Logo, company name

Description automatically generated

**Licenciatura em Engenharia Informática e de Computadores**

Sistema de Controlo de Acessos

(*Access Control System*)

Pedro Miguens Matutino (pedro.miguens@isel.pt)

Diego Passos (diego.passos@isel.pt)

Manuel Carvalho (manuel.carvalho@isel.pt)

Nuno Sebastião (nuno.sebastiao@isel.pt)

Projeto

de

Laboratório de Informática e Computadores

2022 / 2023 verão

04 de abril de 2023

[1 Introdução 2](#_Toc131410317)

[2 Arquitetura do sistema 3](#_Toc131410318)

[A. Interligações entre o HW e SW 4](#_Toc131410319)

[B. Código *Kotlin* - *HAL* 5](#_Toc131410320)

[C. Código *Kotlin* - *KBD* 6](#_Toc131410321)

[D. Código *Kotlin* - *SerialEmitter* 7](#_Toc131410322)

[E. Código *Kotlin* - *LCD* 8](#_Toc131410323)

[F. Código *Kotlin* - *Door Mechanism* 9](#_Toc131410324)

[G. Código *Kotlin* - *TUI* 10](#_Toc131410325)

[H. Código *Kotlin* - *FileAccess* 11](#_Toc131410326)

[I. Código *Kotlin* - *Users* 12](#_Toc131410327)

[J. Código *Kotlin* - *Log* 13](#_Toc131410328)

[L. Código *Kotlin* da classe *M* 14](#_Toc131410329)

[M. Código *Kotlin* – *Access Control System* *- App* 15](#_Toc131410330)

# Introdução

Neste projeto implementa-se um sistema de controlo de acessos (*Access Control System*), que permite controlar o acesso a zonas restritas através de um número de identificação de utilizador (*User Identification Number – UIN*) e um código de acesso (*Personal Identification Number - PIN*). O sistema permite o acesso à zona restrita após a inserção correta de um par *UIN* e *PIN*. Após o acesso válido o sistema permite a entrega de uma mensagem de texto ao utilizador.

O sistema de controlo de acessos é constituído por: um teclado de 12 teclas; um ecrã *Liquid Cristal Display* (LCD) de duas linhas de 16 caracteres; um mecanismo de abertura e fecho da porta (designado por *Door Mechanism*); uma chave de manutenção (designada por M) que define se o sistema de controlo de acessos está em modo de Manutenção; e um PC responsável pelo controlo dos outros componentes e gestão do sistema. O diagrama de blocos do sistema de controlo de acessos é apresentado na Figura 1.



Figura 1 – Sistema de controlo de acessos (*Access Control System*)

Sobre o sistema podem-se realizar as seguintes ações em modo Acesso:

* **Acesso -** Para acesso às instalações, o utilizador deverá inserir os três dígitos correspondentes ao *UIN* seguido da inserção dos quatro dígitos numéricos do *PIN*. Se o par *UIN* e *PIN* estiver correto o sistema apresenta no LCD o nome do utilizador e a mensagem armazenada no sistema se existir, acionando a abertura da porta. A mensagem é removida do sistema caso seja premida a tecla ‘\*’ durante a apresentação desta. Todas os acessos deverão ser registados com a informação de data/hora e *UIN* num ficheiro de registos (um registo de entrada por linha), designado por *Log File*.
* **Alteração do PIN –** Esta ação é realizada se após o processo de autenticação for premida a tecla ‘#’. O sistema solicita ao utilizador o novo *PIN*, este deverá ser novamente introduzido de modo a ser confirmado. O novo *PIN* só é registado no sistema se as duas inserções forem idênticas.

**Nota:** A inserção de informação através do teclado tem o seguinte critério: se não for premida nenhuma tecla num intervalo de cinco segundos, o comando em curso é abortado; se for premida a tecla ‘\*’ e o sistema contiver dígitos, elimina todos os dígitos, se não contiver dígitos, aborta o comando em curso.

Sobre o sistema, podem-se realizar também as seguintes ações em modo Manutenção. Ao contrário das ações em modo Acesso, as ações em modo Manutenção são realizadas através do teclado e ecrã do PC. As ações disponíveis neste modo são:

* **Inserção de utilizador -** Tem como objetivo inserir um novo utilizador no sistema. O sistema atribui o primeiro *UIN* disponível, e espera que seja introduzido pelo gestor do sistema o nome e o *PIN* do utilizador. O nome tem no máximo 16 caracteres.
* **Remoção de utilizador -** Tem como objetivo remover um utilizador do sistema. O sistema espera que o gestor do sistema introduza o *UIN* e pede confirmação depois de apresentar o nome.
* **Inserir mensagem -** Permite associar uma mensagem de informação dirigida a um utilizador específico a ser exibida ao utilizador no processo de autenticação de acesso às instalações.
* **Desligar –** Permite desligar o sistema de controlo de acessos. Este termina após a confirmação do utilizador e reescreve o ficheiro com a informação dos utilizadores. Esta informação deverá ser armazenada num ficheiro de texto (com um utilizador por linha) que é carregado no início do programa e reescrito no final do programa. O sistema armazena até 1000 utilizadores, que são inseridos e suprimidos através do teclado do PC pelo gestor do sistema.

**Nota:** Durante a execução das ações em modo manutenção, não podem ser realizadas ações no teclado do utilizador e no LCD deve constar a mensagem “*Out of Service*”.

# Arquitetura do sistema

O controlo (designado por *Control*) do sistema de acessos será implementado numa solução híbrida de *hardware* e *software*, como apresentado no diagrama de blocos da Figura 2. A arquitetura proposta é constituída por quatro módulos principais: *i*) um leitor de teclado, designado por *Keyboard Reader*; *ii*) um módulo de interface com o *LCD*, designado por *Serial LCD Controller* (*SLCDC*); *iii*) um módulo de interface com o mecanismo da porta (*Door Mechanism)*, designado por *Serial Door Controller* (S*DC*); e *iv*) um módulo de controlo, designado por *Control*. Os módulos *i*), *ii*) e *iii*) deverão ser implementados em *hardware* e o módulo de controlo deverá ser implementado em *software* a executar num PC.

****

Figura 2 – Arquitetura do sistema que implementa o Sistema de Controlo de Acessos (*Access Control System*)

O módulo *Keyboard Reader* é responsável pela descodificação do teclado matricial de 12 teclas, determinando qual a tecla pressionada e disponibilizando o código desta em quatro bits ao *Control*, caso este esteja disponível para o receber. Caso este não esteja disponível para o receber imediatamente, o código da tecla é armazenado até ao limite de nove códigos. O *Control* processa e envia para o *SLCDC* a informação contendo os dados a apresentar no *LCD*. A informação para o mecanismo da porta é enviada através do *SDC*. Por razões de ordem física, e por forma a minimizar o número de sinais de interligação, a comunicação entre o módulo *Control* e os módulos *SLCDC e SDC* é realizada através de um protocolo série.



Figura 3 – Diagrama lógico do Sistema de Controlo de Acessos (*Access Control System*)

1. Interligações entre o HW e SW
2. Código *Kotlin* - *HAL*

**import isel.leic.UsbPort**

// Virtualiza o acesso ao sistema UsbPort

**object HAL {**

**private var lastWriting = 0**

// Inicia a classe

**fun init() {**

UsbPort.write(lastWriting)

**}**

// Retorna true se o bit tiver o valor lógico ‘1’

**fun isBit(mask: Int): Boolean {**

val temp = mask and UsbPort.read()

return mask == temp

**}**

// Retorna os valores dos bits representados por mask presentes no UsbPort

**fun readBits(mask: Int): Int** = mask and UsbPort.read()

// Escreve nos bits representados por mask o valor de value

**fun writeBits(mask: Int, value: Int) {**

val a = mask and value

val b = mask.inv() and lastWriting

val c = a or b

UsbPort.write(c)

lastWriting = c

**}**

// Coloca os bits representados por mask no valor lógico ‘1’

**fun setBits(mask: Int) {**

writeBits(mask,0xFF)

**}**

// Coloca os bits representados por mask no valor lógico ‘0’

**fun clearBits(mask:Int) {**

writeBits(mask,0x00)

**}**

**}**

1. Código *Kotlin* - *KBD*

**import isel.leic.utils.Time**

**object KBD {** // Ler teclas. Métodos retornam ‘0’..’9’,’#’,’\*’ ou NONE.

**private const val NONE = 0**

**private val arrayTeclas = arrayListOf('1', '4', '7', '\*', '2', '5', '8', '0', '3', '6', '9', '#')**

// Inicia a classe

**fun init() {**

HAL.init()

HAL.clearBits(ACK\_MASK)

**}**

// Retorna de imediato a tecla premida ou NONE se não há tecla premida.

**fun getKey(): Char {**

if (!HAL.isBit(DVAL\_MASK)) {

return NONE.toChar()

}

val key = HAL.readBits(D3\_\_0\_MASK)

while (HAL.isBit(DVAL\_MASK)) { HAL.setBits(ACK\_MASK) }

HAL.clearBits(ACK\_MASK)

return arrayTeclas[key]

**}**

// Retorna a tecla premida, caso ocorra antes do ‘timeout’ (representado em milissegundos), ou NONE caso contrário.

**fun waitKey(timeout: Long): Char {**

var key = NONE.toChar()

val endTime = Time.getTimeInMillis() + timeout

while (endTime >= Time.getTimeInMillis()) {

key = getKey()

if (key != NONE.toChar()) { break }

}

return key

**}**

**}**

1. Código *Kotlin* – *SerialEmitter*
2. Código *Kotlin* - *LCD*
3. Código *Kotlin* - *DoorMechanism*
4. Código *Kotlin* - *TUI*
5. Código *Kotlin* - *FileAccess*
6. Código *Kotlin* - *Users*
7. Código *Kotlin* - *Log*
8. Código *Kotlin* da classe *M*
9. Código *Kotlin* – *Access Control System* *- App*