Lesson1: counter.asm

Description of the task

The main goal of this lesson to acquaint the structure of the assembly program. The program display a message to the screen, and counts the keystroke of „a”. The counter starts from 0, after 9 it displays the proper signs of ASCII table.

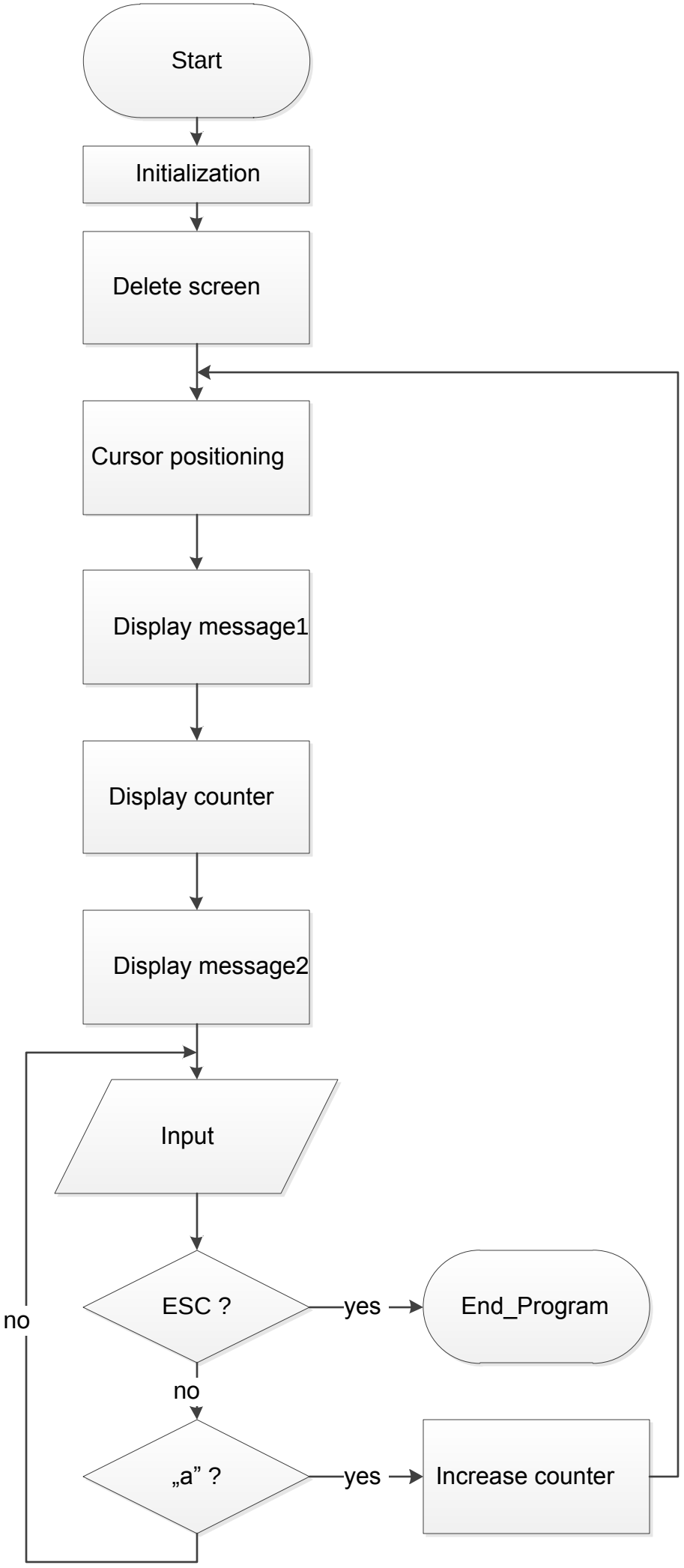
Introducing the program:

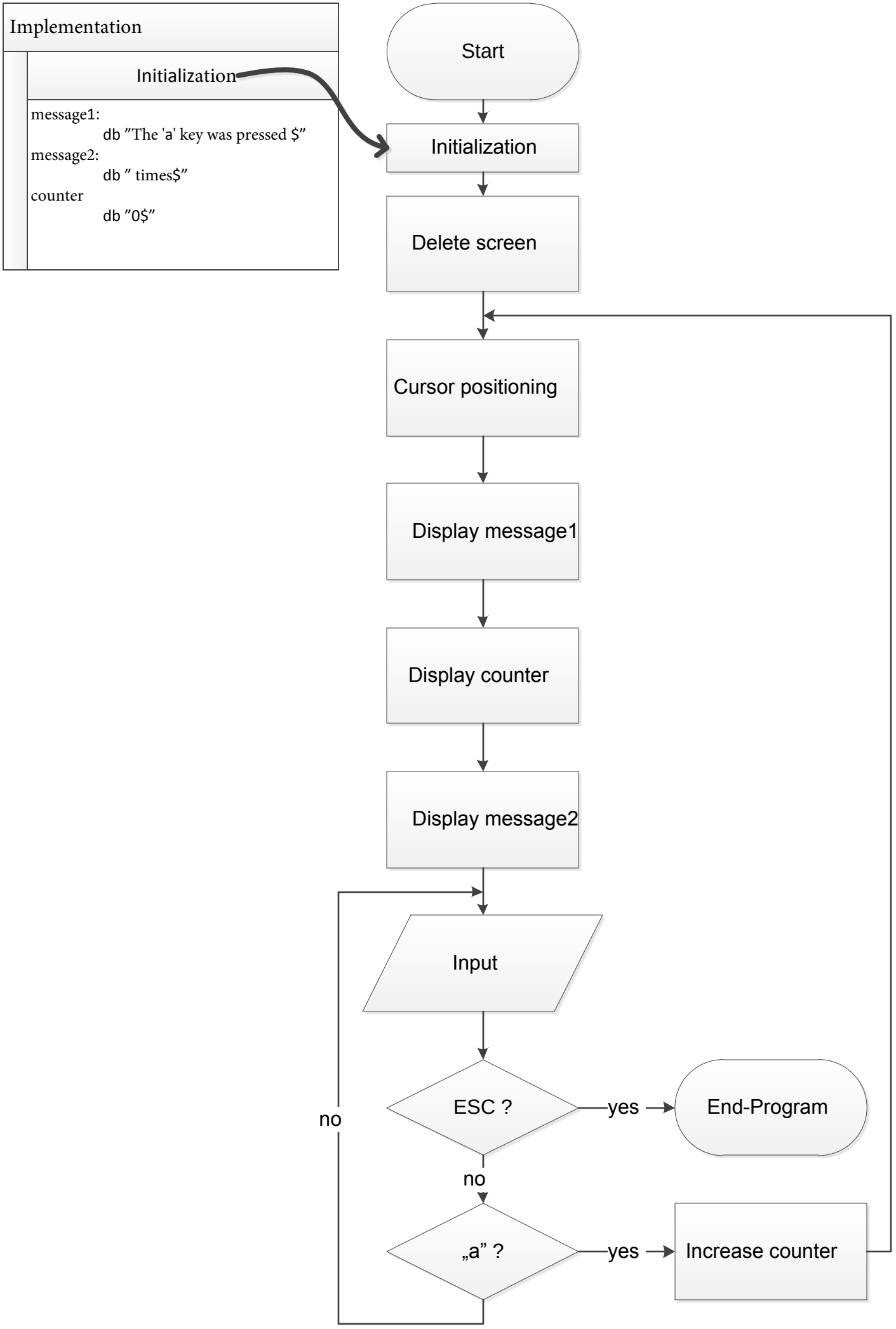
- Creating a blank, but compilable and executable frame. This frame will be the basis for the further programs.
- We take a message into this frame. This is the classical „HELLO WORLD” program. Whit this, the simple string display will be shown with the AH09h, Int21h DOS function.

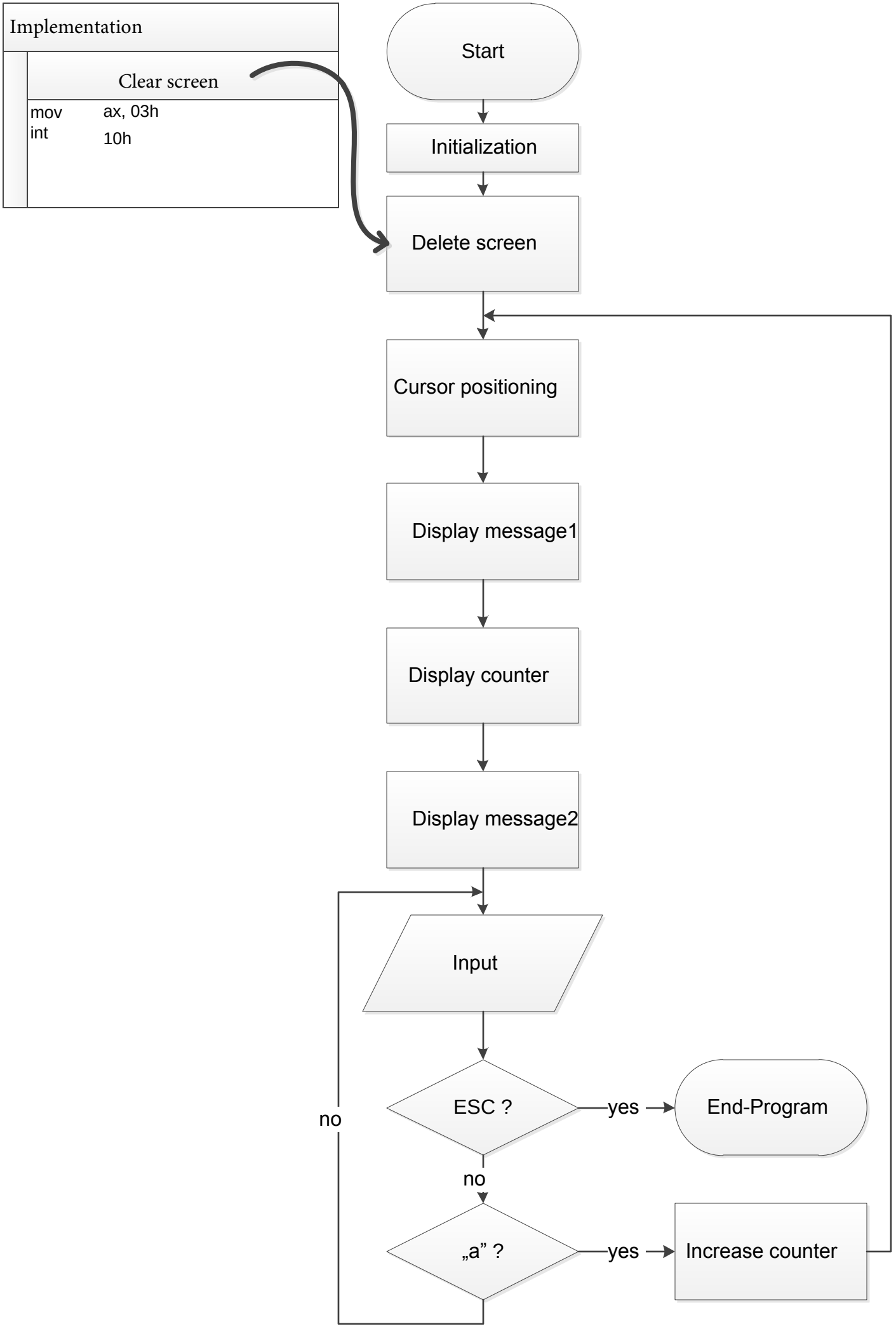
- We have to create a program loop. This can be achieved with a JMP instruction. We have to ensure the exit from the loop to aviod the endless loop. To do so, we must know the AH00h, Int16h function, and we have to write a test condition.

Implementation

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

Start:
        mov     ax, Code
        mov     DS, ax

        mov     ax, 03h
        int     10h
Display:
        mov     ah, 02h
        mov     bh, 0
        mov     dh, 10
        mov     dl, 0
        int     10h

        mov     dx, offset message1
        mov     ah, 09h
        int     21h

        mov     dx, offset
        ;mov    counter ah, 09h
        int     21h

        mov     dx, offset
        ;mov    message2 ah, 09h
        int     21h
Input:
        xor     ax, ax
        int     16h

        cmp     al, 27
        jz      End_Program

        cmp     al, ”a”
        jz      Count
        jmp     Input
Count:
        mov     di, offset counter
        mov     al, [di]
        inc     al
        mov     [di], al

        jmp     Input

End_Program:
        mov     ax, 4c00h
        int     21h

message1:
        db      ”The 'a' key was pressed $”
message2:
        db      ” times$”
counter:
        db      ”0$”

Code    Ends

Data    Segment
Data    Ends

Stack   Segment
Stack   Ends

        End     Start
```


Start
Initialization
Delete screen
Cursor positioning
Display message1
Display counter
Display message2
Input
ESC ?
yes
End_Program
no
no
„a” ?
yes
Increase counter

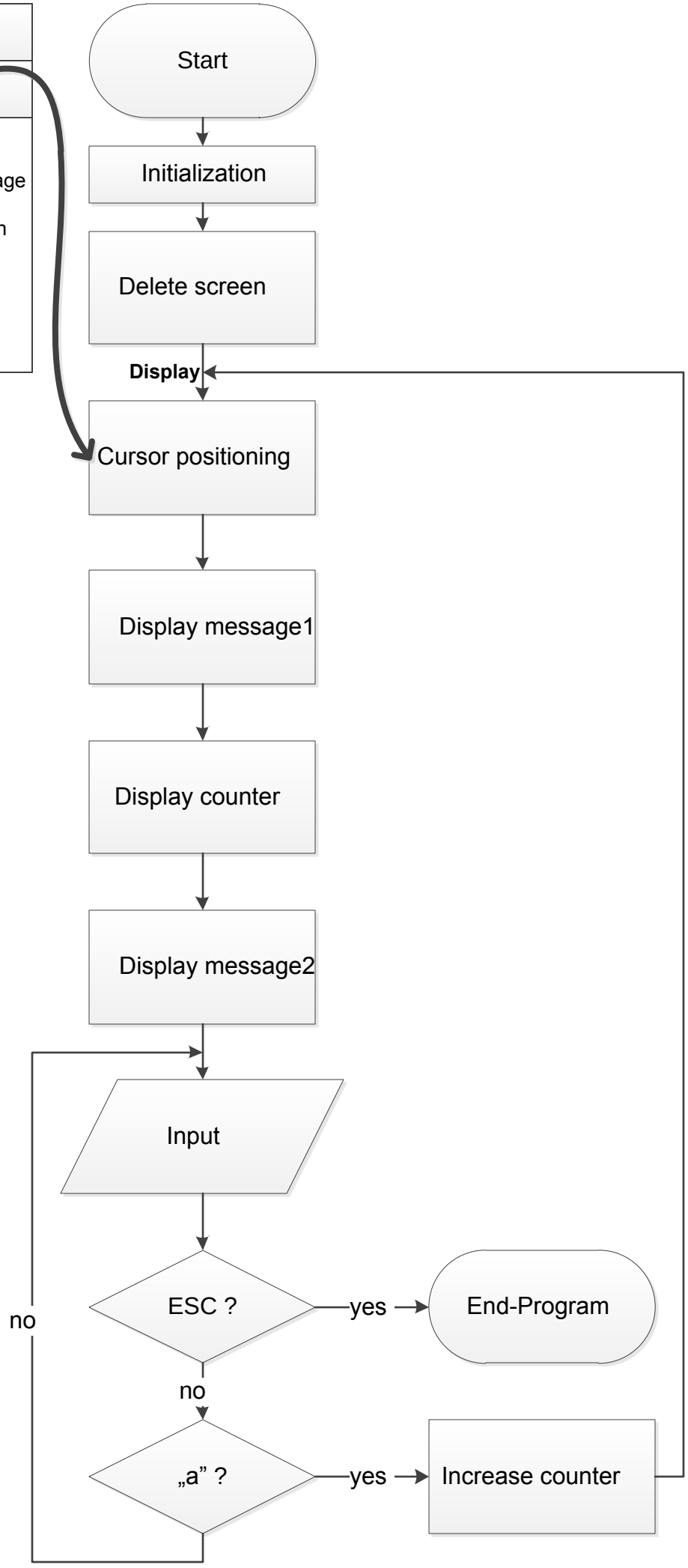

Implementation
Initialization
message1:
db ”The 'a' key was pressed $”
message2:
db ” times$”
counter
db ”0$”
Start
Initialization
Delete screen
Cursor positioning
Display message1
Display counter
Display message2
Input
ESC ?
yes
End-Program
no
no
„a” ?
yes
Increase counter

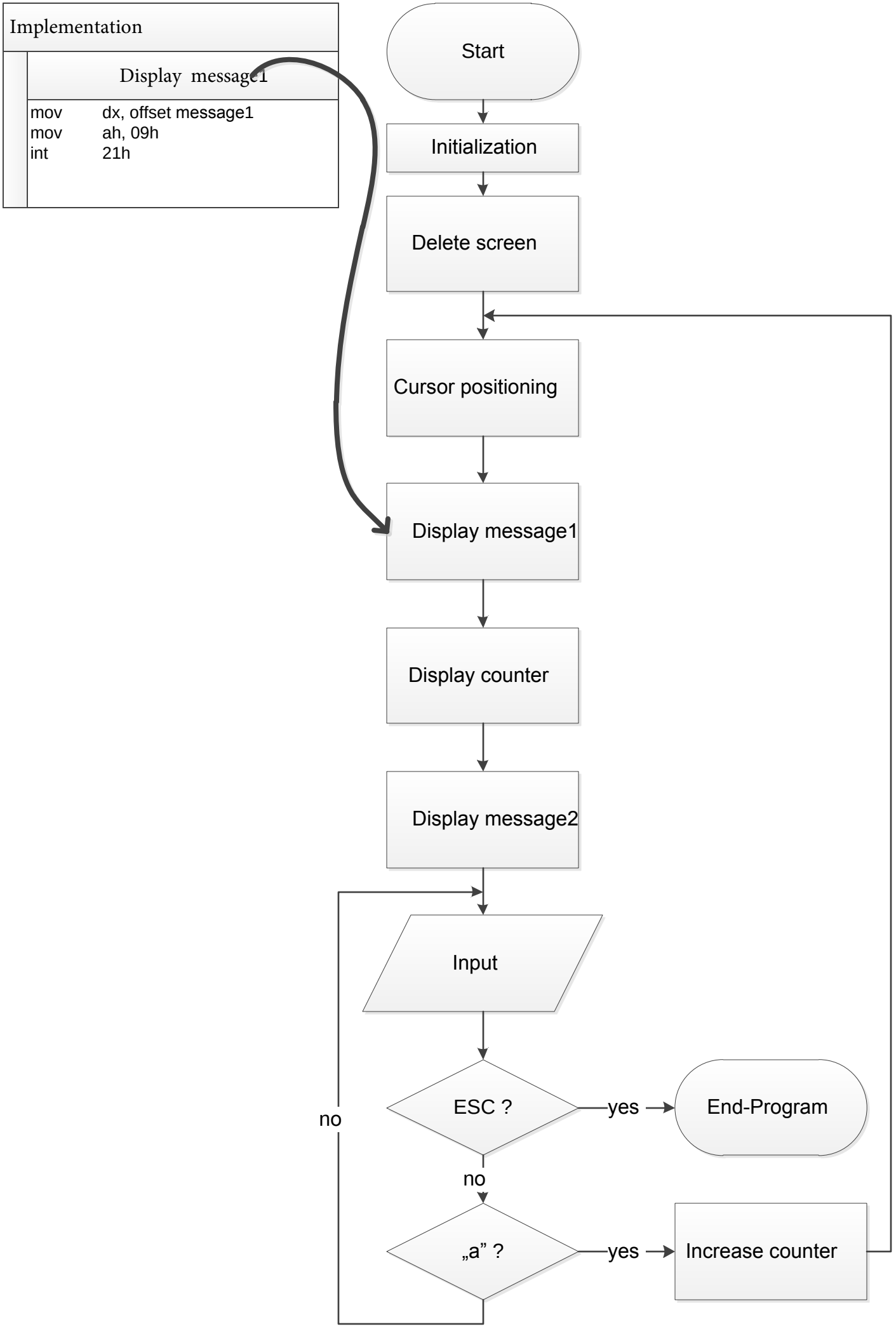

Implementation
Clear screen
mov ax, 03h
int 10h
Start
Initialization
Delete screen
Cursor positioning
Display message1
Display counter
Display message2
Input
ESC ?
yes
End-Program
no
no
„a" ?
yes
Increase counter

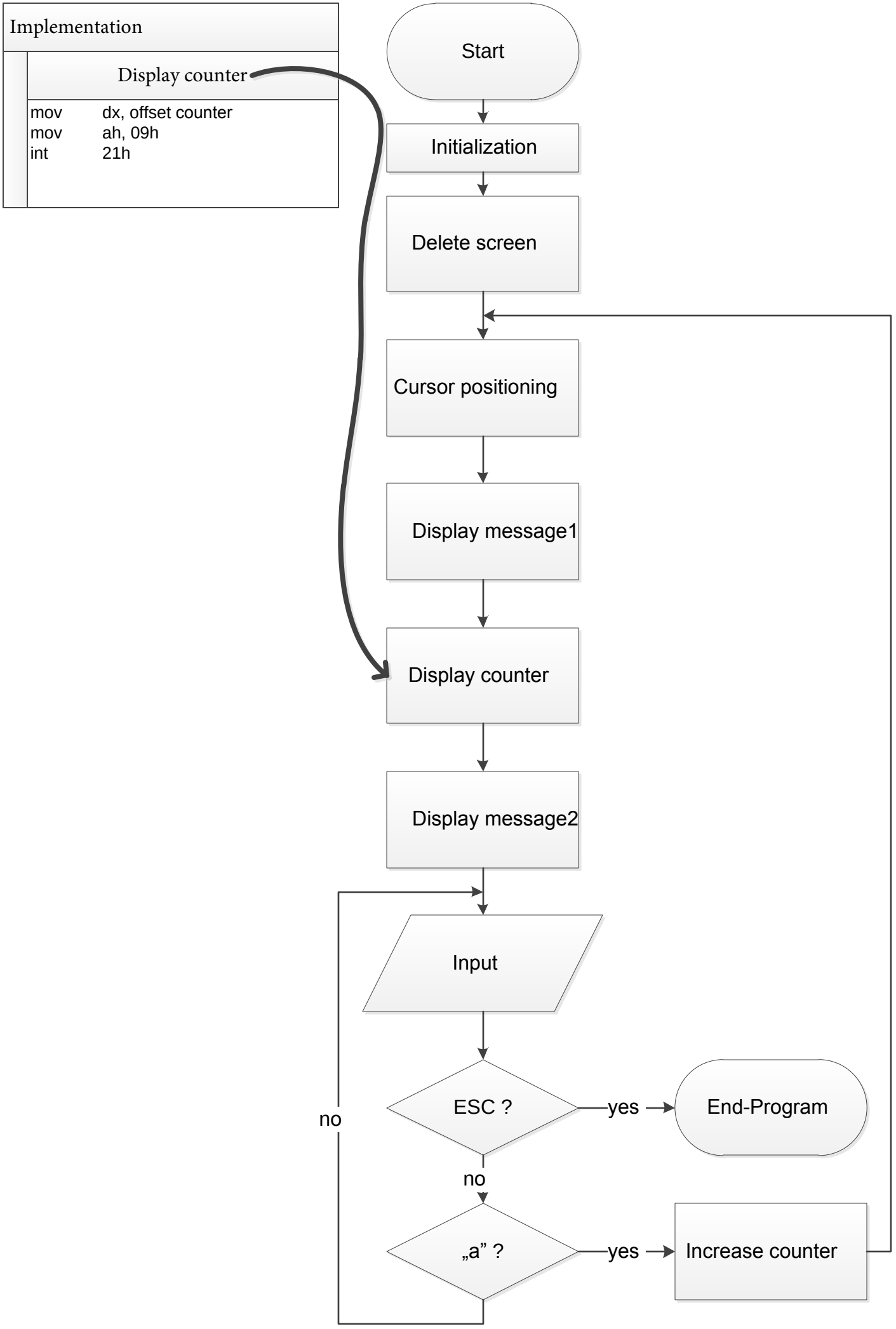
Implementation

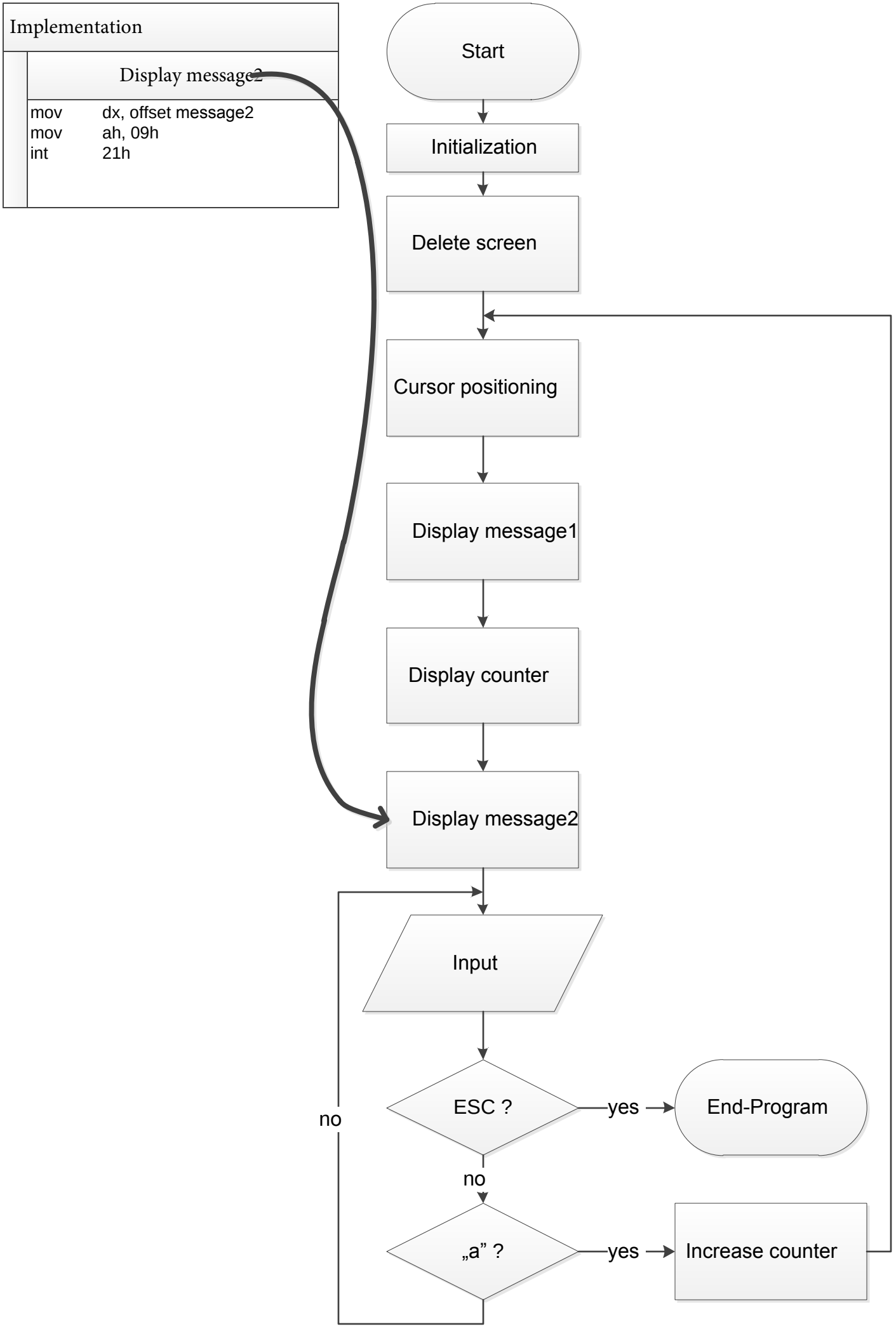
Cursor positioninig

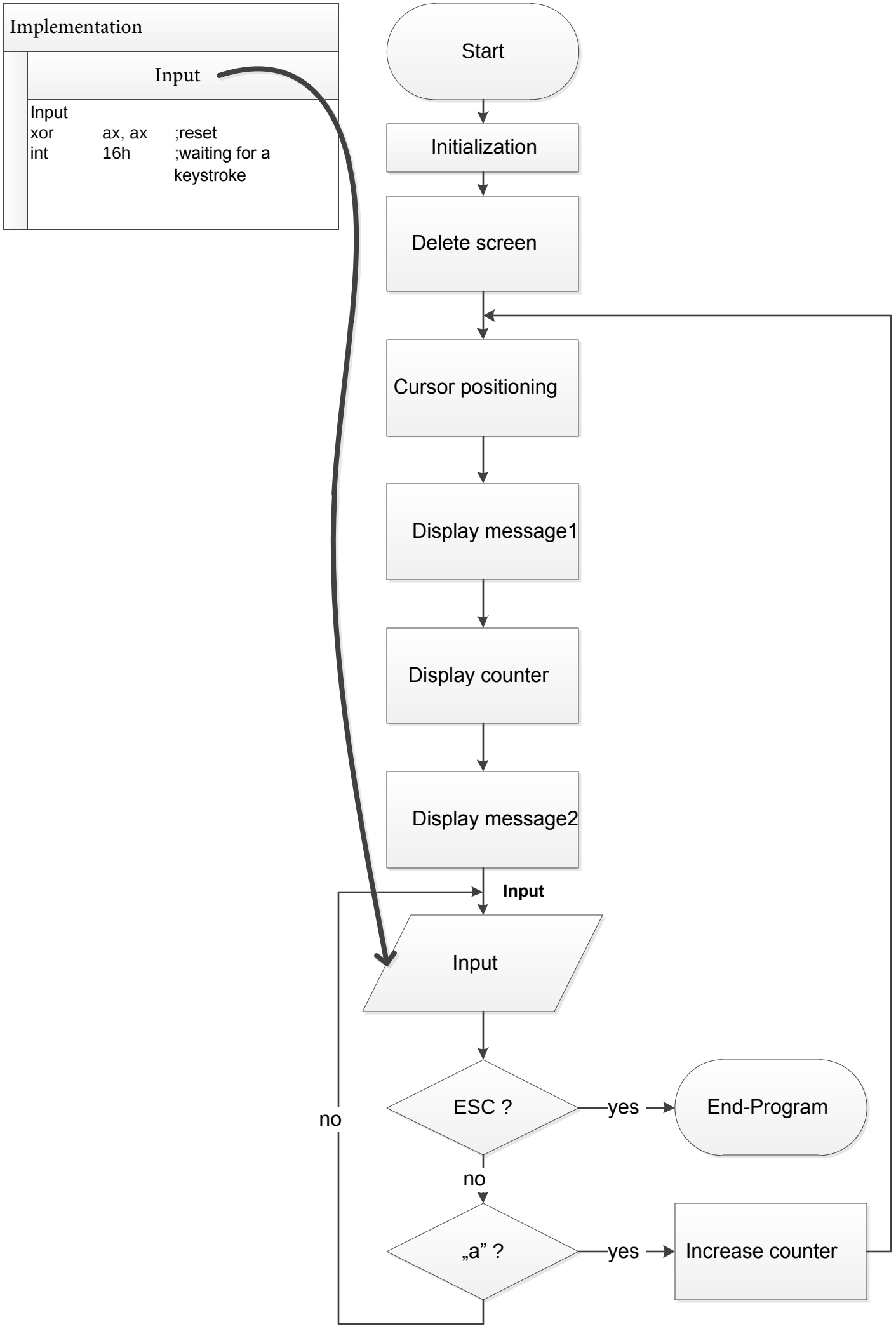
```
Display:
mov     ah, 02h
mov     bh, 0     ;number of video page
mov     dh, 10    ;cursor to 10. row
mov     dl, 0     ;cursor to 0. column
int     10h
```

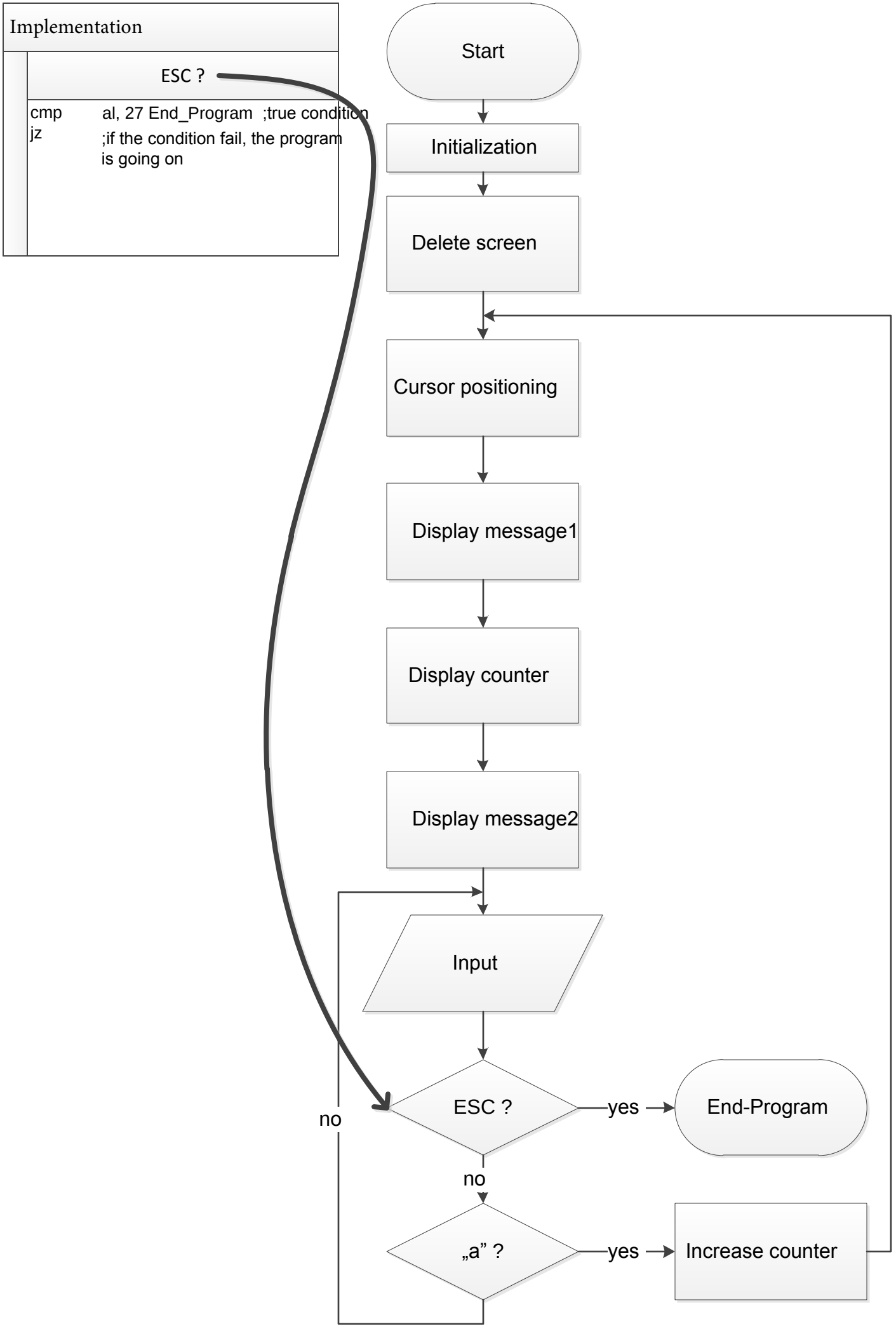


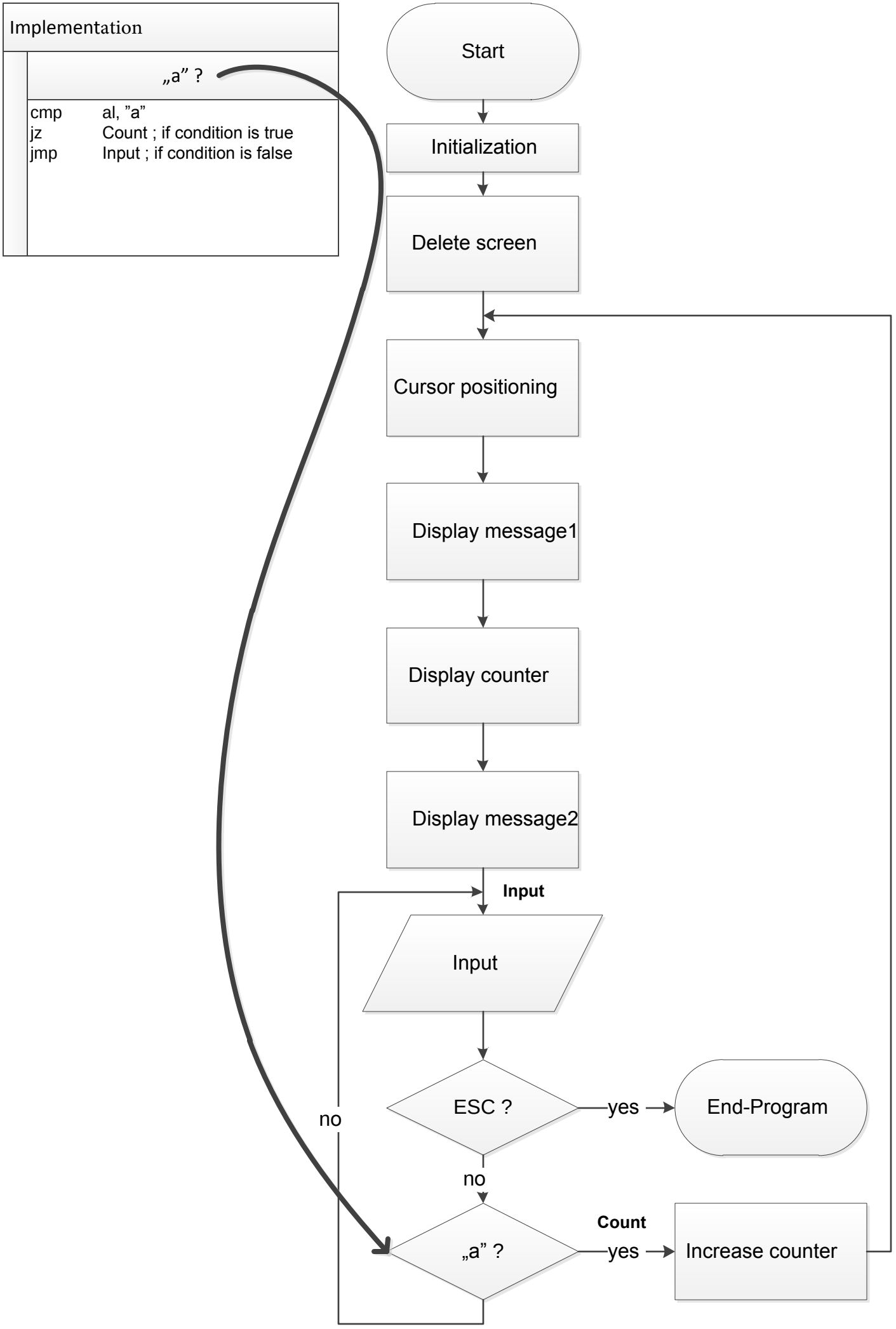

Implementation
Display message1
mov dx, offset message1
mov ah, 09h
int 21h
Start
Initialization
Delete screen
Cursor positioning
Display message1
Display counter
Display message2
Input
ESC ?
yes
End-Program
no
no
„a" ?
yes
Increase counter

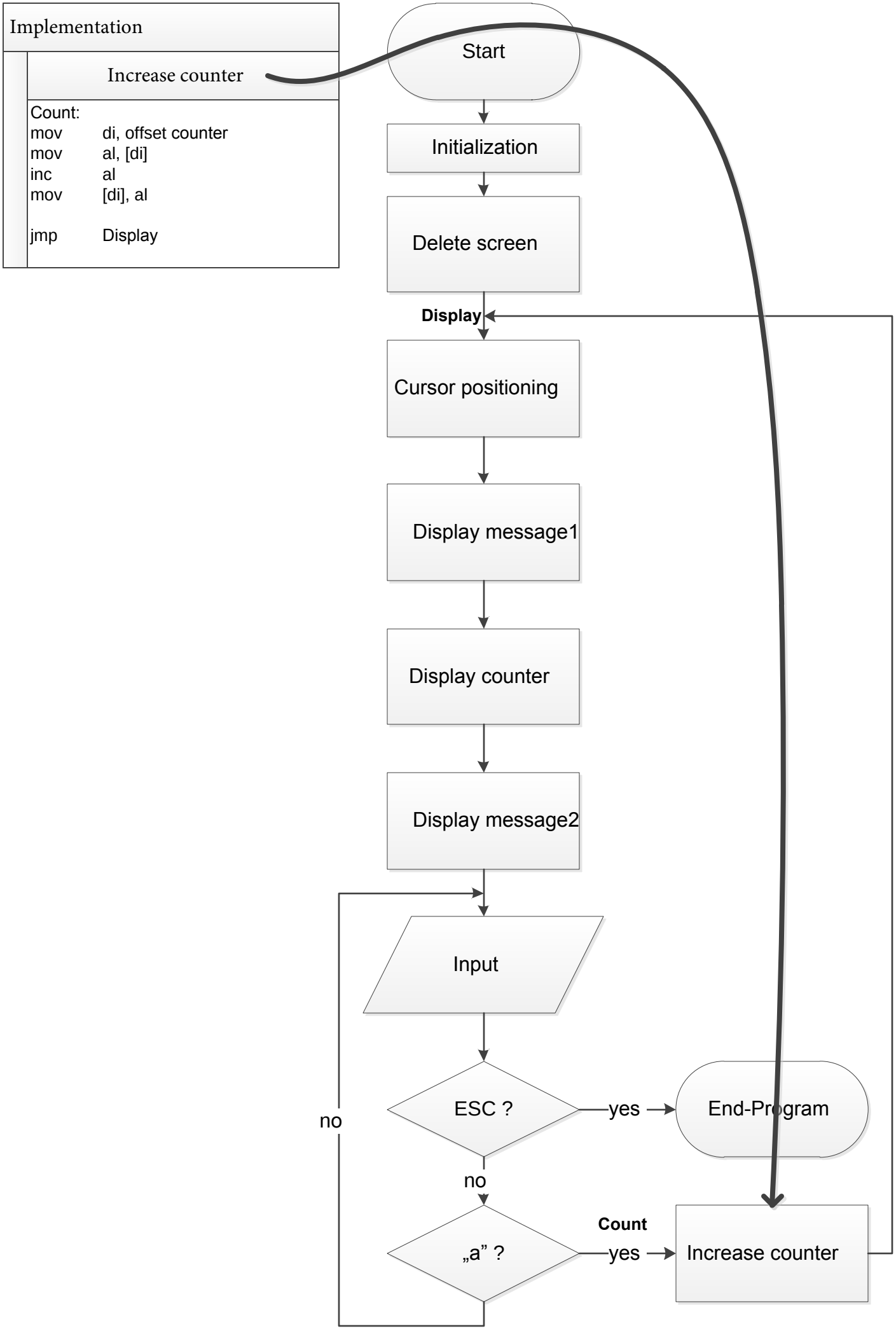

Implementation
Display counter
mov dx, offset counter
mov ah, 09h
int 21h
Start
Initialization
Delete screen
Cursor positioning
Display message1
Display counter
Display message2
Input
ESC ?
yes
End-Program
no
no
„a" ?
yes
Increase counter


Implementation
Display message2
mov dx, offset message2
mov ah, 09h
int 21h
Start
Initialization
Delete screen
Cursor positioning
Display message1
Display counter
Display message2
Input
ESC ?
yes
End-Program
no
no
„a" ?
yes
Increase counter


Implementation
Input
Input
xor ax, ax ;reset
int 16h ;waiting for a keystroke
Start
Initialization
Delete screen
Cursor positioning
Display message1
Display counter
Display message2
Input
Input
ESC ?
yes
End-Program
no
no
„a" ?
yes
Increase counter


Implementation
ESC ?
cmp al, 27 End_Program ;true condition
jz ;if the condition fail, the program is going on
Start
Initialization
Delete screen
Cursor positioning
Display message1
Display counter
Display message2
Input
ESC ?
yes
End-Program
no
no
„a" ?
yes
Increase counter


Implementation
„a” ?
cmp al, ”a”
jz Count ; if condition is true
jmp Input ; if condition is false
Start
Initialization
Delete screen
Cursor positioning
Display message1
Display counter
Display message2
Input
Input
ESC ?
yes
End-Program
no
no
Count
„a” ?
yes
Increase counter


Implementation
Increase counter
Count:
mov di, offset counter
mov al, [di]
inc al
mov [di], al
jmp Display
Start
Initialization
Delete screen
Display
Cursor positioning
Display message1
Display counter
Display message2
Input
ESC ?
yes
End-Program
no
no
Count
„a" ?
yes
Increase counter