Variabilidade dos Tempos de Execução

Fundamentos dos Sistemas de Tempo Real

2ª Edição

Rômulo Silva de Oliveira

Edição do Autor, 2020

www.romulosilvadeoliveira.eng.br/livrotemporeal

Por que o tempo de execução de uma tarefa varia ?

Introdução 1/8

- Tempo de execução de uma tarefa corresponde ao tempo que ela precisa de processador para concluir
  - Considerando que a mesma está sozinha no computador.
  - Não existem outras tarefas
  - Nem mesmo tratadores de interrupções
  - Nem atividades no kernel do sistema operacional

- Isto é diferente do seu tempo de resposta
  - Inclui todas as atrapalhações que ela recebe de outras tarefas
  - e do sistema operacional

Introdução 2/8

- Suponha que uma tarefa é executada muitas vezes
- E o tempo de execução de cada ativação seja medido
- Tempo de execução varia

- Existem aspectos tanto de **software** como de **hardware** capazes de causar esta variação

Introdução 3/8

- Um aspecto fundamental é o **fluxo de controle da tarefa**, isto é, as linhas do código por onde a execução acontece
- Suponha que a tarefa tem um comando IF(EXPRESSÃO)
  - O código a ser executado caso a EXPRESSÃO seja verdadeira é composto por poucas linhas e rápido
  - O código a ser executado caso a EXPRESSÃO seja falsa é composto por muitas linhas e lento
- Mesma coisa para um comando do tipo laço onde o número de iterações é variável
  - Quanto mais iterações forem feitas no laço, a princípio maior será o tempo de execução
- As coisas ficam mais complicadas quando temos, por exemplo
  - um comando IF dentro do laço
  - um laço aninhado dentro de outro laço

Introdução 4/8

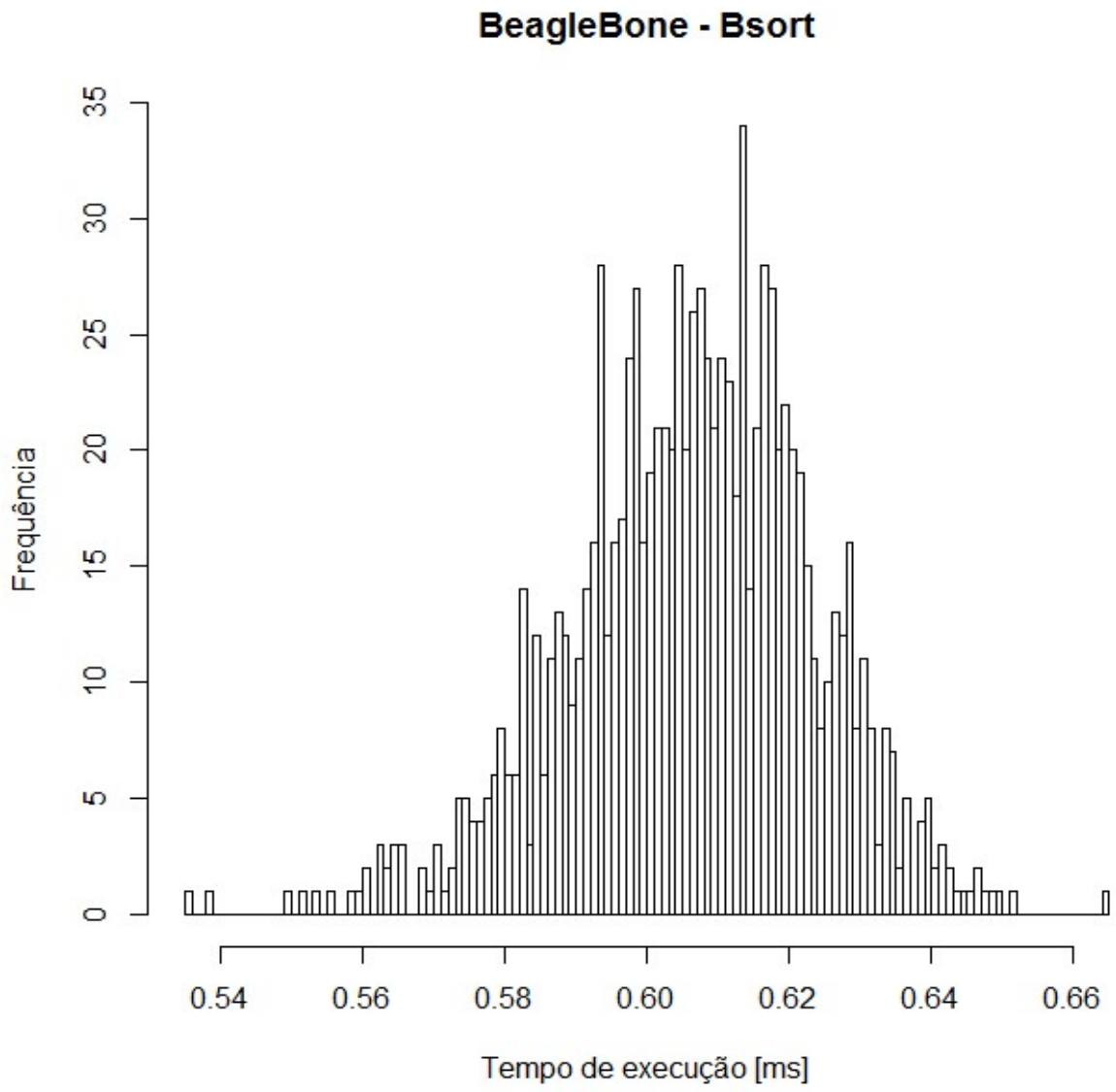
- Processadores modernos contam com vários **mecanismos de aceleração da execução**
  - Apresentam comportamento variável
  - Tempo de execução varia conforme o que foi executado antes
- Por exemplo, memórias cache
- Memória cache é uma memória mais rápida (e mais cara) que mantém dados recentemente acessados no passado
  - Se a tarefa precisar acessar em seguida um dado que está na cache (hit), o acesso será rápido
  - Se o dado não estiver na cache (miss), a memória mais lenta deverá ser acessada

Introdução 5/8

- Considere como exemplo uma tarefa programa na linguagem C
- Uma ordenação de acordo com o algoritmo Bubble Sort
- A cada ativação da tarefa ela ordena um vetor de 100 números inteiros
- A tarefa executa em uma plataforma BeagleBone White
- Após executar a tarefa 1000 vezes, cada vez com um vetor de entrada gerado aleatoriamente, foi construído um histograma

Introdução 6/8

- Tarefa executando bubble sort em uma Beaglebone
- Entradas aleatórias



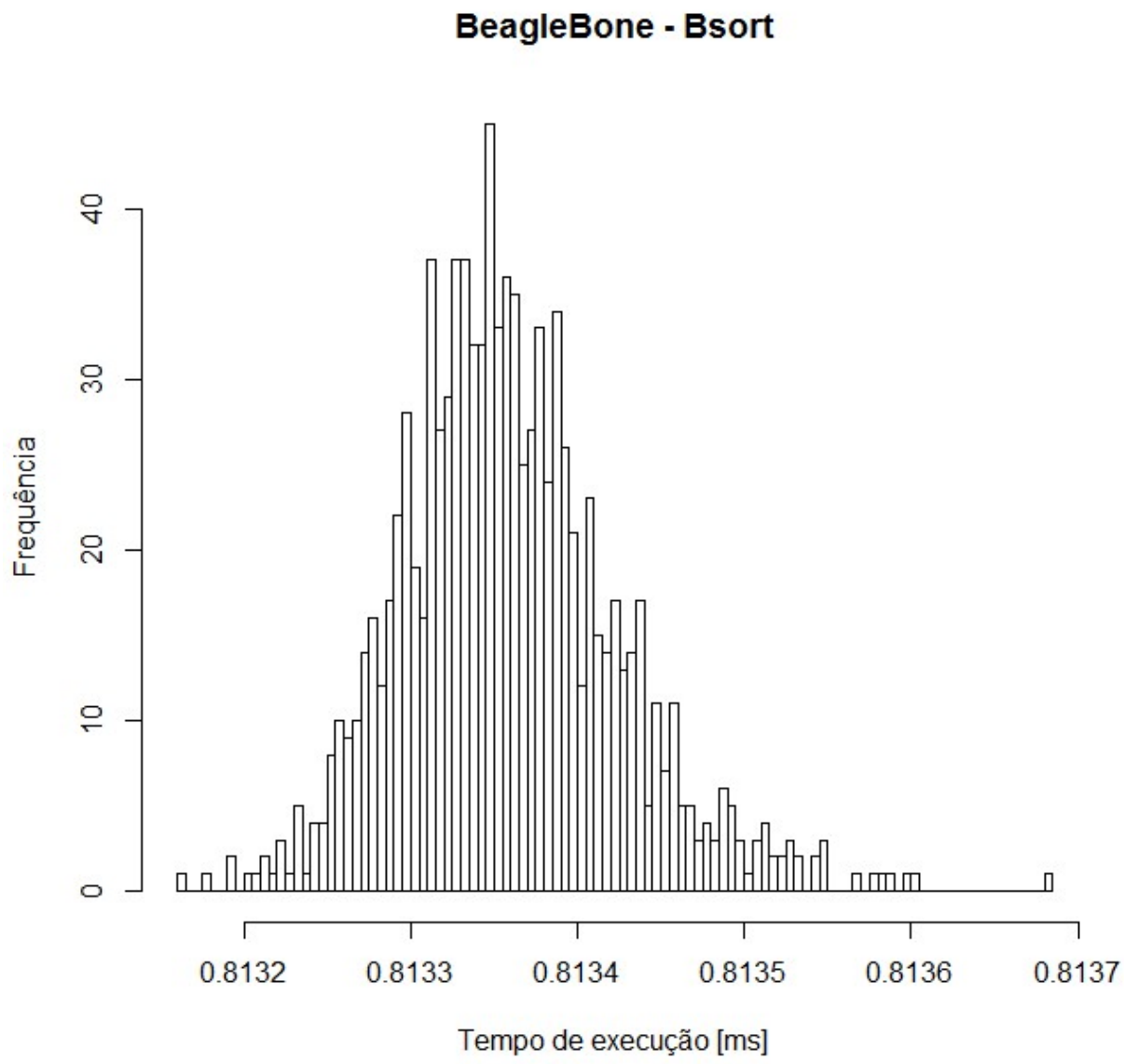
Introdução 7/8

- O que aconteceria se executássemos a tarefa 1000 vezes, porém agora ela recebe sempre o mesmo vetor de entrada ?
- Histograma mostra o resultado para 1000 ativações da tarefa
- Precisa ordenar um vetor de entrada que já está inversamente ordenado
- Este é o cenário que gera o maior número de trocas no bubble sort

- O tempo de execução não é constante
  - Lembre-se que os dados de entrada são sempre os mesmos

Introdução 8/8

- Tarefa executando bubble sort em uma Beaglebone
- Entrada invertida



Sumário

- Introdução
- Variabilidade Causada pelo Software
- Variabilidade Causada pelo Hardware

Variabilidade Causada pelo Software 1/11

- A variabilidade do tempo de execução causada pelo software está relacionada com a ideia de fluxo de controle
- O fluxo de controle da tarefa indica por onde no código da tarefa a execução passa

- Praticamente todos os programas empregam comandos do tipo IF-THEN-ELSE
- Cada vez que cada IF é executado, o fluxo de controle pode seguir por um ou outro caminho

Variabilidade Causada pelo Software 2/11

- Toda linguagem de programação inclui mecanismos para a criação de laços
- Aparecem nas mais variadas formas, tais como comandos FOR, WHILE, REPEAT-UNTIL, DO-WHILE, etc.
- Embora o número de iterações possa ser fixo no código, muitas vezes depende do valor de variáveis do programa
- Cada vez que o fluxo de controle entra no laço, seus comandos podem ser executados um número diferente de iterações

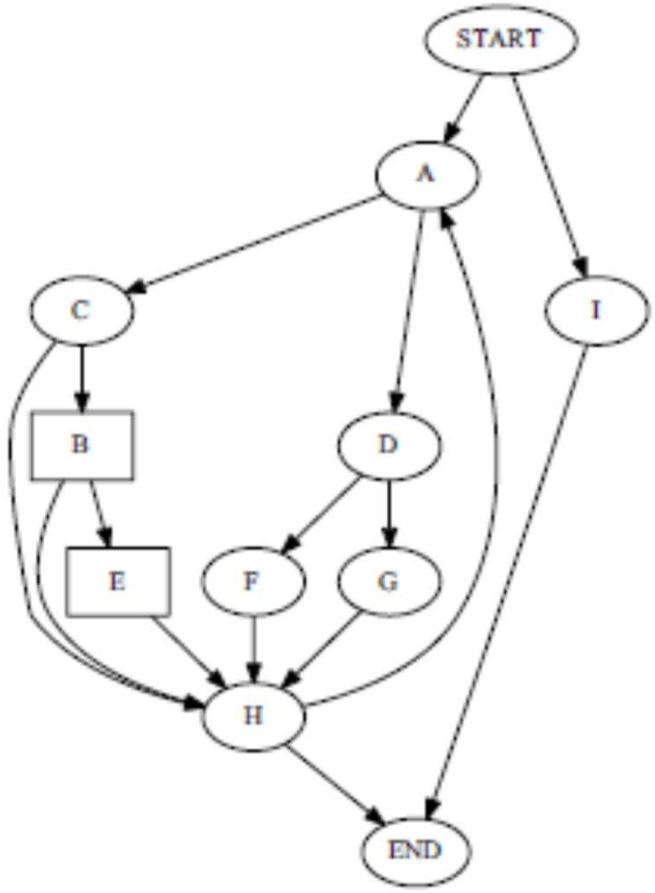
Variabilidade Causada pelo Software 3/11

- A forma usual para representar os caminhos possíveis para o fluxo de controle é o **Grafo de Fluxo de Controle**
  - **GFC**, do inglês *control flow graph*
- Exemplo:
  - START precisa ser sempre executada.
  - Se for verdadeira será executado o comando “I” e depois “END”
  - Se for falsa, a expressão “A” é avaliada
  - “A” verdadeiro o fluxo de controle segue para uma cascata de comandos IF que inclui as expressões “C” e “B” e o comando “E”
  - “A” falso a execução desce para o IF com a expressão D e os comandos “F” e “G”
  - O laço acaba no comando WHILE onde a expressão “H” é avaliada e
  - “H” verdadeiro inicia uma nova iteração do laço (flecha para cima)
  - “H” falso, “END” é executado e a tarefa termina

Variabilidade Causada pelo Software 4/11

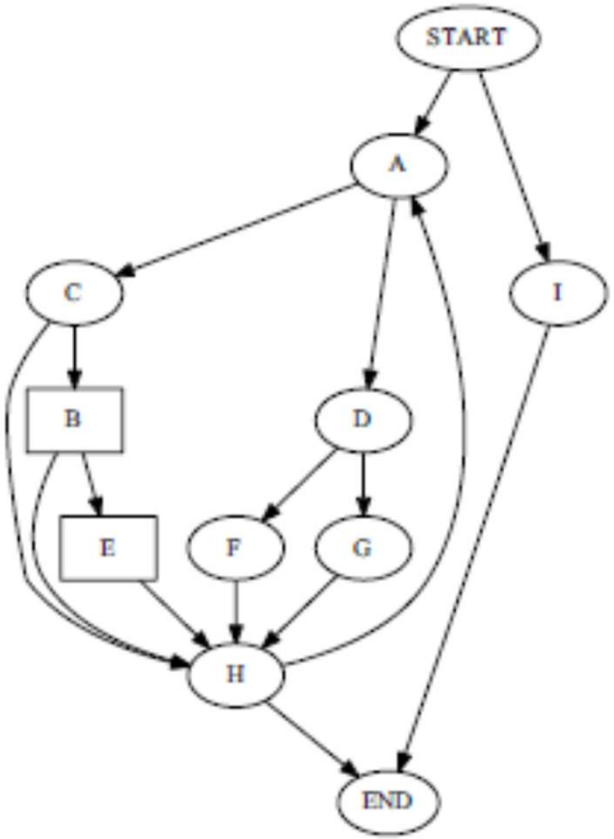
- Exemplo:
  - START precisa ser sempre executada.
  - Se for verdadeira será executado o comando "I" e depois "END"
  - Se for falsa, a expressão "A" é avaliada
  - "A" verdadeiro o fluxo de controle segue para uma cascata de comandos IF que inclui as expressões "C" e "B" e o comando "E"
  - "A" falso a execução desce para o IF com a expressão D e os comandos "F" e "G"
  - O laço acaba no comando WHILE onde a expressão "H" é avaliada e
  - "H" verdadeiro inicia uma nova iteração do laço (flecha para cima)
  - "H" falso, "END" é executado e a tarefa termina

```
if( START )
   I;
else {
   do {
      if( A ) {
         if( C )
            if( B )
               E;
         }
      else {
         if( D )
            F;
         else
            G;
         }
      }while( H );
   }
END;
```



Variabilidade Causada pelo Software 5/11

- Quantos caminhos diferentes existem entre START e END ?
  - Ramo da direita faz I
  - Ramo da esquerda tem um laço contendo 5 ramos:
    - A D F H
    - A D G H
    - A C H
    - A C B H
    - A C B E H
  - Precisa um limite para o número de iterações do laço



Variabilidade Causada pelo Software 6/11

- Precisa um limite superior para o número de iterações do laço
  - Vamos supor 6, repete o laço de 1 a 6 vezes

  - Executa o laço 1 vez: 5^1 possibilidades
  - Executa o laço 2 vezes: 5^2 possibilidades
  - Executa o laço 3 vezes: 5^3 possibilidades
  - Executa o laço 4 vezes: 5^4 possibilidades
  - Executa o laço 5 vezes: 5^5 possibilidades
  - Executa o laço 6 vezes: 5^6 possibilidades

Variabilidade Causada pelo Software 7/11

- Número de caminhos do laço: $\sum_{i=1}^{M}(N)^i$
  - Limite do laço são M vezes
  - Número de ramos no corpo do laço é N

- Dominado por N^M

- Por exemplo, N=4 e M=100
  - Resulta em 4^{100} o que é aproximadamente 10^{60} !!!
  - Completamente intratável processar cada caminho explicitamente

Variabilidade Causada pelo Software 8/11

- **Quantos caminhos existem ?**

- Sem desvio nem laço
  - Apenas 1 caminho

- Com desvios mas sem laços
  - Um certo número de caminhos, depende da combinação dos desvios
  - Porém ainda um numero tratável explicitamente (não são muitos)

- Sem desvios mas com laços
  - Um certo número de caminhos, depende do número de iterações
  - Porém ainda um numero tratável explicitamente (não são muitos)

- Com desvios e com laços
  - Um imenso número de caminhos (ex: laço de 100 com 4 ramos, 4^100)
  - Número intratável explicitamente (ex: 10^60)
  - Pode ter laços dentro de laços

Variabilidade Causada pelo Software 9/11

- O que define qual caminho é executado ?
  - Variáveis de entrada do programa
  - Variáveis globais alteradas em execuções anteriores
  - Data e hora correntes
  - Geração de números aleatórios

- Estes são os caminhos sintaticamente possíveis

- Alguns desses caminhos são semanticamente impossíveis
- Impossível pela semântica do programa
  - Análise de valor pode identificar (parcialmente)
- Impossível pela semântica do ambiente
  - Entradas impossíveis de acontecer na prática

Variabilidade Causada pelo Software 10/11

- IF(A > B)
  THEN X;
  ELSE Y;
  IF(A > B)
  THEN Z;
  ELSE W;

- A princípio 4 possibilidades:
  - “X Z” “X W” “Y Z” “Y W”
- Olhando as expressões nos dois comandos IF, somente 2 caminhos são possíveis:
  - “X Z” quando A>B
  - “Y W” quando A<=B
  - Os caminhos “X W” e “Y Z” são semanticamente impossíveis.
- É comum a existência de caminhos impossíveis, mas não a ponto de mudar a natureza da questão, que é a explosão do número de caminhos

Variabilidade Causada pelo Software 11/11

- Se uma tarefa possui apenas um caminho, então o tempo de execução dela é constante ?
- Infelizmente não é garantido
- Quando a tarefa tem apenas um caminho, então não existe variação do tempo de execução causada pelo software
- Porém, pode ainda existir muita variação do tempo de execução causada pelo hardware

- Mesmo que exatamente as mesmas instruções de máquina sejam sempre executadas pela tarefa
  - a cada execução o tempo de execução será diferente

Sumário

- Introdução
- Variabilidade Causada pelo Software
- Variabilidade Causada pelo Hardware

Variabilidade dos Tempos de Execução

Fundamentos dos Sistemas de Tempo Real

2ª Edição

Rômulo Silva de Oliveira

Edição do Autor, 2020

www.romulosilvadeoliveira.eng.br/livrotemporeal

Parte II: Variabilidade Causada pelo Hardware

Variabilidade Causada pelo Hardware 1/4

- Mesmo quando uma tarefa executa exatamente as mesmas instruções de máquina
  - o seu tempo de execução pode variar
  - ou não
  - dependendo das características do processador

- Nos processadores mais antigos
  - Tempo necessário para executar cada instrução de máquina corresponde a um número inteiro de ciclos de clock
  - Basta inverter o valor da frequência do processador em Hertz
  - Obter a duração do ciclo de clock em segundos
  - E multiplicar pelo número de ciclos de clock necessários
  - Temos quanto tempo demora uma instrução de máquina

Variabilidade Causada pelo Hardware 2/4

- Microcontrolador Intel MCS-51 (Intel 8051)
  - Faz parte de uma família de microcontroladores (*single chip microcontroller*)
  - 8 bits
  - Lançada em meados de 1980 para uso em *embedded systems*
- "MCS 51 MICROCONTROLLER FAMILY USER'S MANUAL"
  - Quando uma frequência de clock de 12MHz é empregada
  - Instrução de máquina **ADD A,<byte>** (soma ao acumulador um valor imediato de 8 bits) demora sempre 1 microsegundo
  - Instrução de máquina **MUL AB** (multiplicação inteira de dois registradores) demora sempre 4 microsegundos
  - **MOV <dest>,<src>** (cópia de um byte entre dois registradores) demora sempre 2 microsegundos.
  - **JZ rel** (jump condicional se zero) demora sempre 2 microsegundos
  - Etc

Variabilidade Causada pelo Hardware 3/4

- No caso do microcontrolador 8051 da Intel, o hardware não introduz variabilidade no tempo de execução da tarefa
- Se as mesmas instruções de máquina são executadas, o mesmo tempo de execução será obtido

- Processadores mais modernos empregam uma gama de mecanismos de hardware que aceleram a execução dos programas
  - Apresentando um comportamento probabilista
- Tais mecanismos tornam a execução das instruções de máquina mais rápida
  - Porém o tempo de execução de uma instrução de máquina é variável

Variabilidade Causada pelo Hardware 4/4

- Exatamente quais mecanismos de aceleração são empregados varia de processador para processador
- Entre os mais importantes podemos citar:
  - Memória cache
  - Pipeline
  - Branch predictor
  - Memórias DRAM (Dynamic Random Access Memory)
  - DMA (Direct Memory Access)
  - TLB (Translation Lookaside Buffer)
- Por exemplo, o processador **ARM Cortex-M0** emprega um pipeline simples porém não emprega branch predictor
- **Intel Core i7** emprega um pipeline muito mais sofisticado, branch predictor e mais uma vasta gama de mecanismos
  - Tempo de execução das instruções de máquina consideravelmente variável

Sumário

- Introdução
- Variabilidade Causada pelo Software
- Variabilidade Causada pelo Hardware
  - Memória Cache
  - Pipeline
  - Branch Predictor
  - Memórias DRAM
  - Acesso Direto à Memória – DMA
  - Translation Lookaside Buffer – TLB
  - Controle de Frequência
  - Modo de Gerência do Sistema
  - Múltiplas Threads em Hardware
  - Impacto dos Tratadores de Interrupção e de Múltiplas Tarefas