# Aula Prática - 1

## Interferências temporais

**Objectivos:**

Observação de interferências no comportamento termporal de programas

1. utilização de processos no SO Linux (PC)
2. utilização de rotinas de interrupção num sistema embedded sem SO (Kit188)

**Procedimentos:**

1. Utilizar o programa *example1-linux.c*, disponível na página da disciplina.
    a. Observar o seu conteúdo. Reparar na técnica utilizada para gerar uma operação aproximadamente periódica recorrendo à função *sleep*. Notar que o Linux não tem suporte ao nível de processos para efectuar activações periódicas.
    b. Lançar o processo com o código daquele programa. Notar que, para períodos longos (>100ms) e sem outros processos *activos*, as reactivações do processo são relativamente periódicas, com um jitter pequeno (inferior a 10%).
    c. Lançar outros processos com o mesmo programa mas sem espera (sem chamada a *sleep*) logo com carga permanente. Observar as variações significativas nos períodos de reactivação do processo inicial no momento em que se lançam os novos. Essa interferência, ainda assim, acaba por ser novamente atenuada pelo mecanismo de escalonamento do Linux, que promove uma distribuição equitativa do CPU pelos vários processos.
    d. Diminuir a prioridade do processo inicial relativamente aos outros, por exemplo usando a função *renice*, e observar as grandes variações introduzidas na sua periodicidade (jitter >100%).

2. Utilizar o programa *aula1kit.c*, disponível na página da disciplina.
    a. Montar o sistema Kit188 com a placa IO188. Observar o conteúdo do programa que contém um ciclo infinito com uma determinada carga, executado com uma frequência baixa, e uma rotina de interrupção curta e rápida que faz piscar um conjunto de *leds*.
    b. Colocar em funcionamento o sistema e observar o piscar periódica dos leds.
    c. Simular uma zona crítica no ciclo do programa principal, inibindo as interrupções no início e restabelecendo-as no final. Observar o bloqueio causado à rotina de interrupção.