International Conference on Control, Engineering & Information Technology (CEIT'13)
Proceedings Engineering & Technology - Vol.2, pp. 124-129, 2013
Copyright - IPCO

# DSC Performance Evaluation and Exploration Case of TMS320F28335

Imene MHADHBI#1, Nabil LITAYEM*2, Slim BEN OTHMEN#3, Slim BEN SAOUD#4

#*LSA Laboratory, INSAT-EPT, University of Carthage, TUNISIA*
1imene.mhadhbi@gmail.com
3slim.BenOthman@ept.rnu.tn
4slim.bensaoud@gmail.com
**Department of Computer Science, College of Arts & Science, Salman Bin Abdalaziz University, KSA*
2*nabil.litayam@gmail.com*

***Abstract*— The rapid advanced of microelectronics and semiconductor technologies has enabled to increase the capacity of digital circuits like Application-Specific Integrated Circuits (ASICs), microcontrollers (MCUs), Digital Signal Processors (DSPs) and Field Programmable Gate Array (FPGAs). Recently, a new digital circuits termed "Digital Signal Controller" (DSCs) has emerged. DSC combines the processing power of the DSP and the functionality of the MCU with several peripheral modules that make it an attractive proposition for practically all embedded systems applications, including communication, audio, medical, aerospace, defence and industrial control. The performance analysis of DSCs processors, presents one of the consideration metric in the choice the best processing element for a special application. In this paper, we will focus on the performance analysis of the TMS320F28335 DSC, basing on benchmarking.**

***Keywords*— Embedded Systems, DSC, Performance, Benchmarking.**

## I. INTRODUCTION

Embedded systems are becoming one of the important factors of the e-industries growth. They are present in practically all human activities such as cellular telephones, personal digital assistants (PDAs), digital cameras, GPS receivers etc. Semiconductor markets have responded to this demand with a bewildering of other solutions for processing such as ASIC (Application Specific Integrated Circuits), FPGAs, DSP, DSC and SoC (System-On-Chip).

Initially, embedded control systems were implemented on microcontrollers (MCUs) due to their small size, efficient input/output communication port and their abilities to perform control applications. In the same era, DSPs are used in telecommunication, image processing and signal processing applications. To improve embedded systems performances, MCUs manufactures tried to increase the date bit size from 8 to 16 bits. Similarly, DSPs manufactures began to include more controllers to have the capacity to be called DSP controller (DSC)[1].

Different studies prove that DSC, an embedded controller with a specific microprocessor designed for typical mathematical operations to manipulate measured digital data, is capable of processing data speedily and generate output data in real-time. DSCs systems can accomplish complex and sophisticated embedded applications that can not be implemented using other processors techniques. They can be used in different application such as image processing [2], digital control processing [3, 4, 5], speech synthesis [6] and control implementation [7, 6].

DSCs integrate the algorithm processing power of a DSP engine with the hard, real-time control abilities of a MCU [1]. Its additional hardware units permit speed up the computational of sophisticated mathematical operations in order to reduce their memory capacity and the number of execution cycles in the processor. DSCs are compact with specific hardware and good performance for the best cost/benefit/performance.

Table 1 presents an illustration of the different features of the DSCs comparing to the MCUs and the DSPs processors.

TABLE I
MCUS, DSPS AND DSCS PROCESSORS FEATURES [1]

| Features | MCUs | DSPs | DSCs |
|---|---|---|---|
| Execute From Flash | ✓ | | ✓ |
| Large Register Set | ✓ | | ✓ |
| Robust Interrupt Capability | ✓ | | ✓ |
| Abundant Mixed Signal | ✓ | | ✓ |
| Single-Cycle MAC | | ✓ | ✓ |
| Dual-operand Fetch | | ✓ | ✓ |
| Zero-Overhead Fetch | | ✓ | ✓ |
| Saturation/Rounding | | ✓ | ✓ |
| Bit-Reverse Modes | | ✓ | ✓ |

Algorithms complexities require that the designer must have a clear idea about the hardware computing performance of the used DSC processor. Currently, there are many DSCs manufactures with many families. The challenges of designers, and especially new users of the DSCs field, are faced up with the various problems in selecting the appropriate processor to implement the most efficient algorithm on the least expensive hardware within given time. Choosing the correct system, can be based on a comparison of the performance of each processor to save energy, money and minimize the risk of the too time to market.

Many interesting studies based on the hardware performance evaluation of processors are presented [8, 9, 10, 11, 12] to allow fair comparisons between processors. Some of them are based on the comparison of the area, energy and cost consumption in a specific applications. Others used performance evaluation techniques to evaluate or compare processors efficiency.

The goal of this paper is to evaluate the performance of the TMS320F28335 DSC processor. The remaining parts of this paper are organized as follows: After this introduction. Section 2 illustrates the performance evaluation techniques and introduces our used methodology. Section 3 presents the HW platform used. Section 4 determines the performance measurement results of the TMS320F28335. Finally, section 5 summarizes the paper.

## II. Performance Evaluation Technique

The performance analysis of embedded systems has multiple aspects depending on the application that the system is made to. It will always be a true challenge for designer of this kind of systems, especially for DSC designers. Performance evaluation help designer to answer the following question: Does a particular DSC platform is appropriate for our application? How fast is the processor? Is it performed for real-time application? What is the memory usage? etc.

### *A. Historical Evaluation Technique*

In the past, analysis was performed by DSC venders such as Texas Instruments, Motorola and Analog Devices. The analysis approaches were chosen by the vender's platform. The first evaluation approach was computed by the number of operation per second [9]. Performance evaluation use a very simple metrics to describe processor performance based on MIPS (millions of instructions per second), MOPS (millions of operations per second) and MACS (Multi-accumulates per second). These metrics are misleading because of the various amounts of work performed by instructions. They become insignificants when RISC architectures appeared.

Actually, many solutions to measure hardware performance are presented. The most part of solutions are based on benchmarking applications.

### *B. Benchmark Evaluation Approach*

Benchmark approach is used by the Standard Performance Corporation in the popular SPEC benchmarks. Benchmarking is a widely recognized to performance evaluation. They are written in a high level programming language and measure the performance of both compiler and processor.

About 25 years ago, we didn't have the authorities of DSP benchmarking. Benchmarking was conduct almost only by chip vendors themselves. Nowadays, several (open source) benchmarks are used: Mibench [13], the most popular, Paranoia [14], LINPACK [15], etc. We can also find commercial benchmarking solutions more efficient like: SPEC (Standard Performance Evaluation Corporation) [16], EEMBC (Embedded Microprocessor Benchmark Consortium) [17], designed for embedded systems or EDN's DSP benchmark.

Since the early beginning of computer and engineering, benchmarking has been playing an exceptionally wide variety of extraordinary important roles, greatly influencing major HW/SW concepts. Recently, benchmarking attracted an exceptionally high attention in both research and industrial CAD communities.

The best benchmark is the application itself. However, in most cases we want a performance estimation of the end product at the initial phase of project.

Benchmarks can be divided into three categories depending on the application. They have been intended for (i.e. control benchmarks, computation benchmarks and I/O benchmarks). Since it's very difficult to fit existing benchmarks solely into one category, it is a better idea to take a combination of these criteria (such as control-computation benchmarks and I/O benchmarks).

Another useful way to categorize benchmarks is whether they are synthetic or application based. Three benchmark types for Microprocessor /MCU / DSP are used [18]:

1) *Synthetic Benchmarks*: developed to measure system specific parameters. Synthetic benchmarks are created with the intention to measure one or more features of systems, processors, or compilers. It try to mimic instruction mixes in real word applications. However, it is not related to how that feature will perform in a real application.

2) *Application Based Benchmarks or "real world" benchmarks*: developed to compare different processors architectures in the same fields of applications. Application based or "real world" benchmarks use the code drawn from real algorithms and they are more common in system-level benchmarking requirements.

3) *Algorithm Based Benchmarks:* (a compromise between the first and the second type) developed to compare systems architectures in special (synthetic) fields of application.

The optimal benchmark program for a specific application is the one who is written in a high-level language, portable across different machines, and easily measurable as well as having a wide distribution.

## III. Benchmark Program Selection and Specification

In our work we have chosen to adopt freely available benchmark solutions. In the first time, we used Synthetic Benchmarks based on the two complementary benchmarks : Dhrystone which report the integer performance of the architecture in Dhrystone MIPS and Whetstone which computes different algorithms and report the characteristics of the floating point units in whetstone MIPS. In the second time, we implement the Algorithm Based Benchmarks that perform mathematical operations with a basic fixed-point/floating-point computation.

### *A. Dhrystone Benchmark*

Dhrystone [19] is a synthetic computation benchmark program developed in 1984 by Reinhold P. Weicker in ADA and translated to C by Rick Richardson. It is intended to be representative of integer performance.

Dhrystone grow to become representative of general processor performance until it was outdated from Standard Performance Evaluation Cooperation. The recent version 2.1

of this benchmark is constituted by 103 high level statements within the main loop, which executes repeatedly during the benchmark execution. User can choose the number of iterations. As result, Dhrystone prints the absolutely time required per iterations through the loop, the performance measured in number of Dhrystone per second (the number of iterations of the main code loop per second).

### B. *Whetstone Benchmark*

Whetstone benchmark [20] is a synthetic benchmark written in 1972 at the National Physical Laboratory in the United Kingdom. It was the first intentionally written benchmark ware to measure processors performance. It originally measured computing power in units of kilo-Whetstone Instructions Per Second (kWIPS). This was later changed to Millions of Whetstone Instructions Per Second (MWIPS).

Both Dhrystone and Whetstone are synthetic benchmark, meaning that they are simple programs that are carefully designed to statistically mimic the processor usage of some set of programs. It difficult stems for the fact that one benchmark cannot effectively represent the variety of embedded applications.

### C. *Algorithm Based Program*

Algorithm Based Program presents a set of simple programs executed to evaluate the DSC processor architecture. They can be separated into different programs:

1) *The Coordinate Rotation Digital Computer (CORDIC):* invented by Jack Volder in 1959 [21, 22], is a simple program designed to estimate the basic elementary functions like trigonometric functions, square roots and exponential functions. Designers use CORDIC in practical all the applications : Biomedical applications to compute Fast Fourier Transforms (FFTs), robotics to determine the position and the movement of robotics joints and limbs, signal processing to generate sine and cosine waves, image processing to implement lighting and vector rotation and controls applications for asynchronous machine. CORDIC encloses two modes: The "Rotation" mode and the "Vectoring" mode : In the Rotation mode, input vector is rotated by a specified angle to compute sin and cosine, while in vectoring mode, the program rotates the input vector to the x axis to record the angle of rotation required to compute : $tan^{-1}(x/y)$.

2) *FIR (Finite Impulse Response Filter*): Filtering two uses [23]: Signal separation and signal restoration. In DSC, the digital filters are classified into FIR and IIR (Infinite Impulse Response). In our paper, we selected to benchmark the FIR filter which requires multiply-and-accumulate (MAC) operations to compute output from a 17-coefficients tap using simulated ADC input data. It has is implemented in practically all digital signal and image processing field such as the measurement of the electrical activity of a baby's heart (ECG signal).

On the following sections, we will present our platform.

## IV. Overview of the DSC HW/SW platform

### A. *Hw Platform*

Texas Instruments is one of the leader company producing DSCs. Depending on applications, three DSCs families are used[6]: C2000 family is efficient for real-time control applications, C5000 family focuses on mobile system and lastly, C6000 family used for audio, image processing and communication applications.

In our study, we choose to evaluate the performance of the C2000 family competent for real-time control application. The selected TMS320F28335 DSC is one of the cutting-edge floating-point DSCs in this series. It operates at 150MHz. Fig. 1 describes the functional block diagram of the TMS320F28335 DSC.

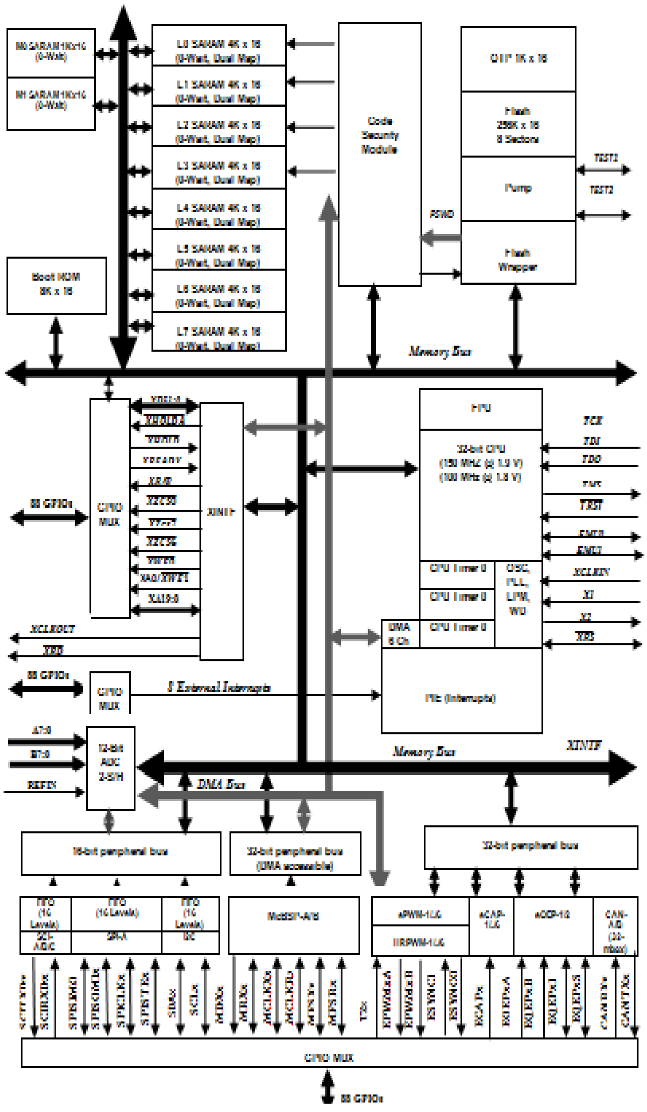

Fig. 1 TMS320F28335 Functional Block Diagram [6]

As mentioned in the Table 2, TMS320F28335 is fitted with a large memory capacity of 512KB on-chip flash, 2KB OPT ROM and 68KB asynchronous SRAM memory that sufficient to storage program. It is also, equipped with 12-bit ADC, an RS232 interface, a CAN interface, etc.

TABLE II
THE MAIN FEATURES OF TMS320F28335 DSC

| Feature | Description |
|---|---|
| Architecture | 32 bits Harvard Bus Architecture |
| Frequency | 150Mhz |
| Cycle Time | 6.61 ns |
| Clock and System Control | Dynamic PLL Ratio Supported, On-Chip Oscillator, Watchdog Timer module, Three 32 CPU Timers. |
| Memory-On-Chip | 512KB Flash, 68KB SARAM, 2KB OPT ROM |
| Peripherals | SPI, I2C, 12-bit ADCs, Internal oscillator, McBSP module, PWM module, watchdog, DMA, RS232, UART and eCAN |

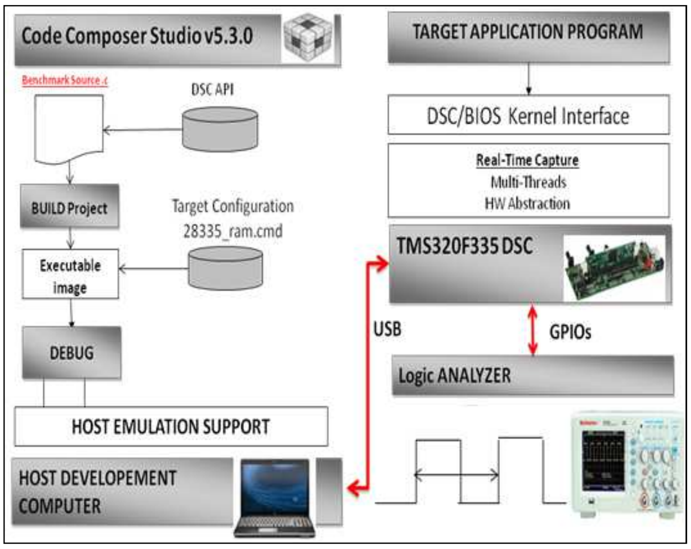

Fig. 2 Hw implementation of Benchmarks programs on TMS320F28335 DSC

## *B. SW Platform*

There are two methods types to implement SW algorithms on the TMS320F28335 DSC [7]:

*1) Using HLS (High Level Synthesis Approach):* The design is developed with MATLAB/Simulink platform and the program can be directly downloaded into the DSC.

*2) Using the CCS IDE (Integrated Development Environment Code Composer Studio) tool:* CCS IDE offers an excellent framework for building and implementing programs written in C language and becoming a standard framework used by many embedded software vendors. It combines the advantages of the Eclipse software with advanced embedded debug capabilities from Texas Instruments resulting in a compelling feature rich development environment for embedded designers.

To implement benchmark program, we used the CCS IDE tool which permits to develop and debug embedded applications. It includes source code editor, compilers for each of Texas Instrument's device families, project build environment, debugger, profiler, simulators and many other features.

## V. PERFORMANCE MEASURES

## *A. Experimental Setup*

The implementation framework for benchmarks programs consist of a : TMS320F28335 DSC, a logic analyzer and a computer as the host. The hardware block diagram of the benchmarks implementation is shown in the Fig. 2.

The benchmark code is required to be written in the CCS. Then, the code is also compiled, linked, downloaded and executed on the processor. After downloaded the executable code on the processor, the code runs wholly on the DSC.

Processor performance can be measured in many ways. The most common metric is the time required for a processor to accomplish defined task. Some architecture use internal CPU clock driver. The total execution time for the code is the clock driver multiplied by the total instruction cycle count. This clock divided is not reflected in the total instruction cycle count number presented.

In our case, Time is measured using an internal Timer, which operates at 150 MHz that triggers an interrupt every 1µs, and a Logic Analyzer, which measure value in order of nanosecond, to have a high precision measurement.

In the next section, we will present results for benchmarks performance analysis.

## VI. BENCHMARKS RESULTS ANALYSIS

Clearly each benchmark can only be compared to itself, as the resulting values are meaningless outside of that benchmarks context. The executed time for each benchmarks is very small, so a number of loops where used to get a time in microsecond range.

## *A. Dhrystone Benchmark results*

Dhrystone benchmark is used to measure the performance of processors in handling pointers, structures and string. It is dominated by simple integer arithmetic, string operations, logic decisions, and memory accesses intended to reflect the processors activities in most general purpose computing applications. Results of the Dhrystone benchmark are based on the speed time: The number of microseconds that Dhrystone program takes to run. To evaluate TMS320F28335 DSC performance, we choose a Loop equal to 100. Dhrystone MIPS (DMIPS) is calculated using the following formulas*:*

$$\mathrm{DMIPS} = (\mathrm{Loop}/\mathrm{Run\_Time})/1757$$

Where:

- Run_Time : The time spent to run Dhrystone benchmark
- Loop : The Loop used
- 1757 : The number of Dhrystones per Second obtained on the VAX 11/780 (Virtual Adress extention), nominally a 1 MIPS machine.

It is interesting to compute the Dhrystone score as a function of the DSC frequency to show the effectiveness of the DSC core rather than how fast it can run. DMIPS/MHz is computed using the following formulas.

$$\mathrm{DMIPS/MHz} = \mathrm{DMIPS}/(\mathrm{Frequency\ Of\ DSC\ in\ MHz})$$

Table 3 report performances resulting from the execution of Dhrystone benchmarks using 8 bits and 16 bits precisions under the TMS320F28335 DSC platform.

TABLE III
RESULTS OBTAINED USING DHRYSTONE BENCHMARK

| | **8 Bits Precision** | **16 Bits Precision** |
|---|---|---|
| Run_Time (µs) | 1491 | 1491 |
| DMIPS | 38.172 | 38.172 |
| DMIPS/MHz | 0.254 | 0.254 |

The use of the 8 bits and the 16 bits precision have not effect on the performance (MIPS) results since the Dhrystone benchmark does not use huge values. It is dominated by single integer arithmetic, string operations, logic decisions, and memory accesses intended to reflect the CPU activities in computing applications.

## B. *Whetstone Benchmark results*

Whetstone benchmark attempts to measure the performance of both fixed-point and floating-point arithmetic in a variety of scientific functions. These functions are divided into modules:

- M1 : Computation with simple identifiers.
- M2 : Computation with array elements.
- M3 : Passing an array as parameter.
- M4 : Performing conditional Jump.
- M5 : Performing integer arithmetic.
- M6 : Computation of Trigonometric Functions.
- M7 : Procedures Call.
- M8 : Array reference and Procedure Call.
- M9 : Integer Arithmetic.
- M10: Computations Standard Functions.

To evaluate the speed (Run_Time) using whetstone benchmark in microsecond, we have to use, for one iteration, a Loop equal to 10. The obtained results of this benchmark are summarized on the Fig. 3 presents the Run_Time (µs) of each whetstone module using the two kinds of floating-point precision: Single precision using float (32 bits precision number) and double precision using double (64 bits precision number).

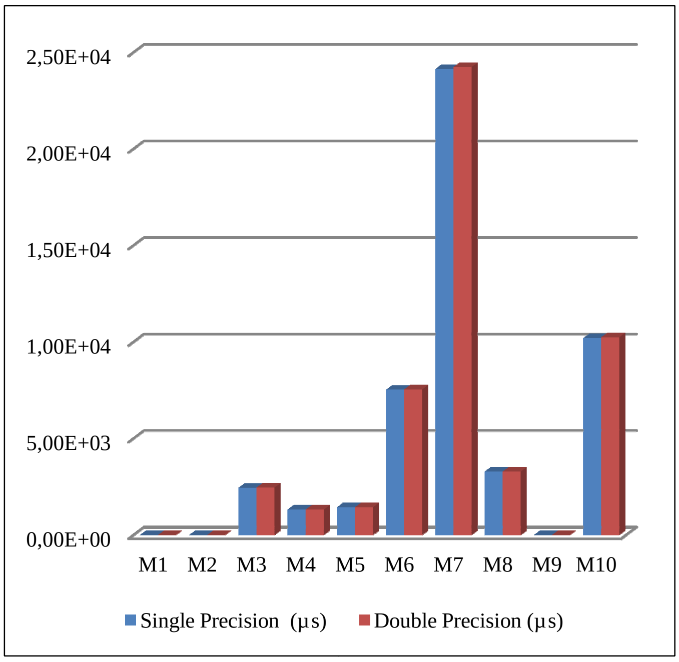

Fig. 3 Results obtained using the two data precision for whetstone benchmark modules

These results prove that the execution time using integer modules (1-5-9) and floating-point units modules (2-6) are rapidly executed comparing to the call procedures modules which require more than 50% of the whole execution time. The number of Whetstone Instruction per second (WIPS) can be measured for all Whetstone benchmarks. It is calculated as follows:

$$\mathrm{WIPS} = (100.0 * \mathrm{Loop})/\mathrm{Run\_Time}$$

Where:

- Loop : The Loop used
- Run_Time : The time spent to run benchmark

Table 4 presents the performance of the TMS320F28335 DSC on Kilo-Whetstone Instructions Per Second (KWIPS).

TABLE IV
RESULTS OBTAINED USING THE TWO DATA PRECISION FOR WHETSTONE BENCHMARK

| | **Single Precision KWIPS** | **Double Precision KWIPS** |
|---|---|---|
| Whetstone calculated | 19.83 | 19.76 |
| Whetstone measured | 19.92 | 19.84 |

## C. *Algorithms Application Benchmarks results*

Algorithms Based Benchmarks use simple programs. Each program is a unique code for testing special parts of the architecture. FIR filter and CORDIC programs, are widely used in the DSCs field. Performance analysis using these Algorithms is based on the time required to achieve each defined task using a Loop equal to 1000.

Table 5 depicts execution results of these Applications benchmarks.

TABLE V
RESULTS OBTAINED USING THE TWO DATA PRECISION FOR ALGORITHMS APPLICATIONS BENCHMARK

| | **Single Precision Run_Time (µs)** | **Double Precision Run_Time (µs)** |
|---|---|---|
| CORDIC "Rotation" Mode | 390.6 | 388.6 |
| Transcendental sine and cosine functions | 1.260 | 1.260 |
| CORDIC "Vectoring" Mode | 760.6 | 760.6 |
| Transcendental tang$^{-1}$ function | 3.018 | 3.010 |
| FIR filter | 130.4 | 130.8 |

CORDIC algorithm is not a very fast algorithm for use compared to the transcendental mathematical functions. It is followed due to its very simple implementation based on simples shift- add operations. So, trigonometric functions should be computed using transcendental mathematical functions.

Results of Whetstone benchmark and Applications Based Benchmarks, CORDIC and FIR filter, indicate that the minimum execution time is provided using double precision. The double precision is actually faster than the single precision for the processors optimized for high-speed mathematical calculations (DSCs, DSPs) using transcendental mathematical functions which return a double values. These results prove the efficiently of the DSC processors on the signal processing applications.

## VII. CONCLUSIONS

The reported benchmarks results cover three complementary benchmarks using single and double data precision. Dhrystone is used to compute integer unit performance. Whetstone is able to characterize floating-point operations. Algorithm Application Benchmarks are used to measure the calculus capacity of processor. The same approach can be used to analysis other embedded systems or other architectures.

In our work, we focused on the hardware excursion speed evaluation in the embedded system design flow. We measured the CPU performance of the TMS320F28335 DSC in term of execution time without using optimization. It's very interesting to compute the effect of optimization techniques of the compiler on execution time. This work can be extended by evaluating the TMS320F28335 DSC performance comparing to other architectures such as FPGAs.

## REFERENCES

[1] S. Mitra, "When MCUs and DSPs Collide: Digital Signal Controllers", Microchip, available : www.microchip.com.

[2] K. Illgner, H. GrubeF, P. Gelaberf, J. Liang, Y. Yoo, W. Rabadiz and R. Talluri, "PROGRAMMABLE DSP PLATFORM FOR DIGITAL STILL CAMERAS", Acoustics, Speech, and Signal Processing, Proceedings., IEEE International Conference, vol. 4, pp. 2235-2238, Mar. 1999

[3] C. Buccella, H. C. Cecati, and H. Katafat, "Digital Control of Power Converters-A Survey", IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, vol. 8, pp. 437-447, Aug. 2012

[4] S. A. Mir, M. E. Elbuluk, and D. S. Zinger, "Fuzzy Implementation of Direct Self Control of Induction Machines", IEEE TRANSACTIONS ON INDUSTRY APPLICATIONS, vol.30, pp.729-735, 1994

[5] P. Dobra, R. Duma, D. Moga, and M. Trusca, "Digital Control Applications using TI Digital Signal Controller", WSEAS TRANSACTIONS ON SYSTEMS AND CONTROL, Issue 6, vol.3, pp. 729-735, June 2008

[6] Texas Instrument Datasheets and Tutorial, (2013), Available : www.ti.com.

[7] C2000 MCU Teaching ROM, "TMS320F2833X Digital Signal Processor Implementation Tutorial", Texas Instrument, 2010

[8] Y. L. Goh, A. K. Ramasamy, F. H. Nagi, A. Azwin, and Z. Abidin, "Evaluation of DSP based Numerical Relay for Overcurrent Protection", INTERNATIONAL JOURNAL OF SYSTEMS APPLICATIONS, ENGINEERING & DEVELOPMENT, Issue 3, vol. 5, ppv 396-403, 2011

[9] N. Litayem, B. Jaafer and S. B. Saoud, "Embedded Microprocessor Evaluation- A case study of theLeon3 Processor", Tecnia Journal of Manajment Studies, vol. 6, pp. 40-46, 2011

[10] Groâschädl J., Tillich S., Szekely, "*Performance Evaluation of Instruction Set extensions for Long Integer Modular Arithmetic on a SPARC V8 Processor*", IEEE DSD. 2007

[11] Kurian L., Lieven Eeckhout, "Performance evaluation and Benchmarking", CRC Press, ISBN 0-8493-3622-8, 2006.

[12] [12] Wolf W. (2007). "High-performance embedded computing", ISBN 13: 978-0-12-369485-0, Elsevier.

[13] M. R. Guthaus, J. S. Ringenberg and D. Ernst, "MiBench M. R.. A free, commercially representative embedded benchmark suite". *In Proc. of 4th Annual IEEE Workshop on Workload Characterization*, 2001

[14] R. KARPINSKI, "Paranoia: A floating-point benchmark". Byte Magazine 10, pp. 223–235, 1985

[15] J.J. Dongarra, J.R. Bunch, C.B. Moler and G.W. Stewart. "LINPACK Users Guide", SIAM Pub, Philadelphia, PA, 1979.

[16] Henning J. L., "SPEC CPU2000: measuring cpu performance in the new millennium, IEEE Computer", vol.7, pp.28–35, July, 2000.

[17] EDN Embedded Microprocessor Benchmark Consortium, available : http://www.eembc.org

[18] Berkeley Design Technology Inc., "Evaluating DSP Processor Performance", 2002.

[19] G. Stitt, R. Lysecky and F. Vahid, "Dynamic Hardware/Software Partitioning: A First Approach", *in Proc. of the 40th Design Automation Conference*, pp. 250-255, 2003

[20] H. J. Curnow and B. A. Wichmann, "Whetstone benchmark: A synthetic benchmark", The Computer Journal, vol. 19, pp. 43-49. 1976

[21] Manoj Arora, R. S. Chauhan, Lalit Bagga, "FPGA Prototyping of Hardware Implementation of CORDIC Algorithm", International Journal of Scientific & Engineering Research, vol. 3, pp. 1-6, January 2012.

[22] T. Menakadevi and M. Madheswaran, "Direct Digital Synthesizer using Pipelined CORDIC Algorithm for Software Defined Radio", International Journal of Science and Technology, vol. 2, June 2012

[23] Application Report Texas Instrument, "MSP430 Competitive Benchmarking", January, 2009.