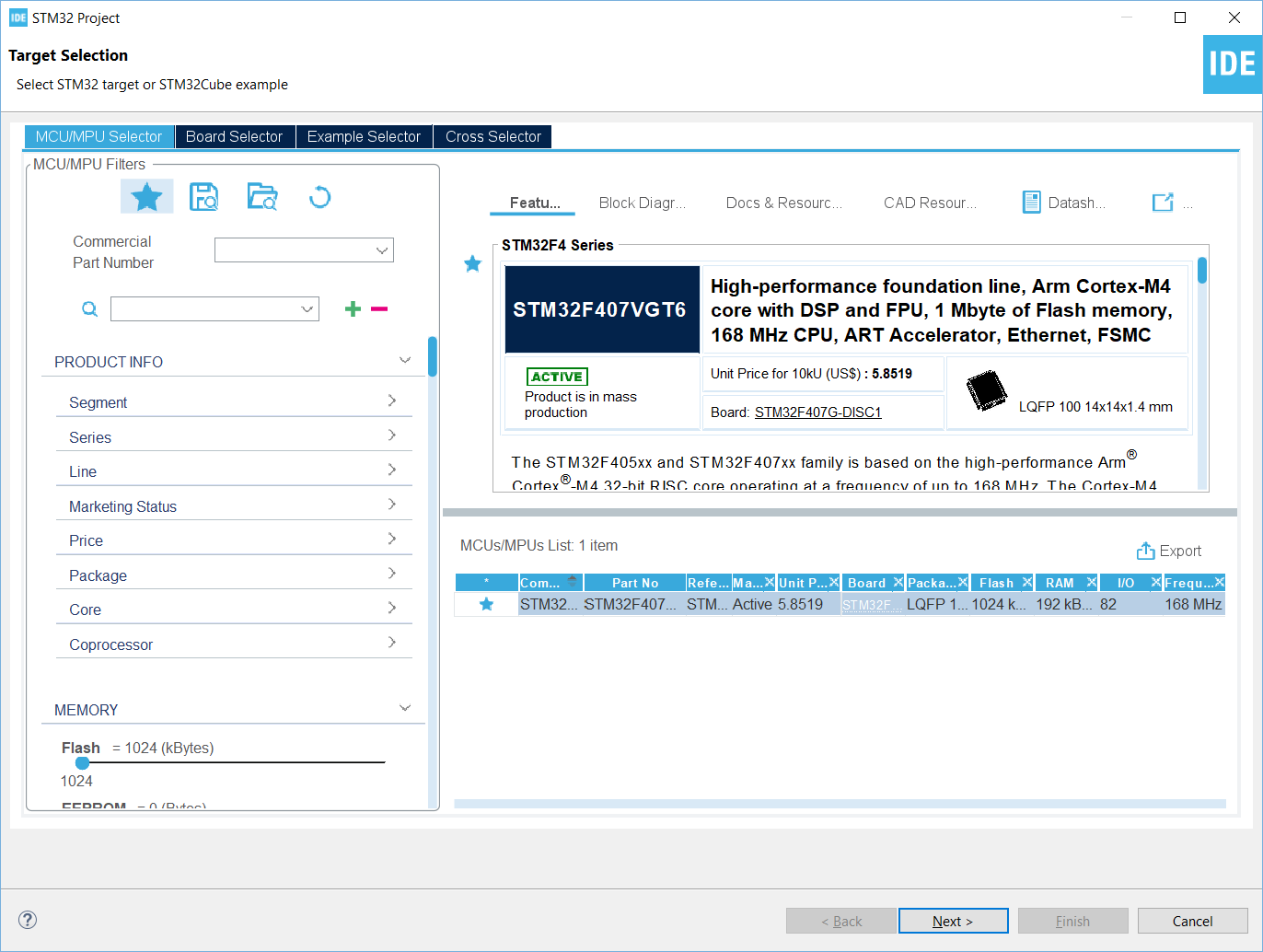
**PROJECT STEPS**

1. Create a project and wait the software to update MCUs.

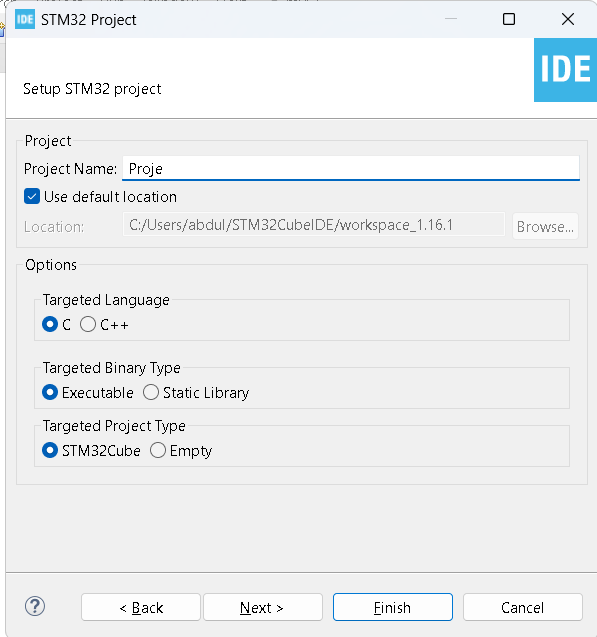
metin, ekran görüntüsü, yazılım, bilgisayar simgesi içeren bir resim

Açıklama otomatik olarak oluşturuldu

1. Select STM32F407V and click next, do not select the board, we want pins to be unconfigured:



1. Give a name to the project, do not change default settings just click finish.



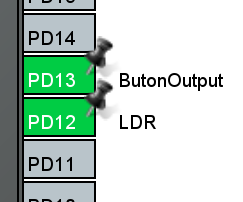
1. From Pinout and Configurations do the followings:
   1. Analog>ADC1: Select IN1 in order to enable it.
   2. Instead of using PA1. We are going to use PA1 pin as analog pin, therefore you should configure PA1 as ADC1\_IN1.

metin, ekran görüntüsü, yazılım, bilgisayar simgesi içeren bir resim

Açıklama otomatik olarak oluşturuldu metin, ekran görüntüsü, yazı tipi, grafik içeren bir resim

Açıklama otomatik olarak oluşturuldu

* 1. Configure the LED’s PD12,PD13 pins as genaral output and PD1 pin as general input. Name them LDR , ButonOutput and Butoninput respectively.

 metin, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

* 1. Now ready to generate code.

metin, yazı tipi, yazılım, web sayfası içeren bir resim

Açıklama otomatik olarak oluşturuldu



* 1. Now press to build button in order to check everything is configured as described.

metin, yazı tipi, ekran görüntüsü, logo içeren bir resim

Açıklama otomatik olarak oluşturuldu



1. Connect the board and start debugging.

metin, ekran görüntüsü, yazı tipi, logo içeren bir resim

Açıklama otomatik olarak oluşturuldu



1. Configure program function.Then upload the code to the device.





Add the following listing to your main.c file and command in the main function. Make sure you put them in the user code blocks.

First, you should include the stdio.h to your main.c file.

/\* Private includes ----------------------------------------------------------\*/

/\* USER CODE BEGIN Includes \*/

**#include** <stdio.h>

/\* USER CODE END Includes \*/

Second, you should include this command for used to monitor the status of LED.

/\* USER CODE BEGIN 2 \*/

uint8\_t led\_action\_done = 0; // Used to monitor the status of LED.

/\* USER CODE END 2 \*/

/\* Infinite loop \*/

/\* USER CODE BEGIN WHILE \*/

// Variables

// Variables

uint32\_t lastButtonTime = 0;

uint32\_t lastLDRTime = 0;

uint32\_t lastADCTime = 0;

uint8\_t buttonPressed = 0;

uint8\_t ldrTriggered = 0;

uint8\_t adcBusy = 0;

uint8\_t ldrOutputActive = 0; // New flag to check if the LDR output is active

Now, we add the variables into the infinite loop.

Now, we insert the main code into the infinite loop

**while** (1)  
{  
 uint32\_t currentTime = HAL\_GetTick();  
  
 // Button control  
 **if** (HAL\_GPIO\_ReadPin(Butoninput\_GPIO\_Port, Butoninput\_Pin) == *GPIO\_PIN\_SET* && !buttonPressed)  
 {  
 buttonPressed = 1;  
 lastButtonTime = currentTime;  
 HAL\_GPIO\_WritePin(ButonOutput\_GPIO\_Port, ButonOutput\_Pin, *GPIO\_PIN\_SET*);  
 }  
  
 // Button LED control  
 **if** (buttonPressed && (currentTime - lastButtonTime >= 250))  
 {  
 HAL\_GPIO\_WritePin(ButonOutput\_GPIO\_Port, ButonOutput\_Pin, *GPIO\_PIN\_RESET*);  
 buttonPressed = 0;  
 }  
  
 // ADC read operation (every 50ms)  
 **if** (!adcBusy && (currentTime - lastADCTime >= 50))  
 {  
 lastADCTime = currentTime;  
 HAL\_ADC\_Start(&hadc1);  
 adcBusy = 1;  
 }

// ADC conversion control  
 **if** (adcBusy && HAL\_ADC\_PollForConversion(&hadc1, 0) == *HAL\_OK*)  
 {  
 uint16\_t adcValue = HAL\_ADC\_GetValue(&hadc1);  
 HAL\_ADC\_Stop(&hadc1);  
 adcBusy = 0;

// LDR control  
 **if** (adcValue > 2500)  
 {  
 **if** (!ldrTriggered && !ldrOutputActive)  
 {  
 ldrTriggered = 1;  
 ldrOutputActive = 1;  
 lastLDRTime = currentTime;  
 HAL\_GPIO\_WritePin(LDR\_GPIO\_Port, LDR\_Pin, *GPIO\_PIN\_SET*);  
 }  
 }  
 **else** {  
 ldrTriggered = 0; // Reset trigger when LDR value drops below threshold  
 }  
 }

// LDR LED control  
 **if** (ldrOutputActive && (currentTime - lastLDRTime >= 1500))  
 {  
 HAL\_GPIO\_WritePin(LDR\_GPIO\_Port, LDR\_Pin, *GPIO\_PIN\_RESET*);  
 ldrOutputActive = 0; // Deactivate output  
 }  
 }  
}

/\* USER CODE END 3 \*/

Now you can use the project.