A Dual-Mode Synchronous/Asynchronous CORDIC Processor

Eckhard Grass, Bodhisatya Sarker and Koushik Maharatna

IHP
Im Technologiepark 25
15236 Frankfurt (Oder), Germany
grass@ihp-microelectronics.com

Abstract

For application in a software defined radio a CORDIC processor has been developed that can operate both in synchronous and asynchronous mode. Each mode of operation has advantages and drawbacks. Depending on the actual application, an optimal trade-off can be achieved by selecting the mode of operation that fits best with system demands. We believe that for a system developer this additional degree of freedom significantly increases the application space.

The design has been implemented on a 0.25 µm SiGe:C BiCMOS process. Simulation results using post-layout extracted parameters indicate that the design is competitive both with purely synchronous and purely asynchronous implementations.

1. Introduction

Fourth generation (4G) wireless and mobile systems are currently the focus of research and development. They will allow new types of services to be universally available to consumers and for industrial applications. Broadband wireless networks will enable packet based high datarate communication suitable for video transmission and mobile Internet applications.

This paper is based on a project which aims to develop a single-chip wireless broadband communication system in the 5 GHz band, compliant with the Hiperlan/2 [1] and IEEE 802.11a [2] standards. Both standards specify broadband communication systems using Orthogonal Frequency Division Multiplexing (OFDM) with data rates ranging from 6-54 Mbit/s. The duration of one OFDM symbol, which can carry 24 to 216 bits of information is 4 µs. For such a system a large portion of the digital baseband processor has to operate on complex signals which are sampled at a rate of 20 MS/s. The most difficult to implement functional blocks of the baseband processor are FFT/IFFT processor, Viterbi Decoder, and Numeric Controlled Oscillator (NCO). The NCO is used to perform a frequency correction on the incoming signals. A Coordinate Rotation Digital Computer (CORDIC) offers an elegant way of implementing an NCO.

To open a broad market for consumer products, low cost of the required hardware is essential. One way to realise low-cost systems is to reduce system complexity and implement all functions of a wireless modem in a single chip. Potentially, a single-chip solution is also advantageous in terms of performance and power dissipation when compared with multi-chip implementations. Fewer signals have to be routed via slow and power-hungry pad drivers. In addition, short interconnections allow faster operation of the system.

Our in-house 0.25 µm SiGe:C BiCMOS technology enables the integration of complex digital baseband functionality together with the analog RF front-end (AFE). This process technology forms the basis for the single-chip implementation of a complete wireless modem. This single-chip wireless modem will hence comprise analog circuitry, processor cores as well as dedicated hardware.

For the single-chip modem, a number of critical design problems have to be solved. One major issue is the reduction of crosstalk from the digital part to the analog circuitry. The high energy which is delivered by clock drivers can influence the AFE and deteriorate its signal to noise (S/N) ratio. The most predominant crosstalk mechanisms are substrate noise, supply voltage variations as well as electro-magnetic effects (EMR) [3]. Secondly, system power dissipation is a critical issue. Since many functional blocks are active for a limited period only, there is potential to reduce power dissipation drastically. A token-flow approach [4] is adopted by adding signals which indicate the validity of data at inputs and outputs of functional blocks. Distributing the control of data flow, this token-flow approach allows easy implementation of clock gating and lends itself to asynchronous implementations of blocks.

Currently a number of extensions to the standards for 5 GHz broadband communication systems are being discussed by the standardisation committee. Therefore, it would be beneficial for customers and suppliers of those systems if hardware would be able to accommodate future communication standards. A software defined radio, in principal, offers this needed flexibility.

Here we extend the idea of software defined radio such that the software developer can chose certain hardware accelerators to operate either synchronously or asynchronously. This additional degree of freedom increases the application space of the complete system. Furthermore, also with a given standard, different application scenarios require different properties of the hardware. We are convinced that having the freedom for certain key blocks of choosing between synchronous and asynchronous operation allows finding a suitable trade-off between power dissipation, crosstalk, data throughput, latency and (self-) testability.

As an example, it may be desirable with very small signal amplitudes on the receiver to reduce the impact of the CLK signal by operating the CORDIC in asynchronous ('whisper') mode. On the other hand, if the receive signal is very strong and for continuous reception of data, the more power efficient synchronous operation in conjunction with clock-gating can be used.

The paper discusses a dual mode implementation of a CORDIC processor and compares the main parameters of this design with a purely synchronous as well as a purely asynchronous implementation. It is demonstrated that asynchronous circuits can be implemented in VHDL using synchronous tools. VHDL synthesis and layout are supported by a number of attributes and constraints which are specific to the tools used. The hardware overhead of the dual mode CORDIC processor compared to the synchronous version is about 12 %. It shows a similar performance when compared to its synchronous counterpart.

The remaining paper is structured as follows: In Section 2 the architecture of the CORDIC processor is derived from the algorithmic specification. Section 3 discusses strategies for implementing the design. In Section 4, a number of issues related to the design flow for asynchronous implementations and the used tools are highlighted. Finally in Section 5 the results of our implementation are reported and conclusions are drawn in Section 6.

2. CORDIC Processor Architecture

The versatility of CORDIC supplemented with its multiplierless feature has propelled digital designers to incorporate it in several arithmetic processors as well as to formulate and implement many modern DSP algorithms [5-9] that require complex number multiplication, elementary plane rotation or lattice operations.

In essence, the CORDIC accepts three input variables x0, y0, and z0 and generates the output x1, y1 and z1. It can be operated in two different modes viz., the rotation and the vectoring mode where, variables z and y are forced to zero respectively. Each of these modes can be utilised in circular, linear and hyperbolic coordinate systems. This illustrates the versatility of the CORDIC and explains why it is regarded as one of the major tools in digital computer arithmetic.

However, for our purpose it is sufficient to use the rotation and vectoring operations in the circular coordinate system only. Hence we concentrate on the realization of a circular CORDIC processor. It's operation relies on the principle of vector rotation. The basic operation is described below:

The rotation of a vector $[x0 \quad y0]^T$ in the Cartesian co-ordinate system can be described as,

$$\begin{bmatrix} x1 \\ y1 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x0 \\ y0 \end{bmatrix} \qquad (1)$$

where, $[x1 \quad y1]^T$ is the final vector and θ is the angle of rotation $(-99.9° \le \theta \le 99.9°)$. In the above equation it was assumed that the rotation takes place in clockwise direction. Now, the target angle θ can be expressed as the summation of a number of elementary angles α_i so that

$$\theta = \sum_{i=0}^{b-1} \sigma_i \alpha_i \qquad (2)$$

where, b is the bit precision of the machine in which the operation is to be implemented and $\sigma_i \in \{1, -1\}$, is known as the direction of rotation. In the conventional circular CORDIC operation α_i is expressed as,

$$\alpha_i = \tan^{-1}(2^{-i}) \qquad (3)$$

Substituting (2) in (1) and using (3) one may write

$$\begin{bmatrix} x1 \\ y1 \end{bmatrix} = \prod_{i=0}^{b-1} \cos\alpha_i \begin{bmatrix} 1 & \sigma_i 2^{-i} \\ -\sigma_i 2^{-i} & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} \qquad (4)$$

$$\text{and } \sigma_i = \mathrm{Sign}\left[\alpha - \sum_{r=0}^{i-1} \alpha_r\right] \qquad (5)$$

$$z_{i+1} = z_i + \sigma_i 2^{-i} \qquad (6)$$

The physical meaning of these equations is that the target angle given as the 'z' variable is driven to zero by to-and-fro motion of the vector. At each iteration step the sign of the residual angle (s_i) is calculated and accordingly the vector is rotated in clockwise or counter clockwise direction. From the hardware implementation point of view, this operation can be carried out using only shift-and-addition operations since multiplication of any quantity by 2^{-i} is a shift of the binary representation of the quantity by i-bit to the right.

However, the conventional CORDIC algorithm suffers from two principal drawbacks, viz., the requirement of a scale factor and the slow rate of convergence. In order to overcome these difficulties we developed a modified CORDIC algorithm which is described below.

In our scheme, the target angle is achieved by rotating the vector only in one direction (either clockwise or counterclockwise) through sufficiently small elementary angles α_i. They can be expressed by the following equation

$$\sin \alpha_i \cong \alpha_i = 2^{-i} \qquad (7)$$

This assumption imposes a lower limit on the value of iteration index i. The lower limit of i can be calculated from the fact that the largest value neglected in the expansion of the sine series is

$$2^{-3i} / 6 = 2^{-(3i+\log_2 6)} = 2^{-(3i+2.585)}.$$

Thus, the condition for preserving the accuracy during such approximation becomes

$$3i + 2.585 \geq b \qquad \text{or}$$

$$i \geq \lfloor (b - 2.585) / 3 \rfloor = p \qquad (13)$$

(since i can adopt only integer values).

The upper limit of i is b-1 since a right shift of any b-bit number by b-bit will result in machine zero. Under the above circumstances, the values of $\sin \alpha_i$ and $\cos \alpha_i$ can be expressed as,

$$\sin \alpha_i = 2^{-i} \qquad (14)$$

$$\cos \alpha_i = 1 - 2^{-(2i+1)} \qquad (15)$$

Considering the approximation, $s_i = +1$ (since the sign of the residual angle is always the same for all iterations) and clockwise rotation of the vector, equation (1) may be rewritten as

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \prod_{i=p}^{b-1} \begin{bmatrix} 1 - 2^{-(2i+1)} & 2^{-i} \\ 2^{-i} & 1 - 2^{-(2i+1)} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} \qquad (16)$$

Equation (16) can be realized in practice using only shift-and-add operations. Furthermore, contrary to equation (4), no scaling term appears in equation (16), which implies a scaling free CORDIC operation. The elimination of the scaling term is a significant step, since it avoids the requirement of extra post-processing circuitry and saves processing time. The datapath components for implementing the operation of equation (16) are shown in Figure 1. This can be viewed as an elementary rotational stage rendering a rotation of α_i to its input vector. This requires four shifters, two subtractors and two adder / subtractors as datapath components. Thus, compared to the datapath of the elementary rotational stage of an unscaled CORDIC, it requires two shifters and two subtractors more. However, for the elementary rotational stages corresponding to $i \geq b/2$, these extra shifters and subtractors can be omitted as the right shift of the input quantity by (2i+1) becomes machine zero and thus does not affect the accuracy any more.

However, equation (7) implies that the convergence range of the CORDIC will be small. The convergence range can be extended over the entire coordinate space by repeating certain iteration steps and by exploiting the symmetry of the coordinate axes. In principle, we found that to cover the whole coordinate space, it is sufficient to compute the angles in the interval [0, 22.5°]. The results of CORDIC rotations for any angle between 22.5° and 360° can be extrapolated from the result of a rotation corresponding to [0, 22.5°].

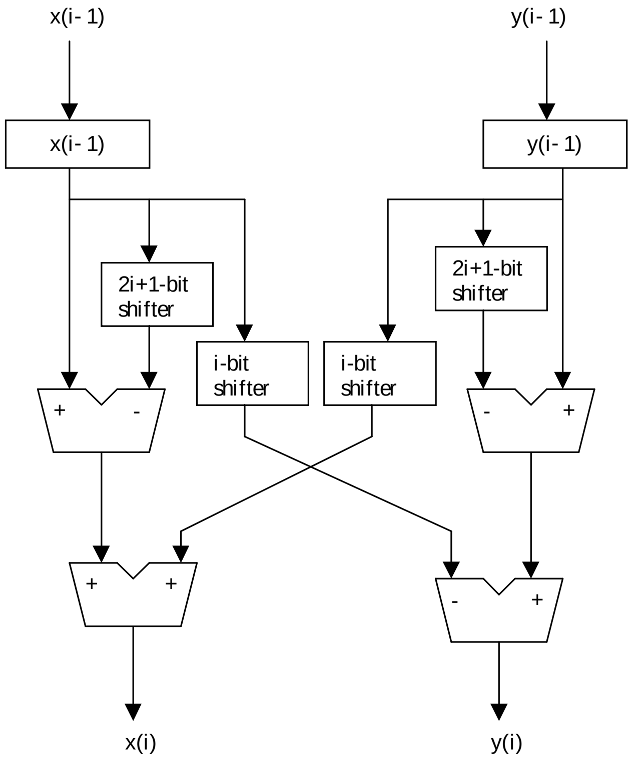

Figure 1. Block diagram of an elementary datapath component (rotor stage i).

The complete CORDIC processor capable to operate in the rotation and vectoring mode over the entire coordinate space has three parts, viz., sign and domain detection circuitry, basic CORDIC section with convergence range – 22.5° and the output circuitry. This is shown in Figure 2. The sign and domain detection circuitry (designated as sgn_detect) detects the sign of the input angle, quadrant and the domain (position of the vector in a quadrant) of the input vector. Accordingly, the input variables are processed and they may be regarded as the modified input to the basic CORDIC processor. The detection of sign and domain, pre-processing and appropriate input operations require two clock cycles. The basic CORDIC processor (designated as cord_pipe) is a bit parallel pipelined implementation with internal wordlength of 16 bits. From equation (13), the value of 'p' for a 16-bit implementation is approximately 4. The pipeline is 17 stages long where the elementary rotational stage corresponding to 2^{-4} is repeated 6 times. Our aim is to approximate z and y for the rotation and the vectoring mode respectively within an accuracy of $O(2^{-16})$. In the construction of the elementary rotational sections, the conventional two's complement addition is used. The information about the quadrant and domain of the initial vector detected at the sgn_detect module is transferred between two consecutive sections of the pipeline along with the data in a local register transfer manner. This means that the data in different sections of the pipeline has a token attributed to it that essentially carries the information about its initial quadrant and domain. A single bit signal, 'sign' and two 2-bit signals 'domain' and 'quad' are used to transfer this information between successive rotational stages.

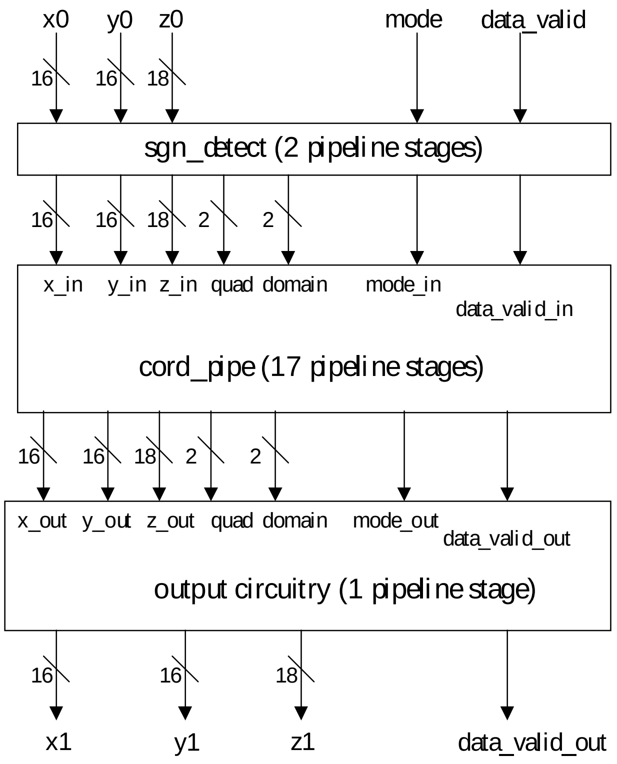

Figure 2. Structure of the synchronous pipelined CORDIC processor

The output circuitry consists of a fixed scaling unit of 1/Ö2 and a couple of adder/subtractor units. This circuit post-processes the output vector emerging from the cord_pipe section in accordance with the information on the initial vector derived by the sgn_detect circuitry and generates the final output. All the operations at the output are completed in one clock cycle.

A mode signal is also provided with the processor. The logic 'HIGH' state of the mode signal makes the processor to operate in rotation mode whereas; its logic 'LOW' state means the vectoring mode of operation is carried out. Apart from the mode signal, the (synchronous) processor is also equipped with an input 'data_valid' and an output 'data_valid' signal that indicates the validity of input and output data respectively.

3. Implementation Strategies

The main components of the OFDM baseband processor for Hiperlan/2 and IEEE802.11a can easily be implemented using a datapath architecture. The most complex blocks in this datapath are CORDIC-Processor, FFT processor and Viterbi Decoder. It is our goal to make the design of the complete baseband processor as much as possible technology independent. Therefore it has been decided to use VHDL for design entry throughout.

In order to de-centralise control, a token-flow approach is used. The validity of data tokens is indicated using additional signals at the input and output of functional blocks.

To allow performance comparison, three different circuits were designed: Circuit 'MIX-CORDIC' can operate both synchronously as well as asynchronously, circuit 'ASY-CORDIC' operates asynchronously and circuit 'SYN-CORDIC' is a 'normal' synchronous design with extra signals to indicate data validity at inputs and outputs.

Dual-mode circuit (MIX-CORDIC):

A 4-phase handshake protocol is deployed since it allows the use of single edge triggered flip-flops. The handshake circuit of the MIX-CORDIC design is implemented using two C-elements as shown in Figure 3. In order to avoid confusion between the type of the signal ('input' or 'output') and the port of the handshake circuit, in this paper we define the 'input' port of the handshake circuit as port 'a' and the output port is labelled port 'b'.

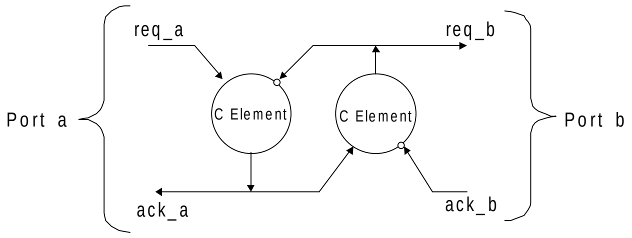

Figure 3. Handshake circuit for circuit MIX-CORDIC

Unfortunately our cell library does not contain any C-elements. RS flip-flops are also not available. Consequently the C-elements had to be constructed from gates. We modelled the behaviour of the C-elements in VHDL and synthesized them. They were then used as components in the VHDL design. It was found that the Synopsys synthesiser does merge the functionality of a number of C-elements into a single circuit during optimization stage. Since in this process timing constraints are violated, the C-element components had to be protected from optimisation using 'don't touch' attributes. The individual C-elements were extensively simulated under a variety of conditions and were found to be functioning correctly throughout. Before application in the CORDIC processor, our handshake circuit was also extensively tested using an asynchronous FIFO.

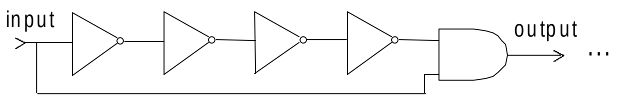

Figure 4. Single stage of an asymmetric delay element

For completion detection a bounded delay method was used deploying a special asymmetric delay element with short recovery time. A single stage of this delay element is shown in Figure 4. Alternatively a cascade of two-input AND gates can be used whereby the input signal is connected in parallel to all AND gates. However the input capacitance of this structure scales with the length of the cascade which creates a high capacitive load on the input signal making the delay highly unpredictable.

A cascade of ten stages of the structure shown in Figure 4 delivers in our circuit a delay of 13.4 ns and 4.8 ns for a rising and a falling transition respectively (measured after back annotation from layout). The recovery time for the falling transition of such a cascade is equivalent to the delay of four inverters of a single stage i.e. approximately 0.8 ns in our case. This recovery time is dimensioned such that it is well below any possible response time of the handshake circuitry. The achieved ratio between rise time and fall time of nearly 3:1 yields a substantial speedup of the handshake circuit when compared to using a 'normal' inverter chain. The circuit does not suffer from high input capacitance which also keeps its power dissipation low.

To allow switching between synchronous and asynchronous mode, for the dual-mode circuit additional logic is introduced to either enable the asynchronous handshake circuit or the synchronous clock to drive the pipeline stages as shown in Figure 5. If the 'sync' signal is in logic high state, the clock signal 'Clock_global' is enabled via the NAND gate at the input. The structure is configured to work in synchronous mode. At the same time 'req_a_top' is disabled via the NOR gate driving 'req_a'. The complete handshake circuit is disabled by forcing 'req_a' to logic low.

On the contrary if 'sync' is in logic low state, the global clock 'Clock_global' is disabled and the circuit is configured for asynchronous operation by enabling 'req_a_top' to drive signal 'req_a'. Currently there is no dynamic switching of mode considered i.e. signal 'sync' is allowed to change state only when 'Clock_global' is low and the pipeline is completely settled. This prevents potential hazards during mode-change operations which could result in undefined states of the system.

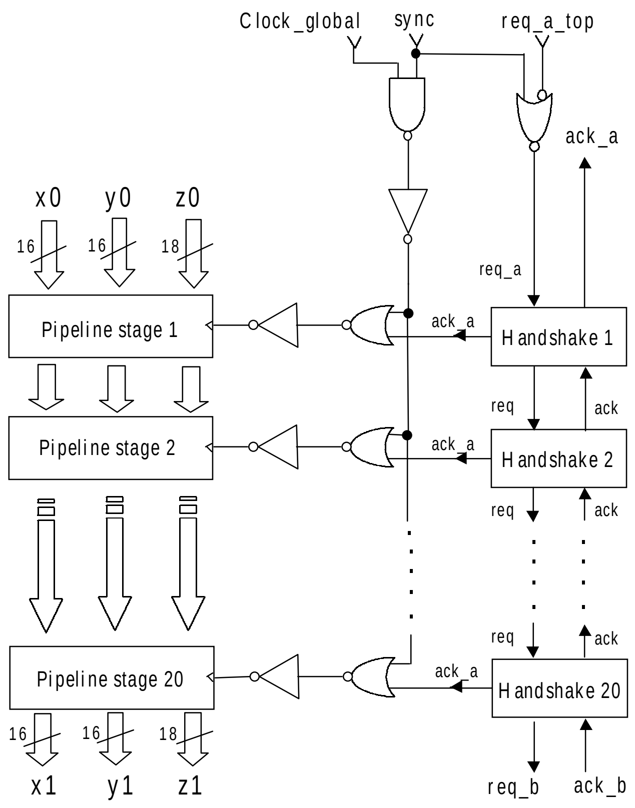

Figure 5. Schematic diagram of dual-mode CORDIC

Asynchronous circuit using latches (ASY-CORDIC):

A second version uses latches instead of flip-flops for buffering the data. Since latches are smaller and consume less power, this can potentially reduce silicon area and power dissipation. For our application, the semi-decoupled latch controller does represent a reasonable compromise between hardware complexity and performance [11]. The asymmetric C-elements used in this controller where designed in the same way as described above. The resulting handshake circuitry was again tested in an asynchronous FIFO. Performance measurements based on simulations using post-layout extracted parameters are given in Table 1.

Synchronous circuit (SYN-CORDIC):

The synchronous implementation was done using exactly the same pipeline architecture as for the other two designs. In order to support the token-flow approach 'data valid' signals where introduced at input and output ports of the pipeline. A performance comparison of the three different implementations is given in Table 1.

Interfacing of the asynchronous outputs with the synchronous environment is done using an additional (synchronous) D-type flip-flop for the request signal at the output ('req_b' in Figure 5). Using a typical clock frequency of 20 MHz for the synchronous circuit will result in a very low probability of metastability of $P=10^{55}$ or an MTBF of $3x10^{39}$ years [12]. For typical applications this will not require any additional measures to reduce metastability probability further.

4. Design-Flow Issues

All versions of the CORDIC processor were modeled in VHDL and are fully synthesized. For simulation and functional verification we used Modelsim from Mentor Graphics. Synthesis was carried out with Synopsys' Design Analyzer and the layout was generated using Silicon Ensemble from Cadence. The design was mapped to our in-house 0.25 mm SiGe:C BiCMOS technology using the IHP digital design kit. The nominal supply voltage for core cells is 2.5 V. Four metal layers are available for interconnect. The CORDIC processor utilizes only the CMOS part of our BiCMOS technology.

Both, for synthesis and layout some attributes and constraints were added to support the asynchronous implementation. After synthesis the gate level Verilog netlist generated by the Design Analyzer was re-simulated including the 'Standard Delay Format' (SDF) file. The correct timing behavior of the processor in both the rotation and vectoring mode was observed.

For driving the local clock signals of the individual stages we fitted the clock signal of every pipeline stage with an appropriate clock buffer. This buffer was synthesized separately and protected with the help of 'don't touch' attributes.

The area of a transparent latch used in the ASY-CORDIC design is 118 mm^2 and the area of the D-type flip-flops used in MIX-CORDIC and SYN-CORDIC is 180 mm^2 (according to IHP cell library). The latch based version can therefore potentially result in smaller silicon area. The design flow for design 'ASY-CORDIC' is described in detail below. In principal, the same approach was used for the MIX-CORDIC circuit. The 'normal' synchronous implementation does not require any extra measures and is therefore generated using the 'standard' synthesis flow with clock-tree generation during layout phase.

The latches within each pipeline stage are controlled with a signal 'lt' which causes the latch to be transparent when low and in hold state when high. Since the data path is in total 50 bit wide, the 'lt' signal must have a reasonable drive strength. Therefore a buffer is used for each pipeline stage to drive signal 'lt'. To keep the silicon area as small as possible, we used latches with an asynchronous reset signal. This is achieved by inserting the Synthesizer specific attribute 'async_set_reset' in the VHDL source code. After separately synthesizing each of the 20 pipeline stages, they were connected with their respective handshake circuit. Each handshake circuit had a delay element (tuned to the critical path of the combinational logic) inserted into the input request signal. A timing margin of 30% was added to guarantee reliable operation after layout. Then the top level design was synthesized separately and the netlists of the individual pipeline stages where integrated into the top-level netlist.

The layout was carried out using the standard cell approach of Silicon Ensemble with 80% row utilization. Post layout parasitic parameters were extracted and a Verilog netlist together with the post layout SDF file were exported. This Verilog netlist was again simulated using the same VHDL test bench that was used for the behavioral model. This simulation confirmed the correct operation and forms the basis for the latency and throughput measurements reported in the following Section.

5. Results and Comparison

For measuring the performance of the asynchronous circuits the various handshake signals are generated from the test bench using an ideal data source and an ideal data sink. The input data is supplied in the test bench from a file. For the asynchronous mode of operation, data is being supplied to the entity with the rising edge of input request signal 'req_a_top' shown in Figure 5. With the delay in the handshake circuit being inserted in the path of signal 'req_a_top', the acknowledge signal 'ack_a_top' goes high after the combinational logic block finishes it's computation. The local clock signal in each of the components is connected to signal 'ack_a' of each respective handshake circuit. Hence, with the rising transition of 'ack_a', the computed data is latched to the data registers. The latency for asynchronous operation is measured as the time interval from the rising edge of input signal 'req_a_top' to the rising edge of output signal 'req_b_top'. This measurement is carried out with an empty pipeline as an initial state and hence assumes operation in the data limited region as defined in [13].

The maximum throughput is measured as the number of data samples emerging at the output when the input is driven by an ideal source and the output is connected to an ideal data sink.

For the synchronous design, the latency is dependent on the clock frequency. The core area for all three designs is measured from the layout and reported in Table 1 together with power dissipation, latency and throughput.

Table 1. Performance comparison of three CORDIC implementations

	ASY-CORDIC	MIX-CORDIC		SYN-CORDIC
		Synch. mode	Asynch. mode	
Core area (in mm^2)	1.8	2.6		2.3
Power (in mW)	(16)	(9.4)	(7.2)	(9.8)
Latency (in ns)	370	444	320	500
Through-put (MS/s)	30	45	45	40

It is shown in Table 1 that the MIX-CORDIC design requires only 13% more silicon area than the SYN-CORDIC. A substantial advantage is the reduced latency. In particular for transmitting acknowledgement frames which have to be generated at a fixed time interval of 16 ms after reception of a frame, any reduction in latency in the datapath is welcome. The data throughput of MIX-CORDIC and SYN-CORDIC are similar. The difference of 45 MS/s versus 40 Ms/s is probably due to the particular layout. Furthermore, the level of manual interaction for the MIX-CORDIC design was higher, with buffers being explicitly inserted into every pipeline stage.

Circuit ASY-CORDIC has significantly smaller silicon area i.e. 22% less than SYN-CORDIC. This is partly due to the smaller size of transparent latches compared to edge triggered flip-flops. Furthermore, in contrast to SYN-CORDIC and MIX-CORDIC, the ASY-CORDIC design does not require any clock tree. The use of local interconnect only, results in considerable savings in silicon area. The higher latency of circuit ASY-CORDIC is due to the more complex handshake circuit for this design. The transparent phase of the latches requires coupling of consecutive stages that is reflected by additional delays.

The power dissipation figures given in Table 1 were derived from the synthesis tool. From previous experience these figures are very unreliable. The distinctly higher power dissipation of ASY-CORDIC may result from the Synopsys software falsely estimating high glitching activity in the downstream circuits during the transparent phase of the latches. Currently work is underway to generate more accurate figures using the power simulator 'Power Mill'.

The layout of the MIX-CORDIC processor, fitted with analog test circuits for crosstalk investigation, is shown in Figure 6. In order to prevent direct coupling from input signals into the analog test circuits a C-shaped placement of the digital pads was undertaken. Four identical oscillators are placed at the east side of this layout, each fitted with a different structure for protection from substrate noise. From top to bottom the deployed protection structures are: 1) GND ring and n-well ring, 2) buried layer shield, 3) no protection, and 4) GND ring only. Additionally three substrate contacts for direct measurement of substrate noise are placed at top end, middle and bottom end of the right edge respectively. We hope that this design will deliver valuable information on the efficiency of different measures for protection from substrate noise, both for synchronous and asynchronous operation of the CORDIC processor.

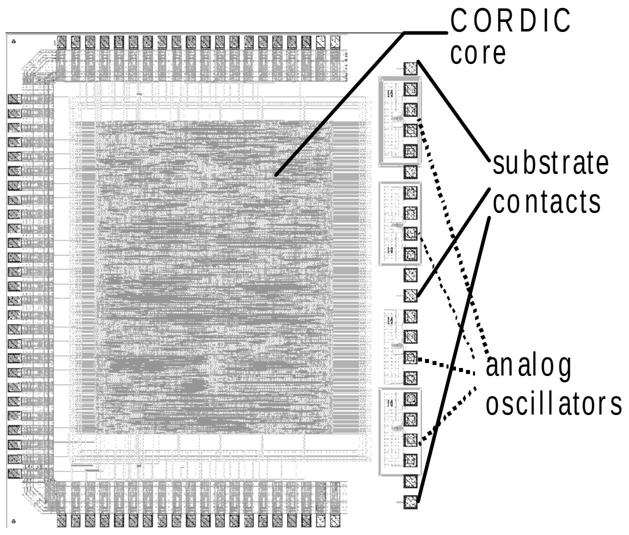

Figure 6. Layout of circuit MIX-CORDIC with analog test circuits at right side

6. Conclusions and Future Work

A pipelined CORDIC processor which can operate both synchronously as well as asynchronously has been designed in VHDL and implemented using a 0.25 mm SiGe:C BiCMOS process. Within a 'software defined radio' framework, the opportunity to operate key hardware components either in synchronous or asynchronous mode can offer additional freedom for the system developer. Using this feature, a more optimal compromise between power-dissipation, crosstalk, throughput, latency and (self-) testability can be found. As an example, to reduce crosstalk, the circuit can be switched into asynchronous mode when the receive signal level is very low. For reasonable signal strength and continuous operation the more power efficient synchronous mode can be chosen. An additional application of our MIX-CORDIC circuit is to serve as an vehicle for detailed investigating of the interference caused by synchronous and asynchronous digital circuits on SiGe:C bipolar and CMOS analog circuits.

Further work will focus on extending our cell library and design kit by a number of cells crucial for asynchronous design. Such cells are, for instance, various C-elements, complete handshake circuits and low-power tunable delay elements. The cells will be designed on transistor level, included into the cell library and made available as components in VHDL and Verilog.

The MIX-CORDIC design reported here, including the analog test circuits, has been taped out for fabrication on our in-house 0.25 mm SiGe:C BiCMOS process in December 2001. First results, including measurements on interference of the analog circuits for synchronous and asynchronous operation of the MIX-CORDIC design will be available by the middle of March 2002.

Acknowledgements

We would like to acknowledge the support from Ulrich Jagdhold for generating the digital layout.

Furthermore, we are grateful of the support from Wolfgang Winkler for design and layout of the analogue test circuits for the dual-mode CORDIC implementation.

References

[1] Standard ETSI DTS/BRAN 030003-1, “Broadband Radio Access Networks (BRAN); HIPERLAN Type 2 Functional Specifications. Part 1 – Physical (PHY) layer”; June 1999.

[2] Standard IEEE P802.11a/D7.0, “Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications: High Speed Physical Layer in the 5 GHz Band”; July 1999.

[3] J. M. Casalta, X. Arogones and A. Rubio, “Substrate Coupling evaluation in BiCMOS Technology”, IEEE Journal of Solid State Circuits, vol. 32, no.4, pp. 598 – 603, April 1997.

[4] J. Buck and E. A. Lee, “The token flow model”, Data Flow Workshop, Hamilton Island, Australia, pp. 267 – 290, May 1992.

[5] D. S. Cochran, “Algorithms and accuracy in the HP – 35”, Hewlett Packard J., pp. 10 –11, June 1972.

[6] A. M. Despain, “Very fast Fourier transform algorithms for implementation”, IEEE Trans. Comput., vol. C – 28, no. 5, pp. 333 – 341, May 1979.

[7] D. T. Lee and M. Morf, “Generalized CORDIC for digital signal processing”, Proc. ICASSP, vol. 3, pp. 1748 – 1751, May 1982.

[8] M. C. Mandal, A. S. Dhar and S. Banerjee, “Multiplierless array architecture for computing discrete cosine transform”, Computers Elec. Eng., vol. 21, no. 1, pp. 13 – 19, 1995.

[9] A. S. Dhar and S. Banerjee, “An array architecture for fast computation of discrete Hartley transform”, IEEE Trans. Circuits and Syst., vol. 38, no. 9, pp. 1095 – 1098, 1991.

[10] I. Sutherland, “Micropipelines”, Communications of the ACM, 32(6), pp. 720-738, June 1989.

[11] P. Day, J. V. Woods, “Investigation into Micropipeline Latch Design Styles”, IEEE Trans. On VLSI Systems, 3 (2), pp. 264-272, 1995.

[12] J. N. Seizovic, “Pipeline Synchronisation”, Proc. International Symposium on Advanced Research in Asynchronous Circuits and Systems, pp. 87-96, November 1994.

[13] T. E. Williams, “Performance of Iterative Computation in Self-Timed Rings” Journal of VLSI Signal Processing, (7), pp.17-31, 1994.