Universidade Estadual de Campinas
Faculdade de Engenharia Elétrica
e de Computação

EA871 - Laboratório de Programação Básica de Sistemas Digitais

Roteiro 6 - Controle de push buttons por polling

Aluno: Fernando Teodoro de Cillo

RA: 197029

Campinas
Maio de 2022

Introdução:

O intuito deste experimento é entender o problema de *bounce* em chaves e formas de evitá-lo, como a implementação de *bedounce* por *hardware* do *shield* FEEC 871. Outros tópicos do experimento são o uso da técnica de *polling* para captura de eventos externos e implementação de uma máquina de estado que altera a cor do *led* RGB.

Experimento:

1 Podemos ver, utilizando um analisador lógico, que as botoeiras são ativas baixas e que há oscilações quando a variação do sinal é muito rápida.

Figura 1: Analisador lógico

3 A figura 2 mostra (à esquerda) que os arquivos .c e .h foram criados ou copiados adequadamente.

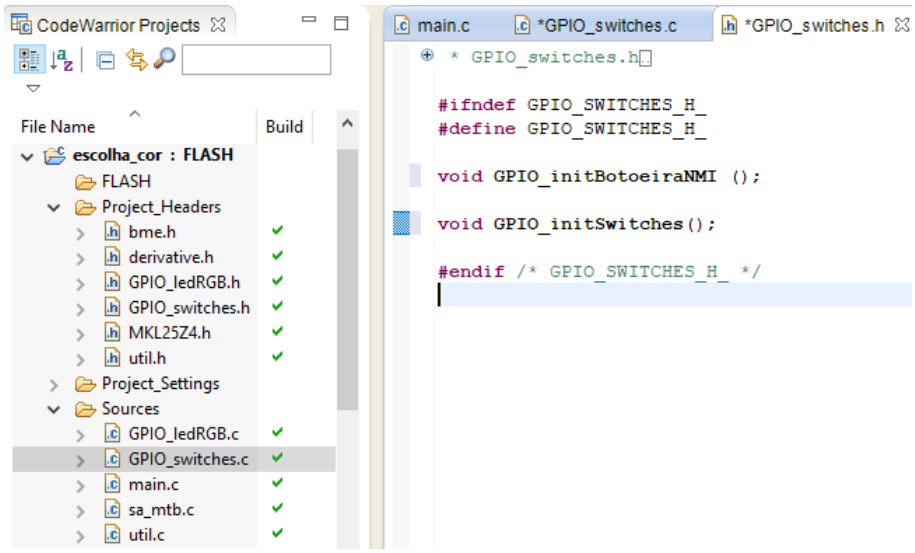

Figura 2: Inicialização

4 A figura 3 mostra um teste do algoritmo que traduz cada código de cor sugerida em níveis lógicos associados aos pinos PTB18, PTB19 e PTD1. Percebe-se, com o uso de um breakpoint, que o valor da variável 'estado' é alterado.

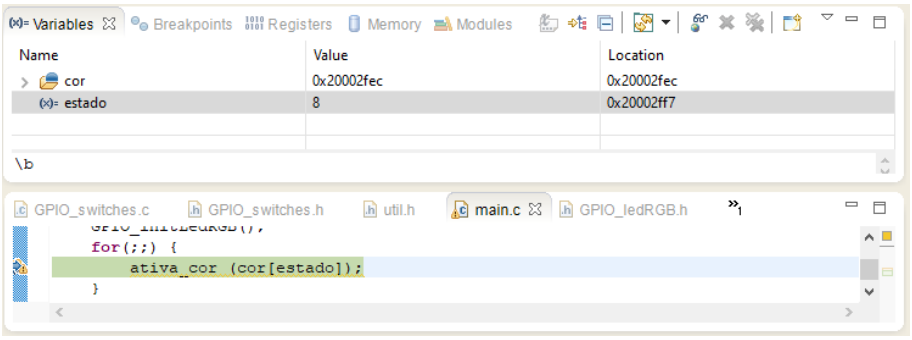

Figura 3: Teste do algoritmo

5 É necessário testar o acionamento (em nível lógico baixo) das botoeiras. Uma forma de fazer isso é com o algoritmo da figura 4, que associa a ativação do botão a um valor (função GPIO_initSwitches()) e esse valor a uma cor do LED.

```
int main (void)
{
    int valor;
    GPIO_initLedRGB();
    GPIO_initSwitches();
    for(;;) {
        GPIO_amostraSwitches (&valor);
        if (valor == 0x001) {
        //acender cor R do led indicando chave NMI
            GPIO_ledRGB(ON, OFF, OFF);
        } else if (valor == 0x002) {
        //acender cor G do led indicando chave IRQA5
            GPIO_ledRGB(OFF, ON, OFF);
        } else if (valor == 0x100) {
        //acender cor B do led indicando chave IRQA12
            GPIO_ledRGB(OFF, OFF, ON);
        }
    }
return 0;
}
```

Figura 4: Trecho do código

6 Pseudo-código:

escolha (estado)

caso 0 [0,0]:

acende cor azul (0,0,1)

chave NMI: transita para o estado [0,1]

chave IRQA5: transita para o estado [1,0]
chave IRQA12: transita para o estado [2,0]
caso 1 [0,1]:
acende cor ciano (0,1 ,1)
chave NMI: transita para o estado [0,2]
chave IRQA5: transita para o estado [1,1]
chave IRQA12: transita para o estado [2,1]

caso 2 [0,2]:
acende cor magenta (1,0,1)
chave NMI: transita para o estado [0,0]
chave IRQA5: transita para o estado [1,2]
chave IRQA12: transita para o estado [2,2]
caso 3 [1,0]:
acende cor verde (0,1,0)
chave NMI: transita para o estado [0,1]
chave IRQA5: transita para o estado [1,1]
chave IRQA12: transita para o estado [2,0]
caso 4 [1,1]:
acende cor ciano (0,1 ,1)
chave NMI: transita para o estado [0,1]
chave IRQA5: transita para o estado [1,2]
chave IRQA12: transita para o estado [2,1]
caso 5 [1,2]:
acende cor amarelo (1,1,0)
chave NMI: transita para o estado [0,2]
chave IRQA5: transita para o estado [1,0]
chave IRQA12: transita para o estado [2,2]
caso 6 [2,0]:
acende cor vermelho (1,0,0)
chave NMI: transita para o estado [0,0]
chave IRQA5: transita para o estado [1,0]
chave IRQA12: transita para o estado [2,1]
caso 7 [2,1]:
acende cor magenta (1,0,1)
chave NMI: transita para o estado [0,1]
chave IRQA5: transita para o estado [1,1]
chave IRQA12: transita para o estado [2,2]
caso 8 [2,2]:
acende cor amarelo (1,1,0)

chave NMI: transita para o estado [0,2]
chave IRQA5: transita para o estado [1,2]
chave IRQA12: transita para o estado [2,0]
fimescolha

8 O fluxo de execução seguiu como esperado, sem nenhum desvio imprevisto. Isso significa que todos os arquivos e funções implementados estão funcionando corretamente e devidamente utilizados, gerando o executável desejado.

10 Duas sequências testadas foram as seguintes:
IRQA5 - IRQA5 - IRQA12 - NMI - IRQA12 - NMI - IRQA5 - IRQA5:
azul - verde - ciano - magenta - ciano - magenta - ciano - ciano - amarelo
NMI - NMI - NMI - NMI - IRQA5 - IRQA5 - IRQA12 - NMI:
azul - ciano - magenta - azul - ciano - amarelo - verde - vermelho - azul

11

```
'Executing target #9 escolha_cor.siz'
'Invoking: ARM Ltd Windows GNU Print Size'
"C:/Freescale/CW_MCU_v10.6/Cross_Tools/arm-none-eabi-gcc-4_7
   text    data     bss     dec     hex filename
   2124      24    2076    4224    1080 escolha_cor.elf
'Finished building: escolha_cor.siz'
' '
```

Figura 5: Função Print Size

Podemos ver, pela figura 5, que a memória alocada para o arquivo executável é principalmente para texto (.txt) e variáveis não inicializadas (.bss).