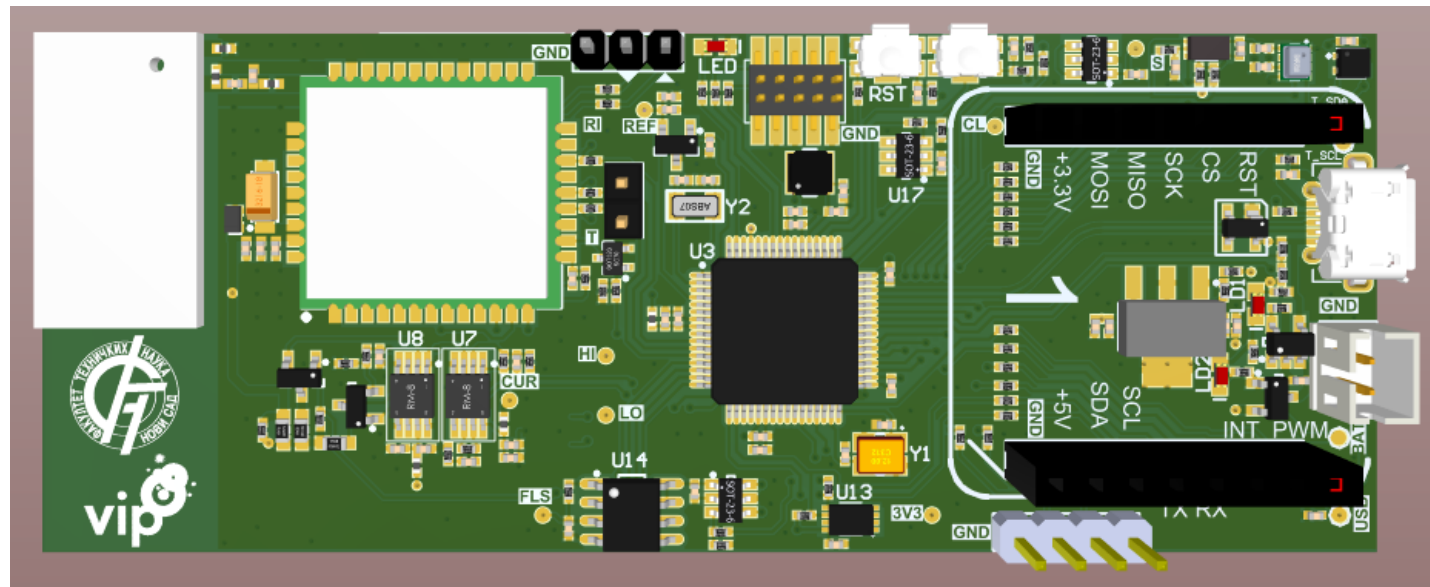
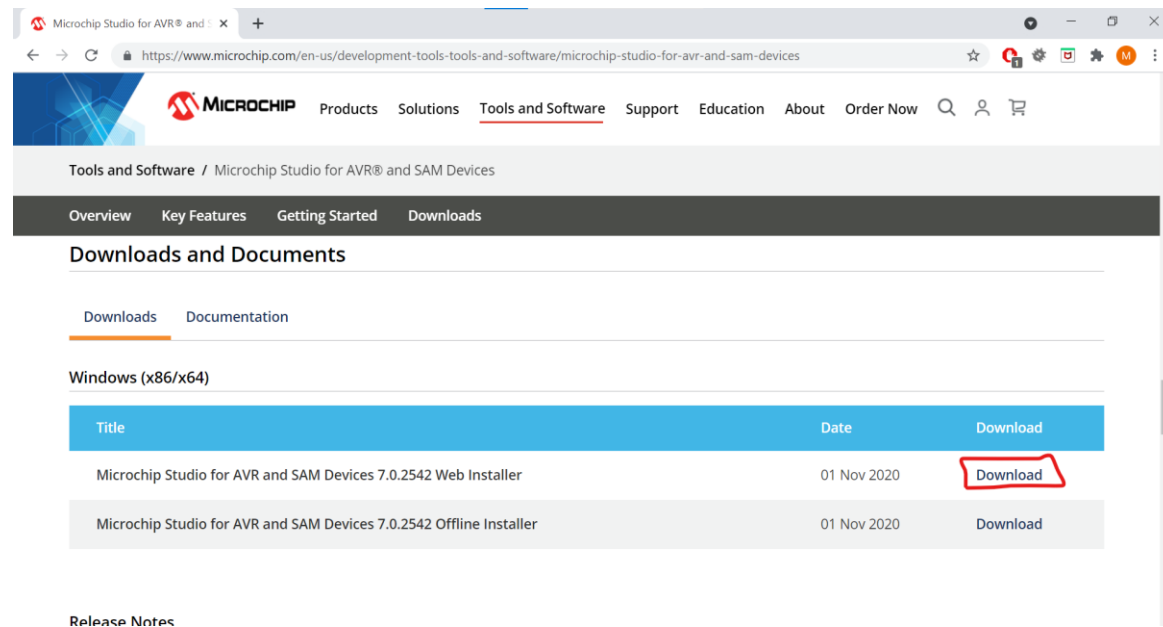


NB-IoT



Microchip Studio - instalacija

- <https://www.microchip.com/en-us/development-tools-tools-and-software/microchip-studio-for-avr-and-sam-devices>
- Prilikom instalacije neophodno je instalirati alate za rad sa ATSAM familijom mikrokontrolera



The screenshot shows a web browser window displaying the Microchip website. The page is titled "Microchip Studio for AVR® and SAM Devices" and is part of the "Tools and Software" section. The navigation menu includes "Products", "Solutions", "Tools and Software", "Support", "Education", "About", and "Order Now". The main content area is titled "Downloads and Documents" and has two tabs: "Downloads" (selected) and "Documentation". Under the "Downloads" tab, there is a section for "Windows (x86/x64)" which contains a table with two rows of download links. The first row is highlighted in blue and has a red box around the "Download" button. The second row is a standard grey row. Below the table, there is a "Release Notes" link.

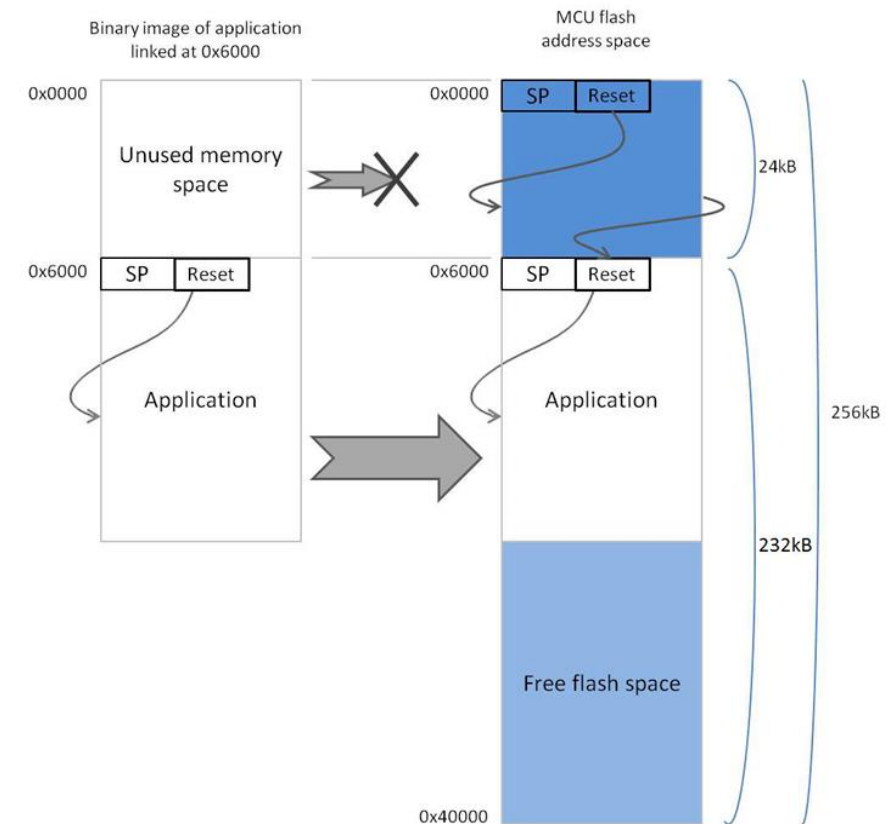
Title	Date	Download
Microchip Studio for AVR and SAM Devices 7.0.2542 Web Installer	01 Nov 2020	Download
Microchip Studio for AVR and SAM Devices 7.0.2542 Offline Installer	01 Nov 2020	Download

SAM-BA instalacija

- SAM-BA v2.18 je pomoćna aplikacija koja će biti korišćena za programiranje mikrokontrolera ATSAML21J18B
- Instalacija:
<https://ww1.microchip.com/downloads/en/DeviceDoc/SAM-BA%20v2.18%20for%20Windows.exe>

Pokretanje bootloader-a i aplikacije

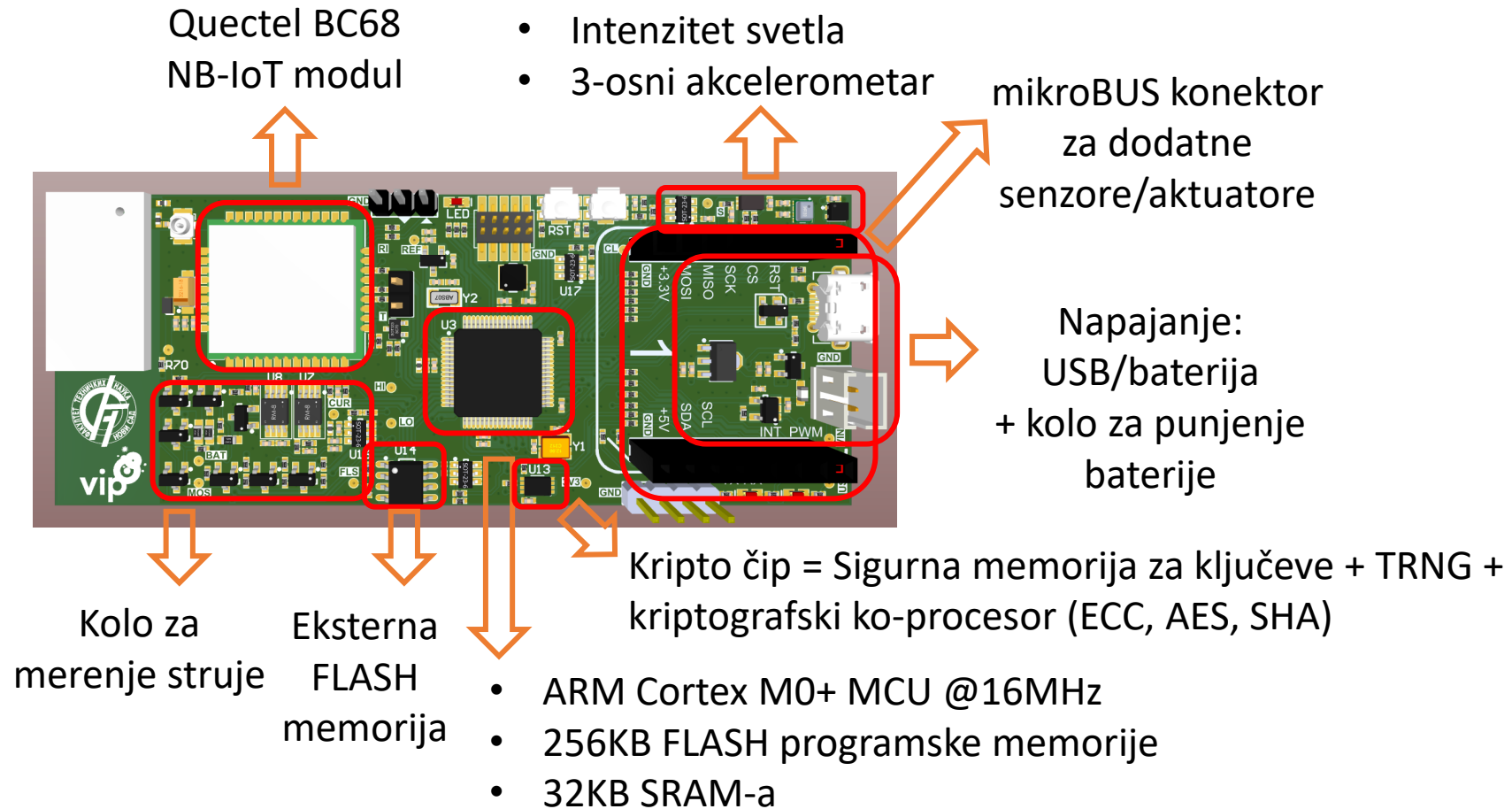
- Pritiskom RST tastera na ploči, uređaj se prebacuje u jedan od dva moguća režima:
 - Ukoliko je u trenutku pritiska RST prethodno bio pritisnut i drugi taster (PRG), uređaj se prebacuje u bootladerski režim i čeka na prijem programskog koda od strane SAM-BA aplikacije
 - Ukoliko je pritisnut samo RST taster, a PRG je u tom trenutku bio pušten, uređaj će startovati korisničku aplikaciju



FTN-VIP ploča: Hardver

Senzori:

- Temperatura + vlažnost vazduha
- Temperatura + vazdušni pritisak
- Intenzitet svetla
- 3-osni akcelerometar

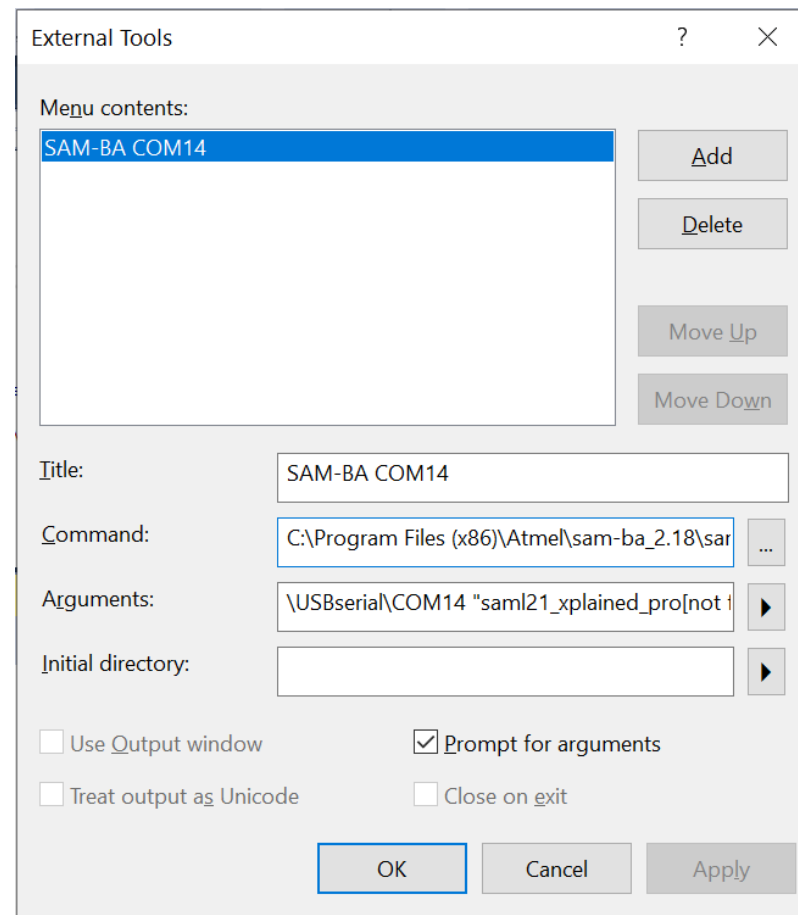


Tehnička dokumentacija

- Mikrokontroler ATSAML21J18B
<https://www.microchip.com/wwwproducts/en/ATSAML21J18B>
- Quectel BC68 set AT komandi
[http://www.elektronika.ftn.uns.ac.rs/umrezeni-embedded-sistemi/wp-content/uploads/sites/176/2018/03/Quectel BC95-GBC68 AT Commands Manual V1.5.pdf](http://www.elektronika.ftn.uns.ac.rs/umrezeni-embedded-sistemi/wp-content/uploads/sites/176/2018/03/Quectel_BC95-GBC68_AT_Commands_Manual_V1.5.pdf)
- Šema uređaja
<http://www.elektronika.ftn.uns.ac.rs/umrezeni-embedded-sistemi/wp-content/uploads/sites/176/2018/03/vip-ftn-bc68-nbiot.pdf>
- GitHub repozitorijum aplikacije
https://github.com/milukic/UES_NB-IoT.git

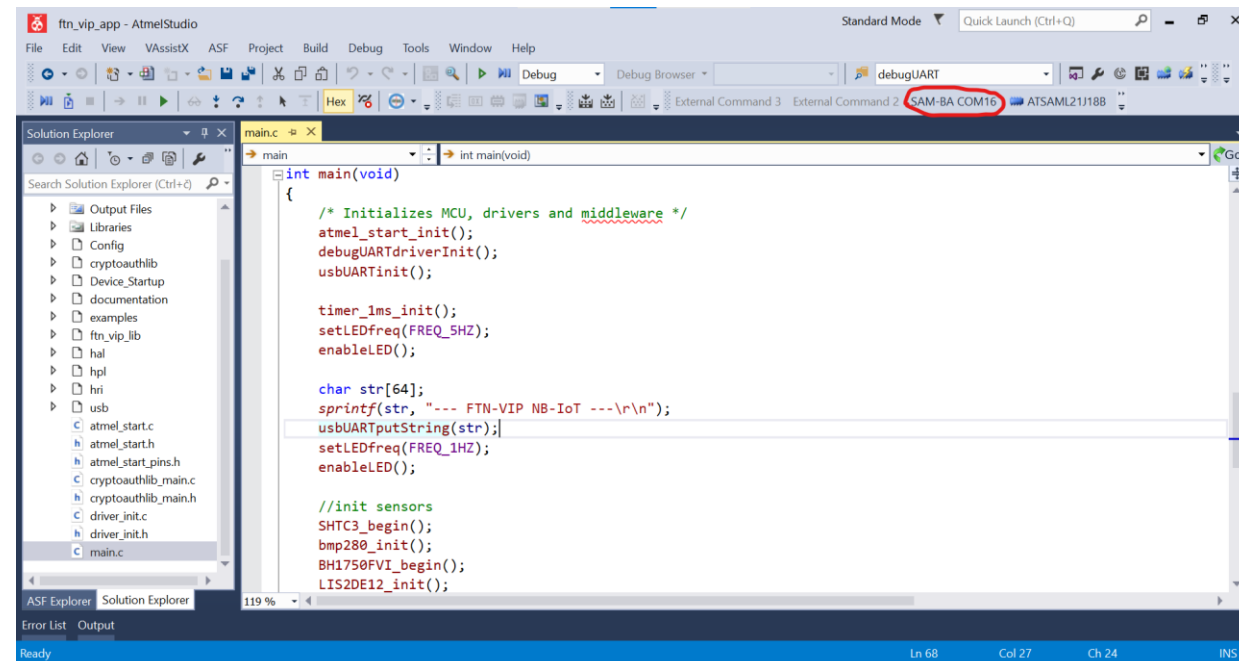
Podešavanje okruženja za rad sa SAM-BA

- U okruženju Microchip Studio izabrati opciju **Tools -> External Tools -> Add**
- U polju **Title** staviti naziv SAM-BA COMxx, gde je xx oznaka COM porta
- U polju **Command** zadati putanju do SAM-BA aplikacije (npr. *C:\Program Files (x86)\Atmel\sam-ba_2.18\samba.exe*)
- U polje **Arguments** upisati:
*\USBserial\COMxx
"sam121_xplained_pro[not factory programmed]"*



Prevođenje aplikacije

- Po otvaranju aplikacije (ftn_vip_app.atsln), razvoj koda se vrši iz okruženja Microchip Studio IDE (ili Atmel Studio, kao starijoj verziji)
- Nakon unošenja izmena u kodu, kreiranje izvršne verzije koda vrši se pomoću opcije **Build -> Build Solution**, ili pritiskom tastera **F7**
- Nakon što je kompajliranje i linkovanje uspešno izvršeno unutar foldera ftn_vip_app/Debug nalaziće se nova verzija izvršnog koda u datoteci **ftn_vip_app.bin**
- Sledeći korak je prebacivanje ploče u režim bootloadera i pokretanje aplikacije za programiranje (**Tools -> SAM-BA COMxx**)

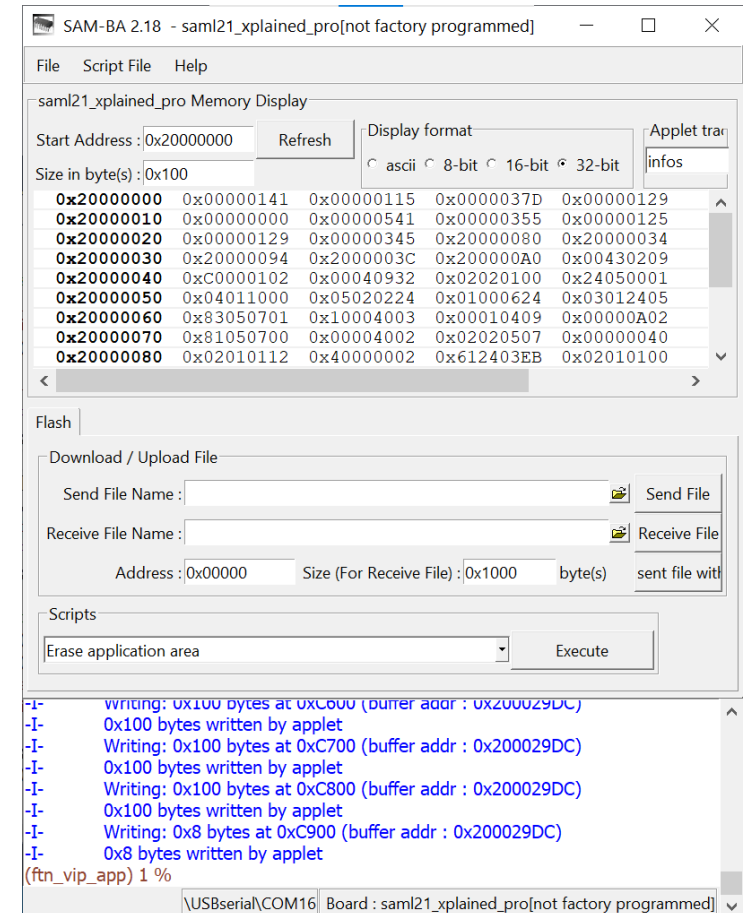


Upis aplikacije u FLASH memoriju mikrokontrolera

- Pokretanjem aplikacije SAM-BA v2.18, omogućen je upis programskog koda u flash memoriju kontrolera ATSAML21J18B
- U okviru datoteke **boot.tcl** smeštene u glavni folder aplikacije **ftn_vip_app**, potrebno je podesiti putanju do izvršne verzije koda, npr:

```
FLASH::Init  
FLASH::ScriptGPNMV 1  
send_file {Flash} "C:/ftn_vip_app/ftn_vip_app/Debug/ftn_vip_app.bin" 0x06000 0
```

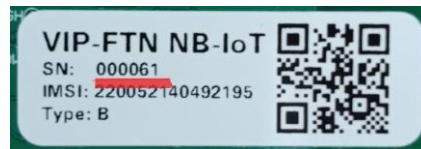
- Pokrenuti opciju **Script File -> Execute Script File**
- Izabrati skriptu **boot.tcl** smeštenu u glavni folder aplikacije **ftn_vip_app**
- Nakon spuštanja koda, aplikacija se pokreće pritiskom tastera RST



Eho server (UDP)

- Serverska skripta osluškuje dolazni UDP port i za svaki primljeni paket u tekstualnom formatu vraća kapitalizovanu verziju teksta (samo sa velikim slovima)
- Skriptu je neophodno snimiti na server u datoteku **udp_echo.py**
- Pre pokretanja skripte potrebno je u kodu podesiti port na vrednost 50000 + serijski broj pločice, ispisan na nalepnici sa donje strane

PRIMER:



port=50061

- Pokretanje servera:

```
$ python3 udp_echo.py
```

- Server se zaustavlja kombinacijom tastera **ctrl-z**, nakon čega je potrebno izvršiti komandu:

```
$ pkill -9 -f udp_echo.py
```

```
from socket import *
import time

port = 50061
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(('', port))
print ('Echo server is ready to receive (port ' + str(port) + ')\n')

msgCnt = 1
while True:
    try:
        messageIn, clientAddress = serverSocket.recvfrom(4096)

        ts = time.localtime()
        print('\033[0;34;40mEcho server (' + str(port) + ') Msg#', str(msgCnt))
        print(time.strftime('%Y-%m-%d %H:%M:%S\033[0;37;40m', ts))
        print('  Rx: ', messageIn)
        print('  Tx: ', messageIn.upper())
        serverSocket.sendto(messageIn.upper(), clientAddress)

        msgCnt += 1
    except:
        print('ERROR in Echo UDP')
```

Echo server (TCP)

```
import socket

HOST = ''                # Symbolic name meaning all available interfaces
PORT = 50061            # Arbitrary non-privileged port
NUM_OF_CLIENTS = 1

tcpSocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
tcpSocket.bind((HOST, PORT))
tcpSocket.listen(NUM_OF_CLIENTS)

print ('Echo server is ready to receive (port ' + str(PORT) + ')\n')

msgCnt = 1
while True:
    try:
        conn, addr = tcpSocket.accept()
        print ('Connected by', addr)
        while True:
            data_in = conn.recv(1024)
            if not data_in:
                break
            data_out = data_in.upper()
            print ('    Rx: ', data_in)
            print ('    Tx: ', data_out)
            conn.sendall(data_out)
    except:
        conn.close()
        print('ERROR in Echo TCP')
```

Packet Sender – pomoćni alat za slanje paketa

- https://github.com/dannagle/PacketSender/releases/download/v7.2.3/PacketSenderPortable_v7.2.3.zip



```
UES student.tlp - test.e01@199.247.17.15:...
test.e01@UES_server:~$ python3 udp_echo.py
Echo server is ready to receive (port 50000)

Echo server (50000) Msg# 1
2021-06-07 23:20:54
  Rx: b'Hello world!'
  Tx: b'HELLO WORLD!'
```

Packet Sender - IPs: 192.168.0.40, fe80::5c76:8e76:2c3e:2c7%wireless_32768

File Tools Multicast Help

Name Packet Name

ASCII Hello world!

HEX 48 65 6c 6c 6f 20 77 6f 72 6c 64 21

Address 199.247.17.15 Port 50000 Resend Delay 0.0/bl... UDP Send Save

Search Saved Packets... Delete Saved Packet Persistent TCP

Send	Name	Resend	To Address	To Port	Method	
Send	Telnet RPG	0	avalon-rpg.com	23	TCP	
Send	UDP Broadcast	0	255.255.255.255	5000	UDP	Hello, broadcas
Send	UDP IPv4 localhost mapp	0	127.0.0.1	5000	UDP	((TIME)) ((RAND

Clear Log (2) Log Traffic Save Log Save Traffic Packet Copy to Clipboard

Time	From IP	From Port	To Address	To Port	Method	Error	ASCII
23:20:54.950	199.24...	50000	You	55625	UDP		HELLO WORLD! 48 45 4C 4C 4
23:20:54.849	You	55625	199.247.17.15	50000	UDP		Hello world! 48 65 6c 6c 6f

UDP:55625 TCP:62845 SSL:62846 IPv4 Mode

Primer: Upis sadržaja UDP paketa u bazu

- U ovom primeru, sadržaj UDP paketa (očitanja senzora u JSON formatu) upisuje se u bazu podataka, u postojeću tabelu **merenja**

PRIMER:

```
{
  "temp": {
    "value": 30.8
  },
  "pres": {
    "value": 1004.56
  },
  "hum": {
    "value": 34.83
  },
  "lum": {
    "value": 33
  }
}
```

The screenshot shows the Packet Sender application window. The configuration for the 'UES server' is as follows:

- Name: UES server
- ASCII: {"temp":{"value":30.8},"pres":{"value":1004.56},"hum":{"value":34.83},"lum":{"value":33}}
- Address: 199.247.17.15
- Port: 50061
- Resend Delay: 0.0/blank
- Method: UDP

The log shows the following traffic:

Time	From IP	From Port	To Address	To Port	Method	Error
10:23:38.339	199.24...	50061	You	50295	UDP	OK!
10:23:38.223	You	50295	199.247.17.15	50061	UDP	{"temp":{"value":30.8},"pres":{"va

```
test.e01@UES_server:~$ python3 udp_senzor.py
UDP server (port 50061)

UDP server (50061) Poruka# 1
2021-06-09 10:14:33
Temperatura:      30.8 °C
Pritisak:         1004.56 mBar
Vlaznost vazduha: 34.83 %
Osvetljenost:    33 lux
```

The screenshot shows a database query result for the 'merenja' table. The query is: `SELECT * FROM 'merenja'`. The result shows 4 rows of data:

br_merenja	vreme	tip_senzora	vrednost
1	2021-06-09 08:14:33	temperatura	30.8
2	2021-06-09 08:14:33	pritisak	1004.56
3	2021-06-09 08:14:33	vlaznost	34.83
4	2021-06-09 08:14:33	osvetljenost	33

```

from socket import *
import time
import mysql.connector
import json

mydb = mysql.connector.connect(
    host = "localhost",
    user = "test.e01",
    password = "...",
    database = "db_test_e01"
)

port = 50061
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(('', port))
print ('UDP server (port ' + str(port) + ')\n')

msgCnt = 1
while True:
    try:
        messageIn, clientAddress = serverSocket.recvfrom(4096)

        ts = time.localtime()
        print('\033[0;34;40mUDP server (' + str(port) + ') Poruka#', str(msgCnt))
        print(time.strftime('%Y-%m-%d %H:%M:%S\033[0;37;40m', ts))

        data = json.loads(messageIn.decode("utf-8"))
        print ("  Temperatura:      ", data["temp"]["value"], "\u00b0C")
        print ("  Pritisak:           ", data["pres"]["value"], "mBar")
        print ("  Vlaznost vazduha:    ", data["hum"]["value"], "%")
        print ("  Osvetljenost:       ", data["lum"]["value"], "lux")

        mycursor = mydb.cursor()
        mycursor.execute("INSERT INTO merenja (tip_senzora, vrednost) VALUES ('temperatura'," + str(data["temp"]["value"]) + ")")
        mycursor.execute("INSERT INTO merenja (tip_senzora, vrednost) VALUES ('pritisak'," + str(data["pres"]["value"]) + ")")
        mycursor.execute("INSERT INTO merenja (tip_senzora, vrednost) VALUES ('vlaznost'," + str(data["hum"]["value"]) + ")")
        mycursor.execute("INSERT INTO merenja (tip_senzora, vrednost) VALUES ('osvetljenost'," + str(data["lum"]["value"]) + ")")
        mydb.commit()

        serverSocket.sendto(bytearray("OK!", "utf-8"), clientAddress)

        msgCnt += 1
    except:
        print('Greska!!!')

```

Source kodovi

- Serverske skripte (Python):

https://github.com/milukic/UES_Python.git

- FW za FTN-VIP ploču:

https://github.com/milukic/UES_NB-IoT.git

Zadaci za vežbu

- Modifikovati FW za VIP-FTN ploču tako da na pritisak tastera šalje očitavanja senzora
- Kreirati web stranicu koja grafički prikazuje pristigle rezultate očitavanja senzora
- **Nešto teži zadatak:** Na web serveru napraviti u HTML-u jednostavnu “Hello world” stranicu. Zatim u FW za VIP-FTN ploču preuzeti sadržaj web stranice. U tu svrhu, potrebno je realizovati aplikativni sloj HTTP protokola, koji koristi TCP konekciju na portu 80.