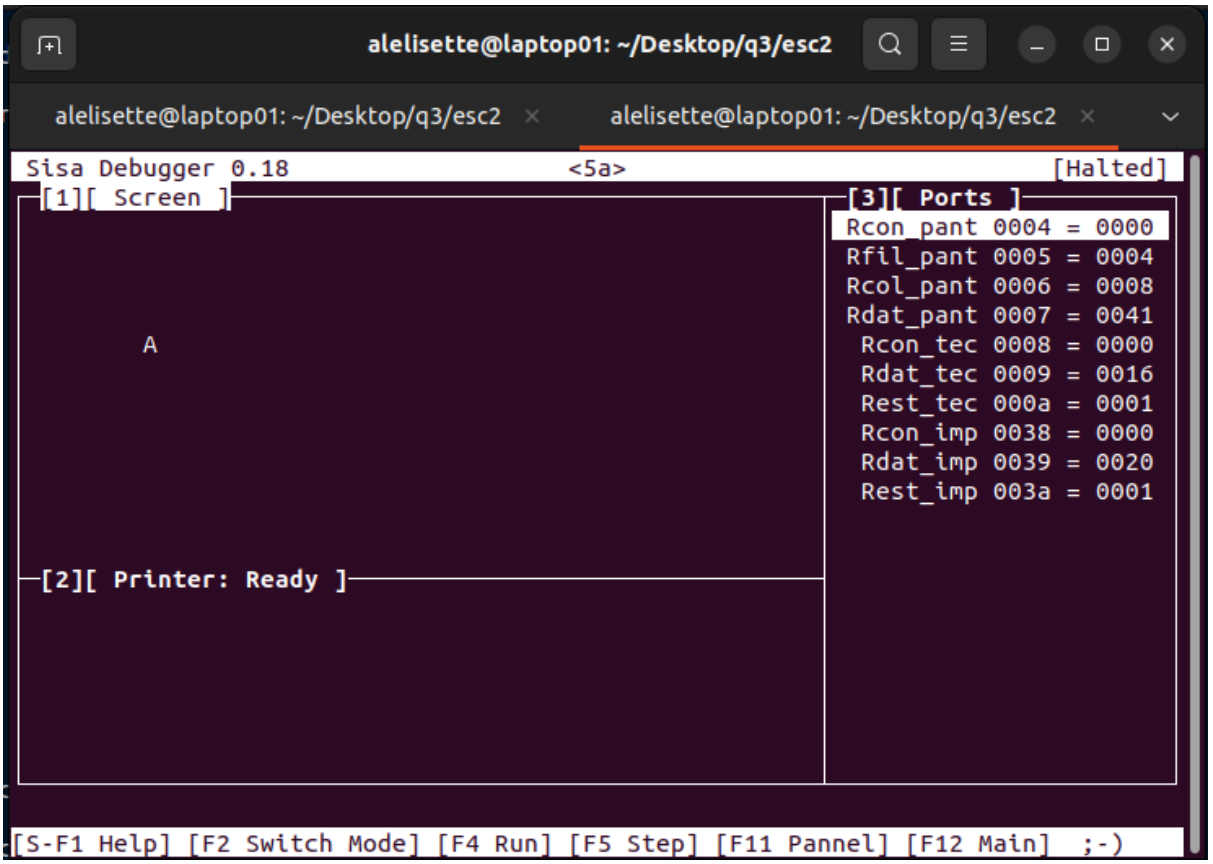
# ESC2: PRÀCTICA LABORATORI 2


**Nom de la professora:** Ana M Heredero
**Nom de l'alumne:** Alejandra Lisette Rocha
**Curs: 2023/2024**

**Activitat 5.A: Rellotge**
**L'objectiu d'aquest apartat és programar el rellotge. Fes l'exercici 5.1.**
**Exercici 5.1: Escriu un programa en alt nivell que esperi fins que hagin transcorregut 5**
**segons, i llavors escrigui una 'A' a la posició [4,8] de la pantalla, amb atribut normal.**
**Escriu el main i la RSI de rellotge**

```
.include "macros.s"
.include "crt0.s"
.data
        ticks: .word 0
        final: .byte 0
        .balign 2

.text

main:

        $MOVEI R0, interrupts_vector
        $MOVEI R1, clock
        ST 0(R0), R1  ;codi id del clock es 0
        MOVI R2, 1
        OUT Rcon_rel, R2      ;permet interrupcions del rellotge
        EI                    ;permet interrupcions i crida a clock per si mateix
        $MOVEI R0, final ;R0 conte la adreca de memoria de final
        $MOVEI R3, 0x8000
        MOVI R4, 4
        OUT Rfil_pant, R4    ;pasa
        MOVI R4, 8
        OUT Rcol_pant, R4
        MOVI R4, 'A
        OUT Rdat_pant, R4

bucle:

        LDB R2, 0(R0)
        BZ R2, bucle
        OUT Rcon_pant, R3
        HALT

clock:

        $MOVEI R0, ticks
        LD R1, 0(R0)
        ADDI R1,R1,1
        ST 0(R0), R1    ; incrementa ticks de un a un
        MOVI R2, 10*5
```

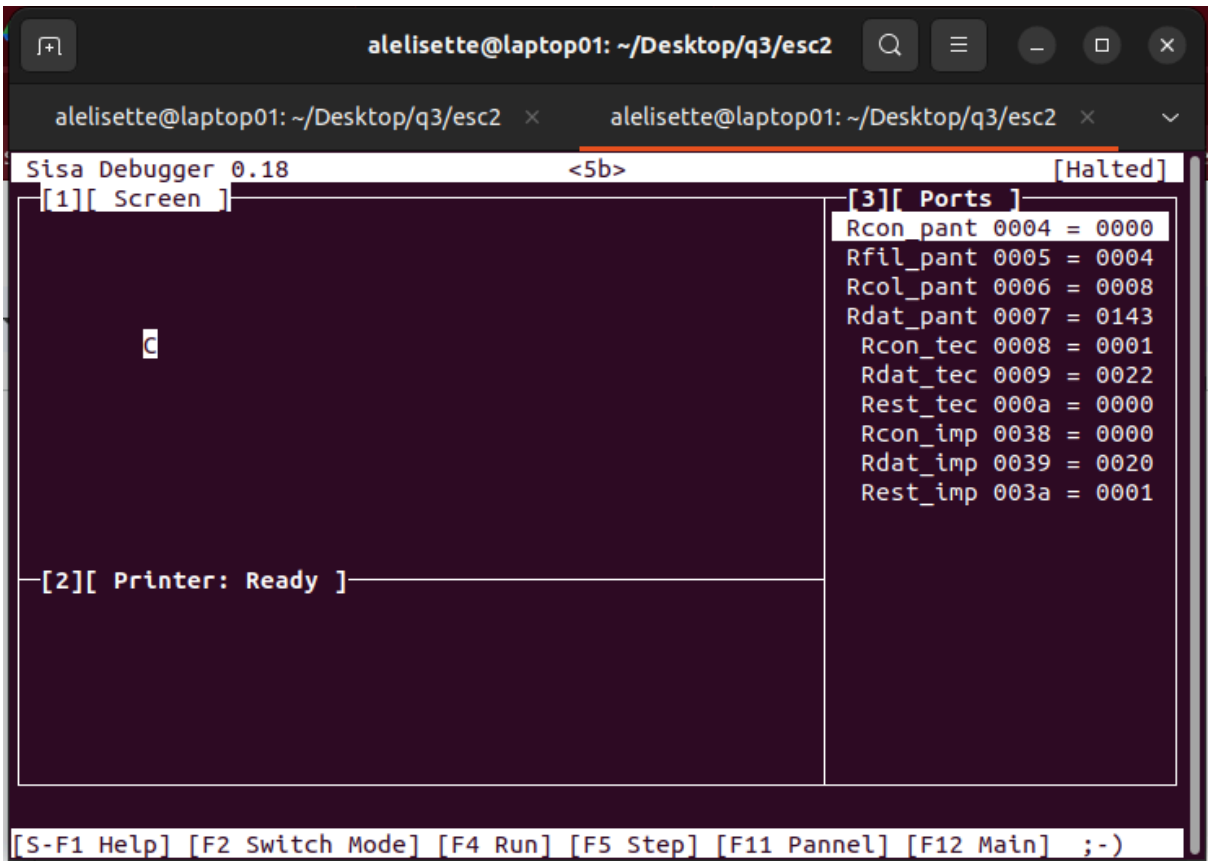
```
        $CMPGE R2, R1, R2 ;compara si ticks es major o igual que 10*5
        BZ R2, fiCLK    ;si R2=0 salta a la etiqueta fiCLK
        MOVI R1, 1
        $MOVEI R0, final
        STB 0(R0), R1  ;R1 te dins final=true

fiCLK:
        JMP R6
```




**Activitat 5.B: Teclat**
**Farem un programa que se sincronitzi per interrupcions amb el teclat. Completa l'exercici 5.2.**
**Exercici 5.2: Escriu un programa en alt nivell que esperi fins que polsem una tecla qualsevol, i llavors escrigui en pantalla, a la posició [4,8], el caràcter associat a aquesta**
**tecla, en mode invers. Escriu el main, la RSI de teclat, i les variables que necessitis.**

```
.include "macros.s"
.include "crt0.s"
.data

final:  .byte 0
        .balign 2
```

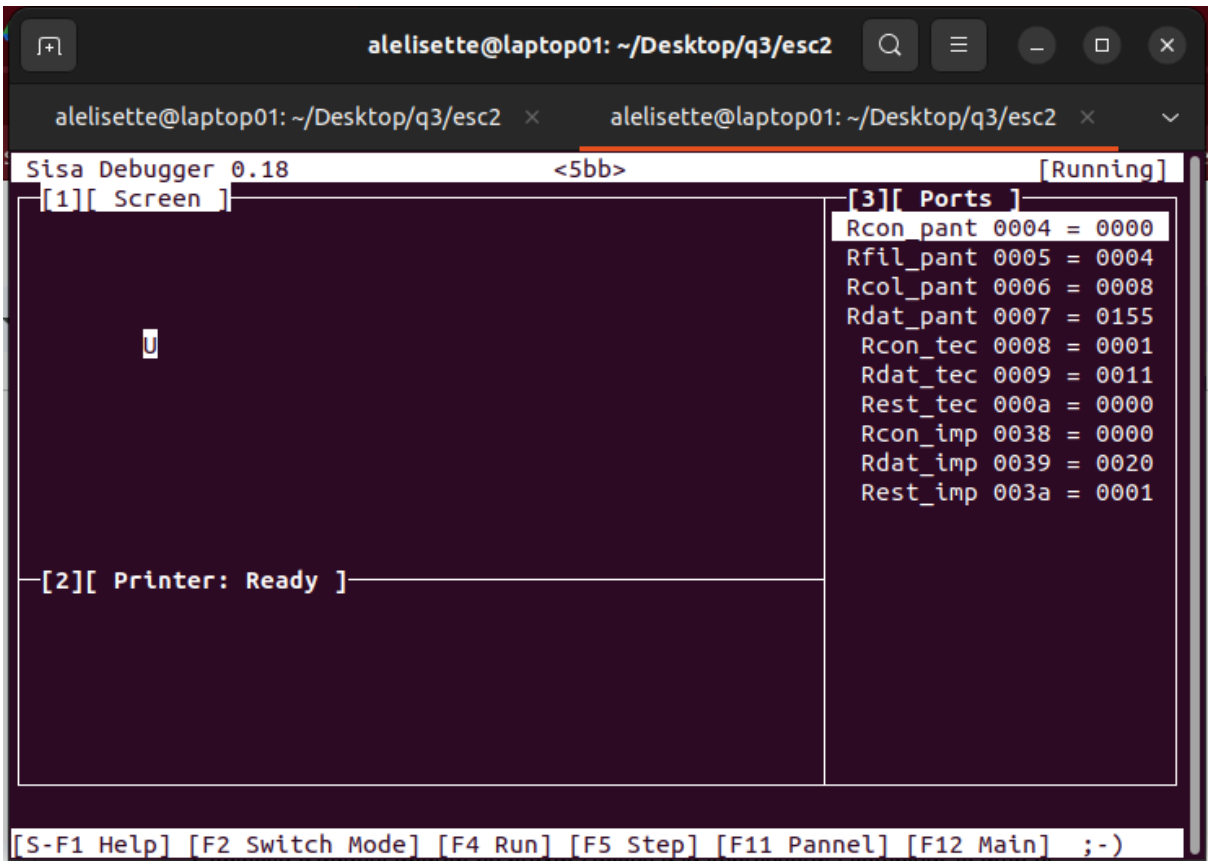
```
.text

main:
        $MOVEI R0, interrupts_vector
        $MOVEI R1, teclat
        ST 1*2(R0), R1          ;codi id teclat es 1*2=2
        MOVI R2, 1
        OUT Rcon_tec, R2
        EI


bucle:
        $MOVEI R0, final
        LDB R0, 0(R0)           ;cargo la adreca de la tecla en R0
        BZ R0, bucle
        $MOVEI R3, 0x8000
        MOVI R4, 4
        OUT Rfil_pant, R4 ;pasa la fila=4 a registre
        MOVI R4, 8
        OUT Rcol_pant, R4 ;pasa la col=8 a registre
        $MOVEI R5, 0x0100  ;R5 te assignat el mode invers
        ADD R0, R0, R5 ;passo a R0 el valor de la tecla en R5
        OUT Rdat_pant, R0
        OUT Rcon_pant, R3

teclat:
        $MOVEI R3, tteclat
        IN R1, Rdat_tec         ;R1=tecla
        ADD R1, R3, R1          ;&tteclat[tecla]
        LDB R1, 0(R1)           ;carga el byte de R1+0 dins de R1 que es la tecla
        $MOVEI R0, final
        STB 0(R0), R1           ;guardem en 0+R0 la tecla que es troba dins de R1
        JMP R6
```

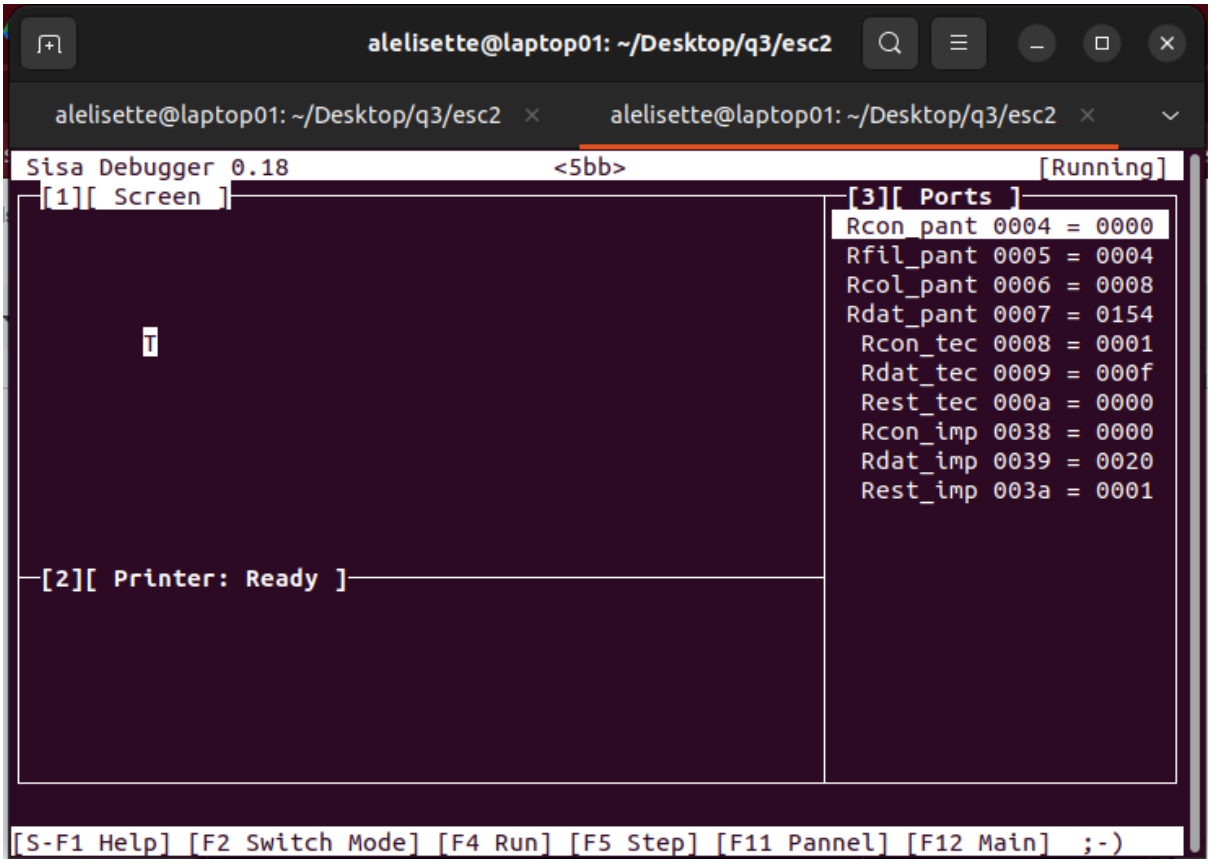


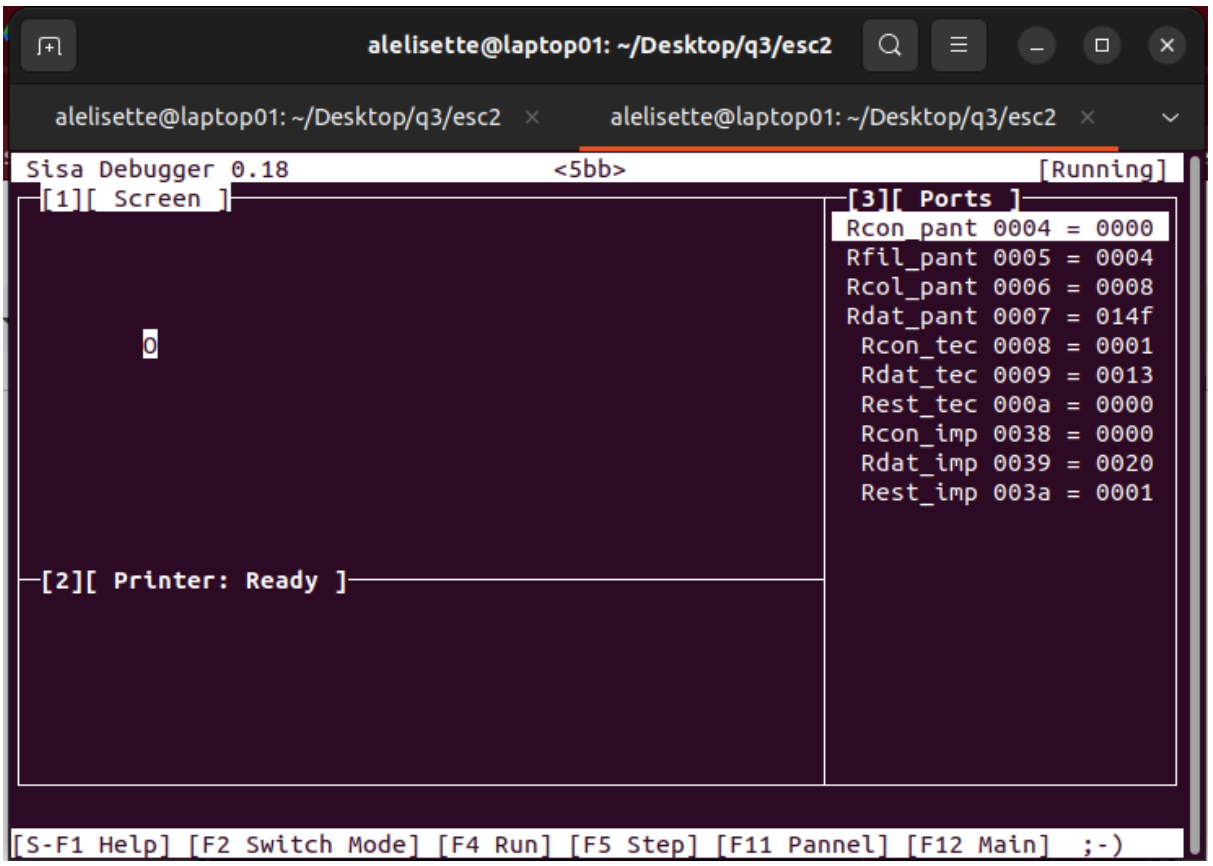

**Exercici 5.3: Modifica el programa anterior perquè repeteixi la mateixa tasca per a cada tecla que polsem, i acabi quan la tecla polsada sigui la 'X'.**
**Tradueix aquest programa a SISA-F en el mateix fitxer s5b.s Verifica'l amb el simulador.**

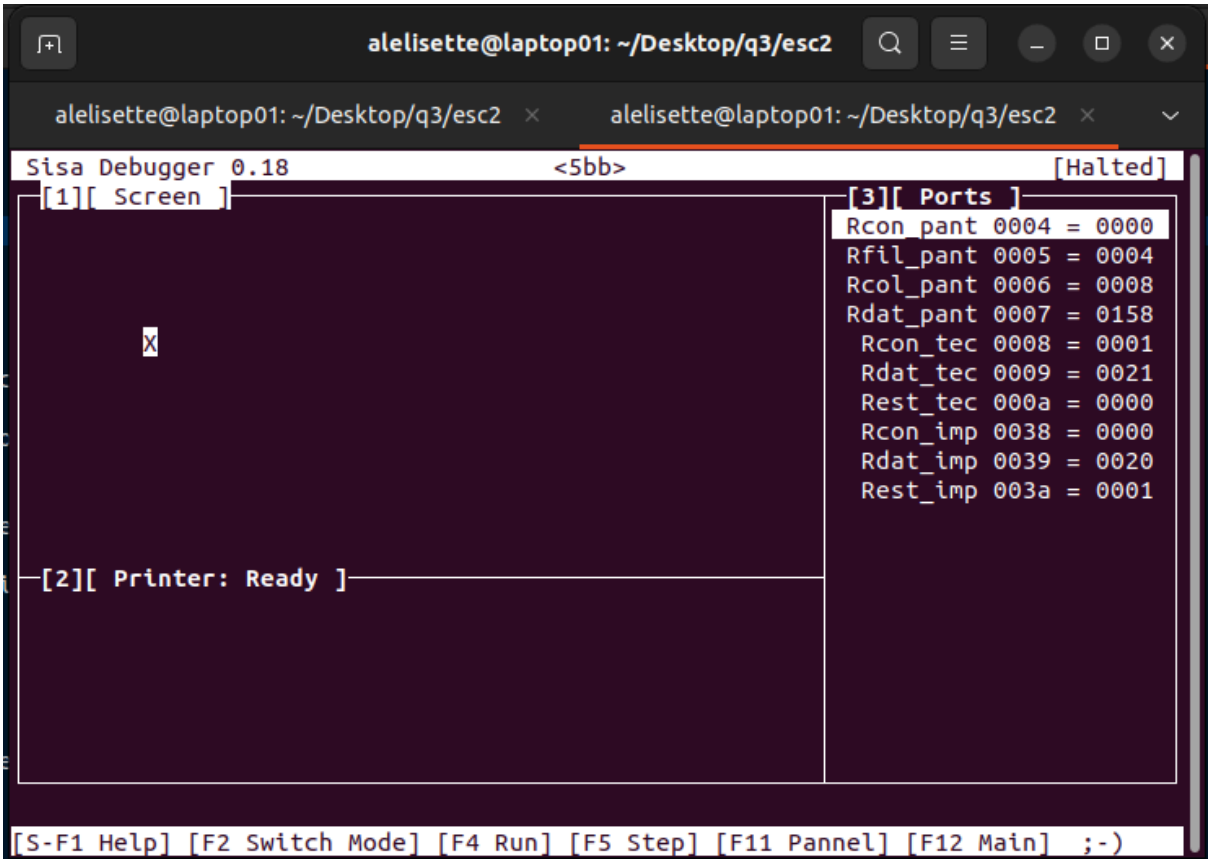
```
.include "macros.s"
.include "crt0.s"
.data
final:    .byte 0
.balign 2
.text

main:
        $MOVEI R0, interrupts_vector
        $MOVEI R1, teclat
        ST 1*2(R0), R1          ;codi id del teclat es 1*2=2
        MOVI R2, 1
        OUT Rcon_tec, R2
        $MOVEI R0, final       ;ens assegurem que final es igual a 0
        MOVI R2, 0
        STB 0(R0), R2           ;final=0
        EI    ;permet interrupcions

bucle:
```

```
        $MOVEI R0, final
        LDB R0, 0(R0)           ;tinc la tecla llegida en byte en R0
        BZ R0, bucle
        $MOVEI R3, 0x8000
        MOVI R4, 4
        OUT Rfil_pant, R4    ;fila=4
        MOVI R4, 8
        OUT Rcol_pant, R4    ;col=8
        $MOVEI R5, 0x0100  ;fem movei R5 i possem el codi en mode invers
        ADD R5, R0, R5
        OUT Rdat_pant, R5
        OUT Rcon_pant, R3   ;permet sortir per pantalla
        MOVI R2, 'X
        CMPEQ R2, R2, R0    ;comparem el valor dins de R0 que dins te el valor de la tecla
amb el caracter X
        BZ R2, main    ;salta a main si R2=0
        HALT


teclat:
        $MOVEI R3, tteclat
        IN R1, Rdat_tec
        ADD R1, R3, R1          ;&tteclat[tecla]
        LDB R1, 0(R1)
        $MOVEI R0, final
        STB 0(R0), R1
        JMP R6
```


alelisette@laptop01: ~/Desktop/q3/esc2
alelisette@laptop01: ~/Desktop/q3/esc2
alelisette@laptop01: ~/Desktop/q3/esc2
Sisa Debugger 0.18 <5bb> [Running]
[1][ Screen ]
U
[2][ Printer: Ready ]
[3][ Ports ]
Rcon_pant 0004 = 0000
Rfil_pant 0005 = 0004
Rcol_pant 0006 = 0008
Rdat_pant 0007 = 0155
Rcon_tec 0008 = 0001
Rdat_tec 0009 = 0011
Rest_tec 000a = 0000
Rcon_imp 0038 = 0000
Rdat_imp 0039 = 0020
Rest_imp 003a = 0001
[S-F1 Help] [F2 Switch Mode] [F4 Run] [F5 Step] [F11 Pannel] [F12 Main] ;-)


alelisette@laptop01: ~/Desktop/q3/esc2
alelisette@laptop01: ~/Desktop/q3/esc2
alelisette@laptop01: ~/Desktop/q3/esc2
Sisa Debugger 0.18 <5bb> [Running]
[1][ Screen ]
T
[2][ Printer: Ready ]
[3][ Ports ]
Rcon_pant 0004 = 0000
Rfil_pant 0005 = 0004
Rcol_pant 0006 = 0008
Rdat_pant 0007 = 0154
Rcon_tec 0008 = 0001
Rdat_tec 0009 = 000f
Rest_tec 000a = 0000
Rcon_imp 0038 = 0000
Rdat_imp 0039 = 0020
Rest_imp 003a = 0001
[S-F1 Help] [F2 Switch Mode] [F4 Run] [F5 Step] [F11 Pannel] [F12 Main] ;-)



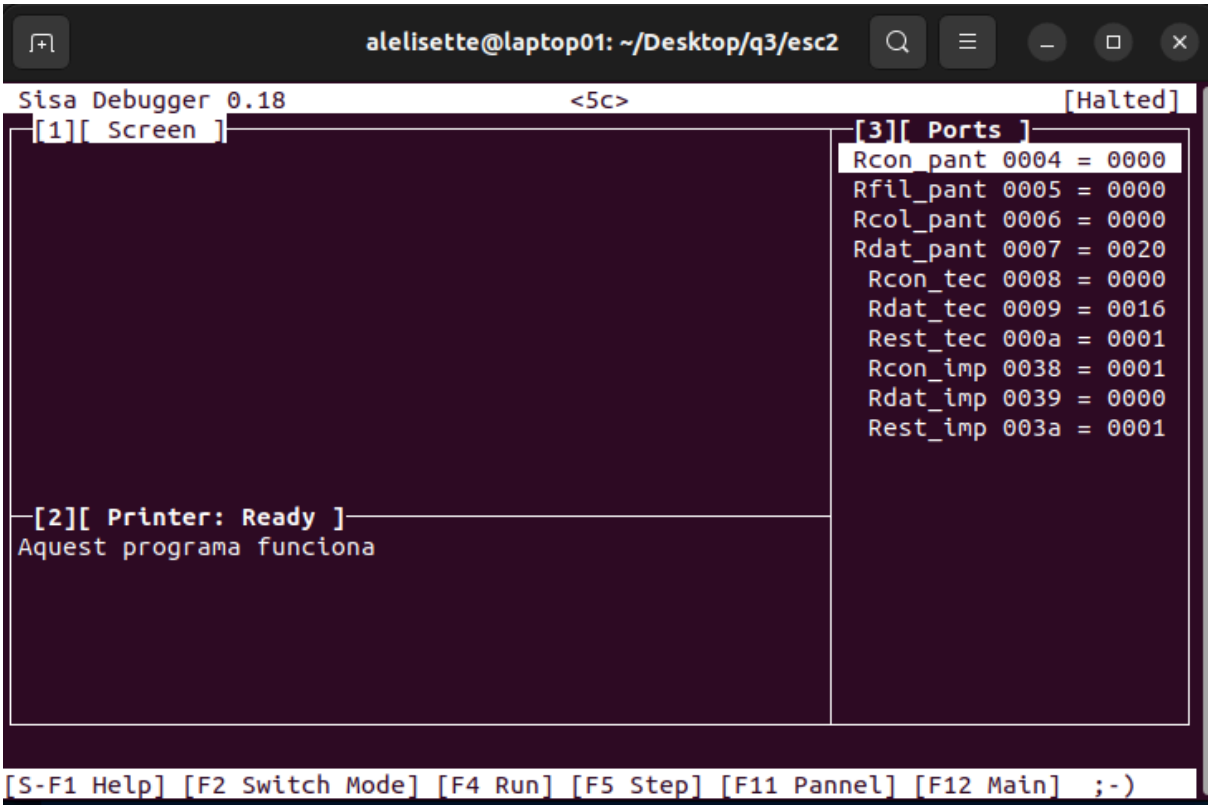




**Activitat 5.C: Rellotge i impressora**

**L'objectiu d'aquest apartat és provar l'accés a la impressora, i combinar-lo amb el rellotge. Tots dos dispositius s'han de sincronitzar per interrupcions. Completa l'exercici 5.4.**
**Exercici 5.4: Escriu un programa en alt nivell que escrigui el vector frase a la impressora, deixant que aquesta resti inactiva durant 1 segon entre lletra i lletra. Cal escriure el main i les dues RSI. Tingues en compte que el vector és un string, i acaba amb un byte que val 0.**
**char frase[32] = "Aquest programa funciona";**
**Tradueix el programa a SISA-F al fitxer s5c.s, i verifica'l amb el simulador. 1. Recorda que, encara que és molt improbable, el simulador pot generar errors de la impressora de forma aleatòria, fent que es quedi ocupada indefinidament. Si sospites que pot estar passant això (comprova-ho al registre d'estat, en el panell dret) sols has de polsar una tecla i la impressora es posarà preparada novament.**

```
.include "macros.s"
.include "crt0.s"
.data
        ticks: .word 0
        seg:   .byte 0
        .balign 2
        ready: .byte 0
        .balign 2
        v: .asciz "Aquest programa funciona"
        .fill 5,1,0
.text
main:
        $MOVEI R0, interrupts_vector
        $MOVEI R1, clock
        ST 0(R0), R1    ;codi id del clock
        MOVI R2, 1
        OUT Rcon_rel, R2     ;permet interrupcions del rellotge
        $MOVEI R1, printer
        ST 2*7(R0), R1
        MOVI R2, 1
        OUT Rcon_imp, R2    ;permet interrupcions
        EI                  ;permet interrupcions i crida a clock i printer
        $MOVEI R2, v ;R2 conte la adreca de v
        $MOVEI R0, seg    ;R0 conte la adreca de memoria de seg

bucle:

        LDB R4, 0(R0)
        BZ R4, bucle    ;si R4=0 salta a la etiqueta bucle
        $MOVEI R4, ready
        LDB R4, 0(R0)
        BZ R4, bucle    ;si ready es igual a 0 salta a la etiqeuta bucle
        MOVI R4, 0
        $MOVEI R5, seg
        STB 0(R5), R4           ;reiniciem els segons seg a 0
```

```
        $MOVEI R5, ticks
        STB 0(R5), R4           ;reiniciem els ticks a 0
        IN R5, Rest_imp
        LDB R4, 0(R2)           ;deso la lletra dins del registre4
        OUT Rdat_imp, R4
        $MOVEI R3, 0x8001
        OUT Rcon_imp, R3    ;el registre fa que imprimeixi la tecla
        ADDI R2, R2, 1          ;va canviant lletra per lletra
        MOVI R5, 0
        $CMPEQ R5, R4, R5 ;compara si v ha arribat a la darrera lletra
        BNZ R5, acaba                   ;si R5 no es igual a 0 salta a la etiqueta acaba
        BZ R5, bucle

clock:
        $MOVEI R0, ticks
        LD R1, 0(R0)
        ADDI R1,R1,1
        ST 0(R0), R1    ;incrementa ticks de un a un
        MOVI R2, 10
        $CMPGE R2, R1, R2 ;compara si R1 que es ticks es major i igual que 10
        BZ R2, fiCLK            ;si R2 es igual a 0 salta a la etiqueta anomenada fiCLK
        MOVI R1, 1
        $MOVEI R0, seg
        STB 0(R0), R1           ;assignem seg=true

fiCLK:
        JMP R6
acaba:
        HALT

printer:
        $MOVEI R0, ready
        MOVI R1, 1
        STB 0(R0), R1           ;assignem ready=true
        JMP R6
```




**Activitat 5.D: Rellotge i teclat**

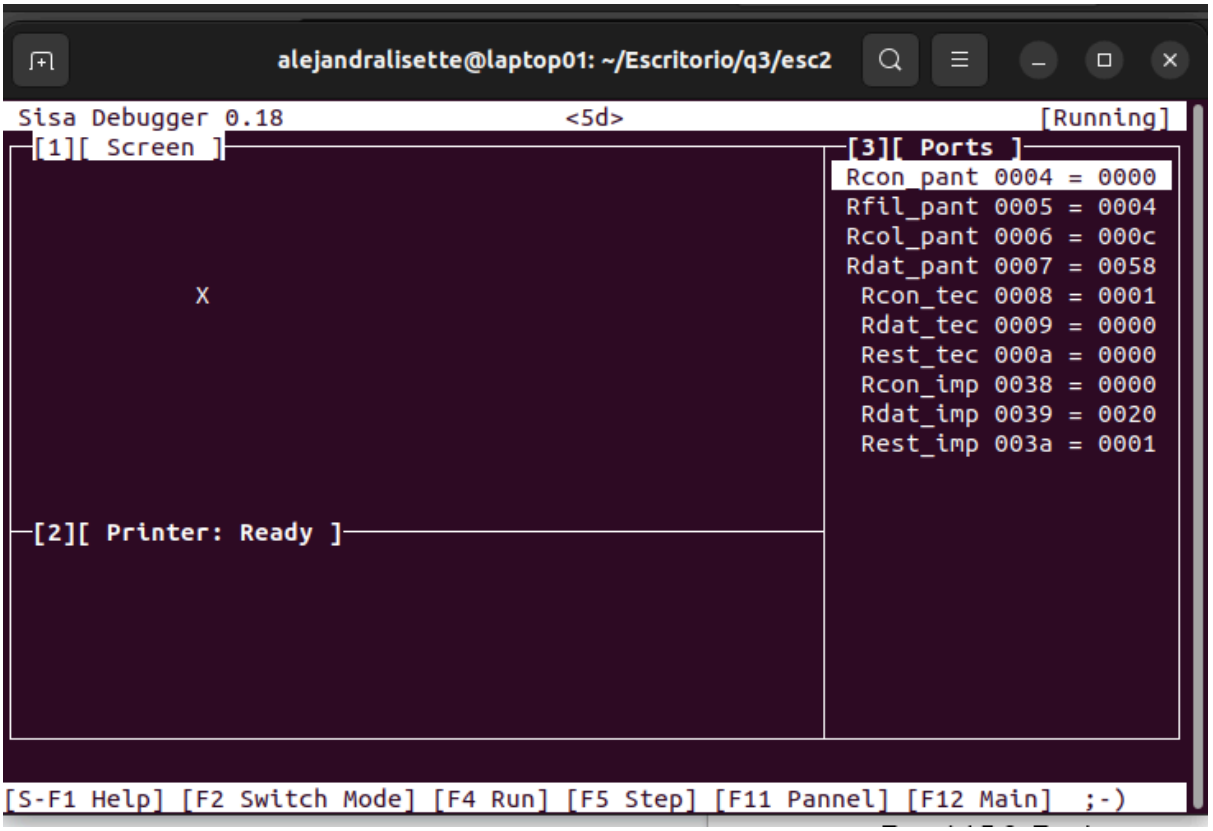
**L'objectiu d'aquest apartat és escriure un petit joc una mica més complex, i que faci ús dels dispositius de rellotge, teclat i pantalla. La sincronització serà per interrupcions.**

**Completa l'exercici 5.5 abans de continuar:**

**Exercici 5.5: Feu un programa en alt nivell que escrigui una 'X' a la posició [4,8] de la pantalla, en mode normal. Cada 0,4 segons, sobreescriurà la 'X' amb un espai en blanc (s'esborrarà) i la reescriurà a la posició adjacent (inicialment considerem adjacent la posició a la dreta de la posició anterior). El programa acaba quan la 'X' surt de la pantalla.**

**Tradueix el programa a SISA-F en el fitxer s5d.s, i verifica'l amb el simulador.**

**Un cop comprovat el programa anterior, completa l'exercici 5.6**




**Exercici 5.6: Escriu una nova versió del programa anterior, amb la següent modificació:**
**en comptes de moure la 'X' sempre cap a la dreta, volem que la direcció es pugui variar a voluntat per mitjà de les tecles 'A','S' (esquerra i dreta),'L' i 'P' (avall i amunt). Quan es polsi una d'aquestes tecles, la 'X' adoptarà la direcció corresponent. Nota: Les polsacions del teclat sols modifiquen la direcció, no "mouen" la 'X'. La 'X' sols es mou a intervals de temps. El programa acaba quan la 'X' se surt de la pantalla. Tingues ben clar quins son els distints casos que cal tractar, tant en la gestió del teclat com del rellotge.**

**Tradueix el programa anterior a assemblador, en el mateix fitxer s5d.s (afegint-hi les modificacions). Assembla'l i comprova'n el funcionament amb el simulador.**

```
.include "macros.s"
.include "crt0.s"
.data
ticks: .word 0
final:    .byte 0
          .balign 2
fila:     .word 4
columna:.word 8
tecla:    .byte 0
          .balign 2

.text
```

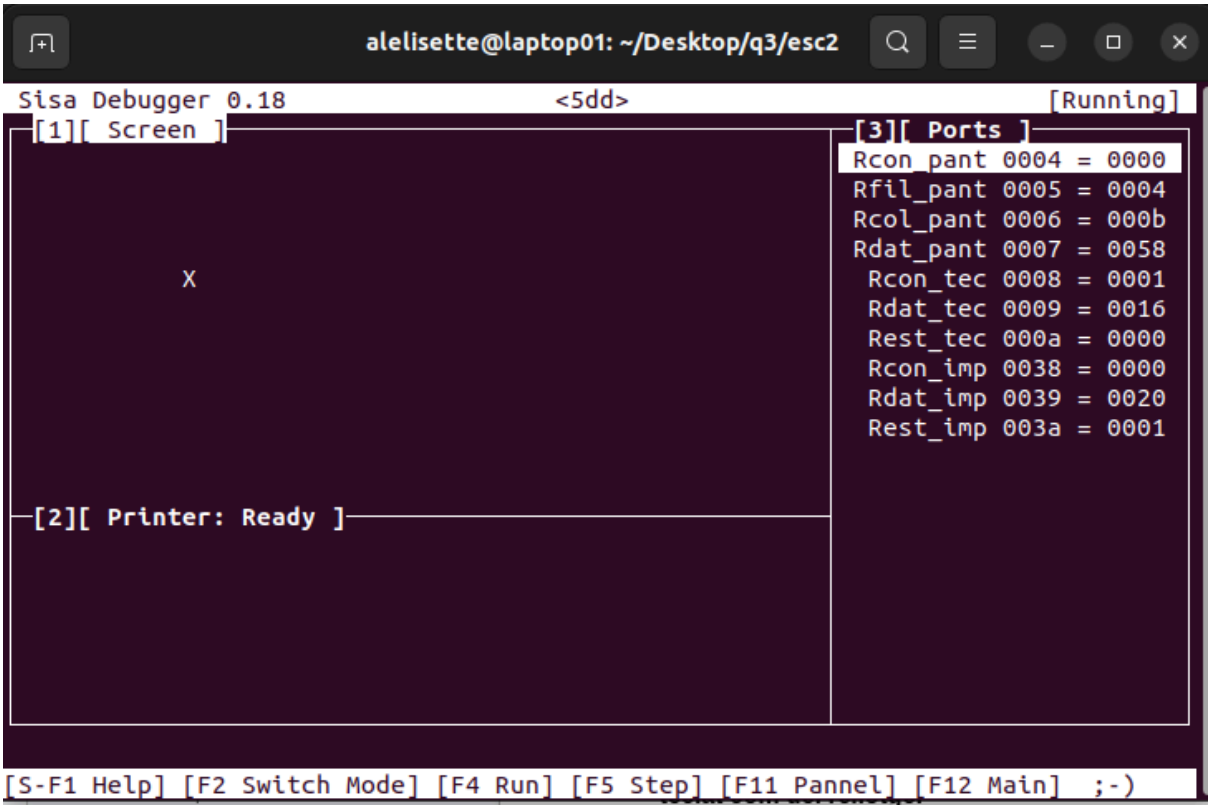
```
main:
        $MOVEI R0, interrupts_vector
        $MOVEI R1, clock
        ST 0(R0), R1    ;codi del clock=0
        MOVI R2, 1
        OUT Rcon_rel, R2     ;el registre permet interrupcions del rellotge
        $MOVEI R1, teclat
        ST 1*2(R0), R1          ;codi id del teclat es 1*2
        MOVI R2, 1
        OUT Rcon_tec, R2
        MOVI R2, 4
        OUT Rfil_pant, R2     ;fila 4
        MOVI R3, 8
        OUT Rcol_pant, R3    ;col 8
        EI


bucle:
        MOVI R1, 'X
        OUT Rdat_pant, R1
        $MOVEI R1, 0x8000
        OUT Rcon_pant, R1   ;surt per pantalla la X
        $MOVEI R1, tecla
        LDB R2, 0(R1)           ;carga dins de R2 la adreca de memoria de la 0+tecla
        MOVI R1, 'L     ;passa el caracter en ascii L a R1
        CMPEQ R4, R1, R2    ;compara si R1 es igual a R2 i el desa dins del registre R4
        BNZ R4, L               ;si R4!=0 salta a la etiqueta L
        MOVI R1, 'P
        CMPEQ R4, R1, R2
        BNZ R4, P
        MOVI R1, 'A
        CMPEQ R4, R1, R2
        BNZ R4, A
        $MOVEI R0, final
        LDB R0, 0(R0)           ;carrega el byte final dins de R0
        BZ R0, bucle        ;si final=0 salta a la etiqueta anomenada bucle
        $MOVEI R0, final
        MOVI R1, 0
        STB 0(R0), R1           ;assignem a la variable final=false
        $MOVEI R0, ticks
        STB 0(R0), R1           ;reiniciem ticks
        MOVI R1, 0x20
        OUT Rdat_pant, R1
        $MOVEI R1, 0x8000
        OUT Rcon_pant, R1   ;treu un espai per pantalla
        $MOVEI R1, columna
        LD R3, 0(R1)        ;carreguem la columna dins de R3
        ADDI R3, R3, 1          ;incrementem columna de un a un
```

```
        ST 0(R1), R3        ;desa el valor dins del registre on es troba la columna
        OUT Rcol_pant, R3
        MOVI R4, 64
        $CMPGE R4, R3, R4    ;compara si R3 es major o igual que 64
        BNZ R4, acaba        ;si R4 es diferent de 0 salta a la etiqueta acaba
        BZ R4, bucle         ;si R4 es 0 salta a la etiqueta bucle

A:
        $MOVEI R0, final
        LDB R0, 0(R0)        ;carrega el final
        BZ R0, bucle         ;si final es igual a 0 salta a la etiqueta bucle
        $MOVEI R0, final
        MOVI R1, 0
        STB 0(R0), R1        ;declarem false a final
        $MOVEI R0, ticks
        STB 0(R0), R1        ;reiniciem ticks
        MOVI R1, 0x20
        OUT Rdat_pant, R1
        $MOVEI R1, 0x8000
        OUT Rcon_pant, R1   ;treu per pantalla un espai
        $MOVEI R1, columna
        LD R3, 0(R1)        ;carreguem el valor en la columna
        ADDI R3, R3, -1      ;decrementem la columna de un a un
        ST 0(R1), R3
        OUT Rcol_pant, R3
        MOVI R4, 0
        $CMPLT R4, R3, R4    ;compara si R3 es menor que R4
        BNZ R4, acaba
        BNZ R1, bucle
L:
        $MOVEI R0, final
        LDB R0, 0(R0)
        BZ R0, bucle        ;si final es igual a 0 repeteix el bucle
        $MOVEI R0, final
        MOVI R1, 0
        STB 0(R0), R1        ;assignem false a final
        $MOVEI R0, ticks
        STB 0(R0), R1        ;reiniciem ticks
        MOVI R1, 0x20
        OUT Rdat_pant, R1
        $MOVEI R1, 0x8000
        OUT Rcon_pant, R1   ;treu per pantalla un espai
        $MOVEI R1, fila
        LD R3, 0(R1)        ;tenim carregat el valor de fila dins de R3
        ADDI R3, R3, 1       ;fila incrementa de un a un
        ST 0(R1), R3
        OUT Rfil_pant, R3
        MOVI R4, 16
```

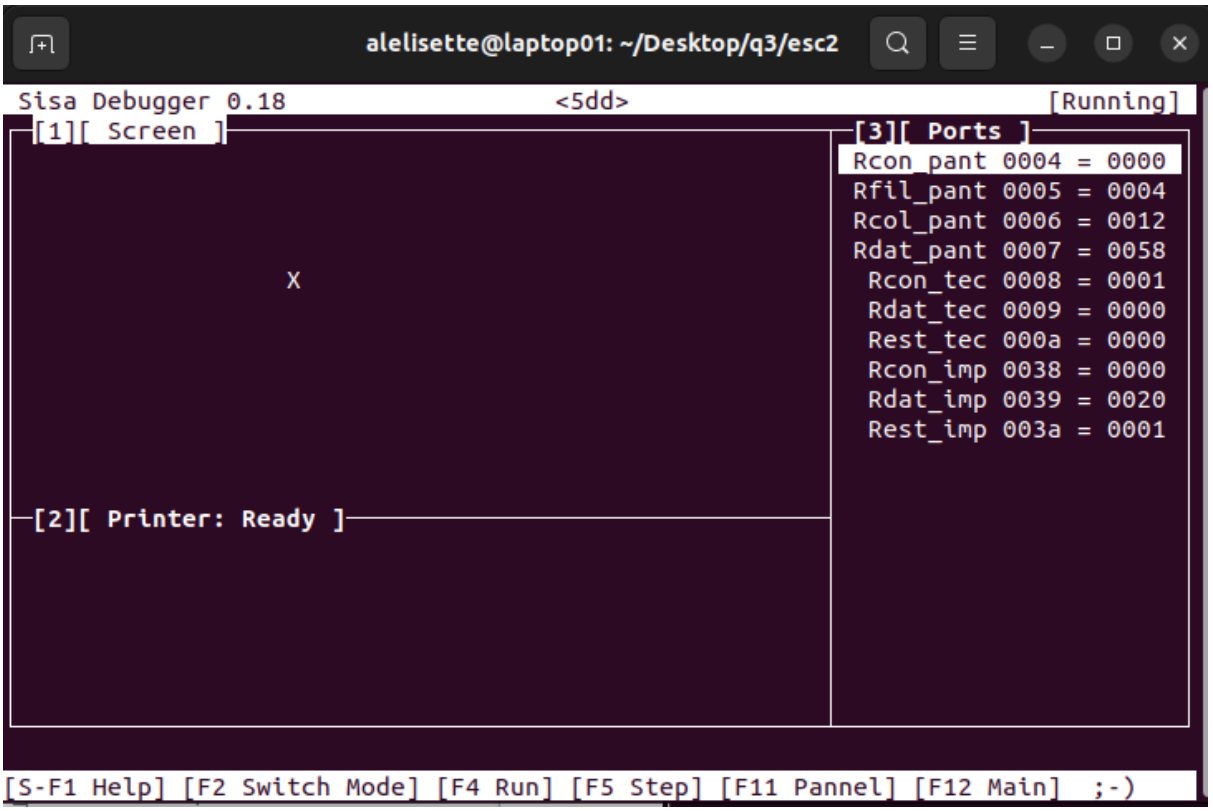
```
        $CMPGE R4, R3, R4
        BNZ R4, acaba
        BNZ R1, bucle


P:
        $MOVEI R0, final
        LDB R0, 0(R0)
        BZ R0, bucle        ;si final es igual a 0 salta a la etiqueta bucle
        $MOVEI R0, final
        MOVI R1, 0
        STB 0(R0), R1           ;assignem false a final
        $MOVEI R0, ticks
        STB 0(R0), R1           ;reiniciem ticks
        MOVI R1, 0x20
        OUT Rdat_pant, R1
        $MOVEI R1, 0x8000
        OUT Rcon_pant, R1
        $MOVEI R1, fila
        LD R3, 0(R1)
        ADDI R3, R3, -1         ;decremeta de un a un fila
        ST 0(R1), R3
        OUT Rfil_pant, R3
        MOVI R4, 0
        $CMPLT R4, R3, R4
        BNZ R4, acaba
        BNZ R1, bucle

acaba:
        HALT

clock:
        $MOVEI R0, ticks
        LD R1, 0(R0)
        ADDI R1,R1,1
        ST 0(R0), R1        ;ticks++
        MOVI R2, 4
        $CMPGE R2, R1, R2 ;compara si ticks es major o igual que 4
        BZ R2, fiCLK
        MOVI R1, 1
        $MOVEI R0, final
        STB 0(R0), R1           ;assignem true a final

fiCLK:
        JMP R6

teclat:
        $MOVEI R3, tteclat
```

```
IN R1, Rdat_tec          ;tecla
ADD R1, R3, R1           ;&tteclat[tecla]
LDB R1, 0(R1)
$MOVEI R0, tecla
STB 0(R0), R1            ;guardo en tecla la tecla polsada
JMP R6
```

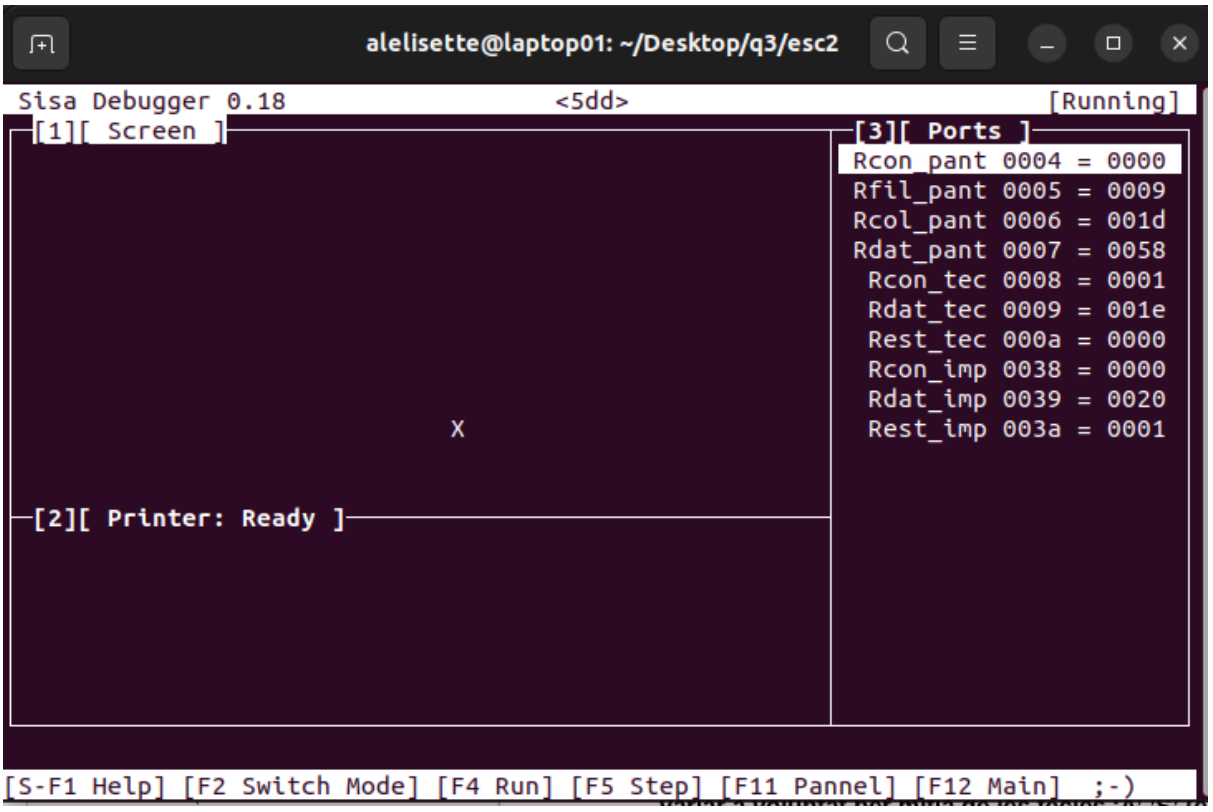
**Prenem la tecla 'A':**



**Prenem la tecla 'S':**




**Prenem la tecla 'L':**



**Prenem la tecla 'P':**