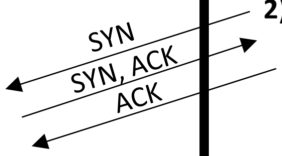
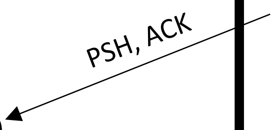
ECHO-TCP server diagram created by vladimirsustek for personal education needs 14.3.2022

LWIP Server (MCU) C

```
LWIP_INIT()
IP4_ADDR(&IP, 192, 168, 100, 1);

struct tcp_pcb *pcb;

1)  pcb = tcp_new();
    tcp_bind(pcb, &IP, port);
    tcp_listen(pcb);
    tcp_accept(pcb, cb_accept);

2)  cb_accept(arg, newpcb, err)
    {
    struct tcp_server_struct *es;

    LWIP_UNUSED_ARG(arg)
    LWIP_UNUSED_ARG(err)

    tcp_set_prio(newpcb, TCP_PRIO_MIN)

    es = (struct tcp_server_struct *)mem_malloc(
      sizeof(struct tcp_server_struct));

    es->state = ES_ACCEPTED;
    es->pcb = newpcb;
    es->p = NULL;

    tcp_arg(newpcb, es);
    tcp_recv(newpcb, cb_recved);
    tcp_err(newpcb, cb_error);
    tcp_poll(newpcb, cb_poll, 1);

    ret_err = ERR_OK;
    return ret_err;
    }

3)  cb_recved(arg, tpcb, p, err)
    {
    struct tcp_server_struct *es;
    err_t ret_err;

    LWIP_ASSERT("arg != NULL",arg != NULL);
    es = (struct tcp_server_struct *)arg;
```

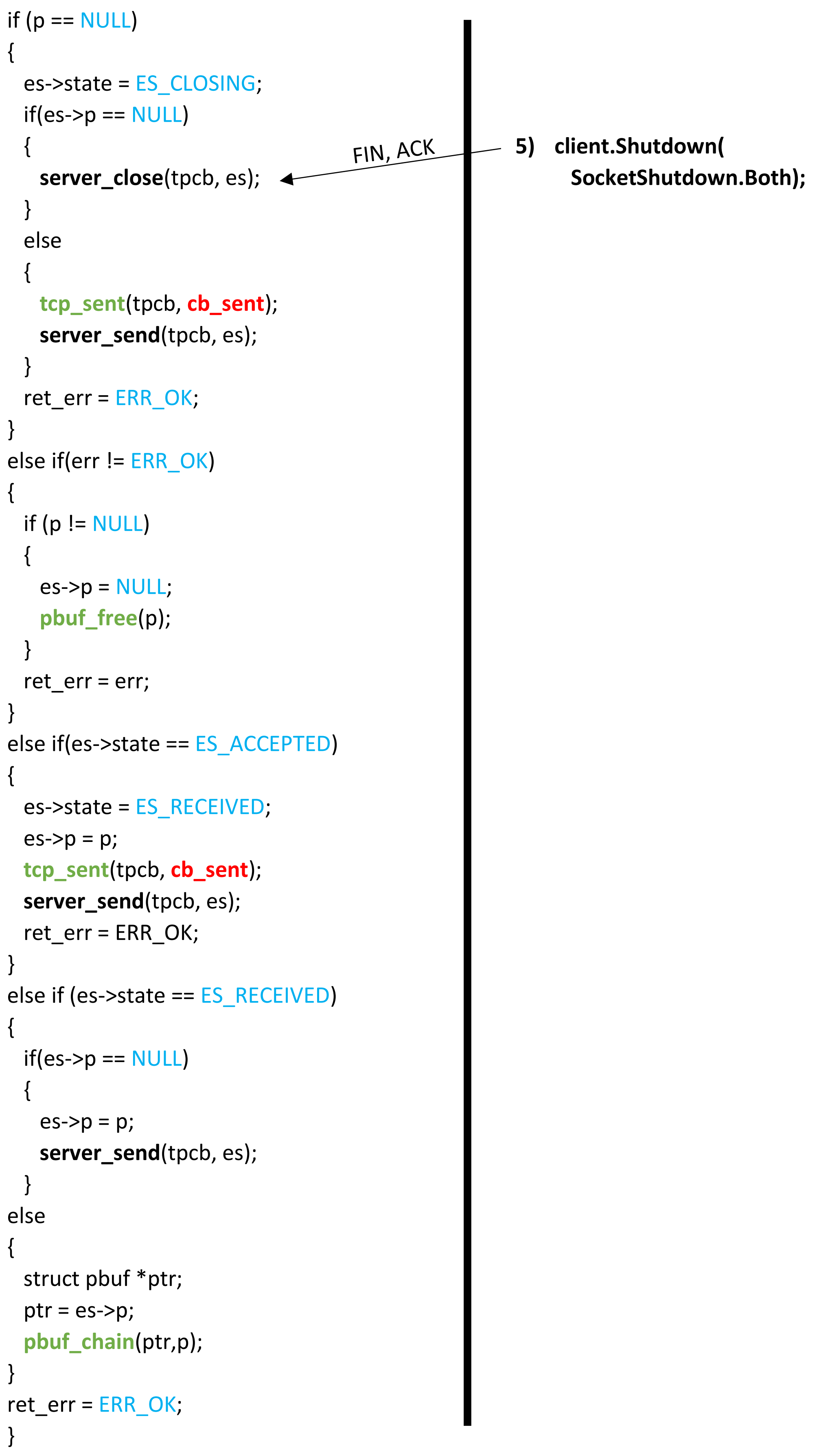




Net.Sockets Client (PC) C#

```
string IP = "192.168.100.1"
IPEndPoint ep = new IPEndPoint(IP, port);

1)  Socket client = new Socket(
                ep.AddressFamily
                SocketType.Stream
                ProtocolType.Tcp);

2)  client.Connect(ep);

byte [] msg = Encoding.ASCII.GetBytes(str)
3)  int sentCnt = client.Send()
```

ECHO-TCP server diagram created by vladimirsustek for personal education needs 14.3.2022

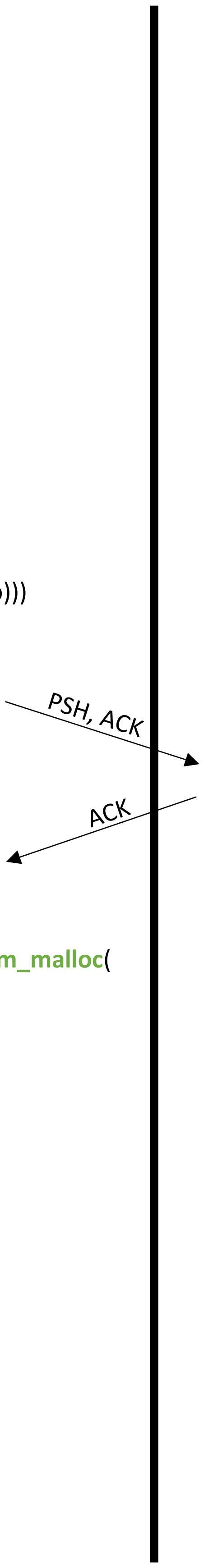


```
if (p == NULL)
{
  es->state = ES_CLOSING;
  if(es->p == NULL)
  {
    server_close(tpcb, es);
  }
  else
  {
    tcp_sent(tpcb, cb_sent);
    server_send(tpcb, es);
  }
  ret_err = ERR_OK;
}
else if(err != ERR_OK)
{
  if (p != NULL)
  {
    es->p = NULL;
    pbuf_free(p);
  }
  ret_err = err;
}
else if(es->state == ES_ACCEPTED)
{
  es->state = ES_RECEIVED;
  es->p = p;
  tcp_sent(tpcb, cb_sent);
  server_send(tpcb, es);
  ret_err = ERR_OK;
}
else if (es->state == ES_RECEIVED)
{
  if(es->p == NULL)
  {
    es->p = p;
    server_send(tpcb, es);
  }
else
{
  struct pbuf *ptr;
  ptr = es->p;
  pbuf_chain(ptr,p);
}
ret_err = ERR_OK;
}
```

**5) client.Shutdown(
SocketShutdown.Both);**

ECHO-TCP server diagram created by vladimirsustek for personal education needs 14.3.2022

```
else
{
  tcp_recved(tpcb, p->tot_len);
  es->p = NULL;
  pbuf_free(p);
  ret_err = ERR_OK;
}
return ret_err;
}

server_send(tpcb, es)
{
struct pbuf *ptr;
err_t wr_err = ERR_OK;

while ((wr_err == ERR_OK) &&
  (es->p != NULL) &&
  (es->p->len <= tcp_sndbuf(tpcb)))
{
  ptr = es->p;
  wr_err = tcp_write(tpcb,
    ptr->payload,
    ptr->len, 1);

  if (wr_err == ERR_OK)
  {
    u16_t plen;
    plen = ptr->len;
    char *pReceived = (char*)mem_malloc(
      (size_t)(plen + 1));

    es->p = ptr->next;
    if(es->p != NULL)
    {
      pbuf_ref(es->p);
    }
    pbuf_free(ptr);
    tcp_recved(tpcb, plen);
  }
  else if(wr_err == ERR_MEM)
  {
    es->p = ptr;
  }
  else
 {}
}

}
```



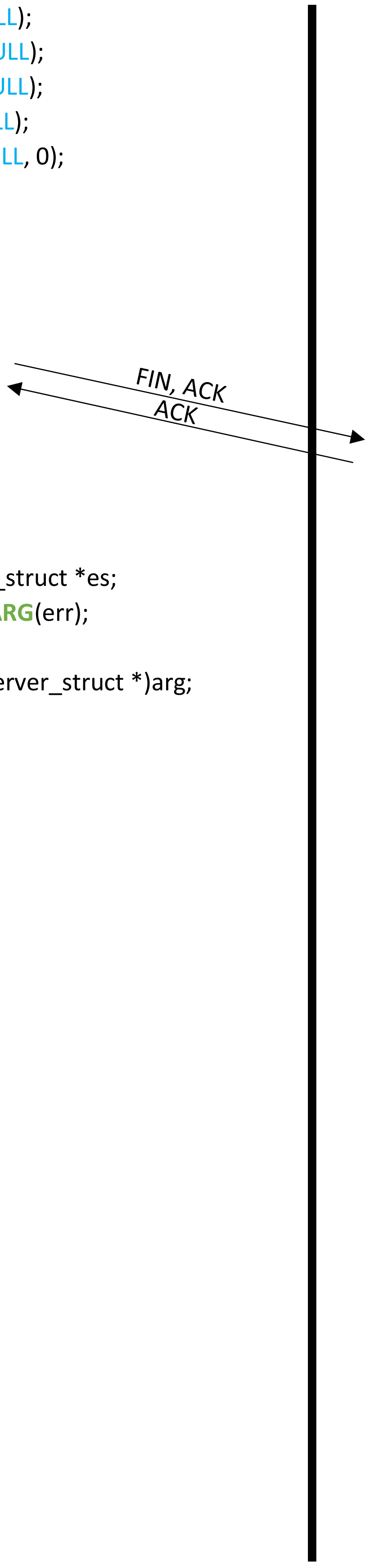
4) int recvdCnt = client.Receive(bytes);

ECHO-TCP server diagram created by vladimirsustek for personal education needs 14.3.2022

```
cb_sent(arg, tpcb, len)
{
struct tcp_server_struct *es;
LWIP_UNUSED_ARG(len);
es = (struct tcp_server_struct *)arg;

if(es->p != NULL)
{
  server_send(tpcb, es);
}
else
{
  if(es->state == ES_CLOSING)
    server_close(tpcb, es);
}

return ERR_OK;

}

server_poll(arg, tpcb)
{
err_t ret_err;
struct tcp_server_struct *es;
es = (struct tcp_server_struct *)arg;
if (es != NULL)
{
  if (es->p != NULL)
  {
    server_send(tpcb, es);
  }
  else
  {
    if(es->state == ES_CLOSING)
    {
      server _close(tpcb, es);
    }
  }
  ret_err = ERR_OK;
  }
else
{
  tcp_abort(tpcb);
  ret_err = ERR_ABRT;
}
return ret_err;
}
```

ECHO-TCP server diagram created by vladimirsustek for personal education needs 14.3.2022

```
server_close(tpcb, es)
{
tcp_arg(tpcb, NULL);
tcp_sent(tpcb, NULL);
tcp_recv(tpcb, NULL);
tcp_err(tpcb, NULL);
tcp_poll(tpcb, NULL, 0);

if (es != NULL)
{
  mem_free(es);
}
tcp_close(tpcb);
}
```



```
cb_error(arg, err)
{
struct tcp_server_struct *es;
LWIP_UNUSED_ARG(err);

es = (struct tcp_server_struct *)arg;

if (es != NULL)
{
  mem_free(es);
}
}
```