

**DISCIPLINA: LABORATÓRIO ORGANIZAÇÃO E ARQUITETURA DE COMPUTADORES**

**RELATÓRIO: LABORATÓRIO 03**

**(2022)**

**ALUNO: Lucas Fausto Medeiros**

**Professor.: Dr. Misael Elias de Morais**

CAMPUS CAMPINA GRANDE, julho de 2022

1. Resumo:

Neste Experimento foi possível visualizar alguns dos conceitos base e importantes na programação baixo nível dos micro-controladores (MCS51), Assembly, pelo Simulador MCY8051 IDE. Ademais, a atividade que foi proposta no Lab. em questão, foi a demonstração do conjunto de instrução do microcontrolador e seus equipamentos. Neste, fizemos o acendimento de LED’s e leitura de teclas ou botões.

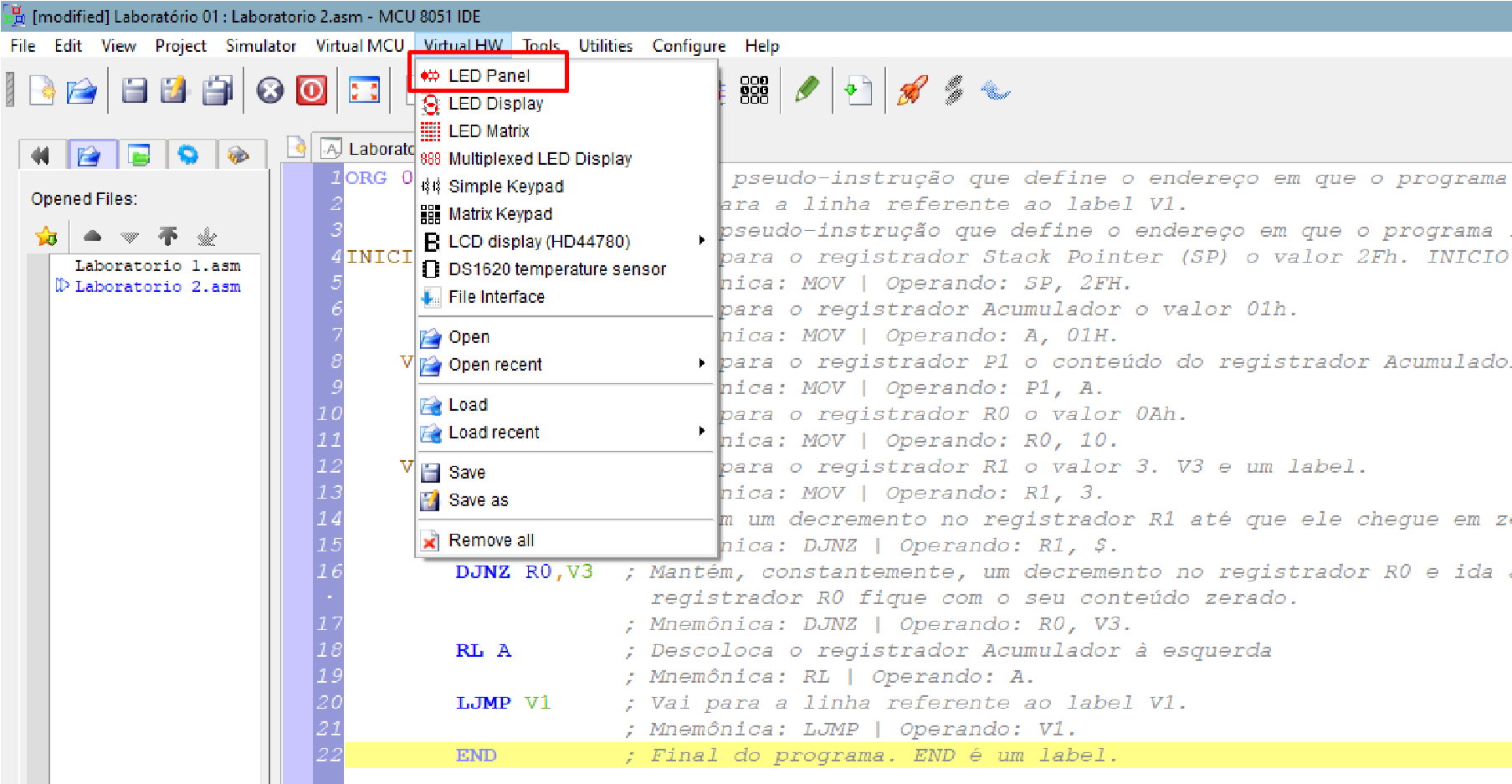
1. Introdução:

O Laboratório III tem o objetivo apresentar o estudo da linguagem Assembly a partir de consultas em seu DataSheet, que nada mais é que uma folha com dados e especificações técnicas de seu desempenho levando em conta o microcontrolador MCS51.

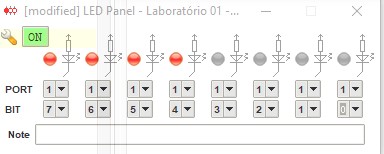
1. Métodos e Materiais:
   1. Objetivo: O experimento teve como objetivo mostrar na pratica o estudo pratico da programação em Assembly, que se fez possível estudar a rotação de led’s além da leitura de dados no simulador MCU8051 IDE.

* 1. Hardware e Software necessários: Computador com o MCU8051 IDE instalado.

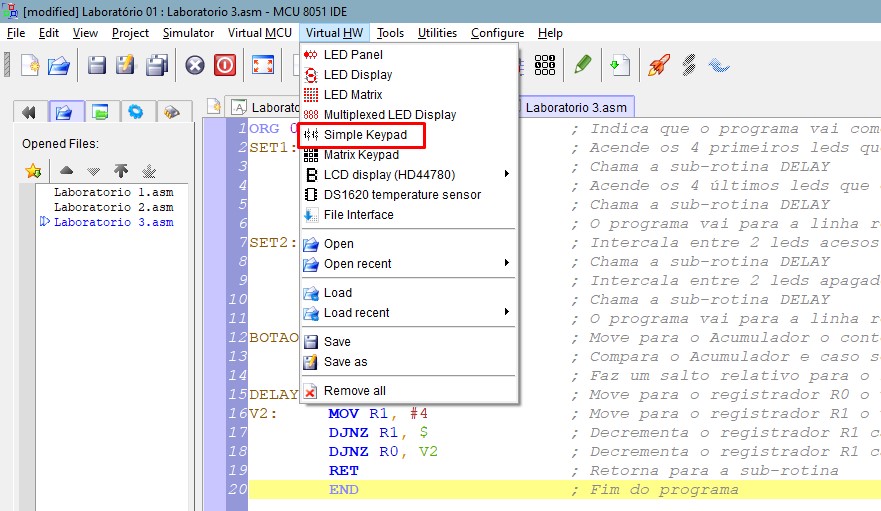
* 1. Procedimentos:
     1. Ligação do painel de LED: Inicialmente, tem de se ligar as portas e seus respectivos bits no painel de led’s do simulador. Como mostra a imagem abaixo:



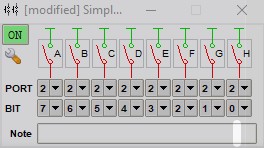
A ativação do painel é feita ativando e relacionando cada um dos oito led’s a uma porta e um bit do microcontrolador.



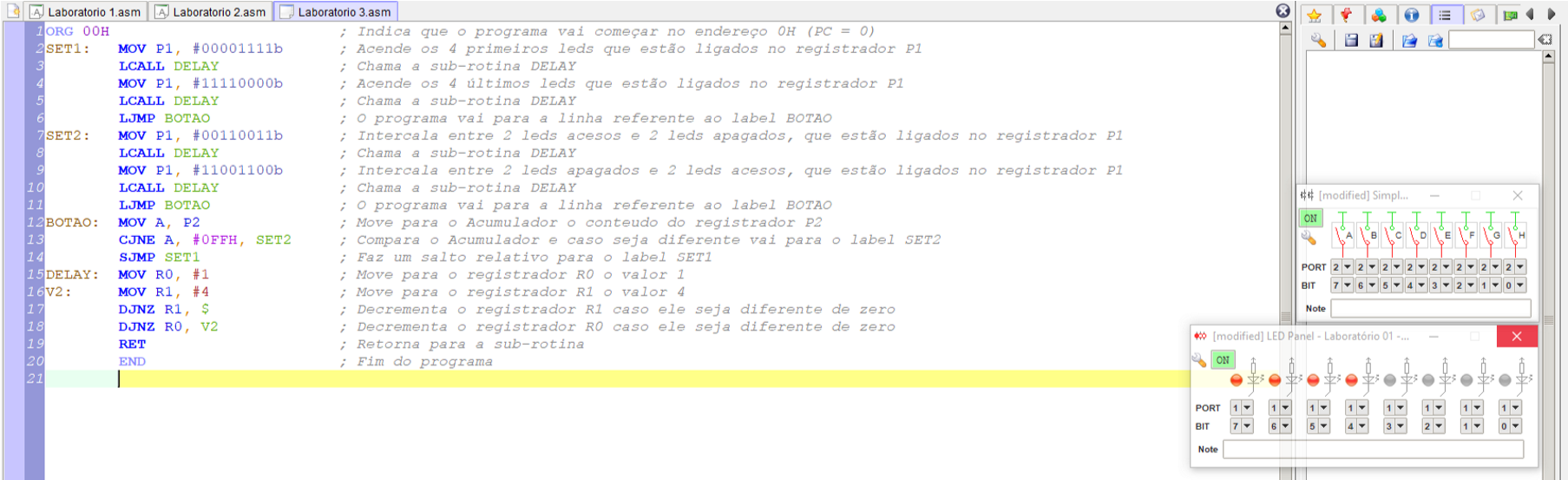
* + 1. Ligação do painel de teclas: Inicialmente, para fazer a ligação das portas e seus respectivos bits no painel teclas. Como mostra a imagem abaixo.



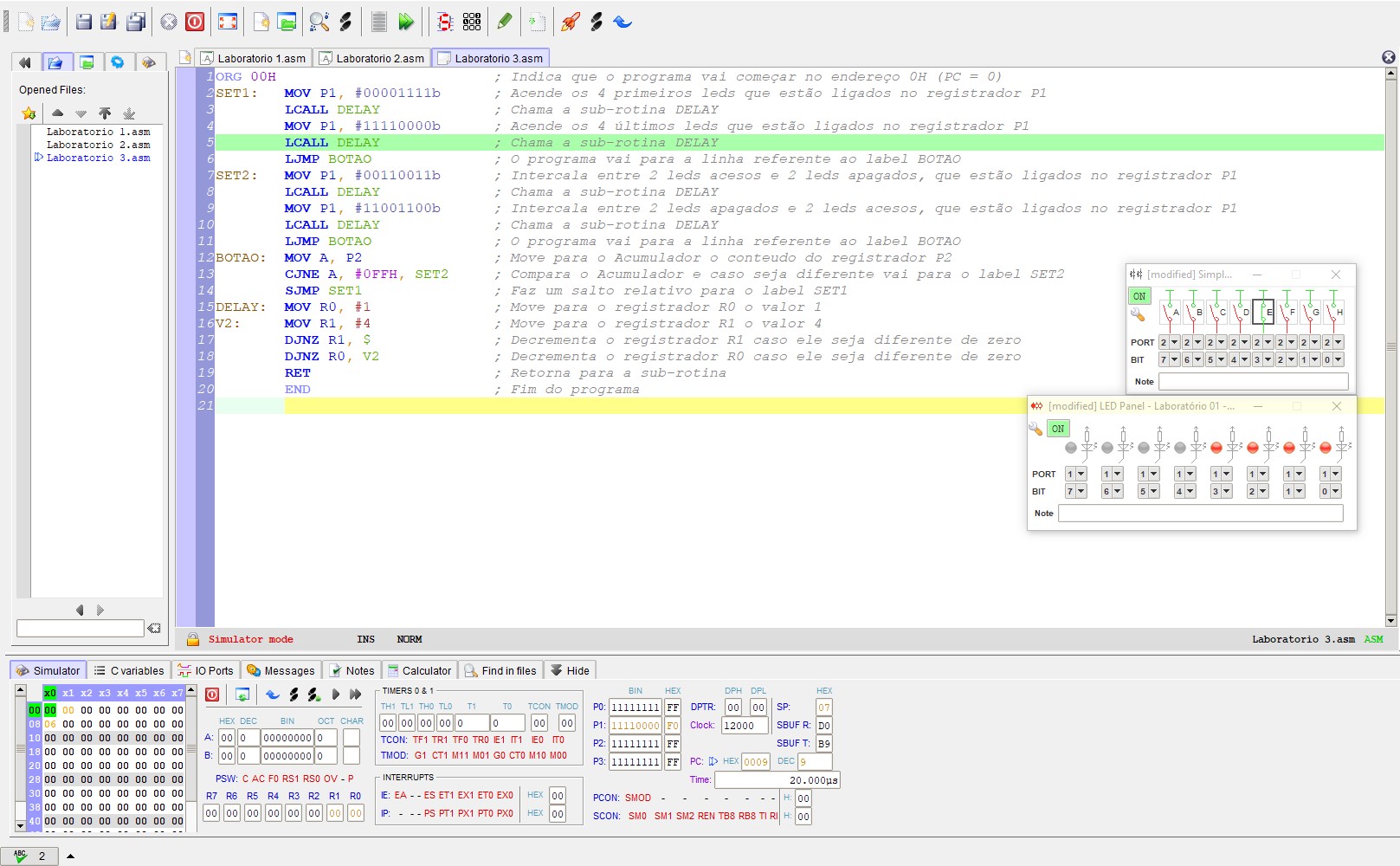
A configuração do painel é feita ativando e relacionando cada um dos oito led’s a uma porta do micro.



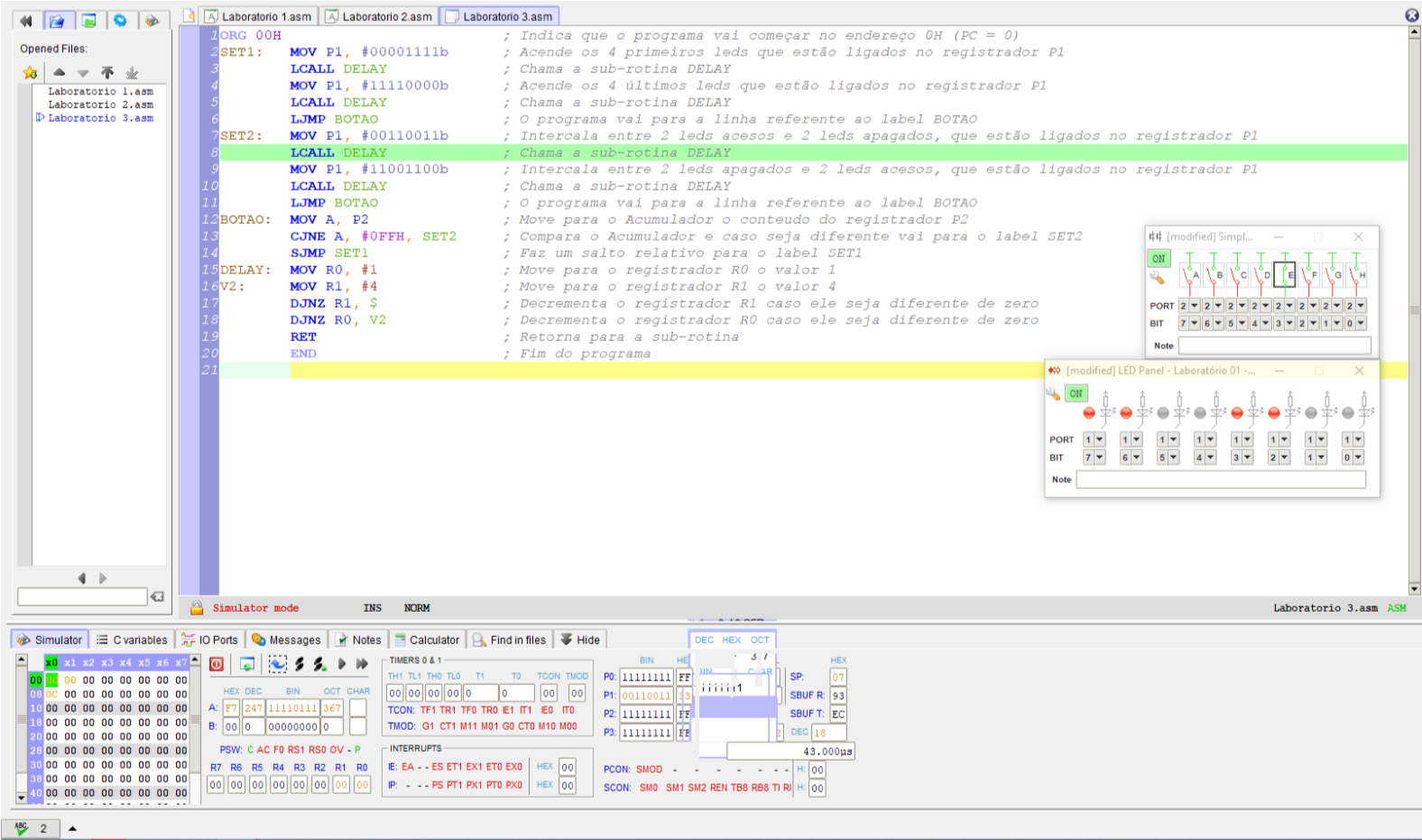
* + 1. Listagem do programa em LM:
       1. **ORH 00H:** Pseudo-instrução que define o endereço que o programa irá começar e continuará.
       2. **MOV P1, #00001111 b:** Move para o registrador P1 o valor #00001111 b.
       3. **LCALL DELAY:** Chamada de sub-rotina DELAY.
       4. **INICIO: MOV SP, #2FH:** Move para o registrador SP o valor de #2FH.
       5. **MOV P1, #11110000 b:** Move para o registrador P1 o valor de #11110000 b.
       6. **LJMP BOTAO:** Instrui que o programa vá para a linha referente ao label BOTAO.
       7. **MOV P1, #00110011 b:** Move para o registrador P1 o valor de #00110011 b.
       8. **MOV P1, #11001100 b:** Move para o registrador P1 o valor de #11001100 b.
       9. **MOV A ,P2:** Move para o Acumulador o conteúdo do registrador P2.
       10. **CJNE A, #0FFH, SET2:** Compara o conteúdo do Acumulador, caso não sejam iguais, vai para o label SET2.
       11. **SJMP SET1:** Faz um salto relativo para o label SET1.
       12. **MOV R0, #1:** Move para o registrador R0 o valor de 1.
       13. **MOV R1, #4:** Move para o registrador R1 o valor de 4.
       14. **DJNZ R1, $:** Decrementa o registrador R1 casa seu conteúdo seja diferente de 0.
       15. **DJNZ R0, V2:** Decrementa do registrador R0 caso seu conteúdo seja diferente de 0;
       16. **RET:** Retorna o programa para a sub-rotina.
       17. **END:** Fim do programa.
    2. Funcionamento do programa: Descrição com detalhes de cada linha do programa que foi utilizado no LAB. Com comentário de cada linha.



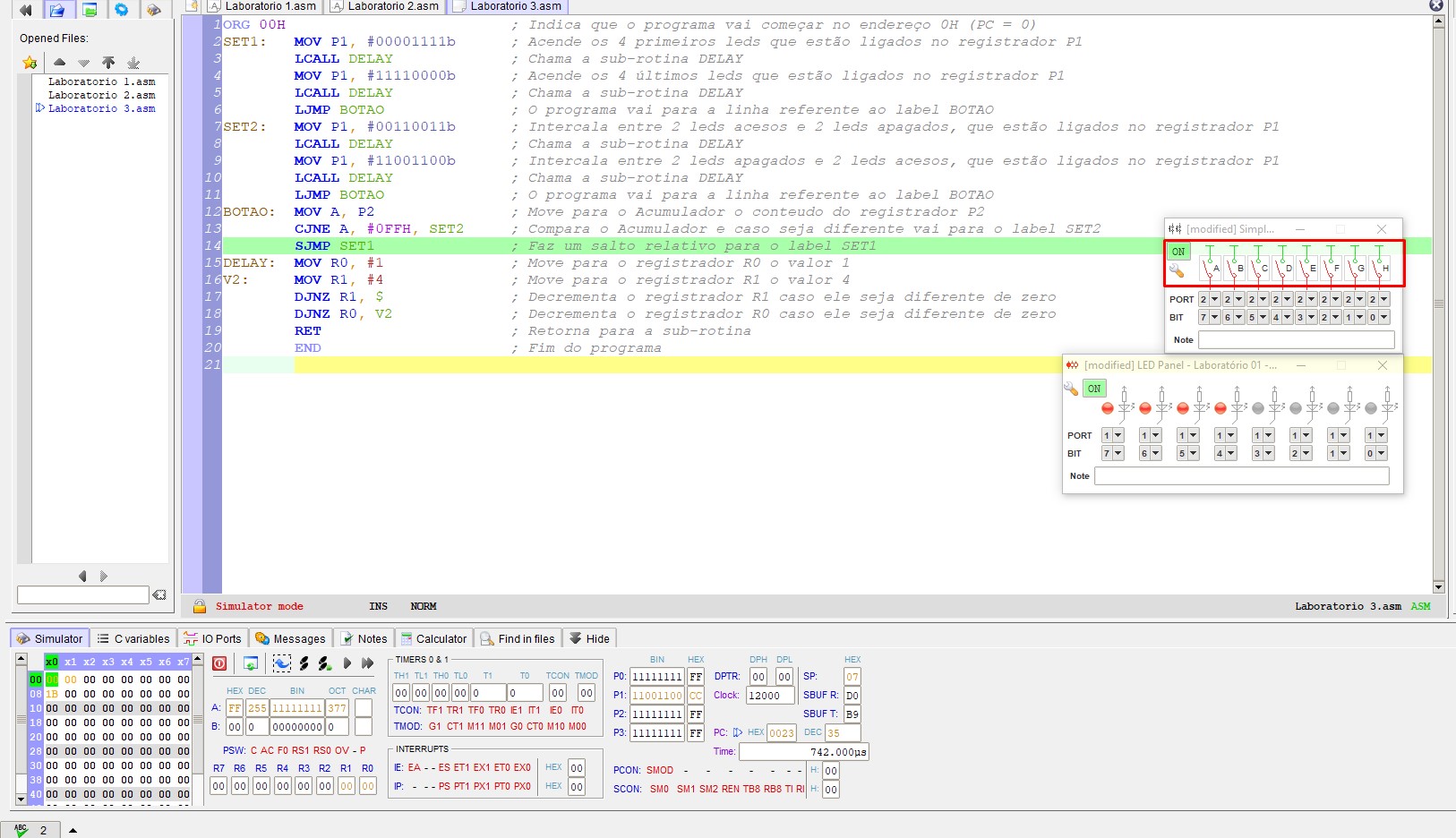
No primeiro passo, o programa irá executar as duas primeiras linhas, o painel de led’s vai acender conforma descrito no comentário da linha 2. No segundo passo, o programa vai cair na sub-rotina definida pelo label **DELAY** que fará com que a próxima mudança no painel de led’s ocorra após um intervalo de tempo definido, Logo após, o programa voltará para a sub-rotina que vai inverter o estado dos led’s ligados ao registrador **P1**. No fim, ao executar o comando **LJMP BOTAO**, linha 6, o acesso ao label **BOTAO** será liberado.



No segundo passo, o programa ao chegar ao label **SET2**, ocorre o comando **MOV P1 #00110011 b**, que tem a função de alterar o padrão dos led’s que estão acesos e desligados, logo após, a sub-rotina **DELAY** é acionada de novo, ao fim da mesma o programa irá executar o comando **MOV P1, #11001100 b**, e vai novamente para a sub-rotina. Por fim, quando chegar ao comando **LJMP BOTAO**, o programa vai acessar a posição do programa que o label **BOTAO** define.

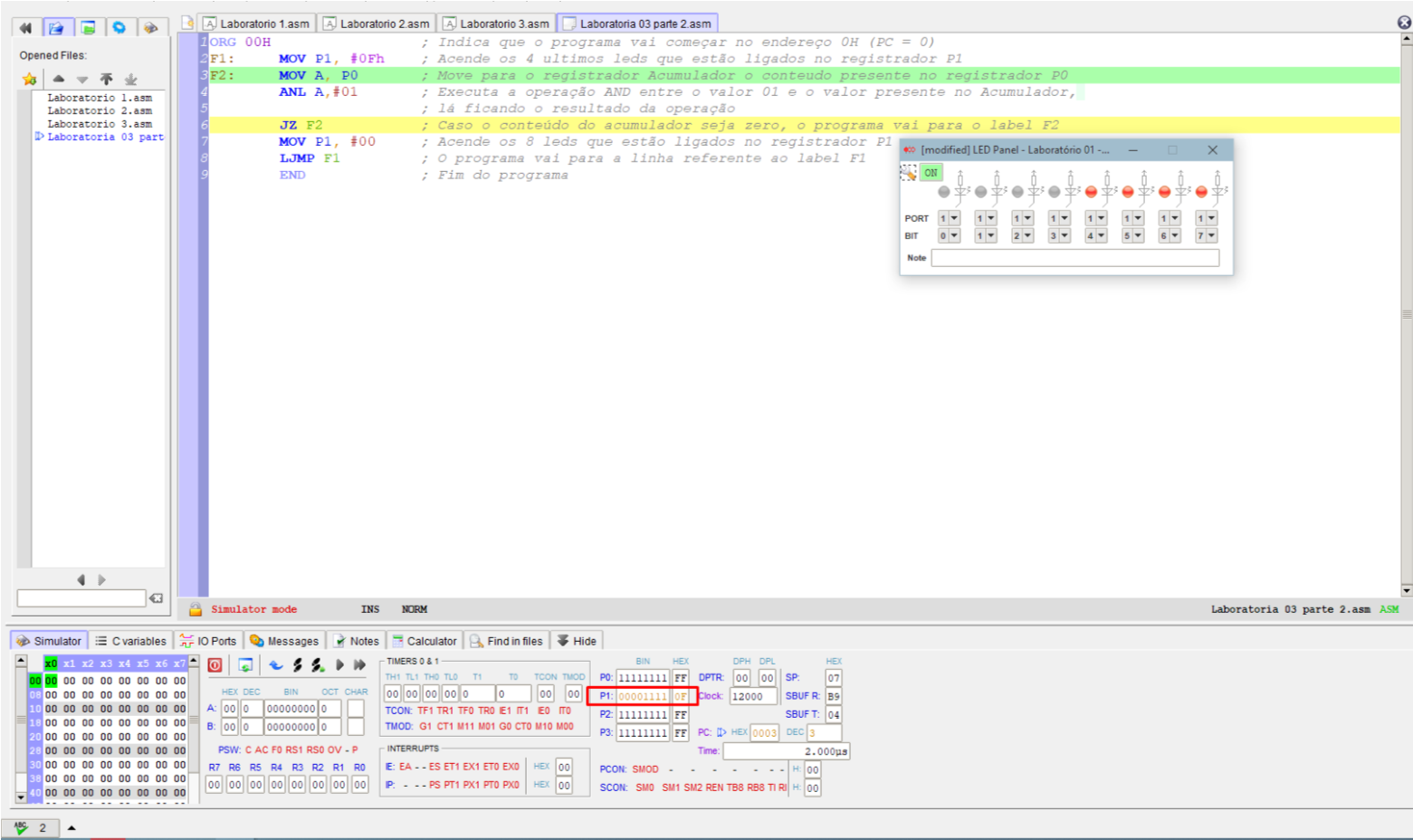


No terceiro passo, será possível ver um método boolean do programa no label **BOTAO**, que define a alteração dos led’s em 4 – 4, ou em 2 – 2. Inicialmente o programa pegará o conteúdo do registrador **P2** e vai move-lo para o acumulador. Logo após, se o dado que foi enviado for diferente de **0FFH**, isto é, se algum botão do teclado virtual for pressionado, o valor será alterado. O comando **CJNE A, SET2N** faz com que o programa vá até o label definido por **SET2**, caso contrário, será passado para instrução **SJMP SET1**, que caminhará para o local que **SET1** está definindo.

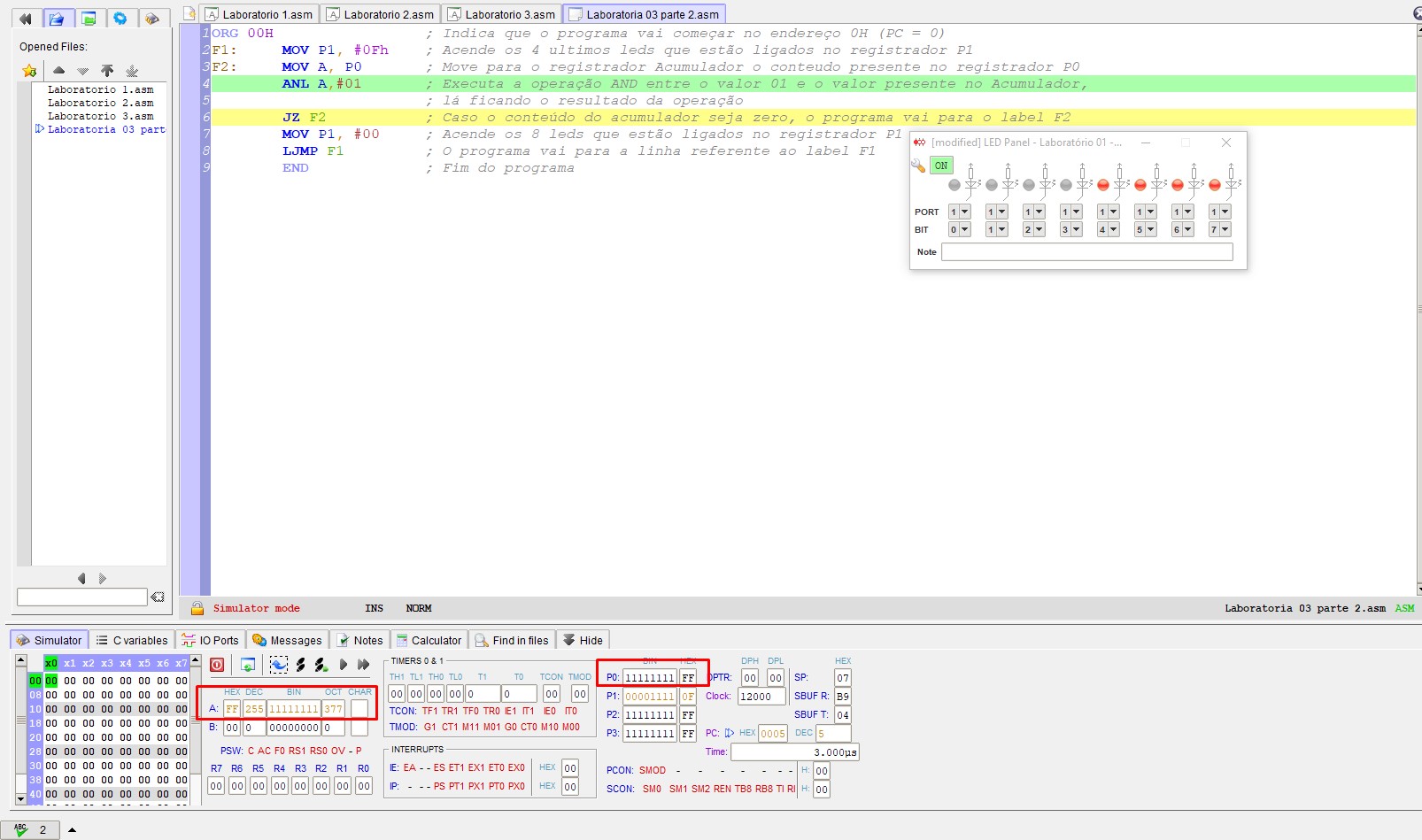


* + 1. Funcionamento do programa fornecido:

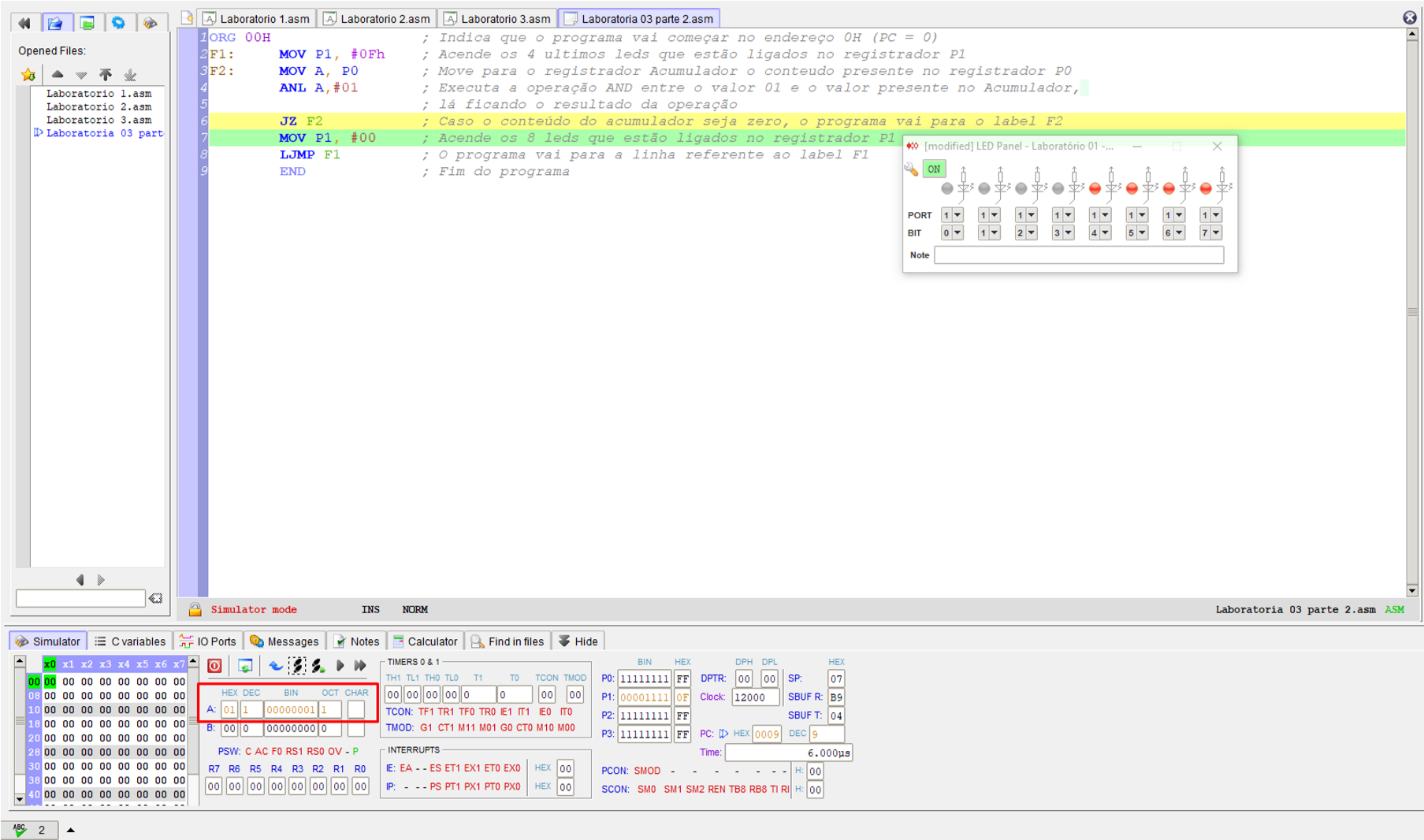
* + - 1. No primeiro passo, o programa inicia movendo o valor **00001111 b** para o registrador **P1**, que tem como função ligar ou desligar os led’s do painel. Após ser iniciado, fica possível notar que os 4 últimos led’s foram acesos por conta da instrução.



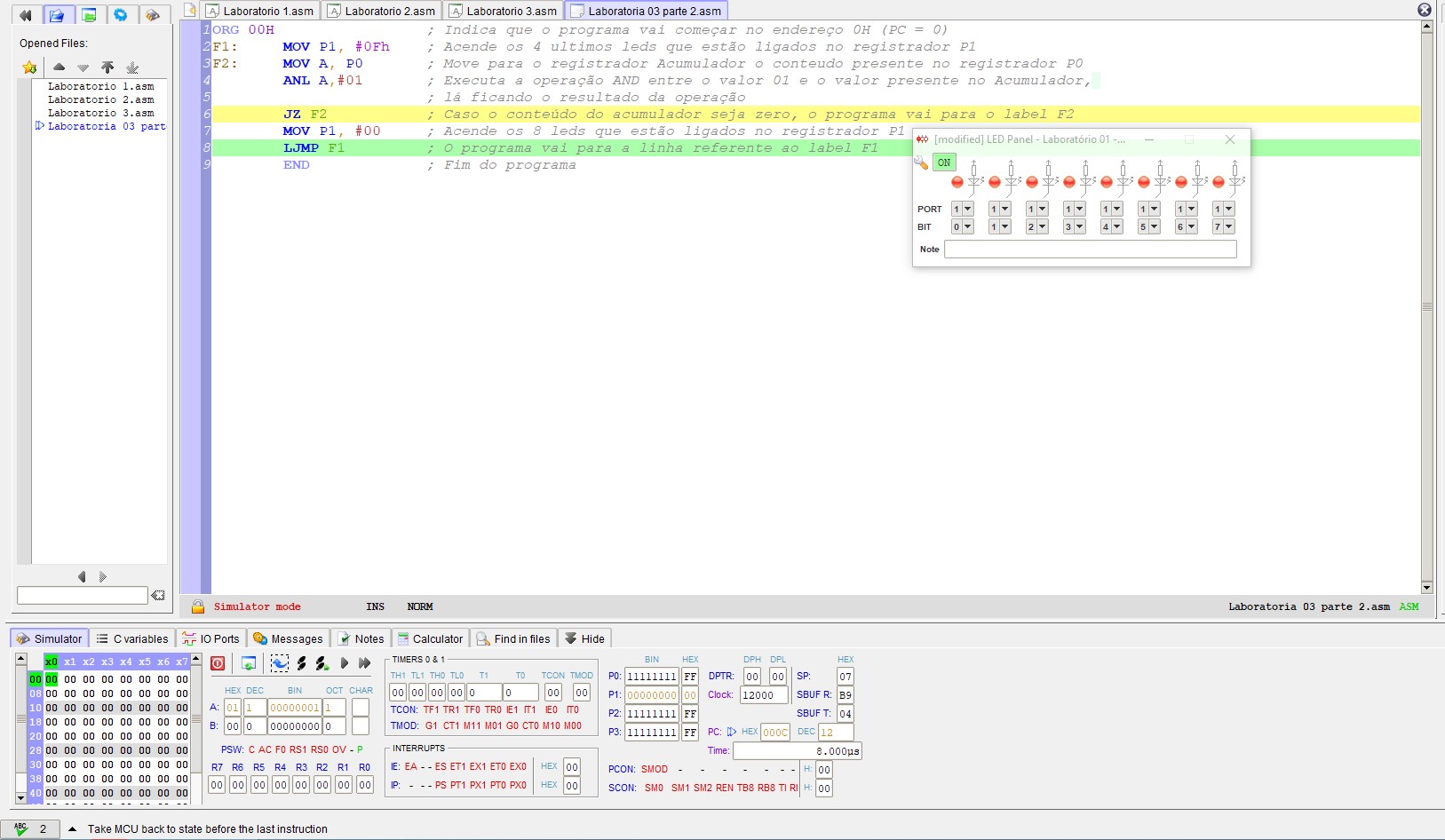
* + - 1. Na segunda linha é perceptível que o programa move para o registrador Acumulador o conteúdo presente no registrador **P0N**, deixando o mesmo com 11111111 b.



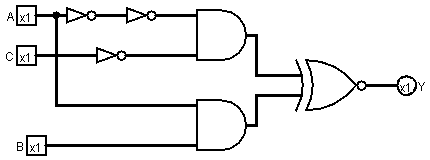
* + - 1. Na terceira e na quarta linha, logo após a operação **AND** é executada entre o conteúdo do registrador Acumulador e o número **01** foi possível ver que o resultado da operação foi armazenada no próprio registrador Acumulador.

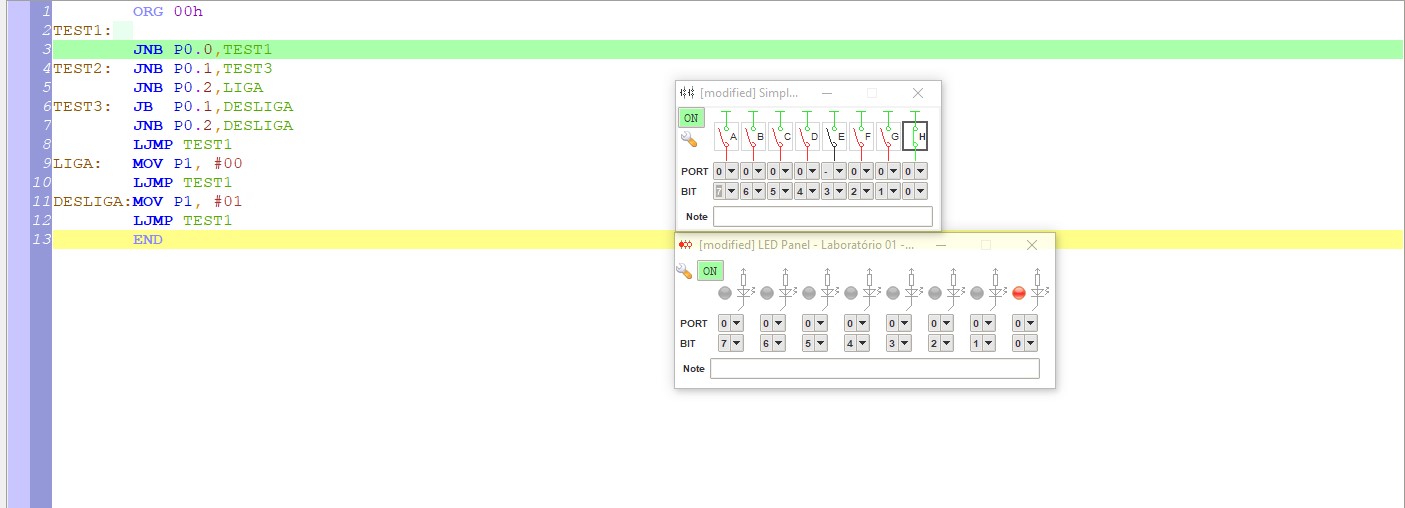


* + - 1. Na última linha, o programa vai para o label definido como **F1**, e após acender todos os led’s.



* + 1. Implementando o circuito Logico:





1. Resultados e Discussão: Ao terminar todo o experimento, é possível compreender de forma pratica o melhor funcionamento da linguagem Assembly e, usar periféricos disponíveis no simulador MCU 5081 IDE, e suas interações.

1. Conclusões: Foi possível concluir que a linguagem Assembly em atuação ao microcontrolador MCU 8051 pode dar uma vasta biblioteca de opções e atuações diante de seus registradores e portas, assim como demonstrado no decorrer do relatório com o exemplo do painel de LED 's e do teclado virtual.
2. Referencias:
   1. MORAIS, Misael. Organização e arquitetura de computadores. [ S. l.], . 2021. Disponível em: https://drive.google.com/drive/folders/0Bwjlecok7TpyfnAtQmx6bE4yZ043amNsbnRxMkF4UFlpWVZhWmRfeVBSeHRRVi1xRzNOZnM?resourcekey=0-WmQ6S1i6hLYyCoGBLbMeGw. Acesso em: 12 jun. 2022