Trabalho do CESAR (2021/2)

Descrição Geral

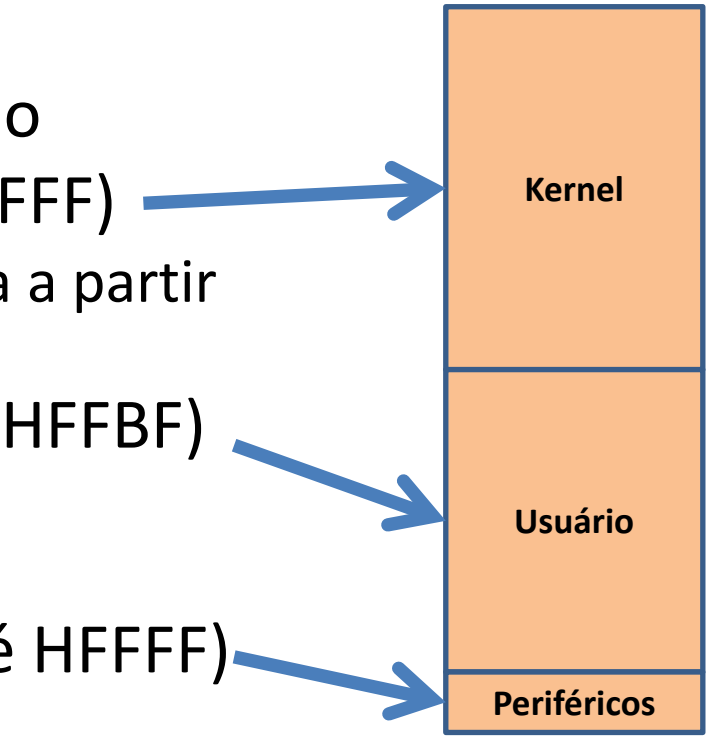
- Sua tarefa é desenvolver o Kernel responsável pelo controle dos periféricos.
- Para isso, você deverá desenvolver funções para controlar:
  - Visor, Teclado e Timer
- Além disso, você deve implementar a inicialização do processador (procedimentos de *reset*), que envolve
  - Inicialização dos periféricos e sistemas de interrupção
  - Inicialização de quaisquer outras variáveis necessárias para as funcionalidades que você está desenvolvendo
- Finalmente, você deverá implementar o tratador de interrupção dos periféricos

Funções do Kernel

- Função para teclado
  - _kbhit e _getchar
- Função para visor
  - _putchar e _putmsg
- Funções para os relógios
  - _getclock, _setclock e _setalarm
- A descrição de cada função encontra-se no arquivo “FuncoesDoKernel.pdf”

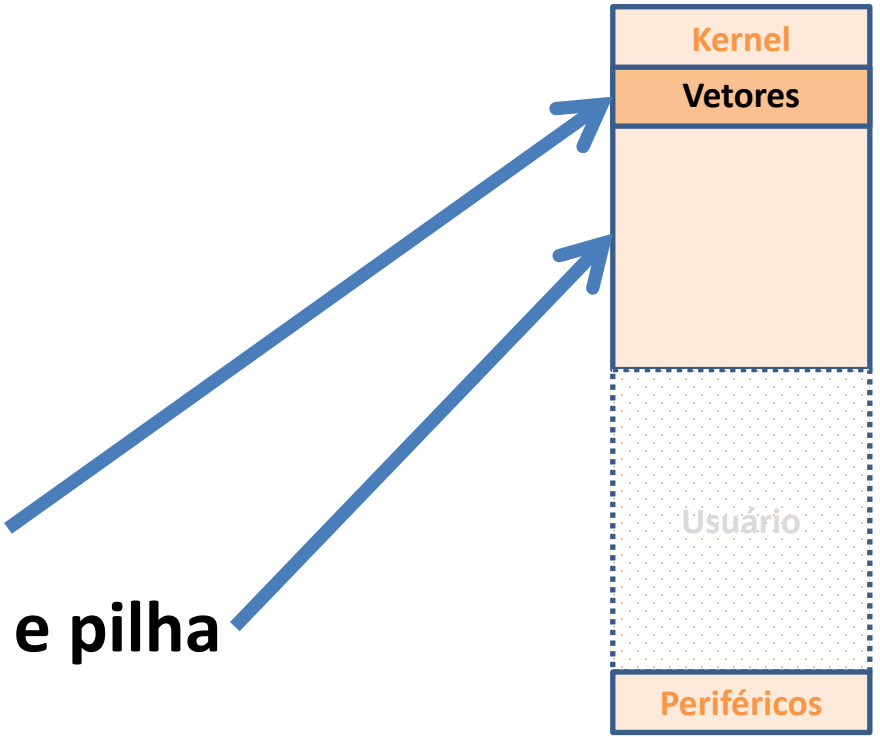
Espaço de Endereçamento

- Forma como o espaço de endereçamento deve ser dividido
- Espaço do kernel (H0000 até H7FFF)
  - Essa parte pode ser desenvolvida a partir do arquivo “KERNEL_REF.CED”
- Espaço da aplicação (H8000 até HFFBF)
  - Essa parte está no arquivo “APP_PROF.MEM”
- Espaço de Periféricos (HFFC0 até HFFFF)



O quê manipular?

- Espaços
  - Espaço de kernel
  - Espaço de periféricos
- No espaço de kernel
  - Os **vetores** de entrada
  - Área de **código, dados e pilha**



Vetores de Entrada

- Estão organizados na forma de uma tabela de ponteiros
  - A aplicação chama as funções através dos ponteiros

```
MOV   #VETOR,R0
JSR   R7,(_VETTAB(R0))
```

  - VETOR é uma constante que depende da função a ser chamada
    - Ex: #2, para a _getchar
    - Ex: #8, para a _get_clock_time
- Existe um vetor para cada função a ser implementada
- Forma e ordem dos vetores
  - Ver arquivo “KERNEL_REF.CED”

```
org     _VETTAB

dw      _kbhit
dw      _getchar
dw      _putchar
dw      _putmsg
dw      _getclock
dw      _setclock
dw      _setalarm
```

Código, dados e pilha

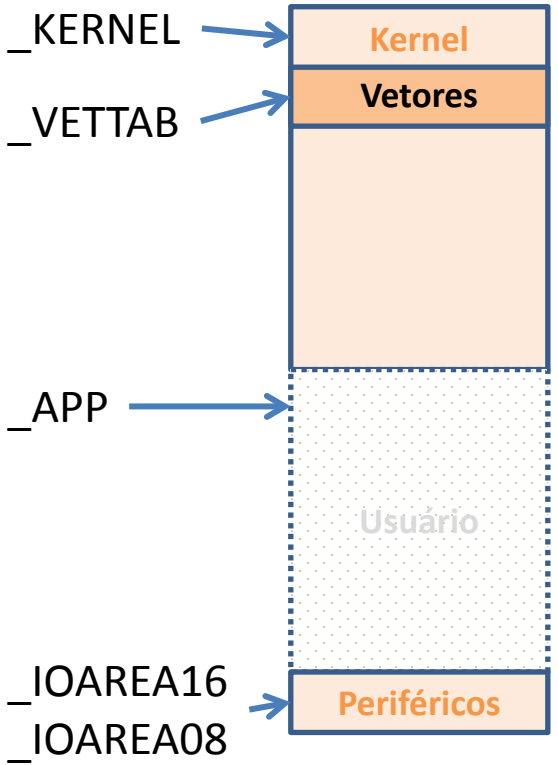
- Área com sua implementação
  - As funções do kernel
  - Os tratadores de interrupção
  - As variáveis necessárias para o seu kernel
  - A área reservada para a pilha do sistema
    - Que será usada pelo kernel e pela aplicação

KERNEL_REF.CED

- Arquivo fornecido pelo professor
- Para ser usado como ponto de partida para o desenvolvimento de seu kernel

kernel_ref.ced

- Definição de vários símbolos úteis (equates)
- Alguns desses símbolos
  - Ponto de entrada no “reset” do processador (_KERNEL)
  - Definição da área de vetores (_VETTAB)
  - Início do programa de aplicação (_APP)
  - Área dos periféricos (_IOAREA16 e _IOAREA08)



KERNEL_REF.CED

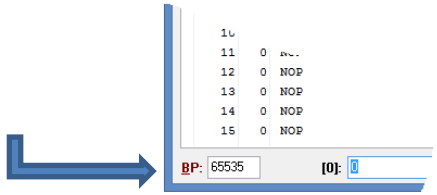
- Lista de procedimentos de reset
  - Nessa área você encontrará uma lista de atividades a serem implementadas, conforme seu projeto de kernel
  - Essa função deve encerrar com um “JMP _APP”
- Protótipos das funções da API do kernel
  - Os detalhes das funções estão no arquivo “FuncoesDoKernel2021_2.pdf”
  - Essas funções devem encerrar com um “RTS R7”

Como preparar seu programa

- Montar seu kernel, usando o Daedalus
- Abrir o simulador
  - Não esquecer de desligar o “Atualizar registradores”
- Carregar seu kernel no simulador (CTRL-C)
- Realizar a Carga Parcial (CTRL-P) da aplicação fornecida pelo professor, com os seguintes endereços:
  - Endereço inicial da memória a copiar: **32768**
  - Endereço final da memória a copiar: **65471**
  - Endereço de destino: **32768**
- Resetar o processador (F10)
- Rodar o programa (F9)

Para depurar seu kernel

- Lembre-se que a interrupção só é chamada quando em execução
  - Durante a execução passo-a-passo a chamada da interrupção está desabilitada
- Para fazer o programa parar dentro da interrupção, é necessário usar “break-points”



Arquivos Fornecidos

- TrabalhoCesar2021_2.pdf
  - Descrição do trabalho
- FuncoesDoKernel2021_2.pdf
  - Descrição das funções do kernel
- app_prof.mem
  - Programa de aplicação, fornecido pelo professor
- kernel_ref.ced
  - Fonte de referência, para iniciar o desenvolvimento
- Apresentacao.pdf
  - Esse conjunto de slides