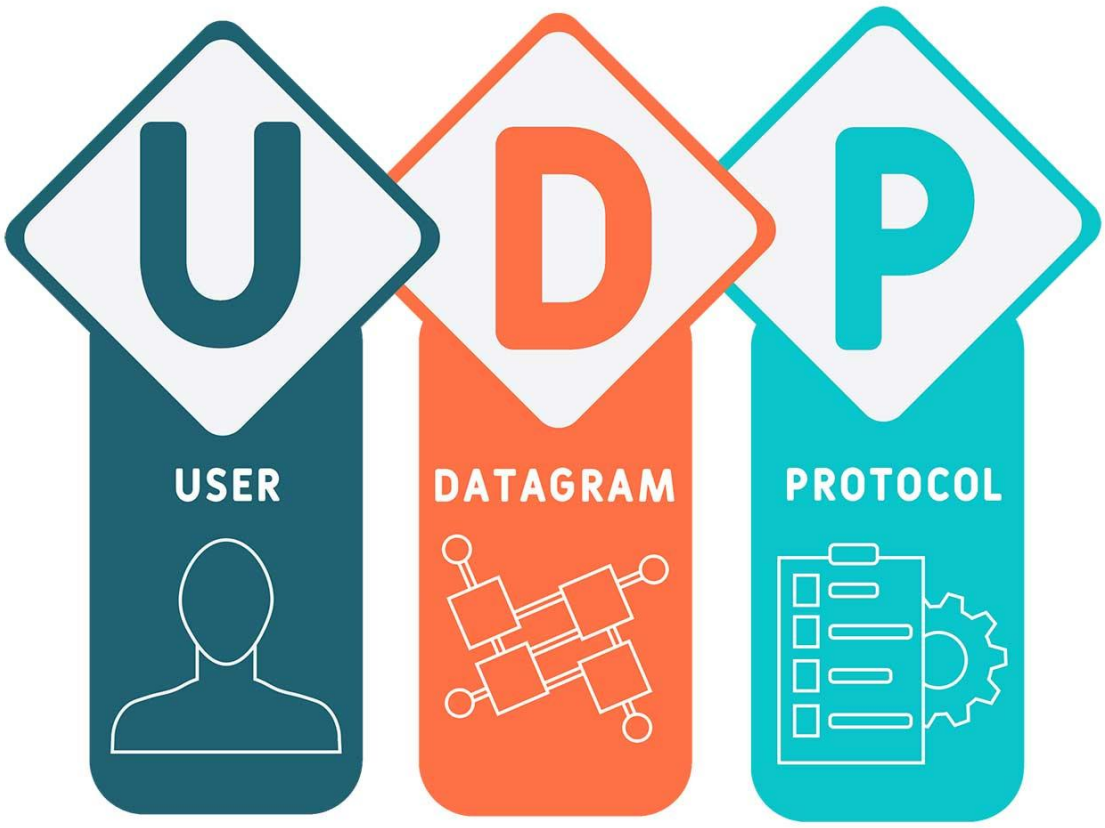
Networking

UIPEthernet

UDP



By Abdelrahman Yasser

Introduction

User Datagram Protocol (UDP) refers to a protocol used for communication throughout the internet. It is specifically chosen for time-sensitive applications like gaming, playing videos, or Domain Name System (DNS) lookups. UDP results in speedier communication because it does not spend time forming a firm connection with the destination before transferring the data.

What uses UDP?

UDP is frequently used when communications are time-sensitive. For users, it is better to have the overall transmission arrive on time than wait for it to get there in a near-perfect state. For this reason, UDP is commonly used in Voice over Internet Protocol (VoIP) applications as well.

For the listener, hearing what the speaker said relatively soon after it was spoken is preferable to waiting several seconds for crystal-clear speech. Similarly, with online gaming, experiencing less-than-ideal video or sound for a few moments is preferable to waiting for a clear transmission and risking losing the game in the interim.

UDP in Arduino

The UIPEthernet Library provides the feature of using both TCP and UDP protocols but in this report, we will be focusing on UPD

begin(uint16_t port) Function

This function initializes and starts listening on the specified port. Returns 1 if successful, 0 if there are no sockets available to use

stop(void) Function

Finish with the UDP socket

beginPacket(IPAddress ip, uint16_t port) / beginPacket(const char *host, uint16_t port) Function

Start building up a packet to send to the remote host specific in ip and port, Returns 1 if successful, 0 if there was a problem with the supplied IP address or port

endPacket(void) Function

Finish off this packet and send it, Returns 1 if the packet was sent successfully, 0 if there was an error

write(uint8_t c) Function

Write a single byte into the packet

write(const uint8_t *buffer, size_t size) Function

Write size bytes from a buffer into the packet

parsePacket(void) Function

Start processing the next available incoming packet, Returns the size of the packet in bytes, or 0 if no packets are available

available(void) Function

Number of bytes remaining in the current packet

read(void) Function

Read a single byte from the current packet

read(unsigned char* buffer, size_t len) Function

Read up to len bytes from the current packet and place them into buffer, Returns the number of bytes read, or 0 if none are available

peek(void) Function

Return the next byte from the current packet without moving on to the next byte

flush(void) Function

Finish reading the current packet

remoteIP(void) Function

Return the IP address of the host who sent the current incoming packet

remotePort(void) Function

Return the port of the host who sent the current incoming packet