

IOT ÉS FELHŐ ALAPÚ ADATMENEDZSELÉS, MŰSZAKI INFORMATIKA

Dr. Békési József, Dr. Kelemen András

SZÉCHENYI 2020



MAGYARORSZÁG
KORMÁNYA

Európai Unió
Európai Szociális
Alap



BEFEKTETÉS A JÖVŐBE

BEVEZETÉS AZ IOT FEJLESZTÉSEKBE

AZ IOT VILÁGA

- **Mit jelent az Internet of Things (IoT)**

IoT is a network of physical objects with embedded electronics that collect and share data

Has been around for quite a while, but only recently has become affordable for personal use

- **Egymással összekapcsolt rendszerek**

- **2020-ra több mint 26 milliárd okos eszköz lesz forgalomban**

- **Jellemzők:**

Ultra Low Power Sensor

Low Power Network Technology

Wireless Connectivity

Autonomous Field Device

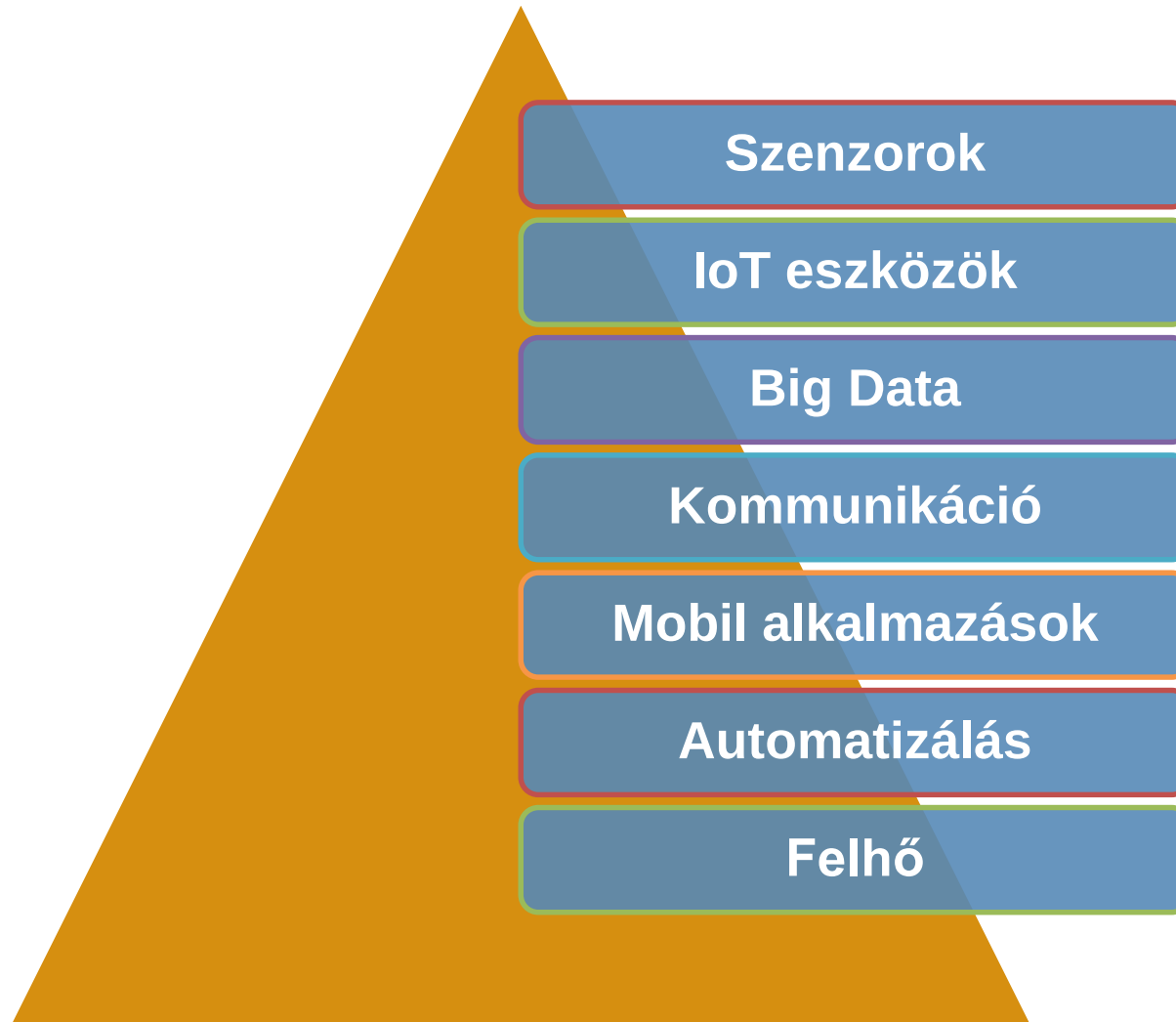
Open, Industry Standard Communication Protocol

Multilayer Encryption

Data Sharing

BEVEZETÉS AZ IOT FEJLESZTÉSEKBE

AZ IOT VILÁGA



BEVEZETÉS AZ IOT FEJLESZTÉSEKBE

TRENDEK AZ IOT VILÁGÁBAN

- Az IoT megjelenése az okos városokban, intelligens otthonokban és az iparban
- Mik is azok az okos eszközök
- Az Okos Készülékek fogalma
- Önvezető autó



BEVEZETÉS AZ IOT FEJLESZTÉSEKBE

TRENDEK AZ IOT VILÁGÁBAN

- Okos otthon, okos zár, okos termosztát
- Okosóra, Okos zokni
- Okos ágy, okos ágynemű



BEVEZETÉS AZ IOT FEJLESZTÉSEKBE

IOT ESZKÖZÖK

- Eszközök

- Szenzorok

- Kommunikációs kihívások

GSM GPRS is „expensive”

Power output 2 Watts, (requires super caps)

Average consumption in milliamps

Huge battery capacity (70/140Ah)

Communication for Installation

Local display or server side web-based app

Needs power and other resources



NB-LTE



BEVEZETÉS AZ IOT FEJLESZTÉSEKBE

IOT ESZKÖZÖK

- Kommunikációs kihívások

Needs power and other resources

Suitable Communication Technology

Power output in milliwatts range

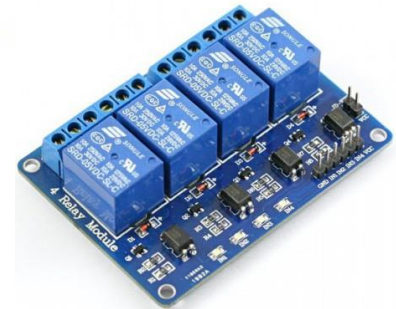
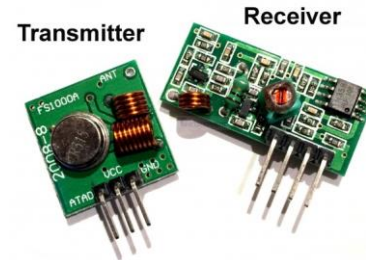
Average consumption in microamps range

Low Energy Communication

BLE 4.x/5

Low power, long range

Tablet or Smart Phone



BEVEZETÉS AZ IOT FEJLESZTÉSEKBE

ESP8266 BEMUTATÁSA

- **ESP8266 mint IoT eszköz**

Espressif Systems által gyártott beágyazott eszközök amely fejlesztők körében egyik legnépszerűbb vezeték nélküli mikrokontroller

- **CPU**

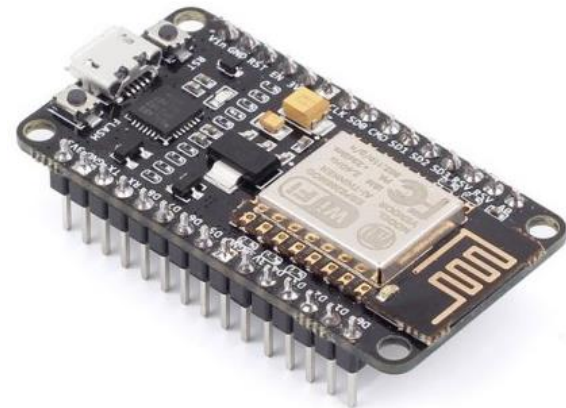
32 bit, 26MHz-52MHz,
64KB instruction RAM,
64KB boot ROM, 96KB data RAM

- **WiFi**

802.11b/g/n, Access Point or Station,
WEP

- **Other**

GPIO, UART, ADC, I2C, SPI, PWM



BEVEZETÉS AZ IOT FEJLESZTÉSEKBE

ESP8266 NODEMCU

- **NodeMCU - Lua**

NodeMCU is an open source IoT platform. It includes firmware which runs on the ESP8266.

- **Elérhető könyvtárak**

File

Wifi – Station and AP

GPIO, PWM, A2D

Net – Server and Socket

I2C, UART, OneWire, SPI

MQTT, COAP

WS2811

- **Egyszerű programozás**

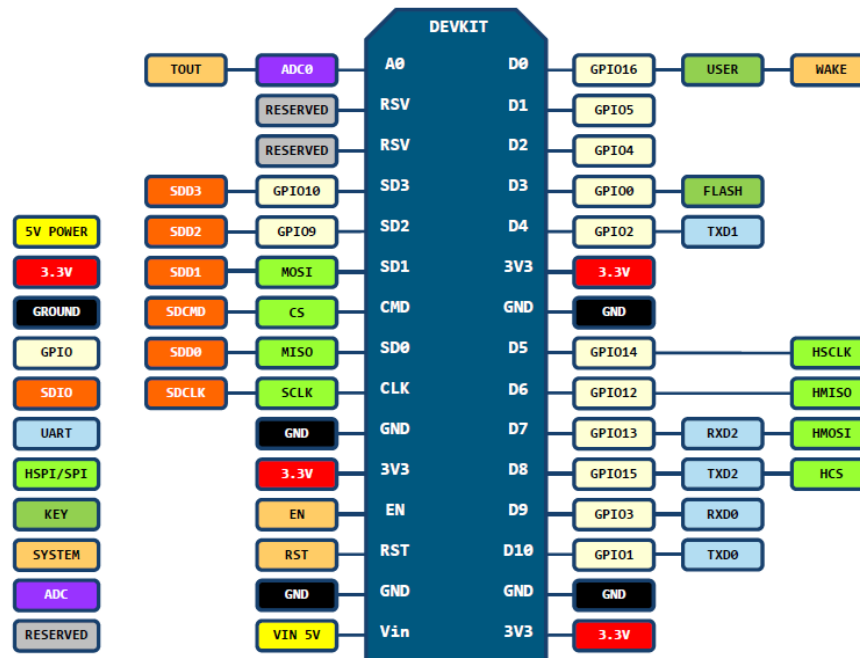
```
wifi.setmode(wifi.STATION)
```

```
wifi.sta.config("ssid", "password")
```

BEVEZETÉS AZ IOT FEJLESZTÉSEKBE

ESP8266 NODEMCU

PIN DEFINITION



BEVEZETÉS AZ IOT FEJLESZTÉSEKBE

ARDUNIO IDE BEMUTATÁS

- **Arduino IDE**

egyszerű syntax highlighting, alap tudás

- **Visual Micro**

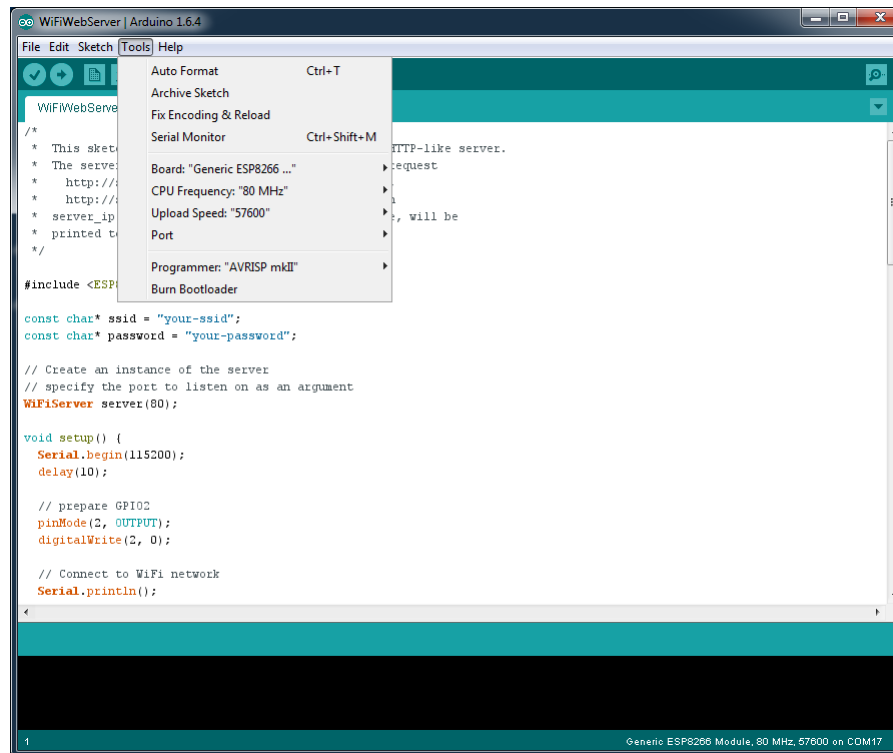
Visual Studio plugin, pro verzió fizetős, Arduino kompatibilis

- **Atmel Studio**

Visual Studio alapú, hivatalos, ingyenes

BEVEZETÉS AZ IOT FEJLESZTÉSEKBE

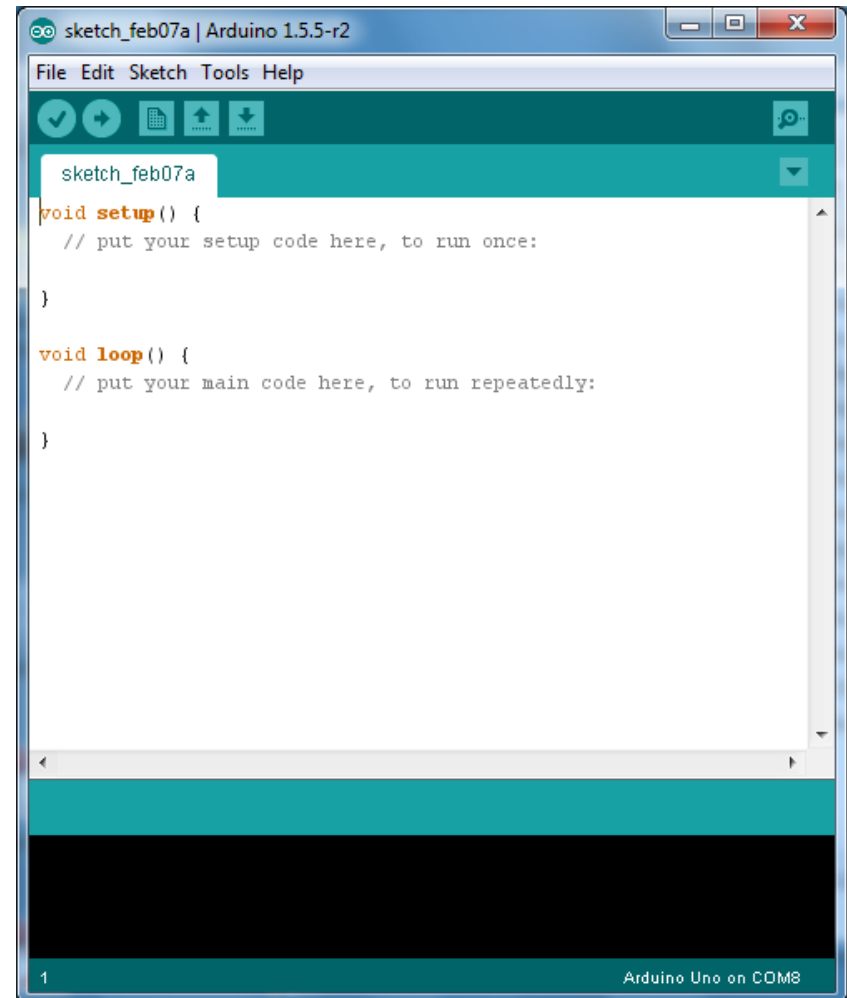
ARDUNIO IDE BEMUTATÁS



BEVEZETÉS AZ IOT FEJLESZTÉSEKBE

ARDUNIO IDE FORDÍTÁS, FELTÖLTÉS

- **Egy program felépítése**
Sketch két fő [void] típusú funkcióhívásból áll, egy setup() és egy loop() blokkból. A [void] jellemzője, hogy nem ad vissza semmilyen értéket, csupán deklarál
- **A Sketch programozási nyelve C++.**
- **Fordítás**
csak teszt, szintaktikai ellenőrzés
- **Feltöltés**



BEVEZETÉS AZ IOT FEJLESZTÉSEKBE

ARDUNIO IDE ELSŐ LÉPÉSEK

- **Soros port osztály**

Inicializásása a `Serial.begin(baudrate)` függvénnyel történik.

Ezt a számot kell a serialMonitor-ban beállítani

`void Serial.print(adat)`: kiírás a portra. Az adat lehet int, char, double és string is.

`Void Serial.println(adat)`: mint `print()`, csak egy soremeléssel lezárva a kiírást.

BEVEZETÉS AZ IOT FEJLESZTÉSEKBE

ARDUNIO IDE ELSŐ LÉPÉSEK

- **Alap műveletek**

`void pinMode(lábszám,üzemmód):` a digitális lábat állítja adott üzemmódra.

Üzemmódok: INPUT, OUTPUT, INPUT_PULLUP. Az analóg input lábak (A0,A1 stb) is használhatóak.

`void digitalWrite(lábszám,adat):` a digitális lábra kimenetet kapcsol. Az adat HIGH vagy LOW lehet, ezek csak definiált konstansok (HIGH=1, LOW=0).

`int digitalRead(lábszám):` beolvas egy digitális bemenet. Az eredmény 0 vagy 1 lesz.

`int analogRead(lábnév):` beolvas egy analóg bemenetet. 10 bites átalakító, alapesetben 5V referencia.

5V=1023

`void analogWrite(lábszám, érték):` pwm képes kimenet (3,5,6,9,10,11) kitöltési tényezőjének állítása 0..255 intervallumban.

`void delay(ezredmásodperc):` adott idejű várakozás.

BEVEZETÉS AZ IOT FEJLESZTÉSEKBE

ARDUNIO IDE DEBUG

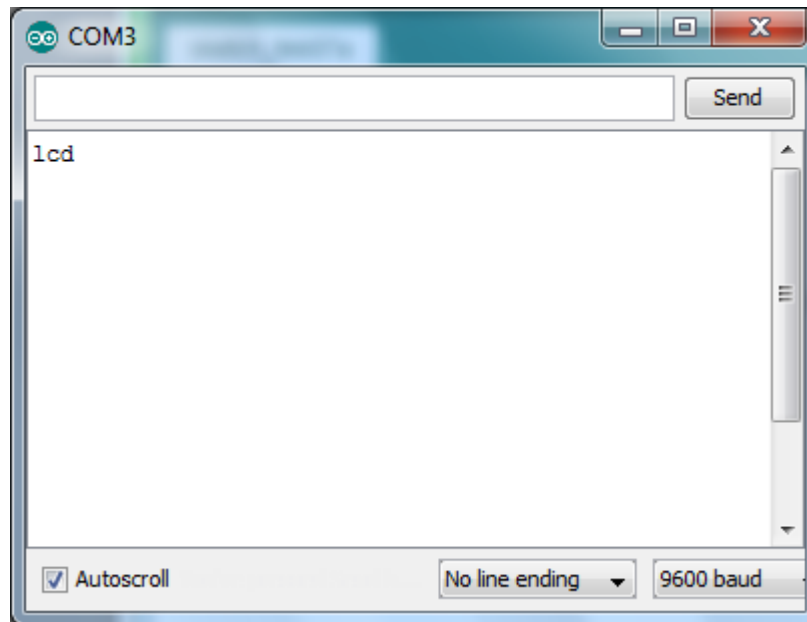
- **Soros monitor**

Ha már kész a feltöltés, a soros port a PC-vel való kommunikációra használható. Feltöltéskor a terminál ablak bezáródik.

```
void loop() {  
  Serial.print("c=");  
  Serial.println(c);  
  delay(1000);  
}
```

BEVEZETÉS AZ IOT FEJLESZTÉSEKBE

ARDUNIO IDE DEBUG



BEVEZETÉS AZ IOT FEJLESZTÉSEKBE

ESP8266 WIFI KÖNYVTÁR

- **#include <ESP8266WiFi.h>**
 - WiFi Class
 - Client Class
 - Server Class
 - IP Address Class
 - UDP Class
- **WiFi class allows to establish a connection with an access point**
 - WiFi.begin(ssid,pass);
 - WiFi.SSID();
 - WiFi.printDiag(Serial);
 - WiFi.macAddress()
 - WiFi.scanNetworks()

BEVEZETÉS AZ IOT FEJLESZTÉSEKBE

FELADAT: WIFI KAPCSOLAT MEGVALÓSÍTÁSA

```
#include <ESP8266WiFi.h>

void setup()
{
  Serial.println(WiFi.scanNetworks());
  delay(500);

  int n = WiFi.scanNetworks();
  for (int i = 0; i < n; i++)
  {
    Serial.println(WiFi.SSID(i));
  }
  Serial.println();

  WiFi.begin("MySSID", "mypass123");

  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }
  Serial.println();
}

void loop()
{
}
```

ESP VÉGPONTI ESZKÖZ PROGRAMOZÁSA

GPIO ALAPOK

- **Digital I/O**

Just like a normal Arduino, the ESP8266 has digital input/output pins (I/O or GPIO, General Purpose Input/Output pins). As the name implies, they can be used as digital inputs to read a digital voltage, or as digital outputs to output either 0V (sink current) or 3.3V (source current).

- **Voltage and current restrictions**

The ESP8266 is a 3.3V microcontroller, so its I/O operates at 3.3V as well. The pins are not 5V tolerant, applying more than 3.6V on any pin will kill the chip. The maximum current that can be drawn from a single GPIO pin is 12mA.

ESP VÉGPONTI ESZKÖZ PROGRAMOZÁSA

GPIO ALAPOK

- **Usable pins**

The ESP8266 has 17 GPIO pins (0-16), however, you can only use 11 of them, because 6 pins (GPIO 6 - 11) are used to connect the flash memory chip. This is the small 8-legged chip right next to the ESP8266. If you try to use one of these pins, you might crash your program.

GPIO 1 and 3 are used as TX and RX of the hardware Serial port (UART), so in most cases, you can't use them as normal I/O while sending/receiving serial data.

ESP VÉGPONTI ESZKÖZ PROGRAMOZÁSA

GPIO PORTOK

- GPIO overview

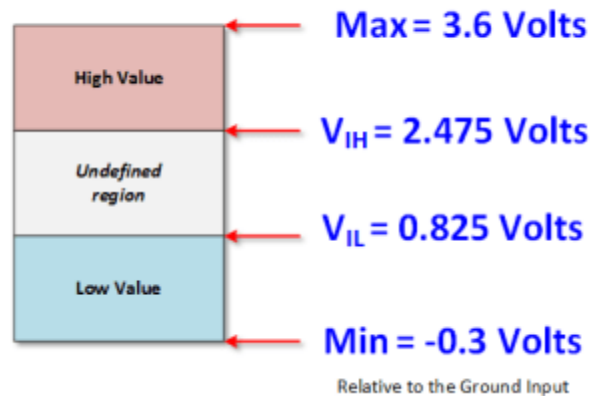
GPIO	Function	State	Restrictions
0	Boot mode select	3.3V	No Hi-Z
1	TX0	-	Not usable during Serial transmission
2	Boot mode select TX1	3.3V (boot only)	Don't connect to ground at boot time Sends debug data at boot time
3	RX0	-	Not usable during Serial transmission
4	SDA (I ² C)	-	-
5	SCL (I ² C)	-	-
6 - 11	Flash connection	x	Not usable, and not broken out
12	MISO (SPI)	-	-
13	MOSI (SPI)	-	-
14	SCK (SPI)	-	-
15	SS (SPI)	0V	Pull-up resistor not usable
16	Wake up from sleep	-	No pull-up resistor, but pull-down instead Should be connected to RST to wake up

ESP VÉGPONTI ESZKÖZ PROGRAMOZÁSA

GPIO HASZNÁLAT

- High and Low Inputs

Assume a 3.3 Volt Digital IO Supply Voltage



ESP VÉGPONTI ESZKÖZ PROGRAMOZÁSA

GPIO HASZNÁLAT

- `pinMode` Command

The GPIO Pin Number

pinMode(pinNum, nMode);

The Mode

INPUT
OUTPUT
INPUT_PULLUP
INPUT_PULLDOWN

ESP VÉGPONTI ESZKÖZ PROGRAMOZÁSA

GPIO ÍRÁS/OLVASÁS

- **Input is read from switch pin** If switch is closed, it will read a **low**

```
int switchPin = 0;
```

```
int LedPin = 16;
```

```
int switchValue;
```

```
void setup() {
```

```
    pinMode(LedPin, OUTPUT);
```

```
    pinMode(switchPin, INPUT_PULLUP);
```

```
}
```

ESP VÉGPONTI ESZKÖZ PROGRAMOZÁSA

GPIO ÍRÁS/OLVASÁS

```
void loop() {  
    // Read the switch value  
    switchValue = digitalRead(switchPin);  
    /* Set the LED output pin the opposit of what is read on the switch  
     * input pin  
     */  
    digitalWrite(LedPin, !switchValue);  
  
}
```

ESP VÉGPONTI ESZKÖZ PROGRAMOZÁSA

LED VILLOGTATÁS

- **Beépített LED villogtatása**

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}
```

ESP VÉGPONTI ESZKÖZ PROGRAMOZÁSA

LED VILLOGTATÁS

```
// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage
level)
  delay(1000);                      // wait for a second
  digitalWrite(LED_BUILTIN, LOW);  // turn the LED off by making the
voltage LOW
  delay(1000);                      // wait for a second
}
```


ESP VÉGPONTI ESZKÖZ PROGRAMOZÁSA

WEB SZERVER KÖNYVTÁR

- `#include <ESP8266WebServer.h>`

```
ESP8266WebServer server(80);
```

- **Szerver „on” parancs**

```
server.on("/", handleRootPath);
```

ESP VÉGPONTI ESZKÖZ PROGRAMOZÁSA

WEB SZERVER KÖNYVTÁR

- **Szerver „send” parancs**

```
server.send(200, "text/plain", "Hello world");
```

- **Szerver „handle” parancs**

```
server.handleClient();
```

ESP VÉGPONTI ESZKÖZ PROGRAMOZÁSA

WEB SZERVER INDÍTÁSA

Szerver indítása

```
#include "ESP8266WiFi.h"
#include "ESP8266WebServer.h"

ESP8266WebServer server(80);

void setup() {

  Serial.begin(115200);
  WiFi.begin("Network name", "Password"); //Connect to the WiFi network

  while (WiFi.status() != WL_CONNECTED) { //Wait for connection
    delay(500); }
}
```

ESP VÉGPONTI ESZKÖZ PROGRAMOZÁSA

WEB SZERVER INDÍTÁSA

```
Serial.print("IP address: ");  
Serial.println(WiFi.localIP()); //Print the local IP  
  
server.on("/other", []() { //Define the handling function for the path  
    server.send(200, "text / plain", "Other URL"); });  
  
server.on("/", handleRootPath); //Associate the handler function to the path  
server.begin(); //Start the server  
Serial.println("Server listening");  
}
```

ESP VÉGPONTI ESZKÖZ PROGRAMOZÁSA

WEB SZERVER RESPONSE

- `server.onNotFound(handleNotFound);`

```
void handleNotFound()
{
    server.sendHeader("Location", "/",true); //Redirect to our html web page
    server.send(302, "text/plain","");
}
```

- `server.on`

```
server.on("/", []() {
    server.send(200, "text/html", HTMLpage);
});
```

ESP VÉGPONTI ESZKÖZ PROGRAMOZÁSA

WEB SZERVER REQUEST

- `server.on` folytatva

```
server.on("/", []() {  
    server.send(200, "text/html", HTMLpage);  
});  
server.on("/data", []() {  
    server.send(200, "text/html", HTMLpage + getdhtdata());  
});  
server.on("/ledON", []() {  
    server.send(200, "text/html", HTMLpage + "<p>LED is ON</p>");  
});
```

ESP VÉGPONTI ESZKÖZ PROGRAMOZÁSA

WEB SZERVER MŰVELETEK

- **Teljes folyamat**

```
server.on("/", []() {  
    server.send(200, "text/html", HTMLpage);  
});  
server.on("/data", []() {  
    server.send(200, "text/html", HTMLpage + getdhtdata());  
});  
server.on("/ledON", []() {  
    server.send(200, "text/html", HTMLpage + "<p>LED is ON</p>");  
    Serial.println("ledON");  
    digitalWrite(TEST_LED, HIGH);  
    delay(1000);  
});  
server.on("/ledOFF", []() {  
    server.send(200, "text/html", HTMLpage + "<p>LED is OFF</p>");  
    Serial.println("ledOFF");  
    digitalWrite(TEST_LED, LOW);  
    delay(1000);  
});
```

ESP VÉGPONTI ESZKÖZ PROGRAMOZÁSA

FELADAT: LED KAPCSOLÁS BÖNGÉSZŐBŐL

```
void setup() {  
  
    Serial.begin(115200);  
    WiFi.begin("Network name", "Password"); //Connect to the WiFi  
network  
  
    while (WiFi.status() != WL_CONNECTED) { //Wait for connection  
        delay(500); }  
  
    Serial.print("IP address: ");  
    Serial.println(WiFi.localIP()); //Print the local IP  
  
    server.on("/", handleRootPath); //Associate the handler function to  
the path
```


ESP VÉGPONTI ESZKÖZ PROGRAMOZÁSA

FELADAT: LED KAPCSOLÁS BÖNГЭСZÖBÖL

```
server.on("/ledON", []() {  
  server.send(200, "text/html", HTMLpage + "<p>LED is ON</p>");  
  Serial.println("ledON");  
  digitalWrite(TEST_LED, HIGH);  
  delay(1000);  
});  
server.on("/ledOFF", []() {  
  server.send(200, "text/html", HTMLpage + "<p>LED is OFF</p>");  
  Serial.println("ledOFF");  
  digitalWrite(TEST_LED, LOW);  
  delay(1000);  
});  
  
server.begin();           //Start the server  
Serial.println("Server listening");  
}
```

LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS RASPBERRY PI MINT KÖZPONT

- **Raspberry Pi 3 Model B főbb műszaki paraméterei**

CPU: Quad Core 1.2GHz BCM2837 – 64bit

RAM: 1GB SDRAM

GPU: Broadcom VideoCore 4

Hálózat: 10/100Mbit/s +wifi (802.11 b/g/n, 2,4 GHz)

BCM43143 chip wifi és bluetooth Low Energy

Bluetooth 4.1 Classic, Bluetooth Low Energy

40pin-es kiterjesztett GPIO



LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS RASPBERRY PI MINT KÖZPONT

4 x USB 2.0 port

4 pólusú jack a sztereó hang és kompozit videó kimenethez

HDMI kimenet

CSI kamera port csatlakozó a Raspberry Pi kamera részére

DSI Display Port csatlakozó érintőképernyőhöz

Push-pull rendszerű micro SD kártya csatlakozó

Micro USB-s tápcsatlakozás maximum 2.5A 5V, azaz nagyobb táp használata tilos!

Mechanikai kompatibilitás a B+ és Pi 2 verziókkal

LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS

NODE-RED FOLYAMAT VEZÉRLŐ

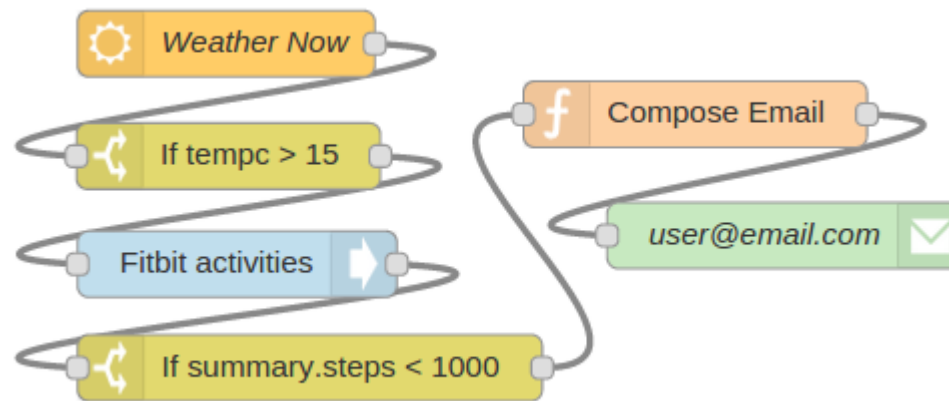
- **Hogyan épül fel a Node-Red?**

A Node-RED „folyamatokból” és „csomópontokból” áll. A csomópontok olyan folyamatok, amelyeket korábban már kialakítottak a Node-RED valamelyik előre definiált funkciójával.

- **Csomópontok**
input, output, funkció, social, storage

LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS

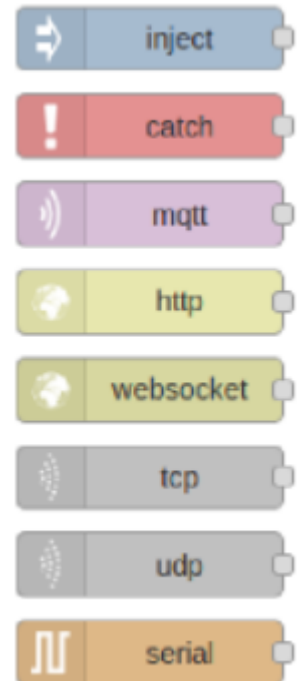
NODE-RED FOLYAMAT VEZÉRLŐ



LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS

NODE-RED INPUT CSOMÓPONTOK

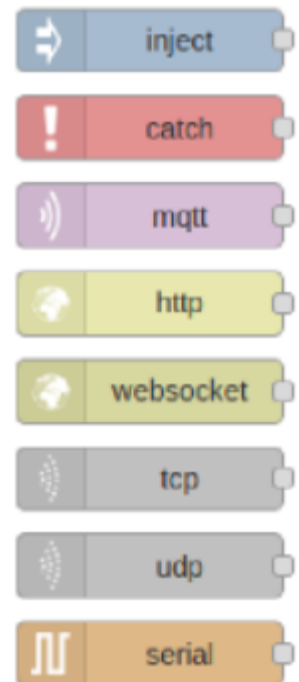
Node name	Description
inject	Injects a timestamp or user-configured text into a message. Can be configured to inject manually, at a set interval, or at specific times (using Cron).
catch	Catches errors thrown by nodes on the same tab. If a node throws an error whilst handling a message, the flow will typically halt. This node can be used to catch those errors returning a message with an error property detailing the error and the source node and type.
mqtt	Subscribes to an MQTT broker and listens on a topic, returns any data published on the topic as a new message. Supports Quality of Service levels and last data retention.
http	Receives HTTP requests, allowing Node-RED to act as a basic web server. HTTP body is delivered as an output message along with any response. Message can contain standard URL-encoded data or JSON.



LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS

NODE-RED INPUT CSOMÓPONTOK

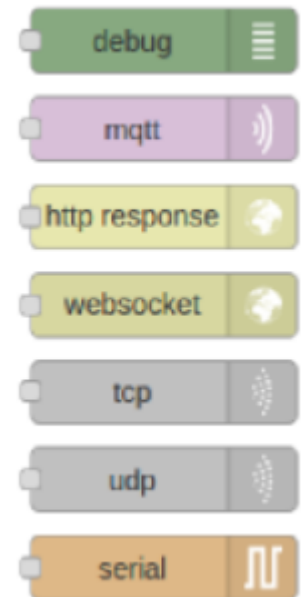
Node name	Description
websocket	Provides an endpoint for a browser to establish a websocket connection with Node-RED. Offers a duplex connection for browser/server combinations.
tcp	Used to accept incoming TCP requests on a specified port or to connect to a remote TCP port. Generates messages containing the TCP data as a single – or stream of – buffer, string or base64 encoded.
udp	Used to accept incoming UDP packets (or multicast packets) on a specified port. Generates messages containing the UDP data as a BUFFER, string or base64 encoded string.
serial in	Reads from a serial port on the local device. Can be configured to read buffers, a specific time period or wait for line breaks.



LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS

NODE-RED OUTPUT CSOMÓPONTOK

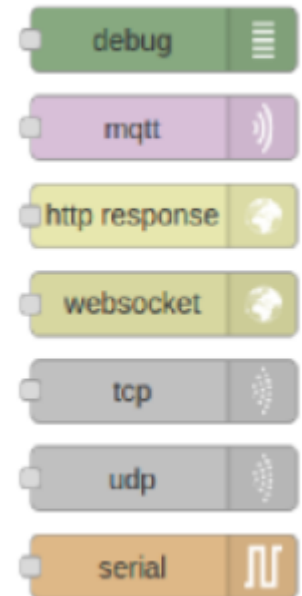
Node name	Description
debug	Provides a simple way to view messages which are displayed in the debug pane. Can be configured to display just the msg.payload or the entire msg object.
mqtt	Subscribes to an MQTT broker and posts any data (msg.payload) it receives in incoming messages to a topic. Supports Quality of Service levels and last data retention.
http	Sends responses back to HTTP requests received from a HTTP Input node. The response body is determined by msg.payload, and can have headers and status code defined.



LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS

NODE-RED OUTPUT CSOMÓPONTOK

Node name	Description
websocket	Sends msg.payload out on the websocket configured. If msg._session is defined, sends to the origination client, otherwise broadcasts to all connected clients
tcp	Replies to a configured TCP port. Can also be used to send to a specific port.
udp	Sends a UDP message to the configured host (ip address) and port. Supports broadcast. Like most nodes, configured through UI or message properties.
serial out	Sends to the defined serial port. Can be configured to send an optional newline after any message payload.



LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS

NODE-RED FUNKCIÓ CSOMÓPONTOK

Node name	Description
function	Generic programmable function node. Using standard JavaScript, the node can be tailored to carry out complex processing on its input messages generating one or more output messages.
template	Configured with a template (using moustache format) of arbitrary complexity, this node takes an input message containing name:value pairs and inserts into the template. Useful for constructing messages, HTML, config files, etc.
delay	A generic node that delays messages by a specific or random time. Can also be configured to throttle a message flow (e.g. 10 msg per sec).
trigger	Creates two output messages separated by a configurable time interval whenever an input message is received. Can also be used as a watchdog timer.
comment	A simple visual comment configured with title and body.



LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS

NODE-RED FUNKCIÓ CSOMÓPONTOK

Node name	Description
http request	Allows you to construct and send a HTTP request to a specific URL. Method (PUT, GET, etc), headers and payload are all configurable through the UI or programmatically.
tcp request	A simple TCP request node. It sends the msg.payload to a server tcp port and expects a response. Can be configured to wait for data, wait for a specific character, or return immediately.
switch	This node routes messages based on their properties. Properties are configured using the UI and can be a variety of logic (>, <, >= etc) applied to a message property.
change	The change node can be used to set, change or delete properties of incoming messages. A variety of configurable rules allow complex changes including search and replace in the msg.payload
range	A simple scaling node that will map numerical input to a new output. Useful for converting or bounding ranges of input values, e.g. temperature. Undefined for non-numeric data.



LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS

NODE-RED SOCIAL CSOMÓPONTOK

Node name	Description
email in	Can be configured to repeatedly read from an IMAP server returning new email as it arrives. Sets msg.topic to email subject and either msg.payload to email text body or msg.html if the email is HTML.
twitter in	Returns tweets as messages. Can be used to search the public or a user's stream for tweets containing the configured search term or all tweets by specific users or direct messages received by the authenticated user.



LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS

NODE-RED SOCIAL CSOMÓPONTOK

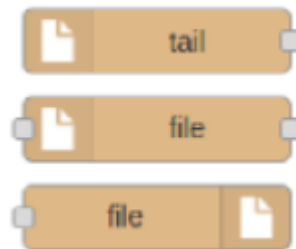
Node name	Description
email out	Sends the incoming message as an email via the configured IMAP server. Topic and recipient all configurable. Will convert binary data to an attachment.
twitter out	Tweets the msg.payload on the configured account. Can send direct messages and will send binary data as an image.



LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS

NODE-RED STORAGE CSOMÓPONTOK

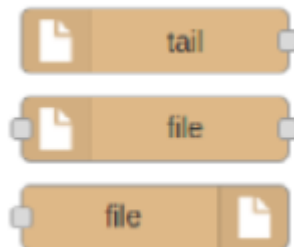
Node name	Description
tail	Tails (i.e. watches for things to be added) to the configured file. (Linux/Mac ONLY) This won't work on Windows file systems, as it relies on the tail -F command.
file in	Reads the specified file and sends the content as msg.payload, and the filename as msg.filename.



LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS

NODE-RED STORAGE CSOMÓPONTOK

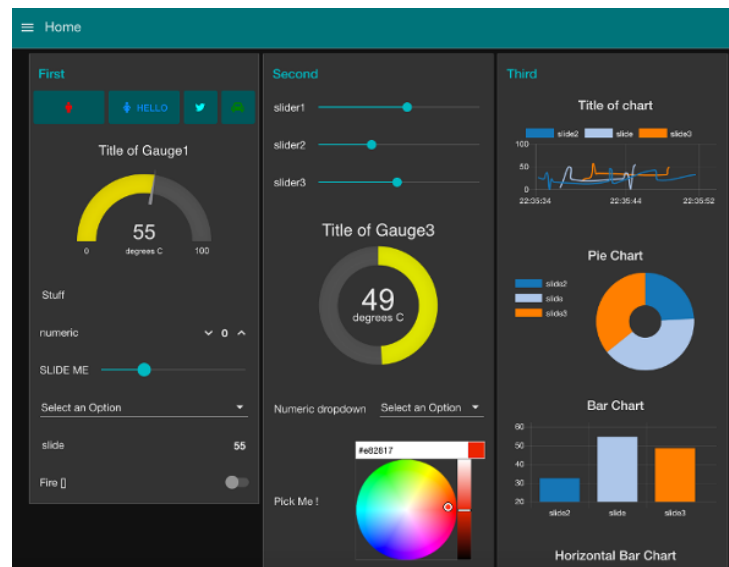
Node name	Description
	The filename can be configured in the node. If left blank, it should be set on msg.filename in an incoming message.
file	Writes msg.payload to the file specified, e.g. to create a log. The filename can be configured in the node. If left blank, it should be set on msg.filename in an incoming message. The default behaviour is to append to the file. This can be changed to overwrite the file each time; for example, if you want to output a “static” web page or report.



LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS NODE-RED DASHBOARD

- Node-RED kiegészítő komponens

Node-RED Dashboard is a module that provides a set of nodes in Node-RED to quickly create a live data dashboard

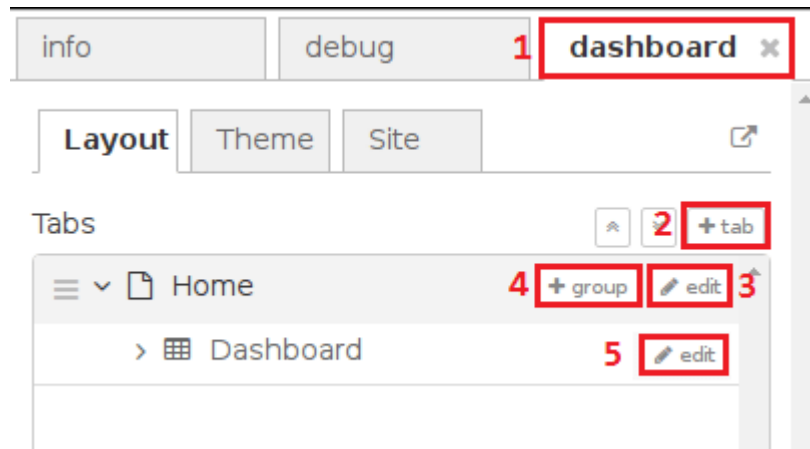


LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS NODE-RED DASHBOARD



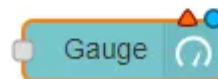
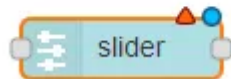
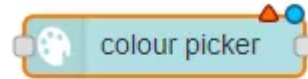
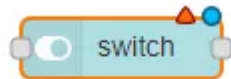
LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS NODE-RED DASHBOARD KÉSZÍTÉS

- Layout és Groups;



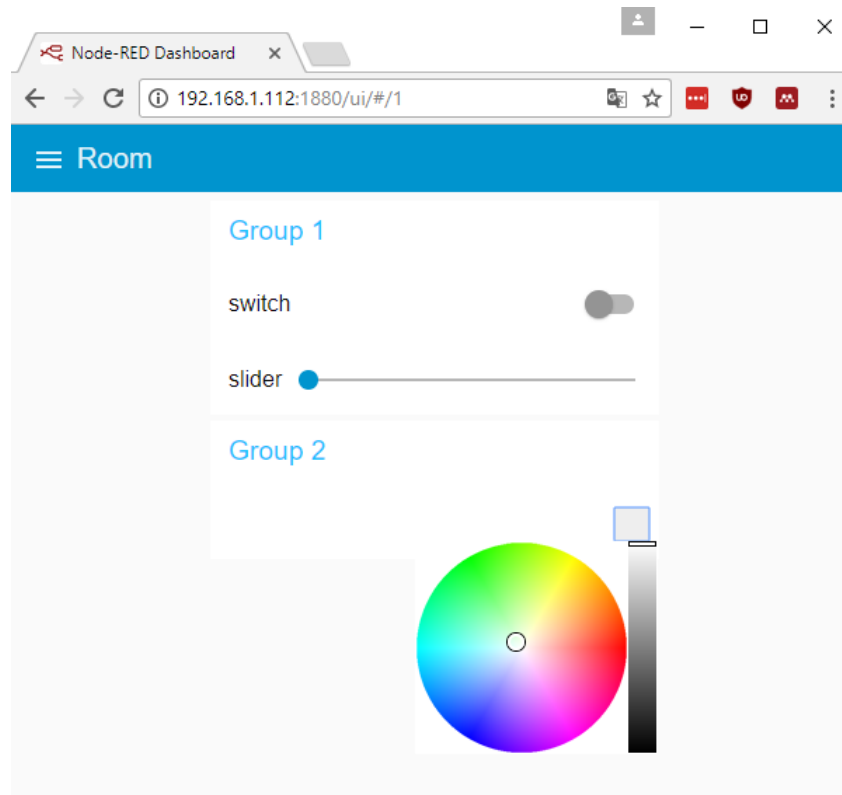
LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS NODE-RED DASHBOARD KÉSZÍTÉS

- Widgets



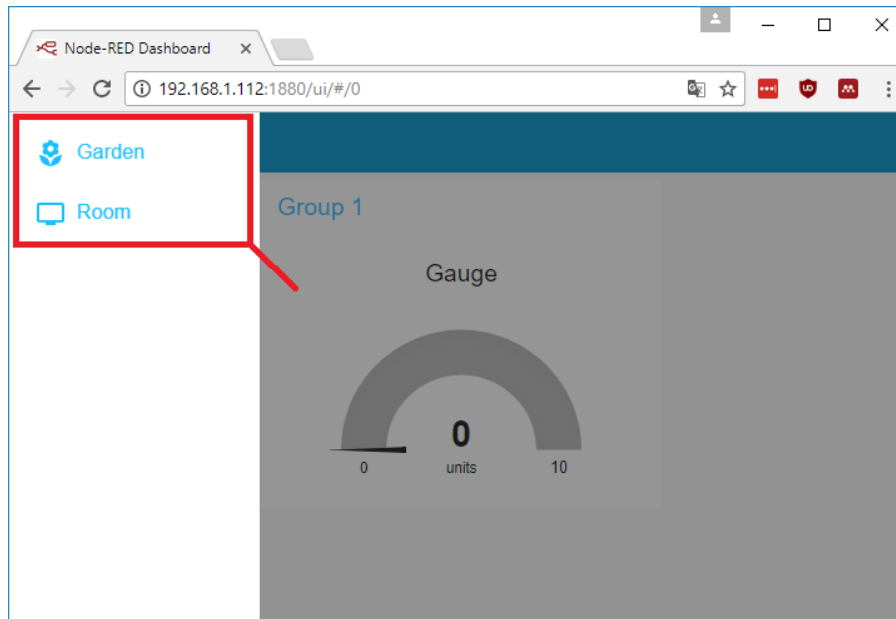
LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS NODE-RED DASHBOARD MEGJELENÍTÉSE

- Megjelenítés böngészőben



LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS NODE-RED DASHBOARD MEGJELENÍTÉSE

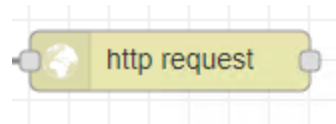
- Megjelenítés böngészőben



LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS

FELADAT: LED KAPCSOLÁS NODE-RED-VEL

- http csomópont



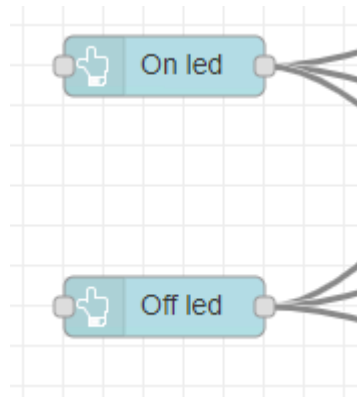
- Inject csomópont



LOKÁLIS ADATGYŰJTÉS ÉS FOLYAMAT VEZÉRLÉS

FELADAT: LED KAPCSOLÁS NODE-RED-VEL

- Nyomógomb dashboard csomópont



IOT RENDSZEREK ADATKOMMUNIKÁCIÓJA

CLOUD PLATFORMOK

- **Miért van rájuk szükség**

As the devices started to become connected, we need a place to send, store, and process all of the information. Setting up your own in-house system isn't practical anymore. The cost of maintaining, upgrading and securing a system is just too high, and there are some great services available.

IOT RENDSZEREK ADATKOMMUNIKÁCIÓJA

CLOUD PLATFORMOK

- **Vezető IT cégek platformjai**

Amazon Web Services (AWS)

Microsoft Azure IoT

IBM Watson

- **Open source platformok**

ThingSpeak

thethings.io

www.openhab.org

IOT RENDSZEREK ADATKÖMUNIKÁCIÓJA

KÖMUNIKÁCIÓS PROTOKOLLOK

- **MQTT**

MQTT is a machine-to-machine (M2M)/"Internet of Things" connectivity protocol. It was designed as an extremely lightweight publish/subscribe messaging transport.

- **AMQP**

„AMQP is the Internet Protocol for Business Messaging”

IOT RENDSZEREK ADATKOMMUNIKÁCIÓJA

KOMMUNIKÁCIÓS PROTOKOLLOK

- HTTP

Használható, de tekintettel a komplexitására M2M esetekben nem kifizetődő

- XMPP, CoAP

IOT RENDSZEREK ADATKOMMUNIKÁCIÓJA

MQTT PROTOKOLL

- **MQTT = Message Queueing Telemetry Transport**

Szűkös erőforrásokkal rendelkező eszközök számára, kevés memória, kis teljesítményű processzor.

- **Publish/subscribe (one-to-many) modell**

A kliensek (client) feliratkoznak (subscribe) ún. topic-okra. Feliratkozás után a topic-ra publikált (publish) összes üzenetet megkapják.

IOT RENDSZEREK ADATKOMMUNIKÁCIÓJA

MQTT PROTOKOLL

- **Szolgáltatási szintek**

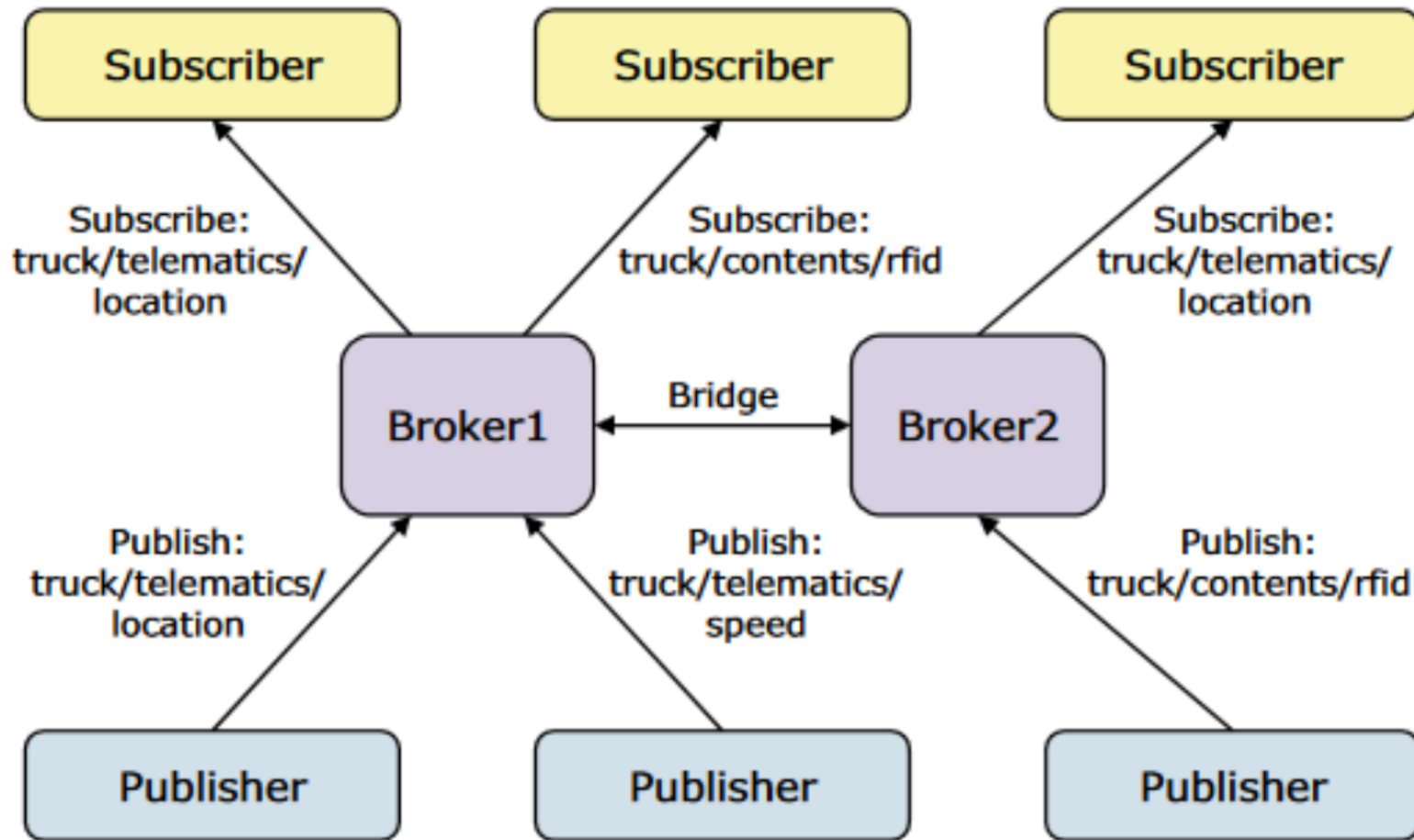
3 QoS szint definiált. Magasabb szolgáltatásminőség nagyobb megbízhatóságú üzenet célba juttatást valósít meg – ennek ára a nagyobb sávszélesség és/vagy késleltetés.

- **Megőrzött üzenetek**

A szerver meg is tartja az utoljára elküldött üzenetet. Egy új feliratkozó esetén egyből elküldi az utolsó üzenetet is a kliensek.

IOT RENDSZEREK ADATKÖMUNIKÁCIÓJA

MQTT MŰKÖDÉSE



Extended

IOT RENDSZEREK ADATKOMMUNIKÁCIÓJA

MQTT BRÓKEREK

- **Szerver, amely implementálja az MQTT protokollt.**

Közvetít az MQTT kliensek közötti kommunikációban.

- **Mosquitto**

Egyszerű, nyílt, ingyenes implementáció (MQTT v3.1)

<http://Mosquitto.org>

IOT RENDSZEREK ADATKÖMUNIKÁCIÓJA

MQTT BRÓKEREK

- **Egyéb brókerek**

HiveMQ: MQTT broker for the connected enterprise.

RabbitMQ

EMQ The Massively Scalable MQTT Broker for IoT

VerneMQ

IOT RENDSZEREK ADATKOMMUNIKÁCIÓJA

MQTT TOPIKOK ÉS FÁK

- **topic**

Az üzenetek osztályozását szolgálja, definiálja az üzenet tartalmát.

Pl: „truck”

- **topic-tree**

Tipikusan a topic-ok hierarchikusan szervezettek.

A „/” karakter használatával hozhatók létre al-topikok.

Pl: „truck/contents”, „truck/contents/rfid”

IOT RENDSZEREK ADATKOMMUNIKÁCIÓJA

MQTT TOPIKOK ÉS FÁK

- **topic-string**

Egy karakterlánc, amely meghatározza az üzenet topikját. (wildcard használható!)

Pl: „truck/contents/#”, „truck/+”

IOT RENDSZEREK ADATKOMMUNIKÁCIÓJA

MQTT KLIENSEK

- MQTT kliens alkalmazás

adatokat gyűjt egy telemetriai eszközről, csatlakozik egy MQTT szerverhez, majd egy topic string segítségével publikálja az információt, ÉS/VAGY

feliratkozik egy topic-ra, ahol fogadja a publikált adatokat, ÉS/VAGY

parancsok segítségével vezérli a telemetriai eszköz(öke)t.

IOT RENDSZEREK ADATKOMMUNIKÁCIÓJA

MQTT KLIENSEK

- **Eclipse Paho MQTT kliens**

<http://www.eclipse.org/paho/>

Széleskörű implementáció

Pl. C, Java és .Net stb

- **PubSubClient**

Arduino Client for MQTT

<https://pubsubclient.knolleary.net>

<https://github.com/knolleary/pubsubclient>

IOT RENDSZEREK ADATKOMMUNIKÁCIÓJA

MQTT ADATKÜLDÉS ELŐKÉSZÍTÉSE

- `#include <PubSubClient.h>`

```
PubSubClient client(server, 1883, callback, wifiClient);
```

- **Connecting**

```
boolean connect(const char* id, const char* user, const char* pass);  
!client.connected()
```

- **Publish**

```
client.publish(topic, "hello from ESP8266")
```

IOT RENDSZEREK ADATKOMMUNIKÁCIÓJA

MQTT ADATKÜLDÉS MEGVALÓSÍTÁS

- **Publishing**

```
void reconnect() {  
  // Loop until we're reconnected  
  while (!client.connected()) {  
    Serial.print("Attempting MQTT connection...");  
    // Attempt to connect  
    if (client.connect("arduinoClient")) {  
      Serial.println("connected");  
      // Once connected, publish an announcement...  
      client.publish("openhab","himitsu sensor, reporting in");  
      // ... and resubscribe
```

IOT RENDSZEREK ADATKOMMUNIKÁCIÓJA

MQTT ADATKÜLDÉS MEGVALÓSÍTÁS

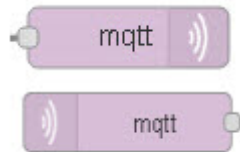
- **Publishing**

```
} else {  
    Serial.print("failed, rc=");  
    Serial.print(client.state());  
    Serial.println(" try again in 5 seconds");  
    // Wait 5 seconds before retrying  
    delay(5000);  
}  
}  
}
```

IOT RENDSZEREK ADATKOMMUNIKÁCIÓJA

NODE-RED MQTT CSOMÓPONT

- Subscribe csomópont konfigurálása



mqtt out > Edit mqtt-broker node

Delete **server name or IP address** Cancel Update

default = 1883

Connection Security Birth Message Will Message

Server 192.168.1.157 Port 1883

☐ Enable secure (SSL/TLS) connection

Client ID node-client **client name must be unique**

Keep alive time (s) 60 ☐ Use clean session

☒ Use legacy MQTT 3.1 support

Default keep alive is 60 secs and clean session is False. It is normal to enable clean sessions

IOT RENDSZEREK ADATKOMMUNIKÁCIÓJA

NODE-RED MQTT CSOMÓPONT

- Subscribe csomópont konfigurálása

The screenshot shows the configuration window for an MQTT Subscribe node in Node-RED. The window has a title bar with 'Delete', 'Cancel', and 'Done' buttons. Below the title bar is a section titled 'node properties' with a dropdown arrow. The main configuration area includes:

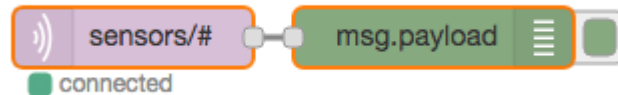
- Server:** A text field containing 'node-client@192.168.1.157:1883' with a dropdown arrow and an edit icon. A red arrow points to this field with the text 'Select server to use or edit or add a new'.
- Topic:** A text field containing 'Topic'. A red arrow points to this field with the text 'Topic to publish on'.
- QoS:** A dropdown menu with a red arrow pointing to it and the text 'Qos'.
- Retain:** A checkbox labeled 'Retain' and a dropdown menu. A red arrow points to the dropdown with the text 'Set retain message flag'.
- Name:** A text field containing 'Name'. A red arrow points to this field with the text 'Name as appears in the flow'.

At the bottom, there is a yellow tip box that reads: 'Tip: Leave topic, qos or retain blank if you want to set them via msg properties.'

IOT RENDSZEREK ADATKÖMUNIKÁCIÓJA

FELADAT: ADATFOGADÁS MQTT-N NODE-RED-BEN

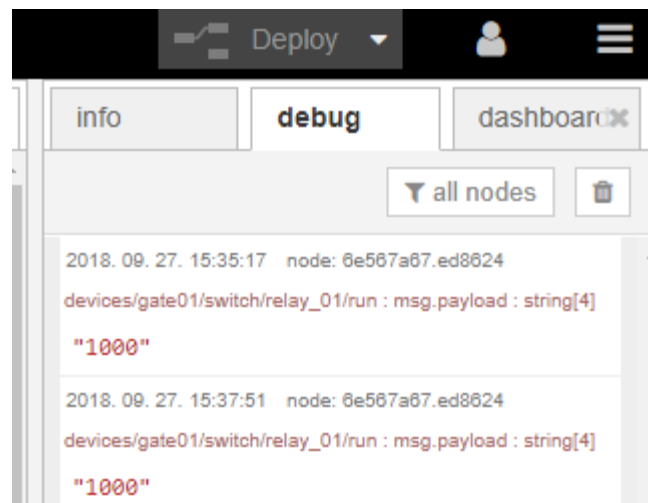
- Node-RED csomópont konfigurálás



IOT RENDSZEREK ADATKÖMUNIKÁCIÓJA

FELADAT: ADATFOGADÁS MQTT-N NODE-RED-BEN

- Node-RED debug csomópont konfigurálása



ADATFOGADÁS IOT ESZKÖZ OLDALON

ADATTARTALMAK

- **Típusok**

Text pl. JSON

Binary pl. Protocol Buffers

- **Payload Design Notes**

“0” or „1”

{“status”:”ON”} or {“status”:”OFF”} – Data

{“set”:”ON”} or {“set”:”OFF”} – –Commands

{“living-room-light”: {“SET”:”ON”}}

ADATFOGADÁS IOT ESZKÖZ OLDALON

TOPIK ELNEVEZÉSI STRATÉGIA

- **When creating the topic naming scheme the approaches are**

Put as much information as possible in the topic field or

Put as much information as possible in the message payload

- **A topic hierarchy can consists of**

High level topic grouping devices.- optional

Assigned Sensor name – optional

function e.g. status,set,get,cmd –optional

ADATFOGADÁS IOT ESZKÖZ OLDALON

TOPIK ELNEVEZÉSI STRATÉGIA

- The message payload can consist of

Payload data

Sensor ID- optional

Function – optional

single-level
wildcard
↓
myhome / groundfloor / + / temperature
only one level

ADATFOGADÁS IOT ESZKÖZ OLDALON

JSON PARSER

- `#include <ArduinoJson.h>`

JSON stands for JavaScript Object Notation. JSON is a lightweight text-based open standard design for exchanging data.

```
{"temperature":27.23, "humidity":62.05, "pressure":1013.25}
```

ADATFOGADÁS IOT ESZKÖZ OLDALON

JSON PARSER

- **Parsing**

```
StaticJsonBuffer<200> jsonBuffer;  
JsonObject& root = jsonBuffer.parseObject(json);
```

```
const char* sensor = root["sensor"];  
long time = root["time"];  
double latitude = root["data"][0];  
double longitude = root["data"][1];
```


ADATFOGADÁS IOT ESZKÖZ OLDALON

MQTT BIZTONSÁG

- **MQTT security**

Network level

Transport level

Application Level

ADATFOGADÁS IOT ESZKÖZ OLDALON

MQTT BIZTONSÁG

- **Securing MQTT with password**

By default, Mosquitto will allow anonymous access.

Set Mosquitto up to require usernames and passwords from clients before they can connect.

Use the utility `mosquitto_passwd` to generate the passwords file `/etc/mosquitto/passwd`:

ADATFOGADÁS IOT ESZKÖZ OLDALON

MQTT TESZTELÉSI LEHETŐSÉGEK

- **mosquitto_pub**

Tesztelési lehetőségek parancssori mosquitto-val
`mosquitto_pub -t 'test/topic' -m 'helloWorld'`

- **MQTTlens**

ADATFOGADÁS IOT ESZKÖZ OLDALON MQTT TESZTELÉSI LEHETŐSÉGEK

The screenshot displays the MQTTlens web interface. On the left, a sidebar shows a 'Connections' list with 'HiveMQ local' connected. The main panel is divided into sections: 'Subscribe' and 'Publish' forms, a 'Message' input field, and a 'Subscriptions' table. The 'Subscribe' form has a topic 'test/mqtt/topic/1' and a QoS of '2 - exactly once'. The 'Publish' form has the same topic and a QoS of '1 - at least once'. The 'Message' field contains 'Hello MQTT!'. The 'Subscriptions' table shows a subscription for 'test/mqtt/topic' with 0/1 messages. Below it, a message log shows a message 'Hello MQTT!' at 3:07:41. The bottom section shows a subscription for 'test/mqtt/topic/1' with 15/15 messages.

MQTTlens

Connections + ^

HiveMQ local **connected** ⚙️ 🗑️

Connection: HiveMQ local

Subscribe ^

Topic QoS **Subscribe**

Publish ^

Topic QoS **Publish**

Message

Subscriptions

Topic: "test/mqtt/topic" Showing the last 1 messages - + 🗑️ Messages: 0/1 ^

#	Time	Topic	QoS
0	3:07:41	test/mqtt/topic	1

Message: Hello MQTT!

Topic: "test/mqtt/topic/1" Showing the last 3 messages - + 🗑️ Messages: 15/15 ^

Fork me on GitHub

ADATFOGADÁS IOT ESZKÖZ OLDALON

MQTT KAPCSOLÓ VEZÉRLÉS

- 2 csatornás relé modul

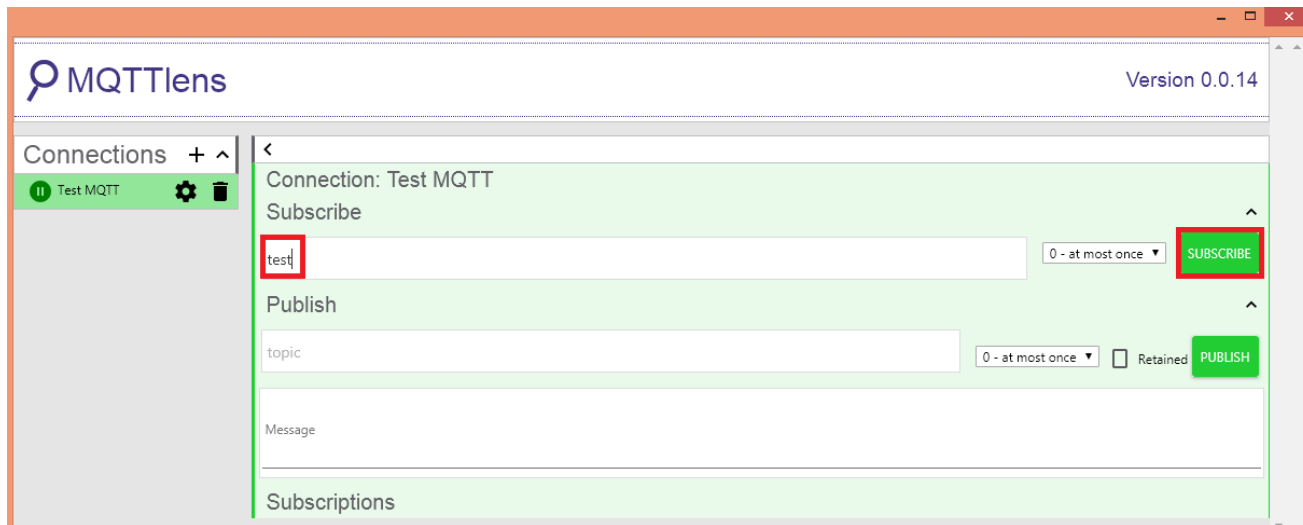
Előkészítve közös tesztre

- MQTTlens beállítása



ADATFOGADÁS IOT ESZKÖZ OLDALON

MQTT KAPCSOLÓ VEZÉRLÉS



ADATFOGADÁS IOT ESZKÖZ OLDALON

MQTT ADATFOGADÁS ELŐKÉSZÍTÉSE

- **PubSubClient Callback**

```
void callback(char* topic, byte* payload, unsigned int length) {  
  if (strcmp(topic,"pir1Status")==0)  
    // whatever you want for this topic  
}
```

ADATFOGADÁS IOT ESZKÖZ OLDALON

MQTT ADATFOGADÁS ELŐKÉSZÍTÉSE

- **PubSubClient Subscribe**

boolean subscribe (topic, [qos])

Subscribes to messages published to the specified topic.

Parameters

topic - the topic to subscribe to (const char[])

qos - optional the qos to subscribe at (int: 0 or 1 only)

ADATFOGADÁS IOT ESZKÖZ OLDALON

MQTT CALLBACK IMPLEMENTÁLÁSA

- **Subscribe implementálás**

```
if (client.connect("arduinoClient")) {  
    client.subscribe("pir1Status");  
    client.subscribe("red");  
    client.subscribe("green");  
    client.subscribe("blue");  
}
```

ADATFOGADÁS IOT ESZKÖZ OLDALON

MQTT CALLBACK IMPLEMENTÁLÁSA

- **Callback implementálás**

```
void callback(char* topic, byte* payload, unsigned int length) {  
    if (strcmp(topic,"pir1Status")==0){  
        // whatever you want for this topic  
    }  
    if (strcmp(topic,"red")==0) {  
        // obvioulsy state of my red LED  
    }  
    if (strcmp(topic,"blue")==0) {  
        // this one is blue...  
    }  
    if (strcmp(topic,"green")==0) {  
        // this orange  
    }  
}
```

ADATFOGADÁS IOT ESZKÖZ OLDALON NODE-RED MQTT ADATKÜLDÉS ELŐKÉSZÍTÉSE

- Node-RED csomópont konfigurálás

Kimeneti csatorna definiálása

Edit mqtt out node

CancelDone

Server

Add new mqtt-broker...

Topic

Topic

QoS

Retain

Name

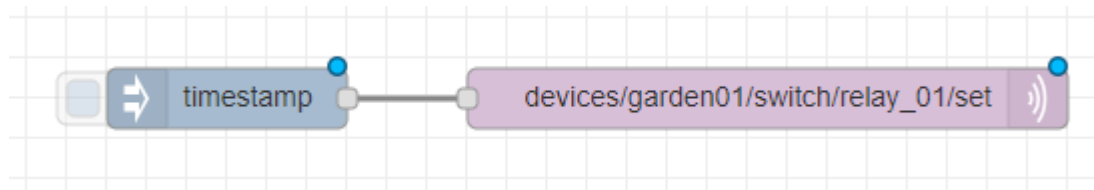
Name

Tip: Leave topic, qos or retain blank if you want to set them via msg properties.

ADATFOGADÁS IOT ESZKÖZ OLDALON

NODE-RED MQTT ADATKÜLDÉS ELŐKÉSZÍTÉSE

- Inject csomópont definiálása



ADATFOGADÁS IOT ESZKÖZ OLDALON NODE-RED DASHBOARD ELŐKÉSZÍTÉSE

- Kapcsoló csomópont definiálása

Edit switch node

CancelDone

 Group

Lamp [Room] 

 Size

auto

 Label

Lamp 

 Icon

Default ▼

 If **msg** arrives on input, pass through to output: ☒

 When clicked, send:

On Payload ▼  on

Off Payload ▼  off

Topic

room/lamp

 Name

Lamp

ADATFOGADÁS IOT ESZKÖZ OLDALON

FELADAT: LED KAPCSOLÁS MQTT-N KERESZTÜL

- Csoport és menüelem felvétele



Edit dashboard group node

Delete Cancel Update

Tab Ecran Principal

Width 6

Name Eléments d'entrée

☒ Display group name

- Telepítés



AZURE MINT PLATFORM

AZURE FELÉPÍTÉSE

- **Tulajdonságai**

Szolgáltatás alapú

Robosztus, redundáns

Skálázható

Globális

AZURE MINT PLATFORM

AZURE FELÉPÍTÉSE

- **Mire használható?**

24/7 szolgáltatások üzemeltetése

Infrastruktúra biztosítása

„Nagy Adat” tárolás

Kutatás, adatfeldolgozás, adatbányászat, HPC

Média sugárzás nagy tömegeknek

Időszakos számítás, kampány csúcs

AZURE MINT PLATFORM

AZURE SZOLGÁLTATÁSOK

- **COMPUTE and DATA**

Virtual machines, Web sites

Mobile services, Cloud services

Storage, SQL Database

HDInsight

Cache, Recovery manager

AZURE MINT PLATFORM

AZURE SZOLGÁLTATÁSOK

- **App services**

- Media services

- Service bus

- Scheduler

- Notification hubs

- BizTalk Services

- Active Directory

- Multi-Factor authentication

AZURE MINT PLATFORM

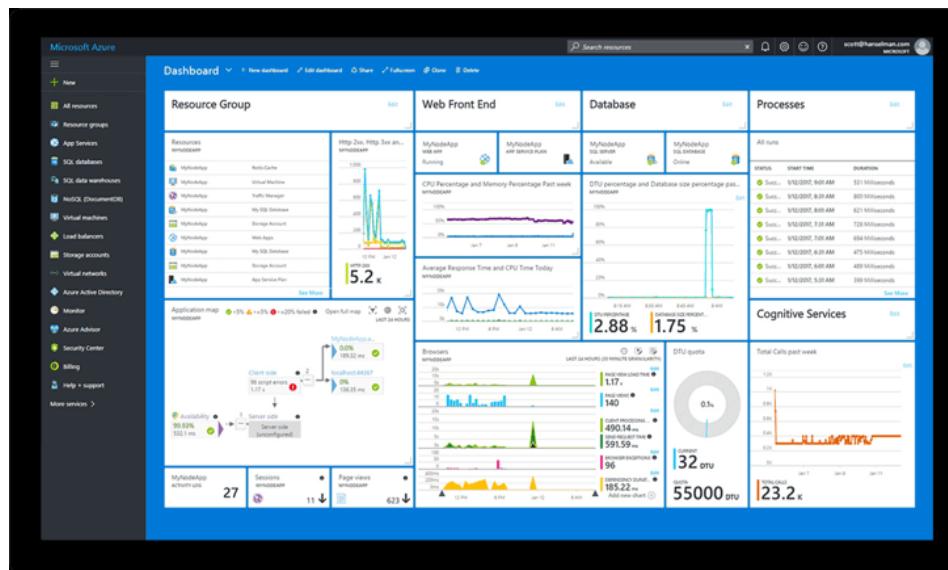
AZURE FIÓK

- Fejlesztési környezetet

Ingyenes Azure-fiók létrehozása

Ingyenes szolgáltatások 12 hónapig

<https://azure.microsoft.com/hu-hu/free/>



AZURE MINT PLATFORM

AZURE EVENT HUB

- **Event Hubs-eseményközpontok**

Big Data streamplatform és eseményfeldolgozó szolgáltatás, amely másodpercenként több millió esemény fogadására és feldolgozására képes

Adatbeolvasás bárholonnan, és platformközi fejlesztés olyan népszerű protokollok támogatásával, mint az AMQP, a HTTPS

AZURE MINT PLATFORM

AZURE EVENT HUB

- **Event Hubs-eseményközpontok**

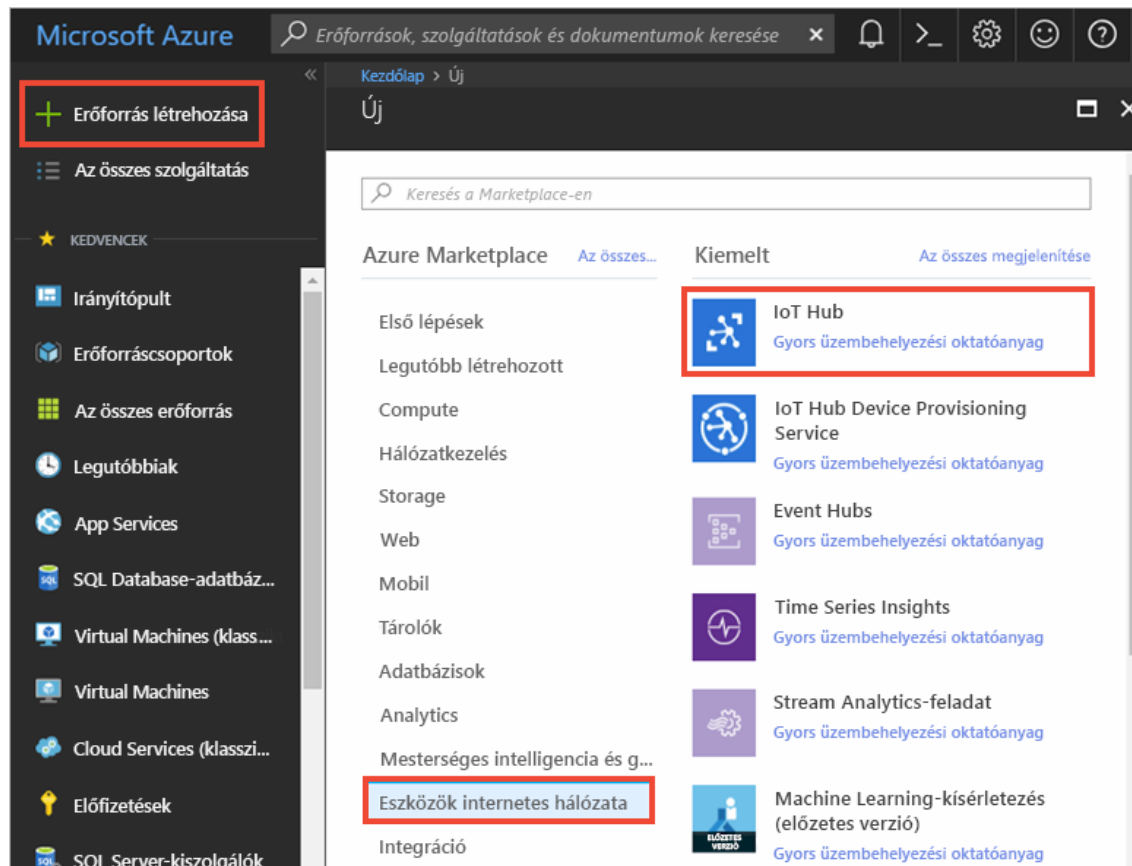
Az Event Hubs képes az elosztott szoftverek és eszközök által generált események, adatok vagy telemetria feldolgozására és tárolására.

Az eseményközpontokba elküldött adatok bármilyen valós idejű elemzési szolgáltató vagy kötegelési/tárolóadapter segítségével átalakíthatók és tárolhatók.

AZURE MINT PLATFORM

AZURE IOT HUB

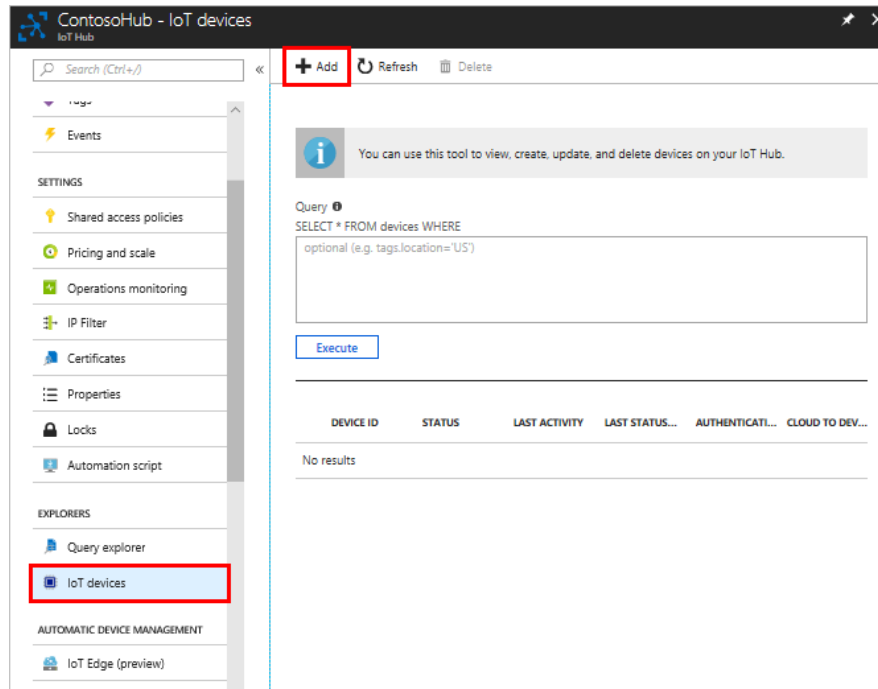
- IoT Hub létrehozása



AZURE MINT PLATFORM

AZURE IOT ESZKÖZ

- Eszköz regisztrálása az IoT hubban



AZURE MINT PLATFORM

AZURE IOT ESZKÖZ

- Eszköz regisztrálása az IoT hubban

Device Details
myDeviceId

Save Device Twin Message To Device Direct Method Add Module Identity Refresh

Device Id ⓘ
myDeviceId

Primary key ⓘ
<primary key>

Secondary key ⓘ
<secondary key>

Connection string—primary key ⓘ
HostName=ContosoHub.azure-devices.net;DeviceId=myDeviceId;SharedAccessKey= <primary key>

Connection string—secondary key ⓘ
HostName=ContosoHub.azure-devices.net;DeviceId=myDeviceId;SharedAccessKey= <secondary key>

Connect device to IoT Hub ⓘ
Enable Disable

AZURE MINT PLATFORM ADATTÁROLÁS

- **Megfelelő adatkezelő platform kiválasztása**

Nagymennyiségű adat

Elsődlegesen írás nincs módosulás

Törlés időközönként nagy mennyiségben

AZURE MINT PLATFORM

ADATTÁROLÁS

- **Reláció adatbázisok**

Nem erre a célra készültek

- **TSDB adatbázisok**

Nagy I/O sebesség

Nagy mennyiségű adat kezelése

AZURE MINT PLATFORM

AZURE IOT ÜZENET MENTÉSE

- StorageContainer létrehozása és beállítás

Adatok irányítása egy tárfiókba

Add endpoint

[Learn more about setting up a custom endpoint.](#)

* Name

StorageContainer

* Endpoint type

Azure Storage Container

* Azure Storage Container

https://contosoresultsstorage.blob.core.windows.net/contosoresults

Batch frequency ⓘ

100 s

Chunk size window ⓘ

100 MB

Blob file name format ⓘ

{iothub}/{partition}/{YYYY}/{MM}/{DD}/{HH}/{mm}

E.g. ContosoTestHub/0/2018/03/25/00/57 ⓘ

AZURE MINT PLATFORM

AZURE IOT ÜZENET MENTÉSE

- StorageContainer létrehozása és beállítás

Adatok irányítása egy tárfiókba

Query string ⓘ

```
1 level="storage"
```

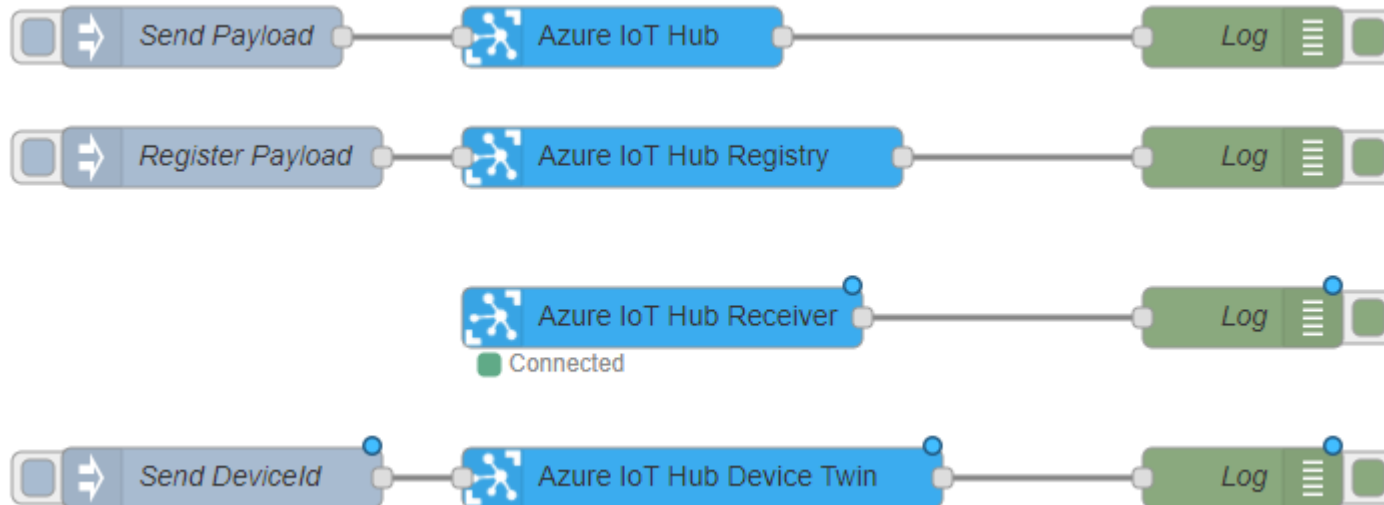
Test the route (optional)

```
1 {  
2   "message": {  
3     "systemProperties": {  
4       "contentType": "application/json",  
5       "contentEncoding": "utf-8",  
6       "iothub-message-source": "deviceMessages",  
7       "iothub-enqueuedtime": "2017-05-08T18:55:31.8514657Z"  
8     },  
9   },  
10 }
```

AZURE MINT PLATFORM

NODE-RED AZURE CSOMÓPONTOK

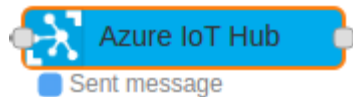
- Azure IoT elérését biztosító csomópontok



AZURE MINT PLATFORM

NODE-RED AZURE KAPCSOLÓDÁS

- Azure IoT Hub csomópont használata



Edit azureiothub node

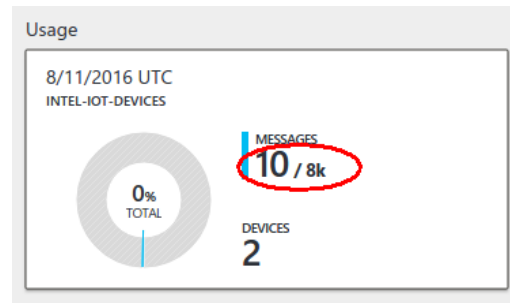
Name	Azure IoT Hub
Protocol	amqp
Connection String	HostName=intel-iot-devices.azure-devices.net;Dev

AZURE MINT PLATFORM

NODE-RED AZURE KAPCSOLÓDÁS

- Azure IoT Hub kapcsolat ellenőrzése

Amennyiben a kapcsolat felépült az Azure portálon nyomonkövethető

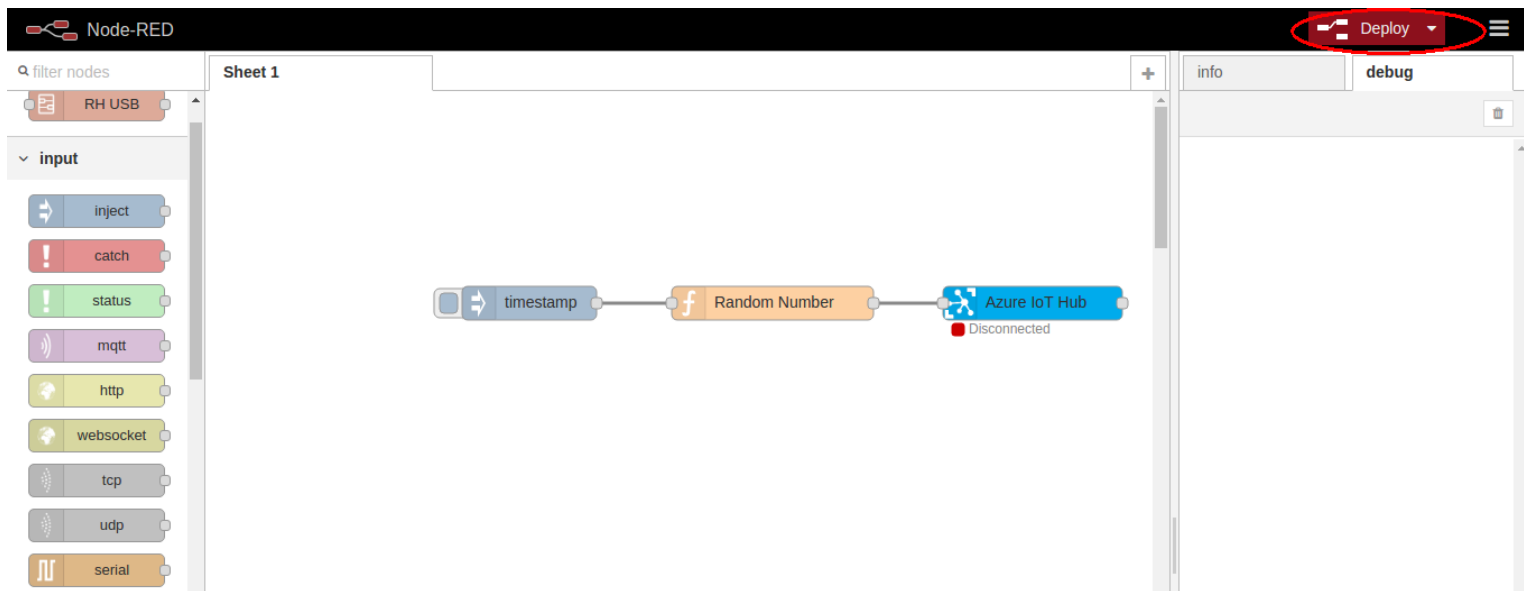


AZURE MINT PLATFORM

FELADAT: ADAT KÜLDÉSE AZURE FELHŐBE

- Véletlenszám küldése az Azure felhőbe

Az inject és funkció csomópontok segítségével adat küldése



SZERVIZ ALAPÚ ADATKISZOLGÁLÁS

RESTFUL ALKALMAZÁSOK

- **REST (Representational State Transfer)**

egy szoftver architektúra típus, amely lehetővé teszi skálázható, nagy teljesítményű elosztott hálózati alkalmazások fejlesztésére

HTTP alapon kommunikál

HTTP utasítások (GET, POST, PUT, DELETE, ...) segítségével

kliens-szerver modell

egységes interfész

állapotmentes kommunikáció

kiegészíthetőség (code on demand)

SZERVIZ ALAPÚ ADATKISZOLGÁLÁS

WEB API KERETRENDSZER

- **Web API**

WebAPI egy RESTful alkalmazások fejlesztését lehetővé tevő keretrendszer

az adatokat alapértelmezetten JSON (Javascript Object Notation) formátumban továbbítja, de a kliens kérésének megfelelően automatikusan tudja a formátumot módosítani

```
{    // objektum
    "id": 1234, // attribútum
    "group": "tool",
    "name": "hammer",
    "resposible": { "name" : "John" },
    "materials": [{ "name": "steel" }, ... ]
}
```

SZERVIZ ALAPÚ ADATKISZOLGÁLÁS

WEB API KERETRENDSZER

- Web API

Action	HTTP method	Relative URI
Get a list of all products	GET	/api/products
Get a product by ID	GET	/api/products/ <i>id</i>
Get a product by category	GET	/api/products?category= <i>category</i>
Create a new product	POST	/api/products
Update a product	PUT	/api/products/ <i>id</i>
Delete a product	DELETE	/api/products/ <i>id</i>

SZERVIZ ALAPÚ ADATKISZOLGÁLÁS

VISUAL STUDIO FEJLESZTŐESZKÖZ

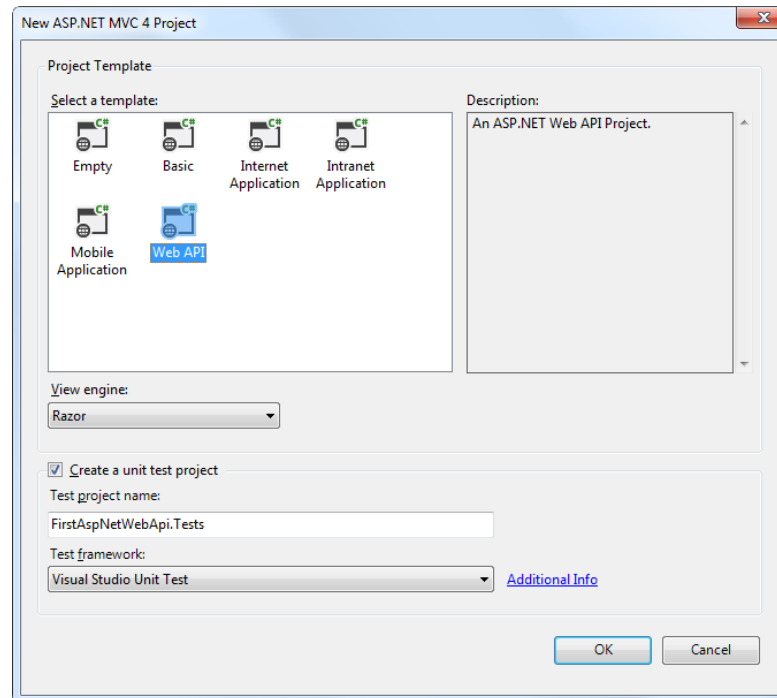
- Visual Studio mint fejlesztői eszköz

Supported Features	Visual Studio Community	Visual Studio Professional	Visual Studio Enterprise
⊕ Supported Usage Scenarios	●●●○	●●●●	●●●●
Development Platform Support ²	●●●●	●●●●	●●●●
⊕ Integrated Development Environment	●●●○	●●●○	●●●●
⊕ Advanced Debugging and Diagnostics	●●○○	●●○○	●●●●
⊕ Testing Tools	●○○○	●○○○	●●●●
⊕ Cross-platform Development	●●○○	●●○○	●●●●
⊕ Collaboration Tools and Features	●●●●	●●●●	●●●●

SZERVIZ ALAPÚ ADATKISZOLGÁLÁS

VISUAL STUDIO PROJEKTEK

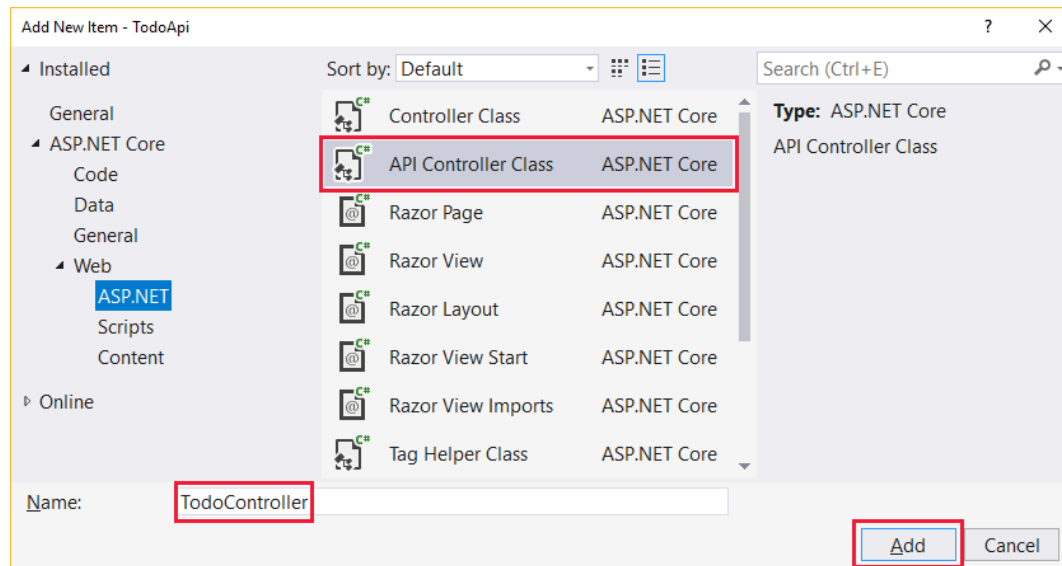
- Előkészített Web API projektek létrehozása



SZERVIZ ALAPÚ ADATKISZOLGÁLÁS

VISUAL STUDIO PROJEKTEK

- Előkészített Web API projektek létrehozása



SZERVIZ ALAPÚ ADATKISZOLGÁLÁS

VISUAL STUDIO WEB API KÉSZÍTÉS

- **Api Controller felépítése**

```
public class ValuesController : ApiController
{
    // GET api/values
    public IEnumerable Get()
    {
        return new string[] { "value1", "value2" };
    }

    // GET api/values/5
    public string Get(int id)
    {
        return "value";
    }
}
```

SZERVIZ ALAPÚ ADATKISZOLGÁLÁS

VISUAL STUDIO WEB API KÉSZÍTÉS

- **Api Controller felépítése**

```
// POST api/values
```

```
public void Post([FromBody]string value)
```

```
{
```

```
}
```

```
// PUT api/values/5
```

```
public void Put(int id, [FromBody]string value) { }
```

```
// DELETE api/values/5
```

```
public void Delete(int id) { }
```

```
}
```


SZERVIZ ALAPÚ ADATKISZOLGÁLÁS DOKUMENTÁLÁS SWAGGER-VEL

- **Mi a Swagger**

Swagger is a language-agnostic specification for describing REST APIs. The Swagger project was donated to the OpenAPI Initiative, where it's now referred to as OpenAPI.

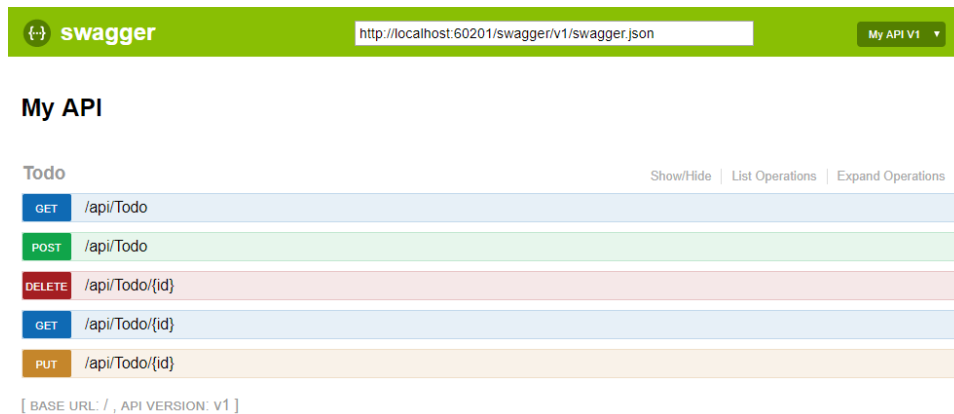
It allows both computers and humans to understand the capabilities of a service without any direct access to the implementation (source code, network access, documentation). One goal is to minimize the amount of work needed to connect disassociated services. Another goal is to reduce the amount of time needed to accurately document a service.

SZERVIZ ALAPÚ ADATKISZOLGÁLÁS DOKUMENTÁLÁS SWAGGER-VEL

- **Swagger UI**

Web-based UI that provides information about the service, using the generated Swagger specification

SZERVIZ ALAPÚ ADATKISZOLGÁLÁS DOKUMENTÁLÁS SWAGGER-VEL



The image shows the Swagger UI interface for an API. At the top, there is a green header bar with the Swagger logo, the URL `http://localhost:60201/swagger/v1/swagger.json`, and a dropdown menu labeled "My API V1". Below the header, the title "My API" is displayed. Underneath, the "Todo" API is listed with five endpoints: `GET /api/ToDo`, `POST /api/ToDo`, `DELETE /api/ToDo/{id}`, `GET /api/ToDo/{id}`, and `PUT /api/ToDo/{id}`. Each endpoint is represented by a colored bar with its method and path. To the right of the endpoints, there are links for "Show/Hide", "List Operations", and "Expand Operations". At the bottom, a status bar indicates "[BASE URL: / , API VERSION: v1]".

swagger `http://localhost:60201/swagger/v1/swagger.json` My API V1

My API

Todo [Show/Hide](#) | [List Operations](#) | [Expand Operations](#)

- GET `/api/ToDo`
- POST `/api/ToDo`
- DELETE `/api/ToDo/{id}`
- GET `/api/ToDo/{id}`
- PUT `/api/ToDo/{id}`

[BASE URL: / , API VERSION: v1]

SZERVIZ ALAPÚ ADATKISZOLGÁLÁS

SWAGGER HASZNÁLATA

- **Swagger.json**

```
{"swagger": "2.0",  
  "info": {  
    "version": "v1",  
    "title": "API V1"  
  },  
  "basePath": "/",  
  "paths": {  
    "/api/ToDo": {
```

SZERVIZ ALAPÚ ADATKISZOLGÁLÁS

SWAGGER HASZNÁLATA

- XML comments

```
/// <summary>
/// Deletes a specific TodoItem.
/// </summary>
/// <param name="id"></param>
[HttpDelete("{id}")]
public void Delete(string id)
{
    TodoItems.Remove(id);
}
```

SZERVIZ ALAPÚ ADATKISZOLGÁLÁS TESZTELÉS SWAGGER-VEL

- Swashbuckle mint eszköz

SuperHero

Show/Hide | List Operations | Expand Operations

GET /api/SuperHero

Response Class (Status 200)

Model | Model Schema

```
[
  {
    "Id": "string",
    "Name": "string",
    "RealName": "string",
    "Universe": 0
  }
]
```

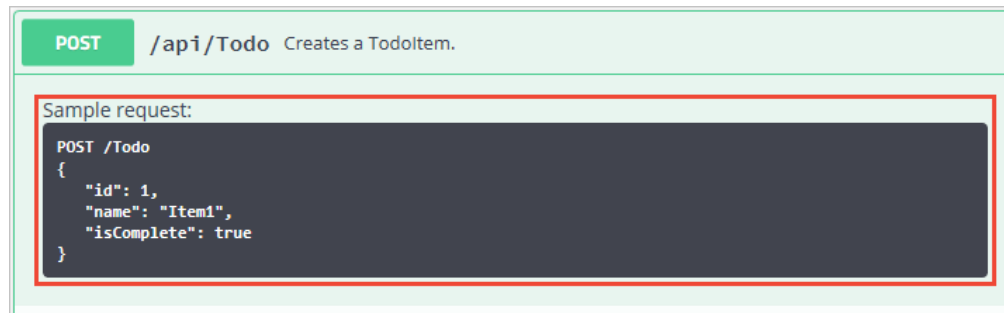
Response Content Type

Try it out!



SZERVIZ ALAPÚ ADATKISZOLGÁLÁS TESZTELÉS SWAGGER-VEL

- Swashbuckle mint eszköz



SZERVIZ ALAPÚ ADATKISZOLGÁLÁS

AZURE APP SERVICE

- **Azure App Service Web Apps**

REST API-kat és mobilháttereket üzemeltető szolgáltatás.

Lehet .NET, .NET Core, Java, Ruby, Node.js, PHP, vagy Python.

Egyszerűen futtatható és méretezhet alkalmazások.

SZERVIZ ALAPÚ ADATKISZOLGÁLÁS

AZURE APP SERVICE

- **Tulajdonságok**

- biztonság
 - terheléselosztás
 - automatikus skálázás
 - automatikus kezelés

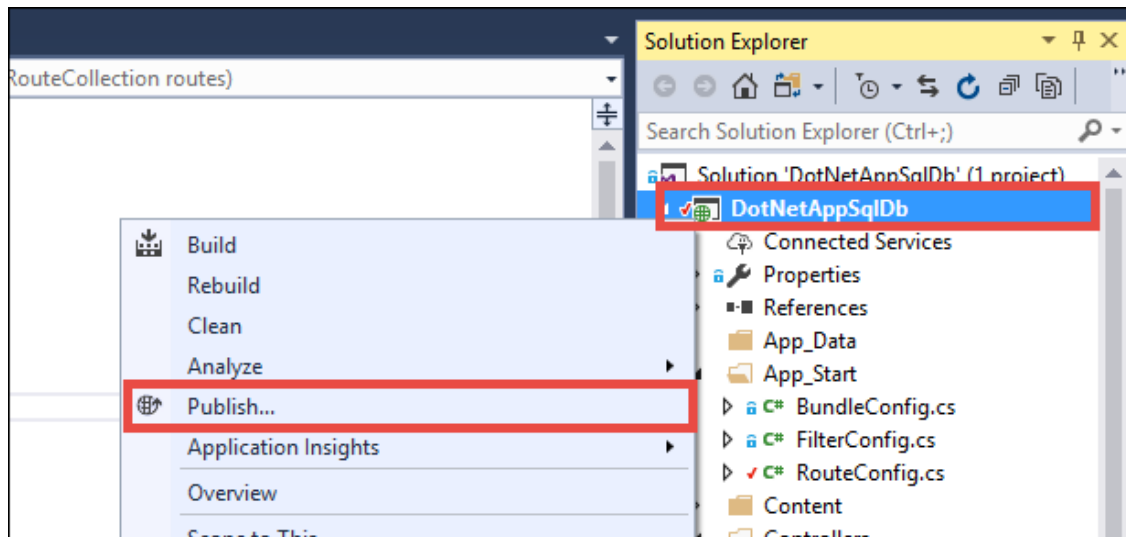
- **DevOps-képességek**

- Folyamatos üzembe helyezés VSTS-ről
 - GitHubról,
 - Docker Hubról

SZERVIZ ALAPÚ ADATKISZOLGÁLÁS

WEB API TELEPÍTÉS AZURE-BA

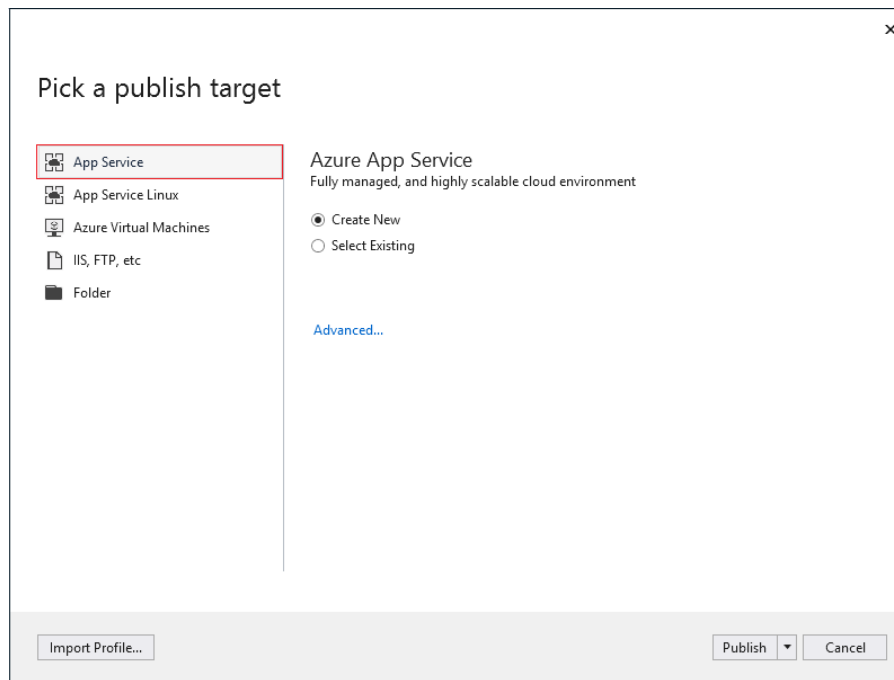
- Visual Studio Publish funkciójának használata



SZERVIZ ALAPÚ ADATKISZOLGÁLÁS

WEB API TELEPÍTÉS AZURE-BA

- Visual Studio Publish funkciójának használata



SZERVIZ ALAPÚ ADATKISZOLGÁLÁS

FELADAT: AZURE WEB API TESZTELÉSE

- Web Api tesztelése Postman segédeszközzel

<https://login.microsoftonline.com/{{tenantId}}/oauth2/token>

▼ Get AAD Token

Add a description

POST ▼ `https://login.microsoftonline.com/{{tenantId}}/oauth2/token`

Auth Headers (1) **Body** ● Pre-req. Tests ●

☐ form-data ☒ x-www-form-urlencoded ☐ raw ☐ binary

	Key	Value	Description
☒	grant_type	client_credentials	
☒	client_id	{{clientId}}	
☒	client_secret	{{clientSecret}}	
☒	resource	{{resource}}	

XAMARIN CROSS-PLATFORM FEJLESZTÉS

CROSS-PLATFORM ESZKÖZÖK

- **Cél**

Egy technológia több platformra
Közös architektúra és kódbázis

- **Típusok**

HTML alapú pl. Cordova
Natív pl. Xamarin

XAMARIN CROSS-PLATFORM FEJLESZTÉS

XAMARIN

- **Ami megvalósul**

Natív alkalmazásfejlesztés

C# programozási nyelv

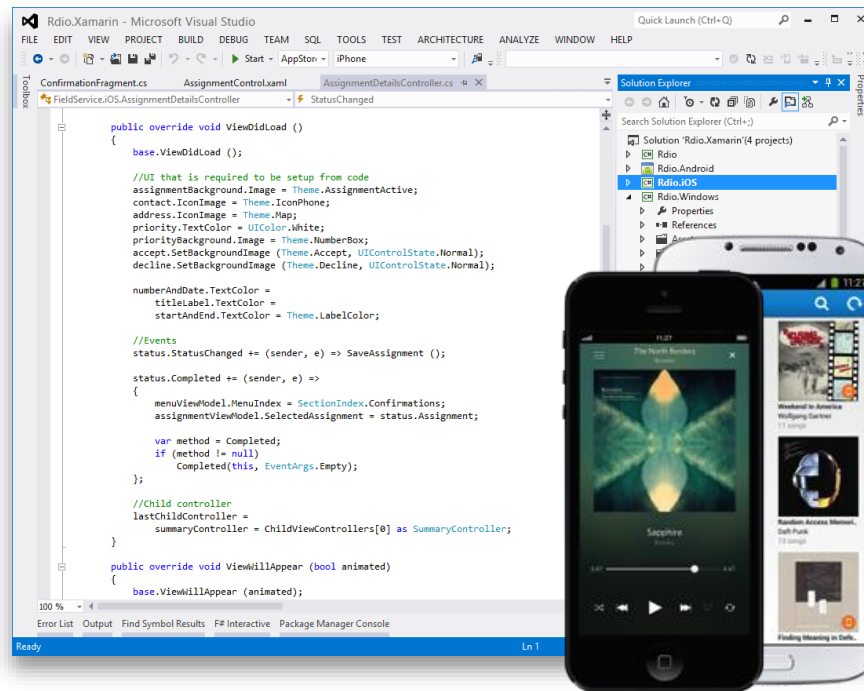
Microsoft támogatás

Visual Studio integráció



XAMARIN CROSS-PLATFORM FEJLESZTÉS

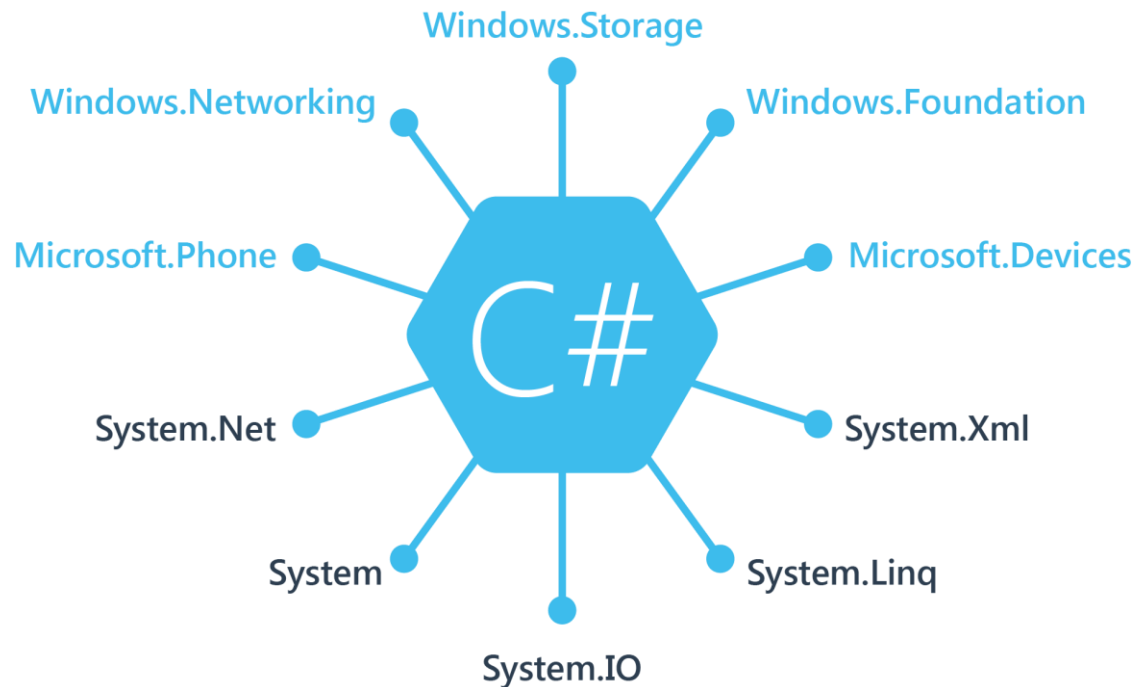
XAMARIN



XAMARIN CROSS-PLATFORM FEJLESZTÉS

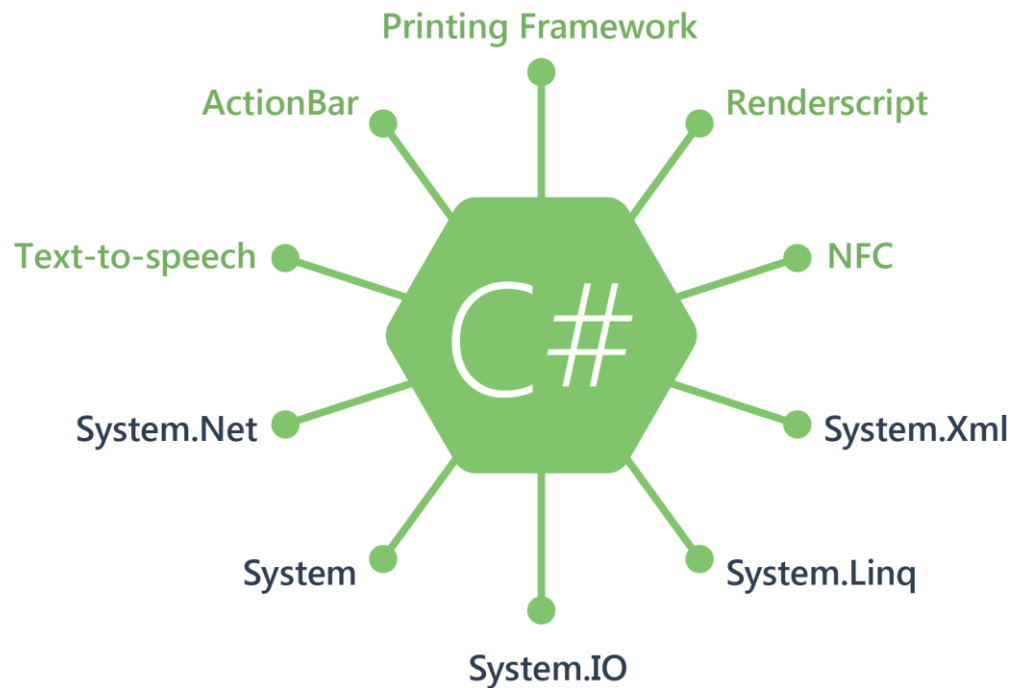
WINDOWS APIS

- Alapértelmezett könyvtárak kiegészítve



XAMARIN CROSS-PLATFORM FEJLESZTÉS ANDROID APIS

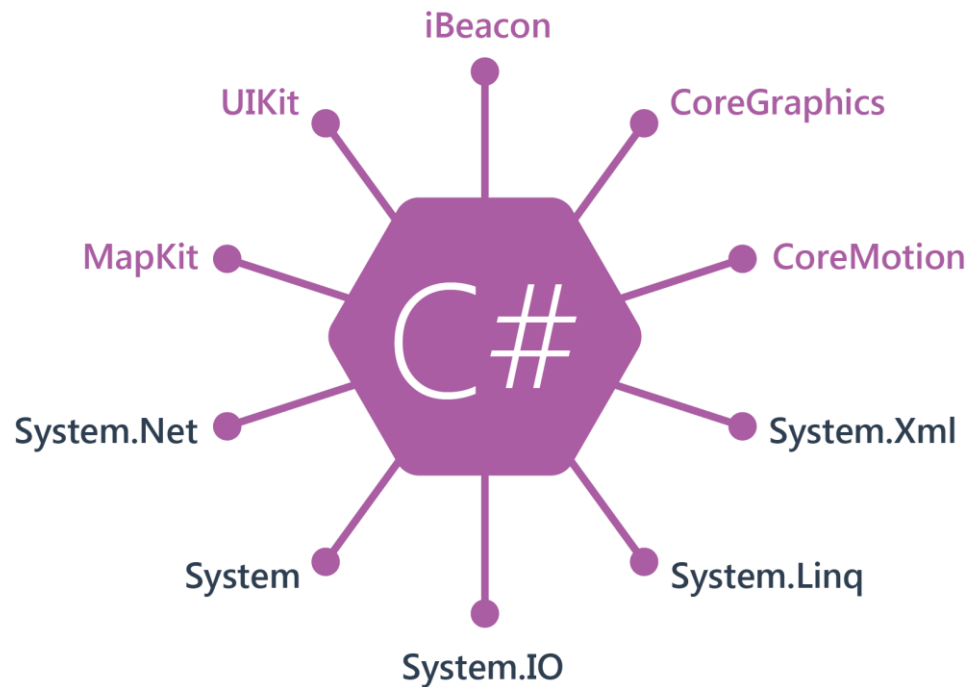
- 100% Adroid API lefedettség



XAMARIN CROSS-PLATFORM FEJLESZTÉS

IOS APIS

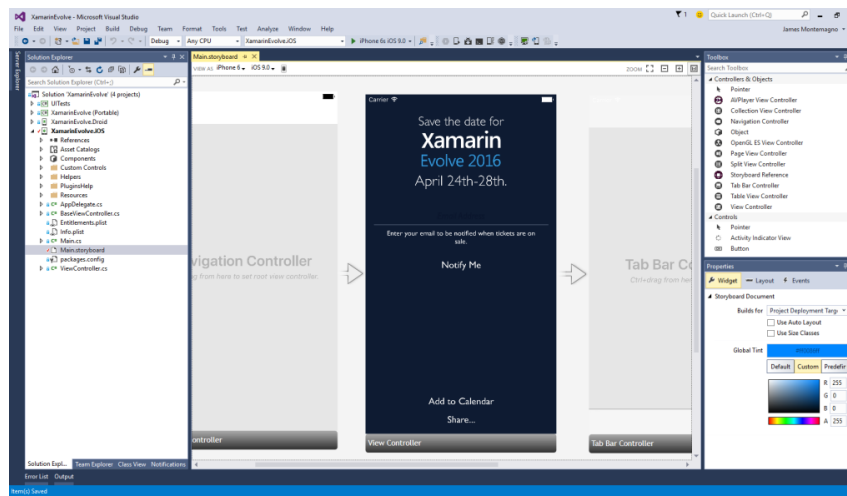
- 100% iOS API lefedettség



XAMARIN CROSS-PLATFORM FEJLESZTÉS FEJLESZTŐESZKÖZÖK

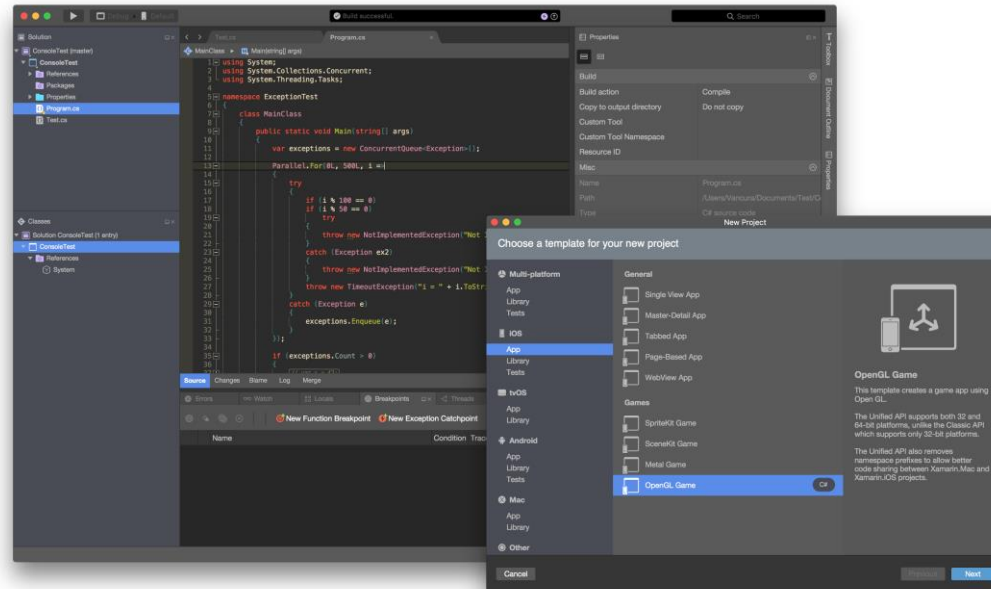
- Visual Studio

Teljeskörű integráció



XAMARIN CROSS-PLATFORM FEJLESZTÉS FEJLESZTŐESZKÖZÖK

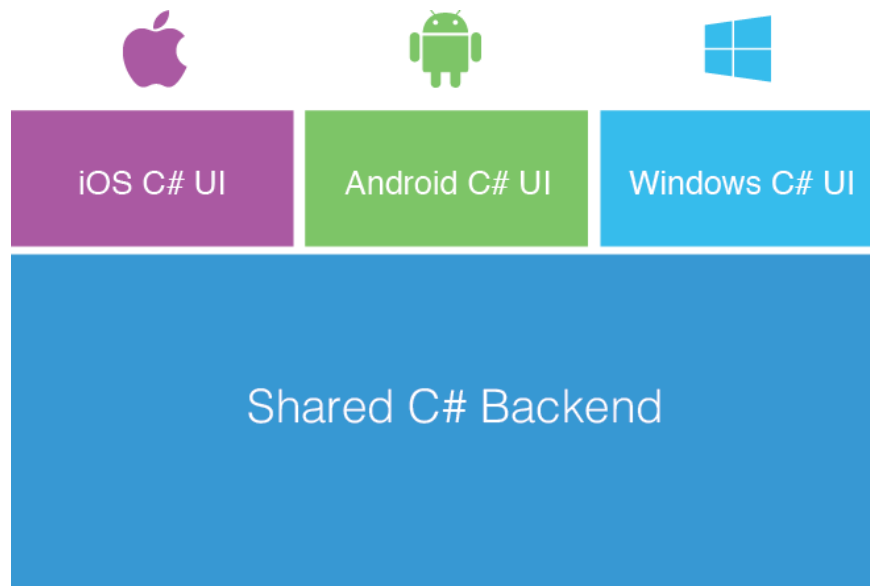
- Xamarin Studio



XAMARIN CROSS-PLATFORM FEJLESZTÉS

XAMARIN FORMS

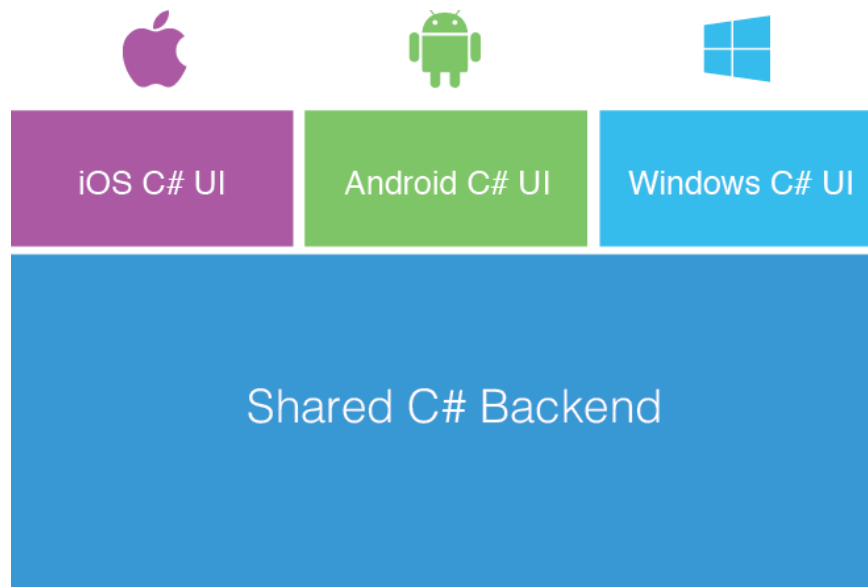
Traditional Xamarin approach



XAMARIN CROSS-PLATFORM FEJLESZTÉS

XAMARIN FORMS

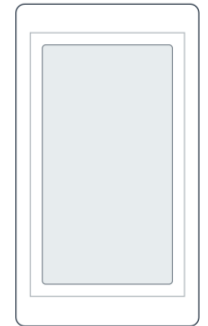
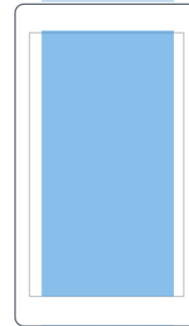
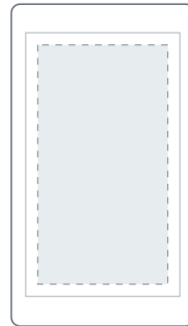
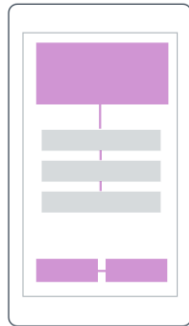
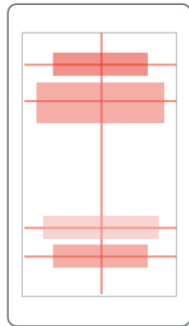
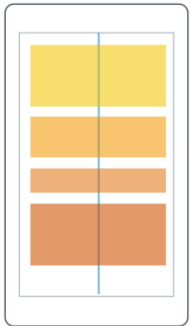
With Xamarin.Forms:
more code-sharing, native controls



XAMARIN CROSS-PLATFORM FEJLESZTÉS

LAYOUT ÉS CONTROLS

Layout elrendezési lehetőségek



XAMARIN CROSS-PLATFORM FEJLESZTÉS

LAYOUT ÉS CONTROLS

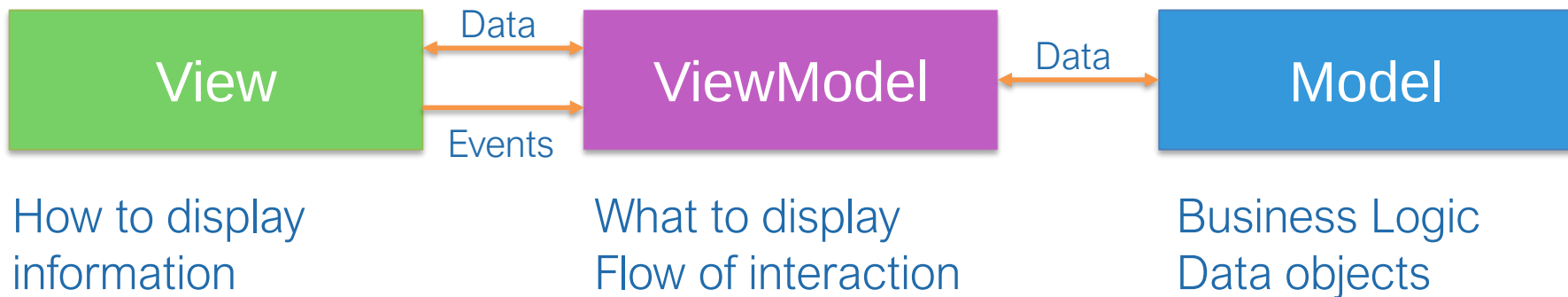
- Layout elrendezési lehetőségek

ActivityIndicator	BoxView	Button	DatePicker	Editor
Entry	Image	Label	ListView	Map
OpenGLView	Picker	ProgressBar	SearchBar	Slider
Stepper	TableView	TimePicker	WebView	EntryCell
ImageCell	SwitchCell	TextCell	ViewCell	

XAMARIN CROSS-PLATFORM FEJLESZTÉS

MODEL-VIEW-VIEWMODEL

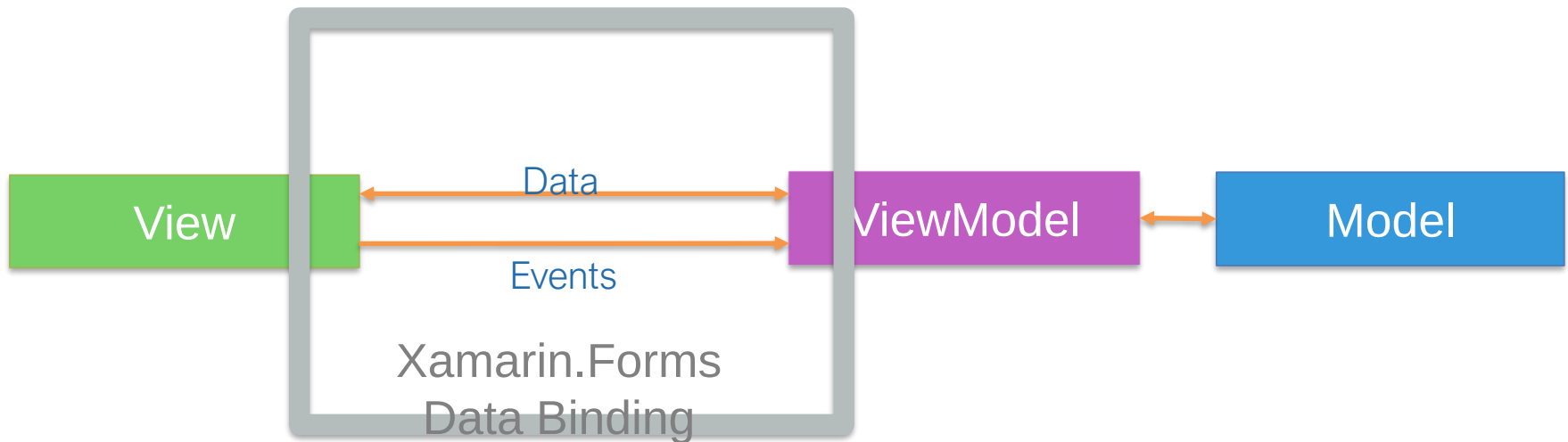
- Általános működés



XAMARIN CROSS-PLATFORM FEJLESZTÉS

MODEL-VIEW-VIEWMODEL

- Xamarin.Forms működés



XAMARIN CROSS-PLATFORM FEJLESZTÉS

DATA BINDING

- Adatkapcsolatok létrehozása

```
public class UserViewModel : INotifyPropertyChanged
{
    #region INotifyPropertyChanged implementation
    public event PropertyChangedEventHandler PropertyChanged;
    #endregion

    public void OnPropertyChanged(string name)
    {
        if (PropertyChanged == null)
            return;

        PropertyChanged (this, new PropertyChangedEventArgs (name));
    }
}
```

```
private string firstname = string.Empty;
public string FirstName
{
    get { return firstname; }
    set {
        if (firstname == value)
            return;

        firstname = value;
        OnPropertyChanged ("FirstName");
    }
}
```

XAMARIN CROSS-PLATFORM FEJLESZTÉS

DATA BINDING

- Adatkapcsolatok létrehozása

```
Label firstName = new Label ();  
firstName.SetBinding (Label.TextProperty, "FirstName");
```

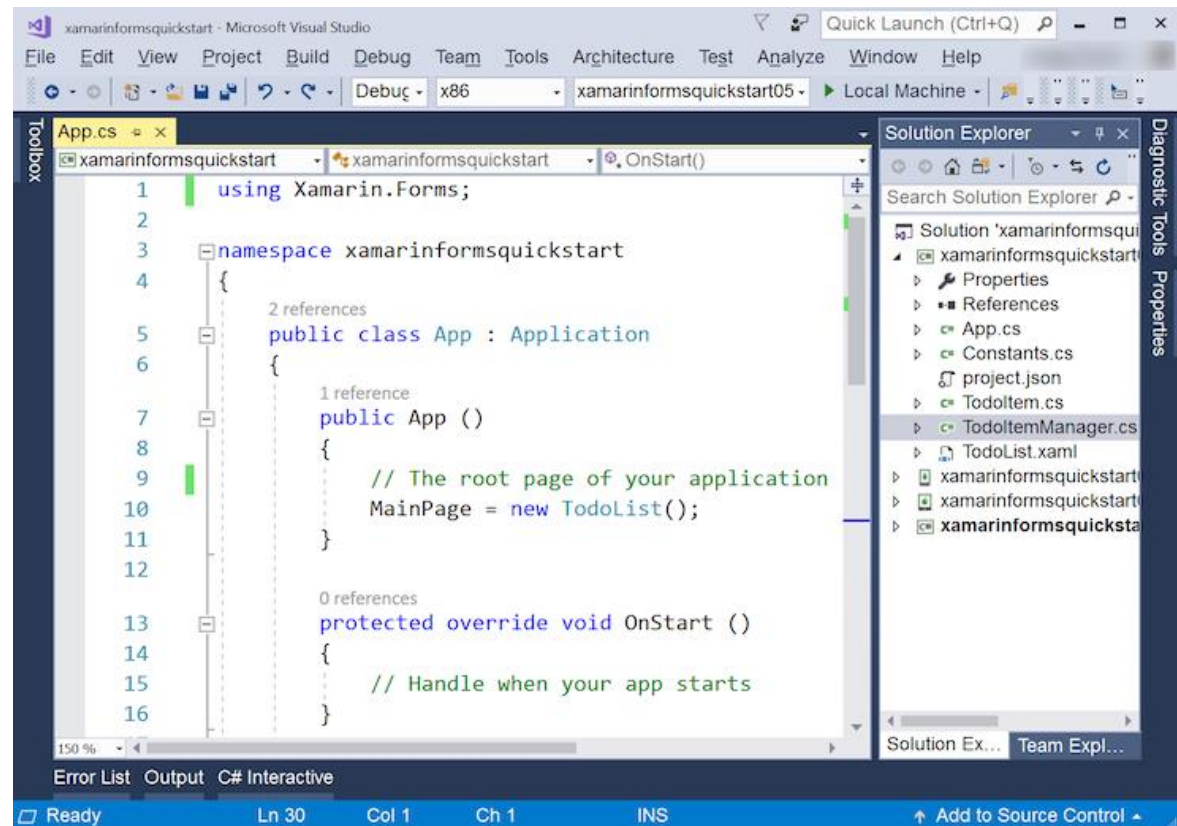
```
Entry firstEntry = new Entry ();  
firstEntry.SetBinding<UserViewModel> (Entry.TextProperty, vm => vm.FirstName, BindingMode.TwoWay);
```

XAMARIN CROSS-PLATFORM FEJLESZTÉS

FELADAT: PÉLDAKÓD TESZTELÉS EMULÁTORRAL

- Visual Studio segítségével „Hello World” alkalmazás készítés

Új Xamarin.Forms projekt léterhozás
Emulator indítás



XAMARIN FORMS HASZNÁLATBAN

TERVEZÉS

- **Tervezési folyamat**

It is advisable to create a of road-map for what you want to achieve before you start coding. This is especially true for cross-platform development, where you are building functionality that will be exposed in multiple ways. Starting with a clear idea of what you're building saves time and effort later in the development cycle.

- **Követelmények**

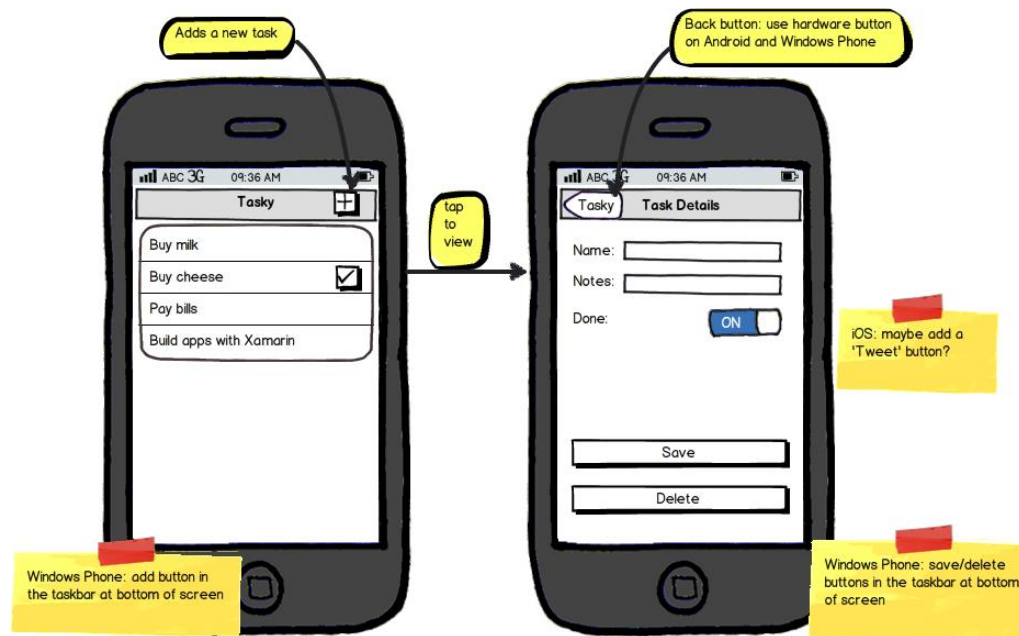
View a list of tasks

Add, edit and delete tasks

Set a task's status to 'done'

XAMARIN FORMS HASZNÁLATBAN TERVEZÉS

- UI tervezés



XAMARIN FORMS HASZNÁLATBAN

ADATMODEL

- **Adatbáziskezelő kiválasztása**

Knowing what data needs to be stored will help determine which persistence mechanism to use. Cross-Platform Data Access information is needed about the available storage mechanisms and help deciding between them.

SQLite.NET is a proper choice.

XAMARIN FORMS HASZNÁLATBAN

ADATMODEL

- **SQLite adatbáziskezelő**

Az SQLite önálló, kisméretű (kb. 500 KiB), C forrású programkönyvtárként (library) megvalósított ACID-kompatibilis relációs adatbázis-kezelő rendszer, illetve adatbázismotor.

- **TaskItem osztály**

Name – String

Notes – String

Done – Boolean

XAMARIN FORMS HASZNÁLATBAN

FUNKCIÓK

- **Alap funkciók**

List all tasks – to display the main screen list of all available tasks

Get one task – when a task row is touched

Save one task – when a task is edited

Delete one task – when a task is deleted

Create empty task – when a new task is created

- **Megvalósítási módszer kiválasztása**

To achieve code reuse, this API should be implemented once in the Portable Class Library

XAMARIN FORMS HASZNÁLATBAN

MEGVALÓSÍTÁS

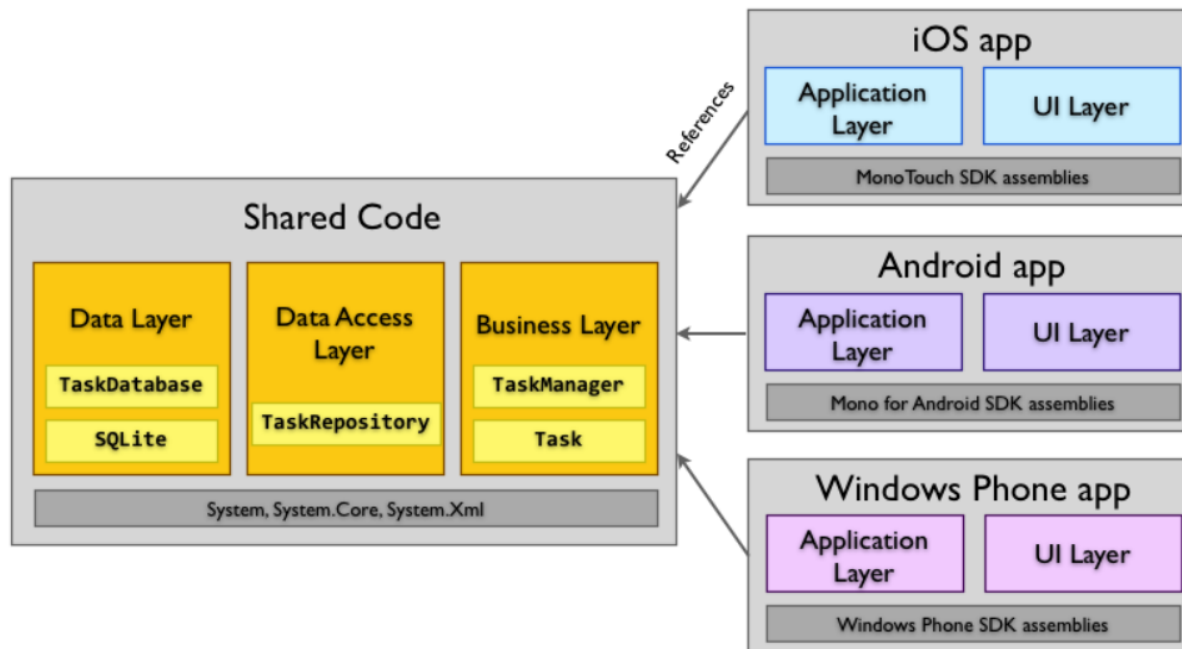
- **Common Code**

A common project that contains re-useable code to store the task data; expose a Model class and an API to manage the saving and loading of data.

- **Platform-specific Code**

Platform-specific projects that implement a native UI for each operating system, utilizing the common code as the 'back end'.

XAMARIN FORMS HASZNÁLATBAN MEGVALÓSÍTÁS



XAMARIN FORMS HASZNÁLATBAN

MEGOSZTOTT KÓD

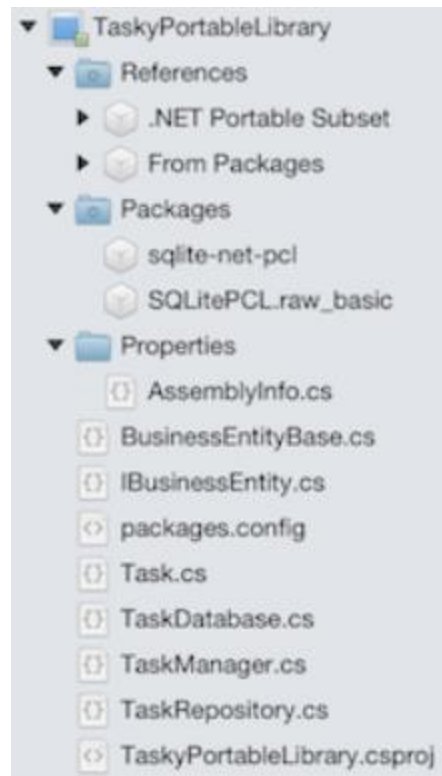
- **Portable uses the Portable Class Library**

All common code, including the data access layer, database code and contracts, is placed in the library project.

The complete PCL project is illustrated below. All of the code in the portable library is compatible with each targeted platform. When deployed, each native app will reference that library.

XAMARIN FORMS HASZNÁLATBAN

MEGOSZTOTT KÓD



XAMARIN FORMS HASZNÁLATBAN

ADATRÉTEG

- **Data Layer (DL)**

The Data Layer contains the code that does the physical storage of data – whether to a database, flat files or other mechanism. The Tasky data layer consists of two parts: the SQLite-NET library and the custom code added to wire it up.

XAMARIN FORMS HASZNÁLATBAN

ADATRÉTEG

- **Interface to Generalize Data Access**

The Data Layer takes a dependency on BusinessIdentity so that it can implement abstract data access methods that require a primary key. Any Business Layer class that implements the interface can then be persisted in the Data Layer.

- **Data Access Layer (DAL)**

The ItemRepository class encapsulates the data storage mechanism with a strongly-typed API that allows TaskItem objects to be created, deleted, retrieved and updated.

XAMARIN FORMS HASZNÁLATBAN

ÜZLETI RÉTEG

- **Business Layer (BL)**

The Business Layer implements the Model classes and a Facade to manage them. In Tasky the Model is the TaskItem class and TaskItemManager implements the Façade pattern to provide an API for managing TaskItems

XAMARIN FORMS HASZNÁLATBAN

ÜZLETI RÉTEG

- **API for Platform-Specific Code**

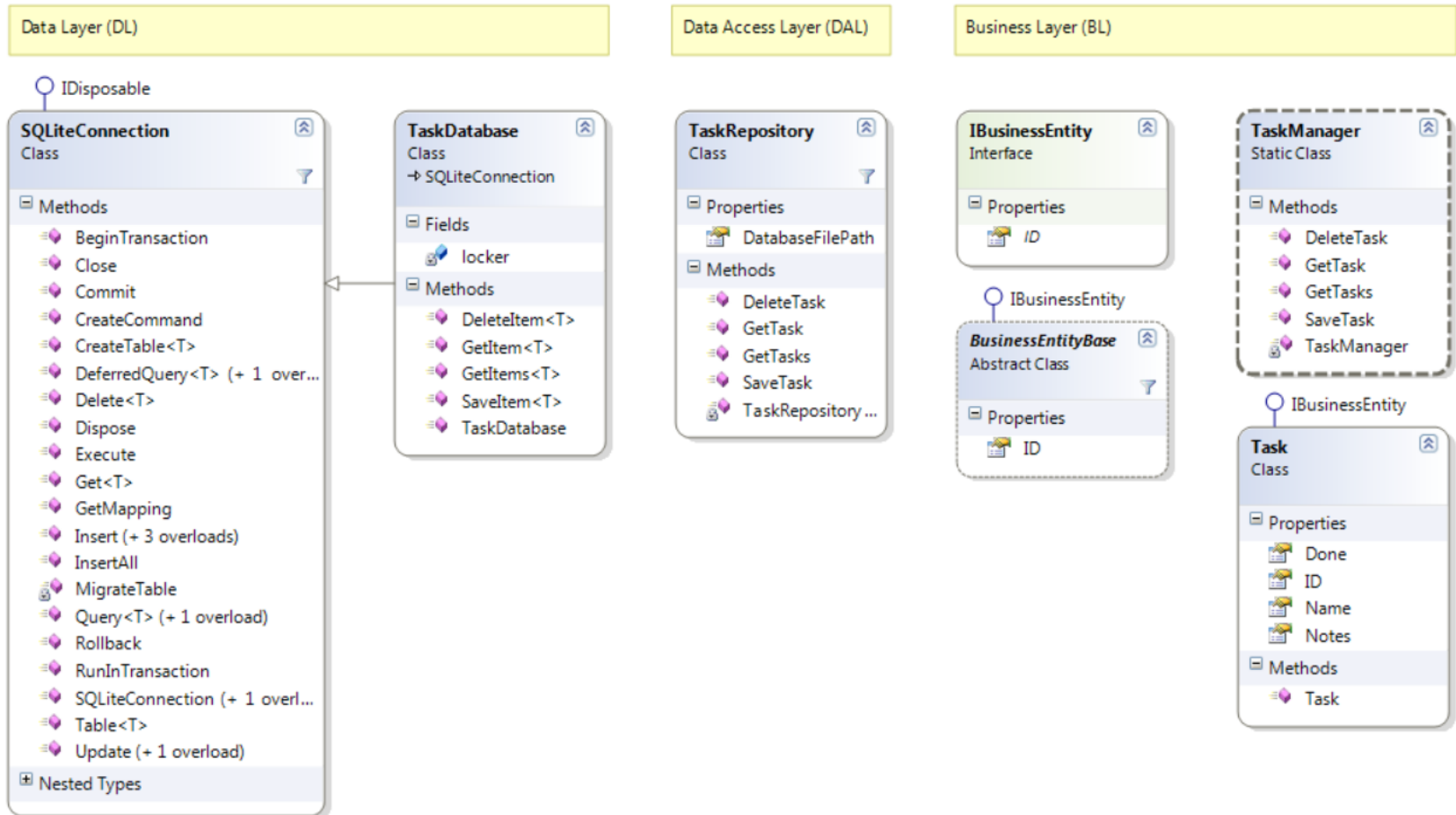
Once the common code has been written, the user interface must be built to collect and display the data exposed by it. The `TaskItemManager` class implements the Facade pattern to provide a simple API for the application code to access.

The code written in each platform-specific project will generally be tightly coupled to the native SDK of that device, and only access the common code using the API defined by the `TaskItemManager`. This includes the methods and business classes it exposes, such as `TaskItem`.

Images are not shared across platforms but added independently to each project. This is important because each platform handles images differently, using different file names, directories and resolutions.

XAMARIN FORMS HASZNÁLATBAN

RÉTEGEK ÁTTEKINTÉSE



XAMARIN FORMS HASZNÁLATBAN

IOS APP

- **Application Layer (AL)**

The Application Layer contains platform-specific classes required to 'bind' the objects exposed by the PCL to the UI. The iOS-specific application has two classes to help display tasks



XAMARIN FORMS HASZNÁLATBAN

IOS APP

- **User Interface Layer (UI)**

AppDelegate – Contains calls to the Appearance API to style the fonts and colors used in the application. Tasky is a simple application so there are no other initialization tasks running in FinishedLaunching

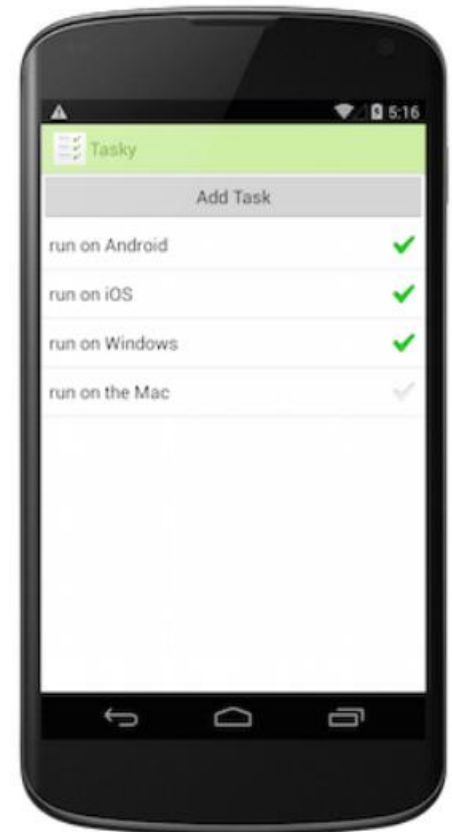
Screens – subclasses of UIViewController that define each screen and its behavior. Screens tie together the UI with Application Layer classes and the common API (TaskItemManager). In this example the screens are created in code, but they could have been designed using Xcode's Interface Builder or the storyboard designer.

Images – Visual elements are an important part of every application. Tasky has splash screen and icon images, which for iOS must be supplied in regular and Retina resolution.

XAMARIN FORMS HASZNÁLATBAN ANDROID APP

- **Application Layer (AL)**

Similar to the iOS version we looked at earlier, the Application Layer in the Android version contains platform-specific classes required to 'bind' the objects exposed by the Core to the UI.



XAMARIN FORMS HASZNÁLATBAN

ANDROID APP

- **User Interface (UI)**

The Android app's User Interface Layer is a combination of code and XML markup.

Resources/Layout – screen layouts and the row cell design implemented as AXML files. The AXML can be written by hand, or laid-out visually using the Xamarin UI Designer for Android.

Resources/Drawable – images (icons) and custom button.

Screens – Activity subclasses that define each screen and its behavior. Ties together the UI with Application Layer classes and the common API

XAMARIN FORMS HASZNÁLATBAN

FELADAT: ALKALMAZÁS LÉTREHOZÁSA

- **Feladatkezelő alkalmazás megvalósítása Android és iOS platformra**

Adatok tárolása lokális SQLite adatbázisban.

Xamarin.Forms portable osztályok használata

XAMARIN FORMS HASZNÁLATBAN

FELADAT: ALKALMAZÁS LÉTREHOZÁSA

- **Adatosztály**

```
public class TaskDialog {  
    [Entry("task name")]  
    public string Name { get; set; }  
    [Entry("other task info")]  
    public string Notes { get; set; }  
    [Entry("Done")]  
    public bool Done { get; set; }  
    [Section ("")]  
    [OnTap ("SaveTask")] // method in HomeScreen  
    [Alignment (UITextAlignment.Center)]  
    public string Save;  
    [Section ("")]  
    [OnTap ("DeleteTask")] // method in HomeScreen  
    [Alignment (UITextAlignment.Center)]  
    public string Delete;  
}
```

XAMARIN ADAT KOMMUNIKÁCIÓ

WEB API

- **HttpClient**

GET – this operation is used to retrieve data from the web service.

POST – this operation is used to create a new item of data on the web service.

PUT – this operation is used to update an item of data on the web service.

PATCH – this operation is used to update an item of data on the web service by describing a set of instructions about how the item should be modified. This verb is not used in the sample application.

DELETE – this operation is used to delete an item of data on the web service.

XAMARIN ADAT KOMMUNIKÁCIÓ

WEB API

- **JSON messages**

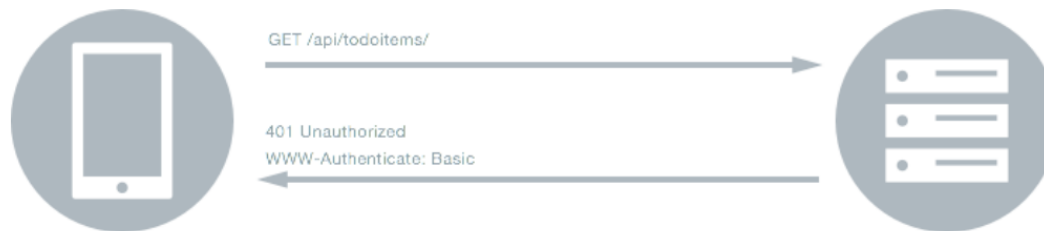
RESTful web services typically use JSON messages to return data to the client. JSON is a text-based data-interchange format that produces compact payloads, which results in reduced bandwidth requirements when sending data. The sample application uses the open source Newtonsoft JSON.NET library to serialize and deserialize messages.

XAMARIN ADAT KOMMUNIKÁCIÓ

WEB API AUTENTIKÁCIÓ

- **Authenticating Users over HTTP**

If a web service receives a request for a protected resource, it rejects the request with an HTTP status code 401 (access denied) and sets the WWW-Authenticate response header, as shown in the following diagram:

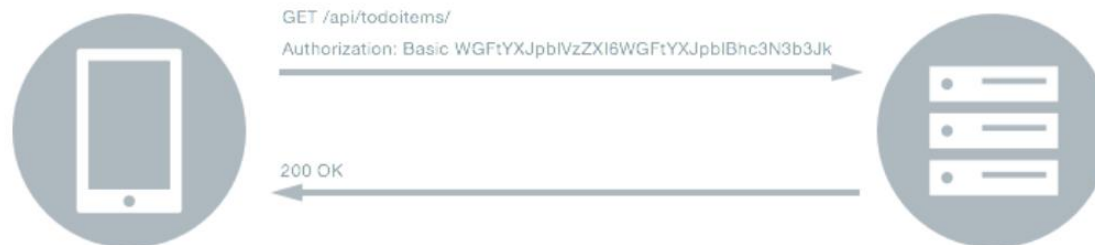


XAMARIN ADAT KOMMUNIKÁCIÓ

WEB API AUTENTIKÁCIÓ

- **Authenticating Users over HTTP**

If a web service receives a request for a protected resource, with the Authorization header correctly set, the web service responds with an HTTP status code 200, which indicates that the request succeeded and that the requested information is in the response. This scenario is shown in the following diagram:



XAMARIN ADAT KOMMUNIKÁCIÓ

WEB API GET

- **Creating the HttpClient Object**

```
HttpClient client;  
public RestService ()  
{  
    client = new HttpClient ();  
    client.MaxResponseContentBufferSize = 256000;  
}
```

XAMARIN ADAT KOMMUNIKÁCIÓ

WEB API GET

- **Retrieving Data**

```
var response = await client.GetAsync (uri);  
if (response.IsSuccessStatusCode) {  
    var content = await response.Content.ReadAsStringAsync ();  
    Items = JsonConvert.DeserializeObject <List<TodoItem>> (content);  
}
```

XAMARIN ADAT KOMMUNIKÁCIÓ

WEB API POST

- **Creating Data**

```
var json = JsonConvert.SerializeObject (item);  
var content = new StringContent (json, Encoding.UTF8, "application/json");
```


XAMARIN ADAT KOMMUNIKÁCIÓ

WEB API POST

- **Posting Data**

```
HttpResponseMessage response = null;
if (isNewItem) {
    response = await client.PostAsync (uri, content);
}
...

if (response.IsSuccessStatusCode) {
    Debug.WriteLine (@"          TodoItem successfully saved.");
}
```

XAMARIN ADAT KOMMUNIKÁCIÓ

MQTT KLIENS

- **System.Net.Mqtt**

System.Net.Mqtt is a light weight and simple implementation of the MQTT protocol version 3.1.1, written entirely in C# and divided in two libraries: System.Net.Mqtt and System.Net.Mqtt.Server.

The foundation of the System.Net.Mqtt libraries is to provide an intuitive and very easy to use API, hiding most of the protocol concepts that are not needed to be exposed, letting the consumers to just focus on the main protocol messages that are: CONNECT, SUBSCRIBE, UNSUBSCRIBE, PUBLISH, DISCONNECT.

XAMARIN ADAT KOMMUNIKÁCIÓ

MQTT KLIENS

- **Connecting an MQTT client**

```
var configuration = new MqttConfiguration();  
var client = await MqttClient.CreateAsync("192.168.1.10", configuration);  
var sessionState = await client.ConnectAsync (new MqttCredentials(clientId: "foo"));
```

XAMARIN ADAT KOMMUNIKÁCIÓ

MQTT MEGVALÓSÍTÁS

- **Creating an MQTT client**

```
var configuration = new MqttConfiguration {  
    BufferSize = 128 * 1024,  
    Port = 55555, KeepAliveSecs = 10, WaitTimeoutSecs = 2,  
    MaximumQualityOfService = MqttQualityOfService.AtMostOnce,  
    AllowWildcardsInTopicFilters = true };  
var client = await MqttClient.CreateAsync("192.168.1.10", configuration);  
var sessionState = await client.ConnectAsync (new MqttCredentials(clientId: "foo"), cleanSession:  
true);
```

XAMARIN ADAT KOMMUNIKÁCIÓ

MQTT MEGVALÓSÍTÁS

- **Subscribing and unsubscribing to topics**

```
await client.SubscribeAsync("foo/bar/topic1", MqttQualityOfService.AtMostOnce); //QoS0  
await client.SubscribeAsync("foo/bar/topic2", MqttQualityOfService.AtLeastOnce); //QoS1  
await client.SubscribeAsync("foo/bar/topic3", MqttQualityOfService.ExactlyOnce); //QoS2
```

```
await client.UnsubscribeAsync("foo/bar/topic1");
```

XAMARIN ADAT KOMMUNIKÁCIÓ

MQTT SZENZOR ADAT FOGADÁSA

- **Adatok fogadása**

client

.MessageStream

.Subscribe(msg => Console.WriteLine(\$"Message received in topic {msg.Topic}"));

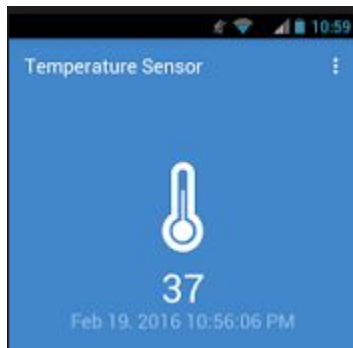
XAMARIN ADAT KOMMUNIKÁCIÓ

MQTT SZENZOR ADAT FOGADÁSA

- Adatok fogadása

client

```
.MessageStream  
.Where(msg => msg.Topic == "foo/bar/topic2")  
.Subscribe(msg => Console.WriteLine($"Message received in topic {msg.Topic}"));`
```



XAMARIN ADAT KOMMUNIKÁCIÓ

PUSH ÉRTESÍTÉSEK

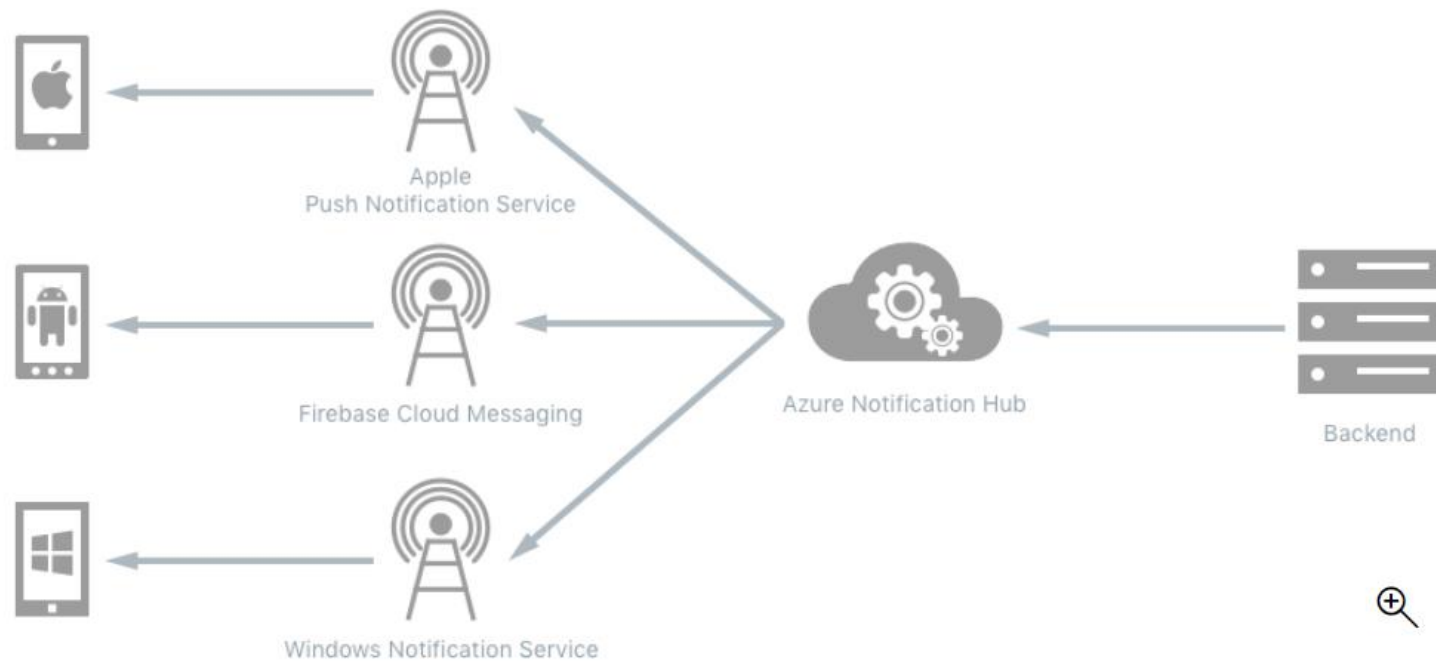
- **Azure Push Notification Hub**

A push notification is used to deliver information, such as a message, from a backend system to an application on a mobile device to increase application engagement and usage. The notification can be sent at anytime, even when the user is not actively using the targeted application

XAMARIN ADAT KOMMUNIKÁCIÓ

PUSH ÉRTESEÍTÉSEK

- Azure Push Notification Hub



XAMARIN ADAT KOMMUNIKÁCIÓ ÉRTESÍTÉSEK ELŐKÉSZÍTÉSE

- **Azure and Platform Notification System**

Creating an Azure Mobile Apps instance.

Configure a notification hub.

Update the Azure Mobile Apps instance to send push notifications.

Register with each PNS.

Configure the notification hub to communicate with each PNS.

XAMARIN ADAT KOMMUNIKÁCIÓ ÉRTESÍTÉSEK ELŐKÉSZÍTÉSE

- **Android**

Firestore Cloud Messaging (FCM)

Register for FCM. A Server API key and a Client ID are automatically generated, and packed in a google-services.json

Configure the notification hub to communicate with FCM.

XAMARIN ADAT KOMMUNIKÁCIÓ

ÉRTESÍTÉS FOGADÁSA

- **Implementing the Firebase Instance ID Service**

```
[Service]
[IntentFilter(new[] { "com.google.firebase.INSTANCE_ID_EVENT" })]
public class FirebaseRegistrationService : FirebaseInstanceIdService
{
    const string TAG = "FirebaseRegistrationService";

    public override void OnTokenRefresh()
    {
        var refreshedToken = FirebaseInstanceId.Instance.Token;
        Log.Debug(TAG, "Refreshed token: " + refreshedToken);
        SendRegistrationTokenToAzureNotificationHub(refreshedToken);
    }
}
```

XAMARIN ADAT KOMMUNIKÁCIÓ

ÉRTESÍTÉS FOGADÁSA

- Implementing the Firebase Instance ID Service

```
void SendRegistrationTokenToAzureNotificationHub(string token)
{
    // Update notification hub registration
    Task.Run(async () =>
    {
        await
        AzureNotificationHubService.RegisterAsync(TodoItemManager.DefaultManager.CurrentClient.Get
        Push(), token);
    });
}
```

XAMARIN ADAT KOMMUNIKÁCIÓ

FELADAT: ALKALMAZÁS LÉTREHOZÁSA

- **Xamarin Forms alkalmazás létrehozása**

Menü 1: RestAPI hívás megvalósítása

Menü 2: MQTT LED kapcsolása

- **Push Notification fogadása**

Android alapon Azure Push Notification elkészítése

1. Bevezetés az IoT fejlesztésekbe

- 1.1 Az IoT világa
- 1.2 Trendek az IoT világában
- 1.3 IoT eszközök
- 1.4 ESP8266 bemutatása
- 1.5 ESP8266 NodeMCU
- 1.6 Ardunio IDE bemutatás
- 1.7 Ardunio IDE fordítás, feltöltés
- 1.8 Ardunio IDE első lépések
- 1.9 Ardunio IDE debug
- 1.10 ESP8266 WiFi könyvtár

Feladat 1: WiFi kapcsolat megvalósítása

Feladat 2: Alkalmazás készítése Ardunio IDE használatával

2. ESP végponti eszköz programozása

2.1 GPIO alapok

2.2 GPIO portok

2.3 GPIO használat

2.4 GPIO írás/olvasás

2.5 Led villogtatás

2.6 Web szerver könyvtár

2.7 Web szerver indítása

2.8 Web szerver response

2.9 Web szerver request

2.10 Web szerver request/response

Feladat 1: Led kapcsolás böngészőből

Feladat 2: Led villogtatás böngészőből

3. Lokális adatgyűjtés és folyamat vezérlés (vizuális programozási környezetben)

3.1 Raspberry Pi mint központ

3.2 Node-RED folyamat vezérlő

3.3 Node-RED input csomópontok

3.4 Node-RED output csomópontok

3.5 Node-RED funkció csomópontok

3.6 Node-RED social csomópontok

3.7 Node-RED storage csomópontok

3.8 Node-RED Dashboard

3.9 Node-RED Dashboard készítés

3.10 Node-RED Dashboard megjelenítése

Feladat 1: LED kapcsolás Node-RED-vel

Feladat 2: Node-RED Dashboard készítés

4. IoT rendszerek adatkommunikációja

4.1 Cloud platformok

4.2 Kommunikációs protokollok

4.3 MQTT protokoll

4.4 MQTT működése

4.5 MQTT brókerek

4.6 MQTT topikok és fák

4.7 MQTT kliensek

4.8 MQTT adatküldés előkészítése

4.9 MQTT adatküldés megvalósítás

4.10 Node-RED MQTT csomópont

Feladat 1: Adatfogadás MQTT-n Node-RED-ben

Feladat 2: Adatküldés MQTT-n Node-RED-ben

5. Adatfogadás IoT eszköz oldalon

5.1 Adattartalmak

5.2 Topik elnevezési stratégia

5.3 JSON parser

5.4 MQTT biztonság

5.5 MQTT tesztelési lehetőségek

5.6 MQTT kapcsoló vezérlés

5.7 MQTT adatfogadás előkészítése

5.8 MQTT Callback implementálása

5.9 Node-RED MQTT adatküldés előkészítése

5.10 Node-RED Dashboard előkészítése

Feladat 1: Led kapcsolás MQTT-n keresztül

Feladat 2: Adatfogadás MQTT-n keresztül

6. Azure mint platform

6.1 Azure felépítése

6.2 Azure szolgáltatások

6.3 Azure fiók

6.4 Azure Event HUB

6.5 Azure IoT Hub

6.6 Azure IoT Eszköz

6.7 Adattárolás

6.8 Azure IoT üzenet mentése

6.9 Node-RED Azure csomópontok

6.10 Node-RED Azure kapcsolódás

Feladat 1: Adat küldése Azure felhőbe

Feladat 2: Adat fogadás Azure felhőből

7. Szerviz alapú adatkiszolgálás

7.1 RESTful alkalmazások

7.2 Web API keretrendszer

7.3 Visual Studio fejlesztőeszköz

7.4 Visual Studio projektek

7.5 Visual Studio Web API készítés

7.6 Dokumentálás Swagger-vel

7.7 Swagger használata

7.8 Tesztelés Swagger-rel

7.9 Azure App Service

7.10 Web API telepítés Azure-ba

Feladat 1: Azure Web API tesztelése

Feladat 2: Tesztelés Swagger-rel

8. Xamarin cross-platform fejlesztés

8.1 Cross-platform eszközök

8.2 Xamarin

8.3 Windows APIs

8.4 Android APIs

8.5 IOS APIs

8.6 Fejlesztőeszközök

8.7 Xamarin Forms

8.8 Layout és Controls

8.9 Model-View-ViewModel

8.10 Data Binding

Feladat 1: Példakód tesztelés emulátorral

Feladat 2: Data Binding használata

9. Xamarin Forms használatban

9.1 Tervezés

9.2 Adatmodel

9.3 Funkciók

9.4 Megvalósítás

9.5 Megosztott kód

9.6 Adatréteg

9.7 Üzleti réteg

9.8 Rétegek áttekintése

9.9 iOS App

9.10 Android App

Feladat 1: iOS alkalmazás létrehozása

Feladat 2: Android alkalmazás létrehozása

10. Xamarin adat kommunikáció

10.1 Web API

10.2 Web API autentikáció

10.3 Web API GET

10.4 Web API POST

10.5 MQTT kliens

10.6 MQTT megvalósítás

10.7 MQTT szenzor adat fogadása

10.8 Push értesítések

10.9 Értesítések előkészítése

10.10 Értesítés fogadása

Feladat 1: Webalkalmazás létrehozása

Feladat 2: MQTT szenzor adat fogadása

MŰSZAKI INFORMATIKA

TÉMAKÖRÖK

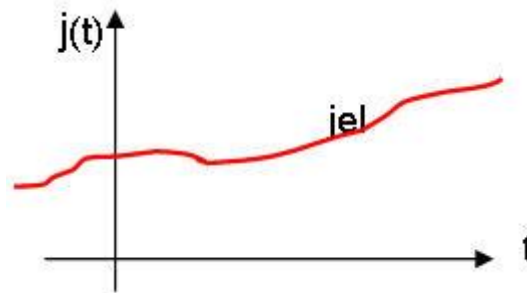
1. Jelek fogalma, osztályozása
2. Jelfeldolgozás alapjai
3. Gyors Fourier transzformáció
4. AD és DA konverzió
5. Digitális jelfeldolgozás
6. Átviteltechnika alapjai
7. Vezetékes átviteltechnika
8. Vezeték nélküli átviteltechnika
9. Irányítás és szabályozástechnika
10. Beágyazott rendszerek

JELEK DEFINÍCIÓJA

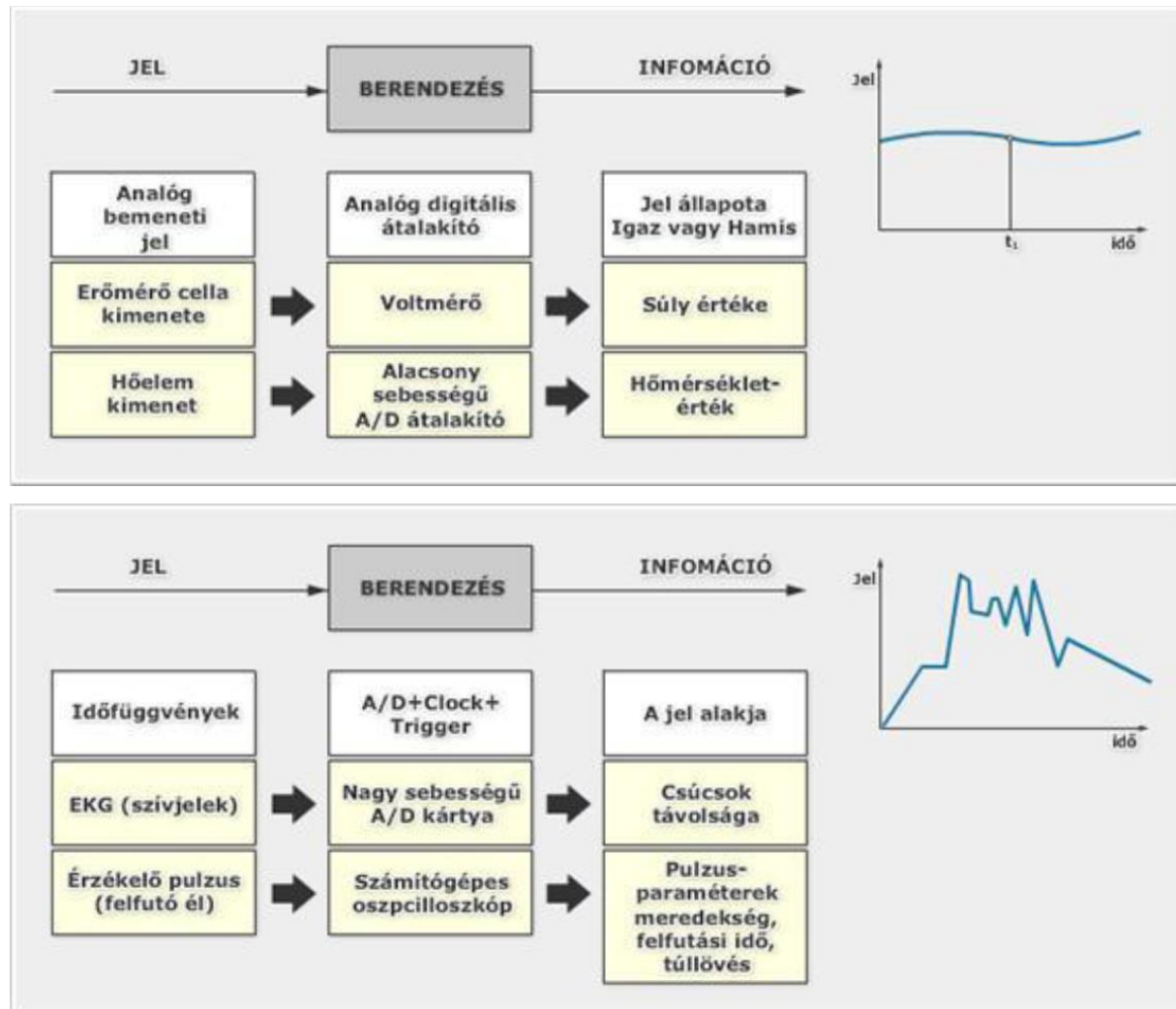
- **Jelnek** nevezzük azokat a fizikai mennyiségeket, amelyek egy fizikai rendszer állapotát írják le a valamely más fizikai mennyiség (pl. hely, idő, hőmérséklet) függvényeként.
- A jel viselkedését általában egyetlen független változó függvényében vizsgáljuk (idő).
- A jel értékkészletét a jel **amplitúdójának** nevezzük
- A jel legfontosabb jellemvonása a jel által hordozott információ
- A jeleket az értelmezési tartományuk és az értékkészletük alapján osztályozzuk.

ANALÓG JELEK

Azokat a jeleket, melyek mind értékkészletükben, mind értelmezési tartományukban folytonosak analóg jeleknek nevezzük.



ANALÓG JELEK TÍPUSAI



PERIODIKUS JELEK

Egy $f(t)$ jelet periodikusnak nevezünk, ha minden egyes t időpontra fenn áll a következő összefüggés:

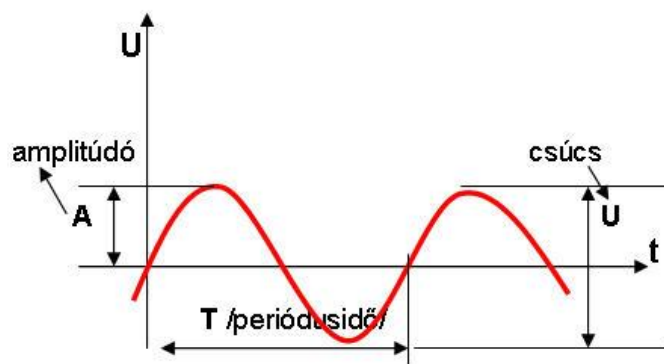
$$f(t) = f(t \pm T)$$

ahol $T > 0$.

Periodikus jel

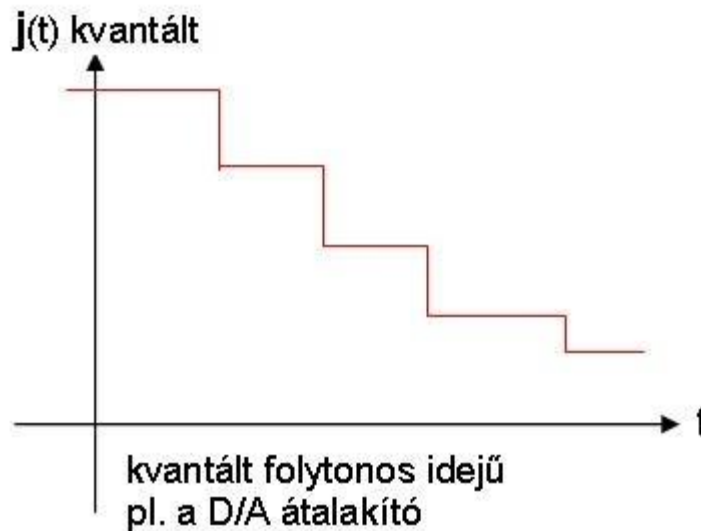
$$x(t) = x(t + n \cdot T) \quad n = -1, 0, 1, 2, 3, \dots$$

$$f = 1/T \quad [1/s = s^{-1} = \text{Hz}] \quad (\text{Hertz})$$



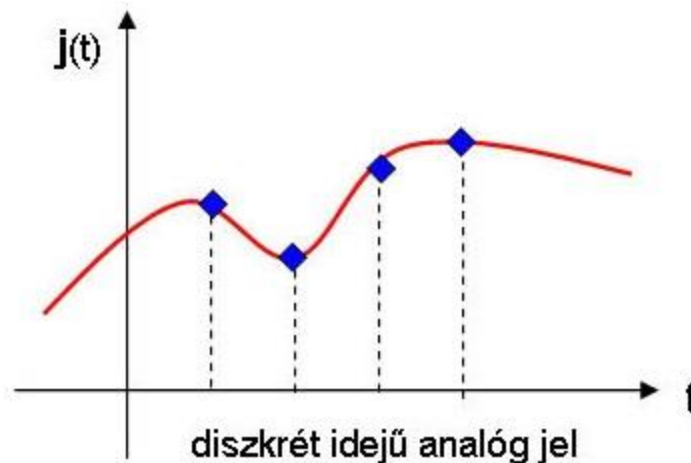
DISZKRÉT AMPLITÚDÓJÚ JELEK

Egy jelet diszkrét amplitúdójúnak nevezünk, ha az amplitúdó értékek ugrásszerűen változnak, míg a jel értelmezési tartománya folytonos.



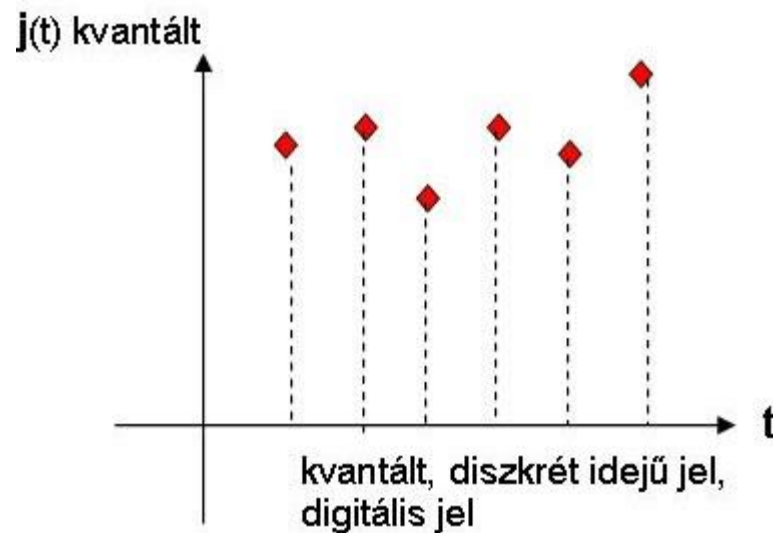
DISZKRÉT IDEJŰ JELEK

Egy jelet diszkrét idejűnek nevezzük, ha a jel értékek csak diszkrét időpontokban állnak rendelkezésre, azonban a jel értékkészlete folytonos.



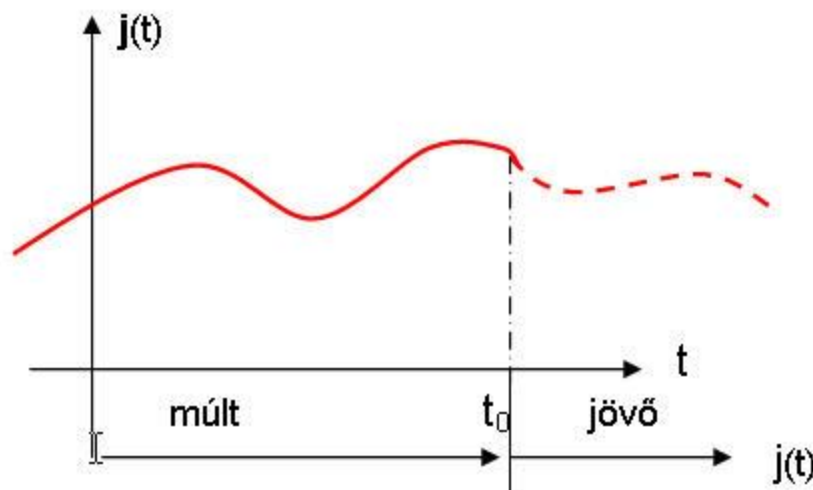
DIGITÁLIS JELEK

Egy jelet digitálisnak nevezünk, ha mind értékkészletében, mind értelmezési tartományában diszkrét.



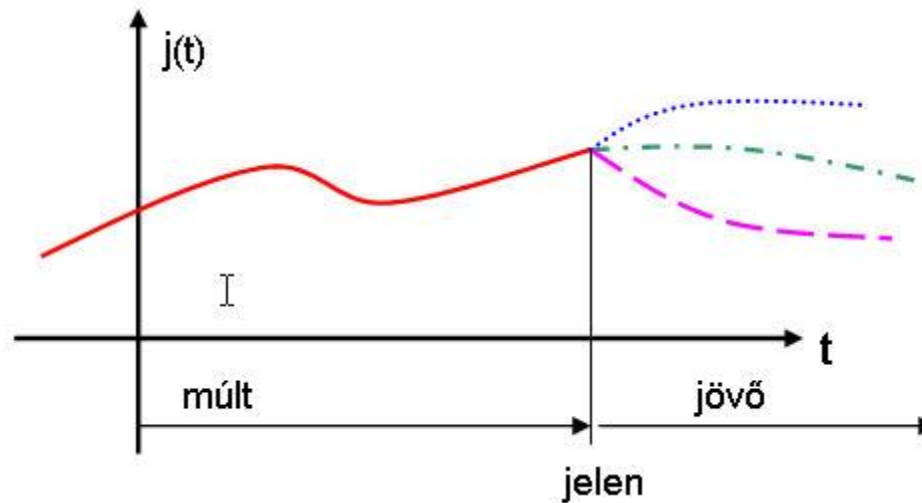
DETERMINISZTIKUS JELEK

Azokat a jeleket, ahol a múltbeli viselkedésből és a jelenkori értékükből meg tudjuk határozni, hogy a jövőben a jel miként viselkedik, determinisztikus jeleknek nevezzük.



SZTOCHASZTIKUS JELEK

Azokat a jeleket, amelyeknél a múltbeli viselkedésből és a jelenkori értékükből nem lehet következtetni a jövőbeli viselkedésre sztochasztikus jeleknek nevezzük.



JELEK ENERGIÁJA

Egy jel energiáján analóg esetben az

$$E = \int_0^{\infty} f^2(t) dt$$

kifejezést, míg digitális esetben az

$$E = \sum_{n=0}^{\infty} f_n^2$$

kifejezést értjük.

VÉGES ENERGIÁJÚ JELEK

Egy f jelet véges energiájúnak nevezünk, ha analóg esetben

$$E = \int_0^{\infty} f^2(t) dt < \infty$$

illetve digitális esetben

$$E = \sum_{n=0}^{\infty} f_n^2 < \infty$$

teljesül.

VÉGES IDEJŰ JELEK

Az $f(t)$ jelet ($t \in (0, \infty)$) véges idejűnek nevezzük, ha van olyan t_1 és t_2 ($t_1 < t_2$) véges időpontpár, amelyre érvényes, hogy ha

$$\forall t < t_1 \text{ és } \forall t > t_2 \text{ akkor } f(t) = 0$$

KORLÁTOS JELEK

Egy jelet korláatosnak nevezünk, ha a jel értelmezési tartományán a jel értékkészlete véges.

TRANZIENS JELEK

Egy jelet tranziens jelnek nevezünk, ha a jel egyszeri, nem periodikus, de a jel által szállított energia véges.

A tranziens jeleket jellemezhetjük a felfutási és lecsengési idejükkel.

JELERŐSSÉG

- A bemenő jel és a kimenő jel hányadosa. Számolási megfontolásokból a hányados logaritmusát használják a gyakorlatban. (A logaritmus ugyanis multiplikatív műveletek helyett additív műveleteket használ.)
- Az arány **mértékegysége** a bel, **dimenziója** nincs. A gyakorlatban ennek tizedét használják

$$X_{db} = 10\log\left(\frac{X}{X_0}\right)^2$$

JELEK FREKVENCIÁJA

A frekvencia egy esemény ismétlődésének a gyakoriságát jelenti. Azt mutatja meg, hogy egy adott időegység alatt az esemény hányszor következett be. Mértékegysége a Hz (Hertz).

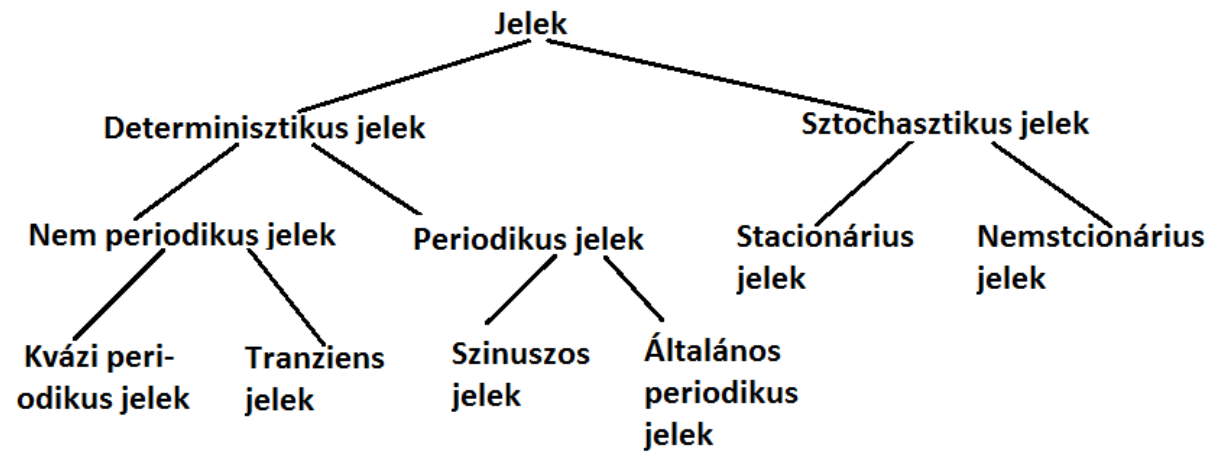
ZAJ

A "zaj" a feldolgozni kívánt információt hordozó jeltől eltérő frekvenciájú és intenzitású jelek zavaró összessége. Más megfogalmazásban: minden olyan jel, amely nem része az átvíendő információnak, de benne van a jelben. A jelfeldolgozás egyik központi feladata a jelben levő zajoktól történő megszabadulás. Ezt a folyamatot a jelek szűrésének nevezik. A technikában általában nem a zaj abszolút nagysága az érdekes, hanem a hasznos jel nagyságához való viszonya. Minél nagyobb ez az érték, annál jobb minőségű az adott rendszer. Mértékegysége a dB (decibel).

ERŐSÍTÉS, CSILLAPÍTÁS

- Csillapításon vagy erősítésen a kimenő (T_{ki}) és a bemenő (T_{be}) jelerősség (teljesítmény) hányadosát értjük. Csillapításnál $T_{ki} < T_{be}$. Erősítés esetén $T_{ki} > T_{be}$.
- **Mértékegysége:** dB (decibel)

JELEK OSZTÁLYOZÁSA



FELADATOK

- Mit nevezünk analóg jelnek? Adjon rá példát!
- Mit nevezünk digitális jelnek?
- Mikor nevezünk egy jelet determinisztikusnak?
- Mikor periodikus egy jel? Adjon rá példát!
- Milyen jelek jellemzéséhez használunk statisztikai jellemzőket?

JELFELDOLGOZÁS ALAPJAI

A jelek által hordozott információ feldolgozási folyamatát ***feldolgozási algoritmusok*** határozzák meg, melyeket az alábbiak szerint lehet csoportosítani:

- Jelanalízis

A jel által hordozott információ tartalom meghatározása

- Jeltárolás

A jel adathordozón történő tárolása

- Jelátvitel

A jel továbbítása más IT eszközre

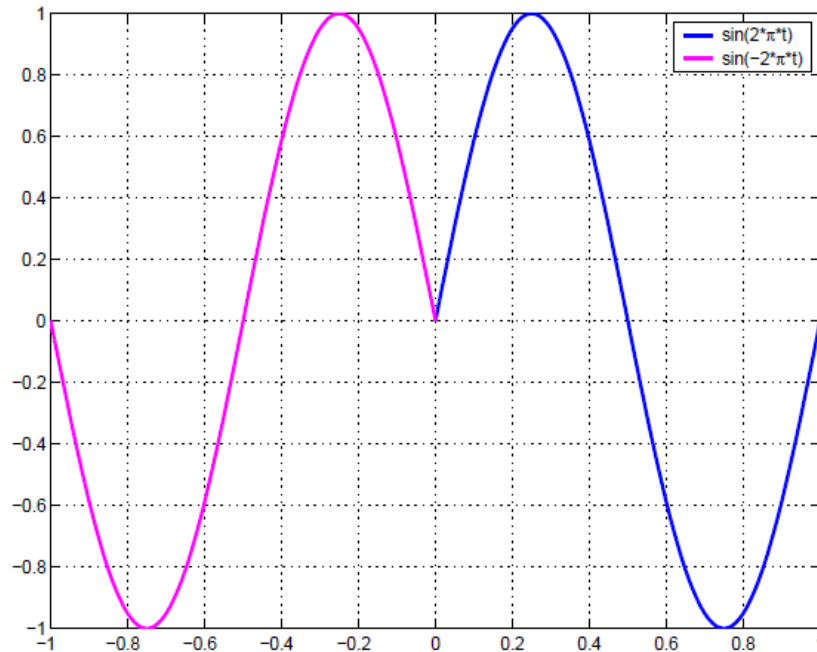
JELANALÍZIS

- **hihetőség figyelés** (jelzésadás, beavatkozás abban esetben, ha mért jel a megadott határokon kívül esik)
- **határszint figyelés** (jelzésadás, beavatkozás abban esetben, ha mért jel a technológiailag veszélyes határ közelébe ér)
- **trend figyelés** (adott időintervallumban a mért jel változásának figyelése)
- **jel szűrés (analóg, digitális)** (spektrális tulajdonságok vizsgálata)
- **korrelációs vizsgálatok**

IDŐALAPÚ JELFELDOLGOZÁS

- Időtengely mentén módosító műveletek:
(reflexió, időeltolás, időintervallum skálázás)
- Amplitúdó tengely módosító műveletek:
(valós számmal való szorzás, invertálás, összeadás, kivonás)

REFLEXIÓ

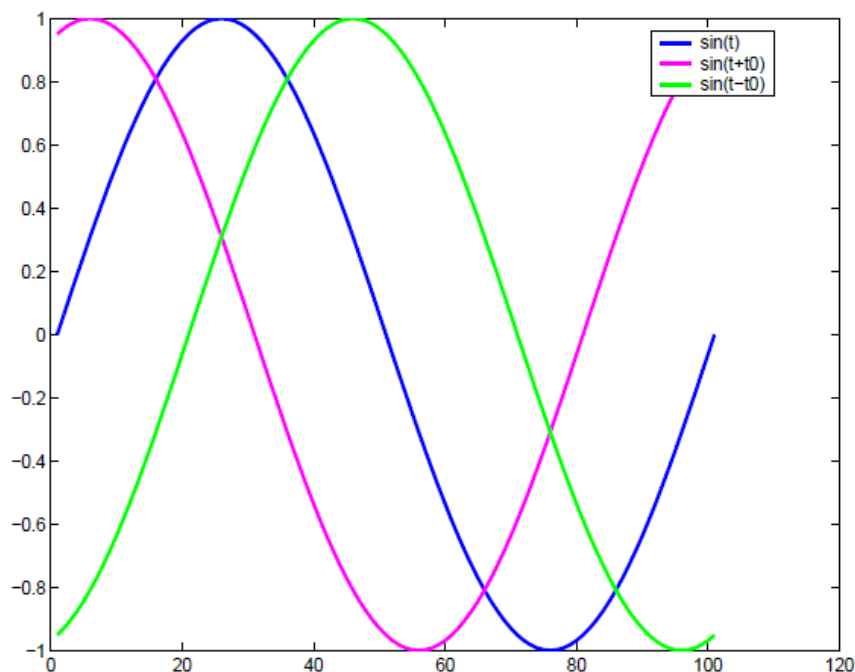


A jel reflexióján az amplitúdó tengelyre való tükrözését értjük. Matematikai alakban analóg jel esetén:

$$y(t) = y(-t)$$

A kék színű az eredeti görbe, a lila az amplitúdó tengelyre tükrözött.

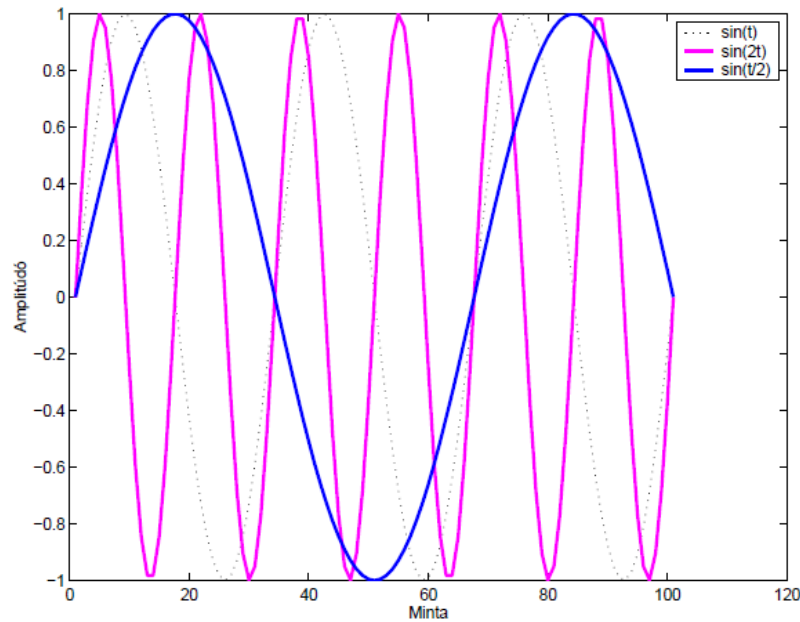
IDŐELTOLÁS



Matematikai formában
analóg jel esetén:

$$y(t) = y(t - t_0)$$

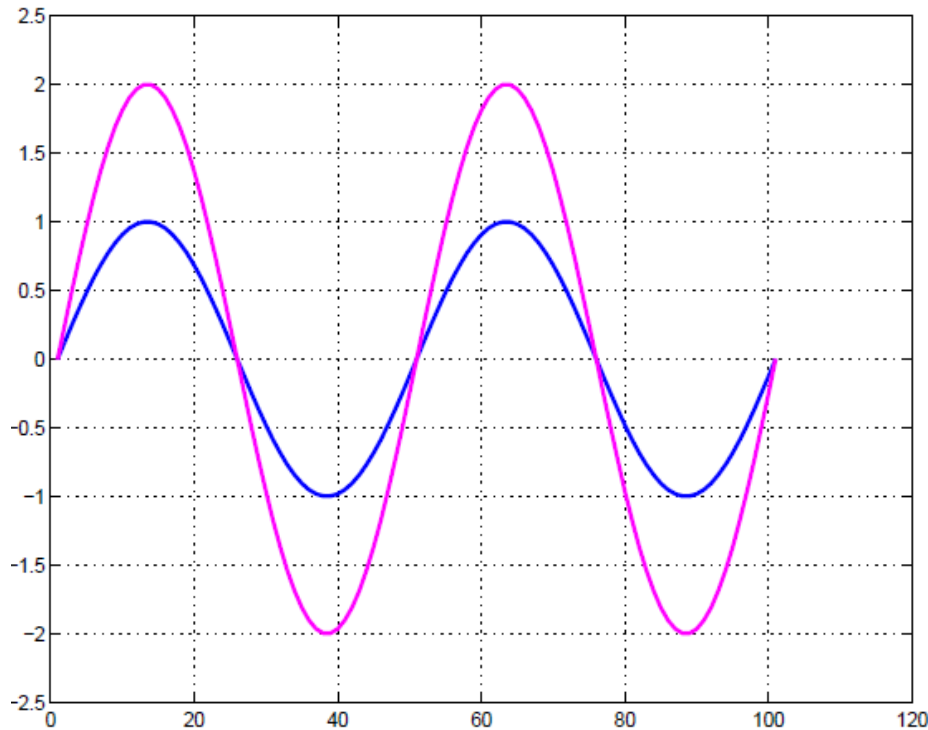
IDŐINTERVALLUM SKÁLÁZÁS



Matematikai formában
megadva analóg jel esetén

$$y(t) = f(at)$$

AMPLITUDÓ SKÁLÁZÁS



Matematikai
formában analóg jel
esetén

$$y(t) = af(t)$$

Technikai értelemben
a valós számmal
való szorzás a jel
erősítésének felel
meg.

INVERTÁLÁS

A jel invertálásán a jelnek a -1 -el történő szorzását értjük.
Matematikai formában analóg jel esetén

JELEK ÖSSZEADÁSA, KIVONÁSA

Jel összegén minden időpontban az adott időponthoz tartozó amplitúdók összegét értjük.

$$f * g = \int_{-\infty}^{\infty} f(x)g(u-x)dx$$

$$y(n) = x(n) * h(n) = \sum_{i=-\infty}^{\infty} x(n)h(n-i)$$

FREKVENCIA ALAPÚ FELDOLGOZÁS



Jean Baptiste Joseph Fourier
(1768-1830)

Minden periodikus jel felírható egyértelműen felírható megfelelő amplitúdókkal és fázisállandókkal bíró harmonikus rezgések összegeként

$$f(t) = a_0 + \sum_{n=1}^{\infty} (a_n \cos n\omega t + b_n \sin n\omega t)$$

$$a_n = \frac{2}{T} \int_0^T f(t) \cos(n\omega t) dt$$

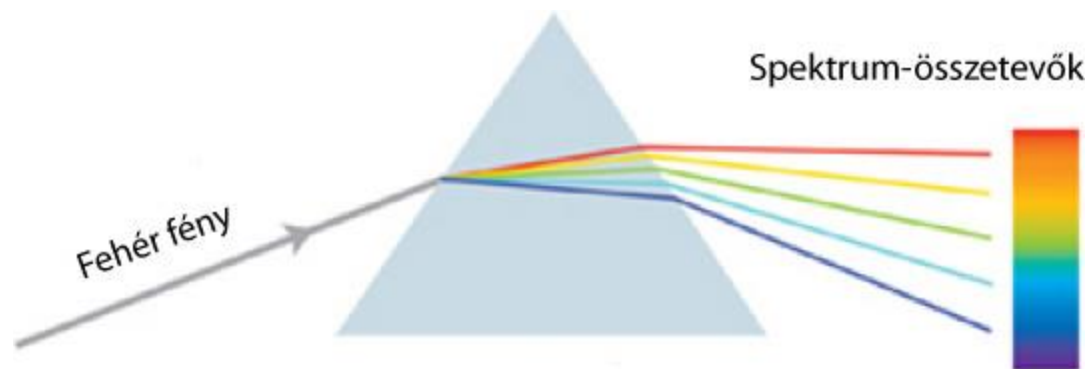
$$b_n = \frac{2}{T} \int_0^T f(t) \sin(n\omega t) dt$$

$$a_0 = \frac{1}{T} \int_0^T f(t) dt$$

JELEK SPEKTRUMA

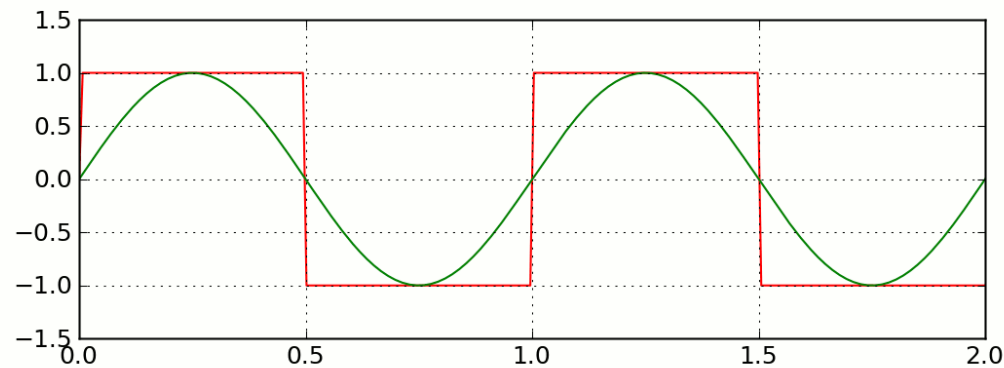
Ha az $f(t)$ periodikus függvény Fourier sorában a jobboldalon levő szinusz függvények mindig pozitívnak választható amplitúdóit ábrázoljuk a szinuszok frekvenciáinak függvényében, akkor megkapjuk az illető periodikus függvény színeképét vagy idegen szóval spektrumát.

Periodikus jel spektruma egymástól elkülöníthető vonalakkól áll.



NÉSGYSZÖGJEL FOURIER SPEKTRUMA

$$f(t) = \frac{4}{\pi} \left(\sin(\omega t) + \frac{1}{3} \sin(3\omega t) + \frac{1}{5} \sin(5\omega t) + \frac{1}{7} \sin(7\omega t) + \dots \right)$$



FOURIER SOR KOMPLEX ALAKJA

$$e^{jx} = \cos(x) + j \sin(x)$$

$$\cos \alpha = \frac{1}{2}(e^{j\alpha} + e^{-j\alpha}) \qquad \sin \alpha = \frac{1}{2j}(e^{j\alpha} - e^{-j\alpha})$$

$$f(t) = \sum_{k=-\infty}^{\infty} C_k e^{(jk\omega_0 t)}$$

NEM PERIODIKUS JELEK FOURIER ANALÍZISE

A nem periódusos jel felfogható úgy is, mint olyan periodikus jel melynek periódus ideje végtelen. Ekkor a spektrumvonalak egymáshoz viszonyított távolsága tart 0-hoz. Azaz a nem periodikus jelek spektruma **folytonos**.

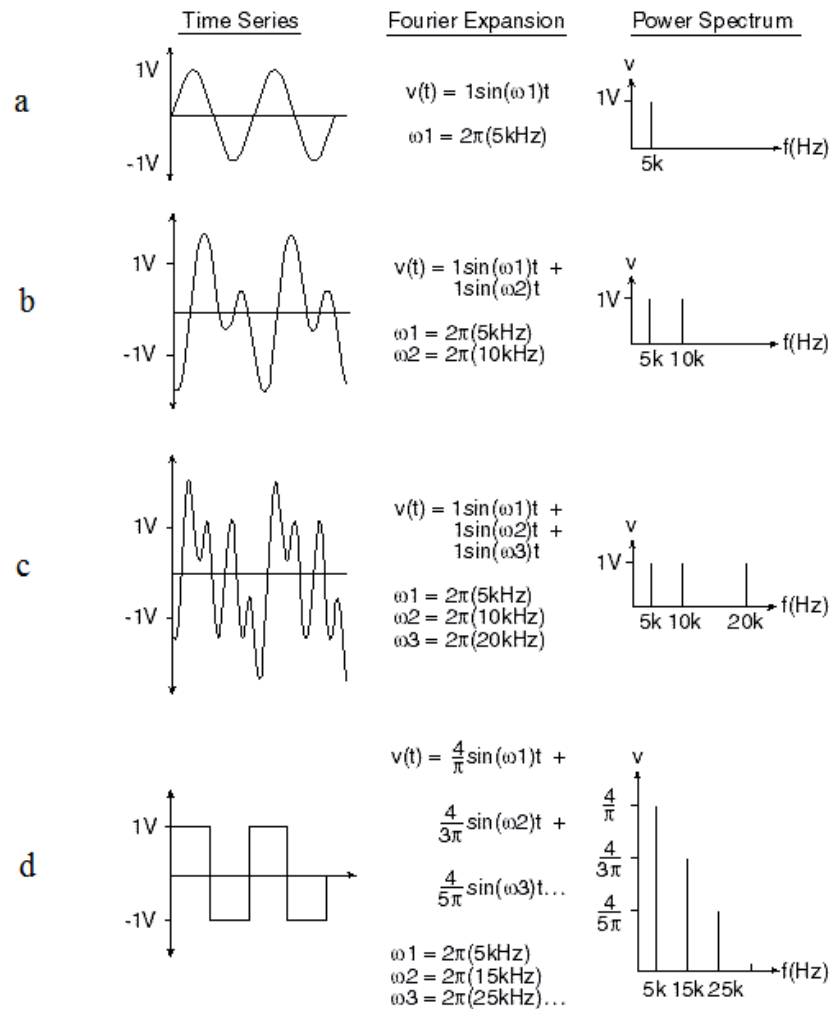
Matematikai alakban:

$$f(t) = \int_{-\infty}^{\infty} C(\omega) e^{j\omega t} d\omega$$

Fourier transzformáció:

$$C(\omega) = \int_{-\infty}^{\infty} f(t) e^{-j\omega t} dt$$

JELEK FOURIER SPEKTRUMA



a: egy 5 kHz frekvenciájú tiszta szinusz jel és annak spektruma,
b: egy 5 kHz-es szinusz és egy 10 kHz-es szinuszból álló jel és spektruma,
c: egy 5 kHz-es szinusz, egy 10 kHz-es szinuszból és egy 20 kHz-es szinuszból álló jel és spektruma,
d: egy négyszög jel és annak spektruma

A TRANSZFORMÁCIÓ TULAJDONSÁGAI

	Tulajdonság	Idő tér	Frekvencia tér
1	A transzformáció lineáris	$a_1 f_1(t) + a_2 f_2(t)$	$a_1 C_1(\omega) + a_2 C_2(\omega)$
2	Rövidebb jelnek szélesebb a spektruma (léptékváltás)	$f(t/a)$	$aC(a\omega)$
3	Az időeltolás nem változtatja meg a spektrum alakját	$f(t-t_0)$	$C(\omega)e^{-j\alpha t_0}$
4	Színuszos moduláló jel eltolja a spektrumot, de nem változtatja meg az alakját	$f(t)e^{j\alpha t_0}$	$C(\omega - \omega_0)$
5	Az időtérben végzett konvolúció a frekvencia térben szorzásnak felel meg	$\int_{-\infty}^{\infty} f_1(t) f_2(t-\tau) d\tau$	$C_1(\omega)C_2(\omega)$

DISZKRÉT FOURIER TRANSZFORMÁCIÓ

Legyen az $x[N]$ vektorban N db minták egy jelből, melyek egyenlő távolságra vannak egymástól az időtengelyen. Ekkor definíció szerint az $x[N]$ vektor diszkrét Fourier transzformáltján (továbbiakban DFT) az alábbi $y[N]$ vektort értjük.

$$y_k = \sum_{n=0}^N x_n e^{\frac{-j2\pi kn}{N}}$$

a transzformáció inverze:

$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} y_k e^{\frac{j2\pi kn}{N}}$$

DISZKRÉT FOURIER TRANSZFORMÁCIÓ

A DFT egy olyan függvény transzformáció (operáció), mely egy N db mintából álló sorozatból egy másik N db mintából álló sorozatot képez, a dimenzió nélküli k index segítségével.

A DFT szempontjából mindegy, hogy a bemenő jel ténylegesen periodikus vagy sem. Úgy tekinti az N db mintát tartalmazó x_n vektort, mint egy végtelen periodikus jel egy periódusát.

A transzformáció eredményeképpen előálló y_k vektor N db komplex számot tartalmaz, ahol az egyes tagok abszolút értékeiből képzett sorozat adja az amplitúdó spektrumot, a fázis szögek sorozata pedig a fázis spektrumot.

DISZKRÉT FOURIER TRANSZFORMÁCIÓ

```
typedef struct
{
    double re;
    double img;
}COMPLEX;

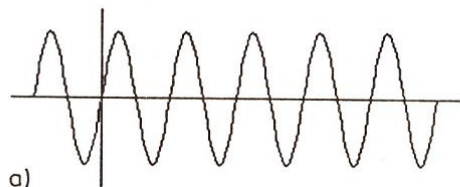
void dft(COMPLEX *in, COMPLEX *out, int N)
{
    double re, im;
    int k, n;
    for(k=0; k<N; k++)
    {
        re=0; im=0;
        for(n=0;n<N;n++)
        {
            re=re+in[n].re*cos(2*PI*k*n/N)+in[n].img*sin(2*PI*k*n/N);
            im=im+in[n].img*cos(2*PI*k*n/N)-in[n].re*sin(2*PI*k*n/N);
        }
        out[k].re=re;
        out[k].img=im;
    }
}
```

FELADATOK

- Nevezze meg azokat a műveleteket, amelyek az időtartományban módosítják a jeleket!
- Nevezze meg azokat a műveleteket, amelyek az amplitúdó tartományban módosítják a jeleket!
- Írja fel az $f(t) = \begin{cases} 1, & 0 < t < \pi \\ 2, & \pi < t \leq 2\pi \end{cases}$ jel Fourier sorát!
- Mikor vonalas egy jelnek a spektruma és mikor folytonos?
- Írja fel az inverz Fourier transzformáció formuláját!
- Mikor használjuk a diszkrét Fourier transzformációt.
- Írja fel a diszkrét Fourier transzformáció formuláját!

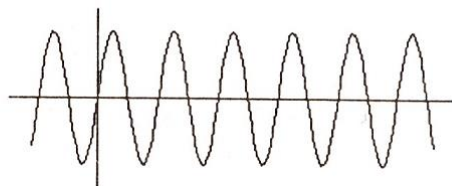
ABLAKOZÁS

$f = 40 \text{ Hz}$



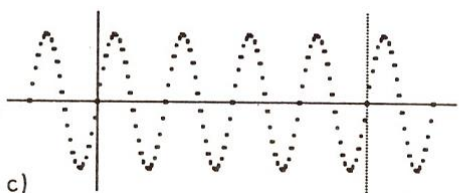
a)

$f = 45 \text{ Hz}$



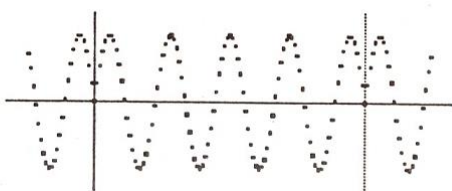
b)

0 100 ms



c)

0 100 ms



a: 40 Hz, és egy 45 Hz szinusz

b: $f_s = 1000 \text{ Hz}$, $\Delta t = 100 \text{ ms}$.

Periodikusan ismételve ezt az eljárást a **c ábrán** látható eredményt kapjuk.

Látható, hogy amikor a mintavételi frekvencia és az eredeti jelben levő frekvencia hányadosa nem egész szám, akkor a mintavételezett jelnél az időablak

szélein torzulások vannak. A torzulás miatt a mintavételezett jel már nem periodikus, ezért nem csak egy szinuszos összetevője van.

Ennek megfelelően a DFT az első esetben a helyes spektrumot fogja szolgáltatni, míg a második esetben a torzult mintavételezett jelnek megfelelően a DFT által számolt spektrum is torzult lesz.

ABLAKOZÁS

A fent említett torzítások jelentősen csökkenthetőek az ablakozási technika segítségével. Az alapötlet a következő: az X_n vektorra alkalmazzunk egy súlyfüggvényt és a DFT-t ezen súlyfüggvénnyel szorzott mintákon végezzük el:

$$y_k = \sum_{n=0}^N W_n x_n e^{\frac{-j 2 \pi k n}{N}}$$

A súlyfüggvényt úgy kell megválasztani, hogy a megfigyelési ablak szélein az értékei kicsik legyenek. Elvárt követelmény még a súlyfüggvénnyel szemben, hogy szimmetrikus legyen. A jelfeldolgozásban a súlyfüggvényeket ablakozó függvényeknek is nevezik. A gyakorlatban az alábbi ablakozó függvényeket szokták használni:

ABLAK FÜGGVÉNYEK

- Négyzetes ablak

$$W_n = \begin{cases} 1, & 0 \leq n \leq N \\ 0, & \text{egyébként} \end{cases}$$

- Háromszög ablak

$$W_n = \begin{cases} 1 - \frac{N/2 - n}{N/2}, & 0 \leq n \leq N \\ 0, & \text{egyébként} \end{cases}$$

- Hanning (Hann) ablak

$$W_n = \begin{cases} 0.5 - 0.5 \cos(n \frac{2\pi}{N}), & 0 \leq n \leq N \\ 0, & \text{egyébként} \end{cases}$$

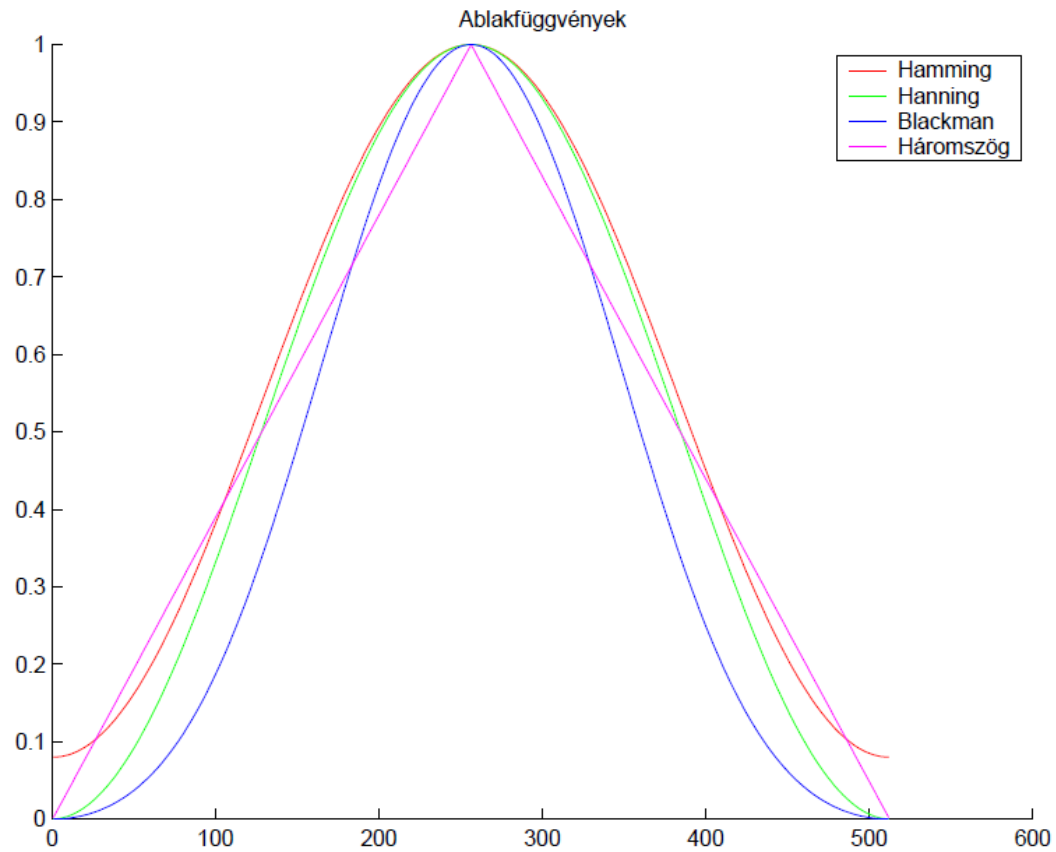
- Hamming ablak

$$W_n = \begin{cases} 0.54 - 0.46 \cos(n \frac{2\pi}{N}), & 0 \leq n \leq N \\ 0, & \text{egyébként} \end{cases}$$

- Blackman ablak

$$W_n = \begin{cases} 0.42 - 0.5 \cos(n \frac{2\pi}{N}) + 0.08 \cos(2n \frac{2\pi}{N}), & 0 \leq n \leq N \\ 0, & \text{egyébként} \end{cases}$$

ABLAKOZÓ FÜGGVÉNYEK



GYORS FOURIER TRANSZFORMÁCIÓ

A diszkrét Fourier transzformációt illetve annak inverzét definíció szerint megvalósító algoritmusokkal bármilyen jel spektruma meghatározható. Azonban az eljárásnak van egy hátránya: az, hogy rendkívül számításigényes. Ennek oka, hogy mind a k mind az n indexek értékeit az eljárás során 0-ról N -re kell növelni az eredmény kiszámításához. Ez N^2 számítási műveletet igényel. Az egyenleteket megvizsgálva látjuk, hogy azok szorzásokból és összeadásokból állnak. Azonban tudjuk, hogy a jelenlegi számítógépeken az összeadás ideje elhanyagolható a szorzásokhoz képest. Így egy $N=1000$ pontos diszkrét Fourier transzformáció végrehajtása 10^6 műveletet igényel. Matematikailag azt mondjuk, hogy a DFT $O(N^2)$ (ejtsd: ordo N^2) műveletet igényel.

GYORS FOURIER TRANSZFORMÁCIÓ

A diszkrét Fourier transzformáció számítási igényét Cooley és Tukey 1965-ben publikált algoritmusukkal jelentősen csökkentették. Algoritmusuk az „oszd meg és uralkodj” elv alapján működik. Mivel az eredeti DFT N^2 szorzást igényel, ezért, ha az adatokat szétválasztjuk két egyforma részre, akkor az egyes részek önmagukban $\left(\frac{N}{2}\right)^2$ szorzást igényelnek. A teljes transzformáció pedig nyilván

$$2\left(\frac{N}{2}\right)^2 = \frac{N^2}{2}$$

műveletet. Így, ha a két transzformáció eredményei összekombinálhatóak, akkor ezzel a módszerrel számítási időt lehet megtakarítani.

GYORS FOURIER TRANSZFORMÁCIÓ

Vezessük be a következő jelölést:

$$W_N = e^{-j\left(\frac{2\pi}{N}\right)}$$

Ekkor a DFT a következő alakba írható:

$$y_k = \sum_{n=0}^N x_n e^{\frac{-j2\pi kn}{N}} = \sum_{n=0}^N x_n W_N^{kn}$$

GYORS FOURIER TRANSZFORMÁCIÓ

Külön összegezve a páros és a páratlan indexű tagokat:

$$y_k = \sum_{n=0}^N x_n W_N^{kn} = \sum_{n=0}^{\frac{N}{2}-1} x_{2n} W_N^{2nk} + \sum_{n=0}^{\frac{N}{2}-1} x_{2n+1} W_N^{(2n+1)k}$$

Másképpen:

$$y_k = \sum_{n=0}^N x_n W_N^{kn} = \sum_{n=0}^{\frac{N}{2}-1} x_{2n} (W_N^2)^{nk} + W_N^k \sum_{n=0}^{\frac{N}{2}-1} x_{2n+1} (W_N^2)^{nk}$$

GYORS FOURIER TRANSZFORMÁCIÓ

Azonban

$$W_N^2 = e^{j\left(-\frac{2\pi}{N}2\right)} = e^{j\left(-\frac{2\pi}{\frac{N}{2}}\right)} = W_{\frac{N}{2}}$$

így:

$$y_k = \sum_{n=0}^N x_n W_N^{kn} = \sum_{n=0}^{\frac{N}{2}-1} x_{2n} W_{\frac{N}{2}}^{nk} + W_N^k \sum_{n=0}^{\frac{N}{2}-1} x_{2n+1} W_{\frac{N}{2}}^{nk}$$

GYORS FOURIER TRANSZFORMÁCIÓ

Vagyis az eredeti DFT felírható páros és páratlan indexű tagok diszkrét Fourier transzformáltjainak összegeként. Igaz a páratlan indexű tagúak transzformáltját még be kell szorozni W_N^k -val. Induláskor érdemes N -et 2 egész kitevőjű hatványának választani, mert ekkor a fenti eljárás tovább folytatható egészen addig, amíg az alsó határhoz a 2 pontos DFT-hez nem érünk.

GYORS FOURIER TRANSZFORMÁCIÓ

Példaként vegyünk egy 8 számból álló sorozatot ($N=8$).

$$x[N]=\{0,1,2,3,4,5,6,7\}.$$

Ezt bontsuk fel két csoportra:

$$x[N]=\{0,2,4,6\} \text{ és } \{1,3,5,7\}.$$

Újra bontsuk két-két csoportra:

$$x[N]=\{0,4\} \{2,6\} \{1,5\} \text{ és } \{3,7\}.$$

Így összesen négy darab 2 pontos DFT-t kell kiszámolni.

GYORS FOURIER TRANSZFORMÁCIÓ

Az algoritmus működésére tekintsünk egy egyszerű példát, ahol a hossz, $N=4$. Ekkor a transzformáció a következő lesz:

$$\begin{aligned} y_k &= \sum_{n=0}^4 x_n W_4^{kn} = \sum_{n=0}^1 x_{2n} W_2^{nk} + W_4^k \sum_{n=0}^1 x_{2n+1} W_2^{nk} \\ &= [x_0 + x_2 W_2^k] + W_4^k [x_1 + x_3 W_2^k] \end{aligned}$$

GYORS FOURIER TRANSZFORMÁCIÓ

Azonban

$$W_2^k = e^{j\left(-\frac{2\pi}{2}k\right)} = e^{j\left(-\frac{2\pi}{4}2k\right)} = W_4^{2k}$$

Így a z FFT a következő lesz:

$$y_k = \left[x_1 + x_2 W_4^{2k} \right] + W_4^k \left[x_1 + x_3 W_4^{2k} \right]$$

GYORS FOURIER TRANSZFORMÁCIÓ

Minden egyes k -ra végig kiszámolva az algoritmus a következő:

$$y_0 = [x_0 + x_2 W_4^0] + W_4^0 [x_1 + x_3 W_4^0]$$

$$y_1 = [x_0 + x_2 W_4^2] + W_4^1 [x_1 + x_3 W_4^2]$$

$$y_2 = [x_0 + x_2 W_4^0] + W_4^2 [x_1 + x_3 W_4^0]$$

$$y_3 = [x_0 + x_2 W_4^2] + W_4^3 [x_1 + x_3 W_4^2]$$

$$W_4^4 = e^{j\left(-\frac{2\pi}{4}\right)} = 1 = W_4^0$$

$$W_4^6 = e^{j\left(-\frac{2\pi}{4}\right)^6} = -1 = W_4^2$$

GYORS FOURIER TRANSZFORMÁCIÓ

Az FFT legfőbb előnye a DFT-vel szemben, hogy a transzformáció végrehajtásához lényegesen kevesebb műveletszám kell. A DFT-nél a műveletszám N^2 . Az FFT fenti megvalósításánál ez az érték

$$N \log_2 N$$

Egy 1024 pontos DFT esetén a műveletszám 1048576 addig az FFT-nél ez 10240, azaz a sebesség növekedés ebben az esetben durván százszoros.

GYORS FOURIER TRANSZFORMÁCIÓ

```
//The initial signal is stored in the array data of length 2*nn,  
//where each even element corresponds to the real part  
//and each odd element to the imaginary part of a complex number
```

```
void four1(double* data, unsigned long nn)  
{  
    unsigned long n, mmax, m, j, istep, i;  
    double wtemp, wr, wpr, wpi, wi, theta;  
    double tempr, tempi;  
  
    // reverse-binary reindexing  
    n = nn<<1;  
    j=1;  
    for (i=1; i<n; i+=2) {  
        if (j>i) {  
            swap(data[j-1], data[i-1]);  
            swap(data[j], data[i]);  
        }  
        m = nn;  
        while (m>=2 && j>m) {  
            j -= m;  
            m >>= 1;  
        }  
        j += m;  
    }  
};
```

GYORS FOURIER TRANSZFORMÁCIÓ

```
// here begins the Danielson-Lanczos section
mmax=2;
while (n>mmax) {
    istep = mmax<<1;
    theta = -(2*M_PI/mmax);
    wtemp = sin(0.5*theta);
    wpr = -2.0*wtemp*wtemp;
    wpi = sin(theta);
    wr = 1.0;
    wi = 0.0;
    for (m=1; m < mmax; m += 2) {
        for (i=m; i <= n; i += istep) {
            j=i+mmax;
            tempr = wr*data[j-1] - wi*data[j];
            tempi = wr * data[j] + wi*data[j-1];

            data[j-1] = data[i-1] - tempr;
            data[j] = data[i] - tempi;
            data[i-1] += tempr;
            data[i] += tempi;
        }
        wtemp=wr;
        wr += wr*wpr - wi*wpi;
        wi += wi*wpr + wtemp*wpi;
    }
    mmax=istep;
}
```

GYORS FOURIER TRANSZFORMÁCIÓ

```
/* *****  
 * The following is a test routine that generates a ramp *  
 * with 10 elements, finds their FFT, and then finds the *  
 * original sequence using inverse FFT *  
***** */  
  
int main(int argc, char * argv[])  
{  
    int i;  
    int Nx;  
    int NFFT;  
    double *x;  
    double *X;  
  
    /* generate a ramp with 10 numbers */  
    Nx = 10;  
    printf("Nx = %d\n", Nx);  
    x = (double *) malloc(Nx * sizeof(double));  
    for(i=0; i<Nx; i++)  
    {  
        x[i] = i;  
    }  
  
    /* calculate NFFT as the next higher power of 2 >= Nx */  
    NFFT = (int)pow(2.0, ceil(log((double)Nx)/log(2.0)));  
    printf("NFFT = %d\n", NFFT);  
  
    /* allocate memory for NFFT complex numbers (note the +1) */  
    X = (double *) malloc((2*NFFT+1) * sizeof(double));  
  
    /* Storing x(n) in a complex array to make it work with fourl.  
    This is needed even though x(n) is purely real in this case. */  
    for(i=0; i<Nx; i++)  
    {  
        X[2*i+1] = x[i];  
        X[2*i+2] = 0.0;  
    }  
}
```

GYORS FOURIER TRANSZFORMÁCIÓ

```
/* pad the remainder of the array with zeros (0 + 0 j) */
for(i=Nx; i<NFFT; i++)
{
    X[2*i+1] = 0.0;
    X[2*i+2] = 0.0;
}

printf("\nInput complex sequence (padded to next highest power of 2):\n");
for(i=0; i<NFFT; i++)
{
    printf("x[%d] = (%.2f + j %.2f)\n", i, X[2*i+1], X[2*i+2]);
}

/* calculate FFT */
four1(X, NFFT, 1);

printf("\nFFT: \n");
for(i=0; i<NFFT; i++)
{
    printf("X[%d] = (%.2f + j %.2f)\n", i, X[2*i+1], X[2*i+2]);
}
```

GYORS FOURIER TRANSZFORMÁCIÓ

```
/* calculate IFFT */
four1(X, NFFT, -1);

/* normalize the IFFT */
for(i=0; i<NFFT; i++)
{
    X[2*i+1] /= NFFT;
    X[2*i+2] /= NFFT;
}

printf("\nComplex sequence reconstructed by IFFT:\n");
for(i=0; i<NFFT; i++)
{
    printf("x[%d] = (%.2f + j %.2f)\n", i, X[2*i+1], X[2*i+2]);
}

getchar();
}
```

FELADATOK

- Mi az ablakozási technika lényege a DFT-nél?
- Milyen tulajdonságokkal kell rendelkeznie egy súlyfüggvénynek.
- Mik a leggyakrabban használatos súlyfüggvények?
- Mennyi a műveletszám igénye egy $N=2048$ pontos DFT-nek, illetve FFT-nek?
- Mi a Cooley – Tukey féle FFT algoritmus alapelve?

ANALÓG DIGITÁLIS ÁTALAKÍTÁS

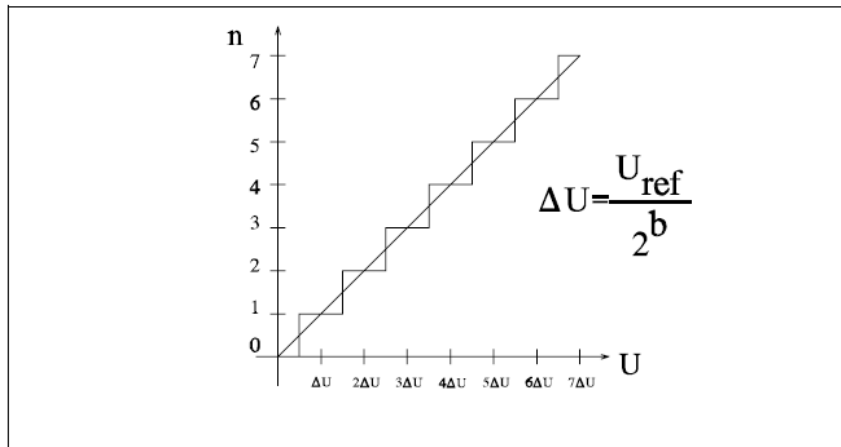
A bementi feszültséget (analóg jelet) ezzel arányos számmá alakítják.

$$Z = \frac{U 2^b}{U_{ref}} + 0.5$$

U_{ref} : referencia feszültség

b : az A/D konverter felbontása

Z : konvertált adat.



A/D ÁTALAKÍTÁS PARAMÉTEREI

- referencia feszültség
- teljes tartomány
- felbontás
- legkisebb szignifikáns bit
- sávszélesség
- mintavételi frekvencia
- konverziós idő

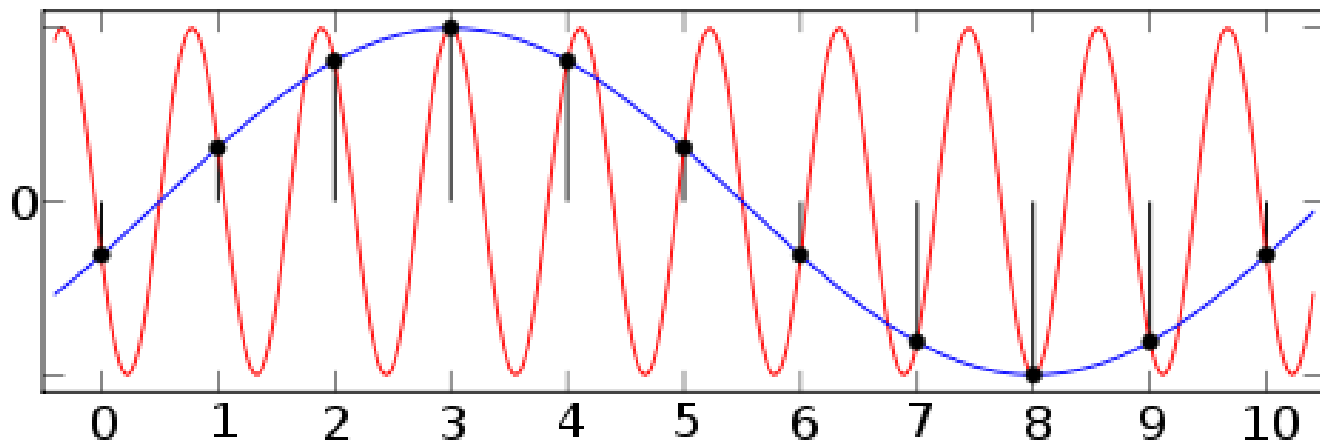
ANALÓG DIGITÁLIS ÁTALAKÍTÁS

Valamennyi A/D konverter a konverzió során az analóg jelen a következő három műveletet hajtja végre:

- Mintavételzés – az idő tengely diszkretizálása
- Kvantálás - az amplitúdó diszkretizálása
- Kódolás – a kvantált amplitúdó érték bináris számmá alakítása

MINTAVÉTELEZÉS

Idő tengely diszkretizálása - adott időnként kiolvassuk az analóg jel értékét.



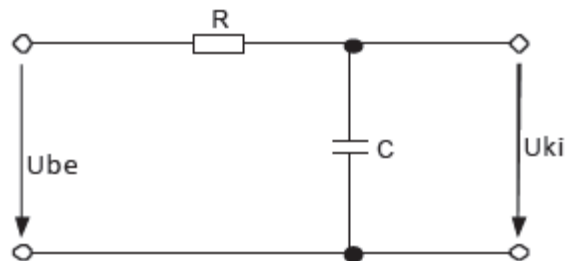
Mintavételi tétel (Shannon, Nyquest):

$$f_{\text{mintavétel}} \geq 2 f_{\text{max}}$$

AULMINTAVÉTELEZÉS (ALIASING)

Alulmintavételezés – a mintavételzett jelből az eredeti jel nem állítható elő.
Az Aliasing effektus furcsa jelenségeket tud produkálni. Tipikus példa a filmekben a hátrafelé forgó kerék.

Aliasing szűrő:



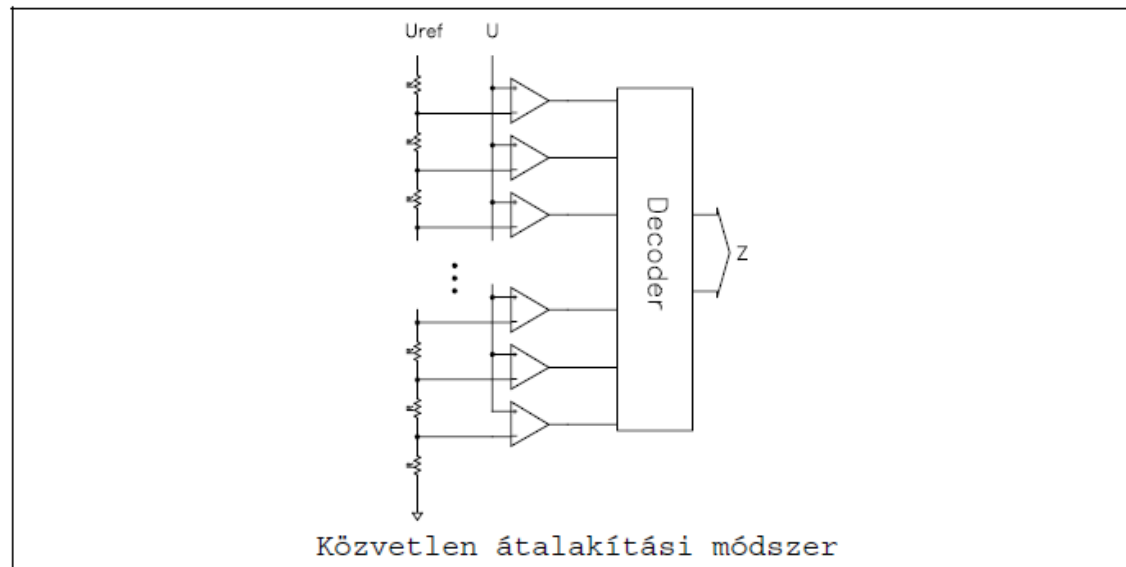
KVANTÁLÁS

Három elvileg eltérő felépítést különböztetünk meg:

- Közvetlen módszer
- Szukcesszív approximációs (fokozatos megközelítés) módszer
- Számlálót alkalmazó módszer

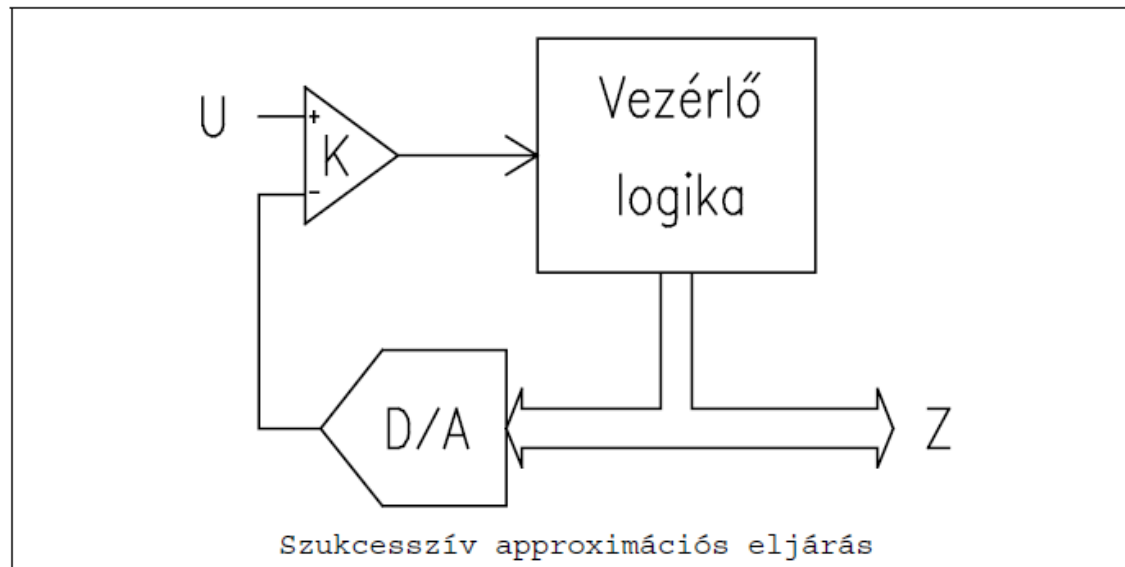
KÖZVETLEN MÓDSZER

A közvetlen módszer esetében a kvantálóra eső pillanatnyi bementi feszültséget összehasonlítjuk n db referencia feszültséggel és meghatározzuk, hogy melyik kettő közé esik.



SZUKCESSZÍV APPROXIMÁCIÓ

A pillanatnyi bementi feszültségnek megfelelő számot nem egy lépésben kapjuk meg. A kvantálás minden egyes lépésénél csak a bináris szám egy helyértékéhez tartozó értéket kapjuk meg. A konverzió mindig a legfelső helyértéken kezdődik.



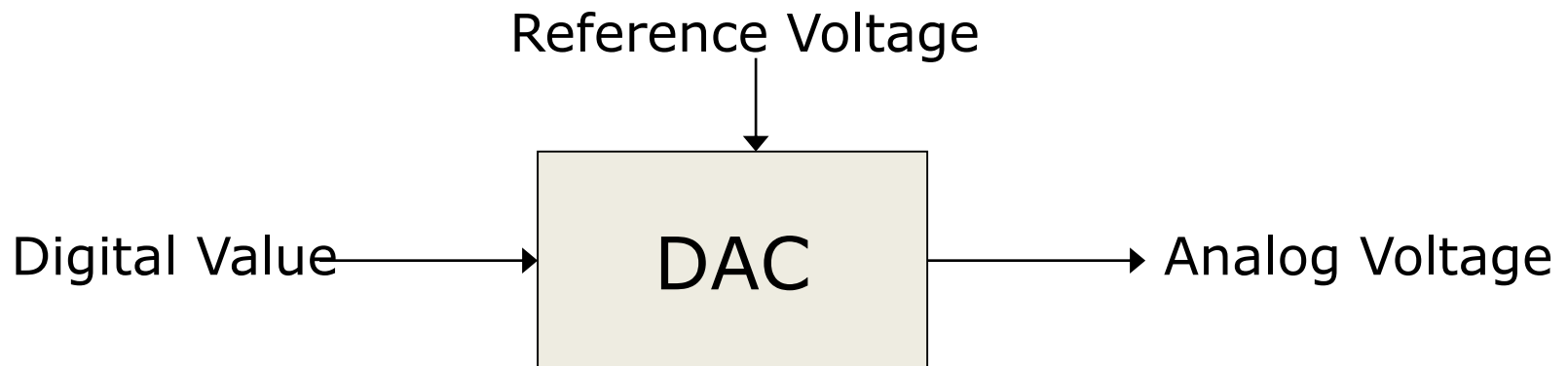
SZÁMLÁLÓT ALKALMAZÓ MÓDSZER

Visszacsatolást használó átalakítók legegyszerűbb változata. Egy ilyen A/D konverter egymás után állítja elő növekvő (vagy csökkenő) sorrendben a részintervallumokhoz tartozó értékeket, majd ezeket egymás után juttatja el egy olyan komparátorba, amely az adott i értékhez tartozó intervallumra hajtja végre az összehasonlítást. A számláló akkor áll le, ha a megadott intervallumban van a feszültség értéke.

DIGITÁLIS ANALÓG KONVERZIÓ (DAC)

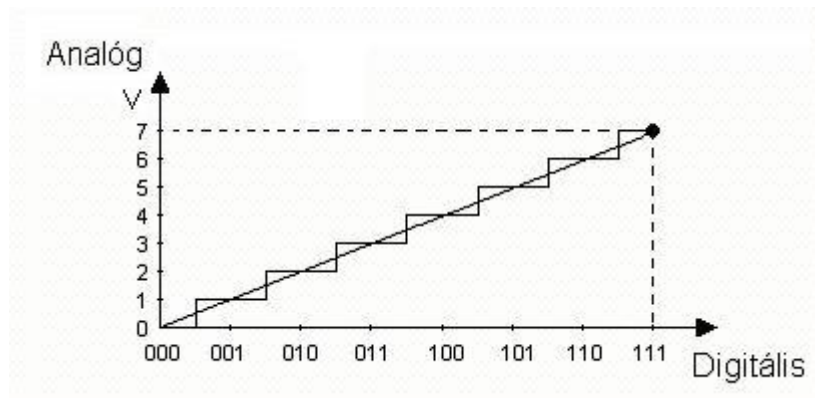
- Digitális értékek analóg feszültség értéké alakítása
- Az ADC inverz művelete

$$U = Z \frac{U_{ref}}{2^b}$$



A DAC JELLEMZŐI

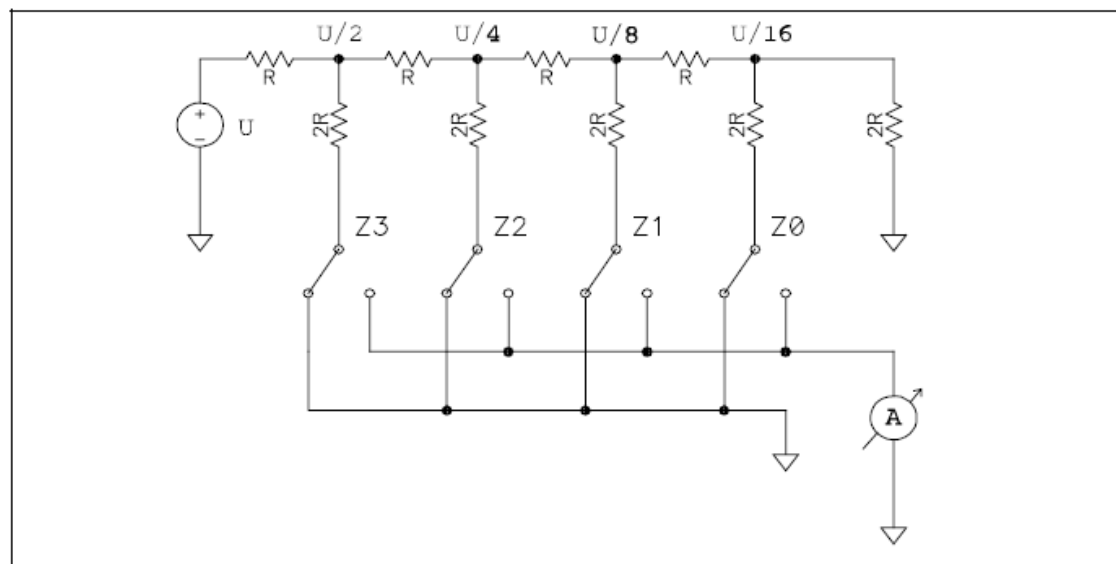
- A kimeneti feszültség tartomány
- Felbontás
- Konverziós idő
- Bitszám
- Átalakítási sebesség



DAC FAJTÁI

- R-2R ellenállás hálózat
- Összegző erősítő átalakító

R-2R ELLENÁLLÁS HÁLÓZAT



R-2R ELLENÁLLÁS HÁLÓZAT

$$I = \frac{V_{REF}}{3R} \left(\frac{B_2}{2} + \frac{B_1}{4} + \frac{B_0}{8} \right)$$

$$V_{OUT} = \frac{R_f}{R} V_{REF} \left(\frac{B_2}{2} + \frac{B_1}{4} + \frac{B_0}{8} \right)$$

B_i = az i-edik Bit értéke

- Általános esetben

$$I = \frac{V_{REF}}{3R} \sum \frac{B_i}{2^{n-i}}$$

$$V_{OUT} = V_{REF} \sum \frac{B_i}{2^{n-i-1}}$$

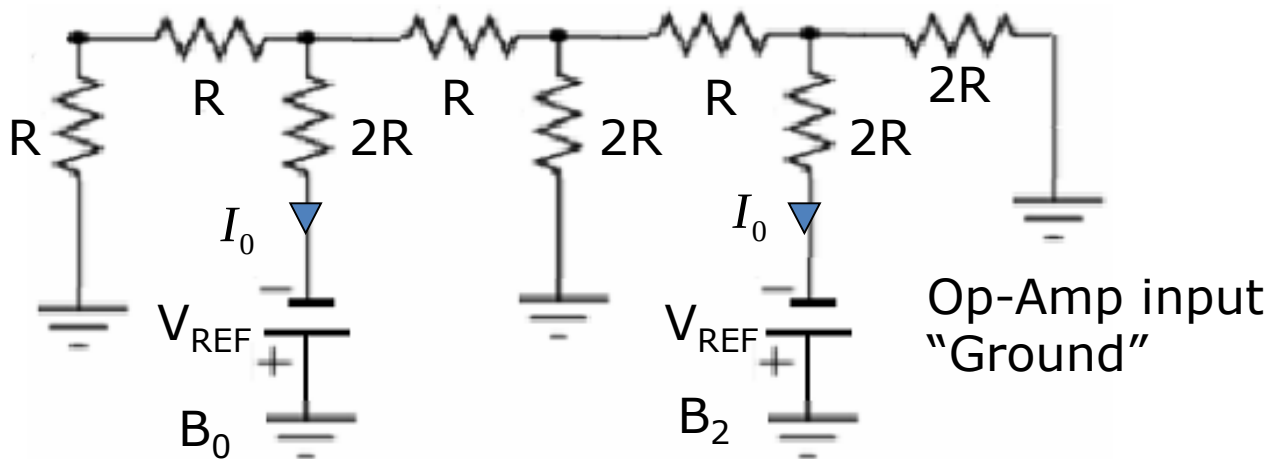
R-2R PÉLDA

- Input = $(101)_2$
- $V_{REF} = 10\text{ V}$
- $R = 2\ \Omega$
- $R_f = 2R$

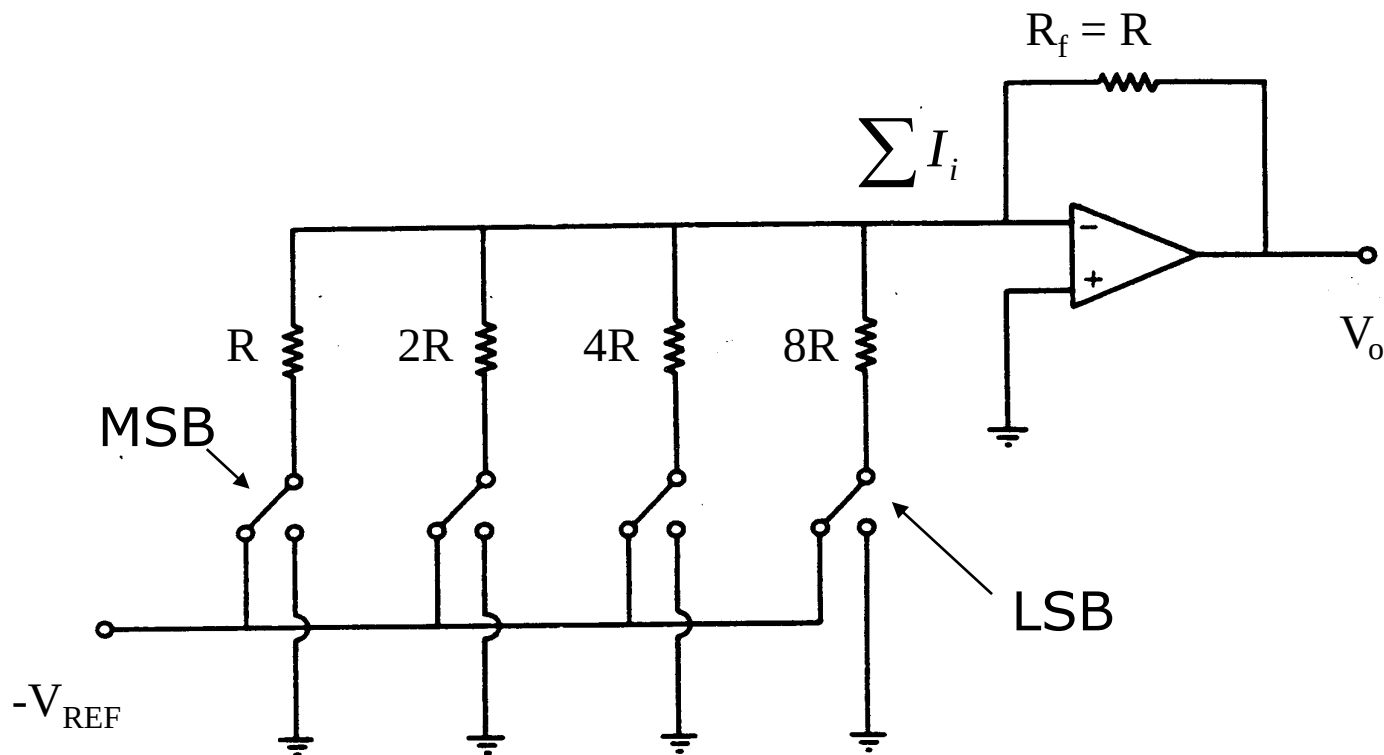
$$I_0 = \frac{-V_{REF}}{2R + 2R \parallel 2R} = \frac{V_{REF}}{3R} = -1.67\text{ mA}$$

$$I_{op-amp} = \frac{I_0}{8} + \frac{I_0}{2} = -1.04\text{ mA}$$

$$V_{OUT} = -I_{op-amp} R_f = 4.17\text{ V}$$



ÖSSZEGZŐ ERŐSÍTŐS ÁTALAKÍTÓ



ÖSSZEGZŐ ERŐSÍTŐS ÁTALAKÍTÓ

$$\sum I = V_{REF} \left(\frac{B_3}{R} + \frac{B_2}{2R} + \frac{B_1}{4R} + \frac{B_0}{8R} \right)$$

$$V_{OUT} = I \cdot R_f = V_{REF} \left(B_3 + \frac{B_2}{2} + \frac{B_1}{4} + \frac{B_0}{8} \right)$$

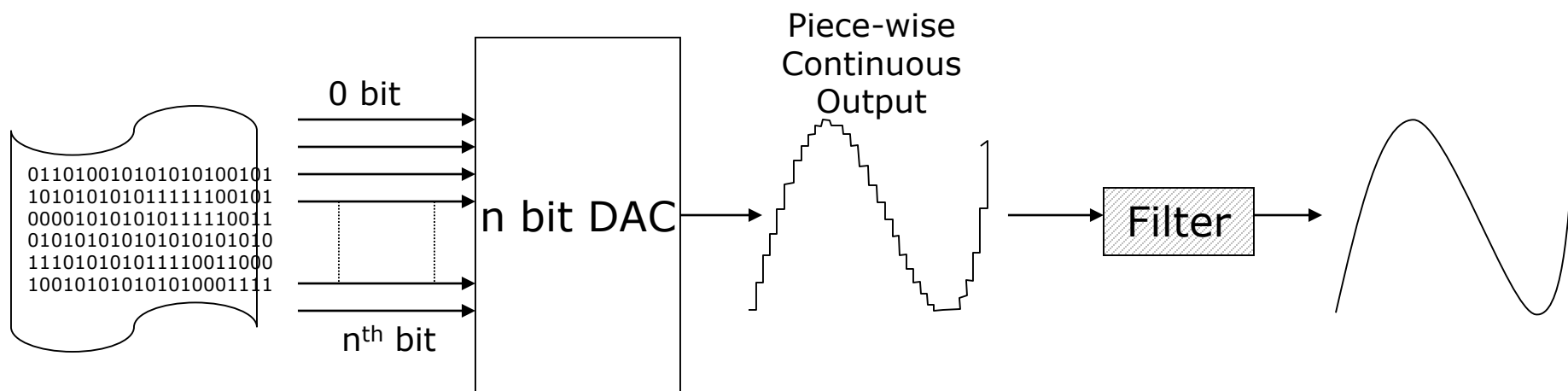
B_i = az i-edik Bit értéke

ÖSSZEGZŐ ERŐSÍTŐS ÁTALAKÍTÓ

- Általános esetben:

$$\begin{aligned} V_{OUT} &= V_{REF} \sum \frac{B_i}{2^{n-i-1}} \\ &= V_{REF} \cdot \text{Digitális érték} \cdot \text{Felbontás} \end{aligned}$$

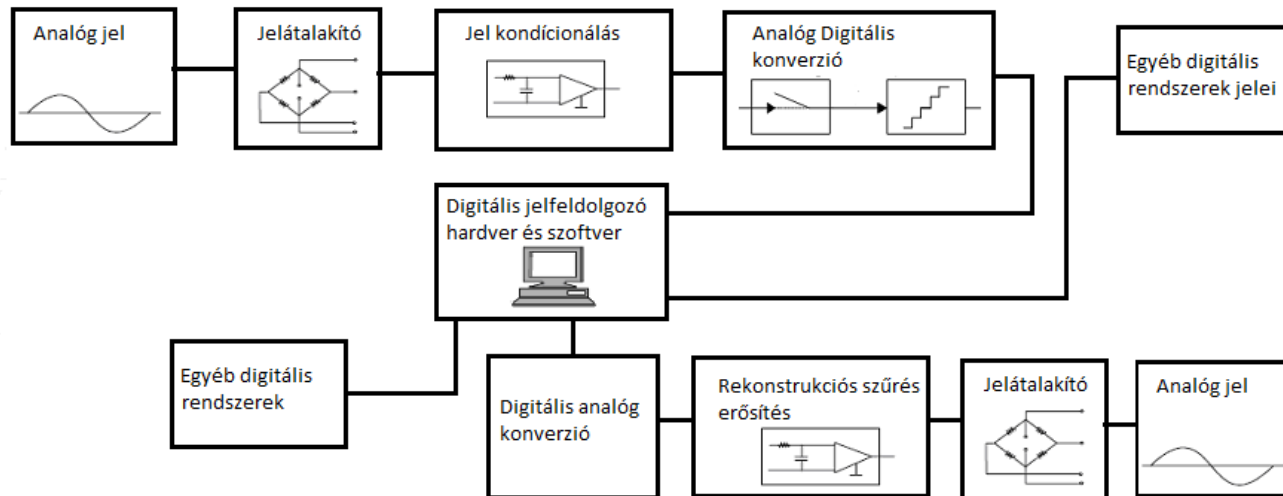
DAC EREDMÉNYE



FELADATOK

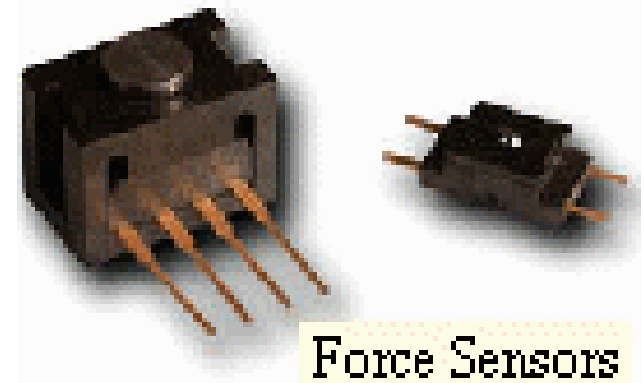
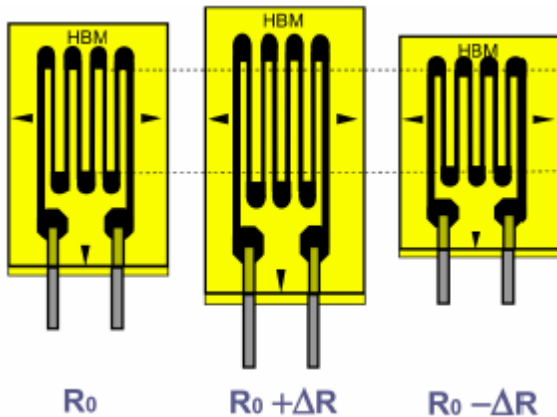
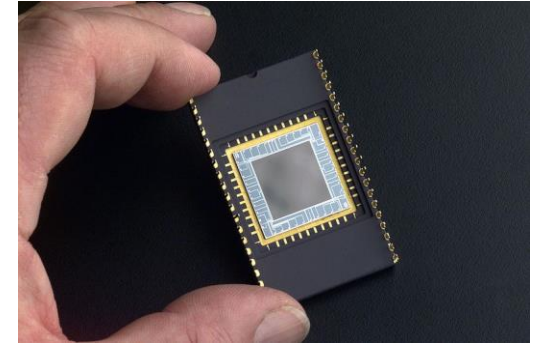
- Az A/D konverternek milyen feladatokat kell elvégeznie?
- Hány byte adatmennyiség keletkezik az A/D konverter kimenetén 10 s alatt, ha a mintavételi frekvencia 144,4 kHz, és a kvantálás hossza 16 bit?
- Sorolja fel a DAC jellemzőit!

DIGITÁLIS JELFELDOLGOZÁS



JELÁTLALKÍTÓK (ÉRZÉKELŐK, SZENZOROK)

Valamilyen fizikai jelet (erő, fény, hang, hőmérséklet) a jel nagyságával arányos analóg elektromos jellé alakít.

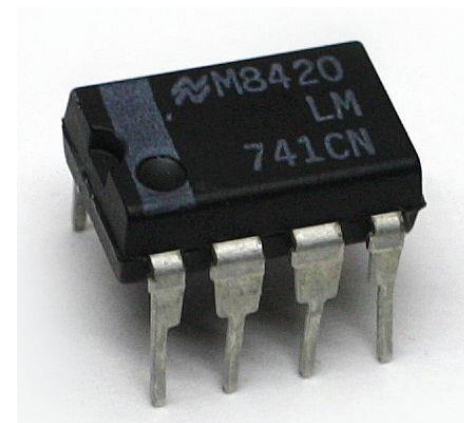
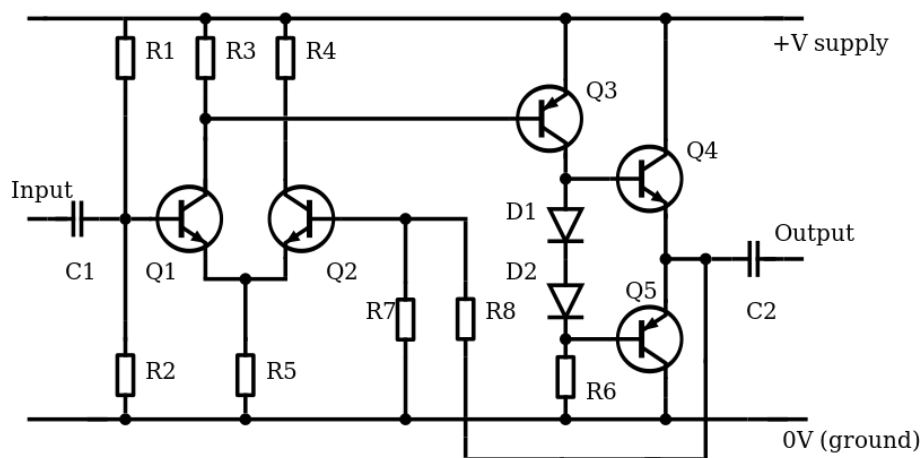


JELKONDITIONÁLÁS

- Erősítéssel vagy csillapítással a megfelelő intenzitású jel előállítása,
- Az analóg jel erősebb „kilengéseit” kezelje, szűrje az analóg jelet a zajtól.

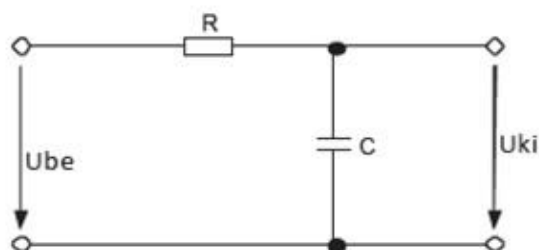
JELERŐSÍTÉS (CSILLAPÍTÁS)

Erősítéssel vagy csillapítással a megfelelő intenzitású jel előállítása.

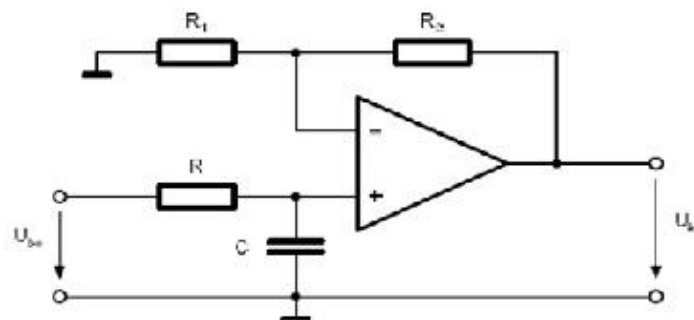


ANTIALIASING SZŰRŐ

Az analóg jel sávkorlátozását végzi el a jel digitalizálása előtt ezzel javítva a jel/zaj viszonyt. Olyan aktív vagy passzív RLC elemekből megvalósított analóg áramkör, mely egy előre adott frekvencia felett (vágási frekvencia) minden olyan komponenst kiszűr az analóg jelből, melynek frekvenciája nagyobb ennél.

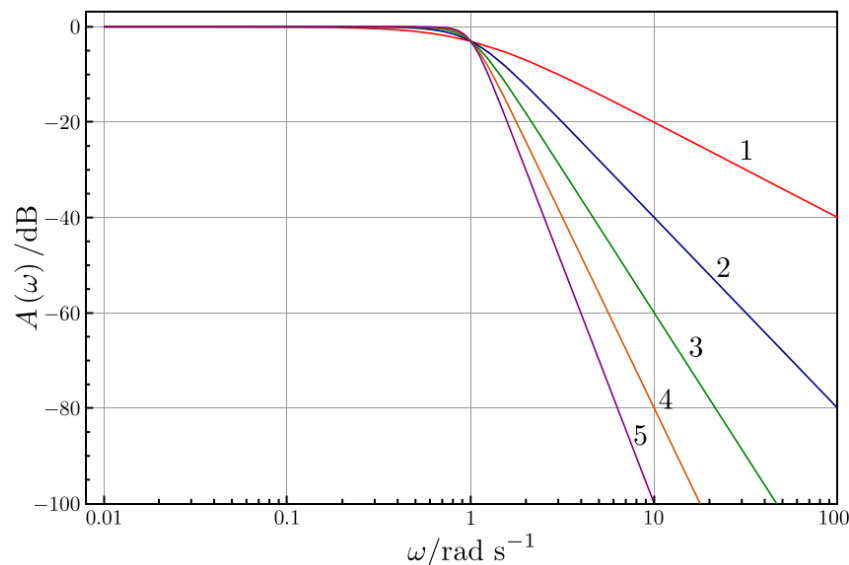


Passzív



Aktív

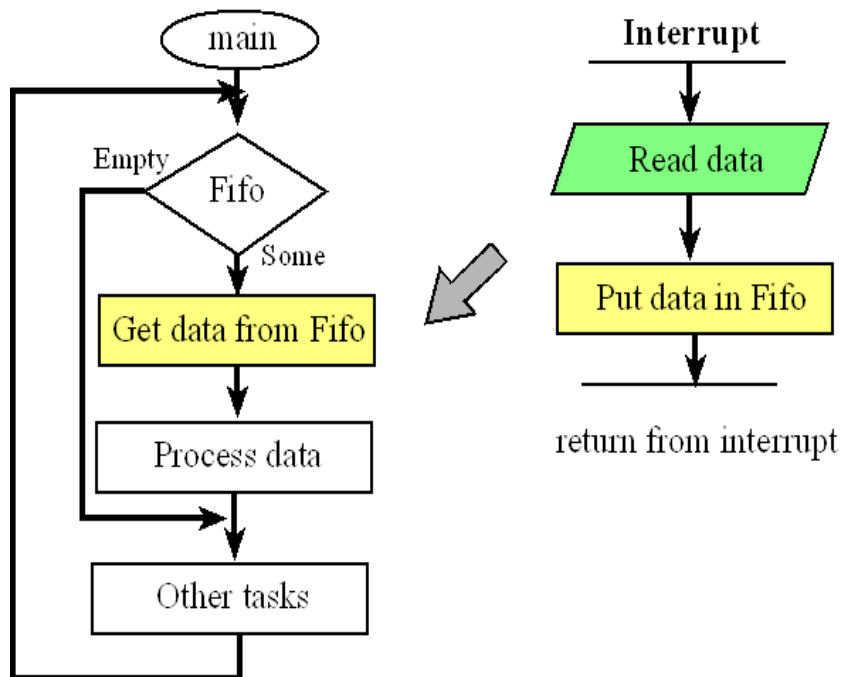
ANTIALIASING SZŰRŐ



A szűrő típus kiválasztásánál fontos, hogy az átteresztési tartományban mennyire sima a frekvencia menet, a vágási frekvenciánál mennyire meredeken vág, valamint mennyire bonyolult az áramköri megvalósítása. A gyakorlatban Butterworth típusú szűrőket használnak erre a feladatra.

$$H(j\omega) = \frac{1}{\sqrt{1 + \varepsilon^2 \left(\frac{\omega}{\omega_c} \right)^{2n}}}$$

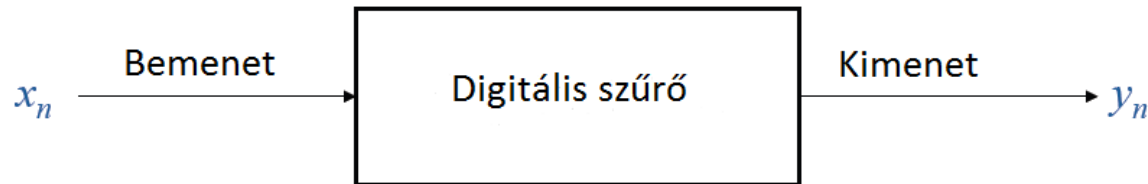
DIGITALIZÁLÁS



Szoftver szempontból nézve az adatvesztést elkerülendő az AD konverternek megszakítás vezérelten kell működnie. Ez azt jelenti, hogy a rendszer egyik időzítőjét a mintavételi frekvenciára programozzuk. Az időzítő minden egyes aktiválásakor az AD konverter regiszterében levő értéket elhelyezzük egy körkörös FIFO puffalba. A fő program pedig folyamatosan futva ellenőrzi, hogy a van –e a FIFO-ban érvényes adat.

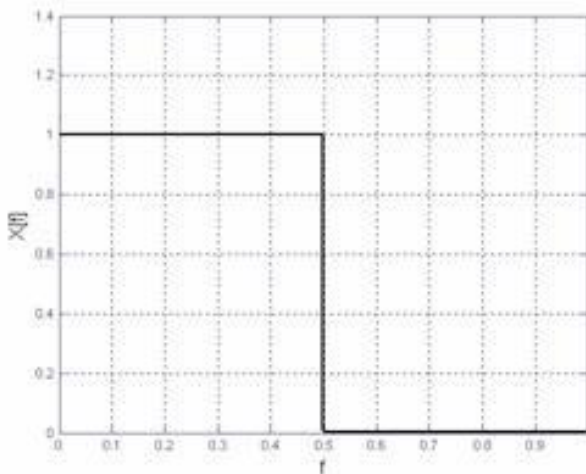
DIGITÁLIS SZŰRŐK

A digitális szűrő olyan szoftveres vagy hardveres eljárás, mely a bemenő x_n számsorozatot y_n kimenő számsorozattá alakítja.



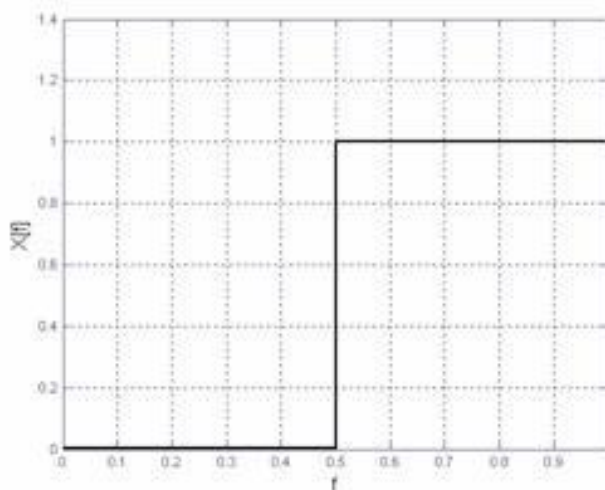
Időtartománybeli vagy frekvenciatartománybeli működésre lehet optimalizálni.

ALULÁTERESZTŐ SZŰRŐ



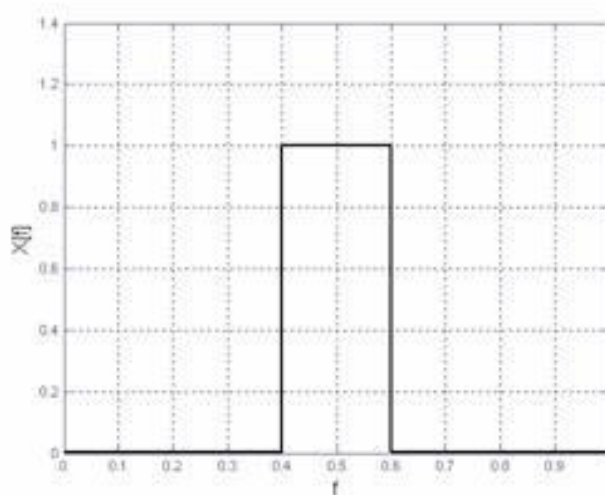
Az **aluláteresztő** szűrő olyan szűrő megvalósítás, mely a szűrő bementére adott jelet úgy módosítja, hogy a jelből, csak a kis frekvenciás komponenseket engedi át. Az átengedés itt azt jelenti, hogy a szűrőre jellemző f_v vágási frekvenciánál kisebb frekvenciájú komponenseket engedi át változatlan amplitúdó értékkel a vágási frekvencia felettieknél viszont az amplitúdó értéket nullázza. Azt a frekvencia tartományt, ahol szűrő átereszt áteresztési tartománynak, ahol pedig nem azt vágási tartománynak nevezzük.

FELÜLÁTERESZTŐ SZŰRŐ



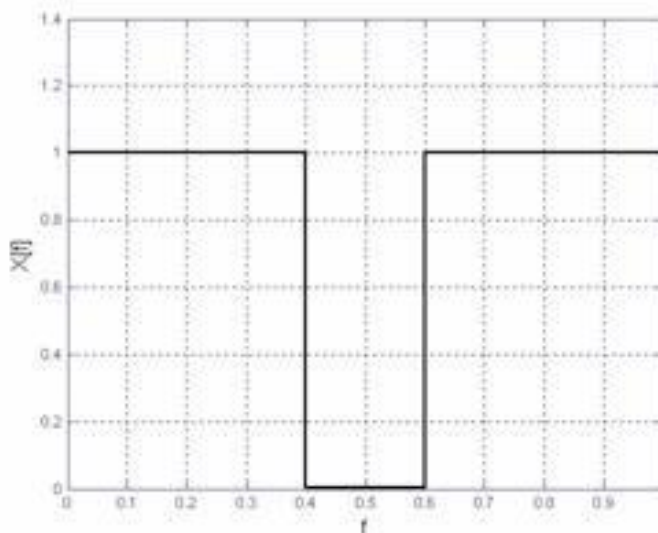
A **felüláteresztő** szűrő az aluláteresztővel pontosan ellentétesen viselkedik. Vagyis a vágási frekvencia felett átereszt, míg alatta szűr.

SÁVÁTERESZTŐ SZŰRŐ



A **sáváteresztő** szűrő egy adott frekvencia tartományon belül ereszt át a bementére adott jelet. A tartományon kívül pedig vág.

SÁVKIZÁRÓ SZŰRŐ



A **sávkizáró** szűrő pedig a sáváteresztő szűrő ellentéte, azaz a frekvencia tartományon belül vág, míg a tartományon kívül átereszt.

DIGITÁLIS SZŰRŐ VÁLASZTÁS

A fentiek alapján azt hihetjük, hogy a digitális szűrők szűrő hatását elég csak a frekvencia tartományban vizsgálni. Ez azonban csak abban az esetben igaz, ha a szűrő feladata olyan, amit a frekvencia tartományban lehet jól elvégezni. Ilyen feladat pl. a jelek szeparálása. Amennyiben a szűrő feladata a jel véletlen zajoktól történő megszabadítása, (simítás), akkor a szűrőnek az időtartománybeli viselkedése a fontos, hiszen a szűrőtől azt várjuk, hogy a zaj kiszűrése után a jelalak lehetőleg változatlan maradjon.

FIR SZŰRŐK

Véges válaszijű (*Finite Impulse Response, FIR*) szűrők

$$y_n = \sum_{i=0}^k a_i x_{n-i}$$

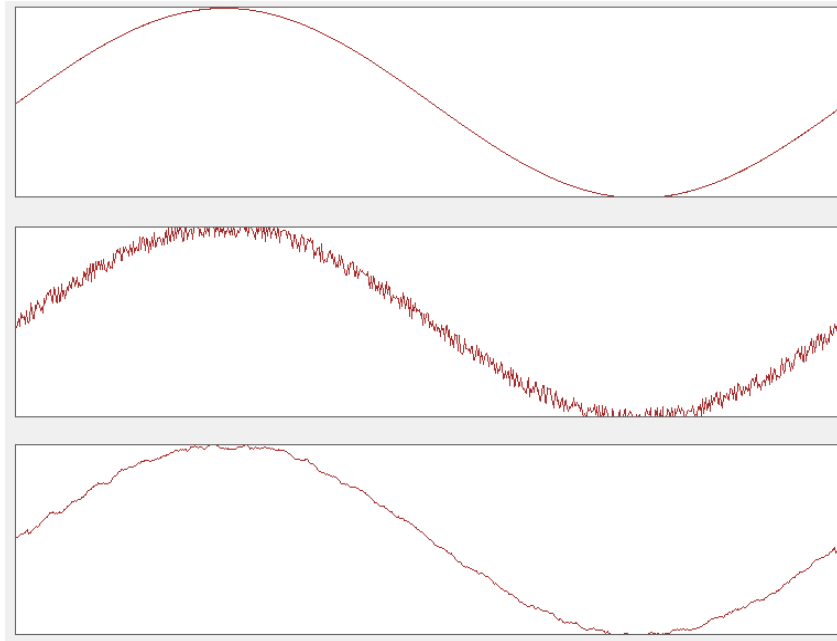
Az $a[k]$ vektor a konvolúció magfüggvénye (*kernel*).

A k számot pedig a konvolúció fokszáma.

FIR SZŰRŐ IMPLEMENTÁLÁSA

```
void filterFIR( float *array, int n, float *a, int window)
{
    float sum=0;
    for (int i = 0; i < (n - window); i++)
    {
        sum = 0.0;
        int k=0;
        // Loop through window numbers from current i position
        for (int j = i; j < i + window; j++)
        {
            sum += a[k]*array[j];
            k++;
        }
        array[i]=sum;
    }
}
```

MOZGÓÁTLAG



Az átlagoló szűrő gyakorlatilag a jelben minden egyes pontot egy adott sugarú környezetének átlagával helyettesít. A véletlen zaj kiszűrésére alkalmas.

A véletlen zaj olyan zaj, melynek spektruma adott frekvenciasávban nagyjából azonos teljesítmény sűrűségű. A véletlen zaj átlaga hosszútávon a nullához tart.

$$y_n = \frac{1}{N} \sum_{i=0}^{N-1} x_{n-i},$$

ÁTLAGOLÓ SZŰRŐ IMPLEMENTÁLÁSA

```
void movingAverage( float *array, int n, int window)
{
    float sum=0;
    for (int i = 0; i < (n - window); i++)
    {
        sum = 0.0;
        // Loop through window numbers from current i position
        for (int j = i; j < i + window; j++)
        {
            sum += array[j];           // Increment sum.
        }

        // Calculate moving average.
        float movingAverage = sum / window;
        array[i]=movingAverage;
    }
}
```

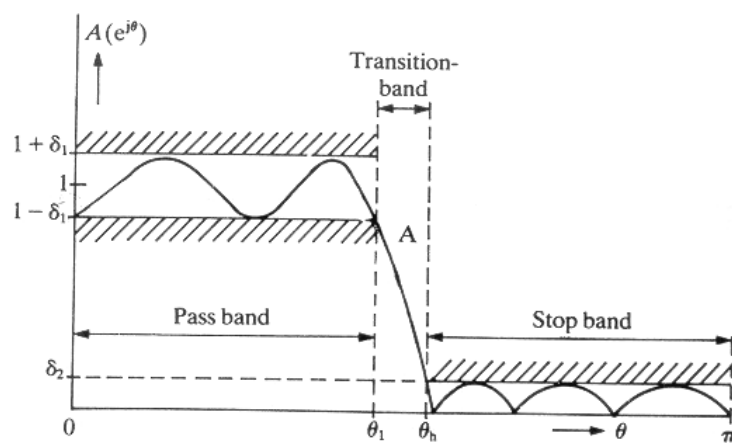
IIR SZŰRŐK

A rekurzív szűrők kimeneti jele nemcsak a bemeneti jeltől, hanem a szűrő korábbi kimeneti jelétől is függ.

$$y_n = \sum_{i=0}^k a_i x_{n-i} - \sum_{i=0}^k b_i y_{n-i}$$

ahol x_n a bemenő jel, a_i és b_i a szűrő viselkedését megadó függvény i -edik értéke, míg y_n a kimenő jel n -edik értéke.

DIGITÁLIS SZŰRŐK MÉRETEZÉSE



A szűrők méretezése a frekvencia tartománybeli amplitúdó karakterisztika ($H(w)$) magadásával kezdődik. Ez a gyakorlatban a tolerancia diagrammal történik.

A diagramon a δ_1 , θ_1 , és θ_h értékek megadásával határozzuk meg, hogy az amplitúdó karakterisztikát meddig tekintjük elfogadhatónak.

DIGITÁLIS SZŰRŐK MÉRETEZÉSE

1. Előállítjuk az $x[nT]$ diszkrét jel Fourier transzformáltját

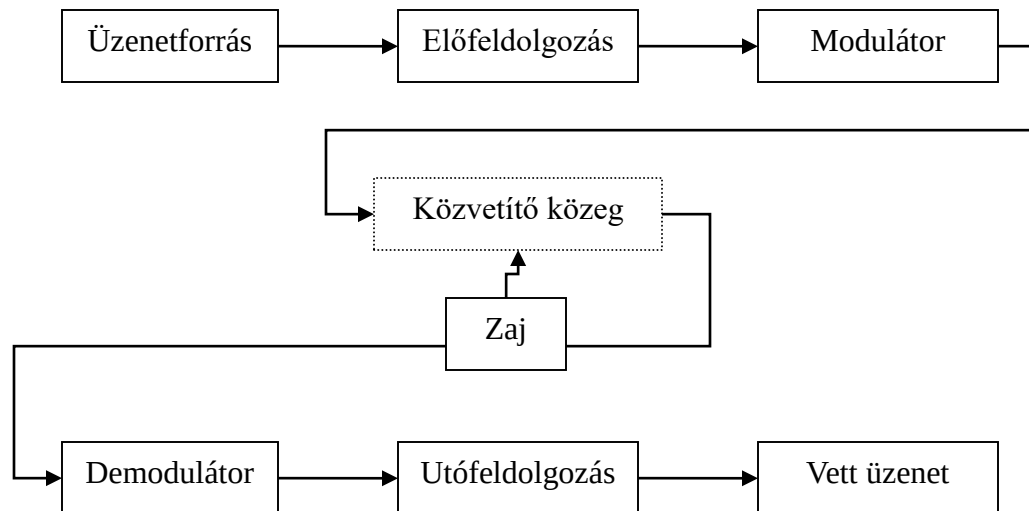
$$X(e^{j\omega T}) = \sum_{n=-\infty}^{n=+\infty} x[nT] e^{-jn\omega T}$$

2. Meghatározzuk, hogy FIR vagy IIR struktúrával akarjuk-e közelíteni az előírt paramétereket
3. Megadjuk a szűrő fokszámát és meghatározzuk az együtthatókat
4. Megválasztjuk a szűrő struktúrát a kvantálás hatásának a figyelembe vételével
5. Ellenőrizzük, hogy a szűrő eleget tesz –e az előírt specifikációknak. Ha nem, akkor módosítjuk a paramétereket és a módosítás után megismételjük az eljárást.

FELADATOK

- Mi az antialiasing szűrő? Milyen szabály alapján kell megválasztani a vágási frekvenciáját?
- Az A/D konverternek milyen feladatokat kell elvégeznie?
- Mit jelent a mintavételezés?
- Mondja ki a Shannon féle mintavételezési tételt!
- Írja fel a FIIR szűrő matematikai alakját!
- Írja fel az átlagoló szűrőt definiáló kifejezést.
- Mire használható az átlagoló szűrő?
- Írja fel a IIR szűrő matematikai alakját!

INFORMÁCIÓ ÁTVITEL



Átviteli közeg: fizikai útvonal az átviteli rendszer adója és vevője között.

A közegek lehetnek:

- vezető
- nem vezető

A kommunikáció mindkét esetben elektromágneses hullámokkal történik.

Vezető közegek:

- csavart érpáras rézvezeték
- koaxiális kábel
- optikai szál

Nem vezető közegek (Vezeték nélküli átvitel (wireless transmission)) :

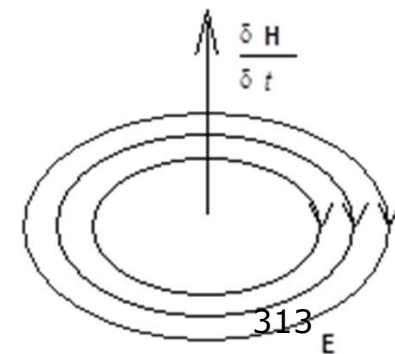
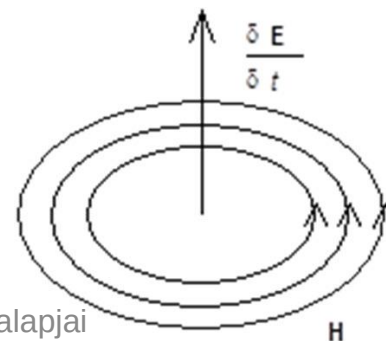
- atmoszféra
- űr

FIZIKAI ALAPOK - ELEKTRODINAMIKA

Maxwell egyenletek

Az elektromos és a mágneses mező kapcsolatát írja le.

- I. Az áramjárta vezető körül mágneses mező alakul ki. A mágneses mező örvényes „jobbkez szabály”
- II. Változó mágneses tér elektromos teret generál. Az elektromos mező örvénymentes. „balkéz szabály”
- III. A mágneses tér forrásmentes, az erővonalak önmagukba záródnak.
- IV. Az elektromos tér forrásos, az erővonalak nem záródnak önmagukba



MAXWELL EGYENLETEK

$$\text{I.} \quad \text{rot} \mathbf{H} \equiv \nabla \times \mathbf{H} = \mathbf{J} + \frac{\partial \mathbf{D}}{\partial t}$$

$$\oint \mathbf{H} \cdot d\mathbf{l} = \int_A \left(\mathbf{J} + \frac{\partial \mathbf{D}}{\partial t} \right) \cdot d\mathbf{A}$$

$$\text{II.} \quad \text{rot} \mathbf{E} \equiv \nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$$

$$\oint_L \mathbf{E} \cdot d\mathbf{l} = -\frac{\partial}{\partial t} \int_A \mathbf{B} \cdot d\mathbf{A}$$

$$\text{III.} \quad \text{div} \mathbf{B} \equiv \nabla \cdot \mathbf{B} = 0$$

$$\oint_A \mathbf{B} \cdot d\mathbf{A} = 0$$

$$\text{IV.} \quad \text{div} \mathbf{D} \equiv \nabla \cdot \mathbf{D} = \rho$$

$$\oint_A \mathbf{D} \cdot d\mathbf{A} = \int_V \rho \cdot dV$$

$$\mathbf{D} = \varepsilon \cdot \mathbf{E} \quad \mathbf{B} = \mu \cdot \mathbf{H}$$

Az átviteli közeg bemenetére adott jel a különböző csillapító és torzító hatások miatt változik, ezért a közeg kimenetén megváltozott jelet érzékelhetünk. A jel átvitelét befolyásoló csillapító és torzító hatások:

- Csillapítás
- Korlátozott sáv szélesség
- Jelkésleltetés
- Vonali zaj

A fenti torzító hatások minden átviteli közegben fellépnek, és együttesen jelentkeznek. A jelek ilyen hátrányos megváltozása hatással van

- az áthidalható távolságra,
- az átvitel sebességére,
- az átvitel megbízhatóságára.

CSILLAPÍTÁS

A jel amplitúdója csökken a jel haladása során az átviteli közegben. Az átviteli közeg hosszát úgy állapítják meg, hogy a jel biztonsággal értelmezhető legyen a vételi oldalon.

Ha nagyobb távolságot kell áthidalni, akkor erősítők (jelismétlők) beiktatásával kell a jelet visszaállítani. A csillapítás frekvenciafüggő, ezért az erősítőknek frekvenciafüggő erősítéssel kell ezt kompenzálniuk.

A csillapítás és az erősítés mértékét *decibel*ben (dB) adják meg:

$$\text{Csillapítás} = 10 \log_{10} \frac{P_1}{P_2} \text{ dB}$$

ahol P_1 és P_2 az átviteli közeg elején és végén mért teljesítmény (Watt).

KORLÁTOZOTT SÁVSZÉLESSÉG

Minden átviteli közeg csak korlátozott frekvenciatartomány (sávszélesség) csillapítás mentes átvitelére képes. Az átviteli közeg sávszélessége és az adat átviteli sebessége között szoros összefüggés van. *Nyquist* meghatározta a maximális adatátviteli sebességet zajtalan csatornára: Ha a jel V diszkrét értékből áll, akkor a

$$C = 2H \log_2 V \text{ bit/s}$$

ahol C a maximális adatátviteli sebesség,
 H az átviteli csatorna sávszélessége.

2-nél több jelszint alkalmazásával jelváltásonként több bitet is kódolhatunk. Például 8 jelszintet használva egy 3000 Hz-re sávkorlátozott telefonvonalon:

$$C = 2 \cdot 3000 \log_2 8 = 18000 \text{ bit/s}$$

JELKÉSLELTETÉS (DELAY)

A jel terjedésének ideje frekvenciafüggő, ezért a jelek szinuszos komponensei eltérő időben érkeznek a vevőhöz, és ún. késleltetési torzítást okoznak.

A torzítás mértéke az adatátviteli sebesség növelésével növekszik.
Ennek oka, hogy bizonyos bitekhez tartozó néhány frekvencia komponens olyan mértékben késik, hogy interferál a következő bit bizonyos frekvencia komponenseivel.

Ezt a torzítást szimbólumok közötti interferenciának (intersymbol interference) nevezik.

VONALI ZAJ (NOISE)

Az átviteli közeg környezetéből származó zavarokat vonali zajnak nevezik.

Az átvitt jelek csillapítása miatt a zajszint összemérhetővé válhat a jelszinttel, és a jelek helyes érzékelése lehetetlenné válhat.

Az átviteli médiumok jellemezhetők az átlagos jelteljesítmény (Signal) és zajteljesítmény (Noise) hányadosával: S/N

Shannon meghatározta a maximális adatátviteli sebességet zajos csatornára:

$$C = H \log_2(1 + S/N) \text{ bit/s}$$

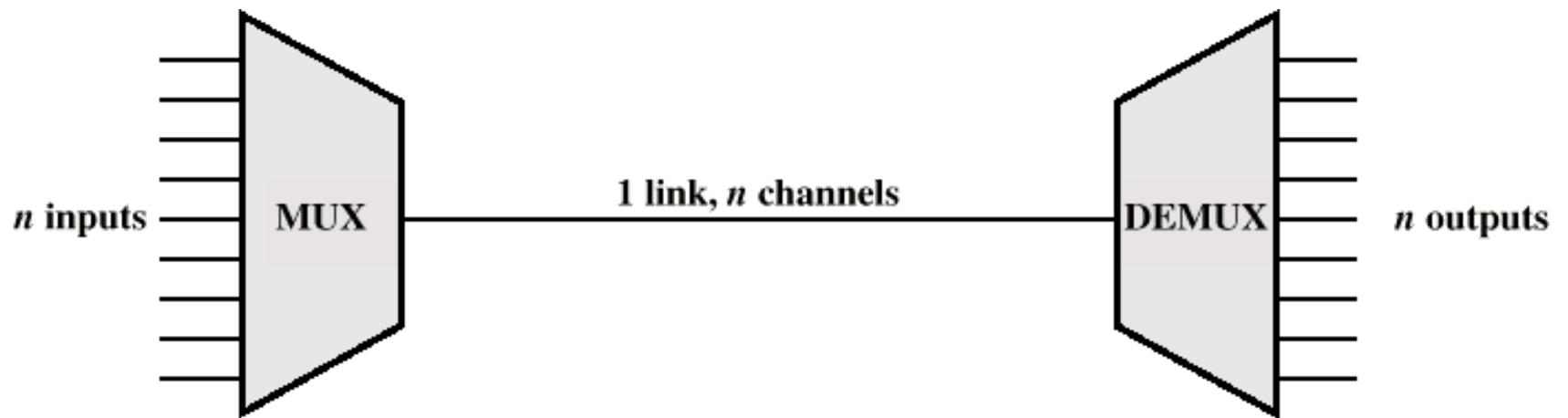
ahol C a maximális adatátviteli sebesség,

H az átviteli csatorna sávszélessége,

S az átlagos jelteljesítmény,

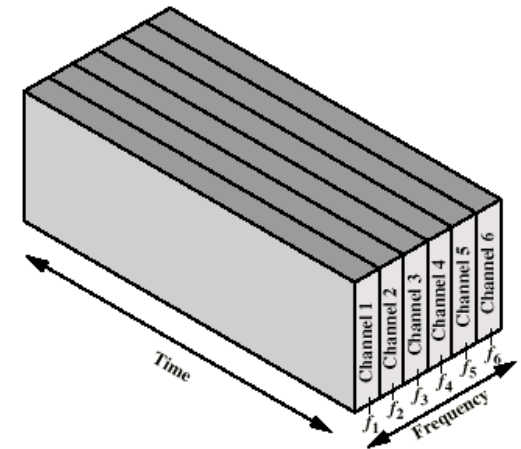
N az átlagos zajteljesítmény.

MULTIPLEXING

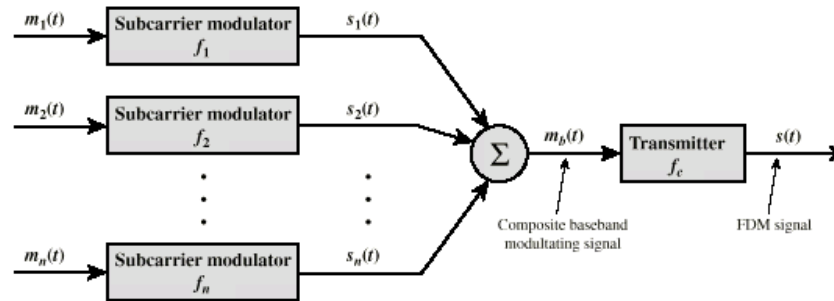


FDM - FREKVENCIA OSZTÁSOS MULTIPLEXING

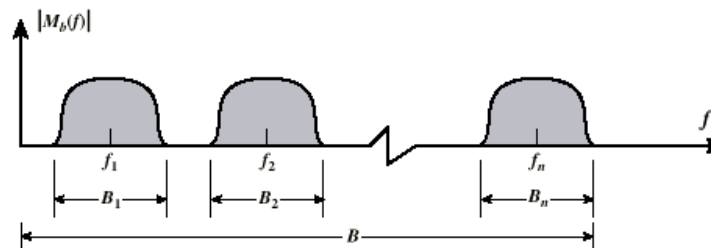
- Akkor lehet használni, ha az átviteli közeg sávszélessége nagyobb mint az átvíendő csatornák együttes sávszélessége.
- Minden csatornának külön vivőfrekvenciája van, melyek olyan távolságra vannak egymástól, hogy a hasznos jel átvitelekor ne legyenek átfedések a csatornák között.
- A csatorna akkor is foglalja a helyet, ha nincs rajta adat
- Pl. Rádió és televízió műsorszórás



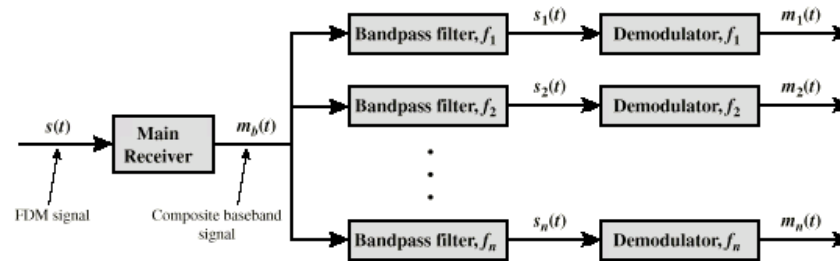
FDM RENDSZER



(a) Transmitter

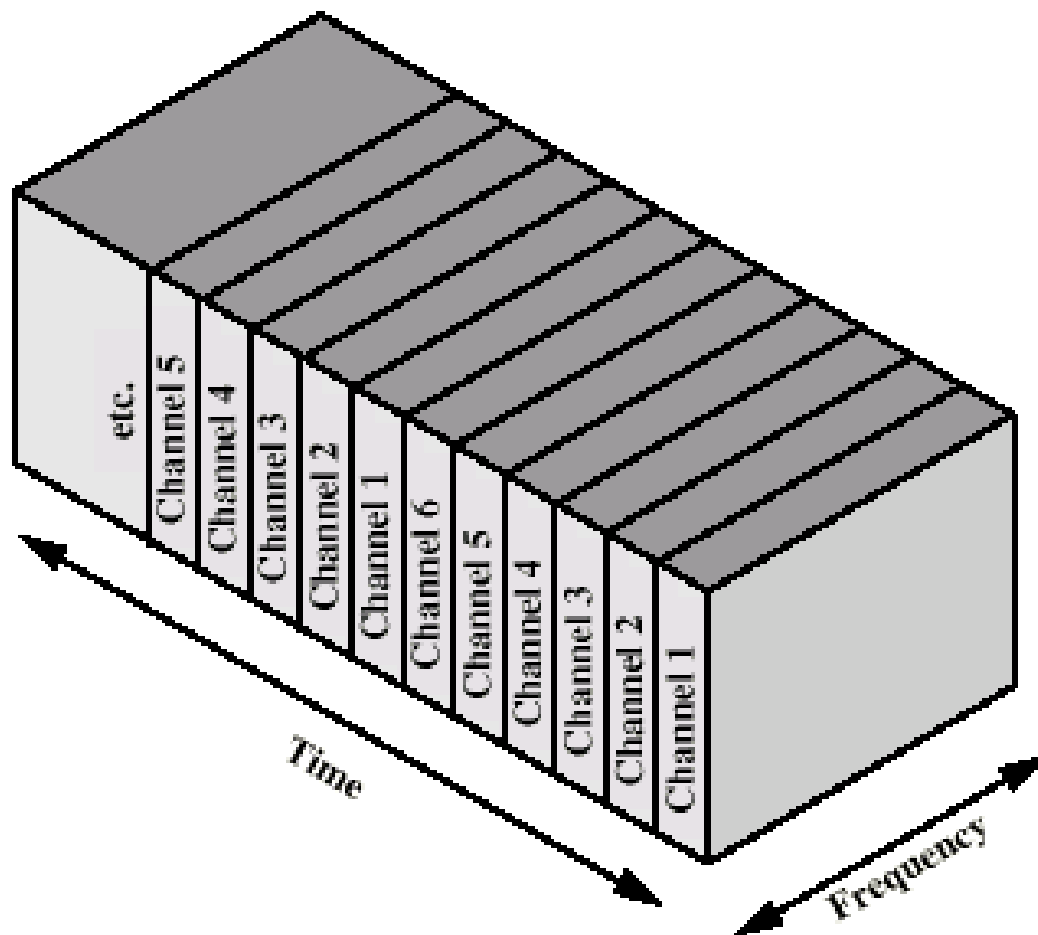


(b) Spectrum of composite baseband modulating signal

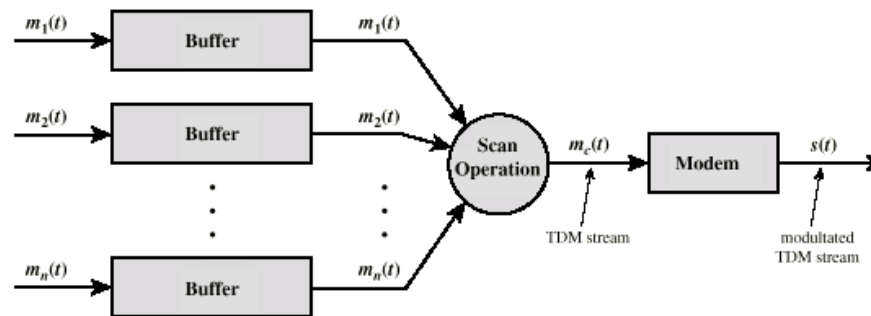


(c) Receiver

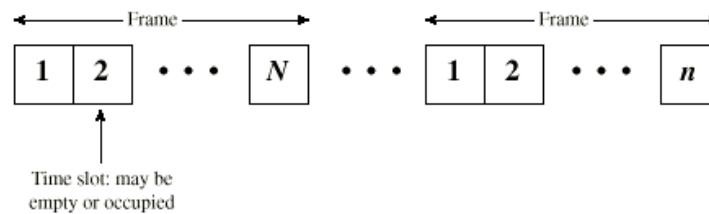
TDM - IDŐOSZTÁSOS MULTIPLEXING



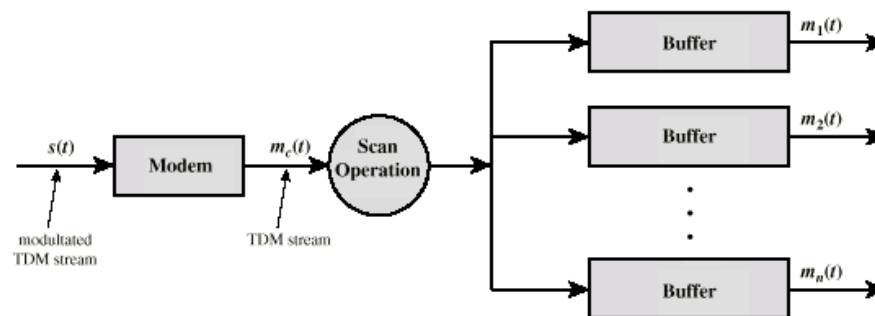
TDM RENDSZER



(a) Transmitter



(b) TDM Frames

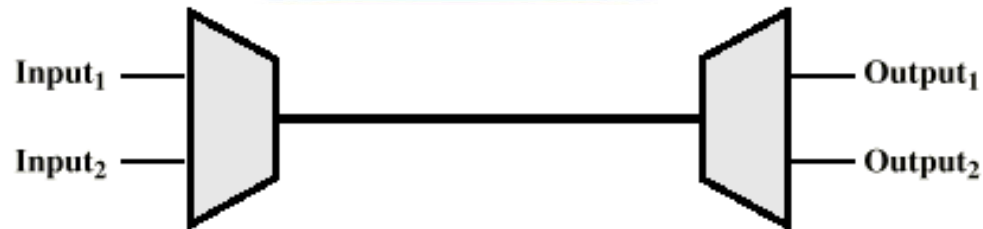


(c) Receiver

SZINKRON TDM

- Akkor használható, ha az átviteli közeg adatátviteli sebessége nagyobb mint az információ forrás adattermelési sebessége
- A csatornák jelei időben elosztva kerülnek rá az átviteli közegre
- Az egyes csatornákhöz rendelt időszeletek akkor is le vannak foglalva a csatorna számára, ha a csatornán nincs adat.
- A csatornák számára lefoglalt időszeleteket nem köztelező azonos hosszúságúra választani

ADAT VEZÉRLÉS A TDM RENDSZERBEN



(a) Configuration

Input₁..... F₁ f₁ f₁ d₁ d₁ d₁ C₁ A₁ F₁ f₁ f₁ d₁ d₁ d₁ C₁ A₁ F₁
 Input₂... F₂ f₂ f₂ d₂ d₂ d₂ d₂ C₂ A₂ F₂ f₂ f₂ d₂ d₂ d₂ d₂ C₂ A₂ F₂

(b) Input data streams

... f₂ F₁ d₂ f₁ d₂ f₁ d₂ d₁ d₂ d₁ C₂ d₁ A₂ C₁ F₂ A₁ f₂ F₁ f₂ f₁ d₂ f₁ d₂ d₁ d₂ d₁ d₂ d₁ C₂ C₁ A₂ A₁ F₂ F₁

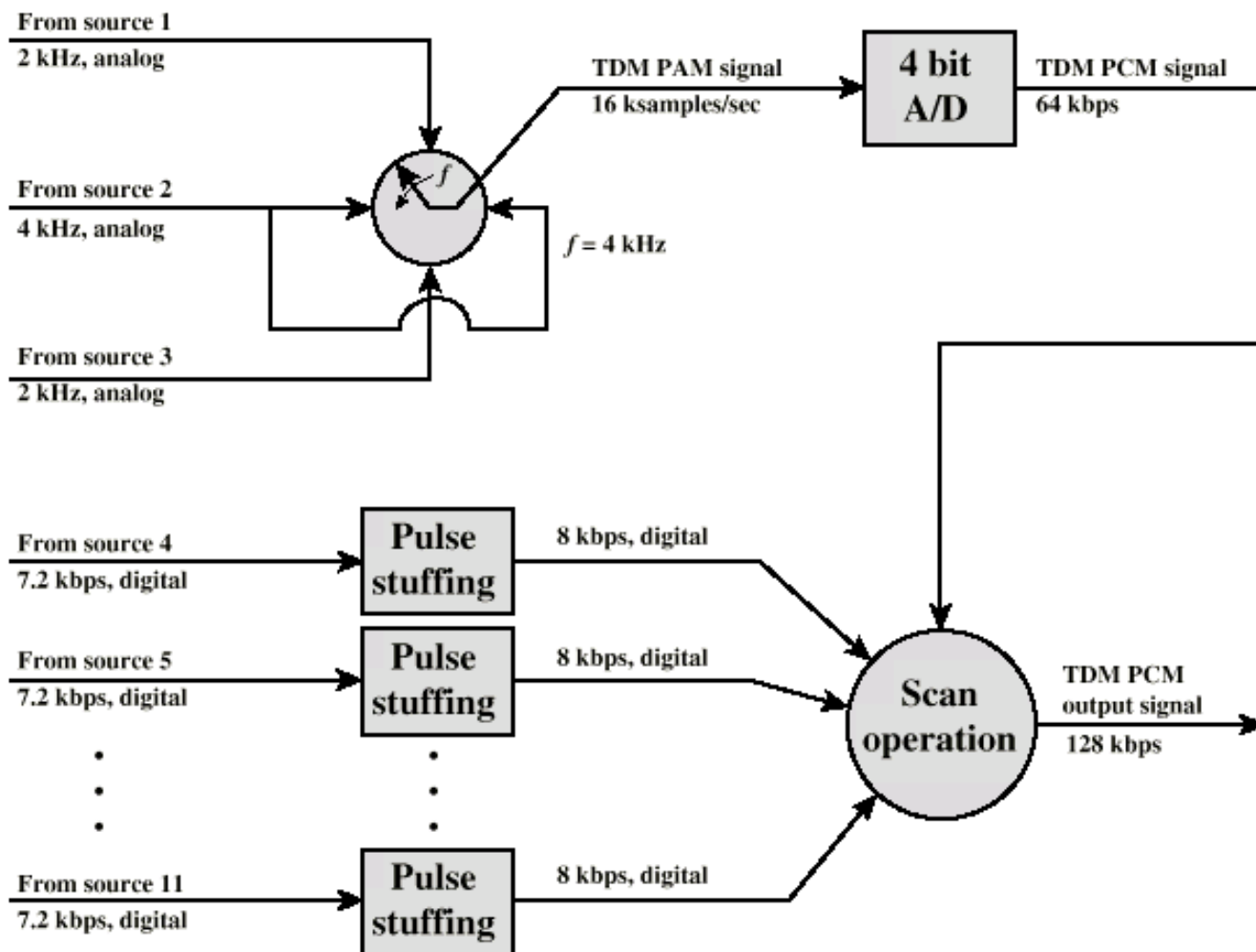
(c) Multiplexed data stream

Legend: F = flag field d = one octet of data field
 A = address field f = one octet of FCS field
 C = control field

KERETEZÉS

- Fizikai szinten a TDM nem használ szinkronizáló jeleket
- A szinkronizálást keretezéssel lehet megoldani
- A keret tartalmazhat:
 - kontroll biteket
 - azonosító bit mintát
 - keret nagyságra vonatkozó információt
 - Hiba ellenőrzést

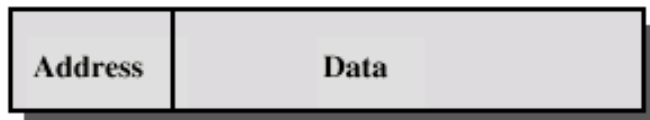
ANALÓG ÉS DIGITÁLIS JELEK IDŐOSZTÁSA



STATISZTIKUS TDM KERET FORMÁTUM



(a) Overall frame



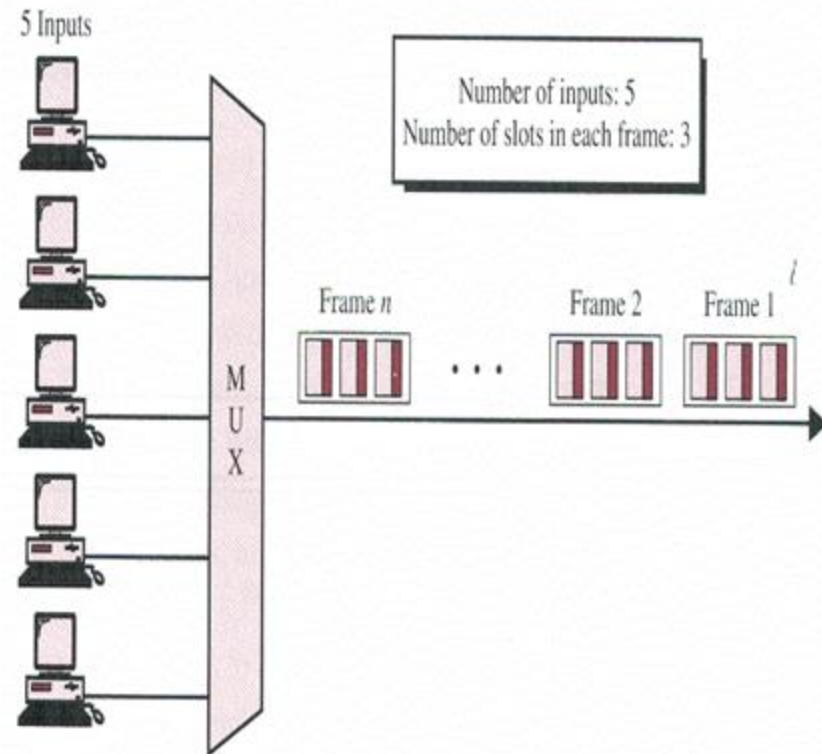
(b) Subframe with one source per frame



(c) Subframe with multiple sources per frame

ASZINKRON TDM

- A szinkron TDM nem garantálja az átviteli közeg teljes kapacitásának kihasználását, mert az időosztás a csatornák között előre megvan határozva.
- Az aszinkron TDM-et, valamint a statisztikus TDM-et arra találták ki, hogy az átviteli közeg teljes kapacitását kihasználja.



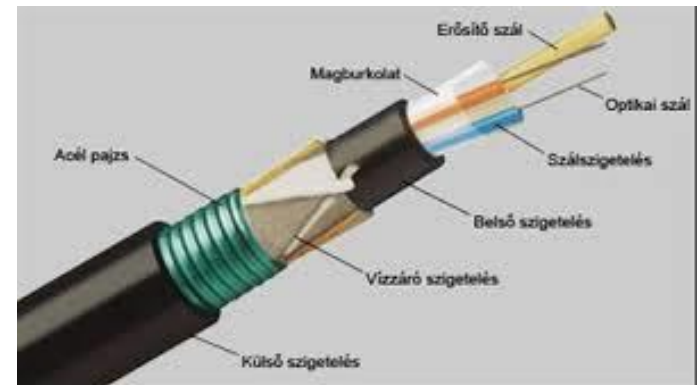
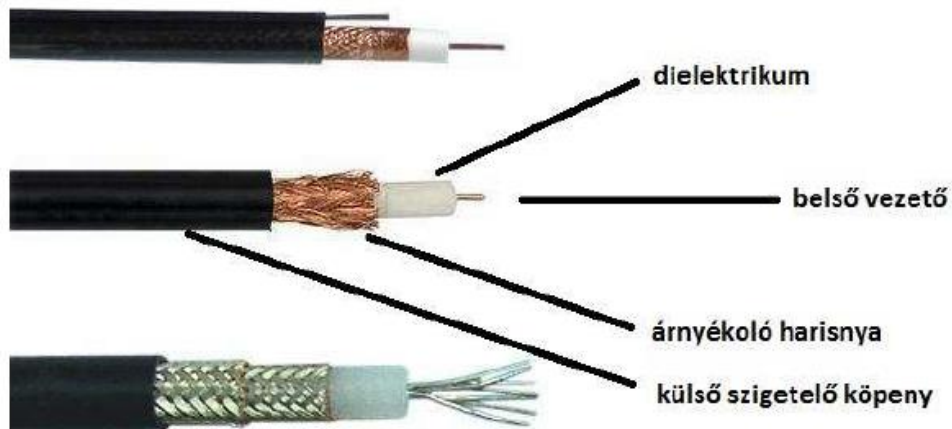
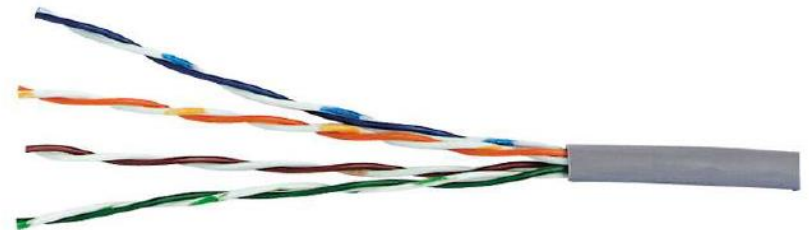
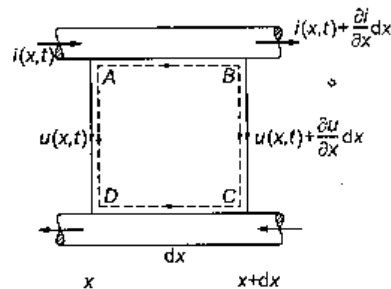
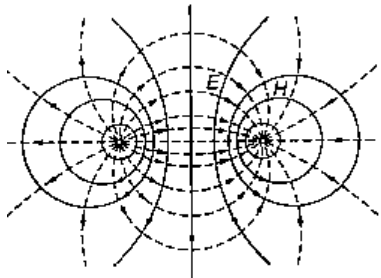
FELADATOK

- Rajzolja le a kommunikációs rendszerek általános sémáját!
- Mi a vonali zaj?
- Mit jelet a multiplexing?
- Mi az aszinkron TDM?

A távvezeték

- mérete a hullámhosszal összemérhető, vagy annak sokszorosa,
- két egymással párhuzamos vezetőből áll,
- a vezetőknek véges ellenállásuk és induktivitásuk van, a vezetők között kapacitás és átvezetés van,
- a vezetők árama, a vezetők közötti feszültség, a z koordináta és t idő függvénye
- az egyik vezető árama a z helyen $i(z,t)$, a másiké $-i(z,t)$. A két vezető közötti feszültség $u(z,t)$,
- az áram és a feszültség időfüggését a távvezetékre kapcsolt gerjesztés időbeli változása szabja meg.

TIPIKUS TÁVVEZETÉKEK



TÁVÍRÓ EGYENLETEK

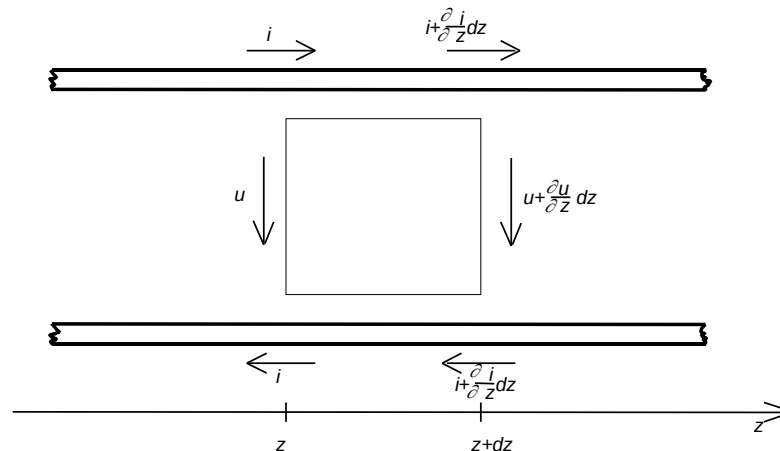
A távvezeték feszültségeinek és áramainak összefüggését a **távíró egyenletek** adják:

$$-\frac{\partial u}{\partial z} = Ri + L \frac{\partial i}{\partial t}$$

$$-\frac{\partial i}{\partial z} = Gu + C \frac{\partial u}{\partial t}$$

$$-\frac{\partial U}{\partial z} = (R + j\omega L)I$$

$$-\frac{\partial I}{\partial z} = (G + j\omega C)U$$



TÁVÍRÓ EGYENLETEK MEGOLDÁSA

- szinuszos gerjesztés esetén vizsgáljuk a távvezetékét,
- ismertnek tekintjük az R,L,G,C hosszegységre vonatkozó jellemzőket,
- a távíró egyenleteket z szerint differenciálva, az első egyenletből I -t a másodikból U -t kiküszöbölve:

$$\begin{aligned} \frac{\partial^2 U}{\partial z^2} - \gamma^2 U &= 0 \\ * \quad \frac{\partial^2 I}{\partial z^2} - \gamma^2 I &= 0 \end{aligned} \quad \gamma = \sqrt{(R + j\omega L)(G + j\omega C)} = \alpha + j\beta$$

- γ a komplex terjedési tényező, α a csillapítási tényező, β a fázistényező

$$\begin{aligned} \alpha &= \sqrt{\frac{1}{2} \left[RG - \omega^2 LC + \sqrt{(R^2 + \omega^2 L^2)(G^2 + \omega^2 C^2)} \right]} \\ \beta &= \sqrt{\frac{1}{2} \left[\omega^2 LC - RG + \sqrt{(R^2 + \omega^2 L^2)(G^2 + \omega^2 C^2)} \right]} \end{aligned}$$

TÁVÍRÓ EGYENLETEK MEGOLDÁSA

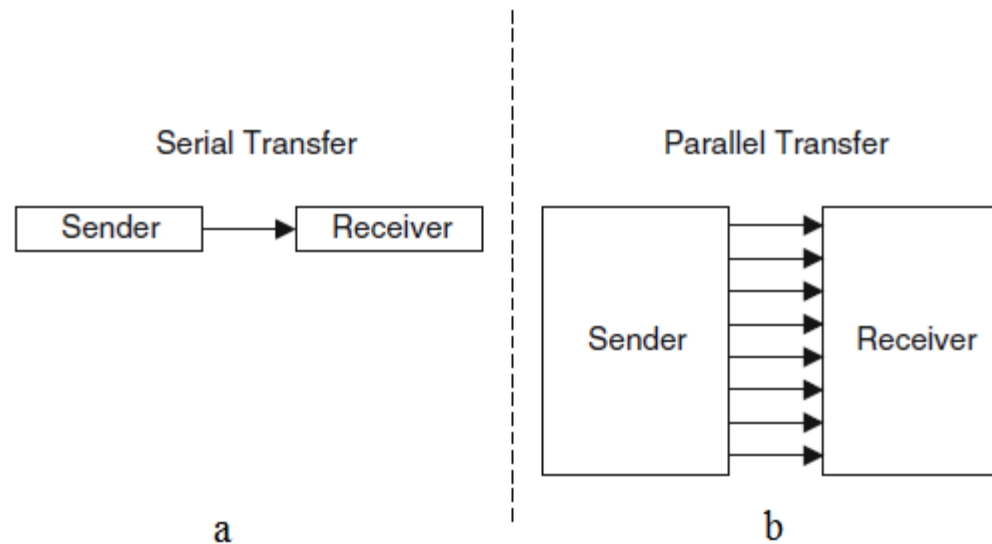
- A $\star$ egyenlet megoldása szinuszos gerjesztés esetén:

$$\begin{aligned} u(z,t) &= U^+ e^{-\gamma z} e^{j\omega t} + U^- e^{\gamma z} e^{j\omega t} \\ i(z,t) &= \frac{U^+}{Z_0} e^{-\gamma z} e^{j\omega t} - \frac{U^-}{Z_0} e^{\gamma z} e^{j\omega t} \end{aligned} \quad Z_0 = \sqrt{\frac{R + j\omega L}{G + j\omega C}}$$

- Z_0 a távvezeték hullámimpedanciája,
- U^+ és U^- a z tengely pozitív, illetve negatív irányában terjedő hullámok komplex amplitúdóit jelölik,
- ha $R=0$ és $G=0 \rightarrow$ csillapítatlanul terjedő hullámokat kapunk $\rightarrow$ *ideális távvezeték, melynek jellemzői a következők:*

$$\begin{aligned} \gamma &= j\beta = j\omega\sqrt{LC} & Z_0 &= \sqrt{\frac{L}{C}} \\ v &= \frac{\omega}{\beta} = \frac{1}{\sqrt{LC}} = \frac{1}{\sqrt{\epsilon\mu}} = \frac{c}{\sqrt{\epsilon_r}\sqrt{\mu_r}} \\ \lambda &= \frac{2\pi}{\beta} = \frac{v}{f} = \frac{c}{f} \frac{1}{\sqrt{\epsilon_r}\sqrt{\mu_r}} = \frac{\lambda_0}{\sqrt{\epsilon_r}\sqrt{\mu_r}} \Rightarrow \beta = \frac{2\pi}{\lambda_0} \sqrt{\epsilon_r}\sqrt{\mu_r} \end{aligned}$$

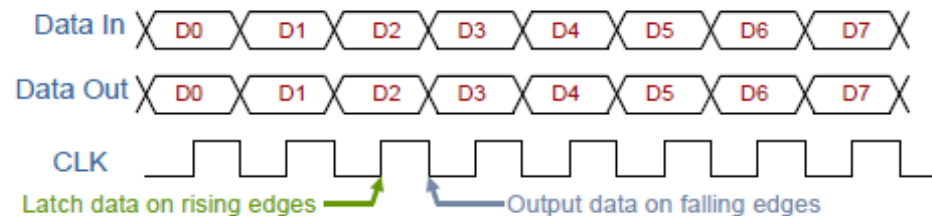
ADATTOVÁBBÍTÁS



Soros (a) és párhuzamos (b) adattovábbítás

SOROS VONALI KOMMUNIKÁCIÓ

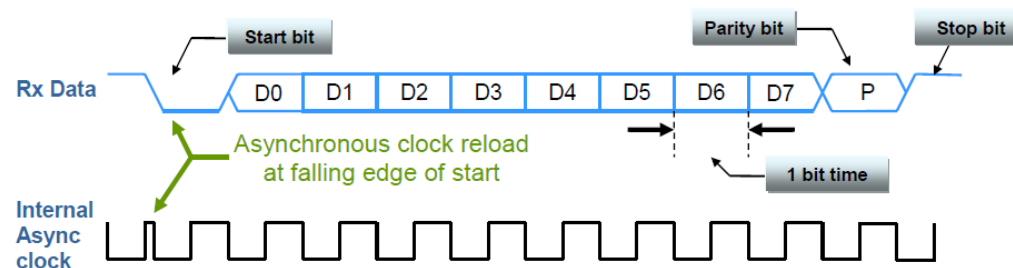
Két eszköz kommunikál egymással: az *adó (Transmitter)* és a *vevő (Receiver)*. A kommunikációs csatornán (vonalon) az adatok bitekre lebontva adott időütem szerint ütemezve bitenként továbbítódnak az adótól a vevőig. Az átvitel sebességét az időütemezés határozza meg. A soros kommunikációnak kétféle változata van: *szinkron* és *aszinkron*.



RS232 PROTOKOLL

Az aszinkron soros kommunikációnál az adó és a vevő csak egyetlen karakter továbbításának idejéig szinkronizálódnak össze.

A D0, D1,...,D7 adatbitek kiegészítő bitek (Start bit, Paritás bit, Stop bit) veszik körül. Az adás kezdetét a start bit lefutó éle jelzi, amely mindig alacsony szintű átmenet. Ezt követik az adatbitek. Az adatbitek a paritás bit követi. A szabvány szerint ennek megadása nem kötelező. Az adatátvitel végét a stop bit jelzi.



RS232 PROTOKOLL PARAMÉTEREI

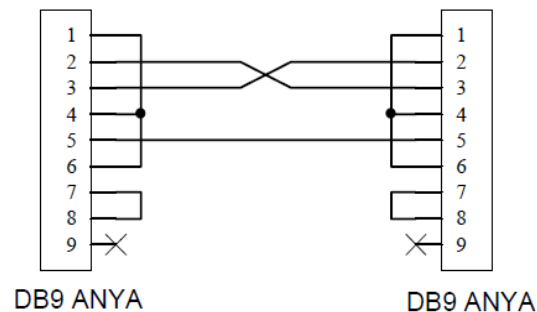
- Az *adatbitek* száma a szabvány szerint 5,6,7 és 8 lehet.
- Szintén a szabvány szerint a *stop bitek* száma 1, 1.5 és 2 lehet. A méret itt természetesen a stop bit idejére vonatkozik, azaz 1.5 stop bit időtartama másfél bitütemet jelent.
- A kommunikáció során az adatátviteli hiba detektálására *paritás biteket* használhatunk. A szabvány szerint ez 0, 1 vagy 2 lehet. A 0 esetén nem használunk paritás ellenőrzést (NO_PARITY). Az 1 esetén páratlan paritás ellenőrzés van (ODD_PARITY). A 2 esetén pedig páros paritás ellenőrzés van. (EVEN_PARITY).
- Az átvitel *sebessége* (BAUD RATE) az időegység alatt átvitt bitek száma. Mértékegysége a *bps* (bit/s). Az alkalmazható Baud Rate-re nincs elvi megkötés. Korlátokat csak az alkalmazott átviteli közeg sávszélessége valamint a rendelkezésre álló órajel jelent. A megbízható kommunikációhoz azonban az RS232 szabvány rögzíti a megengedett sebességeket.

RS232 KÁBEL BEKÖTÉS

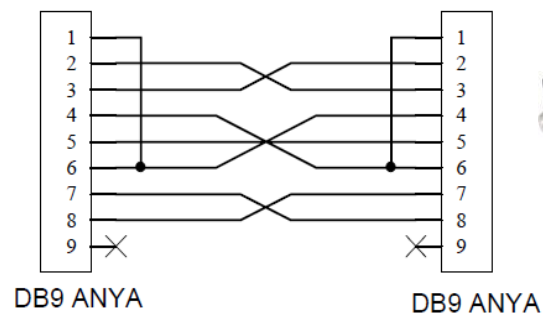
DB9 KIOSZTÁSA

1. DCD - Data Carrier Detect
2. RxD - Receive Data
3. TxD - Transmit Data
4. DTR - Data Terminal Ready
5. SG - Signal Ground
6. DSR - Data Set Ready
7. RTS - Request To Send
8. CTS - Clear To Send
9. RI - Ring Indicator

3 Vezetékes



Teljes



TOVÁBBI SOROS VONALI PROTOKOLLOK

- RS485
Az RS485 szabvány programozói szempontból megegyezik az RS232-vel, azonban ez hardver szinten több eszköz között megvalósított kommunikáció.
- USB
Az USB-t (Universal Serial Bus) az RS232 szabvány felváltására hozták létre.
- SPI
Az SPI (Serial Peripheral Interface) a Motorola cég fejlesztése. Master/Slave rendszerű kommunikációt valósít meg.
- I²C
Az I²C (Inter Integrated Circuit) a Philips cég fejlesztése. Multi master kommunikációt valósít meg.

SOROSVONAL INICIALIZÁLÁS

```
BOOL CSerial::Init( const char *strPort,int nBaud, int nData, BYTE Parity, BYTE stopBit, DWORD
RXQUEUE, DWORD TXQUEUE)
{
    DCB    dcb;
    BOOL    bRc;

    CString sPort;
    int portNum=atoi(&strPort[3]);
    if(portNum>3)
        sPort.Format(_T("\\\\.\\COM%d"), portNum);
    else
        sPort.Format("%s", strPort);

    Close();
    m_hCommPort = CreateFile(sPort,GENERIC_READ|GENERIC_WRITE,0,NULL,
        OPEN_EXISTING,FILE_ATTRIBUTE_NORMAL,NULL);
    if( m_hCommPort==(HANDLE)-1)
    {
        return FALSE;
    }

    SetupComm(m_hCommPort,RXQUEUE,TXQUEUE);
    PurgeComm(m_hCommPort,PURGE_TXABORT|PURGE_RXABORT|PURGE_TXCLEAR|PURGE_RXCLEAR);

    dcb.DCBlength  = sizeof(DCB);

    if (!GetCommState(m_hCommPort,&dcb))
    {
        return FALSE;
    }
    dcb.BaudRate = nBaud;
    dcb.ByteSize = (unsigned char)nData;
    dcb.Parity    = Parity;
    dcb.StopBits = stopBit;

    dcb.fOutxCtsFlow= 0;    // no cts flow control
    dcb.fOutxDsrFlow= 0;    // no dsr flow control

    bRc = SetCommState(m_hCommPort,&dcb);
    return bRc;
}
```

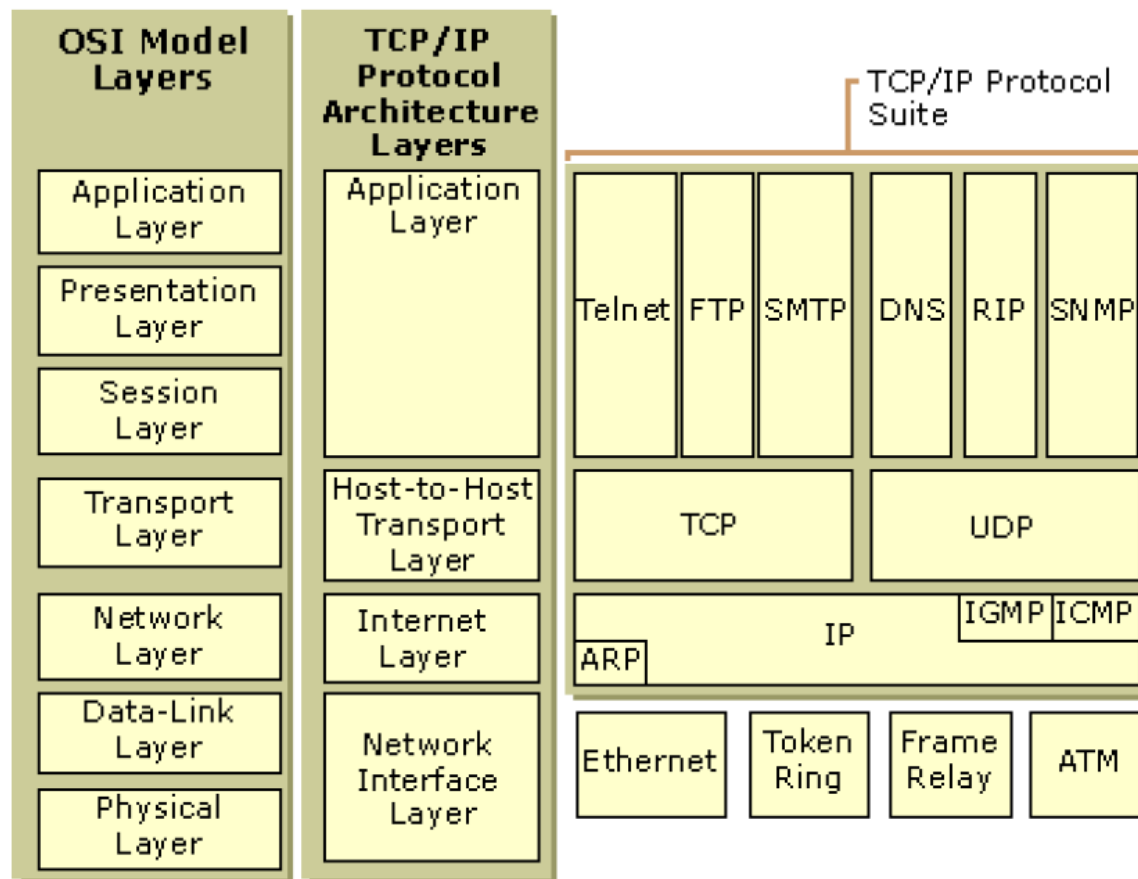
ADATOLVASÁS A SOROS PORTTÓL

```
BOOL CSerial::ReadByte( BYTE *B)
{
    DWORD  dwNumRead,errorcode;
    if( m_hCommPort!=(HANDLE)-1)
    {
        COMSTAT ComStat;

        if(!ClearCommError(m_hCommPort,&errorcode,&ComStat) || !ComStat.cbInQue)
            return FALSE;

        ReadFile( m_hCommPort,B,1,&dwNumRead,NULL);
        if( dwNumRead!=1)
            return FALSE;
        else
            return TRUE;
    }
    return FALSE;
}
```

TCP/IP PROTOCOL RÉTEGEK



TCP (TRANSPORT CONTROL PROTOCOL)

Az IP protokollra épülve megbízható hálózati átvitelt garantál.

- A TCP – akár egy telefontársaság – garantálja, hogy a küldött adat helyes sorrendben érkezzen meg a vevőhöz. Ha ez nem sikerül, hibajelzést küld vissza.
- Olyankor alkalmazzák, amikor a hálózaton átküldött és fogadott adatok sorrendje elengedhetetlen az alkalmazások helyes működéséhez.
- TCP-t használó legismertebb szolgáltatások:
Hyper Text Transfer Protokoll (HTTP),
fájlok átvitele távoli gépekre (*FTP: file transfer*),
távoli gépekre történő belépés (*telnet*),
elektronikus mail (e-mail) szolgáltatás alapját adó *SMTP* protokoll.

UDP (USER DATAGRAM PROTOCOL)

Nem ad megbízható kapcsolatot egy hálózaton levő két számítógép között, mivel ez nem kapcsolat alapú, mint a TCP, inkább az adatok független csomagjait küldi az alkalmazások között. Ezeket a csomagokat adatcsomagoknak is hívjuk.

- A adatcsomagok küldése leginkább a levelek postán át való küldéséhez hasonlít: a szállítás nem garantált, a sorrend nem fontos, és mindegyik üzenet különbözik a másiktól.
- UDP-t használó szolgáltatások:
 - távoli gépeken dolgozó felhasználók azonosítójának lekérdezése (*rwho*),
 - fájlok hordozása (*TFTP*),
 - ping*.

A számítógép egy egyszerű fizikai kapcsolaton keresztül csatlakozik a hálózatra. Honnan tudja a számítógép, hogy melyik alkalmazásnak melyik adatot kell továbbítani?



Port-ok használatával

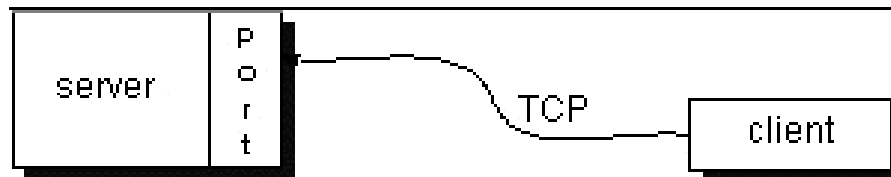
A portok 16 bites számként vannak ábrázolva $\Rightarrow$
értékük 0 - 65535 közötti szám lehet

A 0 és 1023 közötti portok fent vannak tartva olyan ismert szolgáltatásoknak, mint például a HTTP vagy az FTP vagy más rendszerszolgáltatás.

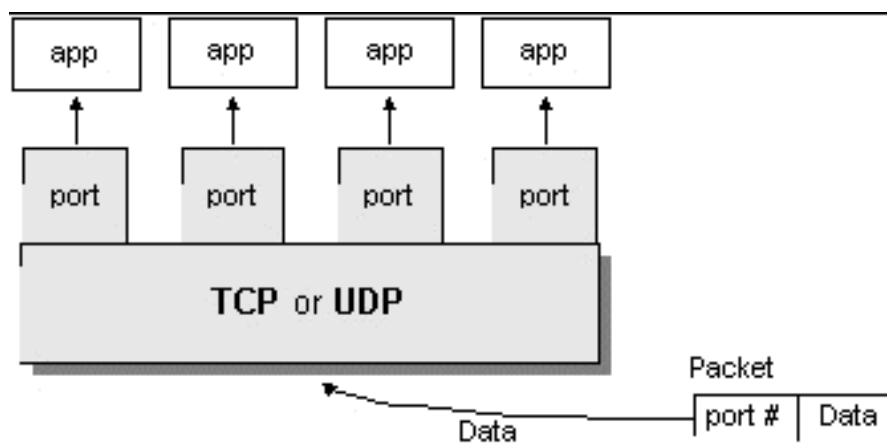
Egy internetes szolgáltatás elérési címe:
IP CÍM : port

A kapcsolat alapú kommunikációnál (TCP) a szerver program leköt egy foglalatot egy jellemző port számára. Így a szerver megkap minden adatot, ami ezen a porton keresztül érkezik.

A kliens ezután a megadott porton keresztül kommunikálhat a szerverrel.



Az adatcsomag alapú kommunikációnál (UDP) az adatcsomagok tartalmazzák a célállomás portszámát, és az UDP irányítja a megfelelő helyre a csomagot.



A socket angol szó, jelentése:
csatlakozó, foglalat, lyuk, tok.

A socket különböző processzek (folyamatok) közötti kommunikációs eszköz, amely független attól, hogy a kommunikáló processzek azonos, vagy különböző gépen vannak-e.

A kommunikáció formája kliens-szerver alapú.
Ez azt jelenti, hogy a szerver valamilyen jól definiált szolgáltatást nyújt, amit a kliensek igénybe vesznek.

Egy Socketnek két vége van: egy a szerver oldalán és egy a kliens oldalán.

Az adatfolyam socket jellemzői:

A TCP protokollra támaszkodó, a telefonhálózathoz hasonló kapcsolat-orientált byte-stream

Az adatcsomag socket (datagram socket) jellemzői:

Kapcsolat nélküli, a postai levélhez hasonló, az UDP protokollra támaszkodó megoldás

FELADATOK

- Milyen fizikai mennyiségek között teremtenek kapcsolatot a távíró egyenletek?
- Milyen soros adattovábbító protokollokat ismer?
- Milyen paramétereit vannak az RS232 protokollnak?
- Mi a socket alapú kommunikáció?

ELEKTROMÁGNESES HULLÁMOK

A Maxwell-egyenletek egyik legérdekesebb tulajdonsága, hogy ezeknek az egyenleteknek a távíró egyenletek mellett van hullámmegoldásuk is.

Vákumra vonatkozólag ezek a következők:

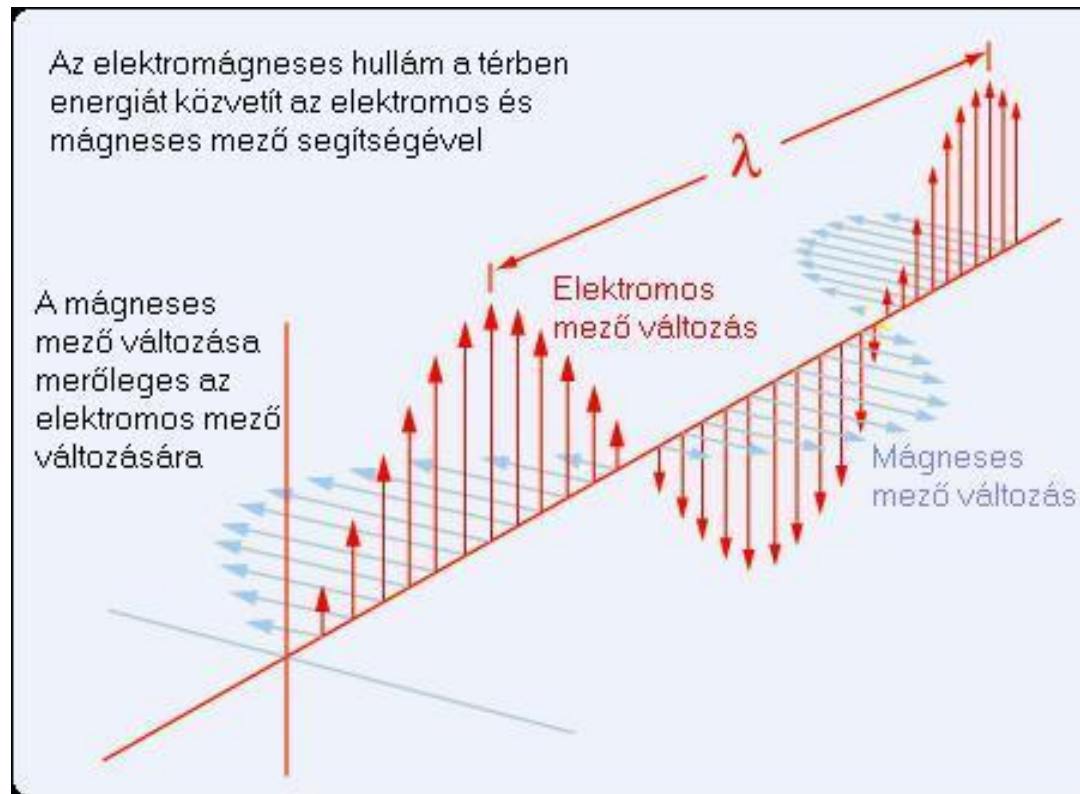
Elektromos komponens

$$\frac{\partial^2 \mathbf{E}}{\partial t^2} = c^2 \Delta \mathbf{E}$$

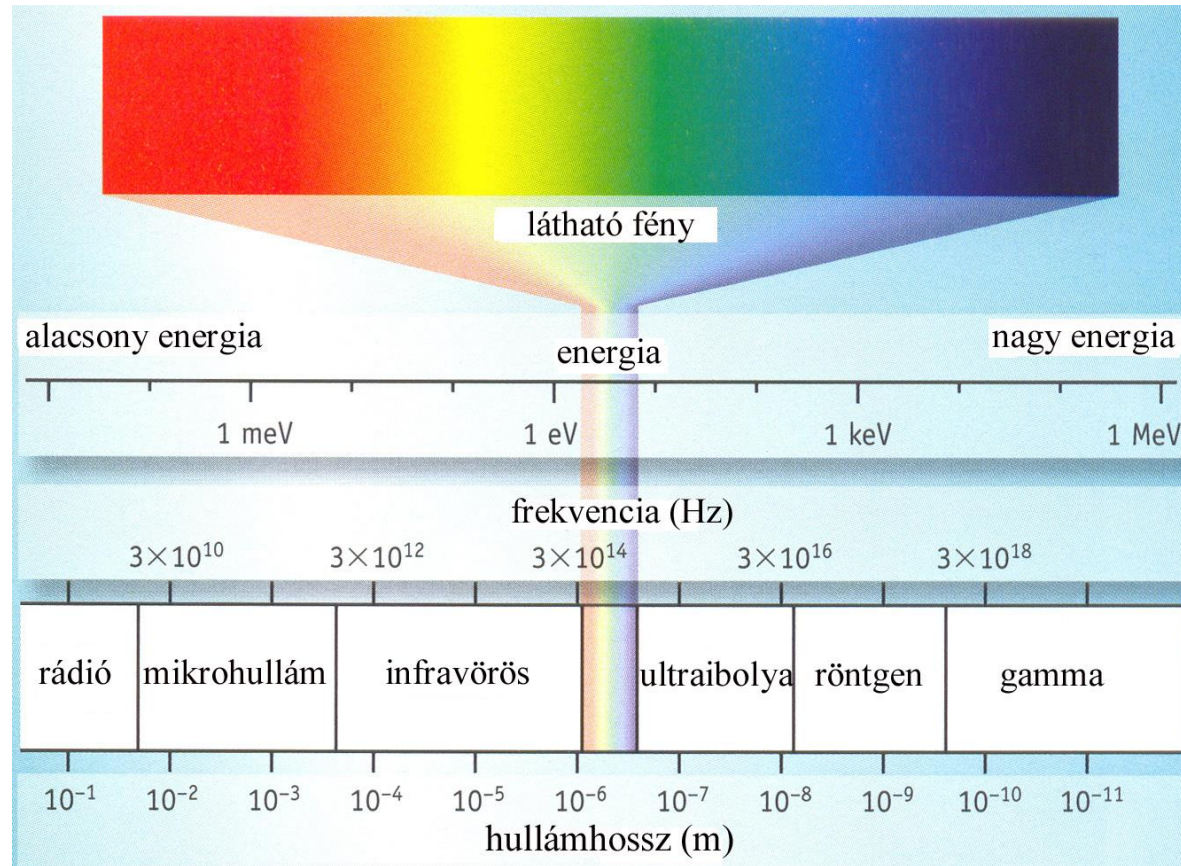
Mágneses komponens

$$\frac{\partial^2 \mathbf{H}}{\partial t^2} = \frac{1}{\epsilon_0 \mu_0} \Delta \mathbf{H}$$

VEZETÉK NÉLKÜLI ÁTVITEL



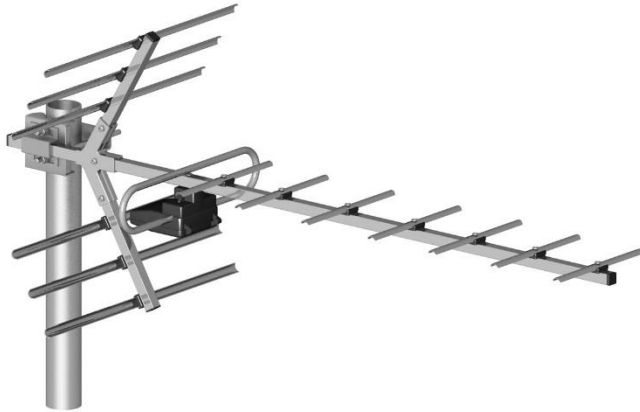
ELEKTROMÁGNESES SPEKTRUM



Az **antenna** elektromágneses hullámok egy tartományának, a rádióhullámoknak a sugárzására vagy vételére alkalmas elektrotechnikai eszköz.

Az antennák az elektromágneses erőterből energiát vesznek fel (vevőantennák) vagy a nagyfrekvenciás generátorok (adók) által keltett elektromágneses hullámok alakjában sugározzák. Ugyanazon antenna vételhez és adáshoz is használható, tulajdonságai és jellemzői mindkét esetben változatlanok.

ANTENNA TÍPUSOK



Vezeték nélküli átvitel

VEZETÉKNÉLKÜLI ÁTVITEL

Az elektromágneses jelek átvitelét és érzékelését antennák végzik.

A sugárzásnak két módja van:

- Irányított
- Mindenirányú (irányítatlan)

Irányított esetben az antenna fókuszált elektromágneses sugarat bocsát ki, a vevő antennát pontosan kell pozícionálni.

Mindenirányú sugárzás sok antennával vehető.

A nagyobb frekvenciájú jelek jobban fókuszálhatók.

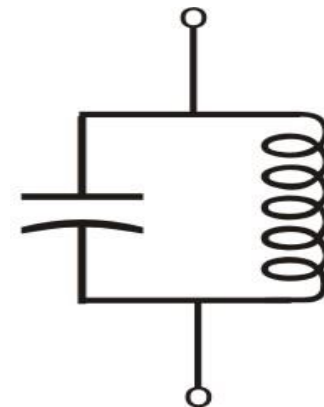
Három frekvencia-tartomány jöhet szóba vezetéknélküli átvitelre:

- 2 - 40 GHz (mikrohullámú átvitel) (irányított)
- 30 MHz - 1 GHz (rádiófrekvencia) (mindenirányú)
- $3 \cdot 10^{11}$ - $2 \cdot 10^{14}$ Hz (infravörös)

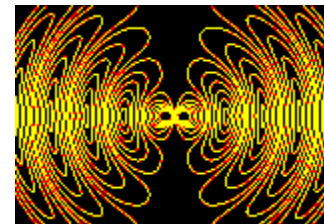
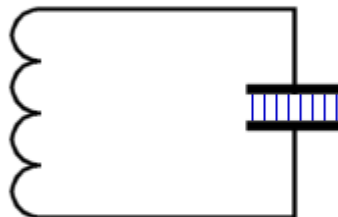
RÁDIÓZÁS ALAPJAI

Rezgőkör: a rezgőkörben elektromos és mágneses energia alakul át egymásba periodikusan

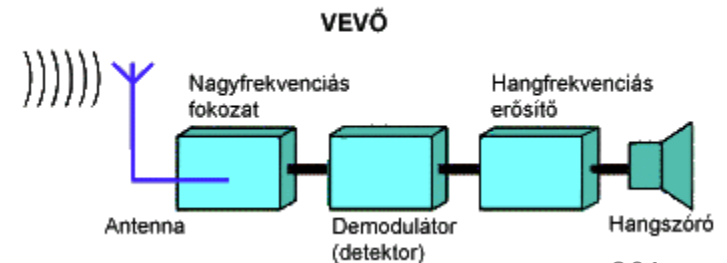
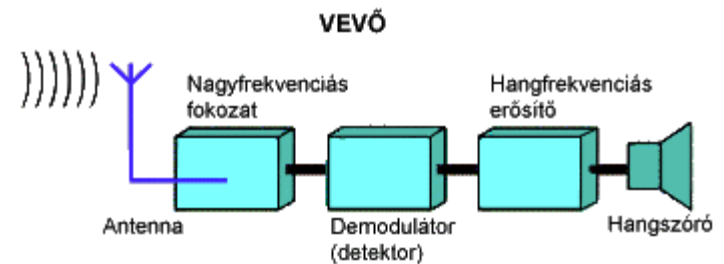
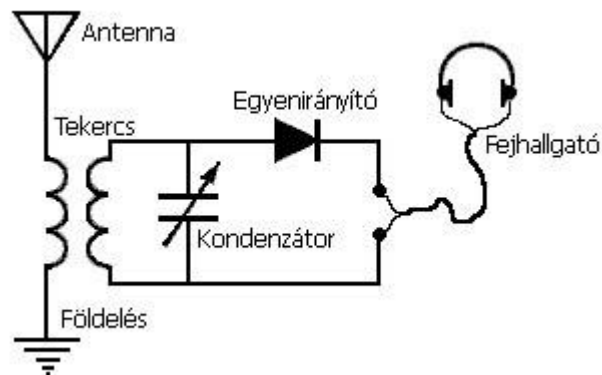
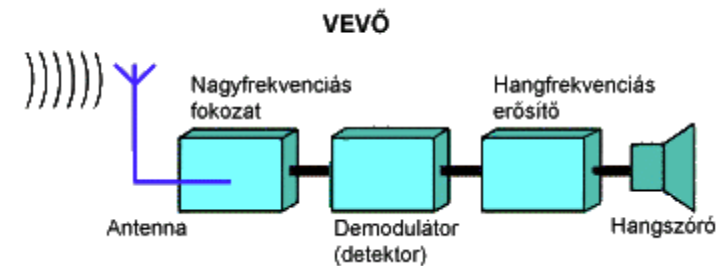
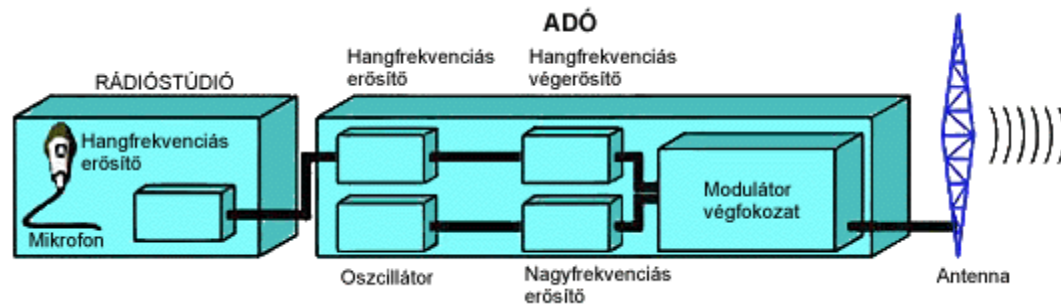
$$f = \frac{1}{2\pi\sqrt{LC}}$$



Antenna:



A RÁDIÓ MŰKÖDÉSE



Vezeték nélküli átvitel

Az az eljárás, melynek segítségével a kisugározni kívánt jelet a vivő jelre ráültetjük, azaz a kisugározni kívánt jellel változtatják az összeköttetést létesítő vivőjel jellemzőit: amplitúdóját, frekvenciáját, fázisát.

Miért kell?

1. Hatásfok
2. Ha minden adó ua. frekvencián sugározna az a káosz.

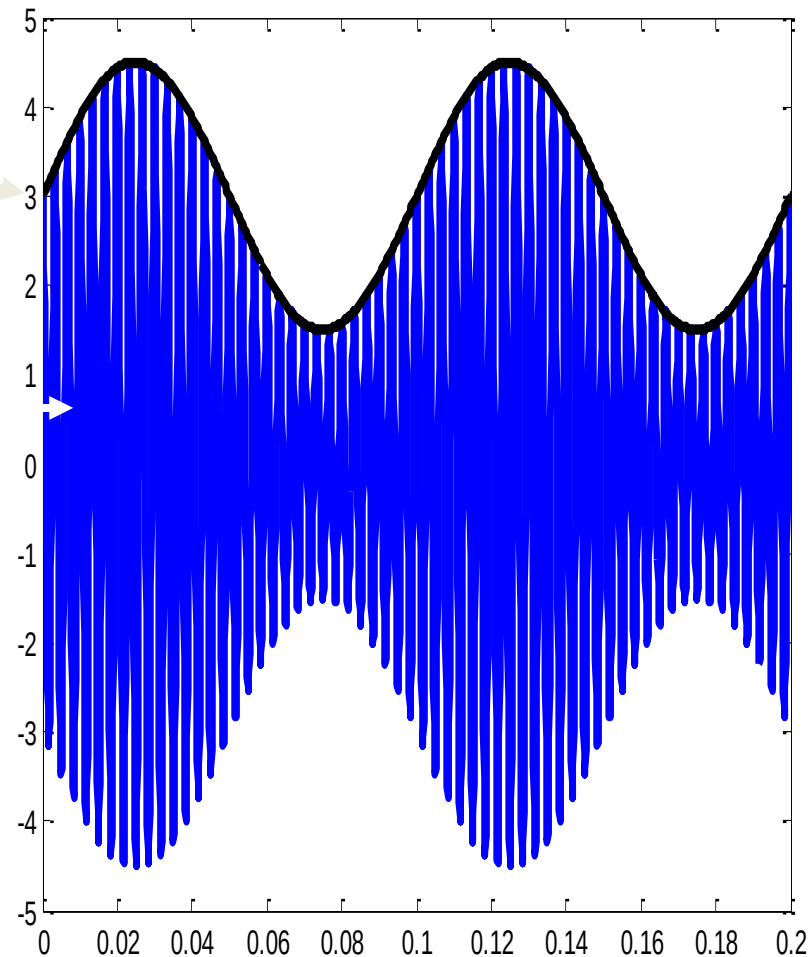
AMPLITÚDÓ MODULÁCIÓ

Burkoló jel:

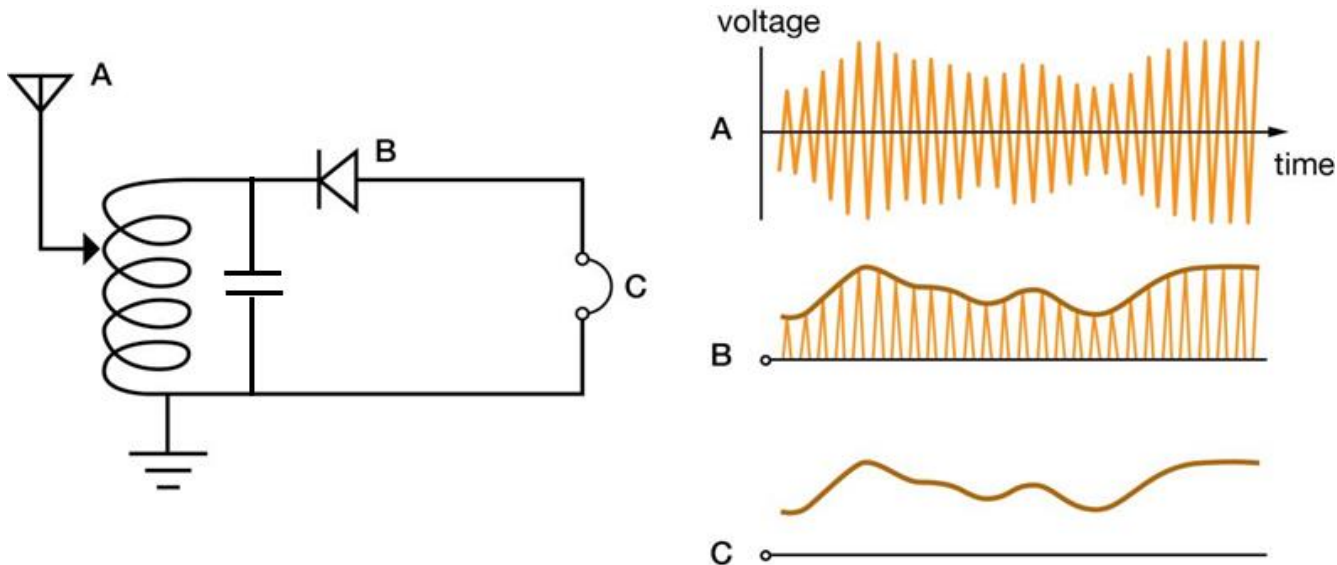
$$s(t) = A_c [1 + k \bullet m(t)]$$

AM jel:

$$s(t) = A_c [1 + k \bullet m(t)] \cos \omega_c t$$



AMPLITÚDÓ MODULÁCIÓ

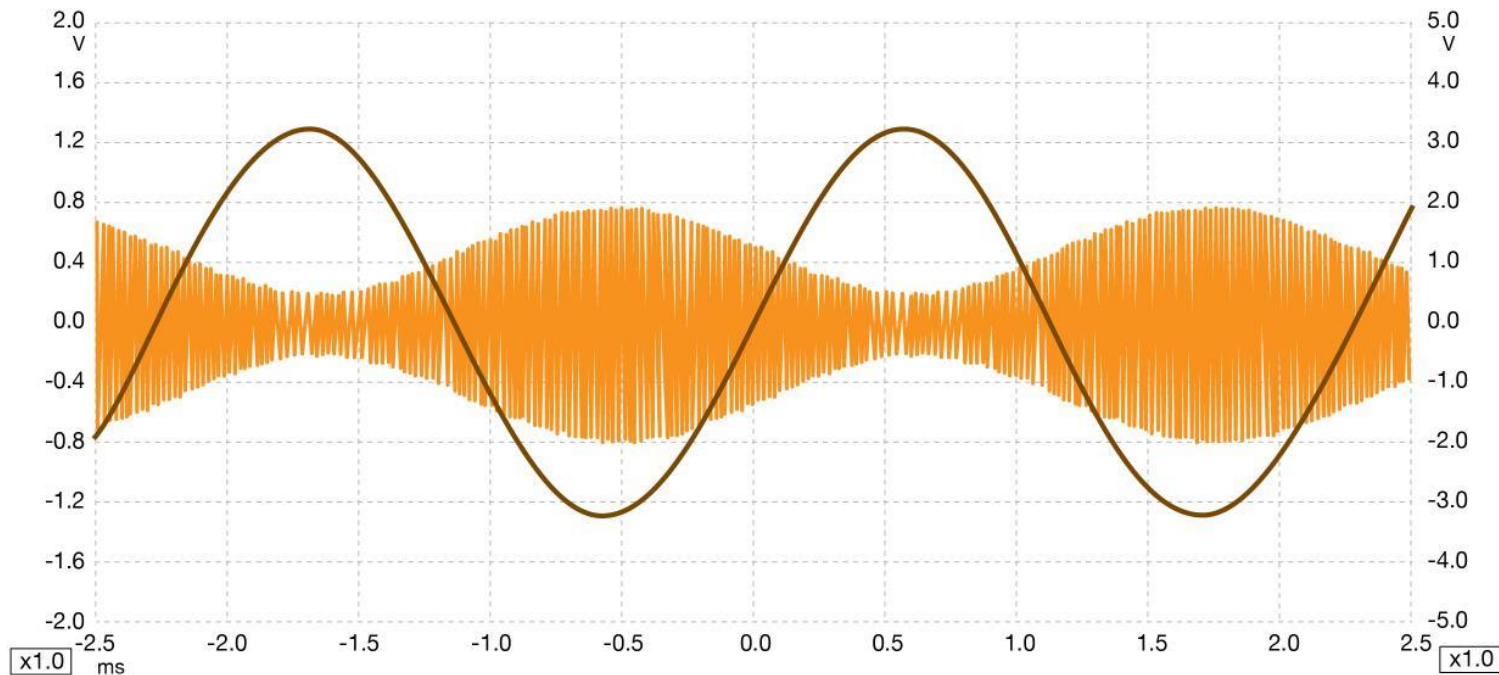


A: AM jel

B: Dióda utáni egyenirányított jel

C: A vivőhullámtól megszűrt jel

AMPLITÚDÓ MODULÁCIÓ



vivőfrekvencia: $X = A_c \sin 2\pi f_c t$

moduláló frekvencia: $Y = A_m \sin 2\pi f_m t$

$$Z = A_c \left(1 + \frac{A_m \sin 2\pi f_m t}{A_c}\right) \sin 2\pi f_c t$$

AMPLITÚDÓ MODULÁCIÓ

A trigonometria szabályai szerint az eredő

$$Z = A_c \left(1 + \frac{A_m \sin 2\pi f_m t}{A_c} \right) \sin 2\pi f_c t$$

jel szinusz és koszinuszos tagok összegeként is előállítható, az alábbiak szerint:

$$Z = A_c \sin 2\pi f_c t + \frac{A_m \cos 2\pi(f_c - f_m)t}{2} - \frac{A_m \cos 2\pi(f_c + f_m)t}{2}$$

AMPLITÚDÓMODULÁCIÓ

Az eredmény azt mutatja, hogy az AM jel három komponensből áll. Az első maga a vivő:

$$A_c \sin 2 \pi f_c t$$

A második komponens a vivő és a modulálófrekvencia különbségét tartamazza. Ezt alsó oldalsávnak (LSB) nevezzük:

$$\frac{A_m \cos 2 \pi (f_c - f_m) t}{2}$$

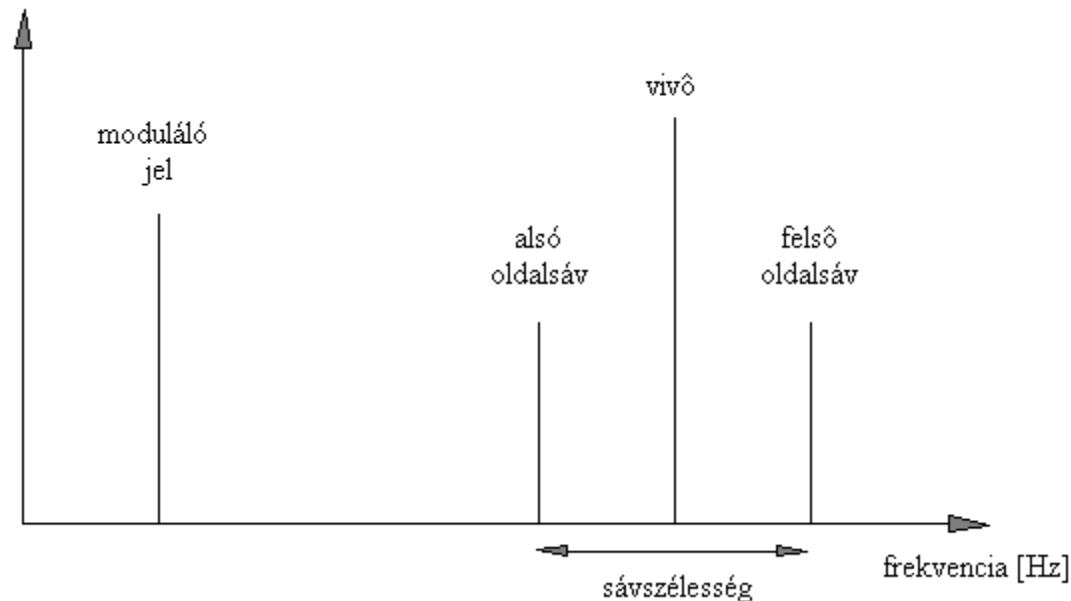
AMPLITÚDÓMODULÁCIÓ

A harmadik komponens a vivő és a moduláló frekvencia összegét tartalmazza, amit felső oldalsávnak (USB) hívnak:

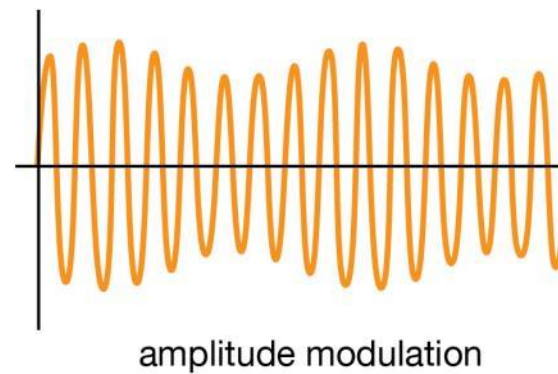
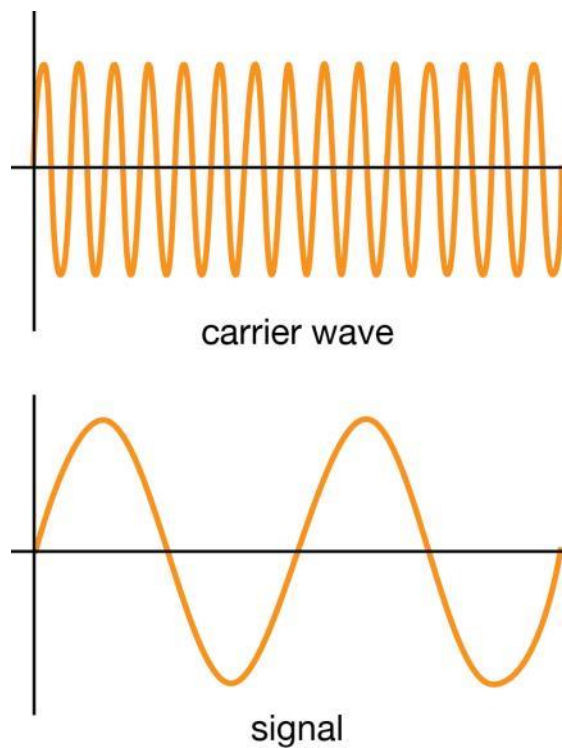
$$\frac{A_m \cos 2\pi(f_c + f_m)t}{2}$$

AMPLITÚDÓMODULÁCIÓ

A sávszélesség értéke $2f_m$. Ha tehát például a 10Hz és 15kHz közé eső zenei hangokat kívánunk sugározni, az adó kétszer akkora tartományt, azaz mintegy 30 kHz-et foglal el.



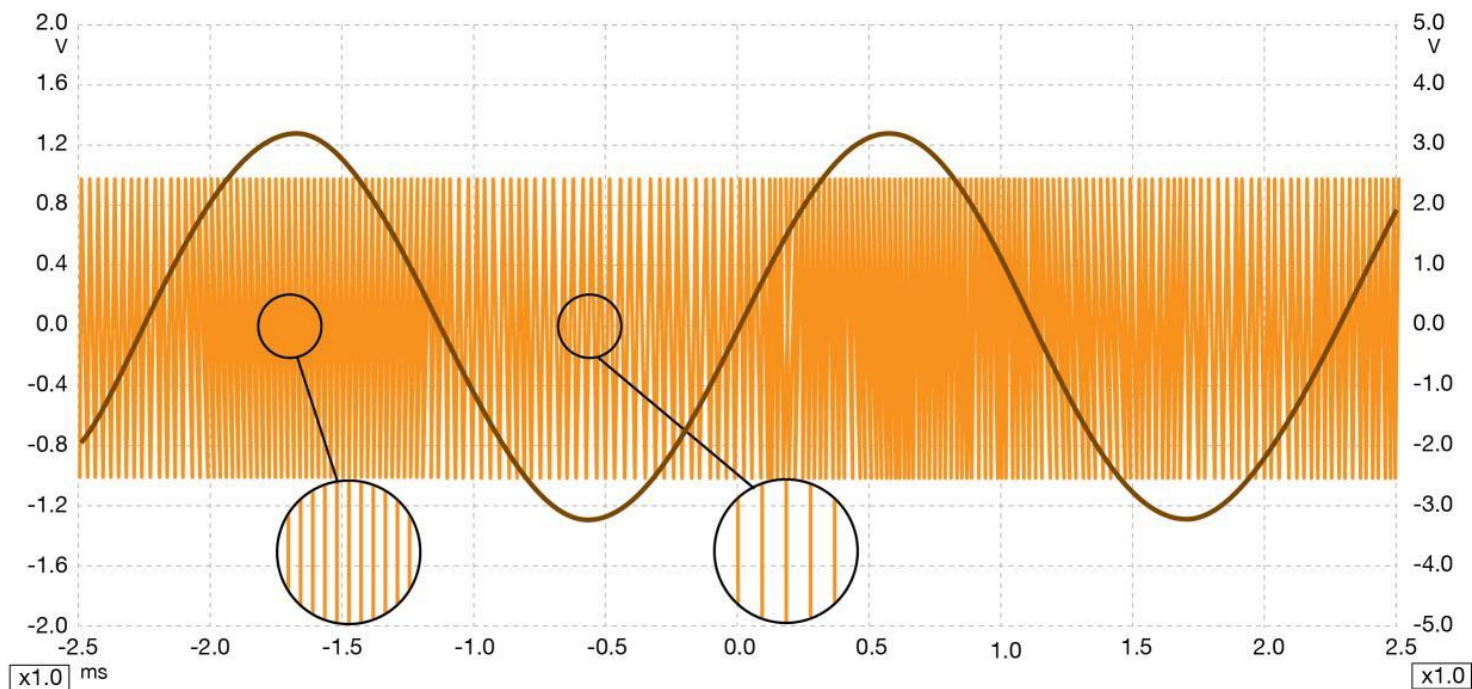
FREKVENCIAMODULÁCIÓ



FREKVENCIAMODULÁCIÓ

Az FM rendszerben (az amplitúdómodulációval ellentétben) az f_c *vivőfrekvencia* nem marad állandó, hanem a moduláló frekvencia Y *amplitúdójának* megfelelően változik. Ez nem okoz gondot, feltéve, hogy a vivőfrekvencia elegendően nagy, - a változás (frekvencialök) nem haladja meg a 10 milliomod részt. (Ez az oka annak, hogy az FM adók 100 Mhz környékén, vagy a fölött működnek.)

FREKVENCIAMODULÁCIÓ



vivőfrekvencia: $X = A_c \sin 2\pi f_c t$

moduláló frekvencia: $Y = A_m \sin 2\pi f_m t$

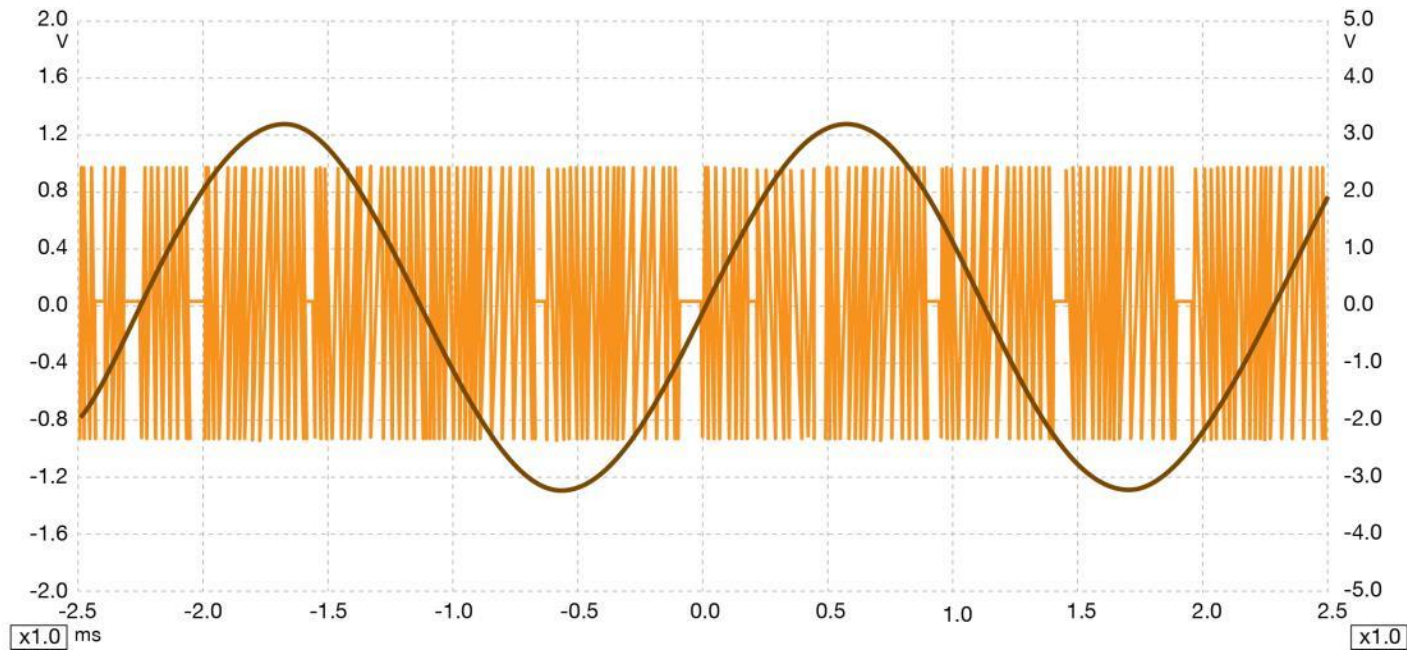
$$Z = f_c + kA_m \sin 2\pi f_m t$$

Az eredő

$$Z = f_c + kA_m \sin 2\pi f_m t$$

jelben a k arányossági tényező a moduláló frekvencia pillanatnyi feszültségértékének frekvenciává való alakítását jelenti. Ez az egyenlet nem olyan magától értetődő, mint az az amplitúdómodulációnál volt, és nem is bontható olyan könnyen frekvenciakomponensekre.

IMPULZUSKÓD MODULÁCIÓ



FELADATOK

- Ismertesse a rezgőkör működési elvét!
- Hogyan osztályozzuk a rádióhullámokat?
- Ismertesse az amplitúdómoduláció elvét!
- Mit értünk egy- és két-oldalsávós modulált jelen?
- Milyen korlátai vannak az amplitúdómodulációnak a rádiózásban?
- Ismertesse a frekvenciamoduláció előnyeit!
- Hogyan alakul a frekvenciamodulált hullám egyenlete?

AUTOMATIKA (ÖNMŰKÖDŐ RENDSZER)

Automata rendszerekben mindig van:

- **Információ szerzés**
- **Döntés** (a megszerzett információ alapján)
- **Beavatkozás** (módosítás, korrekció)

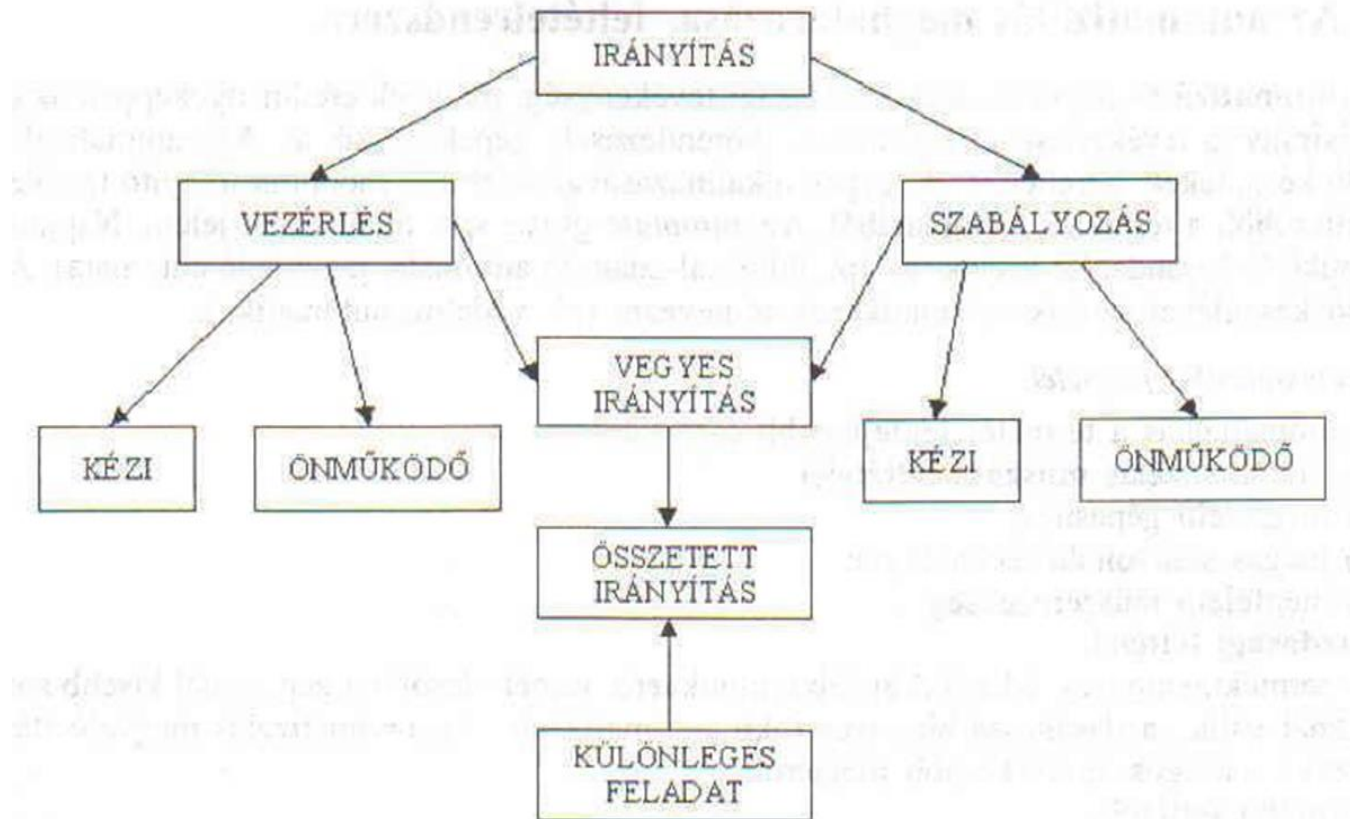
HATÁSOK CSOPORTOSÍTÁSA

Az irányított rendszert érő hatásokat 3 csoportra lehet bontani:

- **Megfigyelt hatások változásaira**
- **Zaj, zavar**
- **Eszközölt beavatkozások**

IRÁNYÍTÁSTECHNIKA FELOSZTÁSA

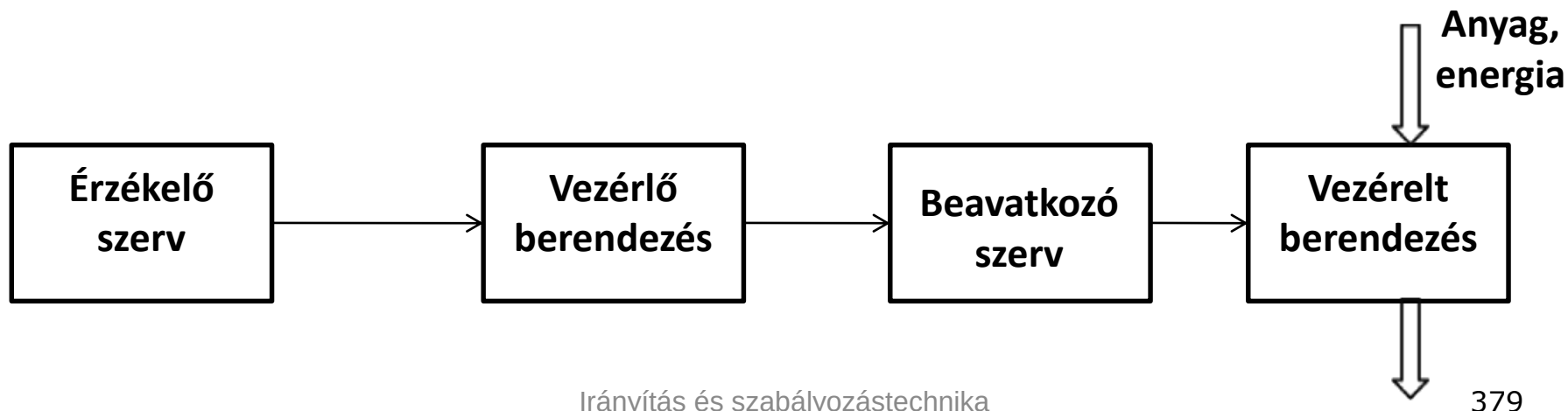
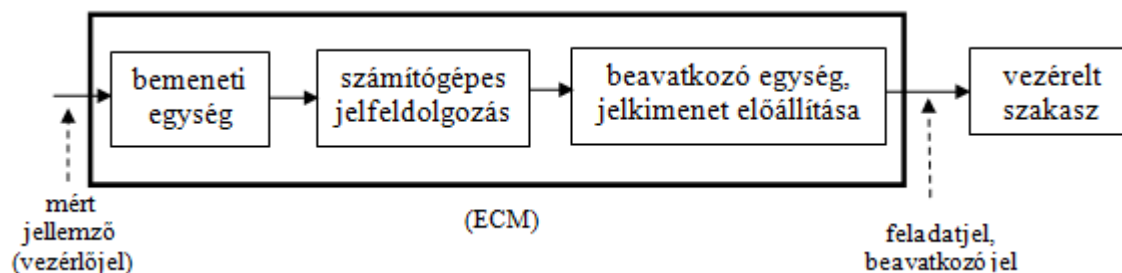
Az automata rendszerek a döntési mechanizmusukban különböznek egymástól.



Nyílt hatásláncú folyamat.

Fogalma: A vezérlésnél az irányított berendezésből nem érkezik visszajelzés. A vezérlés hatáslánca nyílt. A beavatkozás után nem értesülünk a következményekről.

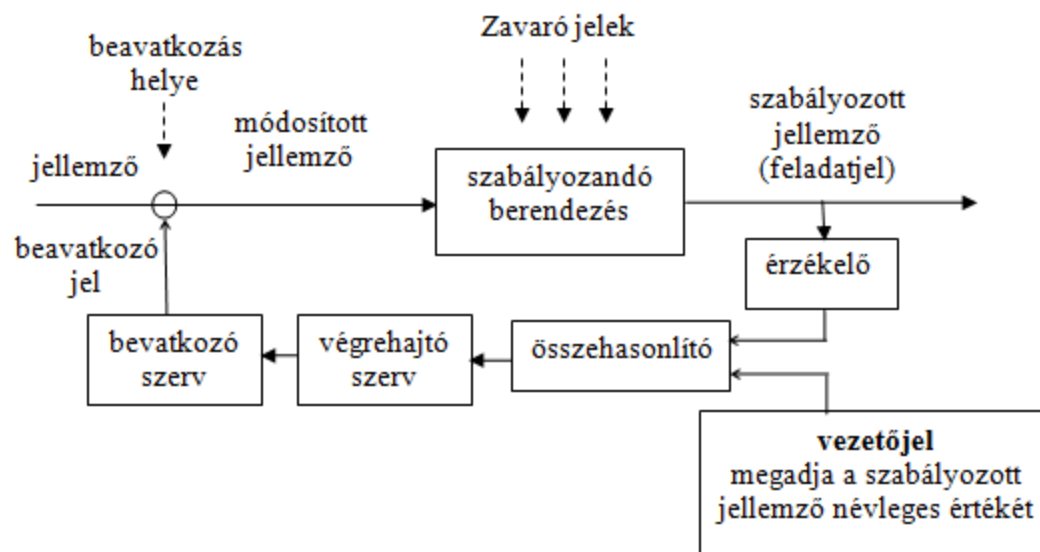
Feladata: Az indítás után egy művelet sor végrehajtása a eredménytől függetlenül.



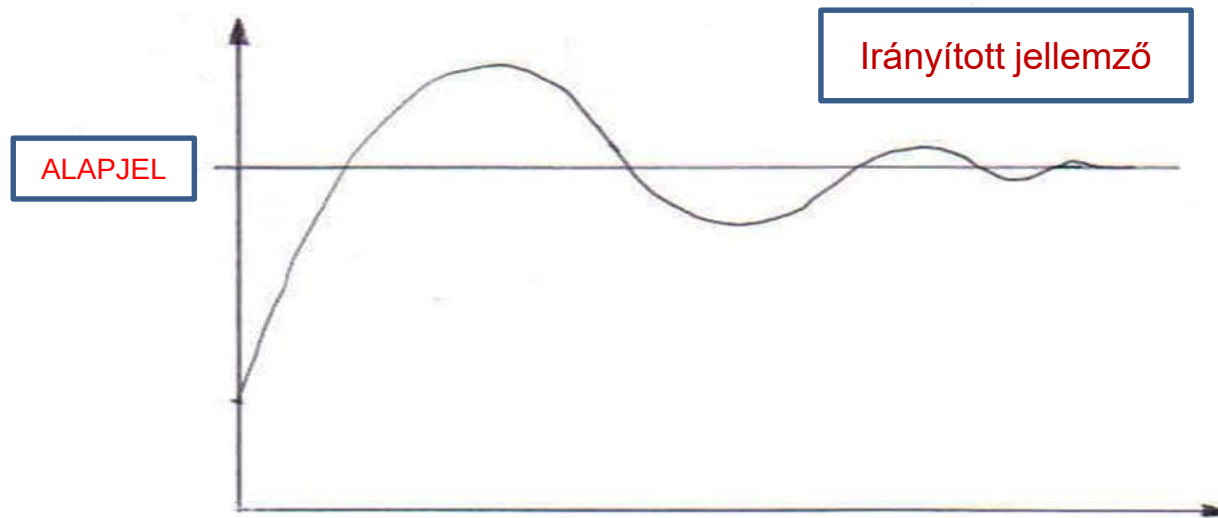
SZABÁLYOZÁS

Zárt hatásláncú folyamat.

A megszerzett információ alapján **visszacsatolással** avatkozik be a döntési algoritmusba.



SZABÁLYOZÁS



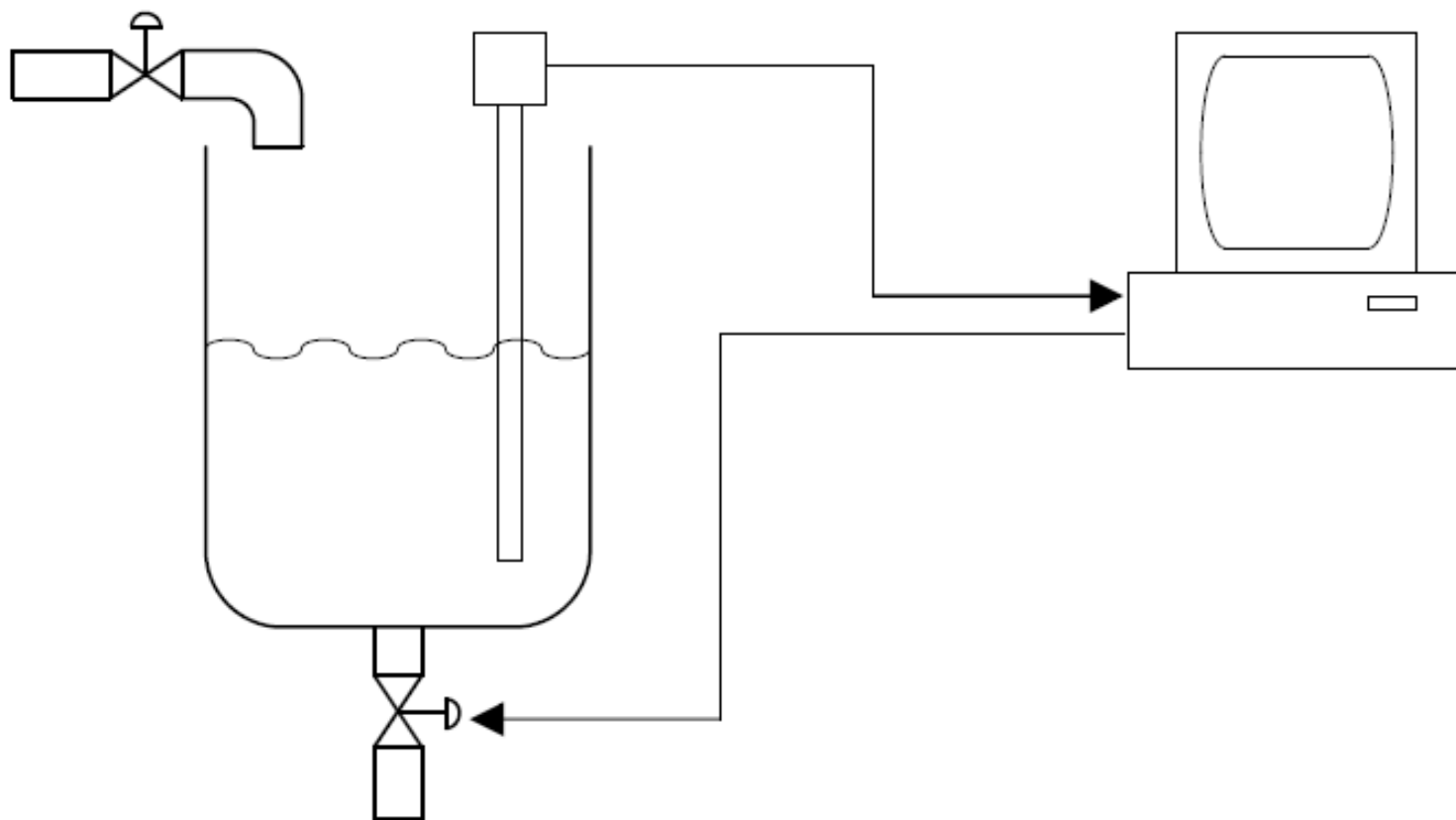
Fogalma: A szabályozásnál az irányított berendezésből érkezik visszajelzés, ezt összehasonlítom az alapjel értékével. A szabályozás hatáslánca zárt. A beavatkozás után folyamatosan értesülünk a következményekről.

Feladata: Az indítás után egy meghatározott érték (alapjel) elérése és megtartása a cél.

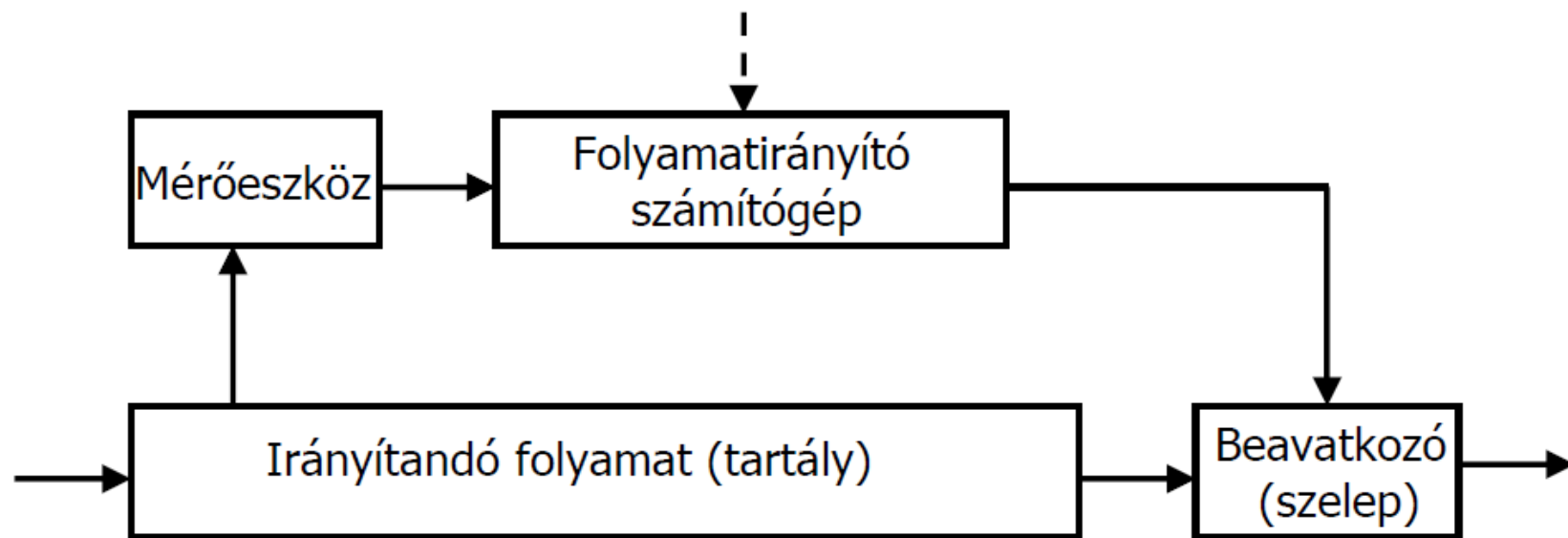
IRÁNYÍTÁSI RENDSZER RÉSZEI, LEÍRÁSA

- irányítási rendszer: - irányított rendszer
- irányító rendszer
- hatáslánc: irányítási rendszernek az irányítás hatását közvetítő sorozata
- irányítási rendszer leírása:
 - szerkezeti vázlat: irányítási szempontból lényeges részek kiemelése szimbólumokkal
 - működési vázlat: szerkezeti részek megjelenítése geometriai elemekkel
 - hatásvázlat: elvi ábrázolás, a jelet módosító elemek jelennek meg geometria jelképekkel ábrázolva

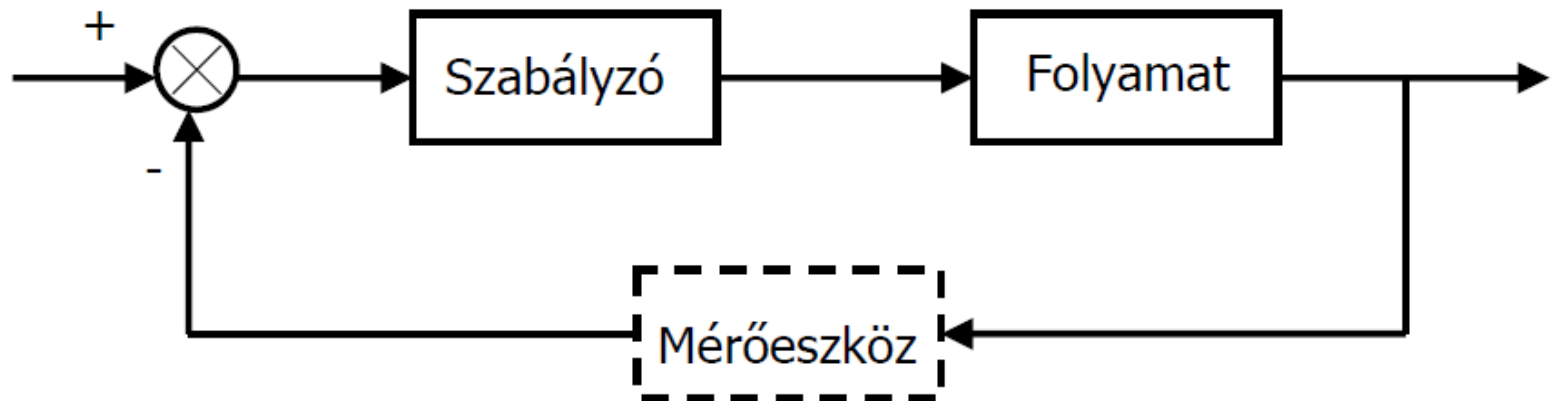
SZERKEZETI VÁZLAT



MŰKÖDÉSI VÁZLAT



HATÁSVÁZLAT



SZERKEZETI RÉSZEK

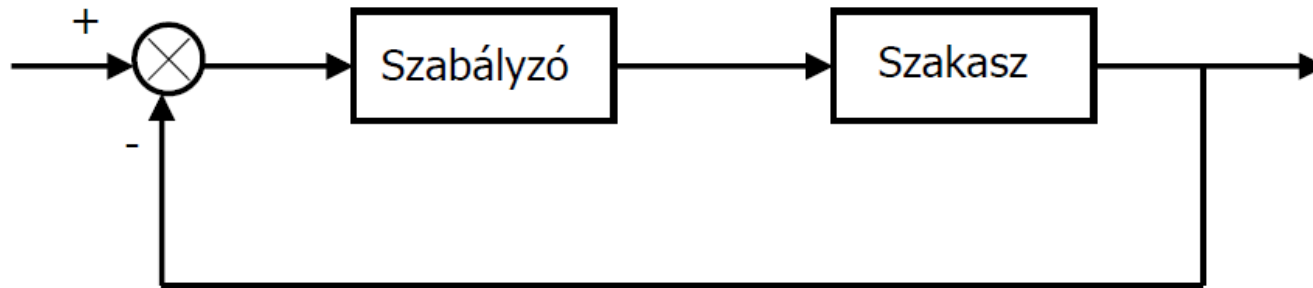
- **készülék:** szerkezetileg körülhatárolt, önálló technológiai egység, melynek irányítási feladata van
- **szerv:** részfeladatok önálló ellátása
- **elem:** irányítástechnikai szempontból tovább nem bontható egység
- **jelvivő vezeték:** jelek továbbítására alkalmas összeköttetés

HATÁSVÁZLAT RÉSZEI

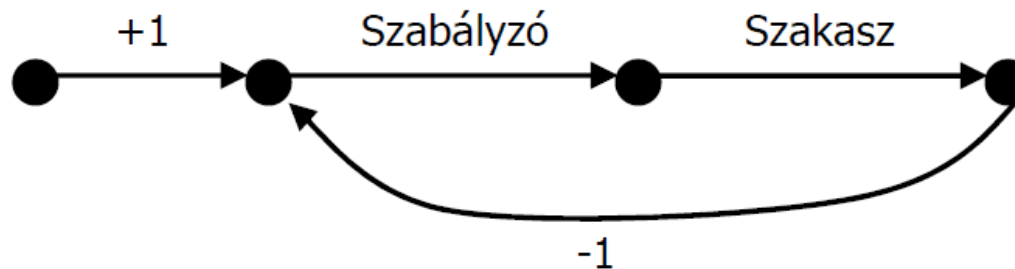
- tag: az irányítási rendszer jelformáló tulajdonságú eleme
 - irányított szakasz: az irányított objektumot leíró tag
 - szabályzó: az irányítás feladatát ellátó tag
- hatásirány: a jelnek a tagokon való áthaladási iránya, egyirányúság, visszahatás-mentesség
 - bemenő jel: tagot működésre készítő, tőle független jel
 - kimenő jel: a tag működése folyamán kialakuló jel

HATÁSVÁZLAT LEÍRÁSA

- tömbvázlat: gráf, csúcsokon a tagok, élek a jelek



- jel-folyam gráf: csúcsokon a jelek, élek a tagok



ÁLLAPOTVÁLTOZÓK

- Az $\mathbf{x}_1(\mathbf{t}), \mathbf{x}_2(\mathbf{t}), \dots, \mathbf{x}_n(\mathbf{t})$ változókat a rendszer állapotváltozóinak nevezzük, ha ismerjük a $\mathbf{t}_0$ időpontban a változók értékeit és a bemenő jelet, akkor ez elegendő ismeret a rendszer bármely más $\mathbf{t} > \mathbf{t}_0$ időpontbeli állapotának meghatározására.
- Egy rendszert lineárisnak nevezünk, ha a bemeneti jelek és a kimeneti jelek kapcsolata lineáris

$$\sum_{i=1}^n x_i = \sum_{i=1}^n c_i y_i$$

(érvényes a szuperpozíció elve)

A SZUPERPOZÍCIÓ TÖRVÉNYE



Külön-külön tetszőleges jelekkel gerjesztve a jelátvivő tagot és mérve a válaszfüggvényeket, majd összegezve a gerjesztő jeleket megismételve a mérést, ha az eredmény az, hogy az első két válaszfüggvény összege elegendő pontossággal azonos az az összegzett jelre adott válasszal, akkor a jelátvivő tag lineárisnak tekinthető

LAPLACE TRANSZFORMÁCIÓ

A Laplace transzformáció az analóg rendszerekre (folytonos függvényekre) vonatkozó Fourier transzformáció általánosítása, melynek matematikai alakja:

$$F(s) = \int_0^{\infty} e^{-st} f(t) dt$$

$$s = \sigma + j\omega$$

$$f(t) = \frac{1}{2\pi j} \int_{\sigma - j\infty}^{\sigma + j\infty} e^{st} F(s) ds$$

ÁTVITELI FÜGGVÉNY

- Csak lineáris rendszerekre definiáljuk
- A kimenő jel Laplace transzformáltjának és bemenő jel Laplace transzformáltjának a hányadosa

$$Z_i(s) = \frac{y_i(s)}{x_i(s)}$$

DINAMIKUS VIZSGÁLAT

- Egy bemeneti változó függvényében vizsgáljuk, a többi üzemi értéken van.
- Az üzemi értéktől való eltérés zavarásként lesz figyelembe véve.
- A dinamikus vizsgálat lineáris jelátviteli tagok esetén jól kidolgozott.
- Mérnöki szempontból egy jelátviteli tag akkor lineáris, ha kellő pontossággal érvényes rá a szuperpozíció.

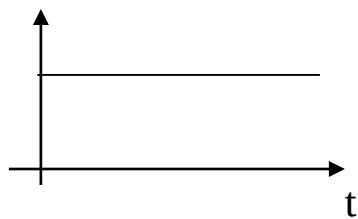
SZABVÁNYOS VIZSGÁLÓ JELEK



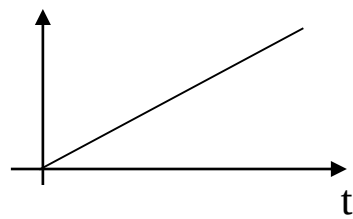
Ha érvényes a szuperpozíció, akkor alkalmazhatók a szabványos vizsgáló jelek. Ezekből a jelekből tetszőleges jel összerakható!



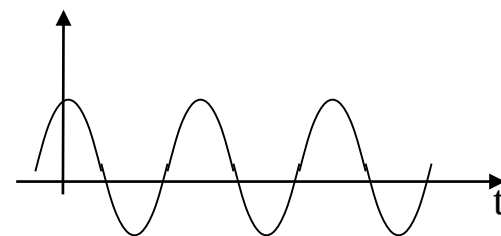
Impulzus
(Dirac delta)



Egység ugrás



Sebesség ugrás

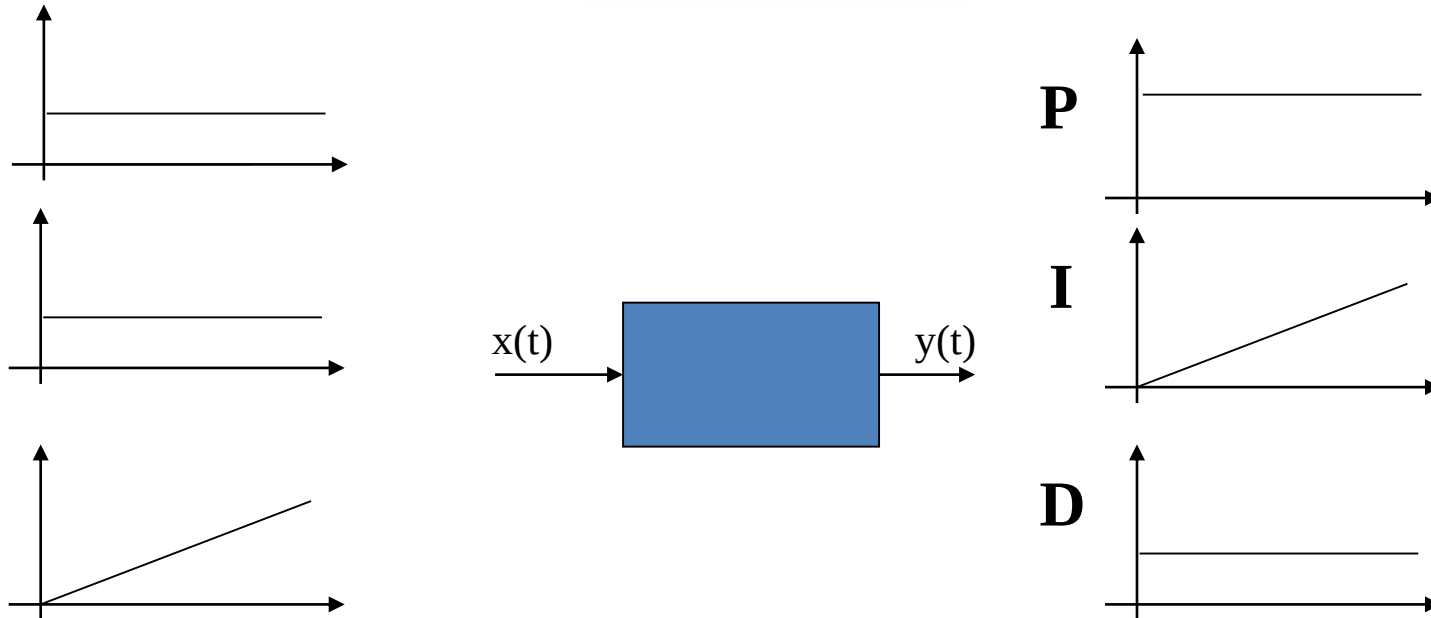


Szinuszos

Az impulzusra adott válasz a súlyfüggvény, az egység ugrásra adott válasz az átmeneti függvény.

A szinuszos jelre adott válasz azonos körfrekvenciájú szinusz.

LINEÁRIS JELÁTVITELI TAGOK JELLEMZÉSE

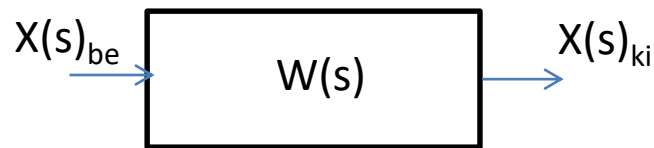


- A jelátviteli tag jellegre lehet arányos (P), integráló (I) és differenciáló (D).
- A tehetetlenségét tekintve lehet egy (T1) vagy két (T2) tárolós (időállandós).
- Lehet időben késleltetés nélküli vagy holtidős (H), azaz késleltetett.

A tehetetlenséget és az időbeni késleltetést az egységnyi arányos hatás mellé rendelve szokás definiálni. Matematikai modellekben P, I, D, PT1, PT2, PH a hat alaptag.

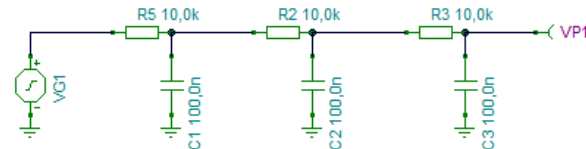
JELÁTVITELI TAGOK LEÍRÁSA

- Frekvencia tartománybeli vizsgálat, leírás átviteli függvényvel

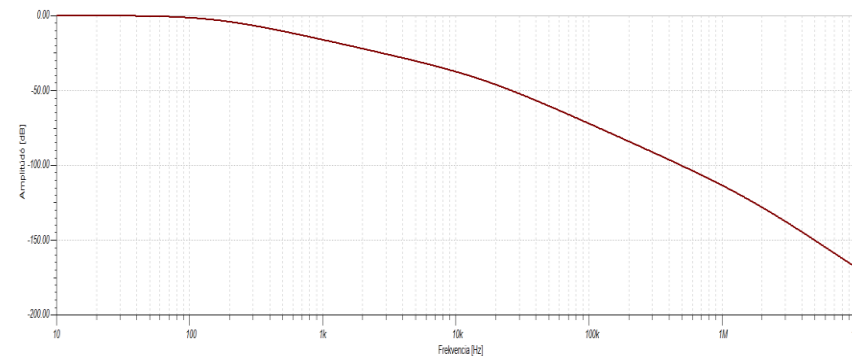


$$W(s) = \frac{X(s)_{ki}}{X(s)_{be}}$$

- Szimulációval:



- **Méréssel:**
Oszilloszkóp és
jelgenerátor segítségével



- Bemenet-kimenet alapú matematikai modell

$$a_0 \frac{d^n y(t)}{dt^n} + a_1 \frac{d^{n-1} y(t)}{dt^{n-1}} + \dots + a_n y(t) = b_0 \frac{d^m u(t)}{dt^m} + b_1 \frac{d^{m-1} u(t)}{dt^{m-1}} + \dots + b_m u(t)$$

- Állapotteres leírás

$$\frac{dx(t)}{dt} = Ax(t) + Bu(t)$$

$$y(t) = Cx(t) + Du(t)$$

Ha: ***j*** számú *u* bemenő jel

k számú *y* kimenő jel

x állapotváltozók száma ***n***

Akkor: ***A*** állapot mátrix (*n*×*n*)

B bemeneti mátrix (*n*×*j*)

C kimeneti mátrix (*k*×*n*)

D együtttható mátrix (*k*×*j*)

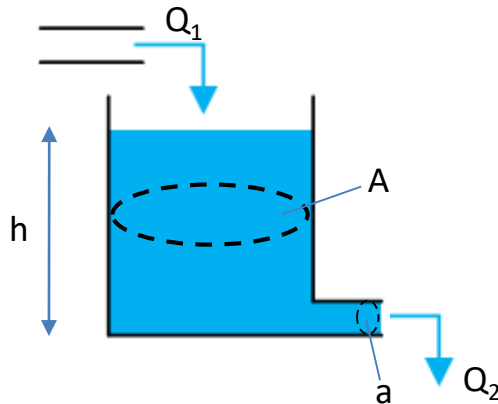
NEM LINEÁRIS RENDSZEREK

- Egy szabályzás nem lineáris, ha a szabályzó kör legalább egy tagjára nem érvényes a szuperpozíció elve.
- Állapotteres leírás:

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}, \mathbf{u}, t) \\ \mathbf{y} &= \mathbf{h}(\mathbf{x}, \mathbf{u}, t)\end{aligned}\quad \dot{\mathbf{x}} = \frac{d\mathbf{x}}{dt}$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad \mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_p \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \quad \mathbf{f}(\mathbf{x}, \mathbf{u}, t) = \begin{bmatrix} f_1(\mathbf{x}, \mathbf{u}, t) \\ f_2(\mathbf{x}, \mathbf{u}, t) \\ \vdots \\ f_n(\mathbf{x}, \mathbf{u}, t) \end{bmatrix} \quad \mathbf{h}(\mathbf{x}, \mathbf{u}, t) = \begin{bmatrix} g_1(\mathbf{x}, \mathbf{u}, t) \\ g_2(\mathbf{x}, \mathbf{u}, t) \\ \vdots \\ g_m(\mathbf{x}, \mathbf{u}, t) \end{bmatrix}$$

VÍZTARTÁLY – NEM LINEÁRIS RENDSZER



Q_1 : a beömlő vízmennyiség [m^3/s]
 Q_2 : a kiömlő vízmennyiség [m^3/s]
 h : a vízoszlop magassága [m]
 A : a tartály keresztmetszete [m^2]
 a : a kiömlő nyílás keresztmetszete [m^2]
 v : a kiömlő víz sebessége
 μ : a folyadékra és a kiömlő nyílásra jellemző állandó

$$\frac{dh}{dt} = \frac{1}{A} Q_1 - \frac{\mu a \sqrt{2g}}{A} \sqrt{h}$$

$$Q_2 = \mu a v$$

$$mgh = \frac{1}{2} m v^2 \rightarrow v = \sqrt{2gh}$$

$$Q_2 = \mu a \sqrt{2g} \sqrt{h}$$

$$Q_1 - Q_2 = A \frac{dh}{dt}$$

Taylor-sorba fejtés

$$\frac{dx_1(t)}{dt} = f_1(x_1(t), x_2(t), u_1(t))$$

$$\frac{dx_2(t)}{dt} = f_2(x_1(t), x_2(t), u_2(t))$$

$$f(x) \approx f(x_0) + \left. \frac{\partial f}{\partial x} \right|_{x=x_0} \Delta x$$

$$\sqrt{h} \approx \sqrt{h_0} + \frac{1}{2\sqrt{h_0}} \Delta h$$

FELADATOK

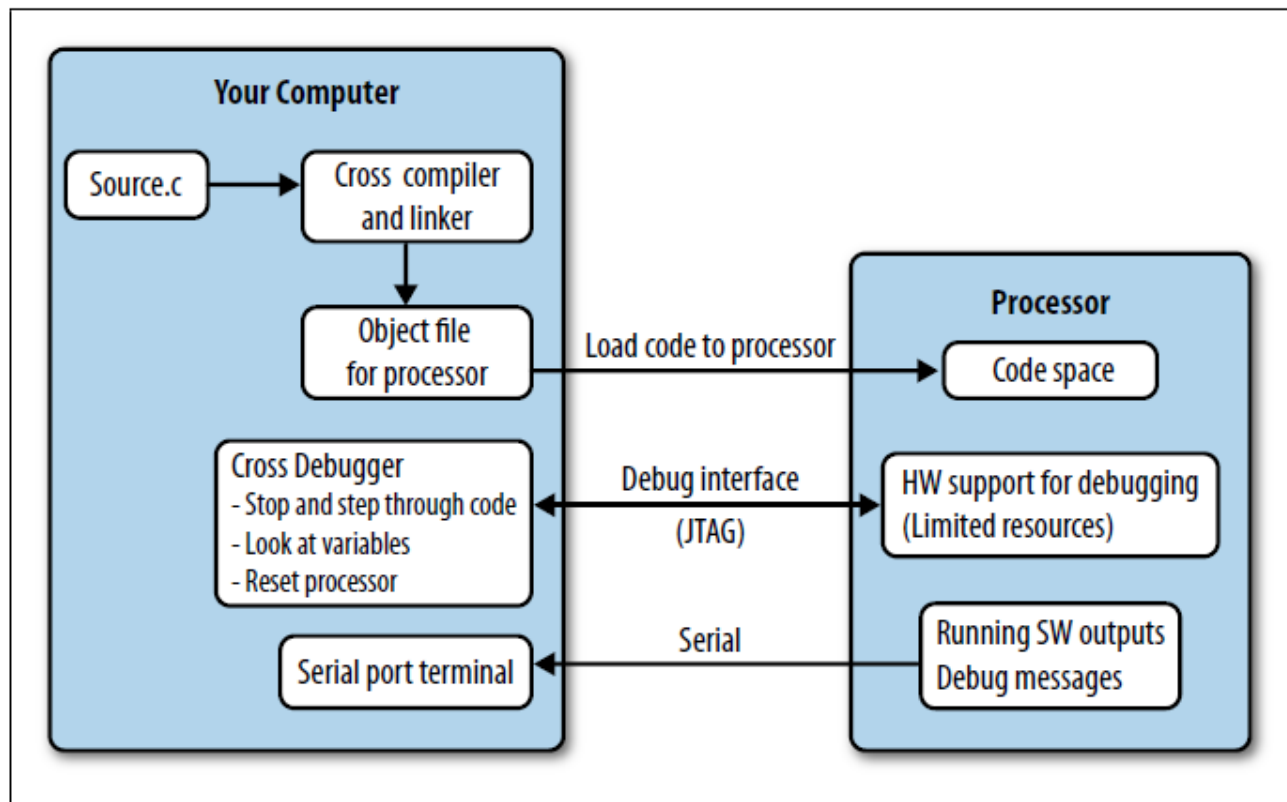
- Mi a lényeges különbség a vezérlés és a szabályozás között?
- Ismertesse a szabványos vizsgáló jeleket!
- Mi az átviteli függvény?
- Mikor nevezünk egy szabályozási rendszert lineárisnak?
- Milyen metódusokkal írhatunk le egy irányítási rendszert?

BEÁGYAZOTT RENDSZEREK

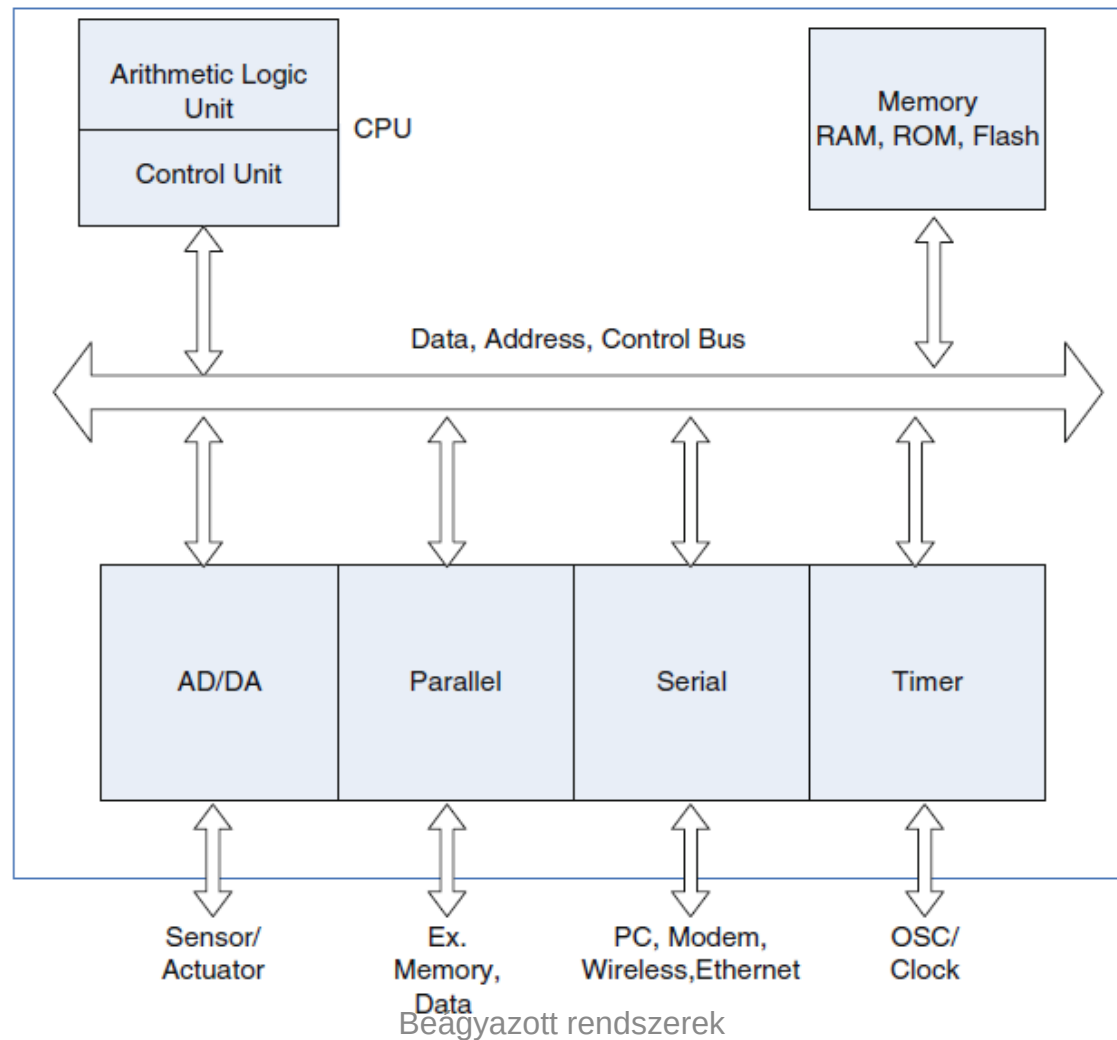
Beágyazott rendszernek nevezzük azokat a számítógépes rendszereket, melyeket egy adott feladat (még ha az bonyolult is) végzésére építettek. Néhány példa beágyazott rendszerekre:

- automata mosógépek vezérlő rendszere,
- házimozsi erősítők,
- LCD és plazma TV-k,
- digitális fényképezőgépek, kamerák,
- gépkocsi elektronika,
- repülőgépek fedélzeti rendszerei,
- bank automaták (ATM).

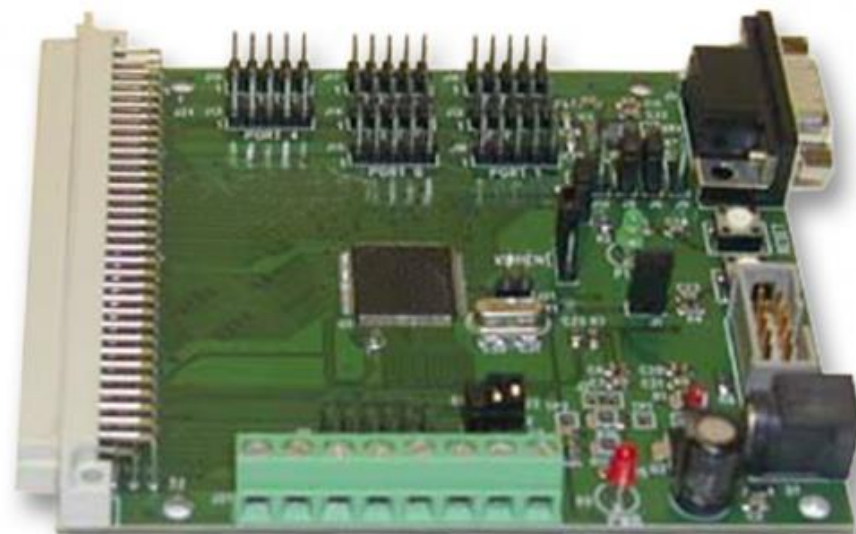
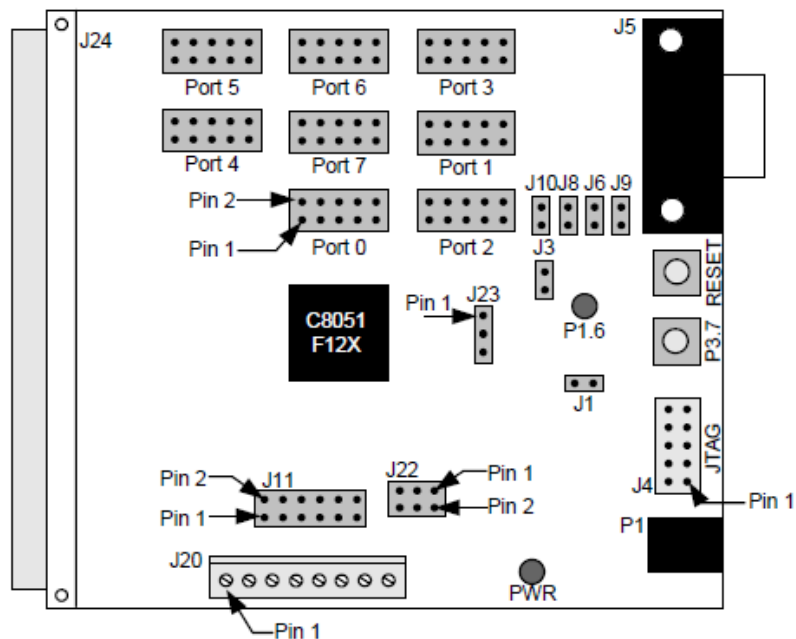
SZOFTVER FEJLESZTÉS BEÁGYAZOTT KÖRNYEZETBEN



MIKROKONTROLLEREK



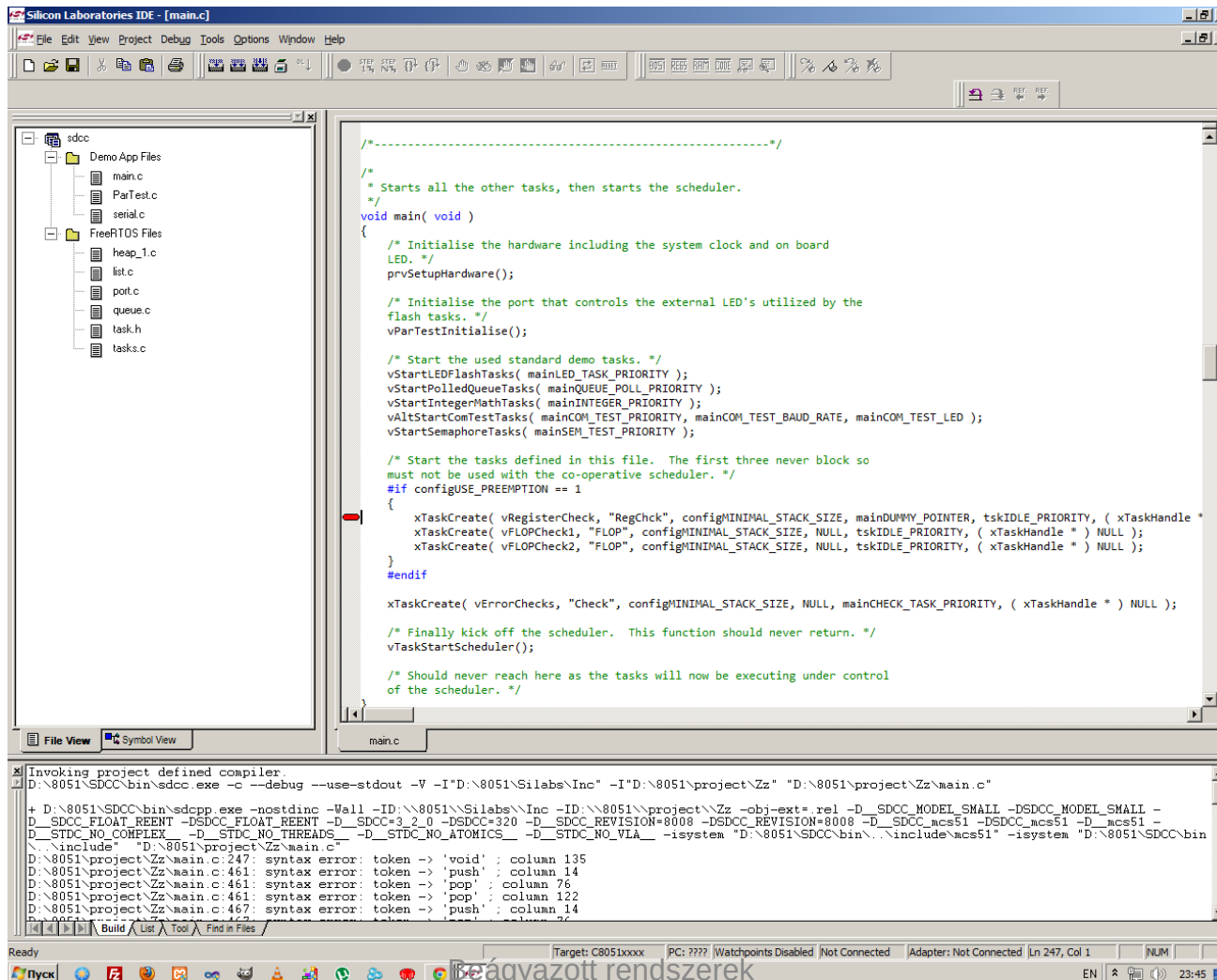
C8051F120 FEJLESZTŐI KÁRTYA



FEJLESZTŐI KÉSZLET



FEJLESZTŐI KÉSZLET - SZOFTVER

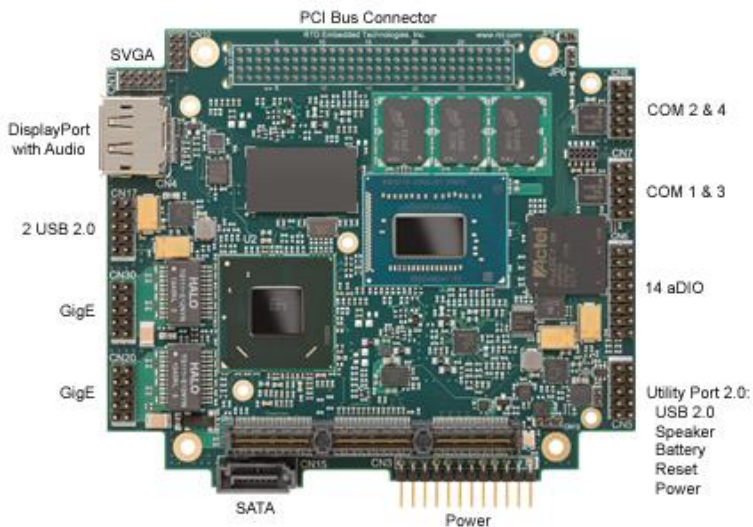


PROGRAMOZÁSI MODEL

Mikrokontrolleres környezetben a *main* függvény soha nem lép ki, mert a program alatt nincs semmilyen operációs rendszer, mely ezt kezelné. Ezért a *main* függvényben el kell helyezni egy hurkot, mely a beágyazott szoftver folyamatos futásához szükséges rutinokat állandóan futtatja. Ilyen hurkot legegyszerűbben végtelen ciklussal tudunk implementálni. Ennek alapján a *main* általános szerkezete a következő:

```
void main(void)
{
    Init();
    while(1)
    {
        func1();
        func2();
    }
}
```

EGYLAPKÁS SZÁMÍTÓGÉPEK



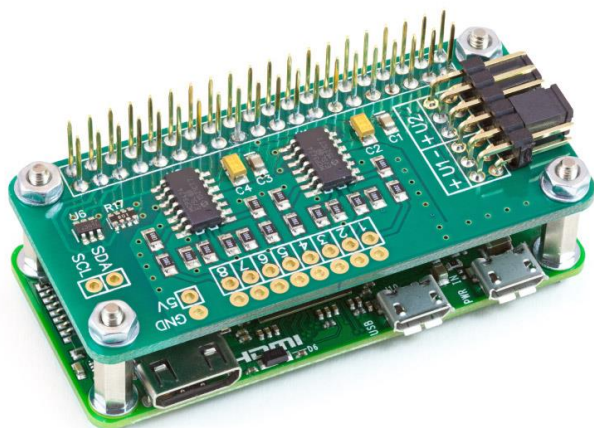
Jellemzők:

- minden egy lapkán van
- operációs rendszertámogatás (Linux, Windows)
- segítségével kompakt műszer építhető.
- drága

RASPBERRY PI 3 B+



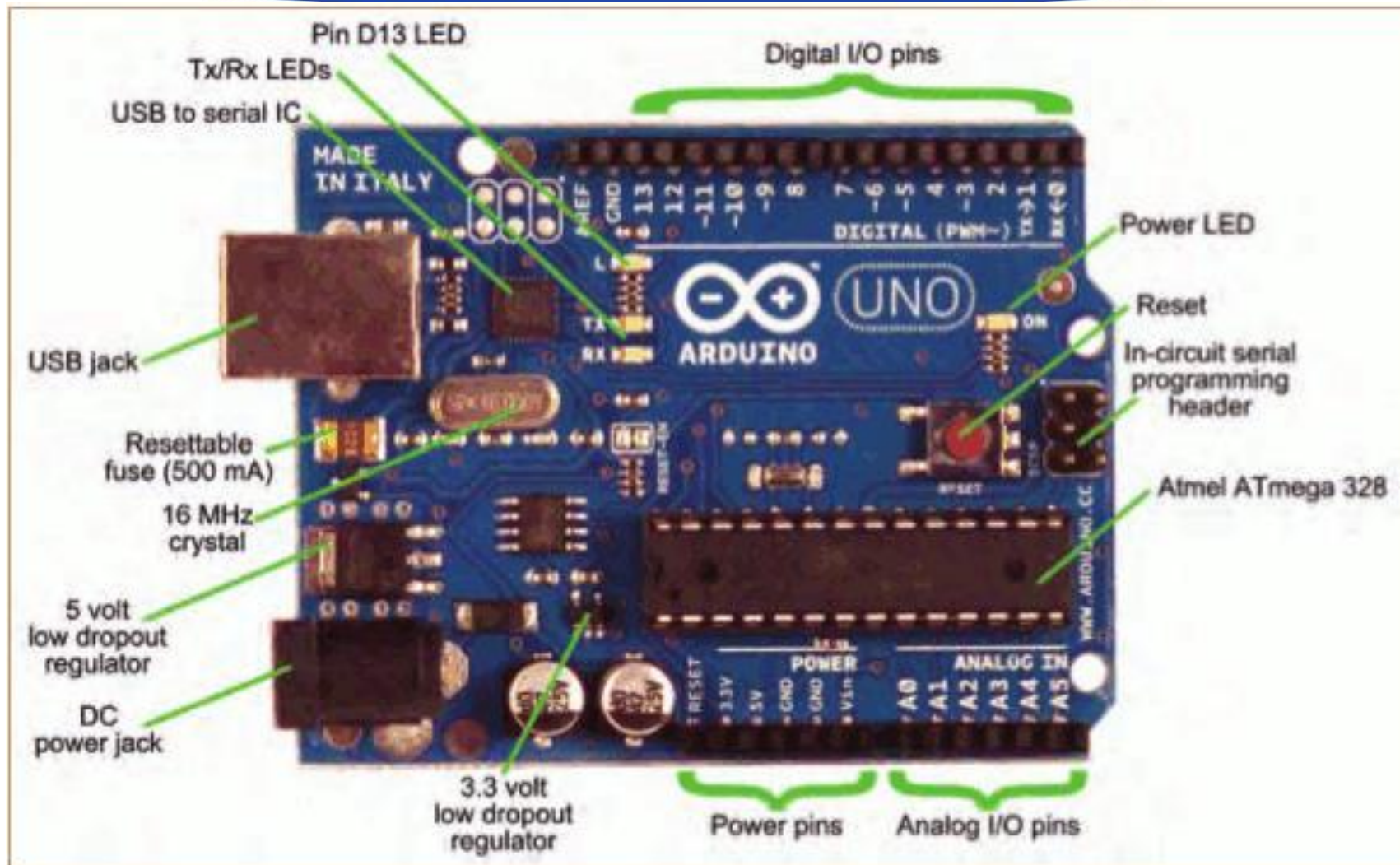
- ARM6 processzor
- 1GB RAM
- DUAL Video Core GPU
- 4 USB 2.0
- HDMI
- GPIO (40 pin)
- SD



Operációs rendszer:

- Raspbian
- Fedora
- Ubuntu Mate
- OSMC (Open Source Media Center)
- Windows 10 on Raspberry Pi2

ARDUINO



ARDUINO PROGRAMOZÁSI ALAPOK

```
int d1 = 1;
int d2 = 2;
int tipkalo = 3;
void setup() {
    // put your setup code here, to run once:
    pinMode(d1, OUTPUT);
    pinMode(d2, OUTPUT);
    pinMode(tipkalo, INPUT);
}

void loop() {
    // put your main code here, to run repeatedly:
    int a = digitalRead(tipkalo);
    if (a == HIGH) {
        stisnuto();
    }
    else {
        otpusteno();
    }
}
```

```
void stisnuto() {
    digitalWrite(d1, HIGH);
    digitalWrite(d2, LOW);
    delay(1000);
    digitalWrite(d2, HIGH);
    digitalWrite(d1, LOW);
    delay(1000);
}

void otpusteno() {
    digitalWrite(d1, HIGH);
    digitalWrite(d2, HIGH);
    delay(50);
    digitalWrite(d2, LOW);
    digitalWrite(d1, LOW);
    delay(50);
}
```


ARDUINO ADC PÉLDA

- ```
// These constants won't change. They're used to give names to the pins used:
const int analogInPin = A0; // Analog input pin that the potentiometer is attached to
const int analogOutPin = 9; // Analog output pin that the LED is attached to

int sensorValue = 0; // value read from the pot
int outputValue = 0; // value output to the PWM (analog out)

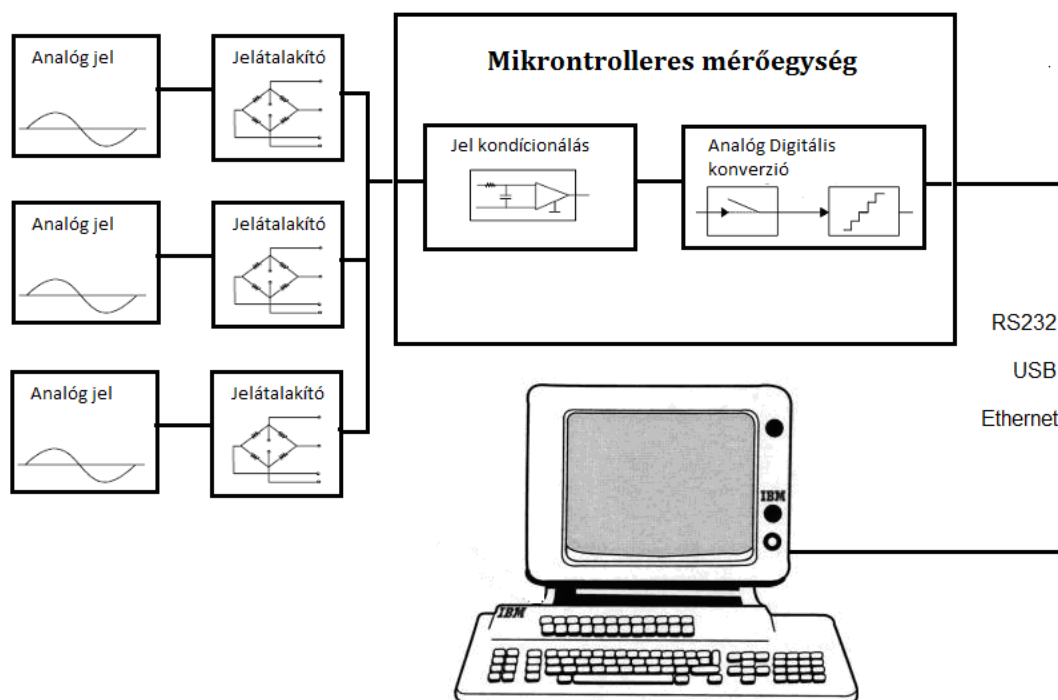
void setup() {
 // initialize serial communications at 9600 bps:
 Serial.begin(9600);
}

void loop() {
 // read the analog in value:
 sensorValue = analogRead(analogInPin);
 // map it to the range of the analog out:
 outputValue = map(sensorValue, 0, 1023, 0, 255);
 // change the analog out value:
 analogWrite(analogOutPin, outputValue);

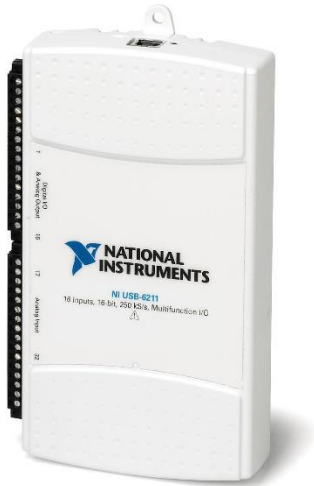
 // print the results to the serial monitor:
 Serial.print("sensor = ");
 Serial.print(sensorValue);
 Serial.print("\t output = ");
 Serial.println(outputValue);

 // wait 2 milliseconds before the next loop
 // for the analog-to-digital converter to settle
 // after the last reading:
 delay(2);
}
```

# SZÁMÍTÓGÉPES MÉRÉSTECHNIKA



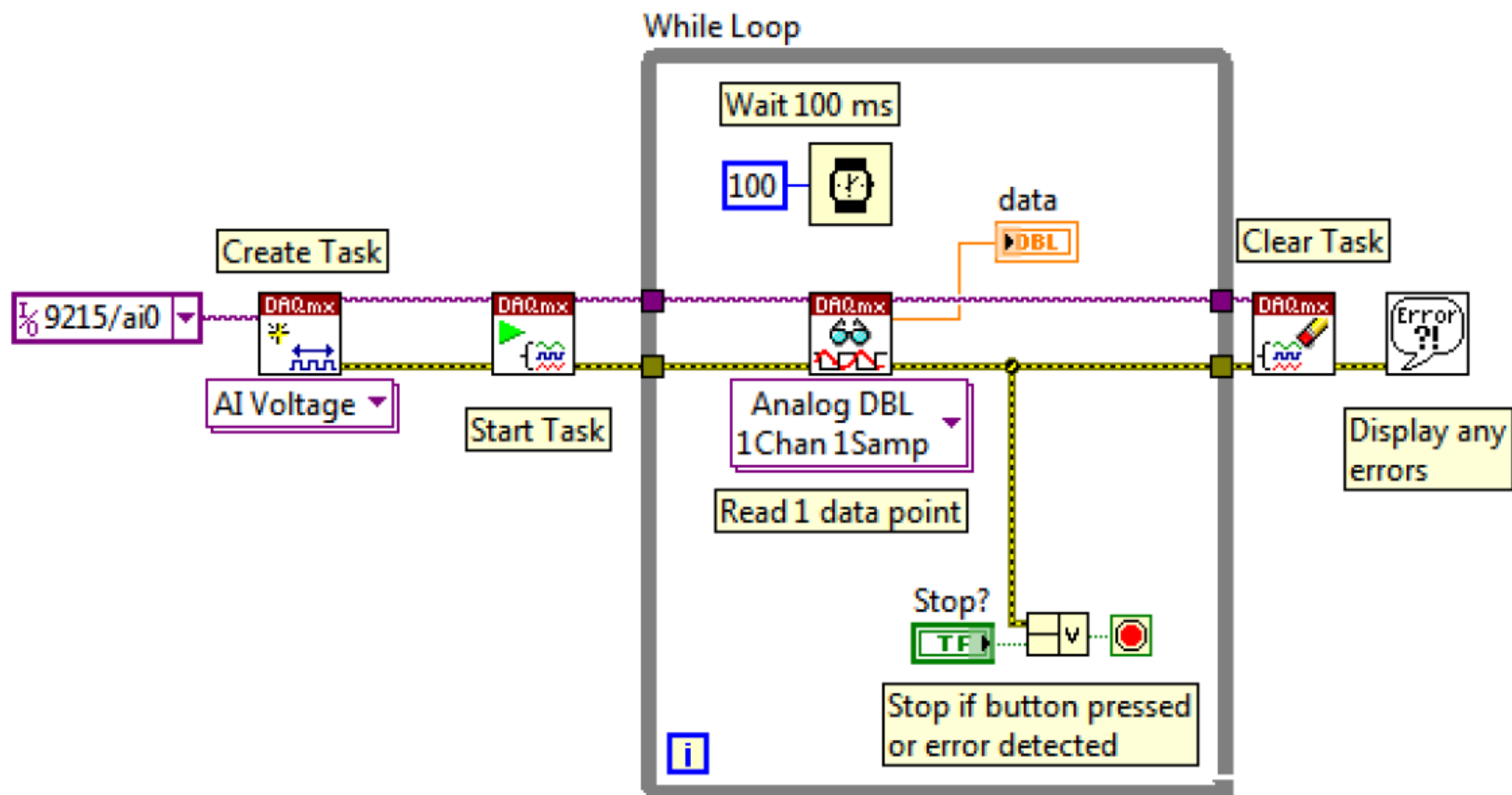
# NI- MULTIFUNKCIÓS IO ESZKÖZÖK



- Sok csatornás analóg I/O, digitális I/O és időzítők egyetlen készülékbe összerakva.  
Ingyenes C++ API (NI-DAQmx)
- Measurement Stúdió
- Labview
- LabWindows CVI

- A LabVIEW egy grafikus programnyelv
- Alkalmas számítógépes adatgyűjtésre, elemzésre és mérésre, mérési jegyzőkönyvek készítésére.
- A LabVIEW programok virtuális eszközök, melyek megjelenésükben és működésükben utánozzák a fizikai eszközöket, műszereket.
- Platform-semleges

## Analóg jel folyamatos egysatornás mérése



Duzzadás során mérhető fizikai mennyiségek

- Térfogatváltozás
- Tömegváltozás
- Hőmérsékletváltozás
- Nyomás

# TÖMEGVÁLTOZÁS

## Diffúziós modell

$$\frac{\partial m(\vec{r}, t)}{\partial t} = \nabla [D(m, \vec{r}) \nabla m(\vec{r}, t)] \longrightarrow \frac{\partial m(x, t)}{\partial t} = D \frac{\partial^2 m(x, t)}{\partial x^2}$$

$$\frac{M_t}{M_\infty} = 2 \left( \frac{Dt}{L^2} \right)^{\frac{1}{2}} \left\{ \pi^{-\frac{1}{2}} + \sum_{n=1}^{\infty} (-1)^n \operatorname{ierfc} \left( \frac{nL}{\sqrt{Dt}} \right) \right\} \longrightarrow \frac{M_t}{M_\infty} = k t^{\frac{1}{2}} \quad k = 2 \frac{D}{\sqrt{\pi} L^2}$$

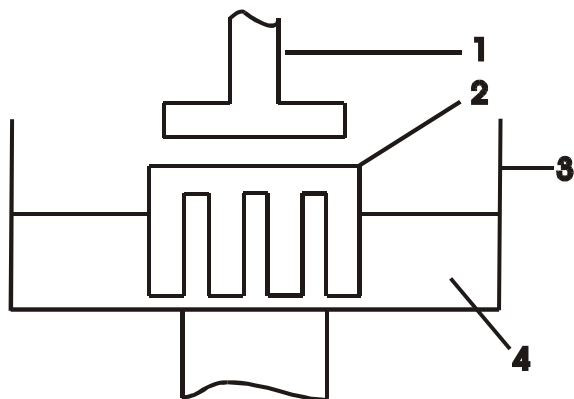
## Peppas

$$\frac{M_t}{M_\infty} = k t^n$$

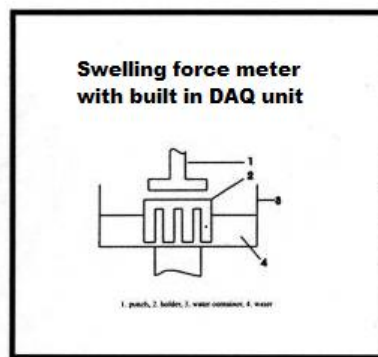
k: front faktor  
n: duzzadási exponens

| Duzzadási exponens (n) | Duzzadási mechanizmus |
|------------------------|-----------------------|
| 0.5                    | Fick diffúzió         |
| 0.1 < n < 0.5          | Anomális diffúzió     |

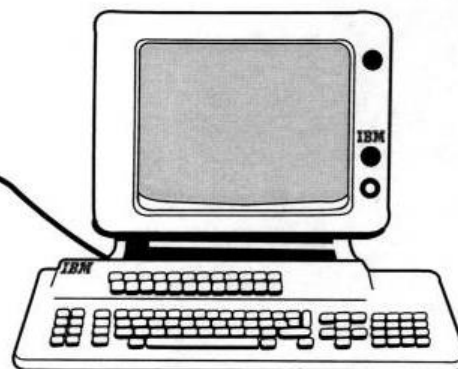
# DUZZADÁSI ERŐMÉRŐ KÉSZÜLÉK



1. nyomóidom,
2. mintatartó,
3. víztartály,
4. víz

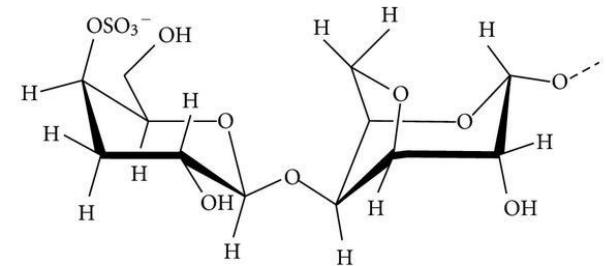
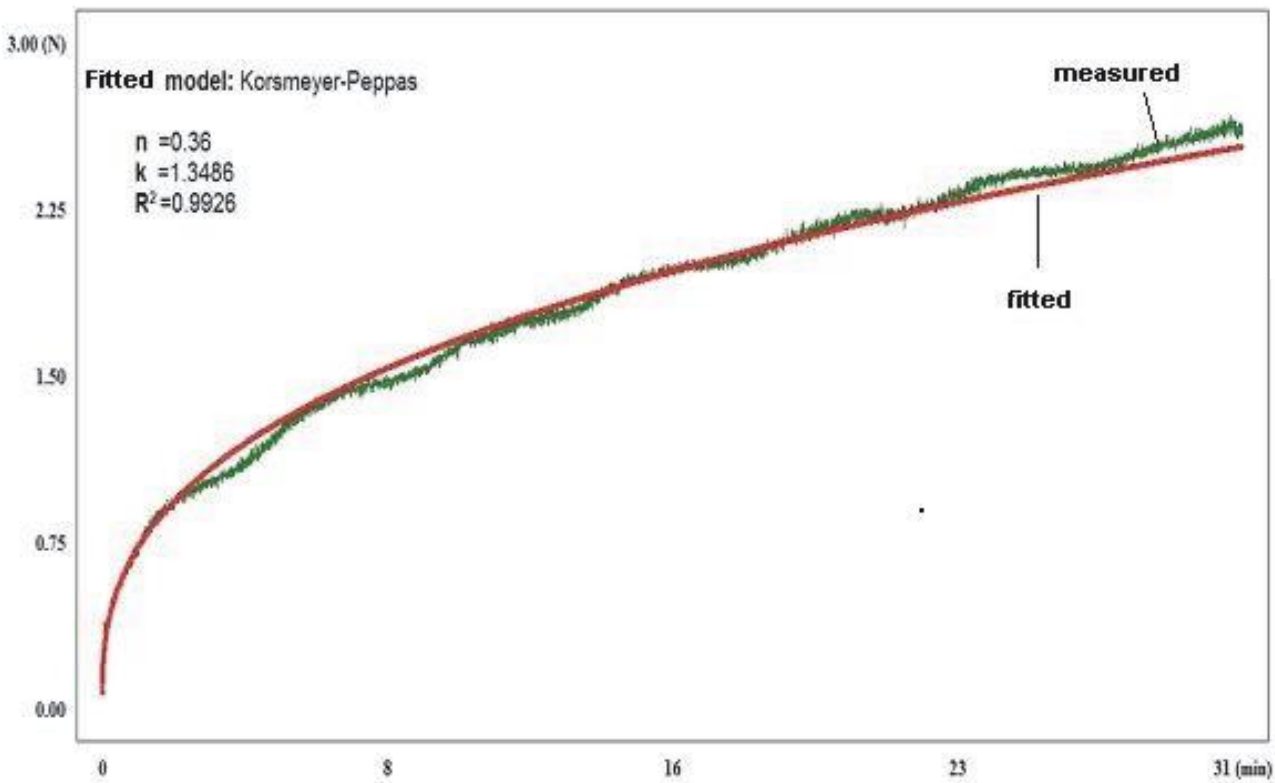


RS232





# IOTA-KARRAGÉN DUZZADÁSA



Préserő: 15kN

András Kelemen et al.: J Pharm Biomed Anal 112: pp. 85-88. (2015)

# FELADATOK

- Mit nevezünk beágyazott rendszernek?
- Mi a különbség a mikrokontroller és a mikroprocesszor között?
- Mire használható a LabView?

# KÖSZÖNÖM A FIGYELMET!

**SZÉCHENYI** 2020



MAGYARORSZÁG  
KORMÁNYA

Európai Unió  
Európai Szociális  
Alap



**BEFEKTETÉS A JÖVŐBE**