Micropipeline DSP-ASIC for a DS-SS receiver

Bengt Oelmann[1,2] and Hannu Tenhunen[1]

[1] Royal Institute of Technology
Electronic System Design Laboratory,
ELECTRUM 209, S-164 40 Kista, Sweden
Phone: +46-8-752 2100, Fax: +46-8-759 54 30
Email: bengt@ele.kth.se

[2] Mid-Sweden University
Department of Information Technology
S-851 70 Sundsvall, Sweden
Phone: +46-60-188939, Fax: +46-60-188830
Email: bengt@ele.kth.se

Abstract --- This paper presents a fully asynchronous DSP for a Direct-Sequence Spread Spectrum (DS-SS) radio receiver. Issues relevant to micropipeline VLSI implementation and performance are discussed. The circuit has been designed using a standard cell library in 0.8μm CMOS and contains 100.000 transistors. The receiver handles up to 48 million samples per second and the power consumption is 600 mW.

I. INTRODUCTION

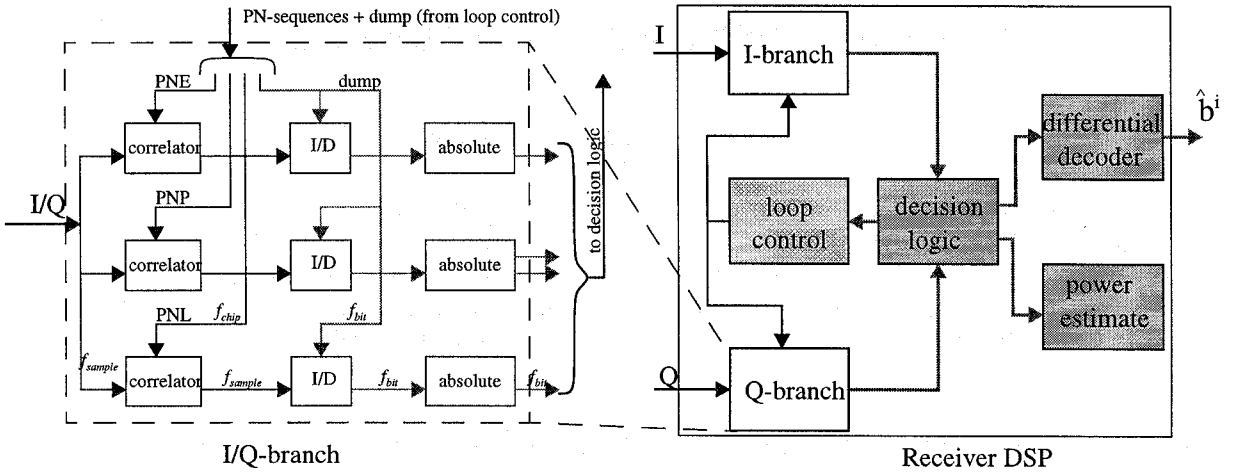
It is often stated that asynchronous circuits have several advantages compared to synchronous ones, e.g in [6]. When the IC process is down-scaled, delays in the logic gates will no longer be the limiting factor for the system performance. Gate delays are scaled with feature size, but the RC-delays in the interconnections remain approximately constant. This will give rise to two principal problems. First, it is hard to estimate the performance of the final system when the delays are dominated by interconnection delays. The top-down design approach will not be fully applicable since detailed knowledge about wire lengths must be known. The design must be taken down to the physical level in order to determine the performance. Second, the system relies on a global clock signal that must be distributed all over the chip. In order to achieve synchronism, a large design effort is required to fine tune the clock distribution net. This involves design work at the physical level. High clock frequencies demand large drivers that will dissipate a large part of the total power budget. Using asynchronous circuits is one proposed way to avoid problems with clock distribution and other global control signals. It makes it possible to handle all communication and synchronization at a local level and leads to circuits that scale better with the IC process used. In spite of this, the asynchronous design technique has not been widely spread. One reason for this is that it is quite different from the traditional synchronous design methods and often, available CAD-tools are not sufficient. Recently published complex asynchronous designs have been designed with special CAD-tools supporting the asynchronous methodology used, e.g [1,2,5]. Another reason is that it is not clear in which situations one can expect to gain performance by using asynchronous circuits. Extra logic is needed to handle the synchronization. At the local level this means that asynchronous circuits will be slower, dissipate more power and occupy larger area. The benefits are expected for larger circuits where the removal of the clock net will pay off.

In this paper we present the performance of an asynchronous DSP circuit for a Direct-Sequence Spread Spectrum (DS-SS) radio receiver. We base our circuit on the asynchronous design methodology *micropipelines*, presented by Sutherland in [8]. An industry standard cell library is used together with commercial available CAD-tools.

II. RADIO ARCHITECTURE

The asynchronous implementation of the receiver in this paper is based on a DS-SS receiver architecture presented in [4]. The main objectives here were to develop a digital radio interface suitable for a high degree of integration and with low power consumption. At the top of figure 1 an overview of the radio system is shown. The transmitter is sending a bit stream (b^i). The binary information is first multiplied with a 13-bit long pseudo-noise (PN) sequence and then modulated using differential binary phase shift keying (BPSK). At the receiver's side the signal is, after down-mixing in a quadrature mixer, A/D converted and fed in to the receiver DSP (Rx DSP). The DSP's tasks are synchronization and demodulation in order to deliver a bit stream ($\hat{b}^i$) to the end-user. In the block I/Q-branch, see figure 1, the received data are first correlated with three shifted versions of the PN-sequence and after that integrated over a time of one bit period ($1/f_{bit}$) in the integrate and dump unit (I/D). The output signals are used in the block *decision logic* for controlling both synchronization and detection. Different parts in the chip are running on different speeds. Thanks to the event-driven nature of asynchronous circuits the transition between these comes natural. The circuit is always running at the lowest possible speed to keep the power dissipation as low as possible. In figure 1 the different speeds are indicated with different shades of gray (dark gray means low speed).

0-7803-3302-0/96 $5.00 © 1996 IEEE



f_{sample}	sample rate	PNE = PN-sequence Early
$f_{bit}=f_{chip}/13$	bit rate	PNP = PN-sequence Punctual
$f_{chip}=f_{sample}/N$	chip rate	PNL = PN-sequence Late
N = 3.5	oversampling, times per chip	

Figure 1. Block diagram over the digital part of the receiver with the block I/Q-branch expanded. Sample rate signals are drawn with black lines and bit-rate signals are drawn with gray lines

III. DSP DESIGN USING MICROPIPELINES

In this design, the design methodology *micropipelines* [8] has been used. Here the communication among different modules is made using a two-phase transition signalling handshake protocol to control the data transfer. A correct data transfer is guaranteed by the bundled data convention. The bundled data interface consists of the data signals and two control signals, *request* and *acknowledge*, see figure 2. The sender informs the receiver that a new set of data is valid on its inputs with a transition on the request line. The receiver responds with a transition on the acknowledge line when accepting the data.

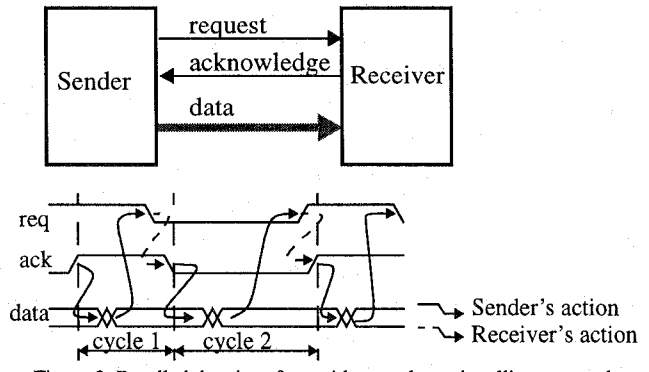


Figure 2. Bundled data interface with two-phase signalling protocol.

A micropipeline system is built from computational logic, event logic and latches that can act together with the two-phase protocol. The computational logic is the same as for synchronous circuits.

The event logic, that controls the data flow in the circuit, is built from a small number of pre-designed event logic cells. These basic cells are *event-OR* (XOR gate), *event-AND* (C-muller element), and *Selector*. More complex control is built from these basic cells. The latches we use are based on Sutherland's capture-pass latches [8].

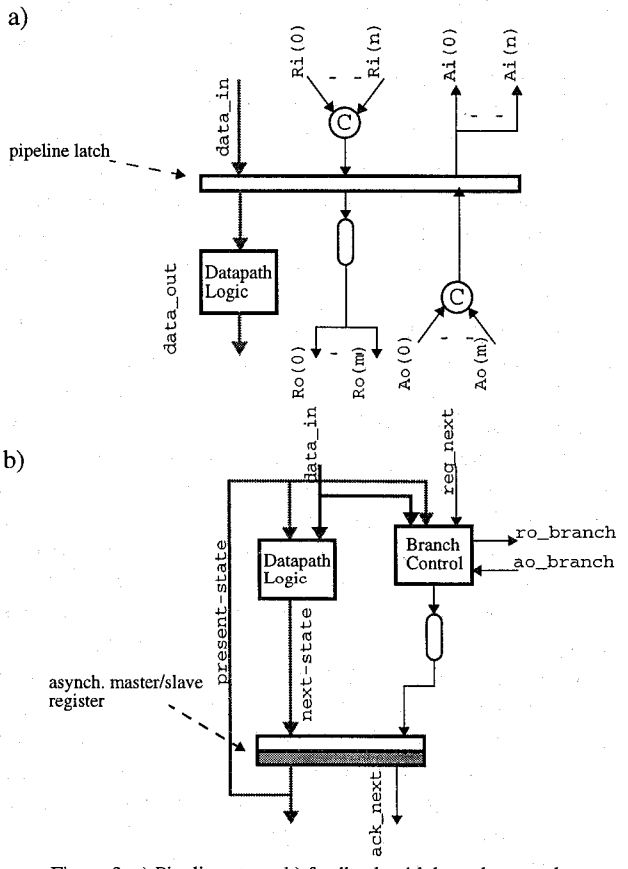


Figure 3. a) Pipeline stage, b) feedback with branch-control

The DSP is composed of two generic structures, a pipeline stage and a feedback structure. The data processing parts are built as straight micropipeline stages, see figure 3a. Asynchronous state-machines are built as figure 3b shows. In these we have the possibility to change the data rate. Computation in a side-branch is conditionally initiated with a transition on *ro_branch*. The state-machine is held in the same state until an acknowledge (*ao_branch*) has been received.

IV. VLSI IMPLEMENTATION

A traditional top-down/bottom-up approach was used. The design specification was captured in behavioural VHDL. It was then further refined down to a level where specific macro cells could be identified. The macro cells, that are composed of standard cells, come from a library designed for circuits in this application area [6]. The final layout was completed by automatic place and route tools.

The macro cells are built according the generic structures shown in figure 2. In table 1 below, a set of typical macro cells that are used is listed. The accumulator and the controller are of the type shown in figure 2b. In these the input data rate is higher than the output rate. The asynchronous control is conditionally directing the transitions to the outputs. In the case of the accumulator, the outputs are updated when a control bit tagged to the input data is zero. The 5-bit counter is a frequency divider. For a specific state in the controller the control transitions are directed to the output. The speed performance of the feedback structures is often limited by the asynchronous control path. In the accumulator, the speed is limited by the data path (16-bit carry look-ahead adder) while in the controller the speed is limited by the control path. In feedback structures with branch control the cycle time will be dominated by the delay in the asynchronous control when the propagation delay through the data path logic is comparable to the delay in a 16-bit CLA adder.

Table 1. Samples of macro cells

Type	Cell example	Cycle time [ns]	Transistor count	Area [mm^2]
Accumulator	16-bit integrate and dump	25	2882	0.659
Controller	5-bit counter	22	688	0.166
Shift register	16-bit shift reg.	15	1388	0.264
Write/read register	16-bit register	read: 8 write: 16	1720	0.333
Latch	16-bit latch	6	684	0.124
Data processing	16-bit adder	16	2424	0.433

V. FABRICATION AND MEASUREMENTS

The micropipeline receiver chip has been fabricated through Europractice MPW service, see figure 4. The circuit has been designed with a standard cell library in a 0.8 µm CMOS process from Austrian Mikro Systemes. The characteristics of the chip are summarized in table 2.

Table 2.

Max. sample rate @ 5V	48 MS/s
Max. data rate	1 Mbit/s
Chip area core/total	21 / 30 mm^2
Power @ 5V, 48 MS/s	600 mW
Number of transistors	97 645
Number of standard cells	18 700
IC Technology	0.8 µm CMOS
Package	40 pin DIL

The chip is working over a wide range of power supply voltages. The measurement results, shown in figure 5, have been carried out in 25C ambient temperature. The peak performance for the different power supply voltages is shown in figure 5a. Micropipelines are using bundled data signalling convention where a delay element has to be inserted in the control path in order to guarantee a correct operation. In the critical path, we choose to have a 30% delay margin for the delay element. This margin is directly affecting the maximum speed. Since the power consumption is directly proportional to the sample rate, see figure 5c, we plot the power consumption in units of tsample rate, see figure 5b.

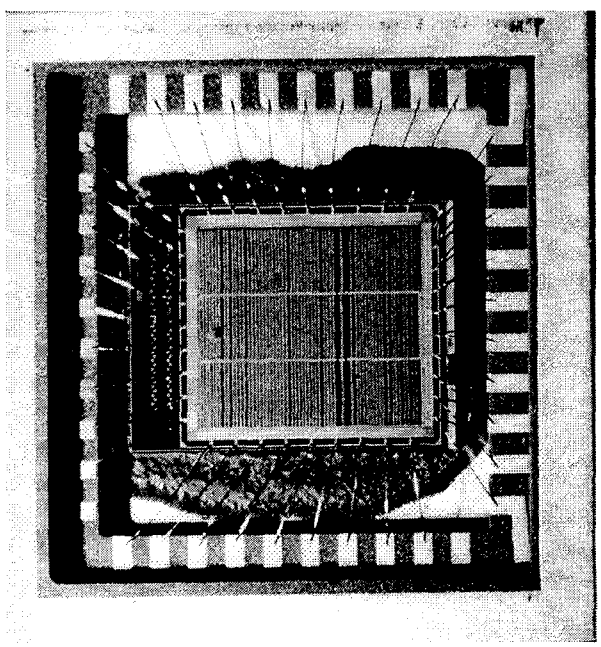


Figure 4. Die photograph of the fabricated chip

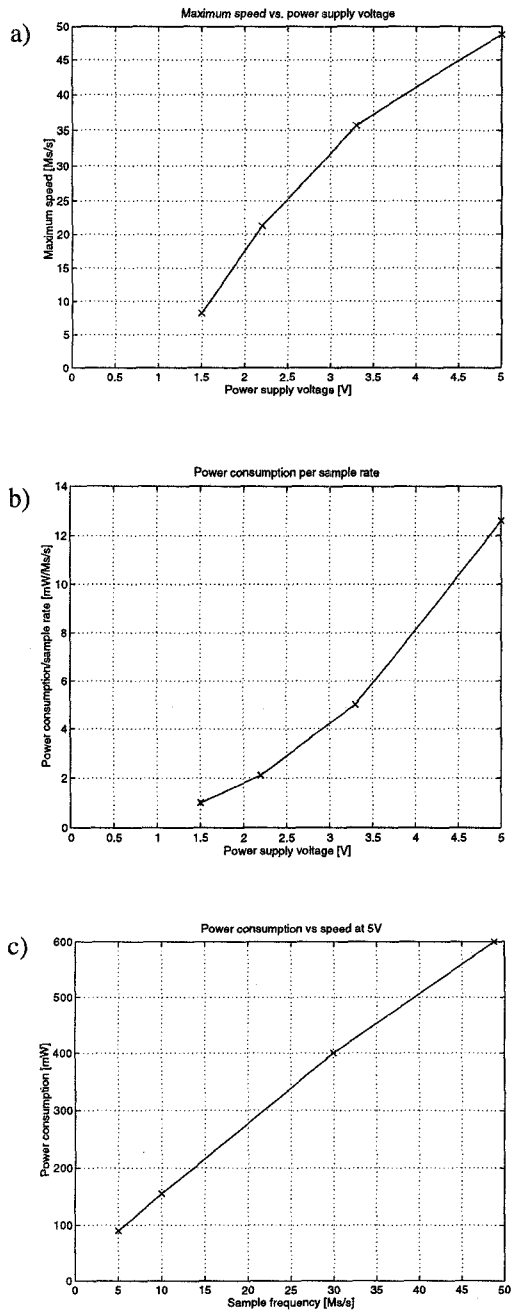


Figure 5. Measured performance of the circuit

The power consumption scales with both power supply voltage and speed with good agreement to the theory. The power consumption is proportional to the square of the power supply voltage. From the graphs in figure 5 it is possible to determine the lowest possible value for the power supply voltage for the actual speed requirement. Large power savings can be made by running at lowest possible voltage. If, for example, the speed requirement is 8MS/s the circuit could run at 1.5V with a power consumption of 12mW, instead of running at 5V with a power consumption of 100mW.

VI. CONCLUSIONS

A fully asynchronous DSP for a DS-SS receiver has been designed and fabricated. The micropipeline methodology enables us to design a complex asynchronous circuit with today's standard tools and technology. The absence of a global clock distribution net and an architecture that allows local interconnections exclusively, give us a design that can scale with improved technology. The measurements show that it is possible to design asynchronous circuits with reasonable good speed performance that operate over a wide range of power supply voltages. Compared to a synchronous solution, we cannot claim that the asynchronous receiver chip has better performance. The circuit overhead in the asynchronous circuit is too large. The capture-pass latches, that operate with the two-phase protocol, are quite area consuming. An asynchronous master-slave register is more than three times larger than a comparable synchronous implementation. In order to improve the speed performance, faster asynchronous control circuits are required. For the future work we will move from two-phase protocol to four-phase protocol that allows conventional pass-transistor latches or single-phase latches [3].

VII. REFERENCES

[1] Kees van Berkel, Ronan Burgess, Joep Kessels, Ad Peters, Marly Roncken, Frits Schalij, A Fully-Asynchronous Low-Power Error Corrector for the DCC Player, Proc. Int. Solid State Circuits Conference, February 1994, pp. 88-89.

[2] Bill Coates, Al Davis, and Ken Stevens, The Post Office Experience: Designing a Large Asynchronous Chip, Integration, the VLSI journal, vol. 15, no. 3, October 1993, pp. 341-366.

[3] Paul Day and J. Viv. Woods, Investigation into Micropipeline Latch Design Styles, IEEE Transactions on VLSI, vol. 3, no. 2, June 1995.

[4] Daniel Kerek, Henrik Olsson, and Hannu Tenhunen, Direct Sequence CDMA Technology and its applications to future portable multimedia, IEEE ISSSTA'94, Oulu, Finland, pp. 445-449.

[5] Alain J. Martin, Steven M. Burns, T.K. Lee, Drazen Borkovic, and Pieter J. Hazewindus, The Design of an Asynchronous Microprocessor, Advanced Research in VLSI: Proceedings of the Decennial CalTech Conference on VLSI, Editor: Charles L. Seitz, MIT Press, 1989.

[6] Teresa H. Meng, Synchronization Design for Digital Systems, Kluwer Academic Press 1991, ISBN: 0-7923-9128-4

[7] Bengt Oelmann, Henk Martijn, and Hannu Tenhunen, Design of an asynchronous macro cell library, Norchip conference '95, Copenhagen, Denmark, November 1995.

[8] I. E. Sutherland, Micropipelines, The 1988 Turing Award Lecture, Communications of the ACM, vol. 32, no. 6, June 1989, pp. 720-738.