U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Arquitetura de Computação Embarcada

1º TRABALHO LABORATORIAL

AMPULHETA ELETRÓNICA

Mestrado em Engenharia Eletrotécnica e de Computadores

Duarte Ribeiro Afonso Branco up201905327@edu.fe.up.pt

Maria Inês Agostinho Simões up201904665@edu.fe.up.pt

Outubro 2022

INTRODUÇÃO

O presente relatório tem como objetivo a criação de uma ampulheta eletrônica que irá atuar como temporizador.

A partir de apenas dois botões foi possível criar vários tipos de ação que alteram o comportamento dos *LEDs*, tendo em conta o número de vezes e o tempo que se pressiona.

O funcionamento base desta ampulheta implica que, ao pressionar qualquer um dos botões, os *LEDs* de ordem 1 a 6 liguem-se, e, depois de um intervalo definido de tempo, o LED de menor ordem atualmente ligado, desliga-se. No final desta sequência, ou seja, quando os *LEDs* de ordem 1 a 6 estão apagados, o *LED 7*, que até este momento se encontrava desligado, liga-se.

CIRCUITO

O esquemático do circuito utilizado durante este trabalho está representado na figura seguinte:

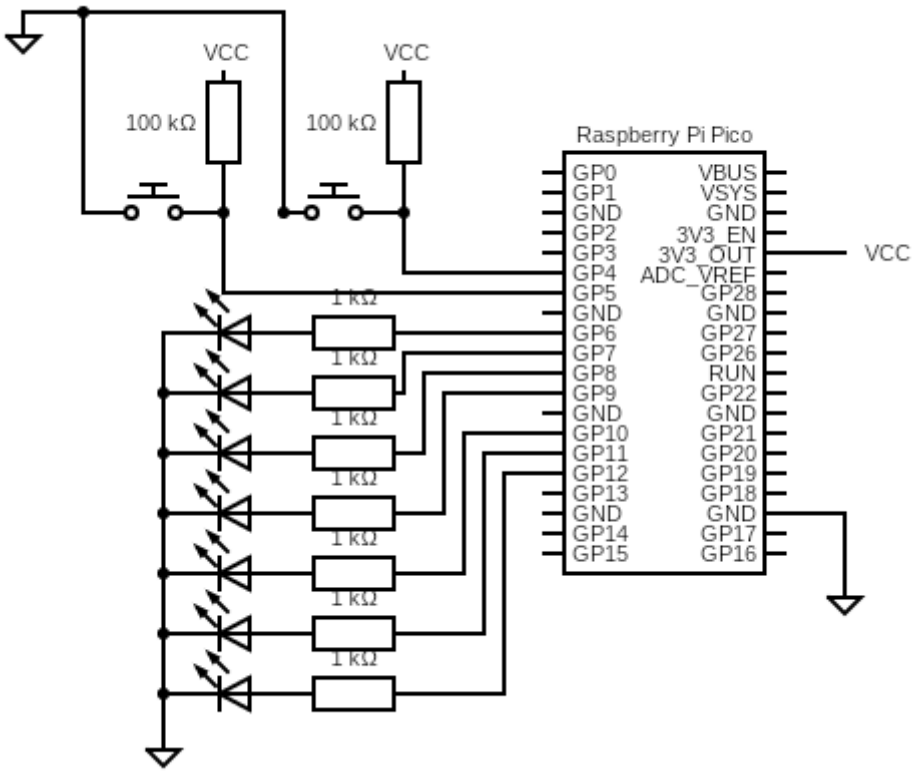

Figura 1 – Esquemático do circuito

A seguinte figura apresenta o circuito montado para o efeito deste trabalho.

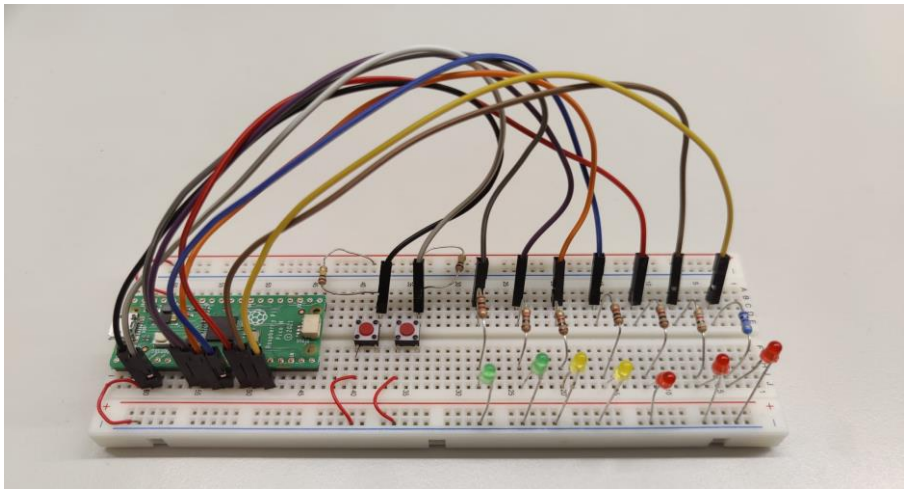

Figura 2 – Breadboard com o circuito montado

MÁQUINAS DE ESTADOS

fsm_BUTTONS

A primeira máquina de estados que foi implementada envolve as funções *reset* e *pause*, assim como a deteção de um *double click* ou de um *long press*, tendo como nome fsm_BUTTONS. Para executar estas funcionalidades foram criadas as seguintes variáveis:

timer – indica o valor atual do contador de tempo usado nas transições das outras máquinas de estado;

cur_time – guarda o número de milissegundos que passaram desde que o programa foi inicializado;

prev_time – guarda o valor da variável *cur_time* no final de um ciclo do *loop*;

last_timer – guarda o valor da variável *timer* no final de um ciclo do *loop*.

As variáveis globais são inicializadas a 0 e a variável *cur_time* recebe o valor da função *millis()* no início de cada *loop*. No final deste, os valores são atualizados segundo o seguinte excerto de código:

```
timer = cur_time - prev_time + last_timer;
prev_time = cur_time;
last_timer = timer;
```

O timer é obtido através da diferença entre *cur_time* e *prev_time*, corresponde ao tempo decorrido entre dois *loops* consecutivos, que é incrementado ao valor do temporizador anterior, *last_timer*. Seguidamente são atualizados os valores de *prev_time* e *last_timer*.

Na figura seguinte é apresentado o esquema da máquina de estados fsm_BUTTONS:

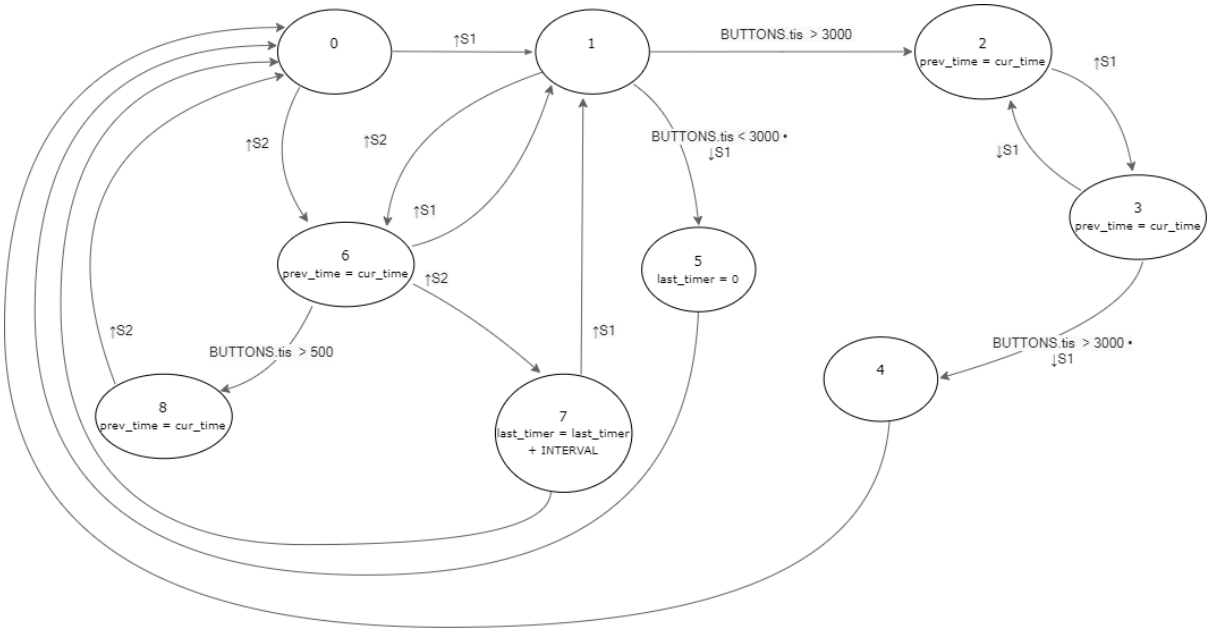

Figura 3 - fsm_BUTTONS

O estado 1 e o estado 6 são estados de decisão que irão verificar se irá ocorrer um *long press* de S1 ou então um *double click* de S2 ou apenas *click* normal de cada um dos botões, respetivamente.

Se o estado 2 ocorrer, ou seja, se ocorreu um *long press* de S1, sabe-se então que se deve proceder à entrada no modo de configuração. O estado 3 serve, como estado de decisão verificando-se a ocorrência de um *long press*, saindo do modo de configuração se tal ocorrer. Nestes dois estados o *timer* é pausado de modo a garantir a continuação da sequência dos *LEDs* após a saída do modo de configuração.

O estado 5 ocorre quando o botão S1 é pressionado e não constitui um *long press,* ou seja, por um intervalo de tempo menor que 3 segundos, devendo-se, portanto, proceder a um *reset* do *timer*, igualando a variável *last_timer* a 0.

O estado 7 ocorre quando o *double click* do botão S2 acontece e, tal como é indicado no enunciado, o *timer* incrementa um intervalo, ou seja, acrescenta-se à variável *last_timer* o valor da o intervalo atual.

Se ocorrer o estado 8 a ocorrer existe um simples *click* de S2 e, portanto, o *timer* é pausado, ação que se consegue ao igualar a variável *prev_time* a *cur_time*. Para continuar a incrementação do *timer* basta pressionar o botão S2 novamente. Note-se que no estado 6 o *timer* é pausado, de modo a impedir a contagem do tempo nos 500 milissegundos necessários à verificação de *double click*.

É de notar que, como inicialmente se pode iniciar a sequência com qualquer um dos botões, S1 ou S2, se se utilizar o botão S2 os *LEDs* irão simultaneamente ligar e entrar logo de seguida em *pause*, ou seja, vão piscar.

fsm_WhichConfig

A máquina de estados seguinte, fsm_WhichConfig, apenas indica qual a configuração atualmente a ser alterada, sendo indicada com os *LEDs* 1 a 3 de acordo com o estado atual desta máquina de estados.

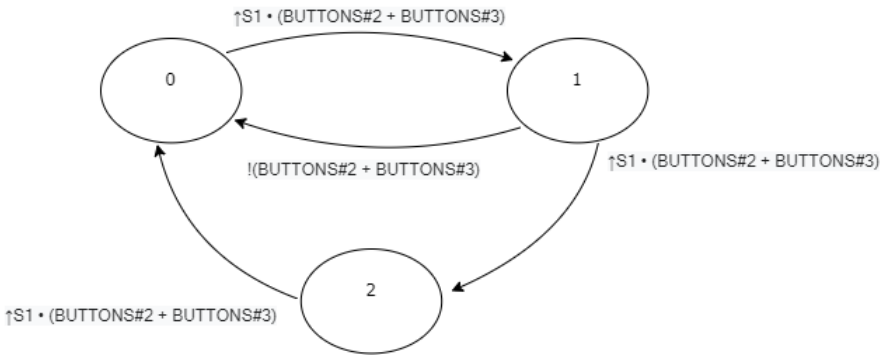

Figura 4 - fsm_WhichConfig

Para cada um dos modos de configuração foram criadas as máquinas de estado fsm_Config0, fsm_Config1 e fsm_Config2, que guardam o estado atual da configuração respetiva, apresentadas de seguida.

fsm_Config0

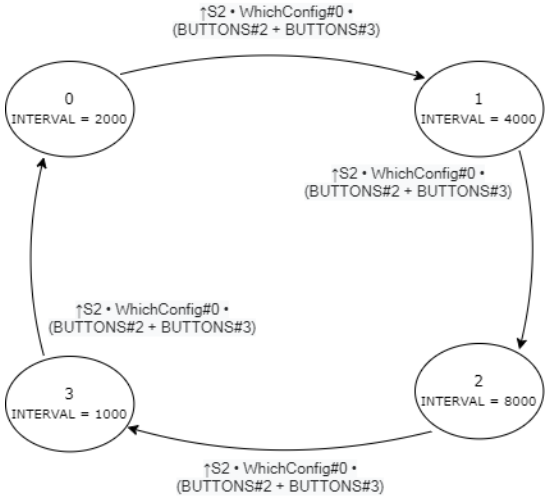

Figura 5 - fsm_Config0

No primeiro modo de configuração é expectável que a variável *INTERVAL* varie sempre que o botão S2 é pressionado. Assim, como o valor inicial da variável é 2000 milissegundos, é a esse valor que o estado 0 iguala o *INTERVAL*, percorrendo a sequência de 2000 para 4000, para 8000 e, finalmente, 1000 milissegundos, retornando depois ao valor inicial.

fsm_Config1

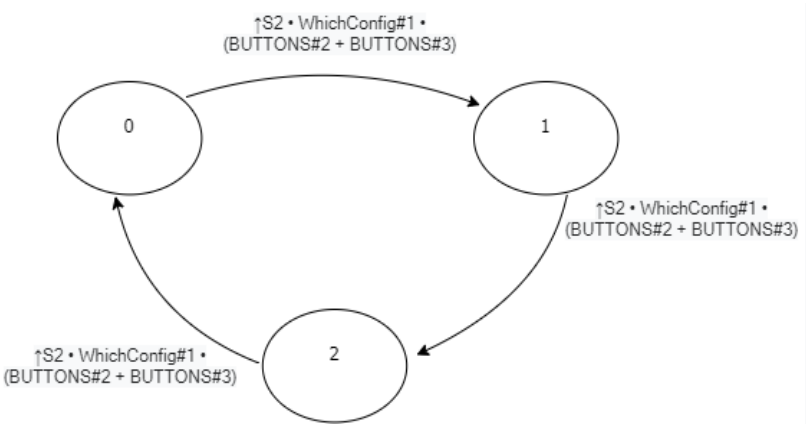

Figura 6 - fsm_Config1

O segundo modo de configuração varia entre o modo normal em que o *LED* desliga depois do seu intervalo, estado 0, o *LED* pisca na segunda metade do seu intervalo, estado 1, ou o *LED* diminui a sua luminosidade gradualmente de 100% até 0% ao longo do intervalo.

fsm_Config2

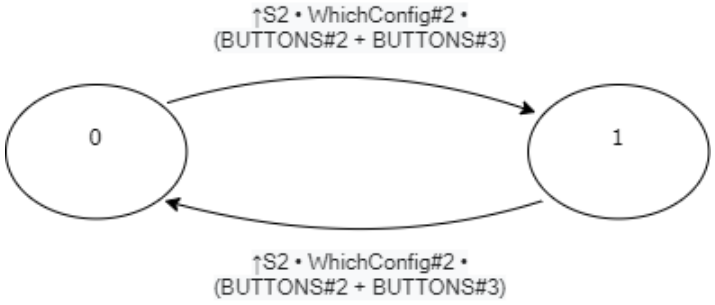

Figura 7 - fsm_Config2

Por fim, o terceiro modo de configuração remete para o *LED7* ligado no final do tempo (estado 0) ou então os *LEDs* e 1 a 6 piscarem no fim do tempo (estado 1).

fsm_LEDxisON

Foi criada uma máquina de estados genérica para cada um dos *LEDs* que apenas indica se este se encontra ligado (estado 1) ou desligado (estado 0). Por razões de simplicidade, apenas se representou uma máquina modelo com uma variável x que representa o número do LED.

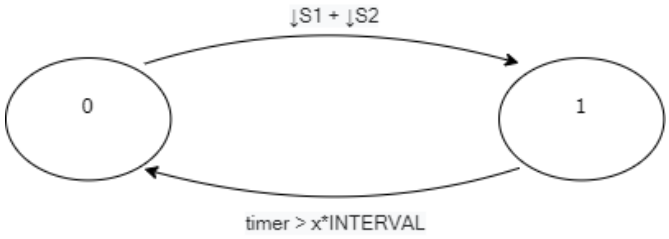

Figura 8 - fsm_LEDxisON

fsm_LED1 a fsm_LED3

A cada uma das três primeiras *LEDs* corresponde uma máquina de estados com funcionamento muito semelhante, reagindo às alterações das configurações e à entrada no modo de configuração, notando-se a diferença nas seguintes transições:

- o estado na máquina de estados fsm_WhichConfig (para o *LED1* o estado deverá ser 0, para o *LED2* o 1 e, finalmente, para o *LED3* o estado 2);
- nome da máquina de estados que indica se o *LED* atual se encontra ligado (LEDxisON, em que x indica o número do *LED*);
- valor do intervalo depois do qual se deve mudar de estado que se dá pela equação (x-1) + ½ , onde x indica o número do *LED*.

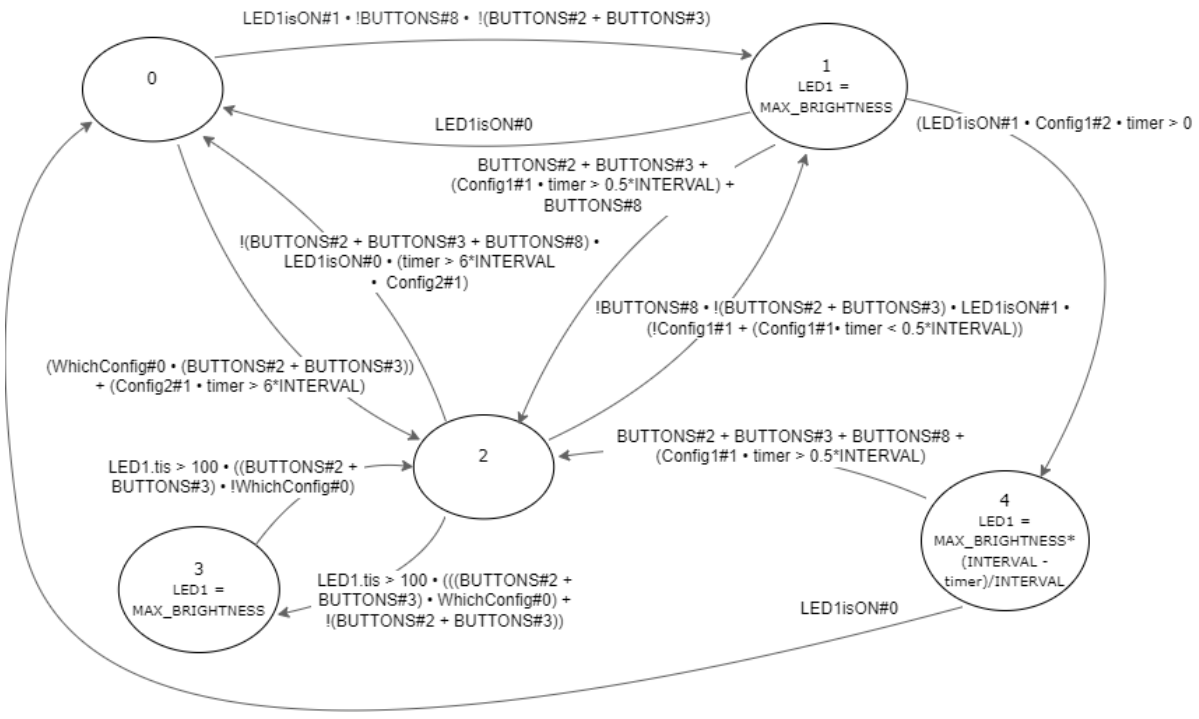

Figura 9 - fsm_LED1

Assim sendo, existem 3 modos possíveis para o *LED*:

- ligada com o máximo de *brightness* (estado 1);
- alterar a variável *brightness* de modo a realizar-se o *fade* proporcional ao intervalo do respetivo *LED* (estado 4);
- piscar, ou seja, alternar entre dois estados, 2 e 3, a cada 100 milissegundos, em que um deles o *LED* se encontra desligado e no outro está com o máximo de *brightness;*

A equação genérica que permite a realização do *fade* é obtida através do produto do valor máximo de *brightness* e da percentagem do intervalo percurrido:

```
LED_x = MAX_BRIGHTNESS*(x*INTERVAL - timer)/(INTERVAL);
```

Com o *click* de qualquer um dos botões e não estando no modo de configuração nem no *pause*, a máquina de estados evolui para o estado 1. No entanto, se estiver no modo de configuração ou no modo que exige que os *LEDs* de 1 a 6 pisquem no final do intervalo irá diretamente para o estado 2.

Para entrar no estado 2 é necessário estar em pelo menos uma destas situações que implicam que o *LED* pisque:

- no modo de configuração;
- no *pause* e *LED* ligado;
- no modo que exige que os *LEDs* de 1 a 6 pisquem na segunda metade do intervalo;
- no modo que exige que os *LEDs* pisquem no final do intervalo;

Só é possível entrar no modo do *fade* se se estiver no modo de configuração correspondente.

fsm_LED4 a fsm_LED6

As máquinas de estado correspondentes aos *LED4* a *LED6* são semelhantes às desenvolvidas para os *LED1* a *LED3*, no entanto, não tem de identificar a configuração a atualmente se alterada. Modifica-se novamente:

- nome da máquina de estados que indica se o *LED* atual se encontra ligado (LEDxisON, em que x indica o número do *LED*);
- valor do intervalo depois do qual se deve mudar de estado que se dá pela equação (x-1) + ½ , em que x indica o número do *LED*;

A máquina de estados correspondente ao *LED4*, fsm_LED4, está representada na figura seguinte.

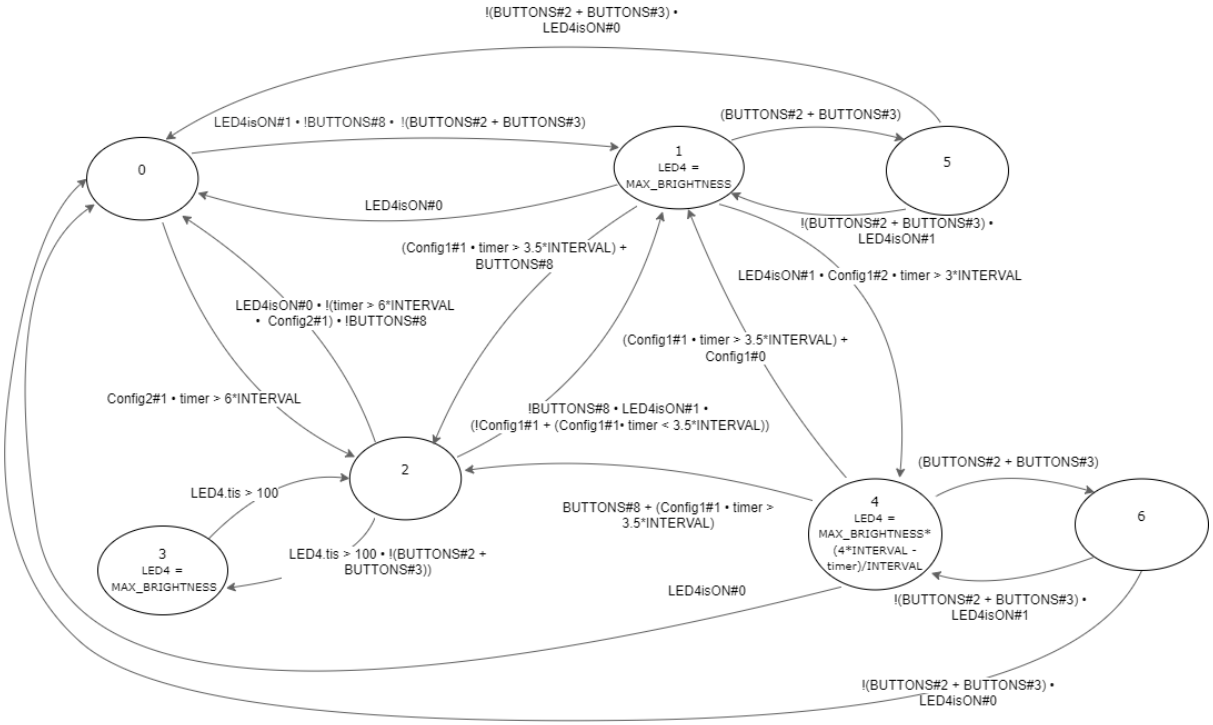

Figura 10 - fsm_LED4

Os *LEDs* têm, mais uma vez apenas três modos de funcionamento, iguais aos descritos na máquina de estados anterior. Foram adicionados os estados 5 e 6 utilizados para desligar o *LED* com a entrada do modo de configuração.

Os *LEDs* deverão piscar, ou seja, entrar no estado 2, nas seguintes situações:

- no *pause* e *LED* ligado;
- no modo que exige que os *LEDs* de 1 a 6 pisquem na segunda metade do intervalo;
- no modo que exige que os *LEDs* pisquem no final do intervalo;

Mais uma vez, só é possível entrar no modo do *fade* se realmente se estiver no modo de configuração correspondente.

fsm_LED7

Por fim, a última *LED* apresenta uma máquina de estados bastante mais complexa e distinta de todas as outras, mas com os mesmos três modos de genéricos de funcionamento. Nos estados 1 e 2, o *LED* apresenta o valor máximo de *brightness* e, no estado 4, ocorre o *fade*.

Devido à complexidade e quantidade de transições teve de ser criada uma legenda que apresenta as condições de mudança de estado. No início de cada condição são apresentados os estados entre os quais ocorre a transição com a estrutura, por exemplo, 0 - 2, que representa a transição do estado 0 para o estado 2.

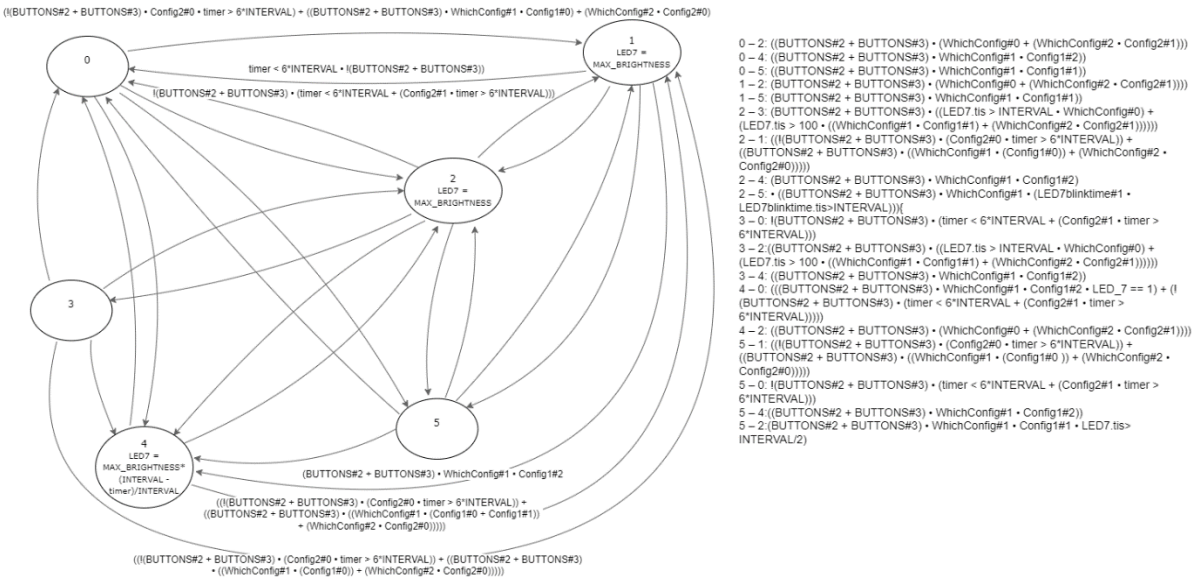

Figura 11 - fsm_LED7

CONCLUSÃO

Através da realização deste trabalho foi possível a implementação de uma ampulheta eletrónica e a sua representação com *LEDs*. Para tal, utilizaram-se máquinas de estado de modo a interpretar diferentes tipos de *input*, verifica-se o correto funcionamento das funções de *pause* e incrementação de intervalo, assim como todas as configurações especificadas.