Pontifícia Universidade Católica do Paraná
Escola Politécnica
Bacharelado em Ciência da Computação / Cibersegurança

Curitiba, 05, abril de 2021.
Disciplina: Sistemas Operacionais Ciberfísicos
Professor: Jhonatan Geremias
Curso: <nome do curso>
Nome: <nome do estudante>

Projeto FreeRTOS - Fase 2
Tarefas FreeRTOS

Descrição da Atividade:

O projeto é estruturado em duas etapas. Esta atividade compreende o conceito do uso de tarefas no FreeRTOS.

Entrega:

Esta atividade deverá ser entregue até o dia 26/04/2022 no Canvas.

O estudante deverá implementar o código utilizando o FreeRTOS, seguindo a especificação definido neste roteiro. O código deverá ser implementado apenas dentro do arquivo "example.c".

Entregar apenas o arquivo "example.c" onde foi codificado, este deve conter o nome dos estudantes e curso adicionados no cabeçalho do arquivo como comentário. A atividade poderá ser realizada em equipe, com no máximo 3 estudantes.

Roteiro da Atividade:

Contexto:

Para que um quadricóptero possa voar é necessária a existência de um diferencial de pressão entre a parte superior e a parte inferior de suas quatro hélices, este diferencial é produzido pela rotação delas. Conforme a velocidade de rotação aumenta, a pressão sobre as hélices diminui, surgindo assim um vetor resultante de força para cima, esta força é denominada empuxo. Conforme o empuxo aumenta, diminui ou iguala-se a força peso, o quadricóptero subirá, descerá ou permanecerá em voo pairado, respectivamente (SILVA, 2011). Para manter-se o equilíbrio as hélices de um quadricóptero são posicionadas em paralelo, no mesmo plano horizontal e equidistantes do centro de massa (SILVA, 2011). O torque total produzido pelas quatro hélices deve resultar em nulo. Se isso não ocorrer, o equilíbrio e manobrabilidade da aeronave é comprometido. Para se anular o torque um par de hélices deve girar no sentido horário e o outro no sentido anti-horário.

Com o controle independente da velocidade de rotação de cada motor, um quadricóptero pode realizar três manobras básicas: guinada (yaw, eixo vertical), arfagem (pitch, eixo lateral) e rolagem (roll, eixo longitudinal). A execução das manobras ocorre por meio de alterações na velocidade de rotação de seus motores, aumentando a velocidade de um par e diminuindo a de outro. Para que o quadricóptero não tenha sua altitude alterada durante a execução de um comando específico é necessário que o empuxo permaneça constante. Para satisfazer tal condição deve-se aumentar e diminuir a rotação dos pares de motores de maneira proporcional.

Dados conhecidos:

A seguir são apresentados os conceitos básicos para se efetuar as manobras mencionadas:

a. guinada:
   - sentido horário: aumentar a velocidade dos motores 0 e 2, diminuir a velocidade dos motores 1 e 3;
   - sentido anti-horário: diminuir a velocidade dos motores 0 e 2, aumentar a velocidade dos motores 1 e 3;

b. arfagem:
   - mover para frente: aumentar a velocidade dos motores 2 e 3, diminuir a velocidade dos motores 0 e 1;
   - mover para trás: diminuir a velocidade dos motores 2 e 3, aumentar a velocidade dos motores 0 e 1;

c. rolagem:
   - virar à direita: aumentar a velocidade dos motores 0 e 3, diminuir a velocidade dos motores 1 e 2;
   - virar à esquerda: diminuir a velocidade dos motores 0 e 3, aumentar a velocidade dos motores 1 e 2.

Supondo que se deseja conceber o software para o controle de um quadricóptero, em que é necessário realizar controles de arfagem, rolagem e guinada. Para estabilizar o quadricóptero no ar depende da realização das três operações. Assume-se que os requisitos de tempo para esses controles são:

- O controle de arfagem deve ser realizado 25 vezes por segundo (período de 40 ms);
- O controle de rolagem deve ser realizado 50 vezes por segundo (período de 20 ms);
- O controle de guinada deve ser realizado 100 vezes por segundo (período de 10 ms).

Especificação do projeto – Fase 1:

Neste projeto vamos implementar um conjunto de tarefas para simular o comportamento das manobras realizadas por um quadricóptero. Segue a especificação do projeto.

a. Implementar um programa no FreeRTOS destinado a controlar um software de quadricoptero;
b. Nesta fase do projeto deverá ser criado três tarefas, cada tarefa é responsável por uma manobra no quadricóptero (arfagem, rolagem e guinada);
c. A criação das três tarefas deve ser realizada no main_;
d. Deverá ser criada quatro variáveis que irão simular a velocidade de cada um dos motores do quadricóptero;
e. A Tarefa criada para manobra de guinada deve verificar se o sentido é horário ou anti-horário, alterar a velocidade dos motores correspondente a manobra da guinada, neste sentido incrementar ou decrementar a variável em 100 vezes conforme o sentido;
f. A Tarefa criada para manobra de arfagem deve se mover para frente e para trás, alterar a velocidade dos motores correspondente a manobra de arfagem, neste sentido incrementar ou decrementar a variável em 25 vezes conforme a direção;
g. A Tarefa criada para manobra de rolagem deve se mover para esquerda e para direita, alterar a velocidade dos motores correspondente a manobra de rolagem, neste sentido incrementar ou decrementar a variável em 50 vezes conforme a direção;
h. Todas as tarefas criadas devem ter a mesma prioridade;

i. Efetuar a passagem de parâmetros para cada tarefa: guinada (horário ou anti-horário), arfagem (para frente ou para trás) e rolagem (esquerda ou direita);
j. A tarefa deve fornecer como saída no console a manobra que está sendo realizada, ação que foi passada por parâmetro para função e a velocidade dos motores – utilizar as funções vPrintString() e printf();
k. Utilizar a função vTaskDelay() configurando o tempo necessário para cada tarefa, arfagem(40ms), rolagem(20ms) e guinada (10ms).
l. Todas tarefas devem definir sua exclusão explicita utilizando a tarefa vTaskDelete();
m. O código deve ser documentado, utilizar os comentários em toda a extensão do programa.

Especificação do projeto – Fase 2:

Esta é a segunda etapa do projeto, consiste em otimizar o código estruturado na fase 1. Nesta etapa o objetivo é aplicar os mecanismos que permitem garantir o fluxo corrente das Tarefas responsáveis pelas manobras do quadricóptero. Segue a especificação da fase 2 do projeto.

a. Criar uma estrutura semáforo do tipo binário, utilizar a função vSemaphoreCreateBinary();
b. Nesta fase do projeto você deverá modificar as três tarefas responsáveis pelas manobras no quadricóptero (arfagem, rolagem e guinada) para utilização da estrutura do semáforo binário;
c. Cada tarefa deve garantir o uso exclusivo do acesso as variáveis que simulam a velocidade dos motores do quadricóptero, utilizar as funções xSemaphoreTake() e xSemaphoreGive();
d. Alterar o escopo das variáveis locais que recebem como parâmetro as operações para tarefas (guinada, arfagem e rolagem), tais variáveis devem ser definidas como variáveis globais (sentido, direção e orientação);
    - A variável “sentido” para manipular a tarefa responsável pela guinada deve alterar o sentido horário ou anti-horário;
    - A variável “direção” para manipular a tarefa responsável pela arfagem deve alterar a direção para frente ou para trás;
    - A variável “orientação” para manipular a tarefa responsável pela rolagem deve alterar a orientação direita e esquerda;

e. Criar uma quarta tarefa rádio frequência, esta tarefa será responsável por alterar as manobras do quadricóptero;
   - A Tarefa rádio frequência deve ter prioridade inferior as outras tarefas de manobra;
   - Para alterar as manobras deverá ser modificado o valor das variáveis "sentido", "direção" e "orientação";
   - O primeiro valor atribuído as variáveis "sentido", "direção" e "orientação" é recebido como parâmetro pela função xTaskCreate() correspondente a cada tarefa, na sequência deverão ser alteradas apenas na execução da tarefa "rádio frequência" ;
   - Esta função também deverá utilizar semáforos, garantir o acesso exclusivo as variáveis globais ("sentido", "direção" e "orientação");
   - Todas as variáveis globais devem utilizar o modificador de acesso volatile;
   - Para as variáveis do tipo string definir um tamanho fixo (Ex.: char sentido[10];), evitar alocação dinâmica de recurso;

```
// Variáveis para armazenar a velocidade dos motores
volatile long velocidadeMotor0;
volatile long velocidadeMotor1;
volatile long velocidadeMotor2;
volatile long velocidadeMotor3;

volatile char sentido[10];
volatile char direcao[10];
volatile char orientacao[10];
```

   - As manobras devem ser alteradas aletoriamente, a cada execução da tarefa rádio frequência será sorteado três números diferentes (um para cada variável);
   - Sortear os valores na faixa de 0 a 100, (utilizar a função rand() – exemplo: x = rand() % 100;);
   - Verificar individualmente se o número sorteado para cada uma das três variáveis é par ou ímpar;

```
x =  rand() % 100;
y = rand() % 100;
z = rand() % 100;

if (x % 2 == 0) {
    sprintf(sentido, "horario");
}
else{
    sprintf(sentido, "antihorario");
}
```

- Para os números pares atribuir o seguinte valor para variável: sentido=“horario”, direção=”frente” e orientação=”direita”);
- Para os números ímpares considerar (sentido=“antihorario”, direção=”trás” e orientação=”esquerda”);
- Utilizar a função sprintf() para atribuir um novo valor a variável do tipo string.

n. Modificar o tempo de atraso das tarefas para obter o tempo em Tick por milissegundos;
   - Utilizar a constante portTICK_RATE_MS, multiplicar o valor da constante pelo valor definido na função vTaskDelay() conforme especificação do item “k” especificado na fase 1;
   - Exemplo: vTaskDelay (portTICK_RATE_MS * 40ms);

o. A tarefa de rádio frequência deve gerar um atraso de 100ms em Tick, semelhante as demais tarefas.

p. O código deve ser documentado, utilizar os comentários em toda a extensão do programa.