
  HAL_GPIO_WritePin(LED_port, orange, 1);
  status = WARNING;
}
```

- **TURN_OFF_state:**

مشابه با دو تابع بالاست و دیگر به if نیازی ندارد و اینکه کار های مورد نیاز برای حالتی که سیستم باید خاموش شود را انجام می دهد.(مثلا اینکه باید به مدت نیم ثانیه LED قرمز به صورت چشمک زن روشن باشد و سپس همه چیخاموش شود. برای این کار، ابتدا همه چی را خاموش می کنیم، سپس PW< مربوط به LED قرمز را فعال میکنیم و سپس 500 میلی ثانیه با استفاده از HAL_Delay ، تاخیر ایجاد میکنیم. سپس همه چیز را خاموش میکنیم.) تیکه کد زیر مربوط به این قسمت می باشد:

```
void TURN_OFF_state(void){
  printString("Turn Off");
  newLine();
  stopBlinking();
  turnOffLEDs();
  HAL_TIM_PWM_Start(&htim3, TIM_CHANNEL_1);
  set7Segment(-1);
  HAL_Delay(500);
  stopBlinking();
  turnOffLEDs();
  status = TURN_OFF;
}
```

- **DANGER_state:**

این تابع نیز تا حد زیادی مشابه با توابع بالایی می باشد، اما این تابع کار های لازم برای وقتی که در حالت DANGER هستیم را انجام می دهد که کمی متفاوت از کار های عادی می باشد.

در این تابع، کار های زیر مرحله به مرحله انجام می شوند:

۱- ابتدا همه چیز متوقف می شود و LED قرمز رنگ به حالت چشمک زن فعال می شود.(PWM آن فعال می شود.)

۲- سپس 500 میلی ثانیه منتظر میمانیم که آیا کاربر کلید cooling را فشار می دهد یا خیر(این کار با استفاده از روش polling می باشد). اگر فشار نداد که به در قسمت else می رویم و کار های لازم را انجام میدهیم(کار های لازم برای وضعیت خاموش را انجام میدهیم.)، اگر فشار داده شد به سراغ بدنه if می رویم. (توضیح در مرحله بعد)

۳- وارد وضعیت cooling می شویم. در این حالت ابتدا باید ۲ ثانیه منتظر بمانیم تا دما اندازه گیری شود.(البته با توجه به اینکه هر بار اندازه گیری دما 200 میلی ثانیه زمان میبرد، پس گویا باید 10 بار دما را اندازه بگیریم.(توضیح حلقه for))

در بدنه این حلقه برای اندازه گیری دما، هر بار با استفاده از متغییر tempCounter (توضیح آن در بالا داده شده است) میایم و آن را صفر میکنیم و تا وقتی که خودش دوباره صفر نشده است، پیش می رویم.(اولین بار که آن صفر است همانجا انقدر می مانیم تا حداقل یک بار صدا زده شود و سپس پس از آن در یح حلقه دیگر آنقدر busy wait می کنیم تا دوباره این مقدار به صفر برسد. هنگامی که این اتفاق رخ داد، مطمئن میشویم که دما دقیقا یک بار اندازه گیری شده است.)

۴- حال با توجه به مقدار جدید دما وضعیتی که باید در آن باشیم را تعیین میکنیم.

تیکه کد مربوط به اسن قسمت به دلیل طولانی بودن، آورده نشده است.

- **HAL_GPIO_EXTI_Callback:**

این تابع برای هندل کردن interrupt های ناشی از کلید های reset و start می باشد. اگر هر کدام در وضعیتی که باید فشار داده شوند، فشرده شوند(برای start وقتی که هنوز کاری انجام نشده است و در وضعیت آغازین (BEFOR_START) هستیم و برای restart هم وقتی که سیستم خاموش شده است)، به حالت CHECKING می رویم و در خود main کار های لازم را انجام می دهیم.(چرا همانکار ها را اینجا انجام نمیدهیم با توجه به اینکه ساده تر بود؟ به دلیل وجود اولویت ها (priority) به باگ ها و مشکلات زیادی بر میخوردیم و هندل کردن آنها خیلی طولانی تر میشد.)
تیکه کد مربوط به این قسمت به صورت زیر می باشد:

```
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin){
  if(GPIO_Pin & SET1(0)){
    if(status == BEFOR_START){
      status = CHECKING;
      stopBlinking();
      turnOffLEDs();
      HAL_TIM_PWM_Start(&htim3, TIM_CHANNEL_2);
    }
  }else if(GPIO_Pin & SET1(2)){
    if(status == TURN_OFF){
      status = CHECKING;
      stopBlinking();
      turnOffLEDs();
      HAL_TIM_PWM_Start(&htim3, TIM_CHANNEL_2);
    }
  }
}
```

- **TIM2_IRQHandler:**

همانطور که در قسمت مشکلاتی که به آنها برخوردیم ذکر شد، این قسمت، برای پیاده سازی وقفه حاصل از تایمر 2 ،که برای اندازه گیری دما می باشد، است.
تیکه کد مربوط به این قسمت به صورت زیر میباشد:

```
void TIM2_IRQHandler(void){
  if(tempCounter == 0)
    temprature = 0;
  uint16_t a = getTemprature();
  temprature += a;
  tempCounter++;
  printString("Temprature Timer Intrrupt");
  newLine();
  printNumber(a);
  newLine();
  if(tempCounter >= 10){
    tempCounter = 0;
    temprature /= 10;
    set7Segment(temprature);
  }
}
```

- **تابع main:**

در ان تابع ابتدا برای اینکه چشمک زدن LED ها غیر فعال شود، ابتدا کانال PWM آنها را غیر فعال میکنیم. تیکه کد ایت قسمت به صورت زیر است:

```
HAL_TIM_PWM_Stop(&htim3, TIM_CHANNEL_1);
HAL_TIM_PWM_Stop(&htim3, TIM_CHANNEL_2);
```

دقیقا قبل از وارد شدن به حلقه، یک مقدار delay قرار می دهیم تا پروتئوس در این تابم load شود و مقدار دمایی که باید نشان بدهد به درستی ست شود. تیکه کد زیر برای این قسمت میباشد:

```
HAL_Delay(350); // wait for proteus to initialize (specialy for the thermometer to generate the appropriate voltage)
```

سپس در بدنه حلقه، حالت هایی که قرار بود در بدنه حلقه به آنها رسیدگی کنیم مانند کاری که باید در وضعیت CHECKING انجام دهیم و ... را بررسی میکنیم.

یکی از وقت هایی که باید کاری که انجام می دهیم را در اینجا هندل کنیم، وقتی است که در وضعیت CHECKING هستیم.(دلیل اینکه در اینجا می خواهیم این وضعیت را هندل کنیم، بالاتر توضیح داده شد.) در این وضعیت یک بار دما را اندازه گیری کنیم (در صورت سوال گفته شده است که باید 200 میلی ثانیه منتظر بمانیم و با توجه به اینکه زمانی که طول می شکد تا یکبار دما اندازه گیری شود، 200 میلی ثانیه است، پس گویا باید یکبار دما را انداره گیری کنیم.). حال برای اندازه گیری دما، ابتدا باید interrupt عهtimer مربوطه را فعال کنیم و سپس در انتهای اندذازه گیری دما (این روش استفاده شده در اینجا مشابه روشی است که بالاتر توضیح داده شد) این تابمر را غیر فعال میکنیم تا وضعیت روشن شود.(در صورت نیاز بعدا آن را در جای مناسب فعال میکنیم.) سپس تابع newTempratureHandler را فراخوانی میکنیم تا عملیات های مناسب را انجام دهد.(توضیح این تابع در بالاتر می باشد.)

تیکه کد مربوط به اسن قسمت به صورت زیر می باشد:

```
if(status == CHECKING){
  tempCounter = 0;
  temprature = 0;
  HAL_TIM_Base_Start_IT(&htim2);// activing tim2_irqHandler
  while(tempCounter == 0) // we wait until we know that we have started finding the temprature.
    ;
  while(tempCounter != 0) // wait until we find the temprature for the first time
    ;
  HAL_TIM_Base_Stop_IT(&htim2); // disabling tim2_irqHandler
  newTempratureHandler();
```

حال باید حالت هایی را هندل کنیم که در آنها در وضعیت OK یا WARNING هستیم.(ای دو حالت را چرا جدا کردیم و در جاهای دیگر هندل نکردیم؟ به دو دلیل. یک اینکه باز هم به خاطر بحث های مربوط به priority راهمان هم طولانی تار میشد هم باید دقتی بیشتری صرف میکردیم که از کجا به آنجا رفته ایم. دو اینکه برای بحث های مربوط به نمونه برداری از سیگنال سینوسی ورودی به مشلاتی میخوردیم که هندل کردن آنها به مراتب سخت تر بود.)

در اینجا باید در حین اندازه گیری دما، نمونه برداری از سیگنال سینوسی ورودی نیز صورت بگیر و کار های مربوط به آن (که در تابع setDAC می باشد) انجام شود. در اینجا نیز همان ساز و کار قبلی برای اندازه گیری دما استافده شده است، اما باید در اینجا این مطلب را که می خواهیم سیگنال سینوسی را هم نمونه برداری کنیم را هم لحاظ کنیم و اگر لحاظ نمیکردیم ممکن بود به مشکلاتی بربخوریم.
در حالتی که می خواهیم تابع setDAC را فراخوانی کنیم، بای د وقفه تایمر را از کار بیاندازیم. اما چرا؟
چون اگر وقفه تایمری فعال باشد و در وسط تابع setDAC تایمرمان یک وقفه ایجاد کند، ممکن است باعث شود که config ها و تنظیماتی که برای کانال های خواندن پین آنالوگ داریم با یکدیگر قاطی شوند و مقدار اشتباهی در هر کدام یک از آن ها برود. برای همین منظور هر وقت که می خواهیم تابع setDAC را فراخوانی کنیم، ابتدا وقفه تایمر آن را قبلش غیر فعال میکنیم و سپس دوباره پس از اتمام اجرای تابع setDAC، وقفه تایمری را دوباره فعال میکنیم. در این صورت به مشکلی بر نمیخوریم.

تیکه کد مربوط به اسن قسمت به صورت زیر می باشد:

```
else if(status == OK || status == WARNING){
 tempCounter = 0;
 temprature = 0;
 HAL_TIM_Base_Start_IT(&htim2);// activing tim2_irqHandler
 while(tempCounter == 0){ // we wait until we know that we have started finding the temprature.
   HAL_TIM_Base_Stop_IT(&htim2);
   setDAC();
   HAL_TIM_Base_Start_IT(&htim2);
 }
 while(tempCounter != 0){ // wait until we find the temprature for the first time
   HAL_TIM_Base_Stop_IT(&htim2);
   setDAC();
   HAL_TIM_Base_Start_IT(&htim2);
 }
 HAL_TIM_Base_Stop_IT(&htim2); // disabling tim2_irqHandler
 newTempratureHandler();
```

منابع:

- https://www.youtube.com/watch?v=ABWU7FlM1T0
- https://www.youtube.com/watch?v=2T7ThV6ng4E
- https://www.electro-tech-online.com/threads/using-bus-in-proteus.143102/
- https://www.circuitstoday.com/digital-integrated-circuits-proteus
- https://www.researchgate.net/post/How-to-draw-no-connection-between-two-crossing-wires-using-Proteus-7-ISIS-7-professional
- https://www.edaboard.com/threads/bus-wire-using-proteus.195134/
- https://www.slideshare.net/DedarulHasan/tutorials-proteus-schematic
- https://componentsearchengine.com/library/proteus?gclid=EAIaIQobChMI-uOe87Ku9wIVyI1oCR2f-AeGEAMYASAAEgKYC_D_BwE
- https://www.labcenter.com/
- https://electronics.stackexchange.com/questions/86195/how-to-connect-two-buses-with-an-offset-in-proteus
- https://electel.blogspot.com/2016/01/how-to-use-bus-in-proteus.html
- http://8051programming.blogspot.com/2014/01/how-to-use-bus-in-proteus.html
- https://www.edaboard.com/threads/how-to-use-bus-wire-in-proteus.206863/
- https://dspnor.com/products/destreamer-lan-to-analog-legacy-format/?utm_source=google&utm_medium=cpc&utm_campaign=destreamer&gclid=EAIaIQobChMIsIrC6K-u9wIVGc53Ch3ZCgZREAAYASAAEgKrAPD_BwE
- https://www.arduino.cc/reference/en/language/functions/analog-io/analogwrite/
- https://www.ti.com/lit/ds/symlink/dac7642.pdf?ts=1650859569351&ref_url=https%253A%252F%252Fwww.google.com%252F
- https://www.ti.com/lit/ds/symlink/dac8811.pdf?ts=1650859330646&ref_url=https%253A%252F%252Fwww.google.com%252F
- https://www.sparkfun.com/datasheets/Components/SMD/ATMega328.pdf
- https://www.labcenter.com/peripherals/
- https://atmega32-avr.com/proteus-simulation-based-avr-projects/
- https://microcontrollerslab.com/pt8211-dual-channel-16-bit-dac-pinout-working-features-applications/
- https://www.researchgate.net/figure/Simulation-scheme-of-an-8-bit-DAC-with-a-R-2R-matrix-created-in-the-Proteus-environment_fig1_343126363
- https://www.pinterest.com/pin/316096467599587584/
- https://www.aparat.com/result/Proteus
- https://dl.acm.org/doi/10.5555/3199700.3199755
- https://www.studocu.com/row/document/comsats-university-islamabad/electric-machines/lab-1-oscilloscope-and-function-generator-in-proteus/8990924
- https://www.edaboard.com/threads/signal-generator-in-proteus.368228/
- https://www.eca.ir/forums/thread700.html

- https://www.snapeda.com/parts/DAC5578SPWR/Texas%20Instruments/view-part/
- https://www.ele.uva.es/~jesman/BigSeti/ftp/Cajon_Desastre/Software-Manuales/EBook%20-%20Proteus%20Library.pdf
- https://deepbluembedded.com/convert-pwm-to-a-dac-using-pwm-to-generate-analog-waveforms/
- https://www.programiz.com/c-programming/c-enumeration
- https://forum.arduino.cc/t/how-to-seperate-numbers-on-a-single-2-digit-7-segment-led-display/358580
- https://electronics.stackexchange.com/questions/230195/display-double-digit-numbers-on-2-seven-segment-displays
- https://www.microchip.com/forums/m1079262.aspx
- https://www.microchip.com/forums/m1079261.aspx
- https://maxembedded.com/2013/01/seven-segment-multiplexing/
- https://en.wikipedia.org/wiki/Oscilloscope
- https://www.dideo.ir/v/ap/3R98q/%DA%86%D8%B7%D9%88%D8%B1%DB%8C-%D8%A7%D8%B2-%D8%A7%D8%B3%DB%8C%D9%84%D9%88%D8%B3%DA%A9%D9%88%D9%BE-%D8%AF%D8%B1-%D9%BE%D8%B1%D9%88%D8%AA%D8%A6%D9%88%D8%B3-%D8%A7%D8%B3%D8%AA%D9%81%D8%A7%D8%AF%D9%87-%DA%A9%D9%86%DB%8C%D9%85
- http://www.barghebeheshti.blogfa.com/post/802
- https://community.st.com/s/question/0D50X0000AurR9hSQE/i-am-using-the-adc-on-nucleo-stm32f401re-board-adc-is-programmed-using-interrupts-and-the-code-does-not-work-beyond-10khz-of-input-signal-what-am-i-missing
- https://stackoverflow.com/questions/56454651/adc-on-nucleo-stm32f401re-board-does-not-work-beyond-10khz-of-input-signal
- https://forum.allaboutcircuits.com/threads/arduino-uno-simulation-using-proteus-analog-sampling-1hz-sine-signal.141406/
- https://devzone.nordicsemi.com/f/nordic-q-a/67343/not-able-to-read-analog-value-from-a-proteus-iii-board-nrf52840-soc
- https://www.edaboard.com/threads/problem-reading-analog-signal-in-proteus-using-arduino.361965/
- https://www.edaboard.com/threads/reading-an-analog-current-signal-into-a-pic-micro.299494/
- https://componentsearchengine.com/learn-more?gclid=EAIaIQobChMI59fqnuys9wIVj9rVCh1D5QnAEAAYASABEgLZHfD_BwE
- https://componentsearchengine.com/library/proteus?gclid=EAIaIQobChMI59fqnuys9wIVj9rVCh1D5QnAEAAYASAAEgIUsvD_BwE
- https://agetintopc.com/proteus-8-free-download/
- https://softvela.com/proteus-8-download/

- https://www.labcenter.com/downloads/
- https://www.st.com/resource/en/user_manual/um1724-stm32-nucleo64-boards-mb1136-stmicroelectronics.pdf
- https://files.amperka.ru/datasheets/nucleo-usermanual.pdf
- https://rusefi.com/forum/viewtopic.php?f=5&p=45048
- https://www.labcenter.com/wiring/
- https://www.youtube.com/watch?v=GYONjVWahGw
- https://www.youtube.com/results?search_query=stm32f401re+nucleo+ADC+in+proteus
- https://www.youtube.com/results?search_query=stm32f401re+nucleo+ADC
- https://www.youtube.com/
- https://www.youtube.com/watch?v=VNs9jnrGuTU
- https://www.youtube.com/?gl=US&tab=r1
- https://www.electro-tech-online.com/threads/help-with-adc0804-in-proteus.123594/
- https://arduino.stackexchange.com/questions/84526/potentiometer-only-display-analog-voltage-value-of-1023-in-proteus
- https://www.edaboard.com/threads/problem-with-proteus-software.227248/
- https://www.microchip.com/forums/m901253.aspx
- https://forum.arduino.cc/t/analog-inputs-not-working/350004
- https://forum.arduino.cc/t/analog-inputs-not-working/350004/11
- https://forum.arduino.cc/t/analog-inputs-not-working/350004/13
- https://forum.arduino.cc/t/analog-inputs-not-working/350004/10
- https://forum.arduino.cc/t/analog-inputs-not-working/350004/9
- https://forum.arduino.cc/t/analog-inputs-not-working/350004/8
- https://forum.arduino.cc/t/analog-inputs-not-working/350004/6
- https://forum.arduino.cc/t/analog-inputs-not-working/350004/4
- https://www.youtube.com/watch?v=9ezj3bdq2g8
- https://www.youtube.com/watch?v=ls9Ag9QX1vo
- https://www.st.com/resource/en/application_note/cd00211314-how-to-get-the-best-adc-accuracy-in-stm32-microcontrollers-stmicroelectronics.pdf
- https://www.microchip.com/forums/m1202313.aspx
- https://www.theengineeringprojects.com/2015/12/arduino-library-proteus-simulation.html
- https://www.ele.uva.es/~jesman/BigSeti/ftp/Cajon_Desastre/Software-Manuales/EBook%20-%20Proteus%20Manual.pdf
- https://www.arduino.cc/en/Tutorial/BuiltInExamples/ReadAnalogVoltage
- https://www.arduino.cc/en/Tutorial/BuiltInExamples/ReadAnalogVoltage/
- https://www.edaboard.com/threads/replace-models-in-proteus-help.249166/
- https://www.edaboard.com/threads/h-bridge-cicuits-in-proteus.233594/

- https://www.dideo.ir/v/yt/e1oJPHstBbw/stm32-nucleo-keil-5-ide-with-cubemx%3A-tutorial-4
- https://www.dideo.ir/v/yt/kI_Xngn3G8c/stm32-nucleo-tutorial-3-adc-coding-in-keil
- https://visualgdb.com/tutorials/arm/stm32/adc/
- https://electronics.stackexchange.com/questions/235832/nucleo-f401re-adc-sampling-time
- https://www.st.com/en/evaluation-tools/nucleo-f401re.html
- https://www.st.com/en/microcontrollers-microprocessors/stm32f401re.html
- http://www.disca.upv.es/aperles/arm_cortex_m3/llibre/st/STM32F439xx_User_Manual/stm32f4xx_hal_adc_8c_source.html
- http://www.disca.upv.es/aperles/arm_cortex_m3/llibre/st/STM32F439xx_User_Manual/group__adc__exported__functions__group2.html#ga421008ca3885339acb12f400958ffbe4
- http://www.disca.upv.es/aperles/arm_cortex_m3/llibre/st/STM32F439xx_User_Manual/group__adc__exported__functions__group2.html
- https://www.ti.com/lit/ds/symlink/ads5242.pdf?ts=1650745609717&ref_url=https%253A%252F%252Fwww.google.com%252F
- https://www.analog.com/media/en/technical-documentation/data-sheets/AD7829-1.pdf
- https://www.delftstack.com/howto/c/how-to-convert-an-integer-to-a-string-in-c/
- https://www.computerhope.com/jargon/n/newline.htm#:~:text=In%20programming%20languages%2C%20such%20as,which%20is%20an%20escape%20sequence.
- https://flaviocopes.com/c-string-length/#:~:text=Use%20the%20strlen()%20function,string%20as%20an%20integer%20value.
- https://www.edaboard.com/threads/proteus-virtual-terminal.12101/
- https://www.theengineeringprojects.com/2013/05/how-to-use-virtual-terminal-in-proteus.html
- https://cpp.hotexamples.com/examples/-/-/HAL_UART_Transmit/cpp-hal_uart_transmit-function-examples.html
- https://www.electro-tech-online.com/threads/proteus-variable-power-supply.110648/
- https://os.mbed.com/questions/54029/USART-Problem-with-NucleoF401RE/
- https://wiki.st.com/stm32mcu/wiki/STM32StepByStep:Step3_Introduction_to_the_UART
- https://www.st.com/resource/en/application_note/an4908-stm32-usart-automatic-baud-rate-detection-stmicroelectronics.pdf
- https://www.st.com/resource/en/application_note/an4908-stm32-usart-automatic-baud-rate-detection-stmicroelectronics.pdf
- https://www.carminenoviello.com/2015/03/02/how-to-use-stm32-nucleo-serial-port/
- https://www.st.com/en/development-tools/stsw-link007.html

- https://os.mbed.com/questions/3017/STM32F401RE-Nucleo-Virtual-COM-Port-is-n/
- https://trust.salesforce.com/en/blocked/
- https://next-hack.com/index.php/2020/02/15/how-to-interface-a-3-3v-output-to-a-5v-input/
- https://electronics.stackexchange.com/questions/191336/how-to-unhide-hidden-pins-in-proteus
- https://electronics.stackexchange.com/questions/374671/vcc-and-gnd-in-microcontroller-on-proteus
- https://www.edaboard.com/threads/changing-the-power-voltage-of-the-chips-in-proteus.222677/
- https://www.dideo.ir/v/yt/5XpzEhHYzQo/proteus-tutorial-%231%3A-how-to-add-ground%2C-5v%2C-3.3v
- https://stackoverflow.com/questions/56454651/adc-on-nucleo-stm32f401re-board-does-not-work-beyond-10khz-of-input-signal
- https://community.st.com/s/question/0D50X00009XkYTjSAN/why-my-adc-value-alternates-when-signal-is-steady
- https://www.st.com/resource/en/datasheet/stm32h747xi.pdf
- https://forum.arduino.cc/t/fast-hardware-adc-any-samples-for-this-are-there-any-plans-to-make-some-examples-for-this/979310/3
- https://forum.arduino.cc/t/fast-hardware-adc-any-samples-for-this-are-there-any-plans-to-make-some-examples-for-this/979310/4
- https://forum.arduino.cc/t/fast-hardware-adc-any-samples-for-this-are-there-any-plans-to-make-some-examples-for-this/979310/2
- https://forum.arduino.cc/t/fast-hardware-adc-any-samples-for-this-are-there-any-plans-to-make-some-examples-for-this/979310
- https://stackoverflow.com/questions/65402721/cannot-get-adc-value-and-put-it-into-a-array-stm32f103c8t6
- https://vimsky.com/examples/detail/cpp-ex-----HAL_ADC_ConfigChannel-function.html
- http://subzero6969.site/enter/?mark=20220414--kabinetalmari.com/4traru&tpl=3&engkey=stm32+hal+adc+multi+channel+example
- https://github.com/openenergymonitor/STM32/blob/master/images/STMCubeADC1.png
- https://forum.arduino.cc/t/portenta-h7-adc-dma-first-steps/931669
- https://github.com/openenergymonitor/STM32/blob/master/docs/Analog.md
- https://letanphuc.net/2016/07/stm32f0-adc/
- https://www.eevblog.com/forum/microcontrollers/stm32-problem-with-adc-conversion/
- https://electronics.stackexchange.com/questions/202938/stm32-adc-conversion-using-hal
- https://i.stack.imgur.com/atCJU.png

- https://cpp.hotexamples.com/examples/-/-/HAL_ADC_GetValue/cpp-hal_adc_getvalue-function-examples.html
- https://www.ece.lsu.edu/koppel/proteus/proteusl_8.html
- https://www.avrfreaks.net/forum/proteus-8-not-showing-eeprom-initialized-variables-using-eseg
- https://www.keil.com/support/man/docs/uv4/uv4_armcompilers.htm
- https://www.keil.com/support/man/docs/uv4/uv4_dg_adsld.htm
- https://www.keil.com/update/check.asp?P=MDK&V=5.36.0.0&S=
- https://www.keil.com/support/man/docs/ARMLINK/armlink_pge1362075624402.htm
- https://support.labcenter.com/forums/index.php
- https://stackoverflow.com/questions/58419485/how-to-read-and-show-adc-value-of-stm32f4-using-hal-library
- http://www.disca.upv.es/aperles/arm_cortex_m3/llibre/st/STM32F439xx_User_Manual/group__adc__exported__functions__group2.html
- https://microcontrollerslab.com/adc-stm32f4-discovery-board-with-hal-adc-driver/
- http://www.disca.upv.es/aperles/arm_cortex_m3/llibre/st/STM32L486xx_User_Manual/group__adcex__exported__functions__group1.html
- https://deepbluembedded.com/stm32-adc-read-example-dma-interrupt-polling/
- https://deepbluembedded.com/stm32-adc-tutorial-complete-guide-with-examples/
- https://microcontrollerslab.com/adc-stm32f4-discovery-board-with-hal-adc-driver/
- https://computer.howstuffworks.com/c7.htm
- https://www.electronics-tutorials.ws/combination/analogue-to-digital-converter.html
- https://learn.sparkfun.com/tutorials/analog-to-digital-conversion/all
- https://www.google.com/search?q=how+to+use+device+buses+in+proteus&rlz=1C1GGRV_enIR889IR889&ei=1kRnYtfiL4G8kwW__6a4DA&oq=how+to+use+device+buse&gs_lcp=Cgdnd3Mtd2l6EAMYADIFCCEQoAEyBQghEKABMgUIIRCgATIECCEQFToECAAQQzoOCC4QgAQQxwEQowIQ1AI6BQgAEIAEOgsILhCABBDHARCvATolCC4QgAQQxwEQowI6BQguEIAEOggILhCABBDUAjoGCAAQFhAeOggIABAWEAoQHjoICCEQFhAdEB5KBAhBGABKBAhGGABQAFifJWCONGgAcAF4AIABqQKIAfgpkgEEMi0yMpgBAKABAcABAQ&sclient=gws-wiz#kpvalbx=_GkZnYpzwBK_xsAfcl5SQBA17
- https://www.youtube.com/watch?v=pUgvPpdJV_g
- https://www.ti.com/lit/ds/symlink/dac7642.pdf?ts=1650859569351&ref_url=https%253A%252F%252Fwww.google.com%252F
- https://www.st.com/resource/en/application_note/an4908-stm32-usart-automatic-baud-rate-detection-stmicroelectronics.pdf
- https://www.st.com/resource/en/datasheet/stm32h747xi.pdf
- https://www.elforum.info/topic/140317-circuite-si-aplicatii-cu-adc-mcp3551-ltc2440-si-dac-mcp4921-ltc1655-cu-pic-mikroc-proteus/
- https://m.z-mass.com/forums/forums/topic/how-to-use-bus-in-proteus/
- https://www.youtube.com/watch?v=pUgvPpdJV_g

- https://www.youspice.com/getting-started-with-proteus/2/
- http://www.picbasic.co.uk/forum/showthread.php?t=16869
- https://componentsearchengine.com/learn-more?gclid=EAIaIQobChMI54vH8Mmw9wIVitxRCh2vUgOwEAAYASABEgKAHfD_BwE
- https://www.youtube.com/watch?v=Hrgw39QaJ3s
- https://www.tehnium-azi.ro/forums/topic/7054-circuite-si-aplicatii-cu-adc-mcp3551-ltc2440-si-dac-mcp4921-ltc1655-cu-pic-mikroc-proteus/
- https://www.ijntr.org/download_data/IJNTR04010006.pdf
- https://www.taskit.de/en/products/embedded-products/converter/?p=1
- https://www.youtube.com/watch?v=sBetu5sTNzw
- https://dspnor.com/products/destreamer-lan-to-analog-legacy-format/?utm_source=google&utm_medium=cpc&utm_campaign=destreamer&gclid=EAIaIQobChMIq8H5qdOw9wIVso1oCR3yDwn_EAAYAiAAEgItNPD_BwE
- http://site.iugaza.edu.ps/olatif/files/2010/02/Lab-10_Digital-To-Analog-Converter.pdf
- http://site.iugaza.edu.ps/olatif/files/2010/02/Lab-9_Analog-To-Digital-Converter.pdf
- https://microcontrollerslab.com/dac0808-8-bit-digital-to-analog-converter/
- https://microcontrollerslab.com/dac7715-dac-pinout-features-example-circuit-applications-features/
- https://www.shf-communication.com/products/high-speed-modules/?gclid=EAIaIQobChMIq8H5qdOw9wIVso1oCR3yDwn_EAMYASAAEgKXNfD_BwE
- https://www.youspice.com/spiceprojects/spice-simulation-projects/general-electronics-spice-simulation-projects/analog-to-digital-converters-spice-simulation-projects/analog-to-digital-converter-with-microcontroller-8051/
- https://www.researchgate.net/figure/LM35-interfaced-to-ADC-and-microcontroller-as-designed-using-Proteus-Professional-7_fig3_311328027
- https://www.ti.com.cn/cn/lit/ds/symlink/dac0808.pdf?ts=1650941583894&ref_url=https%253A%252F%252Fwww.google.com%252F
- https://microcontrollerslab.com/pt8211-dual-channel-16-bit-dac-pinout-working-features-applications/
- https://microcontrollerslab.com/dac-introduction-types/
- https://microcontrollerslab.com/analog-to-digital-adc-converter-working/
- https://www.microchip.com/forums/m711227.aspx?tree=true
- https://www.youtube.com/watch?v=p8OBzPUQ1FA
- https://www.taborelec.com/Sample-Resolution-vs-DAC-Resolution
- https://saeedsolutions.blogspot.com/2013/07/8051-dac-using-dac0808-code-proteus.html?view=sidebar
- https://www.edaboard.com/threads/testing-of-dac-outputs.360285/
- https://www.microchip.com/forums/m646755.aspx

- https://www.microchip.com/forums/download.axd?file=2;646755
- https://www.microchip.com/forums/download.axd?file=0;648347
- https://www.machineseeker.com/mss/proteus?gclid=EAIaIQobChMIxvjjneCw9wIVkKl3Ch2J3gcXEAMYASAAEgJCG_D_BwE
- https://electronics.stackexchange.com/questions/194182/reopen-oscilloscope-window-in-proteus-8#:~:text=In%20order%20to%20show%20again,the%20window%20will%20be%20visible.
- www.edaboard.com
- https://www.howtodowith.com/electronics-engineering/proteus-oscilloscope-not-showing/
- https://stackoverflow.com/questions/69310954/hal-gettick-always-returns-0
- اسلاید های استاد
- رفرنس منوآل
- دیتا شیت
- یوزر منوآل