

*****Draft*****

Ultibo UDPSERVER running a RPi4

Sending a message to RPi4 running Linux on UDP port 514

06/06/21

*****Draft*****

Original code was found at

<https://github.com/ultibohub/Examples/tree/master/Advanced/UDPServer/RPi2>

gcc server.c -o server

server.c: In function 'main':

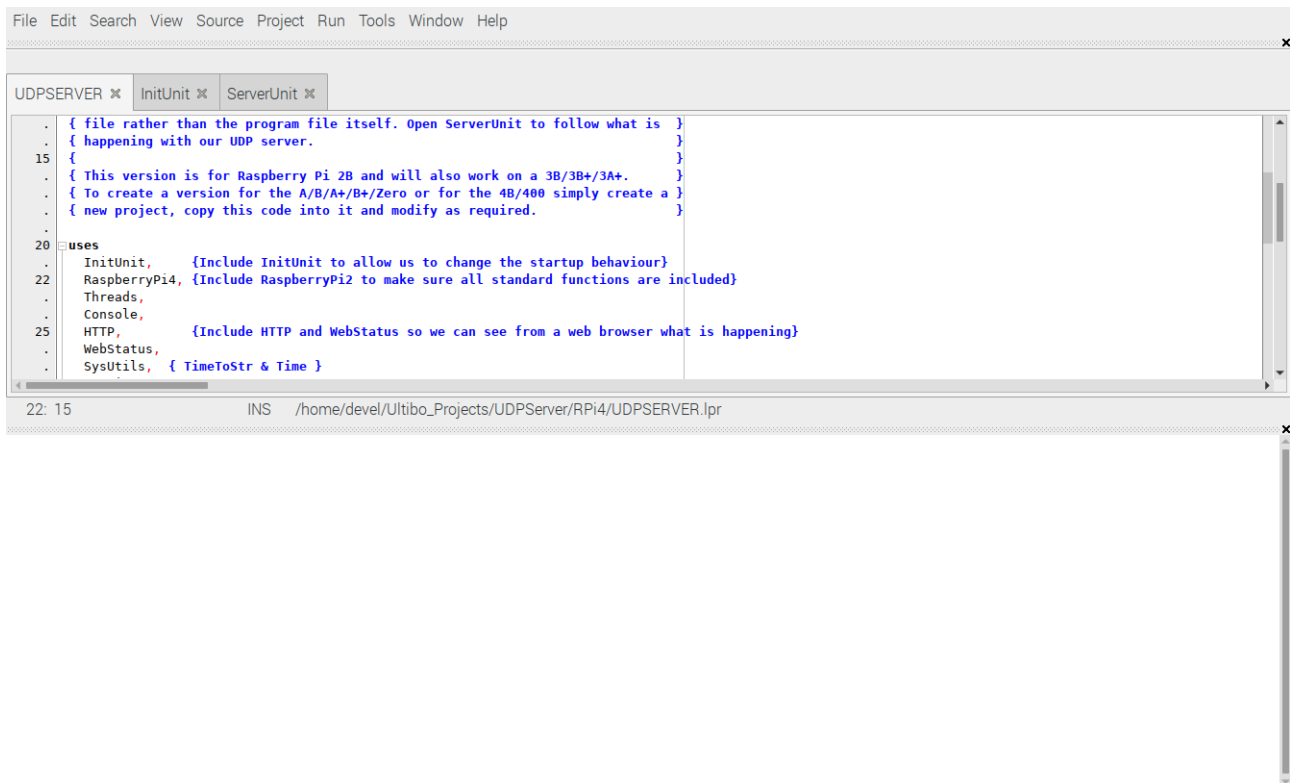
server.c:68:5: warning: implicit declaration of function 'close'; did you mean 'pclose'? [-Wimplicit-function-declaration]

close(s);

^~~~~~

pclose

Lazarus IDE (Ultibo Edition)



Run/Compile

The screenshot displays the Ultibo IDE interface. At the top is a menu bar with options: File, Edit, Search, View, Source, Project, Run, Tools, Window, and Help. Below the menu bar, there are tabs for the open files: UDPSERVER.lpr, InitUnit, and ServerUnit. The main editor window shows the source code for UDPSERVER.lpr. The code is as follows:

```
. { file rather than the program file itself. Open ServerUnit to follow what is }  
. { happening with our UDP server. }  
15 { }  
. { }  
. { This version is for Raspberry Pi 2B and will also work on a 3B/3B+/3A+. }  
. { To create a version for the A/B/A+/B+/Zero or for the 4B/400 simply create a }  
. { new project, copy this code into it and modify as required. }  
  
20 uses  
.   InitUnit,      {Include InitUnit to allow us to change the startup behaviour}  
22   RaspberryPi4, {Include RaspberryPi2 to make sure all standard functions are included}  
.   Threads,  
.   Console,  
25   HTTP,          {Include HTTP and WebStatus so we can see from a web browser what is happening}  
.   WebStatus,  
.   CpuUtil,       {TimeToGet & Time }
```

Below the editor window, a status bar shows the current file path: 22: 15 INS /home/devel/Ultibo_Projects/UDPServer/RPi4/UDPSERVER.lpr. At the bottom of the IDE, a green message bar displays the text: "Compile Project, OS: ultibo, Target: UDPSERVER: Success".

The green bar indicates that the kernel7l.img was successfully created.

```
tftp 192.168.1.143 < cmdstftp
```

```
tftp> tftp> Sent 2672408 bytes in 5.3 seconds
```

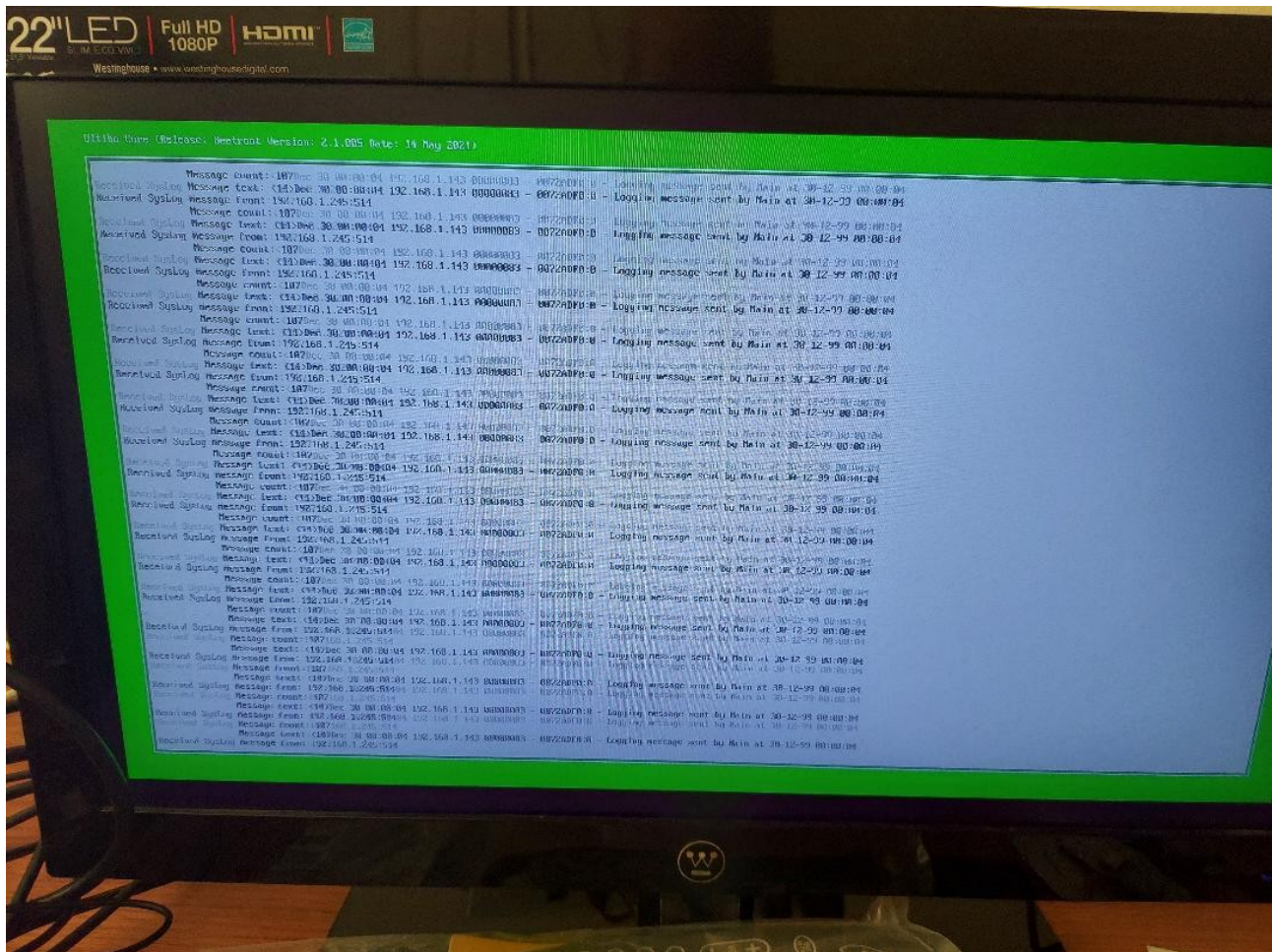
```
sudo ./server
```

```
[sudo] password for devel:
```

Waiting for data...

```
File Edit Tabs Help
```

```
Data: <30>Jun 07 00:52:24 192.168.1.143 Services: 0000007D - 00A7E0C0:3 - Service  
es: NTP: Setting time to 7-6-21 00:52:24 (from 30-12-99 00:00:03)  
Waiting for data...Received packet from 192.168.1.143:514  
Data: <30>Jun 07 00:52:24 192.168.1.143 Services: 0000007D - 00A7E0C0:3 - Service  
es: NTP: Setting time to 7-6-21 00:52:24 (from 30-12-99 00:00:03)  
Waiting for data...Received packet from 192.168.1.143:514  
Data: <30>Jun 07 00:52:24 192.168.1.143 Services: 0000007D - 00A7E0C0:3 - Service  
es: NTP: Setting time to 7-6-21 00:52:24 (from 30-12-99 00:00:03)  
Waiting for data...Received packet from 192.168.1.143:514  
Data: <30>Jun 07 00:52:24 192.168.1.143 Services: 0000007D - 00A7E0C0:3 - Service  
es: NTP: Setting time to 7-6-21 00:52:24 (from 30-12-99 00:00:03)  
Waiting for data...Received packet from 192.168.1.143:514  
Data: <30>Jun 07 00:52:24 192.168.1.143 Services: 0000007D - 00A7E0C0:3 - Service  
es: NTP: Setting time to 7-6-21 00:52:24 (from 30-12-99 00:00:03)  
Waiting for data...
```



server.c

```
/*
 * Simple udp server
 */
#include<stdio.h> //printf
#include<string.h> //memset
#include<stdlib.h> //exit(0);
#include<arpa/inet.h>
#include<sys/socket.h>

#define BUFLen 1024 //Max length of buffer
#define PORT 514 //The port on which to listen for incoming data

void die(char *s)
{
    perror(s);
    exit(1);
}

int main(void)
{
    struct sockaddr_in si_me, si_other;
```

```

int s, i, slen = sizeof(si_other) , recv_len;
char buf[BUFLen];

//create a UDP socket
if ((s=socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP)) == -1)
{
    die("socket");
}

// zero out the structure
memset((char *) &si_me, 0, sizeof(si_me));

si_me.sin_family = AF_INET;
si_me.sin_port = htons(PORT);
si_me.sin_addr.s_addr = htonl(INADDR_ANY);

//bind socket to port
if( bind(s , (struct sockaddr*)&si_me, sizeof(si_me) ) == -1)
{
    die("bind");
}

//keep listening for data
while(1)
{
    printf("Waiting for data...");
    fflush(stdout);

    //try to receive some data, this is a blocking call
    if ((recv_len = recvfrom(s, buf, BUFLen, 0, (struct sockaddr *) &si_other, &slen)) == -1)
    {
        die("recvfrom()");
    }

    //print details of the client/peer and the data received
    printf("Received packet from %s:%d\n", inet_ntoa(si_other.sin_addr),
    ntohs(si_other.sin_port));
    printf("Data: %s\n" , buf);

    //now reply the client with the same data
    if (sendto(s, buf, recv_len, 0, (struct sockaddr*) &si_other, slen) == -1)
    {
        die("sendto()");
    }
}

close(s);
return 0;
}

```