**POLITECNICO DI TORINO**

**Master’s Degree in Computer Engineering**



**Energy Management for IoT 01UDGOV**

**Lab 1 Report**

**Group-8**

**Supervisor:**

**Prof. Massimo Poncino**

**Candidates:**

**Diego Porto (s313169)**

**Mohamed elsayed Mahmoud elsayed**

**elsisy (s330451)**

**A.Y. 2024/25**

**Table of Contents**

[1. Timeout Policy 3](#_Toc191032045)

[1.1. RUN-IDELE states 3](#_Toc191032046)

[1.1.1 Workload 1- (fast sensor response’s) 3](#_Toc191032047)

[1.1.2 Workload 2 – (slow sensor response’s) 4](#_Toc191032048)

[1.2. RUN-SLEEP states 6](#_Toc191032049)

[1.2.1 Summary of Insights – WORKLAOD-1 13](#_Toc191032050)

[1.2.2 Summary of Insights – WORKLAOD-2 13](#_Toc191032051)

[1.3. Three PSM STATE MACHINE 13](#_Toc191032052)

[1.3.1 Examples 13](#_Toc191032053)

[2. History Policy 15](#_Toc191032054)

[2.1. Examples and results 15](#_Toc191032055)

[2.2. Conclusion: 16](#_Toc191032056)

1

# Timeout Policy

In the first part of timeout policy where will run the simulator using the two given workloads from two different sensor response’s with the default timeout policy. With transition constrain only between \*\*RUN\*\* and \*\*IDLE\*\* states are allowed. The PSM file used is `dpm-simulator/example/psm.txt`.

The differences in the sensor's response times between workload\_1.txt and workload\_2.txt can significantly impact the DPM policy's effectiveness and behaviour.

For example, fast Sensor (workload\_1.txt): With a response time of 4ms, it’s better to the system to stay in RUN for state compared to transition to IDLE, due to the overhead in the transitions, but in case the time out is 0 to 1, it may better to move to idle since there is saving energy 80-30 mJ.

Slow Sensor (workload\_2.txt): With a response time of 100ms, it's convenient to the system to switch to IDLE in the meantime, because it could happen that the overhead due to transition is still better than stay in RUN state all the time.

Shorter timeout values (especially the most energy efficient one (0)) may be more effective in managing power consumption, as the system can quickly transition to low-power states when the sensors are not in use. But we have also avoid frequent transitions, which can be counterproductive due to the high transition overhead.

## RUN-IDELE states

### Workload 1- (fast sensor response’s)

This workload uses "fast" sensors, so the value is returned in 4 ms. the file contains two values on each line: the first value is the arrival time of the task, and the second value represents the duration of that task. the following 4 different timeout values: 0ms, 1ms, 20ms, 40ms, 60ms the command to run the first workload with a timeout of 1ms is:

The command to run the first workload with a timeout of 1ms is:

Shell. /dpm\_simulator -t 1 -psm example/psm.txt -wl ../workloads/workload\_1.txt > results/workload\_1/workload\_1ms

The results of the simulation are stored in the folder results, divided by workload. In the output, the [sim] section provides detailed results of the simulation: Active and Inactive Time: The total time the system spent in active and inactive states. Total Time: The total simulation time, both with and without DPM. State Times: The total time spent in each state (Run, Idle, Sleep). Timeout Waiting Time: The time spent waiting for the timeout to expire. Transitions Time: The total time spent in state transitions. Number of Transitions: The total number of state transitions. Energy for Transitions: The total energy consumed during state transitions. Total Energy: The total energy consumption with and without DPM.

The results for Different Timeout Values In this first workload, we can observe the following values:

Timeout 1ms:

Number of Transitions: 138

Total Energy with DPM: 0.6997882530J

Timeout Waiting Time: 0.086400s

Timeout 20ms:

Number of Transitions: 18

Total Energy with DPM: 0.7070946960J

Timeout Waiting Time: 0.400100s

Timeout 40ms:

Number of Transitions: 18

Total Energy with DPM: 0.7119600960J

Timeout Waiting Time: 0.580100s

Timeout 60ms:

Number of Transitions: 18

Total Energy with DPM: 0.7168254960J

Timeout Waiting Time: 0.760100s

As it’s shown, different timeout values affect the number of state transitions, total energy consumption, and

The time spent in each state. Shorter timeouts lead to more frequent transitions and overall

Lower energy consumption, while longer timeouts result in fewer transitions and higher energy consumption. This is

Because the overhead caused by the transition between the RUN and IDLE states has a lower impact on energy

Consumption compared to staying in the RUN state during that time.

Conclusion Overall, from the results, we observed that lower timeout values lead to more frequent state transitions, which results in higher transition time values and energy consumption. However, they also lead to lower overall energy consumption. On the other hand, higher timeout values result in fewer

Transitions, which means lower transition times and energy consumption, but they also lead to higher overall

Energy consumption. Given these results, for the first workload (workload\_1, fast sensors), despite the high

Number of transitions, it is better to keep the timeout threshold at lower values, such as 1ms.

### Workload 2 – (slow sensor response’s)

This workload uses "slow" sensors, meaning the value is returned in 100 ms. the file workload\_2.txt contains two values in each line: the first value is the arrival time of the task, and the second value stands for the duration of that task. Due to the higher response time of the sensors, the timeout values will differ from

Those used in Workload 1. Timeouts: 80ms 100ms 120ms 140ms 160ms 180ms 200ms.

Summary of Results for Different Timeout Values Total Energy Consumption:

The energy consumption without DPM is always Tot. Energy w/o DPM = 30.1388136000J.

The total energy consumption with DPM for different timeout values is:

timeout\_80ms:

Tot. Energy w DPM = 0.9073267669J

timeout\_100ms:

Tot. Energy w DPM = 0.9464934979J

timeout\_120ms:

Tot. Energy w DPM = 0.9552641029J

timeout\_140ms:

Tot. Energy w DPM = 0.9606701029J

timeout\_160ms:

Tot. Energy w DPM = 0.9660761029J

timeout\_180ms:

Tot. Energy w DPM = 0.9714821029J

timeout\_200ms:

Tot. Energy w DPM = 0.9768881029J

\*\*State\*\*: - Analyze the total time spent in different states (RUN and IDLE) for each timeout value:

timeout\_80ms: `Total time in state Run = 10.469800s`, `Total time in state Idle = 1081.439100s` -

timeout\_100ms: `Total time in state Run = 11.968700s`, `Total time in state Idle = 1079.995400s` -

timeout\_120ms: `Total time in state Run = 12.306200s`, `Total time in state Idle = 1079.671900s` -

timeout\_140ms: `Total time in state Run = 12.506200s`, `Total time in state Idle = 1079.471900s` -

timeout\_160ms: `Total time in state Run = 12.706200s`, `Total time in state Idle = 1079.271900s` -

timeout\_180ms: `Total time in state Run = 12.906200s`, `Total time in state Idle = 1079.071900s` -

timeout\_200ms: `Total time in state Run = 13.106200s`, `Total time in state Idle = 1078.871900s`

Number of Transitions:

State Transitions and Energy Consumption:

timeout\_80ms: Number of transitions = 194 Energy for transitions = 0.0019400000J

timeout\_100ms: Number of transitions = 56 Energy for transitions = 0.0005600000J

timeout\_120ms, timeout\_140ms, timeout\_160ms, timeout\_180ms, timeout\_200ms:

Number of transitions = 20 Energy for transitions = 0.0002000000J

Timeout Waiting Time:

Timeout Waiting Time for Different Timeout Values:

timeout\_80ms: Timeout waiting time = 7.922800s

timeout\_100ms: Timeout waiting time = 9.421700s

timeout\_120ms: Timeout waiting time = 9.759200s

timeout\_140ms: Timeout waiting time = 9.959200s

timeout\_160ms: Timeout waiting time = 10.159200s

timeout\_180ms: Timeout waiting time = 10.359200s

timeout\_200ms: Timeout waiting time = 10.559200s

Conclusion Optimal Timeout Value: As the timeout value increases, the total energy consumption with DPM

Slightly increases. For instance, the energy consumption with DPM is 0.9073J at 80ms and increases to 0.9769J

at 200ms. This indicates that shorter timeout values are more energy-efficient. Number of Transitions:

The number of state transitions decreases significantly as the timeout value increases. For example, there are

194 transitions at 80ms compared to only 20 transitions at 200ms. While fewer transitions can reduce the

Overhead and energy consumption associated with state changes, it is still beneficial to switch to the IDLE state

Despite the transition overhead. Timeout Waiting Time: The timeout waiting time increases with higher timeout

Values. This suggests that higher timeout values lead to longer periods of inactivity before transitioning to

A low-power state.

Final Conclusion: With workload\_2, similar to workload\_1, shorter timeouts lead to more frequent transitions

But overall lower energy consumption. On the other hand, longer timeouts result in fewer transitions but

Higher energy consumption. Additionally, as expected, the energy consumed with DPM is significantly lower

Than without DPM across all timeout values for both workloads.

## RUN-SLEEP states

Modify the timeout policy to enable transitions also to \*SLEEP\*. To do this it is necessary to modify the `dpm\_decide\_state` function inside `dpm\_policies.c` changing to the next macros "PSM\_STATE\_SLEEP” which in psm.h file, as well as changing the struct of “dpm\_timeout\_params” in “dpm\_policies.h" and create array of two double numbers which will contain timeout\_0 and timeout\_1 for run and sleep “ psm\_time\_t timeout[2]” plus edit the in file “utilities.h” and edit the struct of what has been done “dpm\_policies.h”. the editing in the code in the three file is attached done as well as Let's compare the results the results/workload\_2 folder. We will focus on the energy consumption and the time spent in different states.

“psm.h”

typedef struct {

/\* Day2: you can add/change stuff here \*/

psm\_time\_t timeout[2];

} dpm\_timeout\_params;

“utilities.h”

tparams->timeout[0] = atof(argv[++cur]);

tparams->timeout[1] = atof(argv[++cur]);

printf("the time out values as input are <<<<<<<<<<<>>>>>>>>>>>>>>\n");

printf("the value of time out idle is %f \n",tparams->timeout[0]);

printf("the value of time out sleep is %f \n",tparams->timeout[1]);

“dpm\_policies.c”

if((t\_curr > t\_inactive\_start + tparams.timeout[1]) && (tparams.timeout[1] > tparams.timeout[0]))

\*next\_state = PSM\_STATE\_SLEEP;

else if((t\_curr > t\_inactive\_start + tparams.timeout[0]) && (tparams.timeout[1] < tparams.timeout[0]))

\*next\_state = PSM\_STATE\_IDLE;

else {

\*next\_state = PSM\_STATE\_RUN;}

break;

**Comparison for workload\_2 with IDLE and SLEEP**

**Timeout\_0ms vs timeout\_0ms\_sleep**

**Total Time in States**  
timeout\_0ms: More time in Idle state (1089.350400s), no time in Sleep state.  
timeout\_0ms\_sleep: All inactive time spent in Sleep state (1088.938200s), no time in Idle state.

**Timeout Waiting Time**  
timeout\_0ms: 0.009900s  
timeout\_0ms\_sleep: 0.009900s

**Transitions Time**  
timeout\_0ms: 0.079200s  
timeout\_0ms\_sleep: 0.495000s

**Number of Transitions**  
timeout\_0ms: 198transitions.  
timeout\_0ms\_sleep: 198 transitions.

**Energy for Transitions**  
timeout\_0ms: 0.0019800000J  
timeout\_0ms\_sleep: 0.1999800000J

**Total Energy with DPM**  
timeout\_0ms: 0.6934801680J  
timeout\_0ms\_sleep: 0.3685548779J

**Timeout\_80ms vs timeout\_80ms\_sleep**

**Total Time in States**  
timeout\_80ms: More time in Idle state (1081.439100s), no time in Sleep state.  
timeout\_80ms\_sleep: All inactive time spent in Sleep state (1081.076900s), no time in Idle state.

**Timeout Waiting Time**  
timeout\_80ms: 7.922800s  
timeout\_80ms\_sleep: 7.906200s

**Transitions Time**  
timeout\_80ms: 0.077600s  
timeout\_80ms\_sleep: 0.460000s

**Number of Transitions**  
timeout\_80ms: 194 transitions.  
timeout\_80ms\_sleep: 184 transitions.

**Energy for Transitions**  
timeout\_80ms: 0.0019400000J  
timeout\_80ms\_sleep: 0.1858400000J

**Total Energy with DPM**  
timeout\_80ms: 0.9073267669J  
timeout\_80ms\_sleep: 0.5716452409J

**Timeout\_100ms vs timeout\_100ms\_sleep:**

**Total Time in States**  
timeout\_100ms: More time in Idle state (1079.995400s), no time in Sleep state.  
timeout\_100ms\_sleep: All inactive time spent in Sleep state (1079.949900s), no time in Idle state.

**Timeout Waiting Time**  
timeout\_100ms: 9.421700s  
timeout\_100ms\_sleep: 9.353200s

**Transitions Time**  
timeout\_100ms: 0.022400s  
timeout\_100ms\_sleep: 0.140000s

**Number of Transitions**  
timeout\_100ms: 56 transitions.  
timeout\_100ms\_sleep: 56 transitions.

**Energy for Transitions**  
timeout\_100ms: 0.0005600000J  
timeout\_100ms\_sleep: 0.0565600000J

**Total Energy with DPM**  
timeout\_100ms: 0.9464934979J  
timeout\_100ms\_sleep: 0.4822010109J

**Comparison for workload\_1 with IDLE and SLEEP**

**Timeout\_0ms vs timeout\_0ms\_sleep:**

**Total Time in States**  
timeout\_0ms: More time in Idle state (1079.576000s), no time in Sleep state.  
timeout\_0ms\_sleep: All inactive time spent in Sleep state (1079.443000s), no time in Idle state.

**Timeout Waiting Time**  
timeout\_0ms: 7.922800s  
timeout\_0ms\_sleep: 7.906200s

**Transitions Time**  
timeout\_0ms: 0.068000s  
timeout\_0ms\_sleep: 0.205000s

**Number of Transitions**  
timeout\_0ms: 170 transitions.  
timeout\_0ms\_sleep: 82 transitions.

**Energy for Transitions**  
timeout\_0ms: 0.0017000000J  
timeout\_0ms\_sleep: 0.0828200000J

**Total Energy with DPM**  
timeout\_0ms: 0.6979953200J  
timeout\_0ms\_sleep: 0.2607854300J

**Timeout\_1ms vs timeout\_1ms\_sleep:**

timeout\_1ms: More time in Idle state (1079.510900s), no time in Sleep state.  
timeout\_1ms\_sleep: All inactive time spent in Sleep state (1079.414800s), no time in Idle state.

**Timeout Waiting Time**  
timeout\_1ms: 0.086400s  
timeout\_1ms\_sleep: 0.047300s

**Transitions Time**  
timeout\_1ms: 0.055200s  
timeout\_1ms\_sleep: 0.190000s

**Number of Transitions**  
timeout\_1ms: 138 transitions.  
timeout\_1ms\_sleep: 76 transitions.

**Energy for Transitions**  
timeout\_1ms: 0.0017000000J  
timeout\_1ms\_sleep: 0.0767600000J

**Total Energy with DPM**  
timeout\_1ms: 0.6997882530J  
timeout\_1ms\_sleep: 0.2559152120J

**Timeout\_4ms vs timeout\_4ms\_sleep:**

timeout\_4ms: More time in Idle state (1079.388800s), no time in Sleep state.  
timeout\_4ms\_sleep: All inactive time spent in Sleep state (1079.383400s), no time in Idle state.

**Timeout Waiting Time**  
timeout\_4ms: 0.256100s  
timeout\_4ms\_sleep: 0.223700s

**Transitions Time**  
timeout\_4ms: 0.007200s  
timeout\_4ms\_sleep: 0.045000sv

**Number of Transitions**  
timeout\_4ms: 18 transitions.  
timeout\_4ms\_sleep: 18 transitions.

**Energy for Transitions**  
timeout\_4ms: 0.0001800000J  
timeout\_4ms\_sleep: 0.0181800000J

**Total Energy with DPM**  
timeout\_4ms: 0.7032023760J  
timeout\_4ms\_sleep: 0.2022010260J

**Timeout\_20ms vs timeout\_20ms\_sleep:**

**Total Time in States**  
timeout\_20ms: More time in Idle state (1079.244800s), no time in Sleep state.  
timeout\_20ms\_sleep: All inactive time spent in Sleep state (1079.239400s), no time in Idle state.

**Timeout Waiting Time**  
timeout\_20ms: 0.400100s  
timeout\_20ms\_sleep: 0.367700s

**Transitions Time**  
timeout\_20ms: 0.007200s  
timeout\_20ms\_sleep: 0.045000s

**Number of Transitions**  
timeout\_20ms: 18 transitions.  
timeout\_20ms\_sleep: 18 transitions.

**Energy for Transitions**  
timeout\_20ms: 0.0001800000J  
timeout\_20ms\_sleep: 0.0181800000J

**Total Energy with DPM**  
timeout\_20ms: 0.7070946960J  
timeout\_20ms\_sleep: 0.2061624660J

**Timeout\_40ms vs timeout\_40ms\_sleep:**

**Total Time in States**  
timeout\_40ms: More time in Idle state (1079.064800s), no time in Sleep state.  
timeout\_40ms\_sleep: All inactive time spent in Sleep state (1079.059400s), no time in Idle state.

**Timeout Waiting Time**  
timeout\_40ms: 0.580100s  
timeout\_40ms\_sleep: 0.547700s

**Transitions Time**  
timeout\_40ms: 0.007200s  
timeout\_40ms\_sleep: 0.045000s

**Number of Transitions**  
timeout\_40ms: 18 transitions.  
timeout\_40ms\_sleep: 18 transitions.

**Energy for Transitions**  
timeout\_40ms: 0.0001800000J  
timeout\_40ms\_sleep: 0.0181800000J

**Total Energy with DPM**  
timeout\_40ms: 0.7119600960J  
timeout\_40ms\_sleep: 0.2111142660J

**Timeout\_60ms vs timeout\_60ms\_sleep:**

**Total Time in States**  
timeout\_60ms: More time in Idle state (1078.884800s), no time in Sleep state.  
timeout\_60ms\_sleep: All inactive time spent in Sleep state (1078.879400s), no time in Idle state.

**Timeout Waiting Time**  
timeout\_60ms: 0.760100s  
timeout\_60ms\_sleep: 0.727700s

**Transitions Time**  
timeout\_60ms: 0.007200s  
timeout\_60ms\_sleep: 0.045000s

**Number of Transitions**  
timeout\_60ms: 18 transitions.  
timeout\_60ms\_sleep: 18 transitions.

**Energy for Transitions**  
timeout\_60ms: 0.0001800000J  
timeout\_60ms\_sleep: 0.0181800000J

**Total Energy with DPM**  
timeout\_60ms: 0.7168254960J  
timeout\_60ms\_sleep: 0.2160660660J

### Summary of Insights – WORKLAOD-1

**Energy Consumption**

The simulations with the "sleep" parameter (timeout\_4ms) show significantly lowest total energy consumption with DPM compared to their counterparts without the "sleep" parameter.

The simulations with the "IDLE" parameter (timeout\_0ms) show significantly lowest total energy consumption with DPM compared to their counterparts without the "IDLE" parameter.

Therefore the best value for timeout for Sleep mode with respect to workload\_1 is 4 ms and this is very convenient when know that average response is 4 ms for fast sensor response.

The time out value for IDLE state is also convenient to be zero to that be logic to transit from run to idle and save energy overall BUT also the 4 ms for IDLE is also can better more than zero if we take into consideration number of transition is 18 not 82, the 0 for idle timeout and 4 ms for sleep timeout has been tested therefore this result is based on the experiment. Overall 4ms/sleep-4ms/idle or 4ms sleep/ 0 idle.

### Summary of Insights – WORKLAOD-2

**Energy Consumption**

The simulations with the "sleep" parameter (timeout\_0ms\_sleep) show significantly lowest total energy consumption with DPM compared to their counterparts without the "sleep" parameter.

The simulations with the "IDLE" parameter (timeout\_0ms\_IDLE) show as well significantly lowest total energy consumption with DPM compared to their counterparts without the "IDLE" parameter.

## Three PSM STATE MACHINE

In this three PSM state machine, its not binary state machine from idle to run and vice versa or sleep to run and vice versa, but the transition can be idle to run to sleep or sleep to run to idle or moving from run to any state (constrain ). The editing in the code is to get the timeout of sleep and time out of idle where the tparams.timeout[1] is for sleep timeout and tparams.timeout[0] is for idle, the transition will be for higher priority time out sleep if we pass time out sleep higher than time idle otherwise it goes to idle state, but during the workload the PSM process will be up to the duration of idleness of the system so it will be some time idle and some sleep achieving the best saving amount of energy at the same time the constrain of transition between the states will be as it needs in the report.

### Examples

For example and as practical example when the result for best value for sleep 4 ms and 4ms idle or 4 ms sleep and 0 idle for workload\_1, and when insert in the three state system the value for idle timeout 0 and for the sleep 4ms the result will be as follows :

Energy w DPM = 0.1976708950J

Energy for transitions = 0.0195400000J

N. of transitions = 154

Transitions time = 0.099400s

Timeout waiting time = 0.006800s

Total time in state Sleep = 1079.376200s

Total time in state Idle = 0.170100s

Total time in state Run = 2.930800s

And as at shown the value of energy resulting will be lower than putting the system in sleep mode in idleness time with time out 4 ms which save energy to be timeout\_4ms\_sleep: 0.2022010260J or to put the system in 0 for idle time out which save energy till 0.6979953200J.

The second example is for workload 2 when the value for sleep and idle is 0 when pass the value to three state machine the result will be as the PSM is always in sleep mode, but the result always good or at least like sleep mode. The result of 0/0 will be the same like 0 in sleep mode for workload\_2.

if(tparams.timeout[1] > tparams.timeout[0] && (t\_curr > t\_inactive\_start + tparams.timeout[1]))

{

if((prev\_state == PSM\_STATE\_SLEEP) || (prev\_state == PSM\_STATE\_RUN) )

{

\*next\_state = PSM\_STATE\_SLEEP;

}

else {

\*next\_state = PSM\_STATE\_RUN;}

}

else if((t\_curr > t\_inactive\_start + tparams.timeout[0]))

{ if((prev\_state == PSM\_STATE\_IDLE) || (prev\_state == PSM\_STATE\_RUN) )

{ \*next\_state = PSM\_STATE\_IDLE;

}

else {

\*next\_state = PSM\_STATE\_RUN;}

} else {

\*next\_state = PSM\_STATE\_RUN; } break;

# History Policy

The History Policy is a predictive policy. it estimates the time to transition based on the previous workload pattern. The computed value *Tpred* is compared to the time to Idle and Sleep thresholds. After a decision is taken the system is able to reach any one of the low-power states independently.

In the code section that is related to DPM history in dpm\_policies.c , the implementation of the code as follow is it inside the inactive time ? Yes? Ok, start to calculate the *Tpred* which method? In the code as it’s shown there is two methods the polynomial and regression which everyone can be activated through #ifdef macros after that and once time prediction is calculated, it goes to the second step where the time prediction has to select to go to sleep or idle and this must be done through achieving these terms for example in order to go to sleep state it should time prediction be higher time breakeven, time of threshold sleep as well as the priority to sleep than idle so the sleep must be higher than idle threshold if its so then the next state will be sleep otherwise it goes for second priority idle and the time prediction has to validate these terms firstly must be higher than time breakeven of idle as well as threshold of idle time otherwise failing to meet terms of the sleep or idle makes the next stage moving to run since the transition must be from sleep to run or idle to run and vice versa. The code is edited with the way to move from the three states.

## Examples and results

For example at workload\_1 with the best threshold timeout policy was 4 ms for sleep mode and 0 for idle when combines 4 ms/sleep mode with 0ms for idle in three state with timout policy the result of energy was 0.1976708950J, in the case of history policy using polynomial with the following coefficient k1,0.9 k2,0.80 k3,.70 k4,.60 k5, .50 with threshold 4ms for sleep and 0 for idle it results:

Energy w DPM = 0.2587870080J

Energy for transitions = 0.0808200000J

N. of transitions = 82

Transitions time = 0.200800s

Timeout waiting time = 0.004100s

Total time in state Sleep = 1079.444700s

Total time in state Idle = 0.002500s

Total time in state Run = 2.928100s

The result is pretty fine in general and close to the result of three state time out policy the changing in the coefficient lead to more saving or increasing to DPM energy but from practical examples in case 4ms/sleep and 0ms it’s around 0.25 to 0.27 J

Example two since the predictive three state machine can also work as binary machine run to idle and vice versa or sleep to run and vice versa, repeating threshold equal to 1 ms for idle period

With predictive history DPM and ignoring threshold of sleep for example disable it “0” so the machine can olny trigger between idle and run with using the last coefficients k1,0.9 k2,0.80 k3,.70 k4,.60 k5, .50 it results as follows

Predcitve\_1ms\_idle: Energy w DPM = 0.6980541630J

Predcitve\_1ms\_Sleep: Energy w DPM = 0.2587870080J

timeout\_1ms: 0.6997882530J  
 timeout\_1ms\_sleep: 0.2559152120J

With workload\_1 the result of predictive is close to timeout policy, and better from point of idle threshold then time out idle policy while isn’t better than timeout sleep policy but it gives close results.

For workload\_2 using the same coefficient the result for threshold idle and sleep is better than using time out policy for idle or sleep for example

Predcitve\_80ms\_idle :0.7203257410J

Predcitve\_80ms\_sleep: 0.3670321360J

timeout\_80ms: 0.9073267669J  
timeout\_80ms\_sleep: 0.5716452409J

Predcitve\_100ms\_idle : 0.7203257410J

Predcitve\_100ms\_sleep: 0.3670321360J

timeout\_100ms: 0.9464934979J  
timeout\_100ms\_sleep: 0.4822010109J

## Conclusion:

In workload\_1 using threshold idle with predictive policy is better than using threshold idle with time out policy but using threshold sleep with time out policy better than use predictive policy in threshold sleep.

In workload\_2 using the predictive of both threshold sleep/idle gives better in saving energy than using predictive policy, the predictive policy result for both idle and sleep still acceptable but not better than time out policy.

|  |
| --- |
| if(t\_curr > t\_inactive\_start)  {  #ifdef T\_Pre\_Poly  for( int i =0; i < DPM\_HIST\_WIND\_SIZE; i++)  {  T\_Pre += hparams.alpha[i] \* pow(history[i],i);  }  #endif  #ifdef T\_Pre\_N\_Reg  for( int i =1; i < DPM\_HIST\_WIND\_SIZE; i++)  {  T\_Pre += hparams.alpha[i]\*(history[DPM\_HIST\_WIND\_SIZE-(i)]);  }  T\_Pre += hparams.alpha[0];  #endif  if((T\_Pre >= T\_B\_E\_S) && (T\_Pre >= Thr\_Sleep) && (Thr\_Sleep > Thr\_Idle ))  {  if((prev\_state == PSM\_STATE\_SLEEP) || (prev\_state == PSM\_STATE\_RUN) )  {  \*next\_state = PSM\_STATE\_SLEEP;  }  else {  \*next\_state = PSM\_STATE\_RUN;}  }  else if((T\_Pre >= T\_B\_E\_I) && (T\_Pre >= Thr\_Idle))  {    if((prev\_state == PSM\_STATE\_IDLE) || (prev\_state == PSM\_STATE\_RUN) )  {  \*next\_state = PSM\_STATE\_IDLE;  }  else {  \*next\_state = PSM\_STATE\_RUN;}  }  }  else {  \*next\_state = PSM\_STATE\_RUN;  }  break;  #endif |