International Journal of Engineering, Pure and Applied Sciences,
Vol. 1, No. 1, 2016

Embedded Systems Dilemma of Chip Memory Diversity by Scratchpad Memory for Cache On-chip Memory

Ravi Kumar Sharma
Research Scholar, Department of Computer Application, Surya World University, Rajpura, India
Email: ravirasotra@yahoo.com

Abstract: -The aim of this paper is the problem of chip memory selection for computationally intensive applications by proposing scratch pad memory as an alternative to cache. Area and energy for different scratch pad and cache size are computed using CACTI tools while performance was evaluated using the trace results. In this paper, an algorithm integrated into a compiler is presented which analyses the application and selects program and data parts which are placed into the scratchpad. Comparisons against a cache solution show remarkable advantages between 12% and 43% in energy consumption for designs of the same memory size.

Key Words-Scratchpad Memory, Cache Memory, CACTI, Compiler, Memory.

1. INTRODUCTION

The salient feature of portable devices is light weight and low power consumption. Applications in multimedia, video processing, speech processing[1], DSP applications and wireless communication require efficient memory design since on chip memory occupies more than 50% of the total chip area [2]. This will typically reduce the energy consumption of the memory unit, because less area implies reduction in the total switched capacitance. On chip caches using static RAM consume power in the range of 25% to 45% of the total chip power [3]. Recently, interest has been focused on having on chip scratch pad memory to reduce the power and improve performance. On the other hand, they can replace caches only if they are supported by an effective compiler. Current embedded processors particularly in the area of multimedia applications and graphiccontrollers have on-chip scratch pad memories. In cache memory systems, the mapping of program elements is done during runtime, whereas in scratch pad memory systems this is done either by the user or automatically by the compiler using suitable algorithm. Although prior studies into scratch pad memory behavior for embedded systems have been conducted, the impact on area has not been addressed. This paper compares cache/scratch pad area models along with their energy models. Specifically we address the following issues. To support comparison of memory systems we generate area models for different cache and scratchpad memory. Further, energy consumed per access for cache and scratchpad is computed for different sizes of cache and scratchpad. We develop a systematic framework to evaluate the area-performance tradeoff of cache/scratch pad based systems.

Experimental environment requires the use of a packing algorithm (which is a compiler support) to map the elements onto the scratchpad memory. Finally, we report the performance and energy consumption for different cache and scratchpad sizes, for the various applications. We include the main memory energy consumption to study the complete system energy requirements. The rest of the paper is organized as follows. In section Awe explain the scratch pad memory area and energy models. In section B, we present cache memory used in our work. Section C describes the methodology and the experimental setup and section VI contains the results. In section VII we conclude and also specify the future work.

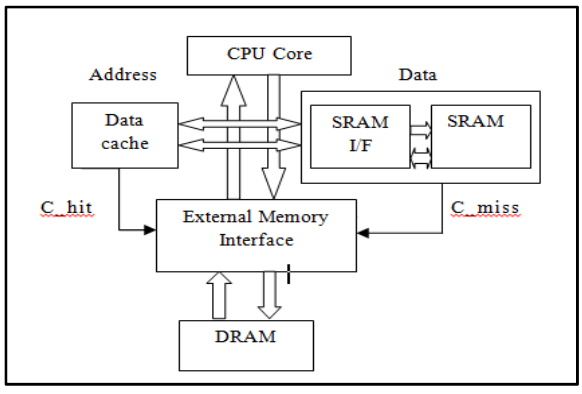

Fig 1: Block Diagram of Embedded Processor Application

2. SCRACTH AND MEMORY

Scratch pad is a memory array with the decoding and the column circuitry logic. This model is

designed keeping in view that the memory objects axe mapped to the scratch pad in the last stage of the compiler. The assumption here is that the scratch pad memory occupies one distinct part of the memory address space with the rest of the space occupied by main memory. Thus, we need not check for the availability of the data/instruction in the scratch pad. It reduces the comparator and the signal miss/hit acknowledging circuitry. This contributes to the energy as well as area reduction. The cell has one R/W port. Each cell has two bit-lines, bit and bit bar, and one word-line. the area of the scratch pad is the sum of the area occupied by the decoder, data array and the column circuit. The scratch pad memory energy consumption can be estimated from the energy consumption of its components i.e.decoder and memory columns.Energy in the memory array consists of the energy consumed in the sense amplifiers, column multiplexers, the output driver circuitry, and the memory cells due to the word line, pre-charge circuit and the bit line circuitry. The major energy consumption is due to the memory array unit. The procedure followed in the CACTI tool to estimate the energy consumption is to first compute the capacitances for each unit. Then, energy is estimated. As an example we only describe the energy computation for the memory array. Similar analysis is performed for the decoder circuitry also, taking into account the various switching activity at the inputs of each stage.In the preparation for an access, bit-lines are pre-charged and during actual read/write, one side of the bit lines are pulled down. Energy is therefore dissipated in the bit-lines due to pre-charging and the read/write access. When the scratch pad memory is accessed, the address decoder first decodes the address bits to find the desired row. The transition in the address bits causes the charging and discharging of capacitances in the decoder path. This brings about energy dissipation in the decoder path. The transition in the last stage, which is the word-line driver stage triggers the switching in the word-line. Regardless of how many address bits change, only two word-lines among all will be switched. One will be logic 0 and the other will be logic 1.

3. CACHE MEMORY

Cache memory is random access memory that a computer microprocessor can access more quickly than it can access regular RAM[6].As the microprocessor processes data, it looks first in the cache memory and if it finds the data there (from a previous reading of data), it does not have to do the more time-consuming reading of data from larger memory. Cache memory is sometimes described in levels of closeness and accessibility to the microprocessor. In addition to cache memory, one can think of RAM itself as a cache of memory for hard disk storage since all of RAM's contents come from the hard disk initially when you turn your computer on and load the operating system (you are loading it into RAM) and later as you start new applications and access new data. RAM can also contain a special area called a disk cache that contains the data most recently read in from the hard disk. Caches are mainly used to exploit the temporal and spatial locality of memory accesses. Area model that we use in our work is based on the transistor count in the circuitry. All transistor counts are computed from the designs of circuits.

4. SCRACTHPAD MEMORY PRAPHERNALIA ANDS METHODS

There are different type's scratchpad memory paraphernalia and method like static memory and dynamic memory. Static memory location don's change at runtime and dynamic memory locations change at runtime. Find a technique for efficiently exploiting on-chip SPM by partitioning the application's scalar and array variables into off-chip DRAM and on-chip SPM Minimize the total execution time of the application. The method is to Use profiling to estimate reuse, Copy variables in to SRAM when reused, Cost model ensures that benefit exceeds cost, Transfers data between the on chip and off chip memory under compiler supervision, Compiler-known data allocation at each point in the code.

4.1. Scratch pad memory accesses

From the trace file, it is possible to do the performance estimation. As the scratch pad is assumed to use part of the total memory address space, from the address values obtained by the trace analyzer, the access is classified as going to scratch pad or memory and an appropriate latency or check points are added to the overall program delay. One cycle is assumed if it is a scratch pad read or writes access. If it is a main memory 16 bit access then we take it as one cycle plus 1 wait state (refer to Table 1). If it is a main memory 32 bit access then, we consider it as one cycle plus 3 wait states. The total time in number of clock cycles taken is used to conclude the performance.

4.2. Cache memory accesses

From the trace file it is possible and easy to obtain the number cache *read* hits, *read* misses, *write* hits and *write* misses.

Table 1: Memory access cycles

Access	Number of Cycles
Cache	Using Table 2
Scratch Pad	1 Cycle
Main Memory 16 bit	1 Cycle + 1 wait state
Main Memory 32 bit	1 Cycle +3 wait state

From this data, we figure out the number of accesses to cache based on Table 2, where the number of cycles required for each type of access is given in Table 1. The cache is a write through cache. There are four cases of cache access that we consider in our model.

1) **Cache read hit**: When the CPU requires some data, the tag array of the cache is accessed. If there is a cache read hit, then data is read from the cache. No write to the cache is done, and main memory is not accessed for a read or writes.
2) **Cache read miss**: When there is a cache read miss, it implies that the data is not in the cache, and the Line has to be brought from main memory to cache. In this case we have a cache read operation, followed by L words to be written in the cache, where L denotes the line size. Hence, there will be a main memory read event of size L with no main memory write.
3) **Cache write hit:** If there is a cache write hit, we have a cache write, followed by a main memory write.
4) **Cache write miss:** In case of a cache write miss, a cache tag read (to establish the miss) is followed by the main memory writes. There is no cache update in this case.

Table 2: Cache memory interaction model

Access Type	Ca_{read}	Ca_{write}	Mm_{read}	Mm_{write}
Read hit	1	0	0	0
Read miss	1	L	L	0
Write hit	0	1	0	1
Write miss	1	0	0	1

4.3. METHODOLOGY

Clock cycle estimation is based on the ARMulator trace output for cache or scratch pad memory. This is understood to directly focus the performance which means the larger the number of clock cycles the lower the performance. The theory that the change in the on-chip memory configuration (cache/scratch pad memory and its size) does not change the clock period. This hypothesis though restrictive doesn't affect our results. Because we always compare the same size cache with scratch pad memory and the delay of cache implemented with the same technology will always be higher as compare to others. Thus, the performance improvement predicted for scratch pad can only increase if both effect the clock period. The identification and assignment of critical data structures to scratch pad was based on a packing algorithm.

5. RESULTS AND DISCCUSION

5.1. Permeation Test

The diffusivity and the permeation rate of the interstitial free steel at 30^0 C with a constant charging current density (10 mA/c^{m2}) are listed in Table 2. The hydrogen diffusivity of annealed interstitial free steel is lower than those of pure iron [4 – 6]. It is due to the hydrogen trap and causing titanium hydride formation matrix. The data also clearly show a decreased in D_{eff} but an increased in J ∞ L as cold-rolled percentage increased for annealed specimens. The value of D_{eff} is decreased with increasing cold work, is due to the more hydrogen trapping site resulting from dislocations and deformation- induced micro voids [9-10] .Cold work increases the J ∞ L , this effect has been explained by short-circuit diffusion paths down dis- locations networks as well a s by low energy trapping of hydrogen to dislocation [7-8] .

5.2. Tensile Testing

Tensile proper ties of hydrogen charged and uncharged specimens are listed in Table 3. The tensile data show a slight loss in mechanical proper ties with a 5-day hydrogen charging for all cold-rolled specimens, but slight improvement for annealed specimen. For the un charged specimen s, it is mainly simple ductile fracture, even the 80% cold- rolled specimen as show n in Fig. 2, while the factor graph shows a trangronular cleavage effect with a partial ductile fracture surface for a 5-day hydrogen charged. This results can be explained as more hydrogen trapping site in cold worked specimen with higher dislocation density. The hydrogen precharged annealed speci- means show slight improvement in strength and elongation can be explained as the tiny titanium hydride formation in the matrix, providing the easy glide of dislocation, and the precipitates also enhancing strength.

6. EXPERIMENTAL SETUP AND FLOW DIAGRAM

Fig. 2 shows the flow diagram. The energy aware (encc) compiler [5] generates the code for the ARM7 core. It is a research compiler used for exploring the design and new optimization techniques. The input to this compiler is an application benchmark written in C. As a post pass option, encc uses a special packing algorithm, known as the knapsack algorithm [4], for assigning code and data blocks to the scratch pad memory.

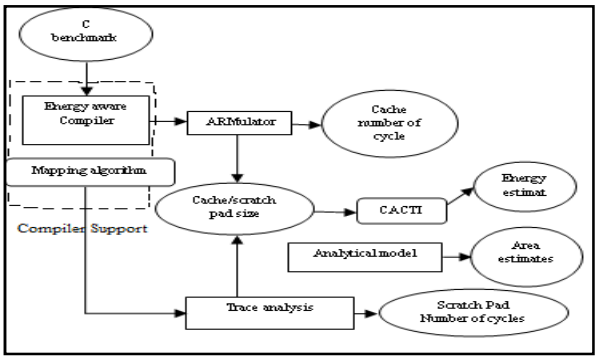

Fig. 2 Experimental flow diagram

This algorithm identifies the frequently referred data and instruction blocks and maps to the scratch pad memory address space. The cost of additional jumps introduced due to mapping consecutive blocks to scratch pad and main memory is accounted for by the algorithm. The result is that blocks of instructions and data which are frequently accessed, and axe likely to generate maximum energy savings, axe assigned to the scratch pad. The output of the compiler is a binary ARM code which can be simulated by the ARMulator to produce a trace file. For on-chip cache configuration, the ARMulator accepts thecache size as parameter and generates the performance as the number of cycles.

7. RESULT AND DISCUSSION

To demonstrate the merits of using on-chip scratch pad memory and on-chip caches, we have conducted a series of experiments for both of these configurations. The trace analysis after the compilation phase. We use a 2-way set associative cache configuration for comparison. This area represents dynamic number of transistors. These consume the area from the cache and scratch pad organization, and obtain the results. The comparison of area of the cache and scratch pad memory for varying sizes. We find that on an average the area occupied by the scratch pad is less than the cache memory by 35%.Thus; we take the main memory energy, along with the on-chip memory energy consumption into account. The energy consumed for bi-quad, matrix-melt and quick-sort, which are examples for both cache and scratch pad. In all the cases we have analyzed, that scratch pad consumes less energy for the same size of cache, except for quick-sort with cache size of 256 bytes. On an average, we found energy consumption to be reduced by 42% using scratch pad memory.

8. CONCLUSION

In this paper, we have presented an approach for selection of on-chip memory configurations. The paper presents a comprehensive methodology for computing area, energy and performance for various sizes of cache and scratch pad memories. Results indicate that, scratch-pad based compile time memory outperform cache-based run-time memory on almost all counts. We observe that the area-time product (AT) can be reduced by 47% (average) by replacing cache by the scratch pad memory. We found that, for most applications and memory configurations, the total energy consumption of scratch pad based memory systems is less than that of cache-based systems. The average reduction was 42% in the application considered which better compare than other systems.

REFERENCES

[1] Rajeshwari Banakar, Stefan Steinke, Bo_Sik Lee, M. Balakrishnan, Peter Marwedel, Scratchpad Memory: A Design Alternative for Cache Onchip memory in Embedded Systems

[2] Doris Keitel-Sculz and Norbert Wehn., Embedded DRAM Development Technology, Physical Design, and Application Issues, IEEE -Design and Test of Computers, Volume 18 Number 3, Page 7 to15, May-June 2001.

[3] M.T. Milan, D. Spinelli, W.W. Bose Filho, Int. J. Fatigue 23 (2001) 129.

[4] T.Y. Zhang, Y.P. Zhang, Acta Mater. 46 (1998) 5023.

[5] D.L. Johnson, J.K. Wu, J. Mater. Energy Syst. 8 (1987) 402`

[6] R.D. McCright, “Stress Corrosion Cracking and Hydrogen Em-brittlement of Iron Base Alloys”, NACE, Houston, TX, 1977, p. 306

[7] http://searchstorage.techtarget.com/definition/cache-memory

[8] Http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.17.34 46