CSC 714 Project Report 3

Nachiket S Deshpande
nsdeshpa@ncsu.edu

"Study of memory leakage on an IBM PowerPC 405LP Embedded Processor and ways to reduce energy consumption, by combining sleep and low-power modes."

Project URL: http://www4.ncsu.edu/~nsdeshpa/project.html

Solved Issues

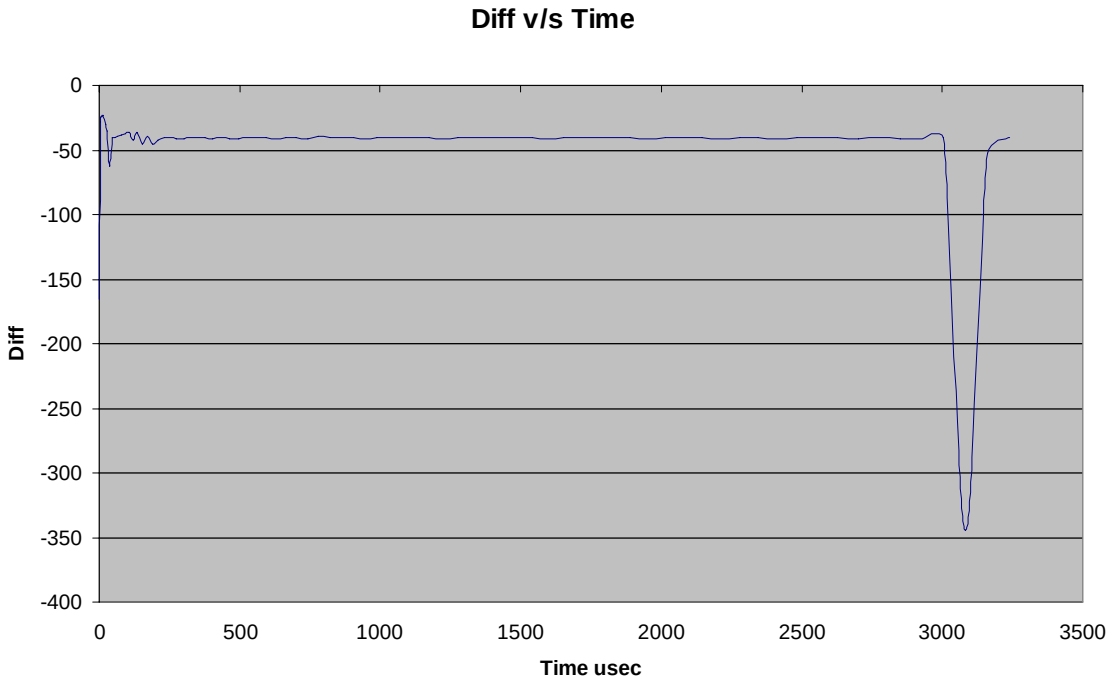
- It was learnt that the PIT, FIT and all related timers get suspended when the processor goes to sleep, so the approach involving the use of these timers was shelved.
- The other option was to use KURT Linux, which would have involved patching the existing Linux distribution on the PC, experiment with the new patch and see the effects on the board. It would have involved modifying the kernel code for the processor test board, and as a result, this approach was nulled out as well.
- A function, defined in the KURT Linux documentation, called nanosleep_test, made use of the nanosleep function provided on the current distribution to simulate microsecond granularity sleeps. This function was integrated into the code and testing was done.
- The new approach involved calling nanosleep successively after wakeup for small durations (1us, 2us, 3us..) and observe the values of sleep actually reported using successive gettimeofday() calls. The nanosleep function was placed between 2 gettimeofday() calls, and the values read were stored into 2 separate arrays. This was attempted to be done for 120 datapoints, but since the system starts trying to re-establish a network connection, the actions taken interfere with the test code and so only 80 values could be plotted.
- A third, result array was used to store the difference observed between the 2 gettimeofday() calls. These values were noted down as the "actual" sleep values.
- A table was generated, in which I compared the ideal sleeps (1us, 2us..) with the actual sleeps observed and a third column was used to take the difference between these two.
- A graph of the differences was plotted versus the timebase (The timebase was obtained by adding successive actual sleep values (1us + 2us + 3us...)

- The table generated was as follows:

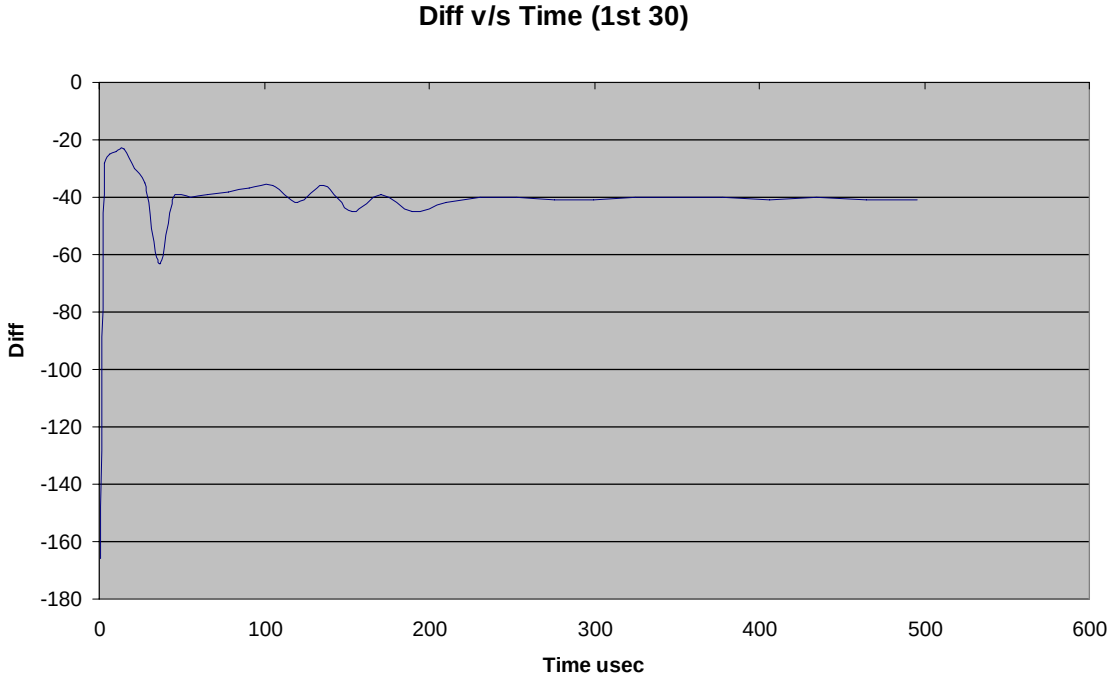
Sample	Desired nanosleep time (usec)	Actual nanosleep time (usec)	Timebase (usec)	Diff (Desired – Actual) (usec)
0	1	167	1	-166
1	2	30	3	-28
2	3	28	6	-25
3	4	28	10	-24
4	5	28	15	-23
5	6	36	21	-30
6	7	42	28	-35
7	8	71	36	-63
8	9	49	45	-40
9	10	50	55	-40
10	11	50	66	-39
11	12	50	78	-38
12	13	50	91	-37
13	14	50	105	-36
14	15	57	120	-42
15	16	52	136	-36
16	17	62	153	-45
17	18	57	171	-39
18	19	64	190	-45
19	20	62	210	-42
20	21	61	231	-40
21	22	62	253	-40
22	23	64	276	-41
23	24	65	300	-41
24	25	65	325	-40
25	26	66	351	-40
26	27	67	378	-40
27	28	69	406	-41
28	29	69	435	-40
29	30	71	465	-41
30	31	72	496	-41
31	32	72	528	-40
32	33	73	561	-40
33	34	74	595	-40
34	35	76	630	-41
35	36	76	666	-40
36	37	77	703	-40
37	38	79	741	-41

38	39	78	780	-39
39	40	80	820	-40
40	41	81	861	-40
41	42	82	903	-40
42	43	84	946	-41
43	44	84	990	-40
44	45	85	1035	-40
45	46	86	1081	-40
46	47	87	1128	-40
47	48	88	1176	-40
48	49	90	1225	-41
49	50	90	1275	-40
50	51	91	1326	-40
51	52	92	1378	-40
52	53	93	1431	-40
53	54	94	1485	-40
54	55	95	1540	-40
55	56	97	1596	-41
56	57	97	1653	-40
57	58	98	1711	-40
58	59	99	1770	-40
59	60	100	1830	-40
60	61	101	1891	-40
61	62	103	1953	-41
62	63	103	2016	-40
63	64	104	2080	-40
64	65	105	2145	-40
65	66	107	2211	-41
66	67	107	2278	-40
67	68	108	2346	-40
68	69	110	2415	-41
69	70	110	2485	-40
70	71	111	2556	-40
71	72	112	2628	-40
72	73	114	2701	-41
73	74	114	2775	-40
74	75	116	2850	-41
75	76	117	2926	-41
76	77	117	3003	-40
77	78	422	3081	-344
78	79	132	3160	-53
79	80	120	3240	-40

A graphical representation for the diff V time values was drawn as follows:



From the graph and the data values from the table, it was seen that the diff values stabilize at around 210 usec (seen in the next graph)



- Thus, it was concluded that the overhead associated with the transition from asleep to fully awake is 210usec. Since there is a 120 usec offset associated with the system activity, the wakeup overhead = 210 – 120 = 90 usec.

Open Issues

- The system does not recover back to full functionality after coming out of sleep – the user has to manually reset the board to make it restart.
- All experimentation performed can be observed only with a minicom connection (the experimentation results cannot be seen if the user telnets to the board). The string to be written and computation done before going to sleep is echoed back as there is an active connection, but after waking up from sleep, the activity is not reflected in the telnet connection scenario. Only using minicom works – this is a hardware design issue.
- The overall accuracy of measurements could possibly be refined by using a way thorugh which it would be possible for the user to access the hardware from user space using assembly instructions for accessing the PIT written in a C test code, however whether this is possible or not, is something to be explored (I tried using assembly within C, but the approach did not work as expected).
- Similarly, even KURT Linux could be tried out, but it would make the solution specific to that particular processor and would lose its generality, as is the case right now.
- Currently, the sleep can be specified only in hh:mm:ss format. Granularity below this has not been provided by the manufacturer as of now, and this is a hardware issue as well.
- Combining these techniques with the existing DVS algorithms already incorporated and then performing experiments to see the effectiveness of using sleep modes in combination with DVS.