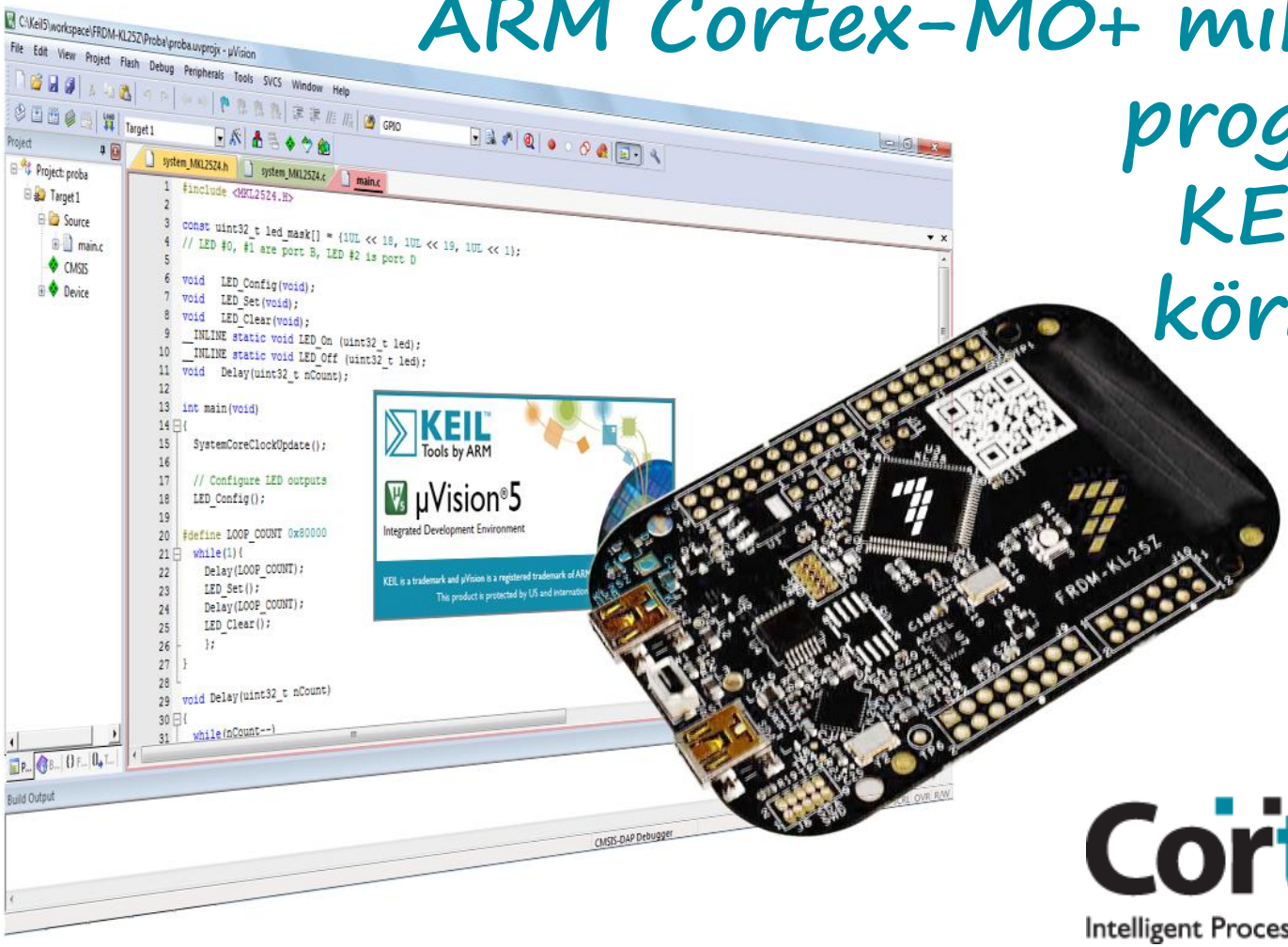
ARM Cortex-M0+ mikrovezérlő programozása KEIL MDK 5 környezetben



Cortex
Intelligent Processors by ARM

5. Időzítők, számlálók – 1. rész

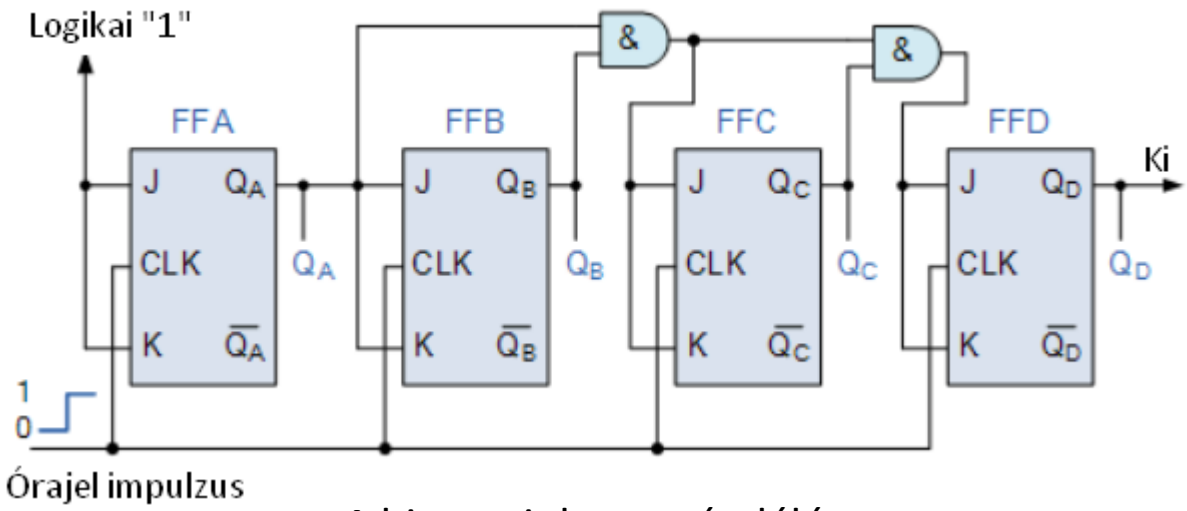
Felhasznált anyagok, ajánlott irodalom

- Joseph Yiu: **The Definitive Guide to ARM® Cortex®-M0 and Cortex-M0+ Processors** (2nd Ed.)
- Muhammad Ali Mazidi, Shujen Chen, Sarmad Naimi, Sepehr Naimi: **Freescale ARM Cortex-M Embedded Programming**
- ARM University Program: **Course/Lab Material for Teaching Embedded Systems/MCUs** (for the FRDM-KL25Z board)
- Trevor Martin: **The Designer's Guide to the Cortex-M Processor Family**
- Electronics tutorials: **Counters**
- Freescale: **MKL25Z128VLK4 MCU datasheet**
- Freescale: **KL25 Sub-Family Reference Manual**
- Freescale: **FRDM-KL25Z User Manual**

Számlálók és időzítők

- **A számláló** (counter) olyan digitális áramkör, amellyel feszültségimpulzusokat tudunk leszámlálni.
- **Időzítőről** (timer) akkor beszélünk, ha a leszámlálandó impulzusok nem külső forrásból származnak, hanem ismert, állandó frekvenciájú jelet vezetünk a számláló bemenetére

Számláló megvalósítása: sorbakötött billenőáramkörökkel. Egy n-bites számláló 2^{n-1} állapotot képes felvenni. Pl. egy 4 bites számláló 0000-tól 1111-ig, azaz 0-tól 15-ig számol.

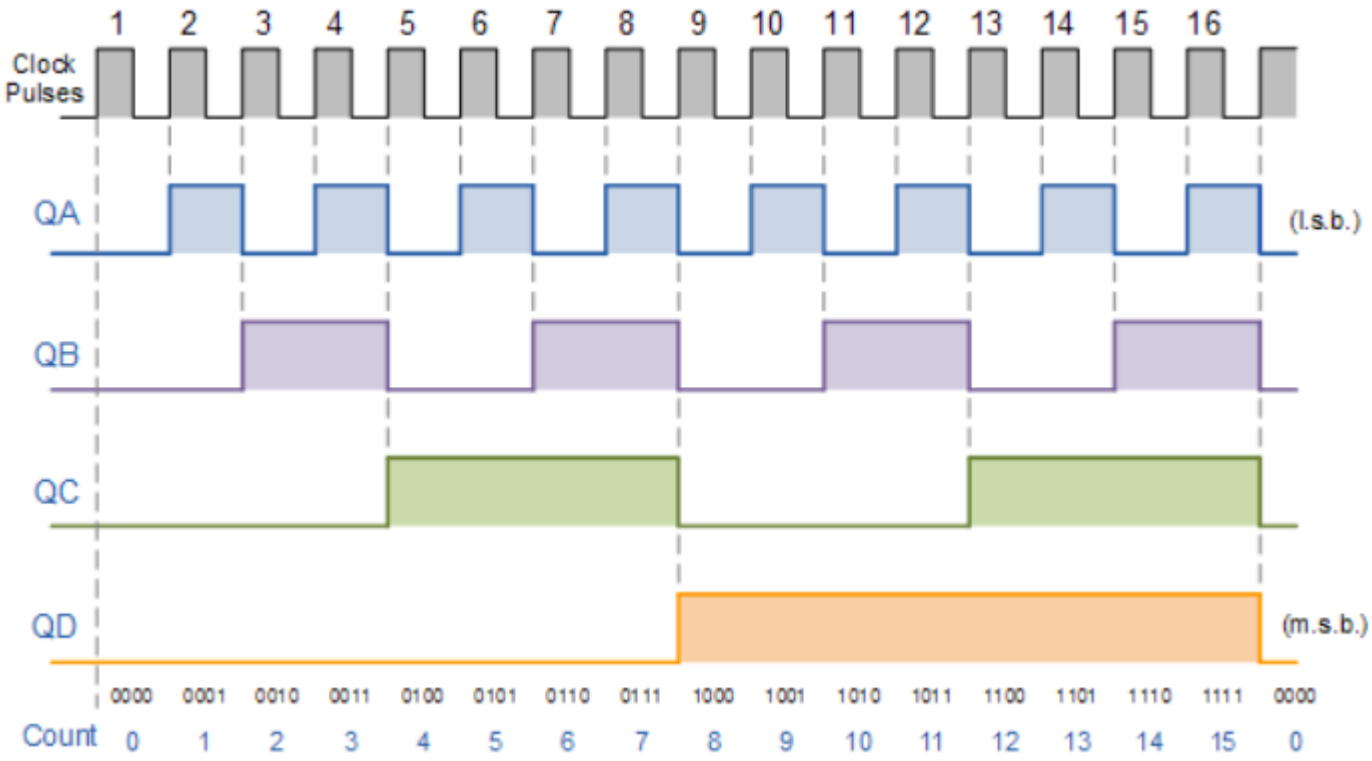


4-bites szinkron számláló.

Forrás: http://www.electronics-tutorials.ws/counter/count_3.html

A mikrovezérlőknél különösen fontos a központi órajellel szinkronizált működés.

A 4-bites számláló jelalakja



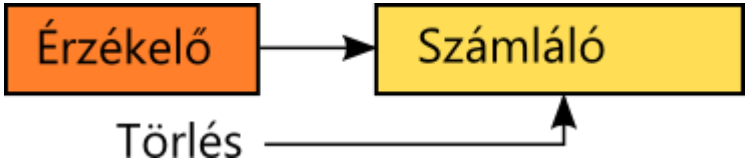
Forrás: http://www.electronics-tutorials.ws/counter/count_3.html

Számlálók felhasználási területei

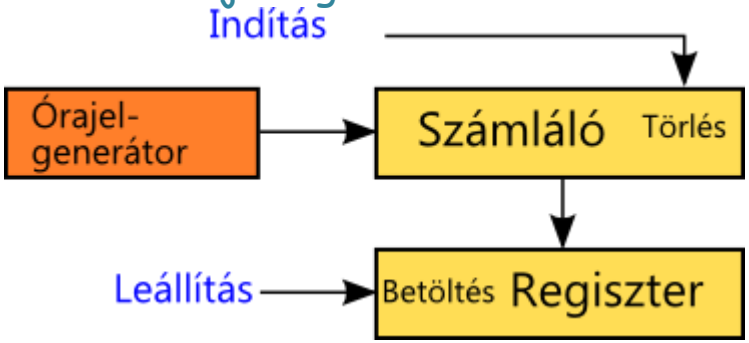
A beágyazott rendszerekben a számlálókat többféle célra is használhatjuk. Néhány tipikus felhasználási terület:

- Események számlálása
- Késleltetések előállítása (a számlálót időzítőként használjuk)
- Két esemény bekövetkezte között eltelt idő mérése
- Impulzusszélesség-moduláció vezérlése
- Események triggerelése (ADC konverzió vagy DMA átvitel indítása)
- Frekvencia osztók (pl. UART, SPI, I2C adatsebesség beállítása)

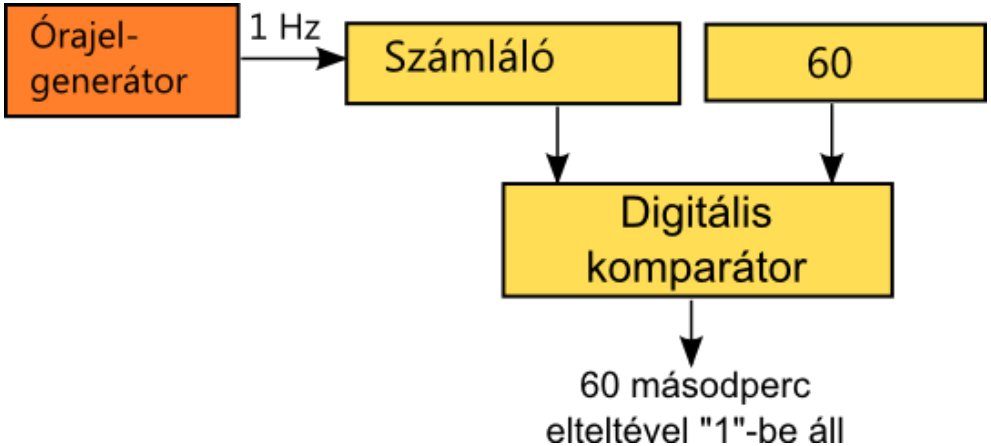
Eseményszámláló



Bemeneti jelrögzítés



Időzítőként használt számláló

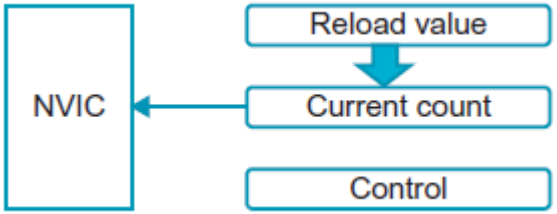


Időzítők/számlálók az MKL25Z128VLK4 mikrovezérlőben

- **Systick** 24-bites (vissza)számláló, az **ARM Cortex-M** mikrovezérlő mag része. Többnyire az operációs rendszerek (pl. RTOS) használják a feladatok ütemezéséhez.
- **TPM0, TPM1, TPM2** általános célú 16-bites számlálók/időzítők, amelyekhez 2-6 db impulzus-szélesség modulációra (PWM), bemeneti jelrögzítésre (Input Capture), vagy digitális komparálásra (Output Compare használható csatorna is tartozik.
- **Periodic Interrupt Timer (PIT)** 2 x 32 bites időzítővel rendelkezik, amelyek egy 64 bites időzítővé is összekapcsolhatók. Periodikus megszakítások előállítására szolgál, de más célra, pl. a legutolsó RESET óta eltelt folyó időt mérésére is hasznnálhatjuk.
- **Low-Power Timer (LPTMR)** kisfogyasztású 16-bites időzítő/számláló. Mindegyik energiatakarékos módban üzemképes, külső események számlálására vagy belső órajellel időzítésre, vagy programozott ébresztésre használható.
- **Real-Time Clock (RTC)** modul, melyben található egy 32-bites másodperc számláló, egy 16-bites előszámláló, egy 16-bites időkompenzációs regiszter és egy 32-bites regiszter az ébresztés beállítására. A **FRDM-KL25Z** kártya esetén a használatot némiképp korlátozza, hogy az RTC órajelét az **OpenSDA** áramkör biztosítja, s az **MKL25Z128VLK4** mikrovezérlő **PTC1** kivezetését (**RTC_CLKIN**) lefoglalja.

A SysTick időzítő

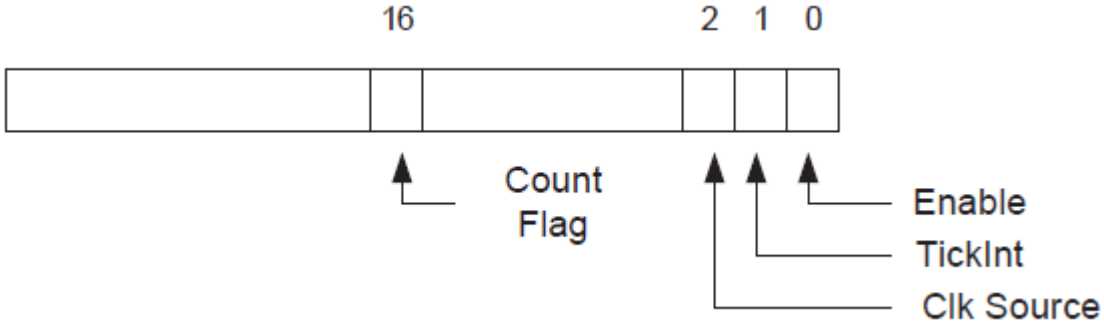
A **SysTick** időzítő egy 24 bites visszafelé számlálót tartalmaz, amely nullára futáskor automatikusan újratöltődik (a töltési érték programozható), s egy megszakítást generál.
A **SysTick** megszakítások természetesen letilthatók, s lekérdezéses módban is vizsgálhatjuk a számláló állapotát – ahogy ebben a fejezetben mi is tesszük.



A SysTick időzítő regiszterkészlete:

Cím	Regiszternév	CMSIS név	A regiszter funkciója
0xE000_E010	SYST_CSR	**SysTick->CTRL**	Vezérlő és állapotjelző regiszter
0xE000_E014	SYST_RVR	**SysTick->LOAD**	Újratöltési érték regiszter
0xE000_E018	SYST_CVR	**SysTick->VAL**	Számláló pillanantnyi értéke
0xE000_E01C	SYST_CALIB	**SysTick->CALIB**	Kalibrációs regiszter

A **SysTick_CTRL** vezérlő és állapotjelző regiszter bitjeinek kiosztása:



Program5_1: SysTick konfigurálása

A **SysTick** számlálót szabadonfutó módba állítjuk, s a periodikusan kiolvasott számláló értékét 4 bináris helyiértékkel jobbra léptetjük, és a B portra kiírjuk.
A számláló 23. bitje a **PTB19** (zöld LED), a 22. bit pedig a **PTB18** (piros LED) lábat vezérli.

```
#include <MKL25Z4.H>

int main (void) {
  int c;
  //--- A GPIOB port konfigurálása ----
  SIM->SCGC5 |= 0x0400;          // Port B engedélyezése (bit 10)
  PORTB->PCR[18] = 0x100;        // PTB18 legyen GPIO (MUX = 1)
  PORTB->PCR[19] = 0x100;        // PTB19 legyen GPIO (MUX = 1)
  PTB->PDDR |= 0xC0000;          // PTB18 és PTB19 legyen kimenet (bit18 és 19=1)

  //--- SysTick konfigurálása ---------
  SysTick->LOAD = 0xFFFFFF;      // Újratöltés érték legyen a maximális
  SysTick->VAL  = 0;             // Kezdéshez alaphelyzetbe állunk
  SysTick->CTRL = 5;             // Engedélyezés, nincs interrupt, rendszer órajel

  while (1) {
    c = SysTick->VAL;            // Számláló kiolvasása
    PTB->PDOR = c >> 4;          // legfelső két bit kijelzése
  }
}
```

Program5_2: 250 ms késleltetés

Ebben a programban 250 ms késleltetésnek megfelelő számú számlálási ciklust végeztetünk a **SysTick** számlálóval, s figyeljük a **COUNT** jelzőbit bebillenését. A késleltetés leteltével átbillentjük a **PTB18**-ra kötött piros LED állapotát.

Újratöltési érték = CPU freq. * időtartam = 20 970 000 1/s * 0,25 s = 5 242 500

A 24 bites újratöltési érték nem lehet nagyobb 0xFFFFFF-nél, azaz 16 777 215-nél!

```
#include <MKL25Z4.H>

int main (void) {
  int c;
  //--- A GPIOB port konfigurálása ----
  SIM->SCGC5 |= 0x0400;           // Port B engedélyezése (bit 10)
  PORTB->PCR[18] = 0x100;         // PTB18 legyen GPIO (MUX = 1)
  PTB->PDDR |= 0x40000;           // PTB18 legyen kimenet (bit18)
  //--- SysTick konfigurálása ---------
  SysTick->LOAD = 5242500;        // Újratöltés érték 250 ms-hoz
  SysTick->VAL  = 0;              // Kezdéshez alaphelyzetbe állunk
  SysTick->CTRL = 5;              // Engedélyezés, nincs interrupt, rendszer órajel
  while (1) {
    if(SysTick->CTRL & 0x10000)   // Ha a COUNT jelző bebillent...
      PTB->PTOR = 0x040000;       // billentsük át a piros LED állapotát!
  }
}
```

Program5_3: delayMs() újratervezése

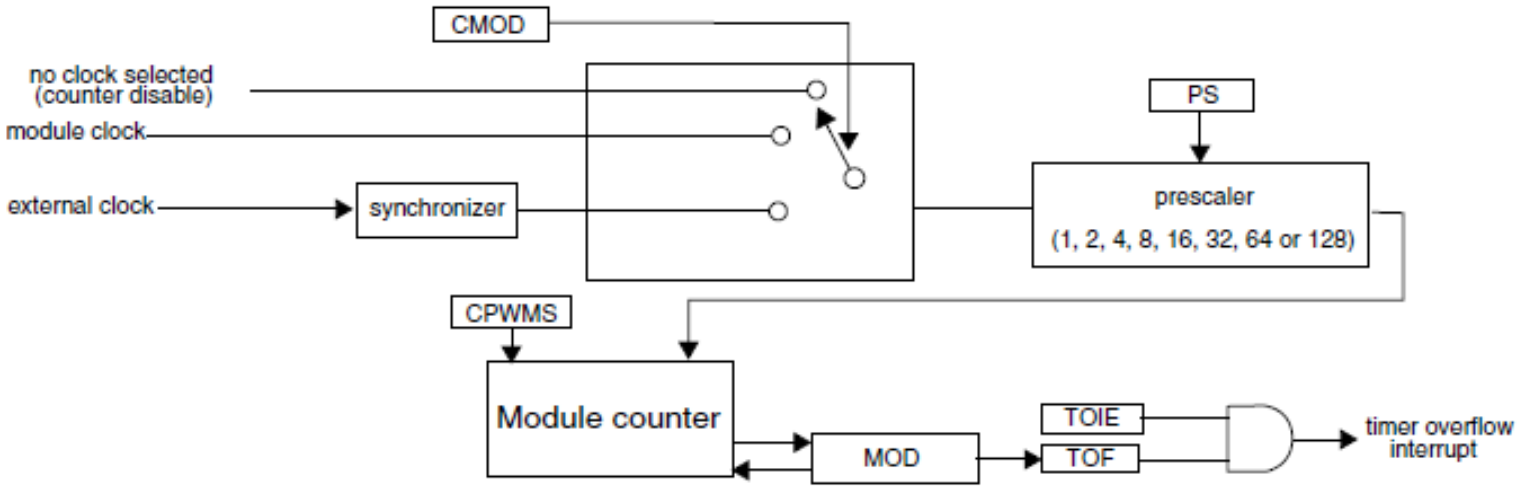
A **SysTick** számlálót 1 ms késleltetésre használjuk a korábbi **delayMs()** függvény belső for ciklusa helyett. A programban a zöld LED-et villogtatjuk 1000 ms-onként átbillentve az állapotát. Ilyen hosszú késleltetést közvetlenül a **SysTick**-kel nem tudnánk előállítani.

```
#include <MKL25Z4.H>
    void delayMs(int n);          // Késlelteto függvény
int main (void) {
    SIM->SCGC5 |= 0x400;          // A GPIOB port enegedélyezése
    PORTB->PCR[19] = 0x100;       // PTB19 legyen GPIO módban!
    PTB->PDDR |= 0x080000;        // PTB19 legyen kimenet!
    while (1) {
        delayMs(1000);            // 1000 ms késleltetés
        PTB->PTOR = 0x080000;     // Átbillentjük a zöld LED állapotát
    }
}
void delayMs(int n) {
    int i;
    SysTick->LOAD = 20970 - 1; // Újratöltési érték 1 ms késleltetéshez
    SysTick->VAL  = 0;         // Számláló törlése
    SysTick->CTRL = 0x5;       // Engedélyezés, nincs interrupt, rendszer órajel
    for(i = 0; i < n; i++) {
        while((SysTick->CTRL & 0x10000) == 0);   // A COUNT jelzőre várunk
    }
    SysTick->CTRL = 0;         // SysTick leállítása
}
```

Általános célú időzítők (TPM)

Az **MKL25Z128VLK4** mikrovezérlő 3 db általános felhasználású időzítő/számláló egységgel rendelkezik: **TPM0, TPM1, TPM2**, nevük a **Timer/PWM Module** elnevezés rövidítése.
Részletes leírás: Freescale KL25 Sub_Family Reference Manual 31. fejezetében.
Mindegyik számlálóhoz több impulzus-szélesség modulációra (PWM), bemeneti jelrögzítésre (Input Capture), vagy digitális komparálásra (Output Compare használható csatorna is kapcsolódik: **TPM0** 6 db, **TPM1** és **TPM2** pedig két-két csatornával rendelkezik.

Egy **TPM** számláló/időzítő részegységének vázlata az alábbi ábrán látható.
A főbb összetevők: a bemeneti jelválasztó, az előszámláló, a modulo számláló, a modulo adatregiszter, valamint a túlcsordulást jelző és megszakításkérő áramkörök.



A TPMx modulok regiszterkészlete

TPM0 báziscíme: 0x4003 8000
TPM1 báziscíme: 0x4003 9000
TPM2 báziscíme: 0x4003 A000

Ebben a részben ezzel a három regiszterrel lesz dolgunk

A **TPM0** modul regiszterei

Regiszternév	A regiszter funkciója	Cím
TPM0_SC	Status and Control (állapotjelző és vezérlő regiszter)	4003_8000
TPM0_CNT	Counter (számláló regiszter)	4003_8004
TPM0_MOD	Modulo (számlálási határ)	4003_8008
TPM0_C0SC	Channel 0 Status and Control (0. csatorna állapotjelző és vezérlő)	4003_800C
TPM0_C0V	Channel 0 Value (0. csatorna adatregiszter)	4003_8010
TPM0_C1SC	Channel 1 Status and Control (1. csatorna állapotjelző és vezérlő)	4003_8014
TPM0_C1V	Channel 1 Value (1. csatorna adatregiszter)	4003_8018
. . .	. . .	. . .
TPM0_STATUS	Capture and Compare Status (Jelrögzítés és komparálás státusza)	4003_8050
TPM0_CONF	Configuration (Konfiguráció)	4003_8084

TPMx_SC: állapotjelző és vezérlő

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							DMA	TOF	TOIE	CPWMS	CMOD		PS		
W									w1c							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Csak a legalsó 9 bitet használjuk. A **TOF** jelzőbitet ’1’ írásával lehet törölni.

A bitek szerepe:

DMA - DMA átvitel engedélyezése a TOF jelzőbitre

TOF - Timeout jelzőbit (TOF = 1, amikor a számláló a MOD értéket meghaladja

TOIE - Megszakításkérés engedélyezése (**0:** megszakítás tiltva, **1:** megszakítás engedélyezve)

CPWMS - Szimmetrikus (Center-aligned) PWM mód választása (**0:** a számláló felfelé számlál, **1:** a számláló fel/le számlál)

CMOD - Órajel választás (**00:** számláló letiltva, **01:** belső órajel, **10:** külső órajel, **11:** fenntartva)

PS - Előosztó osztási arány választás: 000 Divide by 1, 001 Divide by 2, 010 Divide by 4, 011 Divide by 8, 100 Divide by 16, 101 Divide by 32, 110 Divide by 64, 111 Divide by 128

Fölfelé számlálás

Modulo számlálás fölfelé: **CPWMS = 0** esetén a számlálás a **TPMx_MOD** regiszterben megadott értékig történik, utána nullára vált a számláló és ’1’-be áll a TOF bit.

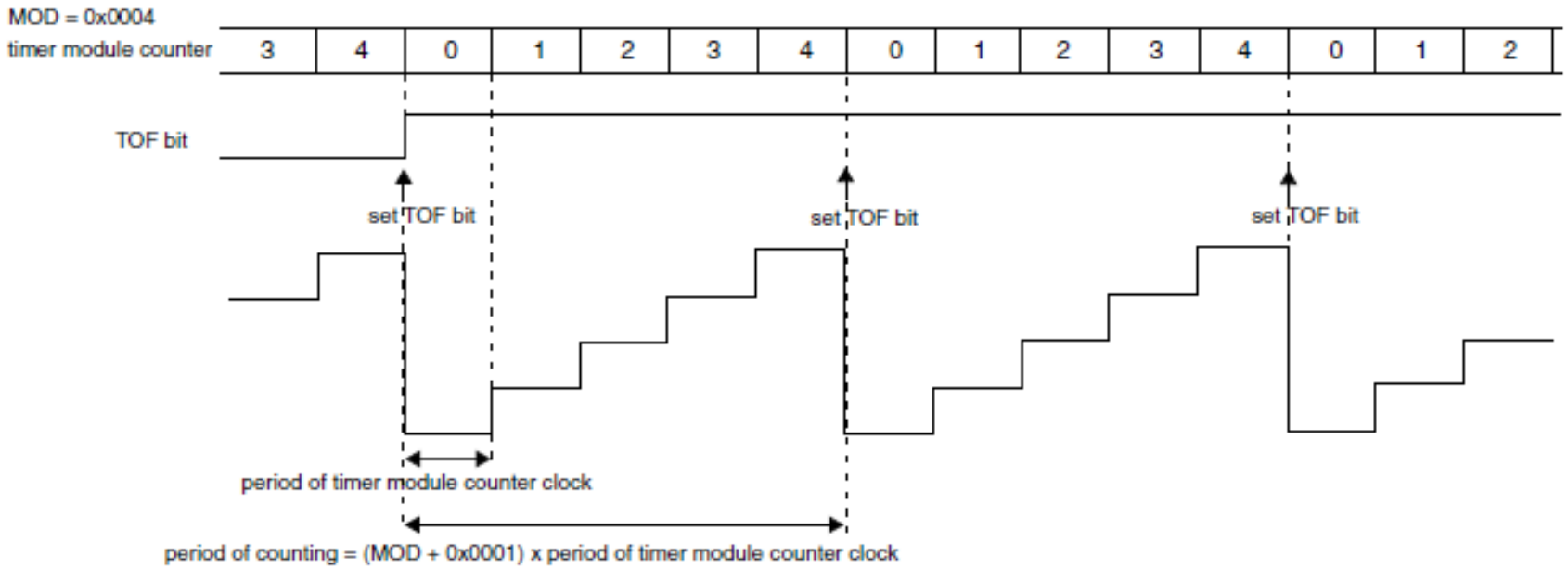


Figure 31-79. Example of TPM Up Counting

Számlálás fel- le

Modulo számlálás fel és le: **CPWMS = 1** esetén a felfelé számlálás a **TPMx_MOD** regiszterben megadott értékig történik, utána MOD-1-re vált a számláló és ’1’-be áll a TOF bit, a számlálás pedig nulláig folytatódik.

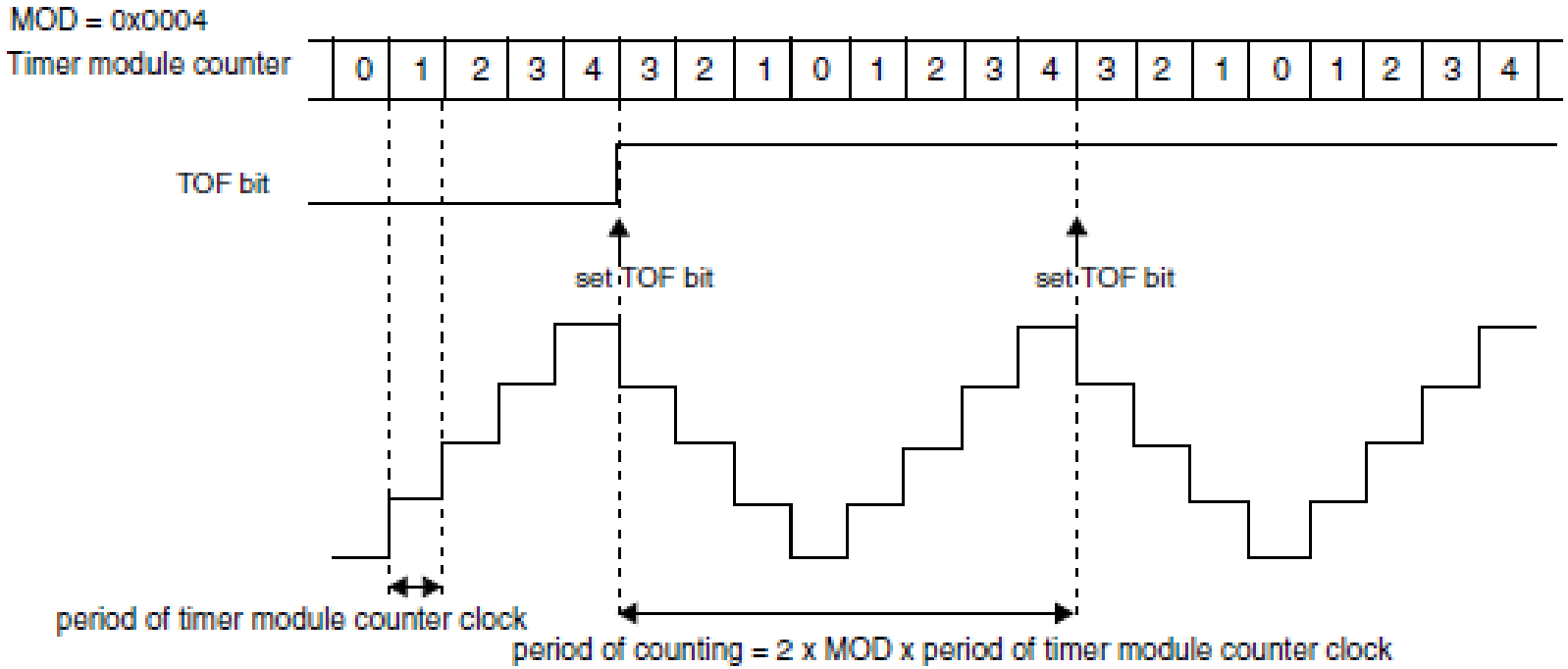


Figure 31-80. Example of Up-Down Counting

A TPMx modulok engedélyezése

Mielőtt valamelyik **TPMx** időzítőt használatba vesszük, engedélyezni a működését a System Clock Gating Control 6 (**SIM_SCGC6**) nevű regiszter megfelelő bitjének '1'-be állításával az alábbi ábra szerint.

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DAC0	0	RTC	0	ADC0	TPM2	TPM1	TPM0	PIT	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

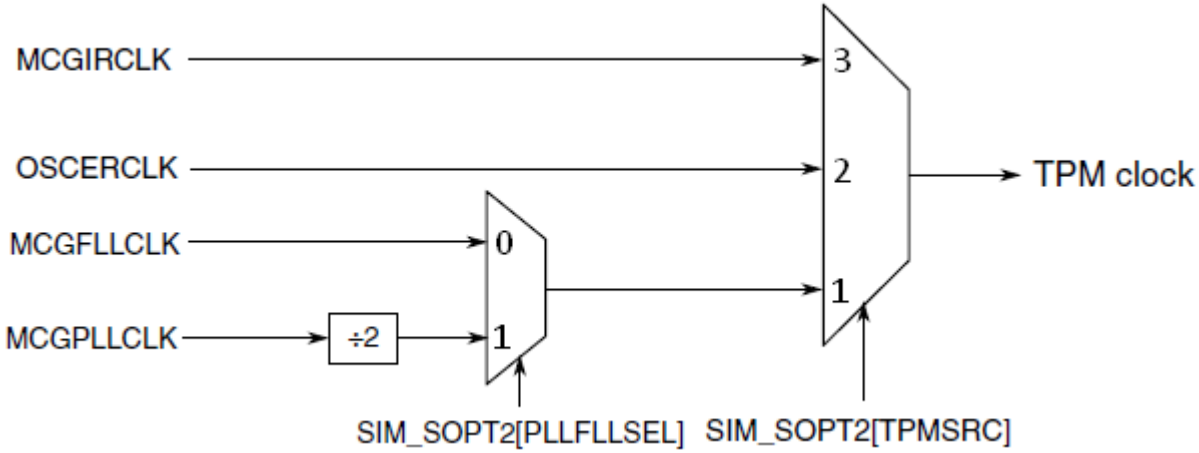
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0													DMAMUX	FTF
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

A **SIM_SCGC6** regiszter a Rendszerintegrációs Modulhoz (**SIM**) tartozik, részletes leírása a Freescale KL25 Sub_Family Reference Manual 12. fejezetében található.

A TPM órajel kiválasztása

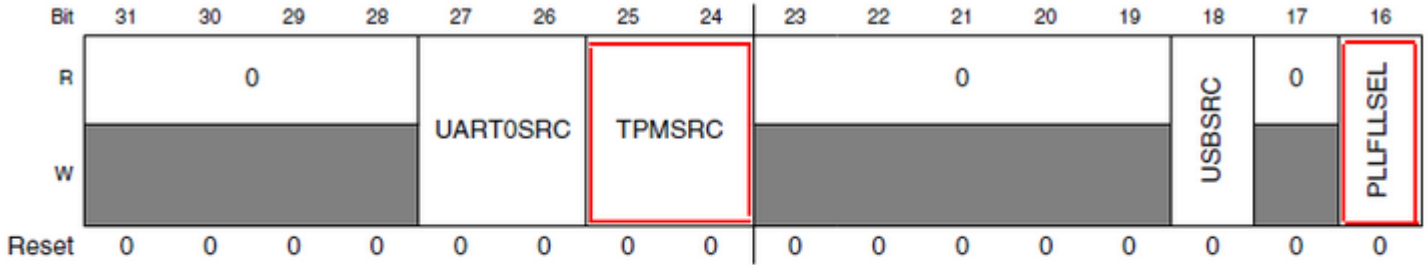
Az órajel kiválasztása a **SIM** rendszerintegrációs modul **SIM_SOPT2** regiszterének konfigurálásával történik. A választott órajel rendelkezésre állása függ az **MCG** órajel generátor konfigurálásától, esetünkben az RTE Device Setup állományokban beállított **MCG** módtól (**CLOCK_SETUP** definiált értéke a **system_MKL25Z4.h** állományban)

CLOCK_SETUP	M5CGFLLCLK	MCGPLLCLK/2	OSCERCLK	OSCIRCLK
0	20,97 MHz	–	–	32,768 kHz
1	–	48 MHz	8 MHz	–
2	–	–	–	4 MHz
3	–	–	4 MHz	–
4	–	48 MHz	8 MHz	–



A SIM_SOPT2 regiszter

A **SIM_SOPT2** regiszter bitcsoportjai közül **TPMSRC[1:0]** a **TPMx** modulok közös órajelének kiválasztására szolgál (**00:** órajel letiltva, **01:** MCGFLLCLK vagy MCGPLLCLK/2, **10:** OSCERCLK, **11:** MCGIRCLK), a **PLLFLLSEL** bit az **FLL** vagy **PLL** áramkör kimenő órajelének kiválasztására szolgál (**0:** MCGFLLCLK, **1:** MCGPLLCLK/2), s ez a bit más perifériák (pl. **UART0**) működésére is hatással van!



Az **MKL25Z128VLK4** mikrovezérlő órajeleinek előállításáról és elnevezéseiről a **KL25 Sub-Family Reference Manual** 5. fejezetében, a **SIM** rendszerintegrációs modul regisztereinek szerepéről és bitkiosztásairól a 12. fejezetében, a többcélú órajel generátor (**MCG**) felépítéséről és konfigurálásáról pedig a 24. fejezetben találunk részletes információt.

TPMx konfigurálásának lépései

Egyszerű időzítő üzemmód (**TOF** figyelése lekérdezéses módban) beállításának lépései:

1. **A TPMx modul engedélyezése:** írjunk 1-et a **SIM_SCGC6** regiszter megfelelő bitjébe!
2. **Válasszuk ki a TPM órajelet:** állítsuk be a kívánt értékre a **SIM_SOPT2** regiszter **TPMSRC[1:0]** és **PLLFLLSEL** bitjeit az előző oldalakon leírtak szerint!
3. **Tiltsuk le a TPMx modult a konfigurálás idejére:** írjunk nullát a **TPMx_SC** regiszterbe!
4. **Töltsük fel a modulo regisztert:** a megfelelő értéket írjuk be a **TPMx_MOD** regiszterbe!
5. **Töröljük a TOF jelzőbitet:** bármilyen meglepő, a törléshez 1-et kell írnunk a **TPMx_SC** regiszter **TOF** bitjébe!
6. **Töröljük a számláló regisztert:** írjunk nullát a **TPMx_CNT** regiszterbe!
7. **Engedélyezzük a számláló működését** a megfelelő üzemmód beállításával együtt: egyszerű időzítéshez a **TPMx_SC** regiszter **CMOD** bitcsoportját állítsuk **01**-be, a **PS** bitcsoportot pedig állítsuk be a kívánt előszámlálási aránynak megfelelően (000: 1x, 001: 2x, 010: 4x, ... 111: 127x leosztás)!
8. **Az időzítéshez:** várjunk addig, amíg a **TOF** jelzőbit '1' nem lesz!

Program5_4: TPM0 szabadonfutó módban

A kék LED-et villogtatjuk, s a **TPM0** időzítőt használjuk a késleltetés előállításához. A **TPM0** 16 bites számlálója legfeljebb 65536/20 970 000 Hz = kb. 3,125 ms késleltetést biztosít, ezért ***for*** ciklusban 160 darab túlcsordulást várunk meg, tehát 3,125 ms * 160 = 500 ms telik el két állapotváltás között. A kék LED a **PTD1** lábra csatlakozik, ezt GPIO módba konfiguráljuk.

```
#include <MKL25Z4.H>
int main (void) {
    int i;
    SIM->SCGC5 |= 0x1000;        // A D port engedélyezése
    PORTD->PCR[1] = 0x100;       // PTD1 pin legyen GPIO módban!
    PTD->PDDR |= 0x02;           // PTD1 legyen kimenet!
    SIM->SCGC6 |= 0x01000000;    // A TPM0 idozíto engedélyezése
    SIM->SOPT2 |= 0x01000000;    // MCGFLLCLK legyen az órajel!
    TPM0->SC = 0;                // Konfigurálsához letiltjuk az idozítot
    TPM0->MOD = 0xFFFF;          // Maximális modulo érték (65536-1)
    TPM0->CNT = 0;               // A számláló törlése
    TPM0->SC |= 0x80;            // TOF törlése
    TPM0->SC |= 0x08;            // Számláló engedélyezése (CMOD=01, PS=000)
    while (1) {
      for(i = 0; i < 160; i++) {
        while(!(TPM0->SC&0x80));// TOF-ra várunk...
          TPM0->SC |= 0x80;      // TOF törlése
      }
      PTD->PTOR = 0x02;          // Kék LED (PTD1) átbillentése
    }
}
```

Program5_5: TPM0 előosztóval

A **TPM0** időzítőt most 128-szoros előosztással használjuk, így 65536 * 128 / 20 970 000 Hz = kb. 400 ms telik el két állapotváltás között. A villogtatni kívánt kék LED a **PTD1** lábra csatlakozik, ezt GPIO módba konfiguráljuk.

```
#include <MKL25Z4.H>

int main (void) {
    SIM->SCGC5 |= 0x1000;          // A D port engedélyezése
    PORTD->PCR[1] = 0x100;         // PTD1 pin legyen GPIO módban!
    PTD->PDDR |= 0x02;             // PTD1 legyen kimenet!

    SIM->SCGC6 |= 0x01000000;      // A TPM0 időzítő engedélyezése
    SIM->SOPT2 |= 0x01000000;      // MCGFLLCLK legyen az órajel!
    TPM0->SC = 0;                  // Konfiguráláshoz letiltjuk az időzítőt
    TPM0->SC = 0x07;               // Az előszámláló leosztása 128 legyen!
    TPM0->MOD = 0xFFFF;            // Maximális modulo érték (65536-1)
    TPM0->CNT = 0;                 // A számláló törlése
    TPM0->SC |= 0x80;              // TOF törlése
    TPM0->SC |= 0x08;              // Számláló engedélyezése (CMOD=01)

    while (1) {
      while(!(TPM0->SC&0x80));     // TOF-ra várunk...
      TPM0->SC |= 0x80;            // TOF törlése
      PTD->PTOR = 0x02;            // Kék LED (PTD1) átbillentése
    }
}
```

Nincs ***for*** ciklus!

Program5_6: TPM1 használata

Ez a program lényegében megegyezik a **Program5_5** mintapéldával, de most a **TPM1** időzítőt használjuk a késleltetés előállításához. A programban a **TPM0** -> **TPM1** átnevezések mellett a **SIM->SCGC6** regiszterben most másik bitet kell '1'-be állítanunk…

```
#include <MKL25Z4.H>

int main (void) {
    SIM->SCGC5 |= 0x1000;        // A D port engedélyezése
    PORTD->PCR[1] = 0x100;       // PTD1 pin legyen GPIO módban!
    PTD->PDDR |= 0x02;           // PTD1 legyen kimenet!
    SIM->SCGC6 |= 0x02000000;    // TPM1 időzítő engedélyezése
    SIM->SOPT2 |= 0x01000000;    // MCGFLLCLK legyen a TPM órajel!
    TPM1->SC = 0;                // TPM1 letiltása a konfiguráláshoz
    TPM1->SC = 0x07;             // Az előszámláló leosztása: 128
    TPM1->MOD = 0xFFFF;          // Maximális modulo érték (65536-1)
    TPM1->CNT = 0;               // A számláló törlése
    TPM1->SC |= 0x80;            // TOF törlése
    TPM1->SC |= 0x08;            // Számláló engedélyezése (CMOD=01)

    while (1) {
      while(!(TPM1->SC & 0x80));// TOF-ra várunk...
      TPM1->SC |= 0x80;          // TOF törlése
      PTD->PTOR = 0x02;          // Kék LED (PTD1) átbillentése
    }
}
```

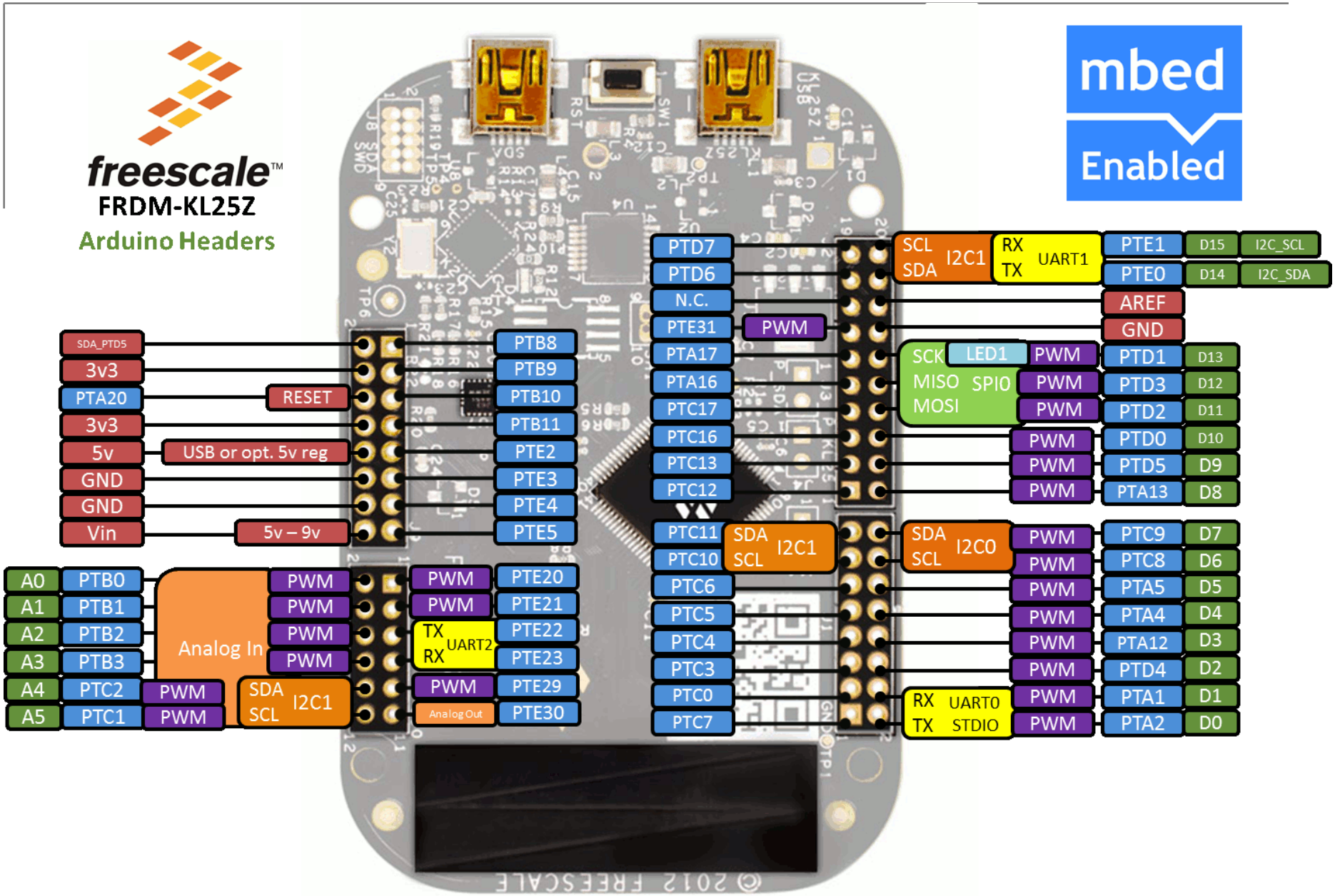
Program5_7: MCGIRCLK órajel használata

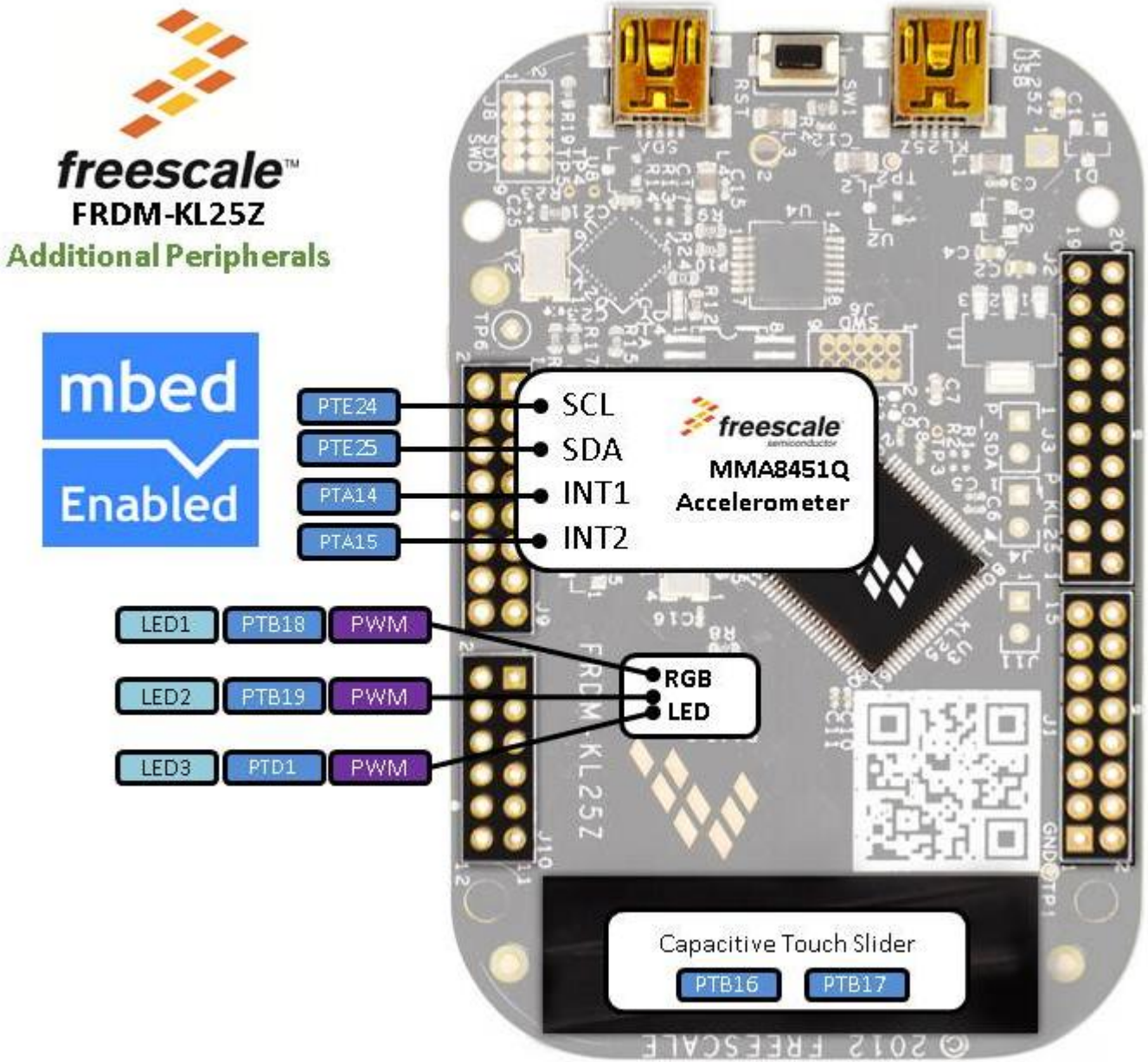
Most az **MCGIRCLK** órajelet választjuk ki **TPM** órajelnek. A **TPM0** időzítő előosztóját 32-szeres leosztásra konfiguráljuk, így 1024 számláló inkrementálás 1 s késleltetésnek felel meg.
A programban szereplő 5120-1 modulo beállítással a LED állapotváltásai között 5 s telik el.
Megjegyzés: Alapértelmezett módban **MCGIRCLK** nincs engedélyezve!

```
#include <MKL25Z4.H>
int main (void) {
    MCG->C1     |= 2;              // Enables IRCLK for use as MCGIRCLK
    SIM->SCGC5 |= 0x1000;          // A D port engedélyezése
    PORTD->PCR[1] = 0x100;         // PTD1 pin legyen GPIO módban!
    PTD->PDDR |= 0x02;             // PTD1 legyen kimenet!

    SIM->SCGC6 |= 0x01000000;      // A TPM0 időzítő engedélyezése
    SIM->SOPT2 |= 0x03000000;      // MCGIRCLK legyen a TPM órajel
    TPM0->SC = 0;                  // Konfiguráláshoz letiltjuk az időzítőt
    TPM0->SC = 0x05;               // Előosztási arány 32 legyen
    TPM0->MOD = 5120 - 1;          // 0 - 5119-ig számlálunk (5 sec)
    TPM0->SC |= 0x80;              // TOF törlése
    TPM0->SC |= 0x08;              // Számláló engedélyezése (CMOD=01)

    while (1) {
      while(!(TPM0->SC & 0x80)); // TOF-ra várunk...
      TPM0->SC |= 0x80;            // TOF törlése
      PTD->PTOR = 0x02;            // Kék LED (PTD1) átbillentése
    }
}
```


freescale
FRDM-KL25Z
Arduino Headers
mbed
Enabled
SDA_PTD5
3v3
PTA20
3v3
5v
GND
GND
Vin
RESET
USB or opt. 5v reg
5v – 9v
PTB8
PTB9
PTB10
PTB11
PTE2
PTE3
PTE4
PTE5
A0 PTB0
A1 PTB1
A2 PTB2
A3 PTB3
A4 PTC2
A5 PTC1
Analog In
PWM
SDA SCL I2C1
PWM
PWM
PWM
PWM
PWM
PWM
PTE20
PTE21
TX RX UART2
PTE22
PTE23
PWM
PTE29
Analog Out
PTE30
PTD7
PTD6
N.C.
PTE31
PWM
PTA17
PTA16
PTC17
PTC16
PTC13
PTC12
PTC11
PTC10
SDA SCL I2C1
PTC6
PTC5
PTC4
PTC3
PTC0
PTC7
SCL SDA I2C1
RX TX UART1
PTE1 D15 I2C_SCL
PTE0 D14 I2C_SDA
AREF
GND
SCK MISO MOSI SPI0
LED1
PWM
PWM
PWM
PTD1 D13
PTD3 D12
PTD2 D11
PWM
PWM
PWM
PTD0 D10
PTD5 D9
PTA13 D8
SDA SCL I2C0
PWM
PWM
PWM
PWM
PWM
PWM
PWM
PTC9 D7
PTC8 D6
PTA5 D5
PTA4 D4
PTA12 D3
PTD4 D2
PTA1 D1
PTA2 D0
RX TX UART0 STDIO
PWM
PWM
© 2012 FREESCALE


freescale
FRDM-KL25Z
Additional Peripherals
mbed
Enabled
PTE24
PTE25
PTA14
PTA15
SCL
SDA
INT1
INT2
freescale
MMA8451Q
Accelerometer
LED1
PTB18
PWM
LED2
PTB19
PWM
LED3
PTD1
PWM
RGB
LED
Capacitive Touch Slider
PTB16
PTB17