Lab of EMIot
Welcome!

Objective and organization

- Logistics: In-class lab
  - 1 laptop per person
  - May be useful to bring portable multiple sockets

- Necessary software:
  - C
  - MATLAB
    - You can get a free student licence https://www.areait.polito.it/supporto/risultato_serv.asp?serv=matlab&dettaglio=S&id_progetto_servizio=331

LAB schedules

- 20% of the final score
  - 9 points maximum

- Assignments will be evaluated
  - Each student has to deliver his/her own assignment
    - No groups allowed!
  - 1 report per lab
  - Any extension to the minimum assignment **may** lead to an increase in the evaluation
    - Make sure you meet all requirements
    - Do not go out of topic

LAB delivery

- Lab deadline is 23:59 of the day before the 2nd exam
  - **No exception**
  - Late delivery implies <u>zero score</u> for labs

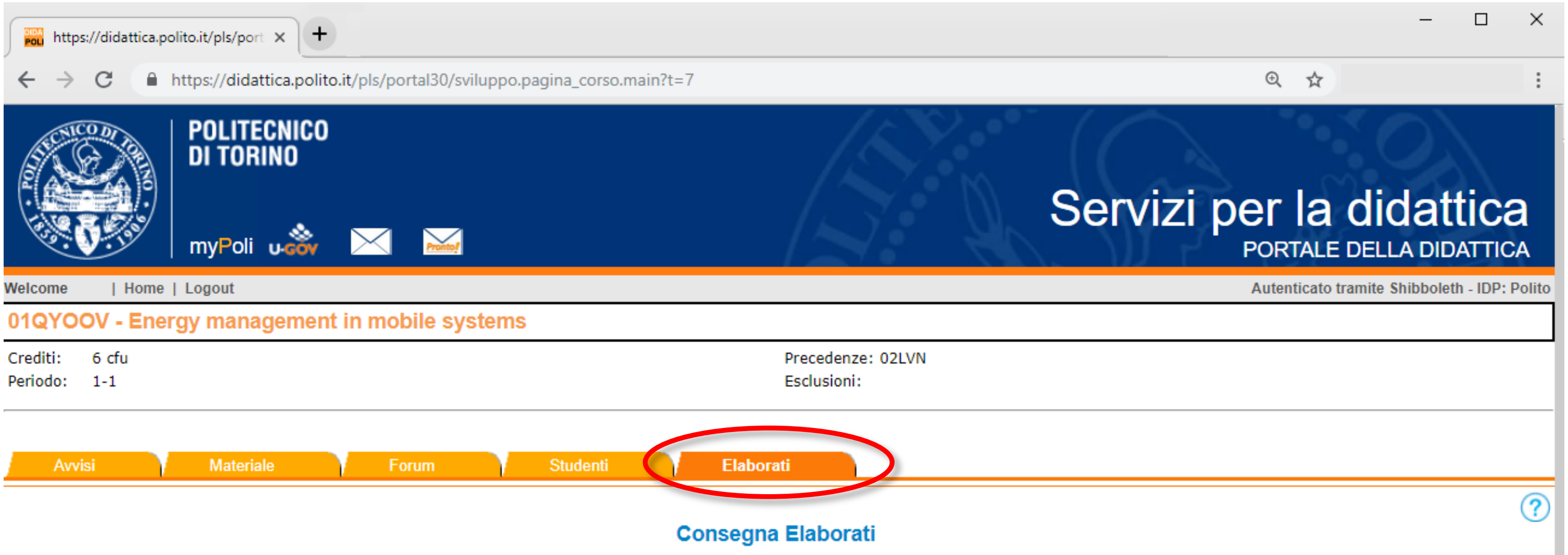
- Format: **one archive**
  - File name = report.zip
  - One subfolder per lab (Lab1/ Lab2/ Lab3/)

LAB delivery

- What to deliver for each lab:
  - Source code
    - All code you modified and/or consider necessary
  - Report
    - 5-10 pages per lab, depending on the depth of experiments
    - PDF format
    - **Analysis of results**
      - This is what gives you points!
      - Implementing the code is not enough!

LAB delivery

- How to deliver:
  - Through the didattica web site
    - «Elaborati» tag



Lab 1 – Day 1
Dynamic Power Management

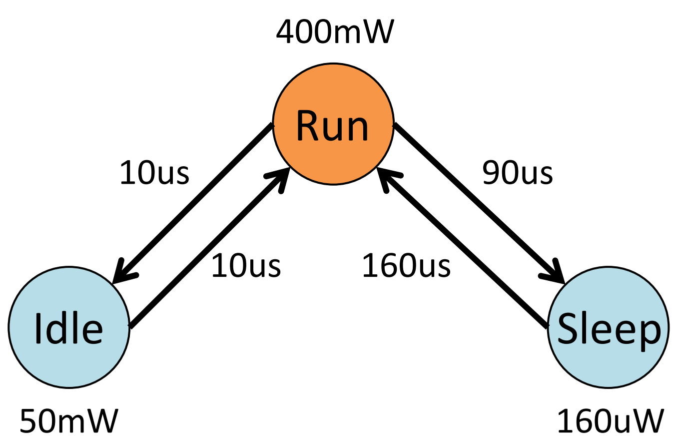
Objective and organization

- Understanding of the basics of DPM
  - Use and modify a simple power state machine simulator in C
  - Evaluation of power management policies

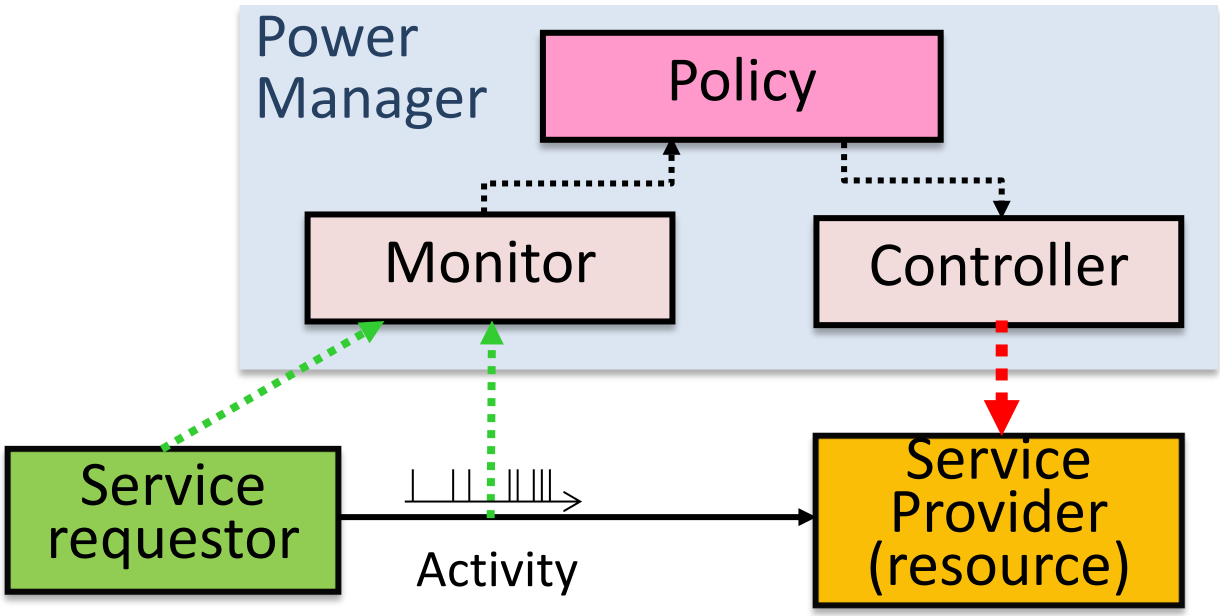
- 1 report and 3 days

Recall

- Dynamic Power Management
  - Reduce power by turning devices to low power when peak performance is not needed
  - Devices abstracted as power state machines
    - Several internal states corresponding to modes of operation
      - Different power and service levels



Recall



- Power manager (PM)
  - Monitors requestor's activity and sets state of provider according to some **policy**
    - E.g., shuts down component after some inactivity time

Recall

- Given a PSM and a workload, determine the optimal allocation of power states over time that minimizes power under performance constraints

- Non-idealities make the problem non-trivial!
  - Transitions costs (time & power) are not zero
    - Transitions must be amortized!
  - Length of idle periods often unknown

DPM simulator

- Goal of the lab:
  - Evaluate on a case study how energy saving changes as a function of
    - The distribution of idle times
    - The PSM parameters
    - The applied DPM policies
    - …

DPM simulator

- C program with the following basic operations
  - Read a power model → a PSM
  - Read a workload profile
  - Simulate two power management policies
    - Timeout
    - History-based prediction

DPM simulator

- **`dpm_simulator [-help] [-t|-h] [-psm <psm file>][-wl <wl file>]`**

  `-help:` prints command line instructions and returns

  `-t <Timeout>:` use a *timeout* policy with <Timeout>

  `-h <Value1> …<Value5> <Threshold1> <Threshodl2>:` use a *history-based predictive policy*
  - Length of history window = 5
  - <Value1-5> are the regression coefficients
  - <Threshold1-2> are the minimum time thresholds

  `-psm <psm file>:` the name of the file describing the *power state machine* (PSM) of the resource

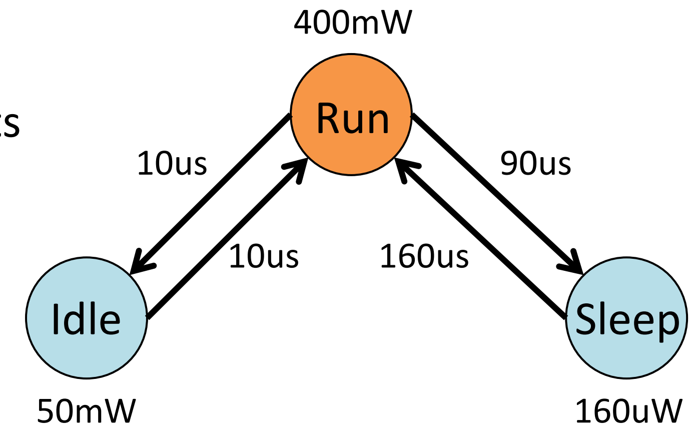
  `-wl <wl file>:` the *workload* file name

Format of the PSM

400	50	0.16
0/0	10/10	10/90
10/10	0/0	-1/-1
30/160	-1/-1	0/0

States power

Transitions costs
(energy/time)



Transition does not exist:

- 0/0: Self-loops (i.e., state does not transition to itself)
- -1/-1: There is no transition between states

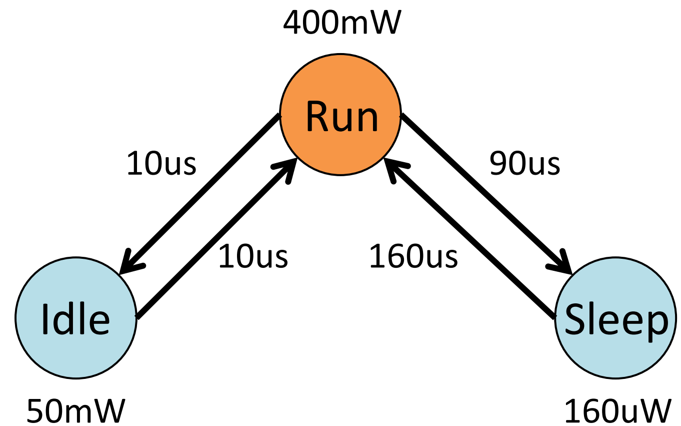
Default time, power, and energy units are: us, mW and uJ

Format of the PSM

400	50	0.16
0/0	10/10	10/90
10/10	0/0	-1/-1
30/160	-1/-1	0/0

State power

Transitions costs (energy/time)



TO STATE…

FROM STATE…

	RUN	IDLE	SLEEP
RUN			
IDLE			
SLEEP	30 / 160		

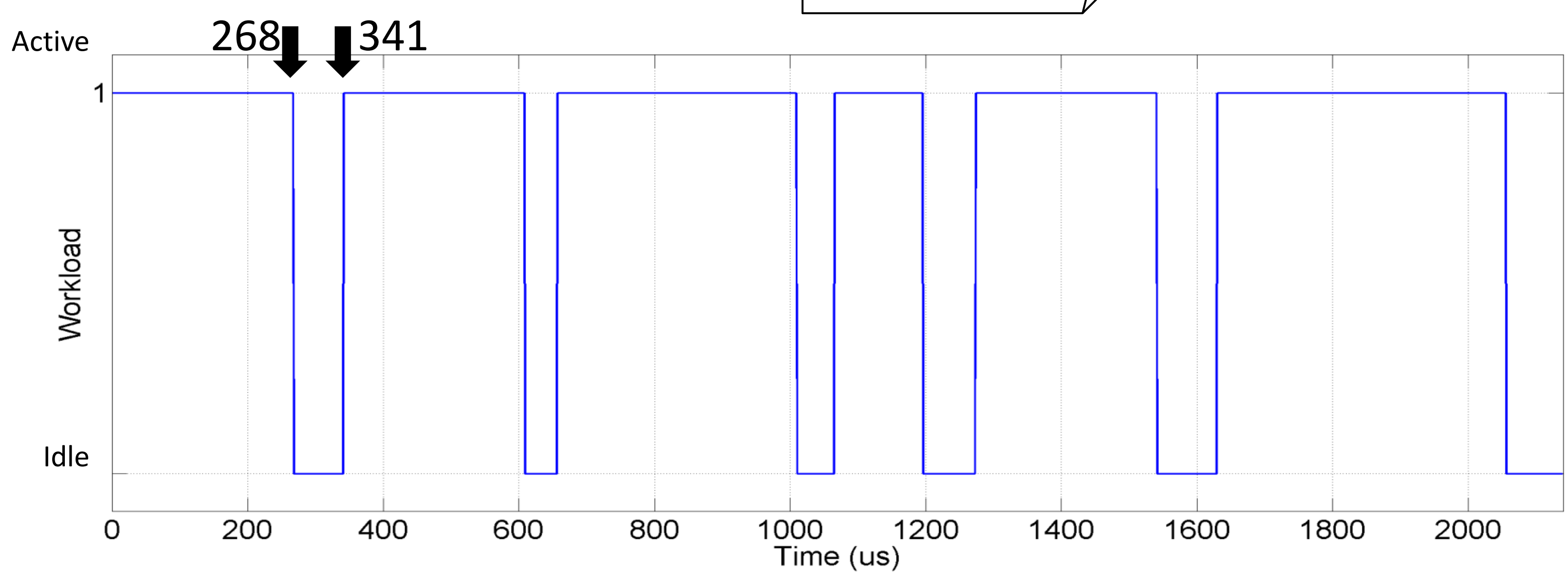
ENERGY / TIME

30 µJ / 160µs

Workload format

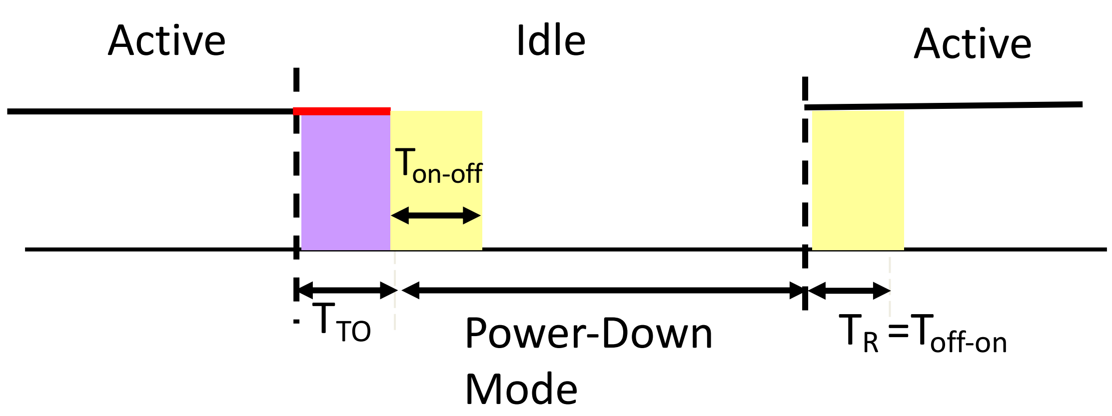
- The workload is given as a list of idle intervals
  - Values are in μs

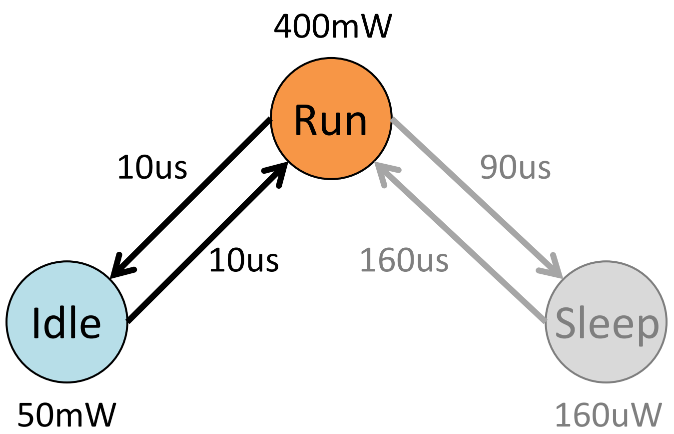
268	341	START/END OF 1st IDLE PERIOD
609	656	START/END OF 2nd IDLE PERIOD
1010	1065	…
1196	1273	
1541	1629	
2056	2139	



DPM policies

- Timeout policy
  - Observe the first part of the current idle period to predict the length of the remaining part
  - Put the device in off state T_{TO} time units after it has entered the idle state





Compile and execute

- Compile (requires gcc):

  make

- Generate Documentation (requires doxygen):

  doxygen Doxyfile

  - Generates «docs» folder with HTML documentation

- Execute:

  ./dpm_simulator -t 20 -psm example/psm.txt –wl example/wl.txt

  - Timeout policy with timeout value 20us

Compile and execute

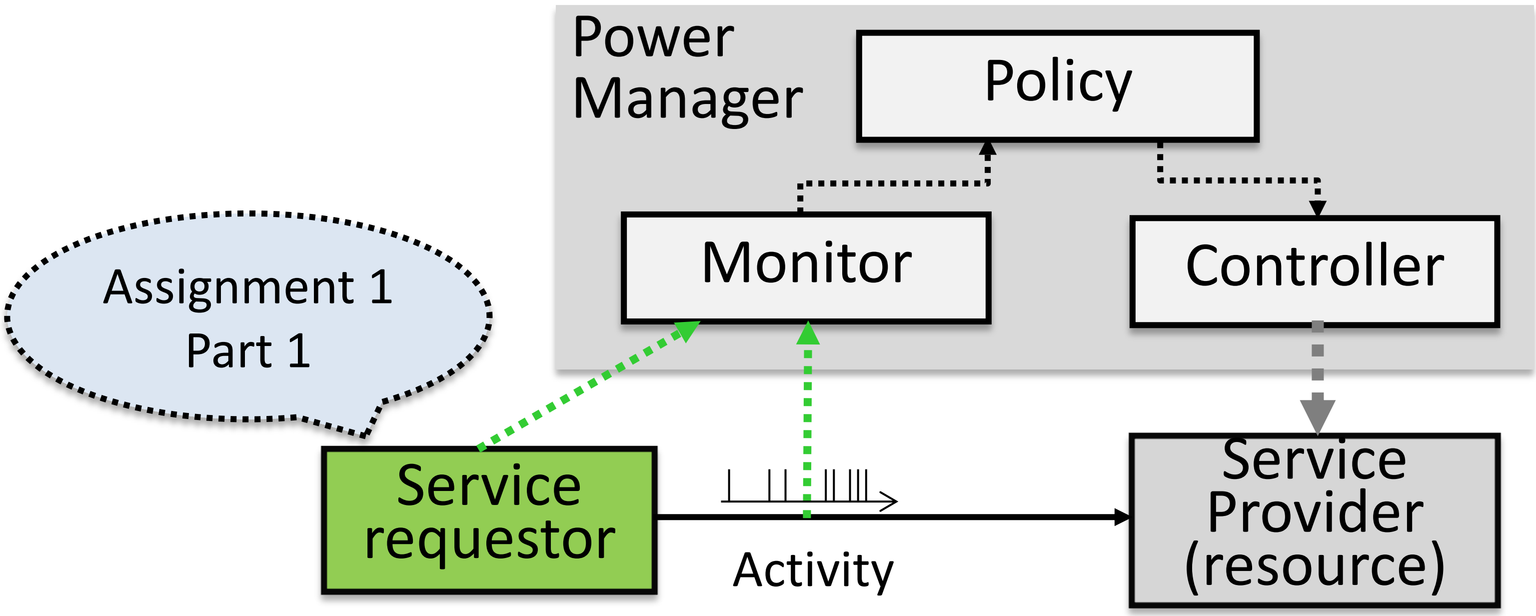
- Compile (requires gcc):

```
➜ dpm_simulator git:(master) ✗ ./dpm_simulator -t 20 -psm example/psm.txt -wl example/wl.txt
[psm] State Run: power = 400.00mW
[psm] State Idle: power = 50.00mW
[psm] State Sleep: power = 0.16mW
[psm] Run -> Idle transition: energy  = 10uJ, time = 10us
[psm] Run -> Sleep transition: energy  = 10uJ, time = 90us
[psm] Idle -> Run transition: energy  = 10uJ, time = 10us
[psm] Sleep -> Run transition: energy  = 30uJ, time = 160us

[sim] Active time in profile = 0.300130s
[sim] Idle time in profile = 0.244066s
[sim] Total time = 0.544196s
[sim] Timeout waiting time = 0.024679s
[sim] Total time in state Run = 0.324809s
[sim] Total time in state Idle = 0.219387s
[sim] Total time in state Sleep = 0.000000s
[sim] Time overhead for transition = 0.022770s
[sim] N. of transitions = 2277
[sim] Energy for transitions = 0.022770J
[sim] Energy w/o DPM = 0.217678J, Energy w DPM = 0.163663J
[sim] 24.8 percent of energy saved.
```

Assignment 1 – Part 1
Workload profile generation

Lab structure



- Workload generation
  – According to specified distributions

Workload format

- Goal: generate the input workload file according to different distributions of active and idle periods

268	341
609	656
1010	1065
1196	1273
1541	1629
2056	2139

Generate the first
ACTIVE period – long
267 time slots

- Length of ACTIVE/IDLE periods?

Workload format

- Goal: generate the input workload file according to different distributions of active and idle periods

268	341
609	656
1010	1065
1196	1273
1541	1629
2056	2139

Generate the first IDLE period: from 268 for 74 time slots

- Length of ACTIVE/IDLE periods?

Workload format

- Goal: generate the input workload file according to different distributions of active and idle periods

268	341
609	656
1010	1065
1196	1273
1541	1629
2056	2139

Generate an ACTIVE period: from 341 for 268 time slots

- Length of ACTIVE/IDLE periods?

Workload format

- Goal: generate the input workload file according to different distributions of active and idle periods

268	341
609	656
1010	1065
1196	1273
1541	1629
2056	2139

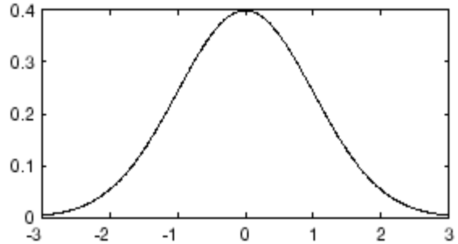
Generate an IDLE period: from 609 for 48 time slots

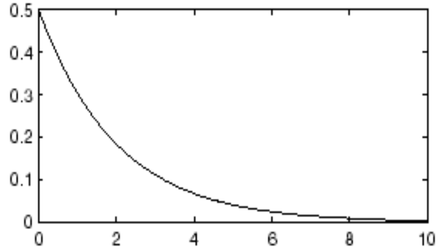
- Length of ACTIVE/IDLE periods?

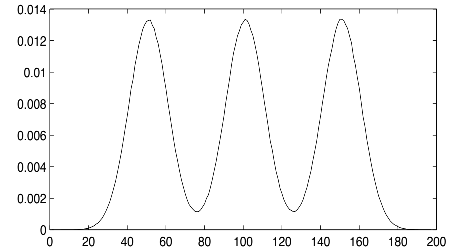
Assignment 1 – Part 1

- Generate workload distributed as follows
  - *Active periods*
    - Uniform distribution, min = 1us, max = 500us, **5000 samples!**
  - *Idle periods* distributed in various ways
    - Uniform distribution, min = 1us, max=100us (high utilization)
    - Uniform distribution, min=1us, max=400us (low utilization)
    - Normal distribution, mean=100us, standard deviation=20
    - Exponential distribution, mean=50us
    - Tri-modal distribution
      - Mean = 50, 100, 150us
      - Standard deviation=10

no zero
no negative
values



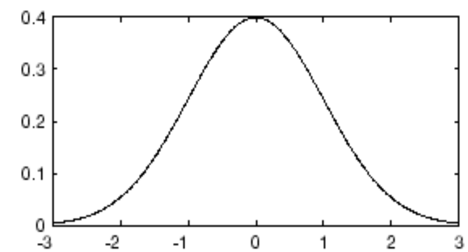


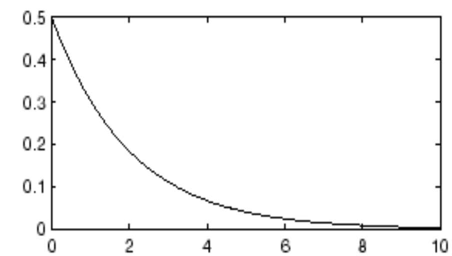


Step by step

Plot a histogram of the length of the generated periods to check that their distribution is correct

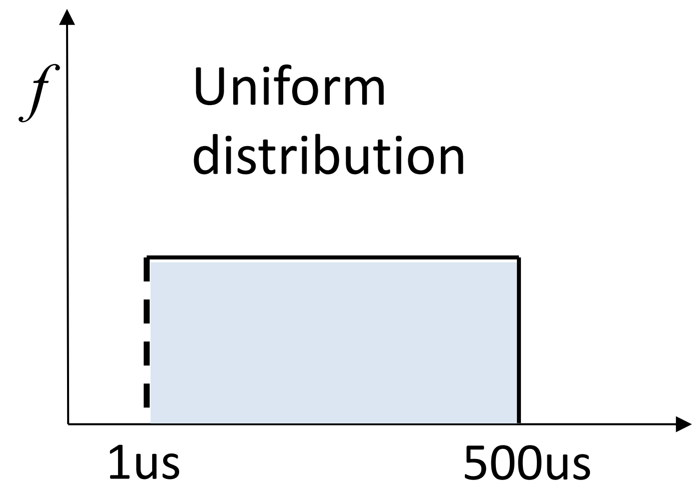
- The distribution defines the lengths of the idle/active periods
  - Normal distribution
    - Periods tend to last as long as the mean
    - Few very short periods
    - Few very long periods
  - Exponential distribution
    - Periods tend to be short
    - Longer periods are fewer
- The adopted distribution influences the effectiveness of the DPM policy





Step by step

1. How do I implement the distribution?
   – E.g., how do I generate random numbers between a minimum and a maximum?
   – This generates the lengths of your idle and active periods



2. Now you have the lengths for idle and active periods..
   Combine them to build the workload!

Assignment 1 – Part 1

- Goal:
  - Compare behavior of policies on workloads with different shapes and characteristics
  - Your generated workloads plus the **two unknown workloads provided in the lab material.**

- Report assignment:
  - Description of generated workload profiles
  - Constraints:
    - Implement in C, MATLAB or Python
    - Use Matlab or Python to view the resulting distributions