Orthogonal Code Generator for 3G Wireless Transceivers

Boris D. Andreev, Edward L. Titlebaum, and Eby G. Friedman

Department of Electrical and Computer Engineering
University of Rochester
Rochester, New York 14627
{bandreev, tbaum, friedman} @ece.rochester.edu

ABSTRACT

Orthogonal variable spreading factor (OVSF) codes are standard in third generation UMTS cellular systems. The efficient generation of these codes is essential for reducing the area and power of wireless transceivers. In this paper, the basic properties of this family of codes are analyzed from an RTL perspective and two efficient hardware code generators are proposed. Tradeoffs and design solutions as well as low power considerations are discussed. These results represent the first reported implementation of an OVSF code generator.

Categories and Subject Descriptors

B.5.1 [**RTL Implementation**] Design: *Arithmetic and Logic Units*
B.6.0 [**Logic Design**] *General*
B.7.m [**Integrated circuits**] *Miscellaneous*

General Terms: Design

Keywords: OVSF codes, CDMA, VLSI, 3GPP, UMTS, WCDMA

1. INTRODUCTION

Modern CDMA cellular systems employ spread spectrum technology to provide multiuser access. In particular, direct sequence spread spectrum has been adopted for improved spectral efficiency, ease of digital implementation, and soft capacity limit. Each user employs a noiselike wideband signal occupying the entire allocated frequency band for as long as necessary. In this way, each user contributes to the background noise affecting all other users. This additional interference limits the overall system capacity but because time and bandwidth resources are virtually unlimited, the resulting capacity is significantly greater than in conventional cellular systems. In the UMTS (Universal Mobile Telecommunication System) wireless standard, part of the Third Generation Partnership Project (3GPP), the spectrum spreading applied to the symbols in the physical channels consists of two operations [1]. The first is the *channelization* operation, which transforms every data symbol into a number of *chips*, thereby increasing the signal bandwidth. The number of chips per data symbol is called the *spreading factor* (SF). The channelization OVSF codes achieve orthogonality between a user's different physical channels. The second operation is the *scrambling* operation, where a scrambling code is applied to the spread signal in order to distinguish signals from asynchronous users. With channelization, data symbols on the in-phase (I) and quadrature-phase (Q) branches are independently multiplied with an orthogonal variable spreading factor (OVSF) code. One control channel and up to six data channels can be simultaneously transmitted. The real-valued chip streams on the I- and Q-branches are combined and treated as a complex-valued stream of chips. With the scrambling operation, the resultant complex signal is multiplied by a complex-valued scrambling code.

An overview of the 3GPP standard defining the OVSF codes is presented in section 2. The generation of OVSF codes for 3GPP systems is considered from an RTL perspective in section 3. Two alternative techniques for designing a hardware OVSF code generator are proposed in sections 4 and 5. The first technique, described in section 4, refers to a standalone logic circuit achieving the target function. A more efficient approach is discussed in section 5, which requires, however, a change in the system software to support a logic circuit of reduced complexity. Power dissipation issues are discussed in section 6. Results from the synthesis of the proposed code generator are reported in section 7 and some related system level issues are discussed in section 8. Final remarks are offered in section 9 to conclude the paper.

2. OVSF CODES IN THE 3GPP STANDARD

OVSF codes are defined in the 3GPP standard [1] by the code tree shown in Fig. 1. A channelization code $C_{ch,SF,N}$ is uniquely described by two numbers: the spreading factor SF in the range [4 - 512] = [2^2 - 2^9], and the identification (ID) number $N \in [0, SF - 1]$. Each level in the code tree defines channelization codes of length SF as shown in Fig. 1.

The control channel is spread by code $C_{256,0}$, which consists of 256 logic zeros. When only one data channel is transmitted, *data channel*$_1$ is spread by code $C_{SF,k}$ where SF is the spreading factor and $k = SF / 4$. When more than one data channel is transmitted, all of the data channels have a spreading factor equal to 4. *Data channel*$_n$ is spread by the code $C_{4,k}$, where $k = 1$ if $n \in \{1, 2\}$, $k = 3$ if $n \in \{3, 4\}$, and $k = 2$ if $n \in \{5, 6\}$.

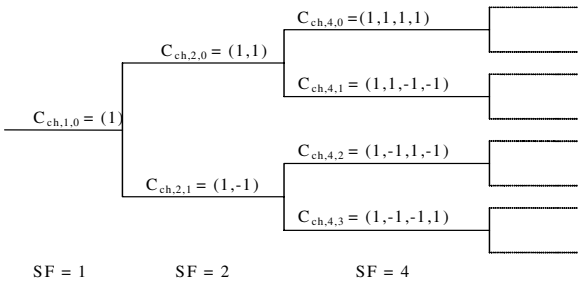

Fig. 1: Code tree of Orthogonal Variable Spreading Factor (OVSF) codes [1]

* This research is supported in part by the Semiconductor Research Corporation under Contract No. 99-TJ-687, the DARPA/ITO under AFRL Contract F29601-00-K-0182, grants from the New York State Office of Science, Technology & Academic Research to the Center for Advanced Technology – Electronic Imaging Systems and to the Microelectronics Design Center, and by grants from Xerox Corporation, IBM Corporation, Intel Corporation, Lucent Technologies Corporation, Eastman Kodak Company, and Photon Vision Systems, Inc.

GLSVLSI'03, April 28-29, Washington, DC, USA.

Table1: Generation of channelization code $C_{8,5}$

	First generated chip							Last generated chip
$C_{8,5}$ =	0	1	0	1	1	0	1	0
	↑ $n_3 = 0$ always	↑ $n_3 \oplus n_2 = n_2$	↑ $n_3 \oplus n_1 = n_1$	↑ $n_3 \oplus n_2 \oplus n_1 = n_2 \oplus n_1$	↑ $n_3 \oplus n_0 = n_0$	↑ $n_3 \oplus n_2 \oplus n_0 = n_2 \oplus n_0$	↑ $n_3 \oplus n_1 \oplus n_0 = n_1 \oplus n_0$	↑ $n_3 \oplus n_2 \oplus n_1 \oplus n_0 = n_2 \oplus n_1 \oplus n_0$

3. RTL PERSPECTIVE ON OVSF CODES

The generation of the OVSF channelization codes is defined in the 3GPP standard [1] by three matrix expressions. The direct implementation of these matrix operations requires a significant amount of resources which is convenient only for a software realization based on a microcontroller or a digital signal processor. In order to design an efficient hardware generator for this family of codes, specific properties of OVSF codes are discussed in this section.

The chip sequence is specified in the binary set {+1, –1}, while digital CMOS logic operates on the set {0, 1}. The mapping {"+1" → "logic 0"} and {"–1" → "logic 1"} is therefore adopted as a convention. The spreading codes serve as control signals for a complex ±1 multiplier [2]. In the remainder of the paper, circuits issues are discussed with logic levels of 0 and 1.

The binary representation of the ID number describes the trajectory of the code generation along the OVSF code tree shown in Fig. 1: a logic 0 for the upper branch or a logic 1 for the lower branch. This property is illustrated by the following example:

Example: Suppose the channelization code $C_{8,5}$ is required.

$$SF = 8_{10} = 1000_2$$
$$N = 5_{10} = 0101_2 = n_3n_2n_1n_0$$

The trajectory of code $C_{8,5}$ is defined as follows:

Stage 1 – a single root, always 0
Stage 2 – lower branch, controlled by n_2
Stage 3 – upper branch, controlled by n_1
Stage 4 – lower branch, controlled by n_0

The individual code chips are controlled by corresponding bits in the ID number, as listed in Table 1. The OVSF chips can therefore be produced by a XOR operation over certain bits of the code ID number. The participation of a specific bit in the XOR operation is periodic in time and can be controlled by a binary counter as listed in Table 2.

Based on observations of the data listed in Tables 1 and 2, a logic level architecture of an OVSF code generator is shown in Fig. 2. The least significant bit (LSB) of the counter enables the MSB of N, bit n_2, to be included in the XOR operation. The MSB of the counter controls n_0, which is the LSB of N.

The 3GPP standard, however, specifies that the code generator should be capable of producing codes with a variable spreading factor over the range SF = 4 to 512. The issue is how best to match the number N and the binary counter in reverse order for different spreading factors. Two techniques are suggested to accomplish this objective:

Table 2: Time counter and XOR operations producing the OVSF code chips

Binary counter	Operation
000	0
001	n_2
010	n_1
011	$n_2 \oplus n_1$
100	n_0
101	$n_2 \oplus n_0$
110	$n_1 \oplus n_0$
111	$n_2 \oplus n_1 \oplus n_0$

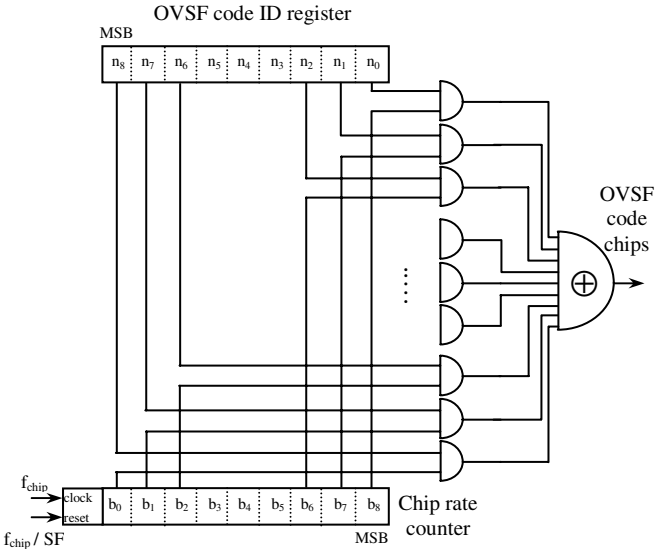

Fig. 2: OVSF code generator

1. Design a special counter, counting incrementally from 0 to $SF - 1$. b_8 is always the MSB, while the LSB is specified by the variable spreading factor SF and is b_x, where $x = \log_2 SF_{max} - \log_2 SF$. In the aforementioned example, the counter has six dummy bits (always at 0) and three active bits. The MSB b_8 controls bit n_0 of the ID register, and bit b_6 is the LSB of the counter, which controls bit n_2.
2. Shift the code ID number such that the MSB of N is always enabled by the LSB of the counter. In this case, a regular binary counter is required to count between 0 and $SF_{max} - 1$.

In the following sections, the advantages and drawbacks of both techniques are described and corresponding logic circuits are proposed.

4. TECHNIQUE 1: DESIGN OF A SPECIAL COUNTER

The design of a special counter requires less additional hardware than a circuit that directly implements the proposed second scheme. An SF register, holding the required spreading factor, controls the counter cycle, while the ID register controls the specific OVSF code. This separation of the high level parameters *SF* and *N* provides modularity of the circuit blocks when multiple OVSF codes are generated. A circuit solution for the counter is proposed in Fig. 3. The LSB of the counter is specified by the only bit of the SF register, which is logic 1. The count is from left to right, where b_8 is always the MSB.

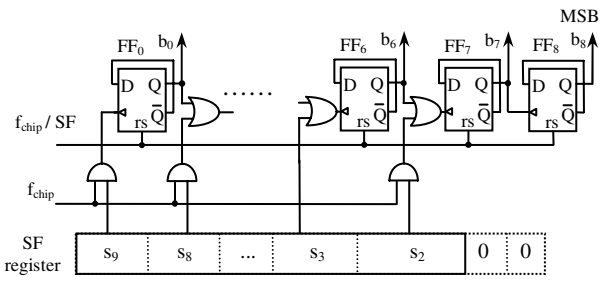

Fig. 3: Chip rate counter from 0 to SF – 1
with variable location of the LSB

The gate level schematics shown in Figs. 2 and 3 provide a general framework, amenable to logic optimization. The logic function can be preserved if all AND gates are replaced by more efficient (in CMOS) NAND gates. OR gates are replaced by NOR gates. The nine-input XOR tree is optimized for area, as the delay is not a critical factor for this application.

5. TECHNIQUE 2: MANAGING THE ID NUMBER

An alternative technique for realizing the variable spreading factor codes is to manage the code number loaded in the ID register. The MSB is always n_8 and controlled by the LSB of a regular binary counter 0 : (SF_{max} – 1). The code number is shifted such that the LSB is n_y, where $y = \log_2 SF_{max} - \log_2 SF$.

A significant amount of additional logic circuitry is required to produce a variable shift (dependent on the SF) of the code number before insertion into the ID register. The information characterizing the spreading factor and ID number of all of the required codes is determined at the system level. The problem of varying the spreading factors can be efficiently resolved by system software when saving the ID numbers in global memory. The logic shift operations can be conveniently performed in software. Once the ID numbers are saved in the appropriate format, the numbers are downloaded into the ID register. The circuit shown in Fig. 2 uses a regular binary counter. This approach is illustrated by a few examples shown in Fig. 4, which complements the circuit illustrated in Fig. 2.

The stored values represent information describing the spreading factor and code ID number required to generate the code. As shown in Fig. 4, these values do not define a single code. The

System memory	Generated OVSF codes
100000000	$C_{4,2}$ or $C_{8,4}$ or $C_{16,8}$ or …
110000000	$C_{4,3}$ or $C_{8,6}$ or $C_{16,12}$ or …
111000000	$C_{8,7}$ or $C_{16,14}$ or $C_{32,28}$ or …
101000000	$C_{8,5}$ or $C_{16,10}$ or $C_{32,20}$ or …
111100000	$C_{16,15}$ or $C_{32,30}$ or $C_{64,60}$ or …
…	

Code ID register (from Fig. 2)

n_8	n_7	n_6	n_5	n_4	n_3	n_2	n_1	n_0

Fig. 4: Realization of variable spreading factor
by shifting the ID code number at the system level

generated chip sequences are identical for codes $C_{SF,N}$, $C_{2SF,2N}$, $C_{4SF,4N}$, etc. These code sequences differ only by the length (SF, 2SF, 4SF, etc.), but this parameter is only important for the sampling moment of the despreading matched filter. The code generator continuously produces chips from the required codes.

6. POWER DISSIPATION

The power consumed by the circuit changes dynamically according to the OVSF code being generated. The fewer the nonzero bits in the ID register, the less activity in the entire circuit, thereby minimizing power. The effect of a variable counter cycle on the power consumption is described in this section. This analysis refers to the first technique (described in section 4 and illustrated in Figs. 2 and 3), where the counter cycle is controlled by a variable spreading factor.

The switching activity is maximum when the code $C_{SFmax,Nmax} = C_{512,511}$ is generated because the counter clock cycle is also maximum and all of the bits of the ID register are logic one. Alternatively, to generate $C_{4,2}$, $SF = 2^4$ and $N = 2$, the counter shifts from 0 to 3, and only one of the AND gates shown in Fig. 2 switches, consuming less power. Minimum power is consumed by the circuit when $C_{512,1}$ is generated. In this case, the single nonzero bit in the ID register is enabled by the most significant counter bit (with the lowest switching activity). Static leakage power is negligible in the target CMOS technologies and circuit densities.

The power consumed by the code generator is the sum of the power consumed by the individual blocks. In order to estimate the power dissipation, the switching activity of each individual block is determined. The initial loading of the SF and ID registers is not considered. The switching activity of the flip flops in a B-bit binary counter (counting from 0 to $2^B - 1$) is listed in Table 3.

Table 3: Switching activity in an N-bit counter

Bit position	B – 1	B – 2	…	2	1	0
Switching activity over one cycle	2^1	2^2	…	2^{B-2}	2^{B-1}	2^B

The combined switching activity of all stages over one counter cycle is the sum of a geometric progression with the first term $a_1 = 2$ and quotient $q = 2$,

$$S_B = a_1 \frac{1-q^B}{1-q} = 2\left(2^B - 1\right) \quad . \tag{1}$$

The switching activity of the counter over F_{chip} clock cycles is

$$A_{counter} = \frac{F_{chip}}{2^N} \cdot 2\left(2^B - 1\right) = \frac{2^B - 1}{2^B} \cdot 2F_{chip} \quad . \tag{2}$$

The switching activity in a binary counter varies between F_{chip} and $2F_{chip}$, approaching the maximum as B increases. Based on this observation and a probabilistic analysis, the switching activity bounds are derived for all blocks. Based on the gate level power dissipation per MHz for the target 0.18 µm CMOS technology [3], minimum and maximum power dissipation bounds are estimated. Power estimates for the first technique, realized by the circuits shown in Figs. 2 and 3, are listed in Table 4. All of the data are reported per MHz. The target technology library is discussed in the following section.

This analysis refers to the general case of any variable cycle counter and is particularly applicable to the circuits shown in Figs.

Table 4: Power dissipation in OVSF generator blocks per MHz of the chip clock rate
Artisan Components standard cell library in 0.18 µm CMOS technology [3]

Block	Switching activity		Gate Power [nW / MHz] 0.18 µm	Power / MHz [nW / MHz]	
	Min	Max		Min	Max
Counter	1.5	2	50	75	100
SF NAND2	1		15	15	15
SF NOR2	1.5	2	15	22	30
ID NAND2	1.5	2	15	22	30
XOR block	2.5	3	40	100	120
Entire code generator				234	295

2 and 3. As discussed in section 5, the hardware required for the implementation of the second technique is limited to the circuit shown in Fig. 2. The switching activity of the counter is maximum since the maximum cycle of $SF_{max} = 512$ is used. Significant power and area savings, however, are realized since the SF register and the control circuit for the counting cycle are not required. The power dissipation is dependent on the OVSF code and, particularly, on the number and location of the nonzero bits in the ID register.

7. LOGIC SYNTHESIS

The chip duration for UMTS systems is defined in [1] as

$$T_{chip} = \frac{1}{F_{chip}} = \frac{1}{3.84 \cdot 10^6} \approx 260\ ns \quad . \tag{3}$$

Since the critical path delay of the proposed circuit is significantly smaller than T_{chip}, the primary optimization criteria for the logic synthesis process are area and power, which are well correlated. A sample code generator is synthesized in Cadence Ambit Buildgates using an Artisan Components standard cell library in 0.18 µm CMOS technology with a supply voltage of 1.8 volts [3]. For the first technique, the total area occupied by the circuits shown in Figs. 2 and 3 is 2100 μm^2 with about 85% of the area occupied by the three registers. Eliminating the cycle control circuit shown in Fig. 3, the code generator based on the second technique occupies approximately 1400 μm^2. The critical path delay in both cases is about 1 ns, such that a large number of OVSF codes can be produced by a single generator.

The power dissipation of the circuit varies with the required code as discussed in the previous section. For a 0.18 µm technology, the power dissipation is approximately 1 mW for the first technique, assuming a standard chip rate clock of 3.84 Mchips/sec. Accounting for the reduced number of logic gates, the second technique shown in Fig. 2 consumes about 0.9 mW.

8. SYSTEM LEVEL CONSIDERATIONS

The OVSF code generator proposed in this paper can be applied to a number of different systems. Such systems include a UMTS mobile terminal supporting up to six data channels or a base station receiver processing the incoming signals from K mobile users. System level tradeoffs are considered together with the overall transmitter/receiver architecture. Different channelization codes can be generated by sharing hardware. The proposed solution produces significant savings in area and power since all channelization codes for a single user (up to six) or multiple users can be generated by a single circuit.

One code generator can serve both the receiver and transmitter of a mobile terminal. As defined in the 3GPP standard [1], there is either one code with a spreading factor in the range 4 to 512, or there are several orthogonal codes but with SF fixed at 4. Both techniques described in sections 4 and 5 can be applied. However, considering the relatively small size of the transceiver circuit and the importance of power dissipation and silicon area in portable applications, the second technique is preferable. In certain cases, however, a standalone solution such as the first technique can be applicable when there is no control over the requirements of the system software.

System specifications are different for a base station receiver, where many codes of different spreading factors are required. The software flexibility of the second solution is preferable due to the reduced hardware complexity. The power efficiency of the switching codes is enhanced as only the ID register is loaded. The required change in system software significantly reduces the area and power of all of the code generators in the base station.

Generally, the second technique described in section 5 offers a better solution with increased power and area efficiency. This method also provides more flexibility for dynamically changing the generated code with minimum overhead in switching activity. A change in the higher level system software is required, however, for a shift operation before the code ID number is saved. Alternatively, if such a requirement cannot be realized, the first solution, discussed in section 4, can be used to design the OVSF code generator. The additional hardware is the primary disadvantage of the first technique.

9. CONCLUSIONS

The generation of orthogonal variable spreading factor codes for third generation wireless systems is discussed in this paper and two efficient solutions are proposed. The simplicity and flexibility of the proposed circuits allow for significant area and power savings since all of the channelization codes for a single user (up to six) or multiple users can be produced by a single code generator. This flexibility is essential for UMTS transceivers, since signals of variable content and data rate requirements can be transmitted or received by changing the number and spreading factors of the channelization codes as specified by the 3GPP standard.

10. REFERENCES

[1] Third Generation Partnership Project; Technical Specification Group Radio Access Network; 25.213, *Spreading and Modulation*, Release 5, www.3gpp.org, September 2002.

[2] B. D. Andreev, E. G. Friedman, and E. Titlebaum, "Efficient Implementation of a Complex ±1 Multiplier," *Proceedings of the ACM Great Lakes Symposium on VLSI*, pp. 83-88, April 2002.

[3] *TSMC 0.18 µm Process 1.8-Volt SAGE-X™ Standard Cell Library Databook*, Release 3.1, Artisan Components, October 2001.