UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
ENGENHARIA DE COMPUTAÇÃO

Wagner Spinato Chittó

TRABALHO 1:

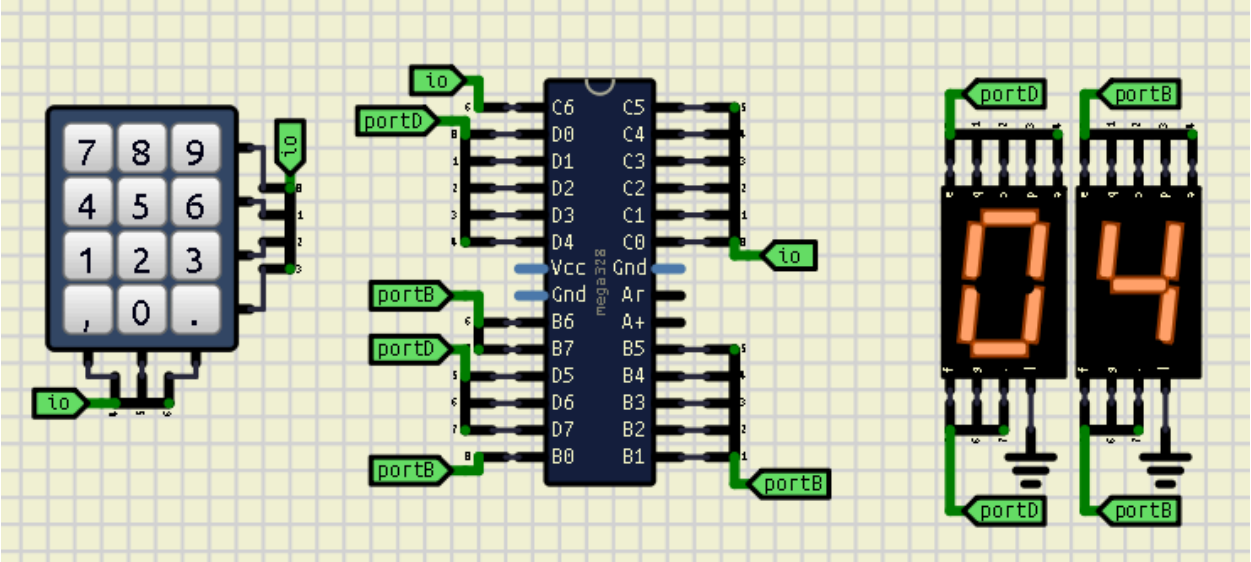
Implementar na prática (SIMULIDE), um teclado matricial, e mostrar o número lido,multiplicado pelo (ultimo numero da sua matricula +1), em 2 displays de 7 segmentos

Santa Maria, RS

19/04/2025

Componentes:

- ATmega328
- Teclado Matricial
    - Ligado pelo bus “io” nos pinos PC do ATmega
    - Linhas ligam nos bits PC[0..3]
    - Colunas ligam nos bits PC[4..6]
- Um display de 7 segmentos para dezenas decimais
    - Alimentado pelos pinos PD do ATmega
    - Ligado pelo bus de rótulo “portD”
- Um display de 7 segmentos para unidades decimais
    - Alimentado pelos pinos PB do ATmega
    - Ligado pelo bus de rótulo “portB”



Código:

```
start:
      ser r16
      out DDRD, r16          ; portD saida dezenas
      out DDRB, r16          ; portB saida unidades

      ldi r16, 0b11110000
      out PORTC, r16         ; portB liga pull up

      ldi r20, 0b1110000     ; r20 = colunas high, linhas low
      ldi r21, 0b0001111     ; r21 = colunas low, linhas high

      out DDRC, r21          ; R1-R4 como saidas | C1-C3 como entradas
      out PORTC, r20         ; portC liga pull up

loop:
      ; redefinicao das entradas apos o loop
      out DDRC, r21          ; R1-R4 como saidas | C1-C3 como entradas
      out PORTC, r20         ; portC liga pull up

      ; Todas colunas estao high
      ; Quando um botao eh pressionado, a coluna vai para low
      in r16, PINC
      com r16                ; r16 recebe bit da coluna ligada

      ; Adiantar a mudanca nas portas
      out DDRC, r20
      out PORTC, r21         ; Inversao das portas
      in r17, PINC
      com r17                ; r17 recebe bit da linha ligada

      ; Testar qual coluna esta ligada
      sbrc r16, 4
      jmp coluna4
      sbrc r16, 5
      jmp coluna5
      sbrc r16, 6
      jmp coluna6
      ; Nenhuma coluna ligada
      jmp loop

coluna4:
      ldi r18, 0x01
```

```
      jmp testelinha
coluna5:
      ldi r18, 0x02
      jmp testelinha
coluna6:
      ldi r18, 0x04
      jmp testelinha

testelinha:
      sbrc r17, 0
      jmp linha0
      sbrc r17, 1
      jmp linha1
      sbrc r17, 2
      jmp linha2
      sbrc r17, 3
      jmp linha3

      jmp loop         ; Nao encontrou linha = erro. volta pro loop

linha0:
      ldi r19, 0x01
      jmp fim
linha1:
      ldi r19, 0x02
      jmp fim
linha2:
      ldi r19, 0x04
      jmp fim
linha3:
      ldi r19, 0x08
      jmp fim

fim:
      ; Shift para gravar em r18 ambos os valores da coluna e da linha
      lsl r18
      lsl r18
      lsl r18
      lsl r18
      or r18, r19
      ; Mapa de entradas
      ; 1= 0b 0001 0100 = 0d 20
      ; 2= 0b 0010 0100 = 0d 36
      ; 3= 0b 0100 0100 = 0d 68
      ; 4= 0b 0001 0010 = 0d 18
      ; 5= 0b 0010 0010 = 0d 34
```

```
      ; 6= 0b 0100 0010 = 0d 66
      ; 7= 0b 0001 0001 = 0d 17
      ; 8= 0b 0010 0001 = 0d 33
      ; 9= 0b 0100 0001 = 0d 65
      ; 0= 0b 0001 1000 = 0d 40
      ; .= 0b 0010 1000 = 0d 24
      ; e= 0b 0100 1000 = 0d 72

      cpi r18, 0b00010100        ; tecla 1
      breq tecla1
      cpi r18, 0b00100100        ; tecla 2
      breq tecla2
      cpi r18, 0b01000100        ; tecla 3
      breq tecla3
      cpi r18, 0b00010010        ; tecla 4
      breq tecla4
      cpi r18, 0b00100010        ; tecla 5
      breq tecla5
      cpi r18, 0b01000010        ; tecla 6
      breq tecla6
      cpi r18, 0b00010001        ; tecla 7
      breq tecla7
      cpi r18, 0b00100001        ; tecla 8
      breq tecla8
      cpi r18, 0b01000001        ; tecla 9
      breq tecla9
      cpi r18, 0b00101000        ; tecla 0
      breq tecla0
      cpi r18, 0b00011000        ; tecla .
      breq tecladot
      cpi r18, 0b01001000        ; tecla e
      breq teclae

      jmp loop

tecla1:
      ldi r22, 1
      jmp mult
tecla2:
      ldi r22, 2
      jmp mult
tecla3:
      ldi r22, 3
      jmp mult
tecla4:
      ldi r22, 4
```

```
      jmp mult
tecla5:
      ldi r22, 5
      jmp mult
tecla6:
      ldi r22, 6
      jmp mult
tecla7:
      ldi r22, 7
      jmp mult
tecla8:
      ldi r22, 8
      jmp mult
tecla9:
      ldi r22, 9
      jmp mult
tecla0:
      ldi r22, 0
      jmp mult
tecladot:
      ldi r22, 0
      jmp displaydot1       ; display do “.” da esquerda
teclae:
      ldi r22, 0
      jmp displaydot2       ; display do “.” da direita

Mult:
      ;poderia usar multiplicacao mas como eh pra multiplicar por 4
vamos usar shift
      ;ldi r23, 4
      ;mul r22, r23
      ;mov r22, r0
      lsl r22
      lsl r22
      ; r22 contem o digito multiplicado por 4 (matricula[0] +1)

displayit:
      ; Conversao para display
      clr r23               ; registrador para dezenas
testedezena:
      cpi r22, 10
      brlo skipdezena
      subi r22, 10
      inc r23
      jmp testedezena
```

```
skipdezena:
      ; r23 (dezena) display para r25 em portD
      ; r22 (unidade) display para r24 em portB

      ; Mapear para display de 7 segmentos

testeportD:
      cpi r23, 1
      breq display1d
      cpi r23, 2
      breq display2d
      cpi r23, 3
      breq display3d
      cpi r23, 4
      breq display4d
      cpi r23, 5
      breq display5d
      cpi r23, 6
      breq display6d
      cpi r23, 7
      breq display7d
      cpi r23, 8
      breq display8d
      cpi r23, 9
      breq display9d
      cpi r23, 0
      breq display0d
displaydot1:
      ldi r25, 0b10000000
      ldi r24, 0x00
      jmp print
displaydot2:
      ldi r24, 0b10000000
      ldi r25, 0x00
      jmp print

display0d:
      ldi r25, 0b00111111
      jmp testeportB
display1d:
      ldi r25, 0b00000110
      jmp testeportB
display2d:
      ldi r25, 0b01011011
      jmp testeportB
```

```
display3d:
      ldi r25, 0b01001111
      jmp testeportB
display4d:
      ldi r25, 0b01100110
      jmp testeportB
display5d:
      ldi r25, 0b01101101
      jmp testeportB
display6d:
      ldi r25, 0b01111101
      jmp testeportB
display7d:
      ldi r25, 0b00000111
      jmp testeportB
display8d:
      ldi r25, 0b01111111
      jmp testeportB
display9d:
      ldi r25, 0b01101111
      jmp testeportB

testeportB:
      cpi r22, 1
      breq display1b
      cpi r22, 2
      breq display2b
      cpi r22, 3
      breq display3b
      cpi r22, 4
      breq display4b
      cpi r22, 5
      breq display5b
      cpi r22, 6
      breq display6b
      cpi r22, 7
      breq display7b
      cpi r22, 8
      breq display8b
      cpi r22, 9
      breq display9b
      cpi r22, 0
      breq display0b
```

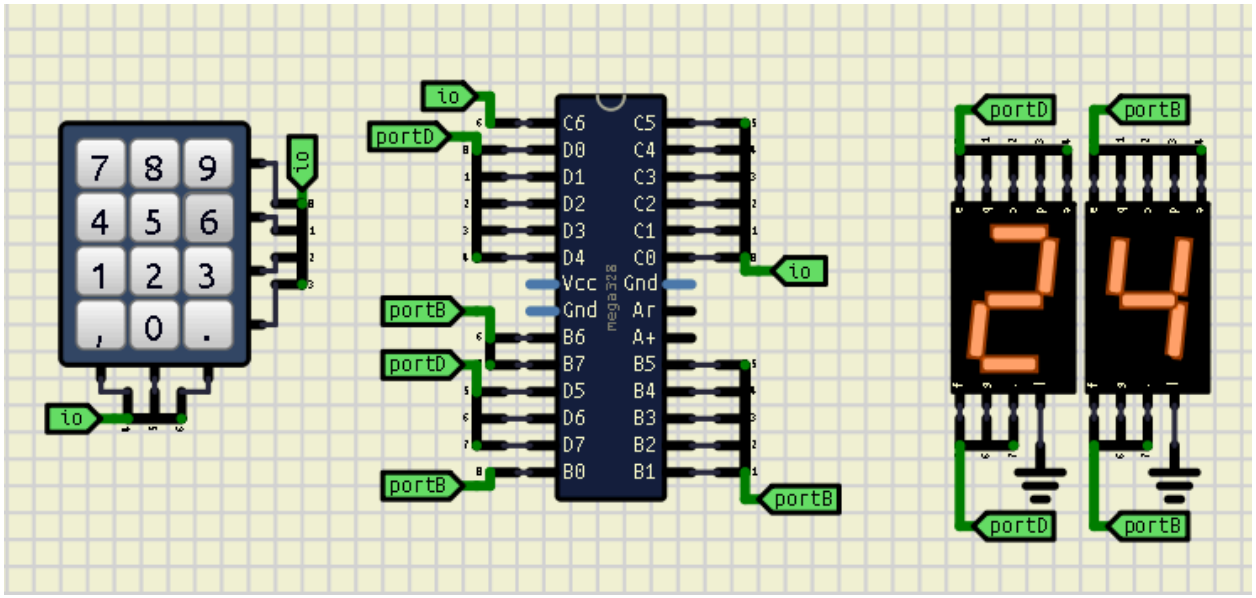
```
display0b:
      ldi r24, 0b00111111
      jmp print
display1b:
      ldi r24, 0b00000110
      jmp print
display2b:
      ldi r24, 0b01011011
      jmp print
display3b:
      ldi r24, 0b01001111
      jmp print
display4b:
      ldi r24, 0b01100110
      jmp print
display5b:
      ldi r24, 0b01101101
      jmp print
display6b:
      ldi r24, 0b01111101
      jmp print
display7b:
      ldi r24, 0b00000111
      jmp print
display8b:
      ldi r24, 0b01111111
      jmp print
display9b:
      ldi r24, 0b01101111
      jmp print

print:
      out PORTD, r25
      out PORTB, r24
      jmp loop
```

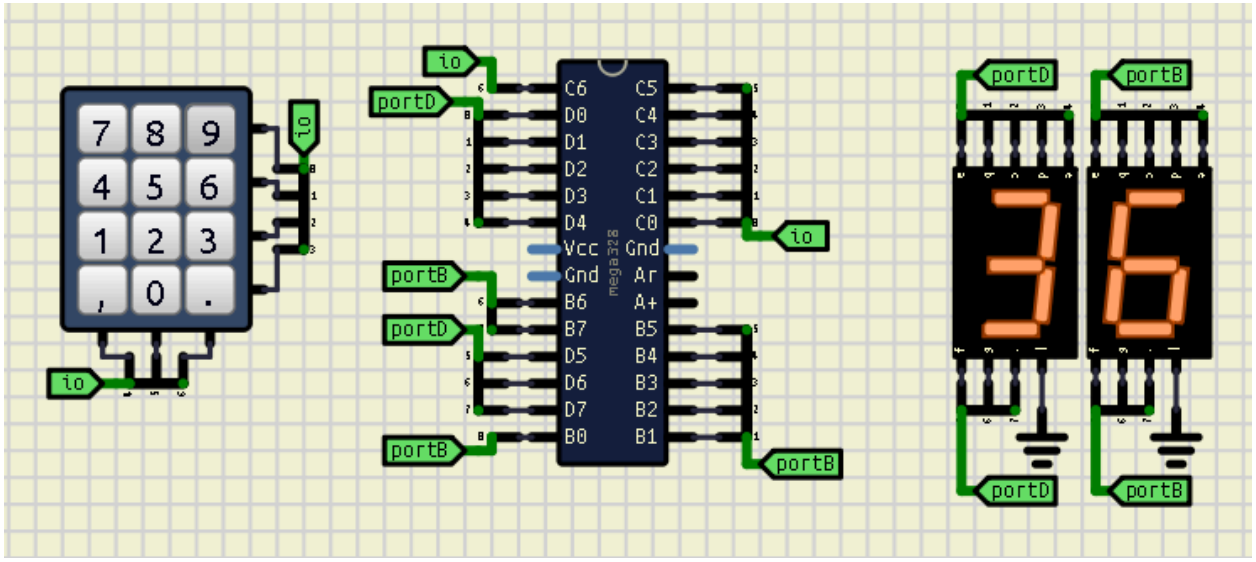
Vídeo do programa em execução:

https://youtu.be/XH7RYizQQAk

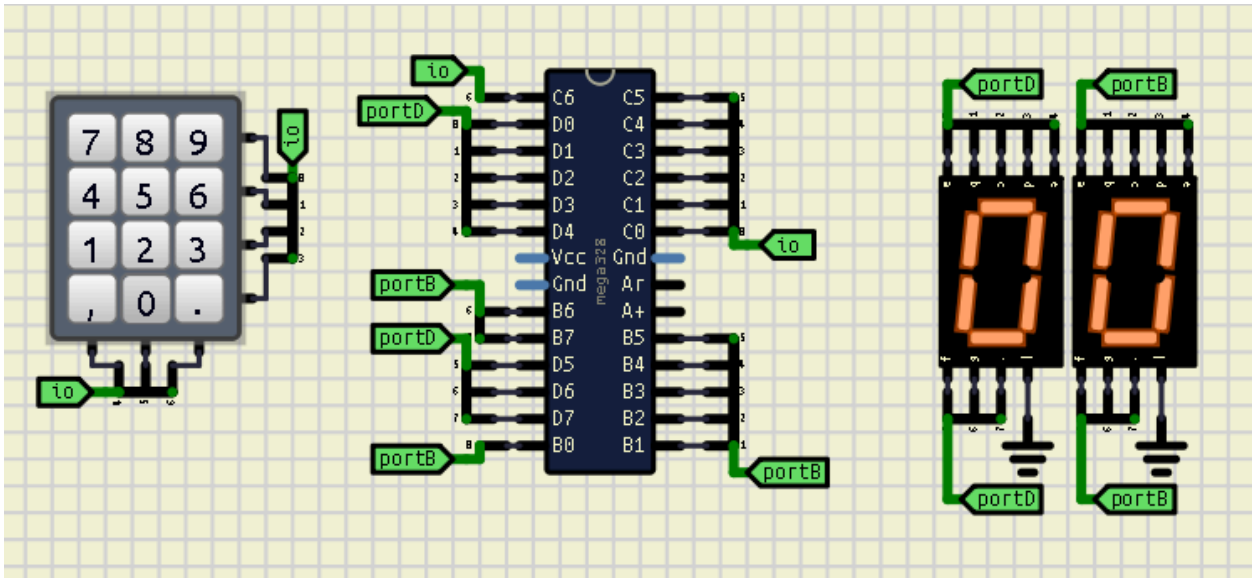
Fotos:



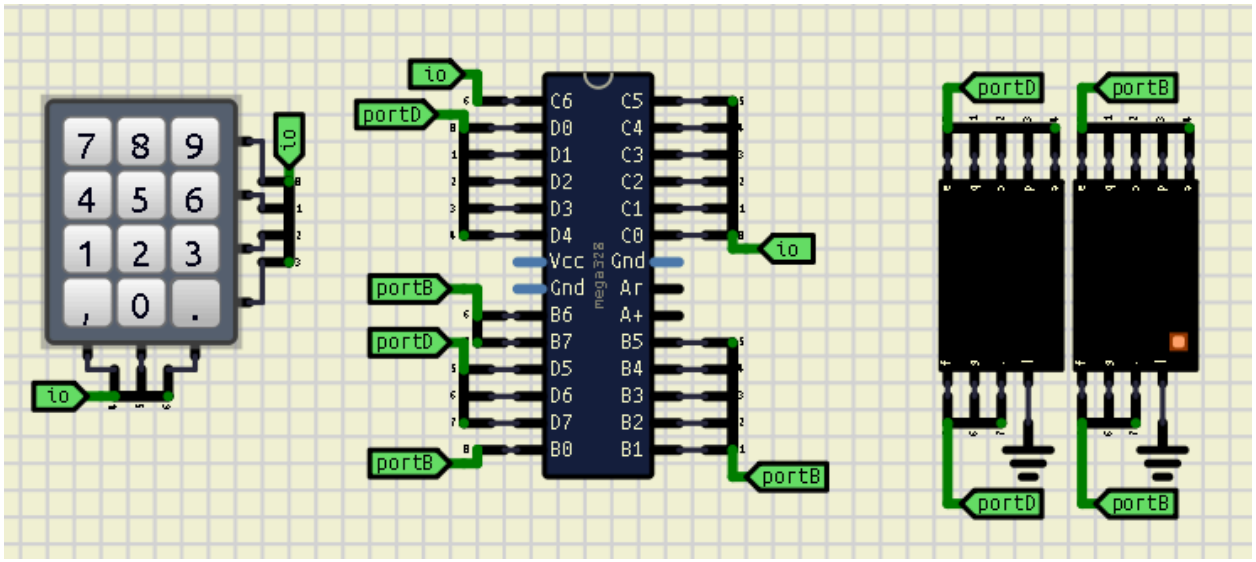
Input: 6
Output = 6x4 = 24



Input: 9
Output = 9x4 = 36



Input: 0
Output = 0x4 = 0



Input: “**.**”
Output = ”**.**” (direito)

Vídeo do programa em execução:

https://youtu.be/XH7RYizQQAk