-

Atividades 3 e 4

Pedro Cunial

February 13, 2017

Contents

1 Cross-Compiler

1.1 O que é cross compilação (cross-compiler)?

Em suma, cross compilação significa compilar um código para uma plataforma diferente da utilizada, por exemplo, compilar um código para uma arquitetura ARM em um x86. Isso é feito pelo chamado cross-compiler (ou compilador cruzado).

2 Embarcados

2.1 O que é RTOS, descreva uma utilização.

RTOS, ou Sistema Operacional de Tempo Real (Real Time Operating System) é um SO voltado à execução de multiplas tarefas cujo tempo de resposta é predefinido e constante, independente de possíveis graus de importância ou relevância do evento.

Desta forma, podemos considerar como fatores chave para a definição de um RTOS a menor existência possível de latências em interrupções e de alternância de tarefas.

RTOS são pricipalmente utilizados em computadores de bordo (ou embarcados) externos, onde um terceiro precisa ter o controle sobre eventos e possíveis interrupções ou falhas no mesmo.

2.2 O que é desenvolvimento de projetos em V (Modelo V)?

O modelo V é um modelo de organização de projetos que surgira com o intuito de substituir o tradicional modelo em cascata nas engenharias de sistemas e desenvolvimento. O modelo V baseia-se na divisão do projeto confiando no sucesso da sua integração pela implementação de tests em cada "subprojeto".

2.3 O que é um DSP? O que difere de um uC?

O DSP (Digital Signal Processor) é um microprocessador especializado no processamento de sinais externos, como de audio, vídeo etc em tempo real.

Ele difere de um uC pela sua velocidade de resposta e simplicidade em geral, uma vez que não vai muito além de um uP, atentando-se somente à uma tarefa, normalmente sendo alguma conversão de sinal.

3 C

3.1 Descreva a funcionalidade do:

3.1.1 Compilador C

O compilador C é encarregado de, principalmente, traduzir um código C para linguagem de máquina, apesar de normalmente também realizar pequenas otimizações no código. O compilador têm várias etapas, sendo o assembler e a linkagem exemplos delas.

3.1.2 Assembler

É um tipo específico de compilador mais simples que traduz código assembly para linguagem de máquina. Um assembler é muito mais simples do que um compilador uma vez que o código assembly é quase uma sequência de "alias" para o verdadeiro código de máquina.

3.1.3 Linker

O linker serve para unir arquivos objetos C em um arquivo executável final. O linker costuma ser a parte final da compilação de um código C.

3.2 Qual a diferença entre C e C++?

A linguagem C é uma linguagem muito próxima da própria linguagem de máquina, permitindo poucas abstrações de alto-nível, como em C++. Por exemplo, enquanto C++ possui o conceito de classes, objetos, variáveis privadas etc, a linguagem C atem-se somente à conceitos básicos como structs. Além disso, o C++ possui algumas facilidades de sintaxe, como a possibilidade de definir variáveis que serão passadas por referência, enquanto em C é necessário passar o endereço de uma variável em uma função que exige um apontador da mesma para obter o mesmo resultado.

4 Paralelismo vs Concorrência

Em sistemas embarcados, a performanse pode ser o diferêncial entre um projeto ser ou não viável; Sendo o tempo de resposta de um microcontrolador à um extimulo externo quase sempre uma operação que não pode levar mais do que microssegundos, além de não poder impedir o funcionamento da tarefa principal do programa.

Mas como isso seria possível? Existem duas principais maneiras de programarmos tarefas que ocorem ao mesmo tempo, sendo elas o paralelismo e a concorrência.

Enquanto o paralelismo baseia-se na distribuição de micro-tarefas e pequenas operações lógicas à dispositimos periféricos ou diferentes do principal (como conversores, microprocessadores ou cores diferentes), a concorrência na verdade ocorre mais como uma abstração, rodando todos os processamentos simultâneos no mesmo core.

Para sistemas embarcados, o paralelismo é essencial para um bom desempenho de uma aplicação. O processador principal de um sistema embarcado

deve ser encarregado principalmente de operações lógicas, como comparações e a distribuição de comandos para seus periféricos baseado nos resultados destas comparações. Repare que não contempla no escopo do core principal a conversão de sinais em valores realmente processaveis para o mesmo, uma vez que esta operação não depende de nenhum outro processamento (podendo ser executada simultaneamente com outros semelhantes).

A programação concorrente é muito mais comum em linguagens de auto-nível. Um bom exemplo disso é em sistemas web, onde a concorrência esta tão enraizada que frameworks que são nativamente concorrentes têm, cada vez mais, crescido em popularidade entre desenvolvedores. Por exemplo, imagine que um usuário faça uma requisição em um sistema web, a qual desencadeia o envio de um email para o mesmo. Realizar o envio deste email pela thread principal do programa simplesmente não faz sentido, uma vez que ela travaria o funcionamento de todo o sistema para o simples envio de um email. Ao mesmo tempo, não faz sentido termos um periférico exclusivamente para o envio de emails e por isso preferimos o uso de concorrência em casos como este.

Existem diversos motivos e vantagens de utilizar programação paralela ou concorrente e cabe ao programador decidir qual a melhor opção para o seu código, seja ele de uma aplicação de alto-nível de abstração ou de baixo.