Low Power and High Speed Adaptive OFDM System Using FPGA

Jatender Kumar Verma[1], K.K. Verma[2]
[1]Mtech Scholar, DPG Institute of technology & Management, Gurgaon
[2]Assistant Professor, DPG Institute of technology & Management, Gurgaon

Abstract—to achieve high spectrum efficiency [bps/Hz], parameters such as cyclic prefix, number of subcarriers, and constellation in an OFDM system have to adapt to the present state of the channel. Since a wireless channel continuously changes its state, adaptation must be performed in real-time. Hence, the physical layer of a mobile high performance OFDM transceiver must be fast, flexible, and energy efficient, i.e. an ASIC with run time flexibility is required. In this Paper it is shown that flexibility can be obtained with a reasonable amount of extra hardware. The flexibility will contribute to a larger set of possible applications and thus to the possibility of larger fabrication volumes and lower price per volume.
The proposed scheme has sacrificed backwards compatibility to existing standards in order to allow a higher degree of optimisation. However, the reduced latency and decreased hardware can be useful in future OFDM standards and applications. Particularly if the system includes real-time services, since real-time communication has an upper limit to the amount of latency that is accepted.

Keywords—OFDM System, ASIC, bps/Hz

I. INTRODUCTION

Analogue multi-carrier systems have been around since the 50's and the concept of orthogonal frequency division multiplexing (OFDM) with overlapping sub channel spectra was introduced by Chang in the mid 60's [1]. However, it was not until the end of the 90's that OFDM finally found its way into the public market, in wireless network cards for laptops and ADSL modems. In the near future most of our radio and television receivers will probably use the OFDM scheme. Digital audio broadcast (DAB) [2], will replace today's analogue radio broadcasting system in a couple of years (all major cities in Sweden are already covered). It has been announced that before the end of this decade the Swedish analogue TV-network will close down and be replaced with digital video broadcast (DVB) [3]. If we look further into the future, there is an ongoing discussion to include an OFDM transceiver in the fourth generation mobile system. With an OFDM transceiver the fourth generation mobiles can connect with a high data rate to an increasing number of hot spots, wireless local area networks that are installed in, e.g. coffee shops and offices. Another hot research topic involving OFDM is personal area networks (PAN), where the idea is that all your personal things like camera, smart cards, and medical sensors have the ability to communicate through a central unit, e.g. your personal digital assistant (PDA). The PDA will then act like a gateway to the outside world. In the European IST project PACWOMAN the OFDM scheme is chosen for the outgoing communication through the PAN gateway [4].

The great strength of OFDM is its spectrum efficiency [bps/Hz] and its ability to deal with multipath channels, i.e. the type of channels that appear in wireless environments. Since OFDM is computationally demanding and therefore power hungry, it was not until recently the technology made it possible to build mobile OFDM devices with an adequate operation time. Thus, with a large market for wireless devices operating in a multipath environment and the technology to build energy efficient devices, the time of OFDM has finally come.

II. OFDM SYSTEM

OFDM is a broadband multicarrier modulation method that offers superior performance and benefits over older, more traditional single-carrier modulation methods because it is a better fit with today's high-speed data requirements and operation in the UHF and microwave spectrum.
The implementations presented in later chapters mostly cover the transmitting side of an OFDM system. Forward error correction is not addressed in this document, since it is outside the scope of this project. To get a feeling for how much flexibility that is required in a flexible design, some existing OFDM standards are discussed and common parameters are pointed out.

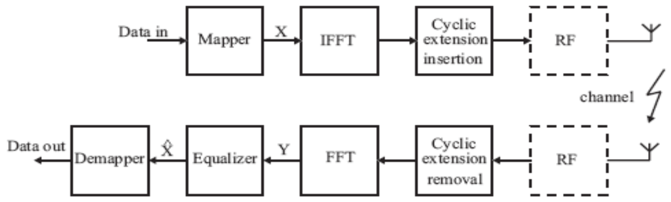

Fig.1: The digital baseband parts of an OFDM transceiver.

OFDM is based on the concept of frequency-division multiplexing (FDD), the method of transmitting multiple data streams over a common broadband medium. That medium could be radio spectrum, coax cable, twisted pair, or fiber-optic cable. Each data stream is modulated onto multiple adjacent carriers within the bandwidth of the medium, and all are transmitted simultaneously. A good example of such a system is cable TV, which transmits many parallel channels of video and audio over a single fiber-optic cable and coax cable.

The digital baseband parts of an OFDM transceiver [2], is shown in Figure 1. The basic idea of OFDM is to divide the available spectrum into N orthogonal subchannels. In the mapper, data is converted to signals located in the frequency domain where each subchannel is assigned one signal. The signals are then transformed to the time domain with the inverse fast Fourier transform (IFFT). The FFT is an efficient method to implement the DFT algorithm, based on a divide and conquer approach [3]. The implementation of an FFT from now on, only the FFT notation will be used. The last digital part of the transmitter inserts a cyclic extension to remove the effects of intersymbol interference (ISI) and interchannel interference (ICI).

The FFT is a variation of the discrete Fourier transform (DFT). Fourier, as you may remember from your college math days, was the Frenchman who discovered that any complex signal could be represented by a series of harmonically related sine waves all added together. He also developed the math to prove it. The math is difficult, and even early computers couldn’t perform it quickly. Cooley/Tukey developed the fast Fourier transform in the 1960s as a way to greatly speed up the math to make Fourier analysis more practical. In general, you can take any analog signal, digitize it in an analog-to-digital converter (ADC), and then take the resulting samples and put them through the FFT process. The result is essentially a digital version of a spectrum analysis of the signal. The FFT sorts all the signal components out into the individual sine-wave elements of specific frequencies and amplitudes—a mathematical spectrum analyzer of a sort. That makes the FFT a good way to separate out all the carriers of an OFDM signal.

III. MAPPING AND DEMAPPING

The mapper converts input data into complex valued signal points, according to a given signal constellation, e.g. BPSK, QAM, or 16-QAM, as shown in Figure 2 where the in-phase (I) axis corresponds to the real part and the quadrature (Q)axis corresponds to the imaginary part of the output signal. The amount of data transmitted on each subcarrier depends on the constellation, e.g. BPSK and 16-QAM transmit one and four data bits per subcarrier, respectively. The number of data bits that a constellation point corresponds to can be seen in Figure 2. For example, in the 16-QAM constellation there are four possible values on both the I and Q axis and thus two bits are needed to specify the location on each axis. The quality of the channel will decide which constellation to use. In a channel with high interference a small constellation like BPSK is favorable, since the required signal to noise ratio (SNR) in the receiver is low, whereas in interference free environments a larger constellation is more beneficial due to the higher bit rate. That larger constellations are more sensitive to noise is seen in Figure 2, since the distance between signals points decrease as the constellation grows, given that the average signal power is the same in all constellations. In the Hiperlan/2 and IEEE 802.11a standards, several constellations are included to provide the means to adapt to different environments, but the choice is restricted to a single constellation per OFDM frame. An even more efficient method, called bit-loading, allows each subcarrier to use a different constellation [4]. The constellation in a bit-loading algorithm is chosen based on the frequency response in each subchannel. A subchannel with high SNR will get a larger constellation and vice versa.

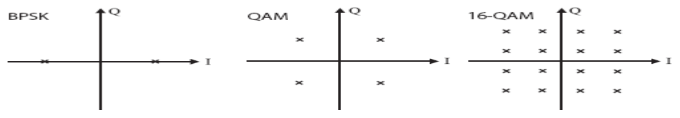

Fig.2: Typical signal constellations for wireless applications.

Demapping is performed to return from signal points to data bits. Since the received signal points are distorted by the channel, they will not be located at precisely the same point in the constellation diagram as they were transmitted. Hence, a decision has to be made about the origin of the received signal point; this decision is either hard or soft. With hard decisions the constellation point closest to the received point is estimated to be the transmitted one and the data bits are extracted accordingly. Simply put, soft decisions take the channel into account and base the output on how good or bad the channel is and how far from the closest constellation point the received point is. The extracted data is soft, i.e. can obtain values in between zero and one, for example strong zero, weak zero, weak one, and strong one. Soft data is used together with a decoder, e.g. a Viterbi decoder, and results in a coding gain of about 3 dB compared to hard decision data [15].

A. IFFT and FFT

The Inverse Fast Fourier Transform (IFFT) transforms signals from frequency domain to time domain and the FFT performs the reverse operation. To keep signals in the frequency domain simplifies some signal processing operations, e.g. convolution in the time domain becomes multiplication in the frequency domain. The transformation into the time domain is done in order to reduce the number of backend RF-oscillators and demodulators [5].

The available bandwidth (B) in frequency is split into N subchannels, one for each subcarrier, where each has a power spectrum shape of a squared sinc pulse. The individual power spectra, after the IFFT in the transmitter, of a number of subcarriers are shown in Figure 3. Since the IFFT is a linear operation the sub-carriers can be separated again with the FFT, even though there spectrum overlap. This is also true after the signals have passed a multipath channel, due to the cyclic extension. The property that each subcarrier is unaffected by the other subcarriers is called orthogonally, hence the name OFDM

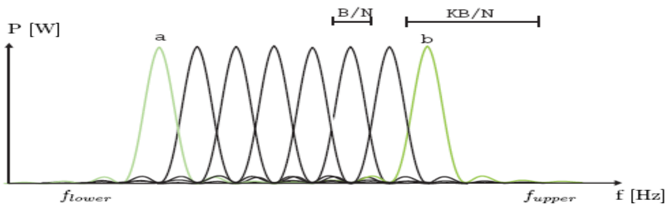

Fig.3: The power spectra of the individual subchannels in an OFDM signal.

The number of subcarriers that should be used is a trade-off between several factors. The more subcarriers used, the less overhead is introduced by the cyclic extension. In addition, with many subcarriers spectrum efficiency is higher since the two subcarriers at the outer edges, a and b in Figure3, contribute more than the central subcarriers to the width of the power spectra. Subcarrier a and b have a spectrum width of approximately KB/N [Hz], while all other subcarriers only have a width of approximately B/N [Hz]. The factor K depends on the upper and lower frequency bounds f_{lower} and f_{upper}, which are governed by national spectrum regulations.

B. Cyclic extension

There are two types of cyclic extensions, the cyclic prefix (CP) and the cyclic suffix (CS). The CP is a copy of the last n samples from the IFFT, which are placed at the beginning of the OFDM frame, as shown in Figure 4. There are two reasons to insert a CP and then discard it at the receiver. Assuming that the CP is longer than the channel impulse response, the convolution between the data and the channel impulse response will act like a circular convolution and therefore no ICI will occur. Moreover, interference from the previous symbol will only affect the CP, i.e. all ISI is avoided. However, if the number of samples in the CP is large, the data transmission rate will decrease, since the CP does not carry any useful data. The data rate will decrease with the factor R as R = N/(N + n). In the receiver the CP is discarded before the FFT, but can be used to support synchronization due to the correlation with the last part of the OFDM frame [8].

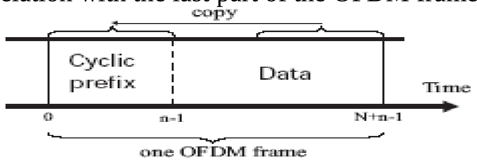

Fig.4: Cyclic prefix

The CS is a copy of the first n samples from the IFFT, which are placed in the end of the OFDM frame, as shown in Figure5. Just as in the case of a CP both ISI and ICI are avoided with the CS, but the input to the receiver FFT will be rotated. The CP is the most commonly used extension [19], but in some cases it is advantageous to use a cyclic suffix.

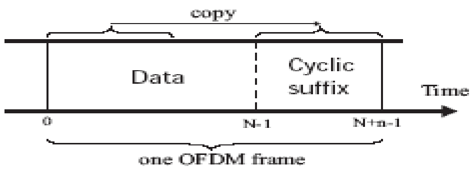

Figure5: Cyclic suffix.

IV. SYSTEM DESIGN

Design strategy is an important issue when designing a chip and becomes even more important when the number of design criteria increase. In this case, a broad spectrum of application areas requires a high flexibility, the ability to handle high as well as low speed, and all this with high power efficiency

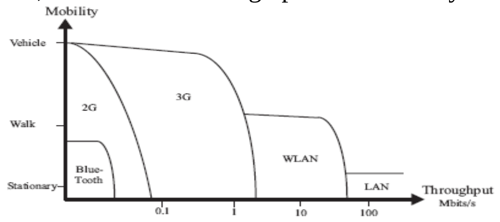

Fig.6: Throughput versus mobility in the most common systems.

In Figure6 an application area diagram for existing wireless systems is shown, a similar diagram for OFDM applications is shown in Figure7. There is a striking resemblance between the two figures: what is covered with several different systems in Figure 6 is covered with OFDM only in Fig8. Thus a flexible OFDM transceiver has the possibility to be used for all kinds of applications; one solution serves all. However, it might not be possible to cover all applications with one flexible design, but if it can be done the benefits are high enough to try.

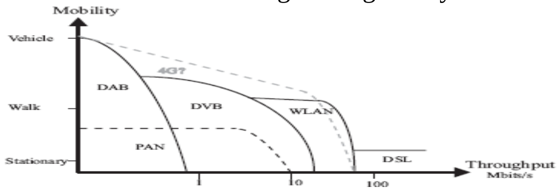

Fig.7: Throughput versus mobility in OFDM systems

In Figure8 the design space is shown. In the foreground is power, throughput, and area, and in the background, not as easy to see but always there, is money. The designer’s goal is to find the optimal point in the design space, one with enough throughputs for the application and minimum power and area requirements. However, an optimal search takes time and effort and therefore money and the money will often drain before the search is over. Hence, the designer most often settle for a ”good enough” point in the design space, a point that just about pass the requirements. When designing for flexibility we are not looking for the optimal point, but for the optimal line, as shown in Figure 9. The optimal line is the line that optimises power for all throughputs with a minimal area, i.e. minimize the angle a in

Figure 9. To find an optimal line is of course even harder than to find a perfect point, but herein lies the challenge.

A. Power aspects

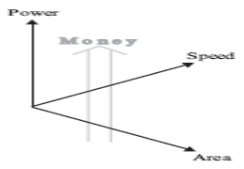

Fig.8: Design space.

The bandwidth (B) determines the operating speed and throughput for a system; large bandwidth corresponds to high throughput but requires high operating speed and thus high power consumption. In section 2.5 it was suggested that a flexible transceiver should be designed to handle bandwidths up to 100 MHz, a number higher than the existing OFDM standards use. Will the fact that a flexible device has to be designed for high speed prevent the device from operating power efficient at low speed? Fortunately not, as will be shown in this section. For more details about the following calculation see [2]

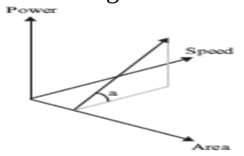

Fig.9: Optimal line in the design space for a flexible design.

Dynamic power consumption (Pd) for a CMOS design depends on the operating speed as

$$P_d \approx C_L f V_{dd}^2 \qquad (1)$$

Where C_L is the total capacitive load on the chip that is switched, Vdd is the operating voltage, and f is the clock frequency. For simplicity, it is assumed that the voltage swing is equal to Vdd.

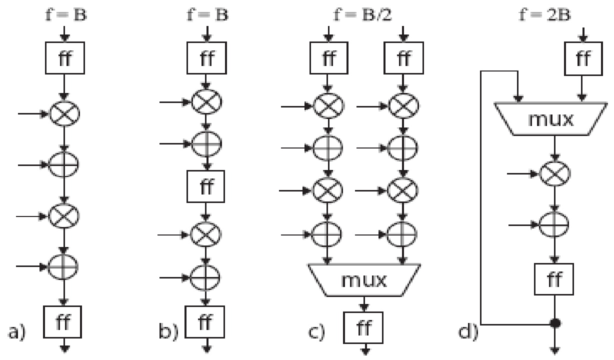

Fig.10: Hardware mapped (a), pipelined (b), parallel (c), and time shared (d) implementations.

The relationship between f and B depends on the hardware implementation. Choosing hardware mapped, time shared, pipelined, or parallel hardware will affect f as shown in Fig 10 [10]. The hardware mapped and pipelined design both produce one sample per clock cycle and thus have f equal to the desired B. The difference is that the pipelined design has a shorter critical path and can therefore reach a higher clock frequency. The parallel design produces two samples per clock cycle and can thus reduce the f to half that of B, while the time shared design has to double f to reach B. Comparing the hardware mapped and the parallel implementation it can at the first glance give the impression that there is nothing to gain from a power perspective by decreasing f, since C_L increase correspondingly.

B. Expression form

$$f = \frac{1}{t} \qquad (2)$$

and

$$t_{pd} \approx \frac{C_{critic} V_{dd}}{K (V_{dd} - V_t)^2}, \qquad (3)$$

Wheret_{pd} is the propagation delay, K is a constant, C_{critic} is the capacitive load in the critical path, and $V_t$is the threshold voltage. Both K and V_t depend on the silicon process. This means that a circuit designed for high speed, can when operated at low speed, reduce the power consumption below that of a circuit only designed for the low speed case.

The parallel design has the same throughput at approximately a quarter of the original design's power consumption and can, if there is need to, achieve doubled throughput at double power consumption. One kind of flexibility is achieved, but as always there is a price to pay, in this case the hardware is doubled. In addition, to design a device with active power control will add constraints on the design and additional verification time. A cell library that is characterized for multiple voltages is needed as well as hardware to generate the voltage levels and logic to control it. But it can be done, e.g. a commercial processor with active power control is found in [9]

In Figure11 an example is shown, to the left is the original design and to the right is a parallelized version of the same design. Since the parallel design produces two samples each clock period, the frequency can be halved and still achieve the same throughput as the original design. If equation 1 and 3 are used, and for simplicity V_T is assumed to be much smaller than Vdd, the dynamic power consumption is calculated as

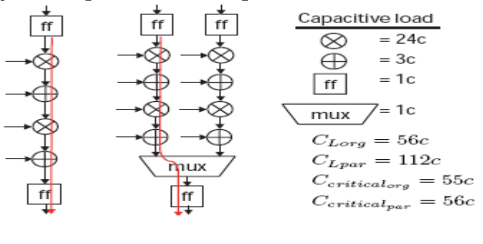

Fig.11: Original and parallelized design.

$$f_{par} = \frac{f_{org}}{2} \qquad (4)$$

Equation 2 and 4 gives

$$t_{par} = 2t_{org} \qquad (5)$$

Where, with the values from Figure11

$$t_{par} \approx \frac{C_{critic_{par}} V_{dd}}{K(V_{dd} - V_t)^2} \approx [V_t << V_{dd}] \approx \frac{C_{critic_{par}}}{KV_{par}} = \frac{56c}{KV_{par}} \qquad (6)$$

Substituting Equation 3.6 and 3.7 into 3.5 yields

$$t_{org} \approx \frac{C_{critic_{org}} V_{dd}}{K(V_{dd} - V_t)^2} \approx [V_t << V_{dd}] \approx \frac{C_{critic_{org}}}{KV_{org}} = \frac{55c}{KV_{org}}. \qquad (7)$$

And finally, using Equation 1

$$\frac{56c}{KV_{par}} = \frac{2 * 55c}{KV_{org}} \Rightarrow V_{par} = \frac{V_{org}}{1.96}, \qquad (8)$$

Note that the approximation that Vt is much smaller than VDD will be more and more rough as the silicon processes shrink, since Vt do not scale down as much as VDD. The power savings will not be as large as in the example above if a small process is used, e.g. 0.13 µm or below.

C. Clock gates

When designing for flexibility there is a great chance that the design will include hardware that is only used at certain times or in certain operation modes. These blocks will consume switching power and contribute with unnecessary load to the clock tree, even though the result is unused. One simple way to deal with this problem is to gate the clocks into these blocks, i.e. turn off the clock. In addition to reducing the clock load it will prevent switching activity, since almost all hardware modules have registers at the input. In addition to this high level clock gating, there are tools that can perform automatic low level clock gating, e.g. Synopsys: Power Compiler. The tool can automatically find small parts of the design, e.g. registers that can be turned off for part of the execution time. Unlike the high level gating no extra control signals are needed, since all control logic is inserted by the tool.

A hardware implementation of a clock gate is shown in Figure12a. With this design timing is not critical as shown in Figure 12b. To turn off the gated clock for one clock cycle, the Enable signal is lowered at any time in the clock cycle and then raised again in the following clock cycle.

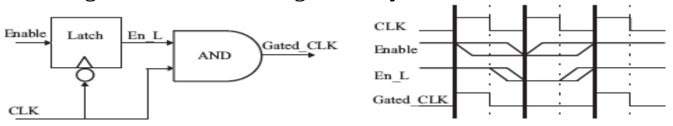

Fig.12: A clock gate (a) and the timing diagram (b).

V. FAST FOURIER TRANSFORM

The first designed chip is an FFT processor. The FFT processor has a central position both in the OFDM transmitter and receiver. The FFT is a computationally demanding operation that requires an ASIC implementation to reach high performance, i.e. high throughput combined with low energy consumption.

A. Architecture

The FFT and IFFT Equation 9 and 10 has the property that, if

FFT(Re(xi)+ jIm(xi)) = Re(Xi)+ jIm(Xi)……..(9) and

IFFT(Re(Xi)+ jIm(Xi)) = Re(xi)+ jIm(xi)…..(10),

Where xi and Xi are N words long sequences of complex valued, samples and sub-carriers respectively, then

1/N * FFT(Im(Xi)+ jRe(Xi)) = Im(xi)+ jRe(xi).

Thus, it is only necessary to discuss and implement the FFT equation. To calculate the inverse transform, the real and imaginary part of the input and output are swapped. Since N is a power of two, scaling with 1/N is the same as right shift the binary word Log_2 (N) bits. Even simpler, is to just remember that the binary point has moved log_2 (N) bits to the left. Not performing the bit shift until, if ever, it is necessary, which depends on how the output from the IFFT will be used.

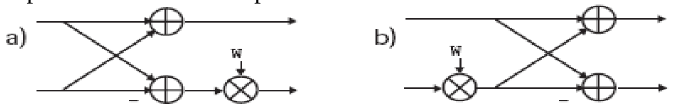

Fig.13: A radix-2 DIF butterfly (a) and a radix-2 DIT butterfly (b), where W is the twiddle factor.

The FFT algorithm can be realized with a butterfly operation as the basic building block [3]. There are two types of butterfly operations, decimation in time (DIT) and decimation infrequency (DIF), both are shown in Figure 13.

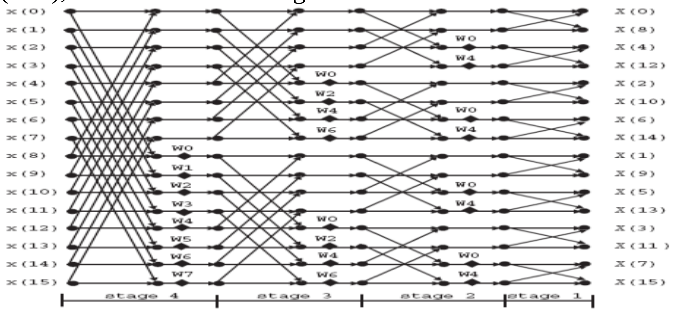

Fig.14: A hardware mapped N = 16-point radix-2 DIF FFT algorithm.

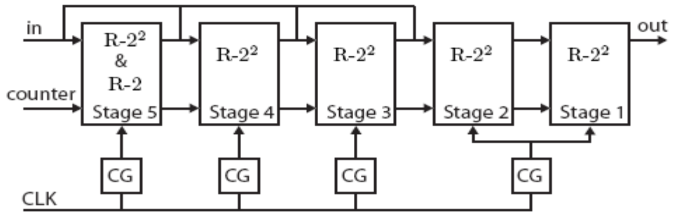

Fig.15: FFT architecture, where CG is a clock gate.

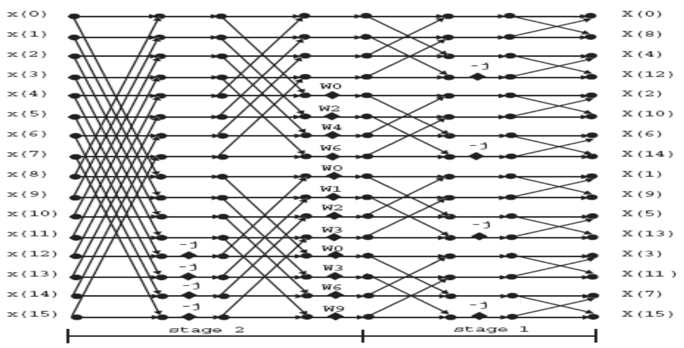

Fig.16: hardware mapped N = 16-point radix-22 DIF FFT algorithm.

VI. SIMULATION RESULTS

One of the more important decisions designing a fixed-point pipelined FFT processor regards the word length in the architecture, since it will affect the precision, number of gates, and power consumption. The most straightforward implementation is to have the same word length throughout the whole FFT processor. However, this will give a poor performance per Kbit memory since the data has to be shifted down before each butterfly, to avoid overflow. As there are ten butterflies in a 1024 point FFT, the ten least significant bits are lost if a fixed word length data path is used.

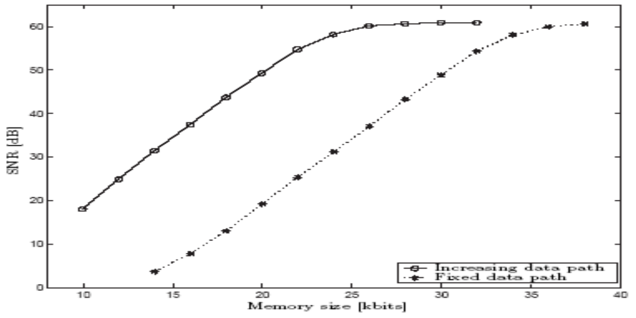

Fig.17: SNR versus memory size, for a 1024 point FFT with increasing and fixed data path.

That the input stages, in a FFT with bit reversed output, contain the largest FIFOs can be used to find a better solution. If the word length is reduced at the input and allowed to increase towards the output, the required memory as well as power consumption and area will decrease. That the data path will be wide close to the output will not affect the size of the memory that much, since the last FIFOs are short; in fact, the last one contains only one word. Allowing the data path to increase with one bit in each butterfly, i.e., no precision is lost in the butterfly, compensates for the loss in Signal to Noise Ratio (SNR) due to the reduced input word length [7]. All noise is quantization noise, due to limited word lengths.

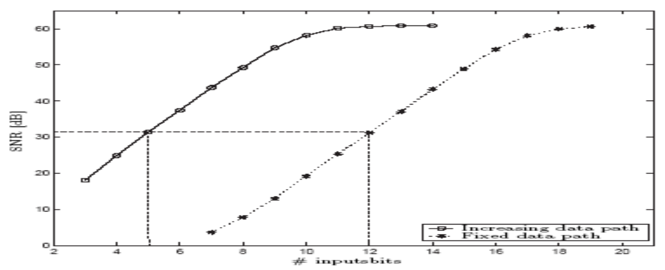

Fig.18: SNR versus number of input bits, for a 1024 point FFT with increasing and fixed data path.

Figure17 and 18 shows SNR versus memory size and number of input bits for a 1024 point FFT with a fixed and an increasing data path. Figure 4.6 shows that, with the same amount of memory, the increasing data path FFT perform about 30 dB better than a fixed data path. In Figure 4.7 it is seen that the word length in the memory critical first stage of the FFT processor has to be 7 bits wider with a fixed data path compared to the increasing path to reach an SNR of 31 dB, as the dashed lines show. The values are obtained from a C-model of a fixed and an increasing data path FFT compared to a 64 bits floating point FFT. The input data, the real and imaginary part, has an upper and lower limit of ±1/ 2, i.e. maximum absolute value of the input is one, to ensure that no overflow occur in the complex multiplier. To multiply with the twiddle factor is the same as rotating the data in phase and thus: to restrict input data to have an absolute value below one will guarantee that output from the complex multiplier has an absolute value below one. The twiddle factors used in this model are 12 bits, which puts the upper limit of the SNR to about 60 dB as seen in Figure 17 and 18.

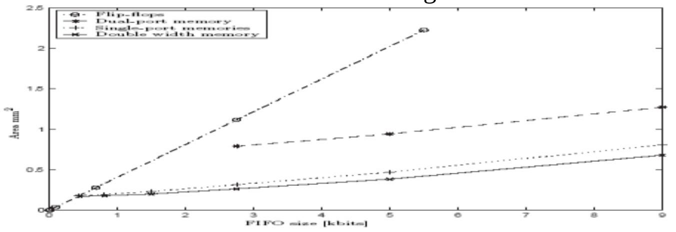

Fig.19: Area versus FIFO size for different implementations.

The designed FFT has 8 input bits (real and imaginary part is 8 bits each) and the width of the data path is increasing, resulting in an 18 bits wide output. The processor has an SNR of 49 dB and uses 20 kbits of memory. The word length is chosen to handle signal constellations up to 64 points, i.e. 64-QAM. The input word consists of 3 bits to represent the constellation, 3 bits for soft decoding, 1 bit to ensure that the data is between ±1/ 2, and 1 bit noise and interference margin [8]. How to implement the FIFOs is another important issue. As there is one read and one write operation in each clock cycle, the FIFO could be implemented in at least four different ways:

1. Flip-flops connected in series: This is an easy solution since there is no need of control logic. In addition flip-flops have a fast access time. However, they do occupy a large area even for small FIFOs.

2. One dual-port memory:A dual-port memory is smaller than flip-flops but is still more than two times larger than two single port memories and needs control logic.
3. Two single-port memories of half length, with alternating read/write:This solution is smaller than a dual port memory but has twice as many memories to place and route.
4. One single-port memory of half-length and double width, which reads and writes every other clock cycle. This is the smallest solution for large FIFOs.

Figure19 shows estimated area for different implementations of FIFOs in a 0.35 µm CMOS technology. There are no interconnections included in the flip-flop area. All memory FIFOs includes control logic and a 50 µm power ring on three sides of the block. In the designed FFT processor the FIFOs were chosen according to Figure19. All FIFOs larger than 8 words, i.e. stage 3 to 5 in Figure17, were implemented as single-port memories of double width and the remaining FIFOs were implemented with flip-flops.

VII. Conclusion

A pipelined FFT processor has been implemented in a standard CMOS 0.35 µm technology with five metal layers. The processor was estimated to compute a 1024 point FFT in less than 13 µs, with a clock frequency of 83 MHz. The FFT processor is resizable between 32-1024 points and unused blocks are deactivated with clock gates. The designed processor reaches an SNR of 49 dB with 8 input bits for a 1024 point FFT. Figure 4.9 shows the FFT processor chip. The chip has 84 pins, three twiddle factor ROMs, and 6 RAMs to implement the FIFOs. The core area is 4.94 mm.

Table 4.1: A comparison between 1024 point FFTs.

Design	Voltage [V]	Frequency [MHz]	Power [mW]
This FFT design	2.0	25	76
Low power FFT	1.5	25	200

Figure 21 shows the power consumption in the core when the chip is operating at 20 MHz. In Figure 4.11 the core power consumption is shown as a function of frequency at a core voltage of 2 V. The FFT processor functioned in all modes up to 50 MHz at 2 V. Since the test equipment did not support any frequencies in between 50 and 100 MHz, it was not
Figure 4.10 shows the power consumption in the core when the chip is operating at 20 MHz. In Figure 4.11 the core power consumption is shown as a function of frequency at a core voltage of 2 V. The FFT processor functioned in all modes up to 50 MHz at 2 V. Since the test equipment did not support any frequencies in between 50 and 100 MHz, it was not possible to verify the estimated max frequency of 83 MHz In Table 1 this design is compared to another pipelined FFT, implemented in the same technology [6]. As seen the presented design consumes less than half the power at the same frequency. Two reasons for this can be found. One, a radix-4 single-path delay communicator is used, which requires 2N words of memory, twice as much as the radix-22 architecture [9]. Secondly, no low power memories are used.

The reason for errors below 2 V is memory related, since the chip functions for frequencies above 20 MHz at 2 V, but not for voltages below 2 V at 20 MHz, as seen when comparing Figure21 and22. If the errors were due to timing at voltages below 2 V the design would not function at frequencies higher than 20 MHz, since both increasing frequency and decreasing voltage results in harder timing constraints, as shown in Equation 3. Another indication is that the only mode that can operate at 1.8 V is the 64-point FFT, which is the only mode that operates without any RAM FIFOs.

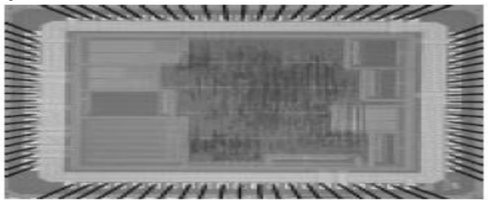
Fig.20: The FFT processor chip.

Another observation is that the 64-point FFT consumes least power, not the 32-point FFT which could be expected. Looking closer it can also be seen that the 256-point FFT consumes less power than the 128-point. One explanation could be that the radix-2 part, involved in the 32 and 128-point FFT, always uses the 256word long FIFO, the second FIFO in Figure 16 (stage = 5), instead of the smallest disabled FIFO. This can however not explain all the difference, since a memory's power consumption is almost as dependent on word length as on the number of words; as the word length increases through the FFT processor and thus, to a large extent, cancels the power saving from using shorter FIFOs. The explanation is that, although the first FIFO in stage-5 is not provided with data when the radix-2 part is used, the clock is still active as there is only one clock gate for the complete stage-5. The solution for future designs is to power down the first FIFO, in stage 5, when the radix-2 part is used. This is easily done with the memory's chip-select signal and would result in a 10-12 mW power reduction, at 2 V and 20 MHz, for the 32, 128, and 512-point FFT.

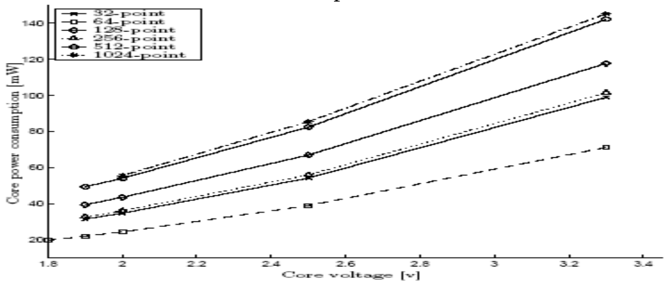

Fig.21: Voltage scaling, with frequency = 20 MHz.

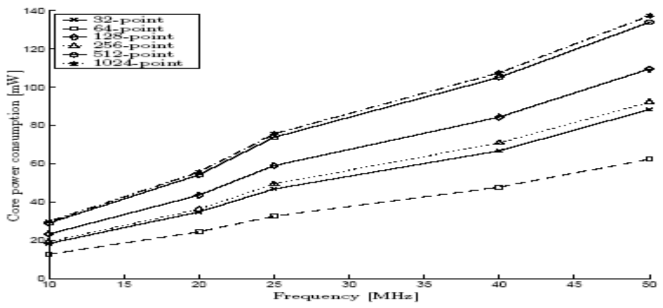

Fig.22: Frequency scaling, with core voltage = 2 V.

VIII. REFERENCES

[1] R. W. Chang, "Synthesis of Band-Limited Orthogonal Signals for Multichannel Data Transmission," Bell System Tech. J., vol. 45, pp. 1775–1796, Dec. 1966.

[2] ETS 300 401, ETSI, "Digital Audio Broadcasting (DAB);DAB to mobile, portable and fixed receivers," 1995.

[3] ETSI EN 300 744, "Digital video broadcasting (DVB);framing structure, channel coding, and modulation for digital terrestrial television," 2001.

[4] European IST Project, "Power Aware Communications for Wireless OptiMised personal Area Networks (PACWOMAN)," http://www.imec.be/pacwoman/.

[5] S. B. Weinstein and P. M. Ebert, "Data transmission by frequency division multiplexing using the discrete Fourier transform," IEEE Transactions on Communications, vol. 19, pp. 628–634, Oct. 1971.

[6] A. Peled and A. Ruiz, "Frequency domain data transmission using reduced computational complexity algorithms," in Int. Conf. Acoustic, Speech, Signal Processing, Denver, CO, 1980, pp. 964–967.

[7] L. J. Cimini, "Analysis and Simulation of a Digital Mobile Channel Using Orthogonal Frequency Division Multiplexing," IEEE Transactions on Communications, vol. 33, pp. 665–675, July 1985.

[8] ETSI TS 101 475, "Broadband Radio Access Networks (BRAN); HIPERLAN Type 2 Physical (PHY) layer, v1.1.1," 2000, http://portal.etsi.org/bran/.

[9] IEEEstd 802.11a, "High-speed Physical Layer in 5 GHz Band," 1999, http://ieee802.org/.

[10] IEEE std 802.11g, "High-speed Physical Layer in 2.4 GHz Band," 2003, http://ieee802.org/