QXD0145 - Sistemas de Tempo-Real

Gerenciamento de Memória em Sistemas de Tempo-Real

André Ribeiro Braga

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

28/03/2023

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Referência

- Phillip A. Laplante , Seppo J. Ovaska ; *Real-Time Systems Design and Analysis*
  - Sessão 3.1.2 *Interrupt-Only Systems*
  - Sessão 3.1.5 *The Task Control Model*
  - Sessão 3.4 *Memory Management Issues*

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Alocação Dinâmica de Memória

- Tópico muitas vezes renegado
- Importância dupla na utilização sob-demanda de memória
  - *Tasks* da aplicação
  - Sistema operacional em si
- Aplicação utiliza memória...
  - Explícitamente por solicitações do *heap*
  - Implícitamente pela manutenção da memória em tempo de execução
- SO deve manter isolamento entre as *tasks*
- Alocação arriscada = Não determinística
  - *Stackoverflow* / *Deadlock*
- Evitar alocação arriscada $\Leftrightarrow$ Reduzir *overhead* com gerenciamento
- *Overhead* é componente significante do tempo de troca de contexto

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Gerenciamento de Pilha e TCB

- Sistemas multitarefas $\Rightarrow$ Salvamento/restauro de contexto
- Modelos de gerenciamento
  - Pilha(s) de execução (*runtime stack*)
    - *interrupt-only* + *foreground/background*
  - Bloco de controle
    - sistemas operacionais

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Pilha(s) de execução

- *Interrupt-only*
  - *jump-to-self loop*
  - *Tasks* escalonadas via interrupções
  - *Dispatching* realizado pelas rotinas de interrupção
  - *Hardware*/*Software*
  - Interrupções desabilitadas durante tratamento
- *foreground/background*
  - *jump-to-self loop* substituido por uma *task-idle*
- Cada *task* associada a uma interrupção

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Pilha(s) de execução

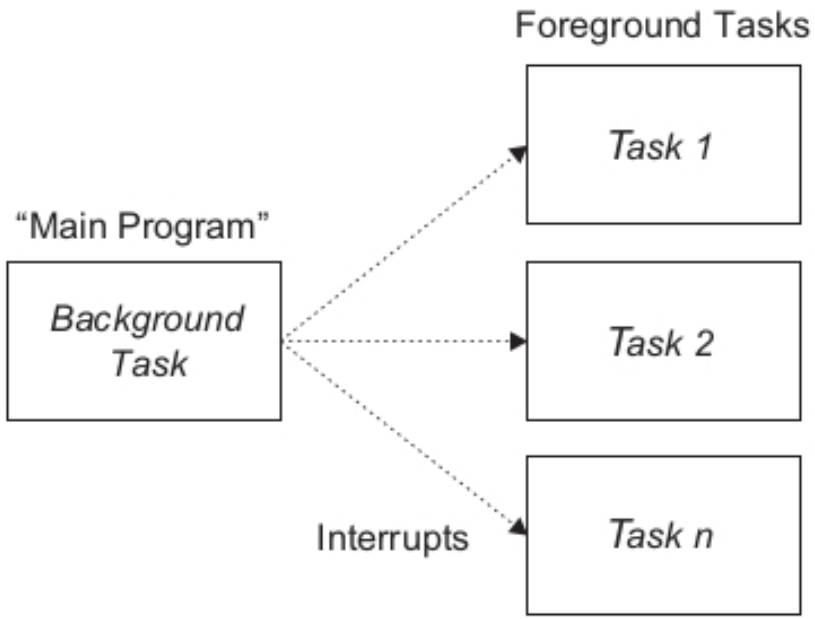
Interrupt-only

```
void main(void)
{
  init();            /* system initialization */
  while(TRUE);       /* jump-to-self */
}
void int_1(void)     /* interrupt handler 1 */
{
  save(context);     /* save context to stack */
  task_1();          /* execute task 1 */
  restore(context); /* restore context */
}
void int_2(void)     /* interrupt handler 2 */
{
  save(context);     /* save context to stack */
  task_2();          /* execute task 2 */
  restore(context); /* restore context */
}
void int_3(void)     /* interrupt handler 3 */
{
  save(context);     /* save context to stack */
  task_3();          /* execute task 3 */
  restore(context); /* restore context */
}
```

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Pilha(s) de execução

Foregroung/background



UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

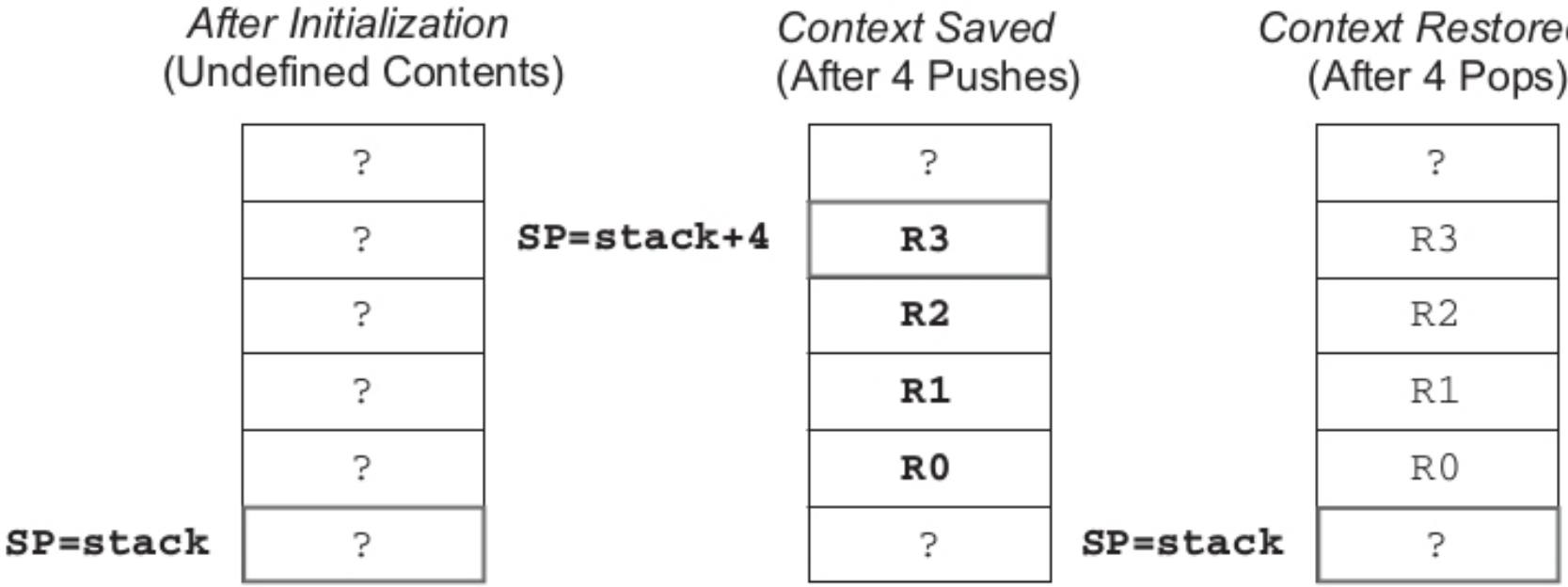
Pilha(s) de execução

- Duas rotinas necessárias
- *save*
  - chamado por um tratador de interrupção
  - salva o contexto do sistema para uma área de pilha
  - imediatamente após desabilitar interrupções
- *restore*
  - imediatamente antes de reabilitar interrupções

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Pilha(s) de execução

Foregroung/background



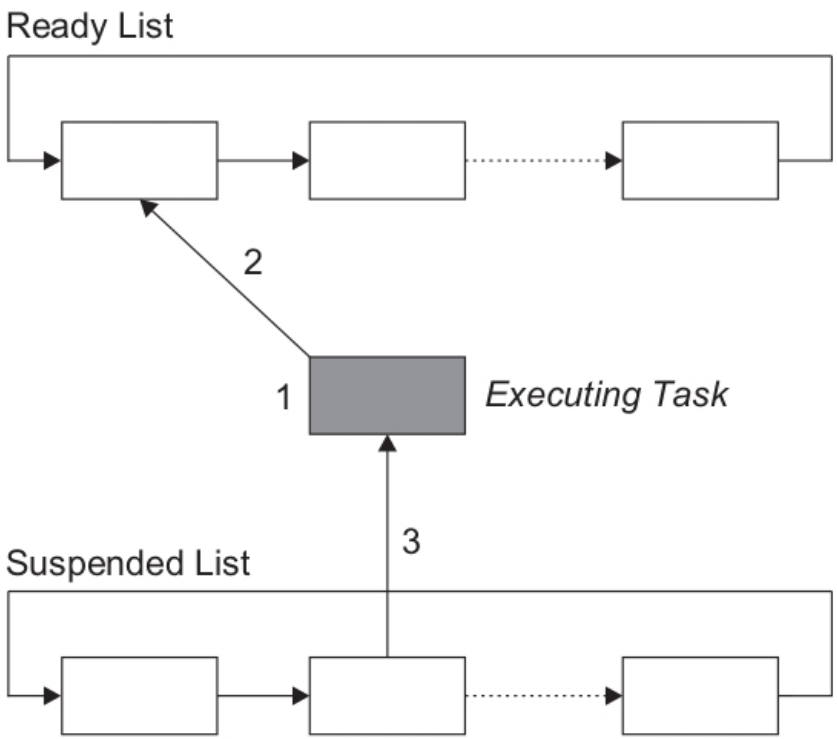
UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Modelo do Bloco de Controle

- Lista de TCBs deve ser mantida
- Pode ser fixa ...
  - n TCBs são alocados durante inicialização
  - Todas com estado *dormente*
  - *Task* criada $\Rightarrow$ *pronta*
  - Política de escalonamento $\Rightarrow$ *pronta* $\longleftrightarrow$ *executando*
  - *Task* deletada $\Rightarrow$ *dormente*
  - Sem gerenciamento de memória em tempo-real necessário
- ou dinâmica ...
  - Listas dinâmicas
  - *Task* criada $\Rightarrow$ *suspensa* + inserida na lista + *heap* alocado
  - Política de escalonamento $\Rightarrow$ *pronta* $\longleftrightarrow$ *executando*
  - *Task* deletada $\Rightarrow$ removida da lista + *heap* liberado
  - Necessidade de gerenciamento de memória *heap*

Modelo do Bloco de Controle

Lista de TCBs



UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Swapping

- Apenas uma *task* coexiste na memória principal
- Troca de *task*
  - Suspenção da *task* em execução
  - Espaço de usuário + contexto $\Rightarrow$ Memória secundária
- Tempo de execução de cada *task*
  - Longo o suficiente para compensar tempo de carregamento
- Variação do tempo de acesso à memória secundária
  - *Context-swich overhead* + resposta em tempo-real
  - Pode arruinar a pontualidade do sistema

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Overlaying

- Programa maior que memória física
- Divisão em seções dependentes
- Exige projeto de *software* cuidadoso
- Limitações de tempo-real
  - Dispositivos de memória secundária não-determinísticos
- Empregada em alguns RTOS comerciais

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Paging

- Pode levar a alta atividade do sistema (*thrashing*)
- Pouco provável de ser empregado em sistemas embarcados
  - *Overhead*
  - Falta de suporte do *hardware*
- Baixa previsibilidade
  - *Memory locking* pode mitigar problema

UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ

Garbage

- Memória alocada e abandonada pela *task*
- Pode ocorrer quando terminam de forma anormal
  - *malloc* utilizado e ponteiro perdido
- Também ocorre em sistemas orientados a objeto
- Tratamento em tempo-real
  - Função importante
  - Suporte da linguagem (e.g. Java)...
  - ou do SO.