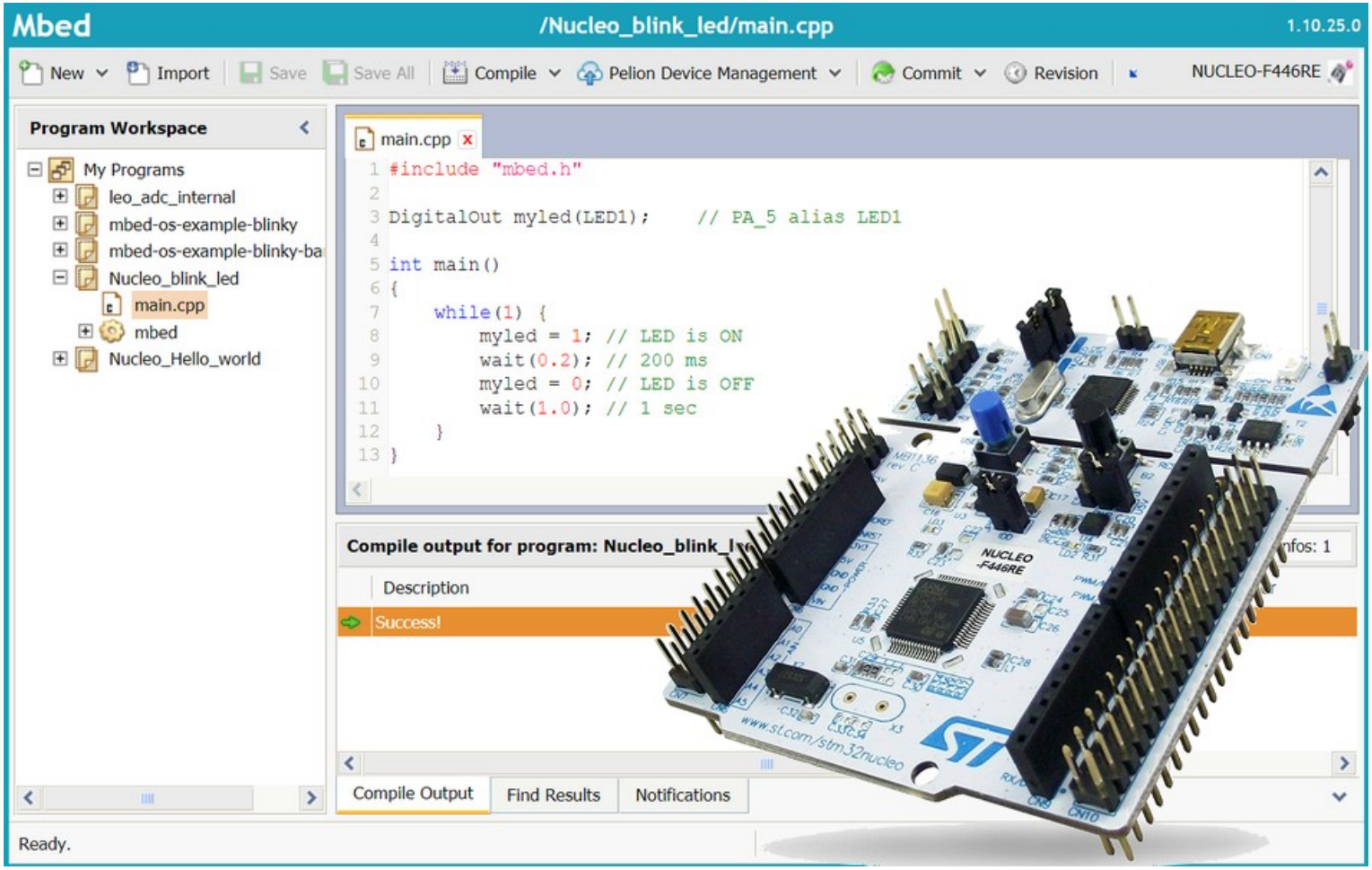
STM32 mikrovezérlők programozása ARM mbed környezetben



STM32 F4

ARM mbed enabled

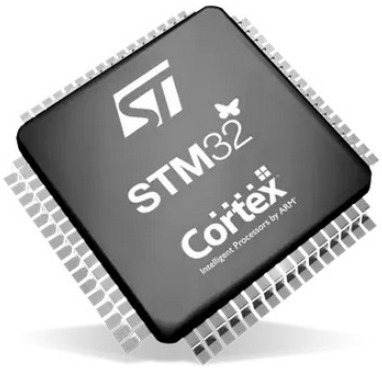
2. Impulzus-szélesség moduláció (PWM)

Felhasznált és ajánlott irodalom

- Cserny István: A FRDM-KL25Z kártya programozása mbed környezetben
- Rob Toulson and Tim Wilmhurst:
  Fast and Effective Embedded Systems Design: Applying the ARM mbed
- Perry Xiao: Designing Embedded Systems and the Internet of Things (IoT) with the ARM mbed
- Dogan Ibrahim: ARM-based Microcontroller Projects Using mbed
- **ARM mbed honlap: https://os.mbed.com/**
- ARM mbed Compiler: https://ide.mbed.com/compiler/
- ARM mbed 2 Handbook (elavult): https://os.mbed.com/handbook/Homepage
- ARM mbed 2 Cookbook (elavult): https://os.mbed.com/cookbook/Homepage
- ARM mbed forráskód: https://github.com/ARMmbed/mbed-os

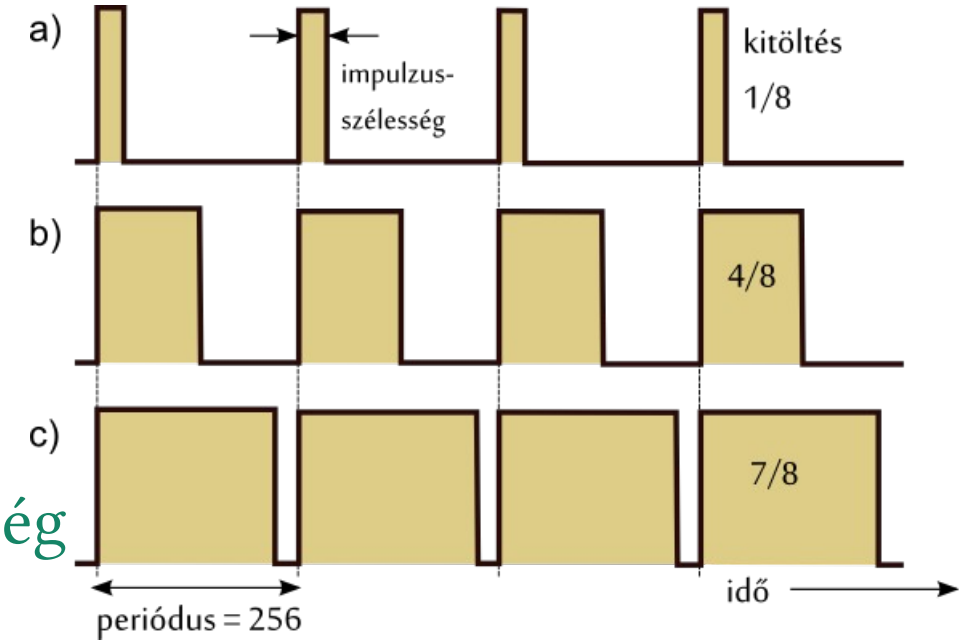
Adatlapok:

- **STM32F446RE** adatlap és termékinfo
- **STM32F446** Family Reference Manual



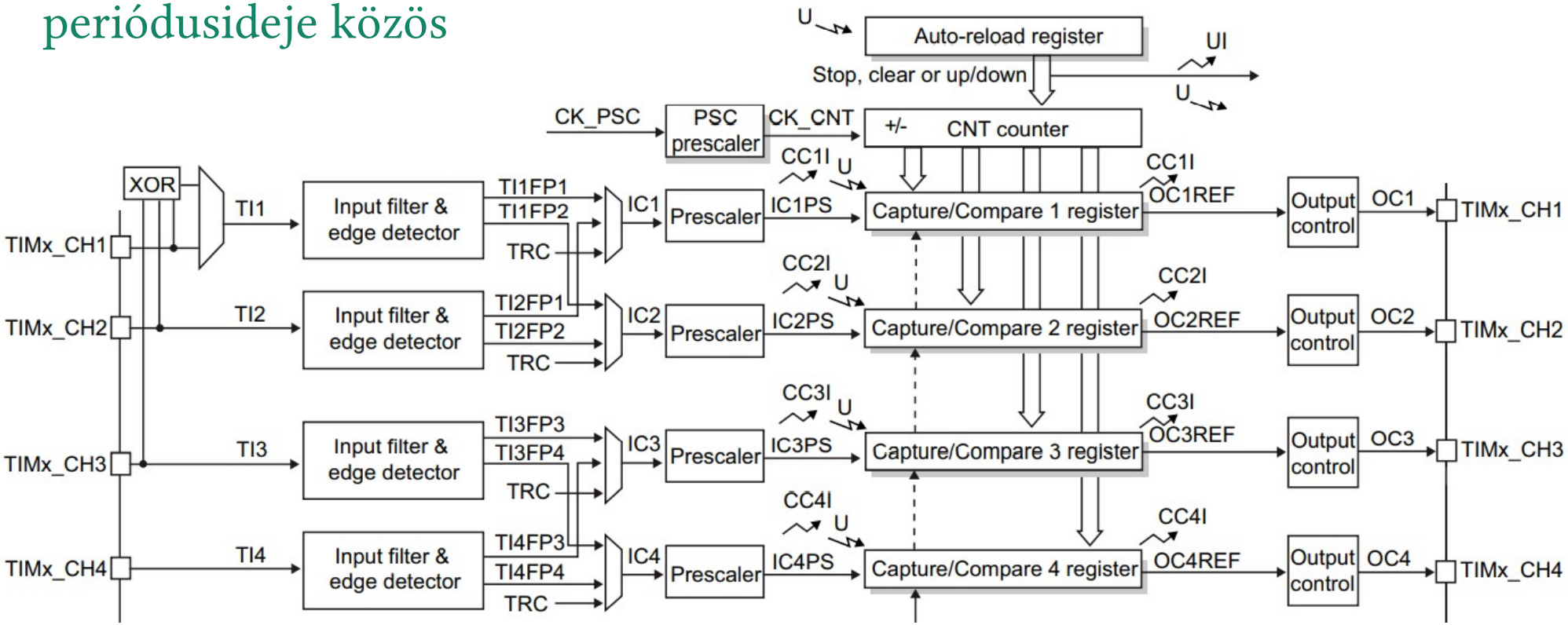
Impulzus-szélesség moduláció (PWM)

- Mikrovezérlő alkalmazásoknál gyakori feladat valamilyen mennyiséget (pl. egy fogyasztó teljesítménye) folyamatos vagy fokozatos szabályozása
- Egyik lehetőség az **impulzus-szélesség moduláció** (Pulse Width Modulation - **PWM**) alkalmazása. Ilyen jeleket hardveresen a mikrovezérlő időzítő (timer) és digitális összehasonlító (comparator) perifériáinak felhasználásával állíthatunk elő.
- Elv: az analóg kimenő feszültség helyettesíthető digitális impulzussorozat-jelekkel, amelynek átlagfeszültsége megegyezik az analóg feszültségjellel
- Jellemző paraméter a **periódusidő** és a **kitöltés,** ami az impulzusszélesség és a periódusidő hányadosa



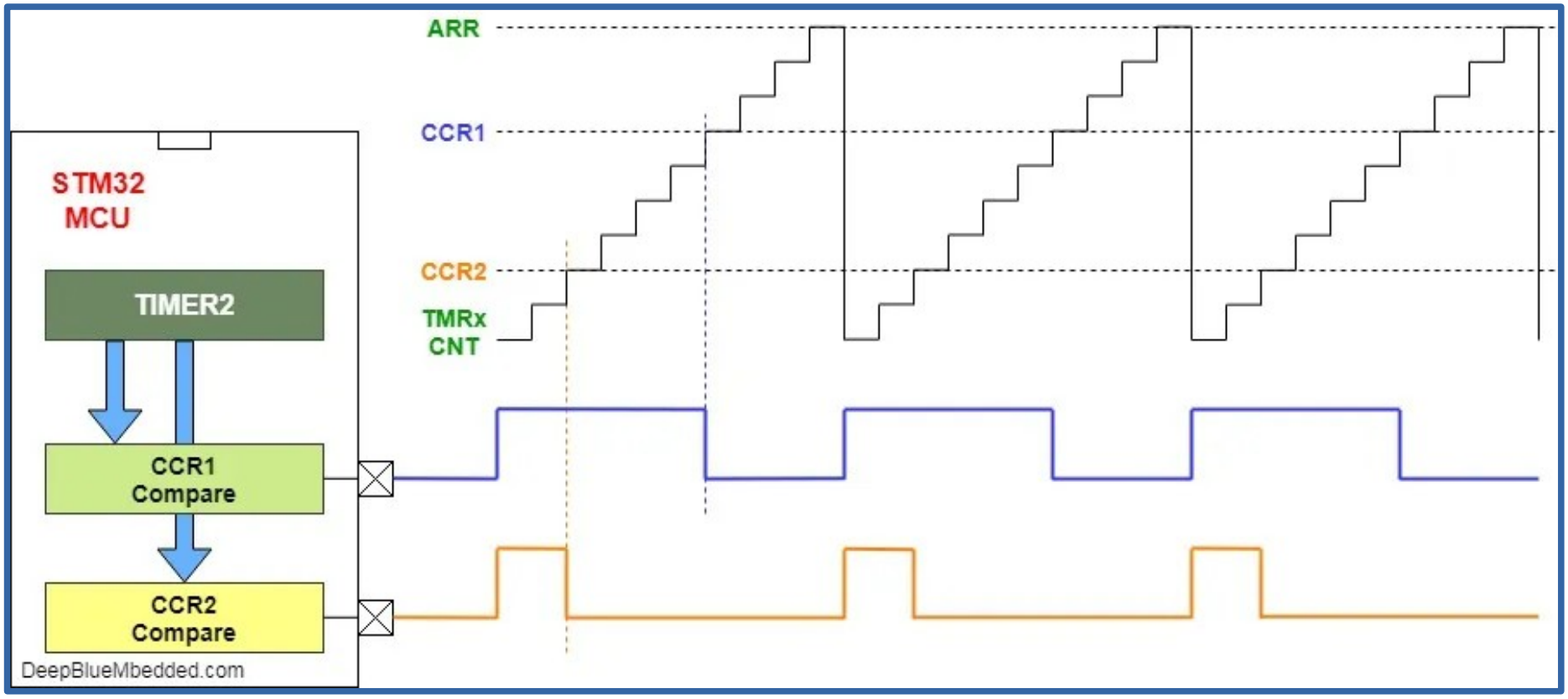
Az STM32F446RE PWM csatornái

- Az **STM32F446RE** mikrovezérlő 10 általános és 2 fejlett időzítővel rendelkezik, amelyek 2-4 csatornában képesek **PWM** jel előállítására (többségük 16 bites, TIM2 és TIM5 32 bites felbontású)
- Az **STM32F446RE** mikrovezérlőnek az adatlap szerint 32 db **PWM** kivezetéssel rendelkezik, de egyidejűleg nem mindegyik használható
- További kötöttség, hogy az egy időzítőhöz tartozó csatornák periódusideje közös



Hogyan működik a PWM?

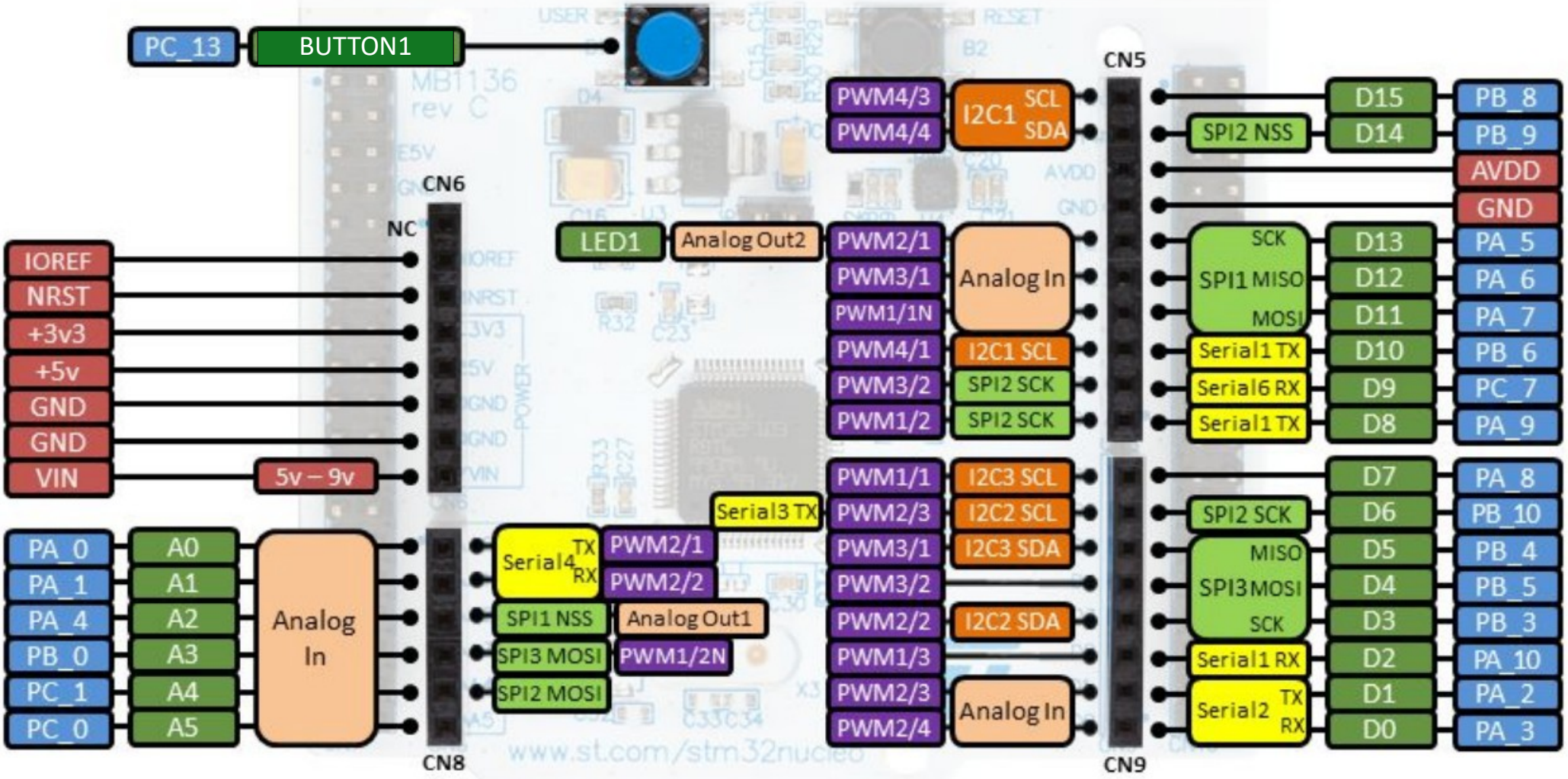
- Az ábra a legegyszerűbb, ún. él igazított (*edge aligned*) üzemmódot mutatja, ahol az **OC***n* kimenetek magas szintre váltanak a számláló nullázásakor, s alacsonyra váltanak vissza, amikor a számláló értéke megegyezik az adott csatorna összehasonlítási értékével (CCR*n*)



Az ábra forrása: deepbluembedded.com/stm32-pwm-example-timer-pwm-mode-tutorial/

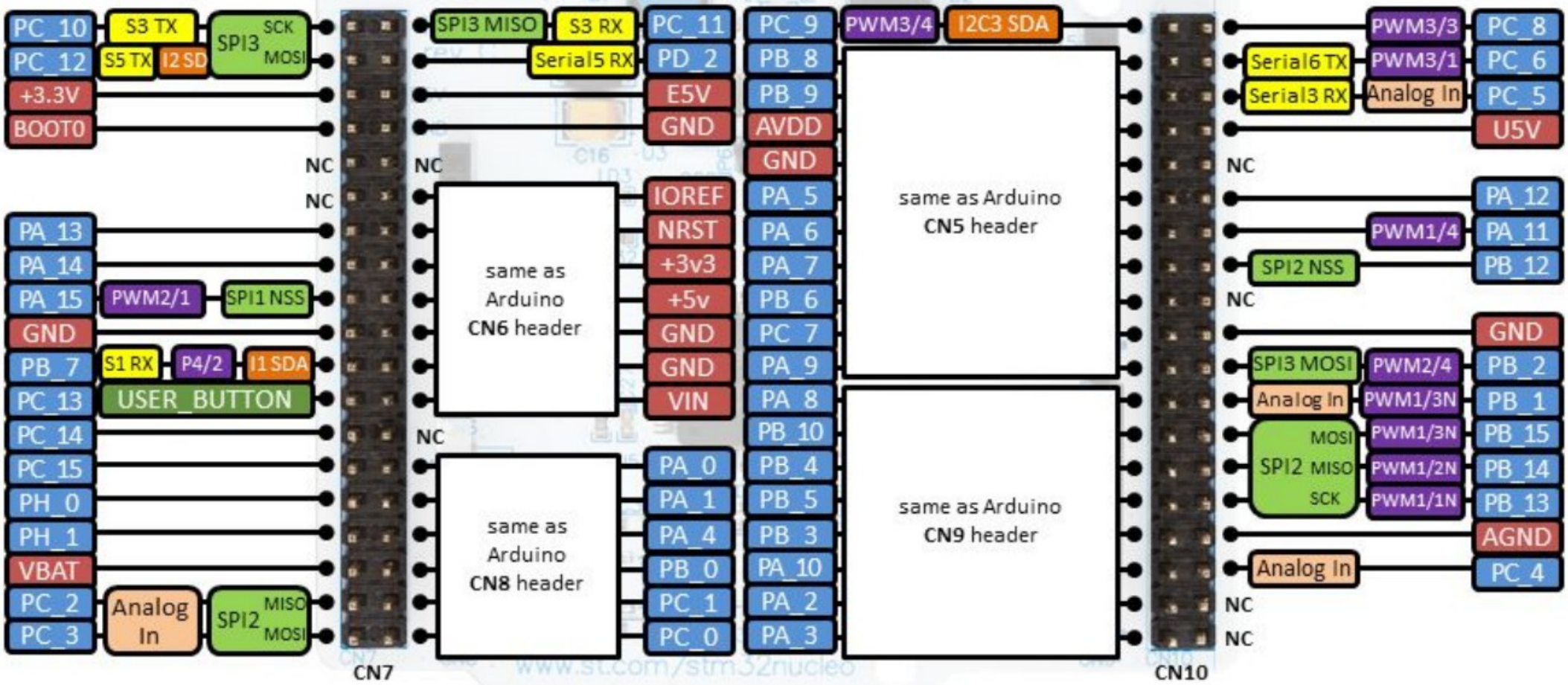
Arduino kompatibilis kivezetések

- Kivezetés azonosításra a kék, illetve a sötétzöld címkék használhatók (pl. PA_5, D13, vagy LED1)



Morpho kivezetések

Nucleo F446RE
Morpho Headers



A PwmOut objektumosztály

- Az **mbed** környezetben **a PwmOut** objektumosztály szolgál a **PWM** kimenetek konfigurálására és kezelésére
- A taggfüggvények felsorolása az alábbi táblázatban található

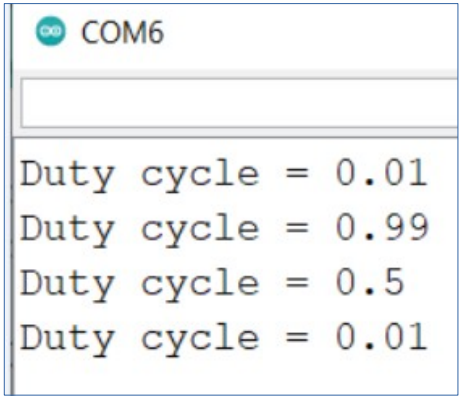
Függvény	Felhasználás
PwmOut név(PinName pin**)**	A konstruktor, amely létrehoz egy "név" nevű **PwmOut** objektumot és a pin paraméterrel megadott kimenethez rendeli
write(float adat**)**	A kitöltési tényező beállítása a megadott (0.0 - 1.0 közötti) értékre
read()	Az utoljára beállított kitöltési tényező adja meg (0.0 - 1.0 közötti float)
period(float sec**)**	A periódusidő megadása másodpercben (float típusú paraméter)
period_ms(int ms**)**	A periódusidő megadása milliszekundumban (int típusú paraméter)
period_us(int us**)**	A periódusidő megadása mikroszekundumban (int típusú paraméter)
pulsewidth(float sec**)**	Az impulzusszélesség (a magas szintű állapot ideje) megadása másodpercben
pulsewidth_ms(int ms**)**	Az impulzusszélesség megadása milliszekundumban (int típusú paraméter)
pulsewidth_us(int us**)**	Az impulzusszélesség megadása mikroszekundumban (int típusú paraméter)
operator = adat	Rövidített alak **név.write(adat)** helyett
operator int()	Rövidített alak **név.read()** helyett

Lab02_pwm_led

- Vezéreljük a beépített LED1 fényerejét a soros porton keresztül!

```
#include "mbed.h"
PwmOut myled(LED1);          // PA_5, a beépített LED
Serial pc(USBTX, USBRX);     // soros terminál a PC-hez

int main() {
    pc.baud(115200);         // Adatsebesség beállítása
    myled.period_ms(20);     // Periódus = 20 ms (50 Hz)
    myled.write(1.0);        // LED kezdetben kikapcsolva
    while(1) {
        char c = pc.getc(); // Beolvasunk egy karaktert
        if(c=='1') {
            myled = 0.01f;  // LED 1 % kitöltéssel
            pc.printf("Duty cycle = 0.01\r\n");
        }
        else if(c=='5') {   // LED 50 % kitöltéssel
            myled = 0.5f;
            pc.printf("Duty cycle = 0.5\r\n");
        }
        else if(c=='9') {
            myled = 0.99f;  //LED 99 % kitöltéssel
            pc.printf("Duty cycle = 0.99\r\n");
        }
        wait(0.2);
    }
}
```

Import into Compiler

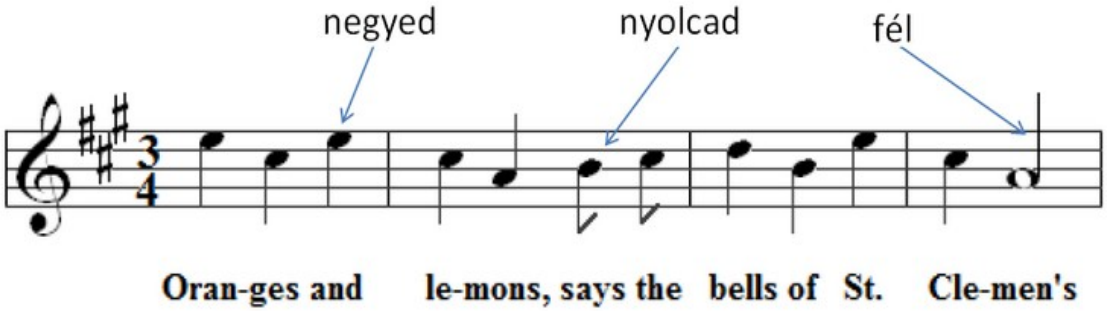


Egy terminálablakból küldjünk ’1’, ’5’ vagy’9’ karaktereket a PWM kitöltés vezérléséhez!

INPUT	PWM duty
1	1 %
5	50 %
9	99 %

Lab02_pwm_music

- A **Lab02_pwm_music** programmal egy rövid zenét játszunk le. Egy piezo csipogót hajtunk meg a keltett **PWM** jellel. A programot, amely az ***Oranges and Lemons*** c. régi angol népdalt játssza le Rob Toulson és Tim Wilmshurst: ARM mbed Course Material - Pulse Width Modulation tananyagából vettük kölcsön



- A kotta hangjegyeihez tartozó frekvencia értékek a mellékelt táblázatban találhatók
- A hangkeltés módja (pl. „normál A” hang):

```
PwmOut buzzer(D3);
buzzer.period(1.0/440);
buzzer = 0.5;
```

Szó/Szótag	Hang	Frekvencia (Hz)	Hossz
oran	E	659	1
ges	C#	554	1
and	E	659	1
le	C#	554	1
mons	A	440	1
says	B	494	½
the	C#	554	½
bells	D	587	1
of	B	494	1
st	E	659	1
clem	C#	554	1
en's	A	440	2

Lab02_pwm_music

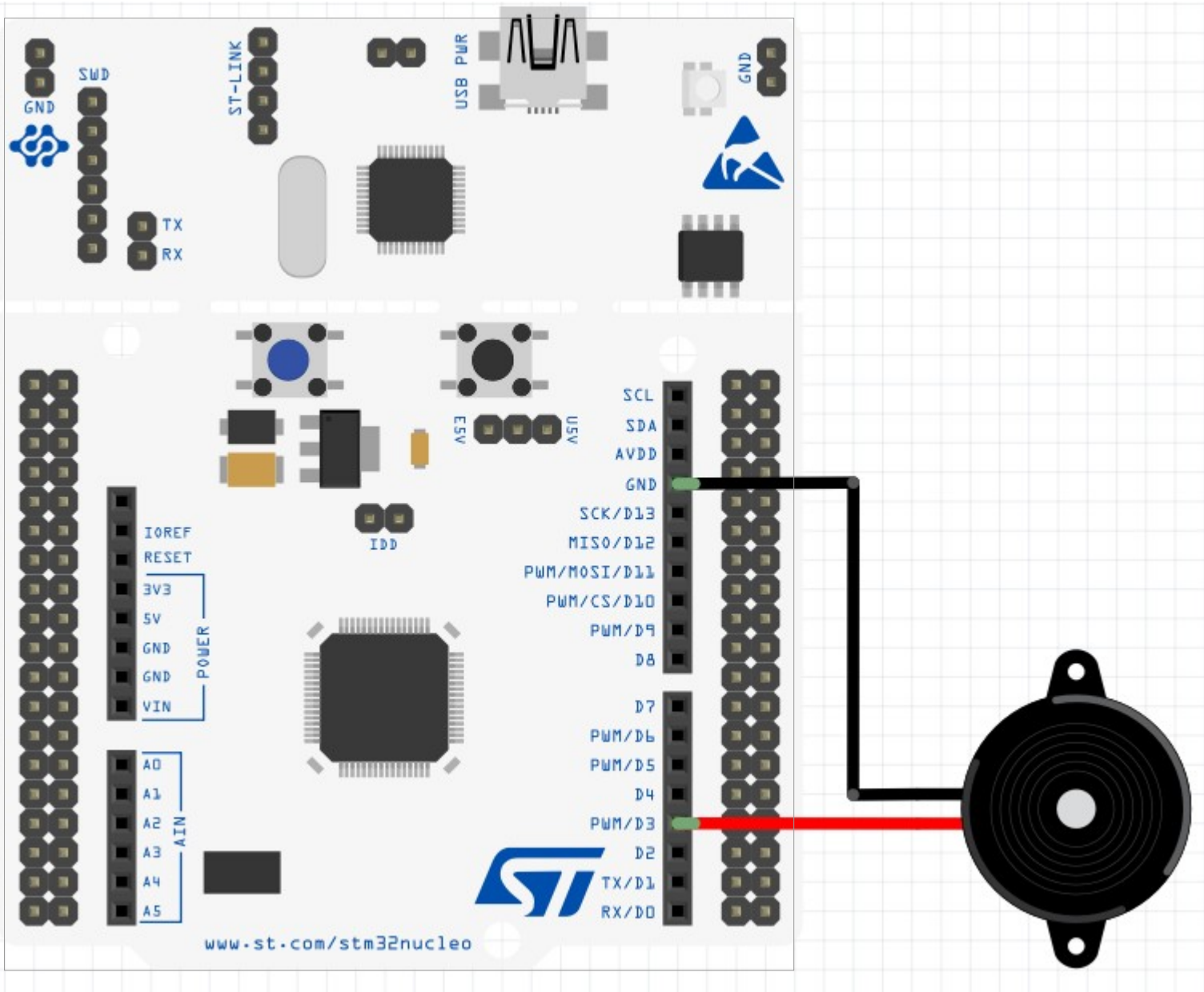
- A hangjegyek frekvenciáját és az időtartamok adatait tömbökben tároljuk és egy for ciklusban
- Itt – bár **PWM** jelet keltünk – a kitöltés mindig 50 % lesz, csak a frekvenciát változtatjuk

```
//Borrowed from Rob Toulson and Tim Wilmshurst: ARM mbed Course Material - PWM
//Link: https://developer.mbed.org/cookbook/Course-Notes
#include "mbed.h"

PwmOut buzzer(D3);
float frequency[]={659,554,659,554,550,494,554,587,494,659,554,440}; //frequency
array
float beat[]={1,1,1,1,1,0.5,0.5,1,1,1,1,2};      //beat array

int main() {
    while (1) {
        for (int i=0; i<12; i++) {
            buzzer.period(1/(frequency[i]));     // set PWM period
            buzzer=0.5;                          // set duty cycle
            wait(0.5*beat[i]);                   // hold for beat period
        }
        wait(2);                                 // két lejátszás közötti szünet
    }
}
```

Import into Compiler

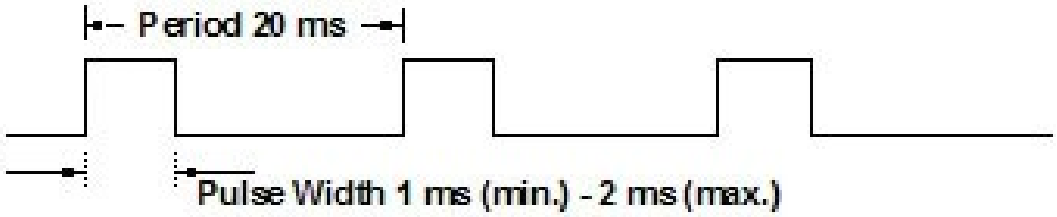
Lab02_pwm_music: a kapcsolás

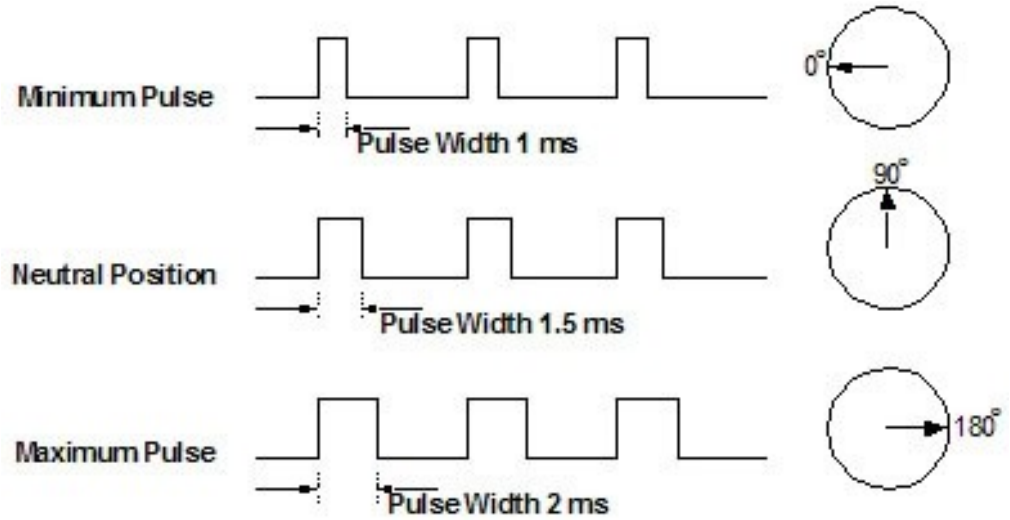
- Kössük a piezo csipogó ’+’ jelű kivezetését a **D3** kimenetre!
- Kössük a másik kivezetést a GND pontra!
- Ügyeljünk rá, hogy ún. passzív piezo csipogót használjunk!

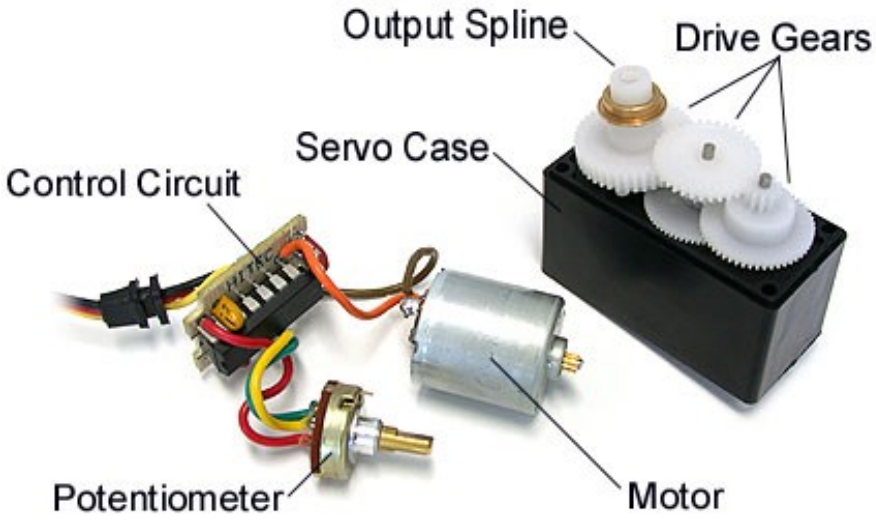


Szervó motorok vezérlése

- A rádiótávirányítású modellekben elterjedt szervó motorokat "RC servo" néven emlegeti az angol nyelvű szakirodalom (RC = Radio Controlled)
- A szervó motorok felépítése és tipikus jelalakja az alábbi ábrákon látható (a képek forrása a ServoCity honlapja).
- A szervó vezérlő jelét **PWM**-mel is előállíthatjuk







Lab02_pwm_servo

- Két szélső helyzet között oda-vissza mozgatunk egy szervo motort
- A vezérlés egy **PwmOut** kivezetés (**D3**) felhasználásával történik
- A periódusidőt 20 ms-ra állítjuk be, az impulzusszélességet pedig 1000 és 2000 µs között változtatjuk, 20 µs-os lépésekben

Import into Compiler

```
#include "mbed.h"
PwmOut servo(D3);                           // D3 (PB_3) lesz a kimenet

int main() {
    servo.period_ms(20);                    //Periódus = 20 ms (f=50 Hz)
    while(true) {
        for(int pw=1000; pw <= 2000; pw=pw+20) {
            servo.pulsewidth_us(pw);        // Új szervó pozíció
            wait_ms(200);
        }
        wait_ms(1000);                      // Visszafordulás előtt várunk
        for(int pw=2000; pw >= 1000; pw=pw-20) {
            servo.pulsewidth_us(pw);        // Új szervó pozíció
            wait_ms(200);
        }
        wait_ms(1000);                      // Visszafordulás előtt várunk
    }
}
```