EEN251-Microcontroladores e Sistemas Embarcados
Pesquisa 22
RTOS

Bruna Tavares, Bruno Campos, Keneth Yamada

November 9, 2016

1 Visão Geral

1.1 Tarefas

Tarefas de um sistema operacional pode ser definido como um ou vários programas em fase de execução. O programa pode ser descrito como um conjunto de processos, com cada processo realizando um funcionamento próprio, sendo ele visível ou não para o usuário e mantendo de forma correta o funcionamento do sistema. As tarefas podem ser controladas pelo usuário, aplicativos ou pelo próprio sistema operacional.

1.2 Hard vs Soft RTOS

Hard RTOS: Sistema de defesa anti míssel. Caso o sistema falhe em processar as informações necessárias no tempo necessário, graves consequências ocorrerão.

Soft RTOS: Sistema de vídeo e áudio de um computador, em caso de falha de atingir a deadline no tempo correto, não acarretará em graves consequências como o Hard RTOS. Outro exemplo de Soft é o receptor de HDTV, um delay irá interferir na qualidade da imagem ou áudio, porém não compromente a sua utilização.

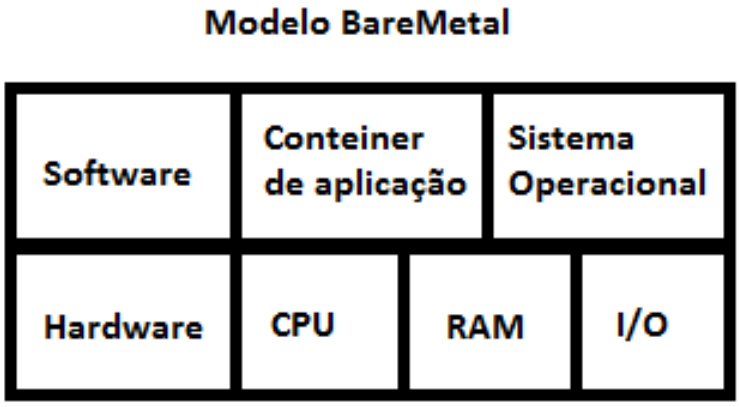
1.3 RTOS

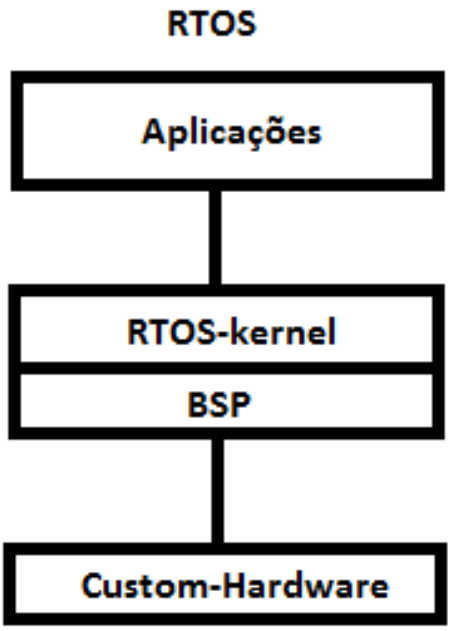
Exemplos de RTOS Open Source, seguido de seus proprietários:

1- Atomthreads, BSD
2- ChronOS, GNU GPL
3- distortos, Mozila
4- eChronos, AGPLv3
5- FreeOSEK, GNU GPLv3
6- iRTOS, GNU LGPL
7- mbed-rtos, MIT

8- RTLinux, GNU GPL
9- Symbian OS, Eclipse
10- TNKernel, BSD

1.4 Camada





1.5 Prioridade

No caso de sistemas operacionais multitarefas, como o Windows, a tarefa seria realizada junto com as outras (com diferentes prioridades). Sendo assim, o sistema operacional dedicará mais para a tarefa que estiver com maior prioridade.

1.6 Latência

Porque o RTOS foi criado para resolver problemas e satisfazer condições de multitasking em rigorosas restrições de tempo.

2 RTOS Conceitos Básicos

2.1 Blocked

A função do estado Blocked é informar se uma tarefa deve aguardar ou não um evento acontecer para que possa ser executada. As tarefas nesse estado não podem ser executadas até que esse evento específico ocorra ou um tempo limite determinado expire.

2.2 Escalonador

Para que um RTOS faça o escalonamento, é necessário que o Kernel assuma o controle da CPU. Com o Kernel no controle, ele irá verificar qual a tarefa com maior prioridade a ser executada no momento.

Assim, a tarefa requsita um delay. Enquanto isso, o kernel irá verificar se há uma tarefa com maior prioridade. Caso haja, uma interrupção no timer do sistema é feita e tratada pelo processador, assim a tarefa em execução é colocada em espera para que uma outra tarefa com maior prioridade possa ser realizada.

Quando o processo com maior urgência ser finalizado, a tarefa em espera retorna para seu estado de execução.