

Optimized porting of PC algorithm on DSP

By implementing algorithm optimization techniques, eInfochips achieved 50 times performance improvement

By Upendra Patel, Manager Embedded Group, eInfochips Inc. www.einfochips.com

Most of the research is done on PC platform initially, as the algorithm research emphasis is more on achieving a particular functionality. Today's PC based systems have ample processing power and memory, so the algorithm developers don't have to worry about these two aspects. When the same algorithm is used for an embedded system, the resources become major bottleneck to achieve real time performance. Hence, the algorithms need to be optimized both for faster execution and lower memory consumption.

This case study discusses a few techniques adopted for improving the performance while porting the image-processing algorithm developed for PC platform to Texas Instrument DSP TMS320C6205. The algorithm identifies specific features of human face and display the modified face image on the LCD of C6205 based hand-held device.

The solution discussed in this case study was ported to DSP platform for our client IDEO, Inc.— a California based engineering company known for designing innovative products. “eInfochips’ accomplishment was impressive. They overcame a series of significant engineering challenges to deliver on time and under budget. Their attention to detail, creative thinking and professional attitude was evident throughout the project.” said Ed Kirk, Director of Software Development at IDEO, on successful completion of the project.

Challenge

Everyone assumes that algorithms written in ANSI C are easily portable across platforms. In this case also, the first attempt to port the algorithm was to recompile the code in Code Composer Studio and use it. But, when we tried to run it on the evaluation board, it was taking 5 seconds to execute against the target execution time of 100 mS. A large performance gap of 50 times, which was a deciding factor for the success of the product.

As usual, the first response was “it is impossible”! But, a more detailed analysis revealed that the seemingly impossible task is doable and the same team finally achieved it.

Porting And Optimization Methodology

We achieved this performance by using only ‘C’, without using DSP assembly coding.

The porting and optimization was divided in four stages:

C++ To C conversion (2 times improvement)

We converted C++ classes to their equivalent C structures. The methods within the C++ classes were converted to global functions. The C++ templates were degrading the performance and hence they were removed. Parameter passing in functions was minimized, resulting in stack size reduction from 48 KB to 3 KB. The program memory was also reduced from 139 KB to 70 KB.

Efficient use of DMA and memory (2 times improvement)

Pixel-by-pixel processing of the images stored in external memory was taking too much of time. The processing was done line by line instead of pixel by pixel and each line was moved to internal memory for processing using DMA.

Logic optimizations (4 times improvement)

The algorithm was written for PC platform, so more emphasis was on flexibility. We broke the algorithm in smaller units based on their execution time and rewrote the time consuming functions in an optimized way. This was an iterative process, but helped in achieving considerable optimization.

Floating-point to fixed-point operations (3 times improvement)

The floating-point operations are considerably slower than fixed-point ones and compiler cannot optimize “for/while” loops that contain floating-point operations. We replaced some of the division operations by shift operations. While converting from fixed-point to floating-point, we selected the appropriate fixed-point data types. The algorithm handled fixed-point decimal values to the power of 9.

Conclusion

By careful optimization, the algorithms developed for PC processors running at GHz speed, can be ported to lower speed DSP platforms and still achieve real-time performance.