

# Laboratório de à Arquitetura de Computadores

IST - LEIC

2021/2022

## Interação do processador com periféricos

### Guião 3

23 a 27 de maio de 2022

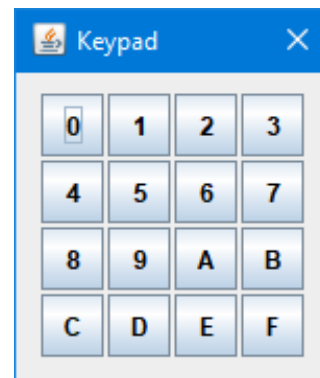
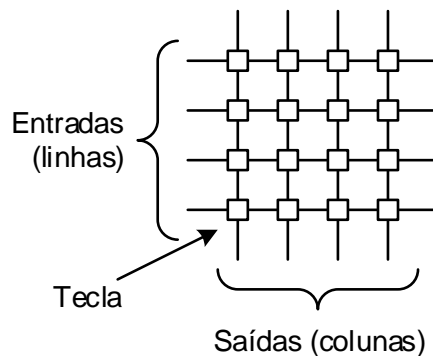
(Semana 3)

#### 1 – Objetivos

Com este trabalho pretende-se que os alunos pratiquem a utilização do microprocessador PEPE (Processador Especial Para Ensino) para acesso a periféricos (um teclado e o MediaCenter).

#### 2 – Funcionamento de um teclado

O teclado de 16 teclas a usar neste guião está organizado internamente como uma matriz de 4 linhas por 4 colunas. Neste caso as teclas disponíveis são os números 0 a 9 e ainda as letras de A a F. O que se pretende fazer do ponto de vista do microprocessador é saber qual a tecla que foi premida (com o cursor em cima da tecla e fazendo clique, simulando o carregar na tecla com um dedo).



O teclado não indica diretamente qual a tecla que foi premida. Para descobrir qual foi, o programa tem de testar sucessivamente as várias linhas do teclado, para ver se alguma tecla foi premida. Quando tal acontecer, a tecla faz um curto-circuito entre a linha e a coluna que a definem, fazendo com que o bit introduzido na linha (seja 0, seja 1) apareça na coluna (saída). Coluna a que nenhuma linha ligue vale 0 no bit de saída.

Aplica-se sucessivamente um 1 apenas a uma das linhas. Ou seja, colocam-se os valores “0001”, “0010”, “0100” e “1000” em formato binário nas entradas (linhas), lendo as saídas do teclado (colunas) em cada caso.

Com o valor 0001b aplicado nas linhas, se se obtiver nas colunas o valor 0001b, quer dizer que a tecla que foi premida foi a tecla “0”. Se o valor for 0010b, quer dizer que foi a tecla “1” que foi premida. Se for 0100b ou 1000b, a tecla premida terá sido “2” ou “3”, respetivamente.

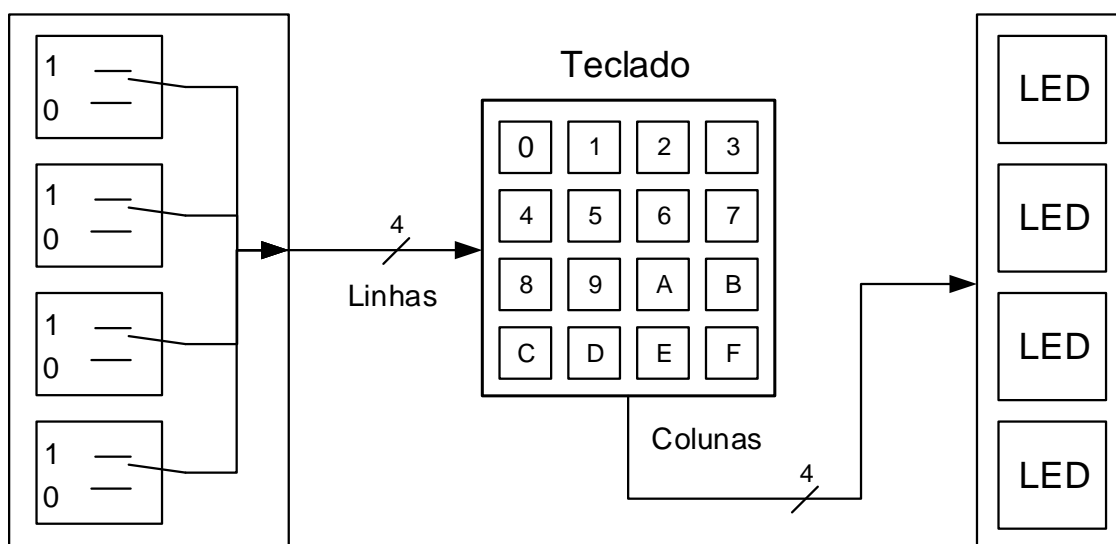
Com o valor 0010b aplicado nas linhas (o 1 está agora na segunda linha), é possível detetar se uma das teclas “4”, “5”, “6” e “7” foi premida, se se ler das colunas o valor 0001b, 0010b, 0100b ou 1000b, respetivamente.

Raciocínio idêntico aplica-se às teclas da 3ª e 4ª linhas do teclado.

Desta forma, fazendo um ciclo de “varrimento” às quatro linhas consegue-se detetar qualquer tecla em qualquer linha.

### 3 – Teste do teclado


Primeiro vamos testar e verificar o funcionamento do teclado com um circuito muito simples, descrito pela figura seguinte. Um conjunto de 4 interruptores permite injetar valores nas linhas do teclado (entradas) e um conjunto de 4 leds permite ver o estado das colunas (saídas).

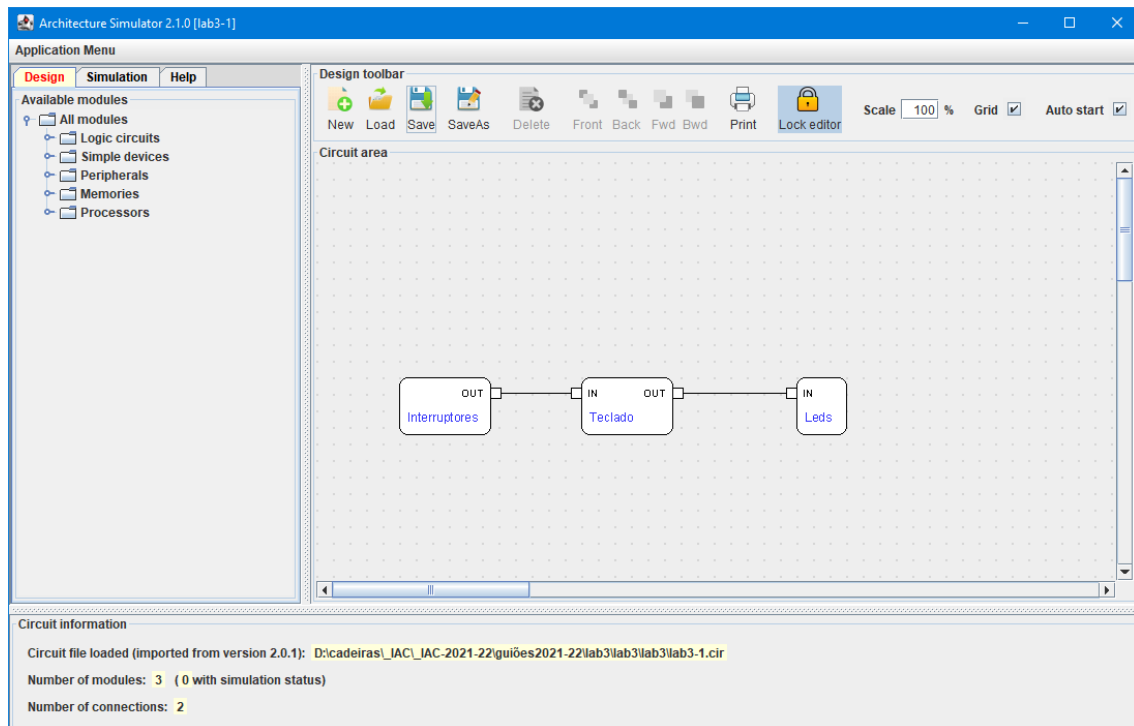


Um led liga apenas quando uma tecla é carregada e o bit injetado na linha dessa tecla está a 1.

O circuito que implementa este diagrama é dado já montado (ficheiro **lab\_teclado.cir**).

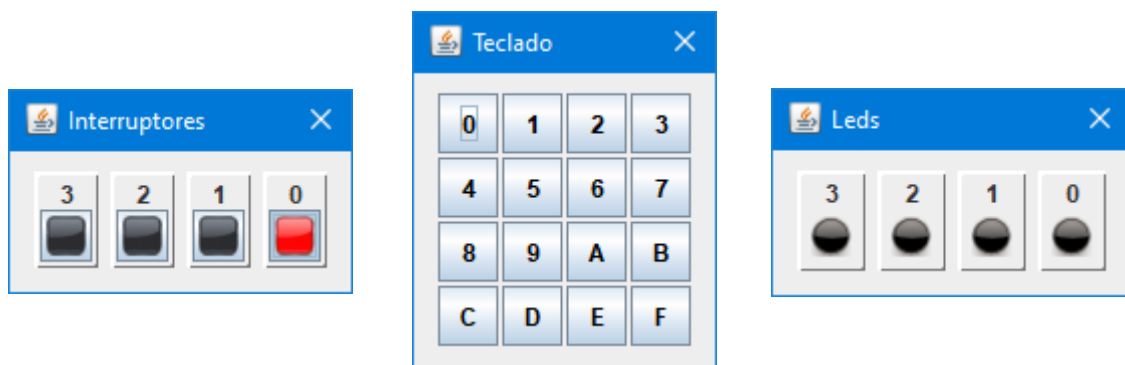
Faça o seguinte:

- Carregue este circuito no simulador (ou pelo botão **Load source**, , ou por *drag & drop*);
- Passe para “Simulation”;
- Abra as janelas de controlo de cada um destes dispositivos (com clique).



Experimente colocar o valor 0001b nos interruptores, fazendo clique no bit 0 dos interruptores.

As janelas de controlo dos dispositivos deverão ter o seguinte aspeto:



Carregue agora nas várias teclas do teclado e verifique que quando carrega numa tecla da primeira linha o led respetivo se liga. Nas teclas das outras linhas, nada acontece.

Note que:

- Só as teclas da primeira linha são detetadas, pois só a primeira linha tem um 1;
- A tecla 0 está à esquerda, enquanto o bit de ordem 0 nas colunas está à direita. É apenas efeito visual. O que interessa é a ordem dos bits lidos das colunas;

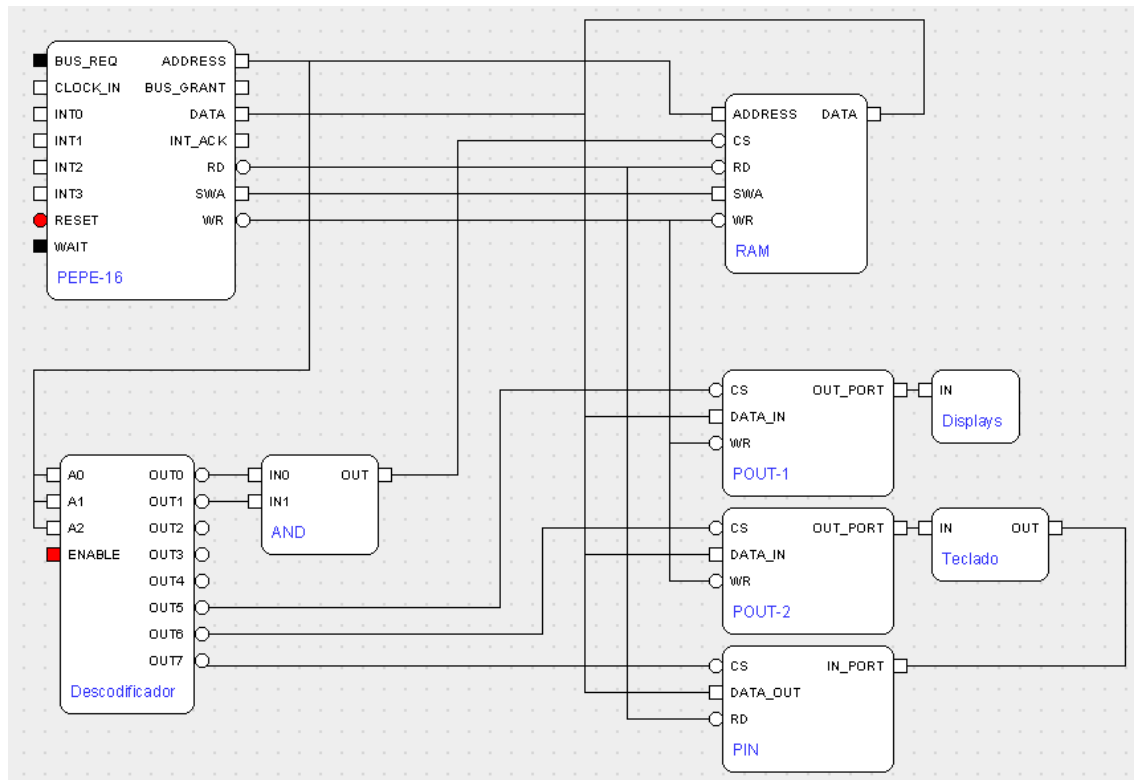
Agora experimente sucessivamente com os valores 0010b, 0100b e 1000b nos interruptores, e verifique que teclas consegue detetar em cada caso. Por fim, experimente com o valor 1111b nos interruptores, clique nas várias teclas de uma mesma coluna e verifique que assim não se consegue distingui-las.

Conclusão: para detetar uma tecla qualquer tem de se testar as diversas linhas, uma de cada vez.

## 4 – Leitura do teclado e escrita nos displays com o PEPE

### 4.1 – Circuito

O circuito para leitura do teclado com um programa é fornecido já montado (ficheiro lab\_pepe\_tec.cir).



São necessários os seguintes módulos:

- **PEPE** (cujo relógio é gerado internamente);
- **RAM** (uma memória de 16 bits de dados, 14 bits de endereço e com capacidade de endereçamento de byte);
- **POUT-1** (periférico de saída – liga aos displays. Este circuito tem 2 dígitos hexadecimais);
- **POUT-2** (periférico de saída – liga às linhas do Teclado);
- **PIN** (periférico de entrada – liga às colunas do Teclado);
- **Teclado** (teclado de 4 linhas e 4 colunas);
- **Descodificador** e **AND** (para permitir aceder aos dispositivos. Neste momento não é relevante. A descodificação de endereços será tratada no guião de laboratório 8).

**NOTAS IMPORTANTES:**

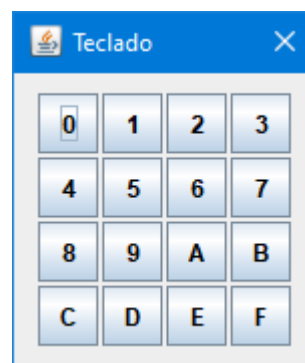
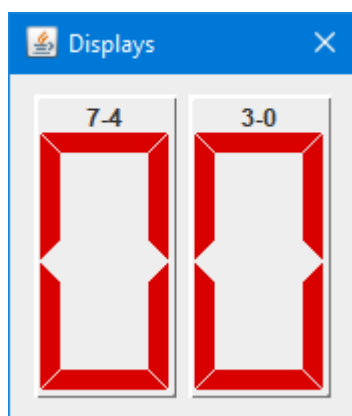
- Os acessos aos periféricos (quer em escrita quer em leitura) devem ser feitos com a instrução **MOV**, pois estes periféricos são de apenas 8 bits;
- Os acessos à memória podem ser feitos com **MOV** (16 bits) ou **MOV** (8 bits);
- A memória e periféricos estão disponíveis nos seguintes endereços:

| Dispositivo                            | Endereços     |
|----------------------------------------|---------------|
| RAM (memória)                          | 0000H a 3FFFH |
| POUT-1 (periférico de saída de 8 bits) | A000H         |
| POUT-2 (periférico de saída de 8 bits) | C000H         |
| PIN (periférico de entrada de 8 bits)  | E000H         |

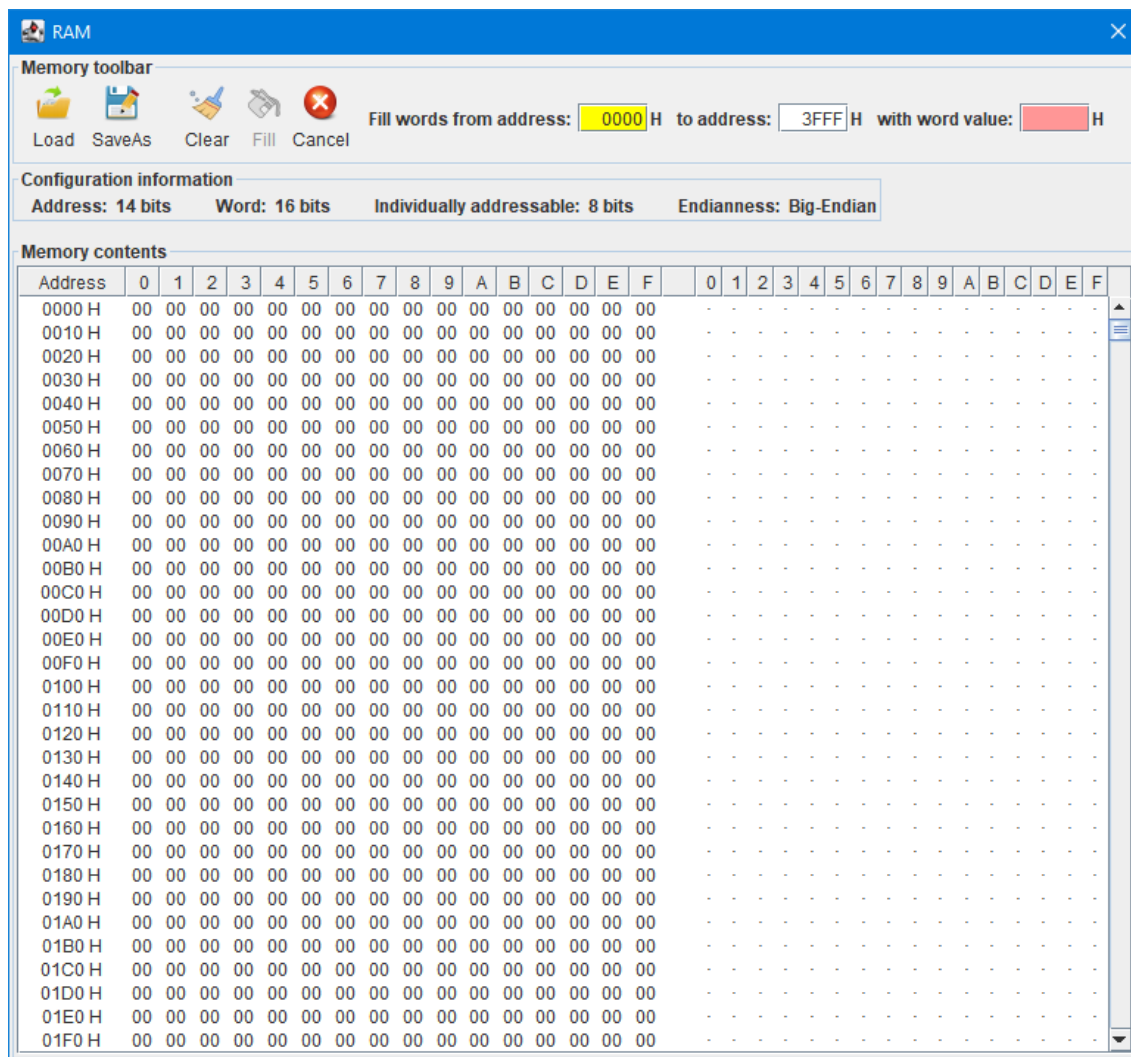
## 4.2 – Simulação do circuito

Passe o simulador para “Simulation”.

Faça clique nos displays e no teclado, para abrir os respetivos painéis, e desloque-os para uma zona do ecrã fora da janela do simulador.







Faça também clique na RAM e abra o respetivo painel, que tem o aspeto indicado na página seguinte. Verifique, fazendo scroll na barra do lado direito, que o endereço máximo é 3FFFH, tal como deve ser atendendo a que esta memória tem 14 bits de endereço ( $2^{14} = 4000H$ ).



### 4.3 – Programa de leitura do teclado e escrita nos displays

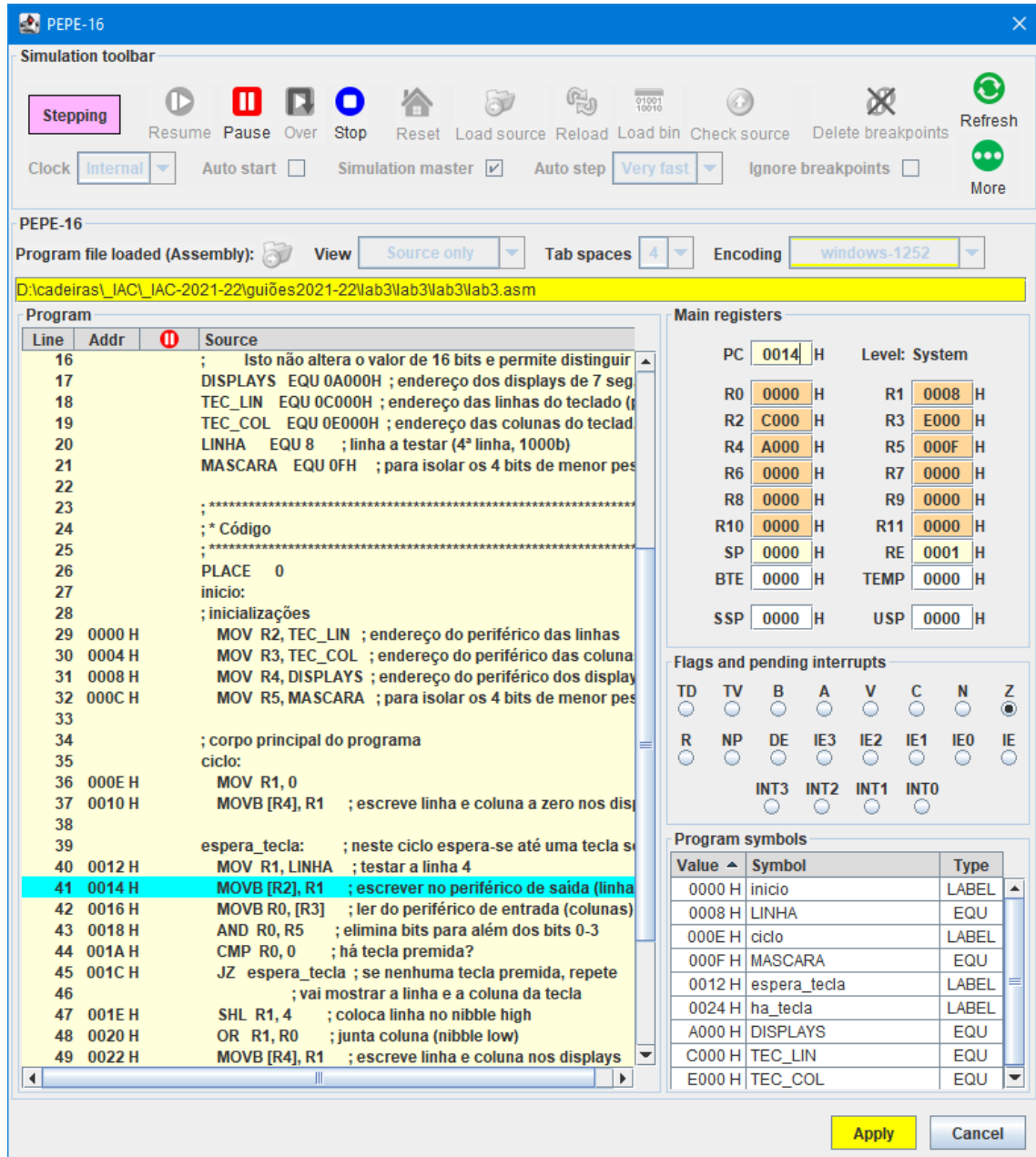
Abra o ficheiro **lab\_pepe\_tec.asm** (com um editor de texto do tipo NotePad++ para PC ou Brackets para Mac) e tente perceber o que faz este programa, pelos comentários e pelo funcionamento descrito na secção 2.

De seguida, execute os seguintes passos:


1. Abra o painel de controlo do PEPE (clique no módulo) e carregue o ficheiro **lab\_pepe\_tec.asm** (pelo botão **Load source**, , ou por *drag & drop*);
2. Execute o programa em modo contínuo, com o botão **Start** () , e carregue numa tecla da linha 4 (a de baixo). Verifique que os displays exibem a linha (8) e a coluna (1, 2, 4 ou 8) correspondentes à tecla em que carregou (ou seja, 81, 82, 84 ou 88);
3. Verifique também que estes valores são exibidos apenas enquanto a tecla estiver a ser carregada. Verifique no programa o que é que permite ter esta distinção de comportamento, consoante a tecla está carregada ou não;
4. Verifique que em qualquer altura pode fazer pausa à execução do programa, carregando no botão **Pause** () do PEPE. A partir daí pode então executar uma instrução de cada vez, ou passo a passo, carregando no botão **Step** () do

PEPE. Em qualquer altura, pode voltar a executar em modo contínuo, carregando no botão **Resume** (▶).

- Note que não se consegue detetar teclas carregadas em execução passo a passo, pois o cursor do rato é só um (e ou se carrega no botão **Step** (▶) ou no teclado). A funcionalidade de auto-step permite resolver este dilema. Selecione a velocidade mais rápida (“Very fast”) e carregue no botão **Step** (▶):



- Verifique que a barra azul circula entre o *label* “espera\_tecla” e pouco mais abaixo, mas que quando carrega no teclado numa tecla da última linha o programa vai até à última instrução (tem de deixar a tecla carregada tempo suficiente para o PEPE ler o teclado na instrução **MOVB R0, [R3]**);
- NOTA – O AND com R5 (0FH) destina-se a forçar a 0 os bits 7 a 4 lidos do periférico de entrada, pois estes estão “no ar” (uma vez que o teclado só liga aos bits 3 a 0) e dão valores aleatórios;
- Termine a execução do programa, carregando no botão **Stop** (●) do PEPE;

9. Altere o valor da constante **LINHA** (no ficheiro **lab\_pepe\_tec.asm**) para 1, 2 ou 4. Guarde o ficheiro (no editor de texto), faça **Reload** () e volte ao passo 2 acima, verificando que agora o programa deteta teclas noutra linha (indicada pela constante **LINHA**).

## 5 – Versão intermédia do Projeto

O programa do ficheiro **lab\_pepe\_tec.asm** dá apenas para uma tecla numa linha. Pretende-se detetar qualquer tecla em qualquer linha. O teclado é um componente essencial para a execução do projeto, cuja **versão intermédia deve ser entregue já dia 1 de junho (até às 23h59)**.

Como parte desta entrega (verifique no enunciado do projeto todos os requisitos da versão intermédia), deve fazer um programa que faça o varrimento (em ciclo) das quatro linhas do teclado e detete a tecla premida, obtendo o seu valor (0 a FH), em vez de apenas os valores separados da linha e da coluna.

Para tal, deve fazer os passos seguintes:

- Faça um programa (pode usar o programa do ficheiro **lab\_pepe\_tec.asm** como base) que varre continuamente, em ciclo, as várias linhas do teclado, verificando se alguma das teclas foi premida e, quando tal acontecer, sai do ciclo com a linha e a coluna da tecla em dois registos (use *breakpoints* e/ou a funcionalidade de auto-step para verificar se os valores obtidos estão certos);
- Na realidade, o que quer obter é um valor entre 0 e FH, correspondente ao valor da tecla carregada, em vez de apenas informação separada sobre a linha e coluna da tecla. Acrescente código para converter esta informação no valor entre 0 e FH. Sugestão: o valor da tecla é igual a  $4 * \text{linha} + \text{coluna}$ , em que tanto linha como coluna são números entre 0 e 3. Logo, terá de converter 0001b, 0010b, 0100b e 1000b (isto é, 1, 2, 4 e 8, valores possíveis quer para a linha, quer para a coluna) em 0, 1, 2 e 3, respetivamente. Tal pode ser feito contando o número de **SHR** (deslocamentos à direita de 1 bit) que se tem de fazer ao valor da linha (ou da coluna) até este ser zero (use um *breakpoint* para verificar se o valor obtido está certo, e se não estiver reinicie o programa e corra-o em *single-step* desde o *breakpoint* do ponto anterior, verificando os registos relevantes em cada passo);
- Acrescente código para mostrar o valor da última tecla carregada nos displays, em ciclo. Sempre que uma tecla é detetada e o seu valor mostrado, o programa deve voltar a esperar a deteção de uma nova tecla;
- Mas, mesmo realmente, o que quer é usar as teclas detetadas para executar funcionalidades no projeto. Simule esta capacidade fazendo com que os displays mostrem agora o valor de um contador hexadecimal (valor de um registo), que é incrementado quando se carrega numa dada tecla (escolha qual) e decrementado quando se carrega numa outra tecla (escolha qual). **Por cada tecla carregada, o contador só deve variar uma vez.**

A tabela A.9 do livro contém informação sobre cada instrução do PEPE.

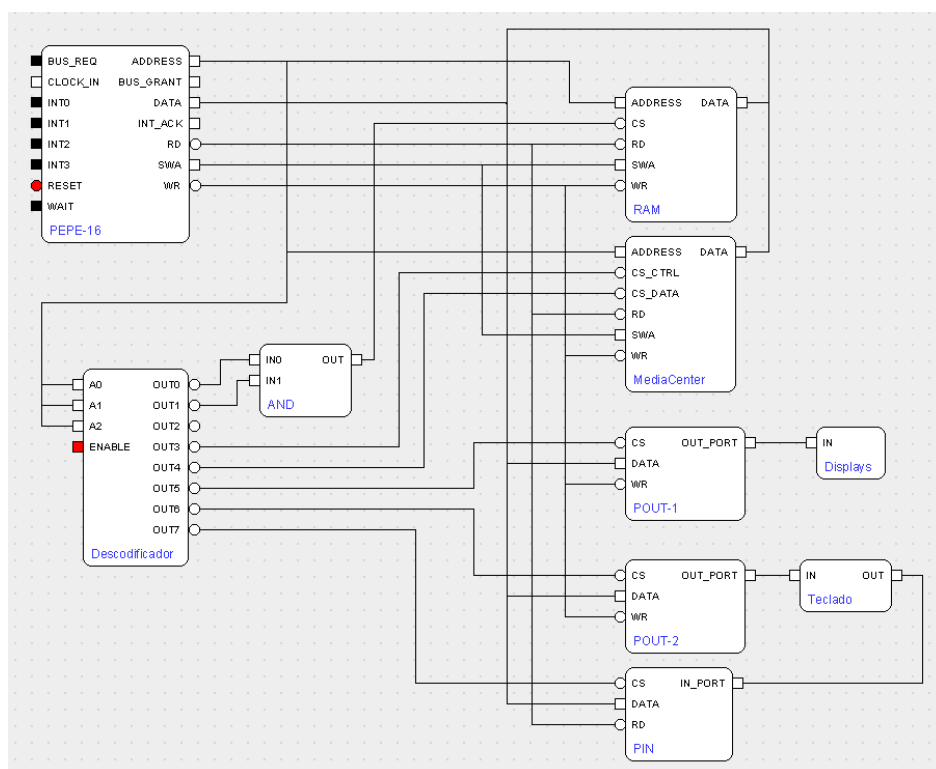
O objetivo deste exercício é fazer o software básico de leitura do teclado, que constitui o coração do controlo do programa do projeto. Será apenas questão de, mais tarde, adaptar o que se faz com o valor da tecla lida.



## 6 – O módulo MediaCenter

### 6.1 – Organização interna

Clique em **Load** (📁) e carregue o circuito contido no ficheiro **lab\_media.cir**. Este circuito é semelhante ao já usado no Guião 3, mas tem agora mais um periférico (MediaCenter, um ecrã gráfico de 32 x 64 pixels, com suporte para áudio e vídeo), e constitui mais um passo em direção ao circuito do projeto, permitindo desenhar objetos gráficos, pixel a pixel.



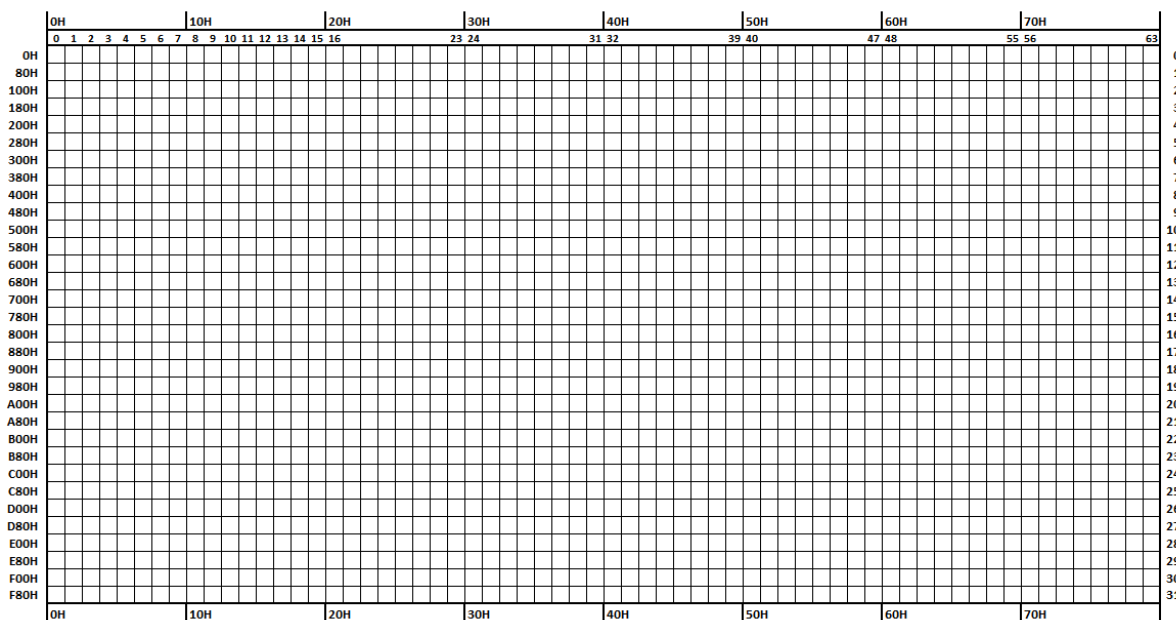
Cada pixel pode ter uma cor diferente, em 4 componentes (*Alpha*, *Red*, *Green* e *Blue*, ou ARGB), todas com 4 bits (valores não negativos, de 0 a 15), pois o PEPE tem 16 bits. A componente *Alpha* define a opacidade (0 é totalmente transparente, 15 totalmente opaco). O pixel pode estar ligado (com a cor definida pelas 4 componentes) ou desligado (todas as componentes a 0, caso em que não se vê).

As coordenadas de um dado pixel no ecrã são dadas em linha (0 a 31) e coluna (0 a 63). É possível alterar a cor de um dado pixel por dois métodos diferentes:

- Enviar comandos específicos para o MediaCenter, para selecionar a linha, a coluna e a cor do pixel (normalmente, o método mais fácil, pois não tem de se preocupar com o endereço de cada pixel);
- Dado que o MediaCenter guarda os pixels numa memória de  $32 \times 64 = 2048$  palavras (cada pixel ocupa 16 bits, ou uma palavra do PEPE), converter a linha e coluna para o endereço interno em que o pixel se encontra nesta memória, somar-lhe o endereço de base do MediaCenter e fazer um acesso de escrita normal (como se de uma RAM se tratasse), escrevendo diretamente a cor do pixel (incluindo 0000H, para o apagar).

A figura seguinte representa a memória do MediaCenter, com 4096 bytes, ou 2048 palavras. Cada quadrado é um pixel, que ocupa 2 bytes, ou 1 palavra:

- No topo, está indicado o endereço relativo (dentro de cada linha) da primeira palavra de cada 8 pixels e a ordem dos 64 pixels de cada coluna (nem todos, para ser mais simples). Note que os endereços das palavras são sempre pares;
- No lado esquerdo, o endereço relativo (dentro da memória do MediaCenter) da primeira palavra de cada linha;
- No lado direito, a ordem dos 32 pixels de cada linha (cresce de cima para baixo).



Desta forma, há uma correspondência direta entre a imagem do MediaCenter onde aparecem os pixels e a memória cujas palavras indicam a cor de cada pixel. Os endereços das palavras variam entre 000H e FFEH (sempre de 2 em 2, pois têm de ser pares). Isto dá 4096 endereços, ou 2048 palavras.

Estes endereços são internos ao MediaCenter. No entanto, o PEPE “vê” a memória interna do MediaCenter a partir do endereço 8000H (determinado pelo sistema de descodificação de endereços deste circuito), pelo que estes endereços, do ponto de vista do PEPE, correspondem ao intervalo 8000H a 8FFFH (endereço do último pixel: 8FFEH).

Tal como a RAM do sistema, a memória do MediaCenter pode ser acedida em *byte* (com **MOVB**) ou em *word* (com **MOV**). No entanto, é mais simples lidar com uma palavra de cada vez, com **MOV**, pois assim lida-se logo com um pixel inteiro.

Em vez de escrever diretamente na memória dos pixels, o PEPE pode dar comandos ao MediaCenter, escrevendo em endereços a partir de 6000H, da forma explicada mais adiante. O MediaCenter tem dois *chip selects*, um para a memória de pixels e outra para os comandos.

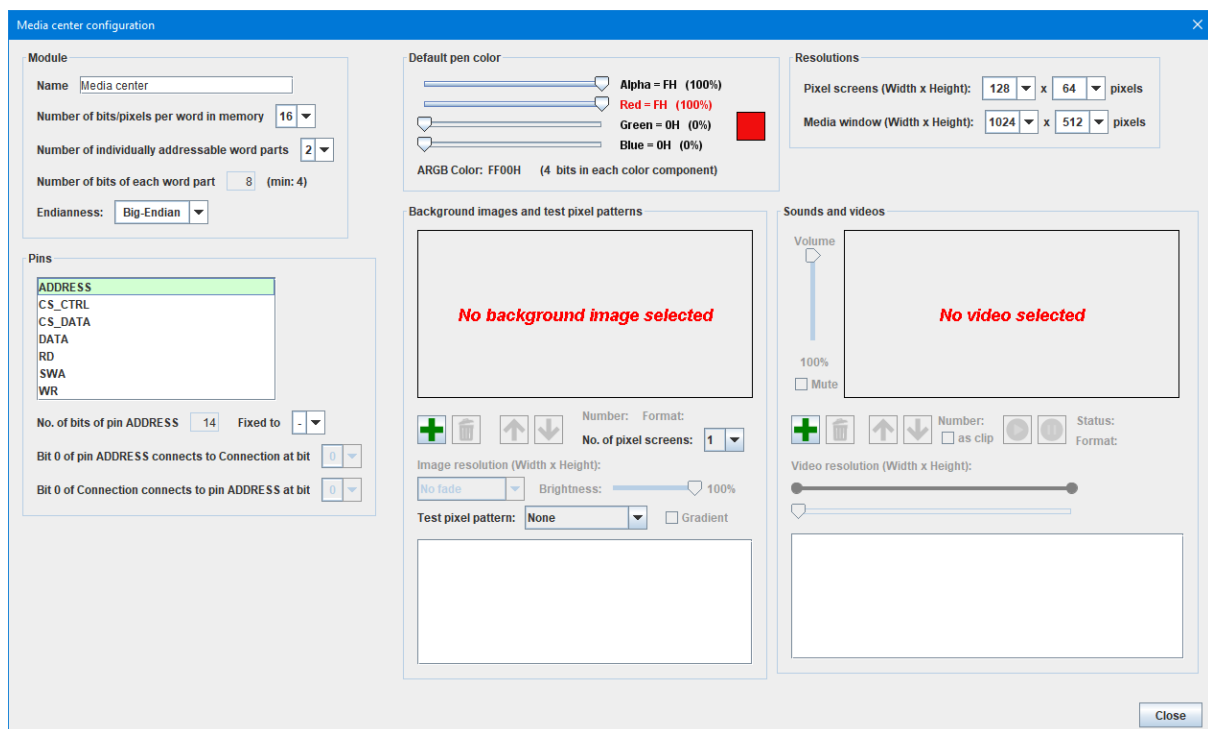
Desta forma, o mapa de endereços (em que os dispositivos podem ser acedidos pelo PEPE) disponível neste circuito é o seguinte:

| Dispositivo                       | Endereços                      |
|-----------------------------------|--------------------------------|
| RAM                               | 0000H a 3FFFH                  |
| MediaCenter (acesso aos comandos) | 6000H a 6063H (ver secção 2.3) |

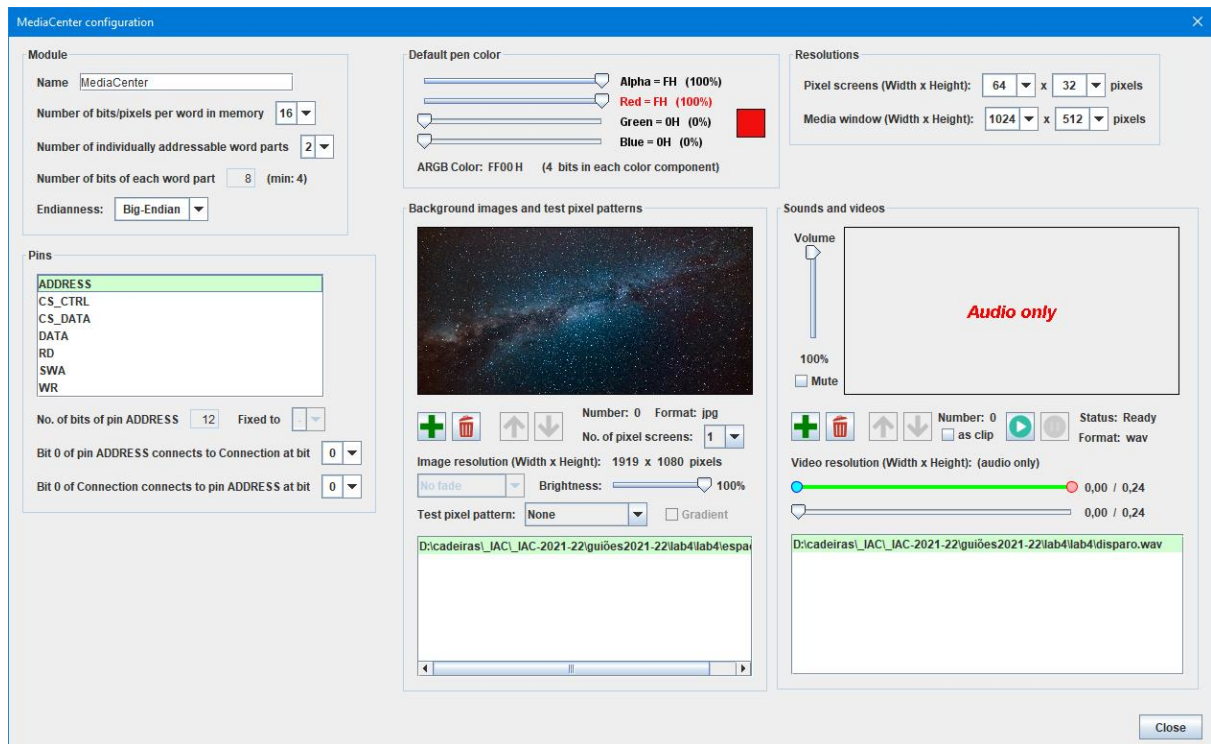
|                                        |               |
|----------------------------------------|---------------|
| MediaCenter (acesso à sua memória)     | 8000H a 8FFFH |
| POUT-1 (periférico de saída de 8 bits) | 0A000H        |
| POUT-2 (periférico de saída de 8 bits) | 0C000H        |
| PIN (periférico de entrada de 8 bits)  | 0E000H        |

## 6.2 – Configuração

A figura seguinte ilustra a interface de configuração do MediaCenter (accedida com clique duplo no MediaCenter em modo “Design”).



Esta é a visão do MediaCenter sem nada definido. No entanto, ele já é fornecido neste guião com uma imagem (**espaço.jpg**) e um som (**disparo.wav**). Pode também acrescentar outras imagens, sons e até vídeos.



Para além da configuração dos pinos, já usada noutros módulos, é possível configurar:

- A cor por omissão da caneta, incluindo opacidade (*Alpha*), com que os pixels serão desenhados no ecrã, em formato ARGB (*Alpha*, *Red*, *Green* e *Blue*);
- A resolução do ecrã de simulação, aquele em que o PEPE escreve os pixels. Na realidade, é possível definir até 16 ecrãs de pixels, suportando sobreposição de objetos diferentes sem interferência;
- A resolução dos cenários de fundo, imagens/vídeos definidos em ficheiros que, se definidos, aparecem por trás dos pixels desligados (que são 100% transparentes). Esta resolução é geralmente bastante maior que a anterior, mas a imagem de menor dimensão é esticada para se sobrepor à outra;
- Os ficheiros com imagens de cenário, havendo um painel para a pré-visualização. As extensões de imagem suportadas são as seguintes: "jpg", "png", "bmp" e "gif". No entanto, outras poderão ser suportadas, dependendo da plataforma. Podem ser adicionadas (carregando no "+" da categoria "Background images and test pixel patterns", apagadas ou mudadas de ordem). Para serem guardadas de forma portátil no ficheiro do circuito, de modo a funcionarem noutro computador, os ficheiros de imagem devem estar no mesmo diretório (ou subdiretório dele) que o ficheiro do circuito (caso contrário, é guardado o *path* absoluto de cada ficheiro);
- Padrões de teste, que sobrepõem à imagem para se ter uma ideia de como ficam os objetos em cima das imagens), e até padrões de *fading* ao mudar de cenário ou vídeo;
- Os ficheiros com sons e vídeos. As extensões de áudio e vídeo suportadas são as seguintes: "aif", "aiff", "mp3", "wav", "mp4", "m4a" e "m4v". Podem ser adicionados (carregando no "+" da categoria "Sounds and videos", apagados ou mudados de ordem). Para serem guardados de forma portátil no ficheiro do

circuito, de modo a funcionarem noutro computador, os ficheiros de som devem estar no mesmo diretório (ou subdiretório dele) que o ficheiro do circuito (caso contrário, é guardado o *path* absoluto de cada ficheiro). Quando houver pelo menos um ficheiro, definido, há um botão para se poder tocar e é possível ver o seu formato e duração, reduzir o seu volume de som e até definir os pontos de início e fim de reprodução.

### 6.3 – Controlo do MediaCenter por comandos

Uma forma de escrever pixels no ecrã do MediaCenter é usando a sua interface de comandos. Para alterar algo no ecrã (a cor de um pixel, seleccionar uma linha ou uma coluna, etc.), o PEPE faz uma escrita num endereço a partir do 6000H, com um valor que corresponde ao argumento do comando. Para ler informação do ecrã (cor de um pixel, que linha está atualmente seleccionada, etc.), o PEPE faz uma leitura de um endereço a partir do 6000H. O valor lido é a informação pretendida.

Portanto, os endereços usados identificam os comandos e os valores (escritos ou lidos) indicam a nova informação, para alterar no ecrã, ou a informação lida do ecrã. Um dado endereço pode ter significados diferentes, consoante seja usado num acesso de escrita ou leitura. Estes acessos pelo PEPE não acedem realmente a uma memória, apenas executam comandos.

**IMPORTANTE** – Porque alguns comandos podem envolver valores que não cabem num byte, todos os comandos são pares. Para usar os comandos deve-se usar instruções de **MOV** (acesso em *word*), somando 6000H ao comando.

Os comandos de escrita atualmente disponíveis no MediaCenter são os seguintes:

| Endereço | Descrição                                                                                                                                      | Valor a escrever                                 |
|----------|------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------|
| 6000H    | Apaga todos os pixels do ecrã especificado                                                                                                     | 0 .. n° ecrãs-1                                  |
| 6002H    | Apaga todos os pixels de todos os ecrãs                                                                                                        | ---                                              |
| 6004H    | Seleciona o ecrã especificado                                                                                                                  | 0 .. n° ecrãs-1                                  |
| 6006H    | Mostra o ecrã especificado                                                                                                                     | 0 .. n° ecrãs-1                                  |
| 6008H    | Esconde o ecrã especificado                                                                                                                    | 0 .. n° ecrãs-1                                  |
| 600AH    | Especifica a linha a usar no próximo comando                                                                                                   | 0 .. n.º linhas-1                                |
| 600CH    | Especifica a coluna a usar no próximo comando                                                                                                  | 0 .. n.º colunas-1                               |
| 600EH    | Especifica o n° do pixel a usar no próximo comando                                                                                             | 0 .. n.º pixels-1                                |
| 6010H    | Especifica o auto-increment. Se ligado, os comandos que alteram a cor de pixels (com *) avançam automaticamente para o(s) pixel(s) seguinte(s) | 0 - desliga; 1 - liga<br>(desligado por omissão) |
| 6012H    | (*) Altera a cor do pixel na posição corrente                                                                                                  | Palavra com ARGB                                 |
| 6014H    | Especifica a cor da caneta                                                                                                                     | Palavra com ARGB                                 |
| 6016H    | (*) Altera a cor do pixel indicado para a cor da caneta                                                                                        | 0 .. n.º pixels-1                                |
| 6018H    | (*) Apaga o pixel indicado                                                                                                                     | 0 .. n.º pixels-1                                |
| 601AH    | (*) Altera a cor do pixel na posição corrente                                                                                                  | 0 - apaga; 1 - caneta                            |
| 601CH    | (*) Altera a cor de 8 pixels, a partir da maior posição múltipla de 8 não é maior que a posição corrente                                       | Cada bit:<br>0 - apaga; 1 - caneta               |
| 601EH    | (*) Altera a cor de 16 pixels, a partir da maior posição múltipla de 16 não é maior que a posição corrente                                     | Cada bit:<br>0 - apaga; 1 - caneta               |
| 6020H    | Desenha um padrão (um dos 12 que se podem indicar na janela de configuração do MediaCenter)                                                    | 0 .. 11                                          |

|       |                                                                                                                         |                                                                |
|-------|-------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------|
| 6022H | Igual ao anterior, mas com gradiente na cor                                                                             | 0 .. 11                                                        |
| ---   | 6024H a 603EH reservados para expansão futura                                                                           | ---                                                            |
| 6040H | Apaga o cenário de fundo (e elimina o aviso: "No background image selected")                                            | ---                                                            |
| 6042H | Seleciona o n.º do cenário de fundo a visualizar                                                                        | 0 .. n.º imagens-1                                             |
| 6044H | Apaga o cenário frontal                                                                                                 | ---                                                            |
| 6046H | Seleciona o n.º do cenário frontal a visualizar                                                                         | 0 .. n.º imagens -1                                            |
| 6048H | Seleciona um som/vídeo para comandos seguintes                                                                          | 0 .. n.º sons/videos-1                                         |
| 604AH | Especifica o volume do som (percentagem) do som/vídeo selecionado                                                       | 0 .. 100                                                       |
| 604CH | Corta o volume ( <i>mute</i> ) do som/vídeo especificado                                                                | 0 .. n.º sons/videos-1                                         |
| 604EH | Retoma o volume do som/vídeo especificado                                                                               | 0 .. n.º sons/videos-1                                         |
| 6050H | Corta o volume ( <i>mute</i> ) de todos os sons/vídeos a reproduzir                                                     | ---                                                            |
| 6052H | Retoma o volume de todos os sons/vídeos a reproduzir                                                                    | ---                                                            |
| 6054H | Especifica o brilho dos vídeos e do cenário de fundo                                                                    | 0 .. 100                                                       |
| 6056H | Especifica um padrão de transição entre vídeos (um dos 5 que se podem indicar na janela de configuração do MediaCenter) | 0- no fading; 1 - very slow; 2 - slow; 3 - fast; 4 - very fast |
| 6058H | Especifica o número de vezes que um som/vídeo deve ser reproduzido                                                      | 0 – repetição até ser parado                                   |
| 605AH | Inicia a reprodução do som/vídeo especificado                                                                           | 0 .. n.º sons/videos-1                                         |
| 605CH | Reproduz o som/vídeo especificado em ciclo até ser parado                                                               | 0 .. n.º sons/videos-1                                         |
| 605EH | Pausa a reprodução do som/vídeo especificado                                                                            | 0 .. n.º sons/videos-1                                         |
| 6060H | Continua a reprodução do som/vídeo especificado                                                                         | 0 .. n.º sons/videos-1                                         |
| 6062H | Pausa a reprodução de todos os sons/vídeos a reproduzir                                                                 | ---                                                            |
| 6064H | Continua a reprodução de todos os sons/vídeos em pausa                                                                  | ---                                                            |
| 6066H | Termina a reprodução do som/vídeo especificado                                                                          | 0 .. n.º sons/videos-1                                         |
| 6068H | Termina a reprodução de todos os sons/vídeos a reproduzir                                                               | ---                                                            |

Os comandos de leitura atualmente disponíveis no MediaCenter são os seguintes:


| Endereço | Descrição                                                                                                                                           | Valor lido                 |
|----------|-----------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------|
| 6000H    | Obtém o n.º de colunas do ecrã (potência de 2)                                                                                                      | 0 .. 2048                  |
| 6002H    | Obtém o n.º de linhas do ecrã (potência de 2)                                                                                                       | 0 .. 1024                  |
| 6004H    | Obtém o ecrã atualmente selecionado                                                                                                                 | 0 .. n.º ecrãs-1           |
| 6006H    | Obtém a visibilidade do ecrã atualmente selecionado                                                                                                 | 0 - escondido; 1 - visível |
| 6008H    | Obtém a linha da posição corrente                                                                                                                   | 0 .. n.º linhas-1          |
| 600AH    | Obtém a coluna da posição corrente                                                                                                                  | 0 .. n.º colunas-1         |
| 600CH    | Obtém o n.º do pixel da posição corrente                                                                                                            | 0 .. n.º pixels-1          |
| 600EH    | Obtém o estado do auto-increment. Se ligado, os comandos que alteram a cor de pixels (com *) avançam automaticamente para o(s) pixel(s) seguinte(s) | 0 - desligado; 1 - ligado  |
| 6010H    | (*) Obtém a cor do pixel na posição corrente                                                                                                        | Palavra com ARGB           |
| 6012H    | Obtém a cor atual da caneta                                                                                                                         | Palavra com ARGB           |



|       |                                                                                                                          |                                                                         |
|-------|--------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------|
| 6014H | (*) Obtém o estado da cor do pixel corrente                                                                              | 0 - apagado; 1 – cor não zero                                           |
| 6016H | (*) Obtém o estado da cor de 8 pixels, a começar na maior posição múltipla de 8 que não é maior que a posição corrente   | Byte, com cada bit:<br>0 - apagado; 1 – cor não zero                    |
| 6018H | (*) Obtém o estado da cor de 16 pixels, a começar na maior posição múltipla de 16 que não é maior que a posição corrente | Palavra, com cada bit:<br>0 - apagado; 1 – cor não zero                 |
| ---   | 601AH a 603EH reservados para expansão futura                                                                            | ---                                                                     |
| 6040H | Obtém n.º de imagens definidas                                                                                           | 0 .. 32                                                                 |
| 6042H | Obtém o n.º do cenário de fundo selecionado                                                                              | -1 .. n.º imagens-1<br>(-1 se nenhum)                                   |
| 6044H | Obtém o n.º do cenário frontal selecionado                                                                               | -1 .. n.º imagens-1<br>(-1 se nenhum)                                   |
| 6046H | Obtém n.º de sons/vídeos definidos                                                                                       | 0 .. 32                                                                 |
| 6048H | Obtém o n.º do som/vídeo selecionado                                                                                     | -1 .. n.º sons/vídeos-1<br>(-1 se nenhum)                               |
| 604AH | Obtém o volume do som (percentagem) do som/vídeo selecionado                                                             | 0 .. 100                                                                |
| 604CH | Obtém o estado de <i>mute</i> do som/vídeo especificado                                                                  | 0 - normal; 1 - muted                                                   |
| 604EH | Obtém o valor do brilho (percentagem)                                                                                    | 0 .. 100                                                                |
| 6050H | Obtém o n.º do padrão de transição entre vídeos (um dos 5 que se podem indicar na janela de configuração do MediaCenter) | 0- no fading; 1 - very slow;<br>2 - slow; 3 - fast; 4 - very fast       |
| 6052H | Obtém o estado do som/vídeo selecionado                                                                                  | 0 - Unknown; 1 - Ready; 2 - Paused; 3 - Playing; 4 - Stopped; 5 - Error |
| 6054H | Obtém o n.º dos sons/vídeos a reproduzir                                                                                 | 0 .. n.º sons/vídeos-1                                                  |
| 6056H | Obtém o n.º de vezes que o som/vídeo selecionado será reproduzido                                                        | 0 - repetição até ser parado                                            |

#### 6.4 – Exemplos de desenho de pixels no MediaCenter (por comandos)

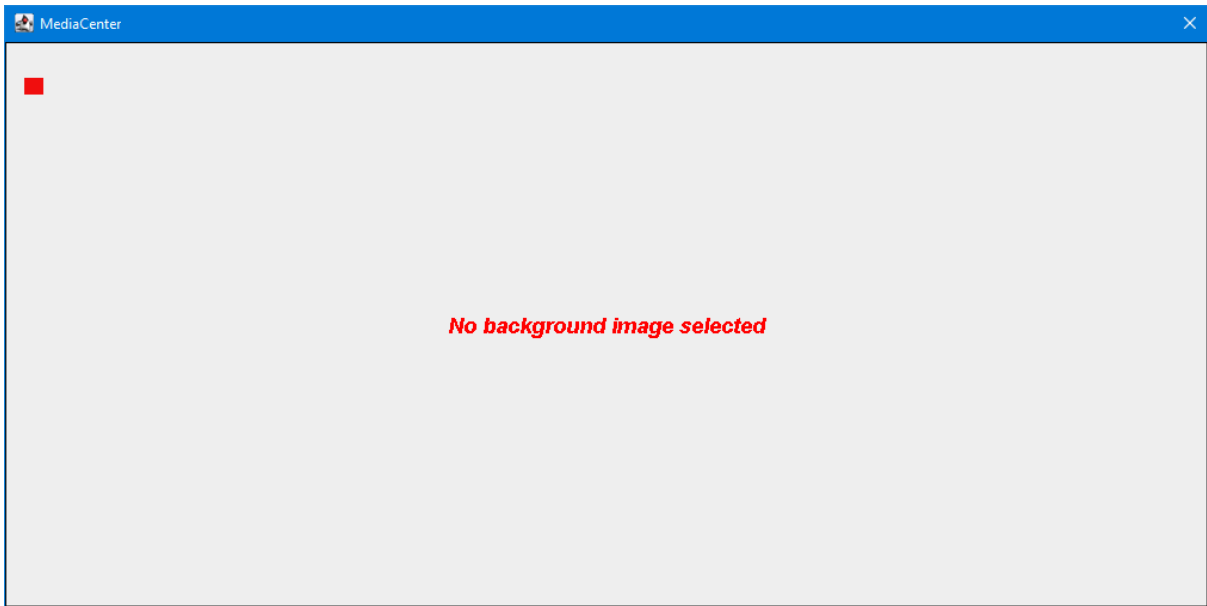
Para exemplificar o uso de comandos com o ecrã, edite e observe o programa contido no ficheiro **lab-comandos.asm**, que desenha um pixel numa dada linha e coluna e com uma dada cor.

Para apagar um pixel já desenhado, basta desenhá-lo de novo, na mesma linha e coluna, mas com cor 0.

Passe o simulador para “Simulation” e carregue o ficheiro **lab-comandos.asm** no PEPE (com o botão **Load source**, , ou por *drag & drop*).

Execute o programa passo a passo, com o botão **Step** () , SEM carregar em **Start** () , e verifique que o pixel da linha 2, coluna 1, se liga e que no fim desaparece.

Note que tanto as linhas como as colunas começam em 0 (canto superior esquerdo do ecrã).



Termine a execução do programa, carregando no botão **Stop** (■) do PEPE.

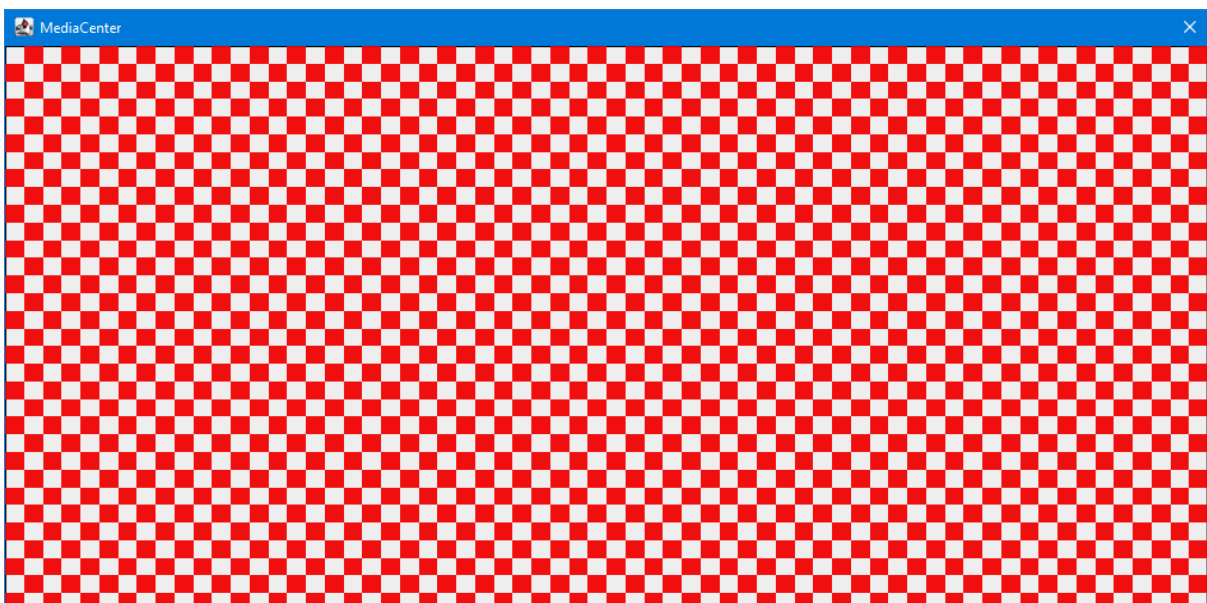
O aviso “No background image selected” existe para lembrar que os cenários de fundo têm de ser explicitamente selecionados, com o comando de escrita 6042H. No entanto, se quiser eliminar o aviso sem selecionar nenhum cenário, basta executar o comando de escrita 6040H. Isto é ilustrado nos dois exemplos seguintes.

Carregue agora o programa contido no ficheiro **lab-comandos-xadrez.asm**, que desenha um padrão de xadrez no ecrã.

Este programa escreve um pixel numa dada linha e coluna, tal como no programa anterior, mas agora dentro de dois ciclos:

- Um para percorrer todos os pixels de uma dada coluna;
- Outro para percorrer todas as linhas do ecrã.

De cada vez que escreve um pixel ou uma linha troca o valor do pixel (entre desligado e vermelho), e o resultado é um padrão em xadrez, tal como se pode ver na figura seguinte.

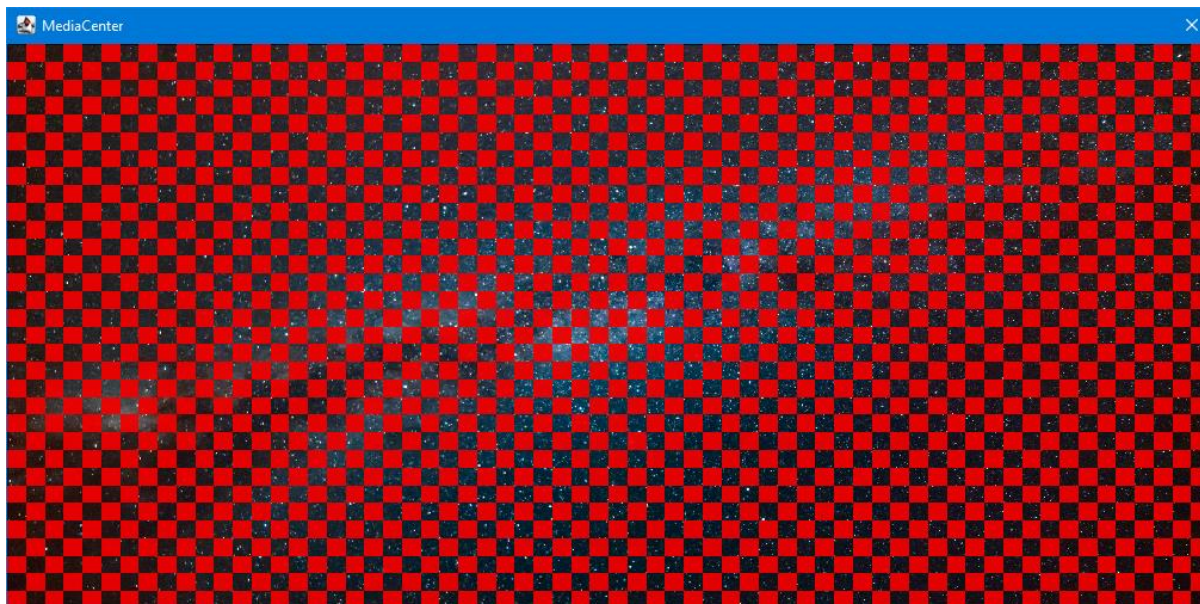




Este programa já apaga o aviso “No background image selected”, mas podemos mesmo seleccionar um cenário de fundo, nomeadamente usando a imagem **espaço.jpg**.

Carregue o programa contido no ficheiro **lab-xadrez-imagem.asm**. Este programa é igual ao anterior, exceto pelo facto de seleccionar um cenário de fundo sobre o qual o padrão de xadrez é desenhado.

Execute o programa e verifique que os pixels desligados são na realidade totalmente transparentes (pois a sua componente de opacidade está a 0).



Termine a execução do programa, carregando no botão **Stop** (■) do PEPE.

### 6.5 – Exemplos de desenho de pixels no MediaCenter (por acesso à memória)

Em vez de dar comandos ao MediaCenter, também é possível escrever directamente na sua memória, que neste ecrã de 64 colunas por 32 linhas é constituída por 2048 palavras (uma palavra por cada um dos 64 x 32 pixels). A primeira palavra está localizada no endereço 8000H e a última em 8FFEh, de acordo com o mapa de endereços indicado atrás.

Os programas equivalentes ao **lab-comandos.asm** e **lab-comandos-xadrez.asm**, mas com escrita dos pixels na memória, estão contidos nos ficheiros **lab-memoria.asm** e **lab-memoria-xadrez.asm**, respetivamente, que também são fornecidos.

Pode comparar os dois casos mais simples (**lab-comandos.asm** e **lab-memoria.asm**), a partir do label “escreve\_pixel”. Com comandos bastam 3 MOVs, mas com acesso à memória são precisas mais algumas instruções.

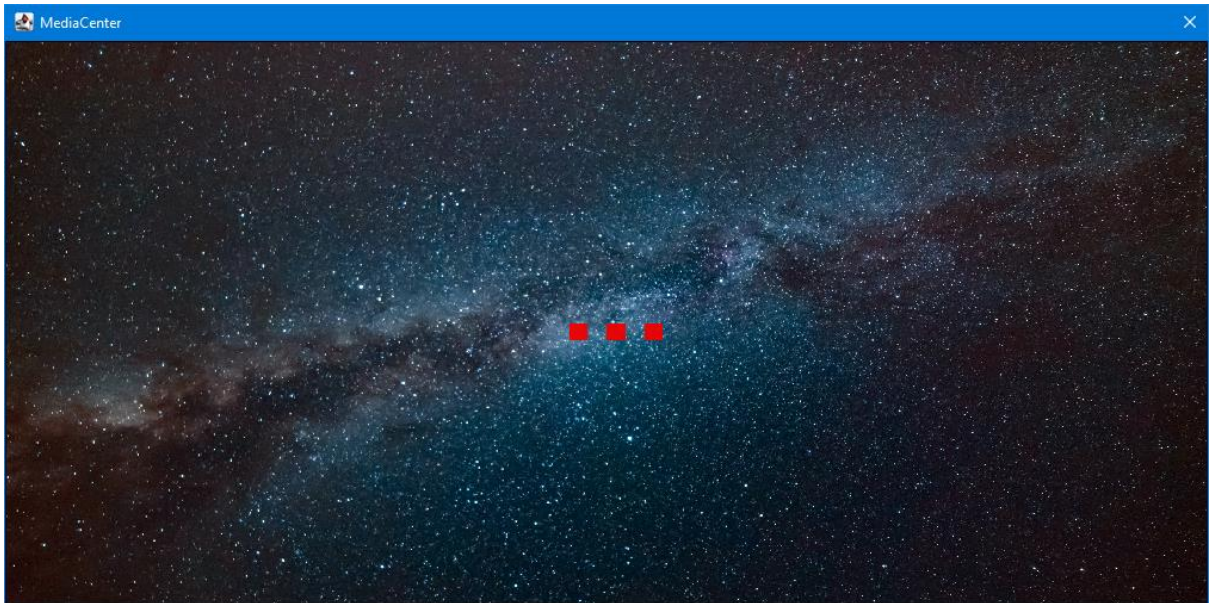
No entanto, mesmo com acessos à memória, os comandos são essenciais para diversas funcionalidades, como apagar o aviso de nenhum cenário de fundo selecionado ou até para limpar o ecrã dos pixels já escritos. O programa do ficheiro **lab-memoria-xadrez.asm**, embora escreva os pixels na memória, usa comandos para estas funcionalidades. Por tudo isto, é mais prático usar apenas comandos para lidar com o MediaCenter.

## 6.6 – Desenho de bonecos

Desenhar um boneco no MediaCenter (vai precisar de o fazer para o projeto) é desenhar um conjunto de pixels, que devem estar definidos numa tabela.

Toma-se um dos pixels do objeto como referência da sua posição no ecrã (por exemplo, o pixel do canto superior esquerdo) e faz-se um programa que percorra essa tabela e vá desenhando pixel a pixel, em posições sucessivas, relativas à posição do pixel de referência.

Carregue o programa **lab-boneco.asm** e execute-o. Deverá obter a figura seguinte.



Note a forma como o boneco é definido, numa tabela feita com várias WORDs consecutivas, em que a primeira indica a largura em pixels do boneco e as restantes as cores dos respetivos pixels. Depois há um ciclo no programa que vai desenhando cada pixel da tabela em colunas sucessivamente mais para a direita.

Este é um boneco só com largura (altura de 1 pixel). Para desenhar bonecos com altura A e largura L, basta indicar tanto a largura como altura (2 WORDs), seguidas de A linhas de L WORDs cada. O código para desenhar o boneco tem agora de ter dois ciclos, o interior com L iterações para desenhar cada linha, e o exterior com A iterações para desenhar as várias linhas. Em cada iteração tem de se ir aumentando a coluna (em cada linha) e a linha (sempre que se passa à linha seguinte).

## 6.7 – Movimento de bonecos

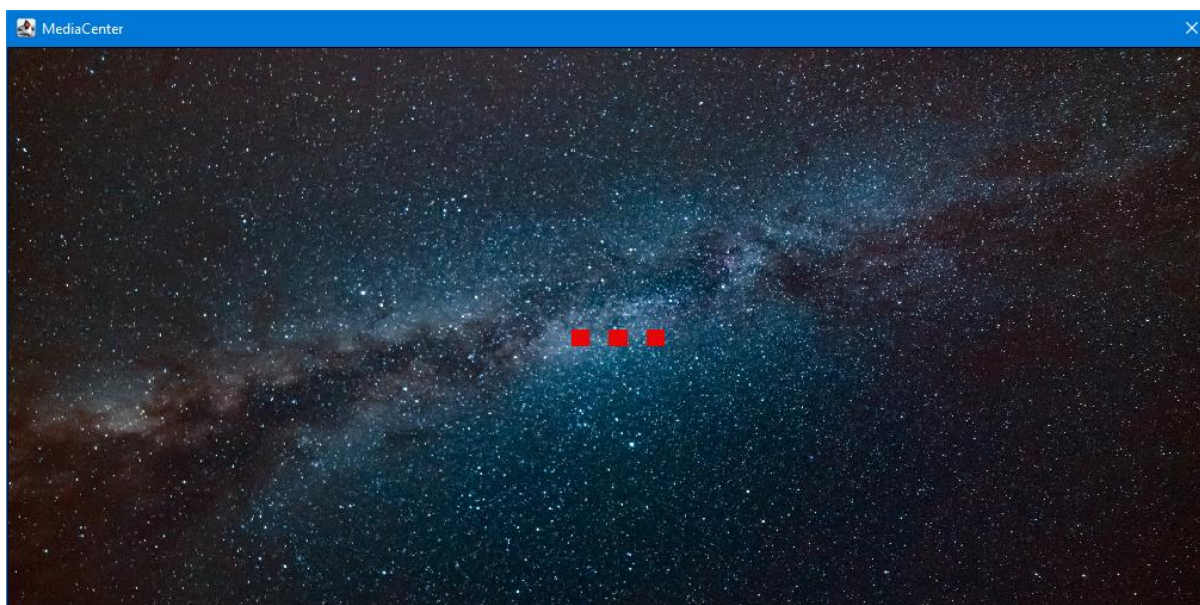
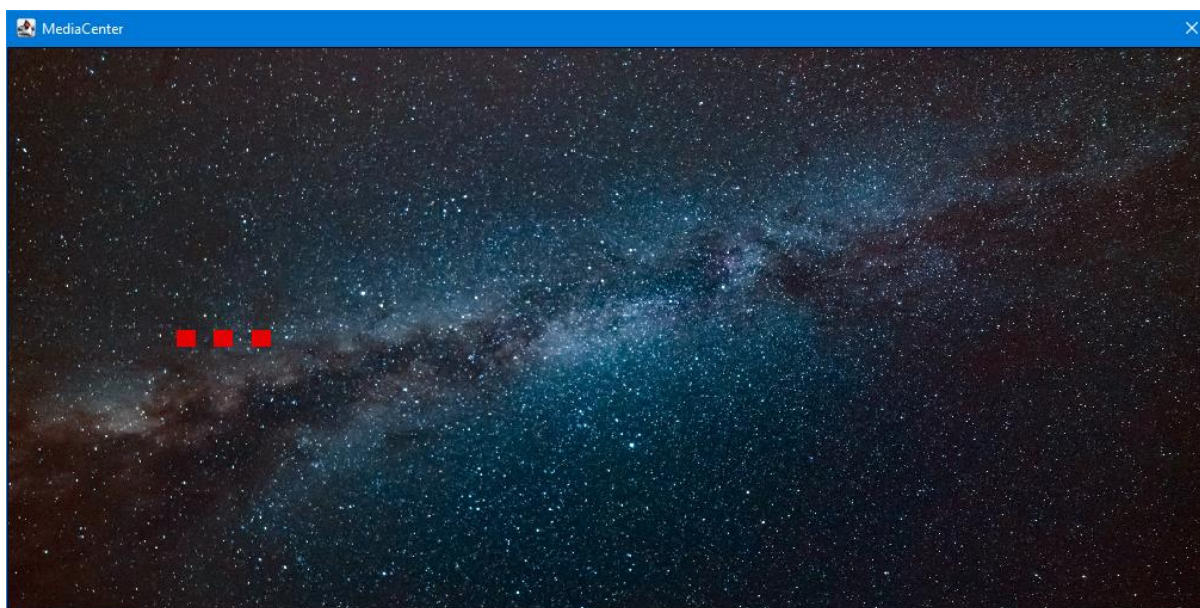
Movimentar um boneco no MediaCenter (vai precisar de o fazer para o projeto) é apagar o objeto na sua posição corrente, mudar a posição corrente e voltar a desenhá-lo na nova posição. O efeito visual é ele mover-se.

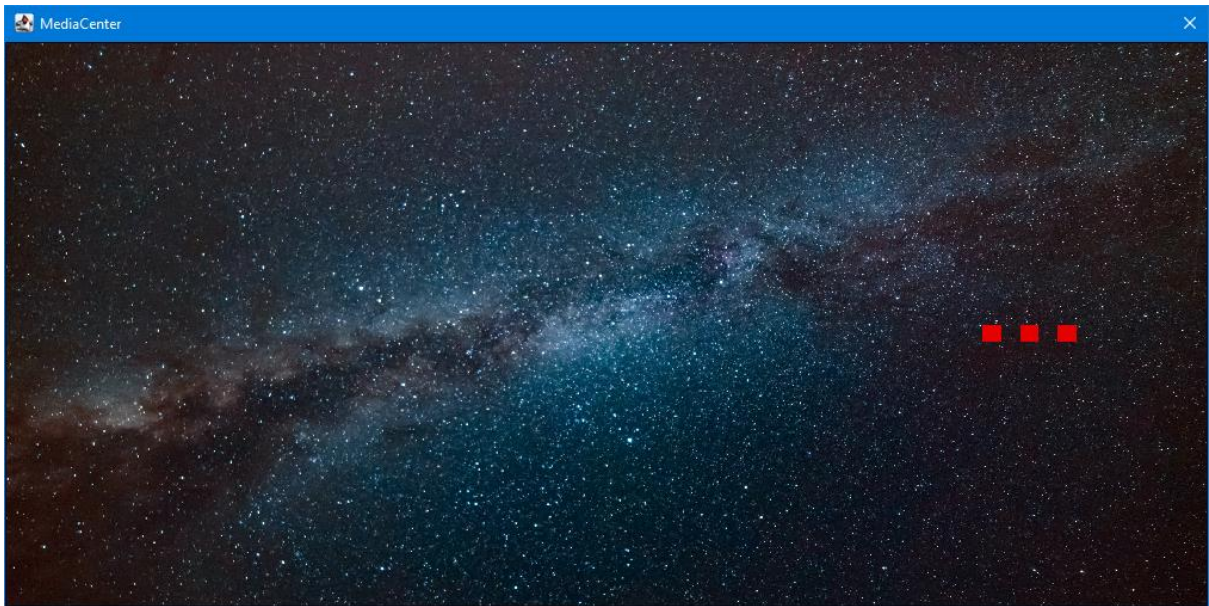
Posição corrente é a posição do pixel de referência do objeto (por exemplo, o pixel do canto superior esquerdo). Todos os restantes pixels são desenhados em posições relativas à do pixel de referência.

Carregue o programa **lab-move-boneco.asm** e execute-o. O boneco deverá movimentar-se de um lado para o outro, tal como ilustrado nas figuras seguintes.



A constante ATRASO é usada para limitar a velocidade do movimento. Poderá ter de a ajustar (maior ou menor), dependendo da velocidade do seu computador.





Este programa desenha o boneco numa dada coluna, faz o ciclo de atraso, apaga o objeto nessa coluna, testa os limites para saber se precisa de inverter o sentido, calcula a nova coluna (somando +1 ou -1 à coluna anterior) e volta atrás para desenhar de novo o boneco e repetir o ciclo.

### **6.8 – Múltiplos ecrãs de pixels**

Na janela de configuração do módulo MediaCenter (accedida com clique duplo no MediaCenter em modo “Design” – ver secção 2.2), é possível especificar o número de ecrãs (pixel screens) que se pretendem (até 16).

Cada ecrã está num plano diferente e a vantagem é que os bonecos que se desenhavam num ecrã podem “passar” por cima ou por baixo de bonecos noutros ecrãs, sem interferência.

Depois de definir o número de ecrãs, é possível o programa seleccionar o ecrã onde os próximos pixels irão ser desenhados, bem como esconder ou mostrar um ecrã inteiro. Tal faz-se por meio de comandos (ver secção 2.3).

### **6.9 – Cenários de fundo e frontal**

Por trás de todos os ecrãs de pixels está uma janela que pode ter uma imagem, que atua como cenário de fundo. Na frente de todos os ecrãs de pixels está outro, que tipicamente tem uma imagem com fundo transparente e assim permite colocar letreiros, por exemplo, à frente de tudo o resto (cenário frontal).

Na janela de configuração do módulo MediaCenter (accedida com clique duplo no MediaCenter em modo “Design” – ver secção 2.2), é possível definir as imagens que se pretendem (até 32), que são automaticamente esticadas para acompanhar a resolução da janela de simulação do MediaCenter. Depois no programa é possível seleccionar a que se quer como cenário, e ir mudando de acordo com os requisitos do programa, por meio de comandos (ver secção 2.3).

### **6.10 – Sons e vídeos**

Na janela de configuração do módulo MediaCenter (accedida com clique duplo no MediaCenter em modo “Design” – ver secção 2.2), é também possível definir sons e vídeos (até 32). Os vídeos são automaticamente esticados para acompanhar a resolução da janela de simulação do MediaCenter e estão em planos entre o cenário de fundo e os ecrãs de pixels.

Depois no programa é possível seleccionar os sons e os vídeos e proceder à respetiva reprodução. Existem vários comandos para este efeito (ver secção 2.3). Na janela de configuração é ainda possível obter informação sobre cada som e vídeo, e até definir os pontos de início e fim do tempo de reprodução.

A escolha “*as clip*” deve ser reservada para efeitos sonoros muito curtos (1 segundo, ou assim). Ocupam mais memória que os sons “não clip”, mas a grande vantagem é a latência (começam a reproduzir mais rapidamente) e podem ser recomeçados antes de acabarem. Em compensação, não se podem terminar a meio, nem fazer pausa, como os outros.

Os ficheiros de áudio e vídeo são carregados quando se passa o simulador para “Simulation”. Embora raro, acontece por vezes o carregamento de um destes ficheiros falhar, caso em que aparece uma mensagem de erro e esse ficheiro não poderá ser reproduzido. Basta passar para “Design” e de novo para “Simulation”.