ÓBUDAI EGYETEM

TIMER modul működésének áttekintése

Modul VI.

Lecke célja

- A lecke célja bemutatni a beágyazott rendszereknél használt időzítési lehetőségeket.
- Bemutatásra kerül az Arduino IDE által biztosított delay(), delayMicroseconds(), millis(), micros() függvények, továbbá egy timer modul regiszter szintű konfigurálása is.

Bevezetés

- A beágyazott rendszerek egyik legfontosabb perifériája az úgynevezett TIMER (időzítő).
- Működés során sokszor valamilyen ciklikus feladatot kell elvégeznie a rendszernek. Ez lehet fix időintervallumonkénti adatgyűjtés, megjelenítés, beavatkozás, vagy két interakció között eltel idő mérése stb.
- A TIMER egyik fő feladata, hogy ezeket az időzített, ütemezett feladatokat menedzselje.
- Külön perifériaként állnak rendelkezésre a mikroprocesszoron belül.

Bevezetés

- A TIMER alapja egy számláló regiszter. A számláló regiszter a bejövő impulzusok változásának a függvényében inkrementálja vagy dekrementálja az értékét.
- Ha ismert az impulzusváltozás frekvenciája, pl. egy fix órajel generátorral (oszcillátorral) egészítjük ki a számláló regisztert, akkor beszélhetünk időzítőről (TIMER).
- A órajel frekvenciájának és a számláló értékének az ismeretében könnyedén fel lehet használni ütemezésre ezt az összeállítást.
- Pl. egy 1MHz-es órajel esetén, a periódusidő 1/1000000 másodperc = 1us időközönként inkrementálódik a számláló érétke

Bevezetés

- Az ATmega2560 mikroprocesszor több TIMER-el is rendelkezik.
- Két csoport érhető el az adatlap alapján:
  - 8bites TIMER 0/2
  - 16 bites TIMER 1/3/4/5
- A bitek száma itt a számláló regiszterre vonatkozik.
- Természetesen nem csak időzítési feladatra használható fel egy TIMER modul, hanem különböző jelek (pl. PWM) generálására is, frekvenciagenerátorként lehet alkalmazni, külső események számlálása stb. (típus függő). Azonban ez most nem a lecke célja.

Millis(), micros()

- Az Arduino IDE által biztosított millis() és micros() függvényeket időmérésre lehet felhasználni.
- A függvény nevében is látható a különbség. Az egyikel milliszekundumban a másikal pedig mikro szekundumban eltelt időt lehet mérni.
- A függvények a TIMER0 időzítőre építenek.
- A visszatérési érték unsigned long típusú.
- Maximális értékből meghatározható, hogy:
  - A millis() kb. 49 nap után túlcsordul
  - A micros() kb. 70 perc után túlcsordul

delay(), delayMicroseconds()

- A delay() és delayMicroseconds() függvényeket késleltetés esetén lehet használni.
- Paraméterként milli vagy mikroszekundumban kell megadni a késletetési időt.
- A legnagyobb hátránya mind a két függvénynek, hogy blokkolnak! Ami azt jelenti, hogy amíg a megadott idő el nem telik, a processzor áll, nem „végez hasznos munkát”.
- Használata egyszerűbb programok esetén nem probléma, azonban ha lehet kerülni kell!

ÓBUDAI EGYETEM

Millis(), micros() példaprogram

idozito_teszt1 | Arduino 1.6.12

Fájl Szerkesztés Vázlat Eszközök Súgó

idozito_teszt1

```
unsigned long t_start=0;
unsigned long t_stop=0;
unsigned long fact_num=0;

unsigned long factorial(int n);

void setup() {
  Serial.begin(9600);
}
```

Mentés kész.

```
Sketch uses 2 402 bytes (0%) of program storage space. Maxim
Global variables use 212 bytes (2%) of dynamic memory, leavi
```

14 Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM6

ÓBUDAI EGYETEM

Video3 Millis(), micros() példaprogram

idozito_teszt1 | Arduino 1.6.12

Fájl Szerkesztés Vázlat Eszközök Súgó

idozito_teszt1

```
unsigned long factorial(int n){
  if (n==0){
    return 1;
  }else{
    unsigned long fact=1;
    for (int i=1; i<n; i++){
      fact=fact*i;
    }
    return fact;
  }
}
```

Mentés kész.

```
Sketch uses 2 402 bytes (0%) of program storage space. Maxim
Global variables use 212 bytes (2%) of dynamic memory, leavi
```

14 Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM6

ÓBUDAI EGYETEM

Millis(), micros() példaprogram

idozito_teszt1 | Arduino 1.6.12

Fájl Szerkesztés Vázlat Eszközök Súgó

idozito_teszt1

```
void loop() {
  t_start=micros();
  fact_num=factorial(10);
  t_stop=micros();
  Serial.print(fact_num);
  Serial.print(", time: ");
  Serial.print(t_stop-t_start);
  Serial.println(" us");
  delay(5000);
}
```

Mentés kész.

```
Sketch uses 2 402 bytes (0%) of program storage space. Maxim
Global variables use 212 bytes (2%) of dynamic memory, leavi
```

14 Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM6

ÓBUDAI EGYETEM

delay(), delayMicroseconds() video4

idozito_teszt2 | Arduino 1.6.12

Fájl Szerkesztés Vázlat Eszközök Súgó

Megnyitás

idozito_teszt2 §

```
#define LED 13 //onboard LED on PIN13
unsigned long t = 0;
unsigned long prev_t_LED = 0;
unsigned long prev_t_UART = 0;

void setup() {
  pinMode(LED, OUTPUT);
  Serial.begin(9600);
  t = millis();
  prev_t_LED = t;
  prev_t_UART = t;
}
```

Mentés kész.

```
Sketch uses 2 750 bytes (1%) of program storage space. Maximum is 253 952 bytes.
Global variables use 212 bytes (2%) of dynamic memory, leaving 7 980 bytes for local
```

2 Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM6

ÓBUDAI EGYETEM

delay(), delayMicroseconds()

idozito_teszt2 | Arduino 1.6.12

Fájl Szerkesztés Vázlat Eszközök Súgó

idozito_teszt2 §

```
void loop() {
  //blink led 1sec
  t = millis();
  if (t - prev_t_LED > 1000)
  {
    digitalWrite(LED, 1 - digitalRead(LED));
    prev_t_LED += 1000;
  }

  //print Serial 3sec
  if (t - prev_t_UART > 3000)
  {
    Serial.println("Hello world!");
    prev_t_UART += 3000;
  }
}
```

Mentés kész.

```
Sketch uses 2 750 bytes (1%) of program storage space. Maximum is 253 952 bytes.
Global variables use 212 bytes (2%) of dynamic memory, leaving 7 980 bytes for local
```

23 Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM6

Timer

- Az előző két példában bemutatásra került, hogyan lehet eltelt időt mérni, illetve a blokkoló késleltetéseket elkerülni.
- A bemutatott példában előfordulhat olyan szituáció, amikor a loop ciklus újrakezdése több időig tart (pl. bonyolult számítás végzése miatt) mint az általunk elvárt legrövidebb ütemezési feladat periódusideje.
- Ebben az esetben az időzítést egy megszakításba kell kiszervezni.
- Minden TIMER túlcsordulás esetén egy megszakítással jelzi ezt az eseményt (programozható). A megszakítást lehet felhasználni ütemezésre.
- Azonban itt is figyelembe kell venni, hogy hosszú, számításigényes feladatot ha lehet, megszakításon belül ne végezzünk, mert addig blokkolva van a processzor!

TIMER 5

- A következőekben a TIMER 5 beállítása kerül bemutatásra.
- Időzítési feladathoz a következő regiszterekre lesz szükség:
  - TCCR5A
  - TCCR5B
  - TCNT5
  - TIMSK5
  - TOIE5
- A TIMER modult a regiszterek egyes bitjeinek a bebillentésével lehet beállítani. Minden egyes bithez tartozik valamilyen beállítás. (adatlap!!!)

TCCR5A és TCCR5B regiszter

- A TCCRnA/B (Timer/Counter Control Regisister) regiszterrel lehet a modult beállítani.
- A mi esetünkben egy sima időzítési feladatot fog ellátni.
- A TCCR5B regiszterben található CS:52-CS:50 bitekkel lehet bekapcsolni és kikapcsolni (órajelet beállítás) a modult.

17.11.8 TCCR5B – Timer/Counter 5 Control Register B

Bit	7	6	5	4	3	2	1	0	
(0x121)	ICNC5	ICES5	–	WGM53	WGM52	CS52	CS51	CS50	TCCR5B
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

TCCR5A és TCCR5B regiszter

- A TIMER5 beállítástól függően a CPU fő órajelét, osztással/osztás nélkül (16MHz) vagy külső órajel forrást használhat. Ezt „CS” bitekkel lehet beállítani az adatlapban megtalálható értékek függvényében:

Table 17-6. Clock Select Bit Description

CSn2	CSn1	CSn0	Description
0	0	0	No clock source. (Timer/Counter stopped)
0	0	1	$clk_{I/O}$/1 (No prescaling
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/64 (From prescaler)
1	0	0	$clk_{I/O}$/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)
1	1	0	External clock source on Tn pin. Clock on falling edge
1	1	1	External clock source on Tn pin. Clock on rising edge

TIMER5

- A fő órajelet leosztva lehet szabályozni a számláló növekedésének periódusidejét.
- A maximális értéket a TCNT5 regiszter szabja meg. A TCNT5 egy 16 bites regiszter, azaz 65535 lehet a maximális érétke.
- A közvetlenül írható és olvasható.
- Külön kezelhető az alsó és felső 8 bit.(TCNT5H és TCNT5L)

17.11.16 TCNT5H and TCNT5L –Timer/Counter 5

Bit	7	6	5	4	3	2	1	0	
(0x125)	TCNT5[15:8]								TCNT5H
(0x124)	TCNT5[7:0]								TCNT5L
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

TIFR5 regiszeter

- Túlcsordulás esetén a TIFR5 (Timer/Counter5 Interrupt Flag Regiszter) regiszterben található TOV5 (Timer/Countern, Overflow Flag) jelzőbit állapota billen át 1 érékre.
- Ezt az állapot változást lehet felhasználni ütemezésre. A jelzőbithez hozzá lehet rendelni egy megszakítást, ami akkor keletkezik, ha 1 lesz az értéke.
- Ezt a TIMSK5 regiszter segítségével lehet aktiválni.

17.11.40 TIFR5 – Timer/Counter5 Interrupt Flag Register

Bit	7	6	5	4	3	2	1	0	
0x1A (0x3A)	–	–	ICF5	–	OCF5C	OCF5B	OCF5A	TOV5	TIFR5
Read/Write	R	R	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

TIMSK5 regiszter

- A TIMSK5 regiszter legkisebb helyiértékű bitjéhez rendelt TOIE5 (Timer/Countern, Overflow Interrupt Enable) a bebillentésével lehet engedélyezni vagy tiltani a túlcsorduláshoz tartozó megszakításkérést.
- A kiváltott megszakításneve TIMER5_OVF_vect.

17.11.36 TIMSK5 – Timer/Counter 5 Interrupt Mask Register

Bit	7	6	5	4	3	2	1	0	
(0x73)	–	–	ICIE5	–	OCIE5C	OCIE5B	OCIE5A	TOIE5	TIMSK5
Read/Write	R	R	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Folyamat rövid áttekintése

- 65535 után a TCNT5 regiszter túlcsordul és a TIFR regiszterben található TOV5 bit bebillen. Mivel a TIMSK5 regiszterben TOIE5 engedélyezve van (és természetesen az SREG regiszter általános megszakításokhoz tartozó „I” (Global Interrupt Enable) is engedélyezve van) akkor a CPU végrehajtja az ISR (Interrupt Service Routin) rutint.
- A kiszolgálás után a TOV5 automatikusan törlődik és a TCNT5 regiszter elkezd 0-tól újra számolni.

1 másodperces ütemező

- Az időzítést a TCNT5 regiszter értékének a manipulálásával lehet beállítani.
- Ismert a TIMER5 modulra kapcsolt órajel, azaz az inkrementálás periódus ideje.
- A feladat egy 1s-os ütemező létrehozása, 1024-es órajel leosztás esetén:
  - TIMER órajele: 16Mhz/1024= 15625Hz
  - Periódusidő: 1/15625 = 0.000064 s
  - 1s időzítő -> 1s/ 0.000064s =15625
  - A TCNT5 számláló érétkét tehát 65535-15625= 49910-ről kell indítani, ekkor 1s-ként fog túlcsordulni, azaz megszakítást eredményezni.

Összefoglalás

- Ebben a leckében bemutatásra került az Arduino IDE által elérhető időzítő függvények, továbbá egy példaprogram keretén belül láthattunk egy 16 bites TIMER modul regiszter szintű beállítását.
- A későbbiekben a robothoz mindenkép szükség lesz egy ilyen megszakításszintű ütemezőre, ugyanis a PID szabályozó algoritmus számára biztosítani kell a fix mintavételezési időt.

ÓBUDAI EGYETEM

1s ütemező példaprogram video 5

idozito_teszt3 | Arduino 1.6.12

Fájl Szerkesztés Vázlat Eszközök Súgó

idozito_teszt3 §

```
#define LED 13

void InitTimer5();

void setup() {
  pinMode(LED, OUTPUT);
  InitTimer5();
}

void loop() {
}
```

Felöltés kész.

```
Sketch uses 1 520 bytes (0%) of program storage space. Maximum is 253 952 byte
Global variables use 9 bytes (0%) of dynamic memory, leaving 8 183 bytes for l
```

14 Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM6

ÓBUDAI EGYETEM

1s ütemező példaprogram

idozito_teszt3 | Arduino 1.6.12

Fájl Szerkesztés Vázlat Eszközök Súgó

idozito_teszt3 §

```
void InitTimer5() {
  TCCR5A = 0; // set entire TCCR2A register to 0
  TCCR5B = 0;// same for TCCR2B
  TCNT5 = 65535 - 15625; //initialize counter value to 0
  TCCR5B |= B00000101; //TCCR1B |= (1 << CS10);   //no prescale
  TIMSK5 |= (1 << TOIE5);
}

ISR(TIMER5_OVF_vect) {
  digitalWrite(LED, 1 - digitalRead(LED));
  TCNT5 = 65535 - 15625;
}
```

Felöltés kész.

```
Sketch uses 1 520 bytes (0%) of program storage space. Maximum is 253 952 byte
Global variables use 9 bytes (0%) of dynamic memory, leaving 8 183 bytes for l
```

14 Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM6