

Universidade Federal de Santa Catarina  
EEL5105: Circuitos e Técnicas Digitais  
Semestre: 2017/1 – Projeto

Test Drive

O projeto final consiste na implementação de um circuito na placa de desenvolvimento *DE1* fazendo uso das estruturas e conhecimentos obtidos durante o curso. O circuito vai implementar a um jogo interativo similar a um jogo de teste de condução. O comportamento do jogo está definido a seguir:

- O usuário inicia no estado *Init* e da início ao jogo pressionando o botão de pressão enter (KEY1). Uma vez no estado *Setup* o usuário deve escolher um dos quatro mapas de obstáculos (nível de jogo) com os *Switches* 8 e 7 (SW8..7).
- Uma vez selecionado o nível de jogo o usuário pulsa *enter* de novo (botão de pressão KEY1) e começa o jogo (estado *Game*).
- Uma vez no estado *Game* o usuário inicia a condução na posição coluna-0 e fila-7 à velocidade 0. O usuário pode mover o veículo para cima (↑) e para baixo (↓) usando os switches SW0 e SW1, respectivamente. Para modificar a velocidade o usuário pode usar os botões de pressão KEY3 e KEY2 para frear e acelerar, respectivamente. A velocidade do veículo vai de 0 (*Stop*) a velocidade 5 (*Max*).
- As posições por onde o usuário pode transitar é uma matriz (mapa de bits de  $16 \times 32$ ), e deve dar duas voltas ao circuito de obstáculos para chegar ao objectivo. Assim, o usuário deve passar 64 colunas para chegar ao final. A linha de horizonte de obstáculos será mostrado nos LEDs LED9.. 0 onde o LED9 mostrará sempre o veículo piscando a 5Hz.
- Durante o modo *Game* painel de controle do veículo será mostrado nos displays HEX5.. 0 com dois modos de selecção usando o switch SW9. Quando SW9 = 0 HEX5 e HEX4 mostram a letra *E* de estados e o numero de estado da máquina de controle, respectivamente. HEX3 e HEX2 mostra a letra *b* de *Bonus* e o numero bonus (número de vezes que o usuário pode bater na corrida). A contagem descendente de 99 a 0 será mostrada no Displays HEX1 e HEX0.
- Quando SW9 = 1 HEX5 e HEX4 mostram a letra *S* de *Speed* e a velocidade atual, respectivamente. O HEX3 indica a letra *P* de *Position* e HEX2.. 0 mostraram a posição do veículo em formato hexadecimal (fila no HEX2 e coluna no HEX1.. 0). Por exemplo se o veículo se encontra na segunda volta na fila 5 e coluna A, os displays HEX3.. 0 mostraram os caracteres *P52A*.
- O usuário deve evitar os obstáculos no tempo de jogo usando os *Switches* SW0 e SW1 e os botões de pressão KEY3 e KEY2. O numero de vezes que o usuário pode bater com um obstáculo depende do numero de bônus. A contagem de bônus ira diminuindo com cada batida. Caso o numero de bônus ou o tempo acabe antes de chegar ao objectivo o jogo terminara (estado *End*) e será mostrado nos Displays os pontos obtidos. Nos displays HEX3... 0, onde HEX3 e HEX2 indicam as letras *Pt* de *Point* e HEX1 e HEX0 indicam os pontos obtidos. Em este estado *End* os displays HEX5 e HEX4 indicaram os caracteres *E3* (Estado 3). Em Após estado *End* o jogo voltará ao estado *Init*.
- Um usuário pode em qualquer momento parar o jogo usando o botão de pressão KEY0 zerando a contagem de alvos, para assim re-iniciar de novo.

O esquema geral do projeto é mostrado na Figura 1 e inclui sete blocos diferenciados:

- *Mapas*: Quatro unidades onde se declaram os mapa de bits referentes aos mapas de obstáculos.
- *Contadores*: Circuitos sequenciais encarregados de gerar três contagens: *i*) contagem decrescente em segundos de dois números em decimal  $99 \rightarrow 0$ , *ii*) Contagem descendente de bônus e *iii*) Contagem ascendente de posição.
- *Comparadores e somador*: Circuitos combinatórios encarregados de gerar bits que determinam se o jogo acabou. O somador é encarregado de fazer a soma de pontos no final do jogo (posição horizontal no final do jogo mais  $2 \times$  bonus).
- *Registradores*: Circuitos sequenciais encarregados de gerar os arrays que definem a velocidade de jogo, posição do veículo e horizonte visível do mapa de obstáculos.
- *Controlador*: Máquina de controle do jogo.
- *Selectores*: Multiplexadores que fornecem sinais de para outros blocos e as saídas para os LEDs e Displays.
- *Debouncer*: Visando evitar problemas de temporização em função do aperto de um KEY por um ser humano durar muitos ciclos de *clock*, o *Button Press Synchronizer* (ButtonSync) será fornecido em conjunto com o projeto deve ser utilizado.

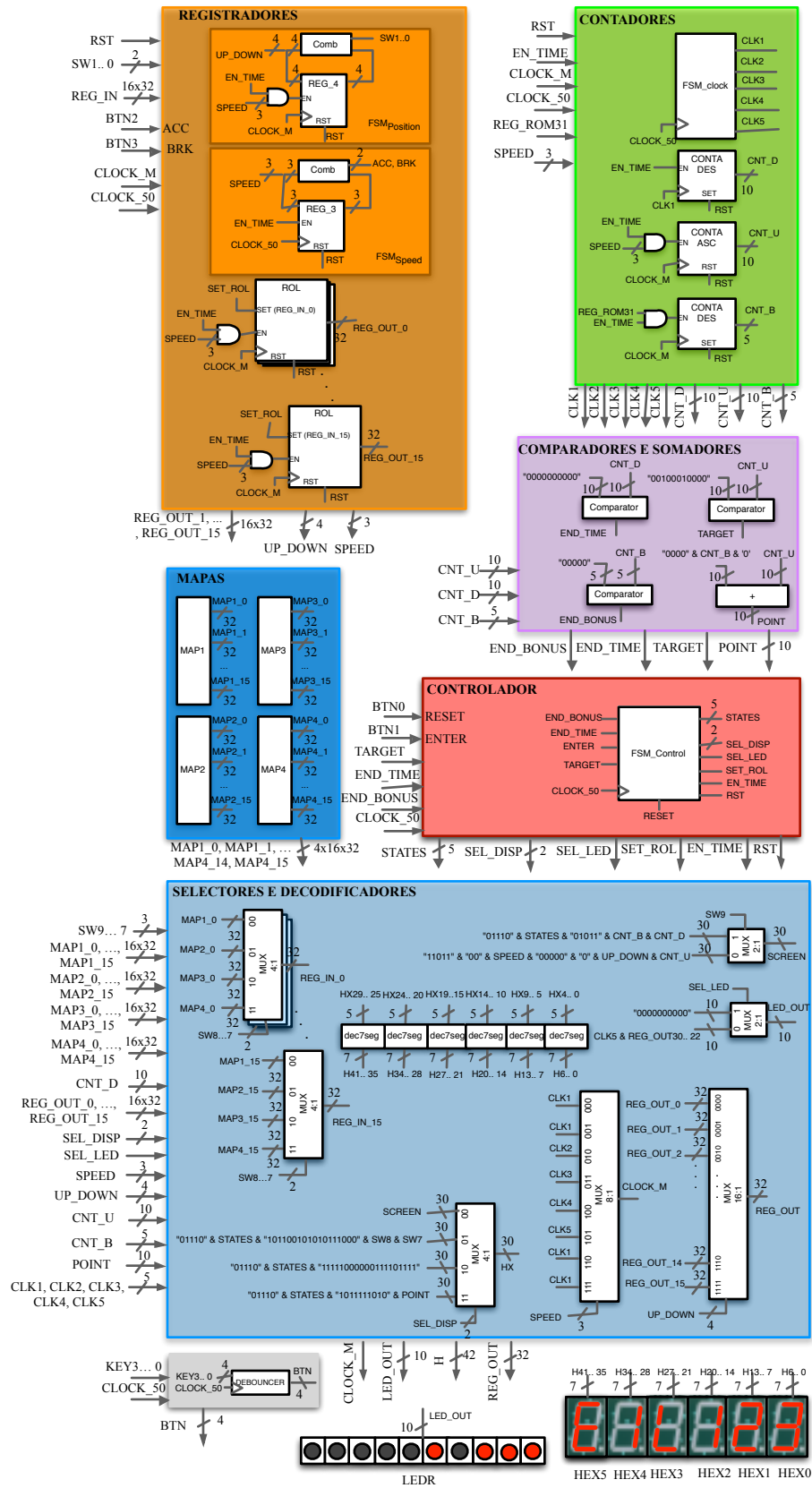


Figura 1: Diagrama de blocos do jogo test drive.

**TESTES** No moodle da disciplina são fornecidas as unidades que fazem os mapas de obstaculos (MAP1.vhd, MAP2.vhd, MAP3.vhd, MAP4.vhd). Duas delas estão sem preencher e os alunos podem anotar a posição dos obstaculos. O exemplo de mapa de obstáculos (MAP1) é mostrado na Figura 2.

*Dica: O aluno pode conferir o mapa da Figura 2 com a unidade MAP1.vhd e descrever as unidades MAP3 e MAP4 de forma similar.*

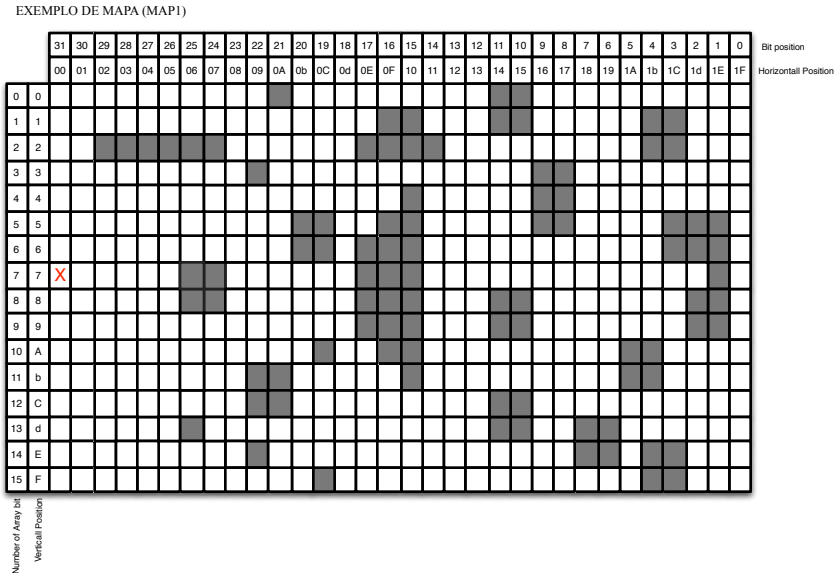


Figura 2: Exemplo de test drive para preenchimento das memorias.

**REGISTRADORES** Os registradores correspondem com 2 FSMs e 16 registradores com deslocamento para esquerda e rotação (ROL).

A primeira FSM corresponde com a FSM de controle de posição, o diagrama de estados é mostrado na Figura 3(a). O aluno pode implementar de forma estrutural (como mostrado na Figura 1) ou comportamental a partir do diagrama de estados.

A segunda FSM corresponde com a FSM de controle de velocidade, o diagrama de estados é mostrado na Figura 3(b). O aluno pode implementar de forma estrutural (como mostrado na Figura 1) ou comportamental a partir do diagrama de estados.

As restantes unidades fornecem o horizonte de mapa de obstáculos, elas podem ser implementadas usando uma operação de deslocamento à esquerda mais rotação (Operação ROL).

*Dica: Para a implementação da operação de deslocamento e rotação o aluno pode usar as dicas fornecidas na aula 9.*

## COMPARADORES E SOMADORES

Circuitos combinatórios encarregado de gerar sinais de controle quando o tempo de jogo/bônus chegar ao fim (sinais *END\_TIME/END\_BONUS*) ou foi atingido a posição da coluna 64 (sinal *TARGET*). Os comparadores deveram gerar um '1' lógico quando sejam ativados ditos sinais.

O somador é encarregado de fazer a soma de pontos no final do jogo (posição horizontal no final do jogo mais 2× bonus). Para fazer a multiplicação ×2 o aluno deverá usar as dicas fornecidas na aula 9.

*Dica: Para a implementação do comparador sugere-se usar uma abordagem estrutural, baseada em portas ANDs.*

## CONTADORES

Neste bloco há uma FSM encarregada de gerar cinco clocks:  $CLK1 = 1Hz$ ,  $CLK2 = 2Hz$ ,  $CLK3 = 3Hz$ ,  $CLK4 = 4Hz$  e  $CLK5 = 5Hz$  a partir do *clock* interno de 50MHz da placa DE1. O  $CLK1$  será utilizado para o contador descendente  $99 \rightarrow 0$  (implementado também neste bloco).

A contagem decrescente de tempo de jogo  $99 \rightarrow 0$  deverá ser mostrada nos *Displays* HEX1.. 0 ( $SW9 = 0$ ) em formato decimal a partir do sinal *CNT\_D*.

A contagem decrescente de bônus de jogo deverá ser mostrada nos *Display* HEX2 ( $SW9 = 0$ ) a partir do sinal *CNT\_B*.

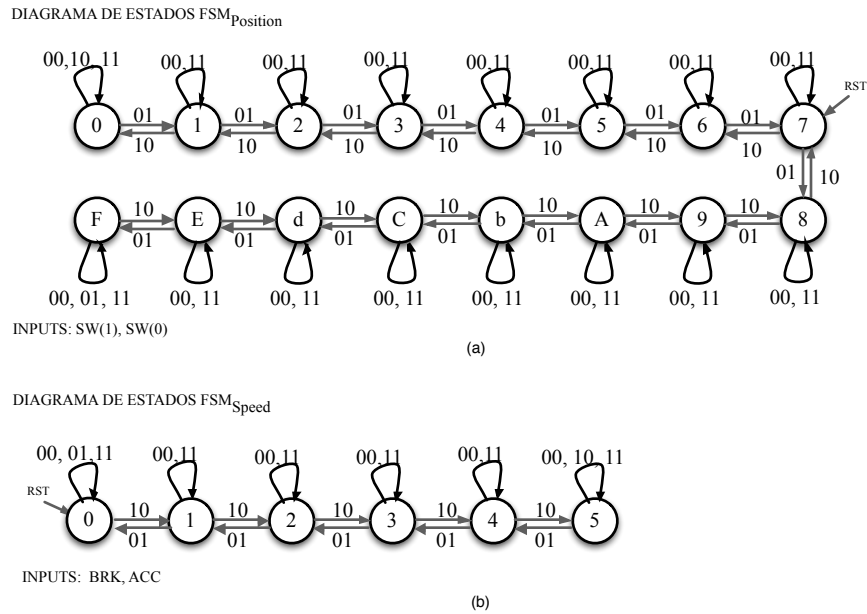


Figura 3: Diagrama de estados do para a selecção de a) posição e b) velocidade.

A contagem crescente de posição  $0 \rightarrow 2F$  deverá ser mostrada nos *Displays* HEX3.. 0 ( $SW9 = 1$ ) a partir do sinal  $CNT\_U$ .

*Dicas: Recomenda-se aplicar os conceitos aprendidos na aula 7b sobre o conversor de frequência  $50MHz \rightarrow 1Hz$ .*

### SELETORES E DECODIFICADORES

Esta unidade consiste em 21 multiplexadores, um de 16:1, um de 8:1, 17 de 4:1 e dois de 2:1. Os multiplexadores deverão atuar para seleccionar o que é esperado nos *Displays* e LEDs para cada estado e fornecer sinais para outros blocos.

O aluno de incluir mais um bit de entrada no decodificador de 7 segmentos desenvolvido na aula 5 e incluir até 16 caracteres adicionais mostrados na Figura 4(a) e alguns exemplos de caracteres para sete segmentos são mostrados na Figura 4(b).

TABELA DEC7SEG

C4	C3	C2	C1	C0	Car
0	0	0	0	0	0
0	0	0	0	1	1
0	0	0	1	0	2
0	0	0	1	1	3
0	0	1	0	0	4
0	0	1	0	1	5
0	0	1	1	0	6
0	0	1	1	1	7
0	1	0	0	0	8
0	1	0	0	1	9
0	1	0	1	0	A
0	1	0	1	1	b
0	1	1	0	0	C
0	1	1	0	1	d
0	1	1	1	0	E
0	1	1	1	1	F
1	0	0	0	0	g
1	0	0	0	1	h
1	0	0	1	0	i
1	0	0	1	1	
1	0	1	0	0	
1	0	1	0	1	
1	0	1	1	0	m
1	0	1	1	1	
1	1	0	0	0	P
1	1	0	0	1	
1	1	0	1	0	
1	1	0	1	1	
1	1	1	0	0	
1	1	1	0	1	t
1	1	1	1	0	
1	1	1	1	1	...

(a)

(b)

Figura 4: Tabela de verdade para obtenção de decodificador de 7 segmentos.

*Dicas: Recomenda-se aplicar os conceitos aprendidos na aula 4 sobre o Multiplexador e aula 5 de decodificador.*

## CONTROLADOR

Este bloco é responsável por gerar os sinais de controle para os outros blocos. O controlador deve implementar o diagrama de estados mostrado na Figura 5. O aluno deve explicar no trabalho o funcionamento da FSM de controle apresentada.

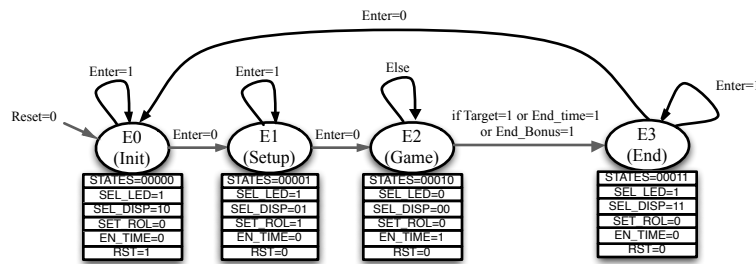


Figura 5: Diagrama de estados para o controlador.

*Dica: É de salientar que o diagrama de estados mostrado não é único, e o aluno possui a liberdade de pode projetar outro diagrama de estados com a mesma funcionalidade.*

## DEBOUNCER

Visando evitar problemas de temporização em função do aperto de um KEY por um ser humano durar muitos ciclos de *clock*, o *Button Press Synchronizer* (ButtonSync) será fornecido em conjunto com o projeto deve ser utilizado. O *ButtonSync* converte apertos das KEYS em pulsos com período de um ciclo de *clock*. Assim, em seu projeto, as KEYS devem ser ligadas nas estradas do *ButtonSync*, e as saídas BTN0 a BTN3 do *ButtonSync* deverão ser utilizadas para controlar o projeto.

### Orientações Gerais:

- Na apresentação, todos os membros do grupo deverão estar presentes:
- tolerados, resultando em nota zero.

Atrasos não serão

- Os alunos devem levar a planilha impressa dos mapas de obstáculos na apresentação do trabalho.