Technology Transfer in Computing Systems

D3.43: Individual TTP43 abstract

Project no.:	609491
Funding scheme:	Collaborative project
Start date of the project:	1st September 2013
Duration:	36 months
Work programme topic:	FP7-ICT-2013-10
Deliverable type:	Report
Deliverable reference number:	ICT-609491 / D3.43
WP and tasks contributing:	WP 3 / all
Due date:	31/07/2016
Actual submission date:	14/07/2016
Responsible Organization:	LUH
Dissemination Level:	Public
Revision:	1.0

TETRACOM D3.43: A Highly Optimized Arithmetic Software Library and Hardware Co-Processors IP for Fixed-Point VLIW-SIMD Processor Architecture

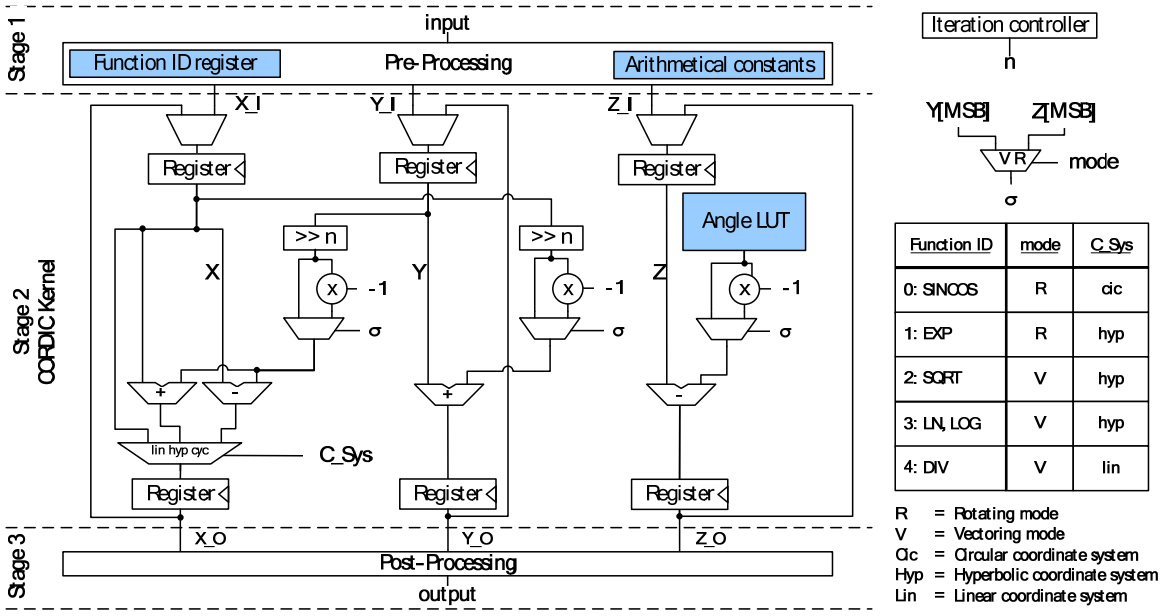
Guillermo Payá-Vayá, Lukas Gerlach, Stephan Nolting; Gottfried Wilhelm Leibniz Universität Hannover, Germany

Carsten Reuter, Hans-Joachim Stolberg; videantis GmbH, Hannover, Germany

The goal of this TETRACOM project has been to transfer the intellectual property of the LibARITH, which represents a collection of approximation methods for widely used complex arithmetic functions, to the videantis v-MP processor system framework. videantis is a Hannover based company, that has specialized on energy efficient, embedded processor systems for media applications. These processors are mainly used in driver assistant systems to provide advanced safety and entertainment systems.

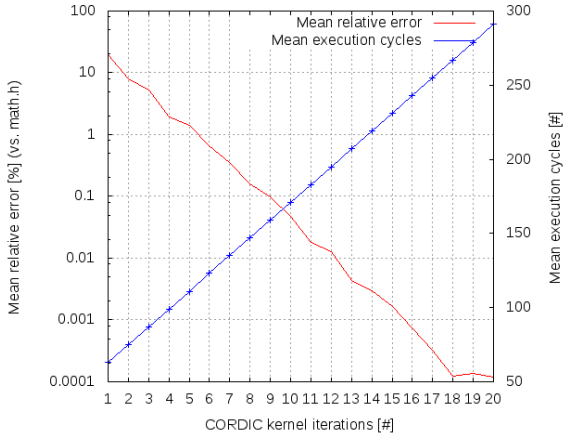
Among others, the LibARITH library, developed at the Institute of Microelectronic Systems at the Leibniz Universität Hannover, provides a framework of CORDIC-based functions to approximate a wide range of arithmetical functions, which present elementary operations for a wide range of embedded computation tasks. These radix-2 CORDIC functions implement vital functions like sine and cosine computations, square root and logarithm.

The main task of this TETRACOM project was portrayed by porting this pure software functions to the videantis v-MP processor architecture and verifying the implemented functions in their new ecosystem. For this work, the sine/cosine, square root, exponential function, natural logarithm, inverse tangent and the generic division function were ported. All these functions are based on 2 generic CORDIC kernel functions (a generic version of the radix-2 CORDIC kernel is shown in the figure below), but require additional pre- and post-processing steps to accomplish the desired task.

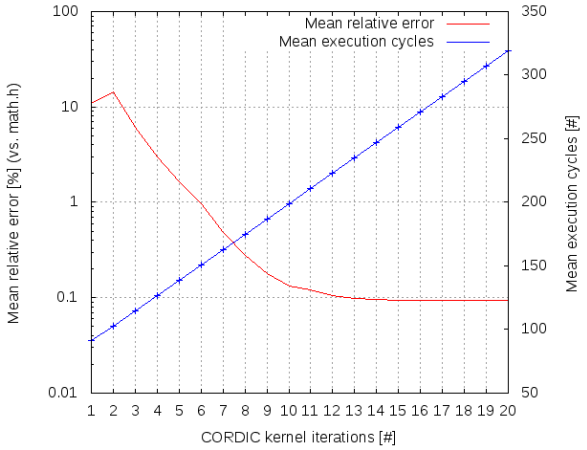


The implemented generic radix-2 CORDIC kernel

These CORDIC-based functions make use of several parameters to configure the algorithm's computation to the needs of the user and the target application, respectively. For instance, the number of CORDIC iterations and the actually used fixed-point representation format can be selected dynamically (during runtime). Hence, an a-priori evaluation of these parameters is required to select the best-fitted configuration. For this purpose, a simple but efficient evaluation framework has been implemented. This evaluation framework allows to determine the best setting for all configuration parameters. The figures below show an exemplary output. Here, the error (red graphs) of the CORDIC-based functions compared to a standard approximation method using the C standard math.h library are shown using a specific fixed-point format and a variable number of CORDIC iterations. Additionally, the required execution cycles are shown (blue graphs).



Computaiton of the **ATAN** function using 20-bit integer part and a variable number of CORDIC iterations.



Computaiton of the **DIV** function using 20-bit integer part and a variable number of CORDIC iterations.

Besides the graphical evaluation results, the evaluation framework also generates more precise reports, which represent information about the actual execution. These include statistical data concerning the relative and absolute error (compared to the math.h C library) as well as execution cycle requirements:

```
Function:              F10         -- Precision (relative) --          -- Timing --
Test cases:            64          Min rel error:          0.000000%   Min exe cycles:  243
Input fp format:       12.20       Max rel error:          2.467811%   Max exe cycles:  254
Output fp format:      12.20       Rel mean error:         0.040154%   Mea nexe cycles: 249.062500
CORDIC iterations:     15          Rel standard deviation: 0.305885%   Exe standard deviation: 4.137764
```

Besides the complex arithmetic functions, the emulated floating point library, which is also part of the LibARITH, was ported to the videantis v-MP processor architecture. The emulated floating point library provides a pure SW implementation of a floating point number format and the corresponding basic operations. In many cases, floating point numbers are very rare in embedded applications, since they require expensive computation cycles or even dedicated hardware for efficient computation. However, in some situations, where the dynamic range or the required precision of a number format cannot be determined a-priori or must be variable during runtime, such implementations are unavoidable.

This library implements a specific number format, consisting of 24 bit for the mantissa and 8 bit for the exponent. Hence, it is not fully IEEE 754 compatible. However, it allows to make efficient use of the available SIMD hardware mechanisms while providing variable precision and dynamic range.

The emulated floating point library implements basic operations, like addition, subtraction and multiplication. Besides these operations, subword-oriented operations like ADDSUB and SUBADD, which perform an addition and a subtraction in parallel on two 32-bit subwords, stored in a single 64-bit wide register. Additionally, conversion functions were ported, that provide means to convert the emulated floating point numbers into a fixed-point format and vice-versa.

To allow an evaluation of the emulated floating point numbers, a plain C converter tool was added to the LibARITH, which provides the conversion between any of the following formats: IEEE 754 conform numbers, fixed-point numbers and emulated floating point numbers. The converter output also shows the resulting conversion errors to allow an approximation of the maximum error.

As a conclusion, the ported LibARITH provides elementary functions required especially for embedded media application tasks. The evaluation framework aids the designer to efficiently match arithmetic demands of complex media application.