Applications such as mobile devices (cellular phones, PDA, consumer electronics) does require both a very powerful processor(s) as well as low power consumption because of the limited power supply on such a device (batteries). Saving power can be achieved on three different levels. First, it could be achieved through low power VLSI technology where the transistors consume less power for switching and maintaining state. Second, it could be achieved on the architecture level by designing processors, memory systems, and instruction set architecture with power saving in mind. Lastly, it could be achieved at the application level by using more efficient algorithms, or by compiling the code to run on either the main processor or some re-configurable logic on or off chip in order to save power. This paper presents a survey of low power processor design concentrating only on the architectural and software/reconfigurable logic only.

1. Introduction

Many applications require low power processors either because the devices are mobile (portable) and depend on a battery for its power supply such as mobile (cellular) phones, portable computers, personal digital assistants, or because it is difficult to reach and recharge its power supply such as electronics in satellites, or sensor networks where it is not easy to reach.

Saving power during the operation of the electronic device could be achieved on more than one level. First on the circuit or VLSI level power could be saved by using less power for state transition (capacitor charging and discharging), and state maintenance.

On the architecture level, power could be saved by the proper implementation of the processor, the cache, and the instruction set. A study on the StrongARM110 processor reveals that the power dissipation in the instruction cache, data cache, and TLB accounts for almost 60% of the power consumption of the processor [5] That means there is a room for power saving by the proper implementation of the memory system.

Power could be saved even on a higher level than architecture by code transformation on the software level, and by executing parts of the code on a separate processor or reconfigurable logic, or by using more efficient algorithms.

In this paper, we survey recent advances of low power processor design, we present examples of power saving on the architectural and software levels, mentioning only the VLSI level when it is combined with the abovementioned levels.

2. Architecture Level

A video codec was proposed in [12] that uses a combination of a low power general purpose DSP, a fast algorithm, and a low complexity noise reduction filter in order to produce a low power H.263 codec.

For the processor, they choose the NEC µPD7701x that uses 16 bit fixed point calculations with a 40 bit ALU. The processor use 3V and consumes 100mW at 50MIPS. The processor executes all the instruction except branching in one cycle.

They also used a faster algorithm for the Motion estimation. The block matching using mean square error between the current block and a reference block is as follows

$$error_{u,v} = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (x_{i,j} - y_{i+u,j+v})^2$$

Where x represents the pixel of the current block, y represents the pixels of the reference block, u, v represents the relative position of the current block with respect to the reference block, an N is the block size of the current block.
size. The squared term consists of three sub terms; $x^2$, $y^2$, and $xy$. The first term, $x^2$ represents the power in the current block. That term is constant for each reference block, and does not contribute anything in finding the minimum and can be ignored. The last term; $xy$ is a 2-D FIR filtering operation. However, because the current block is down sampled, the effect of the 2-D FIR is negligible and this term will be also ignored. The term $y^2$ is the power in the reference block. Since the reference block is displaced by $u,v$ from the current block, there is a lot of overlapping between the references blocks and that could be used to minimize the number of operations performed in the motion estimation part.

They also proposed combining the temporal IIR and the spatial FIR filters that are used to reduce noise in the input frame into a single operation that is performed after the DCT and before quantization. Thus changing convolution into multiplication in the frequency domain and saving operations which is translated into power saving.

In [9] the authors proposed a low power fast inner product processor for 3-D volume rendering applications. The input to their processor is 2 8-bit Cartesian vectors and the output is a one 16-bit signed scalar number. The processor uses modified Booth’s algorithm for fixed point 2-operand multiplication. The Booth’s algorithm reduces the number of partial products almost by half, saving power in the process. They also used a sign extended simplification in order to reduce the number of bits required for each partial product. Finally they used a 4-2 compressor instead of a full adder for the design of Wallace tree in order to produce a Wallace tree with less routing length and highly regular layout.

The processor consists of an initial Booth encoder stage, followed by partial product generation and then a Wallace tree; finally a 2-operand adder is used to calculate the final product.

The processor was designed and manufactured using Complementary Pass Transistor (CPL) logic with PMOS cross-coupled pull-up devices [20]. The authors compared their design, that uses half-swing and full-swing voltage restoration with a design that uses only full-swing voltage restoration after every CPL nodes. For voltages between 1.6v to 2.0v and frequency of 50MHz to 200 MHz. Their design was slower by 8% and the power reduction was 14-16% resulting even in a bigger saving of power delay product.

In [10] the authors proposed a low power correlator array for de-spreading the received signal in a CDMA system in order to receive the transmitted data. They achieved low power by using two different techniques.

First, by rearranging the elements of the different codes as a matrix, they can reduce the number of additions required to re-spread the signal. Second, by noticing that the major reason for power consumption in CMOS circuits is due to signal transition and capacitor charging/discharging. One way to reduce this is to use sign magnitude multiplication instead of 2’s complement. Since in de-spreading, we are multiplying by either 1 or –1, that means only changing the sign bit. Their design resulted in 40% power saving compared to traditional correlator design.

Veljanovski at al in [19] proposed a low power reconfigurable pulse-shaping filter. Their design was targeted towards UMTS terrestrial radio access systems. The objective is to eliminate the adjacent cell interference either by a mobile node in the neighboring cell, or by the base station in the neighboring cell.

Their design is based on the idea of a variable length filter. The filter is capable of increasing or decreasing its length (number of coefficients of the FIR filter used in the computations) based on the level of interference. Thus the power consumption is proportional to the interference between the cells.

They simulated their design in order to estimate the power saving. Using a DSP core controller with advanced power management, and CMOS Alcatel Microelectronics 0.35 micron digital libraries, an average saving of 75% was achieved.

In [8] the authors proposed a cache organization for embedded low power processors. They proposed a small cache in addition to the L1 cache. While a small cache is usually added between L1 cache and the CPU (which result in a longer memory hierarchy and deteriorates the performance), they proposed that the new small cache (Enhanced mini cache, or EM cache) to be at the same level as L1.

The basic idea is the following: If we can make most of the accesses to the much smaller EM cache instead of the L1 cache, that results in a power saving (assuming that the L1 cache will be disabled while accessing EM cache). Since mot of the access is to loops (sequential), they proposed two techniques to disable L1 cache. The first is a hardware-based and is activated when the next instruction is in the EM cache, it sends a signal to deactivate the L1 cache. They also proposed a software based approach by using a special instruction that is inserted by the compiler to deal with access to multiple blocks in the EM cache.

They reported a power saving between 2% to 49% across SPEC95 with average of 36% power saving.

In [5] The authors proposed a new technique to reduce power consumption in the Table Look-aside buffer (TLB). Their technique depends on a 2-way banked filter with a 2-way banked TLB. When a memory address is generated by the CPU, one of the 2 banked filters is searched first, only in case of a miss, we go to the one of the 2 banks of the main TLB. Their results show an improvement of 59% in the miss ratio, and 25% in the Energy delay product.
In [24] the authors proposed a variable line size cache for embedded systems. They showed that by adjusting the cache line size to the application, we can reduce the cache access energy by up to 50%. They also proposed an architecture for a variable line size cache.

3. Algorithm and Software Level

In [22] the authors investigated the design of a low power video processor. They investigated many VLSI and architectural level techniques for power saving (asynchronous design, clock gating, switching time acceleration, and bus switching reduction), which is according to their analysis did not produce any power saving. The promising technique for their design is an algorithmic level technique. They exploited the spatial correlation between the pixels, and used the difference between two pixels instead of the pixel value. They changed the algorithm accordingly however they kept the original design to deal with edges in the image (no spatial correlation at edges). Their design resulted in power saving of up to 15%.

In [18] the authors proposed a high performance and low power video compression techniques. They used a combination of efficient algorithms and new VLSI implementation techniques in order to produce a low-power design.

First, at the algorithm level, in order to convert between YUV and RGB representation of the image [7] they used a fixed point instead of floating point coefficients, thus although not reducing the total number of multiplication, they reduced the power consumption and increased the speed by using fixed point multiplication instead of floating point multiplication. Second, they modified the factorized DCT proposed in [11] and [4] to produce a more regular architecture. They also proposed a multiplier-less version that replaces multipliers by adders and shifters that increased the speed of execution and consumed less power.

On the VLSI level, they used a simplified version of the method proposed in [15] that depends on more than one supply voltage to the chip (5, 3.3, 2.5, and 1.8volts) and they supplied different voltage levels to different nodes. The nodes on the critical path was supplied by a large supply voltage in order to speedup the switching and thus the execution. While the nodes that are not on the critical path was supplied by a less voltage level. That resulted in low-power design without sacrificing the speed.

The authors in [13] proposed a low power sigma delta decimation filter. The authors used a combination of techniques in order to reduce power consumption. First, they used half-band filters and they used canonic signed digit number with shifters and adders instead of multipliers, which added to the power saving of the filter. They implemented their design using Xilinx FPGA 4000 technology and resulted in more the 65% power saving.

The authors in [14] proposed a power consumption frame work for embedded processors. The input to this framework is the program (either assembly or machine level) and a specific processing system. The output is the energy consumption in the processor. This system could be used to estimate the energy consumption of embedded processors.

4. Reconfigurable Logic

Processors with reconfigurable logic on the same chip are getting increasingly popular [1, 17, 21]. Usually these reconfigurable logic are either used to implement coprocessors, or peripherals. However it could be used to save chip power consumption.

In [16] the authors considered an embedded system running the same program. They used profiling in order to map some loops to be executed on the reconfigurable logic while the main processor is in a low power state. They also considered that we can use the faster task completion time (on a reconfigurable logic) in order to use a lower voltage resulting in the same execution time without reconfigurable logic, but much less power consumption. Their results indicate a power saving of up to 70% on 8-bit microprocessor and up to 53% on a 32-bit microprocessor.

A low power reconfigurable processor is being developed in [6] Their system depends on an ultra-low power radiation tolerant CMOS logic family [2]. Their design consists of 16 processing elements (PE’s) connected through a reconfigurable interconnection networks. Each PE consists of a multiplier, general purpose ALU, a conditional multiplexer, data path formatting elements, and two data registers. A micro-sequencer is used to generate control sequences to run a simple program.

In [3] the authors proposed mapping the instructions into binary values of different lengths using an instruction remap table. This remapping can be achieved either statically or dynamically.

In static configuration, each application is profiled to get the instruction usage information, and the most widely used instructions are mapped into a shorter length codes.

For dynamic configuration, the instruction usage may be different from a segment to another segment in the same program. In this case, instructions are remapped on the fly in order to minimize the accessed code size.

Their results show a code reduction of 10% and a saving in the instruction fetch energy of 60%

A power configurable bus for embedded systems was proposed in [23]. The proposed bus provide a tradeoff between power saving and performance. They used bus invert encoding, and address encoding. Their results show address bus power saving of 59%-97% and data bus power saving of 22% to 63%.
5. Conclusion

Power reduction in processor design could be achieved through, circuit design, architecture, algorithms, and reconfigurability. In this paper we surveyed low power processors suitable for embedded systems. And we showed examples for existing systems on the architectural, algorithm and corfigurability levels.

References

7. M. Ghanbari “Video coding an introduction to standards codes. IEE Telecommunication series 42