Problem Set #1 (ver 1.0)

Due: 12:00Noon PST, Nov 16, 2020

URL: https://docs.google.com/document/d/1tVGIc6LBcWSdouvjRj25Wj5vwP3OQaD8E4ahc4bvlAY

Instructions:

1. *This problem set is to be done individually.*
2. *There are no no-penalty late days. However, you may take up to four late days (24 hour intervals) with a penalty of 25% (of the score you get) per full or fractional day. After that score becomes 0.*
3. *You have to submit this homework via Gradescope under PSET #1 (Main PDF). You have already been enrolled there using the official roster. In the PDF that you upload on Gradescope, please start the answer to every problem on a new page and make sure to assign pages to each of problem/subproblem correctly within Gradescope.*
4. *If you had to write software code (Python strongly preferred) to solve any part of the homework, then please also submit your code. Name the files according to the problem they correspond to (e.g. p2a.py for problem 2, part a). Put all the files in a folder hw1 and create hw1.zip, and upload that om Gradescope as well under PSET #1 (Supporting FIles).*

Problem 1: Short Answer Questions [10 points]

1.1. For an MCU will very little data RAM (random-access memory), would you use threads or events? Why?

Events. Because the using threads requires more memory. Since the MCU has little data RAM, it should be better to use events instead of threads although it would need more time to process.

1.2. Many embedded processors do not have memory management units as a result of which all applications and the OS run in a common address space with no memory isolation. Give one pro and one con of this architectural choice.

Pro: It consumes less power, and the program can run faster because the programs can easily access the memory.

Con: Without the memory isolation, the OS cannot protect the programs that should not be accessed by other errant programs. It is easy to cause bug and hard to fix.

1.3. Cache memory is usually considered undesirable in embedded / real-time systems. Why do you think that is the case?

The embedded/real-time systems must have predictable performance so that the hardware can have effective schedule to work. However, the cache memory has unpredictable access times. It leads to a fact the execution times would be different. Therefore, the system might not work well with the cache memory.

1.4. Dynamically allocated memory is usually considered undesirable in embedded / real-time systems. Why do you think that is the case?

Allocation is non-deterministic, and it is not good for real-time systems because they need to have a predictable performance. In addition, allocation can easily access some memories that are used, and cause the systems failure.

1.5. Garbage collection is usually considered undesirable in embedded / real-time systems. Why do you think that is the case?

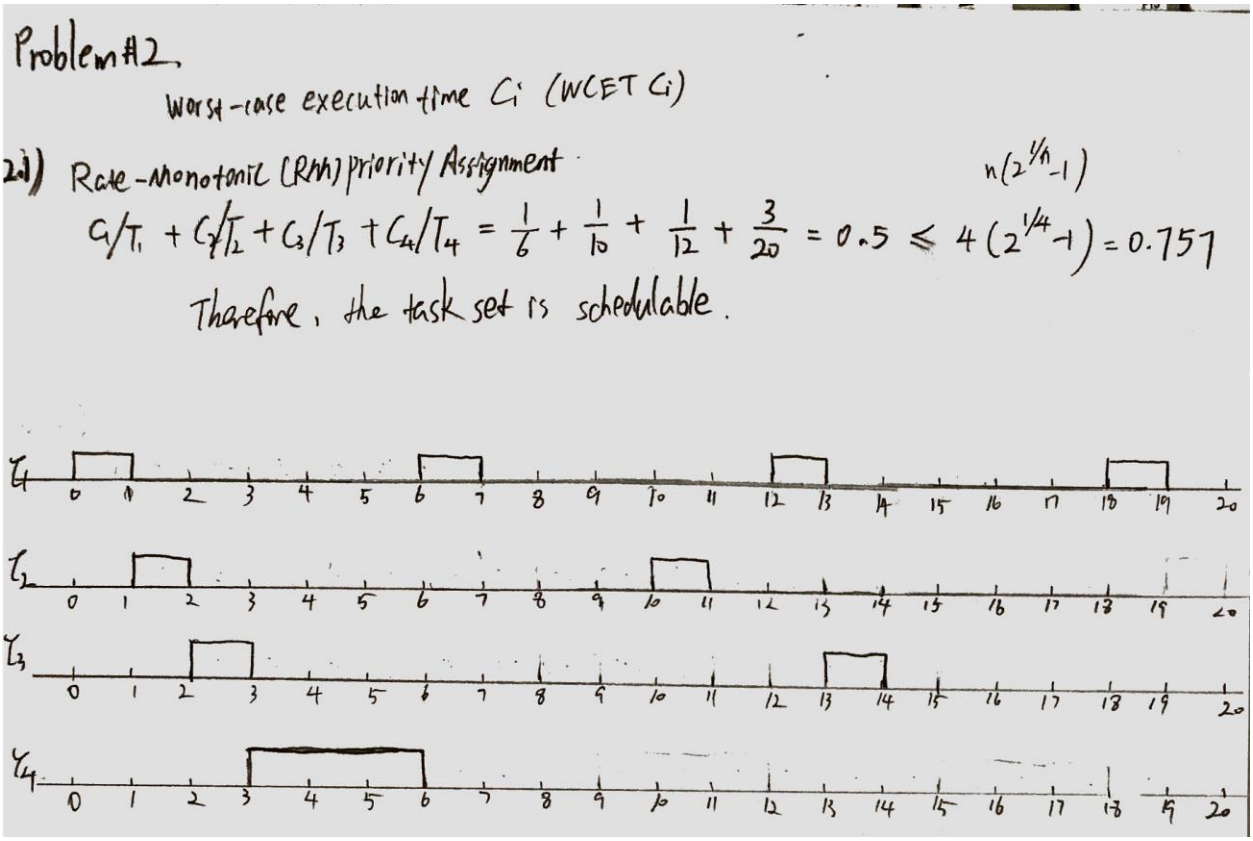
Garbage collection consumes more resources to decide which memory to free, meaning that it needs more power to perform. The Garbage collection can be unpredictable, resulting in pause in the program. This is a very bad idea for the real-time systems

Problem 2: Priority Driven Scheduling [5+5+4+3+3=20 points]

2.1.

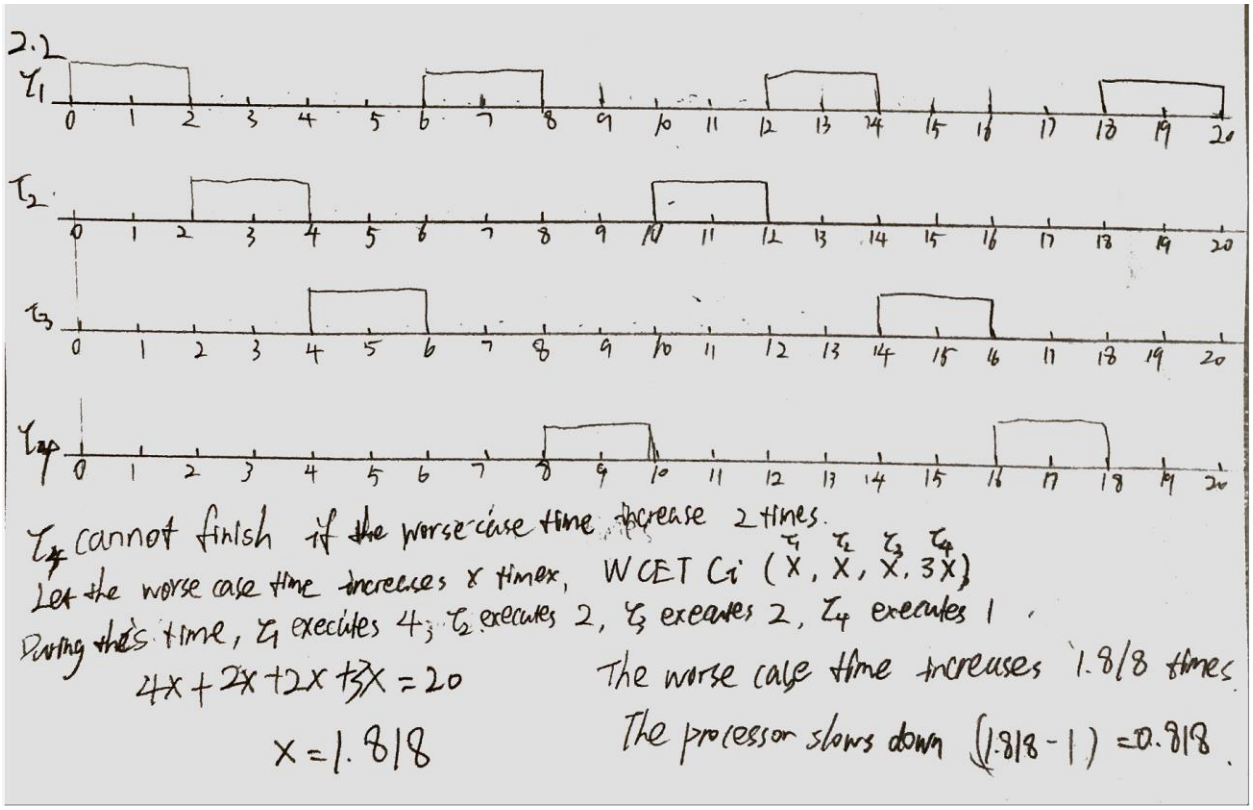
Is the following task set schedulable using a static priority preemptive scheduler? Make sure to justify your answer with appropriate analysis, and provide the priority assignment for the tasks (assume priority=1 is highest, and assign the remaining priorities in sequence).

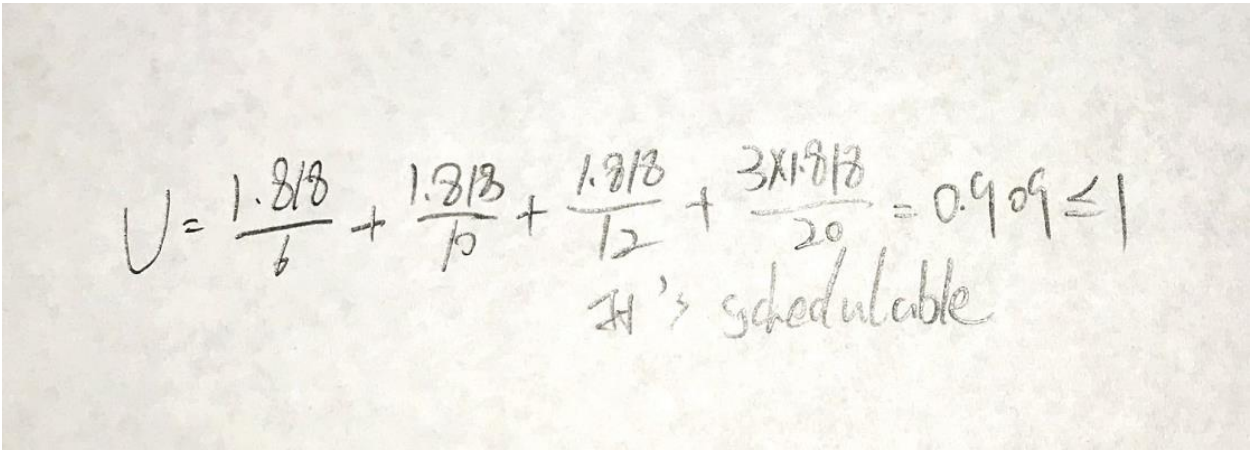
Task i	WCET C_i	Period T_i	Deadline D_i
1	1	6	6
2	1	10	10
3	1	12	12
4	3	20	20



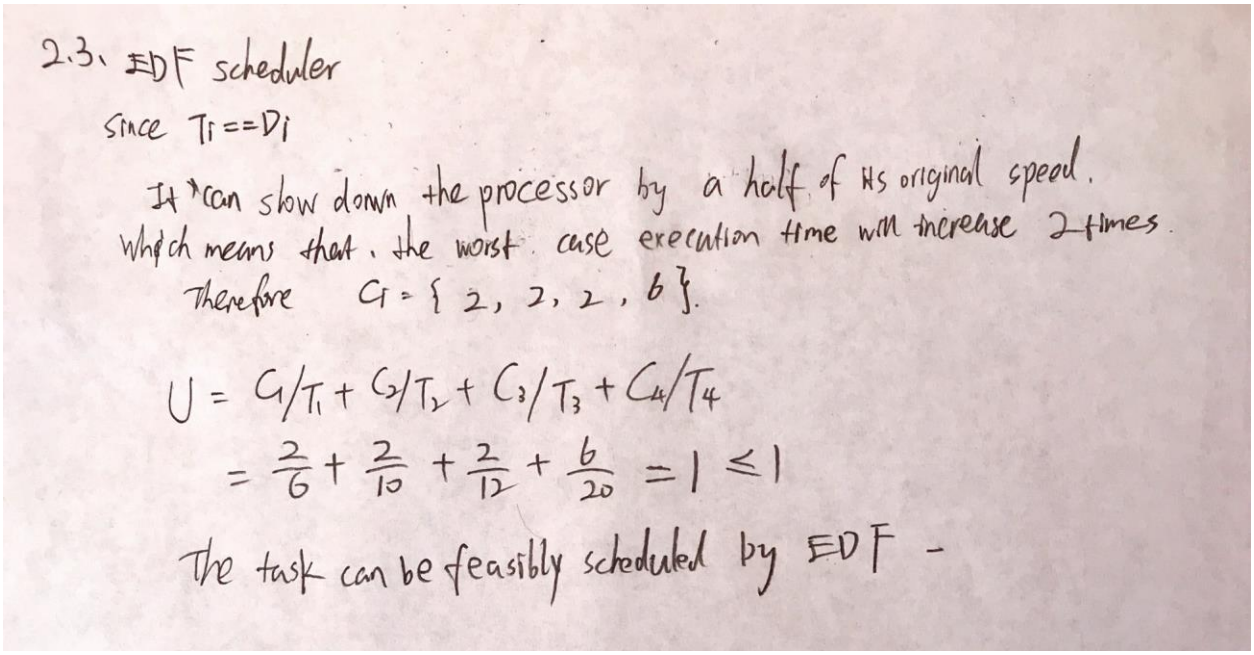
2.2.

If the task set above is schedulable, how much slower a processor can we pick so that the task set still remains schedulable. In other words, at most by what factor can one slow the processor down? On the other hand, if the task set in part (a) is not schedulable, how much faster a processor must we pick so that the task set is schedulable. In other words, at least by what factor must we speed up the processor? Make sure to properly justify your answer (you can write a program if you want, but in that case briefly describe what your program does).





2.3.
Given the same task set but with an OS with a preemptive EDF scheduler, how much would you slow down or speed up the processor so that the system is just schedulable? You must include the reasoning behind your answer to get credit.



2.4.

Consider three periodic tasks τ_1, τ_2, and τ_3 (having decreasing priority), which share three resources A, B, and C, accessed using Priority Inheritance Protocol. Compute the maximum blocking time B_i for each task using the table below which gives the longest duration $D_i(R)$ for a task τ_i on a resource R. Assume that there are no nested critical sections (i.e. a task releases a resource before locking another one). Note that you will need to think about how to generalize the formula given in the lecture.

Task	A	B	C
τ_1	2	0	2
τ_2	2	3	0
τ_3	3	2	5

From lecture 5-35,36

$$B_i = \sum_{k=1}^{K} usage(k,i)CS(k)$$

- Where usage is a 0/1 function:

usage(k, i) = 1
if resource k is used by at least one process with a priority < i,
and at least one process with a priority greater or equal to i.
Otherwise it gives the result 0.

- CS(k) is the computational cost of executing the k-th critical section.

Task1,

usage(A,1) = 1, lower priority τ_2 and τ_3 use resource A, τ_1 also uses resource A

usage(B,1) = 0, τ_1 doesn't use resource B

usage(C,1) = 1, τ_1 uses resource C and lower priority τ_3 uses resource C

CS(A) = 2+3 = 5

CS(C) = 0+5 = 5

B1 = A2+A3+C3 = 2+3+5 = 10

Task2,

usage(A,2) = 1, lower priority τ_3 use resource A, τ_1 and τ_2 also uses resource A

usage(B,2) = 1, lower priority τ_3 uses resource B, equal priority τ_2 uses resource B

usage(C,2) = 1, greater priority τ_1 uses resource C and lower priority τ_3 uses resource C

CS(A) = 3

CS(B) = 2

CS(C) = 5

B2 = A3+B3+C3 = 3+2+5 = 10

Task3,

usage(k, i) = 0 since there no low priority task use the resources

B3 = 0;

2.5.

Repeat above, but now for the case where we use Priority Ceiling Protocol instead of Priority Inheritance Protocol.

$$B_i = \max_{k=1}^{K} usage(k,i)CS(k)$$

As 2.4 we calculated, the critical sections which can block task1 is A2, A3, C3. Maximum blocking time is B1 = 5.

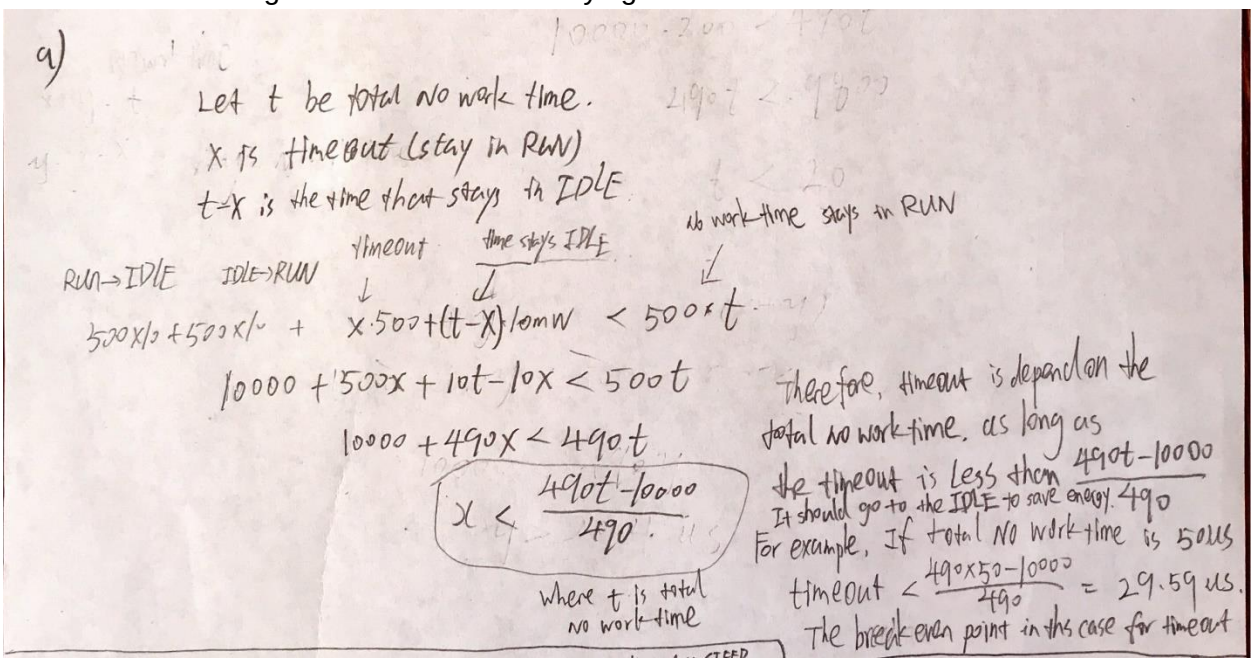
the critical sections which can block task2 is A3, B3, C3. Maximum blocking time is B2 = 5.

B3 = 0 since this is the lowest priority, which it cannot be blocked.

Problem 3: Managing Processor Sleep [10 points]

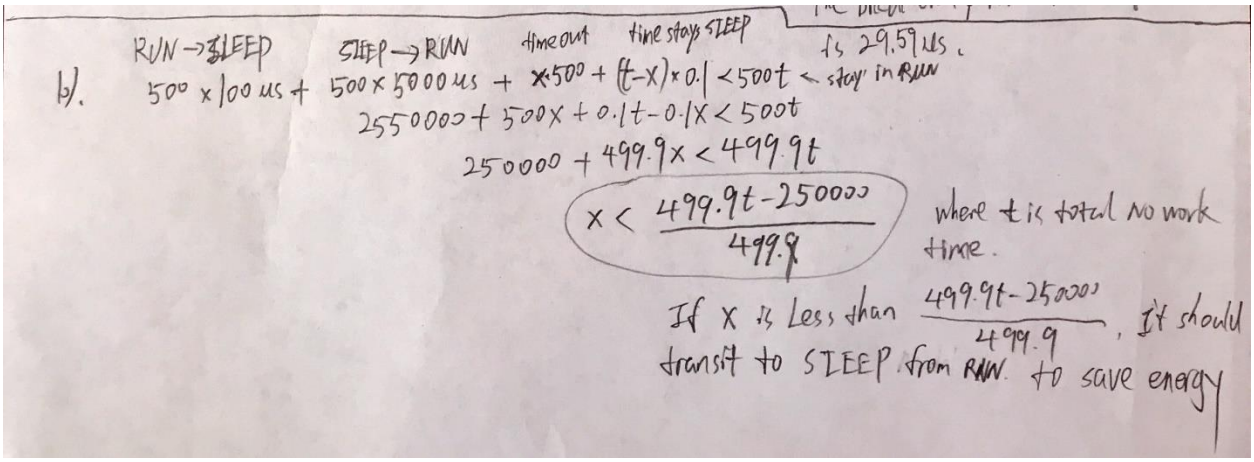
Consider a processor whose power state diagram is as shown below. Assume that power consumption during transitions from one state to another corresponds to the power of the higher power state. So, for example, when going from RUN to IDLE, the power consumption is 500 mW during the 10 microseconds of transition to/from RUN state is the same as in the RUN state, and that the power consumption during the transition from IDLE to SLEEP is the same as that in the IDLE state.

a. Consider a power management strategy whereby the OS transitions the processor to IDLE state from RUN state after a timeout if the processor has no work to do, and transitions back to RUN when there is work to do. What timeout will you pick, and why? Please show any necessary computation. Hint: think in terms of the break-even point when shutting down is better than staying on.



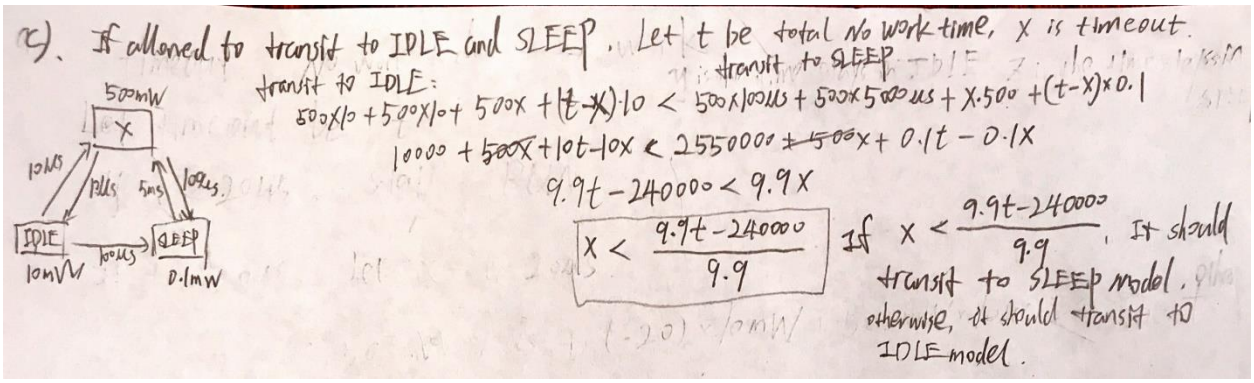
On the other hand, if the timeout is not depending on the total no work time. The break-even point is 20us. If the no work time is longer than 20us, transition to the IDLE state will save more energy. Therefore, the timeout can be any value can be any value if the no work time is longer than 20us. In addition, if the no work time is less than 20us, it should not transit to the IDLE model because it would consume more energy compared to the energy consumed staying in the RUN state.

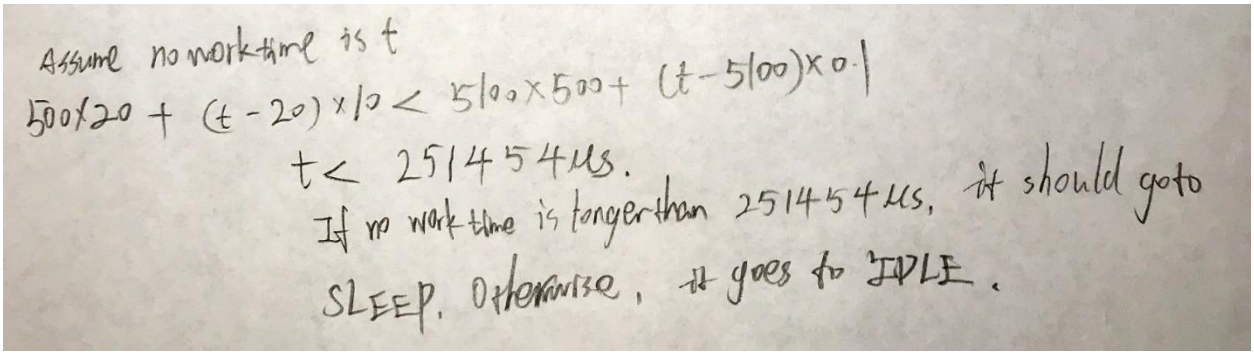
b. How would your choice for timeout change if instead of IDLE the processor were to transition to SLEEP?

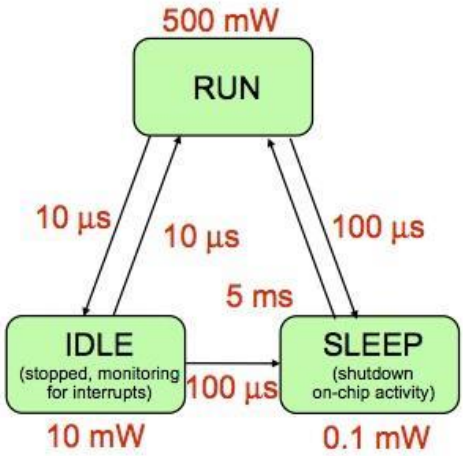


On the other hand, if the timeout is not depending on the total no work time. The break-even point is 5000us + 100us = 5100us. If the no work time is longer than 5100us, transition to the SLEEP state will save more energy. Therefore, the timeout can be any value can be any value if the no work time is longer than 5100us. In addition, if the no work time is less than 5100us, it should not transit to the SLEEP model because it would consume more energy compared to the energy consumed staying in the RUN state.

c. What would your timeout policy be if you were allowed to transition to IDLE as well as SLEEP?







Problem 4: Maximizing Computation Before Battery Dies [10 points]

Note: While the question text is long, the problem is quite a straightforward optimization problem other than some simple algebraic manipulation and high school physics level relationships between energy, power, current, and voltage.

In the lecture, we discussed several objectives relating to energy and power that one may have when designing a system: low power (to keep current and heat under limits for a given rate of work), low energy (to make the battery last longer or to save cost for a given amount of work), and energy proportionality (making power consumed proportional to performance).

To attain these objectives, we discussed some broad strategies, and specifically in the context of CPUs we discussed: dynamic power management or DPM (shutting down the CPU when not needed), dynamic frequency scaling or DFS (changing the clock frequency while keeping voltage fixed), and dynamic voltage-frequency scaling or DVFS (changing clock frequency and simultaneously changing the voltage to the minimum possible value needed to sustain the chosen frequency).

During the class, I made the comment that DFS is not particularly good if our objective is to lower energy consumption in order to make the battery last longer. The reasoning was that if we lower frequency then while power goes down proportionately (in the ideal case), the execution time goes up proportionately and so energy consumed remains constant.

However, this simple analysis ignores two crucial issues.

First, the power consumption of a system is not strictly proportional to processor frequency (speed) due to various sources of static power consumption (e.g. components such as memory and I/O, as well as device leakage within the processor). A simple first-order model of power consumption of a system as a function of frequency while keeping the voltage fixed is $P(f) = \alpha \cdot f + \beta$ where α and β are non-negative constants (dependent on voltage, temperature, etc.) corresponding to dynamic and static components of the power consumption.

Second, the energy capacity of batteries depends on the rate at which we draw energy from the battery, i.e. specifically on power, or equivalently on current assuming the battery has a constant voltage (since $P = VI$). In particular, if the discharge current is high then the battery provides less energy before it dies (the reasons have to do with the mobility of ions and the resulting concentration near the electrodes). This is called the rate-capacity effect. An approximate model for the relationship between the energy capacity of a battery and the current drawn from it is given the ***Peukert's Law***[1], which states that $C_p = I^k t$ where C_p is the capacity at 1 Ampere discharge rate expressed in Ampere-hr (by multiplying this with battery's fixed voltage, one can get energy capacity in the more traditional Watt-hr), I is the discharge current in Ampere, t is the time in hours for which the battery discharges (i.e. its life), and k is the Peukert's constant. Here, C_p and

[1] https://en.wikipedia.org/wiki/Peukert's_law

k are parameters of a battery, while I and t are a function of the system design. The battery parameters C_p and k change as the battery ages and depend on ambient conditions, but can be treated as constant for a short time horizon under static ambient conditions. k is 1 for an ideal battery (i.e. one where battery capacity does not depend on load current as then $C_p = I \quad t$), and typically ranges from 1.1 to 1.3 for real batteries. The Peukert exponent increases with battery age and battery capacity is also impacted by temperature, but we ignore these effects here. Typically for a battery, its capacity in Ampere-Hours at 1 Ampere (i.e. C_p) is not given, and instead, the discharge time t_{ref} at some other reference current I_{ref} (and thus the corresponding battery capacity $C_{ref} = I_{ref} t_{ref}$) is given. Then with some simple algebraic manipulation, one can compute the discharge time for the current at which the system is actually operating.

Imagine you have a battery-operated system with the power model $P(f) = \alpha \cdot f + \beta$ and $f \leq f_{max}$running on a battery with Peukert's constant k and capacity C_{ref} at reference current I_{ref}.

What frequency would you run the processor at so as to maximize the work done by the processor before the battery dies? If a range of frequency is optimum, provide the whole range. Make sure that for the ideal case (energy-proportional system and battery with no rate-capacity effect) your solution makes sense. Please make sure to provide a complete analytic derivation of your answer as otherwise, you will get zero credit.

Problem 4:

Peukert's Law

$C_p = I^k t$

$t = H\left(\frac{C}{IN}\right)^k$ discharge time t_{ref} (in hours) 1+.

$t = t_{ref}\left(\frac{C_{ref}}{I t_{ref}}\right)^k$

$P(f) = \alpha f + \beta$ Power (frequency) $\quad P = VI \Rightarrow VI = \alpha f + \beta$

$I = \frac{\alpha f + \beta}{V}$

$N = t \cdot f = t_{ref}\left(\frac{C_{ref}}{\frac{\alpha f+\beta}{V} \cdot t_{ref}}\right)^k \cdot f$

total cycles (tasks)

$N = t_{ref} \cdot f \left(\frac{C_{ref} \cdot V}{(\alpha f+\beta) \cdot t_{ref}}\right)^k$

take derivative of f: $\frac{dN}{df} = t_{ref}\left(\frac{C_{ref} \cdot V}{t_{ref}(\alpha f+\beta)}\right)^k - \frac{\alpha C_{ref} \cdot f \cdot k V \left(\frac{C_{ref} \cdot V}{t_{ref}(\alpha f+\beta)}\right)^{k-1}}{(\alpha f+b)^2}$

Let $\frac{dN}{df} = 0$ Find f

Derivative:

$$\frac{\partial}{\partial f}\left(t\, f\left(\frac{C\,V}{(a\,f+b)\,t}\right)^k\right) = t\left(\frac{C\,V}{t\,(a\,f+b)}\right)^k - \frac{a\,C\,f\,k\,V\left(\frac{C\,V}{t\,(a\,f+b)}\right)^{k-1}}{(a\,f+b)^2}$$

Alternate forms:

$$\frac{t\,(b-a\,f\,(k-1))\left(\frac{C\,V}{a\,f t+b\,t}\right)^k}{a\,f+b}$$

$$-\frac{t\,(a\,f\,k-a\,f-b)\left(\frac{C\,V}{t\,(a\,f+b)}\right)^k}{a\,f+b}$$

$$-\frac{C\,V\,(a\,f\,(k-1)-b)\left(\frac{C\,V}{t\,(a\,f+b)}\right)^{k-1}}{(a\,f+b)^2}$$

Alternate forms assuming a, b, C, f, k, t, and V are positive:

$$t^{1-k}\left(\left(\frac{C\,V}{a\,f+b}\right)^k - a\,f\,k\,(C\,V)^k\,(a\,f+b)^{-k-1}\right)$$

$$t^{1-k}\left(\frac{C\,V}{a\,f+b}\right)^k - a\,f\,k\,t^{1-k}\,(C\,V)^k\,(a\,f+b)^{-k-1}$$

Roots:

$$t\,(a\,f+b)\neq 0,\quad C=0,\quad \mathrm{Re}(k)>0$$

$$t\,(a\,f+b)\neq 0,\quad \mathrm{Re}(k)>0,\quad V=0$$

$$a\,f\neq 0,\quad t\,(a\,f+b)\neq 0,\quad C\,V\neq 0,\quad k=\frac{a\,f+b}{a\,f}$$

The root will be $k = \dfrac{a\,f+b}{a\,f}$

Therefore, $f = \frac{\beta}{(k-1)*\alpha}$