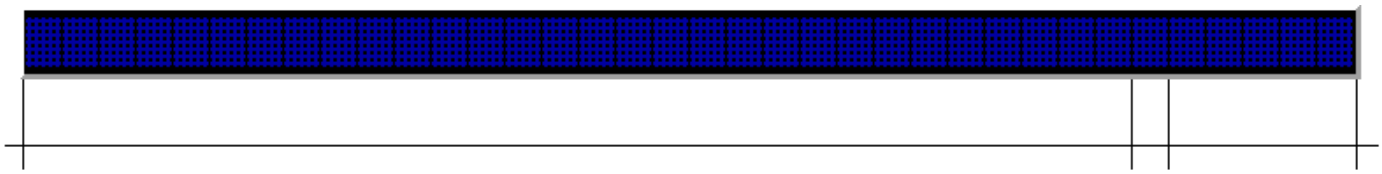
Kernel – 2022/2

Este documento descreve as estruturas a serem gerenciadas pelo Kernel assim como as funções da API de acesso ao Kernel, que deverão ser implementadas no trabalho com o Computador CESAR16i.

Gerência do Visor

O kernel deve gerenciar o visor de maneira que o usuário perceba duas áreas de apresentação de caracteres. A primeira área (caracteres) ocupa os 30 caracteres mais na esquerda do visor; a segunda área (volume de som) ocupa os 5 caracteres mais na direita do visor. Entre as duas áreas há um caractere que não receberá qualquer caractere. A distribuição das áreas está na figura abaixo.



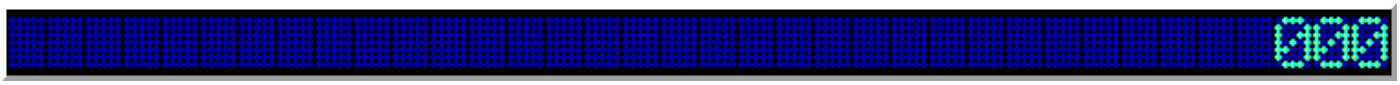
Seu kernel vai receber caracteres para colocar na Área Caracteres através das funções “putchar” e “putmsg”. O kernel terá como responsabilidade coloca-los, corretamente, na posição do visor indicada pelo cursor. Caracteres enviados quando a Área Caracteres estiver cheia (o cursor saiu da área útil) deverão ser ignorados.

A Área Volume é preenchida pelo kernel, com o valor atual do volume de um amplificador de som, alinhado à direita. A escrita na Área Volume é da responsabilidade do kernel. Nessa área deve ser escrito o valor do volume de som, alinhada à direita.

Para a gestão da Área Caracteres, seu kernel deve implementar as funções correspondentes aos caracteres de controle BS (H08) – *back space*, CR (H0D) – *carriage return*, e LF (H0A) – *line feed*.

O *back space* move o cursor uma posição para a esquerda, exceto quando o cursor já estiver na posição mais à esquerda do visor. O *carriage return* posiciona o cursos na posição mais a esquerda do visor e o *line feed* limpa o visor (preenche com SPACE) sem alterar a posição do cursor.

Quando o kernel terminar sua inicialização e passar à execução da aplicação, a Área Caractere deverá estar limpa (preenchida com caracteres SPACE - H20) e a Área Volume deverá apresentar o volume atual do som (que deve ser zero). Abaixo está mostrado o visor, conforme inicialização.



Gerência do Teclado

Seu kernel deverá ser capaz de informar para a aplicação que o usuário do programa digitou alguma tecla. Para isso, seu kernel deverá implementar as funções “kbhit” e “getchar”. Notar que a função “getchar” não coloca os caracteres digitados no visor. A única função do “getchar” é devolver para a aplicação os caracteres digitados. Ainda, a função “kbhit” informa para a aplicação se algo foi digitado. Mas, esta função não retorna nenhuma tecla nem remove a tecla digitada.

Gerência do Volume

Seu kernel deverá gerenciar (incrementar e decrementar) um número que representa a posição do controle de volume de um amplificador de som. Seu kernel deve implementar a função “volume”, que fornece o volume do som para o programa de aplicação.

A alteração do volume deverá ser implementada no kernel. O kernel promoverá essas alterações a partir do acionamento de teclas específicas. Abaixo estão descritas as ações a serem efetuadas no valor do volume, para cada uma das teclas:

Tecla	Ação
+ (mais)	Volume = Volume + 1;
– (menos)	Volume = Volume – 1;
. (ponto)	Volume = Volume + 10;
, (vírgula)	Volume = Volume – 10;
> (maior)	Volume = Volume + 100;
< (menor)	Volume = Volume – 100.

As teclas usadas para controlar o volume não devem ser repassadas para a aplicação.

Gerência do Timer

O kernel desenvolvido deverá dar suporte para um temporizador. A gestão desse temporizador será realizada pelo kernel e será inicializada pela função "timer". Essa função tem dois parâmetros: o tempo para inicialização do temporizador e o endereço da função a ser chamada quando o tempo se esgotar (atingir zero).

O kernel deve implementar a contagem de tempo como um valor de 16 bits, sem sinal. Cada unidade corresponderá a 1 (um) milissegundo (ms) de tempo. Portanto, o tempo máximo dessa temporização é de 65,535 segundos.

A temporização inicia com o valor da contagem de tempo passado quando a função "timer" for chamada. Quando o kernel identificar que o tempo se esgotou (quando a temporização atingir o valor "zero"), o kernel deverá chamar a função cujo endereço foi passado como parâmetro da função "timer".

Importante salientar que, enquanto o tempo passa, o programa de aplicação continua sua operação normal e o tratador de interrupção de tempo deve efetuar, concomitantemente, a contagem adequada de temporização.

Funções do Kernel

A seguir são descritas as funções da API e a forma como serão chamadas pelos programas de aplicação, incluindo os parâmetros de entrada e valores a serem retornados como resultado.

Essas funções permitem que o programa de aplicação possa utilizar os periféricos disponíveis no CESAR16i (teclado, visor e *timer*), sem a necessidade de conhecer o funcionamento do hardware e das portas de acesso a esses periféricos.

As funções a serem implementadas deverão ser colocadas na área de memória reservada para o kernel.

Na área de memória reservada para a Tabela de Vetores do Kernel, você deverá colocar os vetores, na ordem definida neste documento. Cada vetor (ponteiro com 2 bytes) dessa tabela deve conter o endereço, no kernel, onde inicia a implementação da função correspondente ao vetor.

Observar que a ordem desses vetores deve ser obedecida, rigorosamente. Os vetores, ordenados segundo seu número de ordem, e suas funções correspondentes estão indicados abaixo:

Vetor	Função
[0]	kbhit
[1]	getchar
[2]	putchar
[3]	putmsg
[4]	timer
[5]	volume

A seguir, você encontra a descrição das funções a serem implementadas. No título de cada função está indicado o protótipo em "C" da função. Nesse protótipo o tipo "WORD" indica um valor com 16 bits sem sinal e "BYTE" indica um valor com 8 bits sem sinal.

0. Função: "kbhit" – WORD kbhit(void)

Função através da qual a aplicação solicita ao kernel a informação da existência de alguma tecla digitada. A função deve retornar com a informação da existência de tecla. Essa função retorna imediatamente, sem aguardar pela digitação de qualquer tecla.

- *Parâmetro de saída*: registrador R0, com a informação da existência de tecla.

A função retorna no registrador R0 a informação da existência de tecla digitada.

- **Se há tecla, o valor retornado em R0 deverá ser zero**;
- Se não há tecla, o valor retornado em R0 será um valor qualquer diferente de zero.

1. Função: "getchar" – BYTE getchar(void)

Função através da qual a aplicação solicita ao kernel que informe a tecla digitada. Caso não tenha sido digitada uma tecla, a função deve aguardar até que seja digitada uma tecla (a função é "*bloqueante*"). Ao ser digitada uma tecla, a função deve retornar o código ASCII dessa tecla.

- *Parâmetros de entrada*: nenhum.
- *Parâmetro de saída*: registrador R0, com a tecla digitada.

A função só retorna (só termina) se houver uma tecla já digitada ou quando o usuário digitar alguma tecla. O código ASCII da tecla digitada deve ser retornado no registrador R0.

2. Função: "putchar" – void putchar(BYTE c)

A aplicação usa essa função para solicitar que seja colocado um caractere na Área Caracteres do visor, na posição indicada pelo cursor.

- *Parâmetros de entrada*:

- Registrador R5, com o caractere a ser colocado no visor, codificado em ASCII. Só são aceitos valores entre H20 (SPACE) e H7A ("z") e os caracteres de controle BS, CR e LF.

Caso algum dos parâmetros seja inválido, a função não deve apresentar qualquer informação no visor. Além dos caracteres visíveis, a função deve tratar os seguintes caracteres de controle:

Tecla	Operação
BS (H08) – *Back space*	Move o cursor uma posição para a esquerda. Caso o cursor esteja na posição mais a esquerda da Área Caracteres, o cursor não é alterado.
CR (H0D) – *Carriage return*	Move o cursor para a posição mais a esquerda da Área Caracteres.
LF (H0A) – *Line feed*	Limpa a Área Caracteres (preenche com H20)

3. Função: "putmsg" – void putmsg(BYTE *msg)

A aplicação usa essa função para solicitar que seja colocado na Área Caracteres um *string* de caracteres (bytes) terminado por um byte H00 (o mesmo delimitador usado em string "C"). Os caracteres ASCII (visíveis e de controle) que formam o *string* devem ser tratados conforme definido na função "putchar".

Se a mensagem atingir a última posição mais a direita da Área Caracteres, os caracteres restantes devem ser ignorados.

Caracteres inválidos (conforme definido na "putchar") devem ser ignorados e continuar o processamento dos caracteres restantes, como se o caractere inválido não existisse.

- *Parâmetros de entrada*:
  - Registrador R5, com o endereço de memória onde inicia o string.

4. Função: "timer" – void timer(WORD tempo, WORD callback)

Essa função é usada pela aplicação para programar a temporização do kernel. Os parâmetros a serem passados são o "tempo" de temporização e o endereço "callback" que o kernel deve chamar, quando o tempo atingir o valor zero.

- *Parâmetros de entrada*:
  - Registrador R5, com o "tempo" a ser programado no temporizador.
  - Registrador R4, com o endereço de "callback".

Quando a função é chamada, o kernel deve iniciar um temporizador com o valor programado. Esse temporizador deverá ser decrementado conforme o tempo passa. A passagem do tempo é identificada pelo tratador de interrupção de timpo. Quando o temporizador atingir o valor zero, o tratador de interrupção deverá chamar a função que está no endereço programado na chamada da função "timer". A chamada da função deve ser feita usando uma instrução "JSR" e com o registrador "R7". O endereço da função call-back pode ser fornecido usando o modo de endereçamento mais apropriado à programação.

5. Função: "volume" – WORD volume(void)

A aplicação chama essa função para obter informações sobre o volume de som do amplificador controlado pelo programa.

- *Parâmetro de saída*: registrador R0, com a informação atual da velocidade.

O valor retornado pode ser qualquer número entre 0 (zero) e 999, e representa o volume de som do amplificador.

A informação retornada por essa função representa o valor da velocidade controlado pelo usuário, através das teclas "+-.,><".