THE
COMPUTER
ENGINEERING
HANDBOOK

Edited by
VOJIN G. OKLOBDZIJA

CRC
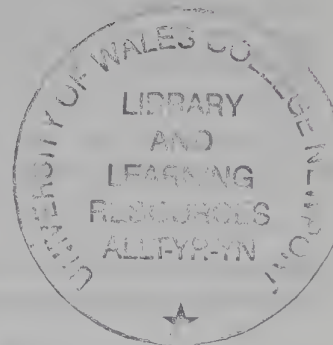
CRC PRESS

Boca Raton   London   New York   Washington, D.C.

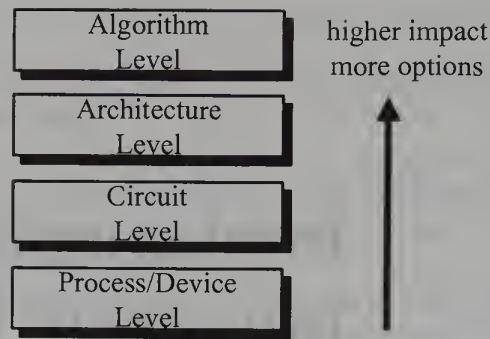**Visit the CRC Press Web site at www.crcpress.com**

FIGURE 19.1   Each level impact for low-power design.

A typical example of algorithm contribution is motion estimation of MPEG encoder. Motion estimation is an extremely critical function of MPEG encoding. Implementing fundamental MPEG2 motion estimation using a full search block matching algorithm requires huge computations [3,4]. It reaches 4.5 teraoperations per second (TOPS) if realizing a very wide search range (±288 pixels horizontal and ±96 pixels vertical), on the other hand the rest of the functions take about 2 GOPS. Therefore motion estimation is the key problem to solve in designing a single chip MPEG2 encoder LSI. Reference [5] describes a good example to dramatically reduce actual required performance for motion estimation with a very wide search range, which was implemented as part of a 1.2 W single chip MPEG2 MP@ML video encoder. Two adaptive algorithms are applied. One is 8:1 adaptive subsampling algorithm that adaptively selects subsampled pixel locations using characteristics of maximum and minimum values instead of fixed subsampled pixel locations. This algorithm effectively chooses sampled pixels and reduces the computation requirements by seven-eighths. Another is an adaptive search area control algorithm, which has two independent search areas with H: ±32 and V: ±16 pixels in full search block matching algorithm for each. The center locations of these search areas are decided based on a distribution history of the motion vectors and this algorithm substantially expands the search area up to H: ±288 and V: ±96 pixels. Therefore, the total computation requirement is reduced from 4.5 TOPS to 20 GOPS (216:1), which is possible to implement on a single chip. The first search area can follow a focused object close to the center of the camera finder with small motion. The second one can cope with a background object with large motion in camera panning. This adaptive algorithm attains high picture quality with very wide search range because it can efficiently grasp moving objects, that is, get correct motion vectors. As shown in this example, algorithm improvement can drastically reduce computation requirement and enable low power design.

19.4   Architecture Level Impact

The architecture level is the next to the algorithm level, also in terms of impact on power consumption. At the architecture level there are still many options and wide freedom in implementation. The architecture level is explained as CPU (microprocessor), DSP (digital signal processor), ASIC (dedicated hardwired logic), reconfigurable logic, and special purpose DSP.

The CPU is the most widely used general-purpose architecture as shown in Fig. 19.2. Fundamentally anything can be performed by software. It is the most inefficient in power, however. The main features of the CPU are the following: (1) It is completely sequential in operation with instruction fetch and decode in every cycle. Basically this is not essential for computation itself and is just overhead. (2) There is no dedicated address generator for memory access. The regular ALU is used to calculate memory address. Throughput of data feeding is not, every cycle, based on load/store architecture via registers (RISC-based architecture). This means cycles are consumed for data movement and not just for computation itself. (CISC allows memory access operation, but this doesn't mean it is more effective; it is a different story, not explained in detail here.) (3) Many temporal storage operations are included in computation procedure.
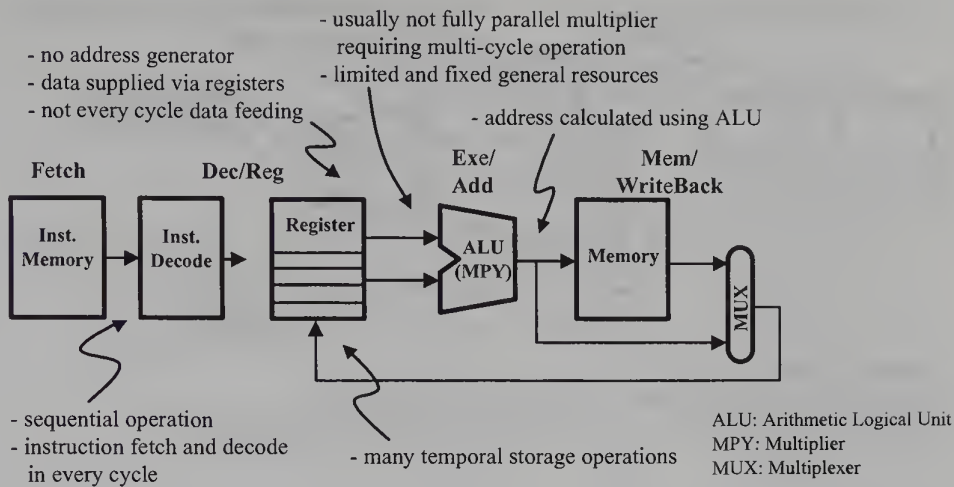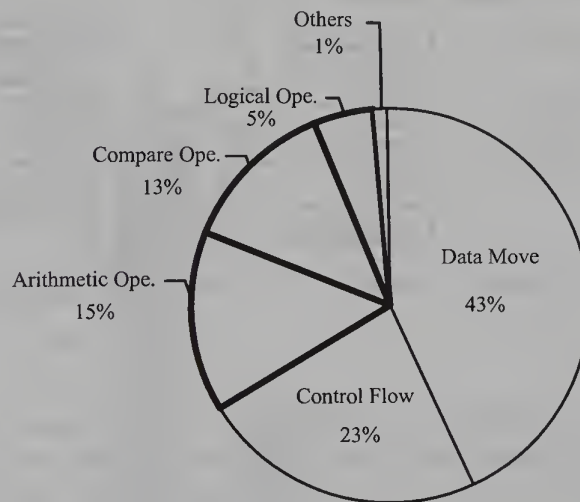
FIGURE 19.2   CPU structure.

FIGURE 19.3   Dynamic instruction statistics.

This is a completely justified overhead. (4) Usually, a fully parallel multiplier is not used, causing multi-cycle operation. This also consumes more wasted power because clocking, memory, and extra circuits are activated in multiple for one multiply operation. (5) Resources are limited and prefixed. This results in overhead operations to be executed as general purpose. Figure 19.3 shows dynamic run time instruction statistics [6]. This indicates that essential computation instructions such as arithmetic operation occupy just 33% of the entire dynamic run time instruction stream. The data moving and control-like branches take two-thirds, which is large overhead consuming extra power.

The DSP is an enhanced processor for multiply-accumulate computation. It is general-purpose in structure and more effective for signal processing than the CPU. But still it is not very power efficient. Figure 19.4 shows the basic structure and its features are as follows. (1) The DSP is also sequential in operation with instruction fetch and decode in every cycle similar to the CPU. It causes overhead in the same way, but as an exception DSP has a hardware loop, which eliminates continued instruction fetch in repeated operations, improving power penalty. (2) Many temporal storage operations are also used. (3) Resources are limited and prefixed for general purpose as well. This is a major reason for causing temporal storage operations. (4) Fully parallel multiplier is used making one cycle operation possible. And also accumulator with guardbits is applied, which is very important to accumulate continuously without accuracy degradation and undesired temporal storing to registers. This improves power efficiency for multiply-accumulate-based
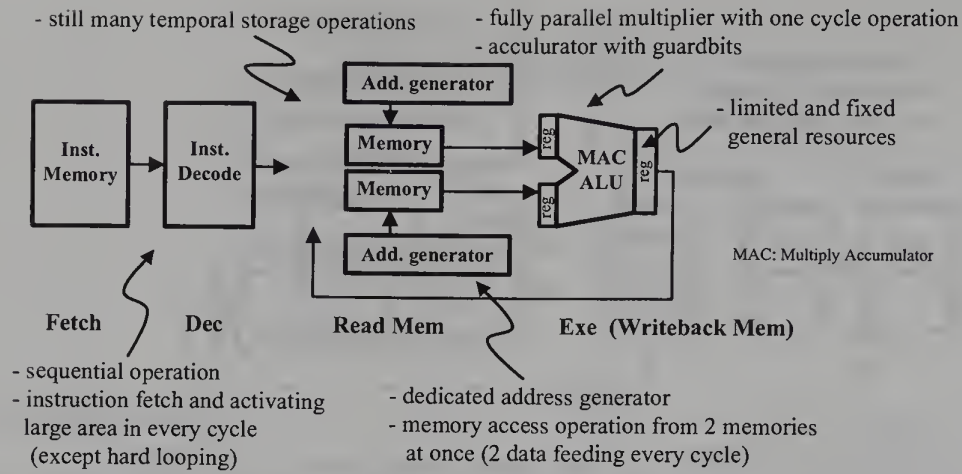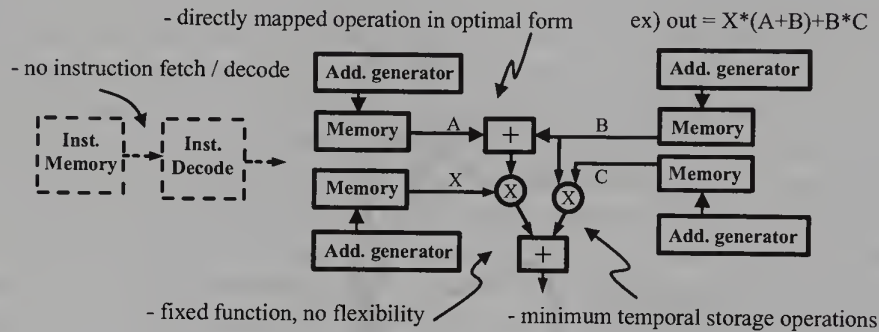
FIGURE 19.4   DSP structure.

FIGURE 19.5   ASIC structure.

computations. (5) It is equipped with dedicated address generators for memory access. This realizes more complex memory addressing without using regular ALU and consuming extra cycles, and two data can be fed in every cycle directory from memory. This is very important for DSP operation. Features (4) and (5) are advantages of the DSP in improving power efficiency over the CPU.

We define the ASIC as dedicated hardware here. It is the most power efficient because the structure can be designed for the specific function and optimized. Figure 19.5 shows the basic image and the features are as follows: (1) Required functions can be directly mapped in optimal form. This is the essential feature and source of power efficiency by minimizing any overheads. (2) Temporal storage operation can be minimized, which is large overhead in general purpose architectures. Basically this comes from feature (1). (3) It is not sequential in operation. Instruction fetch and decode are not required. This eliminates fundamental overhead of general-purpose processors. (4) Function is fixed as design. There is no flexibility. This is the most significant drawback of dedicated hardware solutions.

There is another category known as reconfigurable logic. Typical architecture is field programmable gate array (FPGA). This is gate level fine-grained programmable logic. It consists of programmable network structure and logic blocks that have a look-up table (LUT)-based programmable unit, flip-flop, and selectors as shown in Fig. 19.6. The features are: (1) It is quite flexible. Basically, the FPGA can be configured to any dedicated function if integrated gate capacity is enough to map it; (2) Structure can be optimized without being limited to prefixed data width and variation of function unit like a general 32-bit ALU of CPU. Therefore, FPGA is not used only for prototyping but also where high performance and high throughput are targeted. (3) It is very inefficient in power. Switch network for fine-grain level flexibility causes large power overhead. Each gate function is realized by LUT programed as truth table, for example NAND, NOR, and so on. Power consumption of interconnect takes 65% of the chip, while logic part consumed only 5% [7]. This means major power of FPGA is burned in unessential portion.

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.

fastcase
Smarter legal research.