Feladat6: pi.asm

Fázis

Feladat meghatározása

A program kiszámítja a PI értékét négy tizedes pontosan. Ebből azonban a negyedik tizedes érték hibás 3.1416

A számítás alapja:
PI/4 = 1 - 1/3 + 1/5 - 1/7 + 1/9 -+…

Mivel a program egész értékes számábrázolást használ, ezért 10000-rel felszorozzuk a sorozat minden elemét.

Kiírásnál az egész részt (3) követően kirakunk egy tizedespontot.

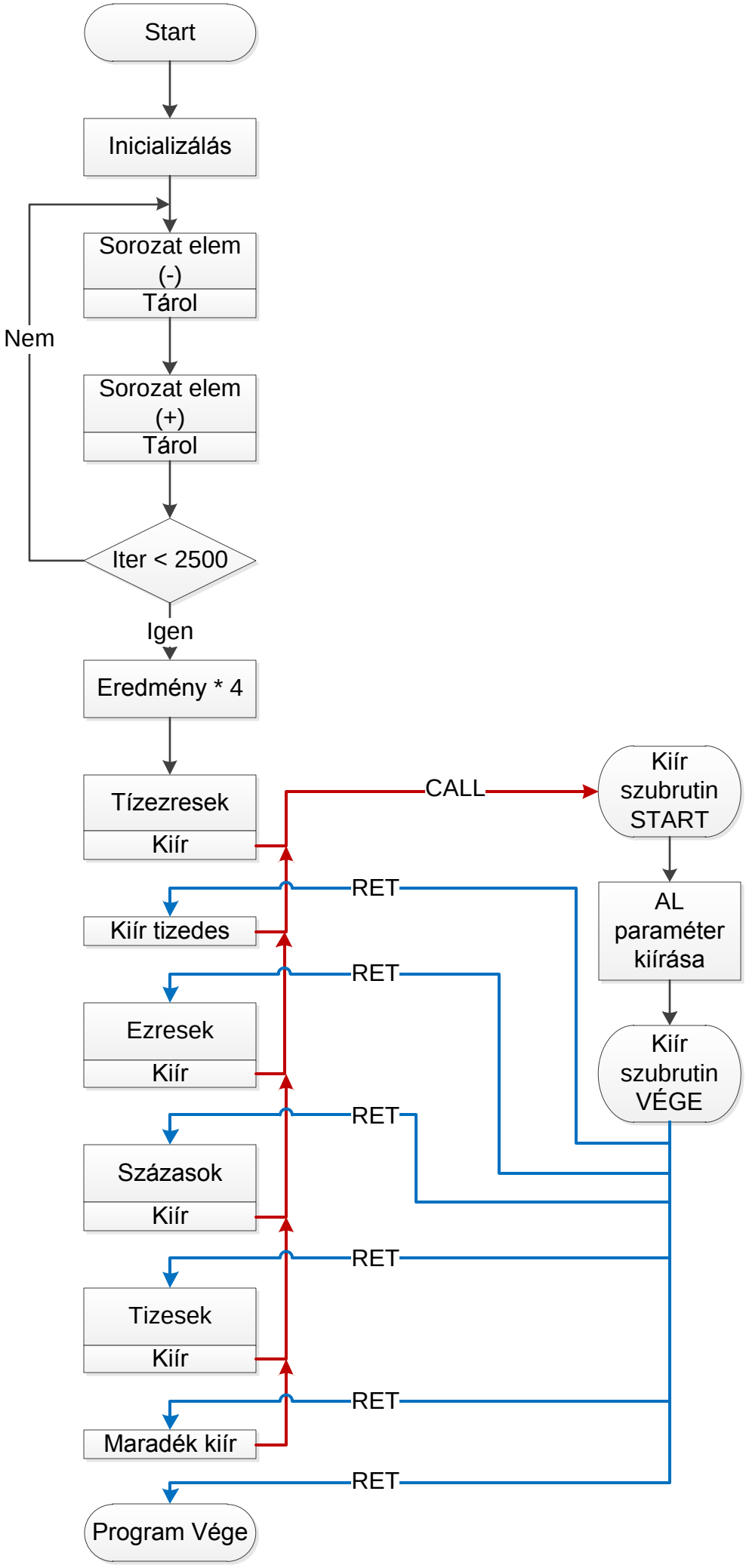
Figyeljük meg, hogy az .exe fájlunk mérete 2 bájttal csökken ebben az esetben!

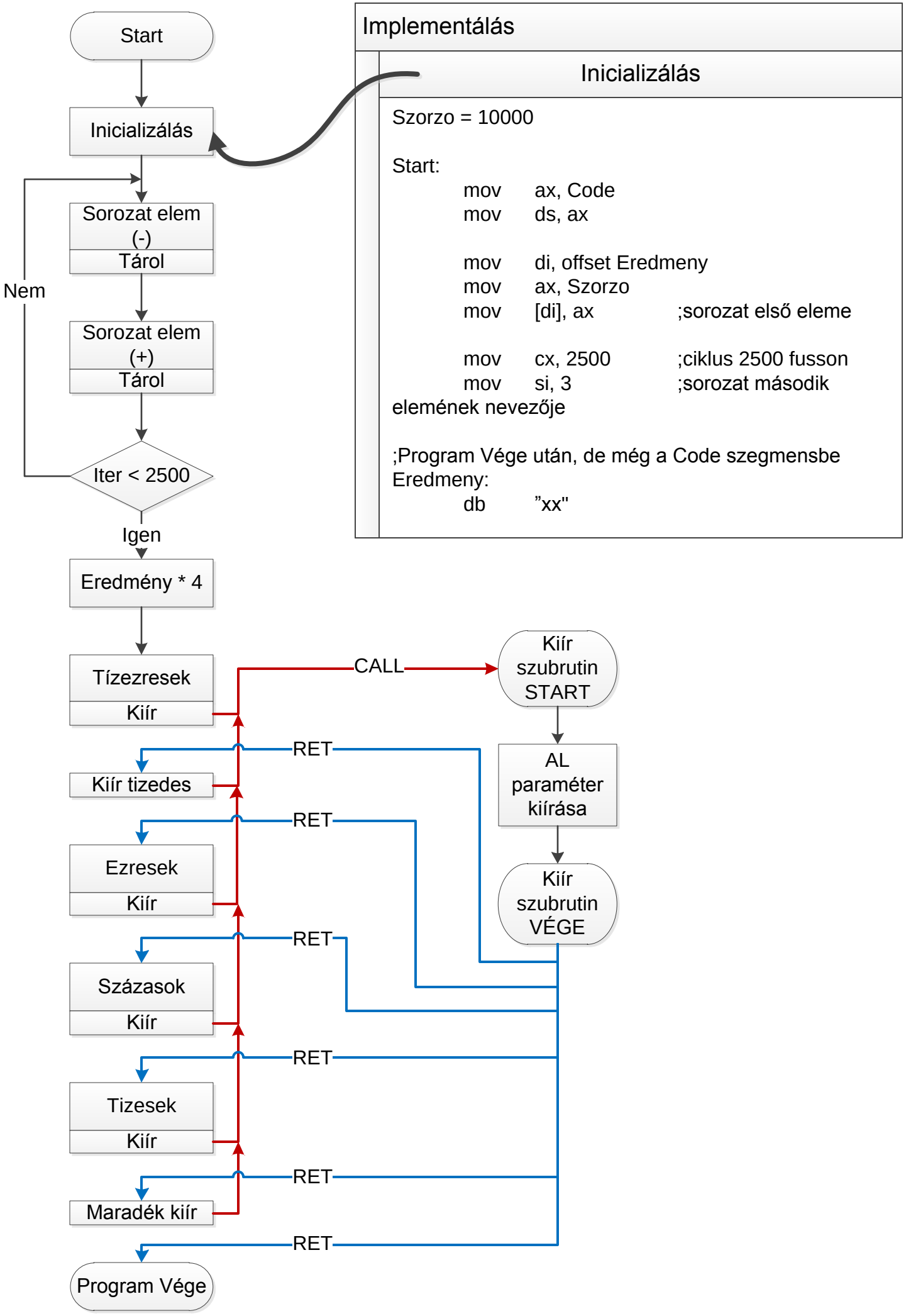
Implementálás

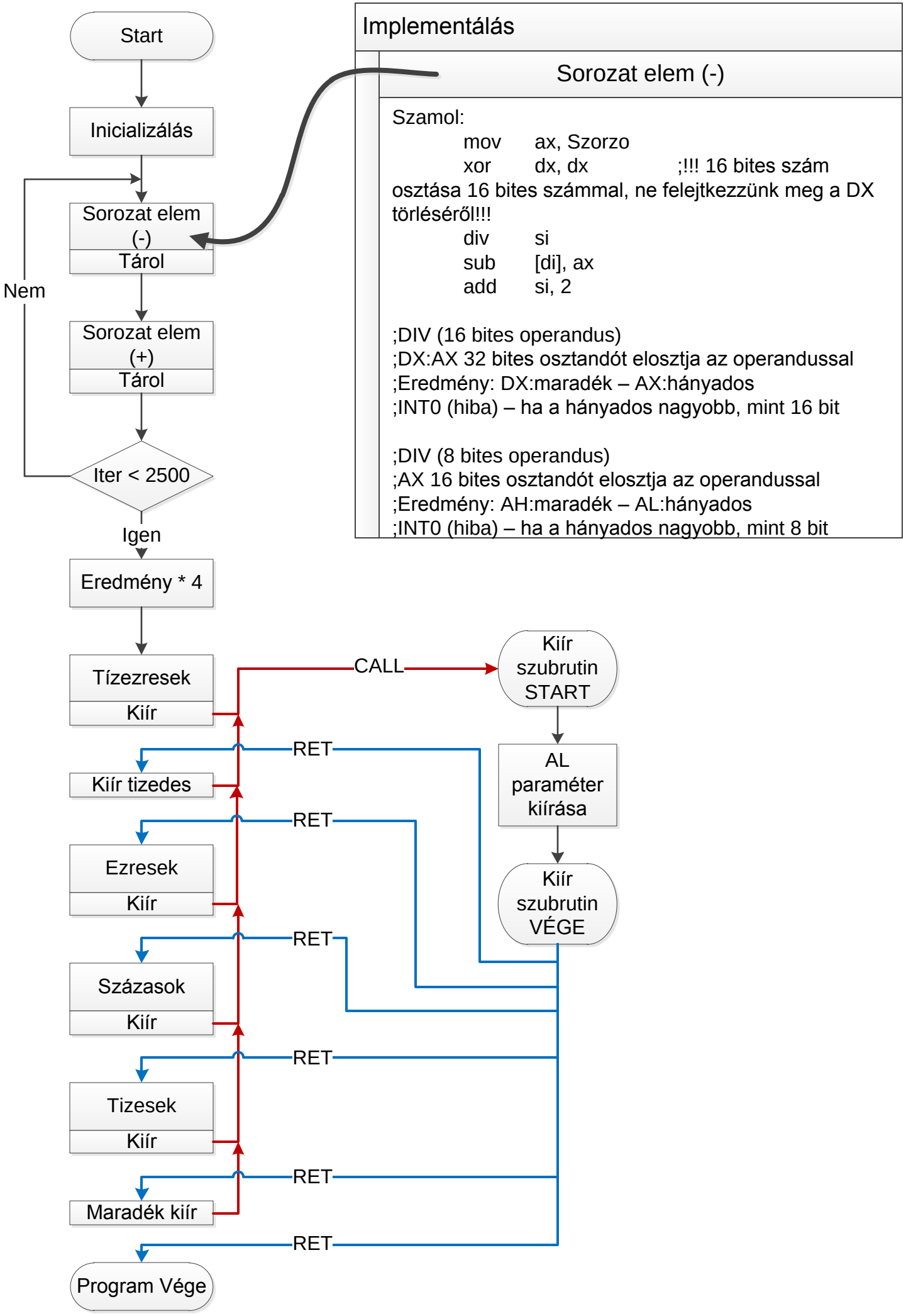
```
Code    Segment
        assume CS:Code,
        DS:Data, SS:Stack

Szorzo = 10000

Start:
        mov     ax, Code
        mov     DS, AX

        mov     di, offset Eredmeny
        mov     ax, Szorzo
        mov     [di], ax

        mov     cx, 2500
        mov     si, 3
Szamol:
        mov     ax, Szorzo
        xor     dx, dx
        div     si
        sub     [di], ax
        add     si, 2

        mov     ax, Szorzo
        xor     dx, dx
        div     si
        add     [di], ax
        add     si, 2

        loop    Szamol

        mov     ax, [di]
        ;shl    ax, 1
        ;shl    ax, 1
        mov     cl, 2
        shl     ax, cl

        xor     dx, dx
        mov     bx, 10000
        div     bx
        push    dx

        call    Kiir

        mov     al, "."
        sub     al, 48

        call    Kiir

        pop     ax
        xor     dx, dx
        mov     bx, 1000
        div     bx
        push    dx

        call    Kiir

        pop     ax
        xor     dx, dx
        mov     bx, 100
        div     bx
        push    dx

        call    Kiir
```

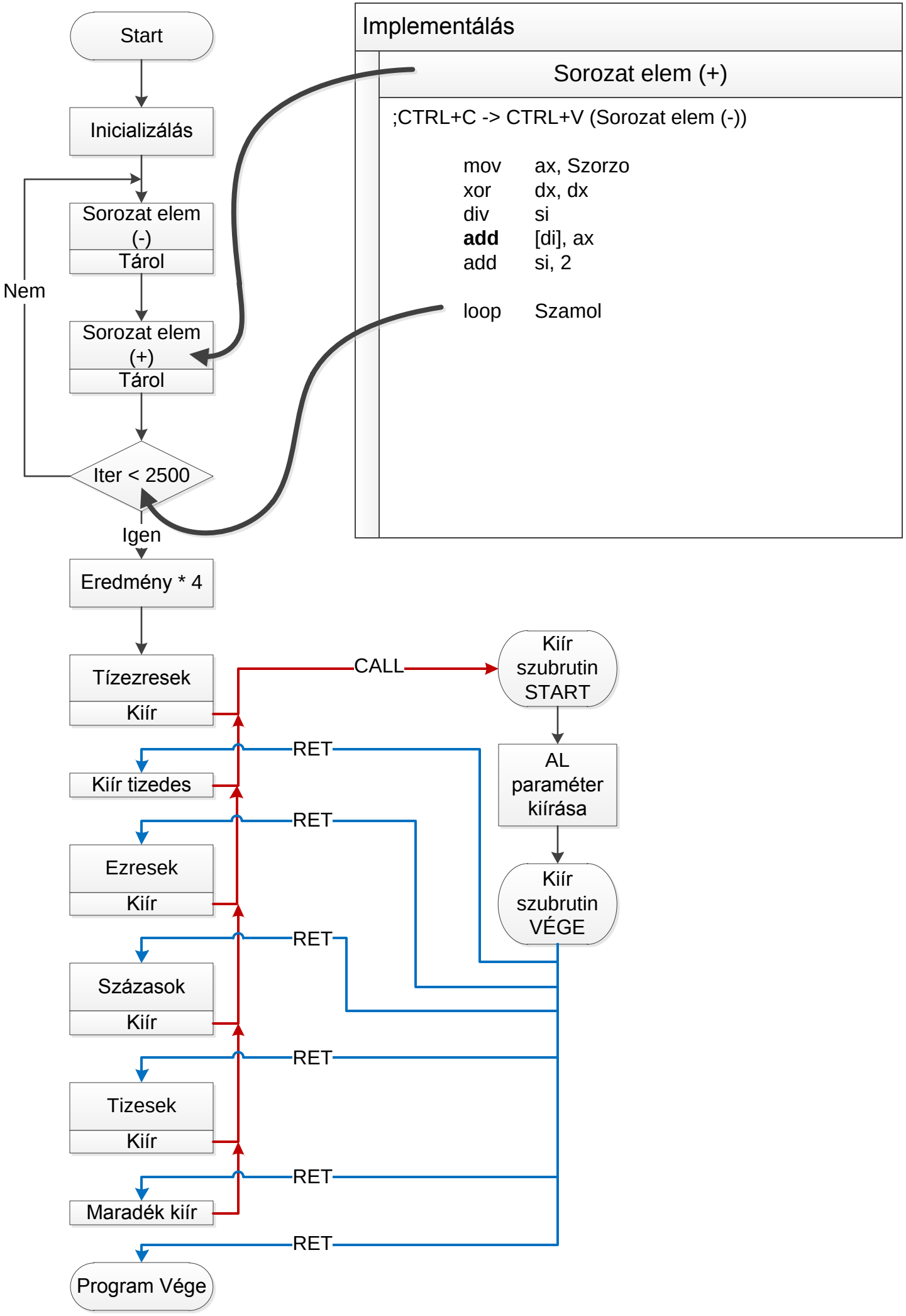
Implementálás

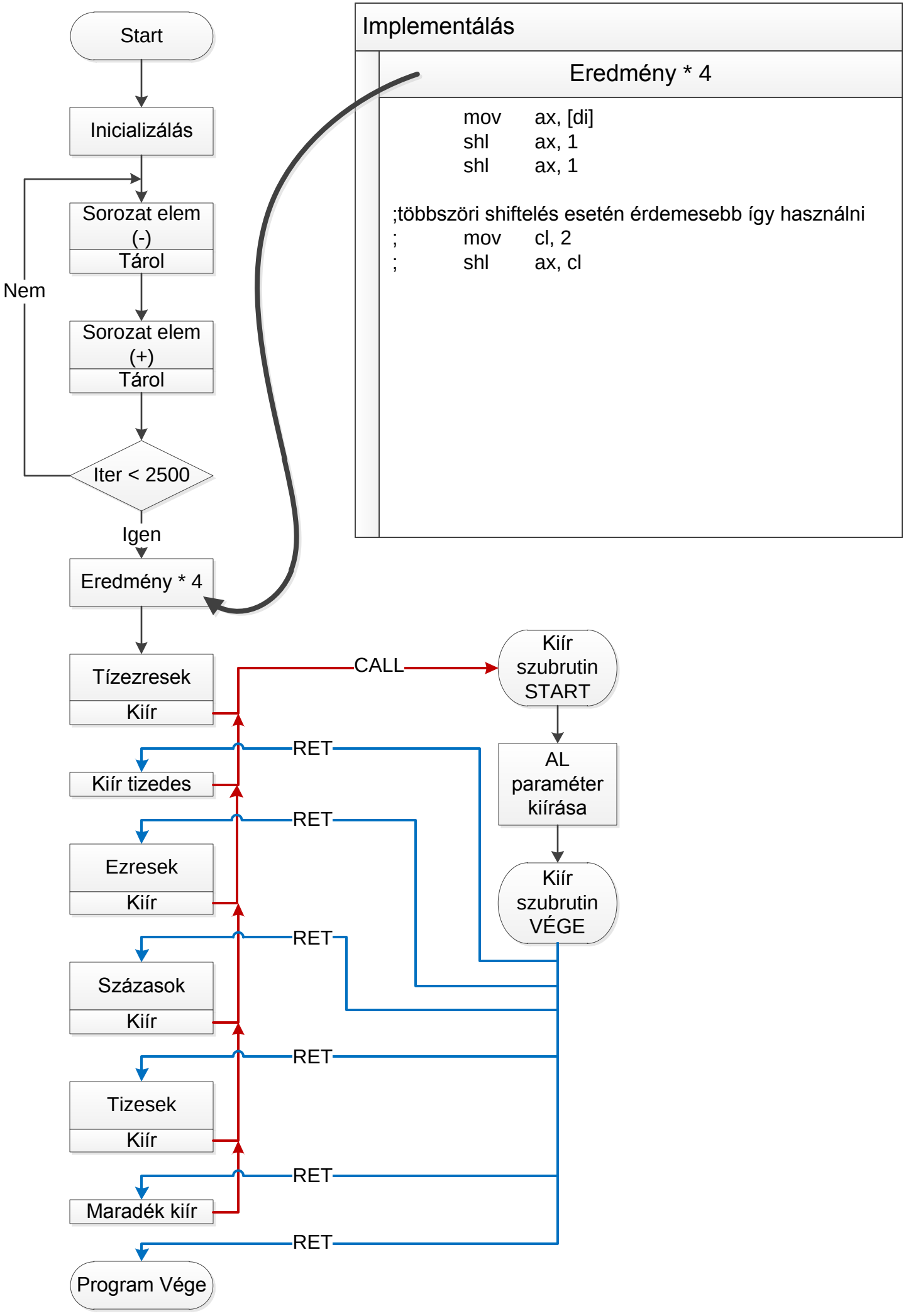
```
        pop     ax
        xor     dx, dx
        mov     bx, 10
        div     bx
        push    dx

        call    Kiir

        pop     ax

        call    Kiir

Program_Vege:
        mov     ax, 4c00h
        int     21h

Kiir:
        ;pushf
        ;pusha

        mov     dl, al
        add     dl, 48
        mov     ah, 02h
        int     21h

        ;popa
        ;popf
        ret

;Eredmeny:
;       db      "xx"

Eredmeny=$+1

Code    Ends

Data    Segment
Data    Ends

Stack   Segment
Stack   Ends
        End     Start
```

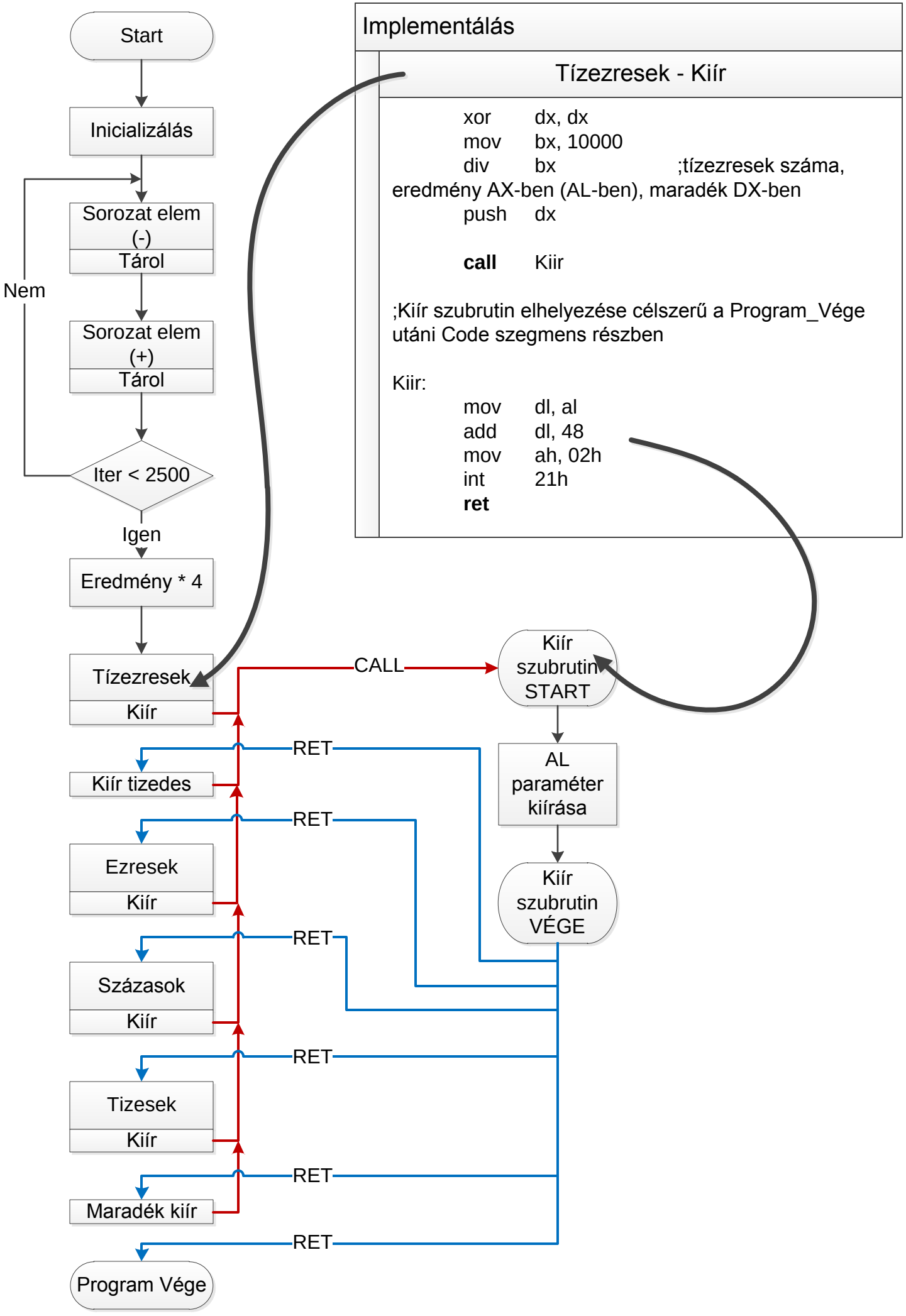

Start
Inicializálás
Sorozat elem (-)
Tárol
Sorozat elem (+)
Tárol
Nem
Iter < 2500
Igen
Eredmény * 4
Tízezresek
Kiír
CALL
Kiír szubrutin START
AL paraméter kiírása
Kiír szubrutin VÉGE
RET
Kiír tizedes
RET
Ezresek
Kiír
RET
Százasok
Kiír
RET
Tizesek
Kiír
RET
Maradék kiír
RET
Program Vége

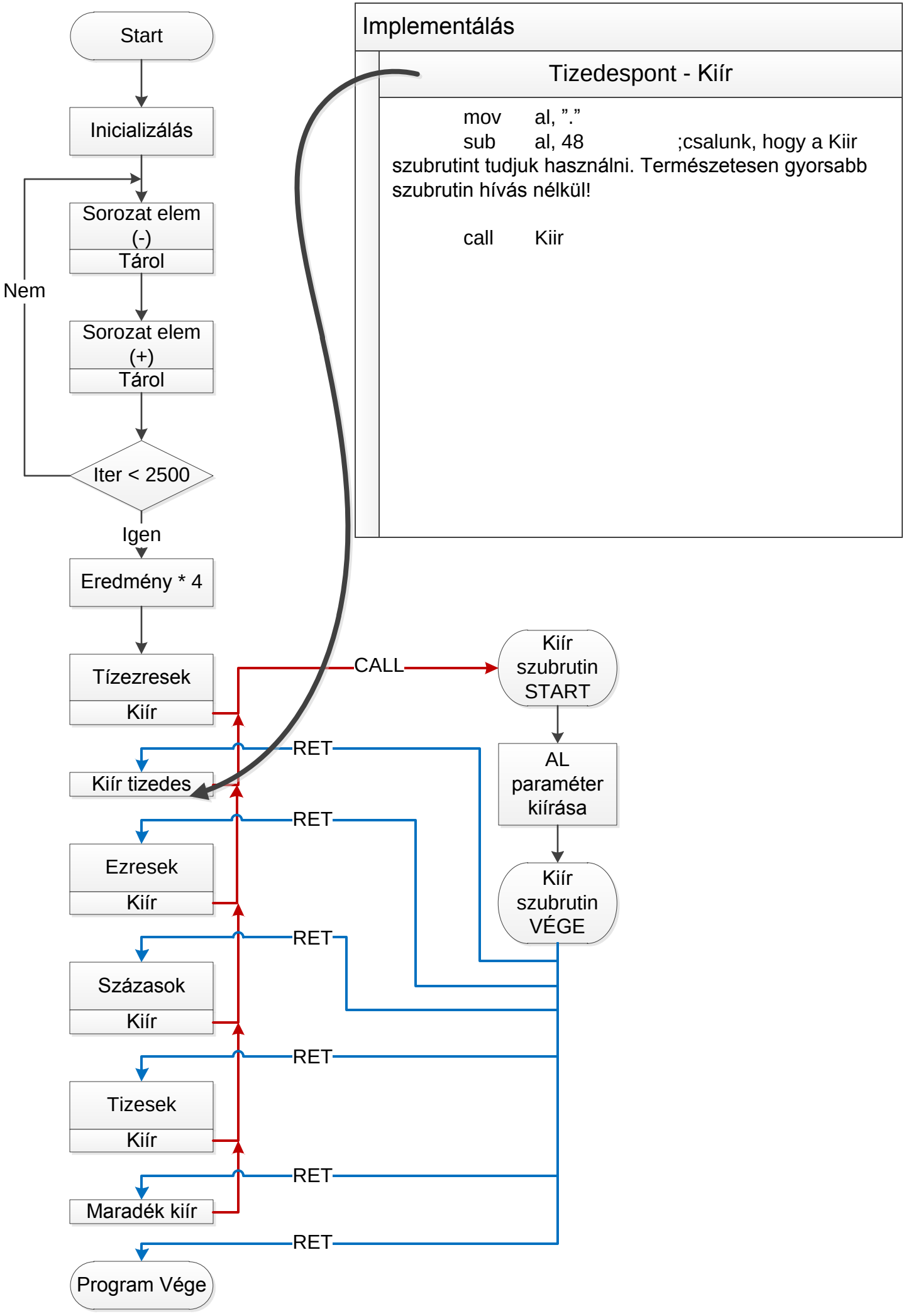

Start
Inicializálás
Sorozat elem (-)
Tárol
Sorozat elem (+)
Tárol
Iter < 2500
Nem
Igen
Eredmény * 4
Tízezresek
Kiír
Kiír tizedes
Ezresek
Kiír
Százasok
Kiír
Tizesek
Kiír
Maradék kiír
Program Vége
CALL
RET
RET
RET
RET
RET
RET
Kiír szubrutin START
AL paraméter kiírása
Kiír szubrutin VÉGE
Implementálás
Inicializálás
Szorzo = 10000
Start:
mov ax, Code
mov ds, ax
mov di, offset Eredmeny
mov ax, Szorzo
mov [di], ax ;sorozat első eleme
mov cx, 2500 ;ciklus 2500 fusson
mov si, 3 ;sorozat második elemének nevezője
;Program Vége után, de még a Code szegmensbe
Eredmeny:
db ”xx"

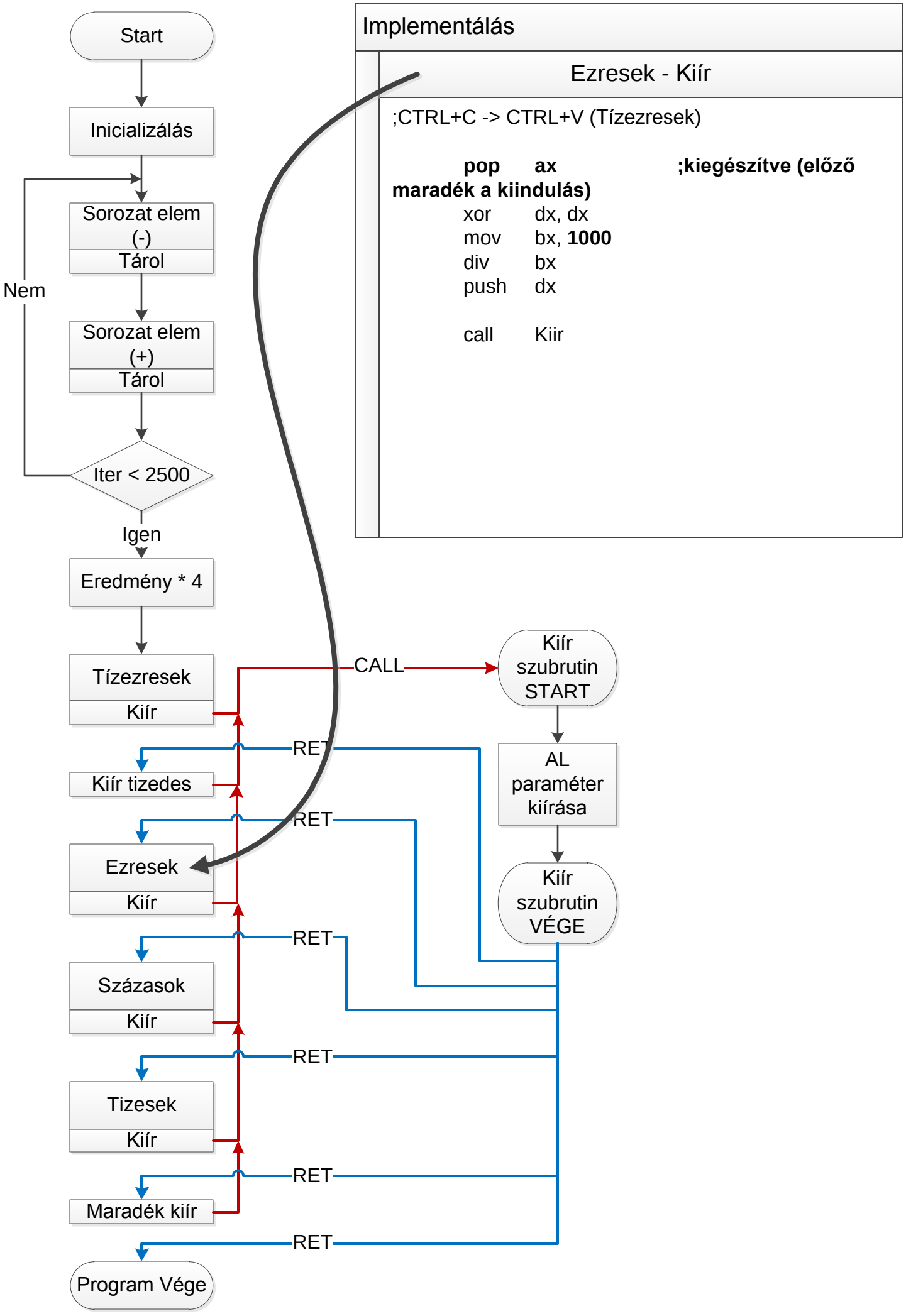

Start
Inicializálás
Sorozat elem (-)
Tárol
Nem
Sorozat elem (+)
Tárol
Iter < 2500
Igen
Eredmény * 4
Tízezresek
Kiír
CALL
Kiír szubrutin START
RET
AL paraméter kiírása
Kiír tizedes
RET
Ezresek
Kiír
Kiír szubrutin VÉGE
RET
Százasok
Kiír
RET
Tizesek
Kiír
RET
Maradék kiír
RET
Program Vége
Implementálás
Sorozat elem (-)
Szamol:
mov ax, Szorzo
xor dx, dx ;!!! 16 bites szám osztása 16 bites számmal, ne felejtkezzünk meg a DX törléséről!!!
div si
sub [di], ax
add si, 2
;DIV (16 bites operandus)
;DX:AX 32 bites osztandót elosztja az operandussal
;Eredmény: DX:maradék – AX:hányados
;INT0 (hiba) – ha a hányados nagyobb, mint 16 bit
;DIV (8 bites operandus)
;AX 16 bites osztandót elosztja az operandussal
;Eredmény: AH:maradék – AL:hányados
;INT0 (hiba) – ha a hányados nagyobb, mint 8 bit

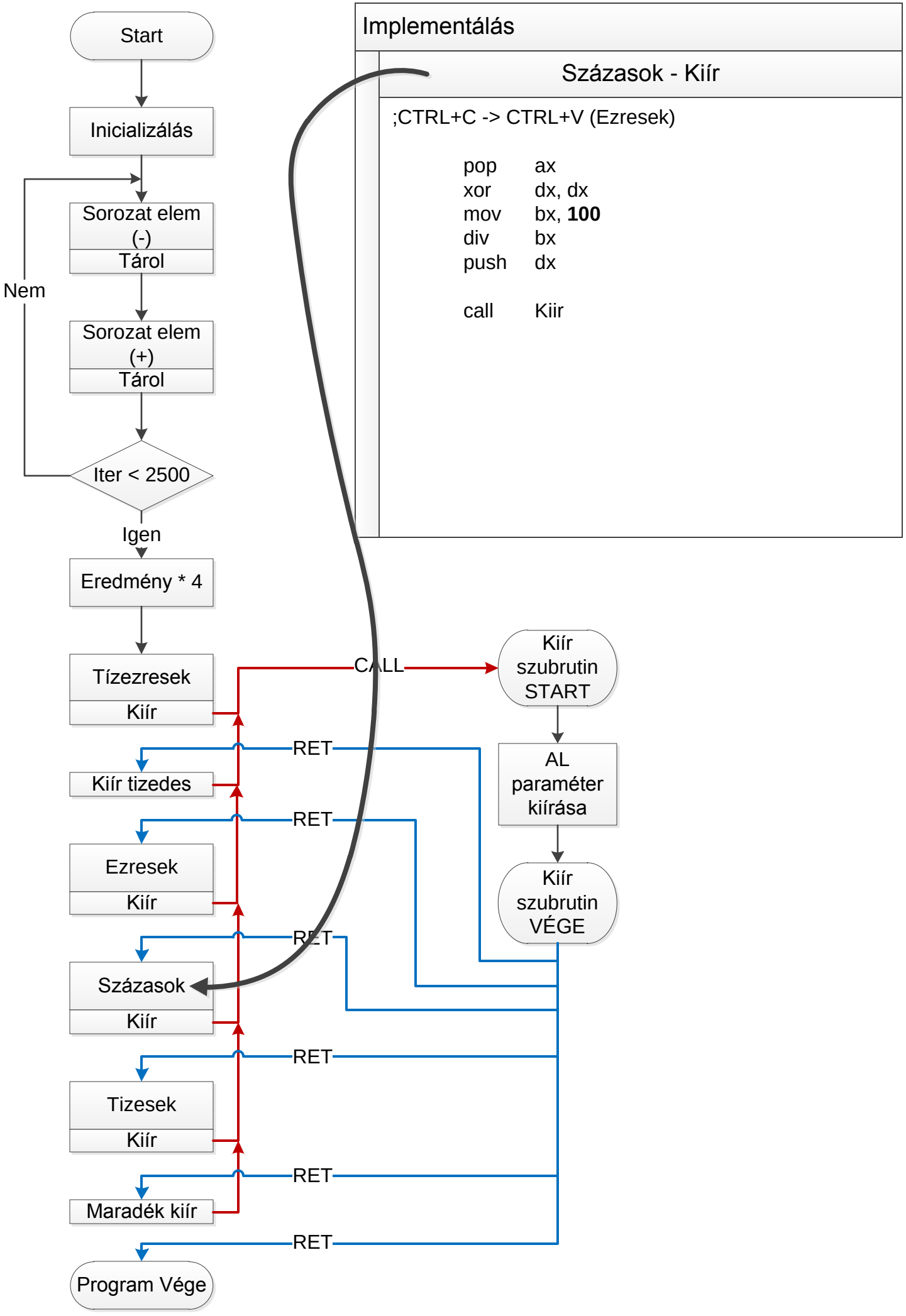

Start
Inicializálás
Sorozat elem (-)
Tárol
Sorozat elem (+)
Tárol
Iter < 2500
Nem
Igen
Eredmény * 4
Tízezresek
Kiír
Kiír tizedes
Ezresek
Kiír
Százasok
Kiír
Tizesek
Kiír
Maradék kiír
Program Vége
CALL
RET
RET
RET
RET
RET
RET
Kiír szubrutin START
AL paraméter kiírása
Kiír szubrutin VÉGE
Implementálás
Sorozat elem (+)
;CTRL+C -> CTRL+V (Sorozat elem (-))
mov ax, Szorzo
xor dx, dx
div si
add [di], ax
add si, 2
loop Szamol

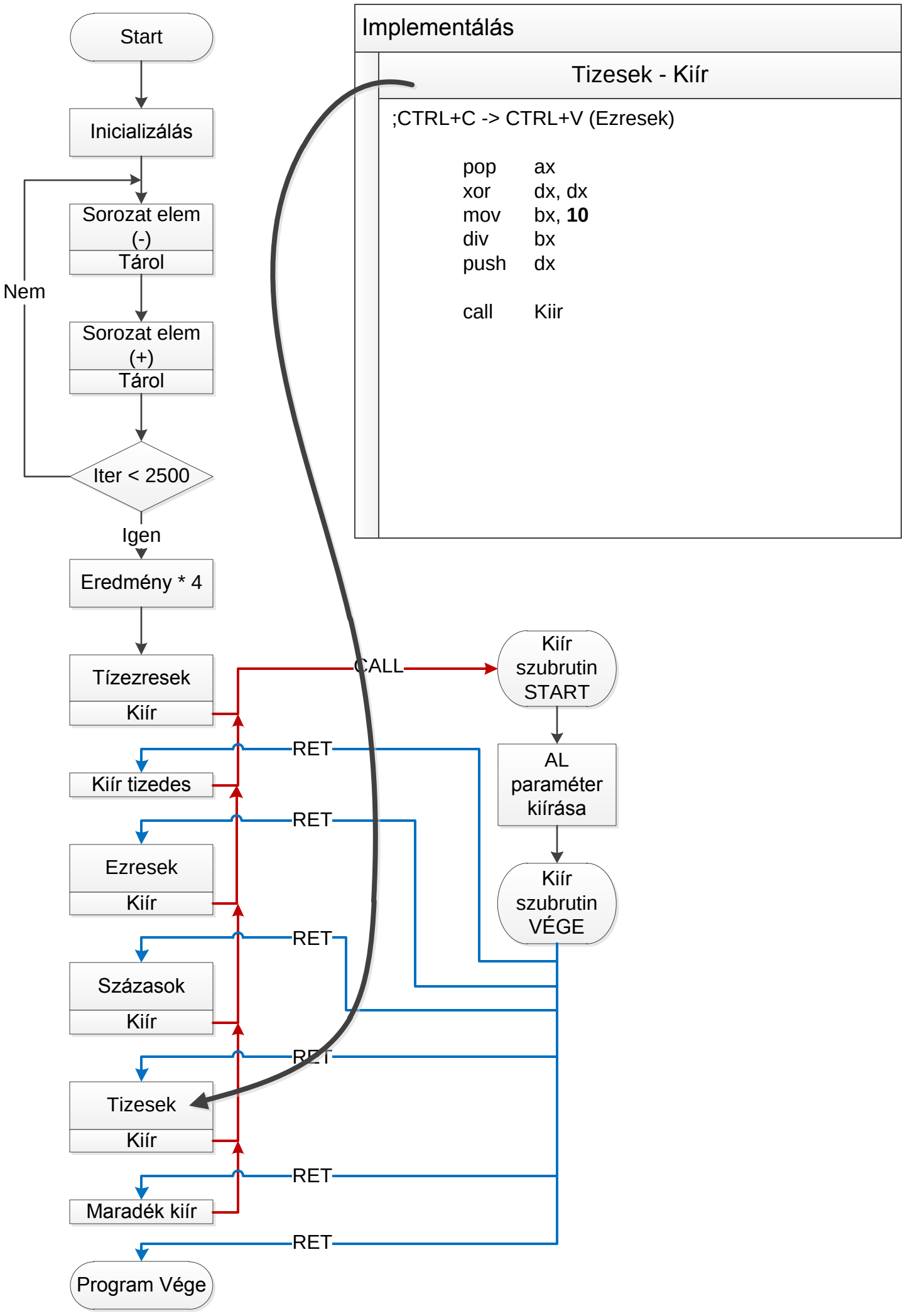

Start
Inicializálás
Sorozat elem (-)
Tárol
Sorozat elem (+)
Tárol
Iter < 2500
Nem
Igen
Eredmény * 4
Implementálás
Eredmény * 4
mov ax, [di]
shl ax, 1
shl ax, 1
;többszöri shiftelés esetén érdemesebb így használni
; mov cl, 2
; shl ax, cl
Tízezresek
Kiír
Kiír tizedes
Ezresek
Kiír
Százasok
Kiír
Tizesek
Kiír
Maradék kiír
Program Vége
CALL
RET
RET
RET
RET
RET
RET
Kiír szubrutin START
AL paraméter kiírása
Kiír szubrutin VÉGE

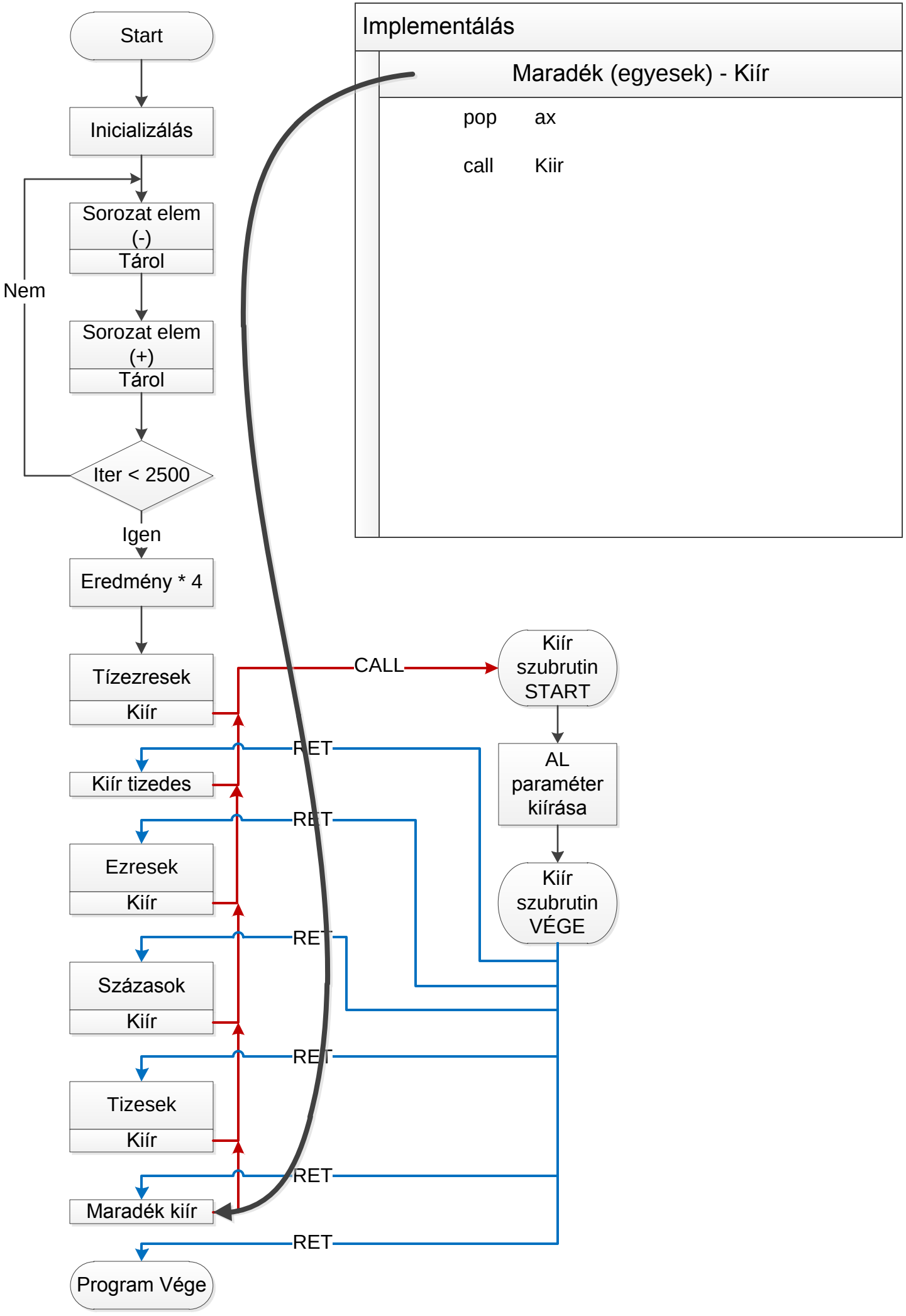

Start
Inicializálás
Sorozat elem (-)
Tárol
Sorozat elem (+)
Tárol
Iter < 2500
Nem
Igen
Eredmény * 4
Tízezresek
Kiír
Kiír tizedes
Ezresek
Kiír
Százasok
Kiír
Tizesek
Kiír
Maradék kiír
Program Vége
CALL
RET
RET
RET
RET
RET
RET
Kiír szubrutin START
AL paraméter kiírása
Kiír szubrutin VÉGE
Implementálás
Tízezresek - Kiír
xor dx, dx
mov bx, 10000
div bx ;tízezresek száma, eredmény AX-ben (AL-ben), maradék DX-ben
push dx
call Kiir
;Kiír szubrutin elhelyezése célszerű a Program_Vége utáni Code szegmens részben
Kiir:
mov dl, al
add dl, 48
mov ah, 02h
int 21h
ret


Start
Inicializálás
Sorozat elem (-)
Tárol
Sorozat elem (+)
Tárol
Iter < 2500
Nem
Igen
Eredmény * 4
Tízezresek
Kiír
Kiír tizedes
Ezresek
Kiír
Százasok
Kiír
Tizesek
Kiír
Maradék kiír
Program Vége
CALL
RET
RET
RET
RET
RET
RET
Kiír szubrutin START
AL paraméter kiírása
Kiír szubrutin VÉGE
Implementálás
Tizedespont - Kiír
mov al, ”.”
sub al, 48 ;csalunk, hogy a Kiir szubrutint tudjuk használni. Természetesen gyorsabb szubrutin hívás nélkül!
call Kiir


Start
Inicializálás
Sorozat elem (-)
Tárol
Sorozat elem (+)
Tárol
Iter < 2500
Nem
Igen
Eredmény * 4
Tízezresek
Kiír
Kiír tizedes
Ezresek
Kiír
Százasok
Kiír
Tizesek
Kiír
Maradék kiír
Program Vége
CALL
RET
RET
RET
RET
RET
RET
Kiír szubrutin START
AL paraméter kiírása
Kiír szubrutin VÉGE
Implementálás
Ezresek - Kiír
;CTRL+C -> CTRL+V (Tízezresek)
pop ax ;kiegészítve (előző maradék a kiindulás)
xor dx, dx
mov bx, 1000
div bx
push dx
call Kiir


Start
Inicializálás
Sorozat elem (-)
Tárol
Sorozat elem (+)
Tárol
Iter < 2500
Nem
Igen
Eredmény * 4
Tízezresek
Kiír
Kiír tizedes
Ezresek
Kiír
Százasok
Kiír
Tizesek
Kiír
Maradék kiír
Program Vége
CALL
RET
RET
RET
RET
RET
RET
Kiír szubrutin START
AL paraméter kiírása
Kiír szubrutin VÉGE
Implementálás
Százasok - Kiír
;CTRL+C -> CTRL+V (Ezresek)
pop ax
xor dx, dx
mov bx, 100
div bx
push dx
call Kiir


Start
Inicializálás
Sorozat elem (-)
Tárol
Sorozat elem (+)
Tárol
Iter < 2500
Nem
Igen
Eredmény * 4
Tízezresek
Kiír
Kiír tizedes
Ezresek
Kiír
Százasok
Kiír
Tizesek
Kiír
Maradék kiír
Program Vége
CALL
RET
RET
RET
RET
RET
RET
Kiír szubrutin START
AL paraméter kiírása
Kiír szubrutin VÉGE
Implementálás
Tizesek - Kiír
;CTRL+C -> CTRL+V (Ezresek)
pop ax
xor dx, dx
mov bx, 10
div bx
push dx
call Kiir


Start
Inicializálás
Sorozat elem (-)
Tárol
Sorozat elem (+)
Tárol
Iter < 2500
Nem
Igen
Eredmény * 4
Tízezresek
Kiír
Kiír tizedes
Ezresek
Kiír
Százasok
Kiír
Tizesek
Kiír
Maradék kiír
Program Vége
CALL
RET
RET
RET
RET
RET
RET
Kiír szubrutin START
AL paraméter kiírása
Kiír szubrutin VÉGE
Implementálás
Maradék (egyesek) - Kiír
pop ax
call Kiir