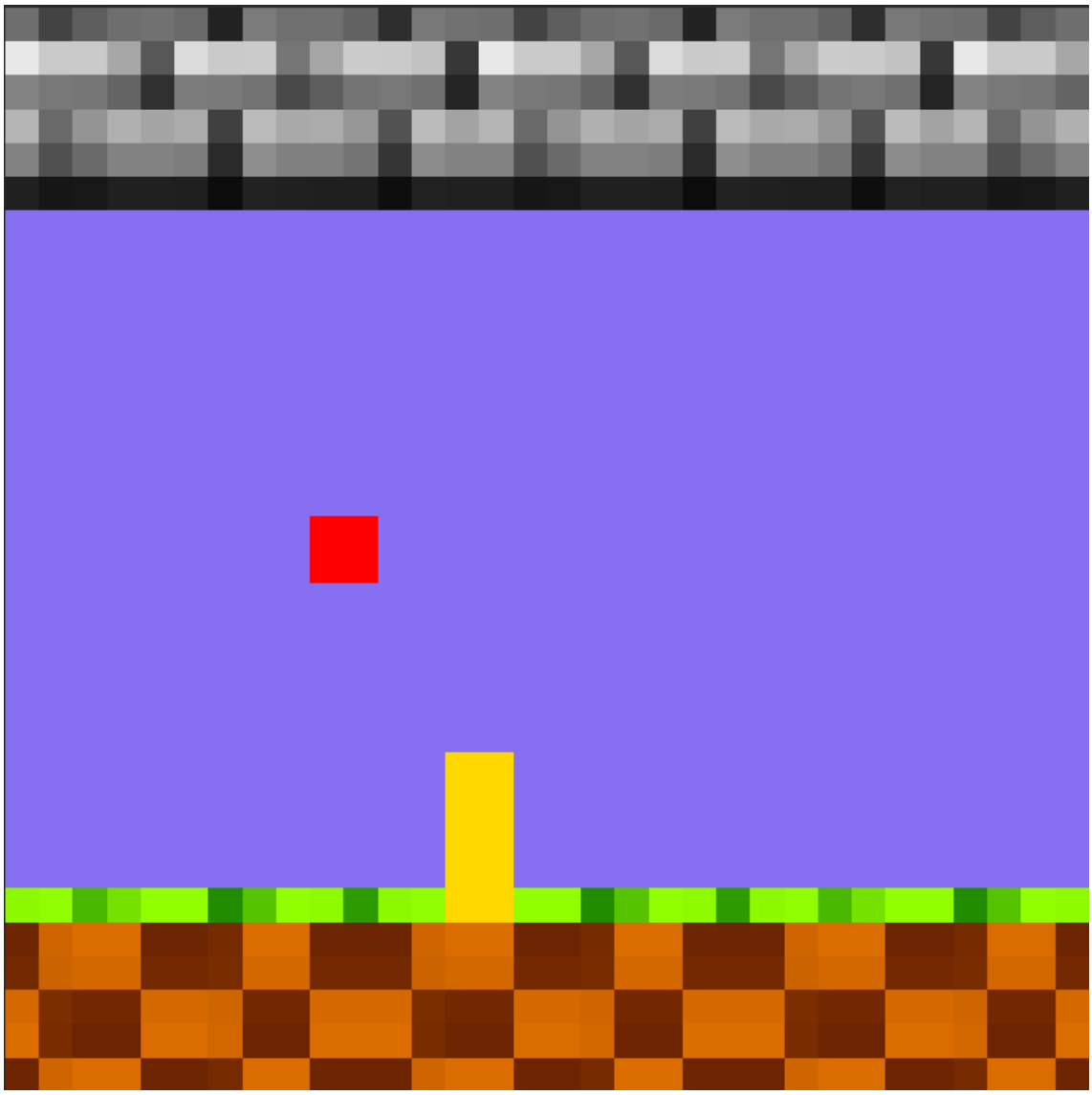
Projeto de Sistemas Digitais - 2020/2021

Square Jumper
PL1- Grupo6

José Pedro Araújo Azevedo - 2016236736

Rafael Alves Vieira - 2017257239

May 30, 2021

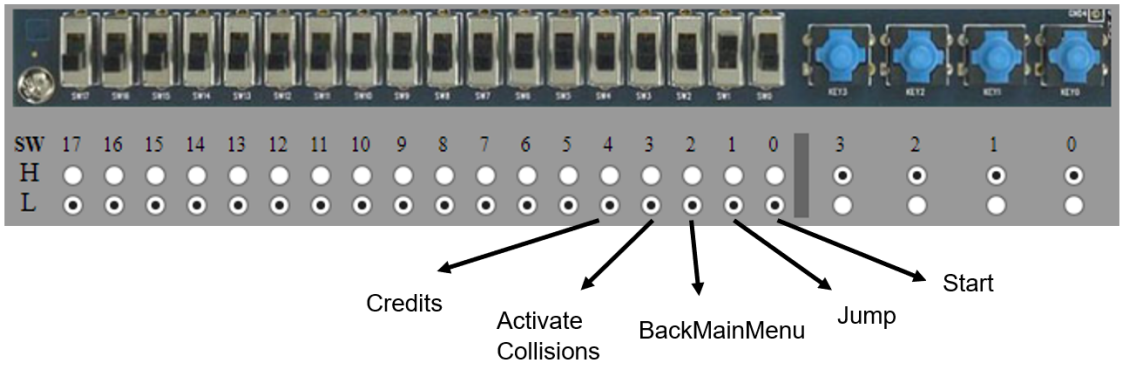


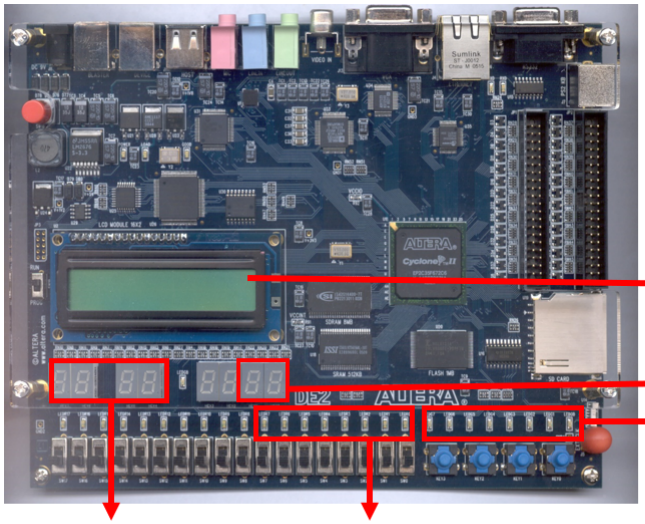
INTRODUÇÃO

Este relatório complenta a implementação realizada do projeto final para a cadeira de Projeto de Sistemas Digitais. Este projeto consiste em um jogo simples no qual, o utilizador necessita de ”saltar” para desviar-se de obstáculos acumulando dessa forma pontos.

GUIA DO UTILIZADOR

Foi implementado um menu principal, no qual o utilizador pode começar a jogar ou então vizualiza os ”créditos” do jogo, nomeadamente autores do projeto e docentes da cadeira. Para começar um novo jogo é necessário premir o switch SW[0]. Encontra-se ilustrado nas figuras seguintes um esquema que resume o guia de utilizador.





Mostra os “créditos” do jogo. Nomeadamente autores do projeto e docentes da cadeira.

Mostra o score.

LEDG[7..0] Ficam intermitentes quando as colisões estão ligadas.

Indica o número do estado que se encontra.

LEDR[7..0] Ativam quando o utilizador colide com um obstáculo

MÉTODO DE PROJECTO RTL

Máquina de Estados Finita de Alto Nível

Inputs:

Start(bit);

Jump(bit);

Over(bit);

BackMainMenu(bit);

Ton(bit);

Credits(bit);

Outputs:

GameOver(bit);

Restart(bit);

Menu(bit);

TurnOn(bit);

JumpG(bit);

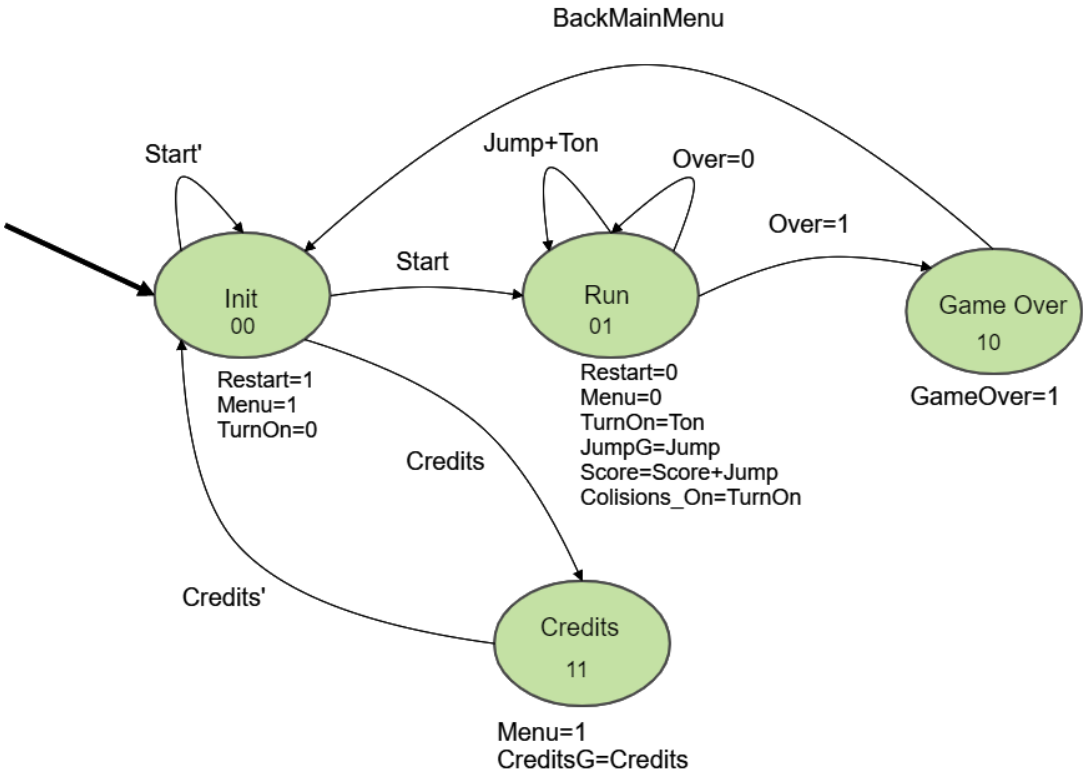
CreditsG(bit);

Local Registers:

Score(5 bits);

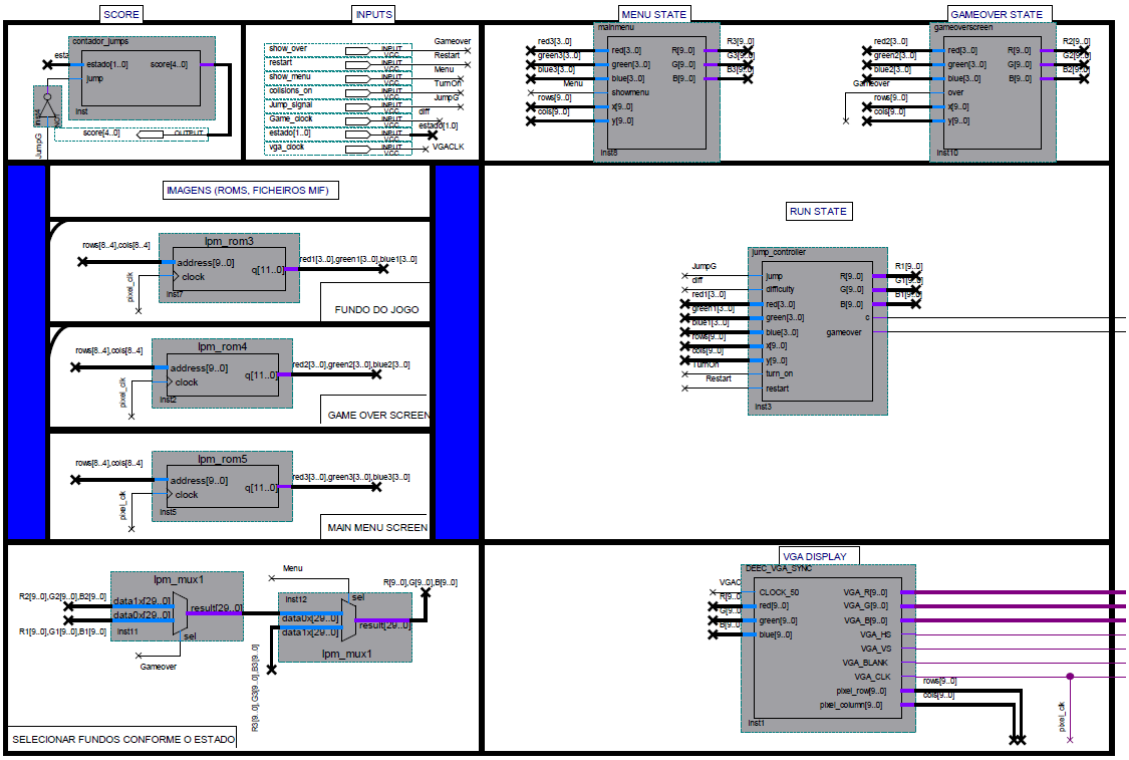
Over(bit);

Colisions_On(bit);



MÉTODO DE PROJECTO RTL

Datapath



Imagens, ROMS e ficheiros MIF

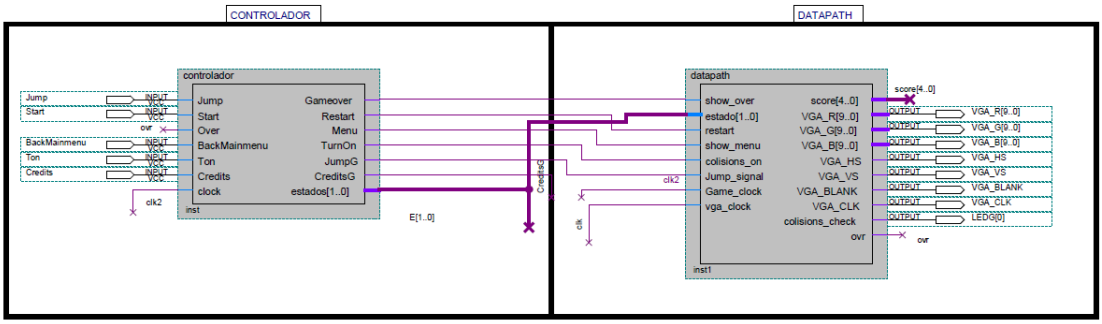
De modo a não sobrecarregar os recursos da placa, e como planeávamos recorrer ao uso de múltiplas imagens utilzámos imagens de resolução baixa, sempre de 32*32 pixeis. Os endereços dos pixeis estão em binário, por isso não precisamos de usar um bloco para convertê-los como se fazia na char_rom. Temos 4 bits para representar a intensidade de cada cor, e em cada endereço associamos 12 bits de *data* (para as 3 componentes de cor), onde a componente vermelha são os 4 bits mais significativos, e a componente azul são os 4 bits menos significativos. Para criar estes ficheiros MIF utilizámos o Matlab (código irá em anexo). Utilizámos três ROMs: para o estado do menu principal, o estado do jogo, e para o estado de *gameover*. Para escolher entre as imagens, temos os seus outputs de côr ligados a multiplexers que as selecionam, conforme os inputs dados pelo controlador.

Run State

Para o estado de *run* e também para os blocos do *score*, *menu* e *gameover* utilizámos descrição comportamental. No bloco *jump_controler* é onde se encontra a lógica que promove o funcionamento do jogo. Para além do fundo do jogo, queremos mostrar um quadrado (que representa o jogador) e um retângulo que representa os obstáculos. Portanto, vamos lendo as coordenadas do pixel, e quando estas coordenadas estão na zona que delimitamos ser o quadrado, escolhemos escrever outro output de cores em vez do fundo. E fazemos de igual forma para o retângulo. O que causa o movimento do quadrado e do retângulo são contadores, no caso do quadrado, sempre que há uma *rising_edge* no switch do jump, incrementa até 200 pixeis de altura e volta a decrementar para a posição inicial. Este contador depois entra nas condições dos *if statements* que usamos para delimitar a zona onde desenhamos um quadrado no ecrã ou então o fundo do ecrã. Um raciocínio semelhante é tomado para os obstáculos.
Para detetar colisões precisamos apenas de verificar se o deslocamento causado pelo contador do obstáculo atingiu a zona do quadrado (no eixo do x) e caso tenha atingido, se o nosso contador que usamos para o *salto* do jogador está acima da altura do obstáculo. Caso isto se verifique, enviamos um sinal de deteção de colisão para o controlador.

MÉTODO DE PROJECTO RTL

Ligação do Datapath ao Controlador



MÉTODO DE PROJECTO RTL

Máquina de Estados Finita do Controlador

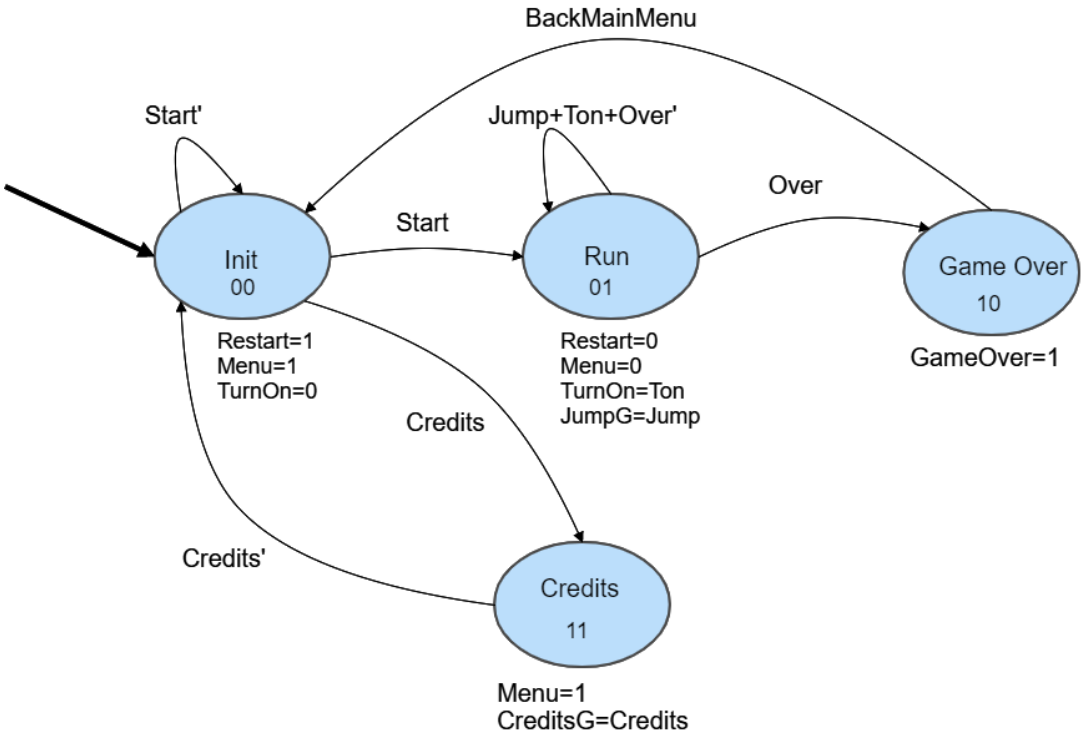


Tabela de Verdade

	Inputs								Outputs							
Estado	S1	S0	Start	Jump	Over	BackMain menu	Ton	Credits	N1	N0	GameOver	Restart	Menu	TurnOn	JumpG	CreditsG
Init	0	0	0	x	x	x	x	x	0	0	0	1	1	0	0	0
			1	x	x	x	x	x	0	1	0	0	0	0	0	0
			x	x	x	x	x	1	1	1	0	0	1	0	0	1
Run	0	1	x	1	0	x	0	x	0	1	0	0	0	0	1	0
			x	0	0	x	1	x	0	1	0	0	0	1	0	0
			x	1	1	x	1	x	1	0	1	0	0	1	1	0
			x	0	1	x	1	x	1	0	1	0	0	1	0	0
Game Over	1	0	x	x	x	0	x	x	1	0	0	0	0	0	0	0
			x	x	x	1	x	x	0	0	0	0	1	0	0	0
Credits	1	1	x	x	x	x	x	0	0	0	0	0	0	0	0	0
			x	x	x	x	x	1	1	1	0	0	0	0	0	1

BIBLIOGRAFIA

Rapid Prototyping of Digital Systems, James O. Hamblen and Michael D. Furman.
biblioteca .de2 disponibilizada no inforestudante.
lcd_example.vhd e *lcd_controller.vhd* disponibilizados no inforestudante.