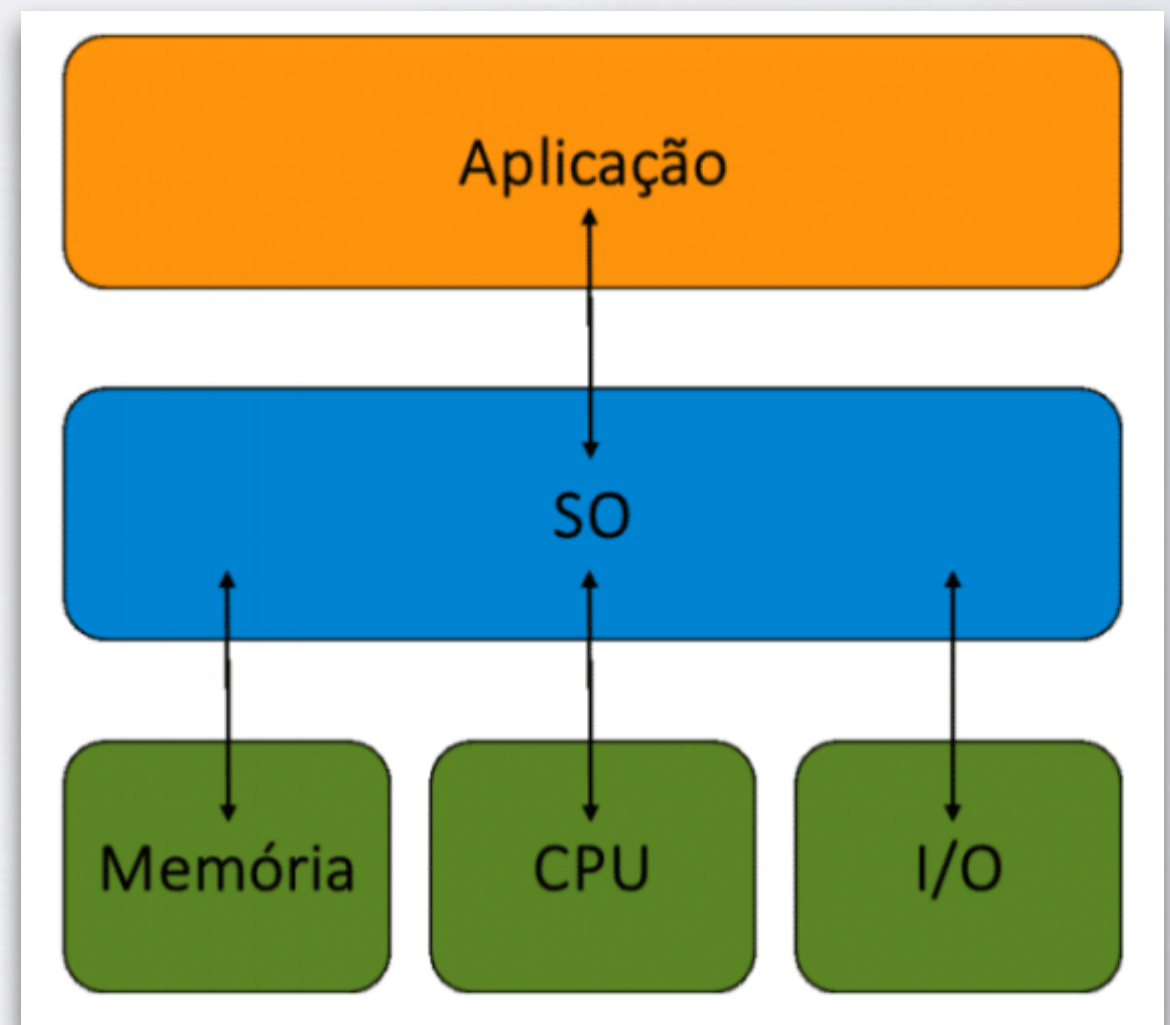


# MICROPROCESSADORES E MICROCONTROLADORES



# SISTEMAS OPERACIONAIS

Software de gerenciamento dos recursos do sistema:

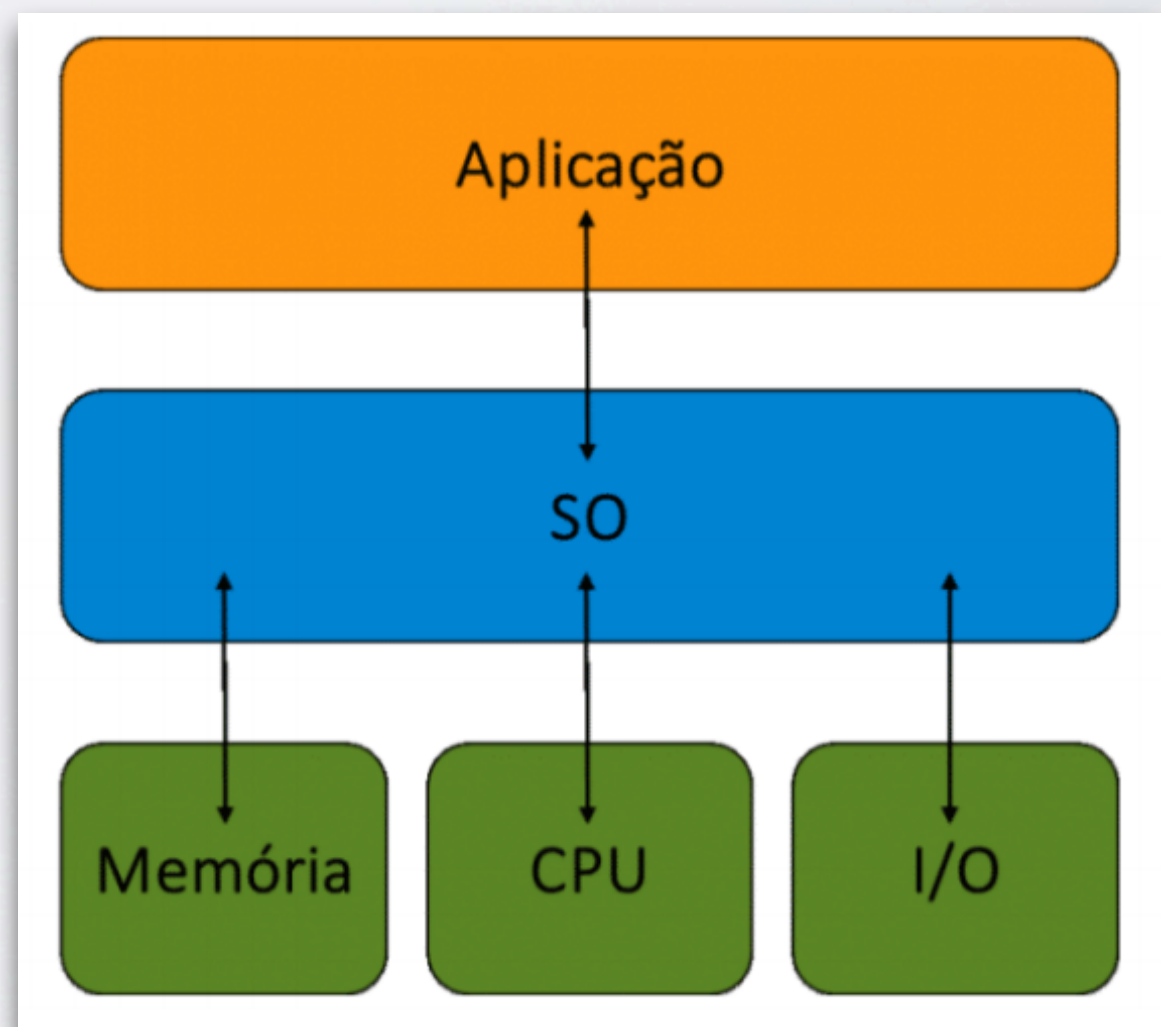




# SISTEMAS OPERACIONAIS

Software de gerenciamento dos recursos do sistema:

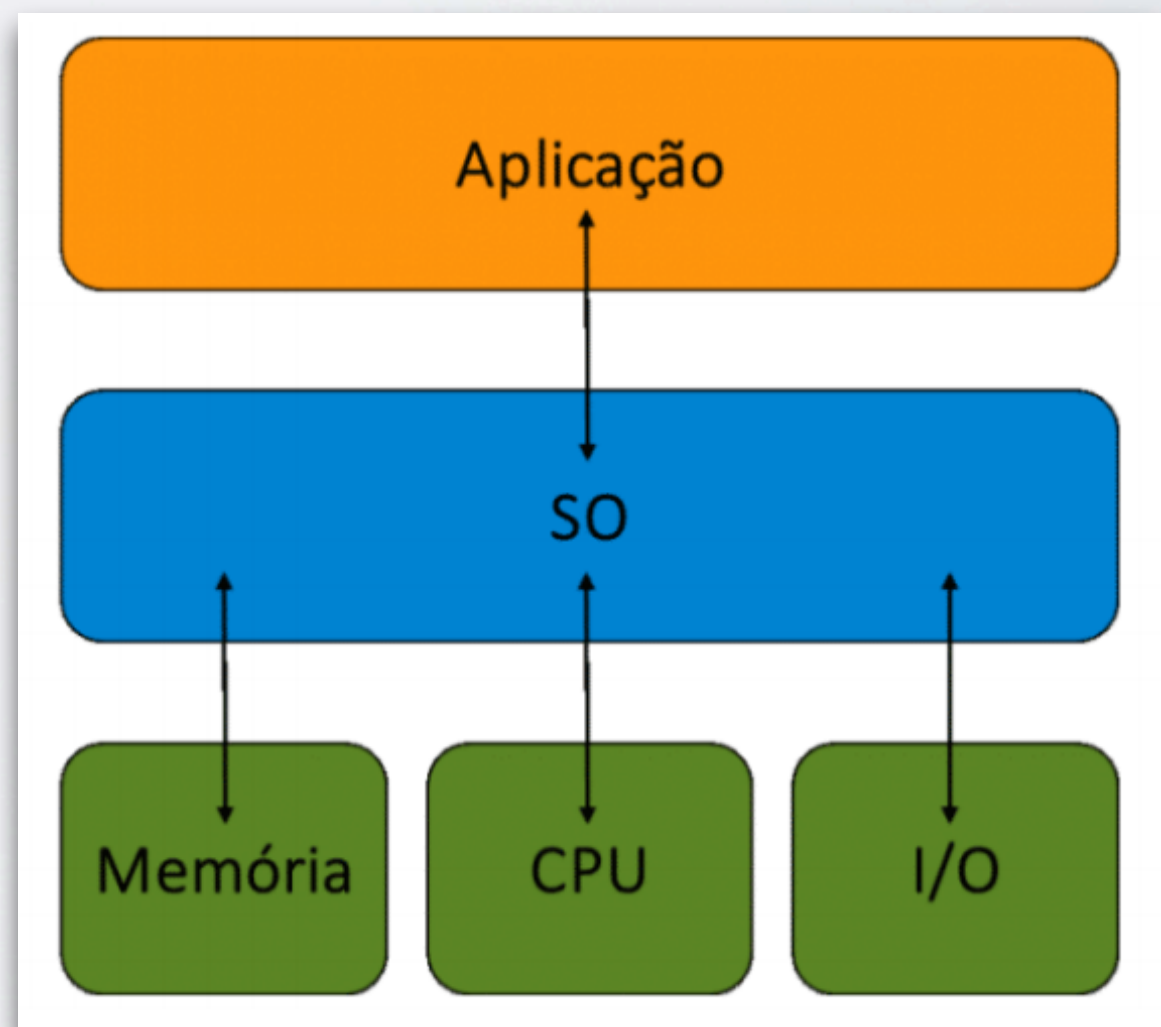
- Uso da CPU pelos diversos programas



# SISTEMAS OPERACIONAIS

Software de gerenciamento dos recursos do sistema:

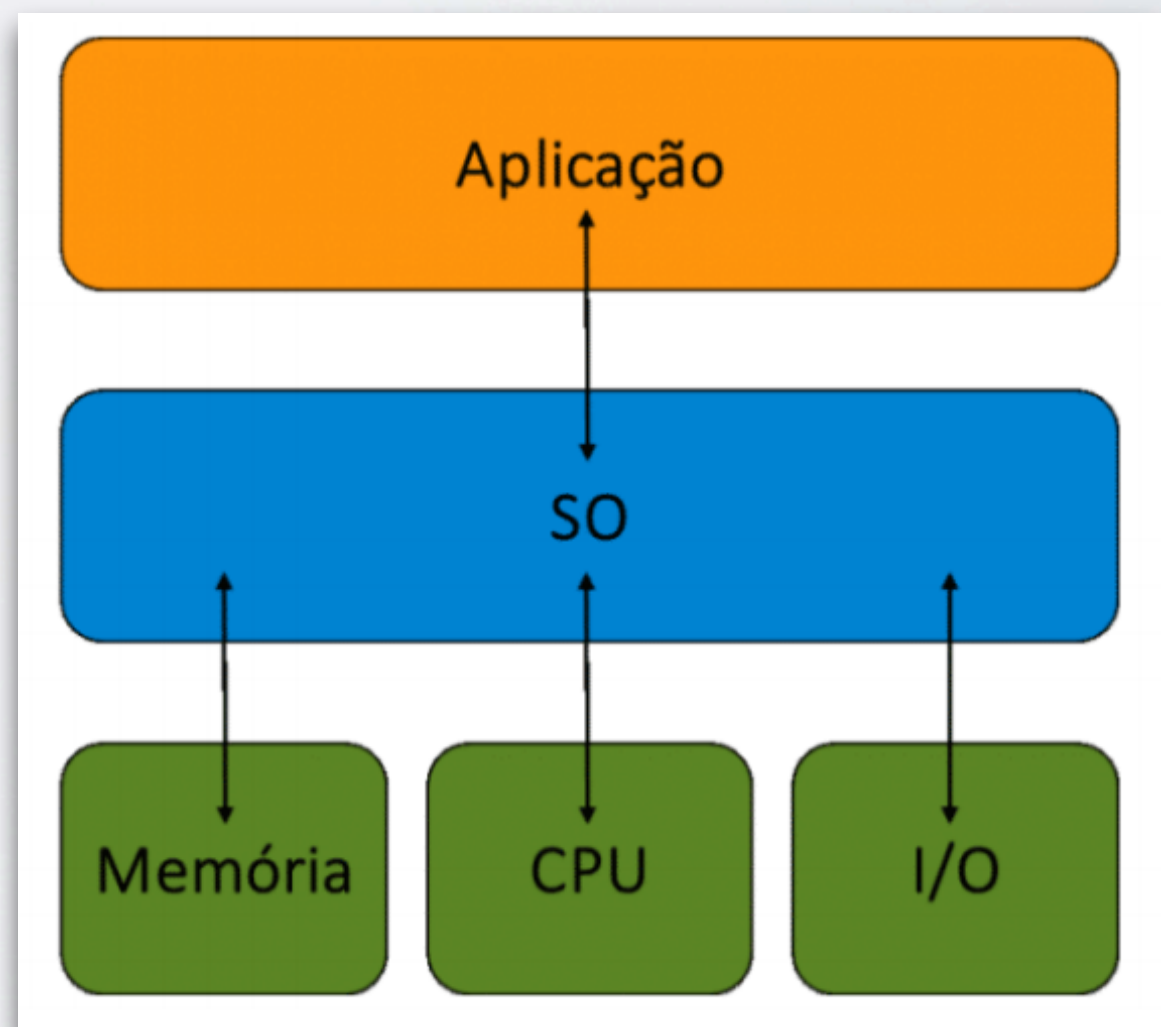
- Uso da CPU pelos diversos programas
- Memória para os programas e arquivos



# SISTEMAS OPERACIONAIS

Software de gerenciamento dos recursos do sistema:

- Uso da CPU pelos diversos programas
- Memória para os programas e arquivos
- Entradas e saídas digitais

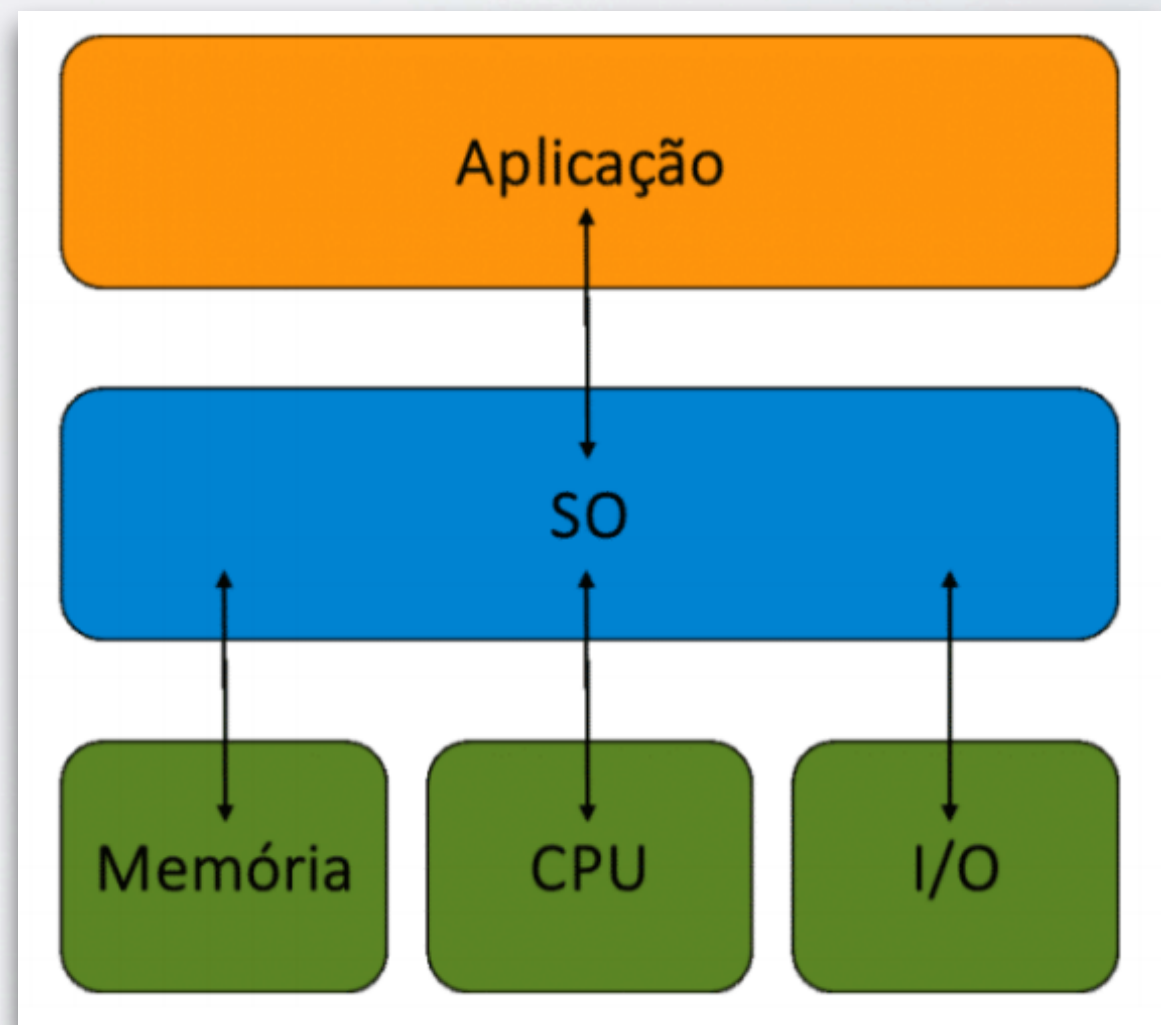




# SISTEMAS OPERACIONAIS

Software de gerenciamento dos recursos do sistema:

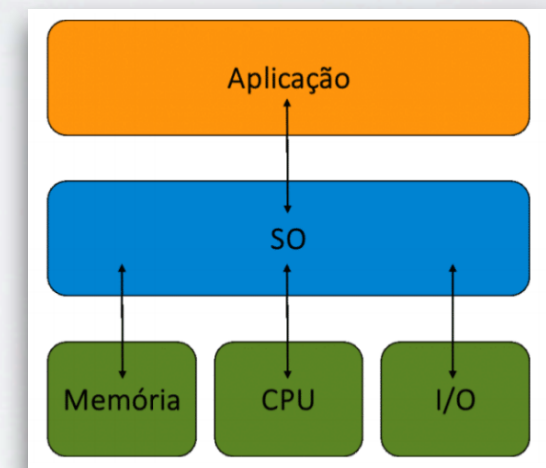
- Interfaces gráficas
- Drivers de dispositivos
- Sistemas de arquivos
- Engines de processamento 3D
- Etc.



# SISTEMAS OPERACIONAIS DE TEMPO REAL

Em sistemas embarcados, temos muitas restrições de memória, CPU e I/O:

- Tempo de execução
- Tamanho da memória usada
- Tamanho do código
- Consumo de energia
- Etc.

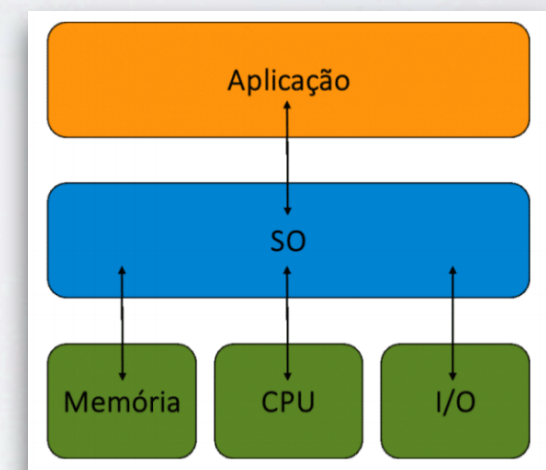


# SISTEMAS OPERACIONAIS DE TEMPO REAL

Em sistemas embarcados, temos muitas restrições de memória, CPU e I/O:

Utilizando microcontroladores como o MSP430:

- CPU de 16MHz
- RAM de poucos kilobytes
- Dezenas de pinos de I/O



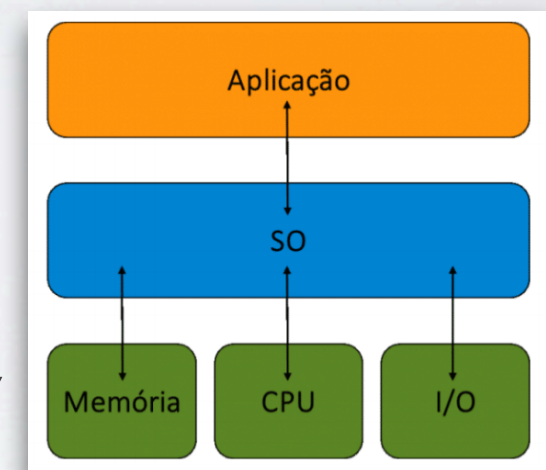


# SISTEMAS OPERACIONAIS DE TEMPO REAL

Em sistemas embarcados, temos muitas restrições de memória, CPU e I/O:

Utilizando *systems on-chip* como o Raspberry Pi:

- CPU de 700-1400 MHz
- RAM de 1GB
- 28-40 pinos de I/O








# SISTEMAS OPERACIONAIS DE TEMPO REAL

RTOS (*real-time operating system*)

Sistemas operacionais simples que permitem o controle do tempo das aplicações executadas.

Alguns exemplos:

Sistema Operacional	Consumo de Flash (mínimo)	Consumo de RAM (mínimo)
VxWorks 	> 75.000	-
FreeRTOS 	> 6.000	> 800
uC/OS 	> 5.000	-
uOS 	> 2.000	> 200
BRTOS 	> 2.000	~ 100

# SISTEMAS OPERACIONAIS DE TEMPO REAL

## **Hard real-time**

- Fortemente baseado em deadlines
- Tolerância "zero" a atrasos
- Não cumprimento de deadlines implica em falhas catastróficas
- Tempo de resposta é determinístico inclusive sob condição de sobrecarga



# SISTEMAS OPERACIONAIS DE TEMPO REAL

## **Hard real-time**

- Controlador de motor (robôs, esteiras)
- Sistemas automotivos (freio ABS, *airbag*)
- Sistemas aeronáuticos (controle de altitude, medição de velocidade)
- Marcapasso

# SISTEMAS OPERACIONAIS DE TEMPO REAL

## **Soft real-time**

- Também baseados em deadlines
- Apresentam certa tolerância a atrasos
- O não-cumprimento de deadlines não implica em falhas catastróficas
- Desempenho é degradado sob condição de sobrecarga

# SISTEMAS OPERACIONAIS DE TEMPO REAL

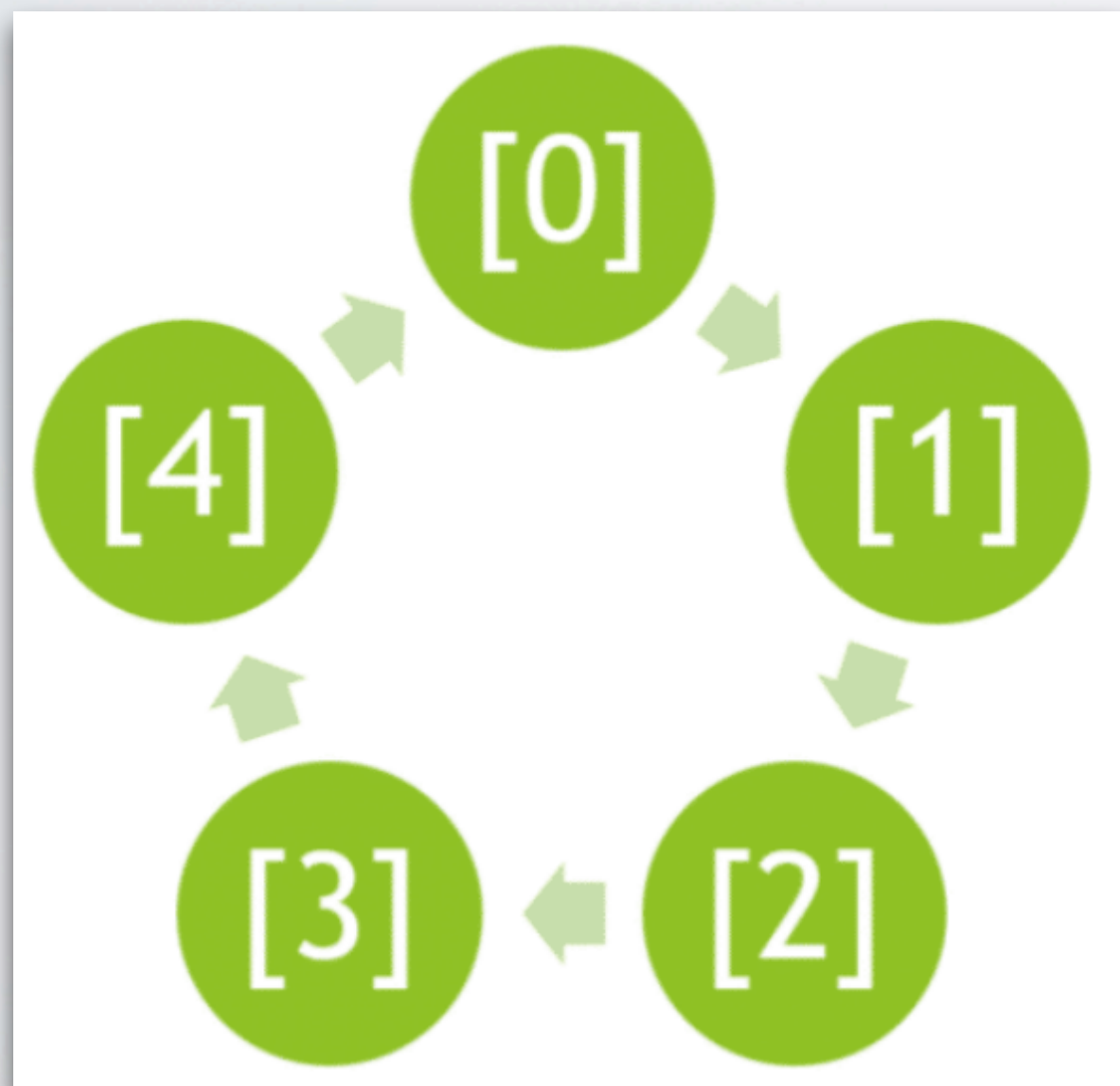
## **Soft real-time**

- Tela do celular/tablet
- Sistemas de entretenimento (CD, DVD, smart TV)
- TV digital



# GERENCIAMENTO DA CPU

*Multitasking* é na verdade o compartilhamento da CPU por uma série de aplicações

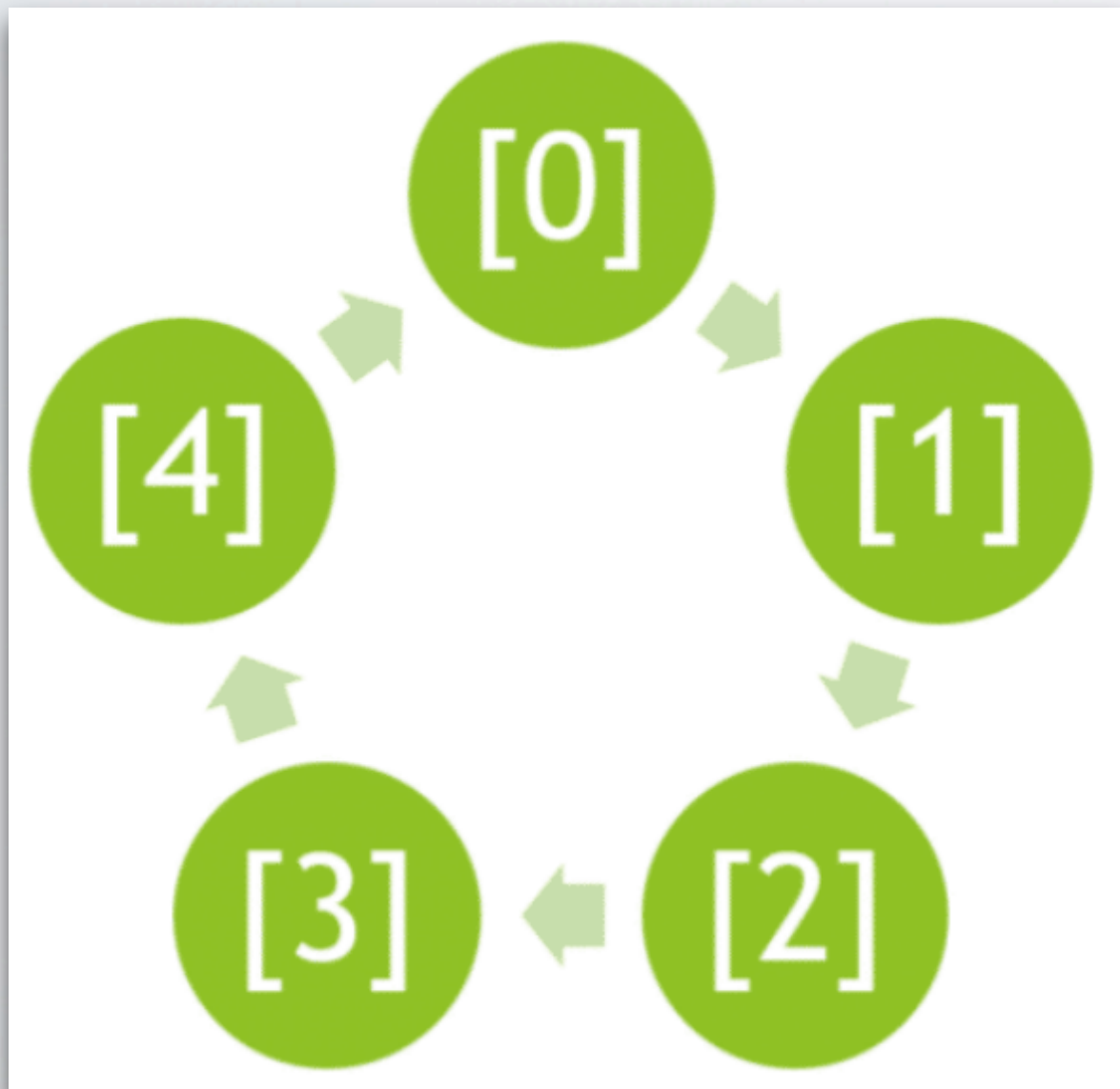


# GERENCIAMENTO DA CPU

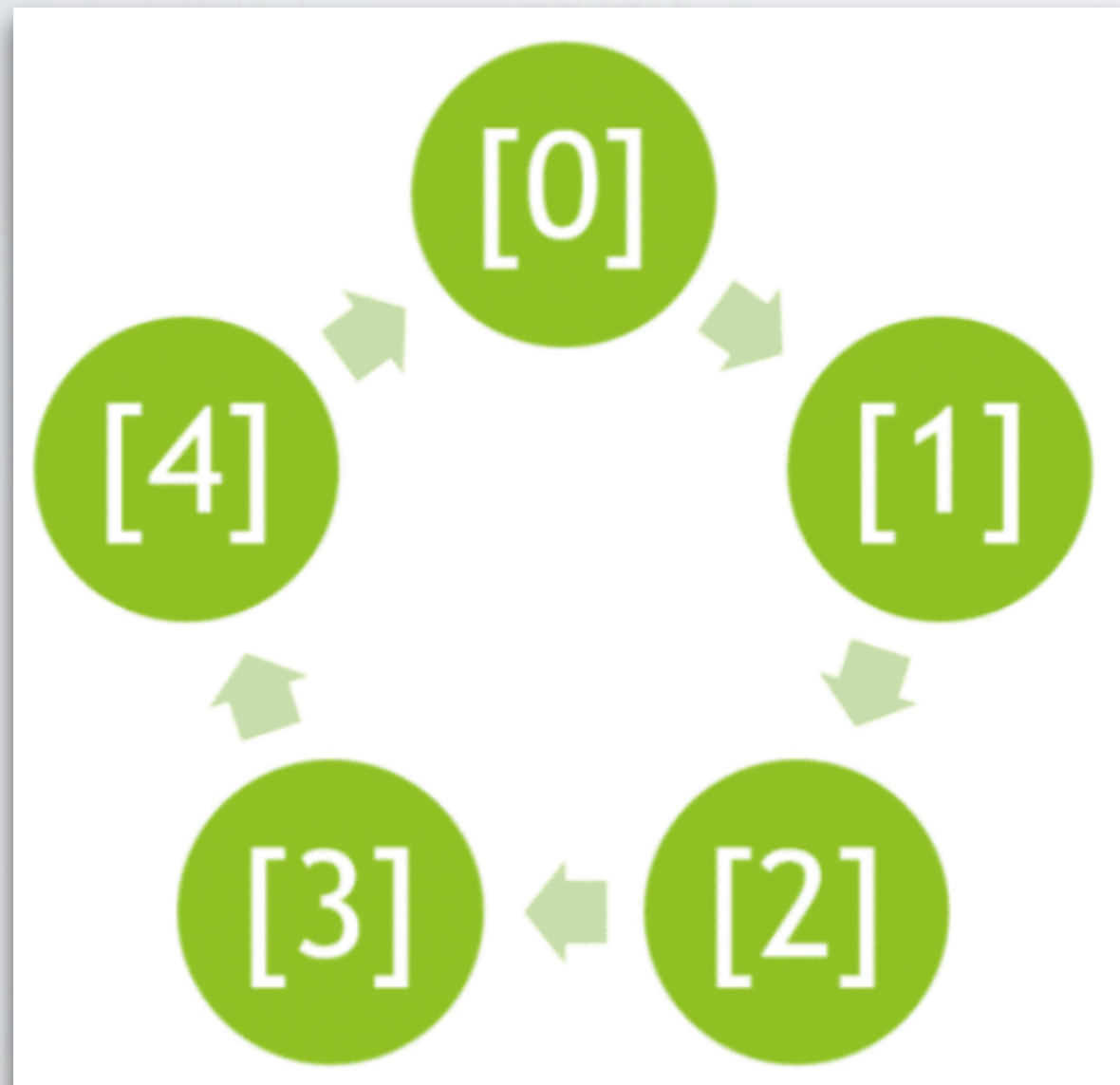
*Multitasking* é na verdade o compartilhamento da CPU por uma série de aplicações

Possíveis critérios:

- Prioridade



# GERENCIAMENTO DA CPU



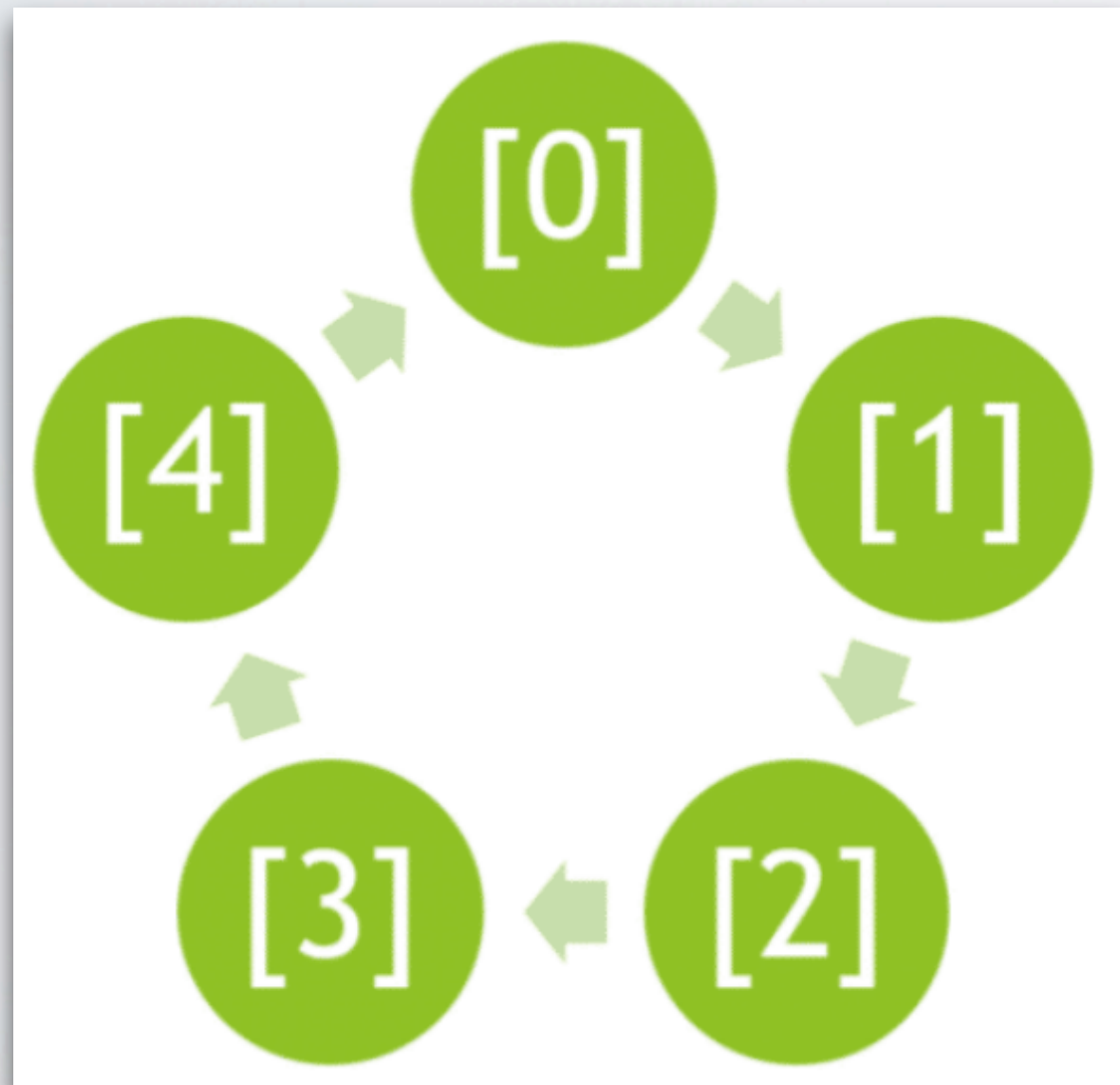
*Multitasking* é na verdade o compartilhamento da CPU por uma série de aplicações

Possíveis critérios:

- Prioridade
- Criticidade



# GERENCIAMENTO DA CPU

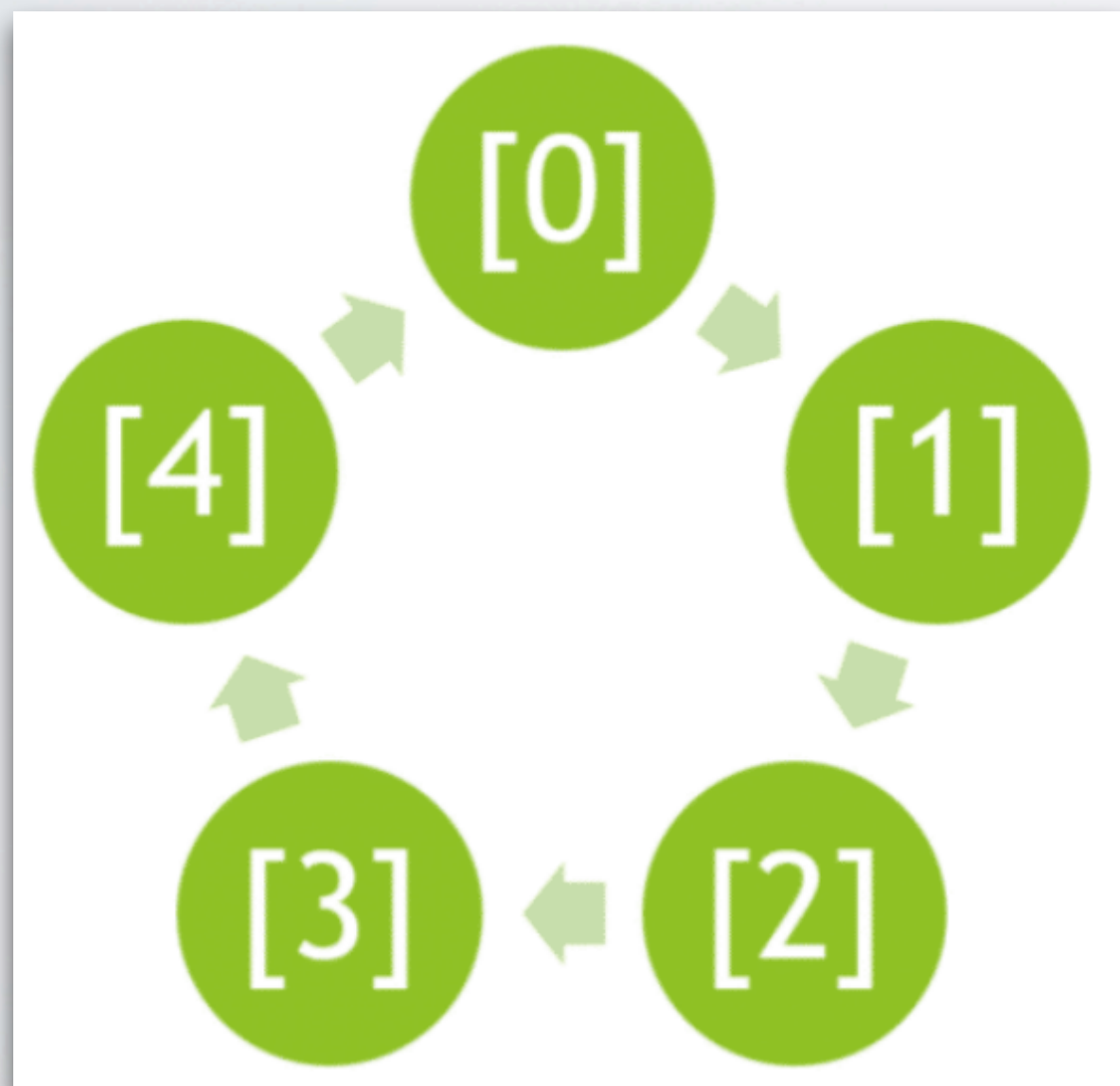


*Multitasking* é na verdade o compartilhamento da CPU por uma série de aplicações

Possíveis critérios:

- Prioridade
- Criticidade
- Sequência

# GERENCIAMENTO DA CPU



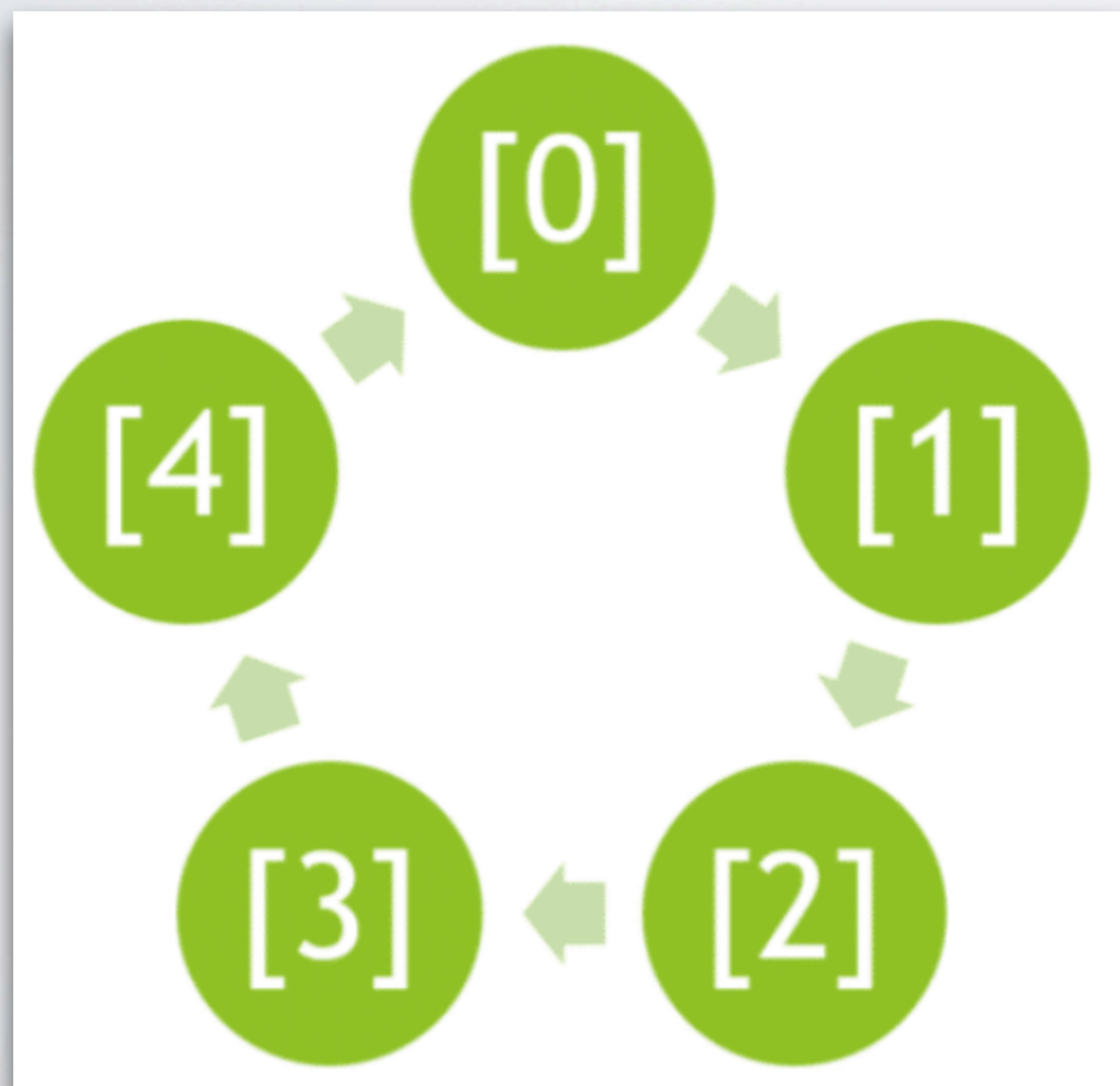
*Multitasking* é na verdade o compartilhamento da CPU por uma série de aplicações

Possíveis critérios:

- Prioridade
- Criticidade
- Sequência
- Tempo

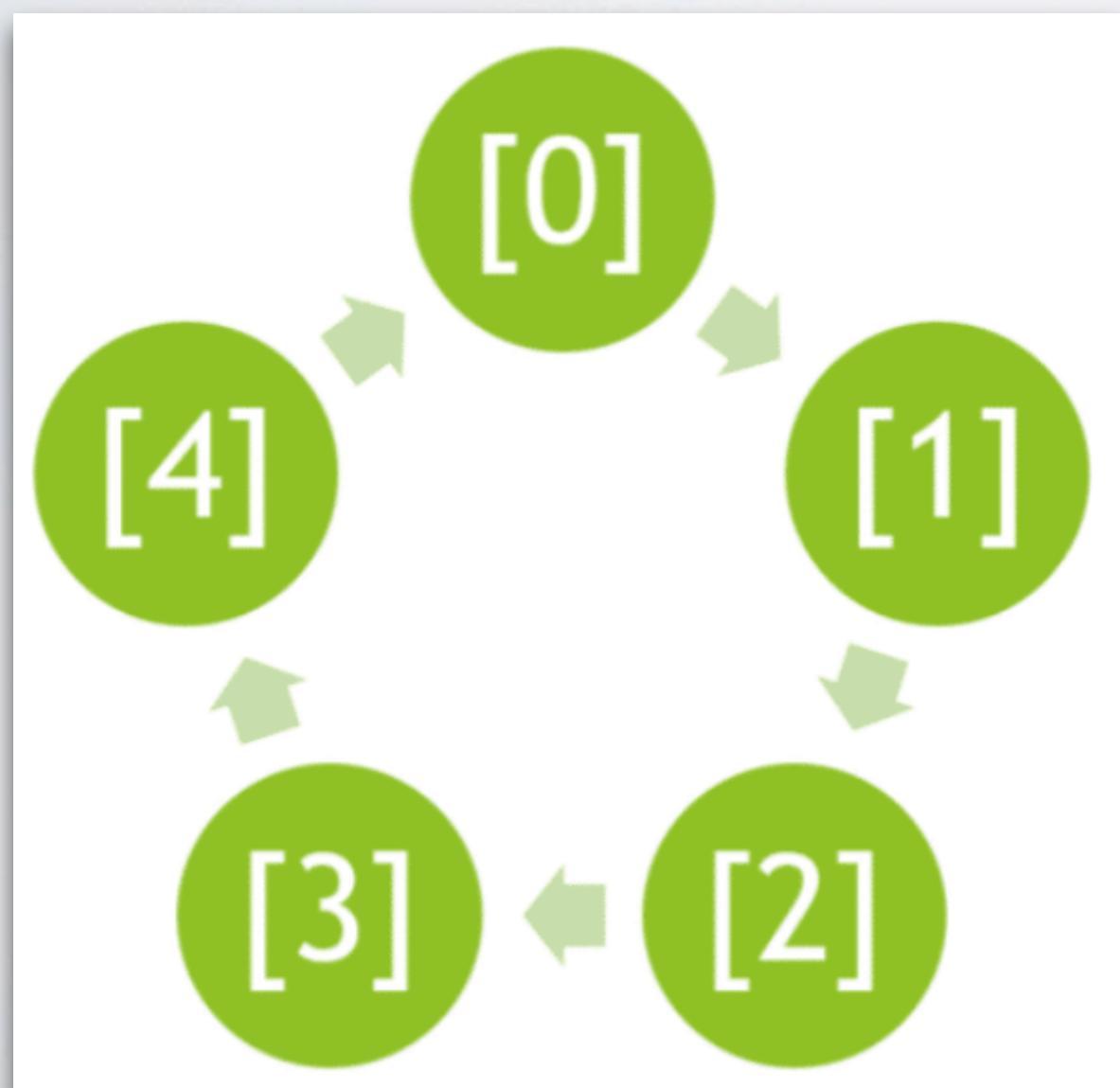
# GERENCIAMENTO DA CPU

O agendamento das tarefas é feita pelo *kernel* do sistema operacional.





# GERENCIAMENTO DA CPU

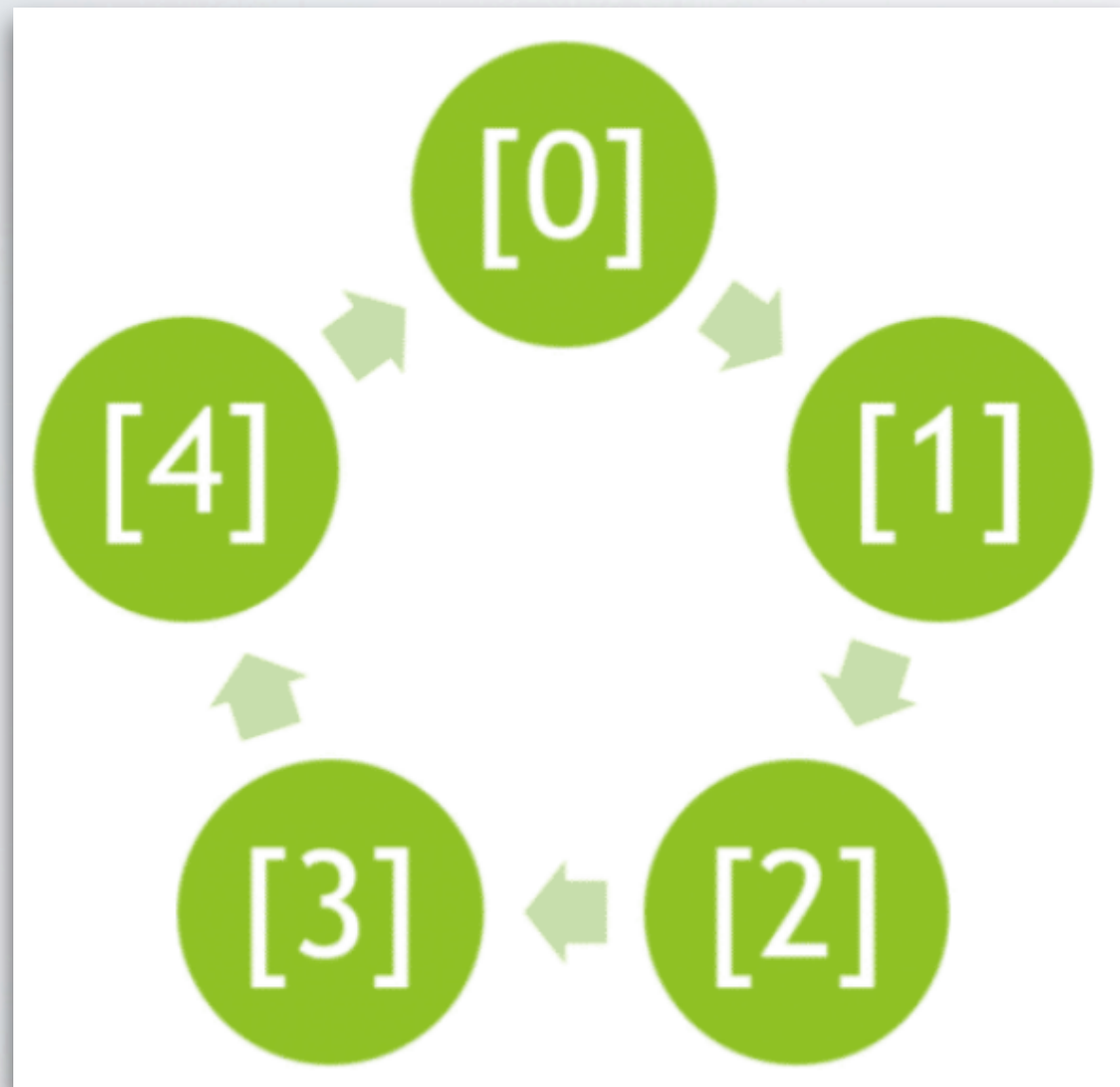


O agendamento das tarefas é feita pelo *kernel* do sistema operacional.

Tipos de *kernel*:

- Cooperativo - cada atividade é executada do começo ao fim; uma nova atividade só começa quando outra termina

# GERENCIAMENTO DA CPU



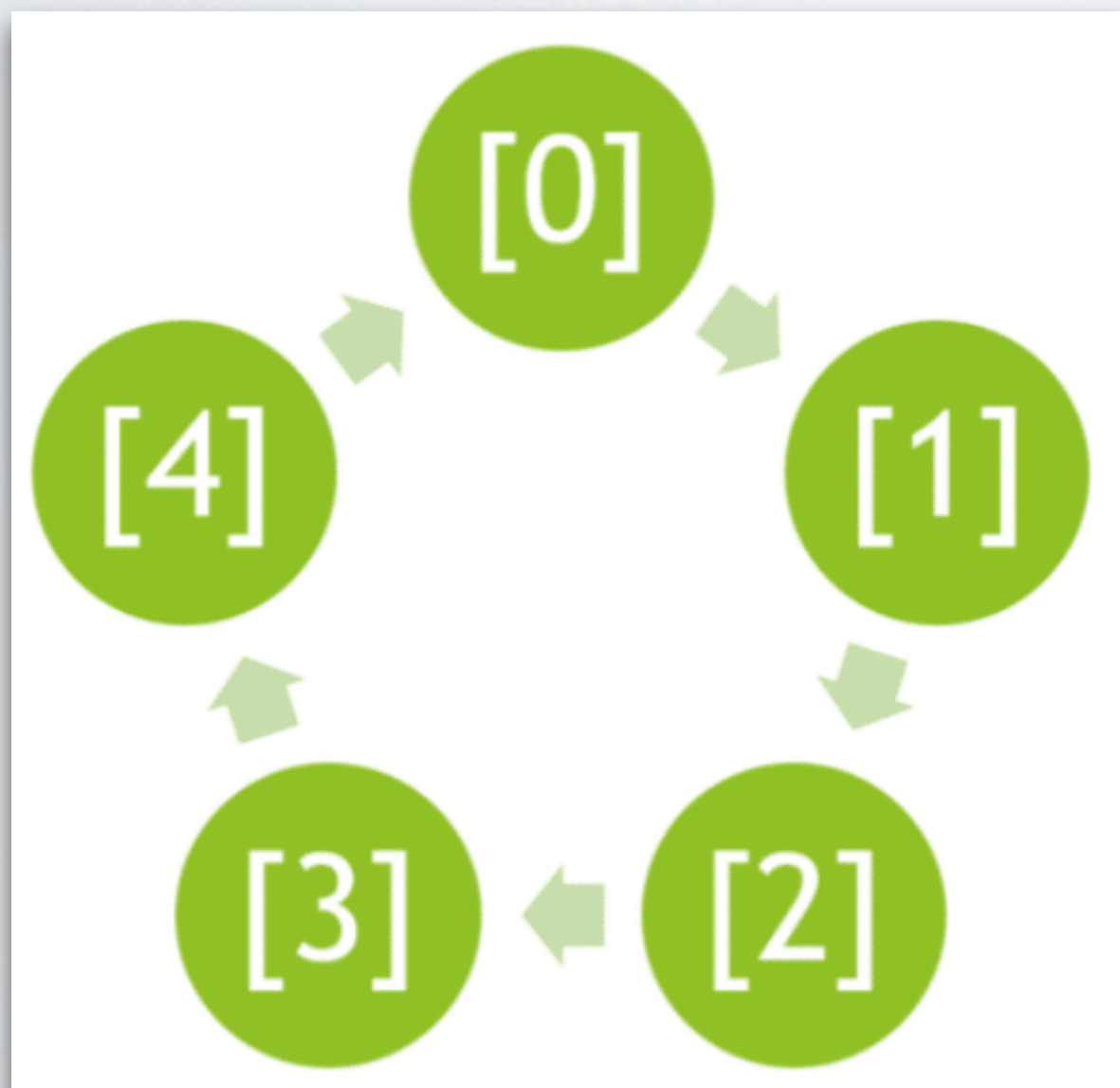
O agendamento das tarefas é feita pelo *kernel* do sistema operacional.

Tipos de *kernel*:

- Preemptivo - atividades podem ser pausadas para que outra atividade seja executada

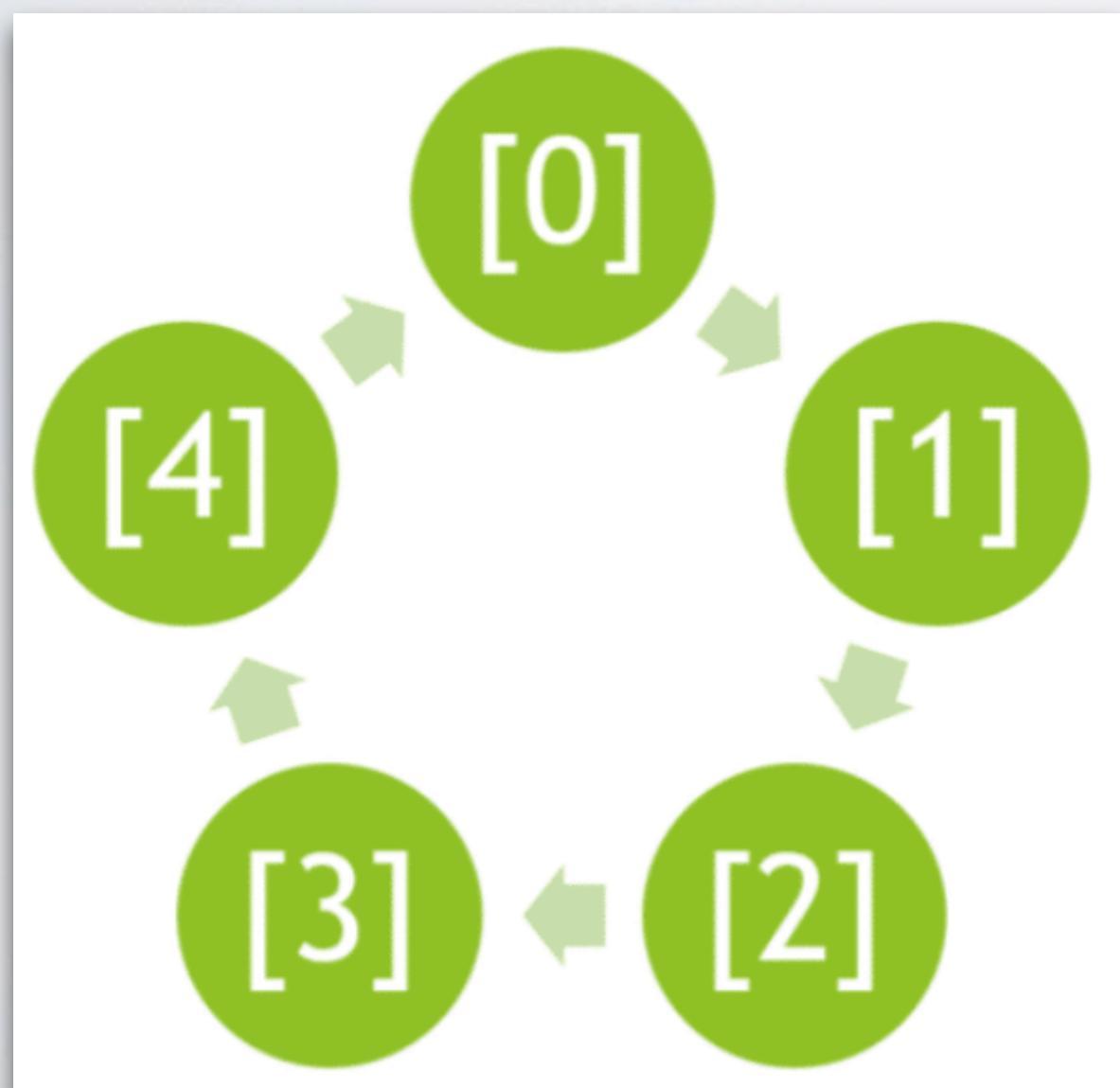
# GERENCIAMENTO DA CPU

O *kernel* cooperativo é uma evolução das máquinas de estado tradicionais





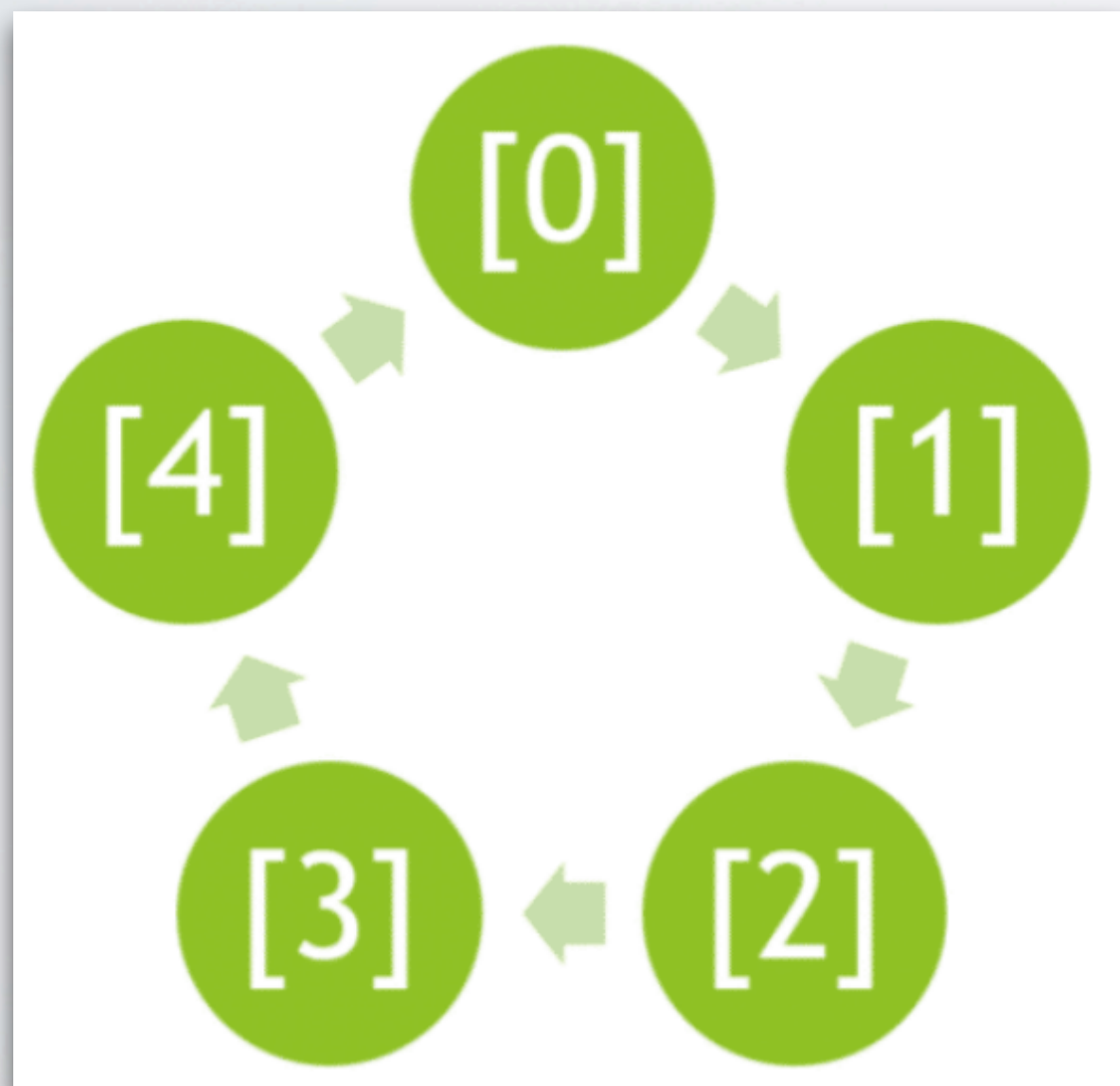
# GERENCIAMENTO DA CPU



O *kernel* cooperativo é uma evolução das máquinas de estado tradicionais

- Para atualizar uma máquina de estados com estados fixos, é preciso analisá-la novamente, gerando um novo projeto e uma nova implementação (reescrever, recompilar e regravar o código do dispositivo)

# GERENCIAMENTO DA CPU

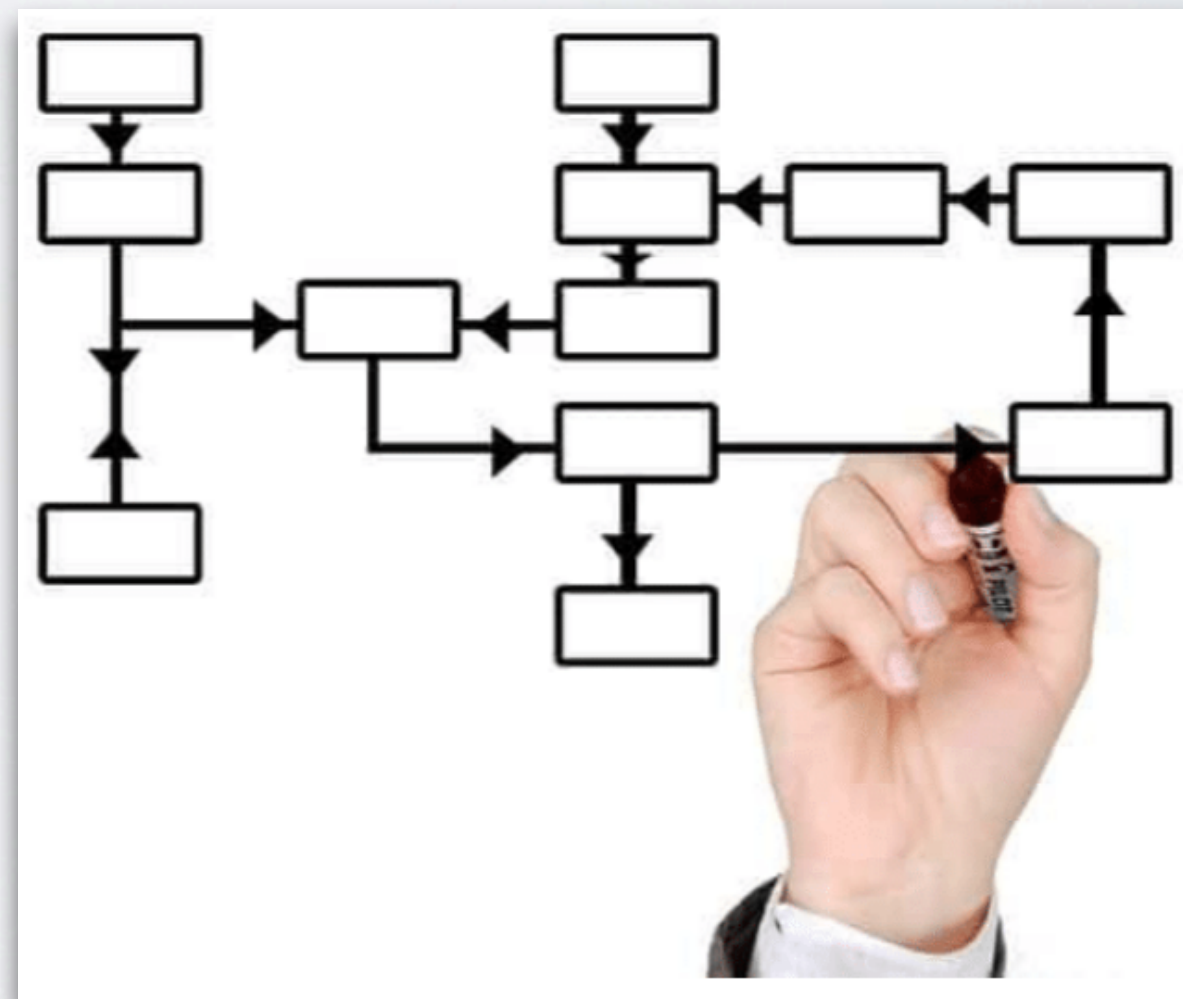


O *kernel* cooperativo é uma evolução das máquinas de estado tradicionais

- Um kernel cooperativo remove essa limitação, permitindo que as tarefas possam ser criadas ou removidas com o sistema em funcionamento

# PROCESSOS E THREADS

Processos e threads são unidades de código executável que pode ser gerenciada de modo independente pelo *kernel*

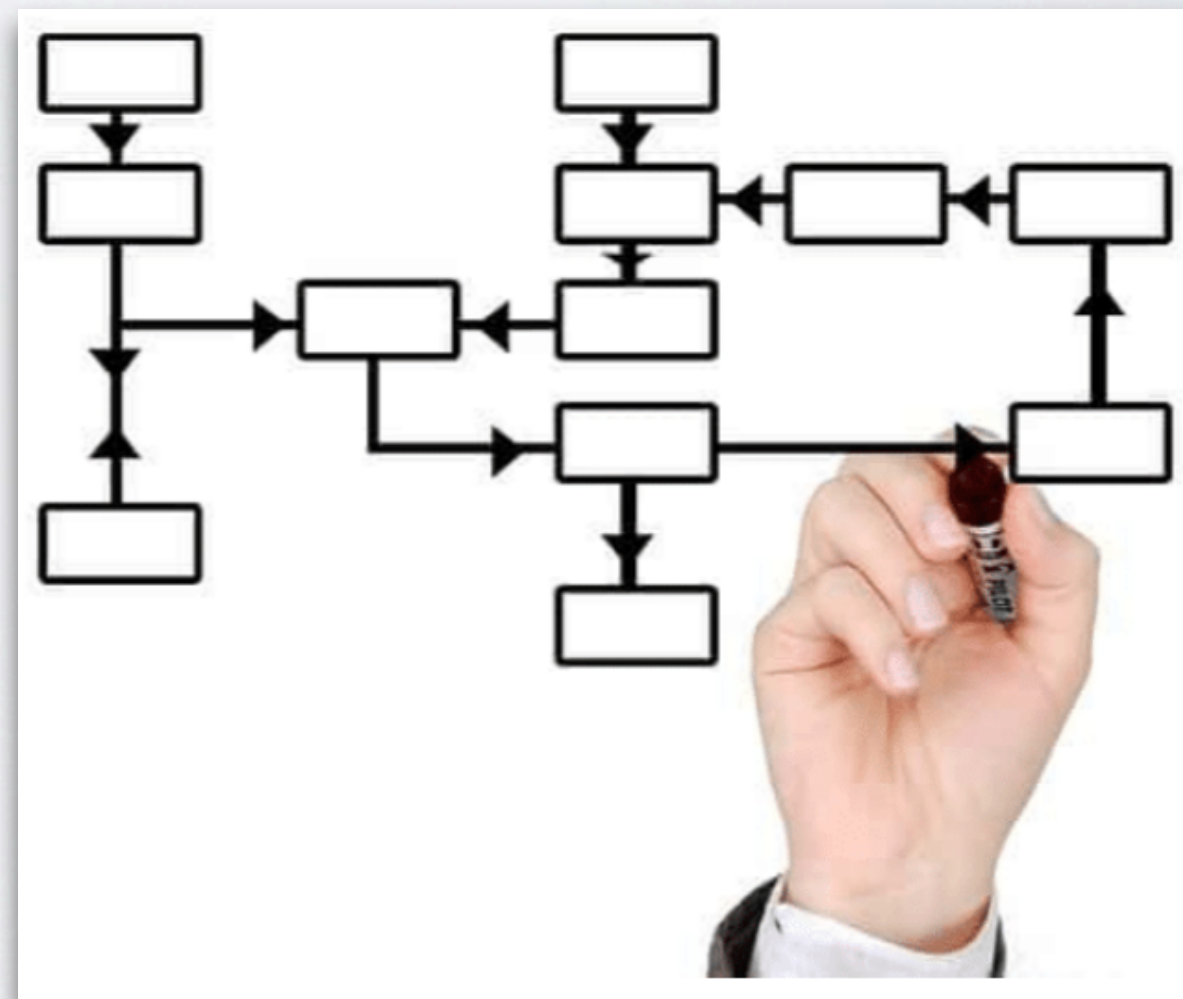




# PROCESSOS E THREADS

Processos e threads são unidades de código executável que pode ser gerenciada de modo independente pelo *kernel*

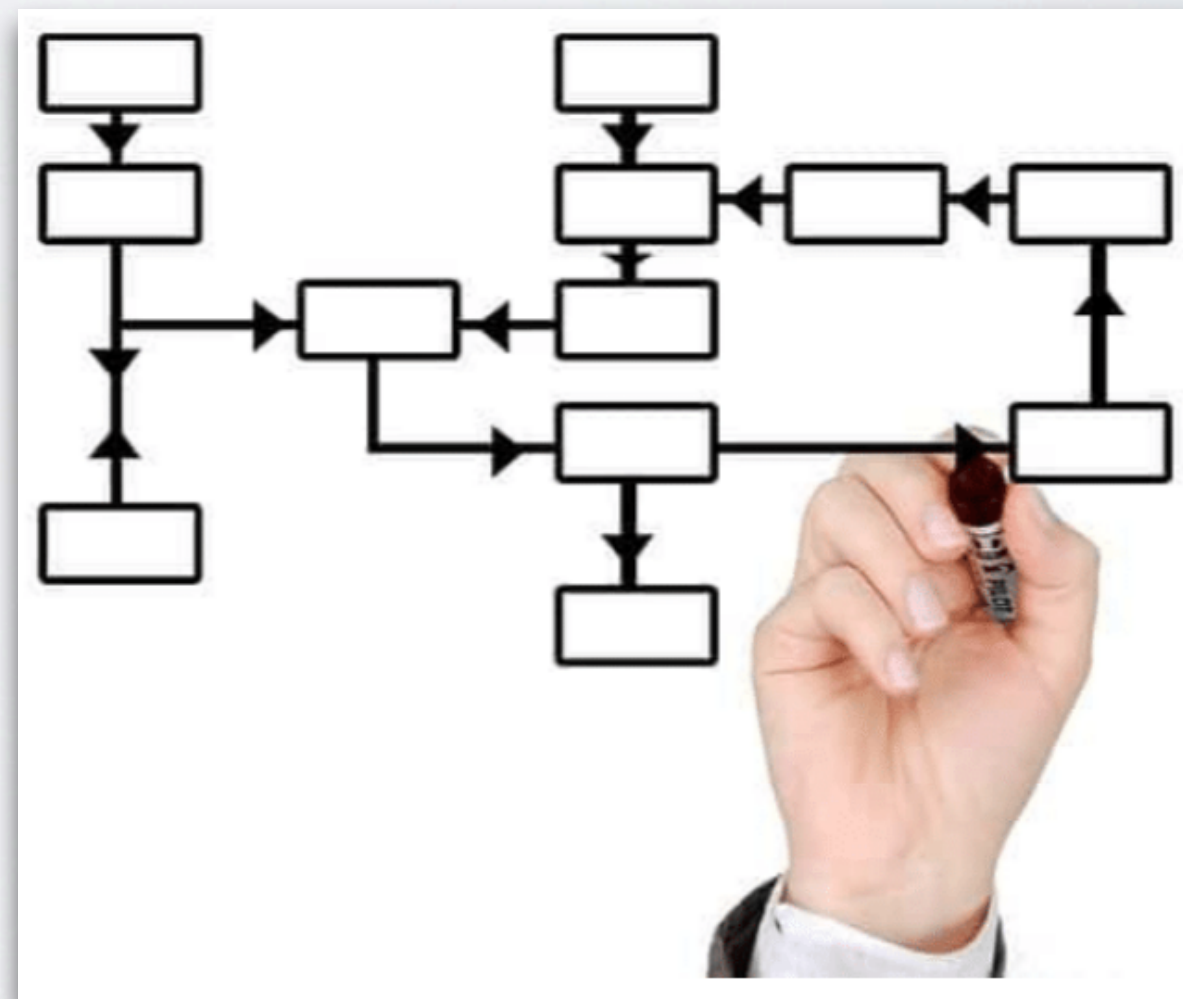
A principal diferença entre processos e threads é o compartilhamento de memória



# PROCESSOS E THREADS

Processos e threads são unidades de código executável que pode ser gerenciada de modo independente pelo *kernel*

- Dois processos só conseguem se comunicar com sistemas dedicados de passagem de mensagens pois as memórias que ocupam são isoladas

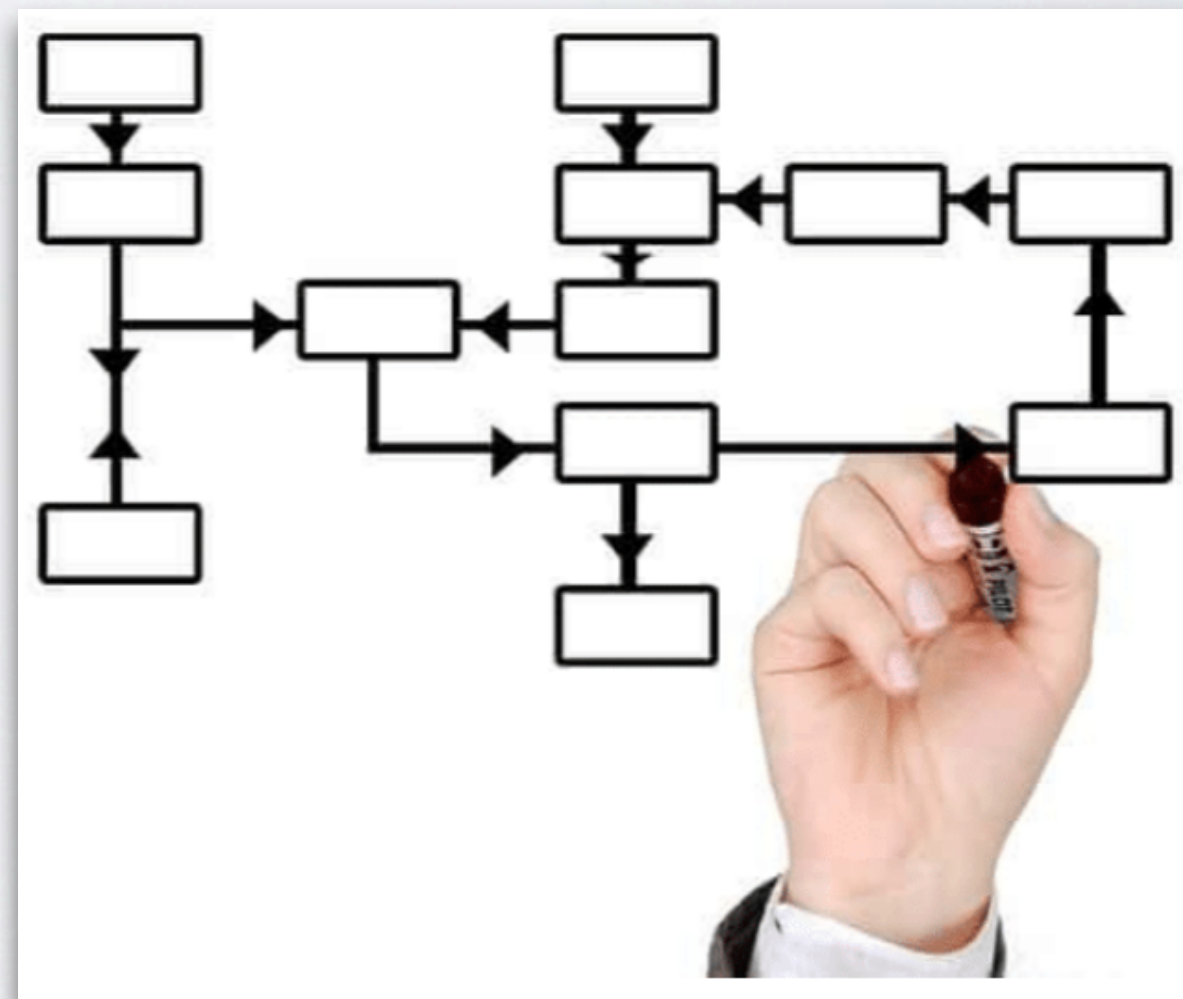




# PROCESSOS E THREADS

Processos e threads são unidades de código executável que pode ser gerenciada de modo independente pelo *kernel*

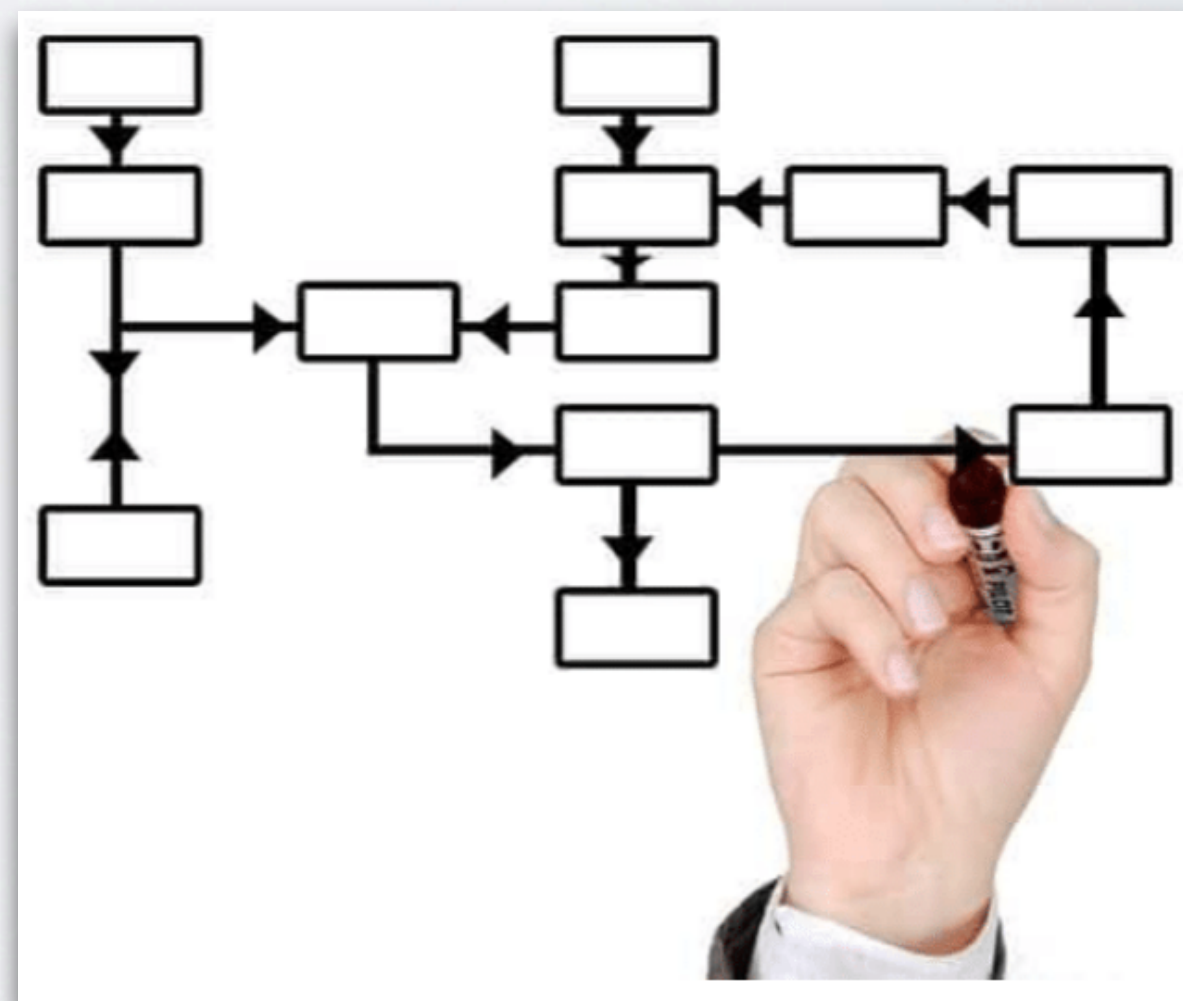
- Threads compartilham a região de memória que se encontram.





# PROCESSOS E THREADS

Em sistemas operacionais voltados para microcontroladores mais simples, principalmente aqueles que não possuem MMU, pode não ser possível implementar um processo, de modo que todas as atividades são por definição threads



# PROCESSOS E THREADS

Para evitar confusões é comum utilizar a nomenclatura task para definir uma aplicação/trecho de código executável por um sistema operacional embarcado

