İçindekiler

[1 TCP/IP ÖZELLİKLERİ 2](#_Toc524349846)

[2 LWIP KULLANIMI 4](#_Toc524349847)

[2.1 Raw API 4](#_Toc524349848)

[2.2 Netconn API 4](#_Toc524349849)

[2.3 Socket API 4](#_Toc524349850)

[3 LWIP ÖRNEKLERİ 4](#_Toc524349851)

[3.1 Statik IP İle lwIP Kullanımı 5](#_Toc524349852)

[3.2 Dinamik IP ile LwIP Kullanımı 7](#_Toc524349853)

[3.2.1 DHCP İstemcisi ile lwIP Kullanımı 7](#_Toc524349854)

[3.2.2 DHCP Sunucusu ile lwIP Kullanımı 9](#_Toc524349855)

[4 LWIP KODLARININ KULLANIMI VE AKIŞI 11](#_Toc524349856)

[4.1 Ağ Birimleri Netif Yönetimi 11](#_Toc524349857)

[4.2 UDP Kullanımı 12](#_Toc524349858)

[4.3 TCP Kullanımı 13](#_Toc524349859)

[4.3.1 Netconn API 13](#_Toc524349860)

[4.3.2 Raw API 15](#_Toc524349861)

[5 WEB SAYFASINUN OLUŞTURULMASI 16](#_Toc524349862)

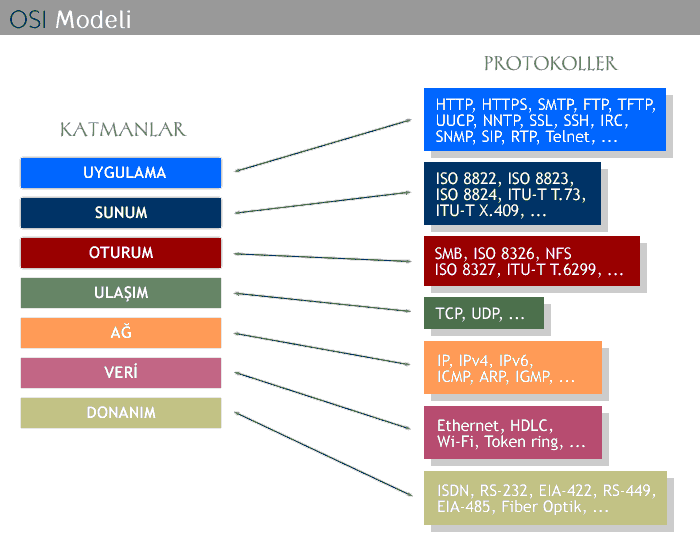
[5.1 Web Sayfalarının İşlemci İçinde Kullanımı 16](#_Toc524349863)

[5.2 HTML Sayfalarının Oluşturulması 17](#_Toc524349864)

# TCP/IP ÖZELLİKLERİ

Bilgisayar ağları kullanılarak bilgisayarların birbirileriyle haberleşmeye başladıkları ilk yıllarda iki bilgisayarın birbiriyle haberleşebilmeleri için aynı marka/model cihazları kullanmaları gerekiyordu. Bunun üzerine farklı üreticiler tarafından üretilen cihazların birbiriyle sorunsuz ve belirli bir düzen içinde haberleşebilmesi için çeşitli standartlar geliştirilmiştir

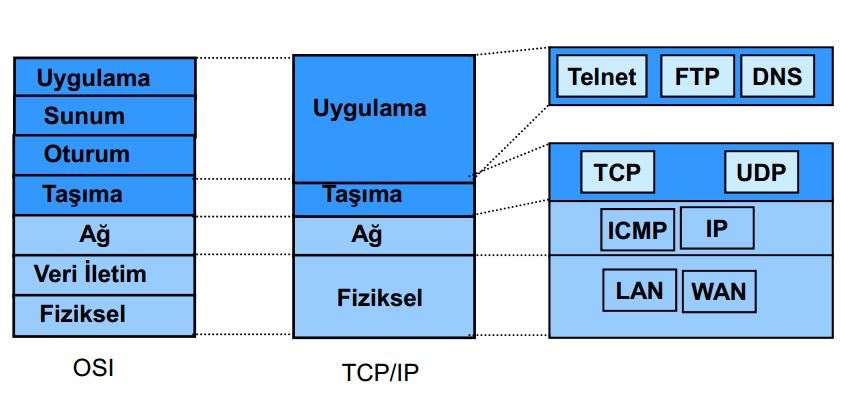
Bunlardan en çok kullanılanı Açık Sistem Bağlantıları komitesi tarafından geliştirilen 7 katmanlı OSI referans modeli ve Amerikan Savunma Bakanlığı tarafından geliştirilen TCP/IP referans modelidir. OSI iki bilgisayar arasındaki haberleşme problemini çözmek için 7 katmanlı (aşamalı) bir ağ sistemi önermiştir.



Şekil 1‑1

OSI referans modelindeki 7 katmana karşılık TCP/IP referans modeli 4 katmanlı bir çözüm sunar ve 7 katmanlı OSI modeline göre daha hızlı bir iletişim imkânı sunar. OSI modeli iletişim standartlarını belirlemeye yöneliktir ve TCP/IP daha uygulanabilir bir model olduğu için daha çok uygulamaya yöneliktir.

TCP/IP birçok protokolün toplandığı bir protokoller ailesidir. Bu referans modeline en çok kullanılan iki protokolün ismi verilmiştir; TCP (Transmission Control Protocol) ve IP (Internet Protocol). Bu referans modelinde 4 farklı katmanda 15’ten fazla protokol vardır. Veriler bu katmanlar arasında sırasıyla paketlenerek gönderilir, alıcıda ise paketlemenin tersi sırayla teker teker açılarak veri ulaştırılmış olur.



Şekil 1‑2

►Uygulama katmanı: Bu katmanda gönderilecek veri tipi ve veriyi işleyen uygulamalar bulunur. Örneğin bir HTML web sayfası ve bu veri tipini kullanan HTTP protokolü bu katmandadır. OSI modelindeki sunum ve oturum katmanları TCP/IP modelinde uygulama katmanı içerisinde yer alır. E-Posta gönderimi için kullanılan SMTP ve dosya gönderimi için kullanılan FTP protokolleri bu katmanda bulunur.

►Taşıma katmanı: Bu katmanda verinin nasıl gönderileceği belirlenir. Veri güvenliği, hata kontrolü gibi işlemler yapılır. TCP ve UDP bu katmandadır. TCP klasik veri aktarımında UDP ise medya aktarımında kullanılır. TCP, UDP ye göre daha güvenli fakat daha yavaş çalışır. Çünkü TCP 'de gönderilen her veri paketinin ardından verinin yerine doğru bir şekilde ulaşıp ulaşmadığı kontrol edilir.

►Ağ katmanı: IP katmanı olarak da adlandırılan bu katman da verilerin gideceği adres veriye eklenir yani veri bu katmandan gönderilir ve yönlendirilir. IPv4 ün gelecekte yetersiz kalma durumuna karşı IPv6 sistemine geçmek için çalışmalar başlatılmıştır. IPv4 32 bit iken IPv6 ile 128 bitlik adresler kullanılacak. Bu sayede daha fazla cihaza IP adresi atanabilecek.

►Fiziksel katman: Bu katman verinin hangi yolla gönderileceği belirlenir. İletişim ortamının özelliklerini, haberleşme hızını ve kodlama şemasını belirler. Ethernet, Wi-Fi, Token Ring, ATM gibi protokoller bu katmanda çalışır.

# LWIP KULLANIMI

Light Weight IP (lwIP), gömülü sistem TCP/IP uygulamalarında kullanılmak için geliştirilmiş bir uygulama platformudur. Günümüzde birçok gömülü sistem üreticileri tarafından kullanılmaktadır. Lwip işletim sistemi ile birlikte kullanılabilmesine rağmen, işletim sistemi bağımlılığı yoktur. İşletim sistemi olmayan platformlarda da çalışabilmektedir. LwIP uygulamalarında, üç çeşit Uygulama Programlama Arayüzü(API) bulunur.

## Raw API

Raw API, lwIP'nin temel API'sidir. Bu API, minimum kod boyutu kullanırken en iyi performansları sunmayı amaçlamaktadır. Bu API'nin dezavantajı ise, uygulama tasarımını karmaşık hale getiren geri çağrıları kullanarak asenkron olayları ele almasıdır.

## Netconn API

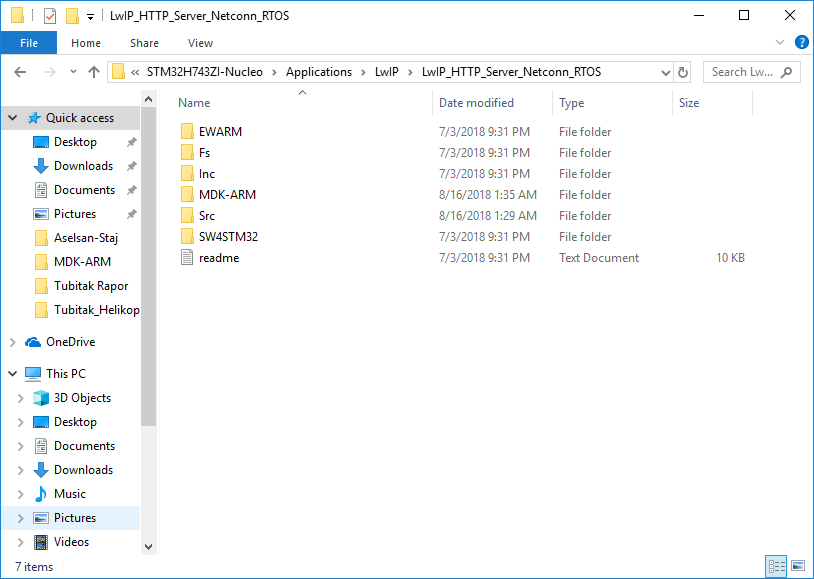
Raw API'nin üzerine yerleştirilmiş sıralı bir API'dir. Multi thread’li işlemlere izin vermektedir. Raw API’ye oranla düşük performans ve daha çok bellek kullanmasına ragmen kolay kullanımı nedeni ile tercih edilebilir.

## Socket API

Soket API, Netconn API'sinin üzerine kurulmuş bir API’dir. Yığın için BSD soket tarzı bir arayüz sağlar. Taşınabilir özelliktedir. Bu özellik RAW API’ye gore avantaj oluşturur. Fakat yüksek perfonmans ve düşük bellek tüketimi gerektiren uygulamalarda bu API’de RAW API’ye göre verimsiszdir.

# LWIP ÖRNEKLERİ

LwIP’nin yapısının anlaşılabilmesi için hazır örneklere bakılmalıdır. Bunun için ise kullanılacak olan işlemcinin CubeMX paketi indirilmelidir. İndirilen pakette Application içinde lwIP uygulaması bulunup açılmalıdır. Örnek olarak kullanılan STM32H743 kiti için CubeMX paketi indirildiğinde örnek uygulamanın bulunduğu dizin Şekil 3‑1 ile verilmiştir. Farklı işlemciler için örnek uygulama olmaması durumunda ise lwIP kütüphanesi manuel olarak eklenerek kullanılabilir.



Şekil 3‑1. Örnek uygulama

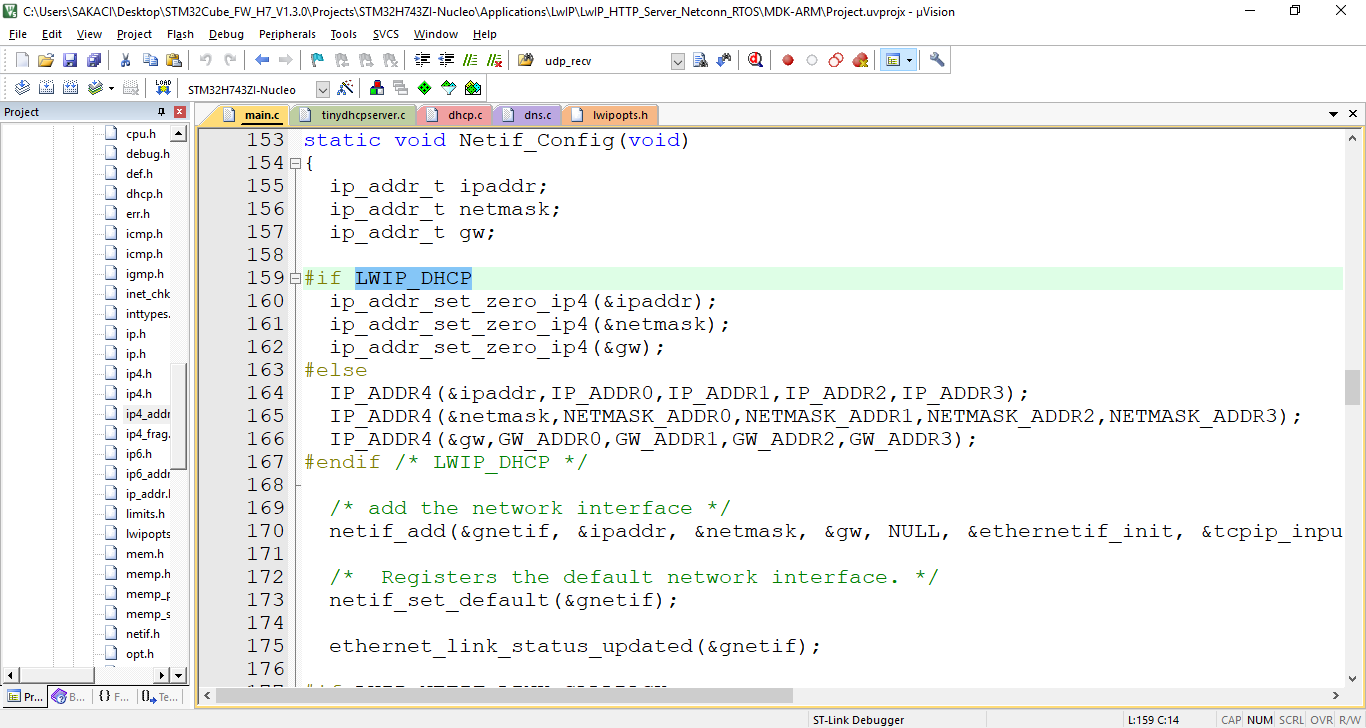
Kullanılacak olan hazır uygulamanın statik ve dinamik IP (DHCP aktif) aldırarak HTTP server açılması mümkündür. Fakat burada STM32H7 işlemcisi istemci(client) olarak kullanıldığı için otomatik ip aldırma durumunda DHCP Server bilgisayar tarafında oluşturulmalıdır. Eğer istemcinin bilgisayar olup sunucunun(server) STM32H7 işlemcisi olması isteniyor ise ek kütüphaneler kullanılarak DHCP Server yapılmalıdır.

## Statik IP İle lwIP Kullanımı

Statik IP adresi değişmeyen, sabit bir IP adresi anlamına gelmektedir. Yani manuel olarak el ile girilerek kullanılmaktadır. Fakat çok kullanıcılı sistemlerde çakışma durumu olmasından dolayı tercih edilmemektedir. Hazır olarak kullanılacak uygulamada ise öncelik olarak çerçevede verilen değişkenin define edildiği yere (lwipopts.h) gidilerek 0 yapılması gerekmektedir.

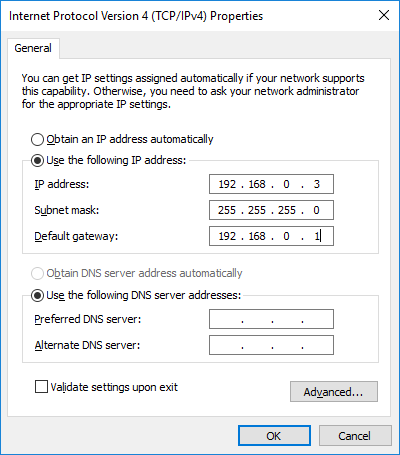
|  |
| --- |
| LWIP\_DHCP |

Daha sonra Şekil 3‑2 ile görüldüğü gibi tanımlanmış sabit ip adresi işlemci ip adresi olarak kullanılacak olur.



Şekil 3‑2

Fakat bilgisayar ile işlemcinin haberleşebilmesi için işlemciye verilmiş olan statik ip’nin komşusu (son hanesi farklı) bir ip adresi bilgisayara Şekil 3‑3 ile gösterildiği gibi verilmelidir.



Şekil 3‑3. IPv4 ayarları

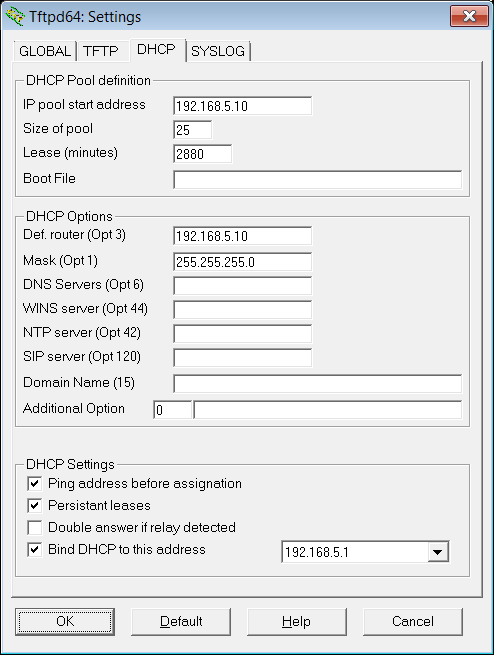
Bu işlem yapıldıktan işlemci içine yazılan ip adresi kullanılan tarayıcıya girilerek işlemci içine gömülmüş web sayfasına gidilebilir.

## Dinamik IP ile LwIP Kullanımı

DHCP sunucusunun kendine bağlanan istemcilere otomatik olarak ip vermesine dinamik ip denir. Dinamik ip özellikle çok kullanıcılı sistemlerde tercih edilmektedir. Ve kiralama mantığı ile çalışmaktadır.

### DHCP İstemcisi ile lwIP Kullanımı

Statik ip örneğinde DHCP\_LWIP değişkeni 0 olarak tanımlanmış ve bilgisayarla statik ip kullanılarak haberleşme sağlanmıştır. Burada ise değişken 1 yapılmalıdır. Değişken 1 yapıldıktan sonra istemci konumunda bulunan STM32H7 işlemcisi 0.0.0.0 ip adresinden 255.255.255.255(tüm yayın bantlarına) hedefine sunucu cevaplayana kadar periyodik olarak discover paketi atmaktadır. Fakat bilgisayarda DHCP sunucusu bulunmadığı için cevap atılamamaktadır. Bilgisayar da sunucu açılabilmesi için TFTPD32 yazılımı kullanılmalıdır. Burada öncelikle bilgisayara yine statik ip verilmelidir. Çünkü dinamik ip aldırma işleminde sunucu statik ip alıp komşu ip’lerini kendine bağlanan istemcilerine vermektedir. Statik ip verildikten sonra yazılım kullanılarak Şekil 3‑4 ile verildiği gibi ayarlar yapılır. Burada 192.168.0.10 bağlanan istemcilere 10’dan başlanarak yukarı doğru ip vermeye başlayacak manasına gelmektedir.



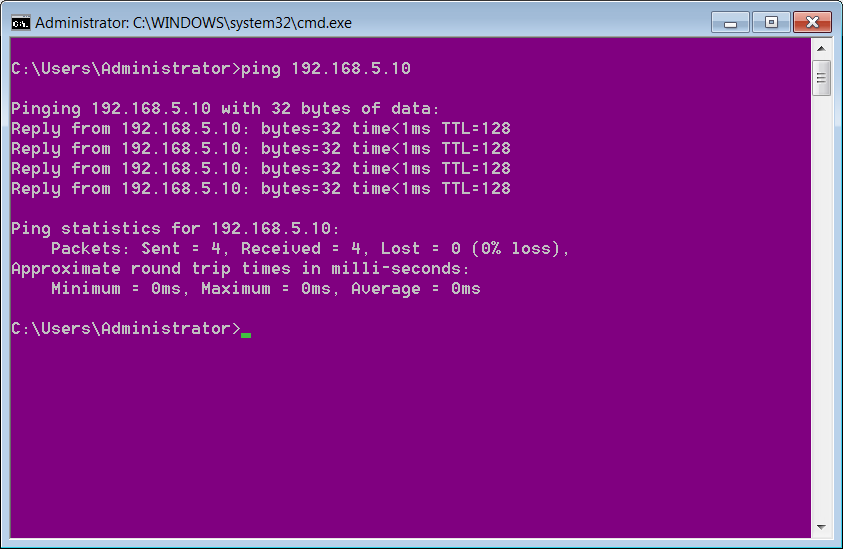
Şekil 3‑4

Yapılan ayarlamalardan sonra discover paketine karşılık bilgisayar offer paketi ile ip’yi gönderir. Bu ip’yi kabul ettğini söylemek için ise işlemci request paketini atar ve bu pakete karşılık ise acknowledge paketi gönderilip bağlantı sağlanır. Burada haberleşme 67. ve 68. portları kullanılarak yapılmaktadır. Paketlerle ilgili detaylı bilgi Tablo 3‑1 ile verilmiştir.

Tablo 3‑1

|  |
| --- |
| **Discover:** DHCP istemci bilgisayar tarafından ağa gönderilen ilk pakettir. Bu paket genel yayım (broadcast) olarak tüm ağa gönderilir. Bu paketin kaynak IP adresi kısmında istemci henüz bir IP adresine sahip olmadığı için 0.0.0.0 adresi bulunur. Hedef IP adresinde ise 255.255.255.255 bulunur. Bu pakette istemci makinenin MAC adresi, kaynak MAC adresi kısmında bulunur. İstemci makine bu mesaja gerekli cevabı alamadığı zaman periyodik olarak bu mesajı yayınlamaya devam eder. |
| **Offer:** DHCP sunucu "discover" mesajını alınca adres alanında kullanılmamış bir IP adresi seçerek bunu bu mesaj ile istemci makineye gönderir. Bu pakette henüz istemci makinenin IP adresi bulunmadığından broadcast olarak yayınlanır. Bu mesajda ayrıca alt ağ maskesi, varsayılan ağ geçidi gibi parametreler de bulunur. Bu mesajın hedef MAC adresi kısmında istemci makinenin MAC adresi bulunur. |
| **Request:** İstemci makine "offer" mesajını alınca gerekli parametreleri kabul ettiğini gösteren bu mesajı broadcast olarak yayınlar. DHCP, istemci makine "Dhcpoffer" mesajını alamazsa 2., 4., 8. ve 16. saniyelerde birer "discover" mesajı daha gönderir. Yine cevap alamazsa her 5 dakikada periyodik olarak birer tane daha gönderir. |
| **Acnowledge:** IP adres önerisini kabul eden istemci makineye DHCP sunucusu tarafından gönderilir. İstemci bu mesajı alamadan makine TCP/IP protokolünü kullanamaz. |

DHCP sunucusunun açıldığından emin olmak için ise Şekil 3‑5 ile gösterildiği gibi açılan sunucu ip’sine ping atılabilir.

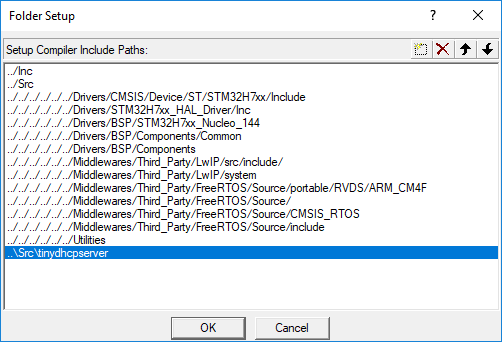


Şekil 3‑5

DHCP sunucusunun açıldığı teyit edildikten sonra istemcinin aldığı ip tarayıcıya girilerek işlemci içinde gömülü bulunan web sayfası açılabilir.

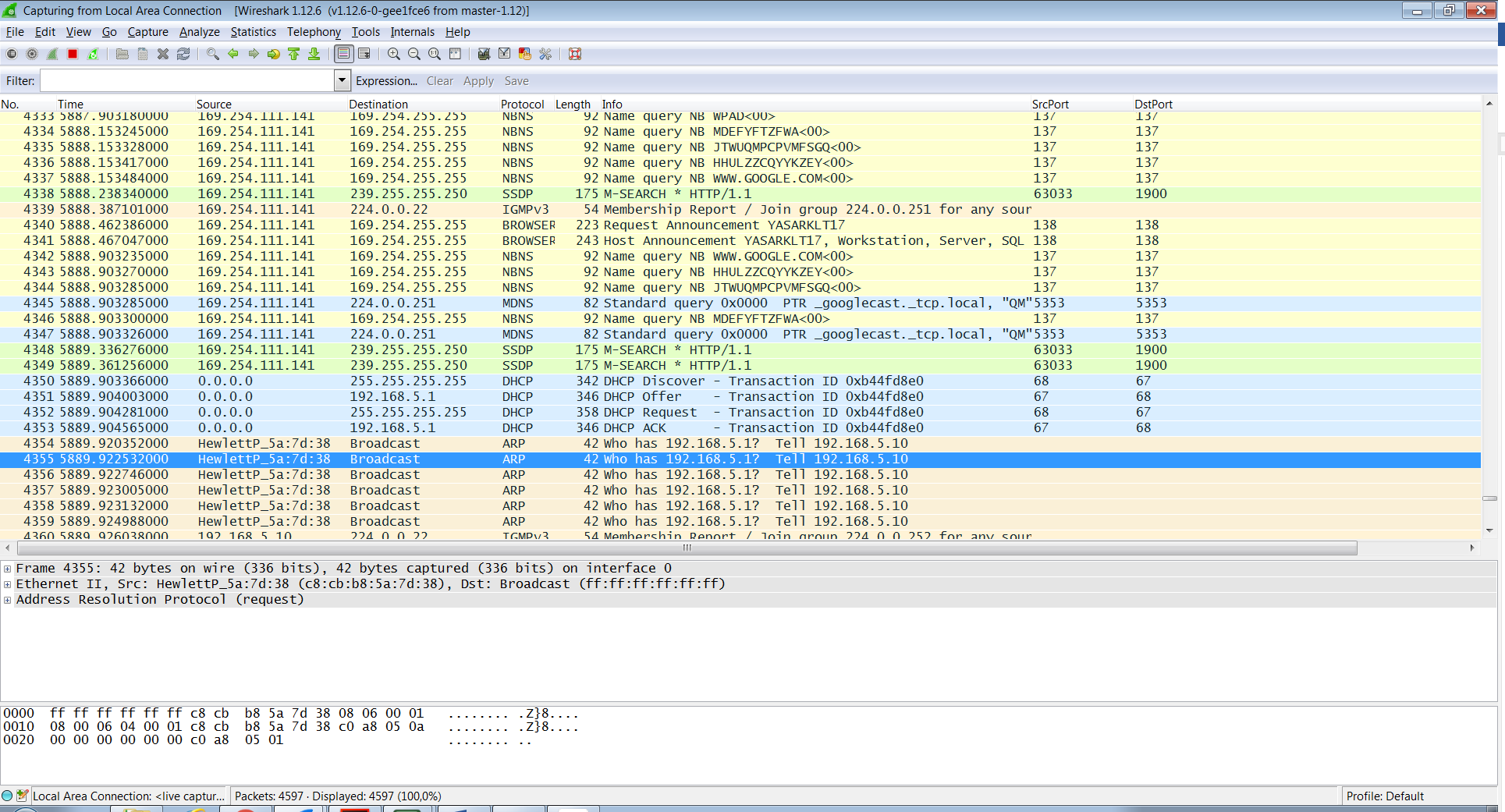
### DHCP Sunucusu ile lwIP Kullanımı

DHCP sunucusunun STM32H7 işlemcisi üzerinde çalışabilmesi için daha önce de bahsedildiği gibi farklı dosyalara da ihtiyaç vardır. Bunun için ATMEL firmasının hazırladığı tinydhcpserver dosyası kullanılabilir. Burada öncelikli olarak bu dosya proje dosyasının içine atılmalıdır. Ve daha sonra başlık dosyaları keil üzerinde Şekil 3‑6 ile verildiği gibi gösterilmelidir. Gösterme işlemi yapıldıktan sonra kaynak kodları projeye eklenmelidir.



Şekil 3‑6

Bu işlemler yapıldıktan sonra sürüm farkından kaynaklı uyumsuzluk hataları giderilmelidir. Burada uyumsuzluk hatasının nedeni tinydhcpserver dosyası lwIP 1.4 için yapılmıştır. Fakat kullanılan lwIP sürümü 2.1’dir. Bu hatalar giderildikten sonra ise main.c dosyası içinde tiny dosyalarının başlık dosyaları eklenmeli ve istemci için bulunnan DHCP thread’i kaldırılıp thread yerine lwip\_tiny\_dhcpserver\_start() denilmelidir. Böylece tinydhcpserver.c dosyası içinde bulunan server başlatma fonksiyon çağırılmış ve DHCP Server başlatılmıştır. Daha sonra bilgisayardan otomatik ip seçilmelidir. Böylece bu sefer bilgisayar 0.0.0.0 adresinden 255.255.255.255 adresine ip’sinin olmadığını ve ona bir router’ın ip vermesini istediğini periyodik olarak discover paketi göndererek söyleyecektir. Bunu alan STM32H7 işlemcisi ise bilgisayara ip verecektir. Paketlerin akışı Wireshark ile alınarak Şekil 3‑7 ile verilmiştir.

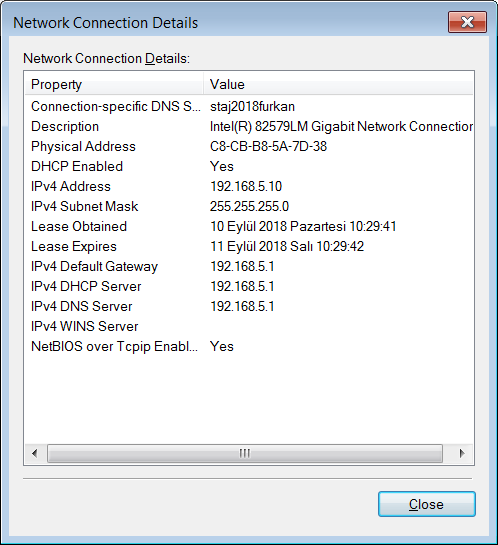


Şekil 3‑7

Burada DHCP paketleri görüldüğü üzere sunucu ve istemci arasında konuşma şeklindedir. Bu paketler atıldıktan sonra ise HTTP Server açılması için uygun ortam sağlanmıştır.

**Not:** Tinydhcpserver.c dosyası içinde offer paketini doldururken tam doldurmamasından dolayı yer açılırken PBUF\_POOL olan kısım PBUF\_RAM yapılmalı ve daha önceden bahsedildiği gibi DHCP Server’ın kendisinin belli bir ip’ye sahip olması gerektiğinden dolayı ip4\_addr.h dosyası içinde bulunan IP\_ADDR\_ANY değişkenine hexadecimal olarak ip adresi verilmelidir.

Sonuç olarak bakıldığında yapılan işlemlerden sonra istemci durumunda olan bilgisayarın aldığı değerler Şekil 3‑8 ile verilmiştir.



Şekil 3‑8.

# LWIP KODLARININ KULLANIMI VE AKIŞI

LwIP ile bağlantı kurulması için öncelikli olarak fizikle ağ donanım ayarlamaları yapılmalıdır. Bu ayarlamalar yapıldıktan sonra ise UDP kullanılarak bağlantı açılmalıdır. Örnek olarak bu bağlantı DHCP Server için Discover, Offer, Request ve ACK paketleri ile sağlanmaktadır. Bağlantı da açıldıktan sonra artık TCP kullanılarak HTTP Server kullanılabilir.

## Ağ Birimleri Netif Yönetimi

LWip’de fiziksel ağ donanımı ile iletişim ağ arabirim (netif) yapısı ile sağlanır. Bu yapı fiziksel ağ donanımı ile lwip arasında aygıt sürücülüğünü üstlenir. Programda yeni bir ağ arabirimi oluşturmak için yeni bir struct netif için kullanım alanı ayrılır ve çerçevede verildiği gibi netif\_add fonksiyonu çağrılır.

|  |
| --- |
| struct \*netif netif\_add(struct netif \*mynetif, struct ip\_addr \*ipaddr, struct ip\_addr \*netmask, struct ip\_addr \*gw, void \*state, err\_t (\* init)(struct netif \*netif), err\_t (\* input)(struct pbuf \*p, struct netif \*netif)); |

Eğer eklenen netif’in varsayılan netif olarak düşünülüyorsa netif\_set\_default fonksiyonu çerçevedeki gibi çağrılarak eklenen netif varsayılan olarak yapılır. Eğer IP yığını paket yönlendirmesi sırasında kullanılacağı arabirimi doğru belirleyemiyorsa, IP yığını bu paketi varsayılan arabirime gönderir.

|  |
| --- |
| netif\_set\_default (struct netif \*netif) |

Oluşturulan netif kullanılamadan önce netif\_set\_up fonksiyonu çerçevedeki gibi kullanılarak netif aktif hale getirilir.

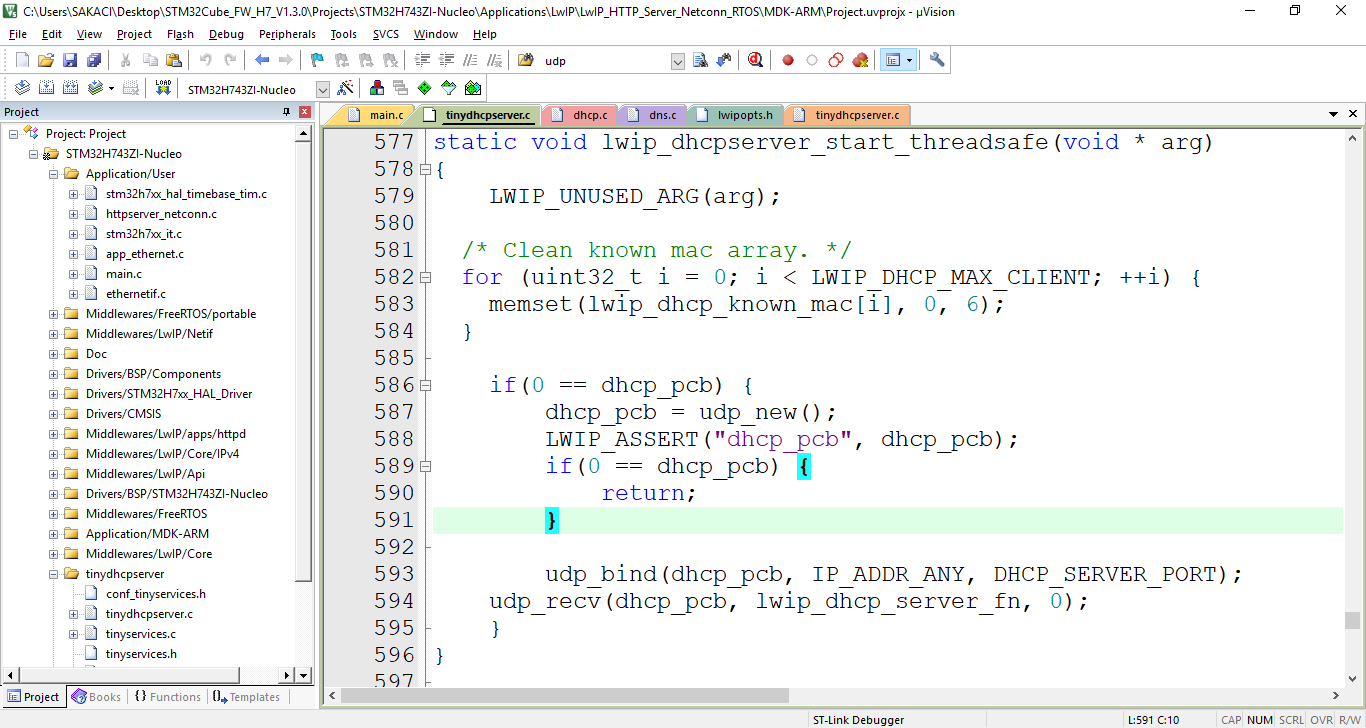
|  |
| --- |
| netif\_set\_up (struct netif \*netif) |

Bu işlemler yapıldıktan sonra arabirim yapısı kurulmuş olur.

## UDP Kullanımı

UDP (User Datagram Protocol) Türkçesi ile Kullanıcı Veribloğu İletişim Kuralları, TCP/IP protokol takımındaki iki aktarım katmanı protokolünden bir tanesidir ve verilerin bağlantı kurmadan yollanmasını sağlamaktadır. STM32H7 içerisinde UDP bilgisayar ile iletişime geçmek için kullanılmıştır. Ağ arabirim yapısı kurulduktan sonra ip yapılandırması için acak ise UDP kullanılarak paketler (Discover, Offer, Request, ACK) gönderilip alınmaktaıdır.

Şekil 4‑1 ile verilmiş tinydhcpserver kaynak kodunda görüldüğü gibi öncelikle udp\_new kullanılarak yeni bir bağlantı tanımlayıcısı (Protokol Kontrol Bloğu) oluşturulmuştur. Oluşturulan bağlantı tanımlayıcısı (PCB) bir yerel yada uzak adrese bağlanmadığı sürece aktif değildir. Daha sonra udp\_bind() kullanılarak oluşturulan bir PCB’nin yerel ip adresi ile port ‘a bağlanmasını sağlar. Eğer herhangi bir yerel ip adresine bağlanılması isteniyorsa ip adresi yerine IP\_ADDR\_ANY ifadesi yazılır. Herhangi bir port numarası için ise port değeri 0 girilir. Ve ayarlamalarda son olarak udp\_recv fonksiyonu kullanılmıştır. Bu fonksiyon ise UDP verisi geldiğinde işlenecek olan fonksiyonu tayin eder. Gelen veri fonksiyonda belirtilen veri paket tamponu(pbuf) formatındaki değere atanır.



Şekil 4‑1

Ayarlamalar yapıldıktan sonra gelen very lwip\_dhcp\_server\_fn fonksiyon içine gidecektir. Bu fonksiyon içine bakıldığında ise gelen pakete göre pbuf doldurulduktan sonra çerçevedeki fonksiyon kullanılarak gönderilme yapılabilir.

|  |
| --- |
| udp\_sendto(dhcp\_pcb, pbuf\_out, &ip\_addr\_c\_broadcast, DHCP\_CLIENT\_PORT); |

Bağlantı sağlandıktan sonra ise HTTP Server açılabilecektir. Fakat bu işlem için TCP gerekmektedir.

## TCP Kullanımı

Bir uygulama başka bir uygulama ile iletişime (TCP kullanarak) geçmek istediğinde; bir haberleşme talebi gönderir. Bu talebin tam adrese gönderilmesi gerekmektedir. İki uygulamanın buluşup bir el sıkışmasından sonra, TCP iki uygulamadan biri tek taraflı olarak bağlantısını kesene kadar çift şeritli bir iletşim hattı kurar. TCP kullanımı için Raw API veya Netconn API kullanılabilir.

### Netconn API

Verilmiş olan örneklerde kullanılan Netconn API kolay kullanımı nedeni ve multi thread’e izin vermesi nedeniyle tercih edilmektedir. Netconn API’de bağlantı açılmasının nasıl yapılacağı Tablo 4‑1 ile verilmiştir.

Tablo 4‑1

|  |
| --- |
| struct netconn \*conn, \*newconn;  err\_t err, accept\_err;  conn = netconn\_new(NETCONN\_TCP);  if (conn!= NULL){  err = netconn\_bind(conn, NULL, 80);  if (err == ERR\_OK) {  netconn\_listen(conn);  while(1) {  accept\_err = netconn\_accept(conn, &newconn);  if(accept\_err == ERR\_OK) {  http\_server\_serve(newconn);  netconn\_delete(newconn);  }}}}} |

Burada görüldüğü üzere ilk önce TCP bağlantısı oluşturulmuştur. Daha sonra oluşturulan bağlantı ip ve port değerleri girilerek açılmıştır. Ve “netconn\_listen” ile dinleme moduna geçilmiştir. Daha sonra TCP paketinin gelmesi durumunda Tablo 4‑2 ile bir kısmı verilen “http\_server\_serve” fonksiyonuna gidilip web sayfaları açılmıştır. Bu fonksiyonun açıklaması ise bölüm 5 ile açıklanmıştır.

Tablo 4‑2

|  |
| --- |
| recv\_err = netconn\_recv(conn, &inbuf);  netbuf\_data(inbuf, (void\*\*)&buf, &buflen);  if ((buflen >=5) && (strncmp(buf, "GET /", 5) == 0))  {  if (strncmp((char const \*)buf,"GET /STM32H7xx\_files/ST.gif",27)==0)  {  fs\_open(&file, "/STM32H7xx\_files/ST.gif");  netconn\_write(conn, (const unsigned char\*)(file.data), (size\_t)file.len, NETCONN\_NOCOPY);  fs\_close(&file);  }  else if (strncmp((char const \*)buf,"GET /STM32H7xx\_files/stm32.jpg",30)==0)  {  fs\_open(&file, "/STM32H7xx\_files/stm32.jpg");  netconn\_write(conn, (const unsigned char\*)(file.data), (size\_t)file.len, NETCONN\_NOCOPY);  fs\_close(&file);  }  else if (strncmp((char const \*)buf,"GET /STM32H7xx\_files/logo.jpg", 29) == 0)  {  fs\_open(&file, "/STM32H7xx\_files/logo.jpg");  netconn\_write(conn, (const unsigned char\*)(file.data), (size\_t)file.len, NETCONN\_NOCOPY);  fs\_close(&file);  }  else if(strncmp(buf, "GET /STM32H7xxTASKS.html", 24) == 0)  { DynWebPage(conn); }  else if((strncmp(buf, "GET /STM32H7xx.html", 19) == 0)||(strncmp(buf, "GET / ", 6) == 0))  {  fs\_open(&file, "/STM32H7xx.html");  netconn\_write(conn, (const unsigned char\*)(file.data), (size\_t)file.len, NETCONN\_NOCOPY);  fs\_close(&file); } } |

### Raw API

Uygulama örneği STM32H7 için bulunmayan Raw API’nin kullanımı Netconn API’ye oranla daha zordur. Öncelikli olarak çerçevede ki kod kullanılarak bağlantı noktası oluşturulmalıdır.

|  |
| --- |
| pcb = tcp\_new(); |

Daha sonra oluşturulan bağlantı noktası çerçevedeki gibi ip ve port girilerek açılır.

|  |
| --- |
| tcp\_bind(pcb, local\_addr, HTTPD\_SERVER\_PORT); |

Bağlantı açıldıktan sonra TCP paketlerini almak için dinleme moduna geçilerek gelen paketlerin hangi fonksiyona gideceği çerçevedeki kod ile söylenir.

|  |
| --- |
| pcb = tcp\_listen(pcb);  tcp\_accept(pcb, http\_accept); |

Burada bağlantı açmak için TCP paketi geldiğinde “http\_accept” fonksiyonuna gitmektedir. Bu fonksiyon içinde ise gelen bağlantının kabul edildiği söylenerek HTTP sunucusu açılması için çerçevedeki geri çağrılar tanımlanır.

|  |
| --- |
| tcp\_accepted(lpcb);  tcp\_recv(pcb, http\_recv);  tcp\_poll(pcb, http\_poll, HTTPD\_POLL\_INTERVAL);  tcp\_sent(pcb, http\_sent); |

Bu geri çağrılar(callback) içinde daha sonra web sayfası açılması için gerekli ayarlar yapılır. Burada http\_recv fonksiyonunda gelen TCP paketlerini alıp ona göre işlem yapar. Http\_poll fonksiyonu ise Specifies the polling interval and the callback function that should be called to poll the application. This feature can be used to check if there are some pending data to be sent or if the connection needs to be closed.

# WEB SAYFASINUN OLUŞTURULMASI

## Web Sayfalarının İşlemci İçinde Kullanımı

Tablo 4‑2 ile verilmiş olan fonksiyonda web sayfalarının kullanımı bulunmaktadır. Burada gelen TCP paketinde GET veya POST protokolü ile yönlendirilen sayfa bilgileri gelmektedir. Genelde gizlilik içeren değerler POST methoduyla gönderilirken içermeyip link kısmında gözükmesi istenen değerler GET methoduyla gönderilmektedir. Link kısmını almak için GET methodu kullanılmıştır. Böylece tarayıcı üzerinden çerçevedeki gibi bir link girildiğinde;

|  |
| --- |
| http://192.168.0.10/STM32H7xx.html |

GET methodu ile “GET /STM32H7xx.html” değeri alınacak ve yapılan kontrolden sonra STM32H7xx.html dosyasına gidilecektir. Ve eğer anlık olarak işlemci içindeki veriler web sayfasında gösterilecekse Tablo 5‑1 ile verilmiş fonksiyon kullanılabilir.

Tablo 5‑1

|  |
| --- |
| void DynWebPage(struct netconn \*conn)  {  portCHAR PAGE\_BODY[512];  portCHAR pagehits[10] = {0};  memset(PAGE\_BODY, 0,512);  nPageHits++;  sprintf(pagehits, "%d", (int)nPageHits);  strcat(PAGE\_BODY, pagehits);  strcat((char \*)PAGE\_BODY, "<pre><br>Name State Priority Stack Num" );  strcat((char \*)PAGE\_BODY, "<br>---------------------------------------------<br>");  osThreadList((unsigned char \*)(PAGE\_BODY + strlen(PAGE\_BODY)));  strcat((char \*)PAGE\_BODY, "<br><br>---------------------------------------------");  strcat((char \*)PAGE\_BODY, "<br>B: Blocked, R: Ready, D: Deleted, S: Suspended<br>");  netconn\_write(conn, PAGE\_START, strlen((char\*)PAGE\_START), NETCONN\_COPY);  netconn\_write(conn, PAGE\_BODY, strlen(PAGE\_BODY), NETCONN\_COPY);  } |

Burada PAGE\_START fonksiyonu web sayfası şeklinde yazılıp hexadecimal şekline dönüştürülmüş bir şablondur. Ve bu şablonun üstüne ise PAGE\_BODY değişkeni oluşturulup giydirilmiştir. PAGE\_BODY aynı zamanda anlık olarak değiştiği için PAGE\_START şablonu içinde JavaScript kodları da bulunmaktadır. Şayet internet üzerinden değer almak isteniyorsa GET ve POST methodlarının kullanımının iyice kavranması gerekmektedir.

**GET Methodu**

Bu method da yapılan web sayfasında GET methodu kullanılarak değer aktarılma yapıldıysa link kısmı örnek olarak çerçevedeki gibi olur.

|  |
| --- |
| http://192.168.0.10/STM32H7xx.html?PWM1=255&PWM2=120&... |

Daha sonra bu link string olarak çekilip parçalama yapılarak kullanılabilir. Ve bu sayade PWM değerleri web sayfası üzerinden ayarlanmış olur.

**POST Methodu**

Bu method da ise gönderilen bilgiler link kısmında gözükmeyip gönderilir. Fakat gömülü sistemde bunun alınmasına örnek yoktur.

## HTML Sayfalarının Oluşturulması

