**SD CARD BOOTLOADER**

Bootloader, mikrodenetleyiciler veya gömülü sistemlerde kullanılan küçük bir programdır. Sistemin açılışında çalışır ve ana uygulamayı başlatmadan önce bazı kritik görevleri yerine getirir. Aslında adından da anlaşılacağı gibi bir ön yükleyicidir. Biz de kodumuzu güncellemek için ön yükleyici ile bazı işlemler yapacağız. Aslında yapılacak ana işimiz , kodumuzun içinden SD card, UART veya Uzaktan bağlantı ile (HTTPS,FTP) güncellemek istediğimiz kodun .bin dosyasını çekeceğiz. Sonrasında çekmiş olduğumuz bu dosyayı belli işlemlerden sonra Flash hafızaya yazacağız. Unutmamız gereken en önemli husus güncelleyeceğimiz kodun başlangıç adresi (Memory den ve Startup dosyasında ayarlayacağız) ile bin dosyasını yazacağımız flash hafıza aynı adres olmalıdır. Aksi takdirde kod istediğimiz gibi çalışmaz .Şimdi adım adım işlemlerimize geçelim.

Şimdi Öncelik olarak yapmamız gereken Ana Kodumuzun Başlangıç adresini ve Vector Table’ ını değiştirmek.

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

Açıklama otomatik olarak oluşturuldu

STM32F415RGTX\_FLASH.ld dosyasının içinde FLASH (rx) tarafını değiştiriyoruz . Normalde 0x8000000 olan adresi , Kaydetmek istediğimiz flash hafıza alanını giriyoruz.

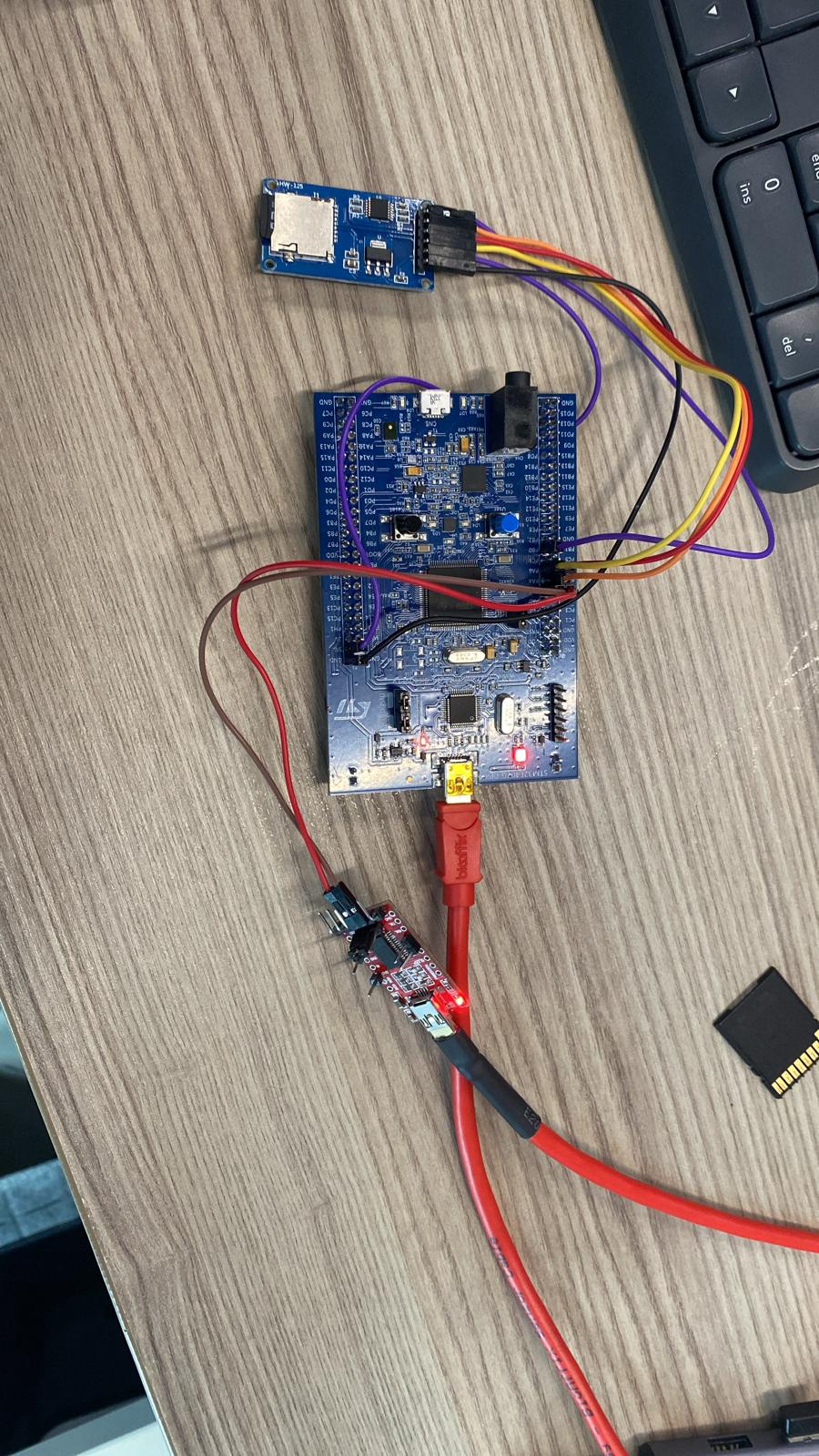
metin, yazı tipi, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Sonrasında startup\_stm32f415rgtx.s dosyasının içinde SystemInit değişkeninin içine girip VECT\_TAB\_OFFSET adresini değiştiriyoruz. Normalde 0x0000000 olan adresi 0x0008000 olarak değiştiriyoruz. Sonrasında kodumuzu yazıp Run ediyoruz.

Run ettiğimiz bu kodun proje dosyasına girip içinden .bin dosyasını almamız gerekiyor.

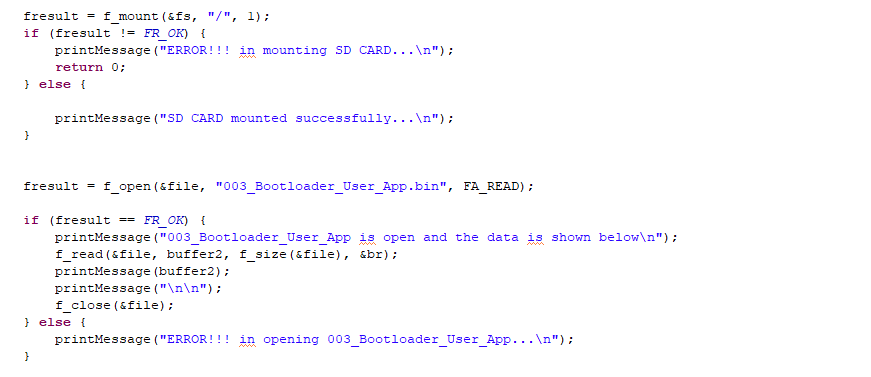
.bin dosyasını kodumuzu yükleyeceğimiz bir SD Kartın içine yüklüyoruz. Bunun için devremizde bir SD Kart olması gerekiyor. Kurduğumuz devre şeması şu şekilde;



Kullanmış olduğumuz SD Kartı SPI ile haberleşmektedir. Kodun takibini kolaylaştırmak için de bir adet USB-TTL bağlantısı da bulunmaktadır. SD Kartın STM32 ile kullanımı , dosya işlemleri için aşağıda linkler verilmiştir. Takip edilerek SD Kart kodu eklenebilir.

<https://controllerstech.com/sd-card-using-spi-in-stm32/>

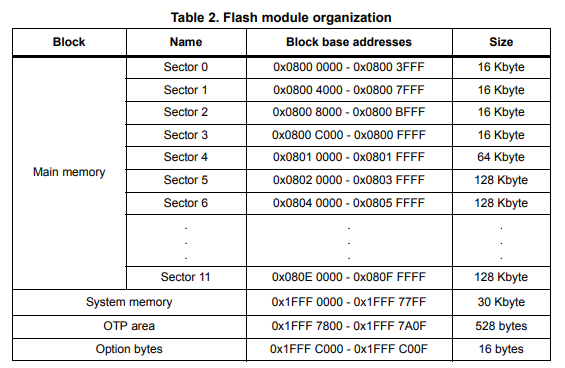
<https://blog.naver.com/eziya76/221188701172>



SD Kartla işlemcinin SPI bağlantılarını yaptıktan ve kütüphaneyi doğru bir şekilde implement ettikten sonra yukarıda ki kodları yazın. PrintMessage ile TTL bağlantısı ile kontroller sağlanabilir.

.bin dosyası çekitikten sonra flaş hafızaya yazma ve atlama fonksiyonlarını yazma sırası geldi. STM32 Flash\_Write fonksiyonunu yazıyoruz. Flash hafıza işlemcinin silinmeyen hafızasıdır. Yani güç kesilse bile hafıza bloğundan kod içinde erease edilmediği sürece veriler kaybolmaz.

Default olarak STM işlemciler 0x8000000 adresinden başlar.



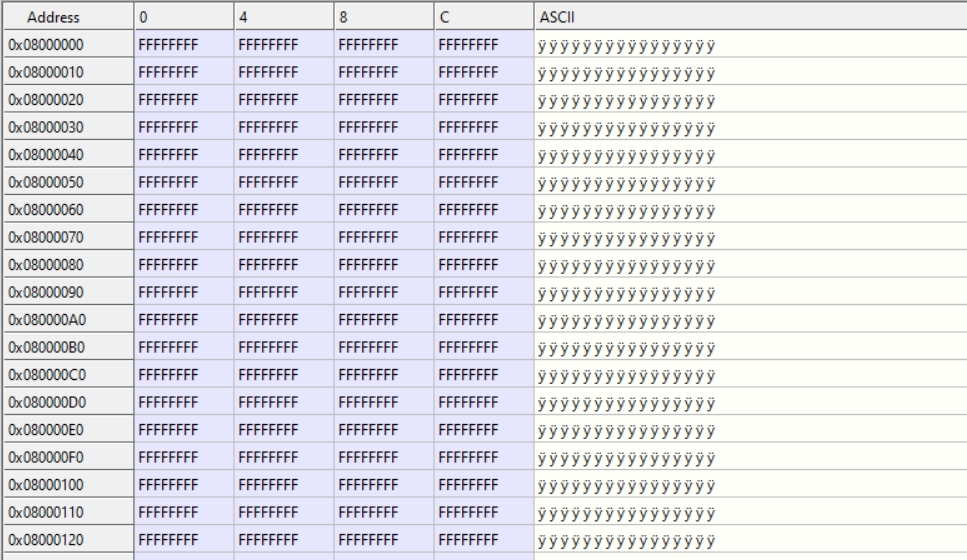
Biz güncelleme işlemi yaparken Bootloader kodu da bu adresten başlar. Yazmış olduğumuz kodlar 0x8000000 hafızasından çalışmaya başlar. .bin dosyasını çektikten sonra flash hafızada ki 0x800 8000 adresine yazarız.

metin, ekran görüntüsü, yazı tipi, doküman, belge içeren bir resim

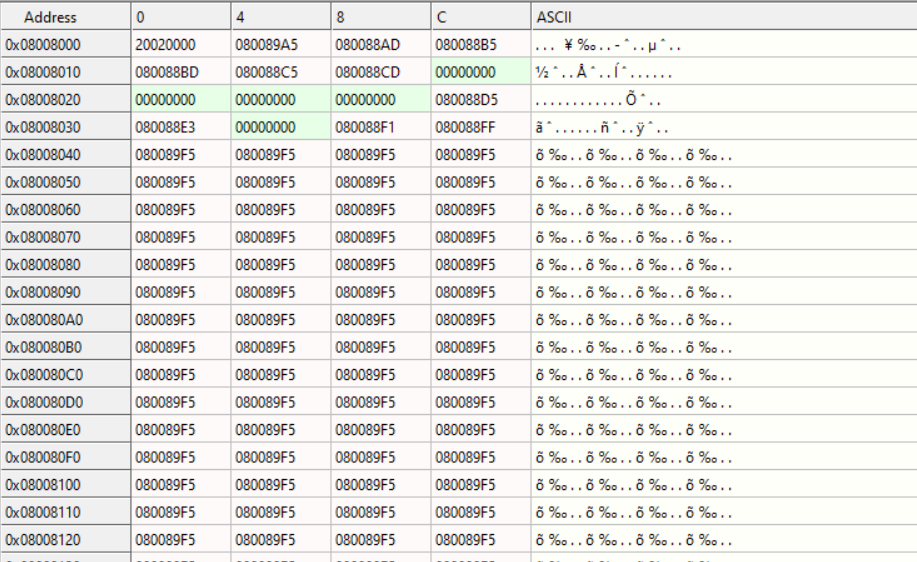
Açıklama otomatik olarak oluşturuldumetin, ekran görüntüsü, yazı tipi, doküman, belge içeren bir resim

Açıklama otomatik olarak oluşturuldu

Bu kod Flash hafızanın gerekli sektörünü önce temizler ve sonrasında belirtilen veriyi o hafıza alanına yazar.



**Tablo 1. Flash Hafızaya Yazmadan Önce Kod**



**Tablo 2. Flash Hafızaya Yazıldıktan Sonra**

Normalde Kod’un 0x800 8000 adresi Tablo 1. de gösterildiği gibidir. Ancak FTP ile çektiğimiz .bin dosyasını yazdıktan sonra artık Flash Hafızamızın 0x8008000 adresi de doldu. Şimdi yapmamız gereken bir şey kaldı o da kodun atlama yapması.

metin, ekran görüntüsü, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

Normalde kodumuz 0x800 0000 adresinden başlar ve aşağı doğru tek tek çalıştırarak gider. Bu fonksiyon ile çalışan kodumuz bulunduğu yerden appAddress’e atlar (Bu kod için 0x800 8000 adresi) .Biz bir önceki adımda bu adrese ana kodumuzun .bin dosyasını yüklediğimiz için artık ana kodumuz cihazımızda çalışmaya başlar.