Especificação, Modelação e Projecto de Sistemas Embutidos

Handson Session

O simulador TrueTime

Paulo Pedreiras
pbrp@ua.pt

Departamento de Electrónica, Telecomunicações e Informática
Universidade de Aveiro

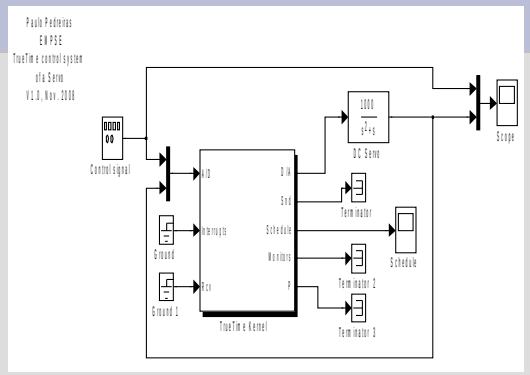
V1.0 Novembro/2008

O simulador TrueTime

- Toolbox do Matlab/Simulink
- Simulação de:
  - **Kerneis** de tempo-real
  - **Redes** wired e wireless
  - Outras **propriedades não funcionais**
    - e.g. bateria, *dynamic voltage-scaling*, …
- Desenvolvido em Lund, Suécia, desde 1999
  - Comunidade de utilizadores alargada
  - *Open Source*
  - *Freeware*

Modelo de Computação

Modelo de Computação



- Simula um kernel baseado em eventos
- Permite ao utilizador criar **tarefas** e ***interrupt handlers***
  - C/C++ ou M-files
- **Diversos algoritmos** de escalonamento de tarefas
- *Set* completo de primitivas
  - **Gestão de tarefas** (criação, activação, prioridades, deadlines, tempos de exec., timers, ...
  - **Mecanismos de IPC e sincronização** (Mailbox, Semáforos, monitores, …)
- Código estruturado em **segmentos**

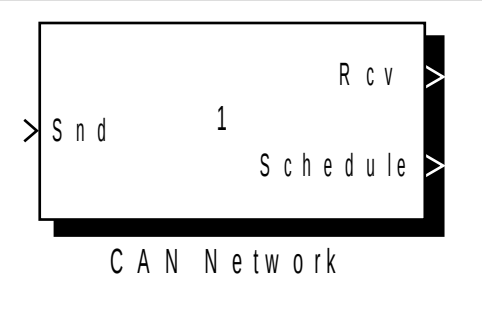
```
function [exectime,data] =
sensortask(seg,data)

 switch seg,
     case 1,
         data.y = ttAnalogIn(1);
         exectime = 0.0001;
     case 2,
         ttSendMsg(2, data.y, 65);
         exectime = 0.0002;
     case 3,
         exectime = -1; % finished
end
```

Modelo de Comunicação

Modelo de Comunicação

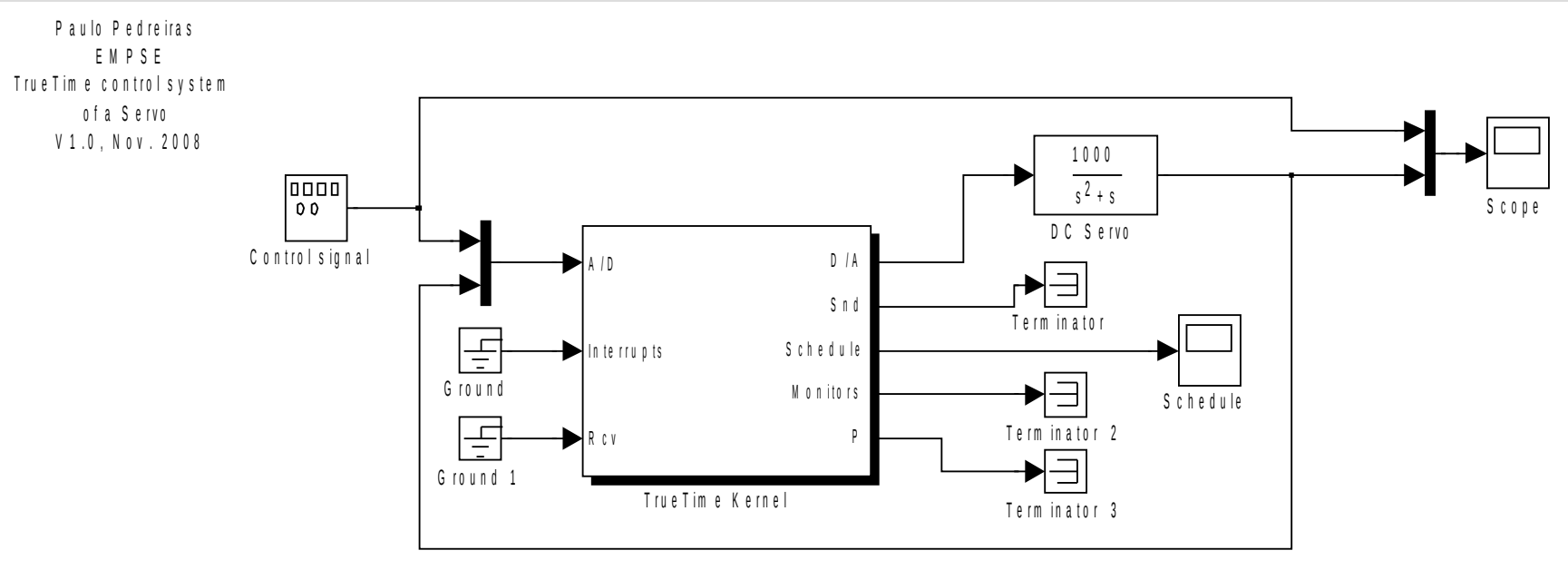
- Suporta **diversos protocolos** de comunicação (wired)
  - Ethernet
  - Switched Ethernet
  - CSMA/NBA (CAN)
  - Round Robin
  - TDMA (TTP)
  - FlexRay, TT-CAN



- Modela o **atraso** no acesso ao meio e o **tempo** de **transmissão**
- **Recepção** de mensagens associado a uma **excepção** (interrupção)
- Suporta ainda redes wireless (Wi-Fi, ZigBee, ultra-sons, ...). Não tratado no âmbito deste curso!

Exemplo: modelação de um sistema centralizado monolítico

Objectivo: pretende-se modelar o controlo de um servo. O controlo é **centralizado** (baseado numa única unidade de processamento) e **monolítico** (uma única tarefa)



Exemplo: modelação de um sistema centralizado monolítico

Como modelar o sistema?

- Abrir a biblioteca do TrueTime
  - Na janela do Matlab **digitar "truetime"**
- **Criar** um novo **modelo** ("File->New Model")
- Arrastar um "**True Time Kernel**" (TTK) para o novo projecto
- Cada **TTK** tem sempre associada uma **função de inicialização** que é executada **uma única vez** quando é efectuada uma **simulação**
- Efectuar um "duplo click" no TTK e definir um nome para a função (e.g. "csystem_init.m")
  - Nesta caixa de diálogo é ainda possível passar um argumento à função, configurar o relógio (*offset* e *drift*, indicar se o sistema é alimentado a baterias)

Exemplo: modelação de um sistema centralizado monolítico

- Adicionar o gerador de sinais, a função de transferência …
  - **Não é necessário efectuar estes passos**! O modelo encontra-se disponível na página da disciplina!
    "O Prof. É porreiro pá" :-)

- Agora é necessário criar a função de inicialização. Esta deve:
  - Inicializar o *kernel*
    - **TtInitKernel()**
  - Inicializar variáveis
  - Criar a(s) tarefa(s)
  - Instalar os *interrupt handlers* (quando necessário)
  - Inicializar a rede (quando necessário)

Exemplo: modelação de um sistema centralizado monolítico

Função de inicialização

```
function servo_init()

% Initialize TrueTime kernel
ttInitKernel(2, 1, 'prioFP'); % nbrOfInputs, nbrOfOutputs, fixed
   priority
...

% Task arguments (local data)
cfgdata.KP = 0.3;      % Proportional gain
...

% Global vars, for IPC
global DATA
DATA.yold = 0;       % Old output value;
...

% Create the control task
ttCreatePeriodicTask('pid_task', offset, period, prio,
   'pidtask',cfgdata);
...

end
```

Código integral disponível na página da disciplina!

Exemplo: modelação de um sistema centralizado monolítico

E finalmente a função de controlo

```
function [exectime,cfgdata] = pidtask(seg,cfgdata)
global DATA
switch seg,

 case 1,
  DATA.r = ttAnalogIn(cfgdata.rChan); % Read reference
  DATA.y = ttAnalogIn(cfgdata.yChan); % Read process output
  pidcalc(cfgdata); % Calculate PID action
  exectime = 0.002;
 case 2,
  ttAnalogOut(cfgdata.uChan, DATA.u); % Output control
   signal
  exectime = -1;
end
```

Código integral disponível na página da disciplina!

Exemplo: modelação de um sistema centralizado monolítico

Procedimento:

- **Descarregar** os ficheiros disponíveis na página da disciplina (modelo funcional)
- **Estudar** com cuidado o ficheiro de inicialização e a tarefa de controlo
- **Simular** o sistema
  - Observe o sinal de referência e o sinal de saída no osciloscópio
- **Experimente** variar (aumentar e reduzir) o período da tarefa de controlo e observe a qualidade do controlo resultante

Tarefa 1: modelação de um sistema centralizado

Tarefa 1

- Na prática é frequente particionar-se o sistema em módulos/tarefas. Modifique o código fornecido, criando **três tarefas periódicas** concorrentes, associadas respectivamente a:
  - **T1** : leitura dos sinais (referência e saída da planta)
  - **T2** : cálculo do valor de actuação
  - **T3** : actuação no sistema

  - **Nota**: use variáveis globais para efectuar a comunicação entre as diversas tarefas

Tarefa 1: modelação de um sistema centralizado

Tarefa 1 (cont)

- Simule o sistema com as seguintes **ordens de prioridades**
  - Prio(T1)>Prio(T2)>Prio(T3) e Prio(T3)>Prio(T2)>Prio(T1)
- Observe e comente o desempenho de controlo em ambos os casos
  - Sugestão: observe o **escalonamento** (osciloscópio "schedule")
- Crie **duas** outras **tarefas** (**carga**) com tempo de execução 0.001s e período idêntico à tarefa de controlo.
  - Simule o sistema com estas tarefas tendo prioridade superior e prioridade inferior às tarefas associadas ao controlo do servo
  - Comentar os resultados

Tarefa 2: modelação de um sistema distribuido

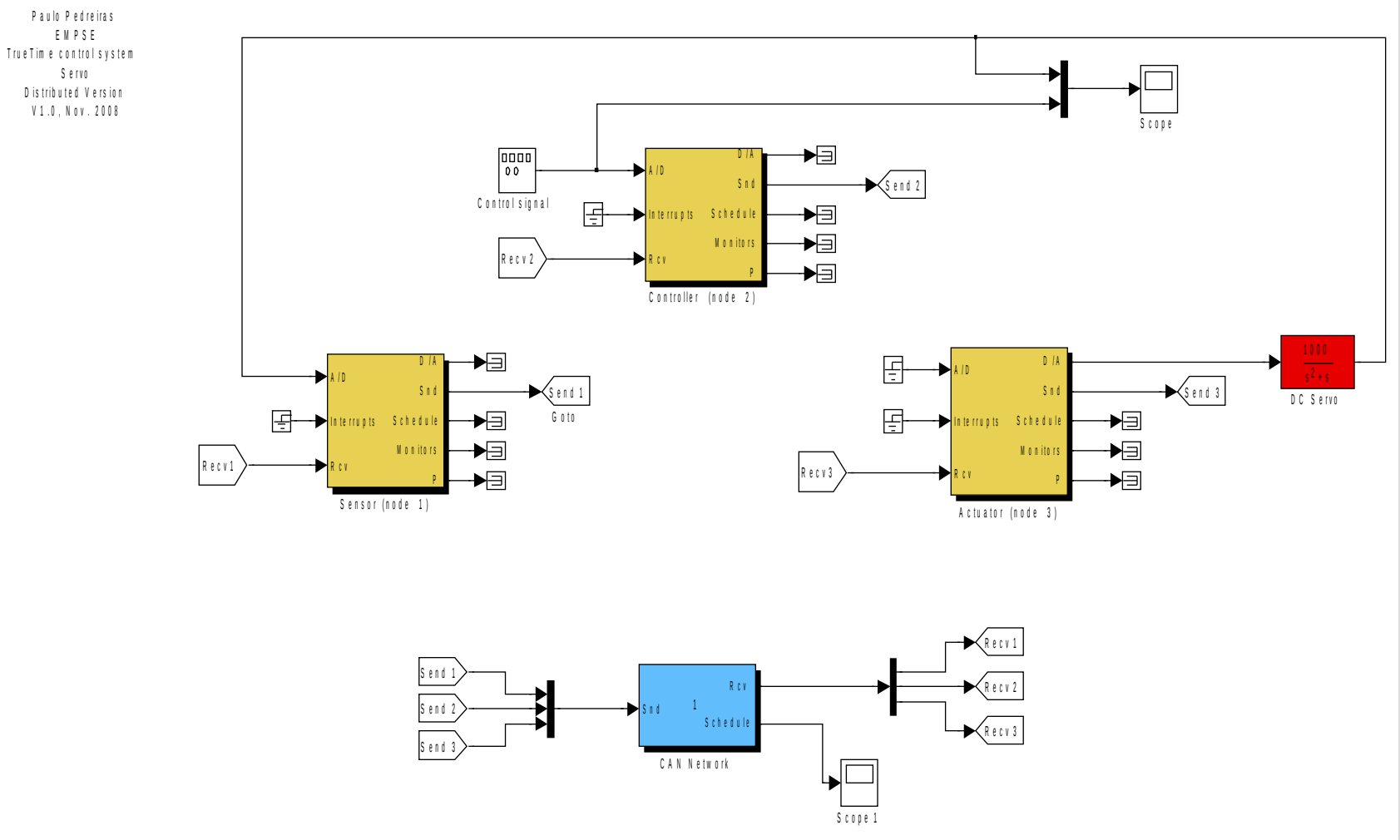
Tarefa 2: De há diversos anos para cá tem havido uma forte tendência para a adopção de arquitecturas distribuídas, em que os sistema é composto por diversos nós "inteligentes" (i.e., com capacidade de processamento) e cooperativos, interligados por redes de comunicação

Objectivo:

- **Fase 1** : modificar o sistema anterior por forma a torná-lo distribuindo
  - Deverão haver 3 nós: **sensor, controlador e actuador**
  - Comunicação por **rede CAN**
- **Fase 2** : adicionar mais nós com o fim de gerar **carga** na rede. Determinar uma configuração em que a perturbação seja evidente.

Tarefa 2: modelação de um sistema distribuído

Modelo do sistema distribuído



Tarefa 2: modelação de um sistema distribuido

- É necessário **adicionar** uma "**True Time Network**" (TTN) e 3 TTK (um para cada nó).
- O módulo **TTN** deve ser configurado para **3 nós**, CSMA/AMP(CAN), 100000 bits/s, *minimum frame size* de 65 bit. Deixe a **rede por defeito** (1)
- Na página da disciplina **já se encontra** o **modelo** bem como o **código** respeitante ao **nó sensor** e ao **nó actuador**.
  - Estude o código que aí se encontra para compreender o funcionamento da rede
  - **Adicione** as funções respeitante ao nó **controlador**