CHAVE DE CONEXÃO

PONTO 2.1

TIME-TRIGGERED SYSTEMS E EVENT-TRIGGERED SYSTEMS

DESCREVEM SISTEMAS DE TEMPO REAL BASEADO EM TEMPO E EM EVENTOS. MOSTRAM VANTAGENS E DESVANTAGENS. MOSTRAM UMA IMPLEMENTAÇÃO EM C DE UMA BIBLIOTECA PARA IMPLEMENTAÇÃO DE UM SISTEMA MULTITAREFA BASEADO EM TEMPO BEM COMO DE UM EXEMPLO. IMPLEMENTAM O MESMO EXEMPLO USANDO A API PROPORCIONADA POR UM KERNEL DE TEMPO REAL

SOLUÇÃO

BASEADO EM TEMPO

BASEADO EM VARREDURA PERIÓDICA

NECESSITA APENAS UM TEMPORIZADOR

TAREFAS SÃO RUN-TO-COMPLETION

BASEADO EM EVENTOS

REAGE A EVENTOS QUE PRODUZEM INTERRUPÇÕES

NECESSITA DE MECANISMO DE INTERRUPÇÃO NO PROCESSADOR

GERALMENTE É USADO COM UM CONTROLADOR DE INTERRUPÇÕES QUE PERMITEM TER VÁRIOS NÍVEIS DE PRIORIDADE

(2)

VANTAGENS DO TTS

ESCALONADOR MAIS SIMPLES

DEMANDA APENAS UMA PILHA, EXIGINDO BEM MENOS RAM

CONCEITUALMENTE MAIS SIMPLES

NÃO EXISTEM PROBLEMAS DE "CORRIDAS"

NÃO DEMANDAM MECANISMOS DE EXCLUSÃO MUTUA

COMPORTAMENTO DETERMINISTICO

DESVANTAGENS

DIFICIL DE MODELAR (O TEMPO DE EXECUÇÃO DE TAREFAS MENOR QUE O PERIODO DE ESCALONAMENTO

MENOS FLEXÍVEL

PODE SER MAIS LENTO (RESPOSTA)

VANTAGENS DO ETS

MAIS FLEXIVEL MAS MAIS COMPLEXO

ESCALONAVEL

DESVANTAGENS

NECESSITA UMA PILHA PARA CADA TAREFA QUE DEVE TER TAMANHO MÁXIMO

OCORREM "CORRIDAS" SIMULTANEAMENTE DEMANDANDO OS MECANISMOS DE SINCRONISMO

IMPLEMENTOS BEM MAIS COMPLEXA MAIOR (ROM E RAM

IMPLEMENTAÇÃO TTS

```
typedef struct {
        void (*func)(void)
        uint periodo;
        uint contador;
        uint ~~[illegible]~~ run;
     } ~~[illegible]~~ TCB_t;
TCB_t TCB[10];
```

DISPATCH EXECUTADO NO main

```
void dispatch(void) {    // verifica posição ocupada
    for(int i=0; i<10; i++) {
        if(TCB[i].~~contador~~ run == 0) {
            TCB[i].func();
            TCB[i].~~contador~~ run = TCB[i].contador;
        } else {
            TCB[i].~~contador~~ run--;
        }
    }
```

UPDATE EXECUTADO ~~NA~~ NA INTERRUPÇÃO PERIÓDICA (SysTick_Handler)

```
void update(void) {

    for(~~int~~ i =0; i<10; i++) {
        TCB[i].contador--;
          // evitar valor <0
    }
}
```

(4)

EXEMPLO

// pisca leds com freq. diferentes

```
void main() {

        Init();
        AddTask(0, FuncA, 10, 1);   // periodo, atraso
        AddTask(1, FuncB, 17, 1);

        while(1) {
                Dispatch();
        }

}

void FuncA() {
        ToggleLED(1);
}
void FuncB() {
        ToggleLED(2);
}
```

// poderiam ser PWM, etc.

USANDO ETS

```
OSSTK StackA[1000];
OSSTK StackB[1000];
void Main() {

OSInit();

OSAdd(FuncA, 10, StackA, 2000);
OSAdd(FuncB, 13, StackB, 1000);

OSStart();

}

void TaskA(void) {

while(1) {
    LED_Toggle(1);
    Delay(10);
}

}

void TaskB(void) {

while(1) {
    LED_Toggle(2);
    Delay(13);
}

}
```

UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO

PROTOCOLO DE ASSINATURA

O documento acima foi assinado digitalmente com senha eletrônica através do Protocolo Web, conforme Portaria UFES nº 1.269 de 30/08/2018, por
MARIA DA CONCEICAO CARNEIRO DA SILVA - SIAPE 1996445
Departamento de Engenharia Elétrica - DEE/CT
Em 05/05/2025 às 17:13

Para verificar as assinaturas e visualizar o documento original acesse o link: https://api-lepisma.prod.uks.ufes.br/arquivos-assinados/1122224?tipoArquivo=O