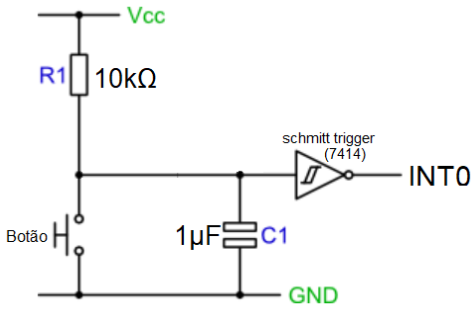
EA869 – Introdução a Sistemas de Computação Digital

Relatório do Projeto Final

Prof. Levy Boccato – 1º semestre de 2019

Bryan Wolff	**RA: 214095**
João Pedro Bizzi Velho	**RA: 218711**
Julia Alves Farias	**RA: 200130**

Para efetuar o projeto final, utilizaremos um circuito *debounce* com um botão conectado no pino D2 do arduino, associado a interrupção externa INT0. Geralmente quando você usa um botão conectado em um circuito digital é necessário de um circuito *debounce* para que uma única pressão no botão não seja detectado várias. E de acordo com o orientador, utilizamos na protoboard o circuito a seguir:



No início do programa foi definido o vetor de interrupção, em 0x0000 está localizado a RSI "reset" e em 0x0002 está a RSI associada ao INT0, que é por onde está conectado o botão externo. A partir de 0x0034 termina o vetor de interrupção, e começa o programa.

O programa foi dividido em 5 partes, onde na primeira, foi definido as rotinas de serviço de interrupção que são responsáveis por tratar o evento externo do botão reset e do botão conectado em INT0. Na rotina "reset", foi configurado a interrupção externa INT0, de modo a ser ativada por borda de subida no pino 2 da porta D, e além disso, os Registradores DDR da porta B e D precisaram ser configurados para definir como os pinos onde estão conectados o botão e o led, de modo a se comportarem como entrada ou saída. Assim, foi setado o bit 5 de DDRB (0x04) para fazer o pino 5 da porta B se comportar como saída (com a finalidade de ligar o led) e zeramos o bit 2 de DDRD para fazer o pino 2 da porta D se comportar como entrada (para receber o sinal de borda de subida do botão). Além disso, zeramos e definimos um registrador com o nome "state" e o valor contido nele corresponderá ao estado em que o led deve estar.

Ainda na primeira parte do programa, configuramos uma rotina "pressed_button" responsável por tratar o evento do botão externo, que quando acionado, incrementará o registrador "state" para o próximo estado, como o programa deve ter apenas 3 estados, o "state" ao ser incrementado poderá passar do valor 3, e portanto, na mesma rotina foi

configurado que se o “state” atingir o valor 4, ele voltará para o valor 1 (antes de finalizar a interrupção), correspondente ao primeiro estado, de modo que o programa fique em um ciclo infinito entre os 3 estados que será abordado na quarta parte do programa.

Na segunda parte, definimos o programa principal que verificará o valor do registrador “state” que se for 1, ele chamará o primeiro estado nomeado como “estado 0”, se for 2, chamará o segundo estado com nome “estado 1”, e analogamente se for 3, desviará para o terceiro estado nomeado como “estado 2”. Os estados foram definidos posteriormente, e ao final da comparação do valor do registrador referente ao estado atual, o programa principal desviará para ele mesmo, para assim efetuar novas comparações de modo a detectar se deve haver uma mudança de estados. Como pode perceber, o programa só irá para o estado 0 quando o valor de “state” for 1, mas anteriormente o registrador “state” foi zerado, portanto, o led começará apagado e ele não entrará no estado 0, até que o botão seja pressionado uma primeira vez, e quando pressionado, o programa ficará num ciclo infinito entre os estados e o programa principal que verifica se houve mudança de estado.

Na terceira parte definimos uma rotina de atraso, que por sua vez consumirá ciclos do processador de modo a gastar tempo, e este tempo dependerá do parâmetro (r20) que deve ser definido antes de começar a rotina de atraso. Para uma variação unitária neste parâmetro, teremos um tempo gasto de 50 ms, portanto se o parâmetro for 5 teremos um atraso de 250 ms, se for 10, teremos um atraso de 500 ms e se for 20 teremos um atraso de 1 segundo.

Na quarta parte do programa, definimos os estados. Os estado farão, a princípio, a mesma coisa, isto é, acender e apagar o led. Porém, o que muda de um estado para outro é o tempo em que o led permanecerá aceso e apagado, ou seja, o parâmetro que deve ser definido para a rotina de atraso assumirá valor diferente em cada estado.

No primeiro estado (estado 0), será definido o parâmetro como 20, depois será chamado uma rotina de acender o led e depois chamará a rotina de atraso de 1 segundo. Logo em seguida, definirá o parâmetro como 20 novamente e chamará uma rotina que apagará o led e vai ser chamado a rotina de atraso de 1 segundo. Assim o led acenderá e permanecerá aceso por 1 segundo, depois vai apagar e permanecer apagado por 1 segundo. O segundo estado (estado 1) será análogo, porém o parâmetro definido será 10 para que o led pisque pela metade do tempo, ou seja, o led acenderá e permanecerá aceso por 500 ms, depois vai apagar e permanecer apagado por 500ms. O terceiro estado (estado 2) será análogo também, porém o parâmetro definido será 5 para que o led pisque pela metade do tempo do estado anterior, ou seja, o led acenderá e permanecerá aceso por 250ms, depois vai apagar e permanecer apagado por 250ms. O final de cada estado, desviará para o programa principal.

Na quinta e última parte do programa estão as rotinas responsáveis por acender e apagar o led. Essas rotinas foram citadas anteriormente e são chamadas de maneira frequente em cada estado. Na rotina “led_on”, foi setado o bit 5 de PORTB (0x05), que jogará nível alto de tensão no pino onde está conectado o led, que por sua vez, acenderá o led. Na rotina “led_off”, foi zerado o mesmo bit 5 de PORTB, que por sua vez apagará o led.

Nesta perspectiva, o programa ficará em um loop infinito entre um estado, que acende e apaga o led, e o programa principal, que verificará se deve ir para outro estado. Se não houve mudança de estado (pressionamento do botão), o mesmo estado será chamado

novamente pelo programa principal para que o led pisque de maneira contínua. E sempre quando o programa estiver no estado 2 e for pressionado o botão, o programa irá retornar para o estado 0, ficando um looping infinito neste ciclo de estados. Caso o botão reset seja pressionado, a rotina “reset” será chamada, que por sua vez vai zerar o registrador “state”, e o led apagará e o programa ficará em um ciclo infinito no programa principal, até que seja apertado o botão de INT0 pela primeira vez para entrar no estado 0, que por sua vez não sairá mais do ciclo dos estados definido anteriormente.