An Optimized DSP Architecture for MEMS Gyroscope Platforms

Mostafa Sakr
Electronics and Communication Engineering Department
Ain Shams University
Cairo, Egypt
m_sakr@ieee.org

Amr Wassal
Computer Engineering Department
Cairo University
Cairo, Egypt
a.wassal@ieee.org

***Abstract*—This paper describes the design of an application specific instruction set processer digital signal processor (ASIP-DSP), for MEMS gyroscope platforms. By profiling the target application, and using the results for optimizing a reference DSP architecture, an optimized DSP architecture can achieve smaller area, and lower power consumption. The reference closed-loop control and signal processing implementation for MEMS gyroscope platform is profiled, and the results show three possible areas of improvement; reduction of the complexity of the RISC core, dedicated hardware for FIR operation, and supporting parallel processing to support multiple axis operation.**

I. INTRODUCTION

Using application-specific instruction set DSPs is more power and area efficient, as it optimizes the DSP architecture for the task at hand. These architectures remove seldom used instructions and features that are used in general purpose DSPs to support a broad range of applications. The design of the optimized ISA starts by defining the target application, then benchmarking this application on the reference architecture [1].

In our case, the target application is based on the closed-loop control and signal processing implementation proposed by [2] for the JPL-Boeing MEMS gyroscope. Fig. 1 shows the main components of the system. The drive loop is responsible for driving the MEMS sensor into resonance, and then it maintains the oscillations. The sense loop picks up the oscillations coupled from the drive axis to the sense axis by means of Coriolis force. These oscillations contain the rotation rate information modulated by the resonance frequency of the MEMS sensor. The demodulation block extracts the rotation rate information, and the optional decimation block, down samples the data to the application required data rate. A thorough description of the principles of the operation of MEMS gyroscopes can be found in [3].

II. APPLICATION PROFILING

The block diagram in Fig. 1 is translated into a C code model. The model is analyzed in two static analysis runs using the high-level C application and the assembly code

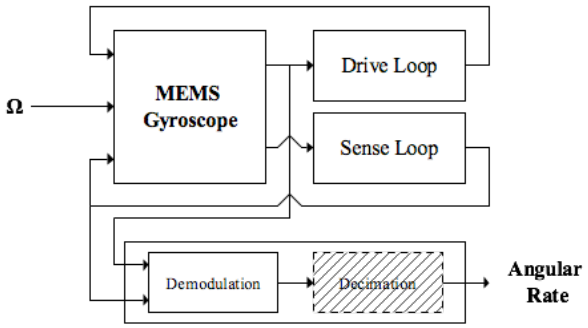

Figure 1. General block diagram of MEMS gyroscope closed-loop system

application. The results of the first run are presented in Fig. 2 and Fig. 3. The ratio of the number of operations required for each basic block, and implemented as a C function, relative to the total number of operations in the whole application is presented in Fig. 2. It is clear that the FIR block is the main contributor in this application. Fig. 3 shows the number of occurrences of each primitive operation in this application. It is evident that data movement operations dominate the execution time of the application. Then, to get the assembly program, the C files are compiled using C2000 compiler from TI, targeting the reference DSP TI C28x [4]. Fig. 4 shows the number of occurrences of the assembly instructions in the application. These results support the observations obtained from C code profiling.

The reference architecture contains 160 assembly instructions [5]. However, the benchmark application only uses 16 of these instructions, or 10% of the complete instruction set, as shown in Fig. 5.

III. PROPOSED ARCHITECTURE

To optimize the ISA, it is necessary to remove all unused instructions, while maintaining functional coverage. The results show that MOVL instruction is the most used one, and that around 55% of these instruction occurrences take place inside the FIR function since the filter requires frequent memory accesses to fetch the filter coefficients multiple times

978-1-4244-8274-0/10/$26.00 ©2010 IEEE

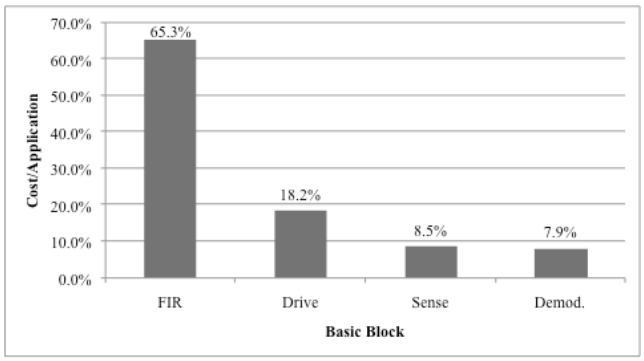

Figure 2. C-code profiling results, showing the contribution of the basic blocks

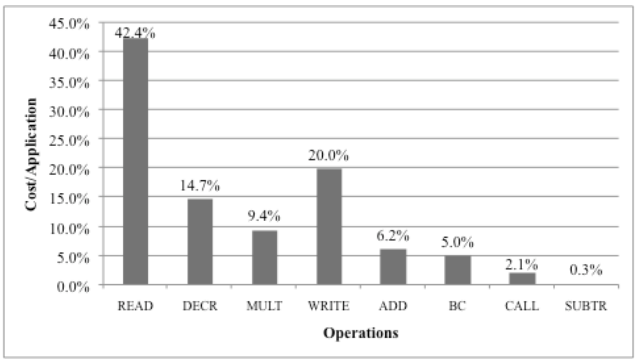

Figure 3. C-code profiling results showing the contribution of the different operations

with every function call. Consequently, the FIR function should be implemented using a special purpose FIR bank to minimize the need to access the memory.

A simplified block diagram of the proposed architecture is presented in Fig. 5. The architectural improvements can be summarized in the following two points:

- Introduce reconfigurable FIR banks to support the filtering functions required by the application.
- Implement a minimalistic RISC core to support the basic arithmetic and data transfer operations.

In order to support multiple axis gyroscopes, the proposed architecture should support vector operations, i.e., homogeneous SIMD architecture, to apply the DSP and control algorithm on data captured from different axis.

IV. CONCLUSIONS

The reduced number of instructions cuts the complexity and power consumption of the instruction decode and control unit hardware in turn. In a typical DSP, this accounts for 28% of the total power consumption [6]. Assuming that the size of control hardware will scale linearly with the instruction set reduction, the power saving while supporting only 50% of the reference ISA, is about 14% of the total power. Additionally, the introduction of parallel architectures enables a reduced operating frequency and a reduced supply voltage for the same workload, which would further reduce the consumed power.

REFERENCES

[1] D. Liu, "Embedded DSP Processor Design: Application Specific Instruction Set Processors," MKP 2008.

[2] Y.-C. Chen, R. M'Closkey, T. Tran, and B. Blaes, IEEE Trans. Cont. Sys. Tech., 13(2008), pp. 286–300.

[3] C. Acar and A. Shkel, "MEMS Vibratory Gyros: Structural Approaches to Improve Robustness," Springer 2008.

[4] Trusov, A.A., et al., Proc. Sensors 2007, pp.244-247.

[5] "TMS320C28x CPU and instruction set reference guide," Technical Report SPRU430E, TI, January 2009.

[6] Jeff Scott, et al, Proc. IEEE Power Driven Microarch. Workshop 1998, pp. 145-150.

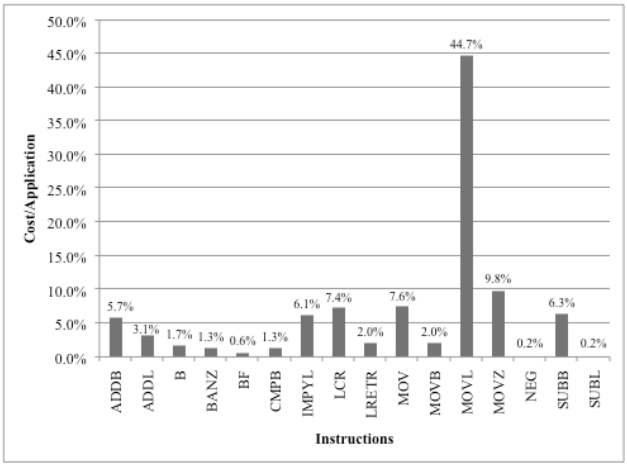

Figure 4. Assembly code profiling results, showing the contribution of each instruction

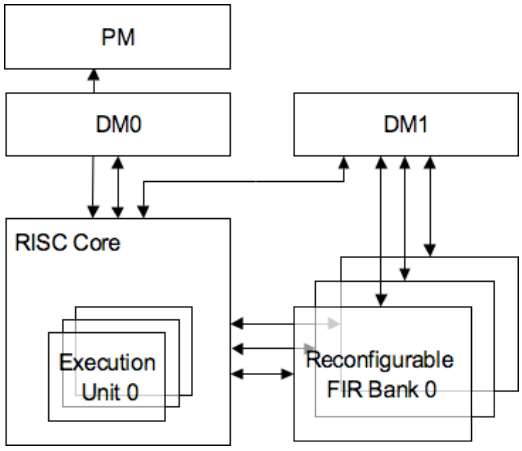

Figure 5. Proposed ASIP DSP architecture