



# 41952 - Arquitetura de Computadores II

<http://elearning.ua.pt>

Pedro Miguel Cabral

Aula 01

# Trabalho Prático Nº 01

AC2  
2021-2022

## Objectivos

- Conhecer o processo de criação de um programa escrito em assembly para correr na placa DETPIC32-IO: compilação, transferência e execução.
- Utilizar os system calls disponibilizados na placa DETPIC32-IO.
- Rever os conceitos associados à manipulação de arrays de caracteres.

# Introdução

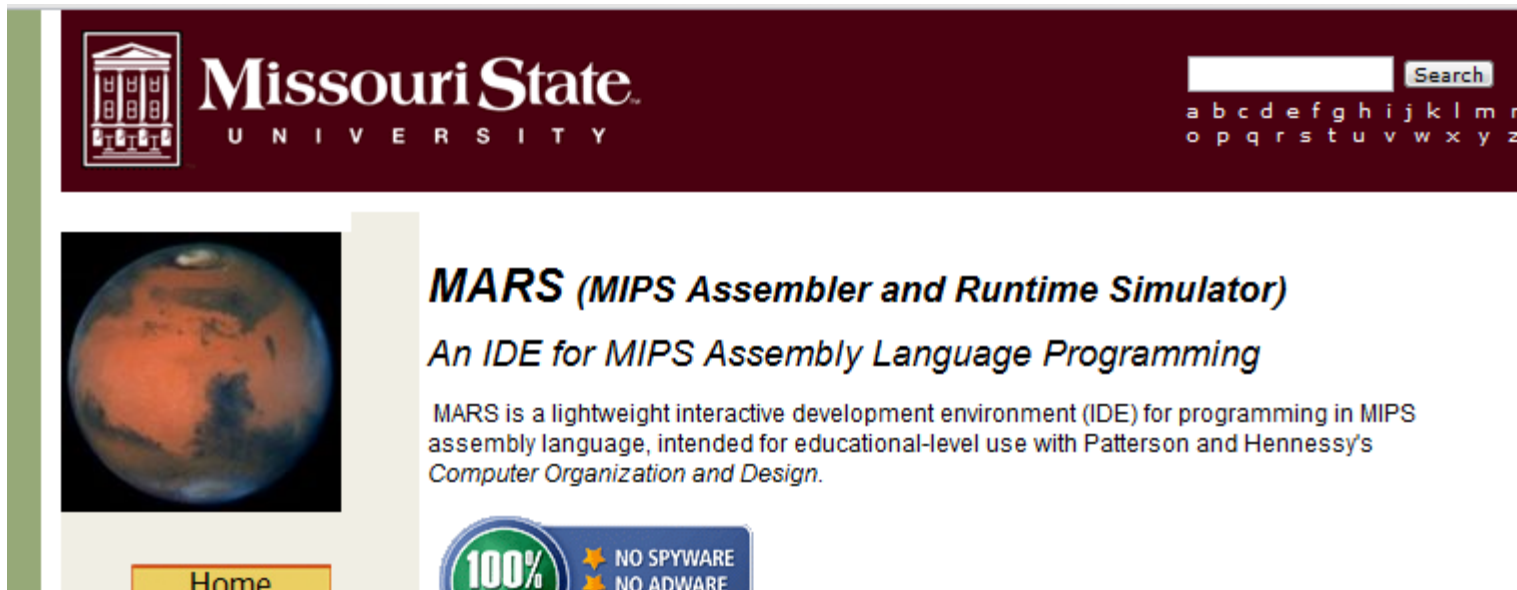
- Microcontrolador:
  - Um dispositivo programável que integra um microprocessador, memória e portos de entrada e saída.
  - Inclui outros dispositivos de suporte tais como *Timers*, Conversores A/D, interfaces de barramentos (RS232, I2C, SPI, etc.)
  - Os pinos de conexão com exterior estão (normalmente) organizados em portos de I/O e são multiplexados, i. e., podem ser configurados para diferentes funções.

# Introdução

- Sistema Embebido (*Embedded System*):
  - **Sistema computacional especializado** - realiza uma tarefa específica ou o controlo de um determinado dispositivo
  - Tem requisitos próprios e executa apenas **tarefas pré-definidas**
  - Tem um **custo muito inferior** a um sistema computacional de uso geral. Os **recursos** disponíveis são, em geral **mais limitados** que num sistema computacional de uso geral (e.g. menos memória, ausência de dispositivos de interacção com o utilizador)
  - É, normalmente, implementado com base num **microcontrolador**

# Introdução

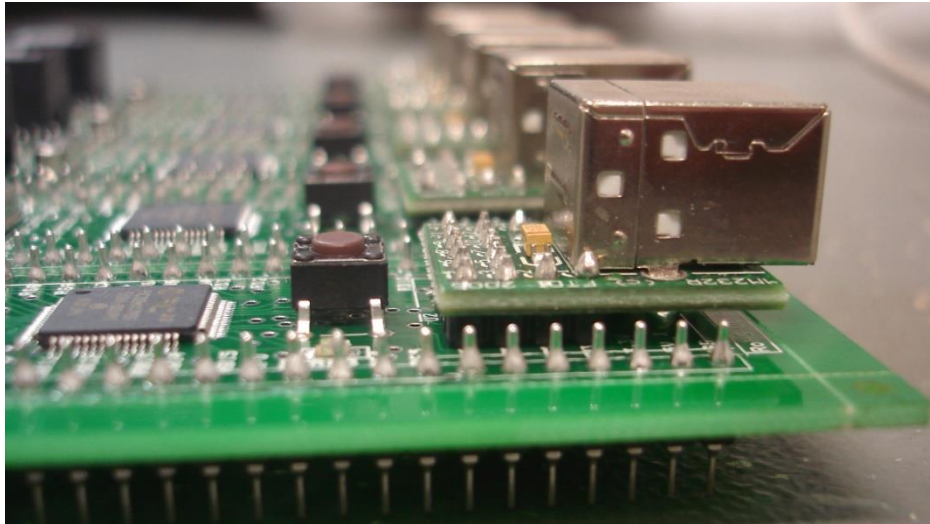
- Em AC1 usámos o Assembly para programar o processador MIPS...
- ... No ambiente de simulação MARS



The screenshot shows the homepage of the MARS (MIPS Assembler and Runtime Simulator) website. At the top is a dark red header with the Missouri State University logo and name on the left, and a search bar with a 'Search' button and a keyboard layout on the right. Below the header, on the left, is a circular image of the planet Mars. To the right of the image, the text reads: **MARS** (*MIPS Assembler and Runtime Simulator*)  
*An IDE for MIPS Assembly Language Programming*  
MARS is a lightweight interactive development environment (IDE) for programming in MIPS assembly language, intended for educational-level use with Patterson and Hennessy's *Computer Organization and Design*.  
At the bottom left is a yellow 'Home' button. At the bottom right is a blue badge with a green circle containing '100%' and the text 'NO SPYWARE' and 'NO ADWARE'.

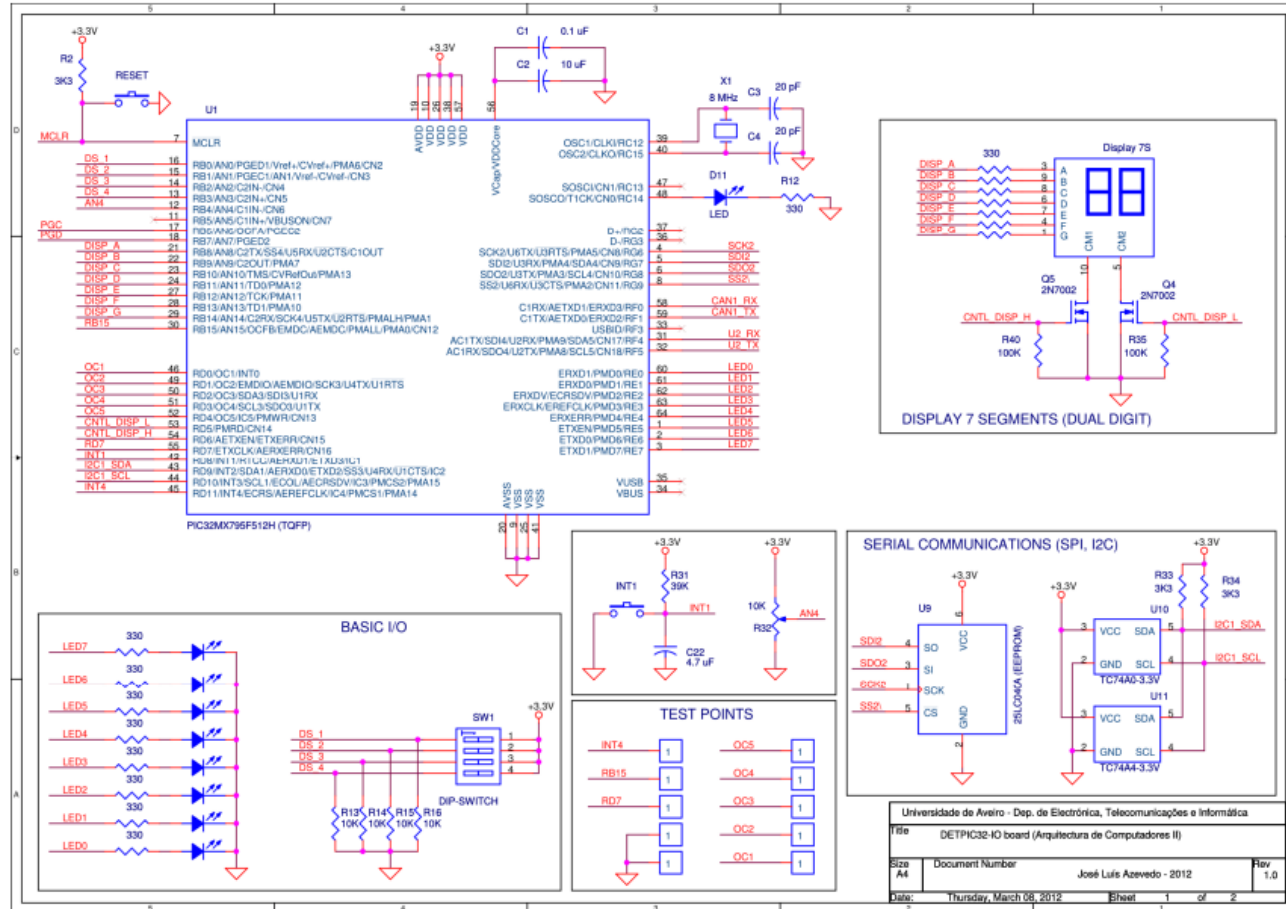
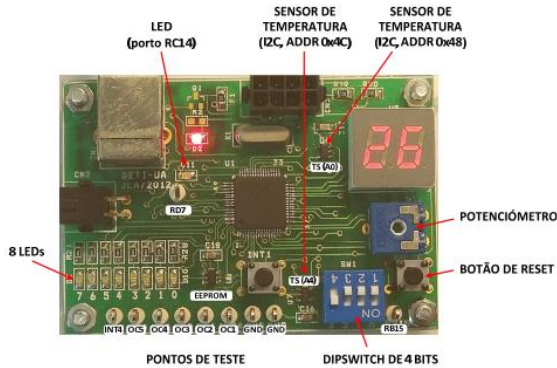
# Introdução

- Agora, o ambiente de simulação tornou-se real...



# Aula Prática Nº 01

AC2  
2021-2022



# Aula Prática Nº 01

- Utilizando um editor de texto (p. ex. o gvim), edite o programa Assembly, dando-lhe a extensão ".s".

**gvim prog1.s**

- Compile o programa introduzindo, na linha de comandos (numa janela de terminal do linux), o seguinte comando:

**pcompile prog1.s**

O comando produz os seguintes ficheiros:

"prog1.o", "prog1.elf", "prog1.map" e "prog1.hex "

- Transfira o programa "prog1.hex" para a memória FLASH do microcontrolador PIC32 da placa DETPIC32, realizando os seguintes passos:
  - ligue a placa à porta USB do PC
  - introduza o comando: **ldpic32 prog1.hex**
  - prima o botão de reset da placa DETPIC32-IO e aguarde que a transferência se processe
- Lance o programa monitor da porta série **pterm**
- Execute o programa premindo novamente o botão de reset.



# Aula Prática Nº 01

## System Calls

| Tabela IV: <i>System Calls</i> do DETPIC32                               |      |                          |         |
|--|------|--------------------------|---------|
| Protótipo equivalent em C  | \$v0 | Parâmetros de entrada    | Retorno |
| <b>char</b> inkey( <b>void</b> )   | 1    |                          | \$v0    |
| <b>char</b> getChar( <b>void</b> )                                       | 2    |                          | \$v0    |
| <b>void</b> putChar( <b>char</b> ch)                                     | 3    | \$a0 = character         |         |
| <b>unsigned int</b> readInt( <b>unsigned int</b> base)                   | 4    | \$a0 = base              | \$v0    |
| <b>int</b> readInt10( <b>void</b> )                                      | 5    |                          | \$v0    |
| <b>void</b> printInt( <b>unsigned int</b> val, <b>unsigned int</b> base) | 6    | \$a0 = val, \$a1 = base  |         |
| <b>void</b> printInt10( <b>int</b> val)                                  | 7    | \$a0                     |         |
| <b>void</b> printStr( <b>char</b> *str)                                  | 8    | \$a0 = str               |         |
| <b>void</b> readStr( <b>char</b> *buffer, <b>unsigned int</b> nc)        | 9    | \$a0 = buffer, \$a1 = nc |         |
| <b>void</b> exit( <b>int</b> code)                                       | 10   | \$a0 = exit code         |         |
| <b>unsigned int</b> readCoreTimer( <b>void</b> )                         | 11   |                          | \$v0    |
| <b>void</b> resetCoreTimer( <b>void</b> )                                | 12   |                          |         |

**printInt()**, "base": **16 lsbits** – [2.. 16] / **16 msbits** – número de caracteres com que o resultado é apresentado (o valor por omissão é 0, i.e. sem formatação)

# Mapa de memória do PIC32 (perspectiva do programador)

AC2  
2021-2022

