طراحی سیستم‌های تعبیه‌شده
Embedded System Design

فصل پنجم ـ قسمت هفتم

پیاده‌سازی سیستم‌های تعبیه‌شده

Implementing Embedded Systems: Hardware / Software Codesign

کاظم فولادی
دانشکده‌ی مهندسی برق و کامپیوتر
دانشگاه تهران

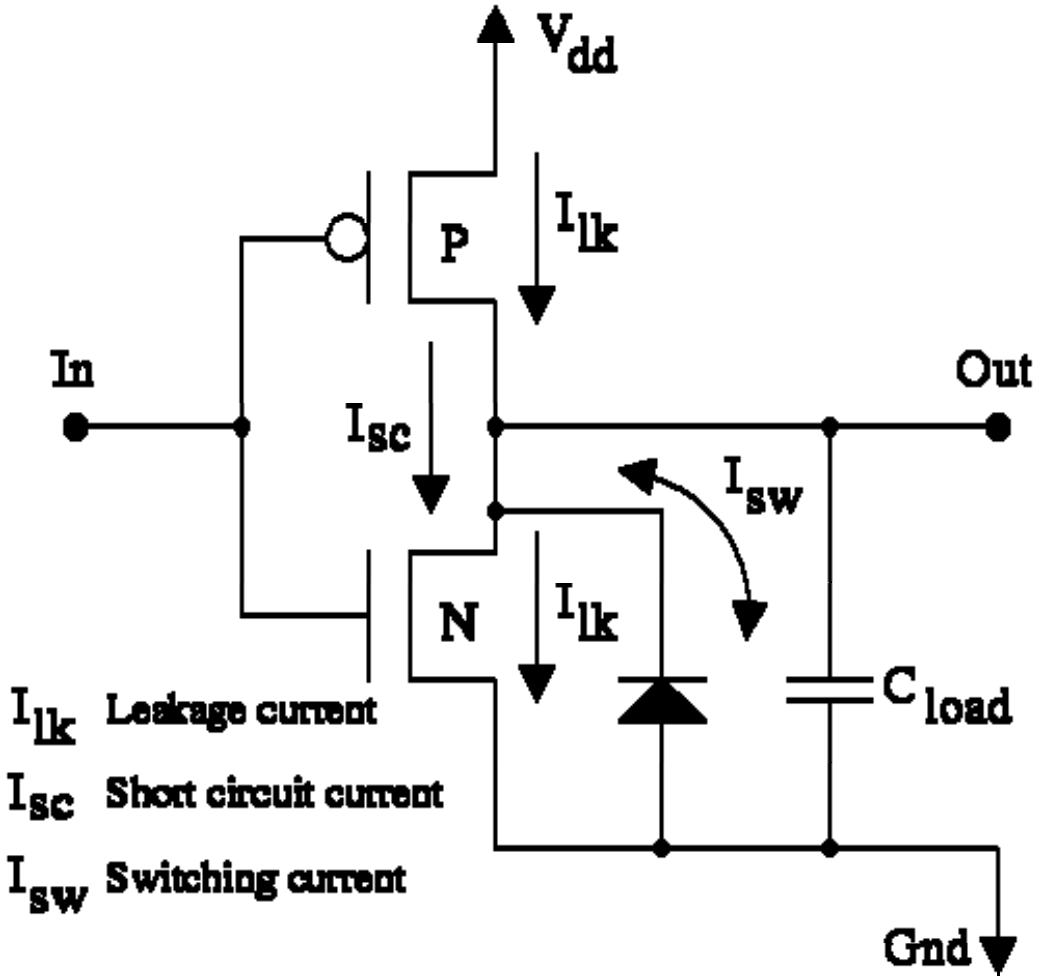
kazim@fouladi.ir

سیستم‌های تعبیه‌شده
Embedded Systems

- Power/Energy Aware Embedded Systems
- Dynami
- Dynami

http://www.hex-tech.co.uk/egg.asp

توان مصرفی یک گیت
Power Consumption of a Gate



Recap from chapter 3: Fundamentals of dynamic voltage scaling (DVS)

Power consumption of CMOS circuits (ignoring leakage):

$$P = \alpha\, C_L\, V_{dd}^2\, f$$

α : switching activity

C_L : load capacitance

V_{dd} : supply voltage

f : clock frequency

Delay for CMOS circuits:

$$\tau = k\, C_L \frac{V_{dd}}{(V_{dd} - V_t)^2}$$

V_t : threshhold voltage

$(V_t << V_{dd})$

☞ Decreasing V_{dd} reduces P quadratically, while the run-time of algorithms is only linearly increased (ignoring the effects of the memory system).

امکان‌های بهینه‌سازی انرژی
Potential for Energy Optimization

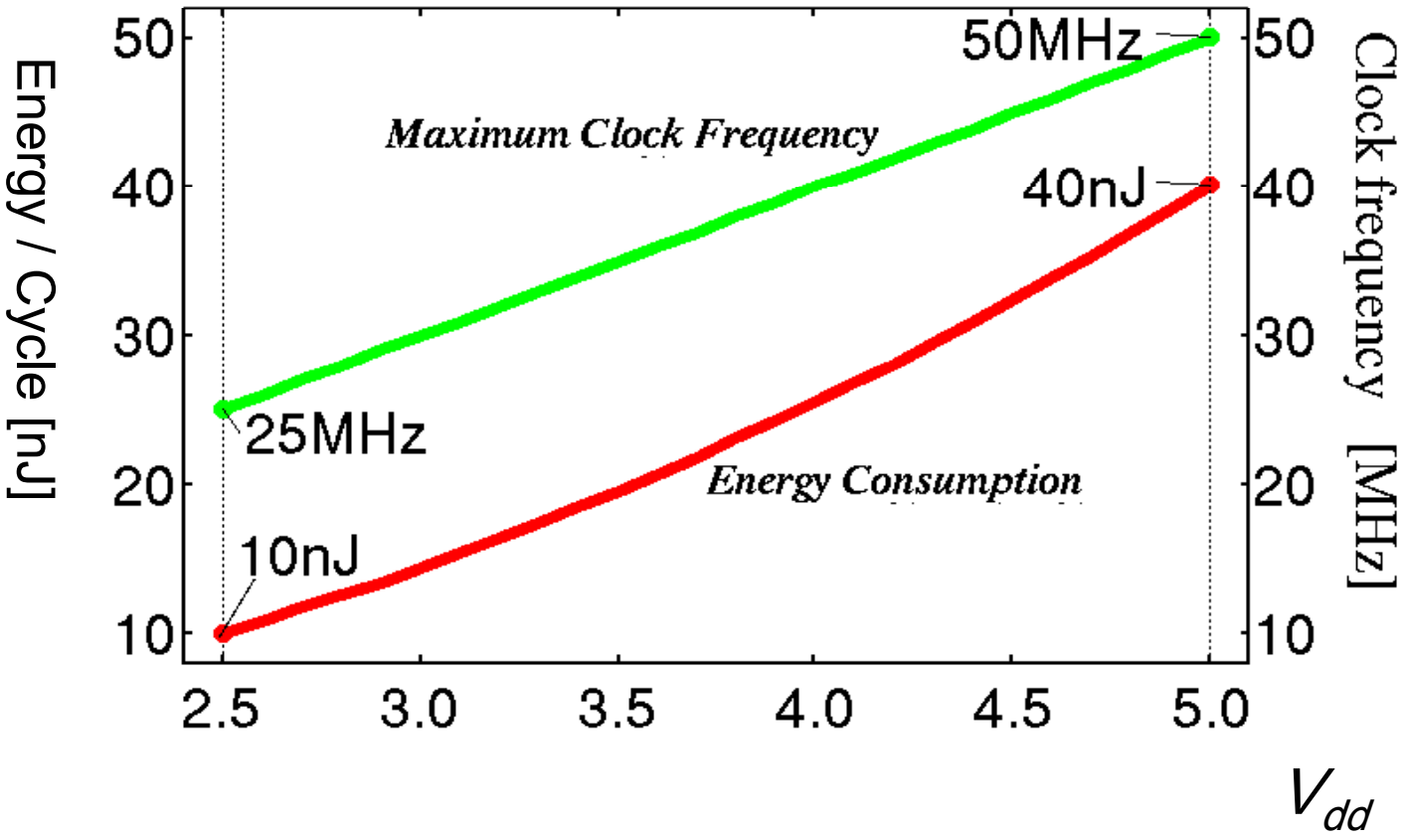
$$P = \alpha \, C_L \, V_{dd}^2 \, f$$

$$E = \alpha \, C_L \, V_{dd}^2 \, f \, t = \alpha \, C_L \, V_{dd}^2 \, \#Cycles$$

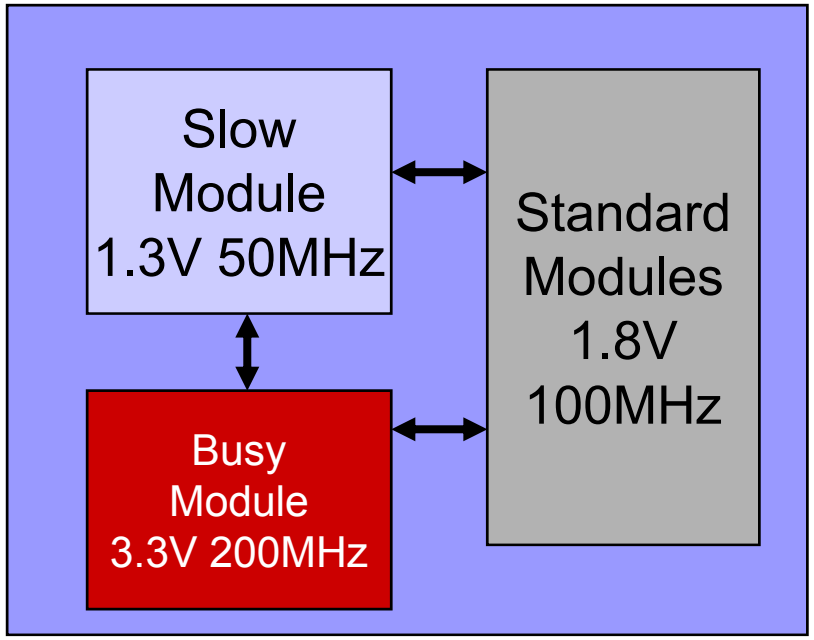
Saving Energy under given Time Constraints:

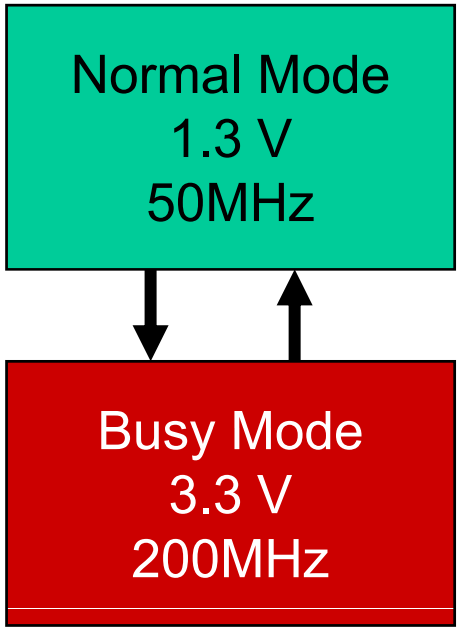
- Reduce the supply voltage V_{dd}
- Reduce switching activity α
- Reduce the load capacitance C_L
- Reduce the number of cycles *#Cycles*

تغییر مقیاس ولتاژ و مدیریت توان، تغییر مقیاس پویای ولتاژ
Voltage Scaling and Power Management Dynamic Voltage Scaling



مدیریت منبع ولتاژ ایستا در مقابل پویا
Spatial vs. Dynamic Supply Voltage Management



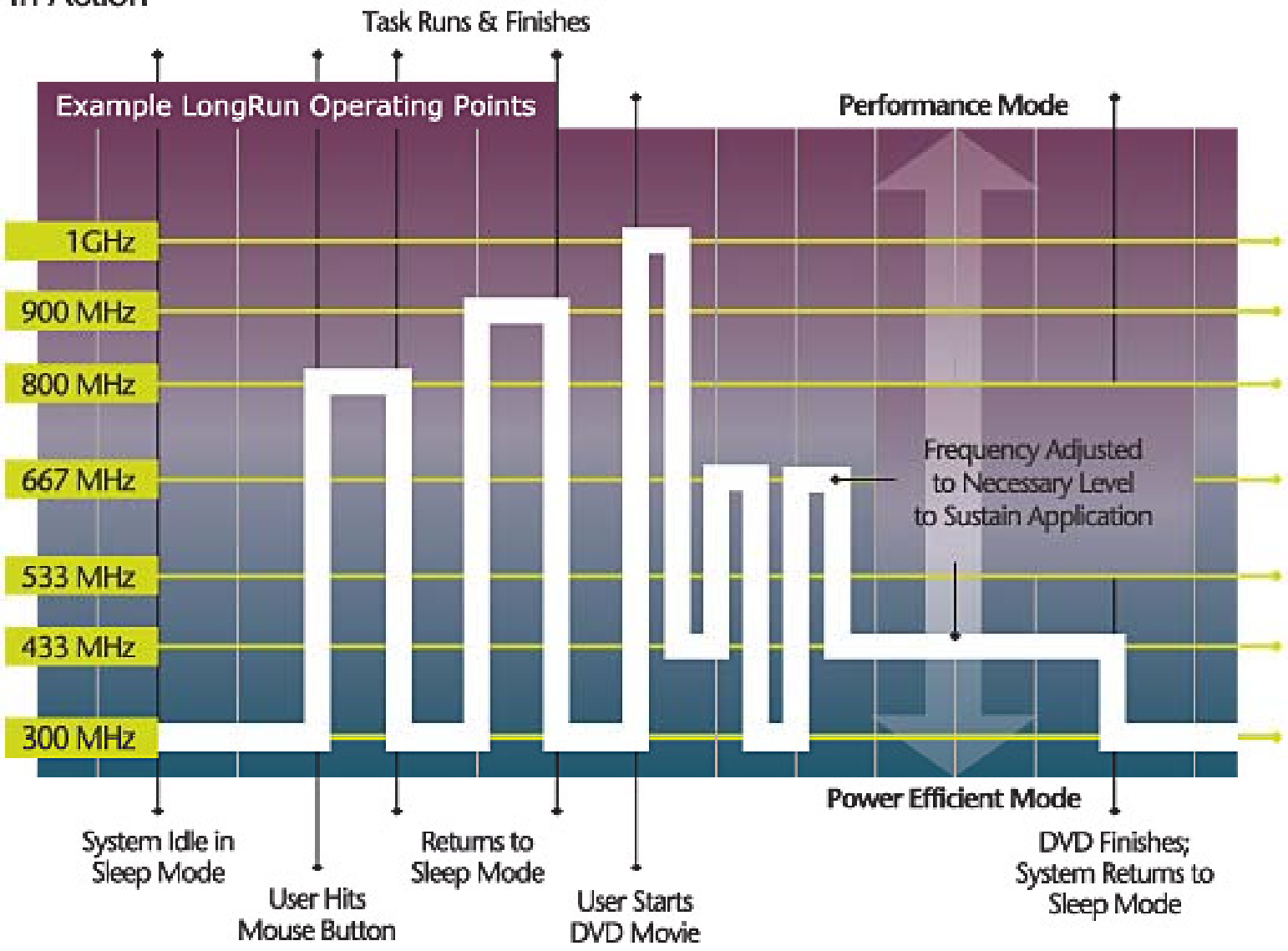


Analogy of biological blood systems:

- Different supply to different regions
- **High pressure:** High pulse count and High activity
- **Low pressure:** Low pulse count and Low activity

Transmeta™ LongRun™ Power Management

In Action



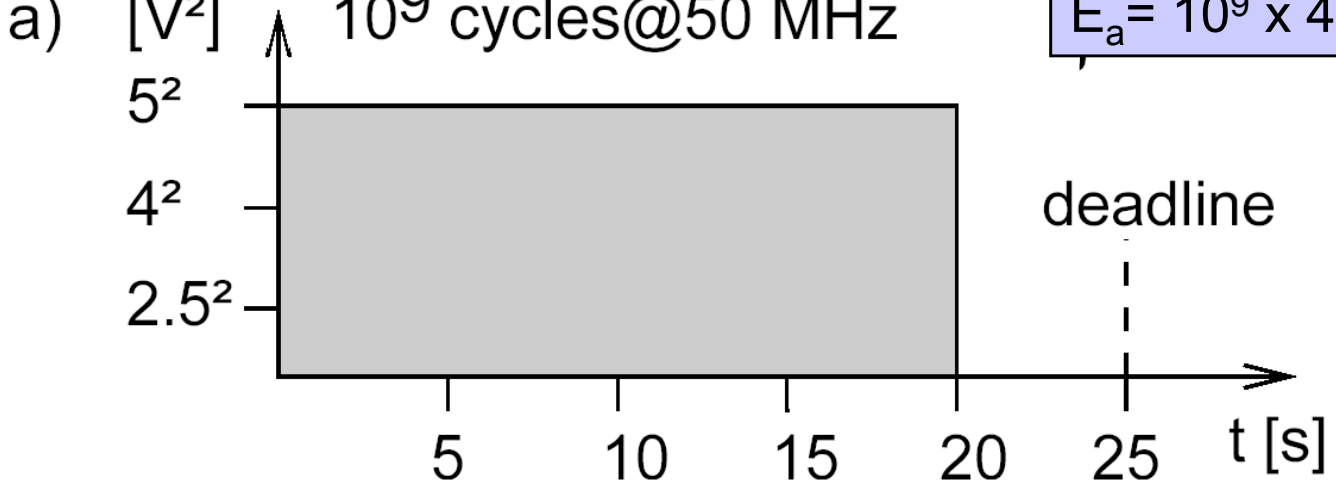
Example: Processor with 3 voltages
Case a): Complete task ASAP

V_{dd} [V]	5.0	4.0	2.5
Energy per cycle [nJ]	40	25	10
f_{max} [MHz]	50	40	25
cycle time [ns]	20	25	40

Task that needs to execute 10^9 cycles within 25 seconds.

a) [V²] 10^9 cycles@50 MHz

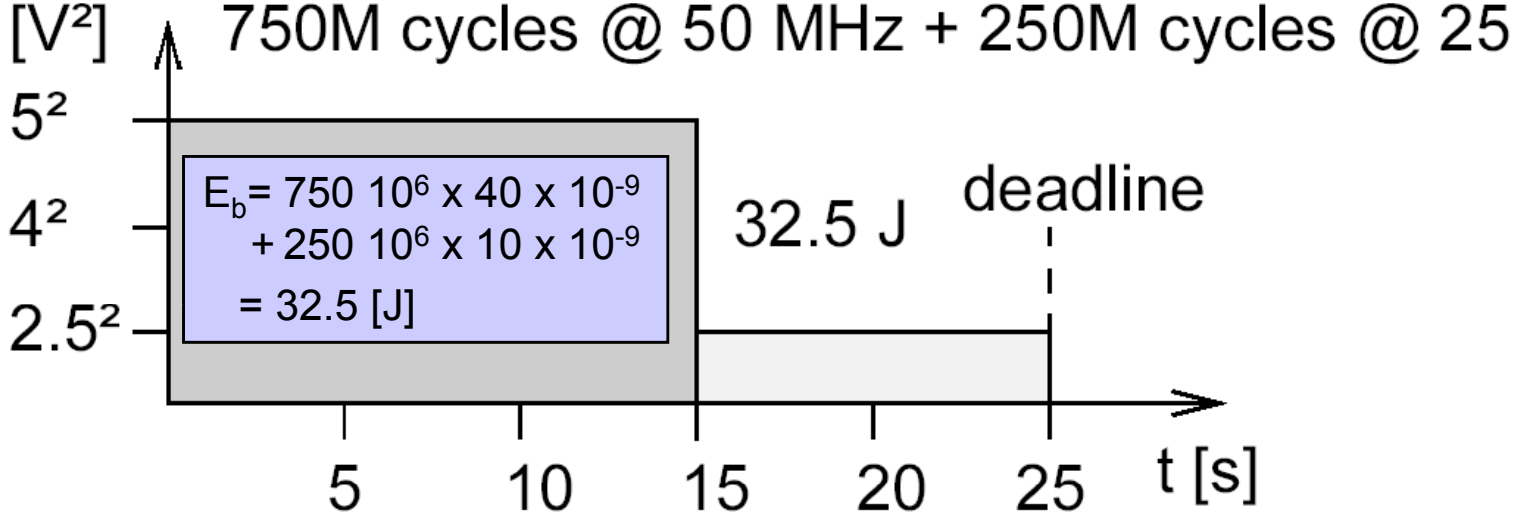
$E_a = 10^9 \times 40 \times 10^{-9} = 40$ [J]



Case b): Two voltages

V_{dd} [V]	5.0	4.0	2.5
Energy per cycle [nJ]	40	25	10
f_{max} [MHz]	50	40	25
cycle time [ns]	20	25	40

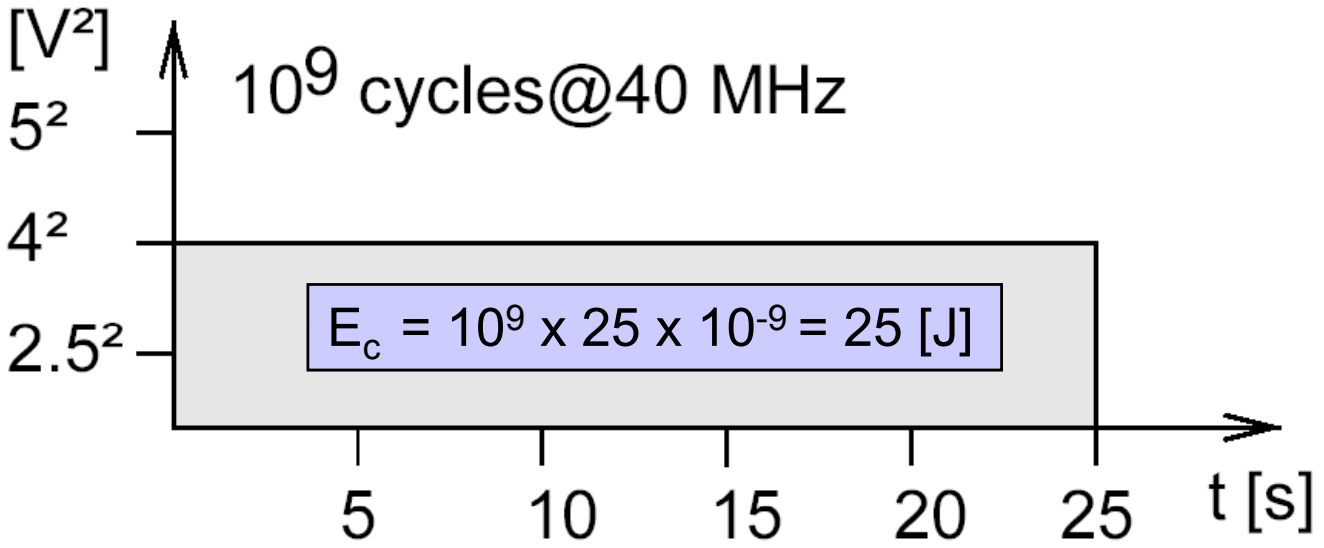
b)



Case c): Optimal voltage

V_{dd} [V]	5.0	4.0	2.5
Energy per cycle [nJ]	40	25	10
f_{max} [MHz]	50	40	25
cycle time [ns]	20	25	40

c)



مشاهدات
Observations

☞ A minimum energy consumption is achieved for the ideal supply voltage of 4 Volts.

In the following: **variable voltage processor =** processor that allows **any** supply voltage up to a certain maximum.

It is expensive to support truly variable voltages, and therefore, actual processors support only a few fixed voltages.

Ishihara, Yasuura: "Voltage scheduling problem for dynamically variable voltage processors", Proc. of the 1998 International Symposium on Low Power Electronics and Design (ISLPED'98)

تعمیم
Generalization

Lemma [Ishihara, Yasuura]:

- If a variable voltage processor completes a task before the deadline, then the energy consumption can be reduced.
- If a processor uses a single supply voltage V and completes a task T just at its deadline, then V is the unique supply voltage which minimizes the energy consumption of T.
- If a processor can only use a number of discrete voltage levels, then a voltage schedule with at most two voltages minimizes the energy consumption under any time constraint.
- If a processor can only use a number of discrete voltage levels, then the two voltages which minimize the energy consumption are the two immediate neighbors of the ideal voltage V_{ideal} possible for a variable voltage processor.

مورد وظایف چندگانه: انتساب بهینه‌ی ولتاژها به مجموعه‌ای از وظایف
Assigning optimum voltages to a set of tasks The case of multiple tasks:

N : the number of tasks

EC_j : the number of execution cycles of task j

L : the number of voltages of the target processor

V_i : the i^{th} voltage, with $1 \leq i \leq L$

F_i : the clock frequency for supply voltage V_i

T : the global deadline at which all tasks must have been completed

SC_j : the average switching capacitance during the execution of task j (SC_j comprises the actual capacitance CL and the switching activity α)

$X_{i,j}$: the number of clock cycles task j is executed at voltage V_i

طراحی یک مدل برنامه‌ریزی صحیح
Designing an IP model

Simplifying assumptions of the IP-model include the following:

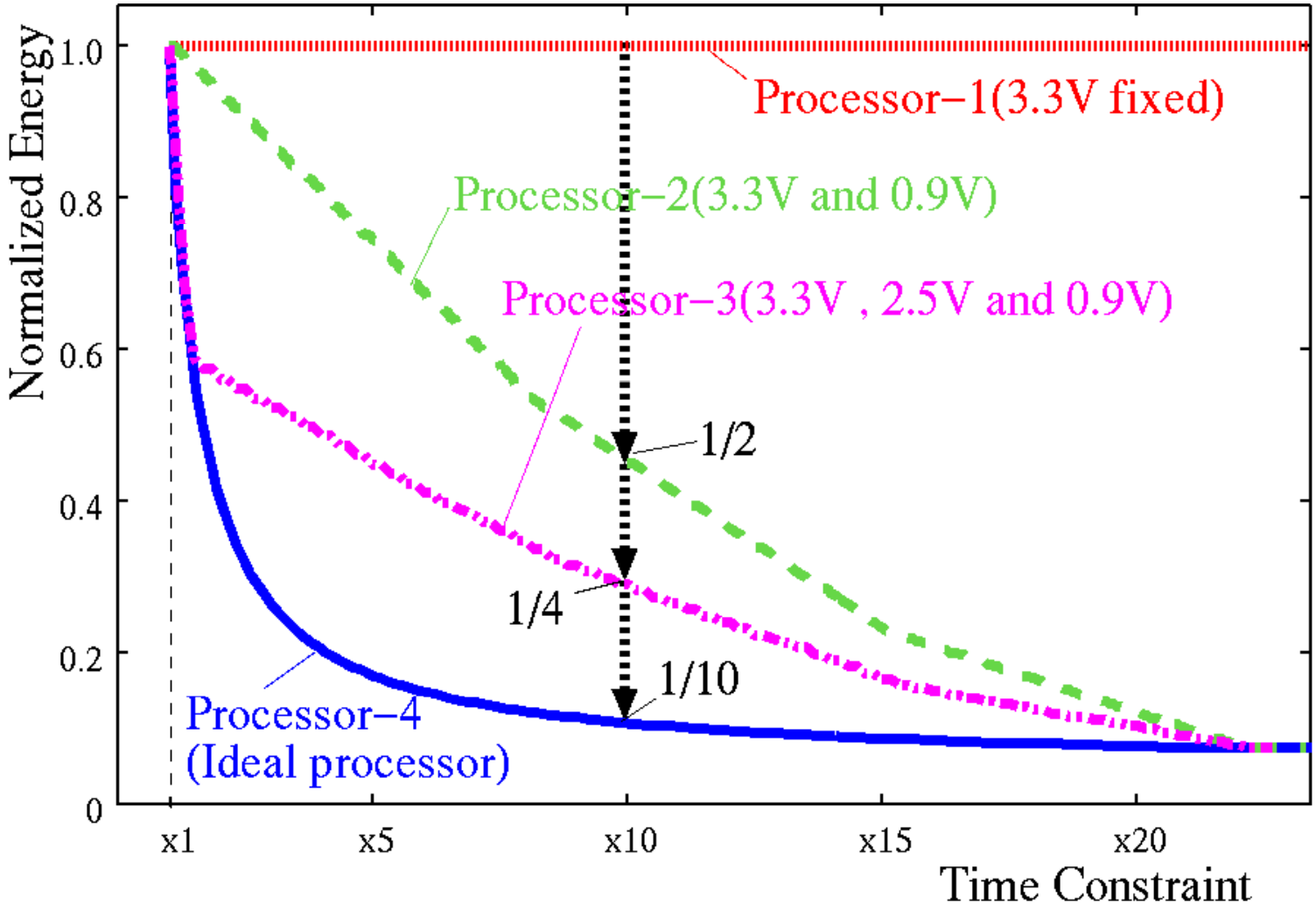
- There is one target processor that can be operated at a limited number of discrete voltages.
- The time for voltage and frequency switches is negligible.
- The worst case number of cycles for each task are known.

Minimize $E = \sum_{j=1}^{N} \sum_{i=1}^{L} SC_j \cdot x_{i,j} \cdot V_i^2$

Subject to $\forall j : \sum_{i=1}^{L} x_{i,j} = EC_j$

and $\sum_{i=1}^{L} \sum_{j=1}^{N} \frac{x_{i,j}}{F_i} \leq T$

نتایج تجربی
Experimental Results



تکنیک‌های زمان‌بندی ولتاژ
Voltage Scheduling Techniques

- **Static Voltage Scheduling**
  - Extension: Deadline for each task
  - Formulation as IP problem (SS)
  - Decisions taken at compile time
- **Dynamic Voltage Scheduling**
  - Decisions taken at run time
  - 2 Variants:
    - arrival times of tasks is known (SD)
    - arrival times of tasks is unknown (DD)

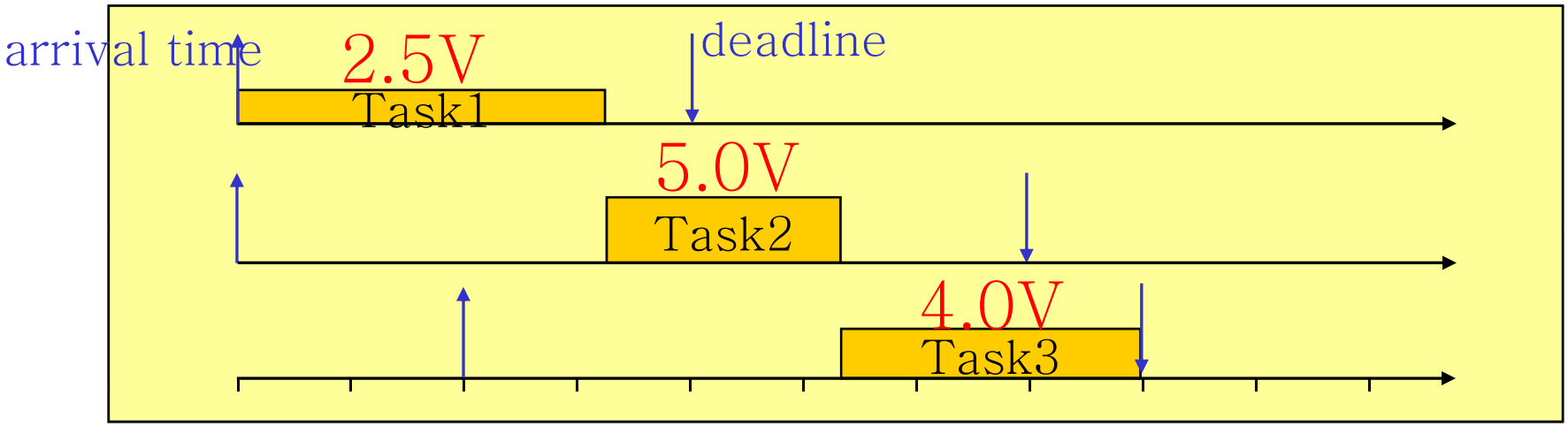
کنترل ولتاژ پویا با سیستم‌های عامل
Dynamic Voltage Control by Operating Systems

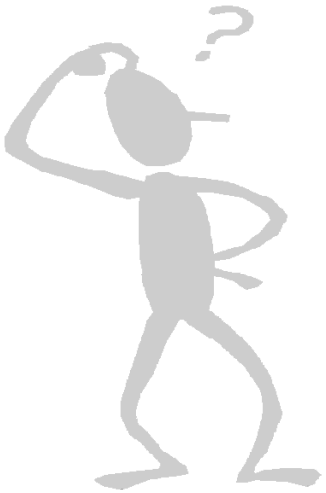
Voltage Control and Task Scheduling by Operating System to minimize energy consumption

Okuma, Ishihara, and Yasuura: “Real-Time Task Scheduling for a Variable Voltage Processor”, Proc. of the 1999 International Symposium on System Synthesis (ISSS'99)

Target:
- single processor system
- Only OS can issue voltage control instructions
- Voltage can be changed anytime
- only one supply voltage is used at any time
- overhead for switching is negligible
- static determination of worst case execution cycles

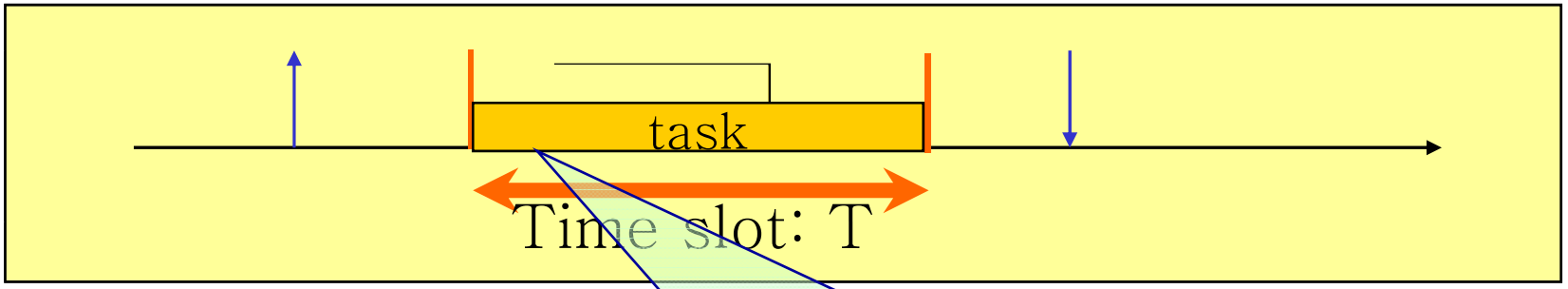
مساله‌ی سیستم‌های عامل
Problem for Operating Systems





What is the optimum supply voltage assignment for each task in order to obtain minimum energy consumption?

سیاست پیشنهاد شده
The proposed Policy



Consider a time slot the task can use without violating real-time constraints of other tasks executed in the future

Once time slot is determined:
- The task is executed at a frequency of WCEC / T Hz
- The scheduler assigns start and end times of time slot

دو الگوریتم
Two Algorithms

Two possible situations:

- The arrival time of tasks is known:
  SD Algorithm
  Static ordering and Dynamic voltage assignment
- The arrival time of tasks is unknown
  DD Algorithm
  Dynamic ordering and Dynamic voltage assignment

	SD	DD
CPU Time Allocation	off-line	on-line
Start Time Assignment	on-line	on-line
End Time Prediction	off-line	on-line

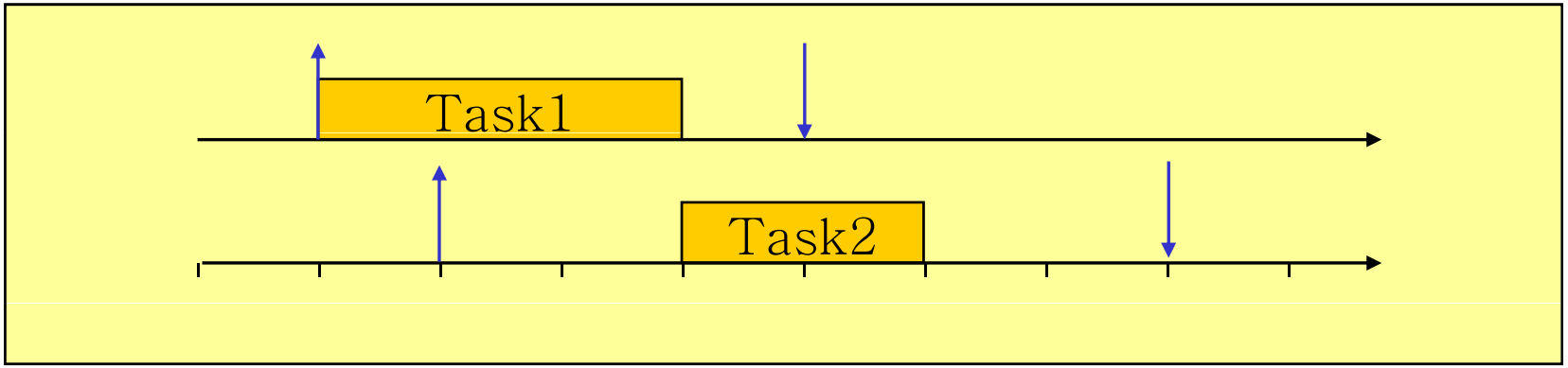
SD Algorithm (CPU Time Allocation)

- Arrival time of all tasks is known
- Deadline of all tasks is known
- WCEC of all tasks is known

→CPU time can be allocated statically

CPU time is assigned to each task:

- assuming maximum supply voltage
- assuming WCEC

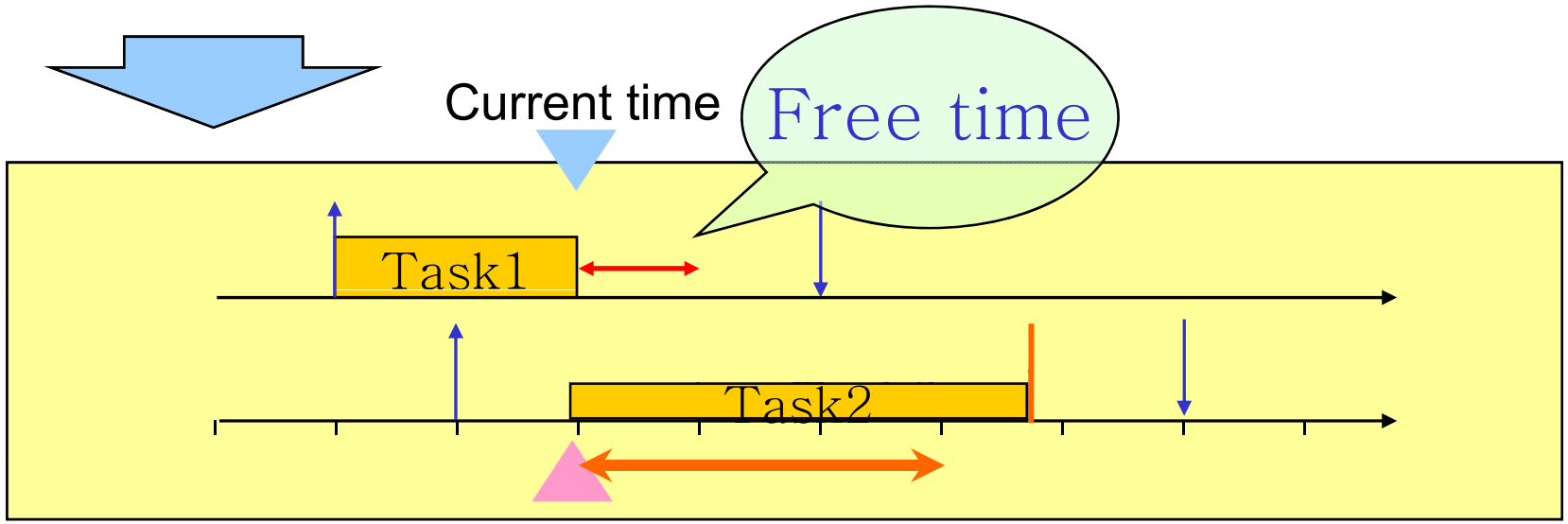


SD Algorithm (Start Time Assignment)

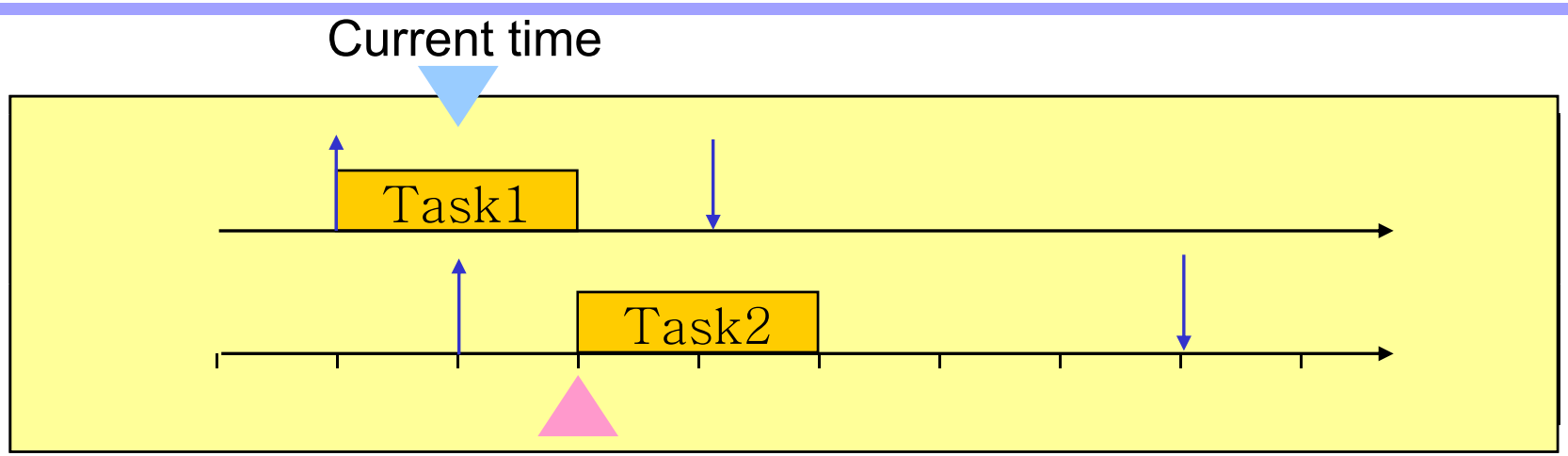
Current time

In SD, it is possible to assign lower supply voltage to Task2 using the free time

• In SS, the scheduler can't use the free time because it has statically assigned voltage



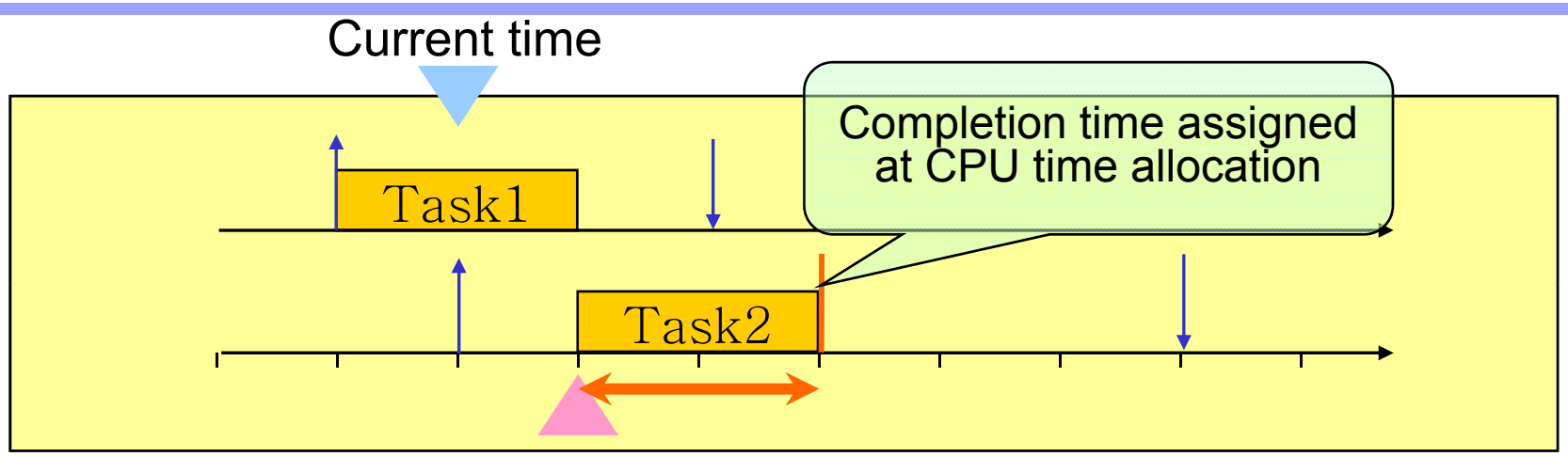
DD Algorithm



Start Time Assignment:

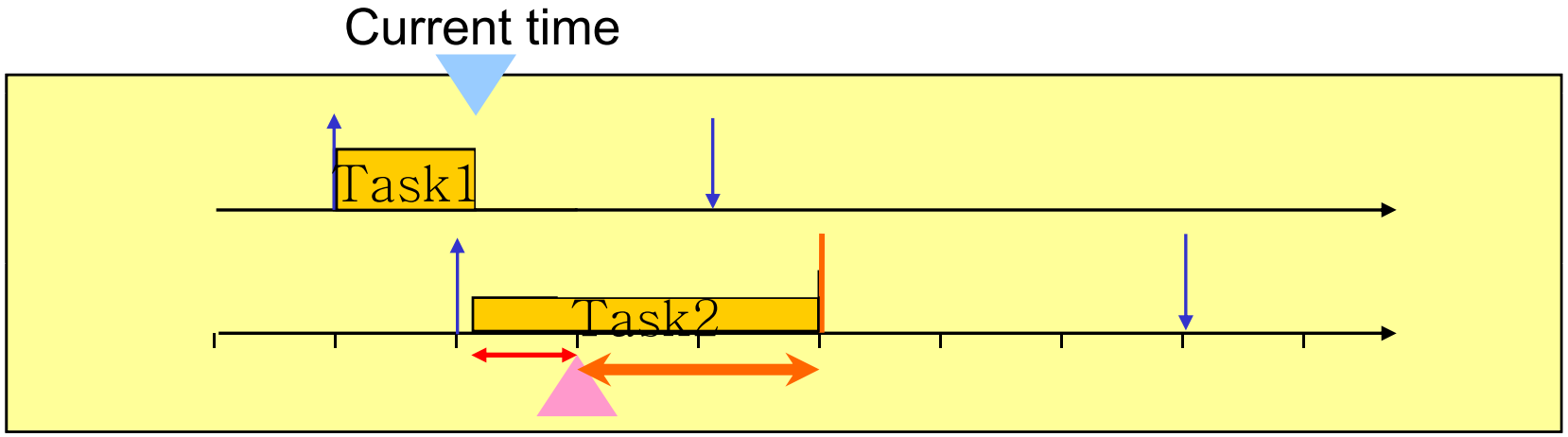
- New task arrives – it either:
  a) Preempts currently executing task
  b) Starts right after currently executing task
  → Starting time is determined

DD Algorithm (cont.)



End Time Prediction:
Based on the currently executing task's end time prediction, add the new task's WCEC time at maximum voltage

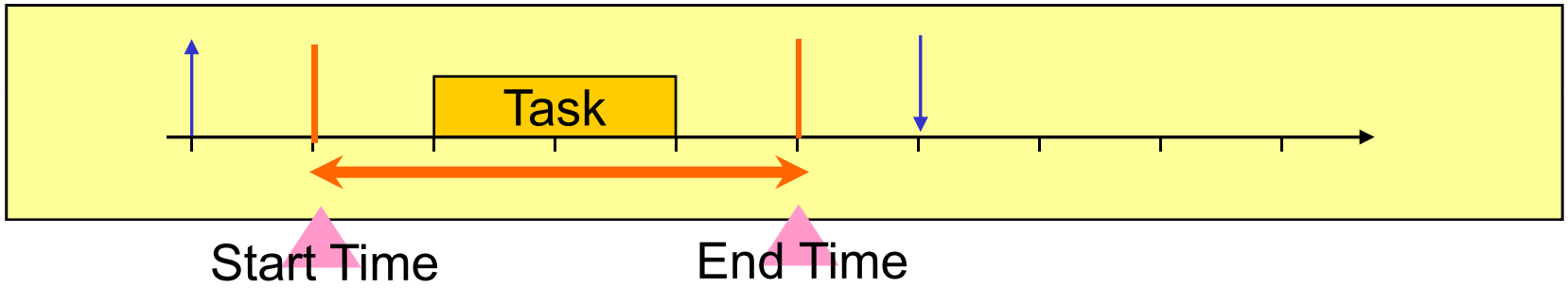
DD Algorithm (cont.)



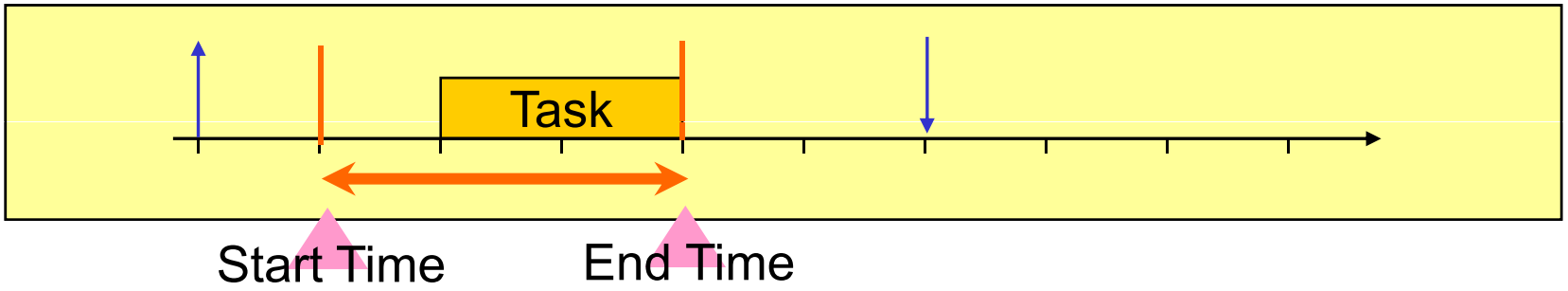
→ If the currently executing task finishes earlier, then new task can start sooner and run slower at lower voltage

Comparison: SD vs. DD

→ SD Algorithm:



→ DD Algorithm:



Experimental Results: Energy

Normal: Processor runs at maximum supply voltage
SS: Static Scheduling
SD: Scheduling done by SD Algorithm
DD: Scheduling done by DD Algorithm

Task	Normal	SS	SD	DD
1	5.0V	4.0V	4.0V	5.0V
2	5.0V	4.0V	4.0V	5.0V
3	5.0V	5.0V	4.0V	5.0V
4	5.0V	2.5V	2.5V	5.0V
3	5.0V	2.5V	2.5V	4.0V
5	5.0V	2.5V	2.5V	4.0V
Energy	**1615J**	**702J**	**685J**	**1357J**

مدیریت توان پویا
Dynamic power management (DPM)

Dynamic Power management tries to assign optimal **power saving states**

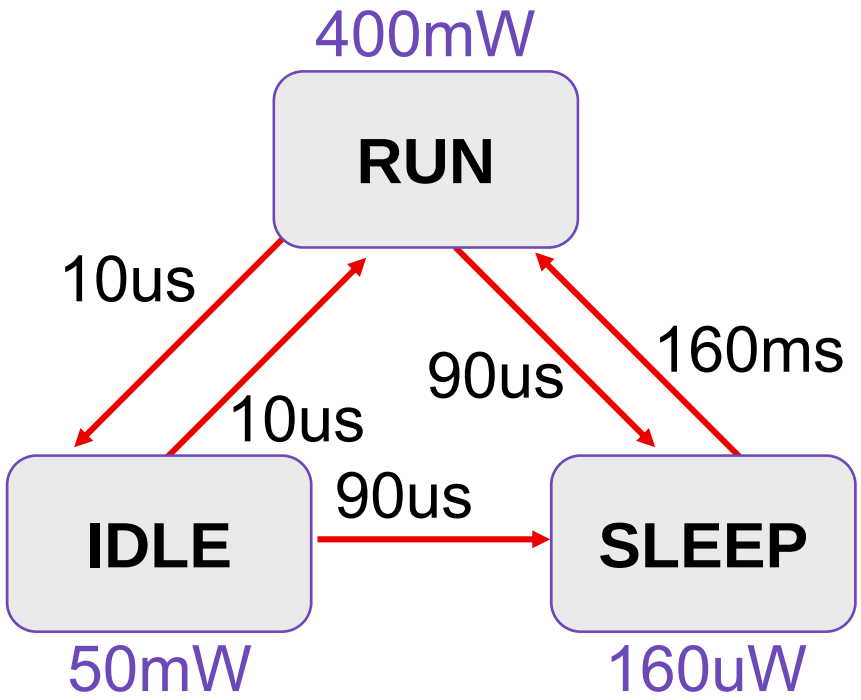
Requires Hardware Support

Example: StrongARM SA1100

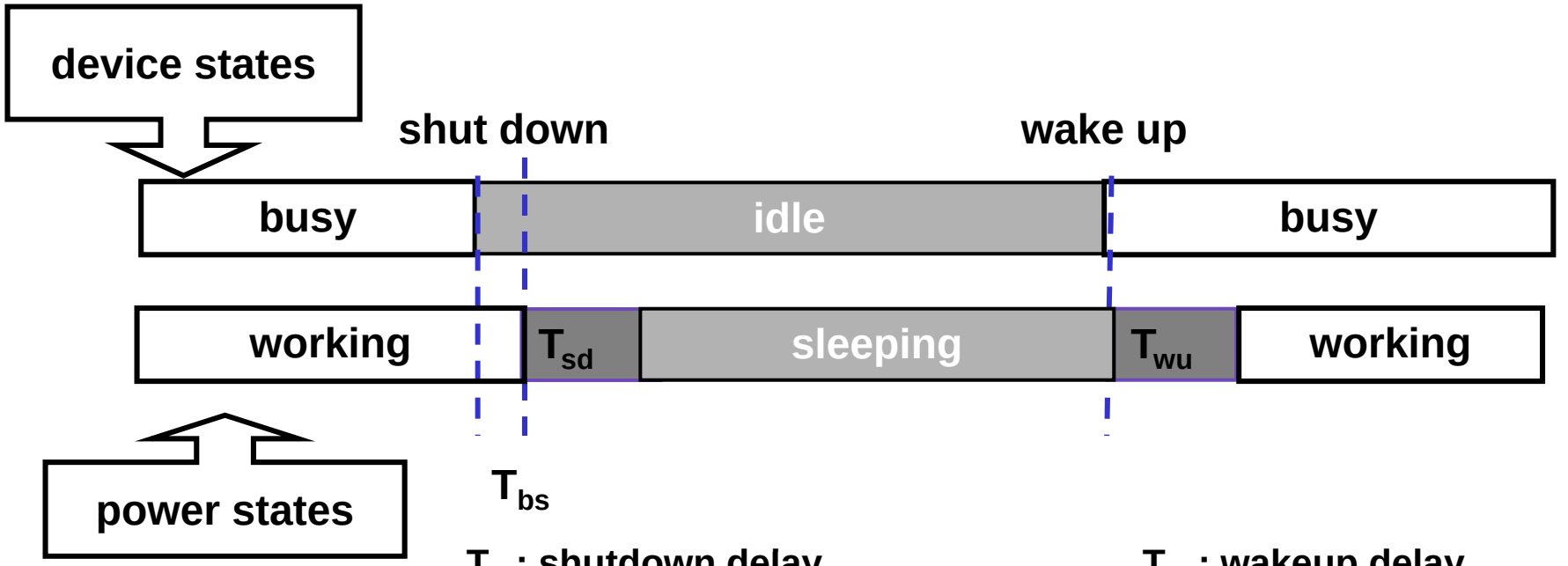
RUN: operational

IDLE: a sw routine may stop the CPU when not in use, while monitoring interrupts

SLEEP: Shutdown of on-chip activity



The opportunity: Reduce power according to workload



T_{sd}: shutdown delay

T_{wu}: wakeup delay

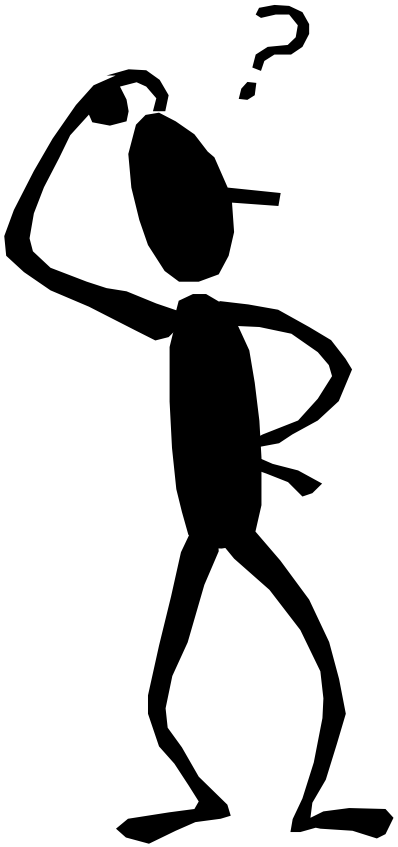
T_{bs}: time before shutdown

T_{bw}: time before wakeup

Desired: Shutdown only during long idle times

→ Tradeoff between savings and overhead

چالش
The challenge



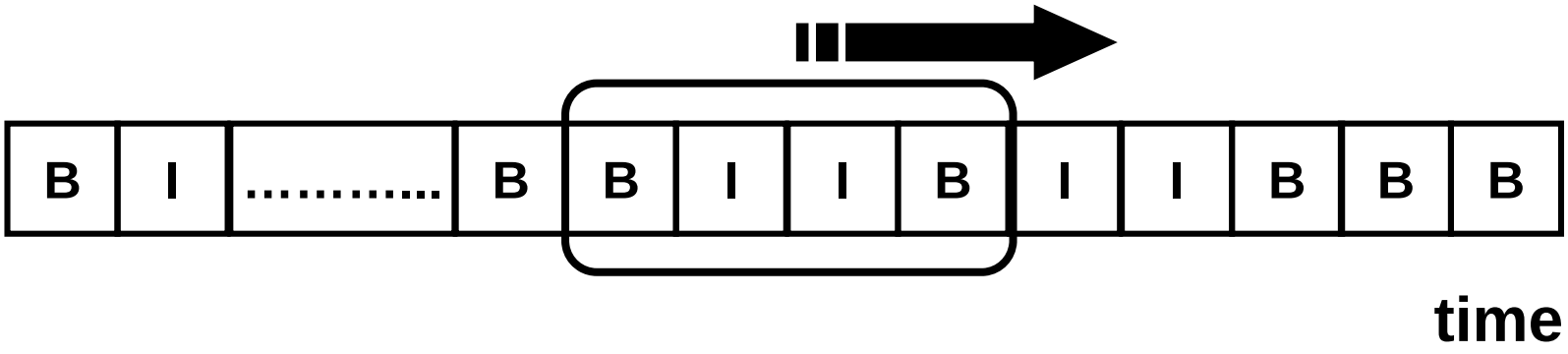
Questions:

- When to go to a power-saving state?
- Is an idle period long enough for shutdown?

→ Predicting the future

مدل‌های تصادفی تطبیقی
Adaptive Stochastic Models

Sliding Window (SW): [Chung *DATE 99*]



- Interpolating pre-computed optimization tables to determine power states
- Using sliding windows to adapt to non-stationarity

مقایسه‌ی رهیافت‌های مختلف
Comparison of different approaches

Algorithm	P	N_{sd}	N_{wd}
off-line	0.33	250	0
Semi-Markov	0.40	326	76
Sliding Window	0.43	191	28
Device-Specific Timeout	0.44	323	64
Learning Tree	0.46	437	217
Exponential Average	0.50	623	427
always on	0.95	-	-

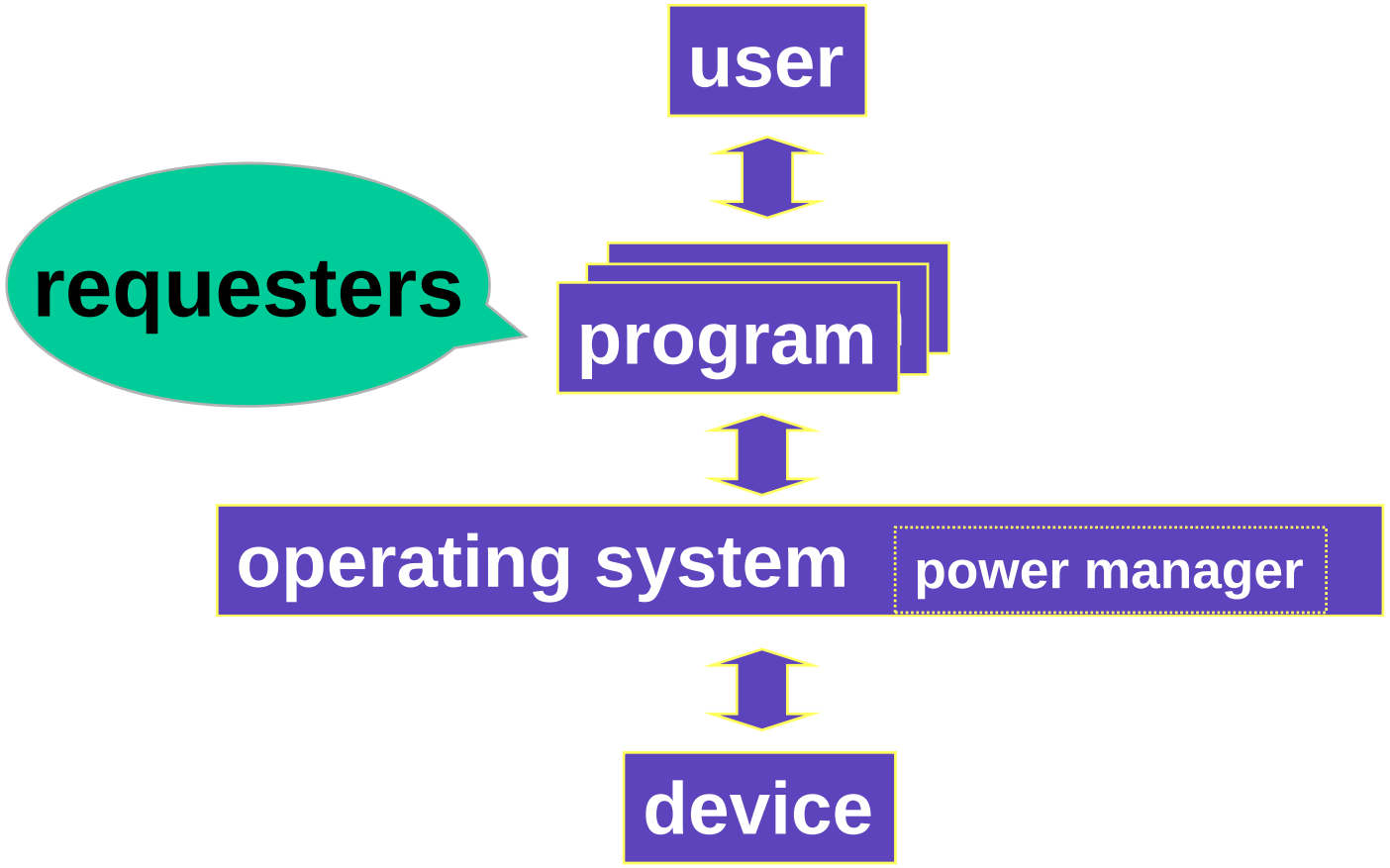
P : average power

N_{sd}: number of shutdowns

N_{wd} : wrong shutdowns (actually waste energy)

در مورد چندوظیفه‌ای؟
What about multitasking?

→ Coordinate multiple workload sources



درخواست کننده‌ها
Requesters

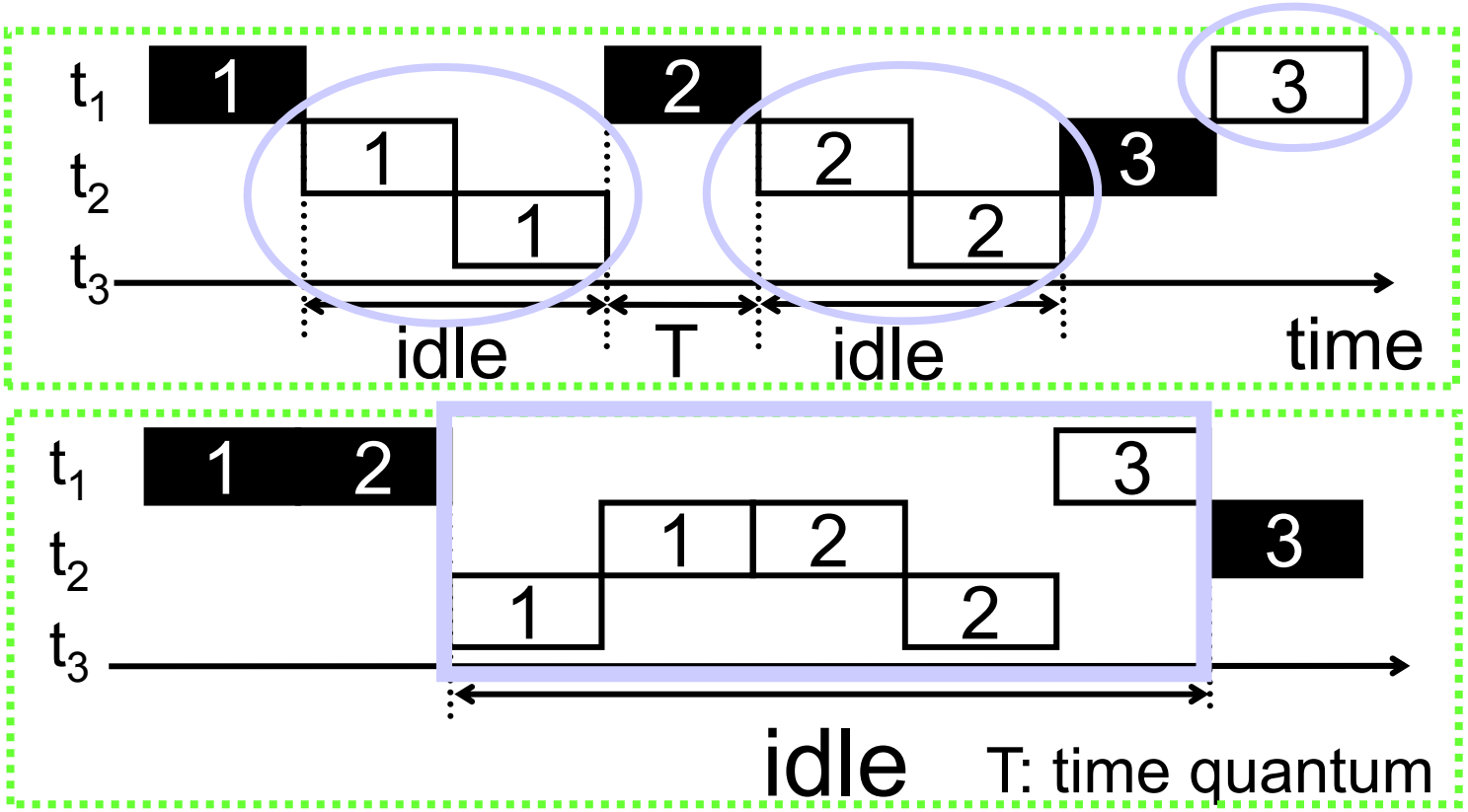
Concurrent processes
- Created, executed, and terminated
- Have different device utilization
- Generate requests only when running (occupy CPU)

Power manager is notified when processes change state

We use processes to represent requesters
requester = process

زمان‌بندی وظایف
Task Scheduling

Rearrange task execution to cluster similar utilization and idle periods



Shutdown-friendly scheduling

- Cluster processes with similar utilization patterns
  → Localize resource usage in time
- Tradeoff against latency
  → A process may be delayed
- Exploit application-knowledge
  → Processes can specify their latency requirements via API calls

→Task scheduling reduces both power and overhead!

به سوی سیستم‌های عامل آگاه از انرژی
Towards the energy aware OS

- Power control for devices and CPU
  - Shutdown
  - Variable frequency
  - Variable voltage
- Application awareness
  - Process aware
  - API for interacting with applications
- Concurrency management
  - Scheduling
  - Mutual exclusion (shared resources)
- Minimize OS impact (lightweight OS)

پیاده‌سازی‌های سیستم عامل آگاه از انرژی
Power-aware OS implementations

- Windows APM and ACPI
  - Device-centric, shutdown based
- Power-aware Linux
  - Good research platform (several partial implementations, es. U. Delft, Compaq, etc.)
  - Quite high-overhead for low-end embedded systems
- Power-aware ECOS
  - Good research platform (HP-Unibo implementation)
  - Lower overhead than Linux, modular
- Micro OSes

مدیریت توان پویای آگاه از کاربرد

Application Aware DPM – Example: Communication Power

NICs powered by portables reduce **battery life**

8 hours

2.5 hours

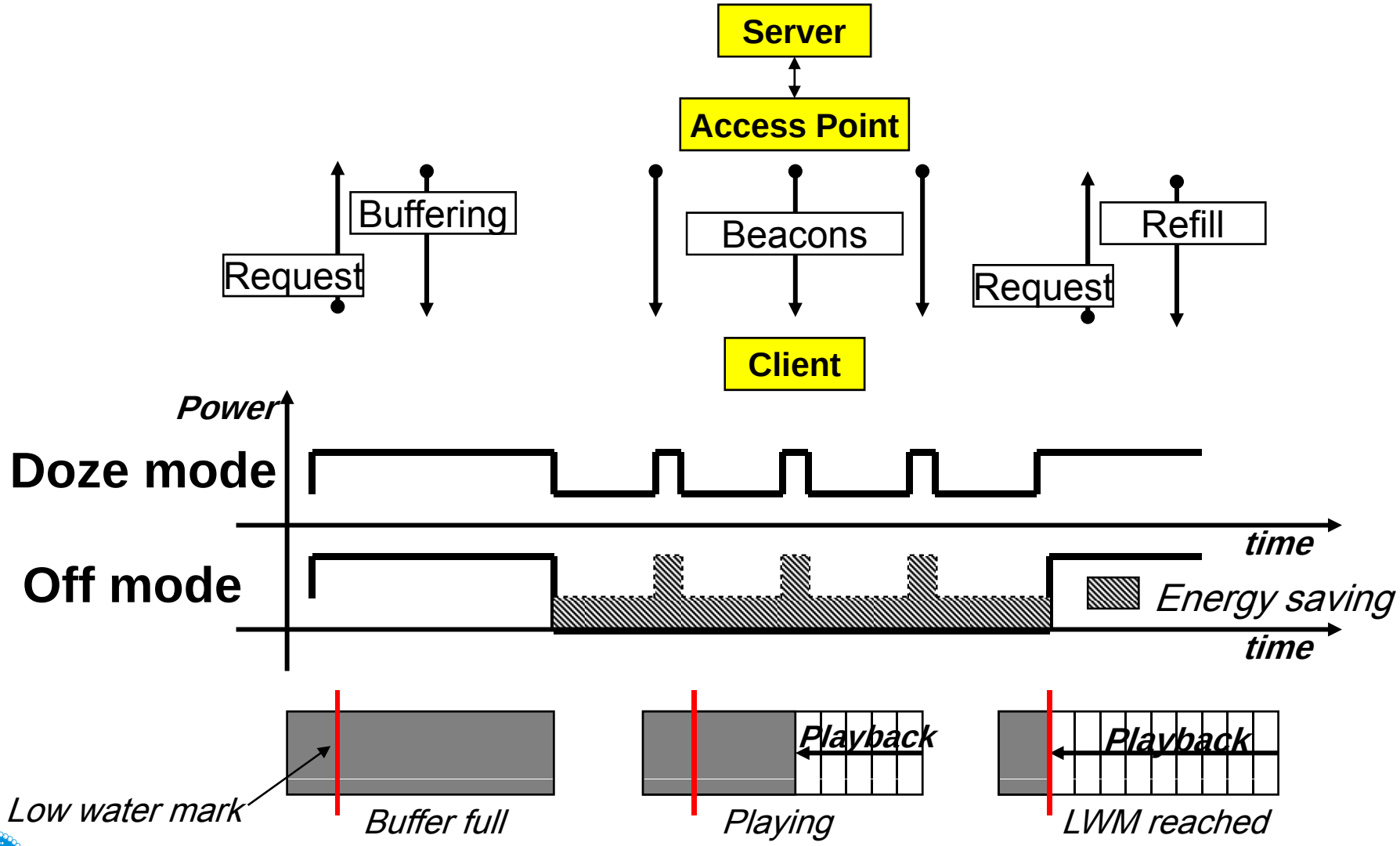
- In general:
  Higher bit rates lead to higher power consumption
- 90% of power for listening to a radio channel

→ Proper use of PHY layer services by MAC is critical!

NIC Power States

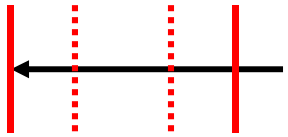
- Transmit mode
- Receive mode
- Doze mode
- Off-mode
  - NIC completely turned off
  - Power overhead for frequent switches
  - Must come with proper **buffering strategies**

Off mode power savings

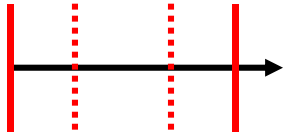


LWM / Buffer characteristics

Where to put the LWM?

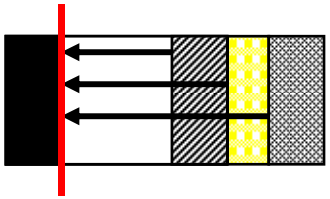


- Higher error probability
- Exploits NIC off-state
- Min. value to allow data acquisition



- lower error probability
- Incurs NIC off-state overhead
- Max. value: Buffer_length–1 block

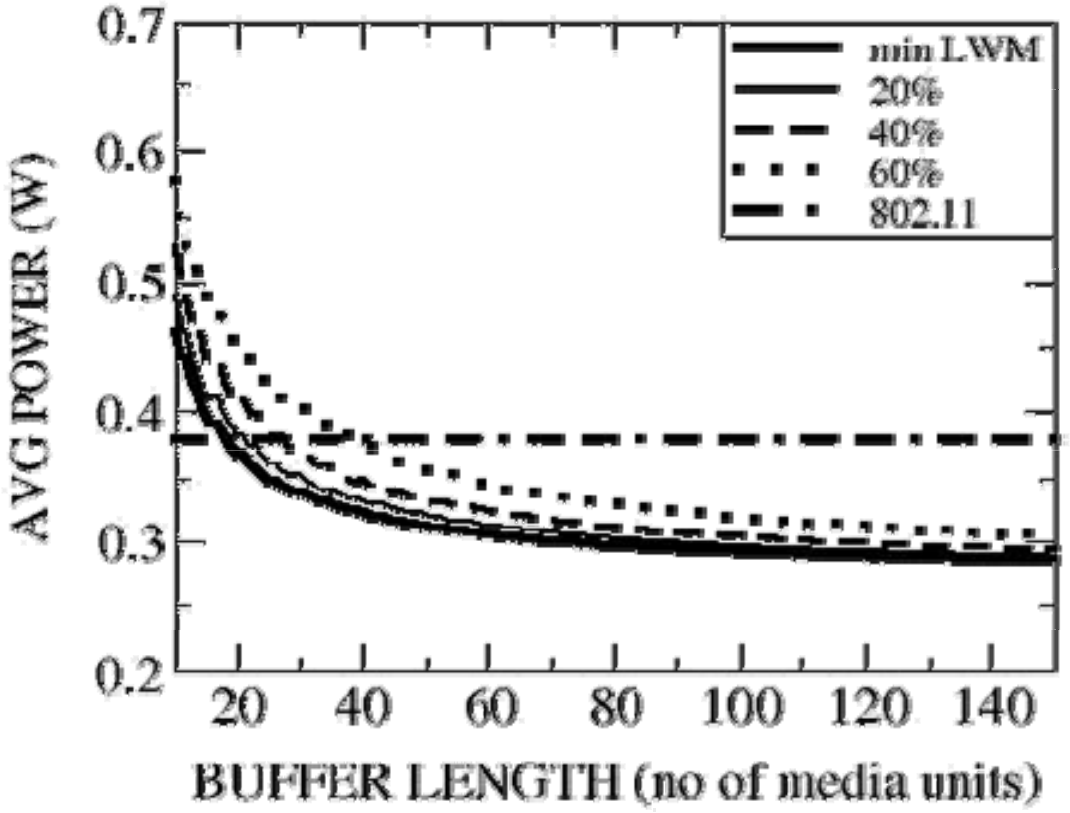
How long should the buffer be?



- Depends on memory availability
- The longer the buffer, the higher the NIC off-state benefits

Buffering Strategies should be Power Aware!

Comparison



- Low length buffers incur off mode power overhead
- Good power saving for high length buffers

بکارگیری دانش کاربرد
Exploiting application knowledge

Approximate processing [Chandrakasan98-01]

- Tradeoff quality for energy (es. lossy compression)
- Design algorithms for graceful degradation

Enforce power-efficiency in programming

- Avoid repetitive polling [Intel98]
- Use event-based activation (interrupts)

Localize computation whenever possible

- Helps shutdown of peripherals
- Helps shutdown of memories