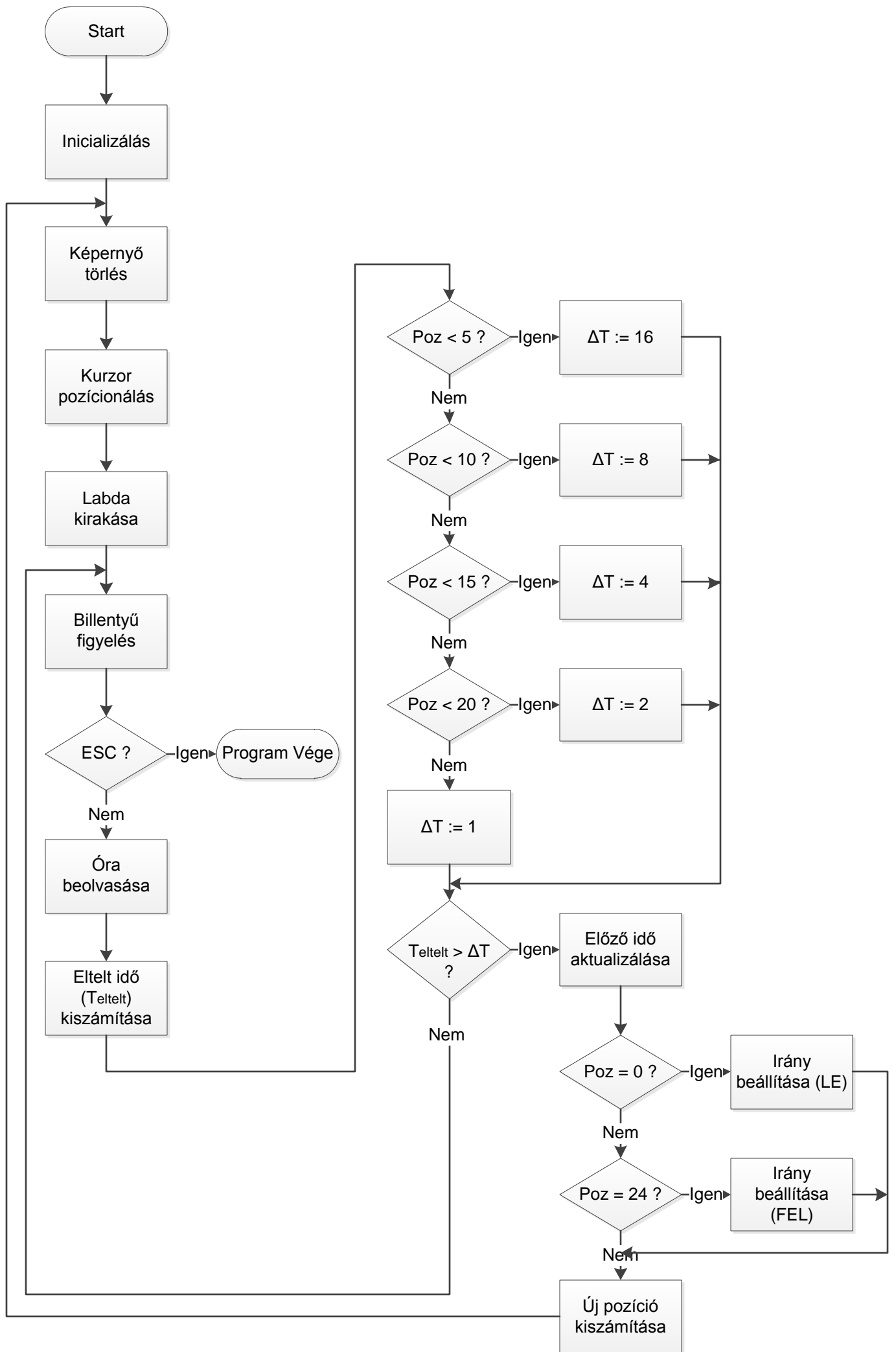


Fealadat3: labda.asm

Feladat meghatározása	Implementálás	Implementálás
<p>A program célja az assembly rutinok időzítesi lehetőségeinek bemutatása. Az időzítés az AH00, INT1Ah funkció segítségével történik. A program egy „labda” leesését szimulálja. A labdát egy „O” karakter szimbolizálja, amely fokozatosan gyorsulva esik le, majd visszapattan. A visszapattanás veszteségmentes, így a labda fokozatosan lassulva eléri a kiindulási pozíciót.</p> <p>Program bemutatása:</p> <ul style="list-style-type: none"> <li>•A programot először a késleltető rutinok nélkül kell megírni. A billentyű figyelés helyett az AH00, INT16h funkcióval megvalósítható billentyű leütésre történő várakozást kell megírni. Így a program csak akkor lép a következő iterációra, ha lenyomunk egy billentyűt. Ellenőrizni kell, hogy a „labda” nem megy ki a képből. A legalsó pozíciót elérve felfelé mozog tovább, a legfelső pozíció elérését követően lefele halad.</li> <li>•A letesztelt programban módosítani kell a billentyű leütésre váró részt az AH01, INT16h funkcióra. Ez a funkció nem állítja meg a program futását, mindössze a FLAG-ek állapotát, illetve az AL értékét módosítja, ha volt billentyű leütés. Be kell fejezni a késleltető programrészt. A késleltetés alapeleme az AH00, INT1Ah funkció, mely a rendszeróra CX:DX regiszterpárba. Egy számláló-változás kb. 1/18 sec.</li> </ul>	<pre> Code      Segment           assume CS:Code,           DS:Data, SS:Stack  Start:     mov    ax, Code     mov    DS, ax      xor    di, di     mov    si, 1      xor    dx, dx     push  dx  Torles:     mov    ax, 03h     int    10h      mov    dx, di     mov    dh, dl      mov    dl, 40     xor    bh, bh     mov    ah, 02h     int    10h      mov    dx, offset Labda     mov    ah, 09h     int    21h  Kesleltet:     mov    ah, 01h     int    16h     ;jnz   Program_Vege      jz     nincsbill     mov    ah, 00h     int    16h     cmp    al, 27     jz     Program_Vege  nincsbill:     xor    ah, ah     int    1ah      pop    cx     push  cx     mov    ax, dx     sub    dx, cx     push  ax      cmp    di, 5     jnc    ldo1     mov    al, 16     jmp    Beallit  ldo1:     cmp    di, 10     jnc    ldo2     mov    al, 8     jmp    Beallit  ldo2:     cmp    di, 15     jnc    ldo3     mov    al, 4     jmp    Beallit  ldo3: </pre>	<pre>     cmp    di, 20     jnc    ldo4     mov    al, 2     jmp    Beallit  ldo4:     mov    al, 1  Beallit:     xor    ah, ah     cmp    dx, ax      pop    ax      jc     Kesleltet      pop    cx     push  ax      cmp    di, 0     jz     Lefele      cmp    di, 24     jz     Felfele  Mozgas:     add    di, si     jmp    Torles  Lefele:     mov    si, 1     jmp    Mozgas  Felfele:     mov    si, -1     jmp    Mozgas  Program_Vege:     pop    cx     mov    ax, 4c00h     int    21h  Labda:     db    "O\$"  Code      Ends  Data      Segment Data      Ends  Stack     Segment Stack     Ends  End Start </pre>



Start

Inicializálás

Képernyő  
törlés

Kurzor  
pozícionálás

Labda  
kirakása

Billentyű  
figyelés

ESC ?

Igen Program Vége

Nem

Óra  
beolvasása

Eltelt idő  
(Teltelt)  
kiszámítása

### Implementálás

#### Inicializálás

#### Verem

```

xor    di, di    ; labda helye (sor)
mov    si, 1     ; lefelé indul a labda (irány vektor)
xor    dx, dx
push   dx       ; verembe a régi idő (most 0)

```

;program vége után, még a Code szegmensbe

Labda: db "o\$"

régi idő (0)

Poz < 5 ?

Igen

$\Delta T := 16$

Nem

Poz < 10 ?

Igen

$\Delta T := 8$

Nem

Poz < 15 ?

Igen

$\Delta T := 4$

Nem

Poz < 20 ?

Igen

$\Delta T := 2$

Nem

$\Delta T := 1$

Teltelt >  $\Delta T$  ?

Igen

Előző idő  
aktualizálása

Nem

Poz = 0 ?

Igen

Irány  
beállítása (LE)

Nem

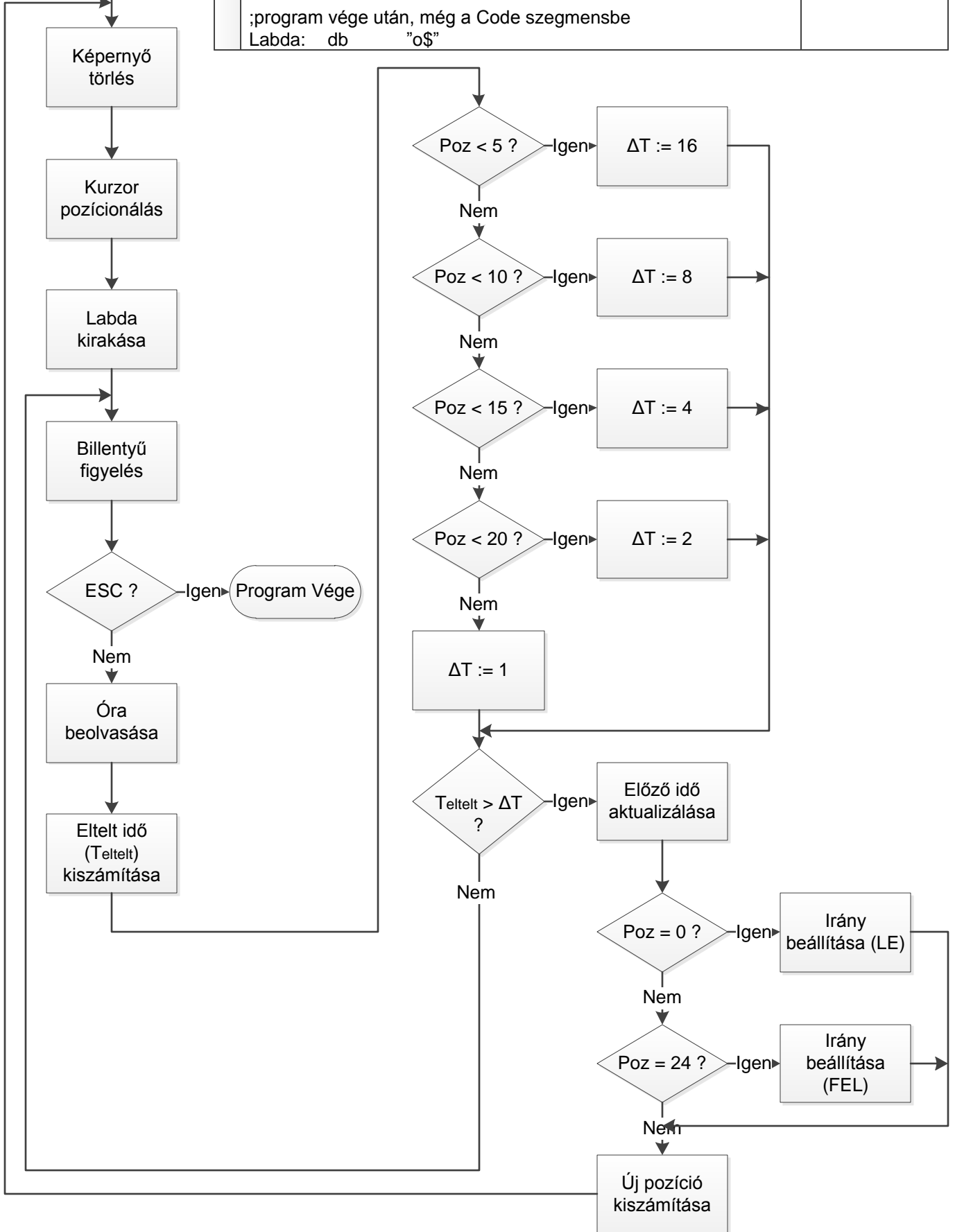
Poz = 24 ?

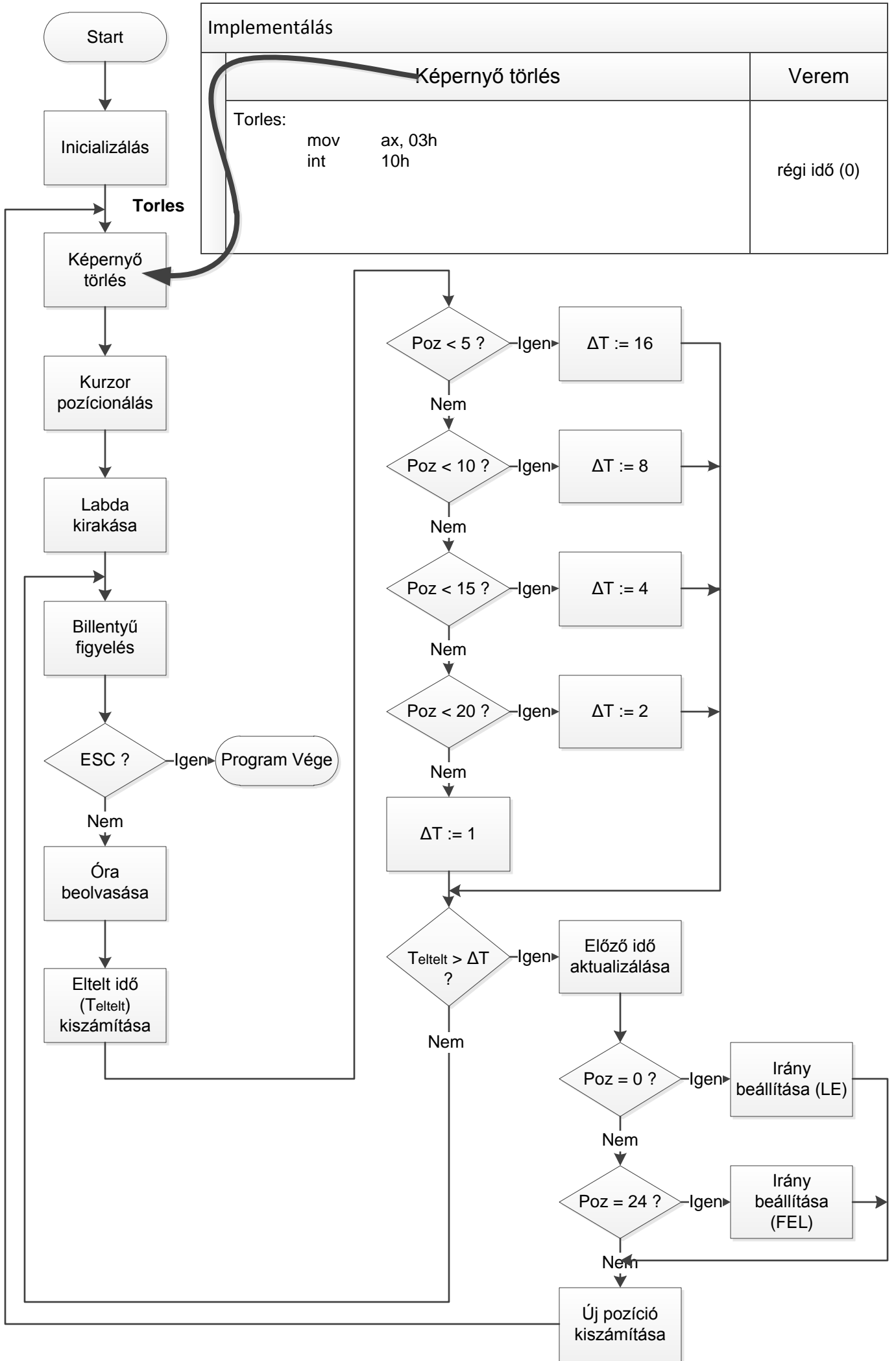
Igen

Irány  
beállítása (FEL)

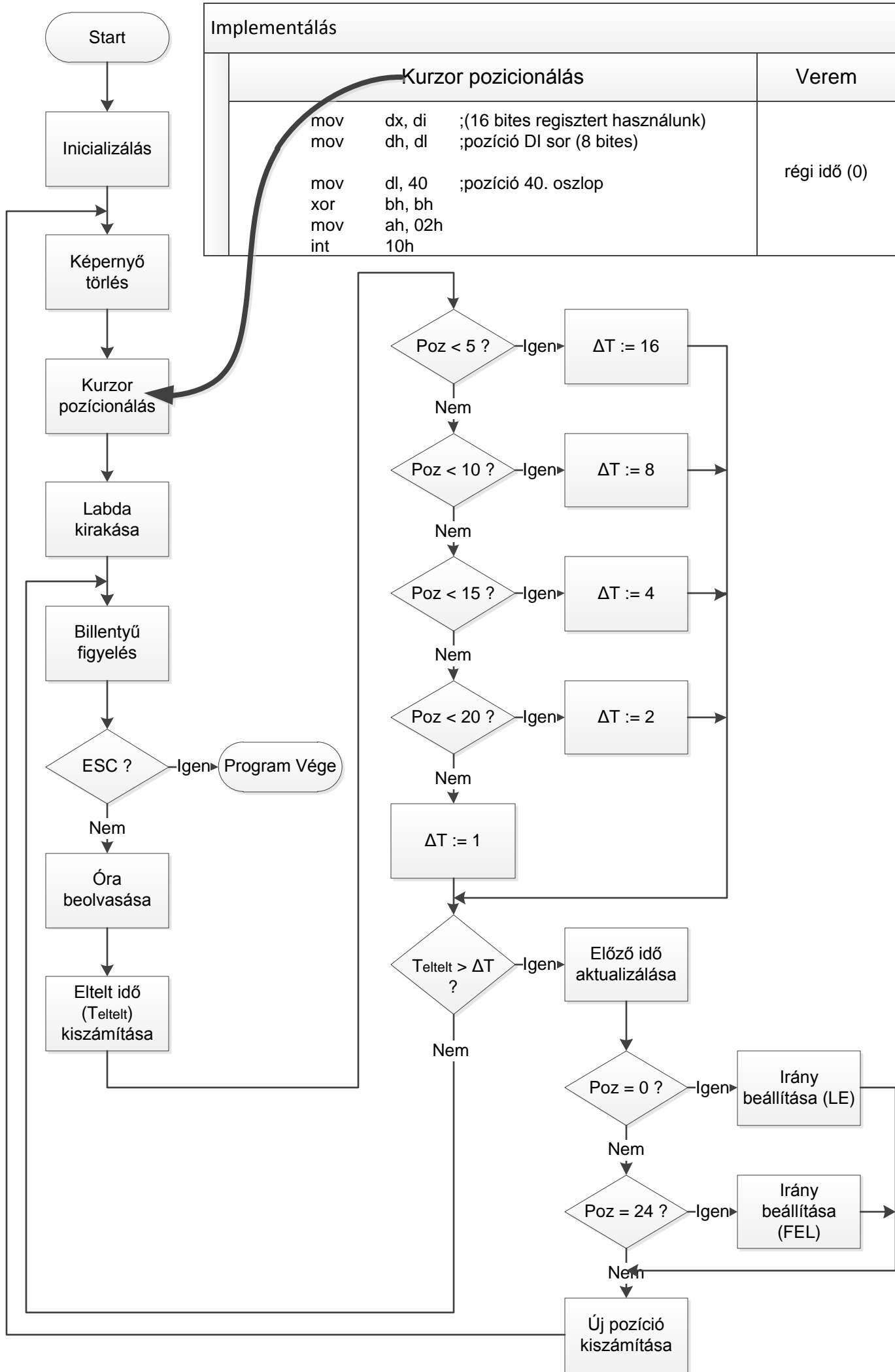
Nem

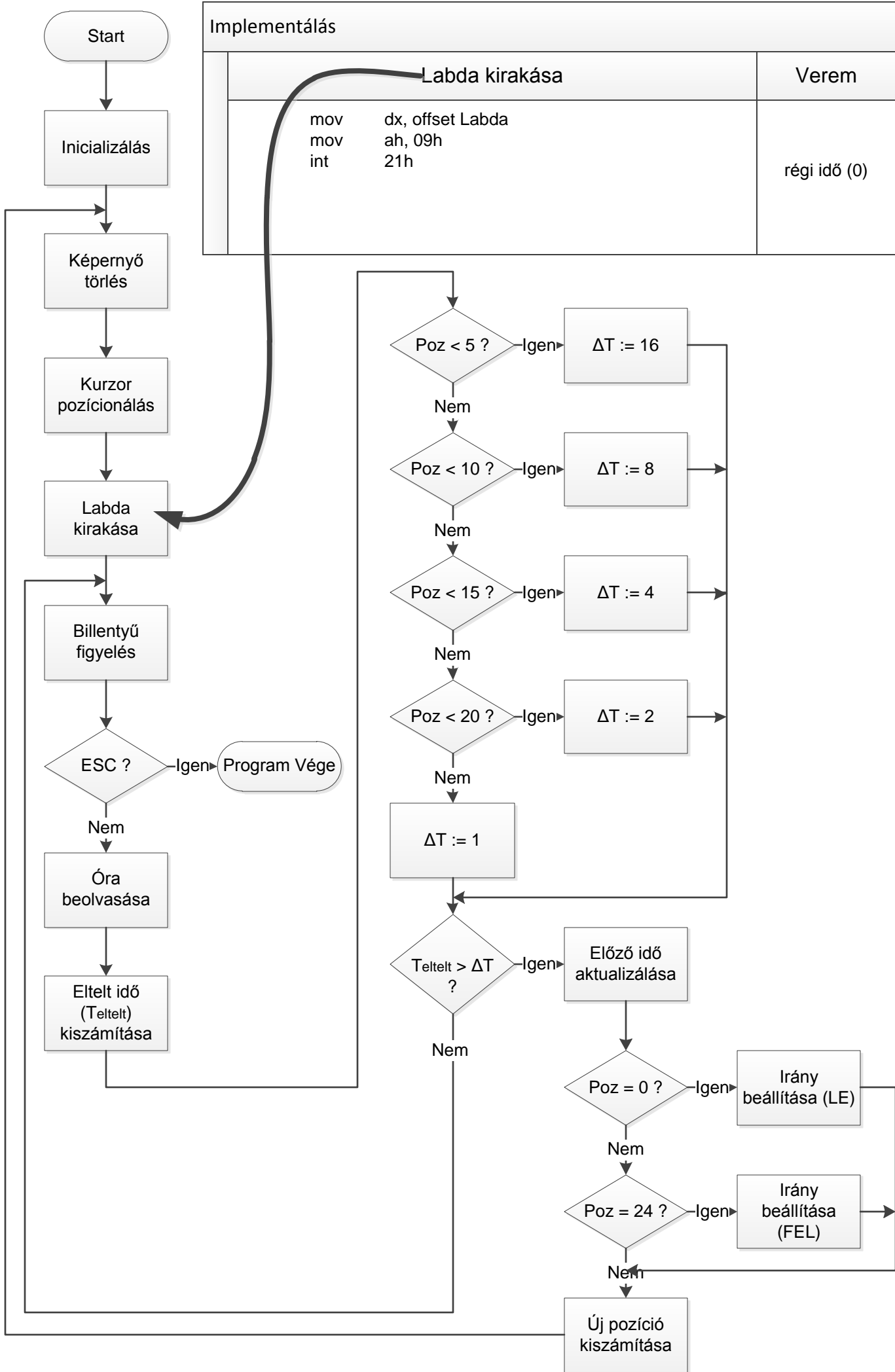
Új pozíció  
kiszámítása





Implementálás		Verem
Képernyő törlés		
Torles: mov ax, 03h int 10h		régi idő (0)





Implementálás		Verem
Labda kirakása		
mov dx, offset Labda		
mov ah, 09h		régi idő (0)
int 21h		

Start

Inicializálás

Képernyő törlés

Kurzor pozícionálás

Labda kirakása

Kesleltet

Billentyű figyelés

ESC ?

Igen Program Vége

Nem

Óra beolvasása

Eltelt idő (Teltelt) kiszámítása

### Implementálás

#### Billentyű figyelés (1)

```

Kesleltet:
    mov     ah, 00h
    int    16h
;időzítő nélküli eset

```

#### Billentyű figyelés (2)

```

mov     ah, 01h
int    16h
;ha van leütött billentyű, tehát
nem üres a billentyűzet puffer,
akkor Z flag értéke 0

```

#### Verem

régi idő (0)

Poz < 5 ?

Igen

$\Delta T := 16$

Nem

Poz < 10 ?

Igen

$\Delta T := 8$

Nem

Poz < 15 ?

Igen

$\Delta T := 4$

Nem

Poz < 20 ?

Igen

$\Delta T := 2$

Nem

$\Delta T := 1$

Teltelt >  $\Delta T$  ?

Igen

Előző idő aktualizálása

Nem

Poz = 0 ?

Igen

Irány beállítása (LE)

Nem

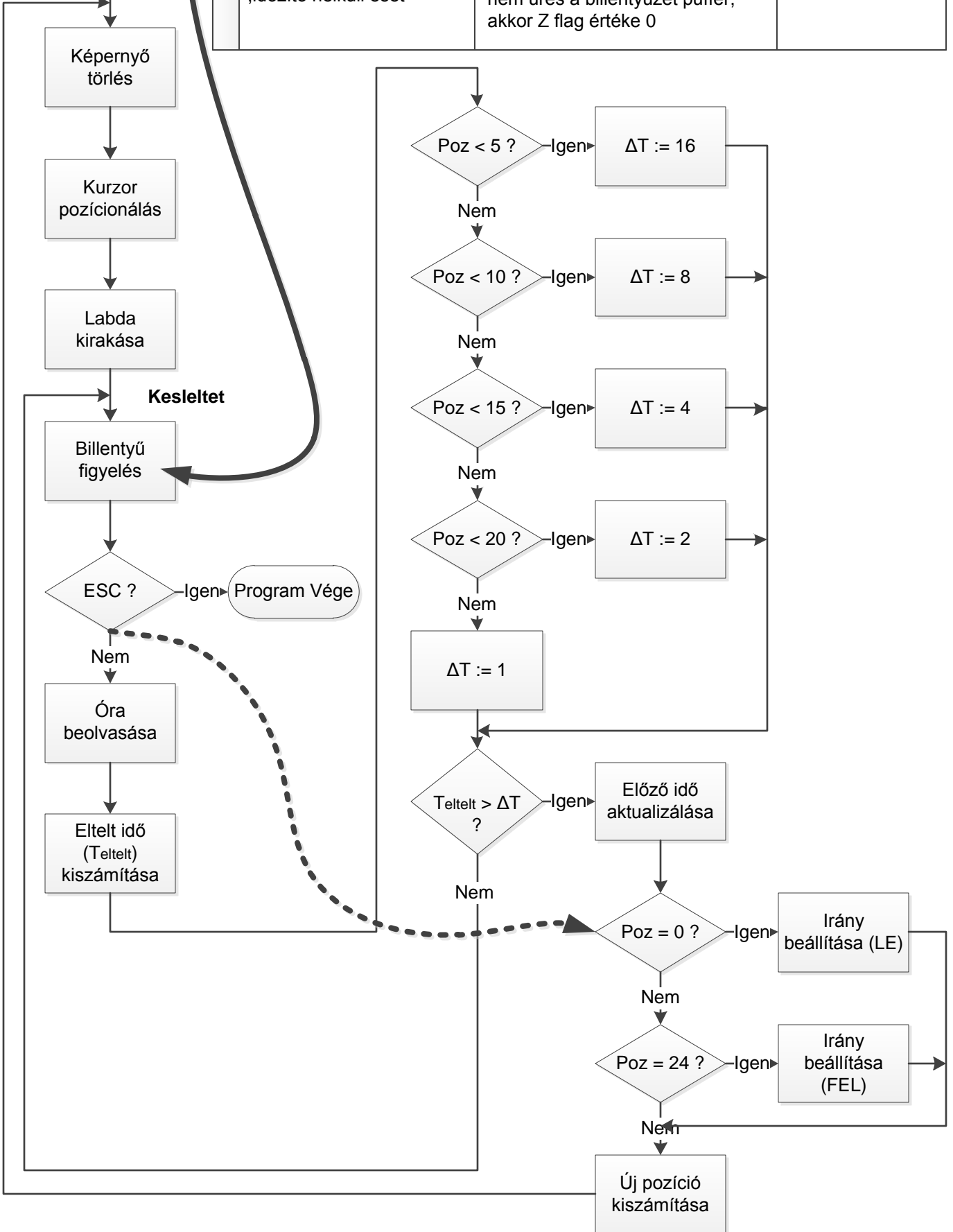
Poz = 24 ?

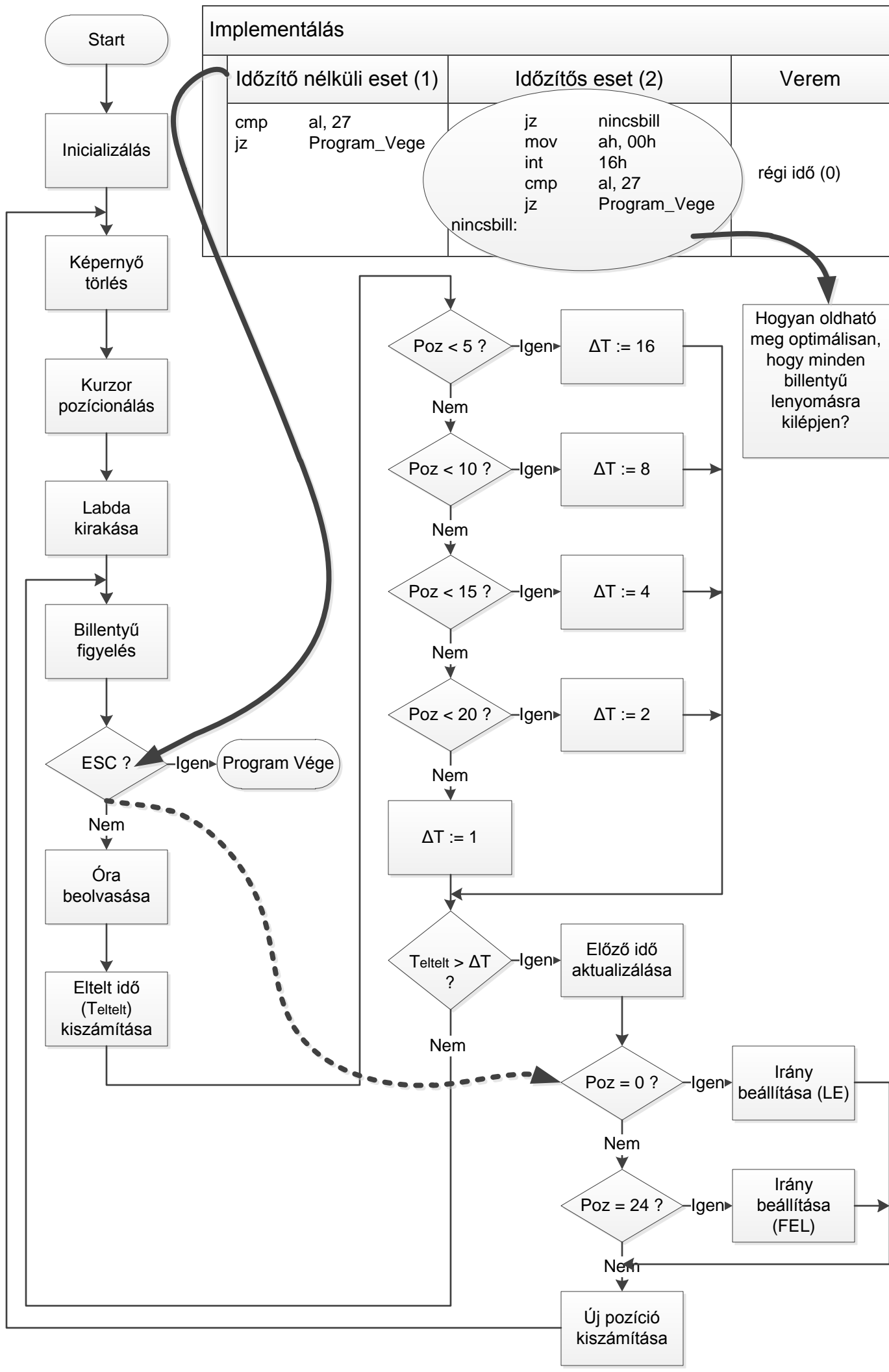
Igen

Irány beállítása (FEL)

Nem

Új pozíció kiszámítása







Start

Inicializálás

Képernyő  
törlés

Kurzor  
pozícionálás

Labda  
kirakása

Billentyű  
figyelés

ESC ?

Igen Program Vége

Óra  
beolvasása

Eltelt idő  
(Teltelt)  
kiszámítása

### Implementálás

Óra beolvasása

Verem

```
xor ah, ah
int 1ah ;óra beolvasása CX:DX -be
```

régi idő (0)

Poz < 5 ?

Igen

$\Delta T := 16$

Nem

Poz < 10 ?

Igen

$\Delta T := 8$

Nem

Poz < 15 ?

Igen

$\Delta T := 4$

Nem

Poz < 20 ?

Igen

$\Delta T := 2$

Nem

$\Delta T := 1$

Teltelt >  $\Delta T$  ?

Igen

Előző idő  
aktualizálása

Nem

Poz = 0 ?

Igen

Irány  
beállítás (LE)

Nem

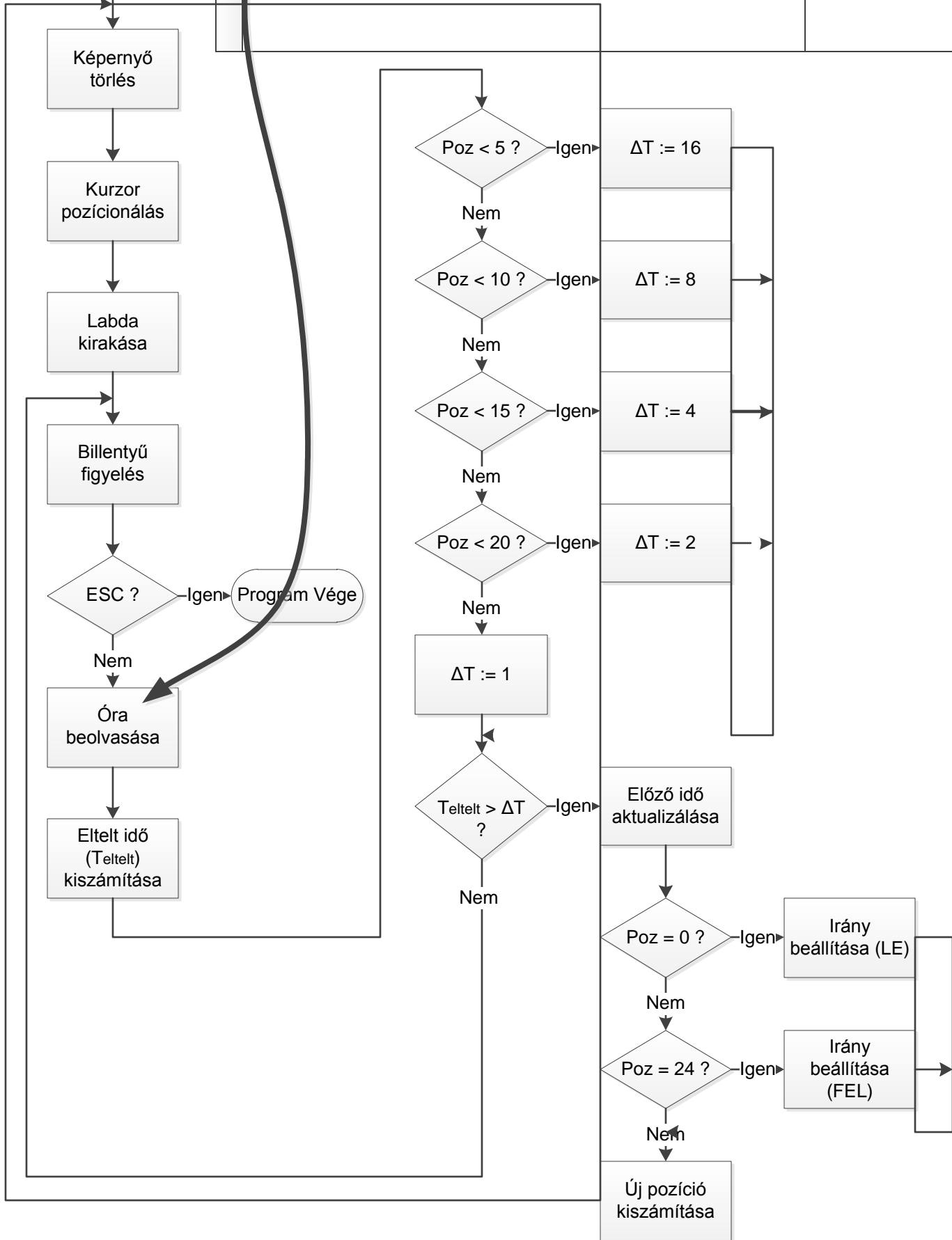
Poz = 24 ?

Igen

Irány  
beállítás (FEL)

Nem

Új pozíció  
kiszámítása



Start

Inicializálás

Képernyő  
törlés

Kurzor  
pozícionálás

Labda  
kírákása

Billentyű  
figyelés

ESC ?

Igen Program Vége

Nem

Óra  
beolvasása

Eltelt idő  
(Teltelt)  
kiszámítása

### Implementálás

Eltelt idő kiszámítása (Teltelt)

Verem

pop	cx	;régi idő kivétele	
push	cx	;régi idő visszatétele	
push	dx	;aktuális idő mentése ax-be	
sub	dx, cx	;dx-ben a Teltelt = Taktuális - Trégi	
		;aktuális idő mentése a verembe	aktuális idő régi idő (0)

Poz < 5 ?

Igen

$\Delta T := 16$

Nem

Poz < 10 ?

Igen

$\Delta T := 8$

Nem

Poz < 15 ?

Igen

$\Delta T := 4$

Nem

Poz < 20 ?

Igen

$\Delta T := 2$

Nem

$\Delta T := 1$

Teltelt >  $\Delta T$  ?

Igen

Előző idő  
aktualizálása

Nem

Poz = 0 ?

Igen

Irány  
beállítás (LE)

Nem

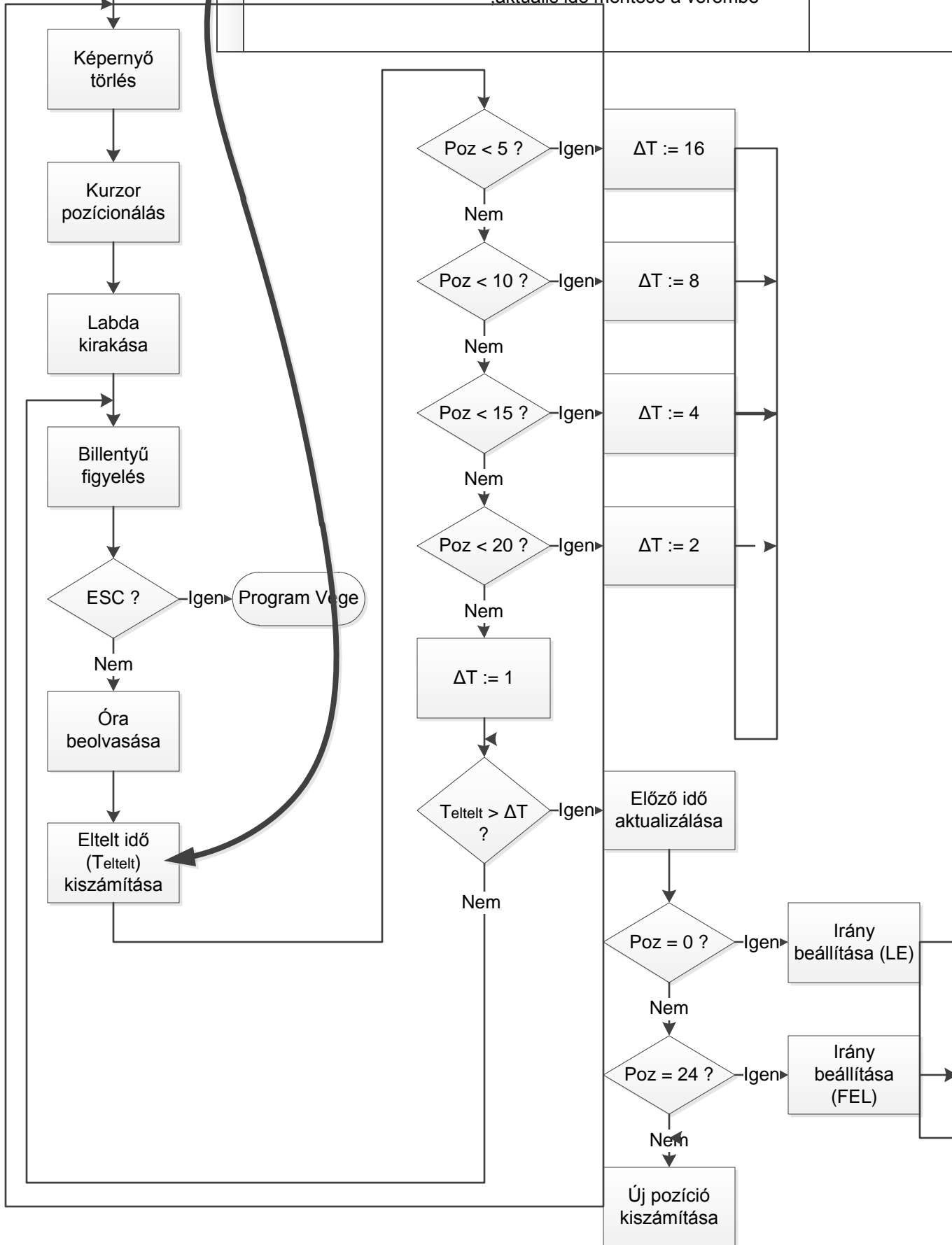
Poz = 24 ?

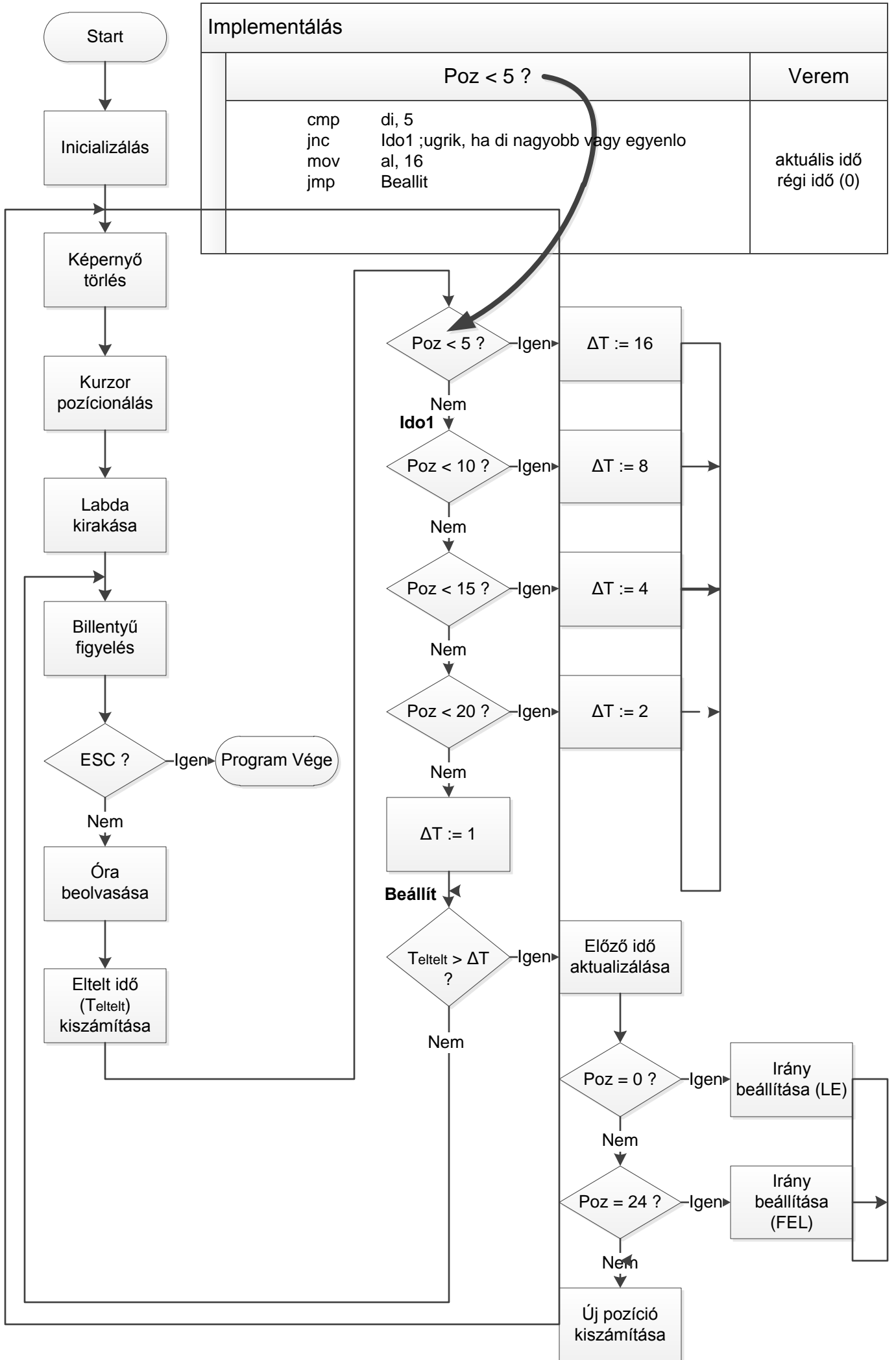
Igen

Irány  
beállítás (FEL)

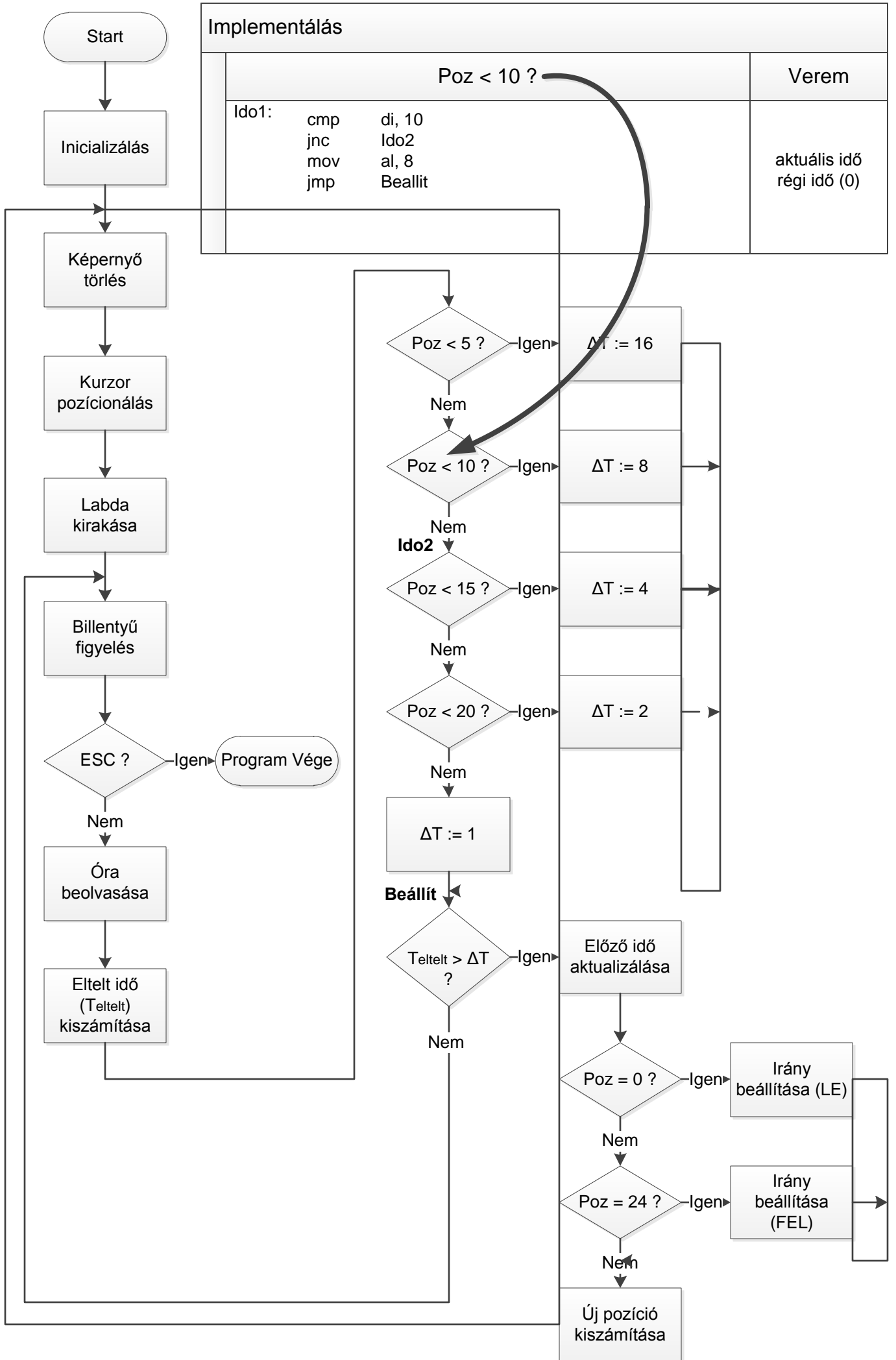
Nem

Új pozíció  
kiszámítása

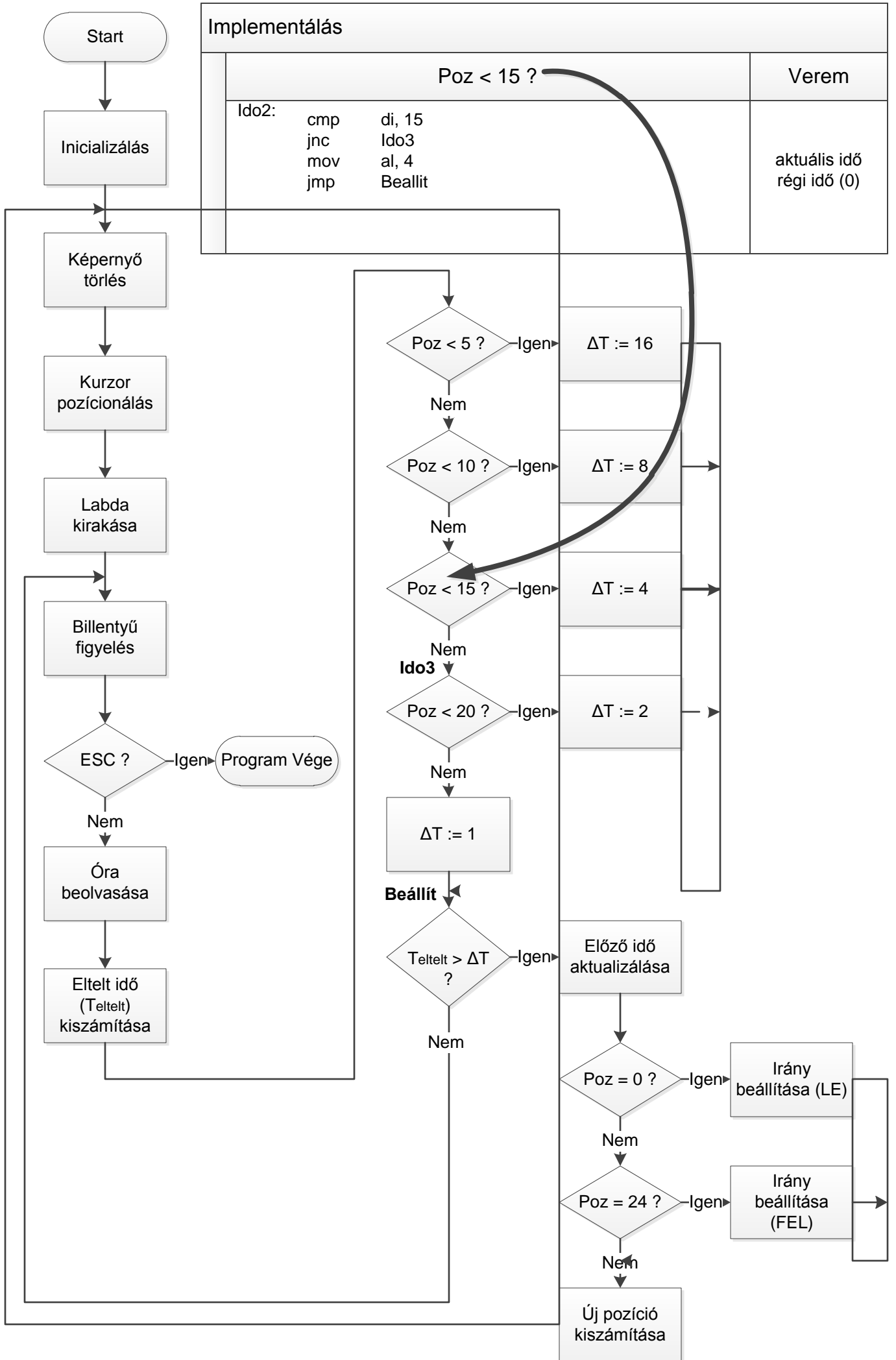




Implementálás		Verem
	Poz < 5 ?	
cmp di, 5 jnc Ido1 mov al, 16 jmp Beallit		aktuális idő régi idő (0)



Implementálás				Verem
Ido1:	cmp	di, 10		aktuális idő régi idő (0)
	jnc	Ido2		
	mov	al, 8		
	jmp	Beallit		



Implementálás		Verem
Poz < 15 ?		aktuális idő régi idő (0)
ldo2:	<pre> cmp    di, 15 jnc    ldo3 mov    al, 4 jmp    Beallit </pre>	

Start

Inicializálás

Képernyő törlés

Kurzor pozícionálás

Labda kirakása

Billentyű figyelés

ESC ?

Igen Program Vége

Nem

Óra beolvasása

Eltelt idő (Teltelt) kiszámítása

### Implementálás

Poz < 20 ?

Verem

```

Ido3:  cmp    di, 20
       jnc    Ido4
       mov    al, 2
       jmp    Beallit

```

aktuális idő  
régi idő (0)

Poz < 5 ?

Igen

$\Delta T := 16$

Nem

Poz < 10 ?

Igen

$\Delta T := 8$

Nem

Poz < 15 ?

Igen

$\Delta T := 4$

Nem

Poz < 20 ?

Igen

$\Delta T := 2$

Nem

Ido4

$\Delta T := 1$

Beállít

Teltelt >  $\Delta T$  ?

Igen

Előző idő aktualizálása

Nem

Poz = 0 ?

Igen

Irány beállítása (LE)

Nem

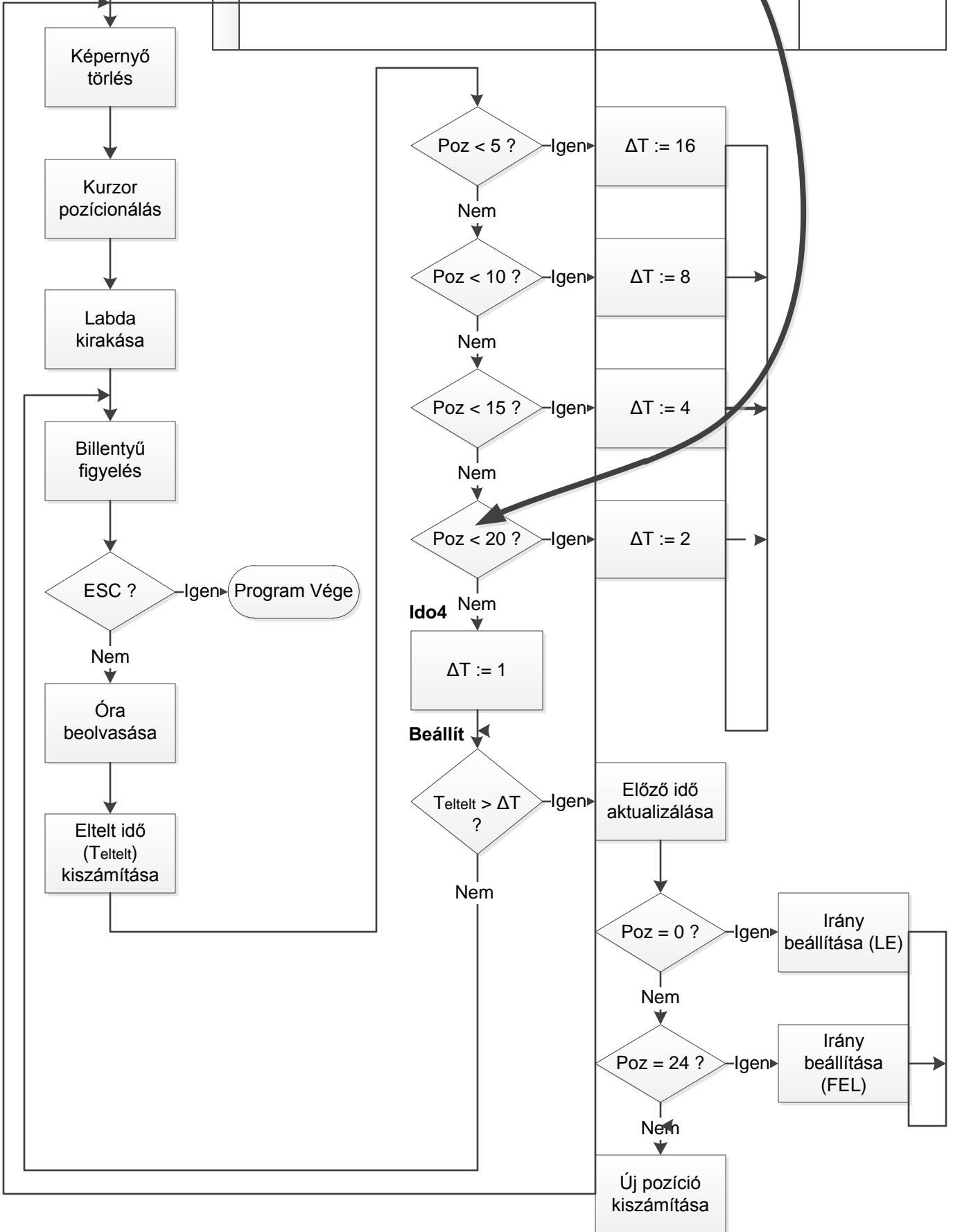
Poz = 24 ?

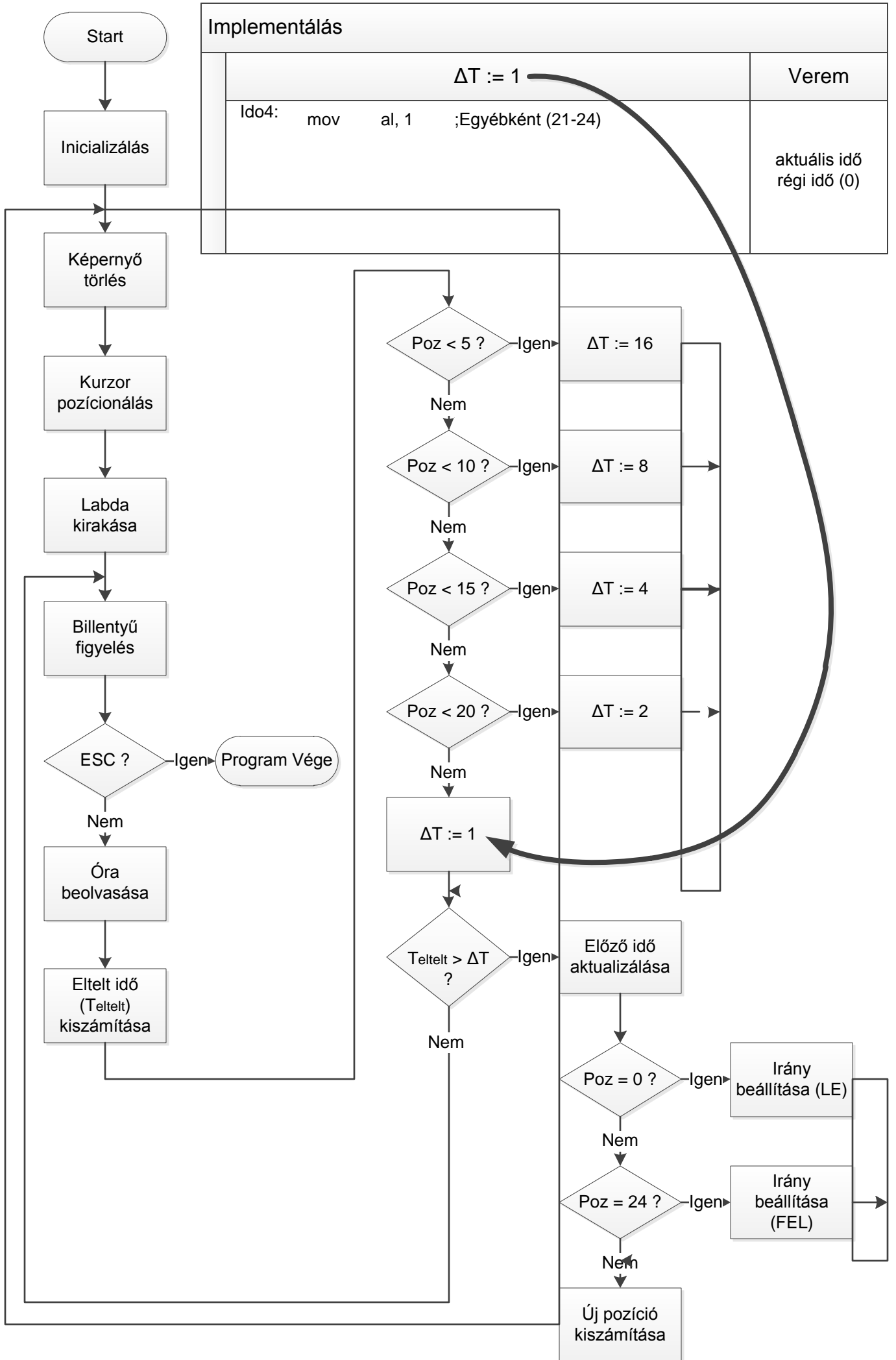
Igen

Irány beállítása (FEL)

Nem

Új pozíció kiszámítása





Implementálás		Verem
$\Delta T := 1$		
Ido4:	mov al, 1 ;Egyébként (21-24)	aktuális idő régi idő (0)

Start

Inicializálás

Képernyő  
törlés

Kurzor  
pozícionálás

Labda  
kirakása

**Kesleltet**

Billentyű  
figyelés

ESC ?

Igen → Program Vége

Nem

Óra  
beolvasása

Eltelt idő  
(Teltet)  
kiszámítása

### Implementálás

Teltet > ΔT ?

Verem

Beállít:

```

xor    ah, ah
cmp    dx, ax
jc     Kesleltet ;ugrik, ha kisebb

```

aktuális idő  
régi idő (0)

Poz < 5 ?

Igen

ΔT := 16

Nem

Poz < 10 ?

Igen

ΔT := 8

Nem

Poz < 15 ?

Igen

ΔT := 4

Nem

Poz < 20 ?

Igen

ΔT := 2

Nem

ΔT := 1

**Beállít**

Teltet > ΔT ?

Igen

Előző idő  
aktualizálása

Nem

Poz = 0 ?

Igen

Irány  
beállítása (LE)

Nem

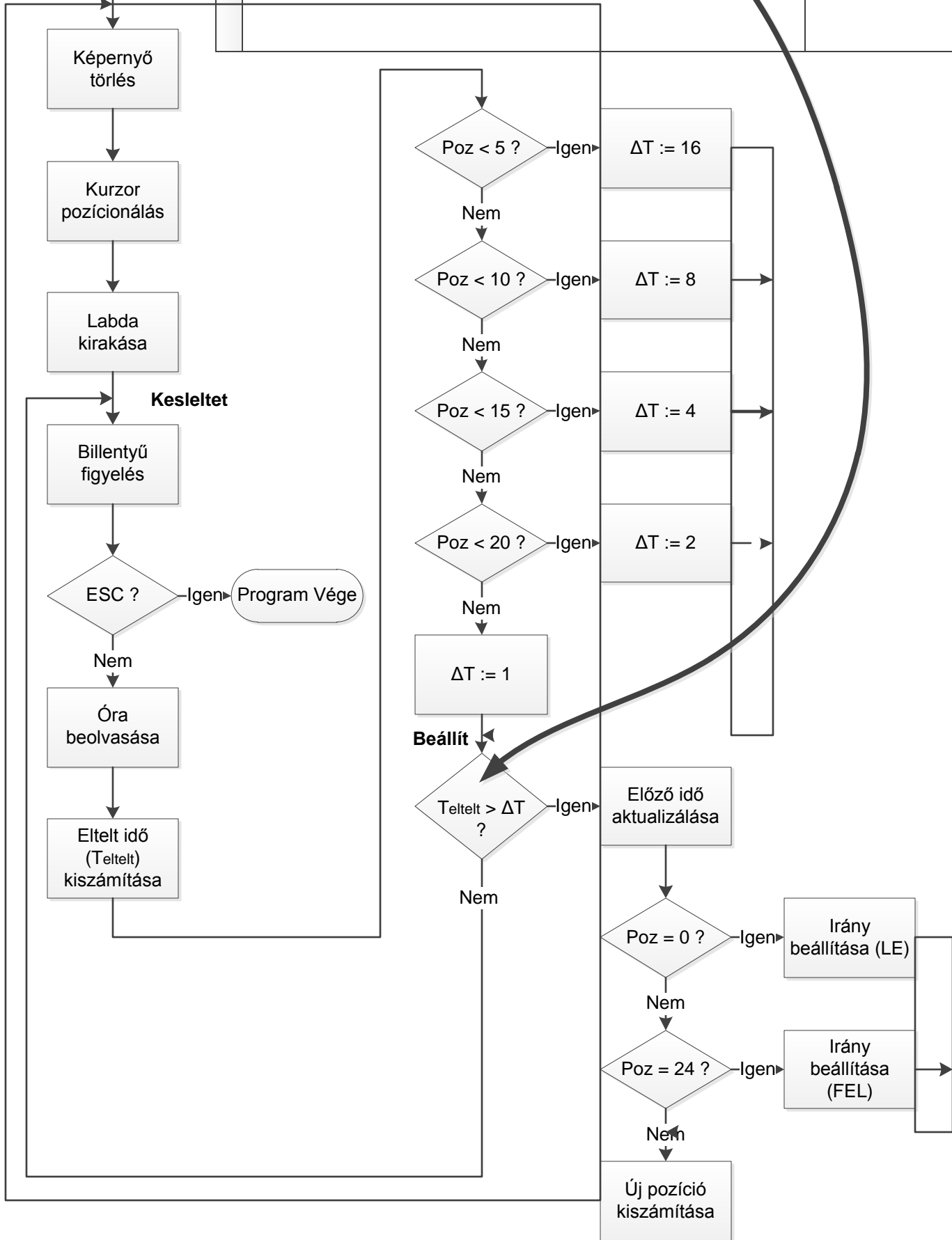
Poz = 24 ?

Igen

Irány  
beállítása (FEL)

Nem

Új pozíció  
kiszámítása





Start

Inicializálás

Képernyő törlés

Kurzor pozicionálás

Labda kirakása

Billentyű figyelés

ESC ?

Igen -> Program Vége

Nem -> Óra beolvasása

Eltelt idő (Teltelt) kiszámítása

### Implementálás

Előző idő aktualizálása			Verem
pop	ax	;előző idő aktualizálásához kell	régi idő (x)
pop	cx	;régi idő kivétele a veremből	
push	ax	;aktuális idő elmentése	

Poz < 5 ?

Igen ->  $\Delta T := 16$

Nem -> Poz < 10 ?

Poz < 10 ?

Igen ->  $\Delta T := 8$

Nem -> Poz < 15 ?

Poz < 15 ?

Igen ->  $\Delta T := 4$

Nem -> Poz < 20 ?

Poz < 20 ?

Igen ->  $\Delta T := 2$

Nem ->  $\Delta T := 1$

$\Delta T := 1$

Teltelt >  $\Delta T$  ?

Igen -> Előző idő aktualizálása

Nem -> Poz = 0 ?

Poz = 0 ?

Igen -> Irány beállítása (LE)

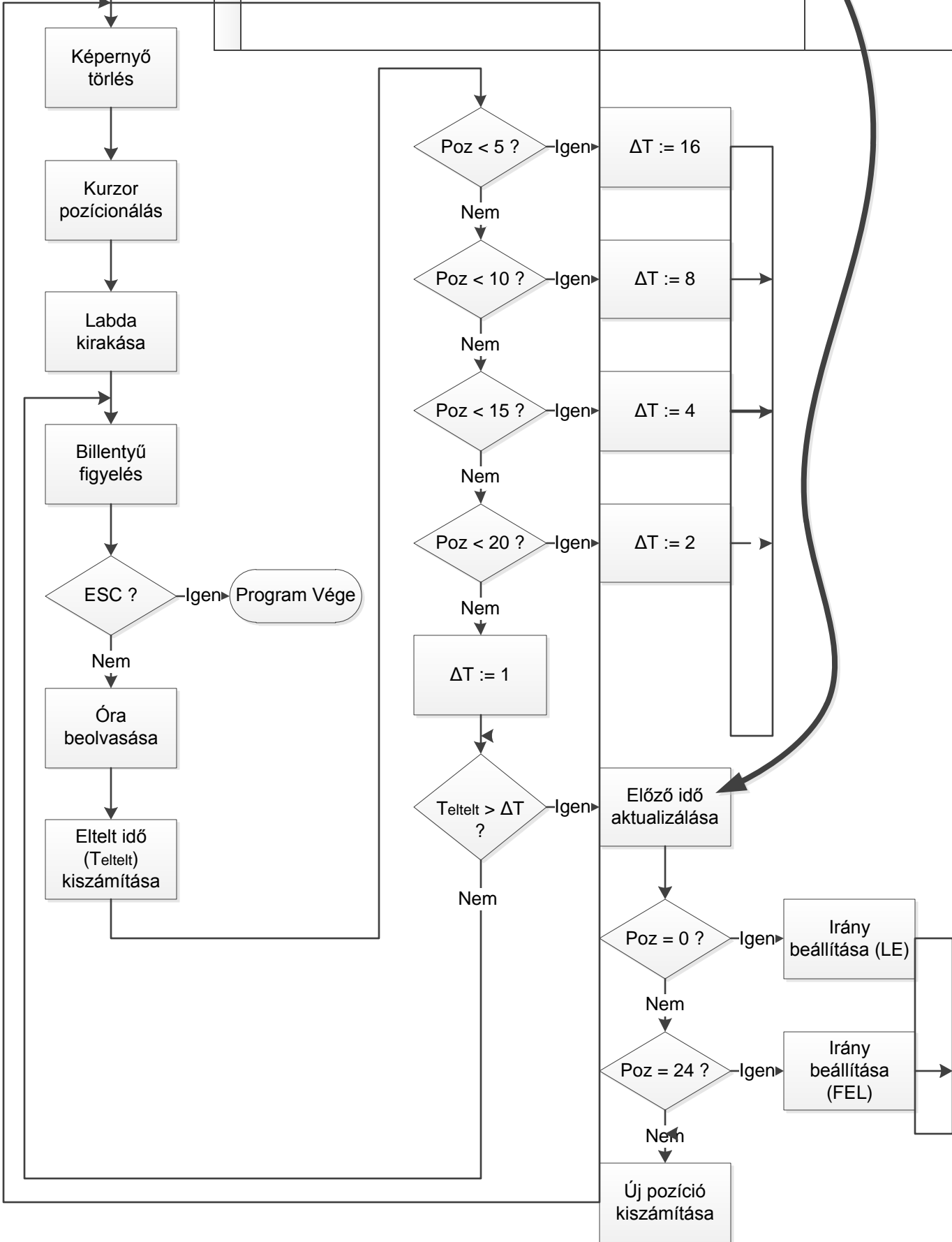
Nem -> Poz = 24 ?

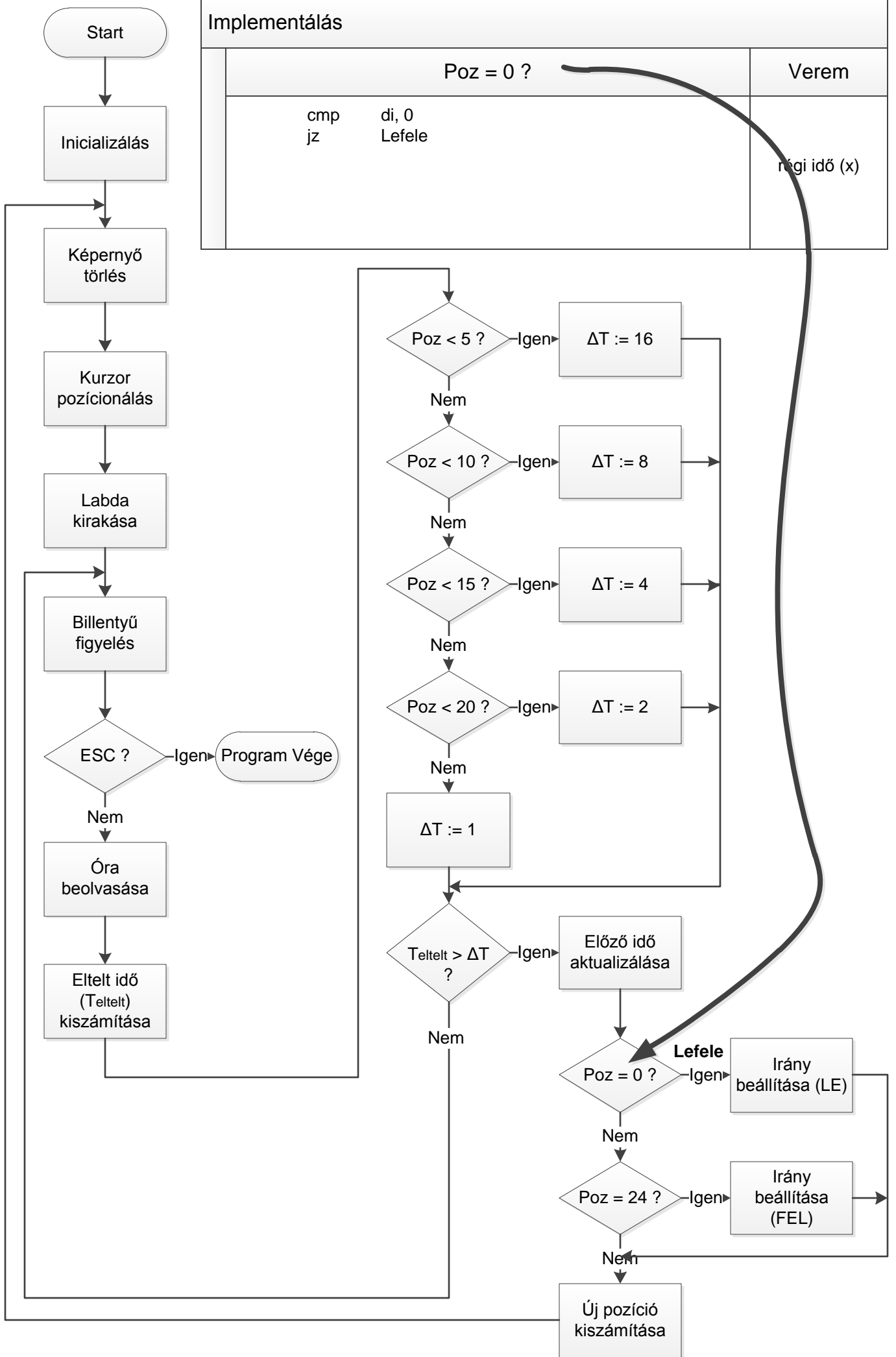
Poz = 24 ?

Igen -> Irány beállítása (FEL)

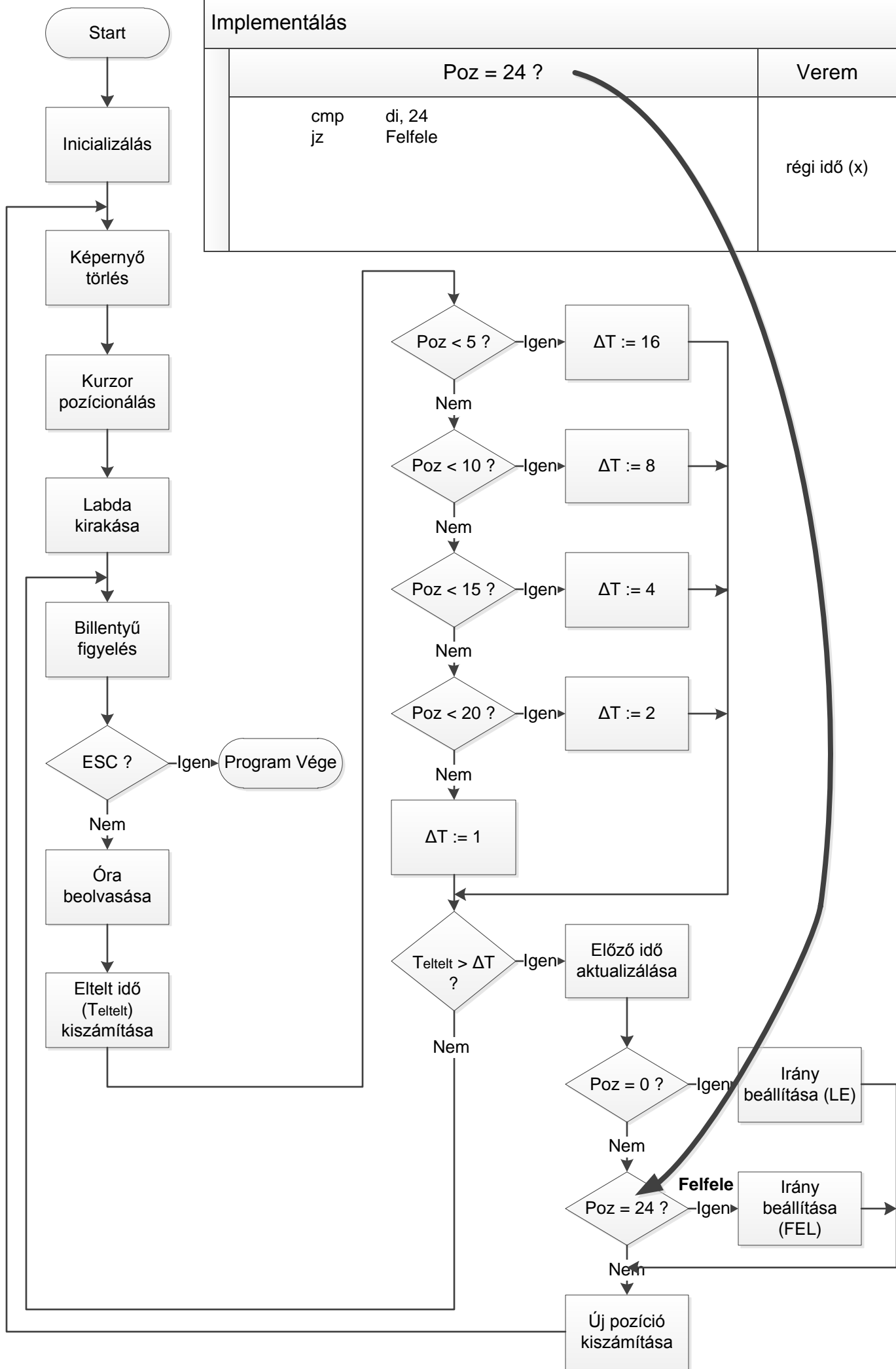
Nem -> Új pozíció kiszámítása

Új pozíció kiszámítása

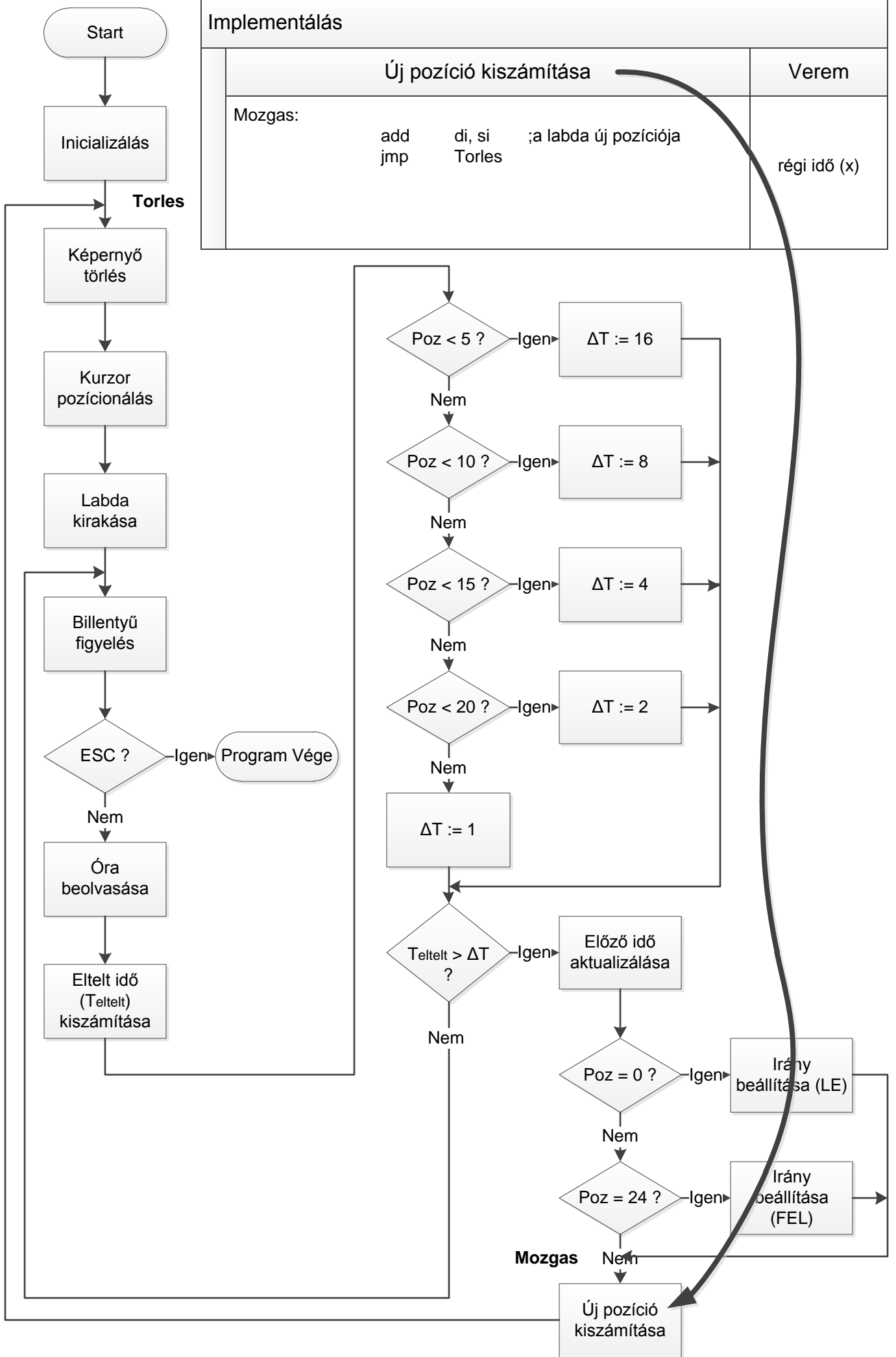




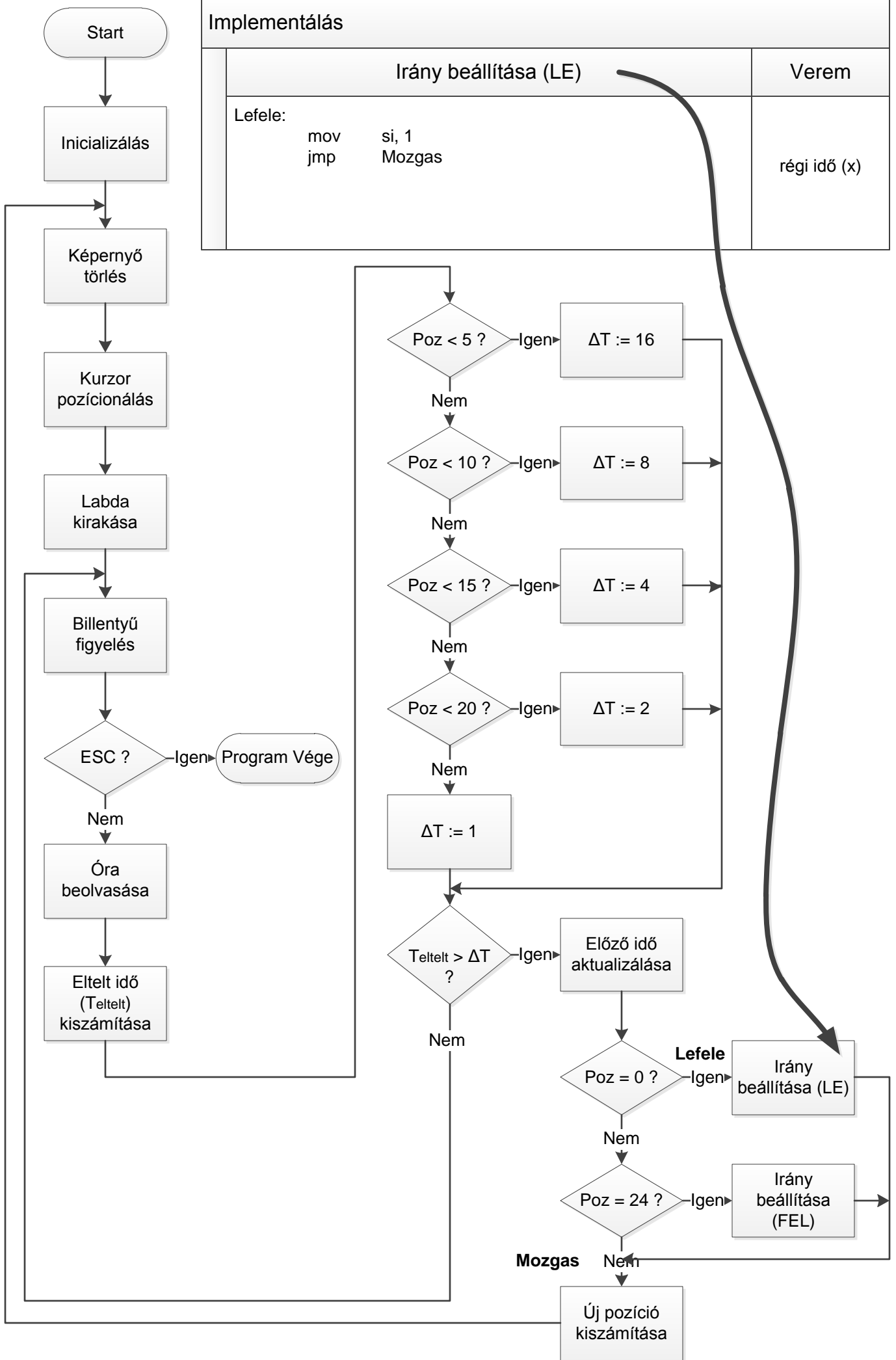
Implementálás		Verem
Poz = 0 ?		régi idő (x)
cmp	di, 0	
jz	Lefe	

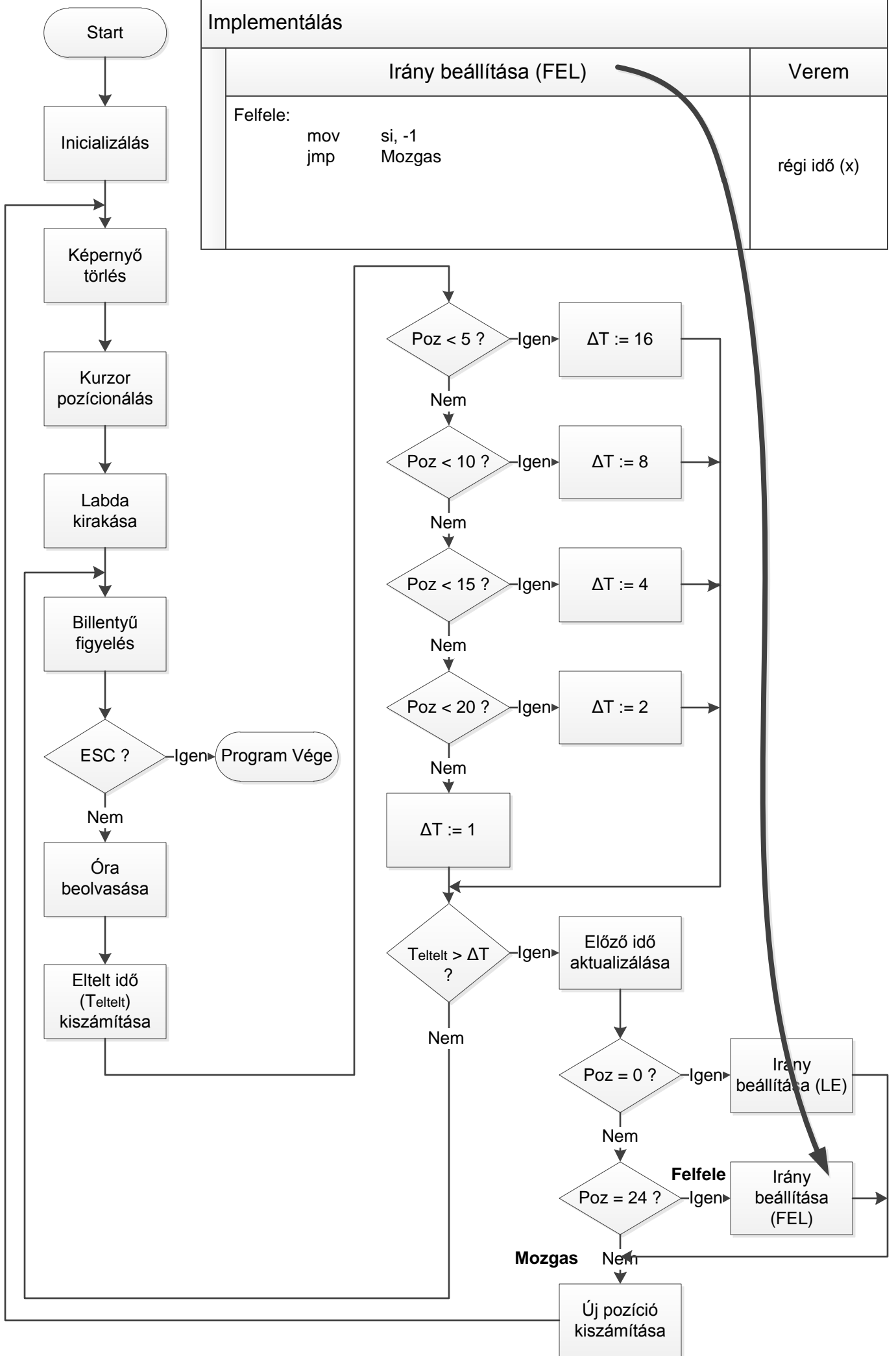


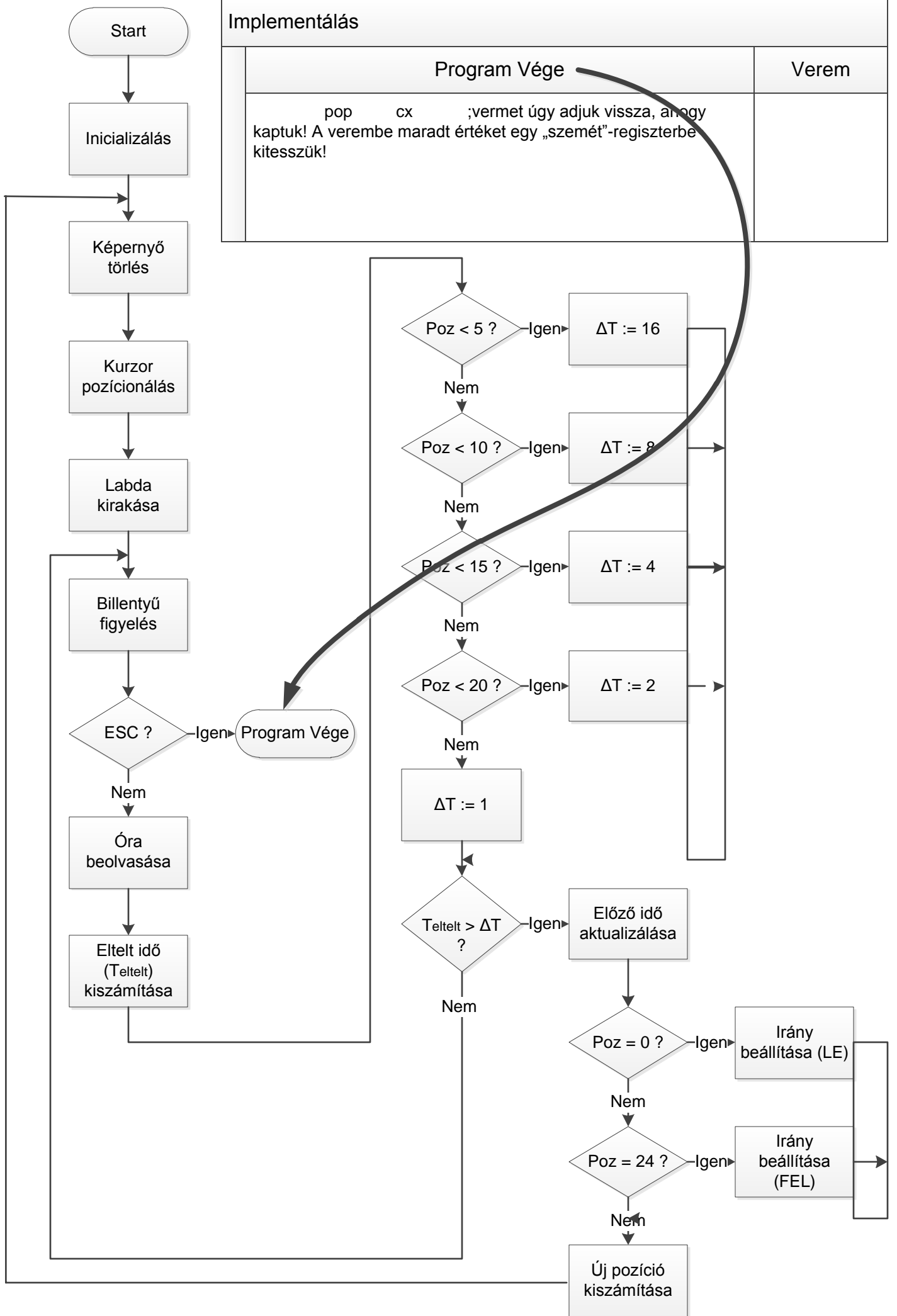
Implementálás		Verem
Poz = 24 ?		
cmp	di, 24	
jz	Felfele	régi idő (x)



Implementálás			Verem
Új pozíció kiszámítása			
Mozgas:	add jmp	di, si Torles	;a labda új pozíciója  régi idő (x)







Implementálás	
Program Vége	Verem
<pre> pop     cx     ;vermet úgy adjuk vissza, ahogy kaptuk! A verembe maradt értéket egy „szemét”-regiszterbe kitesszük! </pre>	