Overloaded CDMA Bus Topology for MPSoC Interconnect

Khaled E. Ahmed, Mohammed M. Farag
Electrical Engineering Department, Faculty of Engineering, Alexandria University, Alexandria, Egypt
Email: k.e.elsayed@ieee.org, mmorsy@alexu.edu.eg

***Abstract*—Intra-chip communication is a major bottleneck in modern multiprocessor system-on-chip (MPSoC) designs. The bus topology is the most common on-chip interconnect technology and bus contention in one of the major issues in bus-based MPSoC designs. Code division multiple access (CDMA) has been proposed as a bus sharing strategy to overcome the bus contention problem. In CDMA, a limited number of orthogonal spreading codes can share the medium due to the Multiple Access Interference (MAI) problem. In wireless communications, overloaded CDMA has been considered to increase the system capacity by adding extra non-orthogonal spreading codes with specific characteristics. We propose a novel CDMA bus architecture leveraging the overloaded CDMA concepts to increase the maximum number of cores sharing the same CDMA bus in MPSoC by 25% at a marginal cost. The overloaded CDMA bus architecture is illustrated, resource- and speed-efficient decoding circuits are presented, and a prototype system is implemented and validated on a Virtex-7 FPGA VC707 evaluation kit. The overloaded and ordinary CDMA bus architectures are compared in terms of resource usage, power consumption, bus operating clock frequency and bandwidth. Evaluation results show an improvement in resource utilization and power consumption per unit (IP core) and the bus bandwidth by approximately %25 while preserving the access delay of the ordinary CDMA bus.**

Keywords*—*MPSoC, CDMA, Bus Topology, On-Chip Interconnect, CDMA Bus, Multiple Access Interference, Overloaded CDMA.

I. Introduction

As transistor size scales down, the number of processing elements that can fit on a silicon die increases. Under scaling, the density of on-chip interconnect increases exponentially to satisfy the connectivity requirement of the MPSoC technology [1]. Such an increase in wiring density has a direct impact to the delay, area, and power consumption, which requires separating the computation and communication aspects of the design to optimize each separately. In modern MPSoCs the main bottleneck is communication rather than computation. Buses and Network-on-Chips (NoCs) are the two widely used technologies for intra-chip communication in MPSoCs [2].

FPGAs are a good candidate for hosting MPSoC designs due to their flexibility and performance. Modern FPGAs can host multiple heterogeneous coarse grained IP cores developed by various developers by applying design reuse and hierarchical design practices. Typical examples of IP cores are soft processors, DSPs, memory controllers, and hardware accelerators. In typical SoC design practices, individual IP cores and components are optimized by the developer leaving the task of connecting and interfacing IP cores and components in a MPSoC to the system designer. Communication between MPSoC modules hosted by FPGAs is implemented on the fine grained fabric. The communication latency needs to be minimized in high-performance MPSoC applications in order to achieve the best efficiency out of hardware parallelism offered by hardware platforms. Minimizing power consumption is another important objective for most modern MPSoC designs, especially those running on battery-enabled systems. Different low-power and latency-optimized communication architectures have been proposed for MPSoC designs implemented in reconfigurable hardware platforms [3].

The straightforward approach to realize on-chip communication is point-to-point links (P2P). The interconnect complexity of the P2P communication topology scales squarely with the number of on-chip cores [4] rendering it a feasible solution only for a small number of cores. Another common approach to realize on-chip communication is the bus topology prevailing contemporary MPSoC designs. In the bus interconnect topology, Time Division Multiple Access (TDMA) is applied, where all cores are connected to the same bus, where bus access is time shared between the different modules according to preassigned access priorities and bus arbitration rules.

As the number of on-chip components increases, the efficiency of the TDMA bus decreases due to the increased communication overhead on the bus. This problem is known as bus contention [5]. Some MPSoC designs attempt to overcome the bus contention problem by employing hierarchical bus topologies at the expense of increasing the interconnect complexity, overhead, and power consumption [6]. The NoC technology is an attractive alternative for both P2P and bus topologies. In NoCs data is treated as packets while IP cores and peripherals are considered as network points which are connected via routers and switching elements. NoCs are a feasible solution for large MPSoCs but they exhibit increased power consumption and large area overheads [7].

Code Division Multiple Access (CDMA) has been proposed as a modification to the bus topology to overcome the bus contention problem raised in TDMA-based bus systems [8]. Direct sequence CDMA (DS-CDMA) is a well known approach for medium sharing in wireless communication systems where the medium is shared by assigning orthogonal spreading codes called signatures to all transmit-receive pairs sharing the communication channel. Code orthogonality enables multiple access and is measured in terms of the cross-correlation between different codes which equals zero for orthogonal spreading codes. In a CDMA system, data from each transmitter is spread with a unique spreading code or signature. The signature codes sent by different transmitters are summed together in the additive communication channel.

The received signal at each receiver is composed of the sum of the spreading codes sent by different transmitters. Spreading codes can be identified on the receiver side by applying correlation operations to the received sum. Each receiver can then extract its intended data from the received channel sum by correlating it with the unique signature assigned for each transmit-receive pair. Other advantages of using CDMA for on-chip interconnect include reduced power consumption, fixed communication latency, and reduced system complexity [9].

The set of spreading codes used in a CDMA system must be orthogonal to each other and any extra codes added to this set will cause Multiple Access Interference (MAI) arising due to the non-zero cross correlation between non-orthogonal spreading codes. MAI can also appear due to the auto correlation between asynchronous orthogonal spreading codes. The MAI problem sets a limit on the maximum number of users in a CDMA communication system. Consequently, the number of IP cores that can simultaneously share the CDMA bus interconnect in MPSoC is limited by MAI. In most spreading code families, the maximum number of synchronous orthogonal codes of length N equals the code length itself.

State-of-the-art CDMA techniques in wireless communications consider adding non-orthogonal codes that can still be separated and identified on the receiver side to increase the CDMA system capacity. These techniques are known as overloaded CDMA techniques and are currently used in synchronous CDMA wireless communication systems [10]. The major drawback of overloaded CDMA is increasing the MAI effect due to the reliance on non-orthogonal codes with non-zero cross correlation which results in reducing the detection distance between different codes, complicating the signature identification and separation, and consequently increasing the bit error rate. In wireless channels, the MAI problem raised by employing non-orthogonal codes is aggravated due to the presence of non-deterministic additive and multiplicative random components in the received signal representing the channel noise and fading, respectively. Such random effects in wireless channels necessitate developing complex receiver structures to enable competent overloaded CDMA systems [10].

On the other hand, for a synchronous binary CDMA bus, the MAI value can be accurately evaluated in the absence of random channel effects appearing in wireless communication channels. Applying overloaded CDMA concepts in MPSoCs can increase the ordinary CDMA bus capacity by adding extra IP cores employing non-orthogonal spreading codes that can still be separated and identified on the receiver side while keeping the decoder complexity unchanged. In this paper, we investigate using overloaded CDMA to increase the CDMA bus capacity in MPSoC designs. We exploit the MAI problem, the synchronous nature of the MPSoC design, the binary system used to encode the channel sum, and the accumulator decoder architecture of the ordinary CDMA bus topology in a smart way to enable the overloaded CDMA bus topology.

We propose a set of MAI-enabled spreading codes for the overloaded CDMA bus topology and present a systematic approach to generate the MAI-enabled codes that can be appended to existing orthogonal code families. The number of MAI-enabled codes equals 25% of the orthogonal code set size, thus adding 25% more cores sharing the same bus. We present the overloaded CDMA bus architecture and provide both area- and speed-efficient decoding alternatives for the proposed bus topology. We compare the implementation results of the overloaded CDMA bus to the ordinary CDMA bus topology. The remaining of this paper is organized as follows: Related work and a brief background about the ordinary CDMA bus topology are presented in Section II. The overloaded CDMA bus architecture and decoding alternatives are presented in Section III. Performance evaluation in terms of resource utilization, maximum bus frequency, power consumption and bandwidth is introduced in Section IV. Conclusions and future work are portrayed in Section V.

II. Background

Most related work addressing the CDMA-based bus relies on orthogonal Walsh codes to enable bus sharing. Nikolic *et. al.* propose a scalable CDMA-based peripheral bus to decrease the number of parallel transfer lines of TDMA buses [11] and present the full CDMA-based bus system in [12]. Bus wrappers convert address and data from IP cores into CDMA encoding while control signals are not encoded to facilitate interconnection to other TDMA buses. Another CDMA bus implementation is compared to a TDMA split transaction bus in [8]. The results show that the CDMA bus outperforms the split transaction bus as the number of processors increases, since the CDMA bus avoids bus contention and queuing delays. CDMA and TDMA are combined in the CT-Bus where data is communicated over both the time and code domains [9]. The CT-Bus shows that the communication overhead of CDMA is less than TDMA as the CDMA bus controller is required to only assign spreading codes while the TDMA controller must perform arbitration every clock cycle. The CT-Bus performance surpasses its TDMA counterpart for the heterogeneous traffic flow since it combines the TDMA bus scalability with the CDMA channel continuity.

CDMA also has been utilized to enable intra-chip communication in NoC topologies. In [13], CDMA is used in the physical layer of the network, enabling collision-free communication between IP cores connected to a router due to the orthogonality property of CDMA codes. The provided network is also hierarchically scalable. The Orthogonal Variable Spreading Factor (OVSF) CDMA code family is exploited in an adaptive scheduling algorithm to enable variable data rate intra-chip communication [14]. The algorithm adapts the number of chips in the OVSF codes according to the number of packets that are required to pass through a node.

Most related work addressing CDMA on-chip interconnect investigate architectural and topological enhancements and performance evaluation for the conventional DS-CDMA communication scheme. In this work we address a different aspect of the CDMA technology for on-chip interconnect. We investigate increasing the bus capacity by applying CDMA expertise gained in the telecommunications field to existing CDMA bus architectures. We make use of the ordinary CDMA bus architecture presented by Nikolic *et. al.* in [11] with some modifications and inclusions to develop the overloaded CDMA bus architecture. Therefore, we present a brief background on the ordinary CDMA bus topology in this section.

Figure 1 shows the block diagram of the ordinary CDMA bus. The system is composed of a number of XOR encoders

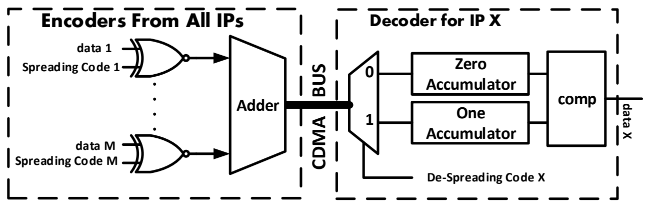

Fig. 1. MPSoC CDMA XOR encoder and Accumulator decoder

and accumulator-based decoders. In the encoder, an N-chip length orthogonal code is XORed with the data bit and sent out serially, indicating that a single bit is spread in a duration of N clock cycles. The relationship between the bus clock frequency f_b and the IP core clock frequency f_c is given by the following equation:

$$f_b = N f_c \tag{1}$$

The number of transmit-receive IP core pairs sharing the bus equals to M where $M \geq N$. For the ordinary CDMA bus topology using Walsh spreading codes $M = N$. Serial streams from all transmitting cores sharing the bus are added together and the sum is represented in binary and sent to a decoding circuit feeding the receiving IP cores. Binary encoding and signaling is preferred over multilevel signaling due to it is ready support in reconfigurable hardware and its superior performance and reliability. Data sent over the CDMA bus is given by the following equation:

$$Bus(i) = \sum_{j=1}^{M} d(j) \oplus S(j,i) \tag{2}$$

where $Bus(i)$ is an m-bit binary number representing the sum carried by the bus at the i^{th} clock cycle, the bus width $m = \lceil \log_2 M \rceil$, $d(j)$ is the data bit from the j^{th} encoder, $S(j,i)$ is the i^{th} chip of the spreading code of j^{th} encoder, and $\oplus$ indicates the XOR operation.

The decoder is implemented as a wrapper that cross correlates the serialized channel sum with the signature code assigned for the transmit-receive pair. The decoding process is periodic and the decoding cycle lasts for N clock cycles. The bus data is passed to the zero accumulator when the current chip value equals to "0" and to the one accumulator when the chip value equals to "1". At the u^{th} decoder, the input to the zero accumulator at the i^{th} cycle $In_z(i)$, and the input to the one accumulator $In_o(i)$ are given by:

$$\begin{aligned} In_z(i) &= \overline{D(u,i)} \cdot Bus(i) \\ In_o(i) &= D(u,i) \cdot Bus(i) \end{aligned} \tag{3}$$

where $D(u,i)$ is the despreading chip of the u^{th} decoder.

The one and zero accumulator circuits accumulate their inputs during the decoding cycle and are reset to zero at the beginning of each decoding cycle. The values hold by the zero and one accumulators are given by the following equations:

$$\begin{aligned} Acc_z &= \sum_{i=1, i\neq j}^{N} In_z(i) \\ Acc_o &= \sum_{j=1, j\neq i}^{N} In_o(i) \end{aligned} \tag{4}$$

where $0 < i,j \leq N$ and the indexes i and j do not take the same value for both Acc_z and Acc_o. Consequently, each accumulator adds $N/2$ different inputs during the decoding cycle because the spreading signature is balanced. At the end of the decoding cycle, the decoder has received the sum of spreading codes or their complements encoded according to the data sent by the transmit IP cores sharing the bus. Decoding the channel sum containing an orthogonal code or its complement using other orthogonal codes results in adding the same value to both accumulators. Decoding the channel sum containing an orthogonal code or its complement using the same code results in increasing the value of the one accumulator than the zero accumulator by the number of ones in the code, which equals to $N/2$ for balanced spreading codes, e.g. for $N = 8$, the difference between the two accumulators is 4. At the end of the decoding cycle, if the zero accumulator content is greater than the one accumulator content, the original data bit is "1"; otherwise, the original data bit is "0". The main advantage of the accumulator-based decoder is replacing the CDMA correlator receiver requiring computational-expensive multiplication operations with an addition-based receiver.

III. Overloaded CDMA bus

The idea behind using the MAI to increase the CDMA-based bus capacity emanates from the steady difference of $N/2$ between the one and zero accumulators at the end of the decoding cycle for all orthogonal codes sharing the bus. If an additional IP core needs to access the bus, this core will use a spreading code that is not orthogonal to the set of orthogonal spreading codes assigned to other cores. If non-orthogonal codes are added to the spreading codes, MAI will appear on the bus and the difference will deviate from $N/2$. If we can detect and identify the MAI value —the difference between the two accumulators in the ordinary CDMA bus— we would add extra spreading codes and, consequently, increase the bus capacity. Compared to wireless communication channels, absence of noise and other random effects in the binary CDMA bus reinforces this idea. The spreading code superset in the overloaded CDMA bus is the union of the orthogonal and MAI-enabled code sets.

Herein we present our approach to construct non-orthogonal but identifiable spreading codes to increase the CDMA bus capacity. Our main objective is increasing the number of IP cores sharing the ordinary CDMA bus while keeping the system complexity unchanged by using a simple encoding circuitry and relying on the accumulator-based decoder with minimal changes. To achieve this goal, we propose some modifications to the ordinary CDMA bus. Figure 2 shows the overloaded CDMA bus architecture for single bit interconnection. The same architecture is replicated for multi-bit CDMA bus. M transmit-receiver IP core pairs share the CDMA bus, where spread data from transmit IP cores are summed together using an arithmetic adder circuit having M binary inputs and an output of m-bit width. Each transmit and receive IP core is interfaced to an encoder or a decoder wrapper enabling data spreading and despreading.

A. Overloaded CDMA Encoder

Unlike orthogonal spreading codes which are XORed with the binary data bit, we utilize an AND gate to encode the

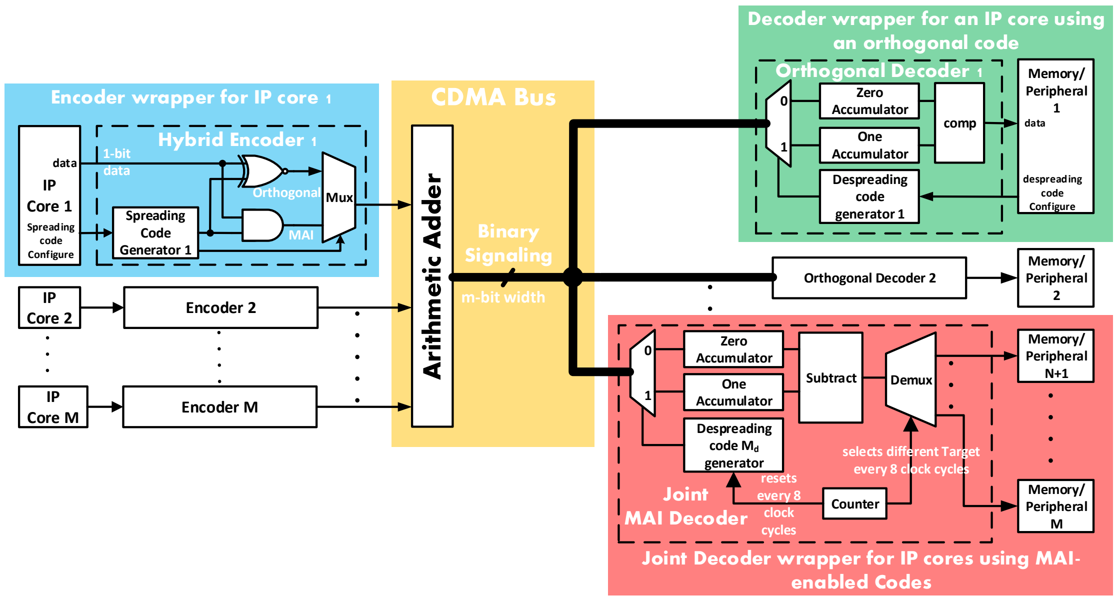

Fig. 2. A CDMA bus system containing the hybrid encoder, and both the orthogonal and overloaded CDMA joint decoders

non-orthogonal MAI-enabled spreading codes with the binary data bit. The AND gate encoder works as follows: if sent data is “0” it sends a stream of zeros that does not cause MAI on the channel and if sent data is “1” the encoder sends a non-orthogonal spreading code. Therefore the additional MAI-enabled spreading code will either contribute an MAI value of one or zero because the encoder is an AND gate. The XOR encoder of the ordinary CDMA bus cannot be used to encode the MAI-enabled spreading codes because it only complements the spreading code chips, so an XOR gate will cause MAI to the bus whether the data is “0” or “1”. A hybrid encoder is developed for both orthogonal and non-orthogonal spreading with an XOR gate, an AND gate, and a multiplexer unit as shown by Figure 2.

B. MAI-enabled Spreading and Despreading Codes

Before describing the bus decoder architecture, we will discuss how to construct the MAI-enabled codes for an arbitrary orthogonal code family. Let us consider a non-orthogonal MAI-enabled spreading code composed of a single chip of “1” and all the remaining chips are “0” in the N clock cycles — data encoding and decoding cycle— as shown in Figure 3 for $\mathbf{data_0}$. Assume this code is assigned to an extra IP core sharing the bus. When this core accesses the bus and sends “1”, this code is sent and the single chip of “1” is passed to either the one or zero accumulators based on the despreading code. This code will contribute an MAI value corresponding to the one chip, and the difference D between the accumulators will be:

$$D = \pm\frac{N}{2} + 1 \tag{5}$$

Similarly, we can add a code composed of two consequent chips in the N clock cycles as shown by Figure 3 for $\mathbf{data_1}$. This code will contribute an MAI value of two corresponding to the two chips of “1”, and the difference D between the two accumulators due to the this code is:

$$D = \pm\frac{N}{2} + 2 \tag{6}$$

This difference can be distinguished from the difference caused by the single-chip MAI-enabled code.

If the two MAI-enabled spreading codes are assigned to two IP cores and the data encoded simultaneously for both cores is “1”, the difference D between the one and zero accumulators due to the extra codes is:

$$D = \pm\frac{N}{2} + 3 \tag{7}$$

Because there are only two codes that cause MAI to the bus, then the only decomposition of the number 3 is 1+2 corresponding to MAI caused by the single-chip and two-chip codes, respectively, indicating that detection is still feasible.

Although these extra spreading codes cause MAI unlike orthogonal codes, the code can still be identified and extracted from other codes sharing the additive bus. For non-orthogonal code decoding, the difference contributed by the additional MAI-enabled codes can be detected using an ordinary accumulator decoder by replacing the comparator unit with a subtractor circuit. For orthogonal code decoding, the difference between the two accumulators is no longer $\pm N/2$, yet a comparator circuit can still compare the two accumulator contents and detect the original bit, accordingly. Errorless detection of data spread by an orthogonal code requires that the total MAI value contributed by non-orthogonal codes is less than $N/2$. Errorless detection of data spread by an MAI-enabled code requires two conditions: (i) the MAI value contributed by a non-orthogonal code must be decomposable meaning that it cannot be composed of MAI values caused by other codes; and (ii) the MAI chips must be passed to the same accumulator to prevent false detection which is controlled via the despreading code M_d of the MAI-enabled decoder. An example of false detection is as follows: if the first code adds an MAI value of one to the zero accumulator and the second code adds an MAI value of two to the one accumulator, the difference will be:

$$D = \pm\frac{N}{2} + 2 - 1 = \pm\frac{N}{2} + 1 \tag{8}$$

which can be confused with the single-chip code sent alone and cause fault detection.

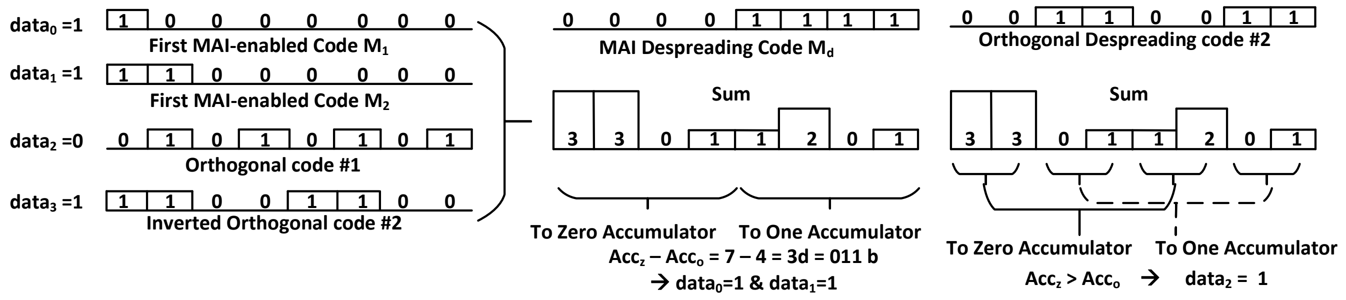
data₀ =1
First MAI-enabled Code M₁
data₁ =1
First MAI-enabled Code M₂
data₂ =0
Orthogonal code #1
data₃ =1
Inverted Orthogonal code #2
MAI Despreading Code M_d
Sum
To Zero Accumulator
To One Accumulator
Acc_z – Acc_o = 7 – 4 = 3d = 011 b
→ data₀=1 & data₁=1
Orthogonal Despreading code #2
Acc_z > Acc_o → data₂ = 1

Fig. 3. An example of a bus sharing MAI-enabled and orthogonal codes

To show the feasibility of the proposed approach, let us assume spreading codes of length $N = 8$, and let the two MAI-enabled codes be M_1 and M_2.

$$\begin{aligned} M_1 &= \{1,0,0,0,0,0,0,0\} \\ M_2 &= \{1,1,0,0,0,0,0,0\} \end{aligned} \tag{9}$$

The despreading code M_d is assigned for the two MAI-enabled codes M_1 and M_2 based on the errorless detection criteria stated above:

$$M_d = \{0,0,0,0,1,1,1,1\} \tag{10}$$

During the first four clock cycles in the decoding cycle, bus contents are passed to the zero accumulator while during the last four clock cycles bus contents are passed to the one accumulator. We should notice that the MAI-enabled codes M_1 and M_2 will increment the zero accumulator only by three ($3 = 2 + 1$), which satisfies all the errorless detection conditions. Figure 3 shows an example of the two MAI-enabled codes and two orthogonal codes sharing the bus simultaneously, and their decoding waveform. Generally, M_1, M_2 and M_d can be of any shape as long as all errorless detection conditions are satisfied.

For $N = 8$, ten IP cores can share the bus following our approach, eight cores using the orthogonal codes and two cores using the MAI-enabled codes. The decoder output R is the difference between the two accumulators A_o and A_z:

$$\begin{aligned} R &= Acc_z - Acc_o = \pm\frac{N}{2} + d_1 + 2d_2 \\ &= \pm 4 + d_1 + 2d_2 = \pm 2^2 + 2^1 d_2 + 2^0 d_1 \\ R_b &= \pm 1 d_2 d_1 \end{aligned} \tag{11}$$

where d_1 and d_2 are binary data of the two added cores respectively. R_b is the binary representation of the received value. Hence we can jointly extract the encoded data of the two cores from R_b.

For $N > 8$, the subsequent MAI value that can be contributed by another non-orthogonal code to facilitate errorless detection is 4 which cannot be decomposed into a combination of a single 1 and 2, the next MAI decomposable value is 8 which 8 cannot be expressed as a combination of a single 1, 2, and 4, and so on. Therefore, the series of contributed decomposable MAI for each new code is $\{1, 2, 4, 8, 16, 32, \cdots\}$, and the last element in the series must be less than $N/2$ to avoid fault detection. For example, for $N = 8$, the added MAI values can be $\{1, 2\}$, and for $N = 16$, the added MAI values can be $\{1, 2, 4\}$. The MAI-enabled set of non-orthogonal spreading and despreading codes will be constructed similar to the one- and two-chip MAI-enabled codes M_1 and M_2 and M_d.

So by doubling N only one additional code can be added to the set of MAI-enabled codes which might not be such a good improvement. However, we can increase the bus capacity by applying a more innovative idea. Assuming a code length of $N = 2^n > 8$, this code can be partitioned into blocks of 8-chip length yielding $N/8$ blocks, each is encoded and decoded independently. By applying this approach, each MAI-enabled code can be constructed from M_1 and M_2 which are assigned to a specific 8-chip block and the rest chips in each code are filled with zeros. For example, if $N = 16$, we can generate two codes by assigning M_1 and M_2 to the first 8 chips and the second 8 chips, respectively, which results in increasing the number of spreading codes by 4. We choose to partition the code into blocks of 8-chip length rather than other length because this leads to maximizing the number of MAI-enabled codes. By applying this approach, the total number of added codes for a code length of N is $2 * N/8 = N/4$. The decoding circuit uses the despreading code M_d to decode each 8 chips independently and sends the decoded data to the corresponding IP core. Figure 4 shows an example for the construction and decoding of the 4 MAI-enabled codes to spreading codes of length $N = 16$ chips. Figure 5 depicts the simulation results of data encoding and decoding in an overloaded CDMA bus using 16-chip spreading code length. As shown by the figure, data encoded of `20'h4D609` is correctly decoded after 16 bus clock cycles.

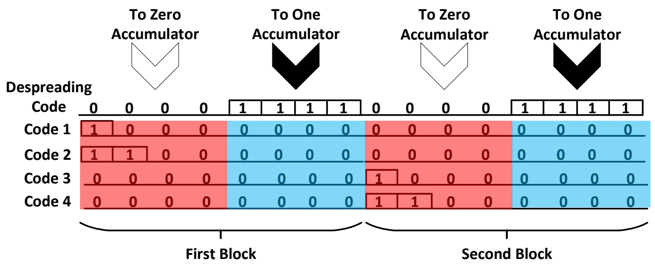

Fig. 4. The construction of 4 MAI codes in a 16-chip spreading code length, the despreading code sends the MAI chips to the zero accumulator

MAI-enabled spreading codes of length $N = 2^n$, and $N > 8$ can be generated recursively from the 8-chip MAI-enabled codes M_1 and M_2 using the following generation matrix:

$$M_8 = \begin{bmatrix} M_1 \\ M_2 \end{bmatrix} \tag{12}$$

$$M_N = \begin{bmatrix} M_{N/2} & 0_{\frac{N}{8} X \frac{N}{2}} \\ 0_{\frac{N}{8} X \frac{N}{2}} & M_{N/2} \end{bmatrix} \tag{13}$$

where M_N is the MAI-enabled spreading codes of length N. This set of MAI-enabled codes of length N are appended to the orthogonal code set to compose the superset of overloaded CDMA spreading codes.

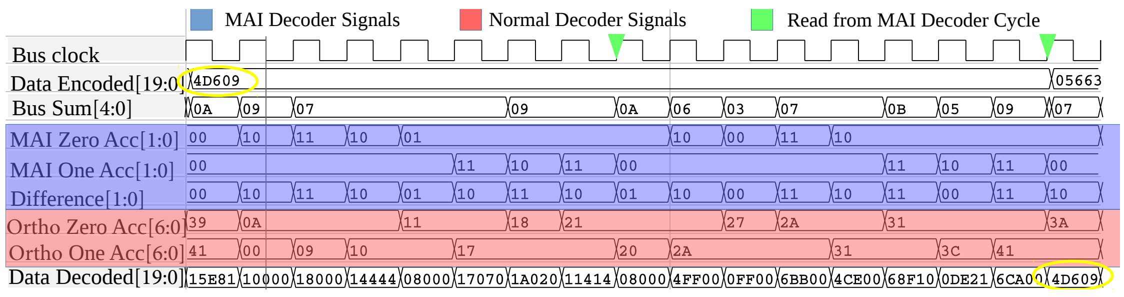

Fig. 5. Simulation results of data encoding and decoding in an overloaded CDMA bus using 16-chip spreading code length

C. Overloaded CDMA Decoder

We present two decoding schemes for the MAI-enabled codes, namely, separate decoders and joint decoder architectures. As the proposed accumulator decoder processes 8 chips independently, we can use $N/8$ separate decoders for the MAI-enabled codes where each decoder performs the correlation on its specific 8 chips using the M_d despreading code. Therefore, each extra IP core using an MAI-enabled code can have its own decoder that processes its specific 8 chips under the control of a controller circuit that assigns the MAI-enabled codes to the extra IP cores and manages their timing, accordingly.

As identical decoders that do not work in the same time slot are used, decoding resources can be shared efficiently without loss in performance. Therefore, only one joint decoder can be shared between the $N/8$ extra IP cores for the resource-optimized architecture. The joint decoder processes each 8-chip block independently, and sends the decoded data from each block to the desired IP core as shown by Figure 2. However, it should be noted that this approach imposes strict placement requirements on the extra IP cores to share the same decoder without needing long extra wires to route the decoded data to a specific IP which can lead to increasing the bus delay.

As illustrated by Figure 2, the transmit IP cores are interfaced to the encoder wrappers, and the receive memory/peripheral units (MPUs) are connected to the decoder wrappers. We apply a static code allocation scheme where each transmit-receive pair has a fixed signature code, the added $N/4$ decoders are connected to $N/4$ MPUs. There are $1.25N$ encoders and $N+1$ decoders, the decoders are decomposed to N orthogonal decoders and a joint MAI decoder that decodes data for the $N/4$ PUs as explained before. Encoders are configured by applying specific spreading codes according to the intended communication link. If the intended link uses orthogonal spreading code, the XOR encoder is selected; otherwise, the AND encoder is selected.

IV. Results

In this section we present the performance results of the overloaded CDMA bus. A system containing a number of IP cores and peripheral devices was built with full capacity, i.e. the number of IP cores is the maximum number offered by the bus. We study the performance of the encoder and decoder wrappers of the overloaded CDMA bus. We compare between the ordinary CDMA bus, overloaded CDMA bus with separate and joint decoder architectures. The joint decoder architecture is synthesized with an objective of minimizing the used resources, while the separate decoder architecture are synthesized with an objective of minimizing the critical-path delay. The separate decoders do not time share the accumulator-based decoders unlike the joint decoder. The system is implemented and validated on a Virtex-7 FPGA VC707 evaluation kit. Resource utilization in slices per IP core, maximum bus frequency, power consumption per IP core, and the bus bandwidth are shown in Figure 6. Synthesis results are evaluated and presented for spreading codes of length $N = \{8, 16, 32, 64\}$.

Resource utilization per IP core and power consumption in mW per IP core of the overloaded MAI CDMA bus is less than the ordinary CDMA bus because the overloaded CDMA increases the number of transmitting IP cores by 25% at a low overhead added by the encoder and decoder circuitry. The joint decoder area and power consumption is less than that of the separate decoder architecture as it uses only a single pair of accumulators. Bus frequency of the overloaded CDMA bus is less than the ordinary one due to the increase in the computation path at the decoder. Delay optimization enables the separate decoders to run at higher bus frequency. Also the bus frequency decreases with increasing N for both overloaded and ordinary CDMA buses due to increasing the computational complexity of the adders and accumulators. Pipelining the adder implementation can fix the bus frequency for different N values but at the expense of increasing the bus latency.

The drop in frequency is compensated by the increase in the offered bandwidth due to the overloaded IP cores as shown by Figure 6. The bus bandwidth is calculated for only a single bit per IP is interconnected via the CDMA bus. Generally, the CDMA bus bandwidth BW is given by the following equation:

$$BW = \frac{N_{bits} * f_b * M}{N} \tag{14}$$

where N_{bits} is the number of interconnected bits per IP, f_b is the bus frequency, M is the number of IP cores sharing the bus, and N is the spreading code length. The bandwidth has significant improvement, about 25% for $N = 8, 16,$ and 32, since the ratio $\frac{M}{N} = 1.25$ in the overloaded CDMA bus compared to the ratio of $\frac{M}{N} = 1$ in the ordinary bus.

V. Conclusions

In this paper, we presented the overloaded CDMA bus topology in which telecommunications research experiences are utilized to increase the CDMA bus capacity by about 25% while preserving the ordinary CDMA bus complexity. We smartly exploited the MAI problem, the synchronous nature of the MPSoC design, the binary signaling scheme used for data transfers in reconfigurable platforms, and the accumulator decoder architecture of the ordinary CDMA bus to enable the overloaded CDMA bus topology and increase the number of simultaneous cores sharing the bus. We presented a systematic

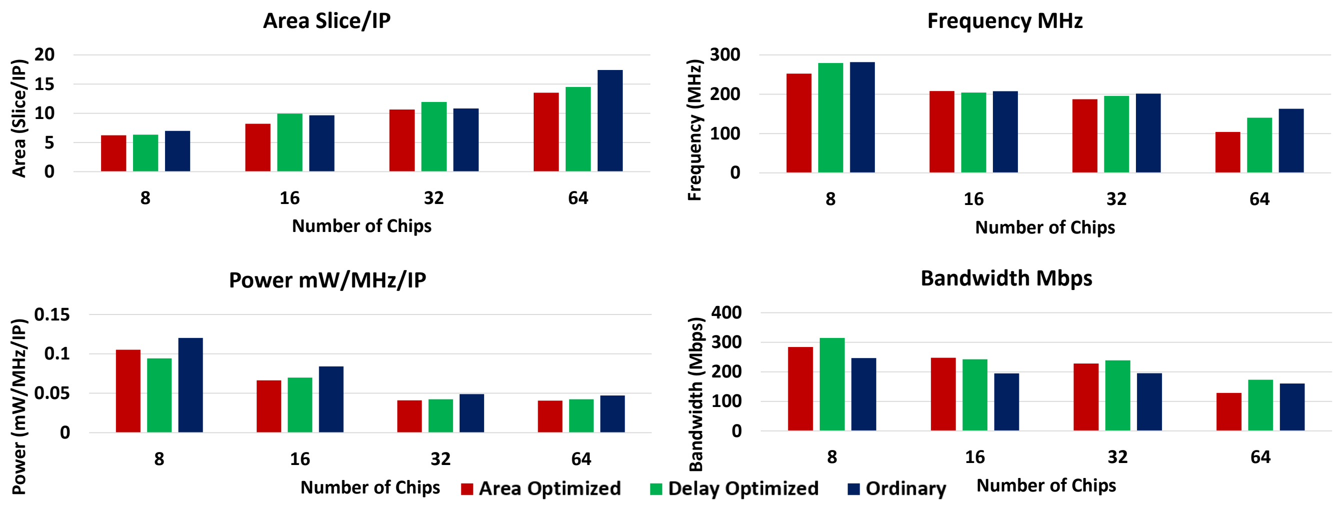

Fig. 6. Implementation results for the overload CDMA bus of length $N = \{8, 16, 32, 64\}$ on a Virtex-7 FPGA

method supported by a recursive code generation matrix to generate the MAI-enabled spreading codes to be appended to current orthogonal code families. We advanced the overloaded CDMA bus architecture and introduced different decoding circuits for resource and delay optimization purposes . We implemented and evaluated a prototype system on a Virtex-7 FPGA VC707 evaluation kit. Evaluation results depict an improvement in the resource utilization and power consumption per IP core, and bus bandwidth by approximately $\%25$ while keeping the ordinary CDMA bus clock frequency unchanged.

The presented approach can enable variable rate communication in the overloaded CDMA bus by assigning more than one MAI-enabled code to the same IP core which facilitates sending more than one data bit per IP core in a single bus cycle. Also extra non-simultaneous IP cores can share the bus using the same MAI-enabled codes while flipping the despreading code M_d to be $\{1, 1, 1, 1, 0, 0, 0, 0\}$. In this case, the one accumulator will receive all MAI chips and its accumulated value will be greater than the zero accumulator content which facilitates identifying the added codes. However, simultaneous use of the two M_d sequences will cause fault detection due to increasing both accumulators. Deeper investigation of the aforementioned points will be a part of our future work. Moreover, we will explore architectural enhancements and alternatives such as pipelining, resource sharing, and retiming of the presented overloaded CDMA bus topology.

Although we could investigate adding extra non-orthogonal codes that can be identified at the receiver, we presented the proposed approach to preserve the ordinary CDMA bus complexity. However, other codes can be added to cause identifiable MAI or cross correlation waveforms, but the decoder will be complicated to enable accurate code detection. Errorless overloaded CDMA and Welch-Bound codes [10] are examples of the code families that can be used in the overloaded CDMA bus yet require complex decoding circuitry. An example of a more complex decoder architecture is the accumulator-based but with cycle-by-cycle decoding capabilities. We will investigate using other overloaded CDMA codes to increase the CDMA bus capacity and study its effect on the bus complexity and other bus metrics in our future work.

References

[1] R. Ho, K.W. Mai, and M.A. Horowitz. The future of wires. *Proceedings of the IEEE*, 89(4):490–504, Apr 2001.

[2] Ling Wang, Jianye Hao, and Feixuan Wang. Bus-based and NoC infrastructure performance emulation and comparison. In *Information Technology: New Generations, 2009. ITNG '09. Sixth International Conference on*, pages 855–858, April 2009.

[3] T.S.T. Mak, P. Sedcole, P. Y K Cheung, and W. Luk. On-FPGA communication architectures and design factors. In *Field Programmable Logic and Applications, 2006. FPL '06. International Conference on*, pages 1–8, Aug 2006.

[4] Earl E. Swartzlander, Jr. *VLSI Signal Processing Systems*. Kluwer Academic Publishers, Norwell, MA, USA, 1986.

[5] Kwok-Tung Fung and H. C. Torng. On the analysis of memory conflicts and bus contentions in a multiple-microprocessor system. *Computers, IEEE Transactions on*, C-28(1):28–37, Jan 1979.

[6] M Mitic, M Stojcev, and Z Stamenkovic. An overview of SoC buses. In Vojin G Oklobdzija, editor, *Digital Systems and Applications*. CRC Press, 2007.

[7] S. Kumar, A Jantsch, J.-P. Soininen, M. Forsell, M. Millberg, J. Oberg, K. Tiensyrja, and A Hemani. A network on chip architecture and design methodology. In *VLSI, 2002. Proceedings. IEEE Computer Society Annual Symposium on*, pages 105–112, 2002.

[8] Jr. Bell, R.H., Chang Yong Kang, L. John, and E.E. Swartzlander. CDMA as a multiprocessor interconnect strategy. In *Signals, Systems and Computers, 2001. Conference Record of the Thirty-Fifth Asilomar Conference on*, volume 2, pages 1246–1250 vol.2, Nov 2001.

[9] B.-C.C. Lai, P. Schaumont, and I Verbauwhede. CT-bus: a heterogeneous CDMA/TDMA bus for future SoC. In *Signals, Systems and Computers, 2004. Conference Record of the Thirty-Eighth Asilomar Conference on*, volume 2, pages 1868–1872 Vol.2, Nov 2004.

[10] Seyed Amirhossein Hosseini, Omid Javidbakht, Pedram Pad, and Farrokh Marvasti. A review on synchronous CDMA systems: optimum overloaded codes, channel capacity, and power control. *EURASIP Journal on Wireless Communications and Networking*, (1):1–22, 2011.

[11] T. Nikolic, G. Djordjevic, and M. Stojcev. Simultaneous data transfers over peripheral bus using CDMA technique. In *Microelectronics, 2008. MIEL 2008. 26th International Conference on*, pages 437–440, 2008.

[12] Tatjana Nikolic, Mile Stojcev, and Goran Djordjevic. CDMA bus-based on-chip interconnect infrastructure. *Microelectronics Reliability*, 49(4):448 – 459, 2009.

[13] Daewook Kim, Manho Kim, and G.E. Sobelman. CDMA-based network-on-chip architecture. In *Circuits and Systems, 2004. Proceedings. The 2004 IEEE Asia-Pacific Conference on*, volume 1, pages 137–140 vol.1, Dec 2004.

[14] Manho Kim, Daewook Kim, and G.E. Sobelman. Adaptive scheduling for CDMA-based networks-on-chip. In *IEEE-NEWCAS Conference, 2005. The 3rd International*, pages 357–360, June 2005.