O módulo *LCD*  implementado é constituído por dois blocos: *i*) o control em software ; e *ii*) o bloco de LCD, conforme ilustrado na Figura 1. Neste caso o módulo de controlo, implementado em *software*, é a entidade de envio para o LCD.

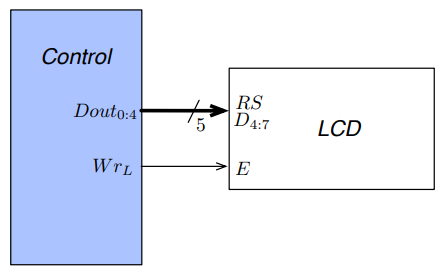
****

Figura 1 – Diagrama de blocos do módulo *LCD*

# Interface com o LCD

O bloco *Key Decode* implementa um descodificador de um teclado matricial 4x3 por *hardware*, sendo constituído por três sub-blocos: *i)* um teclado matricial de 4x3; *ii)* o bloco *Key Scan,* responsável pelo varrimento do teclado; e *iii)* o bloco *Key Control*, que realiza o controlo do varrimento e o controlo de fluxo, conforme o diagrama de blocos representado na Figura 2a. O controlo de fluxo de saída do bloco *Key Decode* (para o módulo *Key Buffer*), define que o sinal *Kval* é ativado quando é detetada a pressão de uma tecla, sendo também disponibilizado o código dessa tecla no barramento *K0:3*. Apenas é iniciado um novo ciclo de varrimento ao teclado quando o sinal *Kack* for ativado e a tecla premida for libertada. O diagrama temporal do controlo de fluxo está representado na Figura 2b.



a) Diagrama de blocos



b) Diagrama temporal

Figura 2 – Bloco *Key Decode*

O bloco *Key Scan* foi implementado de acordo com o diagrama de blocos representado na Figura 3. **[Adicionar a justificação da opção tomada.]**

O bloco *Key Control* foi implementado pela máquina de estados representada em *ASM-chart* na Figura 4. **[Adicionar a descrição da solução apresentada.]**

A descrição hardware do bloco *Key Decode* em CUPL/VHDL encontra-se no Anexo A.

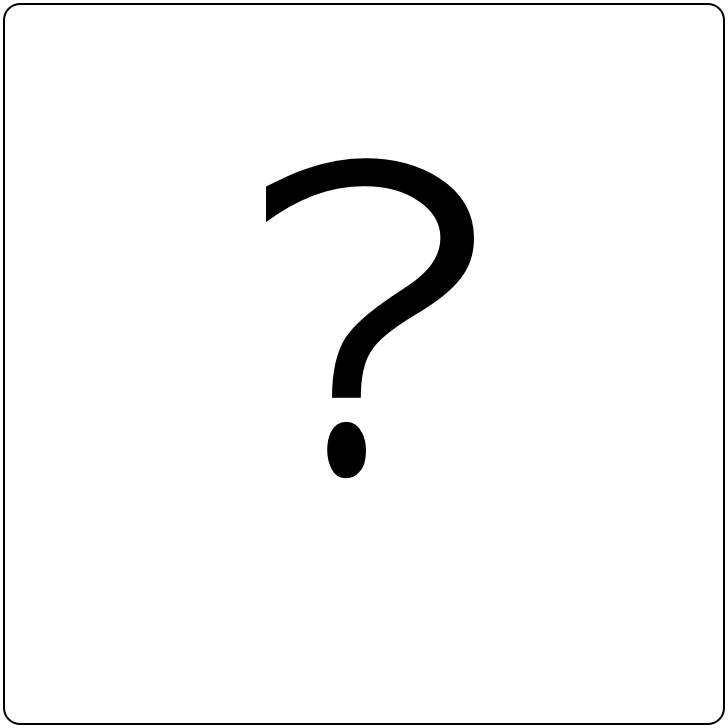


Figura 3 - Diagrama de blocosdo bloco *Key Scan*

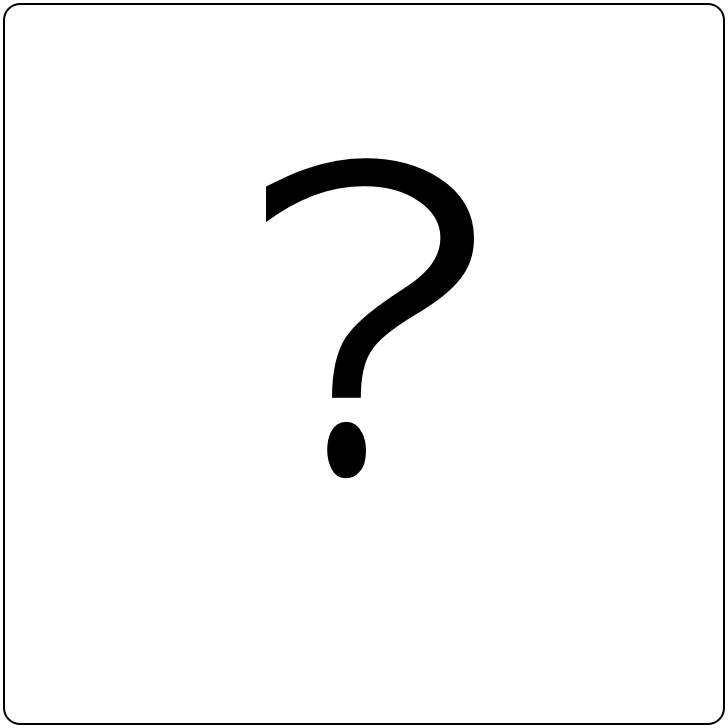


Figura 4 – Máquina de estados do bloco *Key Control*

Com base nas descrições do bloco *Key Decode* implementou-se parcialmente o módulo *Keyboard Reader* de acordo com o esquema elétrico representado no Anexo C. **[Justificar as opções tomadas, como por exemplo, o valor das resistências, as frequências de relógio, etc.]**

O módulo *Key* *Buffer* implementa uma estrutura de armazenamento de dados, com capacidade de uma palavra de quatro bits. A escrita de dados no *Key Buffer* inicia-se com a ativação do sinal *DAV* (*Data Available*) pelo sistema produtor, neste caso pelo *Key Decode*, indicando que tem dados para serem armazenados. Logo que tenha disponibilidade para armazenar informação, o *Key Buffer* escreve os dados *D0:3* em memória. Concluída a escrita em memória, ativa o sinal *DAC* (*Data Accepted*) para informar o sistema produtor que os dados foram aceites. O sistema produtor mantém o sinal *DAV* ativo até que *DAC* seja ativado. O *Key Buffer* só desativa *DAC* depois de *DAV* ter sido desativado.

A implementação do *key Buffer* deverá ser baseada numa máquina de controlo (*Key Buffer Control*) e num registo externo (*Output Register*), conforme o diagrama de blocos apresentado na Figura 5.



Figura 5 – Diagrama de blocos do *Key Buffer*

O bloco *Key Buffer Control* do *Key Buffer* é também responsável pela interação com o sistema consumidor, neste caso o módulo *Control*. O *Control* quando pretende ler dados do *Key Buffer*, aguarda que o sinal *Dval* fique ativo, recolhe os dados e ativa o sinal *ACK* indicando que estes já foram consumidos.

O *Key Buffer Control*, logo que o sinal *ACK* fique ativo, deve invalidar os dados baixando o sinal *Dval*, só deverá voltar a armazenar uma nova palavra depois do *Control* ter desativado o sinal *ACK*.

O bloco *Key Buffer Control* foi implementado de acordo com o diagrama de blocos representado na Figura 6. **[Adicionar a justificação da opção tomada.]**

A descrição hardware do bloco *Key Buffer Control* em CUPL/VHDL encontra-se no Anexo B.

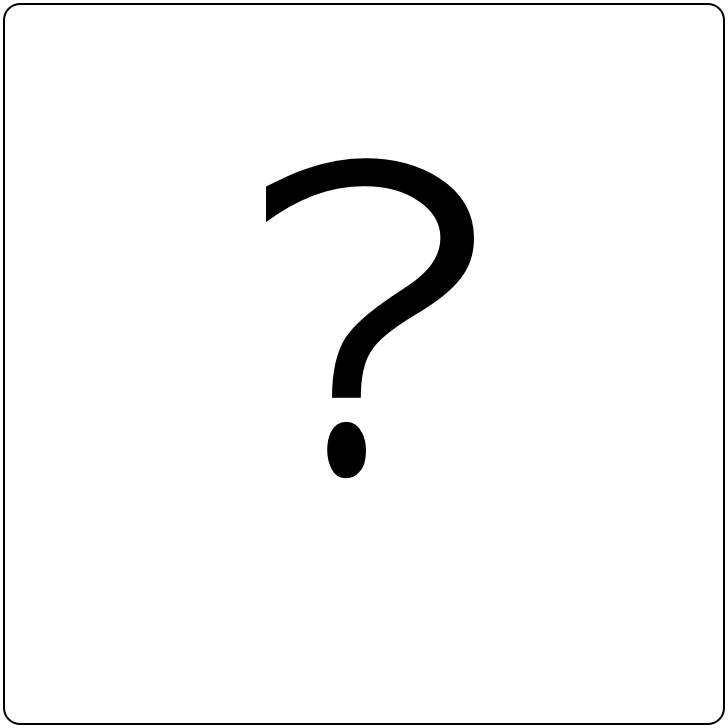


Figura 6 - Máquina de estados do bloco *Key Buffer Control*

Com base nas descrições do bloco *Key Decode* e do bloco *Key Buffer Control* implementou-se o módulo *Keyboard Reader* de acordo com o esquema elétrico representado no Anexo C. **[Justificar as opções tomadas, como por exemplo, as frequências de relógio, etc.]**

# Interface com o *Control*

Implementou-se o módulo *Control* em *software*, recorrendo a linguagem *Kotlin* ou *Java* e seguindo a arquitetura lógica apresentada na Figura 8.



Figura 8 – Diagrama lógico do módulo *Control* de interface com o módulo *Keyboard Reader*

As classes *HAL* e *KBD* desenvolvidas são descritas nas secções 3.1. e 3.2, e o código fonte desenvolvido nos Anexos C e D, respetivamente.

## Classe *LCD*

**[Descrever nesta secção de forma sucinta a classe, referindo se adicionaram métodos.]**

## Classe *TUI*

**[Descrever nesta secção de forma sucinta a classe, referindo se adicionaram métodos.]**

# Conclusões

**[Escrever nesta secção as conclusões da implementação do modulo, incluindo os recursos utilizados, latência na de deteção de tecla, etc.]**

1. Esquema elétrico do módulo *LCD*

1. Código *Kotlin* da classe *LCD*
2. Código *Kotlin* da classe *TUI*