

Introdução

- ▶ O presente trabalho, que é dedicado a Gerenciamento de Tarefas, faz parte do projeto de pesquisa que tem por objetivo o desenvolvimento de um Nanokernel paralelo para Sistemas Embarcados e de tempo real. Esse Nanokernel deverá oferecer suporte para processadores monocores e multicores e será formado por módulos que poderão ser agrupados de maneira a atender as especificações das aplicações alvo.
- ▶ O gerenciamento de tarefas é responsável pelas operações de criação e término de execução de tarefas locais e remotas.

Contextualização do PaNKE

- ▶ O PaNKE é a versão paralela do nanokernel NKE [1]. Possui um módulo de comunicação que implementa operações de envio/recepção de mensagens através de uma rede Ethernet. As tarefas componentes da aplicação são criadas explicitamente, localmente ou a distância e se comunicam com operações send e receive.

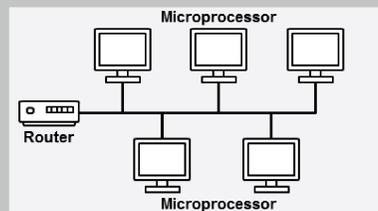


Figura 1: Arquitetura em placas paralelas

Criação de uma tarefa

- ▶ A criação local de uma tarefa, no mesmo processador, é alocada e inicializada em uma estrutura de dados, o registro descritor da tarefa, que contém o nome da tarefa, o número identificador, o processador, a prioridade, é criada uma pilha de execução e a tarefa é inserida na lista de tarefas que estão aptas a disputar o processador. A criação remota de uma tarefa é idêntica à criação local, exceto que a requisição de criação é recebida de uma tarefa rodando em outro processador.
- ▶ Chamada de sistema responsável: TaskCreate(Identificador, Ponto de Entrada, Processador, Período, Tempo Computacional);

Estados de uma tarefa no PaNKE

- ▶ No PaNke uma tarefa pode possuir 5 estados distintos:
 1. INITIAL : Estado inicial de uma tarefa;
 2. READY : Tarefa pronta a disputar o processador;
 3. RUNNING : Tarefa em execução;
 4. BLOCKED : Bloqueio temporario;
 5. TERMINATED : Fim execução, terminada.

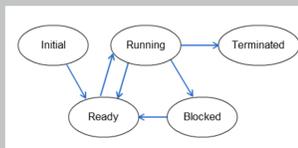


Figura 2: Transições de Estados [1]

Termino de execução de uma tarefa

- ▶ A operação de término normal de uma tarefa ocorre pela execução de seu último comando. A tarefa passa para o estado terminado e a mesma não é mais escalonada (não disputa mais o processador). O término anormal de uma tarefa ocorre em duas situações: por erro de execução ou por término provocado por outra tarefa. Por erro de execução toda a aplicação é terminada, isto é, todas as tarefas pertencentes à aplicação terminam e um erro é enviado pelo NanoKernel. Quando uma tarefa é terminada por outra, executando no mesmo processador ou em algum outro processador (operação remota de término de uma tarefa) as ações executadas são as mesmas do caso de término normal, ou seja, a tarefa passa para o estado terminado e não disputa mais o processador, não sendo mais escalonada.
- ▶ Chamadas de sistemas responsáveis: TaskExit(); TaskKill(Identificador da tarefa);

Resultado: Exemplo de codificação

```
#include "../Kernel/kernel.h"

int t1,t2,t3;
void tarefa1()
{
    char msg[10];
    while(1)
    {
        NkPrint("\n Esperando dado",0);
        Recv( t1, msg, sizeof(msg));
        NkPrint("\n Recebido = %s",msg);
    }
    TaskExit();
}
void tarefa2()
{
    char msg[10]={"Ola Mundo"};
    while(1)
    {
        NkPrint("\n Vou enviar",0);
        Send( t2, msg, sizeof(msg));
        NkPrint("\n Mandei = %s",msg);
    }
    TaskExit();
}
void tarefa3()
{
    while(1);
    TaskExit();
}
int main(int argc, char *argv[])
{
    TaskCreate(&t1, tarefa1, 0, 10, 2);
    TaskCreate(&t2, tarefa2, 0, 15, 4);
    TaskCreate(&t3, tarefa3, 0, 35, 10);
    Start(RM);
    return 0;
}
```

example_application.c

Resultado: Gráfico de escalonamento

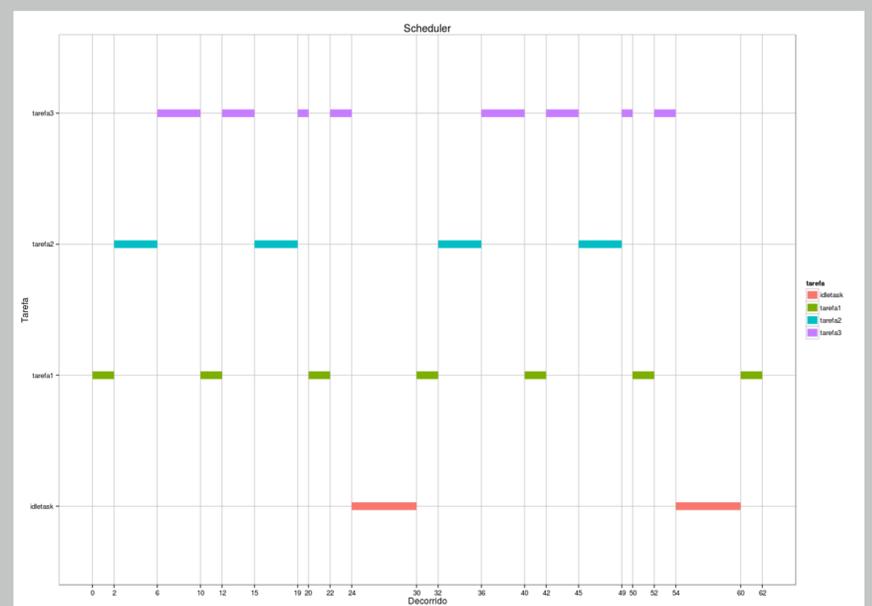


Figura 3: Escalonamento de tarefas utilizando o algoritmo Rate Monotonic

Conclusão

- ▶ Já se encontram operacionais as rotinas de criação e termino normal ou anormal de tarefas locais, periódicas e aperiódicas. Testes foram realizados com diversos programas rodando com um número variado de tarefas e o sistema se manteve estável. Atualmente estão sendo implementadas as primitivas de criação e término de tarefas remotas.

Referências

- [1] Costa, Celso Maciel da; Fragoso, João Leonardo; Murliky, Lucas; Silva, Leonardo da Luz; Fracalossi, Aline; Brasil, Cássio; Debom, Guilherme; Matias Jr., Rivalino. "NKE - Um Nanokernel Educacional para Microprocessadores ARM". IV Simpósio Brasileiro de Engenharia de Sistemas ComputacionaisManaus/AM, 3 a 7 de novembro de 2014. (aceito para publicação).