Article

Symmetry in the Bit-Stream Converter Design

Olga Bureneva *, Nikolay Safyannikov and Sergey Mironov

Department of Computer Science and Engineering, Saint Petersburg Electrotechnical University “LETI”, 197022 Saint Petersburg, Russia; nmsafyannikov@etu.ru (N.S.); semironov@etu.ru (S.M.)

* Correspondence: oibureneva@etu.ru

Abstract: The paper presents the results of the study of the influence of symmetry in the design of bit-stream digital converters. We have shown realizations of the symmetry-based approach at different levels: at the level of basic elements, functional converters, and at the level of processes occurring in bit-streaming devices. Using symmetry in design, we have developed basic bit-stream elements that realize frequently used transformations with good technical performance. As a research result, we present descriptions and implementation results of the designed symmetric bit-stream devices in FPGA chips. Using the proposed elements and the concept of symmetric bit-stream device design, we designed and presented a specialized computing module for a temperature sensor controller.

Keywords: bit-stream converter; symmetric digital design; functional bit-stream converter; PWM signal; negative feedback; sensor transducers

1. Introduction

In the last few years, the paradigm of cyber-physical systems has been actively developing. Such systems support the optimal integration of digital and physical components, which maximizes the synergies between the two and allows achieving significant benefits both in terms of cost, performance, and reliability [1]. The development of cyber-physical systems is due not only to new technologies of data transmission, but also to the possibility of creating integrated elements that simultaneously perform the following functions: primary processing of information about a physical object or phenomenon of various natures into an electrical signal; intermediate transformation to obtain the necessary form of data representation; and computational conversions providing the formation of the output signal transmitted to the core located on the next level of cyber-physical architecture. Due to this integration, part of the computational processing is moved from the processing core closer to the sensing element, realizing the computation and analytics “next to the sensor” paradigm [2,3]. Within the “computation near the sensor” paradigm, there is a need to develop special devices that will provide computation near the sensors. These should be compact, energy-efficient, and fault-tolerant modules that can be manufactured as separate chips or integrated with sensitive elements in a single crystal [4,5].

The hardware implementation of data conversion near the sensor is available in analog, digital, pulse, and mixed formats. Obviously, performing calculations in the formats that are obtained at the sensing outputs will avoid additional operations related to data format conversions. Many sensing elements generate a frequency output signal represented as pulse streams or PWM (pulse-width modulated) signal streams [6–8]. In both cases, the amplitude of the signals is the same and can be considered a logical value of 1. Moreover, the frequency signals near the sensor can be obtained by converting analog signals into the time parameters of single-amplitude pulses using universal micromodules [9] or specialized converters oriented to specific types of sensor elements [10,11]. The listed
pulse forms of signal representation, also called bit-stream, allow the use of digital logic elements for primary data processing [12]. In this case, programmable logic devices of various levels of complexity can be used as the element base for the implementation of near-sensor computing modules [13]. In some cases, it is economically feasible to perform the primary transducers of sensor signals in the form of ASICs [14].

The purpose of our research was to find a formal way to construct bit-stream converters on a digital element basis using the concepts of structural symmetry in the architecture of the converters and functional symmetry in the conversion processes. We have shown the realization of the approach based on structural symmetry at different levels: at the level of basic elements, functional converters, as well as at the level of processes occurring in bit-stream converters.

Using symmetric structures, we have developed the following basic elements: binary rate multiplier, bit-stream multiplier-divider, and bit-stream quadratic converter. We used the proposed elements to design a controller for a temperature sensor that generates an output as a stream of PWM signals. The importance of our study is due to the fact that the proposed approach to the construction of bit-stream converters can be used in the design of compact, specialized controllers placed near the sensors. When implemented as ASICs, such controllers can be placed on the same chip as the sensing element. This will allow us to perform transformations at the point of measurement of physical parameters. The novelty of our study is determined by the use of an original approach to the organization of computations: arithmetic transformations are performed not by sequential calculation of complex functions, but by tracking the input signals with the help of a symmetric dynamical system tending to a dynamic equilibrium mode. The parameters of this mode can be used to obtain the result of computations.

The paper is organized as follows. We first briefly review typical related works on non-algorithmic computation in Section 2. In Section 3, we present the proposed symmetric bit-stream circuits and mathematically describe the processes occurring in them. Section 4 shows the results of the implementation of the developed devices in FPGAs, summarizes the hardware costs required for this, and shows the simulation results. Section 5 analyzes the obtained results. Conclusion and future research directions are given in Section 6. The appendices show the traditional calculators for comparison with the proposed solutions.

2. Related Work

The principle of symmetry in any field of natural science allows us to create new solutions and find new laws. Special attention is paid to electronic circuits because the efficiency of their design is often determined by the symmetry/asymmetry of their architecture, their topological implementation, and the processes occurring in them. The application of the symmetry principle in the design of hardware converters is shown in many sources at different levels: at the level of analog components, and at the level of digital elements, at the level of system modules. The symmetry of device architectures as well as symmetry in layout can be considered.

For example, the authors of [15] showed both theoretically and experimentally the application of symmetry rules at the level of inductively coupled active LRC circuits. The authors have developed the basis of PT (parity-time) symmetric electronics, which allows using the apparatus of PT-symmetric Hamiltonians to obtain important information for understanding the behavior of the system at the level of RLC components. In Ref. [16], it is shown that symmetry constraints in an analog circuit have a significant influence on its performance. This paper proposes an effective way to detect the symmetry pairs of analog devices from the circuit netlist and analyze the symmetry constraints in these pairs. Correction of the detected irregularities can improve the performance of the implementation. Victor K. Tamba et al. [17] present an oscillator with different special features and symmetry. The features and oscillator dynamics are discovered through different tools of nonlinear dynamics. An electronic circuit is designed to mimic the
oscillators dynamics. Moreover, the combined synchronization of two drives and one response oscillator is reported.

The authors of the paper [18] presented the results of a study on the creation of symmetry constraints at the system level. They proposed a new method for detecting system symmetry constraints for analog circuits based on graph similarity. Using graph spectral analysis and graph centrality, the proposed method can be applied to circuits and systems of large scale and different architectures. Pijush Kanti Bhattacharjee [19] has shown that with proper application of symmetric functions, it is possible to easily solve design problems of computing devices and complex control systems with different parameters or variables, as well as problems with a large number of variables consisting of functions. The author also proposed a way to simplify Boolean functions of any kind using symmetric functions. Thus, he developed an approach to simplifying the implementation of the hardware devices based on symmetries. The paper [20] shows a variant in-memory calculation implemented on the basis of eight-transistor (8T) static random-access memory (SRAM). This approach provides execution of multibit relocatable operations. The article shows the architecture of a weighting block with diagonal symmetry and the results of an experiment confirming a 54.4% increase in the performance of the computator compared to traditional computations.

In the paper [21], a new structure of symmetry constraint extraction based on graph learning for analog/mixed circuits is presented. An efficient heterogeneous multigraph representation was proposed to model interconnections in a circuit. It was shown that the algorithm for embedding circuit elements represents the circuit with the most representative substructures. Thanks to the teacherless inductive learning method, the proposed structure can be generalized to any schematic project. Experimental results demonstrated the effectiveness of the proposed structure in detecting symmetry constraints at both system and device levels. The authors of the paper [22] revisited the notion of symmetry in Boolean functions, with a special emphasis on group theory. They proposed a new algorithm that searches for symmetries of Boolean functions under permutations of inputs and outputs. This allows us to use the advantages of functional symmetry in solving problems of synthesis and verification of digital circuits. The authors of the paper [23] show methods for constructing symmetric arithmetic schemes that compute polynomials. In doing so, he proves that the symmetric scheme is provided by the symmetric form of the realized polynomial. An example of the efficiency of symmetry in structure-level design is shown in the example of digital filter design [24]. The authors studied the architecture of a two-dimensional (2-D) IIR filter and showed that new structures of a two-dimensional IIR filter with diagonal and four-fold rotational symmetries. The critical paths in the proposed filters are shorter, which allows for more efficient hardware implementations.

Thus, we can note the relevance of studies related to the application of various types of symmetries in the design and analysis of digital circuits. In this case, various symmetries at the functional and structural levels are considered. It can also refer to the symmetry of the processes occurring in the devices. This type of symmetry corresponds to symmetry with respect to time transfer and is provided by the physical equivalence of different time moments and their homogeneity. The concept of dynamic functional symmetry is also used; under such symmetry, we understand the processes occurring in symmetrical parts of the circuit, with some unequal activity at the same averaged characteristics. Our research focuses on identifying symmetries in bit-stream devices [12] and determining the effectiveness of incorporating symmetry principles into their design.

3. Materials and Methods

The types of symmetry observed in electronic devices can be divided into two types. The first type is related to the architecture and can be considered as symmetry in the device, manifested at the level of modules. The second type is observed at the level of processes occurring in different branches (channels) of the devices. The symmetry of
processes is relevant for devices performing the processing of continuous information flows. In this case, the stream can be considered at different levels: it can be a stream of single pulse signals, a stream of codewords, or a stream of frames or files. In considering bit-stream devices, we will concentrate on analyzing streams of pulse signals with the unit amplitude.

The basic elements of bit-stream devices are logic primitives (OR, AND, NOT), registers, reversible counters, and binary rate multipliers. The architecture of registers and reversible counters is quite well researched, and there are both classical implementations and original devices [25]. Binary rate multipliers can also be designed using different approaches [26–28]. The binary rate multiplier was proposed and actively modified in the 1970s of the last century, but no principal changes in the structure have been proposed so far. Modern integrated circuits, for example [29], are produced according to the ideas shown in the first articles.

3.1. Binary Rate Multiplier

The binary rate multiplier generates a pulse stream of average frequency proportional to the value of the decimal equivalent of the binary code D according to the following formula:

\[ F_{\text{out}} = \frac{F_{\text{ref}} D}{2^n} \]

where \( F_{\text{ref}} \) is input clock frequency of the multiplier; \( D = \sum_{i=0}^{n-1} D_i 2^i \) is the decimal equivalent of the input binary code, \( D_i = \{0; 1\} \) is logic state of the i-th bit of the code D; and n is multiplier bit capacity.

The classical structure of the multiplier implemented in accordance with the approach proposed in [26] is realized in the form of an integrated circuit and has a sequential organization [29]. This structure of a binary rate multiplier has the following disadvantages:

- speed limitations, due to the need to synchronize the triggering of AND components and the organization of a group transfer circuit by the OR component;
- dependence of the number of inputs of AND elements on the multiplier digit capacity, leading to the usage of multi-input AND circuits;
- nonlinear deterioration of the topological characteristics of the circuit with increasing bit capacity, which leads to a large number of traces going to the AND circuits from lower to higher digits along the entire structure, and these traces intersect with synchronization traces.

We propose a new architectural solution based on symmetric representation. The schematic of the symmetric binary rate multiplier is presented in Figure 1. The multiplier consists of two n-bit binary counters, an inverter, n three-input AND elements, and an OR element. The device has a clock frequency input \( F_{\text{ref}} \) and a reset input R.

The symmetric rate multiplier structure is implemented by using two symmetric branches organized with binary counters. The first counter is controlled by the input clock frequency, and the second counter is controlled by this frequency shifted by half a clock. In this way, symmetrical branches with processes shifted by half a clock cycle are formed. Such a displacement provides a shift in time for the processes occurring in different branches. As a result, in the device there is a simultaneous formation of two states of the counting process: the current one, shown by counter 1 (top counter in the figure), and the previous one, shown by counter 2 (lower counter in the figure).
Figure 1. Symmetrical binary rate multiplier: (a) schematic diagram of the multiplier; and (b) timing diagram of the three-digit multiplier operation for two input codes \( D = 111 \) and \( D = 101 \).

The operation of the multiplier is based on the principle of simultaneous formation of two identical counting processes, with the second one shifted relative to the first one by moving the start of the second counter by half a clock cycle. This type of symmetry relies on the physical equivalence of different moments of time and its homogeneity. The comparison of streams allows us to identify a bit that changes state from zero to one during the counting process, i.e., in the code at the counter 1 output, this bit is already equal to 1, but in the code at the counter 2 output, it is still equal to 0. If the corresponding digit of the bus \( D \) expanded in reverse order holds one signal, the output will form a single pulse. The algorithm for output bit-stream formation is presented by us in [30].

The logical expression for the pulse stream at the output has the following form:

\[
F_{\text{out}} = L_0 v L_1 v \ldots v L_{n-1}
\]

where \( L_i = Q_{1i} Q_{2i} D_{n-i-1} \) and \( i = 0, n - 1 \) is the digit number of the \( D \) bus.

Obviously, for any \( i \), the equality \( Q_{1i-1} Q_{2i-1} = 1 \) is achieved when \( Q_{1i-1} = 1 \) and \( Q_{2i-1} = 0 \). This situation occurs for each output state \( Q_{1i} \) equal to a logical one. This is due to the fact that the moment of transition of the output state \( Q_{2i} \) from 1 to 0 (\( Q_{2i} \) from 0 to 1) is shifted by the time \( \tau \) relative to the moment of transition of the output \( Q_{1i} \) from 0 to 1, and is ensured by the clocking of counter 2 by the signal from the inverter output, i.e., signal \( F_{\text{ref}} \).

Thus, a pulse sequence will be formed at the output of any AND element; its frequency is defined as follows:
Impulses in these streams are separated by time, so at the output of the OR element, the cumulated stream is formed, i.e., the average value of the output frequency $F_{out}$ of the multiplier is determined by the following expression:

$$F_{out} = F_0 + F_1 + \ldots + F_{n-2} + F_{n-1}$$

or

$$F_{out} = \frac{F_{ref}}{2^1} D_{n-1} + \frac{F_{ref}}{2^2} D_{n-2} + \ldots + \frac{F_{ref}}{2^{n-1}} D_1 + \frac{F_{ref}}{2^n} D_0$$

taking $F_{ref}$ and $2^{-n}$ out of brackets, we have the following expression:

$$F_{out} = F_{ref} 2^{-n} (D_{n-1}2^{n-1} + D_{n-2}2^{n-2} + \ldots + D_1 2^1 + D_0 2^0).$$

Here, the expression in brackets corresponds to the expanded record of the binary code $D$. Thus, the average value of the output frequency of the pulse stream has the form:

$$F_{out} = \frac{F_{ref} D}{2^n}.$$  

This formula corresponds to the original expression for a binary rate multiplier.

The timing diagrams of the three-bit binary rate multiplier operation are shown in Figure 1b. The counters are set to 0 by the signal $R$. At the same time, the inverse outputs of the second counter $Q_2$, will be set to 1. After completion of the reset signal $R$, the operation is as follows: The conjunctions of the direct outputs of the first counter and the inverse outputs of the second counter produce pulse streams with an average frequency defined by the following formula:

$$F_{out} = \frac{F_{ref}}{2^n}. $$

When the input code $D = 111_2$, all of these streams can be observed at the outputs of the AND elements. The duration of the three-bit multiplier operation period $T$ corresponds to 8 pulses of the $F_{ref}$ stream. During one period $T$, 4 pulses can be observed on the line $A_{out}$, 2 pulses on the line $A_{out}$, and one pulse on the line $A_{out}$. The number of pulses in all streams ($A_{out}$, $A_{out}$, $A_{out}$) during the period is 7, which is shown in the diagram $F_{out}$ and corresponds to the value of the code $D$. When the code $D = 101_2$ is applied, only the streams from the outputs of $A_{out}$ and $A_{out}$ will be summarized at the OR element; the stream at the $A_{out}$ element output is not formed due to the zero value of $D_1$. The number of pulses in the output stream $F_{out}$ is 5, which corresponds to the code $D$.

The structure of the binary rate multiplier is built bit by bit without increasing the number of input elements from bit to bit and is free from the disadvantages of the classical realization.

### 3.2. Bit-Stream Multiplier-Divider

To perform multiplication and division operations on bit-stream signals, the method of converting the streaming data into binary codes and then performing the necessary operations using traditional digital arithmetic converters (adders, multipliers, divisors, etc.) has traditionally been used. We propose an approach to continuous tracking computational transformations in which no change in the data form is required and the computation is performed in a streaming mode. Figure 2 shows the bit-stream multiplier-divider (BSMD) that performs an operation on input bit-stream signals (PWM signals and frequency streams) and binary codes, with the computational results represented as binary code and frequency stream.
The BSMD circuit contains two binary rate multipliers (D/f), two AND elements, a reversible binary counter (CT) and a register (RG). In this unit, there is a symmetry at the level of modules, providing the formation of positive (\(\Theta_1, N_{in}, F_1, D/f_1, \text{AND}_1\)) and negative (\(\Theta_2, N_{out}, F_2, D/f_2, \text{AND}_2\)) branches of the unit. The process of result formation is based on the symmetry of the processes occurring in the corresponding branches.

At the initial moment, the counter and the register are reset to zero. The input binary code (\(N_{in}\)), PWM signals (\(\Theta_1\) and \(\Theta_2\)), and pulse streams (\(F_1\) and \(F_2\)), are fed to the inputs of the unit. Under the influence of the input code, pulses are generated at the output of the first binary rate multiplier; when the signal \(\Theta_1\) is active, these pulses pass through the element AND to the summing input of the reversible counter. No pulses are input to the subtracting input of the reversible counter because the code \(N_{out}\) at the input of the second binary rate multiplier is zero. At the beginning of the next period of PWM signals, by raising of the signal \(\Theta_2\), the code from the output of the reversible counter is written into the register. As in the previous period, under the influence of the code \(N_{in}\), frequency \(F_1\), and PWM signal \(\Theta_1\), pulse stream, \(F_+\) is formed and arrives at the summing input of the counter. The output code \(N_{out}\) has become non-zero, so under its influence, taking into account the frequency \(F_2\) and the PWM signal \(\Theta_2\), a pulse stream \(F_-\) is formed and fed to the subtracting input of the reversible counter.

The negative feedback in the unit provides an exit to the established equilibrium mode characterized by the functional symmetry of the streams in the positive and negative branches of the unit. In this case, the number of impulses \(N_+\) coming to the summing input of the reversible counter and the number of impulses \(N_-\) coming to the subtracting input of the counter during the period \(T\) of pulse-width modulation are equal:

\[ N_+ = N_- \quad \text{or} \quad F_+ = F_- \]

where \(F_+\) and \(F_-\) are the average frequencies of pulse streams at the summing and subtracting inputs of the reversible counter. The average frequencies of the pulse streams at the outputs of binary rate multipliers are determined by the following expression:

\[ F_{D/f1} = \frac{F_1 N_{in}}{2^n} \quad \text{and} \quad F_{D/f2} = \frac{F_2 N_{out}}{2^n}. \]

Taking into account the conjunction operation performed by AND elements, the inputs of the reversible counter receive width-modulated pulse streams with frequencies defined as follows:

\[ F_+(t, \Theta_1) = \begin{cases} \frac{F_1 N_{in}}{2^n} & \text{when} \quad kT \leq t \leq kT + \tau_1 \\ 0 & \text{when} \quad kT + \tau_1 < t < (k+1)T \end{cases} \]
\[
F_{\text{o}}(t, \Theta) = \begin{cases} 
\frac{F_{\text{out}}N_{\text{out}}}{2^n} & \text{when } kT \leq t \leq kT + \tau_2 \\
0 & \text{when } kT + \tau_2 < t < (k+1)T
\end{cases}
\]

where \( k = 0, 1, 2 \ldots \), and \( \tau_1 \) and \( \tau_2 \) are the durations of the values 1 of the PWM signals \( \Theta_1 \) and \( \Theta_2 \) with period \( T \), correspondingly.

The number of pulses coming to the summing and subtracting inputs of the reversible counter during one period of pulse-width modulation is determined by the following equations:

\[
N_+ = \frac{N_{\text{in}}\Theta_1 F_1}{2^n}, \quad N_- = \frac{N_{\text{out}}\Theta_2 F_2}{2^n}.
\]

The contents of the reversible counter are written to the register at the beginning of each period on the rising edge of the modulated signal. The code stored in the register is present at the BSMD output during the whole period.

After the first operating period of the BSMD, the reversible counter will generate the following code:

\[
N_{\text{out}0} = N_{\text{out0}} + \frac{N_{n} \Theta_1 F_1}{2^n} - \frac{N_{\text{out0}} \Theta_2 F_2}{2^n} = N_{\text{out0}} - N_{\text{out0}} \Theta_2 k_2 + N_{\text{in}} \Theta_1 k_1 = \frac{N_{\text{out0}}(1 - \Theta_2 k_2) + N_{\text{in}} \Theta_1 k_1}{2^n}.
\]

where \( N_{\text{out0}} \) is the initial value of the output code stored in the register; \( k_1 = \frac{F_1}{2^n}, \quad k_2 = \frac{F_2}{2^n} \).

After the second period of operation, the code at the register output will be calculated as follows:

\[
N_{\text{out}2} = N_{\text{out}0} - N_{\text{out}0} \Theta_2 k_2 + N_{\text{in}} \Theta_1 k_1 = N_{\text{out}0}(1 - \Theta_2 k_2)^2 + N_{\text{in}} \Theta_1 k_1(1 + (1 - \Theta_2 k_2))
\]

\[
N_{\text{out}} = N_{\text{out2}} - N_{\text{out2}} \Theta_2 k_2 + N_{\text{in}} \Theta_1 k_1 = \frac{N_{\text{out0}}(1 - \Theta_2 k_2)^3 + N_{\text{in}} \Theta_1 k_1(1 + (1 - \Theta_2 k_2) + (1 - \Theta_2 k_2)^2)}{2^n}.
\]

After the \( i \)-th period of the multiplier-divisor operation, the reversible counter will generate the following code:

\[
N_{\text{out}i} = N_{\text{out0}}(1 - \Theta_2 k_2)^i + N_{\text{in}} \Theta_1 k_1(1 + (1 - \Theta_2 k_2) + (1 - \Theta_2 k_2)^2 + \cdots + (1 - \Theta_2 k_2)^{i-1})
\]

The second summand of this expression is a geometric progression with the common ratio \( q = (1 - \Theta_2 k_2) \) and can be represented by the sum calculated as follows:

\[
\sum_{q=1}^{q=(1-\Theta_2 k_2)^{i-1}} = \frac{(1 - \Theta_2 k_2)^{i-1} - 1}{(1 - \Theta_2 k_2 - 1)} = \frac{1 - (1 - \Theta_2 k_2)^{i-1}}{\Theta_2 k_2}.
\]

Using the expression of the geometric progression sum, we obtain the following expression:

\[
N_{\text{out}m} = N_{\text{out0}}(1 - \Theta_2 k_2)^m + \frac{N_{\text{in}} \Theta_1 k_1}{\Theta_2 k_2} = \frac{N_{\text{out0}}(1 - \Theta_2 k_2)^m - N_{\text{in}} \Theta_1 k_1(1 - \Theta_2 k_2)^{m-1}}{\Theta_2 k_2}.
\]

Since the values of \( \Theta_2 \) and \( k_2 \) are in the following ranges: \( 0 < \Theta_2 \leq 1 \), \( 0 < k_2 \leq 1 \), then \( 0 < \Theta_2 k_2 \leq 1 \); hence, the following applies:

\[
\lim_{i \to \infty} (1 - \Theta_2 k_2)^{m-1} = 0.
\]

Thus, in the dynamic equilibrium mode, the characteristic of the BSMD will have the following form:

\[
N_{\text{out}} = \frac{N_{\text{in}} \Theta_1 k_1}{\Theta_2 k_2}.
\]

The time for the BSMD to reach dynamic equilibrium mode depends on how fast the value \( (1 - \Theta_2 k_2)^{m-1} \) tends to zero and is therefore inversely proportional to the bit depth of the unit. The dynamic equilibrium mode corresponds to the symmetry of the processes in the positive and negative branches of the unit. It is in this mode that the code at the output \( N_{\text{out}} \) and, accordingly, the pulse flow at the output \( F_{\text{out}} \) correspond to the result of the multiplicative-divisive operation.
3.3. Bit-Stream Quadratic Converter

A frequently used operation in near-sensor calculations is raising to a power. It is interesting both as an independent operation and as an operation for forming approximate polynomials. Similar to the multiplicative-divisive conversion, the raising to a power can be performed by converting bit-stream data into digital codes, followed by traditional computation using typical arithmetic modules. Also, power functions can be computed directly in pulse form without transforming the form of the data. We will consider this in the example of squaring.

The input signals of the converter are represented in the form of pulse streams with frequencies \( F_1 \) and \( F_2 \), and \( F_1 > F_2 \). Two negative feedback branches are used in the converter. After the initial transients are completed, the converter reaches a dynamic equilibrium mode and further realizes the tracking mode. When the input signal changes slightly, the result at the output of the device will be obtained with the minimum delay. And two memory feedback loops provide increased accuracy in calculations and increased speed.

The schematic of the converter is shown in Figure 3.

![Figure 3. Bit-stream quadratic converter: (a) schematic diagram of the converter; and (b) timing diagram of the converter operation.](image-url)
input signal. The PWM signals also determine the moments when the feedback is triggered. Due to negative feedback during operation, the device reaches the dynamic equilibrium mode. In this mode, the processes in the positive and negative branches of the device are symmetrical, and the codes recorded in the registers are the result of calculations. Violation of the symmetry of processes in the device is characterized by the occurrence of mismatches. Negative feedback provides compensation for mismatches and returns to the equilibrium mode. When the equilibrium in the device is violated, the time interval $T$ is not constant; when the equilibrium is reached, the interval $T$ is constant. In equilibrium mode, the number of pulses arriving at the summing (+) and subtracting (−) inputs of the reversible counters CT1 and CT2 during the time interval $T$ are equal to each other:

$$N_{CT2+} = N_{CT2-} \text{ or } \overline{F}_{CT2+} = \overline{F}_{CT2-},$$  \hspace{1cm} (1)

$$N_{CT1+} = N_{CT1-} \text{ or } \overline{F}_{CT1+} = \overline{F}_{CT1-},$$  \hspace{1cm} (2)

where $\overline{F}_{CT+}$ and $\overline{F}_{CT-}$ are the average values of pulse stream frequencies at the summing and subtracting inputs of the reversible counters. The average value of the frequency at the «+» input of the counter CT2 for the period $T$ of the pulse stream $\overline{F}_{CT2+}$ is determined as follows:

$$\overline{F}_{CT2+} = F_1 \Theta,$$  \hspace{1cm} (3)

where $\Theta$ is the relative duration of the logical one level at the RSff output during the period $T$. The subtraction input (−) of this reversible counter receives a pulse sequence from the output of the binary frequency multiplier $D/f\ 6$:

$$\overline{F}_{CT2-} = \frac{N_{out1}F_1}{2^n},$$  \hspace{1cm} (4)

where $N_{out1}$ is the code stored in register RG4.

The average value of the pulse stream frequency at the summing input of the reversible counters CT1 during the period $T$ is determined as follows: $\overline{F}_{CT1+} = \overline{F}_{CT2-} \Theta$. Substituting (4) into the last expression, we obtain the following:

$$\overline{F}_{CT1+} = \frac{\Theta N_{out1}F_1}{2^n}.$$  \hspace{1cm} (5)

The pulse sequence from the output $D/f\ 5$ is transferred to the subtraction input of the reversible counter CT1. Its average frequency is described as follows:

$$\overline{F}_{CT1-} = \frac{N_{out1}F_1}{2^n}$$  \hspace{1cm} (6)

where $N_{out}$ is the code stored in register RG3.

When the transients are completed, the equality of the number of pulses received at the summing and subtracting inputs of the reversible counters during the period $T$ of the input frequency $F_2$ is achieved. Thus, the following expressions are derived:

- for the counter CT2 from (1), taking into account (3) and (4), we can write as follows:
  $$F_1 \Theta = \frac{N_{out1}F_1}{2^n} \text{ or } N_{out1} = \Theta 2^n;$$  \hspace{1cm} (7)

- for the counter CT1 from (2), taking into account (5) and (6), we can write as follows:
  $$\frac{\Theta N_{out1}F_1}{2^n} = \frac{N_{out1}F_1}{2^n} \text{ or } N_{out} = \Theta N_{out1};$$  \hspace{1cm} (8)

By substituting the value of $N_{out1}$ from expression (7) into (8), we obtain the following:

$$N_{out} = \Theta 2^n.$$  \hspace{1cm} (9)
The relative duration $\theta$ of the signal formed by the CT and RS\textsuperscript{ff} can be expressed as follows:

$$\theta = \tau \frac{F_2}{F_1},$$  \hspace{1cm} (10)

where $\tau = \frac{k}{F_1}$, $k$ is the division factor of the CT counter.

From Equations (9) and (10), the functional characteristic of the converter is derived:

$$N_{\text{out}} = \left(\frac{k}{F_1} F_2\right)^2 \frac{2^n}{2^{2k}} = \left(\frac{2^n}{F_1 F_2}\right)^2.$$ \hspace{1cm} (11)

Time diagrams of the converter operation are shown in Figure 3b.

Registers in feedback improve functional symmetry by fixing codes in the device branches during $T_i$ intervals, which also improves the dynamic properties of the converter. Internal pulse streams formed by the elements of the converter are characterized by a large nonuniformity due to the width modulation by the AND components. Consequently, in order to reduce non-uniformities without data fixation in feedback loops, it is necessary to perform averaging over a much longer conversion time interval $T$ than $T_i$, i.e., $T \gg 1/F_2$. The specific values of $T$ are determined by the allowable errors. Registers in feedback loops improve other characteristics of the converter. For example, the codes at the inputs of the D/f frequency multipliers become more stable, and this improves the uniformity of the pulse streams at their outputs, thus stabilizing the $N_{\text{out1}}$ and $N_{\text{out2}}$ codes, reducing the time to reach the dynamic equilibrium state.

By introducing additional feedback loops, it is possible to modify this converter to realize higher power conversion.

4. Results

The elements proposed in Section 3 were realized on a digital basis using the Cyclone 10 LP FPGA family company Intel FPGA. Simulation of the devices was performed with ModelSimAltera Starter Edition 10.1, and implementation was performed using CAD Quartus Prime 18.0.0. Parameterized VerilogHDL descriptions of devices were prepared for the investigation, which allowed them to easily change the parameters of the devices and perform their research.

4.1. Binary Rate Multiplier

The RTL of the symmetrical 8-bit binary rate multiplier, obtained by compiling the VerilogHDL description, and the timing diagram of its operation are shown in Figure 4.
Figure 4. Symmetrical 8-bit binary rate multiplier: (a) RTL diagram obtained by compiling the VerilogHDL description with the Quartus II compiler; (b) timing diagram of multiplier operation for input code $D = 255$; and (c) timing diagram of multiplier operation for input code $D = 15$.

The pairs $Add0$ and $rg0$ and $Add1$ and $rg1$ represent counters symmetrically placed in the device branches. The first counter is driven by the rising edge of the clock signal, and the second by the falling edge of the clock signal. In the timing diagram Figure 4b, the input code $D = 255$ corresponds to passing $2^n - 1$ pulses of the $F_{\text{ref}}$ stream to the $F_{\text{out}}$...
output during the device period, which is observed in the diagram in the range from 2140.36 ns to 2150.6 ns. The specified range for the frequency $F_{\text{ref}} = 1/40$ ns corresponds to the device operating frequency of 97.656 MHz. On the time diagram Figure 4c, input code $D = 15$ corresponds to the formation of 15 pulses at the output $F_{\text{out}}$ during the period of operation of the device, which is observed on the line $F_{\text{out}}$. The characteristics of the designed device are shown in Table 1.

Table 1. Resources required for realization of the binary rate multiplier and maximum clock frequencies depending on multiplier digit capacity.

<table>
<thead>
<tr>
<th>Bit Capacity</th>
<th>8</th>
<th>10</th>
<th>12</th>
<th>14</th>
<th>16</th>
<th>18</th>
<th>20</th>
<th>22</th>
<th>24</th>
<th>26</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total logic elements</td>
<td>25</td>
<td>32</td>
<td>38</td>
<td>44</td>
<td>51</td>
<td>57</td>
<td>63</td>
<td>70</td>
<td>76</td>
<td>83</td>
</tr>
<tr>
<td>Total combinational functions</td>
<td>25</td>
<td>32</td>
<td>38</td>
<td>44</td>
<td>51</td>
<td>57</td>
<td>63</td>
<td>70</td>
<td>76</td>
<td>83</td>
</tr>
<tr>
<td>Logic element usage by number of LUT inputs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-- 4 input functions</td>
<td>5</td>
<td>5</td>
<td>7</td>
<td>9</td>
<td>9</td>
<td>11</td>
<td>13</td>
<td>13</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>-- 3 input functions</td>
<td>6</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>15</td>
<td>15</td>
<td>18</td>
</tr>
<tr>
<td>-- ≤ 2 input functions</td>
<td>14</td>
<td>18</td>
<td>22</td>
<td>26</td>
<td>30</td>
<td>34</td>
<td>38</td>
<td>42</td>
<td>46</td>
<td>50</td>
</tr>
<tr>
<td>Logic elements by mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-- normal mode</td>
<td>13</td>
<td>16</td>
<td>18</td>
<td>20</td>
<td>23</td>
<td>25</td>
<td>27</td>
<td>30</td>
<td>32</td>
<td>35</td>
</tr>
<tr>
<td>-- arithmetic mode</td>
<td>12</td>
<td>16</td>
<td>20</td>
<td>24</td>
<td>28</td>
<td>32</td>
<td>36</td>
<td>40</td>
<td>44</td>
<td>48</td>
</tr>
<tr>
<td>Total registers</td>
<td>16</td>
<td>20</td>
<td>24</td>
<td>28</td>
<td>32</td>
<td>36</td>
<td>40</td>
<td>44</td>
<td>48</td>
<td>52</td>
</tr>
<tr>
<td>$F_{\text{max}} (F_{\text{ref}})$, MHz</td>
<td>544.0</td>
<td>458.0</td>
<td>346.9</td>
<td>416.4</td>
<td>400.6</td>
<td>412.3</td>
<td>394.3</td>
<td>383.2</td>
<td>320.5</td>
<td>346.4</td>
</tr>
</tbody>
</table>

Analysis of Table 1 shows an almost linear increase in the total number of logic elements required to implement the device. The maximum possible frequency for the input $F_{\text{ref}}$ was estimated using the slow 1200 mV 85 °C model.

4.2. Bit-Stream Multiplier-Divider

The results of the FPGA implementation of the symmetric bit-stream multiplier-divider are shown in Figure 5.
Figure 5. Symmetric bit-stream binary multiplier-divider: (a) RTL circuit obtained from compilation of VerilogHDL description by Quartus Prime compiler; and (b) timing diagram of the multiplier operation for input code Nin = 100, PWM1/PWM2 = 1/2, F1 = F2.

The RTL of the binary multiplier-divider is obtained by compiling its VerilogHDL description with the Quartus Prime compiler. Unlike the circuit in Figure 2, it contains logic elements: XOR clk_revers~0, AND clk_revers~1, and Phase Locked Loop PLL_inst. These elements make it possible to eliminate the simultaneous appearance of active single signals on the summing and subtracting inputs of the reversible counter, which can lead to the execution of only one action and the occurrence of a counting error. PLL is used to generate a short pulse to strobe an increment or decrement operation on the reversible counter. This short pulse is generated in such a way as to prevent the simultaneous change of signals at the P_minus and clk and P_plus and clk inputs, i.e., to avoid violation of the Tsu preset time and Th hold time of the internal triggers of the reversible counter. The timing diagram shows the transient process until the calculation result is reached. The characteristics of hardware costs are shown in Table 2.

Table 2. Resources required for realization of the bit-stream binary multiplier-divider and maximum frequency of the clock signal in dependence on multiplier-divider bit capacity.

<table>
<thead>
<tr>
<th>Bit Capacity</th>
<th>Resource and Frequencies</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>8</td>
</tr>
<tr>
<td>Total logic elements</td>
<td>70</td>
</tr>
<tr>
<td>Total combinational functions</td>
<td>46</td>
</tr>
<tr>
<td>Logic element usage by number of LUT inputs</td>
<td></td>
</tr>
<tr>
<td>-- 4 input functions</td>
<td>8</td>
</tr>
<tr>
<td>-- 3 input functions</td>
<td>22</td>
</tr>
<tr>
<td>-- ≤ 2 input functions</td>
<td>16</td>
</tr>
<tr>
<td>Logic elements by mode</td>
<td></td>
</tr>
<tr>
<td>-- normal mode</td>
<td>27</td>
</tr>
<tr>
<td>-- arithmetic mode</td>
<td>19</td>
</tr>
<tr>
<td>Total registers</td>
<td>48</td>
</tr>
<tr>
<td>Fmax (F1), MHz</td>
<td>423.5</td>
</tr>
<tr>
<td>Fmax (F2), MHz</td>
<td>543.5</td>
</tr>
</tbody>
</table>

There is a linear increase in all resources with the increase in device bit size, which makes it easy to predict the hardware cost as the bit size changes. The maximum possible frequencies F1 and F2 were estimated using the slow 1200 mV 85 °C model. It is impossible to identify a definite trend in the change of frequencies when changing the digit capacity. This is most likely due to the fact that when the bit capacity reaches certain values, the
project trace becomes more complex, and these specific values depend on the FPGA architecture.

4.3. Bit-Stream Quadratic Converter

The results of the FPGA implementation of the symmetric bit-stream quadratic converter are shown in Figure 6.

Figure 6. Symmetric bit-stream quadratic converter: (a) RTL schematic resulting from compilation of Verilog description by Quartus Prime compiler, and (b) timing diagram of device operation for input signals $F_1 = 50$ MHz, $F_2 = 196$ KHz, $k = 120$.

For correct operation of the reversible counter, as in the multiplier-divider, a built-in PLL is used to generate clock pulses coming to the clk input of the reversible counters. The counter CT and flip-flop Dff with an asynchronous reset generate an internal PWM signal based on the input frequency $F_2$. The hardware costs of such an implementation are shown in Table 3.

Table 3. Fitter resource usage summary and maximum frequencies in the implementation of the Cyclone 10 LP FPGA-based bit-stream quadratic converter.

<table>
<thead>
<tr>
<th>Bit Capacity</th>
<th>8</th>
<th>10</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource and Frequencies</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total logic elements</td>
<td>103</td>
<td>125</td>
<td>146</td>
</tr>
<tr>
<td>Total combinational functions</td>
<td>70</td>
<td>84</td>
<td>97</td>
</tr>
<tr>
<td>Logic element usage by number of LUT inputs</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-- 4 input functions</td>
<td>13</td>
<td>12</td>
<td>14</td>
</tr>
</tbody>
</table>
Figure 6b shows the simulation of the device operation when F1 and F2 constant-frequency signals are input to the device. The device reaches dynamic equilibrium at the beginning of the third period of operation. The result of the operation is represented by an integer; the fractional part of the result is lost. To avoid this, it is sufficient to increase the number of digits of the device, which allows scaling the result using coefficients equal to the integer power of the number 2. Additionally, the accuracy can be corrected by the duration of the internal PWM signal. It is desirable that its duration exceeds half of the period of frequency F2, which reduces the non-uniformity of internal bit streams. This non-uniformity is the cause of the loss of calculation accuracy.

5. Discussion

In this paper, we have proposed a circuit for a binary rate multiplier different from the binary rate multiplier traditionally used. The circuit is realized using standard digital logic modules. The proposed variant of the binary frequency multiplier, built on the basis of observing symmetry at the level of digital modules, has a regular structure. This allows to easily increase the digit capacity of the device and does not lead to the growth of the complexity of combinational circuits.

To evaluate the efficiency of our solution, we compared its characteristics with the rate multiplier implemented on the FPGA according to the traditional circuit [29]. For the correctness of the comparison of hardware costs, the auxiliary signals for controlling the operation of the multiplier and cascading were not used in the implementation of the traditional circuit. The RTL representation of the 8-bit circuit is given in Appendix A.

When realizing the circuit [29], it is required to use conjunctors with an increasing number of inputs: the inputs of the last conjuncor receive all output signals of the internal counter, reference frequency signals, and a bit of the input code. So, at the 8-bit realization of the multiplier, the number of inputs of the conjuncor is 10, 8 of which are the outputs of the counter. Two inputs are used for input code bit D and reference frequency F_ref. Obviously, the topological realization of such a device is complicated. When the device is implemented on an FPGA, the conjunction from many arguments is implemented sequentially, as shown in RTL Figure A1a in Appendix A. This implementation reduces the number of inputs on the conjuncor, but lengthens the combinational circuits, resulting in reduced circuit frequency and the manifestation of races, which in turn lead to undesirable switching events known as logic hazards. This phenomenon can be observed in the timing Figure A1c. The proposed symmetric architecture does not have this disadvantage.

The hardware costs of the traditional variant are shown in Table A1. The traditional variant is more economical, which is explained by doubling the number of triggers in the symmetric implementation, which leads to an increase in the number of cells used since each cell contains only one trigger. A comparison of the limiting frequencies shows that the traditional implementation can operate at higher frequencies. However, both the frequencies of the traditional implementation and the frequency characteristics of the proposed symmetric variant significantly exceed the frequency capabilities of the FPGA, so the frequency reduction cannot be considered a serious disadvantage limiting the application of the proposed architecture.

---

| -- 3 input functions | 28 | 40 | 47 |
| -- ≤ 2 input functions | 29 | 32 | 36 |
| Logic elements by mode | | | |
| -- normal mode | 38 | 44 | 49 |
| -- arithmetic mode | 32 | 40 | 48 |
| Total registers | 73 | 89 | 105 |
| Fmax (Dff), MHz | 118.57 | 167.73 | 162.13 |
| Fmax (F1), MHz | 132.56 | 159.64 | 155.67 |
| Fmax (Pll), MHz | 225.6 | 346.74 | 326.26 |
The proposed bit-stream multiplier-divider implements two complex operations, multiplication and division, which in the classical realization in both combinational and pipeline versions require significant costs. In this case, the calculation does not use elements of embedded DSP blocks; it removes restrictions on the families of FPGA chips that can be used to implement the binary multiplier-divider. As an alternative variant of the organization of calculations, it is possible to use traditional algorithms of arithmetic operations and their modifications.

We compared, in terms of hardware costs, our solution and implementation of the same function using library elements: counters for converting frequency streams into codes, multipliers, and dividers. The RTL of the resulting circuit is shown in Appendix B. The counters cnt_inst1 and cnt_inst2 are used to generate codes corresponding to the duration of the signals PWM1 and PWM2. The multiplier Mult_inst and the divider Div_inst are used to perform the corresponding arithmetic operations. D-flip-flop and logic elements provide a short pulse when the PWM1 signal rises; this short pulse is used to reset the counters. An external reset of the circuit is also provided. The peculiarity of the implementation is the use of the built-in multiplier, which imposes a limitation on the families of programmable chips suitable for the implementation of the device. Taking into account that embedded multipliers are nine-bit, increasing the digit capacity to 10 and more complicates the combination of multipliers and, consequently, complicates the tracing of the project, which can be observed in the analysis of Table A2. It is possible to avoid the use of embedded multipliers by implementing, for example, a matrix multiplier on logic cells, but this will lead to a significant increase in the hardware resources consumed. The stream frequencies that can be processed by the device exceed the operating frequencies of the proposed bit-stream symmetric architecture. In addition, the proposed design requires the device to operate for several periods corresponding to a transient to achieve the results. At the same time, the proposed architecture, realizing the tracking mode, compensates for short-term interferences and losses of single pulses and can be used for the computational processing of signals characterizing slowly changing processes.

The considered implementation can be modified in order to minimize hardware costs. Frequency multipliers built on the basis of a pair of counters can be realized on the basis of a single pair with copying of the combinational part for simultaneous multiplication by the input code and by the output code fixed in the feedback circuit.

As an example of such a tracking calculation, we can consider a temperature control implementation using a sensor with a PWM output. To process the sensor signal, the circuit can be simplified, as it is required to calculate the ratio of the duration of a single value of the input PWM signal to the duration of its zero value. In this case, instead of frequencies F1 and F2, it is enough to use one frequency, which will serve as a reference frequency.

Figure 7 shows a schematic of a multiplier-divider converter for a frequency temperature sensor controller, generating the output signal as a signal T with a constant duration of a single value and a variable duration of a zero value, e.g., [31].
This converter realizes the following function:

\[ N = \frac{A T_1}{T_2} \]  

(12)

where \( T_1 \) and \( T_2 \) are the durations of the one and zero values of the transducer output signal. The coefficient \( A \) is determined by the specific sensor model. Two symmetrical branches are realized in the circuit on a shared pair of counters, which are activated depending on the value of the input signal \( T \). At \( T = 1 \), the positive branch operates and forms a pulse stream for the summing input of the reversing counter. At \( T = 0 \), the negative branch operates and generates a flux for the subtractive input of the counter. The peculiarities of the operation of such a device at the behavioral level are shown in detail in [32].

The proposed quadratic bit-stream converter with symmetric architecture realizes the operation of squaring a function whose arguments are input frequencies. Function (11) can be calculated in the traditional way by counting the pulses of streams \( F_1 \) and \( F_2 \) over some time and performing appropriate transformations with the obtained values. Such an implementation is shown in Appendix C. In the implementation of a simplified approach based on the determination of the number of pulses of flow \( F_1 \) corresponding to the period of the signal \( F_2 \). To count the pulses, the counter \( \text{counter}_F1_r \) is used, and the result of its work is written to the register \( \text{reg}_F1 \). A shift chain of flip-flop \( \text{Pulse1} \) and \( \text{Pulse2} \) is used for the correct cooperation of the counter and register. It forms two short pulses after the occurrence of the pulse of the stream \( F_2 \). The first pulse is used to write the code from the counter to the register, and the second pulse is used to reset the counter. To form these pulses, an additional synchronization signal \( F_{\text{ref}} \) is introduced into the circuit; the frequency of this signal must exceed the frequency of \( F_1 \). An important requirement for the implementation of such a converter is the presence of FPGA-embedded multipliers.
that realize the squaring. Similar to multiplier-divider circuits, the project design with a bit capacity greater than 10 bits becomes more complex, and this is reflected in the hardware cost characteristics as shown in Table A3.

Another important feature of the proposed bit-stream devices is the absence of overflow when calculating the intermediate values. All calculations are performed on the basis of increment and decrement operations realized on reversible counters. If changes in digit capacity are taken into account when performing intermediate multiplication and division, the implementation will be more complicated and hardware costs will be higher.

The considered converters, in addition to the output code (Nout), form the output signal as a frequency stream (Fout), which will ensure reliable signal transmission to the next levels of the computing system.

In summary, the proposed bit-stream devices have the following advantages:

- lower hardware costs compared to devices realizing traditional arithmetic calculations;
- high reliability due to the use of negative feedback in devices and the presence of the dynamic equilibrium mode sought by the device;
- absence of necessity to use PLD with embedded multipliers;
- no overflow when performing intermediate calculations;
- obtaining the output signal in frequency form, which is formed naturally in the process of performing calculations.

6. Conclusions

The paper presents the results of the research on the influence of symmetry in the design of bit-stream converters on their characteristics. The research is based on the concept of symmetric construction of converters, taking into account symmetries both at the level of converter structures and at the level of processes occurring in them. It is shown that the approach to design taking into account symmetry allows for more effective technical solutions.

Looking into the future, several promising directions of research can be emphasized. First, the study of the proposed converter realizations in the form of ASIC fragments. We believe that taking symmetries into account at the level of integrated circuit topologies will provide additional advantages. In addition, the extension of our research to more complex converters realizing higher-order functions will allow us to expand the application area of bit-stream devices by improving their characteristics.

The proposed symmetric devices can be used in the design of miniaturized primary converters for frequency sensors, which can be placed side by side. First of all, such controllers will be effective for sensors used to control slowly changing parameters or inertial processes, e.g., rotational speed, temperature, and pressure.

**Author Contributions:** Conceptualization, O.B. and N.S.; methodology, O.B.; software, S.M.; validation, S.M. and N.S.; formal analysis, O.B.; investigation, S.M.; resources, N.S.; data curation, S.M.; writing—original draft preparation, O.B.; writing—review and editing, S.M.; visualization, S.M.; supervision, N.S.; project administration, O.B.; funding acquisition, O.B. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the Ministry of Science and Higher Education of the Russian Federation by Agreement No. 075-15-2022-291 dated 15 April 2022 on the provision of a grant in the form of subsidies from the federal budget for the implementation of state support for the establishment and development of the world-class scientific center «Pavlov Center for Integrative Physiology for Medicine, High-Tech Healthcare, and Stress Resilience Technologies».

**Conflicts of Interest:** The authors declare no conflict of interest.
Appendix A. Realization of Binary Frequency Multiplier according to the Traditional Circuit

Figure A1. Eight-bit binary rate multiplier: (a) RTL diagram obtained by compiling the VerilogHDL description with the Quartus II compiler; (b) timing diagram of multiplier operation for input code \( D = 15 \); and (c) timing diagram of multiplier operation for input code \( D = 127 \), illustrating the appearance of the logic hazard.
Table A1. Resources required for realization of the traditional binary rate multiplier and maximum clock frequencies depending on multiplier digit capacity.

<table>
<thead>
<tr>
<th>Bit Capacity</th>
<th>8</th>
<th>10</th>
<th>12</th>
<th>14</th>
<th>16</th>
<th>18</th>
<th>20</th>
<th>22</th>
<th>24</th>
<th>26</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total logic elements</td>
<td>16</td>
<td>20</td>
<td>26</td>
<td>31</td>
<td>33</td>
<td>39</td>
<td>42</td>
<td>50</td>
<td>54</td>
<td>58</td>
</tr>
<tr>
<td>Total combinational functions</td>
<td>16</td>
<td>20</td>
<td>26</td>
<td>31</td>
<td>33</td>
<td>39</td>
<td>42</td>
<td>50</td>
<td>54</td>
<td>58</td>
</tr>
<tr>
<td>Logic element usage by number of LUT inputs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-- 4 input functions</td>
<td>6</td>
<td>9</td>
<td>11</td>
<td>15</td>
<td>16</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>23</td>
<td>24</td>
</tr>
<tr>
<td>-- 3 input functions</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>8</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>-- ≤ 2 input functions</td>
<td>7</td>
<td>9</td>
<td>12</td>
<td>15</td>
<td>15</td>
<td>17</td>
<td>19</td>
<td>22</td>
<td>24</td>
<td>27</td>
</tr>
<tr>
<td>Logic elements by mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-- normal mode</td>
<td>10</td>
<td>12</td>
<td>16</td>
<td>19</td>
<td>19</td>
<td>23</td>
<td>24</td>
<td>30</td>
<td>32</td>
<td>34</td>
</tr>
<tr>
<td>-- arithmetic mode</td>
<td>6</td>
<td>8</td>
<td>10</td>
<td>12</td>
<td>14</td>
<td>16</td>
<td>18</td>
<td>20</td>
<td>22</td>
<td>24</td>
</tr>
<tr>
<td>Total registers</td>
<td>8</td>
<td>10</td>
<td>12</td>
<td>14</td>
<td>16</td>
<td>18</td>
<td>20</td>
<td>22</td>
<td>24</td>
<td>26</td>
</tr>
<tr>
<td>Fmax, MHz</td>
<td>619.5</td>
<td>540.2</td>
<td>542.0</td>
<td>532.4</td>
<td>517.6</td>
<td>483.5</td>
<td>461.8</td>
<td>434.4</td>
<td>417.8</td>
<td>391.7</td>
</tr>
</tbody>
</table>

Appendix B. Realization of Multiply-Divide Operation with Bit Streams Using Traditional Arithmetic Converters

Table A2. Resources required for realization of the multiplier-divider and maximum clock frequencies depending on multiplier digit capacity.

<table>
<thead>
<tr>
<th>Bit Capacity</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total logic elements</td>
<td>237</td>
<td>294</td>
<td>356</td>
</tr>
<tr>
<td>Total combinational functions</td>
<td>236</td>
<td>293</td>
<td>355</td>
</tr>
<tr>
<td>Logic element usage by number of LUT inputs</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-- 4 input functions</td>
<td>13</td>
<td>17</td>
<td>43</td>
</tr>
<tr>
<td>-- 3 input functions</td>
<td>185</td>
<td>233</td>
<td>262</td>
</tr>
<tr>
<td>-- ≤ 2 input functions</td>
<td>38</td>
<td>43</td>
<td>50</td>
</tr>
<tr>
<td>Logic elements by mode</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-- normal mode</td>
<td>119</td>
<td>147</td>
<td>177</td>
</tr>
<tr>
<td>-- arithmetic mode</td>
<td>117</td>
<td>146</td>
<td>178</td>
</tr>
<tr>
<td>Total registers</td>
<td>17</td>
<td>19</td>
<td>21</td>
</tr>
<tr>
<td>Embedded Multiplier 9-bit elements</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Fmax (F1), MHz</td>
<td>549.45</td>
<td>550.36</td>
<td>503.78</td>
</tr>
<tr>
<td>Fmax (F2), MHz</td>
<td>549.15</td>
<td>549.15</td>
<td>517.6</td>
</tr>
</tbody>
</table>
Appendix C. Realization of the Operation of Squaring of Stream Frequencies Using Traditional Arithmetic Converters

Figure A3. RTL diagram of the 8-bit quadratic converter.

Table A3. Resources required for realization of the quadratic converter and maximum clock frequencies depending on converter digit capacity.

<table>
<thead>
<tr>
<th>Bit Capacity</th>
<th>8</th>
<th>10</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total logic elements</td>
<td>109</td>
<td>157</td>
<td>212</td>
</tr>
<tr>
<td>Total combinational functions</td>
<td>91</td>
<td>135</td>
<td>188</td>
</tr>
<tr>
<td>Logic element usage by number of LUT inputs</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-- 4 input functions</td>
<td>32</td>
<td>50</td>
<td>74</td>
</tr>
<tr>
<td>-- 3 input functions</td>
<td>41</td>
<td>61</td>
<td>84</td>
</tr>
<tr>
<td>-- ≤ 2 input functions</td>
<td>18</td>
<td>24</td>
<td>30</td>
</tr>
<tr>
<td>Logic elements by mode</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-- normal mode</td>
<td>52</td>
<td>75</td>
<td>103</td>
</tr>
<tr>
<td>-- arithmetic mode</td>
<td>39</td>
<td>60</td>
<td>85</td>
</tr>
<tr>
<td>Total registers</td>
<td>26</td>
<td>32</td>
<td>36</td>
</tr>
<tr>
<td>Embedded Multiplier 9-bit elements</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Fmax (F1), MHz</td>
<td>60.48</td>
<td>46.33</td>
<td>36.11</td>
</tr>
<tr>
<td>Fmax (F2), MHz</td>
<td>1135.07</td>
<td>1165.5</td>
<td>1131.2</td>
</tr>
</tbody>
</table>

References


Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.