

Energy Aware Real-Time Scheduling

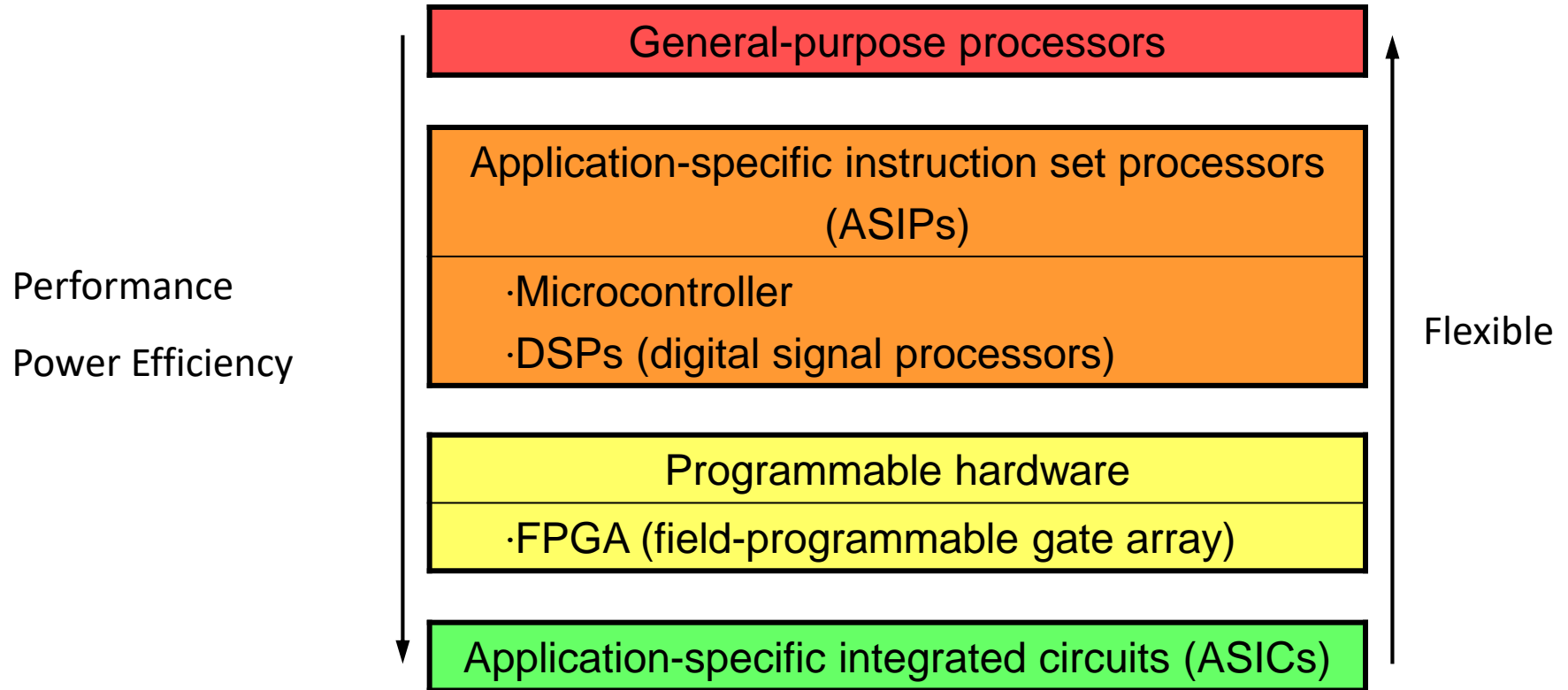
Embedded System Software Design

Prof. Ya-Shu Chen
National Taiwan University of
Science and Technology

Outline

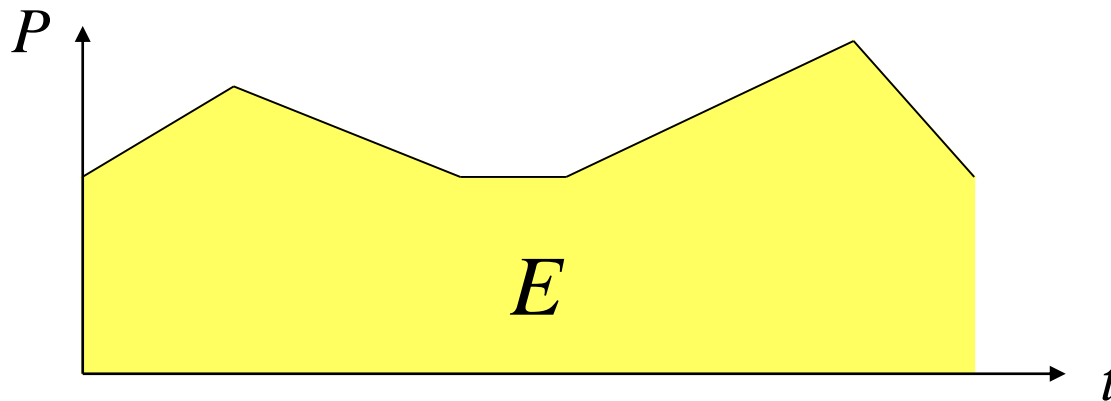
- Power and Energy
- Basic Techniques
- Scheduling Issues

Implementation Alternatives



Power and Energy are Related

$$E = \int P(t)dt$$



In many cases, faster execution also means less energy, but the opposite may be true if power has to be increased to allow faster execution.

Dynamic Voltage Scaling (DVS)

Power consumption of CMOS circuit (ignoring leakage):

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

V_{dd} : supply voltage

α : switching activity

C_L : load capacity

f : clock frequency

Delay for CMOS circuits:

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

V_{dd} : supply voltage

V_T : threshold voltage

$$V_T \ll V_{dd}$$

Decreasing V_{dd} reduces P quadratically (f constant).

The gate delay increases only linearly.

Maximal frequency f_{\max} decreases linearly.

Potential for Energy Optimization

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

$$E \approx \alpha C_L V_{dd}^2 f t = \alpha C_L V_{dd}^2 (\# \text{cycles})$$

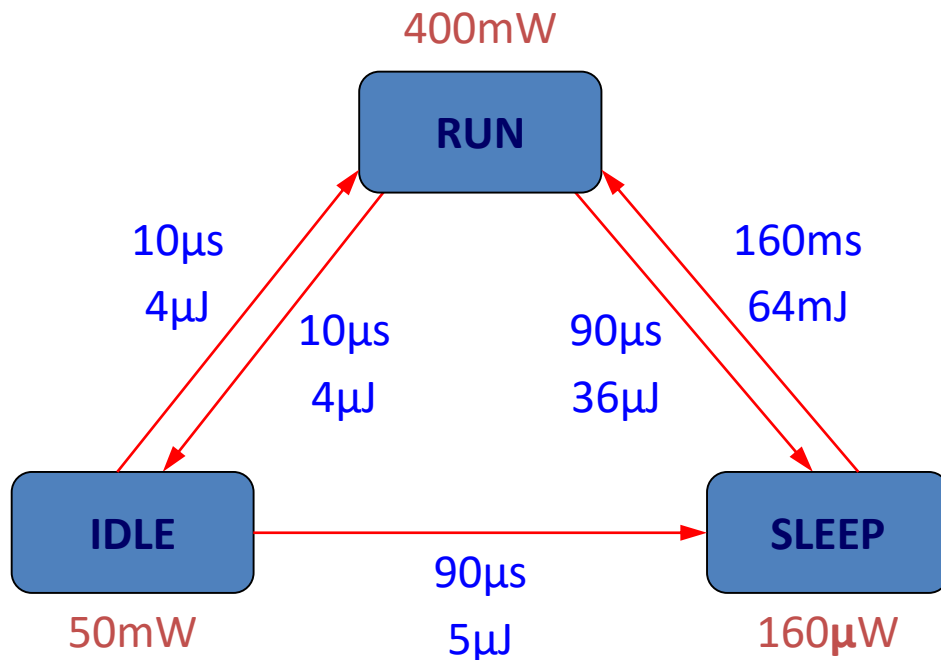
Saving energy for a given task:

- Reduce the supply voltage V_{dd}
- Reduce switching activity α
- Reduce the load capacitance C_L
- Reduce the number of cycles $\# \text{cycles}$

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

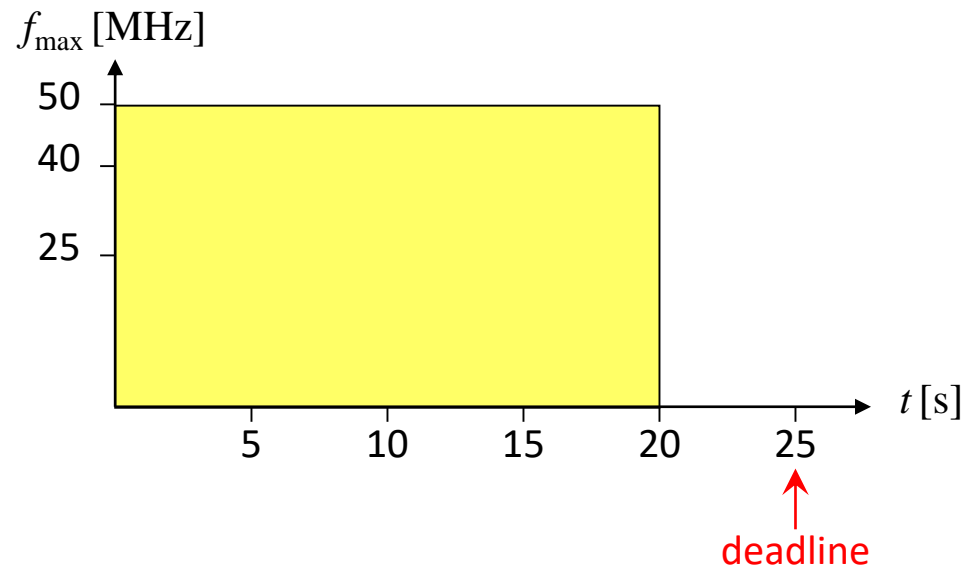
DVS Example:

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) 10^9 cycles @ 50 MHz



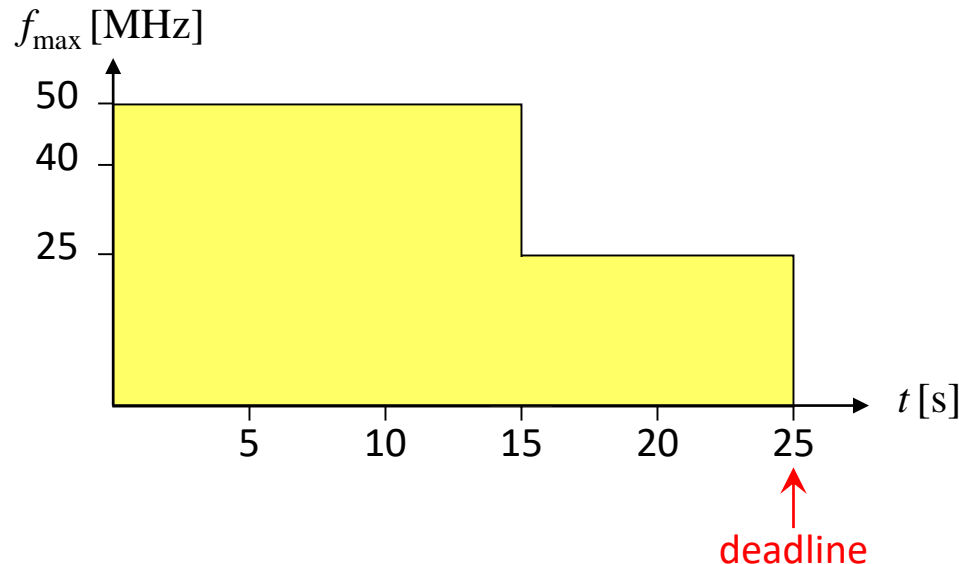
$$E_a = 10^9 \times 40 \times 10^{-9} \\ = 40 \text{ [J]}$$

DVS Example: 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

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

b) 750M cycles @ 50 MHz + 250M cycles @ 25 MHz



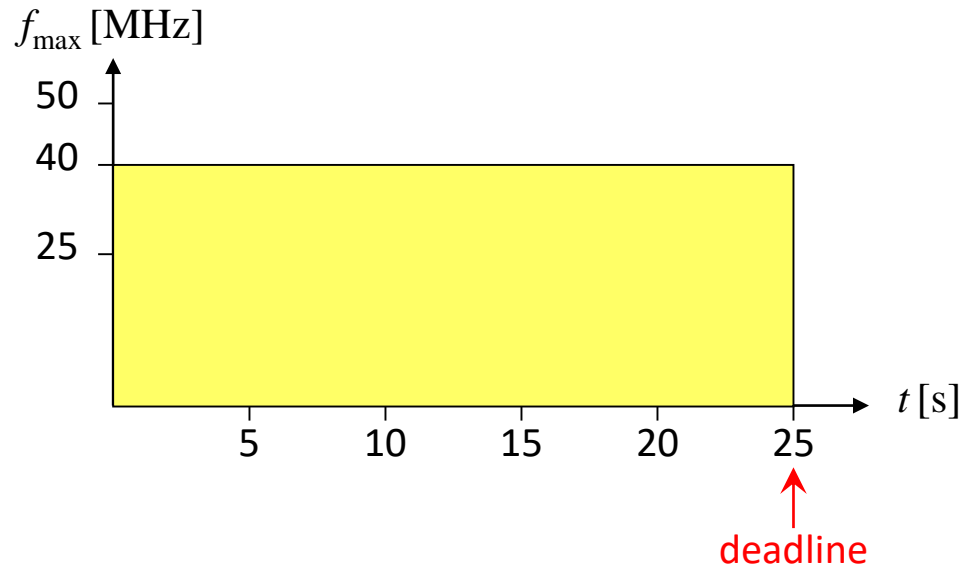
$$\begin{aligned} E_b &= 750 \times 10^6 \times 40 \times 10^{-9} \\ &\quad + 250 \times 10^6 \times 10 \times 10^{-9} \\ &= 32.5 \text{ [J]} \end{aligned}$$

DVS Example: 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

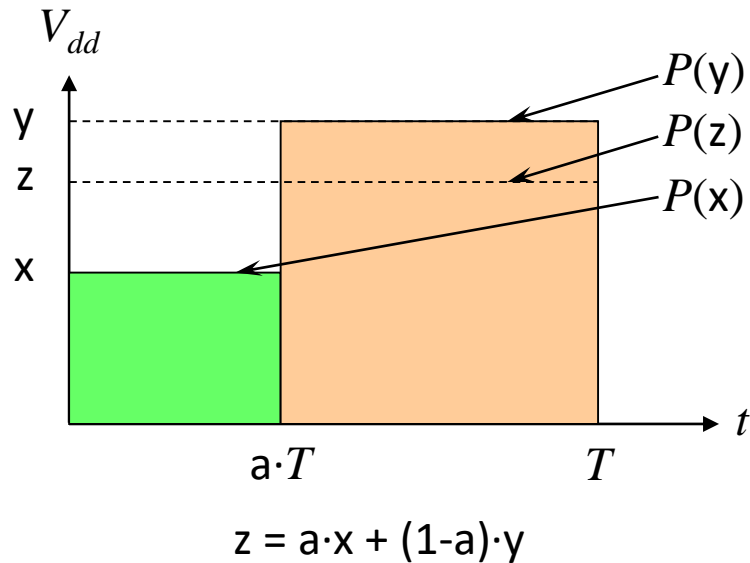
Task that needs to execute 10^9 cycles within 25 seconds.

c) 10^9 cycles @ 40 MHz



$$\begin{aligned} E_c &= 10^9 \times 25 \times 10^{-9} \\ &= 25 \text{ [J]} \end{aligned}$$

DVS: Optimal Strategy



Execute task in fixed time T
with variable voltage $V_{dd}(t)$:

$$\text{gate delay: } \tau \approx \frac{1}{V_{dd}}$$

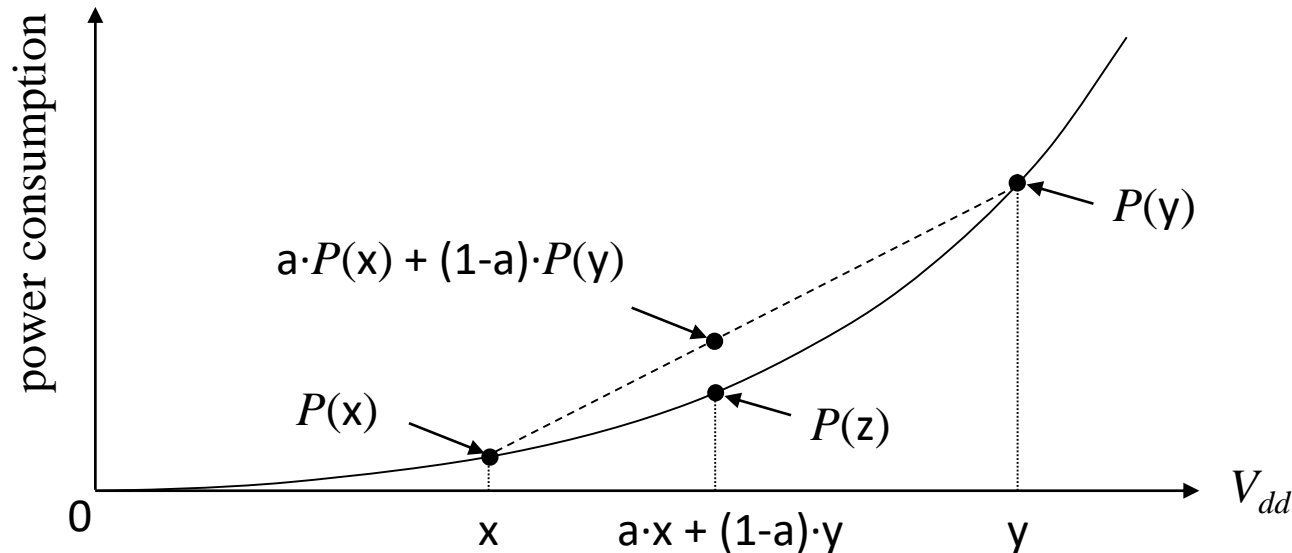
$$\text{execution rate: } f(t) \approx V_{dd}(t)$$

$$\text{invariant: } \int V_{dd}(t) dt = \text{const}$$

- **case A:** execute at voltage x for $a \cdot T$ time units and at voltage y for $(1-a) \cdot T$ time units;
energy consumption $T \cdot (a \cdot P(x) + (1-a) \cdot P(y))$
- **case B:** execute at voltage $z = a \cdot x + (1-a) \cdot y$ for T time units;
energy consumption $T \cdot P(z)$

DVS: Optimal Strategy

Dynamic power is a convex function of V_{dd}



If possible, running at a constant frequency (voltage) minimizes the energy consumption for dynamic voltage scaling:

case A is always worse if the power consumption is a convex function of the supply voltage

DVS: Scheduling on One Processor

- Let us model a set of independent tasks as follows:

We suppose that a task $\tau_i \in \mathbf{T}$

- requires c_i computation time at normalized processor frequency 1
 - arrives at time a_i
 - has (absolute) deadline constraint d_i
- How do we schedule these tasks such that all these tasks can be finished ***no later than their deadlines*** and the energy consumption is ***minimized***?

YDS Algorithm from “A Scheduling Model for Reduce CPU Energy”, Frances Yao, Alan Demers, and Scott Shenker, FOCS, 1995.

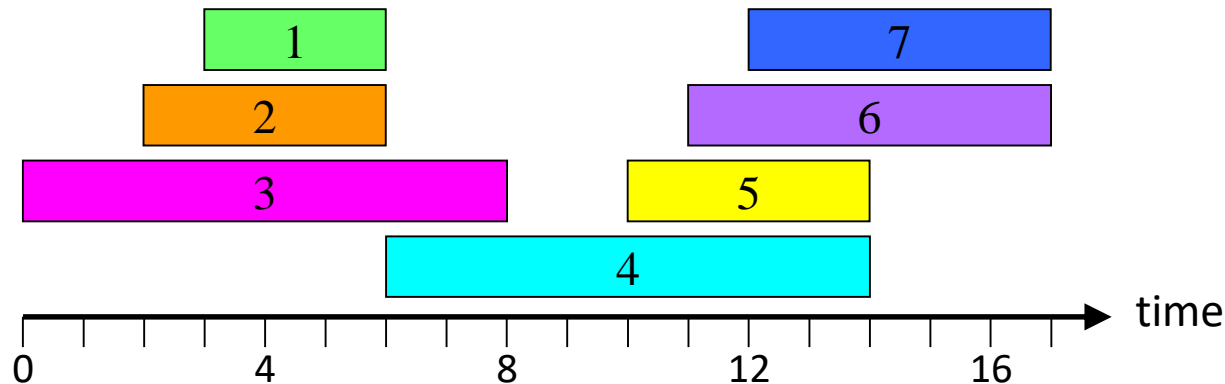
If possible, running at a constant frequency (voltage) minimizes the energy consumption for dynamic voltage scaling.

YDS Algorithm for Offline Scheduling

Define **intensity** $G([b, e])$ in some time interval $[b, e]$:
average accumulated execution time of all tasks that
have arrival and deadline in $[b, e]$

$$\mathbf{T}'([b, e]) = \{\tau_i \in \mathbf{T} : b \leq a_i < d_i \leq e\}$$

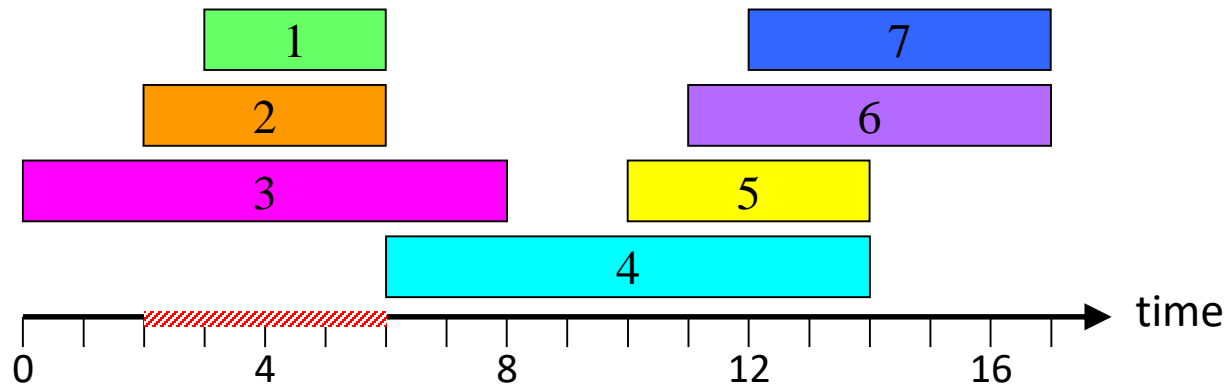
$$G([b, e]) = \sum_{\tau_i \in \mathbf{T}'} c_i / (e - b)$$



a_i, d_i, c_i
3, 6, 5
2, 6, 3
0, 8, 2
6, 14, 6
10, 14, 6
11, 17, 2
12, 17, 2

YDS Algorithm for Offline Scheduling

- Step 1:** Execute jobs in the interval with the highest intensity by using the earliest-deadline first schedule and running at the intensity as the frequency.



$$G([0,6]) = (5+3)/6=8/6, G([0,8]) = (5+3+2)/(8-0) = 10/8,$$

$$G([0,14]) = (5+3+2+6+6)/14=11/7, G([0,17]) = (5+3+2+6+6+2+2)/17=26/17$$

$$G([2, 6]) = (5+3)/(6-2)=2, G([2,14]) = (5+3+6+6) / (14-2) = 5/3,$$

$$G([2,17]) = (5+3+6+6+2+2)/15=26/15$$

$$G([3,6]) = 5/3, G([3,14]) = (5+6+6)/(14-3) = 17/11,$$

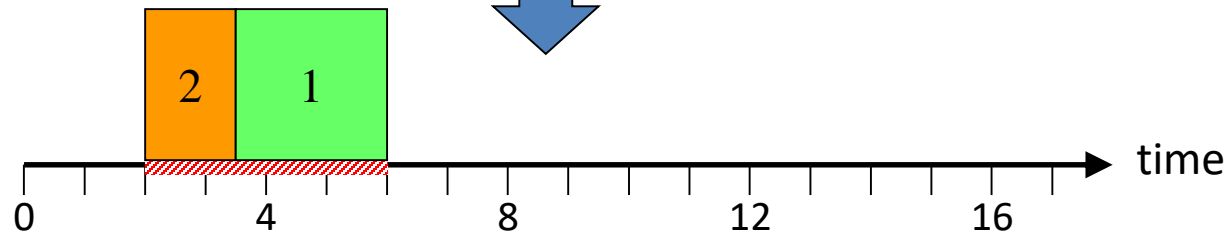
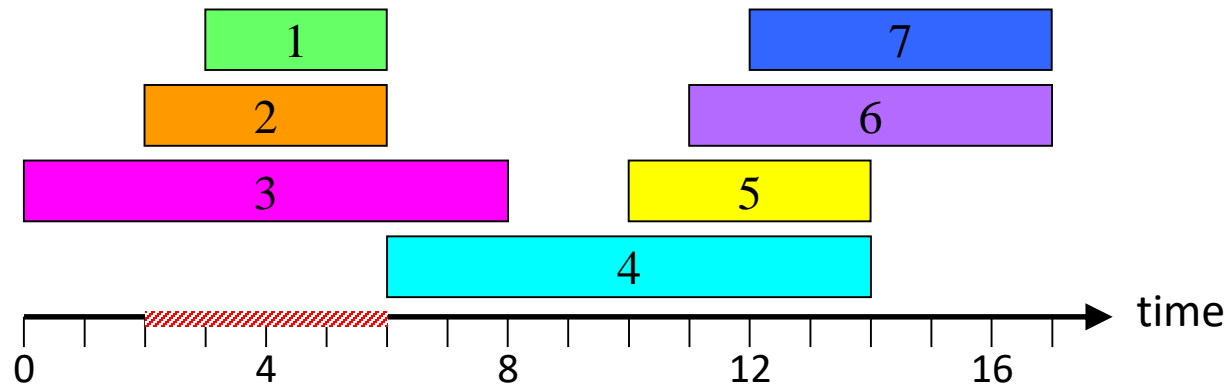
$$G([3,17])=(5+6+6+2+2)/14=21/14$$

$$G([6,14]) = 12/(14-6)=12/8, G([6,17]) = (6+6+2+2)/(17-6)=16/11$$

$$G([10,14]) = 6/4, G([10,17]) = 10/7, G([11,17]) = 4/6, G([12,17]) = 2/5$$

YDS Algorithm for Offline Scheduling

- Step 1:** Execute jobs in the interval with the highest intensity by using the earliest-deadline first schedule and running at the intensity as the frequency.



a_i, d_i, c_i

3, 6, 5

2, 6, 3

0, 8, 2

6, 14, 6

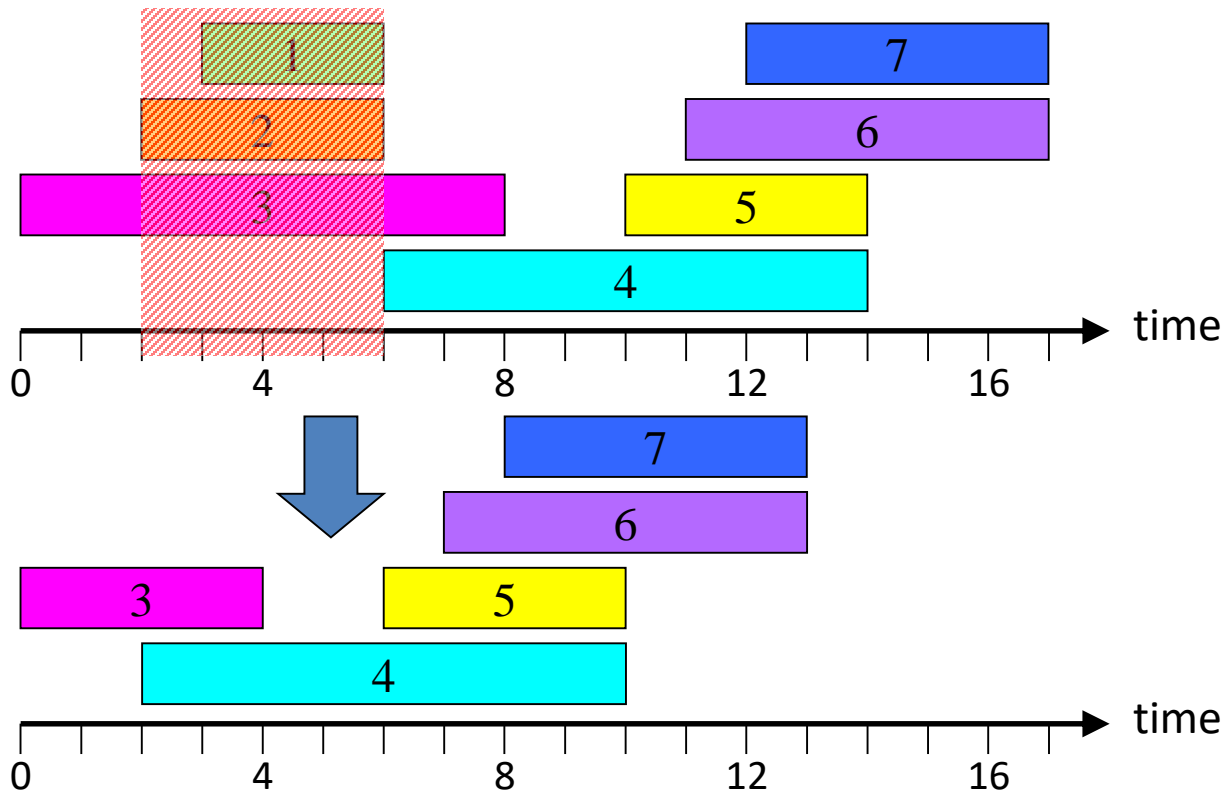
10, 14, 6

11, 17, 2

12, 17, 2

YDS Algorithm for Offline Scheduling

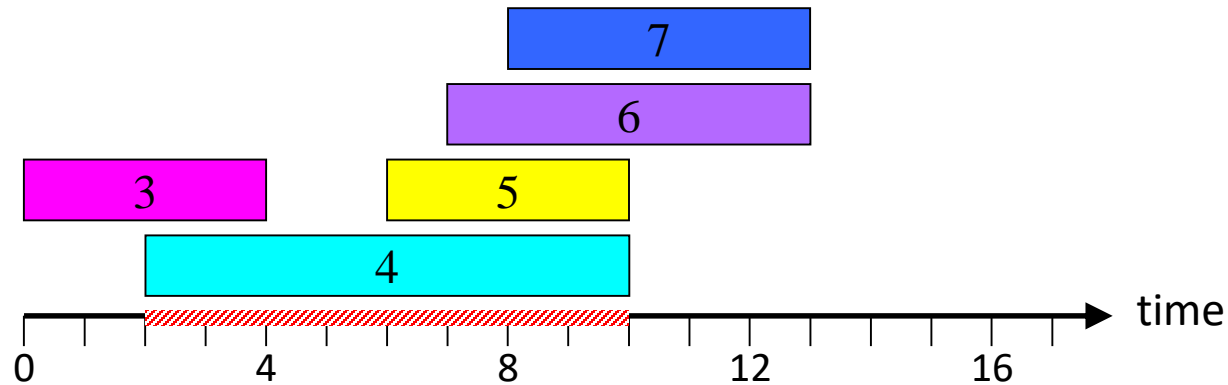
- Step 2:* Adjust the arrival times and deadlines by excluding the possibility to execute at the previous critical intervals.



a_i, d_i, c_i	
3, 6, 5	
2, 6, 3	
0, 8, 2	0, 4, 2
6, 14, 6	2, 10, 6
10, 14, 6	6, 10, 6
11, 17, 2	7, 13, 2
12, 17, 2	8, 13, 2

YDS Algorithm for Offline Scheduling

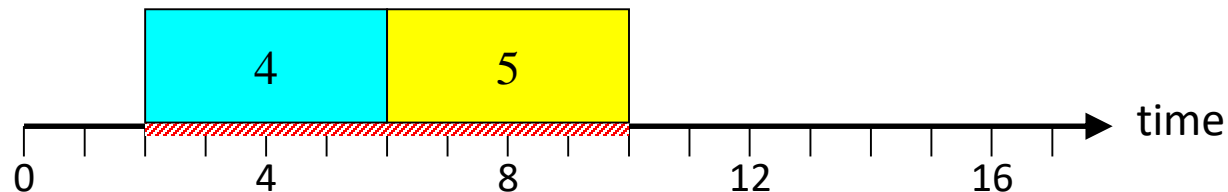
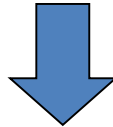
- **Step 3:** Run the algorithm for the revised input again



$$G([0,4])=2/4, G([0,10]) = 14/10, G([0,13])=18/13$$

$$G([2,10])=12/8, G([2,13]) = 16/11, G([6,10])=6/4$$

$$G([7,13])=4/6, G([8,13])=4/5$$



a_i, d_i, c_i

0, 4, 2

2, 10, 6

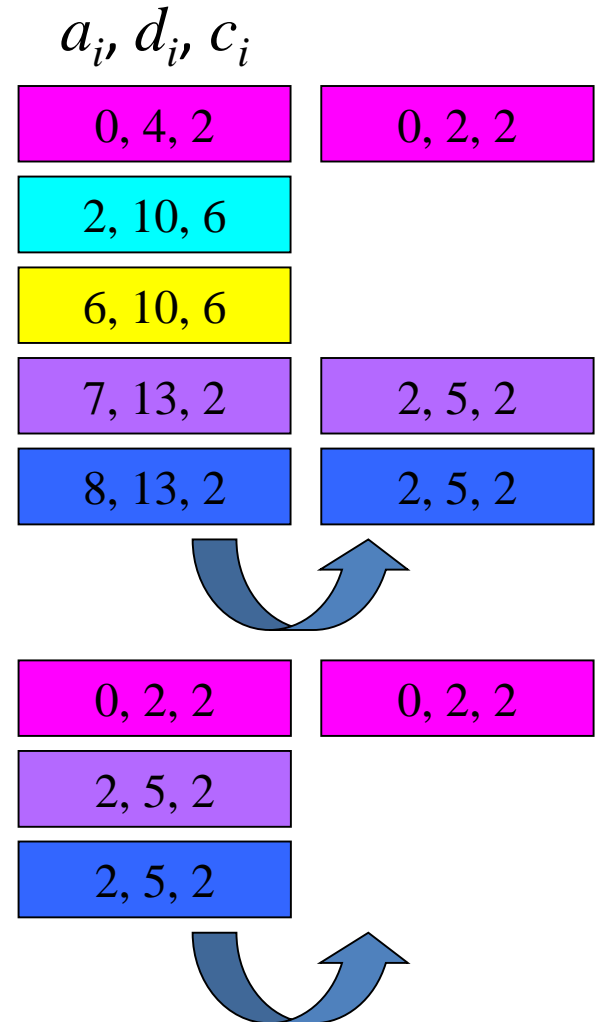
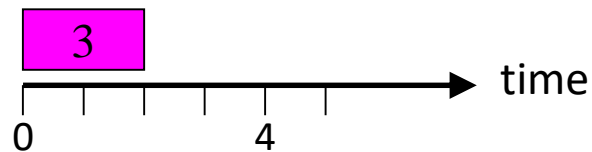
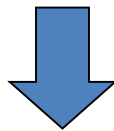
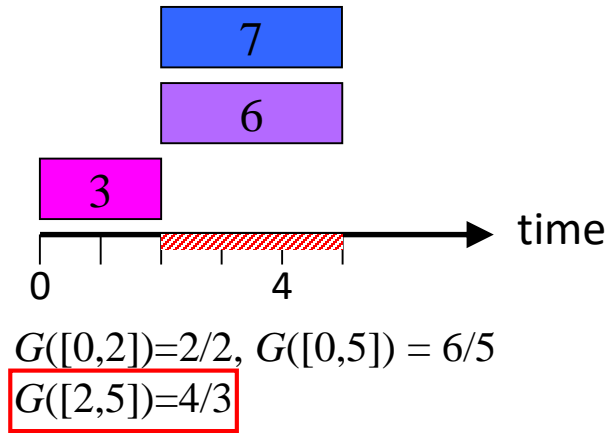
6, 10, 6

7, 13, 2

8, 13, 2

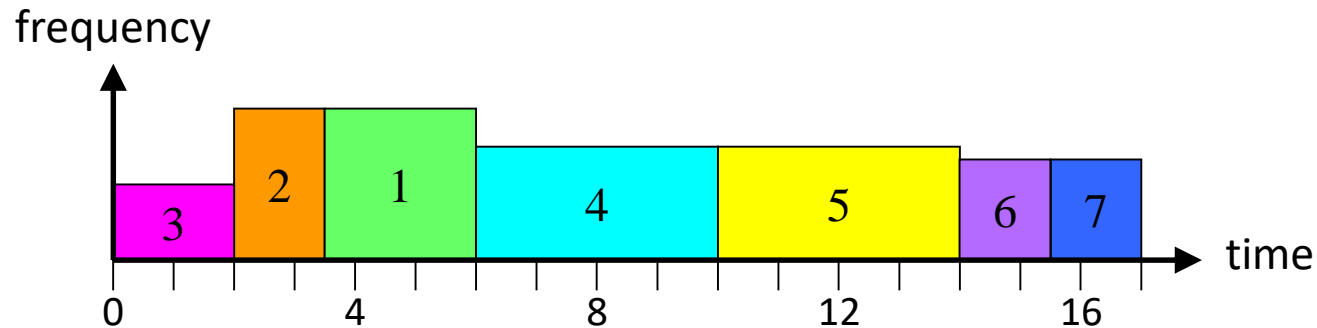
YDS Algorithm for Offline Scheduling

- Step 3:** Run the algorithm for the revised input again



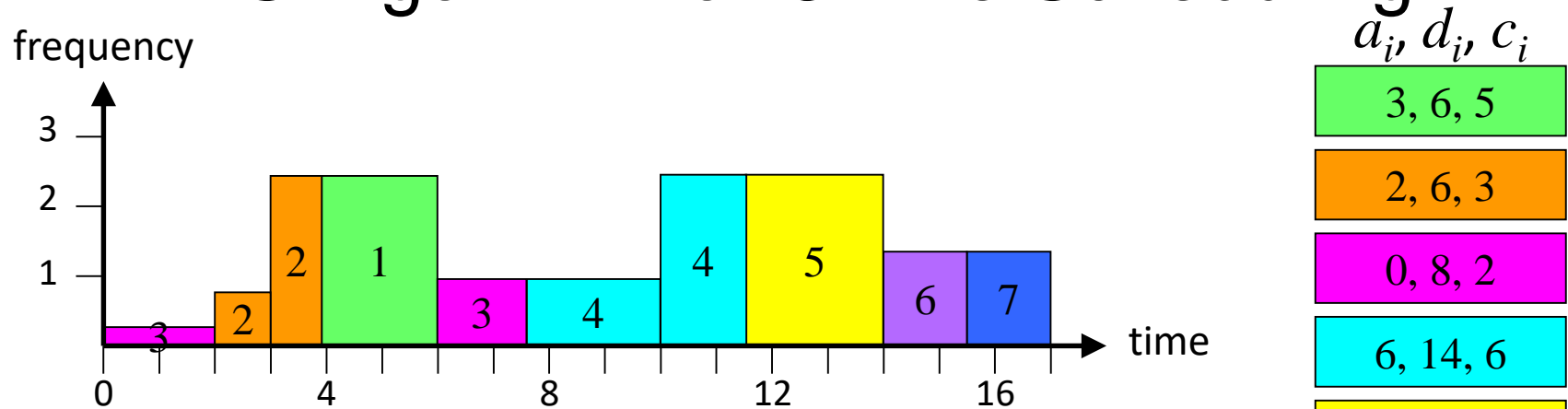
YDS Algorithm for Offline Scheduling

- *Step 4*: Put pieces together



	T_1	T_2	T_3	T_4	T_5	T_6	T_7
frequency	2	2	1	1.5	1.5	$4/3$	$4/3$

YDS Algorithm for Online Scheduling



- Continuously update to the best schedule for all arrived tasks
 - Time 0: task τ_3 is executed at 2/8
 - Time 2: task τ_2 arrives
 $G([2,6]) = 3/4, G([2,8]) = 4.5/6 = 3/4 \rightarrow$ execute τ_2 at 3/4
 - Time 3: task τ_1 arrives
 $G([3,6]) = (5+3-3/4)/3 = 29/12, G([3,8]) < G([3,6]) \rightarrow$ execute τ_2 and τ_1 at 29/12
 - Time 6: task τ_4 arrives
 $G([6,8]) = 1.5/2, G([6,14]) = 15/16 \rightarrow$ execute τ_3 and τ_4 at 15/16
 - Time 10: task τ_5 arrives
 $G([10,14]) = 39/16 \rightarrow$ execute τ_4 and τ_5 at 39/16
 - Time 11 and Time 12
 The arrival of τ_6 and τ_7 does not change the critical interval
 - Time 14:
 $G([14,17]) = 4/3 \rightarrow$ execute τ_6 and τ_7 at 4/3

Remarks on YDS Algorithm

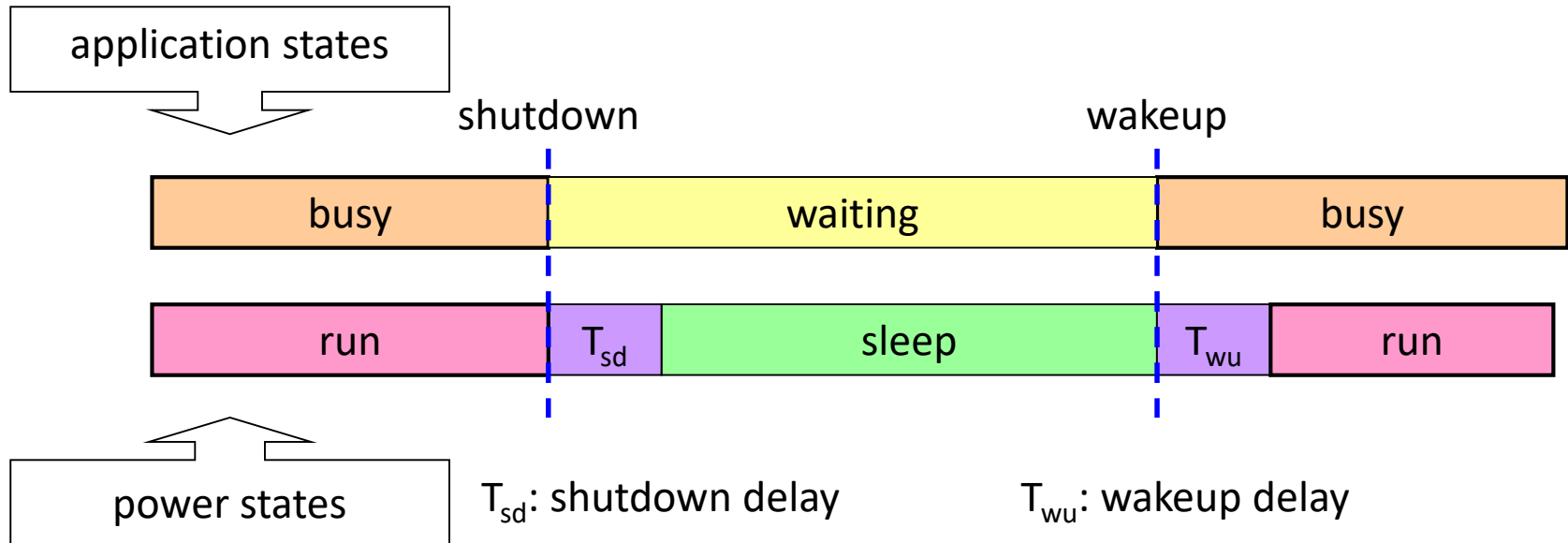
- Offline

- The algorithm guarantees the minimal energy consumption while satisfying the timing constraints
- The time complexity is $O(N^3)$, where N is the number of tasks in \mathbf{T}
 - Finding the critical interval can be done in $O(N^2)$
 - The number of iterations is at most N
- Exercise:
 - For periodic real-time tasks with the same initial phase, running at constant speed with 100% utilization under EDF has minimum energy consumption while satisfying the timing constraints.

- Online

Compared to the optimal offline solution, the on-line schedule uses at most 27 times of the minimal energy consumption.


Reduce Power with DPM




Desired: Shutdown only during long idle times

➔ Tradeoff between savings and overhead

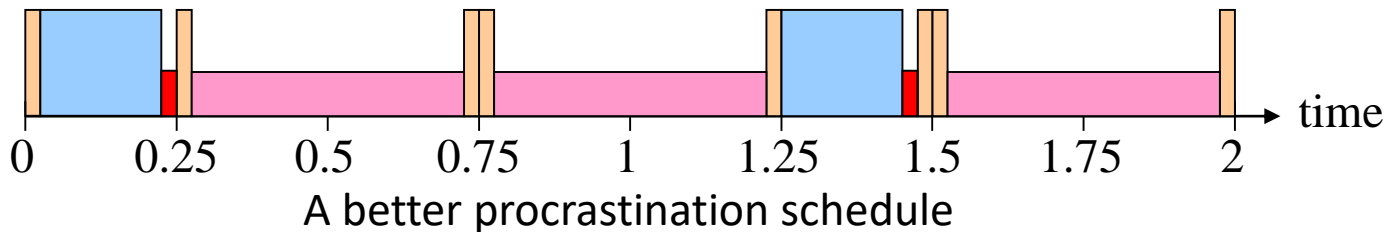
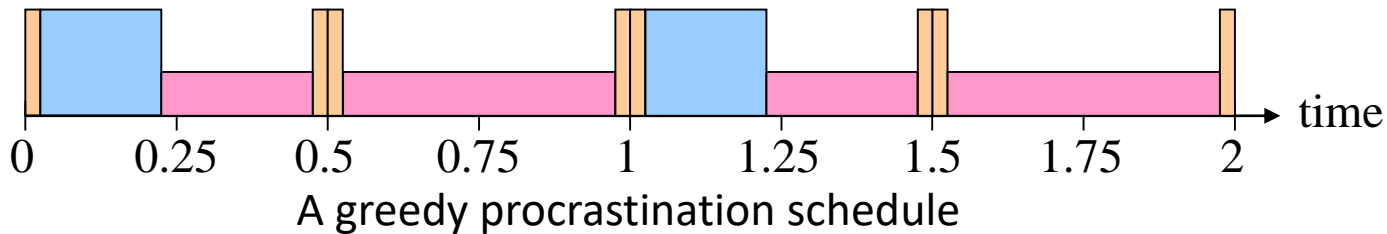
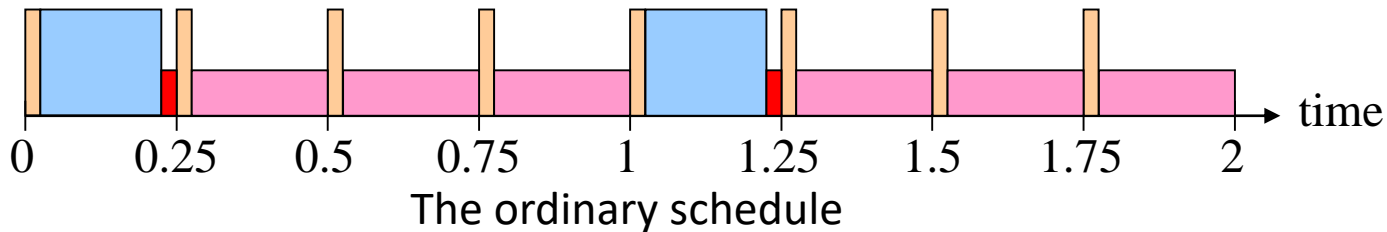
Task Procrastination

 $\tau_1 = (\varepsilon, 0.25)$

 $\tau_2 = (0.25 - 2\varepsilon, 1)$

 Idle

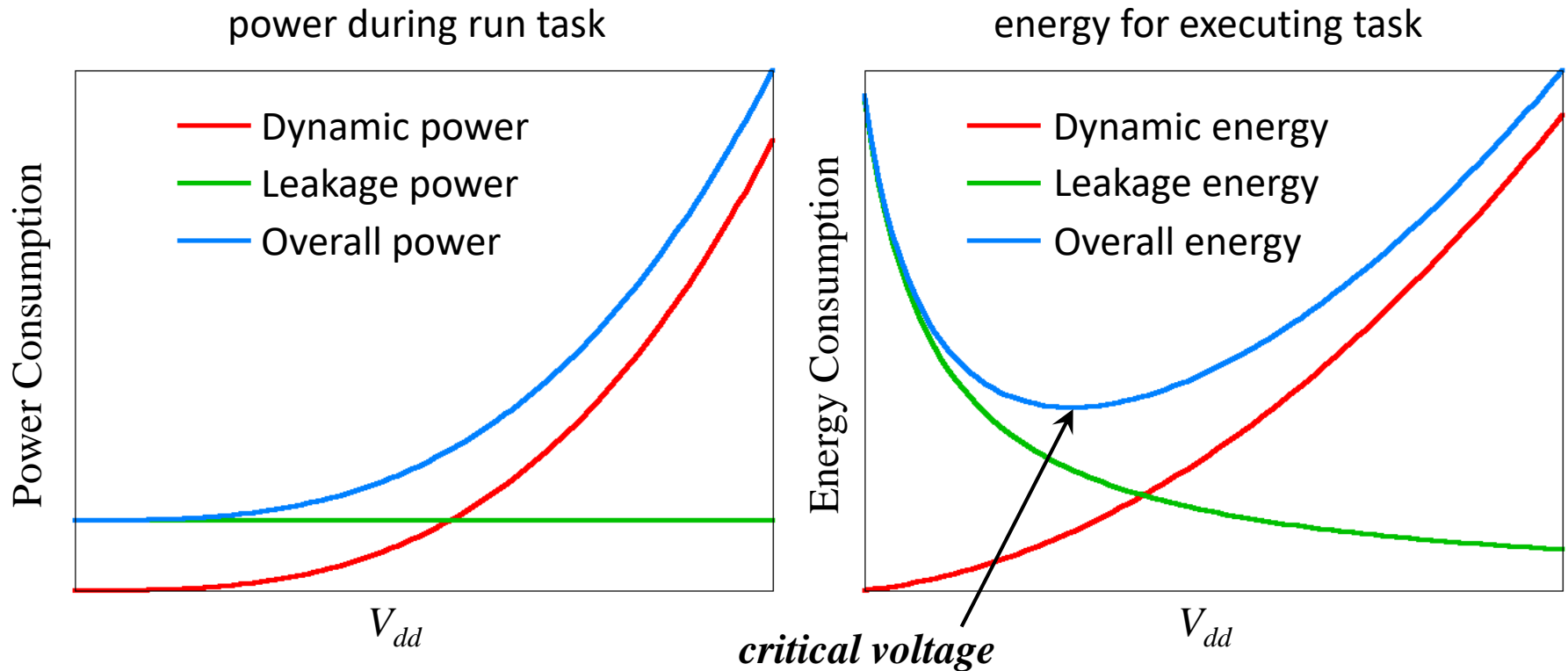
 Sleep



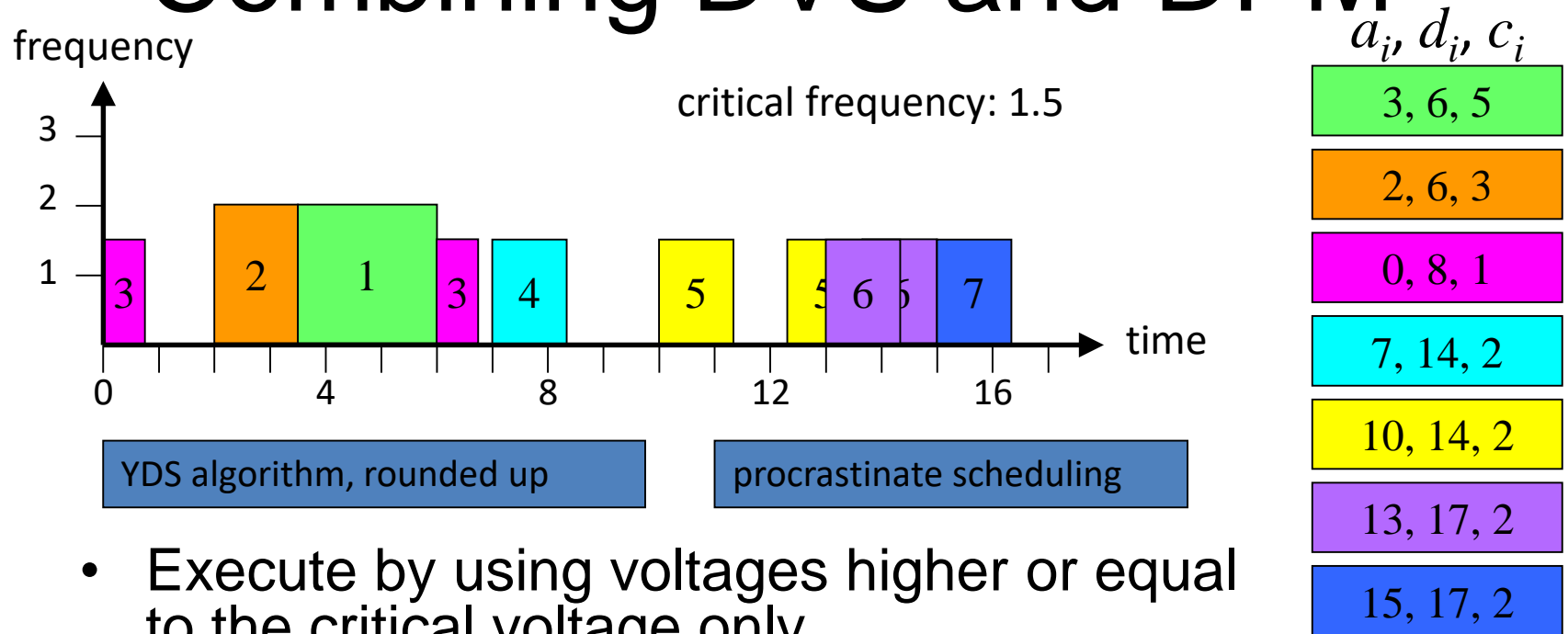
Combining DVS and DPM

DVS critical frequency (voltage):

Running at any frequency/voltage lower than this frequency is not worthwhile for execution.



Combining DVS and DPM



- Execute by using voltages higher or equal to the critical voltage only
 - apply YDS algorithm
 - round up voltages lower than the critical voltage
- Procrastinate the execution of tasks to aggregate enough time for sleeping
 - Try to reduce the number of times to turn on/off
 - Sleep as long as possible

Summary

- Sources of power consumption
 - Dynamic power consumption
 - Leakage power consumption
- Techniques for low-power design
 - Dynamic voltage scaling (DVS)
 - Dynamic power management (DPM)