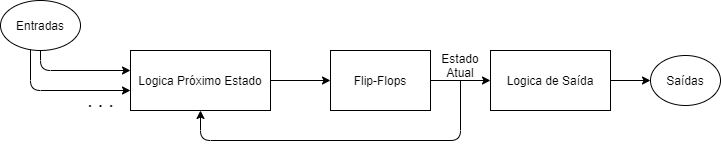
3.3 - Sistema eletrónico para o controlo dos motores (pcbs e ligações)

De forma a ser possível controlar o estado de funcionamento do robô, ou seja, fazê-lo seguir a linha, parar ou rodar 180 °, desenvolveu-se uma máquina de estados sequencial síncrona, mais especificamente uma máquina de Moore, uma vez que as suas saídas dependem apenas do estado atual.

Figura 1 - Máquina de Moore



Numa máquina de estados, o objetivo é determinar a “Lógica do Próximo Estado” e a “Lógica de Saída” do circuito. Como referido, a “Lógica de saída” depende apenas do estado atual. Já a “Lógica do Próximo Estado” depende do estado atual e das entradas do circuito.

Como referido, o objetivo do robô é seguir uma linha, devendo parar sobre uma linha horizontal colocada no final do percurso (Figura 2) e, nesse momento, o robô deve dar uma volta de 180 ° sobre o seu eixo, voltando a seguir a linha em direção ao ponto de partida, após o acionamento de um botão.



Figura 2 - Linha preta usada para os testes

Denominou-se a linha de final de curso como a linha horizontal e a linha determinante do percurso como linha vertical.

De modo a detetar a linha no final do percurso, usaram-se os dois sensores das pontas do array de sensores (S1 e S8, rodeados a amarelo na Figura 4). Usou-se também o sensor do meio (S5, rodeado a verde na Figura 4) de forma a interromper a rotação quando este encontrar a linha preta.

Nesse momento, o robô poderá ser acionado através de um botão de pressão (B) (Figura 3) para que este volte a seguir a linha.



Figura 3 - Botão de Pressão

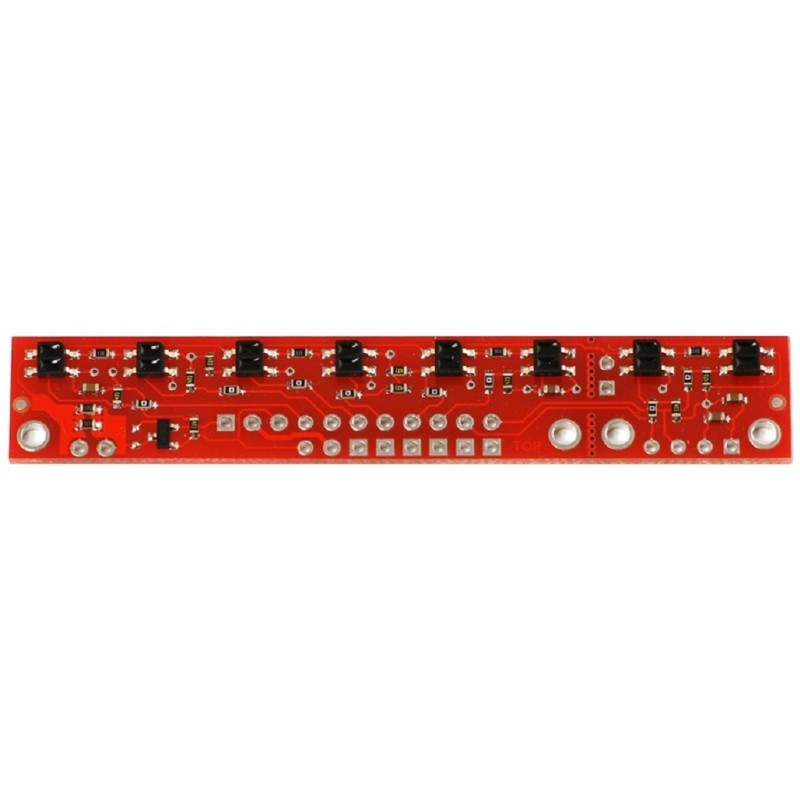


Figura 4 – Sensores usados na máquina de estados: S1, S5 e S8

Os motores são controlados por meio de um *driver* (L298N), podendo estar parados, a rodar no sentido horário ou no sentido anti-horário. O driver possui quatro entradas, duas por cada motor.

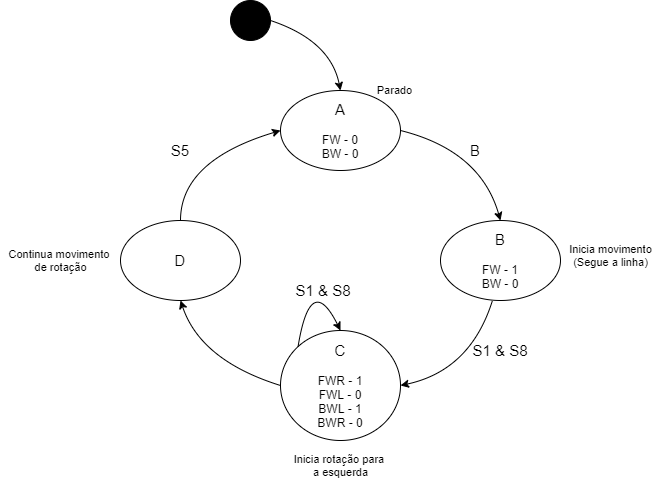
Para isso, na máquina de estados existem duas saídas por cada motor, sendo que uma controla o movimento no sentido horário e outra no sentido anti-horário.

Como o robô usa dois motores, existem quatro saídas, que quando estão ativas (nível lógico alto) despoletam as seguintes respostas:

* FWR – motor da direita roda no sentido horário;
* FWL – motor da esquerda roda no sentido horário;
* BWR – motor da direita roda no sentido anti-horário;
* BWL – motor da esquerda roda no sentido anti-horário.

Na figura seguinte, mostra-se o diagrama da máquina de estados, usada para controlar o modo de funcionamento do robô. FW representa os valores de FWR e FWL e BW representa os valores de BWR e BWL.

Figura 5 - Diagrama da Máquina de Estados



Inicialmente, no estado “A”, o robô está parado (FW = 0 e BW = 0), e espera pelo acionamento do botão para iniciar a marcha. Após o acionamento do botão, B a nível lógico alto, no estado “B”, o robô inicia a sua trajetória, seguindo a linha com os dois motores a rodar no sentido horário (FW = 1 e BW = 0). Quando o *array* de sensores estiver posicionado sobre a linha horizontal preta (fim do trajeto), os sensores S1 e S8 estarão a nível lógico alto, e ocorrerá a transição para o estado “C”. Neste estado, o robô inicia a rotação para a esquerda (FWR = 1, FWL = 0, BWR = 0 e BWL = 1) e não transita de estado enquanto os sensores S1 e S8 estiverem a nível lógico alto, isto é, enquanto o *array* de sensores estiver sobre a linha horizontal. Quando o *array* de sensores sair desta, S1 e S8 estarão a nível lógico baixo, avançando para o estado “D”, no qual o robô continua a rotação, mantendo as suas saídas, até que o sensor S5 esteja sobre a linha vertical, S5 a nível lógico alto. Assim posicionado, o robô volta ao estado inicial.



Tabela 1 - Tabela Completa para Determinação da Lógica Completa da Máquina de Estados

De forma a implementar a máquina de estados, de acordo com o diagrama de estados, determinou-se toda a lógica necessária, construindo a tabela 1.

Adicionaram-se duas colunas que indicam os bits do estado atual (B1 e B0) e também as entradas da máquina de estados (S1, S5, S8 e B). Como existem quatro entradas, existem dezasseis (24) combinações possíveis para cada estado, o que resulta em sessenta e quatro possibilidades no total (24+2 = 26 = 64) que será o número de linhas da tabela 1.

A seguir, “E” representa o estado atual e “E\*” o estado seguinte. “B1\*” e “B0\*” representam os bits do estado seguinte. Estas colunas devem ser preenchidas de acordo com a tabela ao lado (Tabela 2).

Tabela 2 - Atribuição de Estados



Sabendo que cada flip-flop permite a existência de dois estados e que existem quatro estados (A – D), selecionaram-se dois *flip-flops* do tipo J-K, que apresentam a tabela de excitação ao lado (Tabela 3). De acordo com esta tabela, foram preenchidas as colunas “J” e “K” para cada *flip-flop* (Ja e Ka, relativo a B0; Jb e Kb, relativo a B1). Para este efeito, determinou-se a lógica do próximo estado, em função das quatro entradas da máquina de estados, simplificada através de mapas de *Karnaugh*. Resolveram-se quatro mapas de *Karnaugh* (um para cada entrada J e K dos *flip-flops*) e obteve-se os resultados apresentados na figura 6.



Tabela 3 - Tabela de Excitação do Flip-Flop JK



Figura 6 - Lógica do Próximo Estado

Adicionaram-se as colunas das saídas (FWR, FWL, BWR e BWL), as quais foram preenchidas de acordo com o estado atual. Para cada uma destas determinou-se a lógica de saída, que também foi simplificada através de mapas de *Karnaugh*. Os resultados obtidos foram os da figura 7.



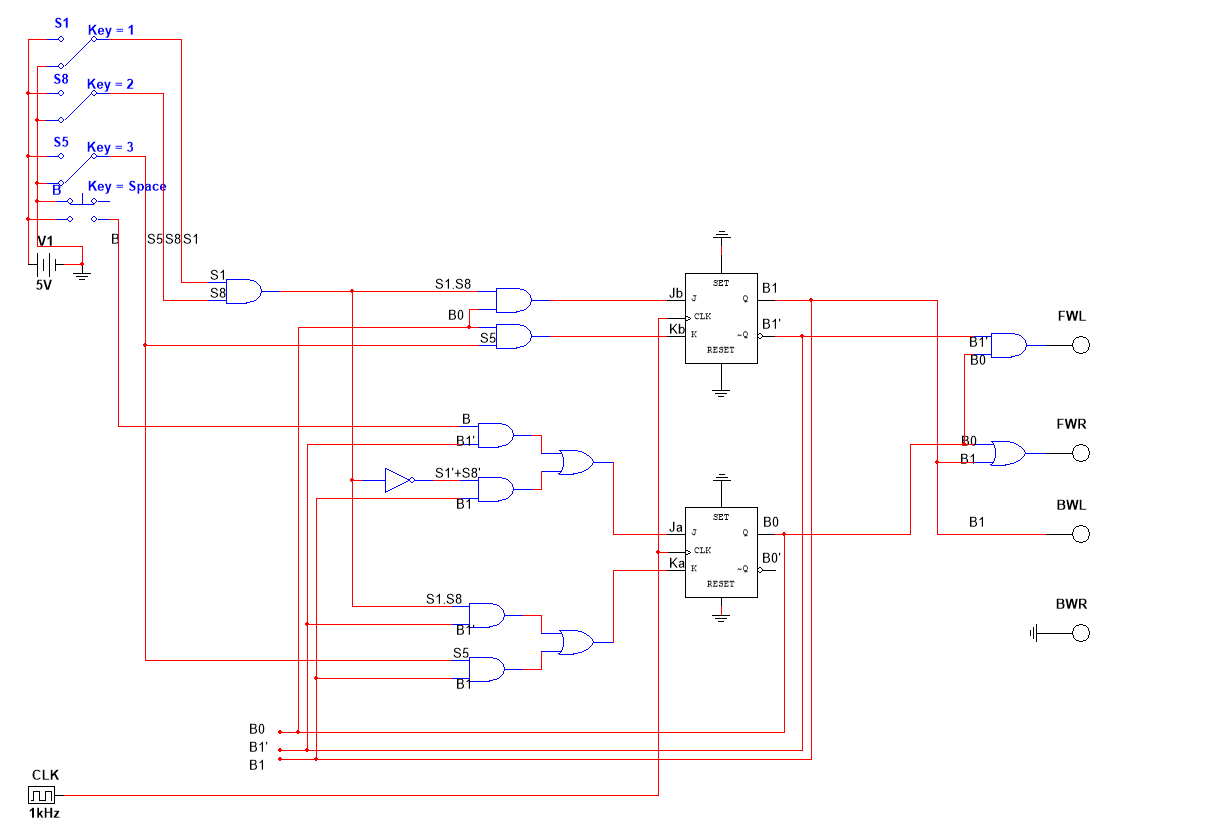
Figura 7 - Lógica de Saída

De acordo com a lógica determinada, desenhou-se o circuito que implementa a máquina de estados, o qual foi testado usando a ferramenta *Multi-Sim*, verificando-se a sua funcionalidade.

Lógica de próximo estado

Entradas

Figura 8 - Desenho da Máquina de Estados



Lógica de saída

Saídas

Flip-Flops

Estado atual

Lógica de Entrada

Tal como mostra a figura 9, cada Flip-Flop J-K necessita de um sinal *clock* como entrada. O *clock* é um sinal em forma de onda quadrada com um *duty-cycle* de, idealmente, 50 %. Dá-se uma transição entre estados sempre que, a cada pulso de *clock*, uma das entradas (J ou K) estiver a nível lógico alto.

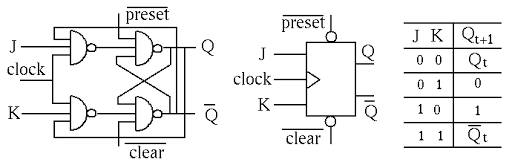
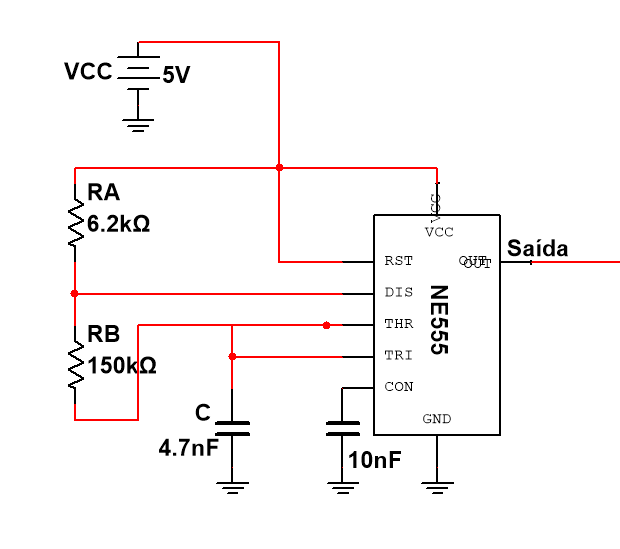


Figura 9 - Esquema do Flip-Flop J-K

Para este efeito, implementou-se um oscilador usando o integrado NE555 numa montagem como multivibrador astável.

Figura 10 - Implementação do Oscilador



O condensador C1 carrega até 2\*VCC/3 através das resistências RA e RB (saída = 1). Quando C1 atinge este valor, irá descarregar por RB (saída=0), até uma tensão de VCC/3 e o ciclo repete-se.

Através dos seguintes cálculos, dimensionaram-se os componentes para obter uma onda quadrada de frequência 1 kHz, *duty-cycle* de 50 %, aproximadamente.

Para e :

Foram selecionados os valores standard das resistências (Série E24) e do condensador (Série E12) mais próximos dos obtidos:

A figura 11 mostra a saída do circuito obtida em simulação com a ferramenta TINA.

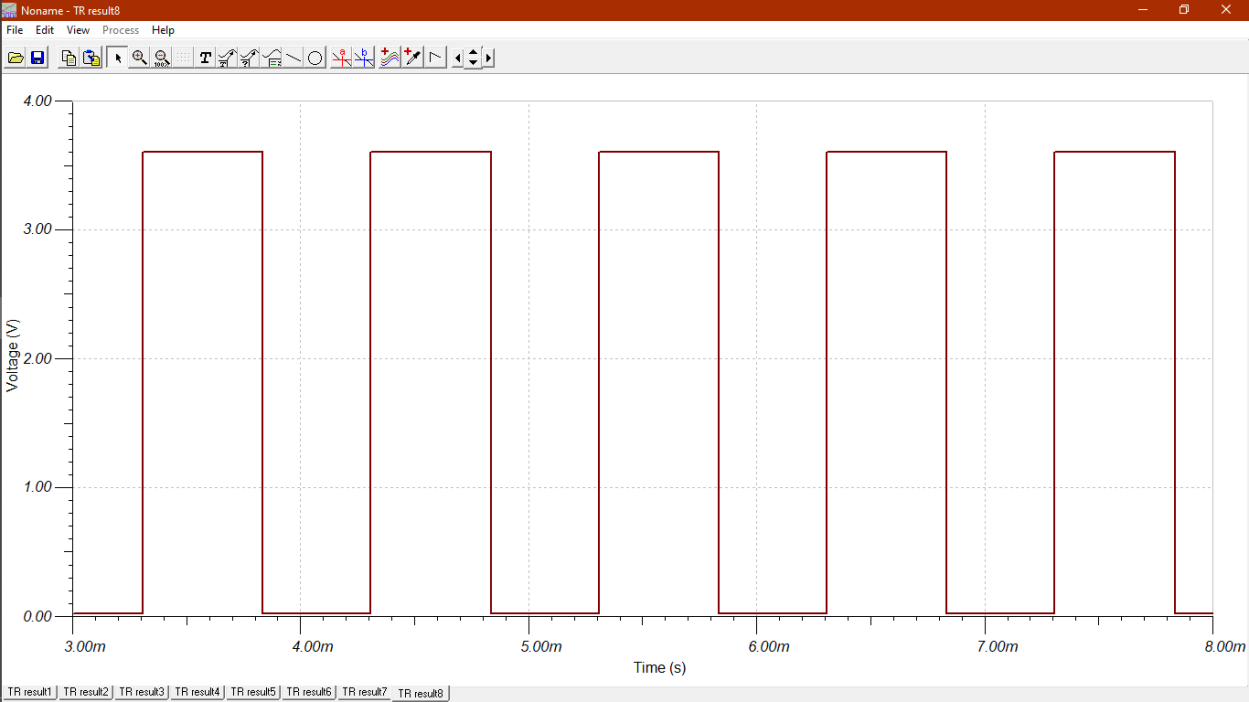
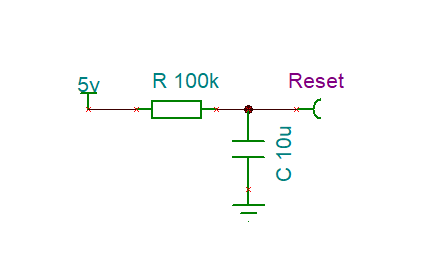
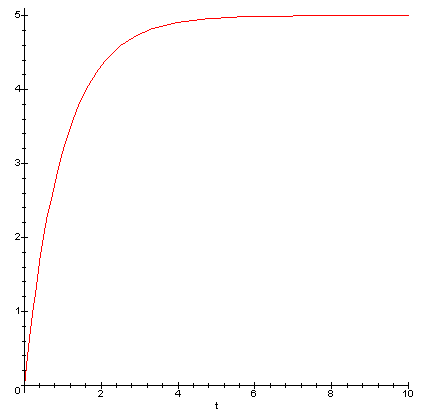


Figura 11 - Resposta do Oscilador de 1 kHz

As entradas *Clear* dos Flip-Flops J-K utilizados são ativas a nível lógico baixo, isto é, se ligarmos esta entrada ao *ground*, ocorrerá um *reset* no flip-flop e sua a saída *Q* virá a nível lógico baixo, independentemente das suas entradas *J* e *K*. Assim, se na máquina de estados as entradas *Clear* forem ligadas *ao* GND*,* B0 e B1 ficarão a nível lógico baixo*.*

Experimentalmente, verificou-se que, quando alimentados, os flip-flops mantêm as saídas do estado anterior. Para que o robô inicie no primeiro estado de funcionamento (estado “A”), utilizou-se um circuito para efetuar um *reset* aos flip-flops, tal como mostra a figura 10.

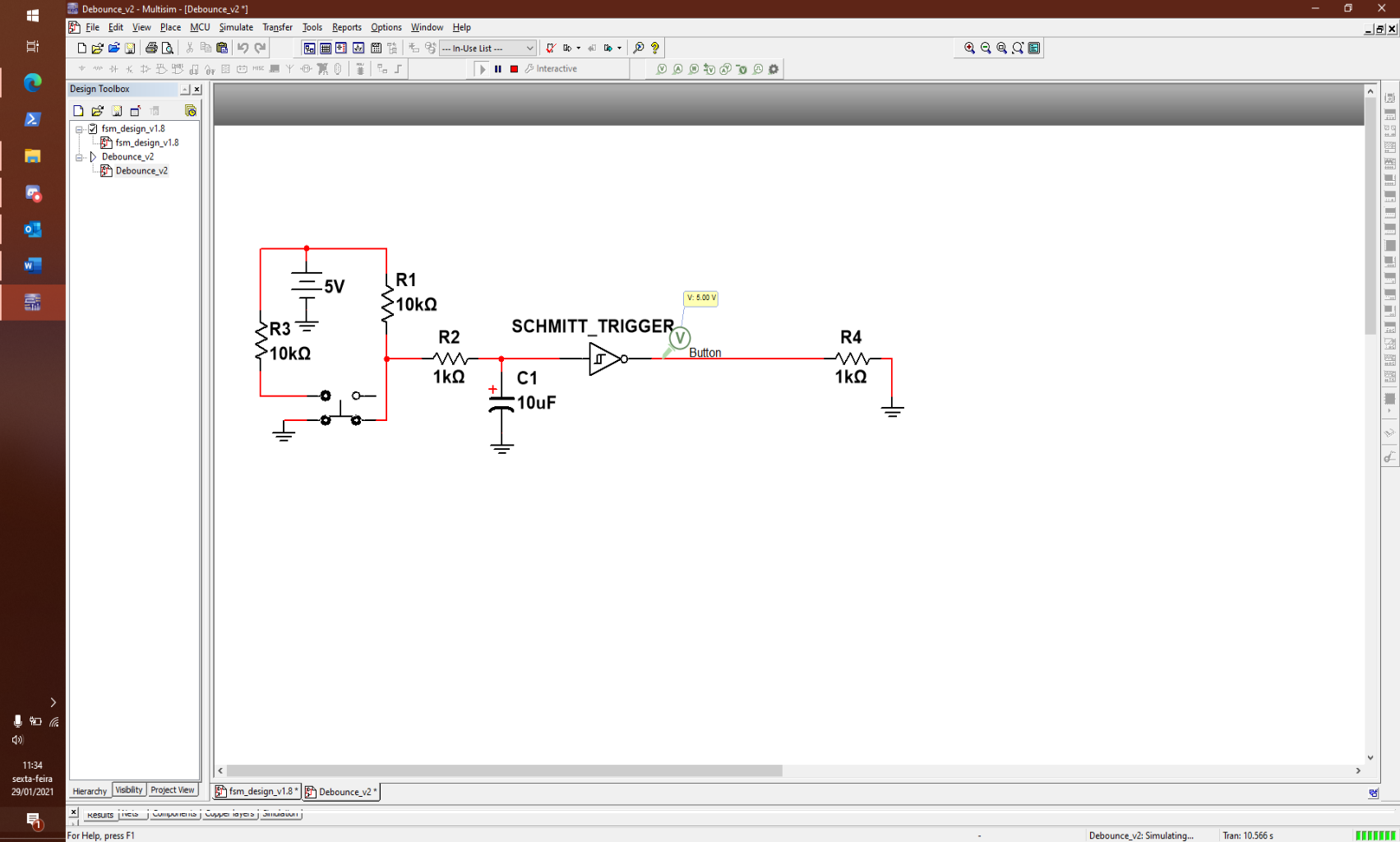
Figura 12 - Circuito RC de Reset e a sua resposta no tempo.



Sabendo que a saída deste circuito estará a nível lógico baixo enquanto o condensador não carregar, a entrada *reset* dos flip-flops estará também a nível lógico baixo, o que resulta na ação de *reset*. Assim, logo que se alimentar a máquina de estados, os flip-flops apresentarão nas saídas “Q” o nível lógico baixo que representa o primeiro estado de funcionamento do robô. Após o condensador carregar, a entrada *reset* estará a nível lógico alto durante o resto do tempo de funcionamento, não provocando um *reset* indesejado.

Foi usado um botão de pressão para que o robô inicie a marcha. No entanto, sabe-se que, quando uma saída provém diretamente de um botão de pressão, esta é afetada por problemas de vibração mecânica (*bounce*) quando o contacto se fecha. De forma a garantir que a mudança de nível lógico se faça sem oscilações, torna-se necessário implementar um circuito de *debounce* como, por exemplo, o apresentado na figura 13.

Figura 13 - Circuito de Debounce do Botão de Pressão (quando premido)



O condensador em conjunto com a resistência funciona como filtro passa-baixo, que ajuda a reduzir o ruído elétrico provocado pelas vibrações mecânicas do botão. Assim, quando o botão não está pressionado, os terminais do condensador apresentarão 5 V e, quando pressionado, apresentarão 0 V. Uma vez que, num circuito sequencial, o nível lógico alto corresponde ao acionamento de uma entrada (botão, neste caso), será necessário utilizar uma lógica inversora na saída do condensado. Para isso, usou-se uma porta lógica *Schmitt-Trigger* (integrado 74HCT14) que funciona como um circuito comparador inversor com histerese centrado em 0 V. Assim, aumenta a imunidade ao ruído e, como inverte, a saída do circuito estará a nível lógico alto quando o botão for pressionado.

Foi feito o teste em simulação do circuito da máquina de estados através da ferramenta *Multi-Sim.* Os resultados obtidos apresentam-se nas figuras abaixo. Verificou-se, portanto, a boa funcionalidade do circuito dimensionado.

Figura 14 - Estado 0 (Parado)

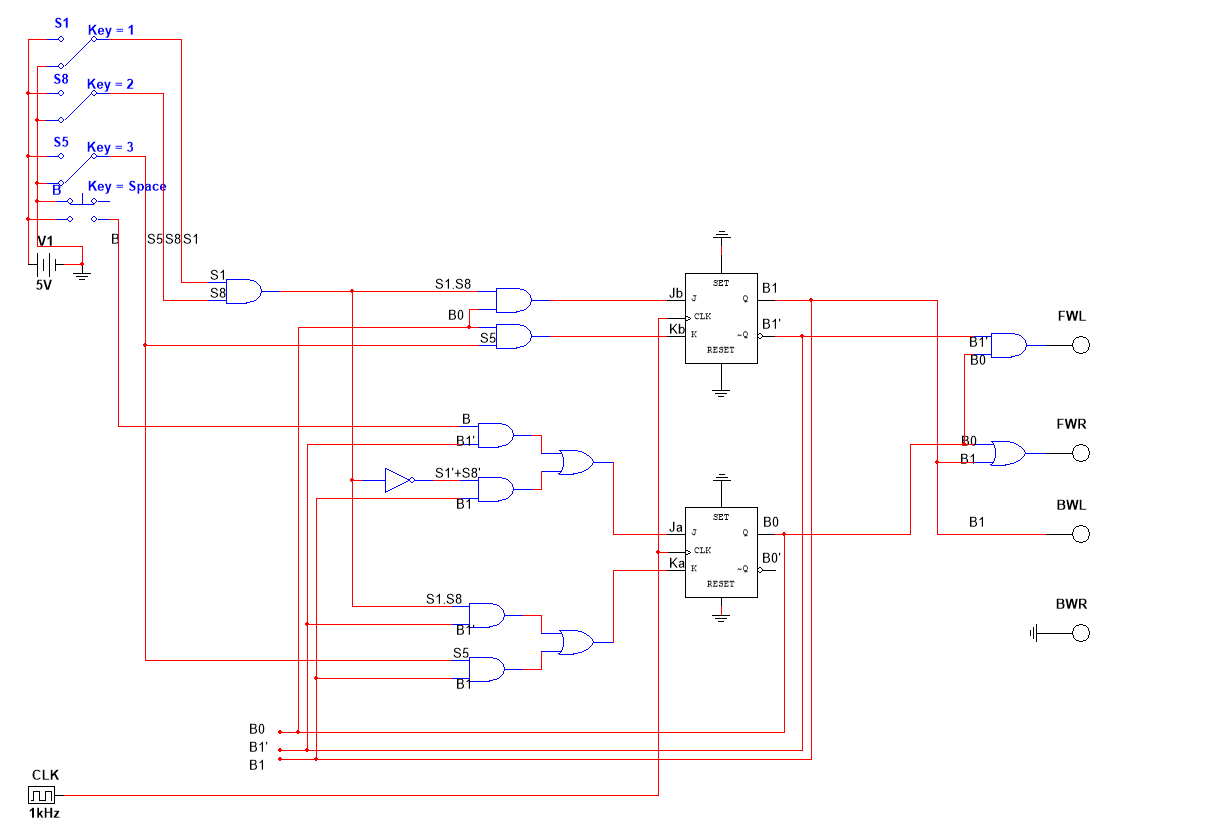


Figura 16 - Estado 1 (Anda para a frente) – Botão foi pressionado

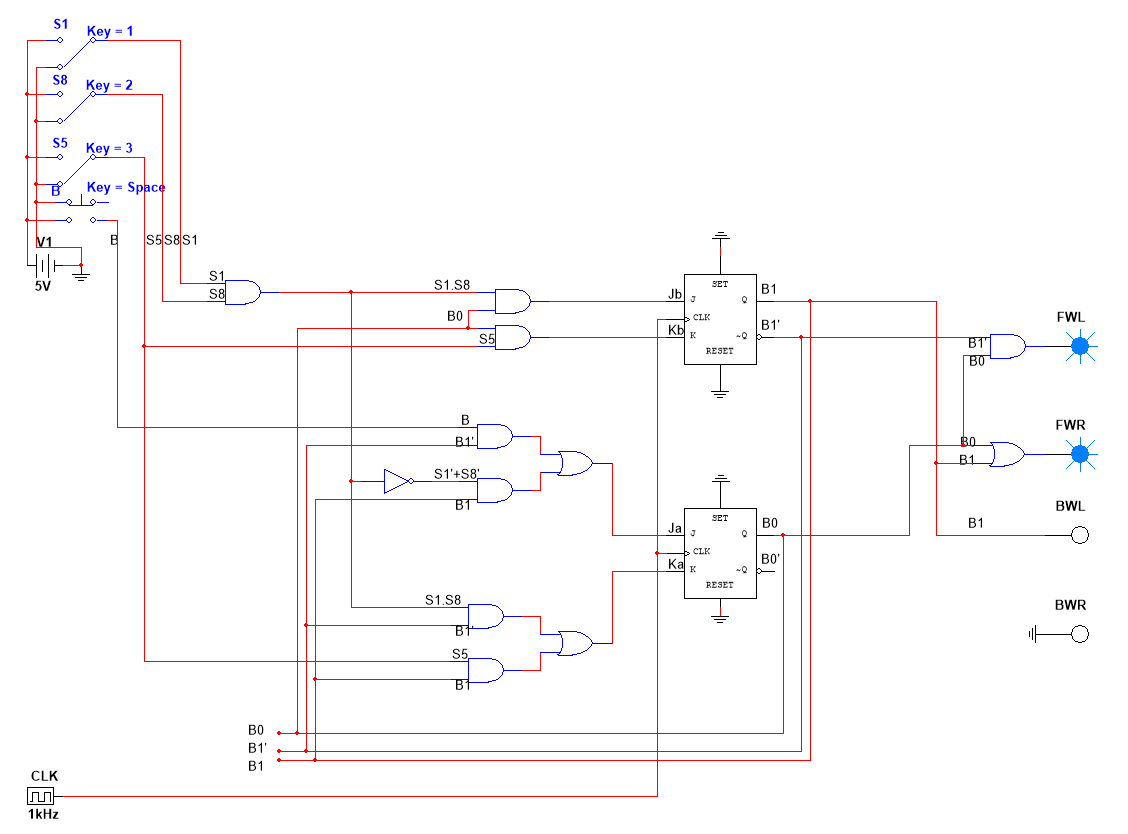


Figura 15 - Estado 2 (Dá a volta para a esquerda) – S1 e S8 a nível lógico alto

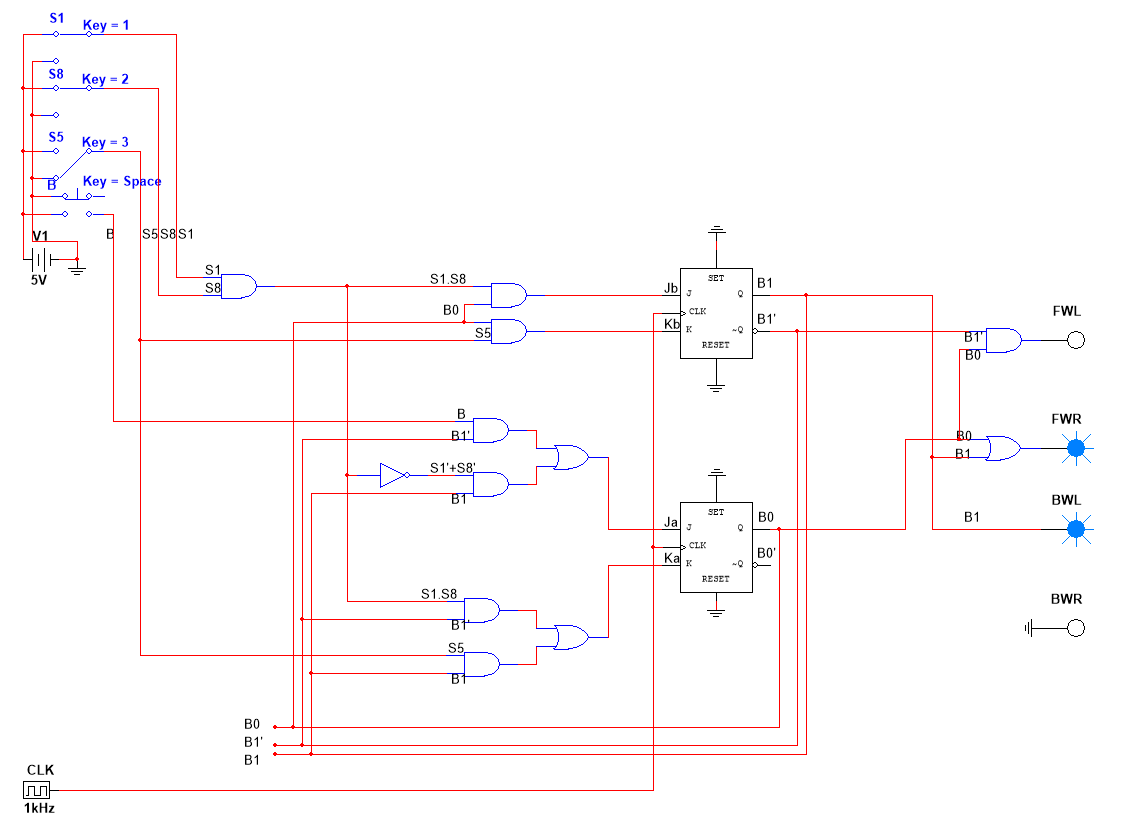


Figura 17 - Estado 3 (Dá a volta) - S1 e S8 a nível lógico baixo

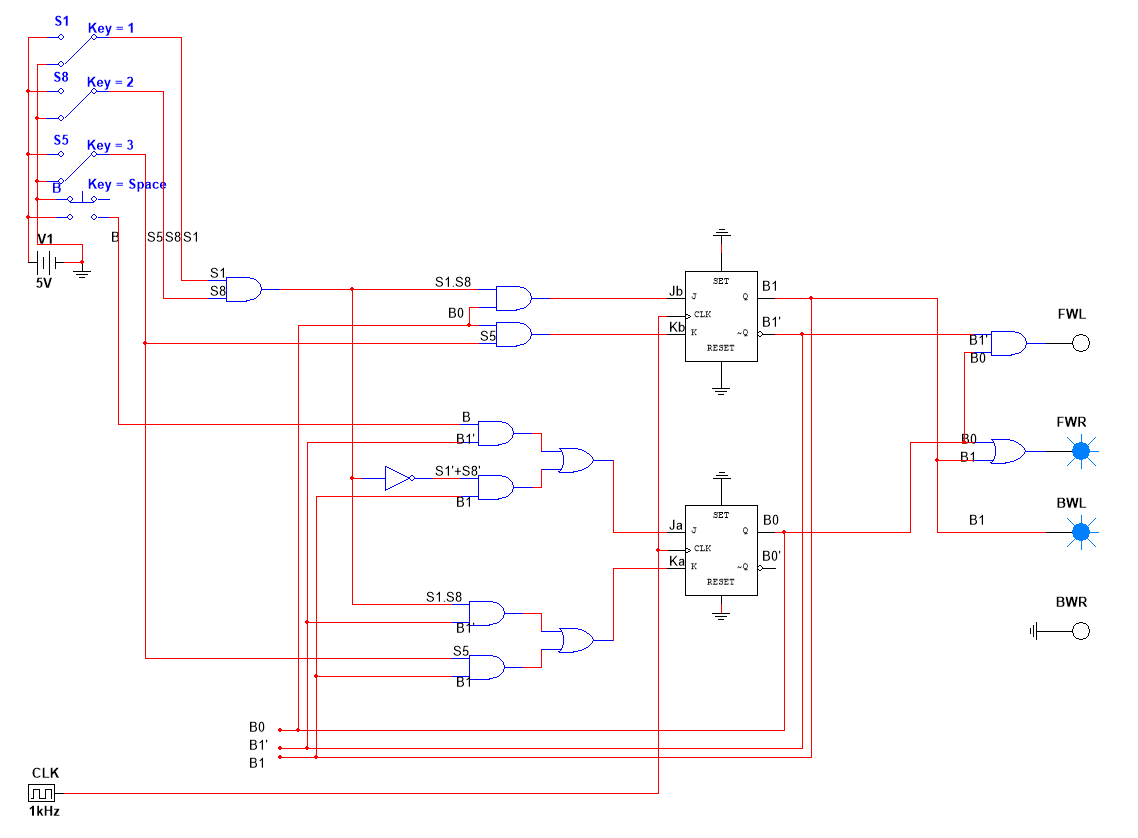


Figura 18 - Estado 0 (Parado novamente) - S5 = 1

