UNIVERSIDADE FEDERAL DO CEARÁ
CURSO DE ENGENHARIA DE COMPUTAÇÃO
DISCIPLINA: MICROPROCESSADORES
PROFESSOR: MARCELO MARQUES SIMÕES DE SOUZA

TRABALHO N° 01 - PIC18F4550

Aluno: Emanuel Dêvid Paulino Felix **Matrícula: 469870**

Sobral – CE
2022.2

1 DESCRIÇÃO DO PROBLEMA

Inicialmente, foi proposto a implementação de um programa em Assembler para o PIC18F4550, que apresentasse no PORTD o valor de um número binário de 8 bits. O programa deveria ler o estado lógico de duas chaves, ou botões, e a cada segundo incrementar o valor do número de 8 bits se o botão 1 tivesse pressionado, decrementar esse mesmo número caso o botão 2 tivesse pressionado e ainda manter o estado do número caso nenhuma ou ambos os botões estivessem pressionados.

2 PROGRAMA EM ASSEMBLY

Inicialmente, foram declaradas as variáveis CONT, KEY1, KEY2, AUX1 e AUX2, onde KEY1 e KEY2 correspondem aos pinos RC0 e RC1, respectivamente do PORTC e foram, como era necessário, configuradas como pinos de inputs. Além disso, CONT foi usada para ser a variável que será incrementada ou decrementada e movida para o PORTD para que seja a saída apresentada neste projeto. Por fim, AUX1 e AUX2 foram usadas para auxiliar nas sub-rotinas de atraso, que foram aproveitas do código que o professor postou no primeiro exemplo do “Polling_Button_Blink_Led”.

A seguir, foi uma lógica usando um loops que testavam a cada 1 segundo os estados das chaves para que então, dependendo desses estados, fossem chamadas as devidas rotinas. Caso somente a chave 1 estivesse pressionada, era então chamada a rotina de incremento. Caso somente a chave 2 estivesse pressionada era chamada a rotina de decremento. Caso nenhuma ou ambas as chaves estivessem sendo pressionadas, o programa era direcionado para loop principal e tudo era começado de novo.

Finalmente, foram implementadas as rotinas ‘INCREMENT’ e ‘DECREMENT’, responsáveis por fazer o incremento e o decremento na variável CONT, respectivamente. Ademais, foram implementadas as sub-rotinas de atraso como dito anteriormente. A seguir são mostradas as imagens do código.

```
#include "p18f4550.inc"

CONFIG FOSC = XT_XT
CONFIG WDT = OFF
CONFIG LVP = OFF

VARIAVEIS UDATA_ACS 0 ; DEFININDO VARIÁVEIS
    CONT RES 1
    KEY1 EQU .0 ; PINO RC0
    KEY2 EQU .1 ; PINO RC1
    AUX1 RES 1
    AUX2 RES 1

RES_VECT  CODE    0x0000            ; processor reset vector
    GOTO    START                   ; go to beginning of program

; TODO ADD INTERRUPTS HERE IF USED

MAIN_PROG CODE                      ; let linker place main program

START
    CLRF CONT           ;INICIALIZANDO CONT1 COM 0
    CLRF AUX1           ;INICIALIZANDO AUX1 COM 0
    CLRF AUX2           ;INICIALIZANDO AUX2 COM 0

    CLRF TRISD          ;TORNANDO A PORTD COMO SAÍDA
    BSF TRISC, KEY1     ;CONFIGURANDO KEY1 COMO PORTA DE ENTRADA
    BSF TRISC, KEY2     ;CONFIGURANDO KEY2 COMO PORTA DE ENTRADA
    CLRF PORTD          ;COLOCANDO ZER0 EM TODOS OS PINOS DE PORTD
```

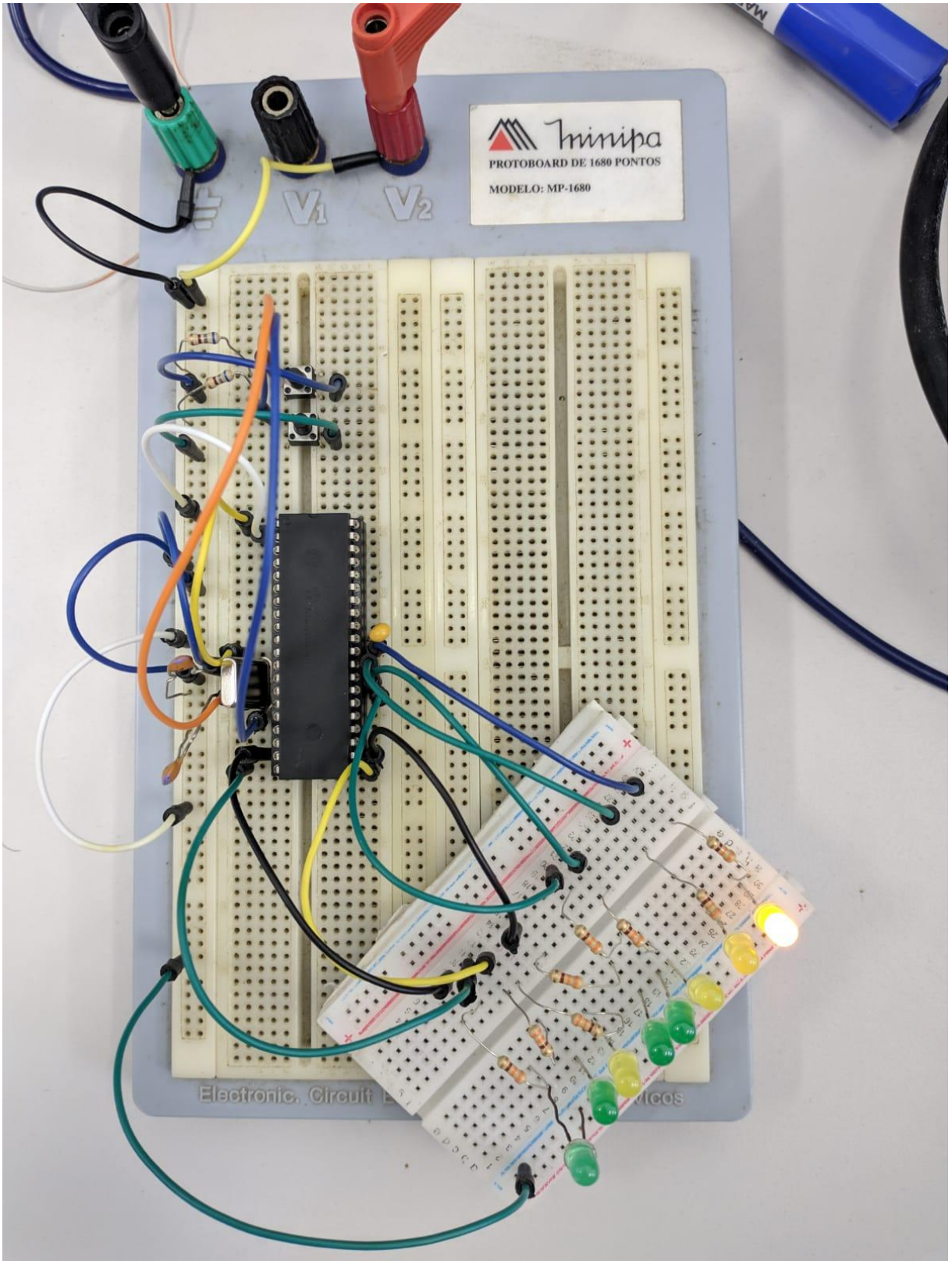
```
LOOP
    BTFSC PORTC, KEY1   ;VERIFICA SE KEY1 ESTÁ DESATIVADA, SE SIM PULA UMA INSTRUÇÃO
    BRA LOOPA           ;DESVIO PARA O LOOPA
    BRA LOOPB           ;DESVIO PARA O LOOPB

    LOOPA
    BTFSC PORTC, KEY2   ;VERIFICA SE KEY2 ESTÁ DESATIVADA, SE SIM PULA UMA INSTRUÇÃO
    BRA LOOPC   ;DESVIO PARA O LOOPC
    BRA LOOPD   ;DESVIO PARA O LOOPD

    LOOPB
    BTFSC PORTC, KEY2   ;VERIFICA SE KEY2 ESTÁ DESATIVADA, SE SIM PULA UMA INSTRUÇÃO
    BRA LOOPE   ;DESVIO PARA O LOOPE
    BRA LOOPF   ;DESVIO PARA O LOOPF

    LOOPC
    GOTO LOOP   ;DESVIO PARA O LOOP

    LOOPD
    CALL INCREMENT      ;CHAMA A ROTINA DE INCREMENTO
    GOTO LOOP   ;DESVIO PARA O LOOP

    LOOPE
    CALL DECREMENT      ;CHAMA A ROYINA DE DESCREMENTO
    GOTO LOOP   ;DESVIO PARA O LOOP

    LOOPF
    GOTO LOOP   ;DESVIO PARA O LOOP
```

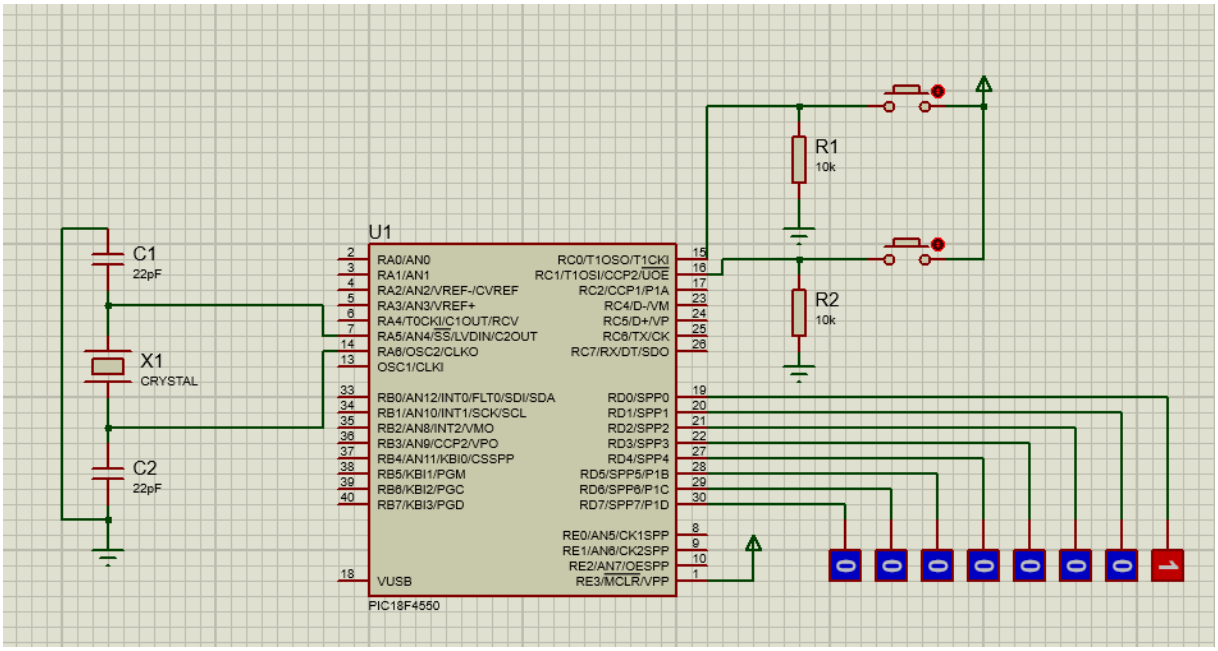
```
63      INCREMENT
64          CALL ATRASO_500ms
65          CALL ATRASO_500ms
66          BTFSC PORTC, KEY1
67          INCF CONT
68          MOVFF CONT, PORTD  ;MOVENDO CONT PARA O PORTD
69          RETURN
70
71      DECREMENT
72          CALL ATRASO_500ms
73          CALL ATRASO_500ms
74          BTFSC PORTC, KEY2
75          DECF CONT
76          MOVFF CONT, PORTD   ;MOVENDO CONT PARA O PORTD
77          RETURN
```

```
79      ATRASO_2ms ; Subrotina que gera atraso de 2 ms
80          MOVLW .152
81          MOVWF AUX2 ; Carrega contador para 200 interações
82      REPETE2       ; Inicio do loop
83          NOP       ; 10 NOPs -> 10 us
84          NOP
85          NOP
86          NOP
87          NOP
88          NOP
89          NOP
90          NOP
91          NOP
92          NOP
93          DECFSZ AUX2
94          GOTO REPETE2  ; Repete interações até que CONTA2 = 0
95          RETURN
96
97      ATRASO_500ms ; Subrotina que gera atraso de 500 ms
98          MOVLW .250
99          MOVWF AUX1 ; Carrega contador para 250 interações
100     REPETE_500   ; Inicio d92o loop
101         CALL ATRASO_2ms ; Espera 2 ms
102         DECFSZ AUX1
103         GOTO REPETE_500 ; Repete interações até que CONTA500 = 0
104         RETURN
105
106         END
```

3 MONTAGEM DO CIRCUITO E SIMULAÇÃO NO PROTEUS

A seguir é mostrado as imagens do circuito montada na protoboard, bem como no Proteus.





Abaixo segue o link do repositório GitHub, onde é possível encontrar as imagens com uma qualidade melhor, os arquivos criados no Proteus e no MPLAB e ainda um vídeo feito no laboratório mostrando o funcionamento do circuito e um que mostra a simulação do mesmo no Proteus:

https://github.com/EmanuelDevid/Microprocessadores.git