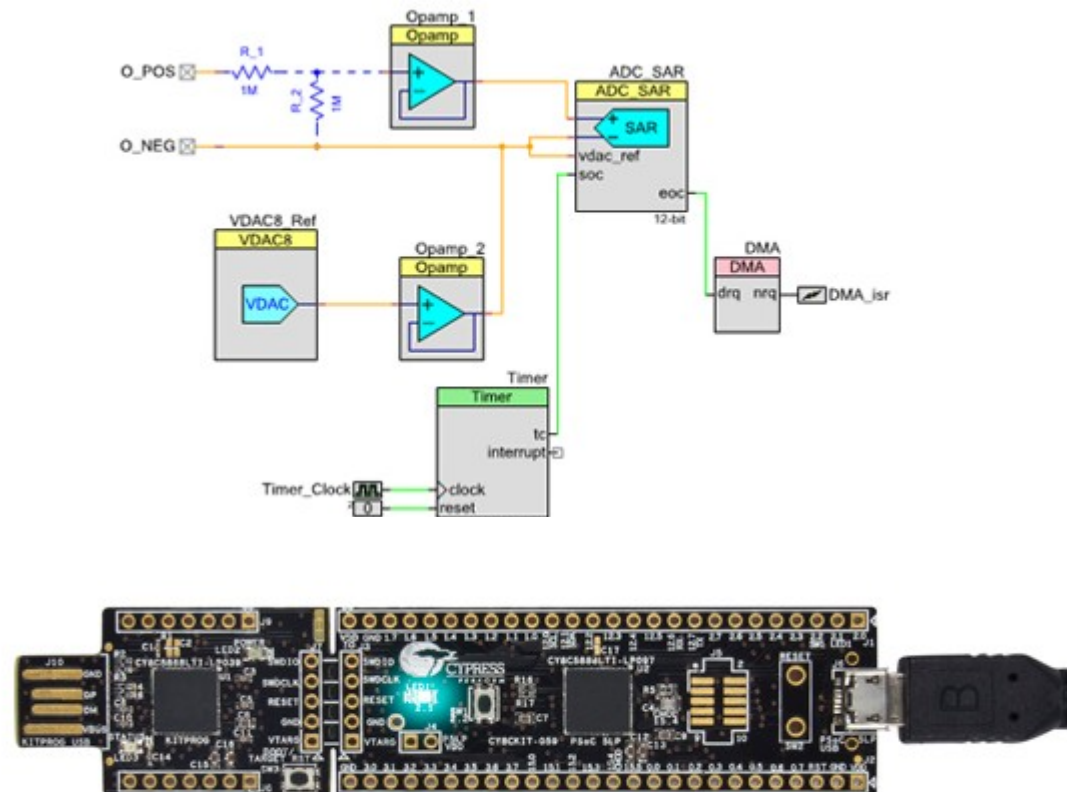
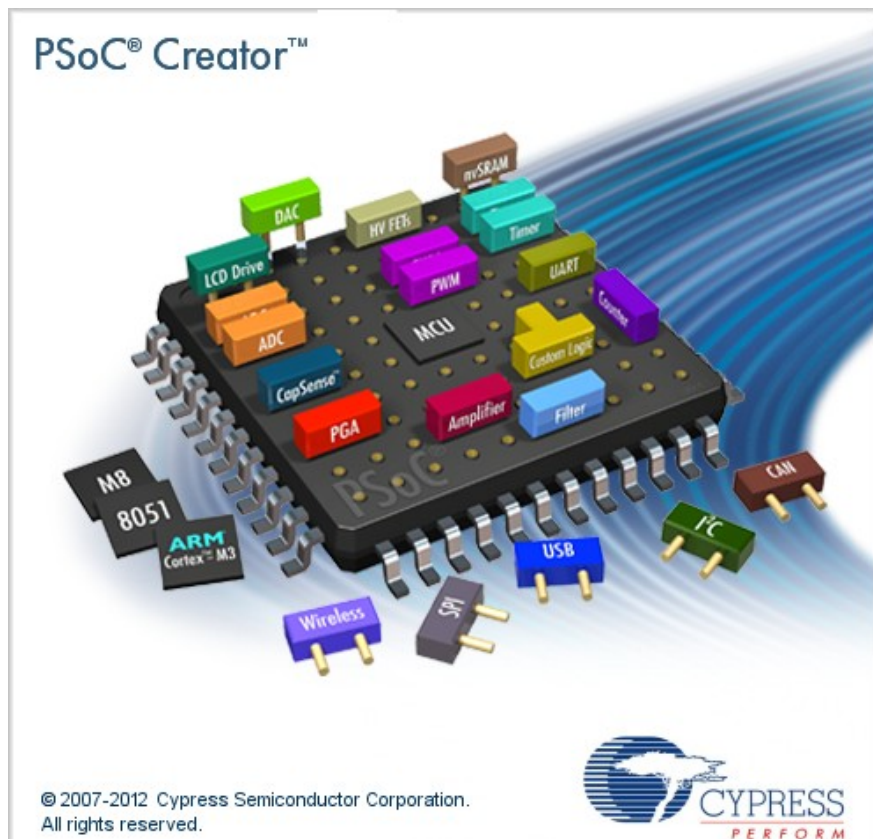


# Újrakonfigurálható eszközök



## 9. Cypress PSOC 5LP prototípus kártya - az első lépések

# Felhasznált irodalom és segédanyagok

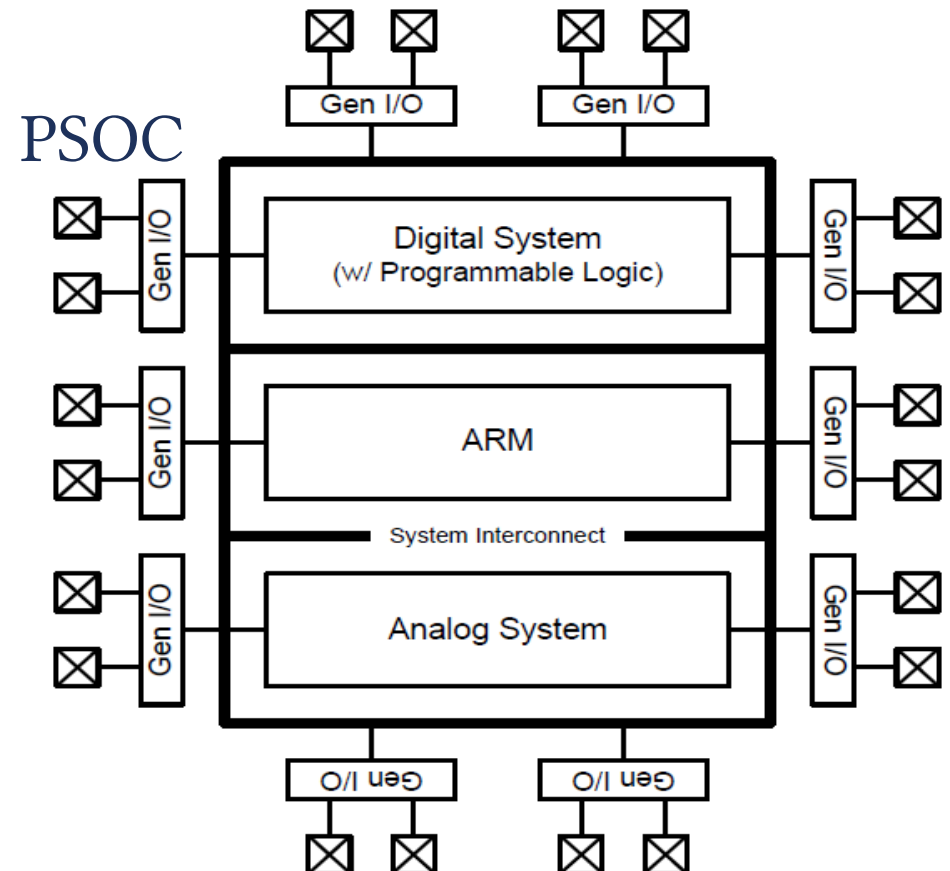
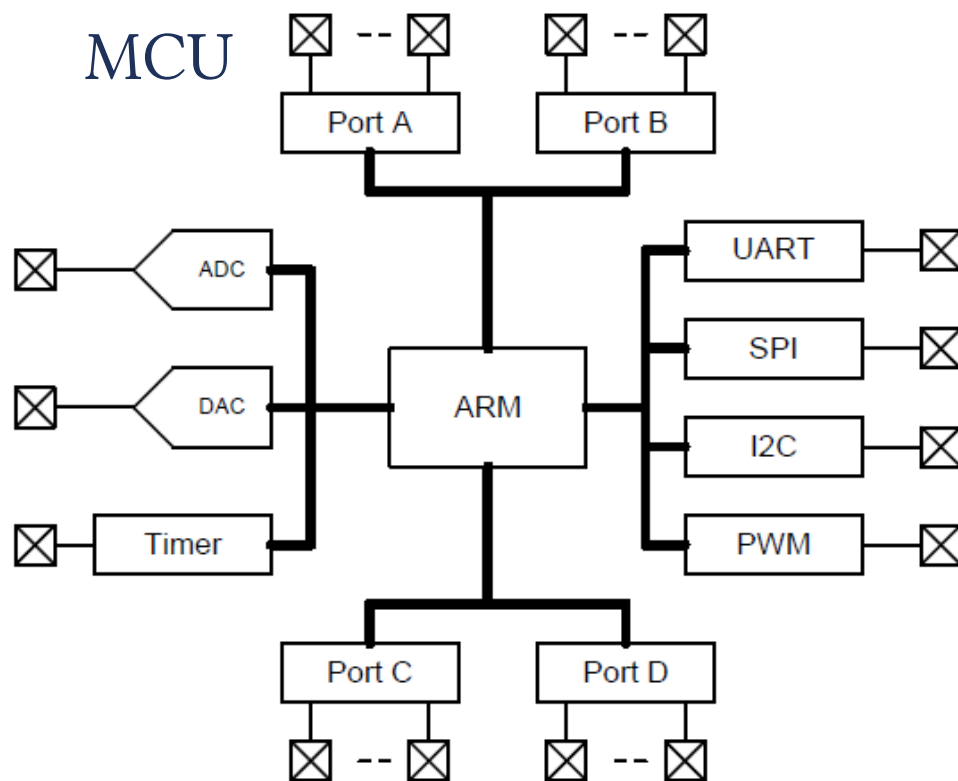
---

- Cypress: CY8C58LP Family Datasheet
- Cypress: PSoC 5LP Architecture Technical Reference Manual)
- Cypress: CY8CKIT-059 Prototyping Kit Guide
- Cypress: AN77759: Getting Started with PSoC® 5LP
- Cypress: PSoC® Creator™ User Guide
- Yuri Magda: Cypress PSoC 5LP Prototyping Kit Measurement Electronics
- Cserny István: PSoC 5LP Mikrokontrollerek programozása

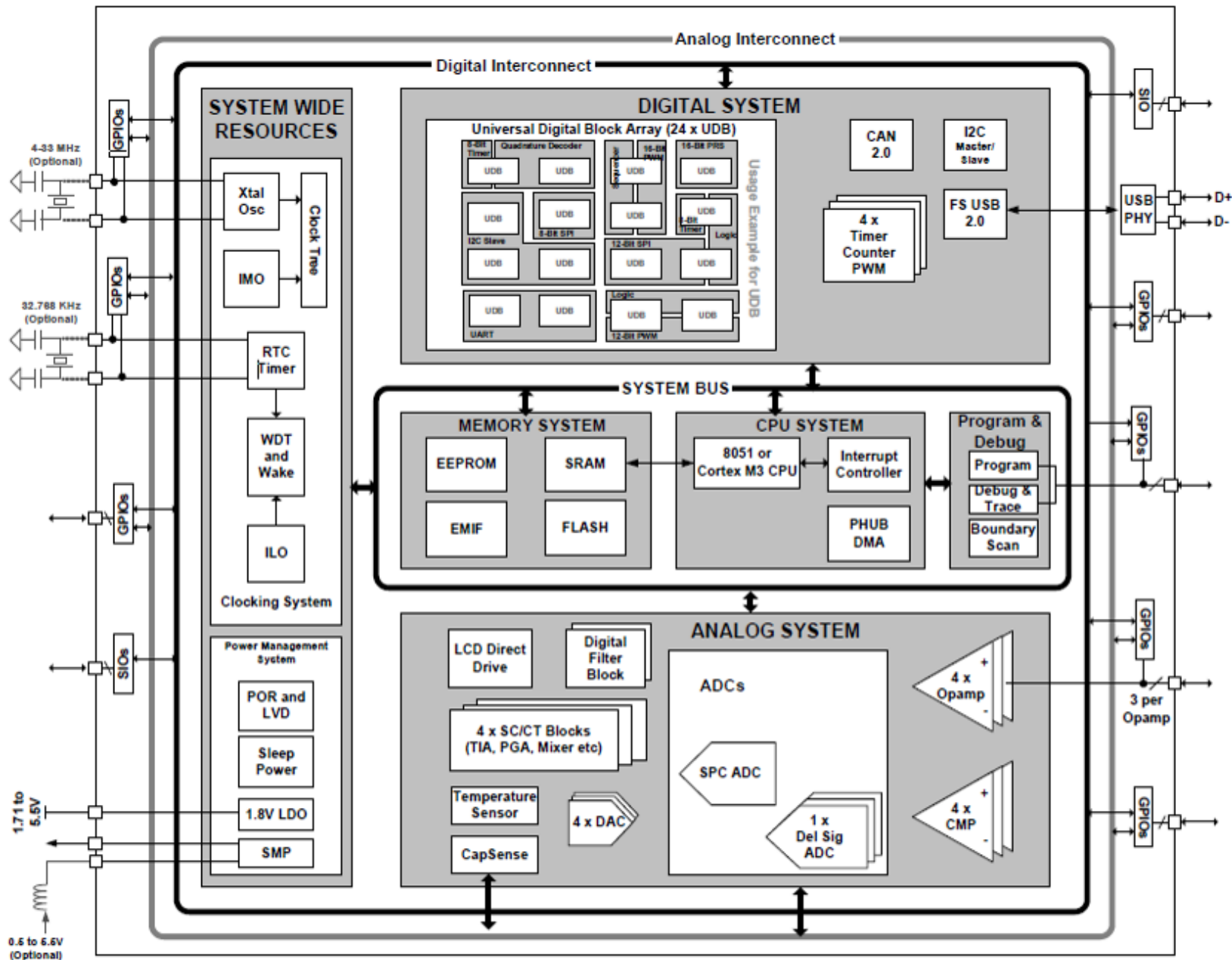


# MCU és PSOC – mi a különbség?

- A hagyományos MCU központi eleme a CPU
- A PSOC esetében a CPU nem központi eleme az adatáramlásnak: a digitális és analóg perifériák és a kivezetések egy jól konfigurálható jel- és adatbusz mátrix hálózaton keresztül kapcsolódnak össze



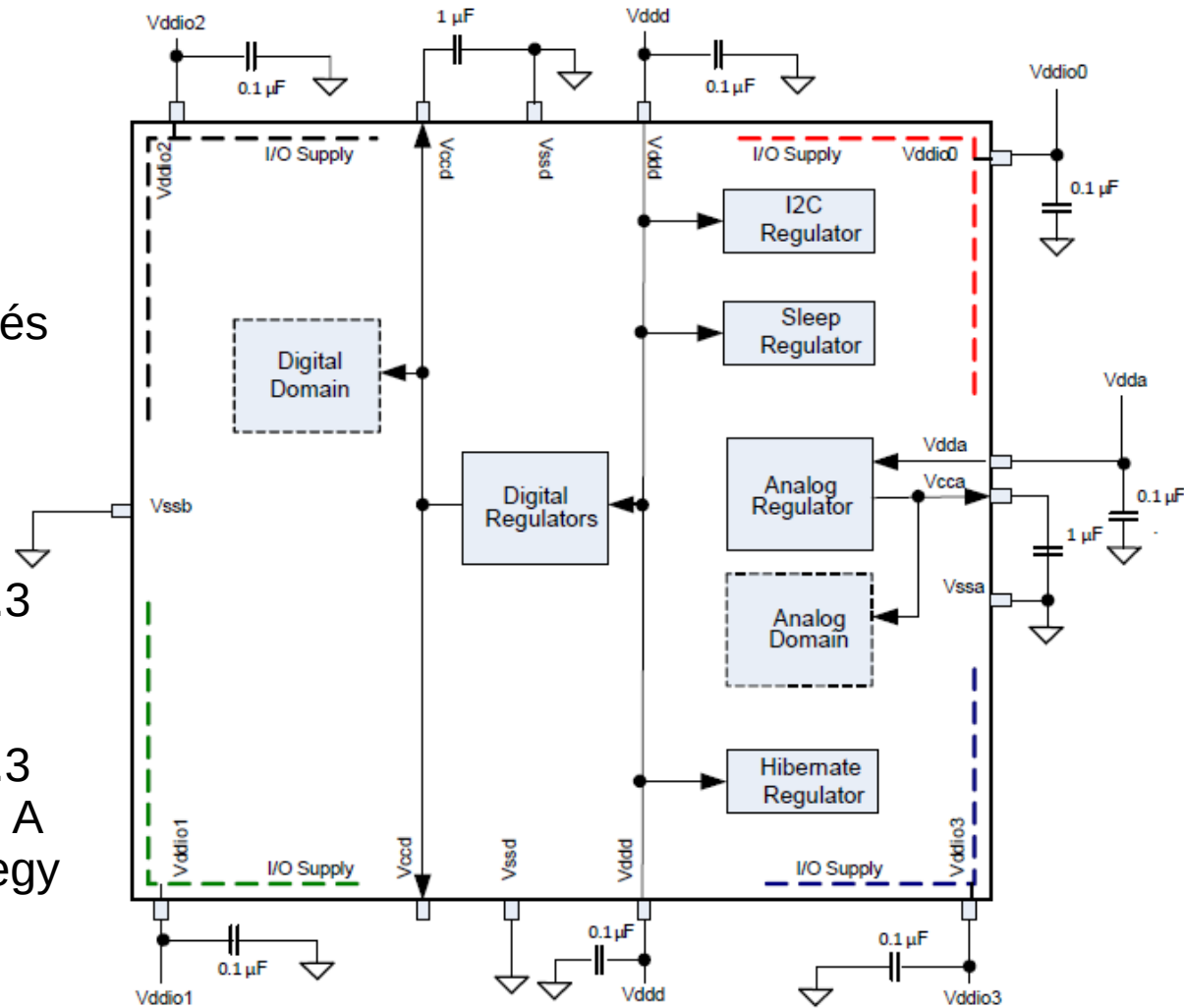
# PSOC 3, PSOC 5LP rendszerszintű vázlat



# Tápellátás és tápfeszültségek

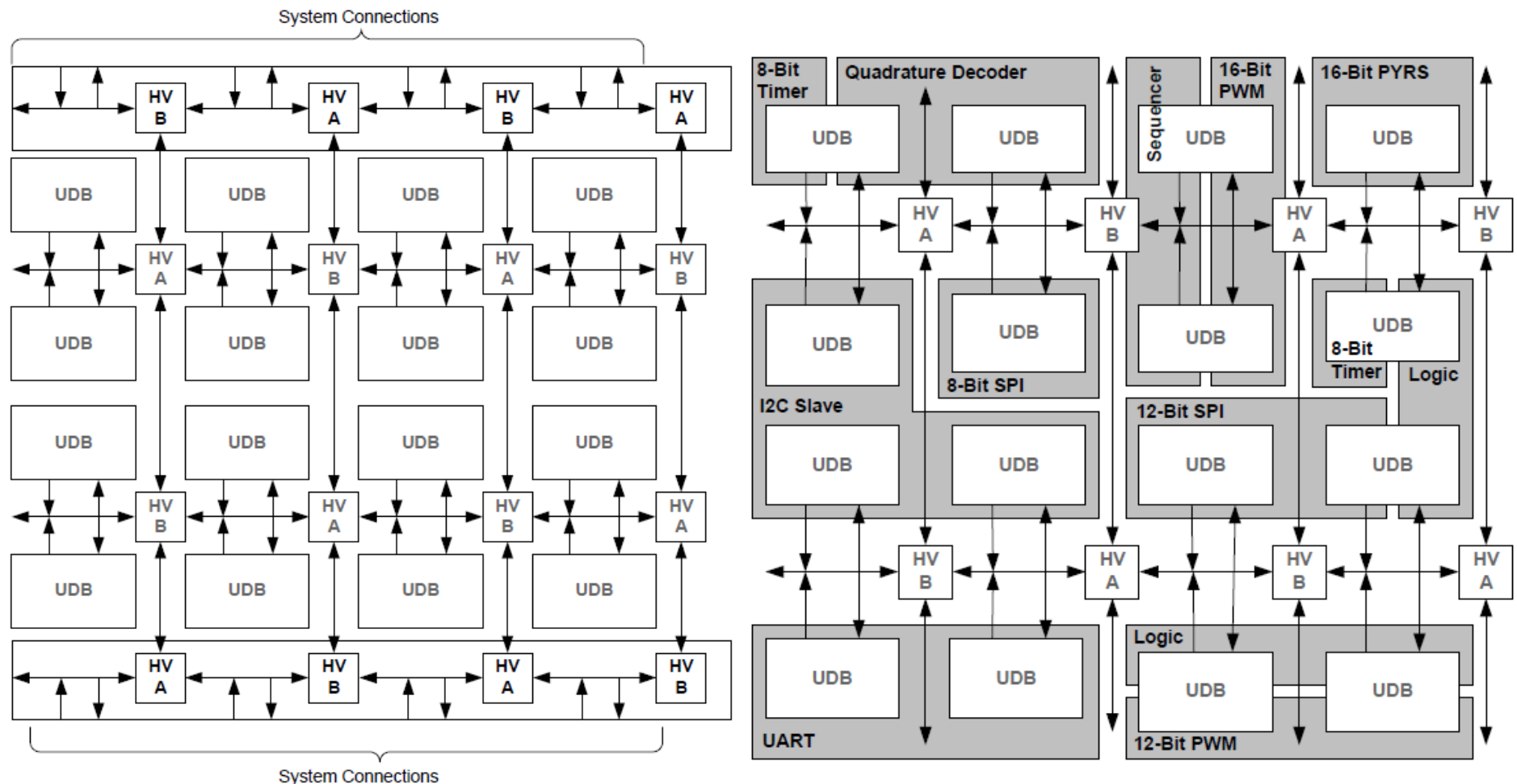
## Standard tápellátási konfiguráció

- Nincs használatban boost töltéspumpa
- $V_{dda} \geq V_{ddio0/1/2/3}$
- $V_{dda} = 1.8 - 5.5V$
- Tápellátási szabályok
- V<sub>dda</sub>: Ez legyen a rendszerben a legmagasabb feszültség. Ellátja a magasabb feszültségű analóg domént és a belső stabilizátorokat.
- V<sub>ddd</sub>: A digitális rendszer belső stabilizátorait táplálja.
- V<sub>cca</sub>: Az analóg rész belső stabilizátorának kimenete. Egy külső 1.3 uF szűrőkondenzátor kell rá a föld felé.
- V<sub>ccd</sub>: A digitális rész belső stabilizátorának kimenete. Egy külső 1.3 uF szűrőkondenzátor kell rá a föld felé. A két V<sub>ccd</sub> kimenetet össze kell kötni, s egy közös 1.3 uF szűrőkondenzátoron osztoznak.
- V<sub>ddio0/1/2/3</sub>: Független I/O tápfeszültségek. Tetszőleges értékűek lehetnek 1.8V és V<sub>dda</sub> között.



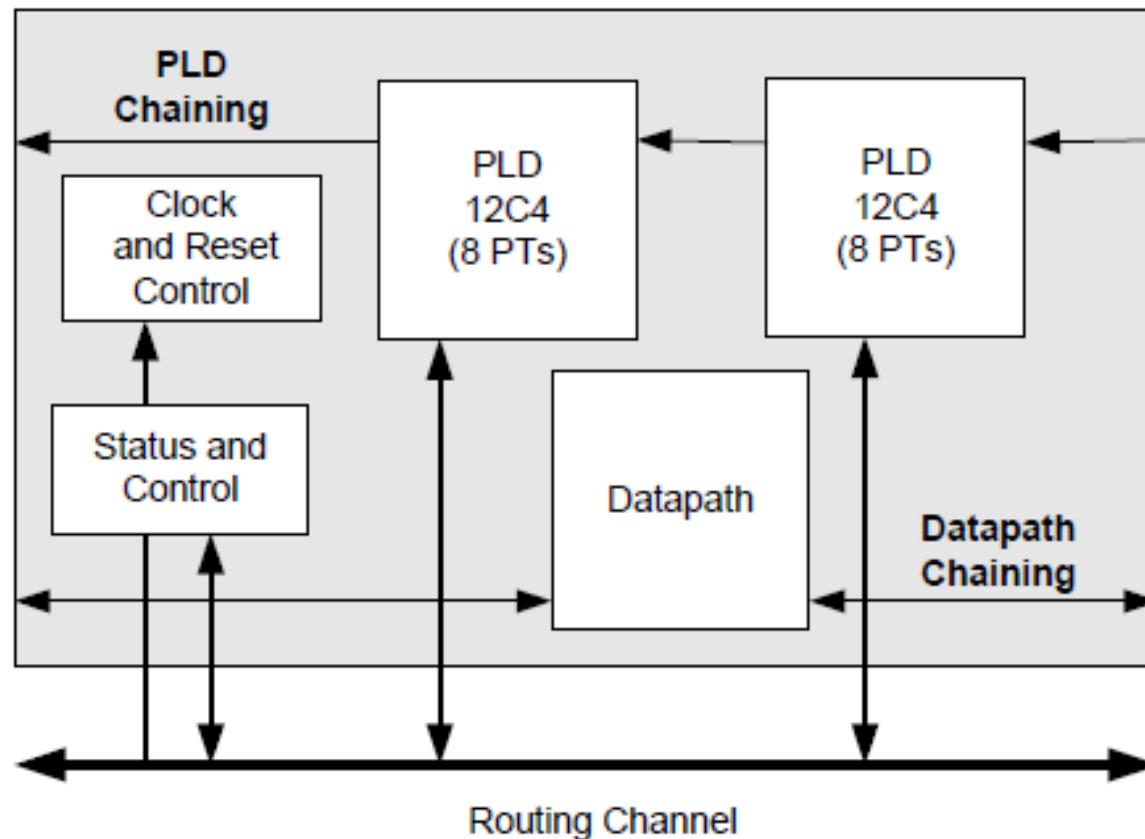
# Univerzális digitális blokkok (UDB)

- Az Univerzális Digitális Blokkokból a CY8C4245 PSOC 4 mikrovezérlők 4 db-ot, a CY8C5868, illetve CY8C5888 PSOC 5LP mikrovezérlők 24 db-ot tartalmaznak.



# Egy univerzális adatblokk felépítése

- Az univerzális adatblokk 2 db 12 bemenetű és 4 makrocella kimenetű szorzat-tag előállító áramkört és egy regiszterekkel, illetve ALU egységgel ellátott „adatút” modult tartalmaz.





# Egy 12C4 PLD blokk felépítése

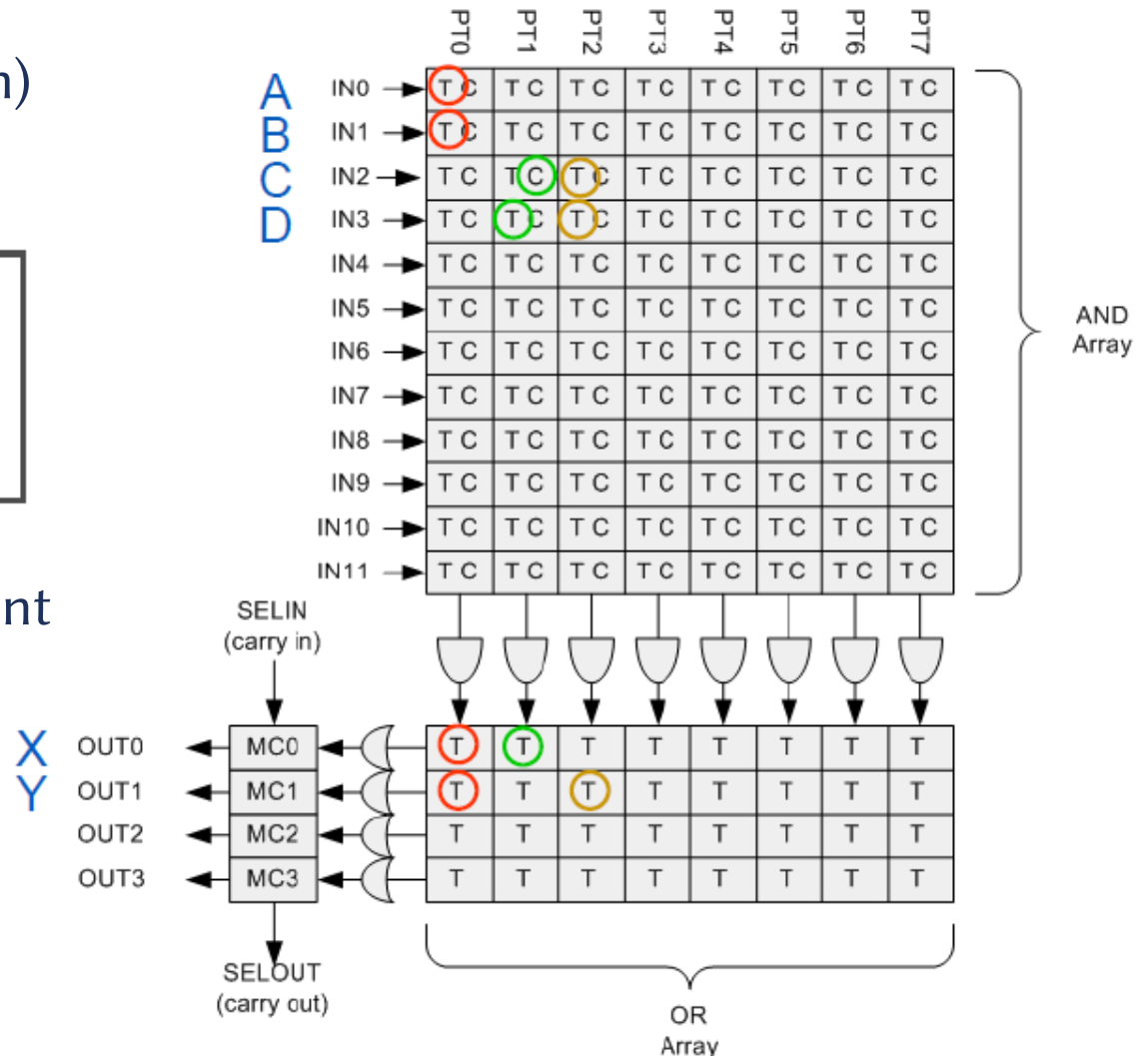
## ■ PLD blokk jellemzők:

- ❖ 12 bemenet
- ❖ 8 szorzat tag (product term)
- ❖ 4 makrocella kimenet

$$X = (A \& B) \mid (\sim C \& D)$$

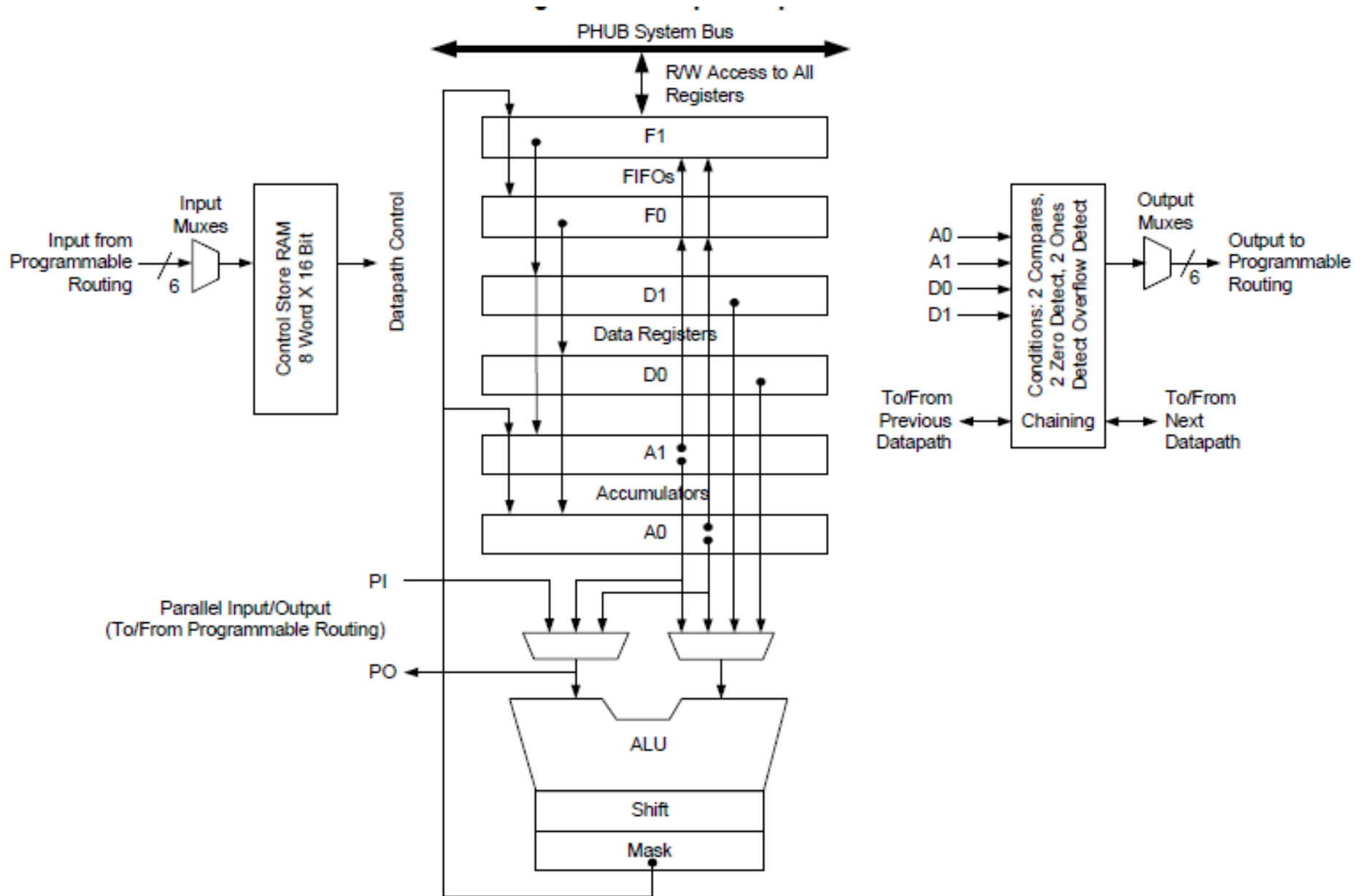
$$Y = (A \& B) \mid (C \& D)$$

- ❖ TC = True vagy Complement
- ❖ AND = logikai ÉS kapuk
- ❖ OR = logika VAGY kapuk
- ❖ MC = makrocella





# Adatút (Data path)



# PSOC 5LP fejlesztőeszközök



## CY8CKIT-050 PSoC® 5LP Development Kit

**CY8C5868AXI-LP035** (100 pin TQFP)  
USB/DMA/12-bit SAR és 20-bit delta-sigma ADC  
24 UDB (digitális blokk), 4 analóg blokk



## CY8CKIT-059 PSoC® 5LP Prototyping Kit

**CY8C5888LTI-LP097** (68 pin QFN)

USB/DMA/12-bit SAR és 20-bit delta-sigma ADC  
24 UDB (digitális blokk), 4 analóg blokk

# CY8CKIT-059 fejlesztői kártya

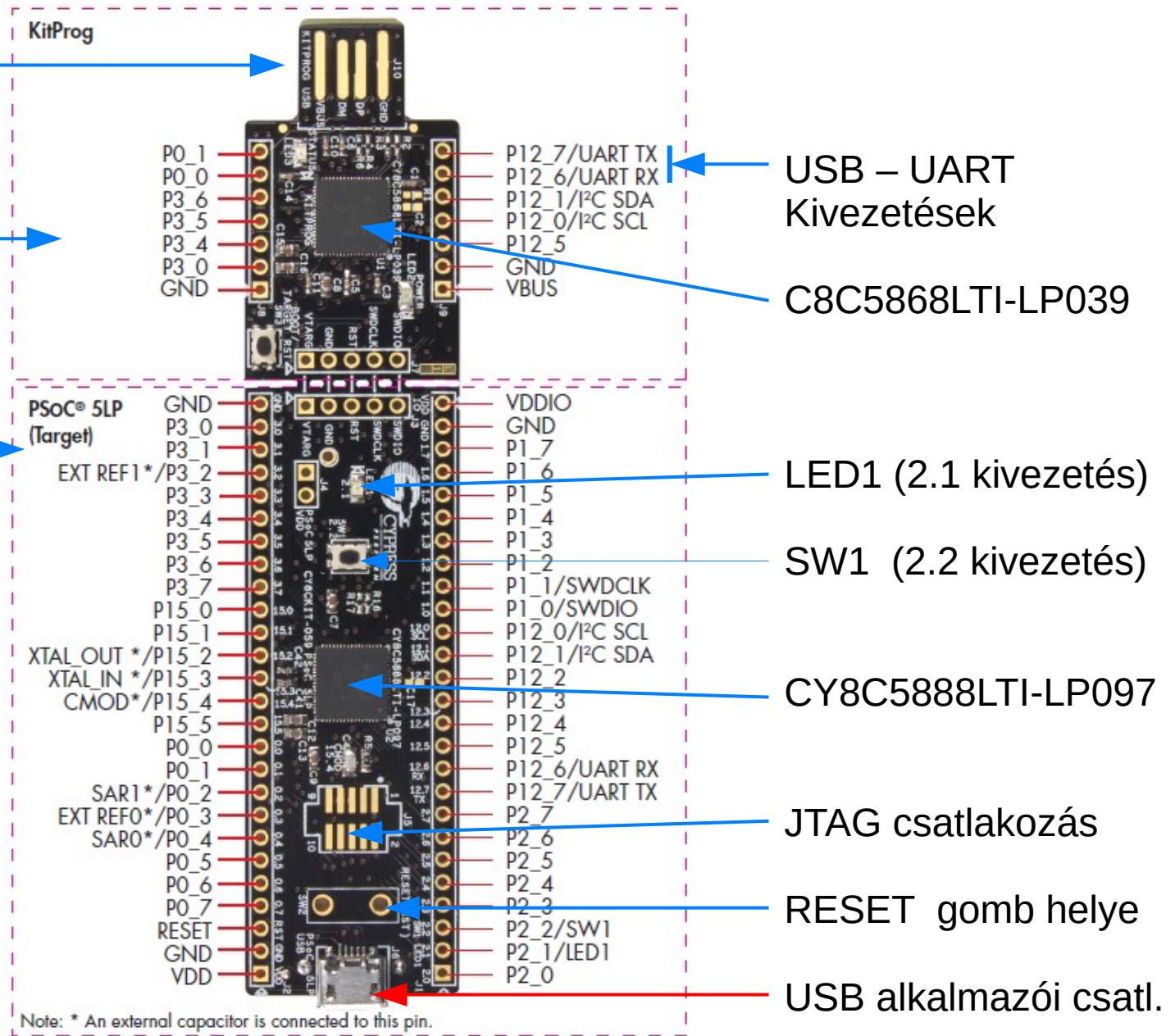
USB csatlakozás  
a PC-hez

KitProg programozó és  
hibavadász

PSOC 5LP  
Target áramkör

A tápellátás történhet a  
programozó felől (5V),  
Az alkalmazói USB  
csatlakozóról (5V),  
vagy a VDD  
csatlakozáson  
keresztül (3,3 – 5 V).

Utóbbi esetben a D1  
és D2 diódákat el kell  
távolítani az USB-re  
csatlakozás előtt!





# Fejlesztői szoftver letöltése és telepítése

---

- Az alábbi szoftverek némelyikének letöltéséhez ingyenes regisztráció szükséges a [www.cypress.com](http://www.cypress.com) webhelyen!
- **PSOC Creator 4.1** (link: [www.cypress.com/psoccreator](http://www.cypress.com/psoccreator))
- **CY8CKIT-059 Kit Only** – Kit tervezői fájlok, Dokumentáció, mintaprojektek (link: [www.cypress.com/cy8ckit-059](http://www.cypress.com/cy8ckit-059))
- **PSOC 5 Programmer**  
(link: [www.cypress.com/products/psoc-programming-solutions](http://www.cypress.com/products/psoc-programming-solutions))
- A fenti szoftverek letöltése és használata ingyenes!

# PSOC Creator indítása, Blinky projekt

1. Projekt Felfedező
2. Tervezőablak
3. Könyvtári modulok listája

The screenshot displays the PSOC Creator 4.1 software interface for the 'Blinky' project. The main workspace shows a schematic diagram of a PWM\_1 component. The component is configured with the following settings:

- Fixed Function (de UDB is lehet)
- 8-bit
- Single Output
- Period = 100 counts (1 sec)
- Compare = 2 counts, Less or Equal

The schematic diagram includes a Kill\_Switch connected to the kill pin of the PWM\_1 component. A Clock\_1 component is connected to the clock pin of the PWM\_1 component. The PWM\_1 component is connected to a PWM\_Out pin, which is connected to an LED1 through a resistor R18 (820 Ohms). The LED1 is connected to Vss.

The Component Catalog on the right shows the PWM [v3.30] component selected. The output window at the bottom shows the log file location: C:\Users\cserny\AppData\I.

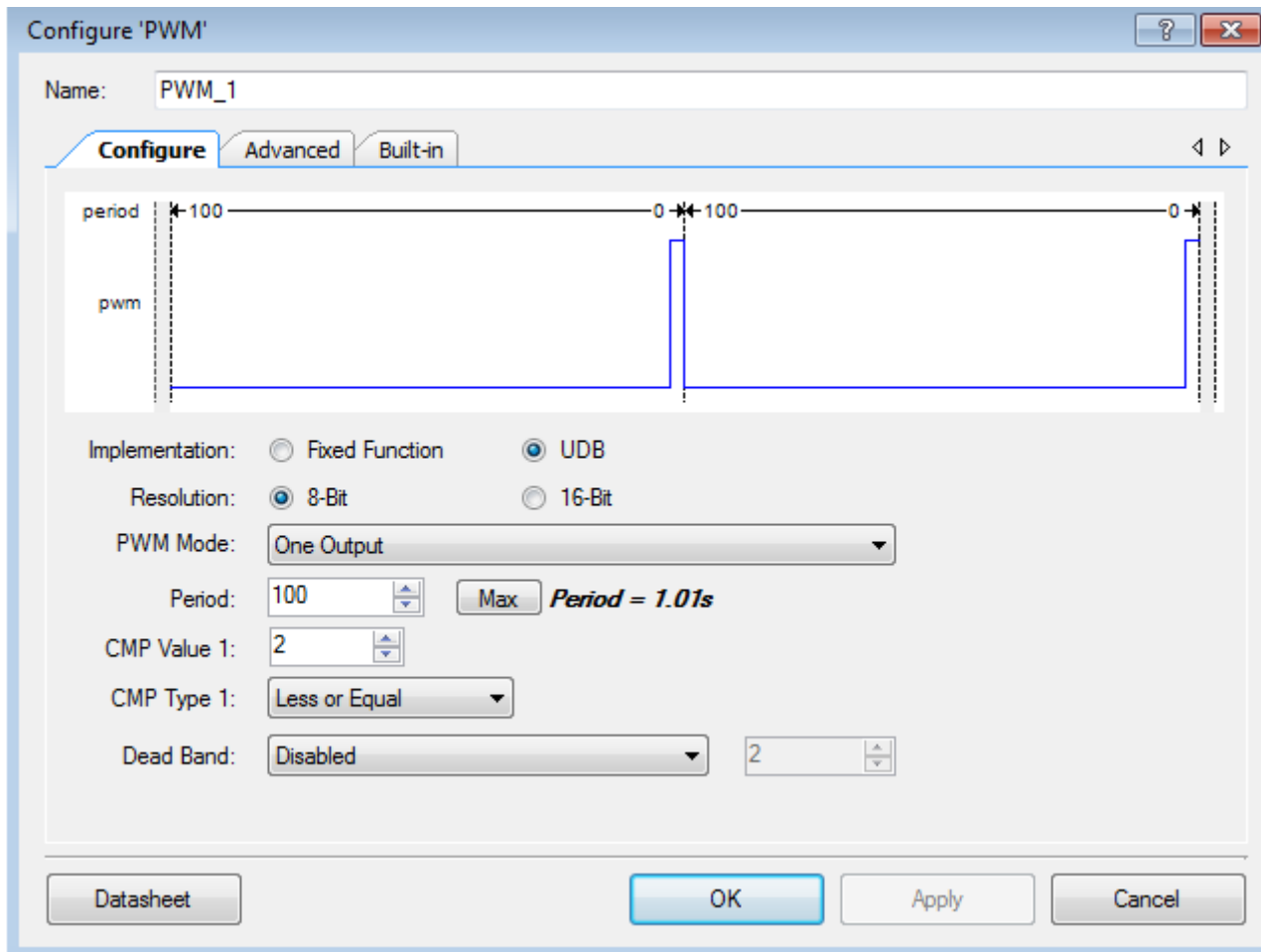
1. 2. 3.

Projekt megnyitás: File → Open → Project/workspace, majd betallózás



# A Blinky projekt konfigurálása

- A CY8CKIT-059 kártya első mintapéldáját (CE95 352 Blinking LED) egy kicsit átszabjuk a PWM\_1 modul átkonfigurálásához. Ehhez duplán kattintsunk az alkatrészeire!



## Beállítások:

1. Fixed function vagy UDB
2. 8-bites mód
3. Egy kimenet
4. A periódus 100 órajel (1 s)
5. A kitöltés 2 órajel
6. Less or Equal mód
7. Holtsáv nem kell

Az Advanced lapon engedélyezzük a Kill funkciót!

# Kivezetések hozzárendelése

- A K1 nyomógomhoz (Kill\_Switch) a P2[2] kiveztés tartozik
- A LED1 kijelzőhöz (PWM Out) a P2[1] kiveztés tartozik

Workspace Explorer (1 project) | Start Page | main.c | Blinky.cydwr | TopDesign.cysch

Workspace 'Blinky' (1 Projects)  
Project 'Blinky' [CY8C5888]  
TopDesign.cysch  
Design Wide Resources  
Pins  
Analog  
Clocks  
Interrupts  
DMA  
System  
Directives  
Flash Security  
EEPROM  
Header Files  
cyapicallbacks.h  
Source Files  
main.c  
Generated\_Source

SWD Debug  
SWD Debug  
SWD Debug  
SWD Debug

Ezeket a programozó lefoglalja

Name	Port	Pin	Value
Kill_Switch	P2[2]	64	
PWM_Out	P2[1]	63	

# A főprogram

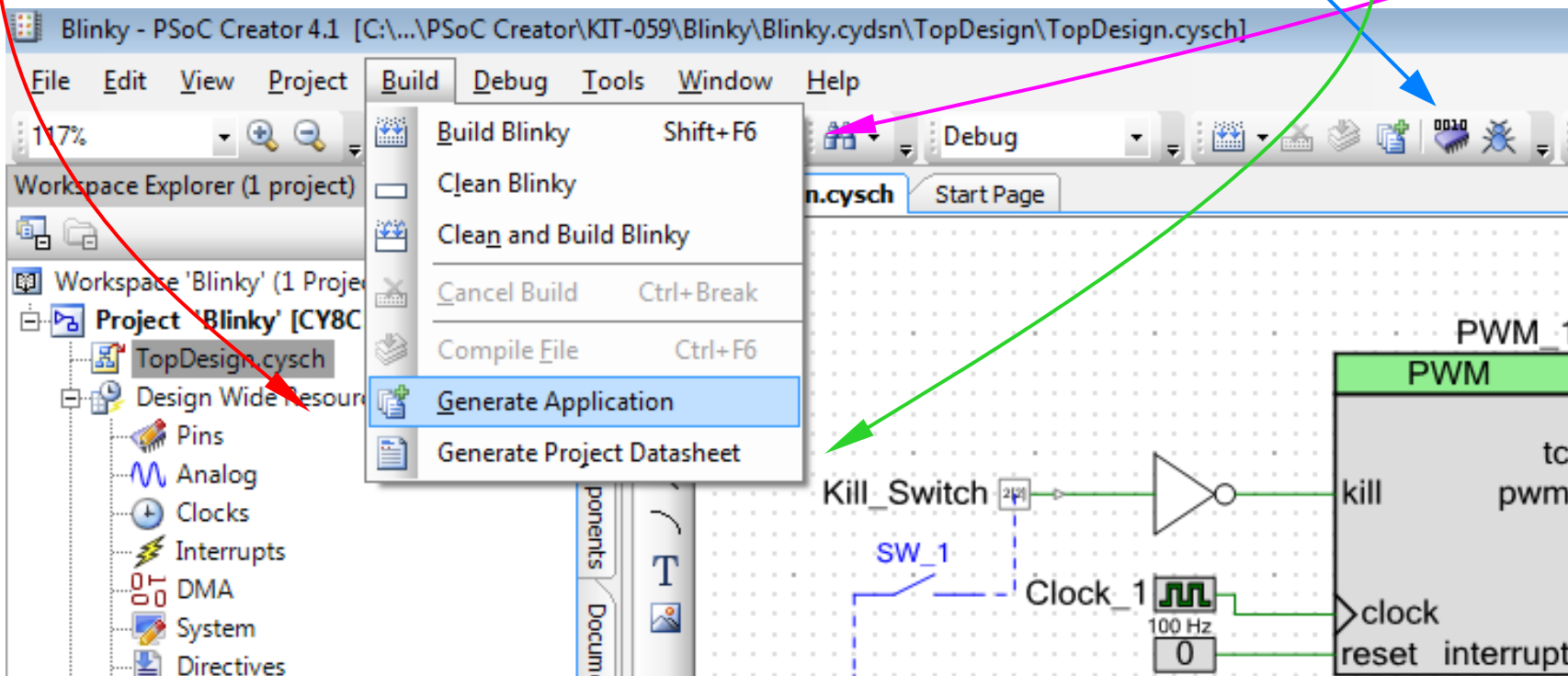
- A PSOC eszközök esetén a hardver működtetéséhez a mikrovezérlőnek is futnia kell (órajelek generálása, modulok engedélyezése, stb.), ezért legalább egy minimális méretű `main.c` programra szükség van.
- A projektbe beillesztett könyvtári alkatrészek/modulok kezelését biztosító API függvényeket a **PSOC Creator** automatikusan beilleszti a projektbe, ezeket hívhatjuk meg a főprogramból

A Blinky projekt `main.c` programjának listája

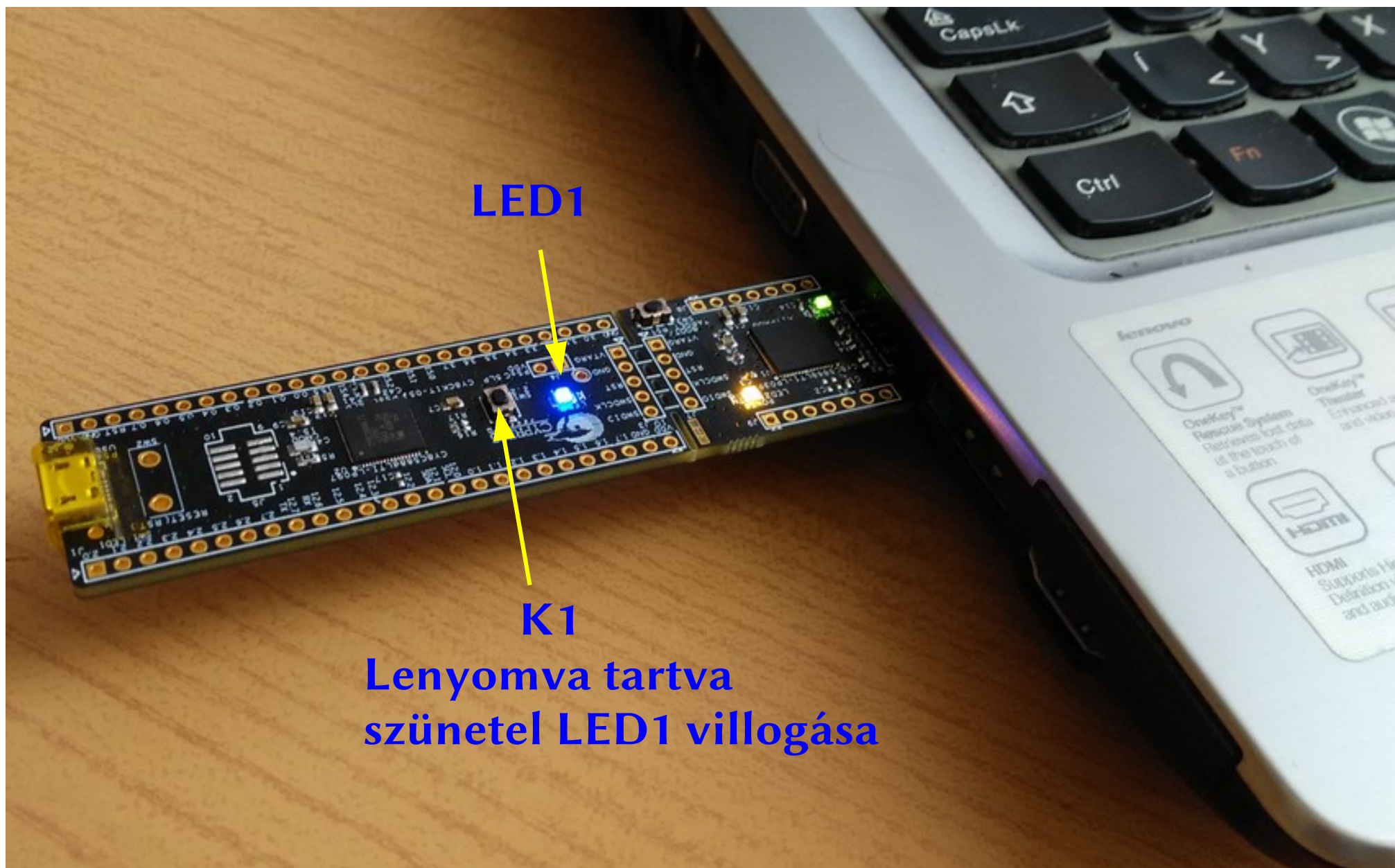
```
#include "project.h"
void main() {
    PWM_1_Start();    // PWM modul indítása
    for (;;) {}
}
```

# Fordítás, programletöltés

- A projekt generálálás, lefordítás és letöltés lépései:
  - 1) Az alkalmazás generálása (API generálás)
  - 2) A projekt lefordítása (build)
  - 3) Dokumentáció generálás (opcionális lépés)
  - 4) A program letöltése (az áramkör csatlakoztatása után)



# A Blinky program futtatása



LED1

K1

Lenyomva tartva  
szünetel LED1 villogása

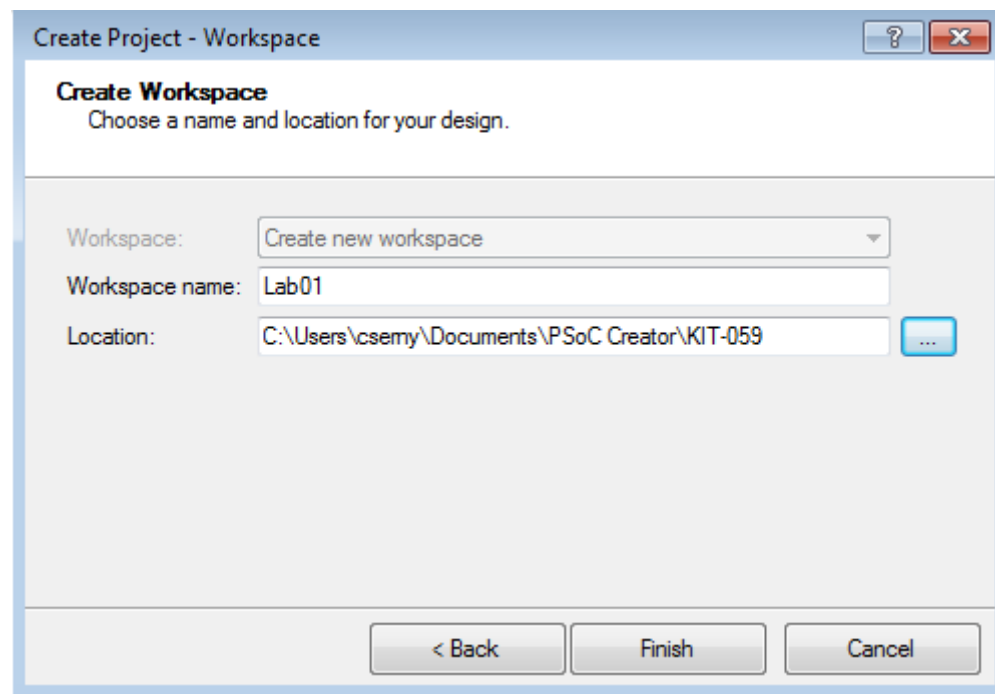
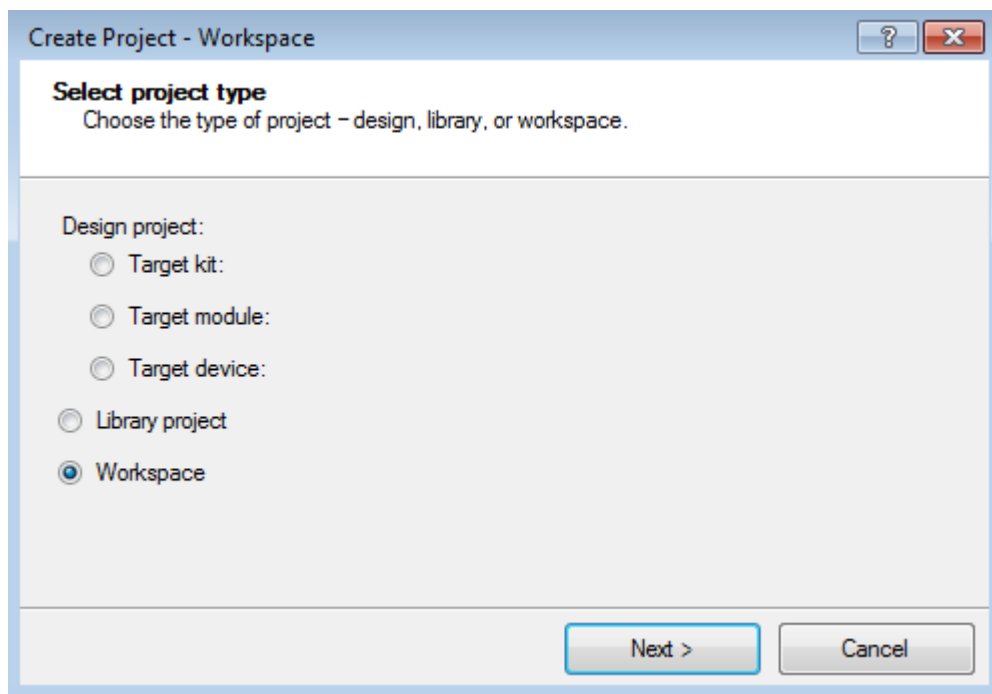
# Új projekt létrehozása lépésről-lépésre

---

- A következőkben egy új **PSOC Creator** projekt létrehozásának menetét mutatjuk be, lépésről-lépésre
- Az új projektünkben a **PSOC 5LP** mikrovezérlő **UART** porton kommunikál a PC-vel, s egy terminálablakból küldött parancsokkal állathatjuk be a **LED1** fényerejét
- Az **UART** kommunikáció a kártyánkra épített **KitProg** eszköz segítségével történik, mivel az **USB-UART** átalakítóként is működik
- A **LED1** teljesítményét egy **PWM** modul segítségével szabályozzuk, s a **PWM** jel kitöltését szabályozhatjuk 5 – 95 % között
- A projekt forrása **Yury Magda** könyvének (Cypress PSoC 5LP Prototyping Kit Measurement Electronics) első mintaprojektje

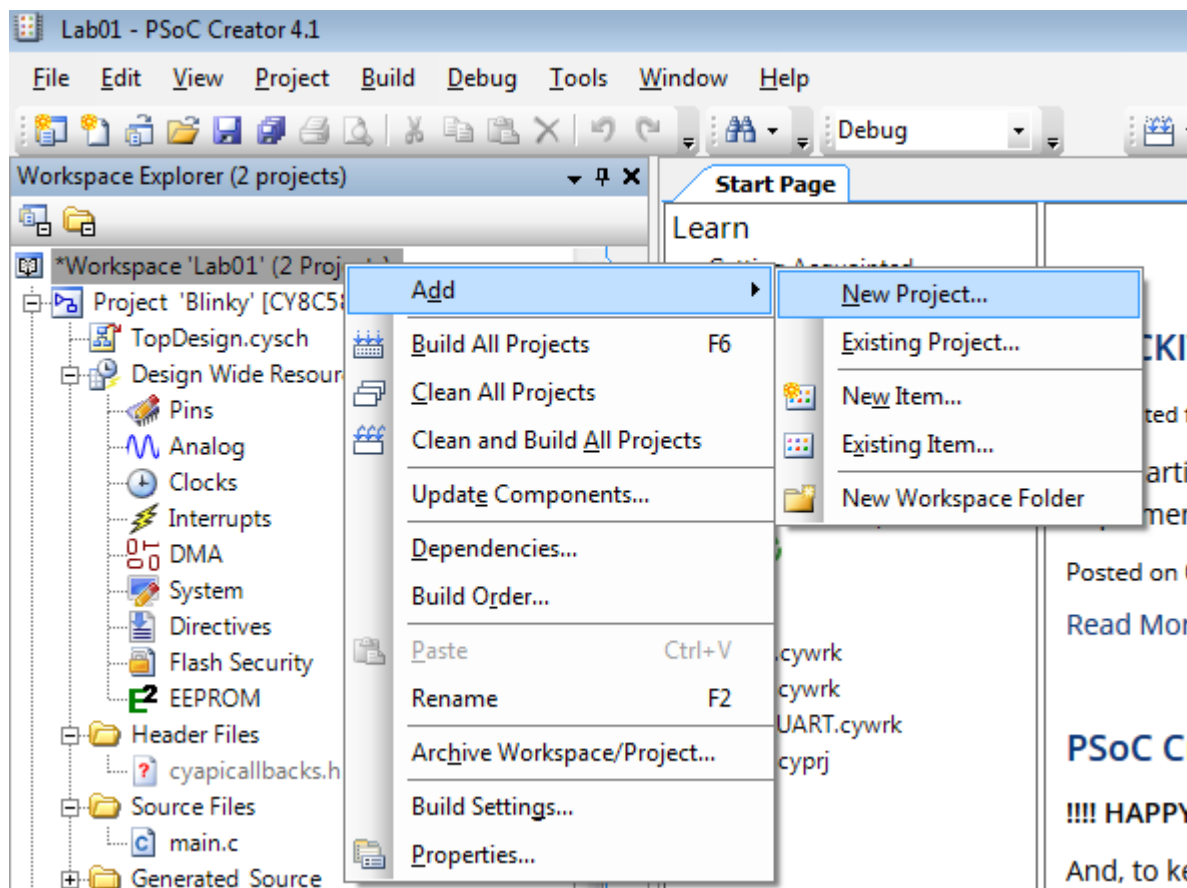
# Új munkaterület létrehozása

- A munkaterület (workspace) egy vagy több projektet tartalmazhat
- Új munkaterület létrehozásának lépései a **PSOC Creator**-ban:
  - ❖ **File** → **New** → **Project** menüpont választás
  - ❖ **Workspace** választás, majd **Next** gombra kattintás
  - ❖ **Munkaterület neve** (pl. **Lab01**) és **hely** megadása, majd **Finish**



# Új projekt létrehozása

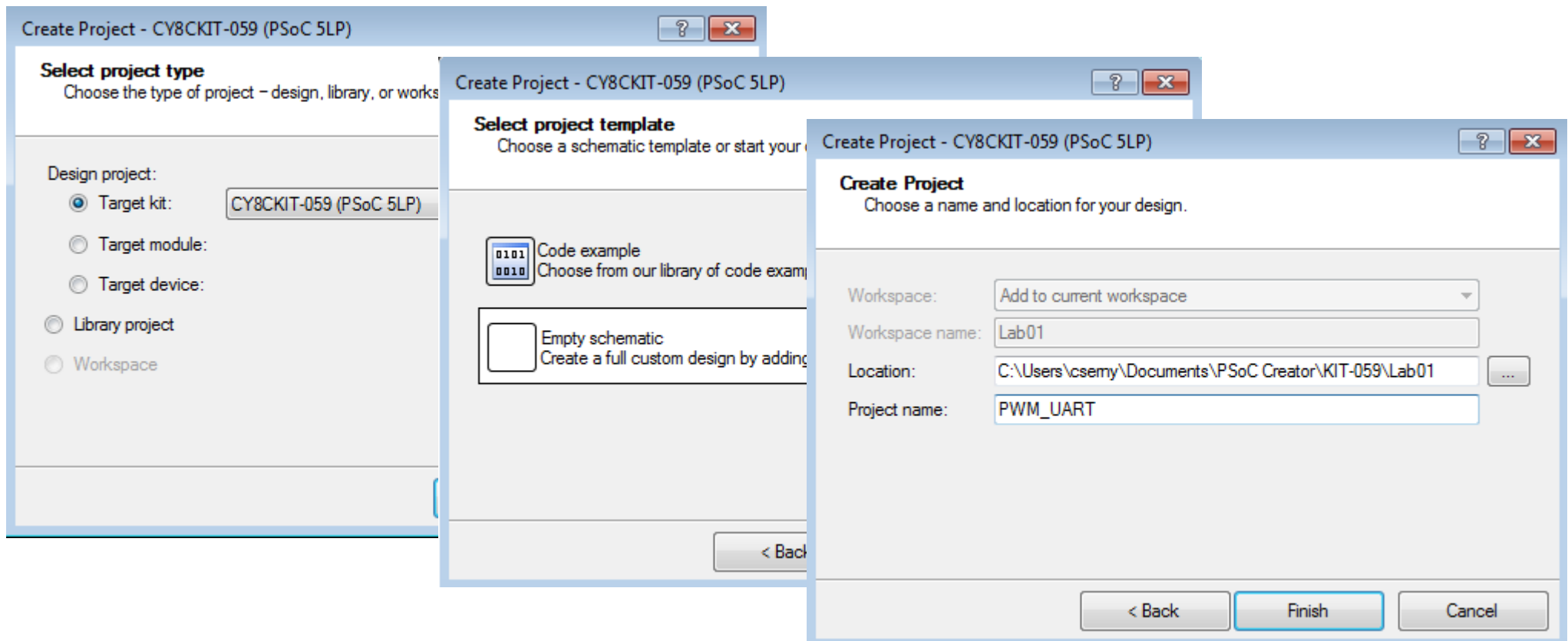
- Jobb gombbal kattintsunk a munkaterület nevére a **Project Explorer** ablakban!
- Válasszuk az **Add → New Project** menüpontot!






# Új projekt létrehozása

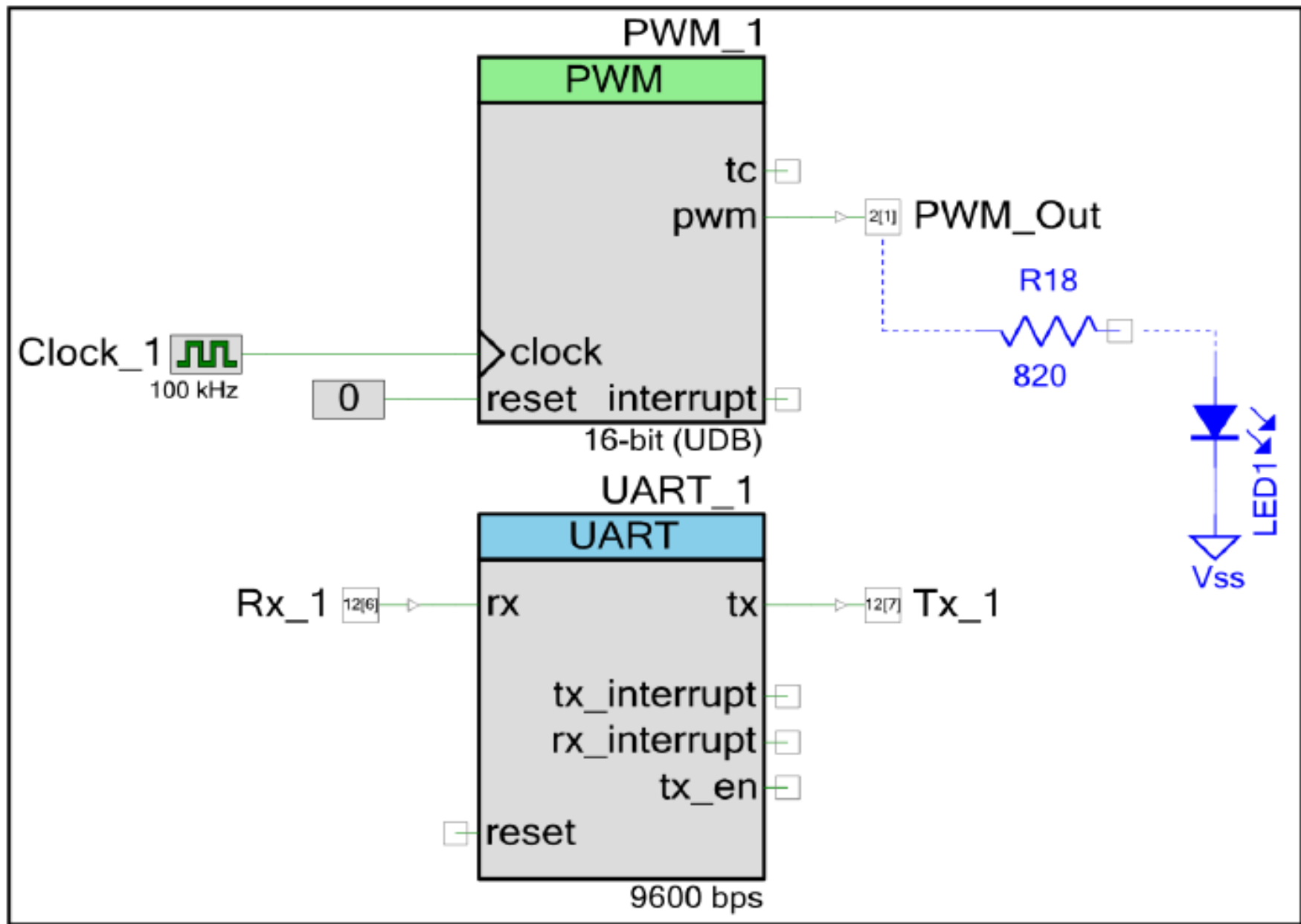
- A felbukkanó ablakban válasszuk a **Target kit** opciót és a **CY8CKIT-059** eszközt, majd kattintsunk a **Next** gombra!
- Válasszuk az Üres (**Empty**) opciót, majd **Next**!
- Adjuk meg a projekt nevét (**PWM\_UART**), majd **Finish**!



# A kapcsolás megrajzolása

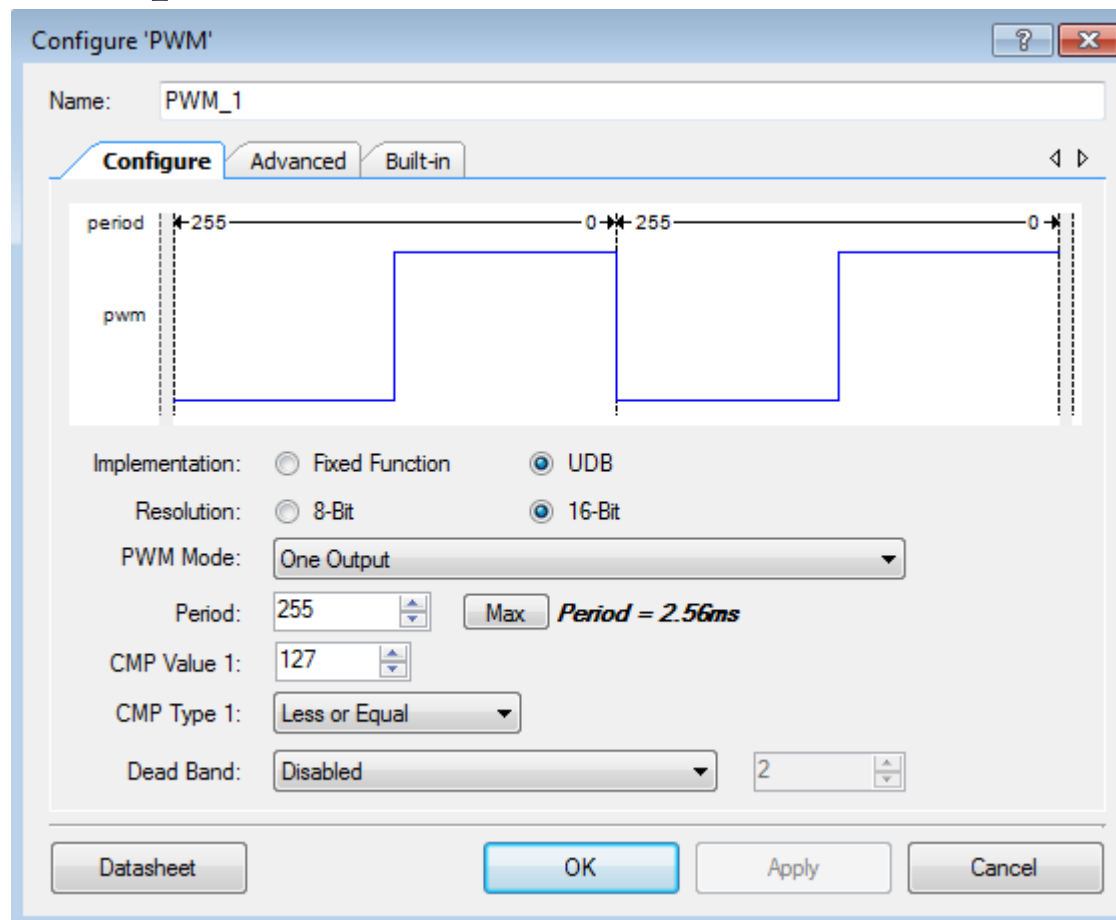
- A kapcsolás összeállítása a **TopDesign.cysch** lapon történik
- Az alkatrészeket a jobboldali menüből tallózhatjuk és húzhatjuk rá a rajzfelületre
  - ❖ **PWM\_1**: Cypress → Digital → Functions → PWM [v3.30]
  - ❖ **UART\_1**: Cypress → Communications → UART [v2.50]
  - ❖ **Clock\_1**: Cypress → System → Clock [v2.20]
  - ❖ **0**: Cypress → Digital → Logic → Logic Low "0" [v1.0]
  - ❖ **PWM\_Out**: Cypress → Ports and Pins → Digital output pin [v2.20]
- Az összekötésekhez a  eszközt használjuk!
- A mikrovezérlőhöz kapcsolódó külső áramköri elemeket az **Off-Chip** alkatrészkönyvtárból keressük ki! Ezeknek az elemeknek nincs hatása a projektépítésre illetve annak működésére, csupán dokumentációs célokat szolgál a tervrajz ilyen kiegészítése.

# Az áramköri terv



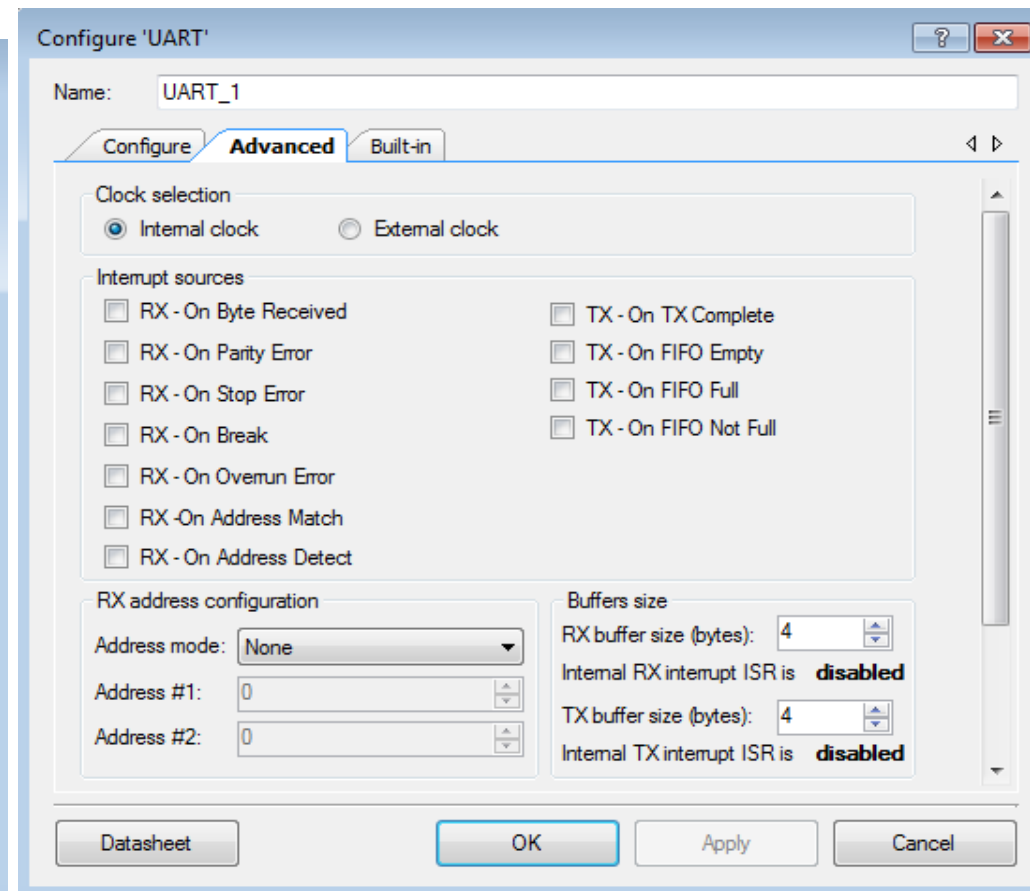
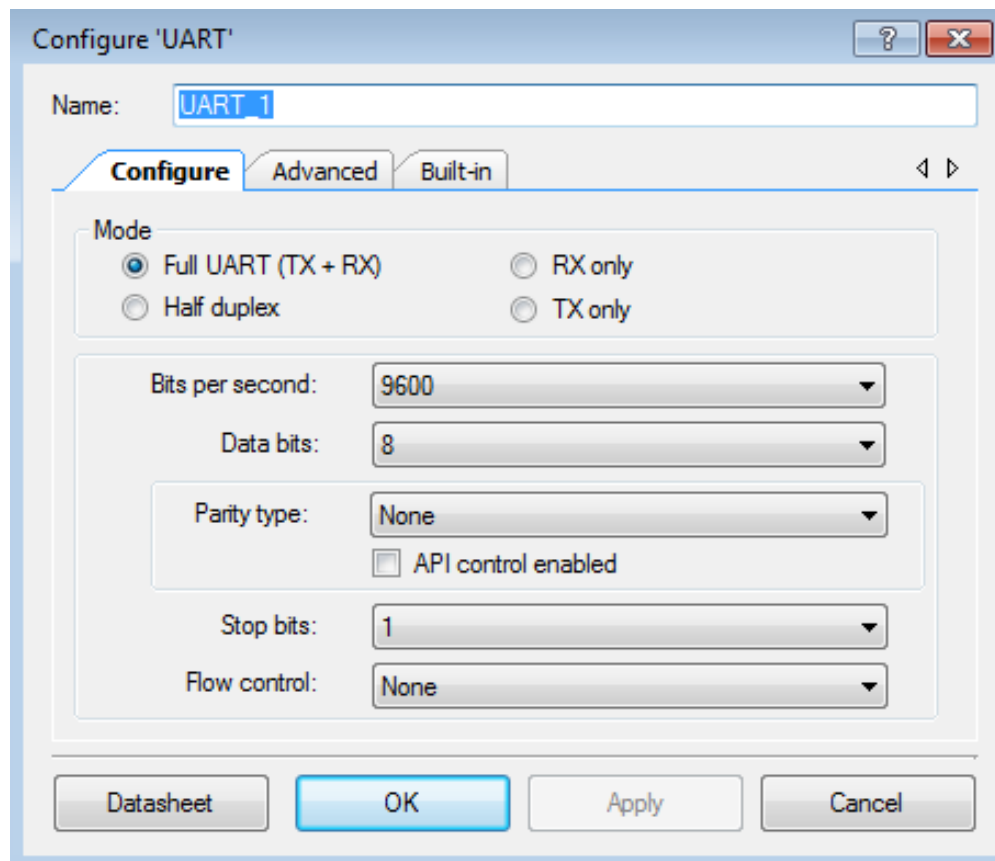
# PWM\_1 konfigurálása

- Duplakattintással nyithatjuk meg a kiválasztott alkatrész konfigurációs felületét
- **PWM\_1**: UDB, 16-bit mód, periódus=255, kitöltés=127, polaritás: Less or Equal, nincs holtsáv



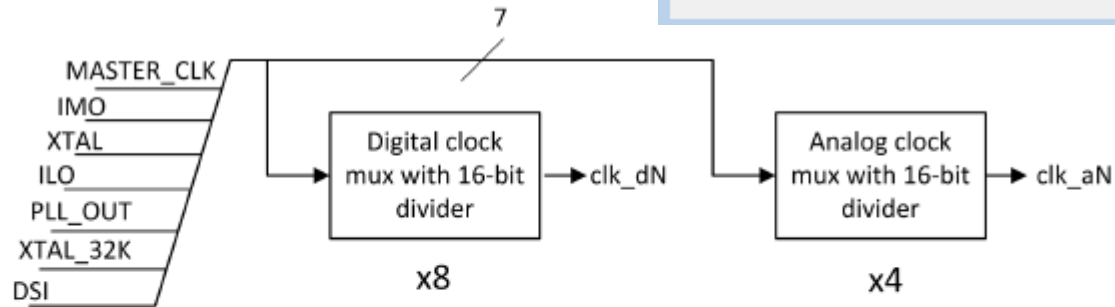
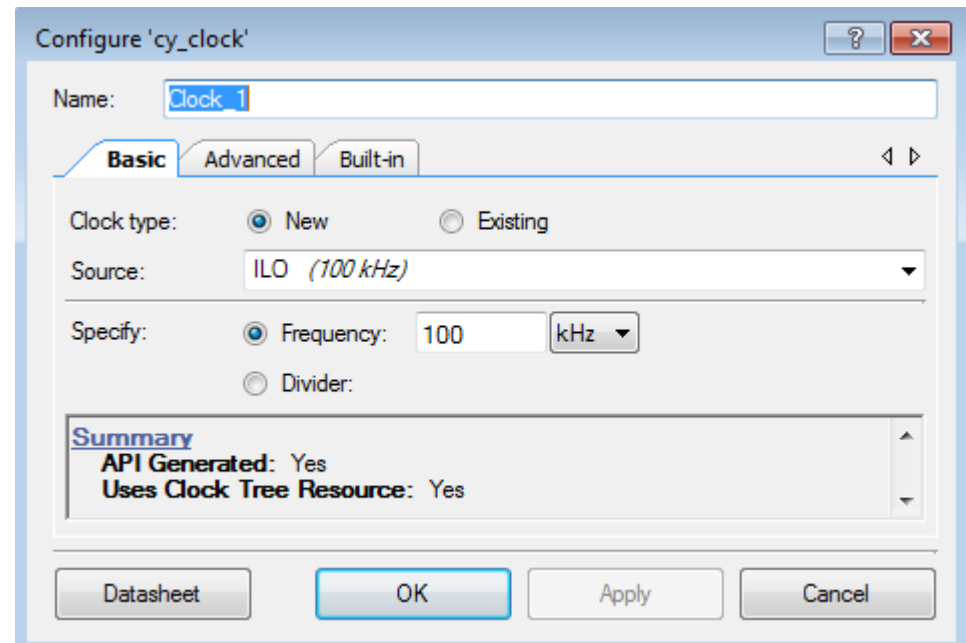
# UART modul konfigurálása

- Az **UART\_1** modullal full duplex kommunikációt valósítunk meg a **KitProg**-on keresztül ami **USB-UART** átalakítóként is szolgál
- Megzsakítást nem használunk, a sebesség 9600 bps, a formátum pedig 8 bit, 1 stop bit, no parity, és nincs adatfolyam-vezérlés



# CLOCK\_1 konfigurálása

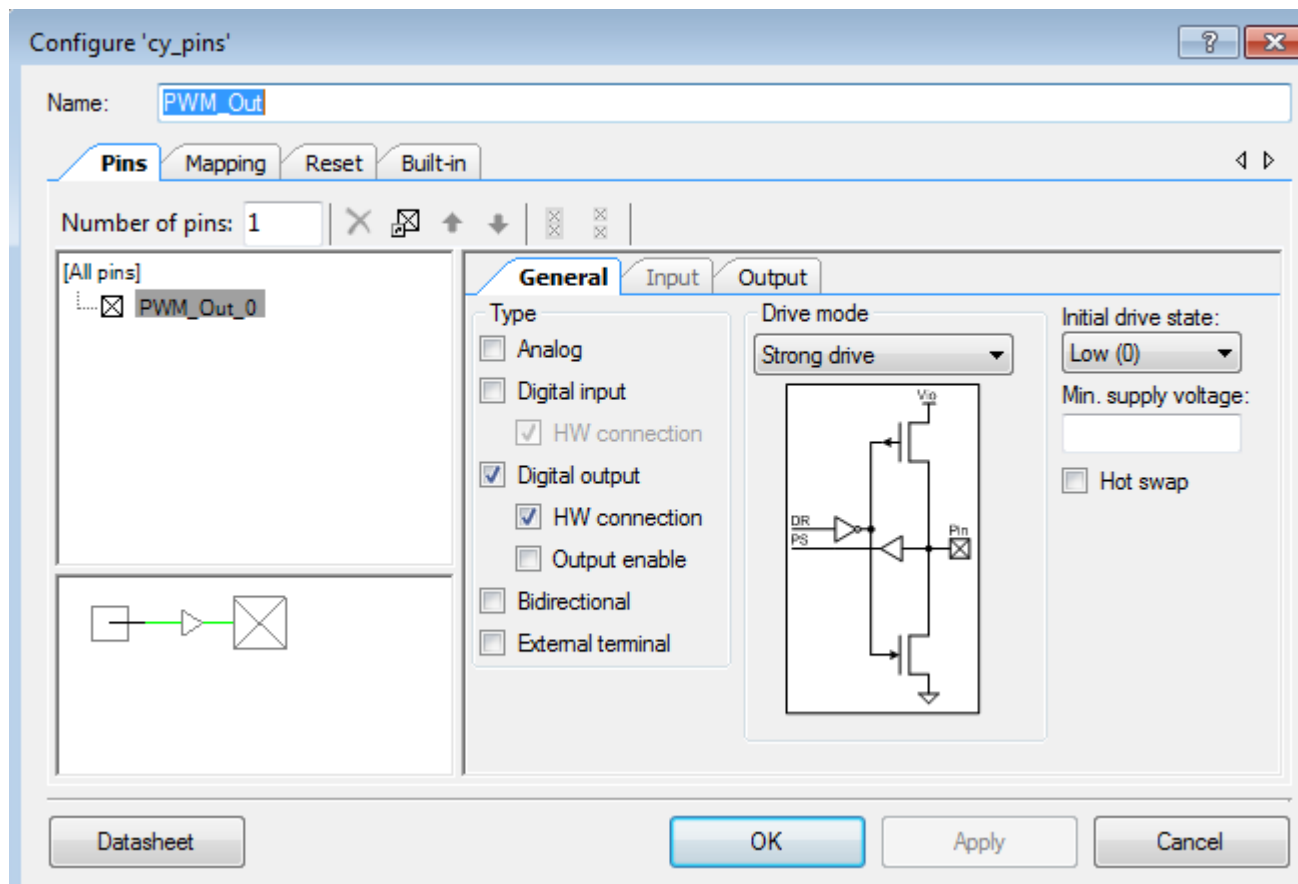
- A rendszer órajeleiből leosztással további órajeleket képezhetünk
- A PWM\_1 modulhoz 100 kHz-es órajelet konfigurálunk
- Azt UART\_1 órajele automatikusan generálódik az adatsebesség megadásakor



Name	Domain	Source	Desired Freq	Nominal Freq	Accuracy (%)	Start at Reset	Enabled
Clock_1	DIGITAL	ILO	100 kHz	100 kHz	-55,+100	True	True
UART_1_- IntClock	DIGITAL	MASTER_CLK	76.8 kHz	76.677 kHz	±1	True	True

# A PWM\_Out kivezetés konfigurálása

- A PWM-Out kimenettel LED1-et akarjuk meghajtani, ezért „erős meghajtóként” (Strong drive) konfiguráljuk
- A földre kötött LED1-et így legfeljebb 4 mA-rel táplálhatjuk



# A kivezetések hozzárendelése

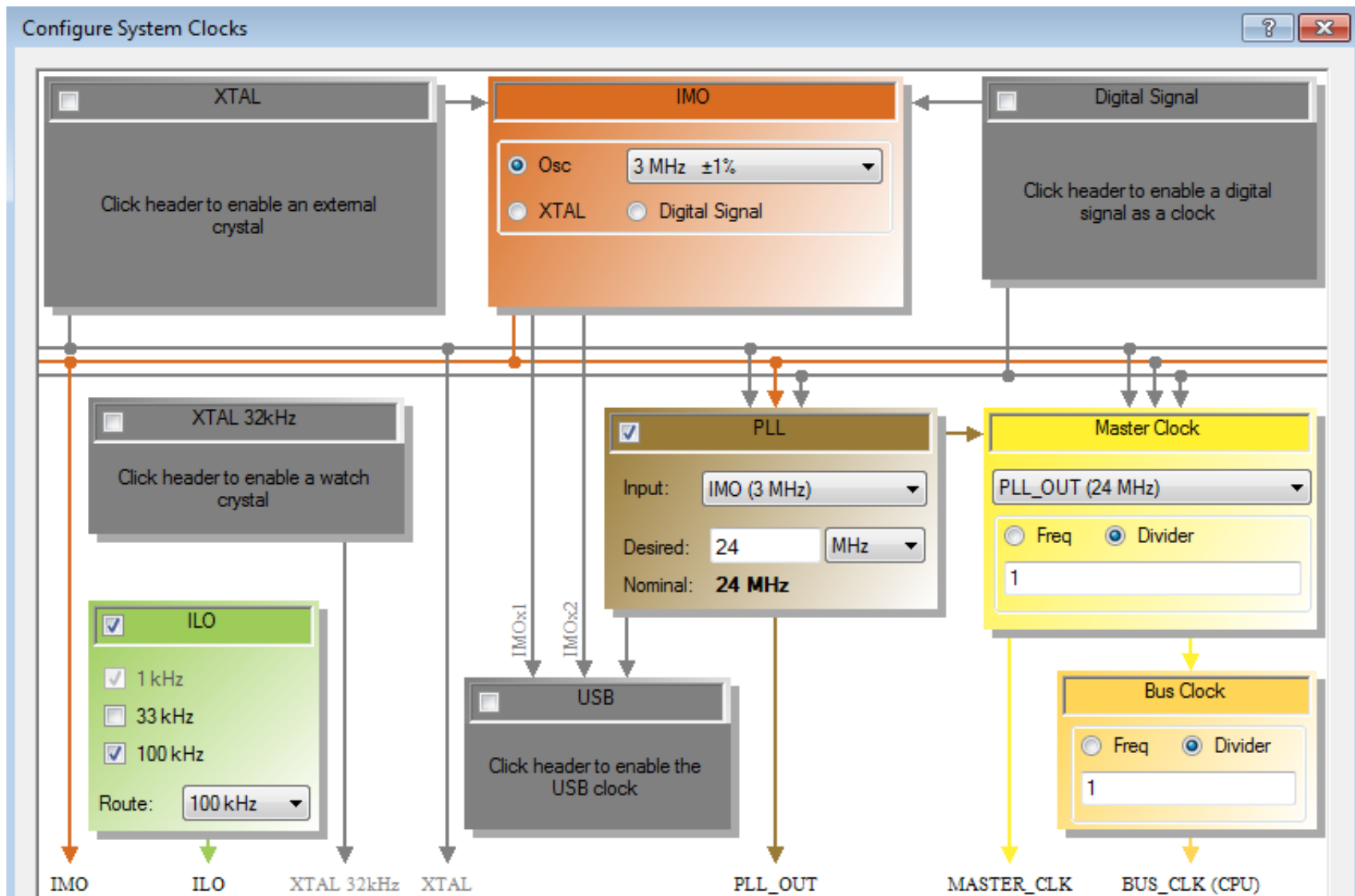
- Mivel a külső áramkör kialakítása már adott számunkra, nekünk kell megadni, hogy mi melyik lábón legyen kivezelve

The screenshot shows the Pin Manager interface for the CY8C5888LTI-LP097 68-QFN chip. The chip pinout is displayed with various pins labeled. On the right, a table lists the assigned pins for PWM\_Out, Rx\_1, and Tx\_1.

Name /	Port	Pin	Lock
PWM_Out	P2 [1]	63	<input checked="" type="checkbox"/>
Rx_1	P12 [6]	20	<input checked="" type="checkbox"/>
Tx_1	P12 [7]	21	<input checked="" type="checkbox"/>



# Az órajelek konfigurálása



A „Design Wide Resources” Clock lapján a rendszer órajeleket konfigurálhatjuk (Edit Clock...-ra kattintva). Itt most ILO állítha elő a 100 kHz-es órajelet. A CPU 24 MHz-en fut.

# A main.c program

```
#include <project.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
uint8 rxState; char buf[32]; char *pbuf = buf; int cmp;
int main() {
    CyGlobalIntEnable;           // Megszakítások engedélyezése
    UART_1_Start();             // UART_1 egység indítása
    PWM_1_Start();              // PWM_1 egység indítása
    for(;;) {
        pbuf = buf;
        UART_1_PutString("Enter duty cycle (5 - 95, max. 2 digits): ");
        UART_1_PutCRLF(0xD);
        UART_1_ClearRxBuffer();
        memset(buf, 0, strlen(buf));
        while (1) {
            if((rxState = UART_1_ReadRxStatus()) == UART_1_RX_STS_FIFO_NOTEMPTY) {
                *pbuf = UART_1_GetChar();
                if((isdigit((uint8)*pbuf) == 0) || (*pbuf == '\n')) {
                    UART_1_ClearRxBuffer(); break; }
                UART_1_PutChar(*pbuf); UART_1_ClearRxBuffer(); pbuf++; }
        }
        UART_1_PutCRLF(0xD);
        CyDelay(500); cmp = atoi(buf);
        if ((cmp >= 5) && (cmp <= 95))
            PWM_1_WriteCompare((int)(cmp*2.56));
    }
}
```

Forrás: Yuri Magda: Cypress PSoC 5LP Prototyping Kit Measurement Electronics

# A projekt lefordítása és futtatása

- A projektépítés munkamenete megegyezik a korábban leírtakkal (Build → Generate Application, Build → Build PWM\_UART)
- A futtatáshoz használhatjuk pl. a **Termite** programot, 9600 bps-re konfigurálva. A beírt számok 5 és 95 közöttiek lehetnek!

