Power Management In PowerPC 405LP : Front Bus Scaling

FINAL REPORT

Mohamed Nishar Kamaruddin(mkamaru)
Santhosh Selvaraj(sselvar)
Instructor: Dr.Frank Mueller

CSC714
REAL-TIME COMPUTER SYSTEMS
NORTH CAROLINA STATE UNIVERSITY

1. PROBLEM STATEMENT

Power management is an important aspect in embedded real-time systems design. Dynamic Power Management(DPM) techniques allow power parameters to be changed even while programs are in execution[1]. Dynamic Volatage Scaling(DVS) and Dynamic Frequency Scaling(DFS) are widely used DPM strategies to reduce processor power consumption.

Previous research work in NCSU had led to the development of Feedback-EDF scheduling algorithms which were shown to produce better energy savings in real-time systems with dynamic workloads[2]. These effectively scale the voltage/frequency of the processor based on feedback from previous tasks. But Feedback-EDF scheduling algorithms are intended only to reduce power consumption by the processor. In embedded systems, memory subsystem is also a considerable contributor to the overall power consumption of the system. In our work, we attempt to study the power conservation that could be achieved by scaling voltage/frequency of the memory subsystem in the existing feedback EDF scheduling framework.

2. LEARNINGS & ACCOMPLISHMENTS

The DPM module(DPM405LP.c) defined the various operating points used by the feedback EDF schedular and exported APIs to change operating points dynamically. The example.c sample application was available to us. This was used to spawn mutiple threads which were dispatched by the feedback EDF scheduler defined in my_threads.c. The voltage and current readings were made through the avg.c program.

2.1 PART I

Feedback EDF scheduler would dynamically select operating points without us able to keep track of those. Hence our first experiments were to execute example.c sample application without feedback EDF scheduling. We defined two different operating points(Table 1) with same processor frequency and with different Processor Local Bus(PLB) frequency. It should be noted that memory subsystem operated at the PLB frequency.

Existing :{ "200/100/24", {1500, 1, 24, 4, 2, 32, 32, 4, 4, 4, 0, 0, -1, -1} },
Newly added :{ **"200/50/24"**, {1500, 1, 24, 4, **4**, 32, 32, 4, 4, 4, 0, 0, -1, -1} }

Table 1. Defined operating points in dpm405lp.c

2.1.1 RESULTS

The modified sample application(example.c) was executed with each of the operating points mentioned in Table1. The avg.c was modified to capture current/voltage for the memory subsystem in addition to the current/voltage of the processor. The results are available in Table2 below.

A: Channel 1 reading
B: Voltage for CPU
C: Channel 2 reading
D: Converted voltage reading from channel 2
E: Current for CPU
F: Channel 3 reading
G: Voltage for I/O subsystem
H: Channel 4 reading

I: Converted voltage reading from channel 4
J: Current for I/O subsystem

Volatile + File Reading application @ PLB 100 MHz:

Time	A	B	C	D	E	F	G	H	I	J
0.000000	2365	1.55067	2079	0.15385	0.05554	2720	3.28449	2086	0.18803	0.06788
0.008260	2368	1.56532	2088	0.19780	0.07141	2720	3.28449	2086	0.18803	0.06788
0.016520	2364	1.54579	2114	0.32479	0.11725	2719	3.27961	2086	0.18803	0.06788
0.024780	2362	1.53602	2083	0.17338	0.06259	2719	3.27961	2086	0.18803	0.06788
0.033040	2367	1.56044	2088	0.19780	0.07141	2719	3.27961	2089	0.20269	0.07317
0.041300	2365	1.55067	2137	0.43712	0.15780	2719	3.27961	2087	0.19292	0.06965
0.049560	2366	1.55556	2077	0.14408	0.05201	2719	3.27961	2086	0.18803	0.06788
0.057820	2365	1.55067	2089	0.20269	0.07317	2719	3.27961	2086	0.18803	0.06788
0.066080	2365	1.55067	2161	0.55433	0.20012	2720	3.28449	2088	0.19780	0.07141
0.074340	2365	1.55067	2092	0.21734	0.07846	2719	3.27961	2087	0.19292	0.06965

Infinite empty Loop @ PLB 100 MHz:

Time	A	B	C	D	E	F	G	H	I	J
0.000000	2367	1.56044	2175	0.62271	0.22481	2720	3.28449	2087	0.19292	0.06965
0.008194	2365	1.55067	2239	0.93529	0.33765	2719	3.27961	2087	0.19292	0.06965
0.016388	2366	1.55556	2160	0.54945	0.19836	2720	3.28449	2086	0.18803	0.06788
0.024582	2365	1.55067	2179	0.64225	0.23186	2720	3.28449	2088	0.19780	0.07141
0.032776	2365	1.55067	2246	0.96947	0.34999	2720	3.28449	2086	0.18803	0.06788
0.040970	2366	1.55556	2161	0.55433	0.20012	2719	3.27961	2087	0.19292	0.06965
0.049164	2367	1.56044	2181	0.65201	0.23538	2720	3.28449	2086	0.18803	0.06788
0.057358	2365	1.55067	2225	0.86691	0.31296	2720	3.28449	2086	0.18803	0.06788
0.065552	2366	1.55556	2163	0.56410	0.20365	2719	3.27961	2087	0.19292	0.06965
0.073746	2365	1.55067	2177	0.63248	0.22833	2720	3.28449	2087	0.19292	0.06965

Infinite empty Loop @ PLB 50 MHz:

Time	A	B	C	D	E	F	G	H	I	J
0.000000	2369	1.57021	2137	0.43712	0.15780	2720	3.28449	2080	0.15873	0.05730
0.008194	2370	1.57509	2143	0.46642	0.16838	2721	3.28938	2079	0.15385	0.05554
0.016388	2371	1.57998	2151	0.50549	0.18249	2720	3.28449	2080	0.15873	0.05730
0.024582	2372	1.58486	2162	0.55922	0.20188	2720	3.28449	2079	0.15385	0.05554
0.032776	2373	1.58974	2190	0.69597	0.25125	2720	3.28449	2078	0.14896	0.05378
0.040970	2368	1.56532	2246	0.96947	0.34999	2720	3.28449	2079	0.15385	0.05554
0.049164	2369	1.57021	2138	0.44200	0.15957	2720	3.28449	2079	0.15385	0.05554
0.057358	2370	1.57509	2144	0.47131	0.17015	2722	3.29426	2079	0.15385	0.05554
0.065552	2371	1.57998	2152	0.51038	0.18425	2720	3.28449	2080	0.15873	0.05730
0.073746	2374	1.59463	2165	0.57387	0.20717	2720	3.28449	2080	0.15873	0.05730

Table 2. Result of initial experiments.

Following are energy measurements for memory subsystem derived from the results from Table 2.

- File Reading application @ PLB 100 MHz:
  Power 234.820145 mWatt over 8.243480 secs
  => Energy 0.537704 mWatt-hours
- Infinite Loop @ PLB 50 Mhz:
  Power - 149.464111 mWatt over 8.177612 secs
  => **E**nergy: 0.339517 mWatt-hours
- Infinite Loop @ PLB 100 MHz:
  Power - 226.028000 mWatt over 8.177612 secs
  => Energy 0.513436 mWatt-hours

Thus if we could detect programs with high memory reference rate and operate them with high PLB

frequency while scaling down the PLB frequency for programs with less or no memory reference could result in energy saving of nearly 40%. Note that this energy savings was achived by scaling d FSB frequency from 100MHz to 50Mhz. Other scaling frequencies could produce different energy savings.

2.2 PART II

It is important to note that the results explained in section 2.1 were without the use of any type of feedbach DVS algorithms. That is, there was no dynamic reselection of operating points as would happen in case feedback DVS schemes are used. This section explains the results obtained when memory scaling was incorporated into the feedback scheduling algorithms.

The feedback DVS-EDF algorithms start with an operating point specified by us. Then after, in response to PID feedback, and in an attempt to dynamically scale processor frequency choses various other operating points during the course of execution. Code walkthrough of the PID schedular showed that PID would always select one of the 5 below listed frequencies(shown in table4),

```
{ "266/133/16",    {1700, 1, 16,  2, 2, 32, 32, 4, 4,  6, 0, 0, -1, -1} },
{ "133/133/16",    {1300, 1, 16,  4, 1, 32, 32, 4, 4,  6, 0, 0, -1, -1} },
 { "66/66/16",     {1100, 1, 16,  8, 1, 32, 32, 2, 2,  3, 0, 0, -1, -1} },
 { "44/44/16",     {1000, 1, 16, 12, 1, 32, 32, 1, 1,  2, 0, 0, -1, -1} },
 { "33/33/16",     {1000, 1, 16, 16, 1, 32, 32, 1, 1,  2, 0, 0, -1, -1} },
```

Table 4: Originally available operating points

The sample applications executed in the body of the threads scheduled by the PID feedback scheduler would be executed in any of the operating points defined in Table4. One other parameter seemed to affect the power readings was the 'max_loop' defined in example.c. So we consider it along with the operating points and all subsequent results would be discussed in the context of the operating points and 'max_loop'.

In an attempt to define lower PLB frequecy operating points we edited the PLB divider parameter from the original operating points as shown in Table5 below,

```
{ "266/133/16",    {1700, 1, 16,  4, 4, 32, 32, 4, 4,  6, 0, 0, -1, -1} },
{ "133/133/16",    {1300, 1, 16,  4, 4, 32, 32, 4, 4,  6, 0, 0, -1, -1} },
 { "66/66/16",     {1100, 1, 16,  8, 2, 32, 32, 2, 2,  3, 0, 0, -1, -1} },
 { "44/44/16",     {1000, 1, 16, 12, 1, 32, 32, 1, 1,  2, 0, 0, -1, -1} },
 { "33/33/16",     {1000, 1, 16, 16, 1, 32, 32, 1, 1,  2, 0, 0, -1, -1} },
```

Table 5: Edited operating points. Note the values in bold.

2.2.1 RESULTS

All experiments were repeated 10 times and the values shown in the section are the averaged out values over the 10 interations.

- ✔ For noop application with max_loop = 1000 and operating points(original) as defined in Table 4: **Memory sub-system energy = 1.5413399 mWatts-Hours**

- ✔ For noop application with max_loop = 1000 and operating points as defined in Table5. **Memory sub-system energy = 1.519984 mWatts-Hours**

- ✔ For a memory intensive application(see section 2.3) with max_loop = 1000 and operating

points(Original) as defined in Table4: **Memory system energy = 1.5459353 mWatts-Hours**

- ✔ For noop application with max_loop = 5000 and original operating points as defined in Table4: **Memory sub-system energy = 1.691270 mWatt-Hours**

- ✔ For noop application with max_loop = 5000 and operating points as defined in Table5. **Memory sub-system energy = 1.343177**

When max_loop(see sec 2.3) is 1000, our PLB frequency-scaled operating points (Table5) produced energy saving of 1.38% over the unscaled original operating points(Table 4). With max_loop increased to 5000, the energy savings jumped up to 12.17%.

In conclusion, we could see that considerable energy savings could be achived by scaling down the FSB frequency for tasks with no or less memory references. But integration of the FSB scaling into the existing feedback-DVS sceduling mechanisms could be challenging as discussed in section 2.4.

2.3 EXPLANATION ON CHANGES TO PROGRAMS

As we cannot decide on the memory profile of an application/thread, we get this input from the user, that is whether the test load is memory-intensive or CPU-intensive. Following modifications were made to incorporate it and to decide on a appropriate operating point based on it.

In example.c, we have added two test cases one with memory-intense file read operation and another with a dummy loop which is CPU-intensive. We also added one more user parameter for getting the type of the load. This is then given to the thread package.

In my_threads.c, we modified the logic to select the appropriate operating points - we included the support to scale down the PLB frequency based on the test load input from the user.

In dpm405lp.c, we have added new operating points with reduced PLB settings.

In avg.c, changes were made to calculate the energy for the memory subsystem.

In beech.c, changes were made to read the channels 3 and 4.

2.4 OPEN ISSUES

* PID feed back scheduling algorithms as explained in [2] and [3] make complex dynamic decisions on what operating points have to be used at various points duting a task's execution. In order for the memory scaling to be integrated into a similar dynamic scaling algorithm we might have to do a through analysis of the memory reference patterns of each task in the system.
* It is not clear how FSB scaling would affect the execution time of the task which would inturn influence the PID feedbacks and hence the choice of the operating points.

REFERENCES

[1] “A survey of design techniques for system-level dynamic power management” Benini, L.; Bogliolo, A.; De Micheli, G. Very Large Scale Integration (VLSI) Systems, IEEE Transactions on Volume 8, Issue 3, Jun 2000 Page(s):299 - 316
[2] “Feedback EDF Scheduling of Real-Time Tasks Exploiting Dynamic Voltage Scaling" by Y. Zhu

and F. Mueller in Real-Time Systems Journal, Vol. 31, No. 1-3, Dec 2005, pages 33-63
[3] Exploiting Synchronous and Asynchronous DVS for Feedback EDF Scheduling on an Embedded Platform by Y. Zhu and F. Mueller in ACM Transactions on Embedded Computing Systems, Vol. 7, No. 1, Dec 2007, pages 1-26.
[4] www.research.ibm.com/arl/publications/papers/DPM_V1.1.pdf
[5] “System level power-performance trade-offs in embedded systems using voltage and frequency scaling of off-chip buses and memory(2002)”. by Kiran Puttaswamy, Kyu-won Choi, Jun Cheol Park, Vincent J. Mooney Iii, Abhijit Chatterjee, Peeter Ellervee. In Proceedings of International Symposium on System Synthesis (ISSS’02)
[6] PowerPC 405 User Manual