IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received November 20, 2020, accepted January 22, 2021, date of publication February 8, 2021, date of current version February 23, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3057866

Area-Efficient Parallel Reconfigurable Stream Processor for Symmetric Cryptograph

YUFEI ZHU[1], (Graduate Student Member, IEEE), ZUOCHENG XING[1], (Member, IEEE), JINHUI XUE[2], ZERUN LI[1], YIFAN HU[1], YANG ZHANG[1], AND YONGZHONG LI[1]

[1]National Laboratory for Parallel and Distributed Processing, National University of Defense Technology, Changsha 410073, China
[2]Institute of Information Science and Technology, Zhengzhou 450001, China

Corresponding author: Zuocheng Xing (zcxing@nudt.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61874140, and in part by the Hunan Science Project Foundation under Grant 2018xk2102.

ABSTRACT Represented by application-specific instruction set processors (ASIPs) and array processors, existing cryptographic processors face challenges in application to mobile terminals with sensitive security requirements. Typically, ASIPs have limited computational efficiency and algorithmic adaptability. An array processor requires massive circuits to perform fully expansive computations for ciphers, and such processors are unaffordable for terminals. To overcome these issues, we propose a highly area-efficient parallel reconfigurable symmetric cipher stream processor combining the characteristics of symmetric cryptograph stream-state processing and the advantages of stream architectures. This processor has a hierarchical stream architecture with multi-dimensional parallelism. It decouples cipher computation from data transmission, facilitating parallelism between algorithm operation and data scheduling. Furthermore, the processor fully excavates vertical pipeline parallelism and horizontal block parallelism in its operation processes. Additionally, it takes multi-granular cipher units and reconfigures computation circuits to map different algorithms efficiently. These mechanisms significantly improve its area efficiency and security intensity. The processor's prototype was verified and synthesised in a 65 nm CMOS process. Many typical ciphers were mapped onto the processor. Experimental results demonstrate excellent performance in feedback/no-feedback modes with a small area (1.26 mm^2). Therefore, its area efficiency is clearly higher than other cipher processors, and it has better prospects for future applications.

INDEX TERMS Stream processor, symmetric cryptograph, area efficiency, parallelism, reconfigurable.

I. INTRODUCTION

With the rapid development of information technology, information security faces diverse challenges. Information networks transmit massive amounts of data, including sensitive information related to business, finance, and daily life [1]–[7]. Therefore, achieving safe transmission and data storage has become a critical problem in open network environments. One effective method for solving this problem is to use cipher technology [8]–[10]. As the main symmetric cipher, block ciphers have the characteristics of fast speed and easy hardware realisation. Block ciphers are typically adopted as core algorithms to ensure data security in information networks, and they are widely used in the information security field.

As a pivotal hardware component for information security, cipher processors [11], [12] have broad application prospects for various types of information terminals [13]–[15], such as mobile terminals and other embedded terminals [16], [17]. Typical cipher processors are represented by application-specific instruction set processors (ASIPs) and array structure cipher processors. Unfortunately, their application prospects are restricted by performance limitations, area resources, and suitability issues. Furthermore, the complex application environments of sensitive information terminals require cipher cores that can flexibly support different algorithms to handle the developing attacks and equipment upgrade pressure. Currently, the application field of cipher processors is limited. Therefore, it is necessary to consider the features of information terminals and design novel cipher processors to meet their requirements.

ASIP cipher processors usually integrate several computational cores, which are suitable for certain types of algorithms or individual algorithms [19], [25]. However, ASIPs fail to effectively develop the parallelism existing in many cipher tasks and pay little attention to excavate the pipeline

The associate editor coordinating the review of this manuscript and approving it for publication was Dušan Grujić.

potentialities of cipher algorithms. These factors negatively affect the processing efficiency of ASIPs, and their long critical paths reduce working frequencies. Additionally, such processors typically have a fixed operation width, leading to mismatch problems when mapping different granularities of cipher algorithms [19], [20] significantly affecting the implementation performance of algorithms and reducing the algorithmic coverage of such processors. The operation units and instruction sets in cipher processors are specially designed for cipher tasks. Cipher operation units are customized units based on the operations required by related cipher algorithms. According to the execution processes of cipher algorithms, their hardware circuits are briefly designed to ensure that cipher processors can complete the encryption/decryption processes of the target algorithms. In general, the structures of ASIPs are relatively simple. They typically include several similar computing cores to perform cipher operations, small-capacity memory modules to support data caching [18], and simple control logic to control algorithm operations. Small memory modules make it difficult to achieve flexible scheduling and data organization for cipher information [21]–[23]. Therefore, it is difficult for such processors to meet the requirements of high-performance cipher computing [24]. Due to structural limitations and simple data transmission methods, it is difficult for ASIPs to achieve flexible algorithmic data mapping, implying that these processors fail to exploit the parallel processing potential of cipher tasks [25]. Additionally, their working frequency and hardware resource utilisation are relatively low. Overall, the cipher task processing performance of processors should be further improved.

An array structure cipher processor typically contains large-scale cipher operation units providing high processing performance [30]–[32]. However, array processors require enormous circuit resources to construct fully expansive operations for cipher algorithms, resulting in complex hardware structures and large areas. These processors are typically implemented based on coarse-grained cipher functional units, that construct processing arrays to process specific types of cipher algorithms using hierarchical network circuits. Functional units have relatively complex designs to enable them to support cipher operations widely. Array structure cipher processors typically contain huge operation modules and massive on-chip register resources. They also tend to adopt complex on-chip network routing structures and cumbersome communication modules. These hardware and software requirements are difficult to satisfy; thus, the costs of design and manufacturing are very high [22]. To achieve strong performance on algorithm processing, an array processor implements batch execution of cipher tasks by setting massive sets of isomorphic units repeatedly, improving the parallelism of cipher operations. Therefore, although the huge hardware resources of array processors achieve excellent performance, such processors have poor resource utilisation. Additionally, the chip area and power consumption of array processors are too large to be suitable for mobile terminals [16], which have strict requirements in terms of hardware cost, implying that the application range of array processors is obviously limited.

In general, existing typical cipher processors face a similar problem in that their encryption performance per unit area is relatively low in both cipher feedback modes and no-feedback modes (e.g., CBC mode and ECB mode). On-chip resources exist distinctly idle, resulting in low area efficiency. In practical applications, a designer must consider performance, area, and reconfiguration capabilities in the context of terminal requirements. Application demands usually aim to maximize performance under conditions with limited hardware area to achieve high area efficiency for cipher processing. Stream architecture is based on the concept of stream-state parallel data processing [36]–[38]; it has computational parallelism and decouples data operations from transmission scheduling, allowing such architectures to achieve high resource utilisation [26]–[28].

The remainder of this paper is organised as follows. Section II presents our proposed parallel reconfigurable round stream processing (PR^2SP) architecture. Section III presents a processor prototype called high area-efficiency reconfigurable block-cipher stream processor (HARBSP), describes three regions of HARBSP according to functional division, and illustrates the advantages of the processor. Section IV describes the designs of the cipher computation region and cipher operation clusters, including their internal computation units. Section V describes the circuit structure for cipher data scheduling in detail. Section VI presents the mapping of three typical cipher algorithms (SP structure, LM structure, and Feistel structure) on the processor prototype. Section VII presents experimental results and compares the processor to other cipher processors. Section VIII summarises the present work and provides our conclusions.

II. PR^2SP ARCHITECTURE

A stream architecture is an efficient flow processing architecture for computationally intensive applications. Such architectures develop the parallelism of stream processing by decomposing a computational process into combinations of operational element and decoupling data computation from stream transfers, which significantly improves processing performance and resource utilisation. We present the PR^2SP architecture for symmetric ciphers in Fig. 1. In Fig. 1(a), the upper box represents an off-chip host system, and the lower box represents a symmetric cipher stream processor. The cipher stream processor can be attached to the bus of the host system as a co-processor to accelerate the processing of cipher tasks. This paper focuses on the cipher stream processor.

The main modules of the stream processing architecture are divided into two parts: stream scheduling modules and stream computing modules. The processor executes instructions and processes relevant data, which are loaded as streams from the host system. Instructions and data in stream processing can be regarded as an indexed association queue in producer-consumer mode. When the flow rate of data in the

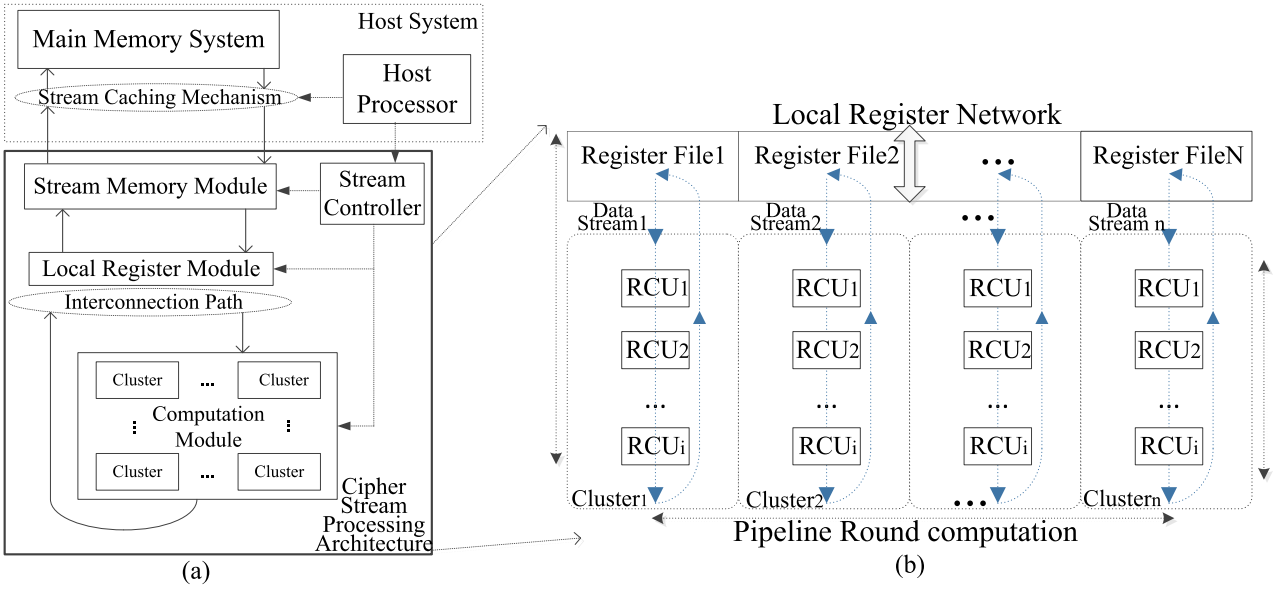


FIGURE 1. (a) Parallel reconfigurable round stream processing architecture (b) Parallel stream processing.

stream queue reaches a high value, and the spacing distribution is stable, a processor can process task streams with very high computational efficiency.

The symmetric cipher fully conforms to the following characteristics of stream processing.

1) The cipher algorithm has the characteristics of natural flow processing, and its operational process is highly deterministic. As shown in Fig. 1, the cipher task executes iterative pipeline operations based on a predetermined algorithm flow. According to the algorithm execution path, batches of message data flow through the relevant computing nodes like streams to realise algorithmic operations. In stream processing, the process of a cipher task is determined when the host system compiles an algorithm program into specific instructions. Therefore, the operation path and data path are highly deterministic.
2) The process of cipher computation can be decomposed into several relatively independent operations. The data sequentially flow through the relevant operational units, and the ciphertext results are generated through round operations. The final results are then output to the host system.
3) PR^{2}SP can decouple parallel computing and data scheduling. Cipher algorithms have the potential for parallel processing, including horizontal block and vertical thread parallel processing. Based on multi-level stream memory modules, algorithm operation and data scheduling are loosely coupled or even independent, which can effectively hide data scheduling delays or parallel computation delays.

PR^{2}SP fully integrates the stream-state processing characteristics of symmetric ciphers (block ciphers) with the advantages of a stream architecture. It was designed as a three-dimensional parallel stream architecture. This three-dimensional parallel structure is represented in Fig. 1(b) by three arrows. PR^{2}SP decouples cipher computation from data transmission, which facilitates parallelism between algorithm operation and data scheduling. Additionally, this architecture fully leverages the vertical pipeline parallelism and horizontal block parallelism in cipher algorithms based on the stream computation module and local register module. Fig. 1 illustrates the operational principles of PR^{2}SP. Within PR^{2}SP, the main structure is composed of a cipher computation region and register transmission region. The cipher computation region consists of multi-way operation clusters that are composed of reconfigurable cipher units (RCU). RCUs are determined on the common computational elements of block ciphers. The register transmission region corresponds to operational clusters in sequence, and the two sides form a joint pair to coordinate their works. Multi-path data streams run through the relevant RCUs of cipher operation clusters in pipeline form, and the data scheduling is performed based on a local register network. The local register network is composed of register files and transmits multi-batch data streams to the relevant operation units, while data are computed in the operation pipeline. This system matches the data cycles of the cipher round operations; therefore, the data load and output can adapt to the clock cycles of the algorithm operations. After the relevant operations are completed, the processed data are output. Based on this structure, PR^{2}SP can realise a multi-stream parallel deep pipeline and achieve high throughput.

A specific hardware mapping circuit is preconfigured for a target algorithm according to its operation paths. This efficiently leverages the parallelism between the horizontal blocks of cipher tasks and the parallelism of algorithm data pipeline iterations in the longitudinal direction. It also realises parallel pipeline operations on algorithm data on multiple cipher operation clusters, which fully leverages the operational efficiency of the hardware circuit. It provides a flexible operation structure and data organisation that can

efficiently map and process the majority of block cipher algorithms at multiple operation granularities. Based on the research of numerous block ciphers, we realised operation granularities for major ciphers including 8-bit, 16-bit, 32-bit, 64-bit, and 128-bit granularities. However, the data width of existing cipher processors is mostly 32-bit, implying that the implementation of different widths of cipher operations requires frequent algorithm sub-operations to complete multi-width data organisation. Therefore, the performance of cipher computation tends to be significantly impaired. Based on this consideration, this paper presents 8-bit basic operation units, 32-bit intra-cluster operation units, and the necessary large-width inter-cluster operation units. Some large-width operations can be achieved by combining 8-bit units or 32-bit units to meet the processing width requirements of different ciphers. We adopt a flexible data scheduling structure to achieve enhanced organisation and allocation of data streams. This data scheduling structure coordinates with cipher computation pipelines in parallel processes to improve processor execution efficiency.

In addition, the supporting compiler is a stream compiler developed for cipher tasks by the cooperative team. The relevant functions are performed by both the compiler and the cipher programs designed by the programmers. The programming model of the processor is VLIW, and the programming language is C. The cipher program explicitly tells the compiler which parts of the program can be parallelized, and the compiler automatically parallelizes those parts and translates them into VLIW instructions. The cipher program prompts the compiler as to which pieces of the program are executed in parallel and which pieces are executed serially. The parallel parts basically correspond to data operations and pipelines, and the serial parts mainly corresponds to data scheduling of source and result to/from host memory and cipher processor memory.

III. CIPHER PROCESSOR BASED ON THE PR^{2}SP ARCHITECTURE

Based on the PR^{2}SP architecture, we designed a processor called HARBSP. Based on block cipher processing characteristics, HARBSP focuses on improving the utilisation of hardware resources and takes full advantage of the pipeline parallelism of cipher tasks. By configuring the circuit structures of the cipher computation region and data scheduling region, it can reconfigure the corresponding processing structures for various block cipher algorithms. The processor is compile-time reconfigurable; it should be reconfigured according to the characteristics of the target algorithm. Fig. 2 presents the hardware structure of HARBSP. Its internal structure is modular and can be divided into three regions: stream control region, data schedule region, and cipher computation region.

The stream control region includes a stream controller, stream file bank, configuration information register (CIR), IO module, and host interface (HI). According to the target cipher task, control information and configuration information are generated by the off-chip host system, and they are loaded into the corresponding controller and configuration register in advance.

The stream controller schedules stream instructions and controls the execution of the corresponding functional components in the processor to complete stream transmission, data allocation, and kernel execution. The kernel represents the relevant cipher processes in the form of an operational instruction set. The stream controller is mainly composed of the following units: an instruction queue, an instruction decoder, a state analyser, and an issue unit. High-level stream programs running on a host system are compiled to generate low-level stream instructions. Each stream instruction contains relevant execution information, and the stream instructions are sent to the stream controller through the HI. The instruction queue stores the instructions loaded from the host system. The instruction queue consists of two dual-port RAM blocks: one for cipher operation instructions and another for data transmission instructions. The execution flows of cipher algorithms are regular and circular, and the operations of a cipher algorithm are limited. As a result, the number of instructions required to implement a cipher task is typically small, which implies that the instruction queue only requires two small-capacity RAM blocks. The state analyser is used to analyse the status signals from other function modules. It records the current state information in scoreboard circuits and makes logical judgments regarding execution instructions. The key units of the state analyser are the scoreboard circuits. The issue unit adopts a dual-issue structure, where one issue port is used for cipher operations, and the other is used for data transmission. The issue unit dynamically issues the corresponding instructions according to the free or busy states of components and the correlation of instruction executions. The controller works to coordinate the stream transmission components and cipher computation components to realise the parallel execution of data transmission and cipher task processing. Simultaneously, the stream controller broadcasts instructions to each compute cluster to realise multi-cluster parallel operations and complete cipher computation.

The configuration information in the CIR is stored in its register pages. The CIR contains four configuration pages corresponding to four operation clusters. Each page contains configuration information for the operational units in one cluster and joint configuration information between clusters. Before executing a cipher task, the cipher computation region and data schedule region preconfigure their circuits based on this configuration information to form special operation structures for cipher algorithms. Additionally, these methods assist in deep pipeline computations based on cipher round operations. Therefore, the CIR provides the reconfigurable computing capabilities of the processor. The details of the computation region are discussed in Section 5.

The stream file bank (SFB) is mainly used to cache and transmit the cipher task stream, which supports data scheduling and allocation of other modules. The SFB contains four

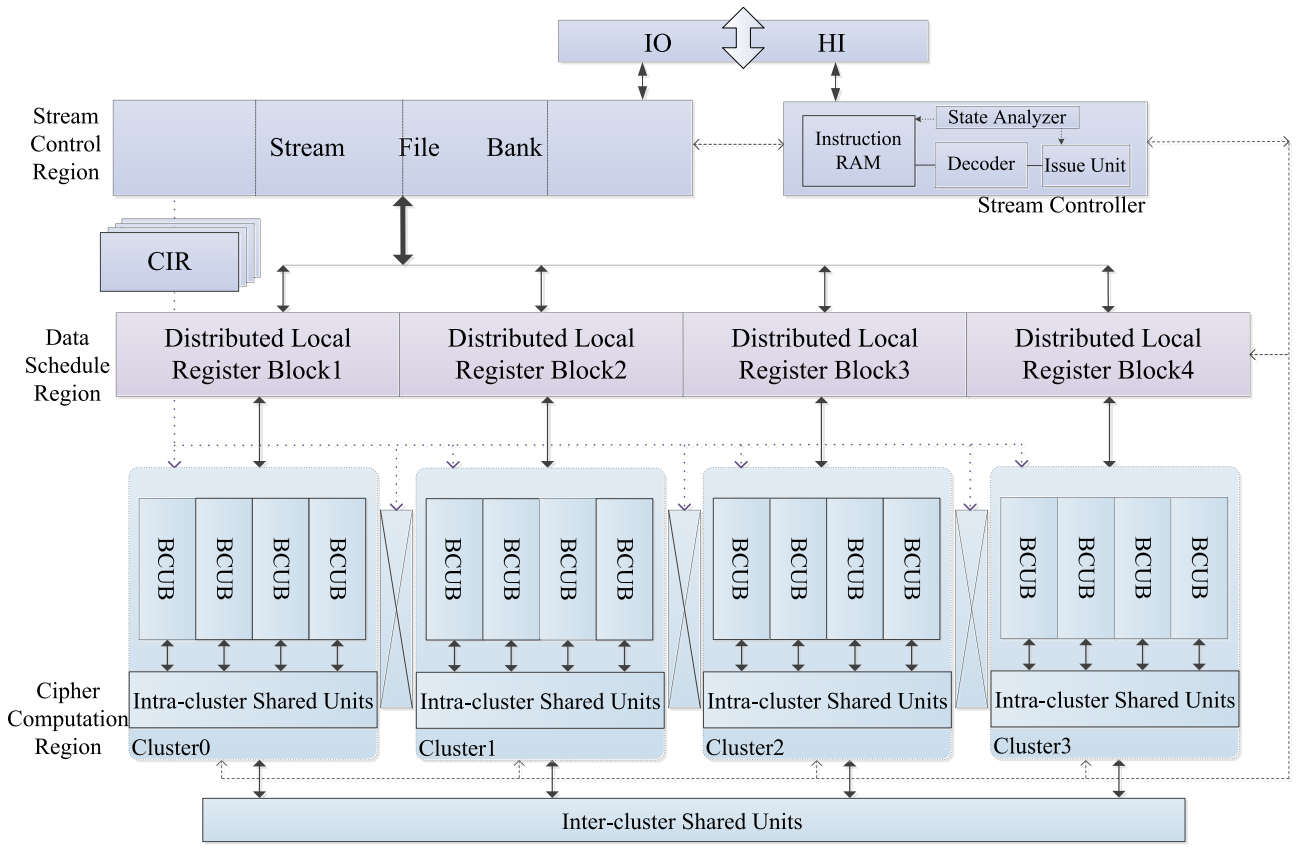


FIGURE 2. High area-efficiency reconfigurable block cipher stream processor.

sub-files that can cache input/output task streams. Therefore, it can realise parallel data stream transmission using local register blocks in the data scheduling region. The SFB cooperates with the data scheduling region to achieve stream access and data scheduling, which forms a hierarchical memory structure for data stream organisation. The IO and HI increase the extensibility and universality of HARBSP. Specifically, the IO and HI enable HARBSP to be used as a secure IP that is mounted on the bus of a host processor to realise cipher co-processing.

The data scheduling region mainly consists of distributed local register blocks (DLRB). The DLRB has flexible data organisation functions and is responsible for data interactions with the cipher computation region. It also temporarily stores intermediate results. Cipher operations are computationally intensive. However, by exploiting data parallelism and task parallelism, a large number of arithmetic operations can be implemented in parallel computation and pipeline execution. Then, the data path needs to provide source operands for cipher operation units and store intermediate results temporarily. The SFB cannot provide the required data paths and bandwidths for cipher operation clusters. Therefore, HARBSP sets up DLRBs corresponding to each cluster, which provide data registers to meet the requirements of the operation clusters. Because every block cipher algorithm only uses a specific set of cipher arithmetic units and the data path for algorithm operation is known, a special and compact DLRB structure was designed to reduce the required number of registers and optimise the utilisation of registers.

The cipher computation region focuses on the computation of cipher tasks and supports multiple-bit-width cipher operations. The cipher computation region is the core component for implementing cipher algorithms. The algorithmic structure of a block cipher mainly includes the SP structure, Feistel structure, and LM structure. By researching and analysing massive block cipher algorithms, we found that algorithms based on similar structures typically consist of the same basic arithmetic units. The algorithmic units involved in block cipher operations can be summarised as S-box units, logical operation units, modular addition/subtract units, modular multiplication units, Galois field multiplication units, shift units, and permutation units. The corresponding operation granularities can be divided into two groups. (1) The first are operation granularities within 32 bits, such as 32 bits, 16 bits, and 8 bits. Their characteristics are small operation widths and high parallelism. (2) The second are operation granularities above 32 bits, such as 64 bits, 96 bits, and 128 bits, which mainly include the shift operation and permutation operation.

As shown in Fig. 2, the cipher computation region mainly consists of four isomorphic cipher operation clusters and large bit-wide cluster common units shared by the four cipher operation clusters. Cipher operation clusters have

TABLE 1. Basic operations of block ciphers.

Cipher Algorithm	S-box	Logic	Modular Addition	Galois-Field Multiplication	Modular Multiplication	Shift	Permutation
AES	8×8	32 XOR		GF(2^8)		32	
IDEA		16 Xor	2^{16}		2^{16}+1		
SMS4	8×8	32 Xor				32	128-128
DES/ 3DES	6×4	32 Xor 48 Xor				28	64-64 32-48 48-32
Camellia	8×8	32 Xor				32	
SHARK	8×8		2^{64}	GF(2^8)			
SAFER+	8×8	8 Xor	2^8			8	
E2	8×8	8 Xor	2^{64}		2^{32}		64-64
KASUMI	7×7	32 Xor 32 And 32 Or				32	
CAST	8×32	32 Xor 32 And 32 Or	2^{32}				
RC6		32 Xor	2^{32}		2^{32}	32	
RC5		32 Xor	2^{32}			32	
Twofish	8×8	32 Xor	2^{32}	GF(2^8)		32	
FOX	8×8	64 Xor 64 Or		GF(2^8)			
CLEFIA	8×8	32 xor		GF(2^8)			
SERPENT	4×4	32 Xor				32	128-128
FEAL		8 Xor 32 Xor	2^8			8	
3-WAY	3×3	32 Xor				32	96-96
Mars	8×32	32 xor 32 and	2^{32}		2^{32}	32	
CRYPTON	8×8	8 Xor				32	128-128

reconfigurable functions that can be widely adapted to known block ciphers. Fig. 2 also presents the internal structure of a cipher operation cluster. A heterogeneous modular design is adopted in the clusters. One cluster consists of four identical 8-bit basic cipher-operation unit blocks (BCUBs). A BCUB contains multiple 8-bit reconfigurable RCUs. There are also 32-bit cipher operation units shared within a cluster (intra-cluster shared units). Besides, inter-cluster shared units are mainly used for large-bit-width operations in cipher algorithms and shared among four clusters. Therefore, the cipher computation region adopts reconfigurable technology, enabling the processor to configure heterogeneous operation units and data paths according to the type of cipher algorithm, bit width, and data stream characteristics of round operations. This means the processor can adapt to cipher tasks with different types of operations and bit widths, to satisfy different user needs for target algorithms.

Cipher algorithm computation and cipher data scheduling are the core components of the HARBSP, and this section has summarized the key elements of the control region, which will not be discussed further. The following sections will mainly discuss the cipher computation region and data scheduling region.

IV. CIPHER TASK COMPUTATION

Stream processing and block cipher operations have natural similarities. Block cipher has a structure of data stream cycle iteration. In the process of a block cipher algorithm, data blocks are executed in multi-round operations. A round operation is composed of corresponding cipher operation units in order. The related data blocks are inserted into the pipeline cycle intervals of cipher operations. Therefore, the block cipher task has evident features of round pipeline processing, which is as same as PR^2SP architecture.

Based on in-depth research and statistical analysis of a large number of typical block ciphers, we have summarised a list of basic operations of block ciphers, which is shown as Table 1.

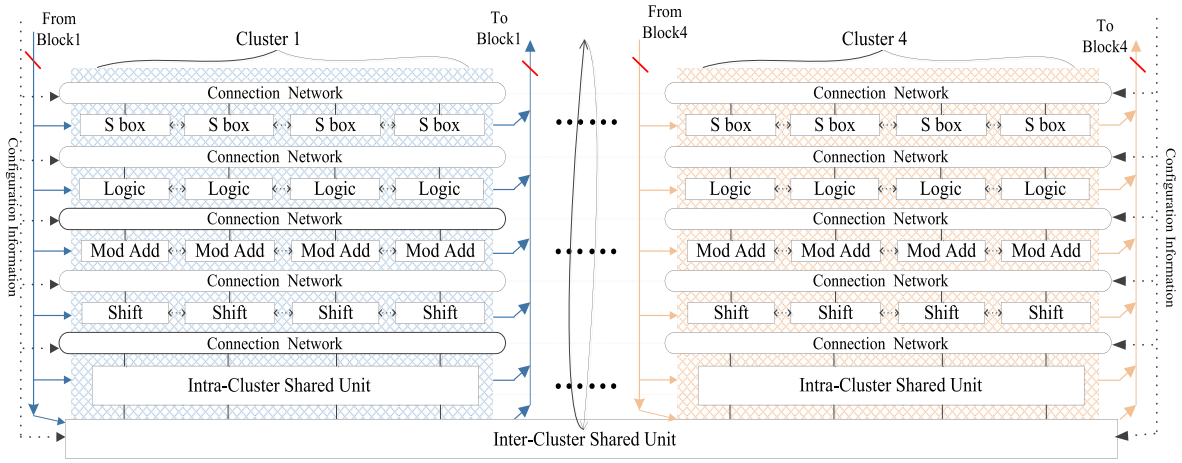


FIGURE 3. Cipher operation clusters.

Block ciphers usually contain multiple algorithm operations, and the operation granularities of different algorithms are usually diverse. The computation region of HARBSP appropriately chooses the high probability computation elements in cipher algorithms, and we convert these elements into hardware operation units. The operation units adopt a heterogeneous granularity design so that they can better satisfy mapping operations of various block ciphers.

In this study, we organically integrate a cipher processing structure with a stream processing structure. Therefore, we further propose stream-structured cipher operation clusters for block ciphers. These operation clusters aim to facilitate the existing algorithm operations in block ciphers. Considering operational performance, hardware cost, algorithm adaptability, and mapping flexibility, we designed reconfigurable cipher operation clusters. These clusters can develop the pipeline parallelism of cipher algorithms at a deep level. They provide high hardware utilisation, high algorithm coverage, and flexible mapping.

As shown in Fig. 3, the processor's cipher computation region consists of four isomorphic cipher operation clusters and 128-bit inter-cluster shared units. A cluster consists of four 8-bit BCUB and individual 32-bit intra-cluster shared units. These units have the configurable function. Figure. 3 shows the work structure of cipher operation clusters. Multiple blocks combine to implement cipher algorithms and complete large width operations based on configuration information. Four clusters share inter-cluster shared units to execute 128-bit operations.

Intra-cluster shared units mainly include modular multiplication units and Galois-field multiplication units. Inter-cluster shared units (including perm units) can be used in permutation operations and shift operations within 128/256 bits. Additionally, they can also be used as data communication units among cipher operation clusters.

For the HARBSP, the RCU, which has the longest delay, determines the critical paths of cipher operation modules. Here, we optimise the circuit structures of all RCUs, which effectively reduces the critical path delay of the RCUs. Based on an analysis of substantial block cipher operations, our design accounts for the coverage of cipher operations. We set up special operation units for cipher operations, which often appear in cipher computation, implying that such operations can be completed within one or two cycles. For the operations which less appear in cipher computation, we comprehensively consider combinations spending and reorganisation costs. Such operations are completed by combining or reorganising the basic functional units discussed above.

A thorough analysis of each operational unit reveals that S-box unit, logic unit, and modular addition/subtraction unit are the most frequently used units. Therefore, special RCUs should be set up for these units to support the corresponding basic operations. Furthermore, in 65nm process, the path delays of these units are less than 1 ns, compared with the entire operation clusters, that will not affect the critical path of clusters. Additionally, modular multiplication unit, finite field multiplication unit, and permutation unit, which are used at higher frequencies, have larger delays. Integrating these large-delay units in the computation region increases the critical path delay of cipher operations to two or three times. However, if such units are not integrated in cipher operation clusters, it typically requires multiple or even dozens of operations to achieve the corresponding cipher operations indirectly. Once these cipher operations were included, which would affect the computational performance of cipher algorithms. Then, this design overall considers the performance and efficiency of cipher operations. Considering the delays of these functional units, the operation times of the large-delay units are set to two or more cycles. HARBSP adopts cyclical pipeline operations and makes full use of the idle cycles within operations to enhance the performance of cipher processing. Additionally, the front/behind XOR binding S-box can effectively reduce the number of execution cycles.

BCUBs directly obtain the required data from DLRBs in the data scheduling region, and multiple BCUBs can realise the computation of large-width operations through a

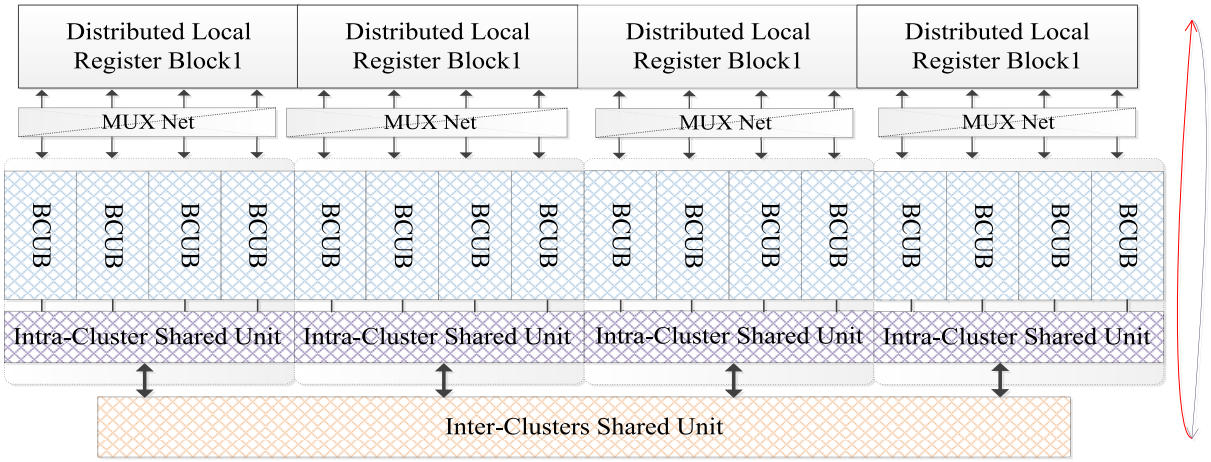


FIGURE 4. Cooperation between the cipher computation region and data scheduling region.

connection network. An intra-cluster shared unit (32 bits) can receive up to 32 bits of data from the corresponding DLRBs simultaneously. An inter-cluster shared unit (128 bits) can process (4×32) bits of data from the four operation clusters simultaneously. This method of mixing small-bit-width operation units with large-bit-width operation units can not only realise parallel execution of different widths operations but also reduce the number of required large-bit-width operation units. This design improves cipher computation performance and increases algorithmic compatibility and cluster resource utilisation.

It is a novel method to implement a reconfigurable cipher processor by using the operation units with heterogeneous granularities in the stream processor field. Each cluster can simultaneously support up to seven operation units to compute cipher data, which greatly improves the utilisation rate of hardware resources. These units lay a foundation for cipher operations in the pattern of stream parallelism. Before executing a task algorithm on the HARBSP, the operation paths of the algorithm are preconfigured according to the computing process of the cipher algorithm and the necessary operation units. Under the direction of preloaded configuration information, HARBSP determines the combination modes of relevant units and connection modes among operation clusters. Therefore, a hardware mapping special structure corresponding to a target cipher task is formed in the cipher computation region.

To ensure cooperation between the cipher computation region and data scheduling region, HARBSP contains special data stream channels between cipher operation clusters and the corresponding DLRBs. As shown in Fig. 4, each cluster is connected to the corresponding DLRB through a multiplexer network, which enables a pair of the cluster and DLRB to interact with the data stream rapidly. 8-bit BCUBs in clusters can connect with 8-bit register files in DLRBs rapidly. Similarly, a 32-bit DLRB ((4×8) bit file group) can directly communicate with the 32-bit intra-cluster shared units within an operation cluster. The combination of DLRBs can flexibly adapt to large-bit-width inter-cluster shared units. Therefore, the combination of four DLRBs can efficiently support eight-way 16-bit cipher operations, four-way 32-bit cipher operations, two-way 64-bit cipher operations, and one-way 128-bit cipher operations. Additionally, the large-bit-width operation unit has downward compatibility and can support multi-channel small-width parallel operations. The circuit structures of operation clusters adopt a design to promote granularity for heterogeneous units. Based on the 8-bit, 32-bit, and 128-bit granularity operation units in clusters, the cipher computation region and data scheduling region can form a combination structure that can adapt to different cipher granularities. These factors efficiently enable HARBSP to achieve high-performance stream processing under different cipher requirements.

V. CIPHER DATA SCHEDULING

A. DLRB CIRCUIT NETWORK

As shown in Fig. 2, HARBSP has a hierarchical distributed memory structure. The structure decouples the cipher operation pipeline from the data path pipeline and facilitates parallel execution between the computation process and data transmission process. These factors enhance the utilisation of hardware circuits and release the processor performance under limited resource conditions.

Fig. 5 presents the structure of the DLRB circuit network (DLRBCN). The circuit structure can flexibly organise and distribute data, which ensures efficient cipher data stream transmission and helps the computation region realise parallel pipeline processing.

Currently, typical cipher processors use 32 bits for their data width. Therefore, to process multi-granularity cipher data, it has been considered necessary and troublesome that such processors must execute fixed-granularity operations frequently to complete data organisation and the computation of multi-width cipher paths. These generally cause the number of cycles required for data scheduling to increase rapidly and significantly reduce the performance of

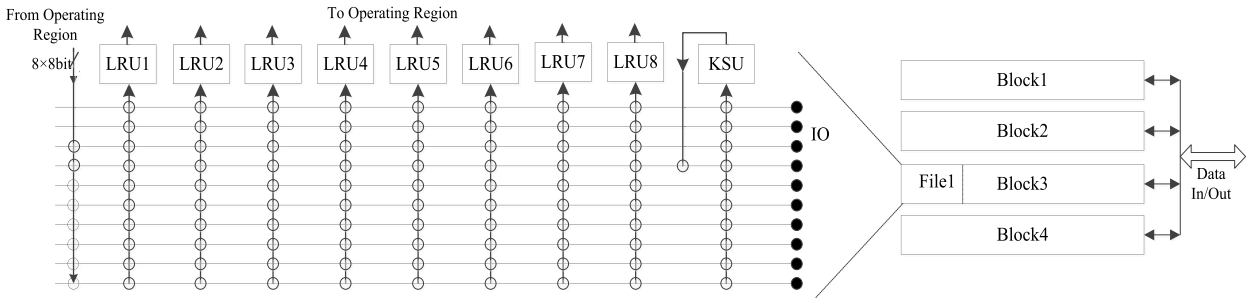


FIGURE 5. Distributed local register blocks' circuit network.

cipher operations. Additionally, block cipher algorithms contain various data paths. A cipher processor must consider algorithmic operation characteristics and use a flexible data scheduling network to adapt to different cipher algorithms. Typical cipher processors usually have simple memory structures and single data paths, which makes it difficult for them to achieve flexible cipher data mapping and scheduling.

The DLRBCN can preconfigure data paths according to the target algorithm structure, block width, and cipher mapping mode. It cooperates with the reconfigurable cipher operation clusters to realise stream-state data scheduling. As shown in Fig. 5, the DLRBCN contains four DLRBs. These DLRBs are called Block1, Block2, Block3, and Block4, and correspond to four cipher operation clusters. Each register block contains four sub-files called File1, File2, File3, and File4, which correspond to four BCUBs in a cipher operation cluster. According to the needs of cipher operations, each file can be directly transferred to the computation region in a single mode or combination mode. The internal structure of a file is presented in Fig. 5. This structure mainly consists of a crossbar network, local register units (LRUs), and a key scratchpad unit (KSU). The data width of a file is 8n bits, where n is an integer.

Fig. 5 also presents the circuit structure of the crossbar network. Each LRU has a vertical data path. The nodal circles in the figure represent selectable connection points between the LRU and horizontal data lines. The capacity of one LRU is 32 bits. One LRU selects and loads the cipher data from the data lines into the corresponding operation unit through a multiplexer. Each LRU chooses intermediate data to receive, which are then computed by the corresponding operation units. Additionally, LRUs register intermediate data to support parallel pipeline computation. Within the DLRBCN, multi-batch data blocks can be processed in vertical pipelines.

When it executes cipher operations, the data scheduling region interacts with the SFB at the times of initial data input and result output. During this process, LRUs store all operands and intermediate results. Each LRU has a write penetration function. Therefore, data transfers from the crossbar network to the cipher operation cluster can be accomplished in one clock cycle. The LRUs of one file can provide 8×8 bits of input data at a time, one BCUB in a cluster can also write back up to 8×8 bits of data at a time. A KSU is a shared unit within an operation cluster, and its memory capacity is 64×32 bits. A KSU can be used to register key information, and it uses a base-address-indexed addressing mode. Its address information is given by the address field in the corresponding VLIW instruction. A KSU can provide sub-keys for encryption/decryption round operations and register intermediate results or parameters, which reduces the register pressure on LRUs.

With this circuit structure, the data schedule region can not only satisfy the typical 32-bit width cipher operation, but it also adapts to the multi-width data organisations and cipher operations, such as 8-bit, 16-bit, and 64-bit data widths. Furthermore, the circuit structure of this region can flexibly support any data width from 8-bit to 128-bit (Data widths of known block ciphers are almost all 8n bits). Flexible data scheduling methods of DLRBCN effectively eliminate the tedious process of data organisation during multi-granularity operation processes. Moreover, DLRBs fully support parallel pipeline operations of different granularity ciphers.

B. CONFIGURATION OF THE DLRBCN

For a given cipher task, DLRBs can be preconfigured for the corresponding transmission structure. If the number of types of operation units required for cipher computation is typically less than five. Based on the statistical analysis of numerous block ciphers, we know that the common operation types of most ciphers conform to this rule. Therefore, we can set the circuit network to use a shared LRU mode. This means that an RCU used in a cipher algorithm can occupy two LRU data paths simultaneously, allowing a cipher task to realise multi-level pipeline operation, which can significantly improve the throughput of cipher computing.

Fig. 6 presents an example cipher task in which the used RCUs include logic unit, S-box unit, shift unit, and permutation unit. Based on the CIR configuration information, the DLRBs are all configured as an interconnected transmission structure.

A black dot indicates that an LRU data path has been selected to connect with this transverse data line. Each operation unit can occupy two LRU paths for deep pipeline registers based on the round stream characteristics of block ciphers.

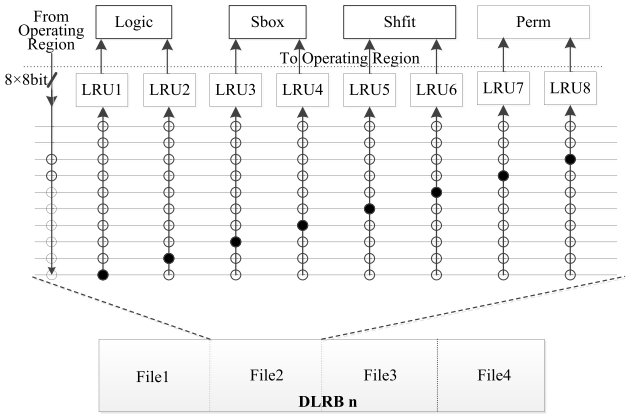


FIGURE 6. Configuration example A.

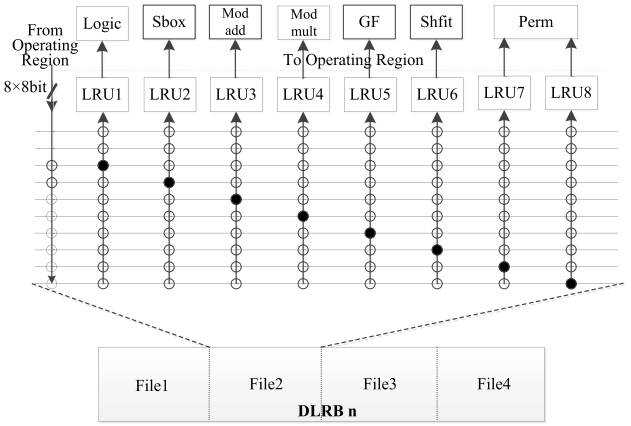


FIGURE 7. Configuration example B.

With four DLRBs cooperating with the four operation clusters in the computation region, HARBSP can efficiently realise three-dimensional parallel deep pipeline processing of cipher algorithms. Because a processor executes one cipher algorithm at a time, the circuit networks of the four DLRBs have the same configuration structure. Therefore, the four associated DLRBs can support the data stream transmission process of a cipher task.

When a cipher operation processes five or more types of operation units, the circuit network can be set to LRU exclusive mode. In this mode, one RCU has one LRU data path. If there are redundant LRU paths, they will be supplemented and allocated to intra-cluster shared units or inter-cluster shared units which are used in cipher operations. For example, Fig. 7 demonstrates that in some extreme cases, a cipher algorithm may require many types of operation units. The RCUs used in the example algorithm are logic unit, S-box, modular addition unit, modular multiplication unit, Galois field multiplication unit, shift unit, and permutation unit. In this case, the circuit network can be configured as a transmission structure. Each of the first six cipher operations allocates one LRU path, and the permutation operation can allocate two LRU data paths to meet its large-width operation requirement. Overall, one can observe that the DLRBCN can provide specific data scheduling for the processing characteristics of given cipher tasks.

VI. TYPICAL CIPHER ALGORITHMS MAPPED ON HARBSP

To evaluate HARBSP objectively and comprehensively, typical block cipher algorithms were mapped onto it based on parallel pipeline processing. In this section, the AES, IDEA, and SMS4 algorithms are discussed as representative algorithms. These algorithms represent three typical algorithm structures (SP structure, LM structure, and Feistel structure). HARBSP has carried out hardware mapping for each algorithm. The following subsections discuss the mapping process and parallelism development for all three algorithms.

A. HARDWARE MAPPING OF THE SP STRUCTURE CIPHER

As a representative of the SP structure cipher, the AES algorithm is widely used in various security fields. The block length and key length of the AES-128 algorithm are both 128 bits. Its operation process is mainly composed of four transformations: a sub-key plus transformation, byte substitution transformation, row shift transformation, and column mix transformation. The corresponding cipher transformations were mapped onto HARBSP as the following operations: XOR operation, S-box operation with pre-binding XOR, Galois field multiplication operation, and permutation operation. These operations correspond to the Logical unit (XOR), XOR-S-box unit, Galois Field unit (GF), and Perm unit, respectively, which are included in the cipher operation clusters.

As in Fig. 8, considering a no-feedback model as an example, when HARBSP maps the AES algorithm, four cipher operation clusters can realise intra-block parallelism. The 128 bits of block data occupy four cipher operation clusters. Additionally, in no-feedback mode, four clusters can also realise longitudinal stream parallelism for multi-batch data blocks. Because Galois field multiplication requires two clock cycles, the GF units are set to two cycles. HARBSP can then longitudinally execute multiple AES blocks in the same period.

B. HARDWARE MAPPING OF THE LM STRUCTURE CIPHER

Herein, the IDEA algorithm was selected as a representative LM structure cipher for mapping on the HARBSP. The IDEA algorithm consists of eight round transformations and one output transformation. Its block length is 64 bits, and its key length is 128 bits. Because the algorithm granularity of IDEA is 16 bits, it is convenient to construct the algorithm mapping circuit using fine-grained units. Based on the hierarchical distributed stream memory modules, when this cipher algorithm is mapped on the HARBSP, the modules can hide the delay times of memory access and data organisation. During the mapping of the IDEA algorithm, stream memory modules composed of SFBs and DLRBs efficiently support data scheduling for message streams. These modules flexibly schedule message blocks and key data to match the mapping mode of the IDEA algorithm.

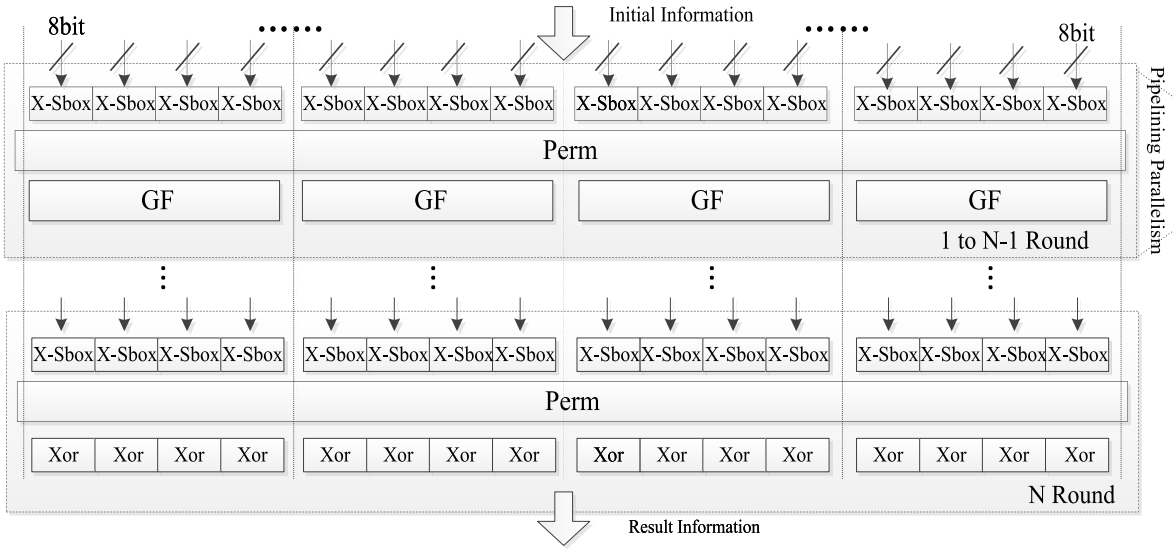


FIGURE 8. AES algorithm.

The processing of the IDEA algorithm includes three main types of operations: XOR operation, modular addition operation, and module multiplication operation. The corresponding operation units that are involved in the implementation of algorithm mapping include the logic unit (XOR), module addition unit, and module multiplication unit.

For cipher algorithms with block lengths less than 128 bits, multiple blocks can be simultaneously mapped on four operation clusters in no-feedback mode to achieve horizontal parallelism of multiple blocks. Furthermore, based on the circuit structure of the HARBSP, it fully supports pipeline round operations and realises vertical pipeline parallelism for multiple blocks. Therefore, HARBSP can develop horizontal and vertical parallelism for the IDEA algorithm simultaneously.

As shown in Fig. 9(a), IDEA is mapped in no-feedback mode. Message set A is divided into four separate files whose data blocks are sequentially mapped to the corresponding cipher operation clusters. Two 64-bit blocks can be executed on the pipeline of a single cluster, which effectively improves the utilisation of functional units within the clusters.

The data operation process based on these clusters is also shown in the figure. If four operation clusters operate simultaneously, the processor can support parallel execution of 8 blocks.

As shown in Fig. 9(b), in CBC mode, there are two separate message sets, A and B, which are divided into multiple batches of message packets. The data in the message packets is defined as follows.

Block A(1*1), Block A(1*2), Block A(1*3), Block A(1*4) ... Block A(i*1), Block A(i*2), Block A(i*3), Block A(i*4);

Block B(1*1), Block B(1*2), Block B(1*3), Block B(1*4) ... Block B(i*1), Block B(i*2), Block B(i*3), Block B(i*4).

Each packet contains four IDEA blocks, and each block is mapped in order on the corresponding cipher operation cluster. The start times of each cluster are separated by one algorithm cycle, and the resulting data from a previous cluster participates in the computation of the next block. These clusters use the time interval of the operation cycle to transfer data and complete data scheduling.

C. HARDWARE MAPPING OF THE FEISTEL STRUCTURE CIPHER

The Feistel structure is widely used in block cipher design. The SMS4 algorithm is considered as a mapping example. This algorithm is commonly applied in the field of information security. The block length and key length of the SMS4 algorithm are both 128 bits. SMS4 adopts a 32 round iteration structure. Its round function formula can be expressed as follows.

$$\begin{aligned} &F(X_i, X_{i+1}, X_{i+2}, X_{i+3}) \\ &\quad = X_i \oplus B \oplus (B <<< 2) \oplus (B <<< 10) \\ &\qquad \oplus (B <<< 18) \oplus (B <<< 24) \end{aligned}$$

and $B = S(X_{i+1} \oplus X_{i+2} \oplus X_{i+3} \oplus R_{ki})$.

Here, the operation process of the formula corresponds to a four-input XOR operation and an S-box operation. The main form of the round function is composed of four shift-binding XOR operations and a two-input XOR operation. These arithmetic operations are physically mapped onto HARBSP as two-input XOR units (double connection), S-box pre-binding XOR units, shift-binding XOR units, and XOR units. Cipher data then streams through these operation units. In the parallel mode of SMS4, stream memory modules efficiently schedule and organise message data and key data. In the no-feedback model, message sets are organised into multiple 128-bit groups that are recorded as Block1, Block2, Block3, and Block4.

Considering Block1 in Fig. 10 as an example, this block is composed of four 32-bit data (X_1, X_{1+1}, X_{1+2}, X_{1+3}) from bottom to top. The sequence from left to right is bits 0 to 7,

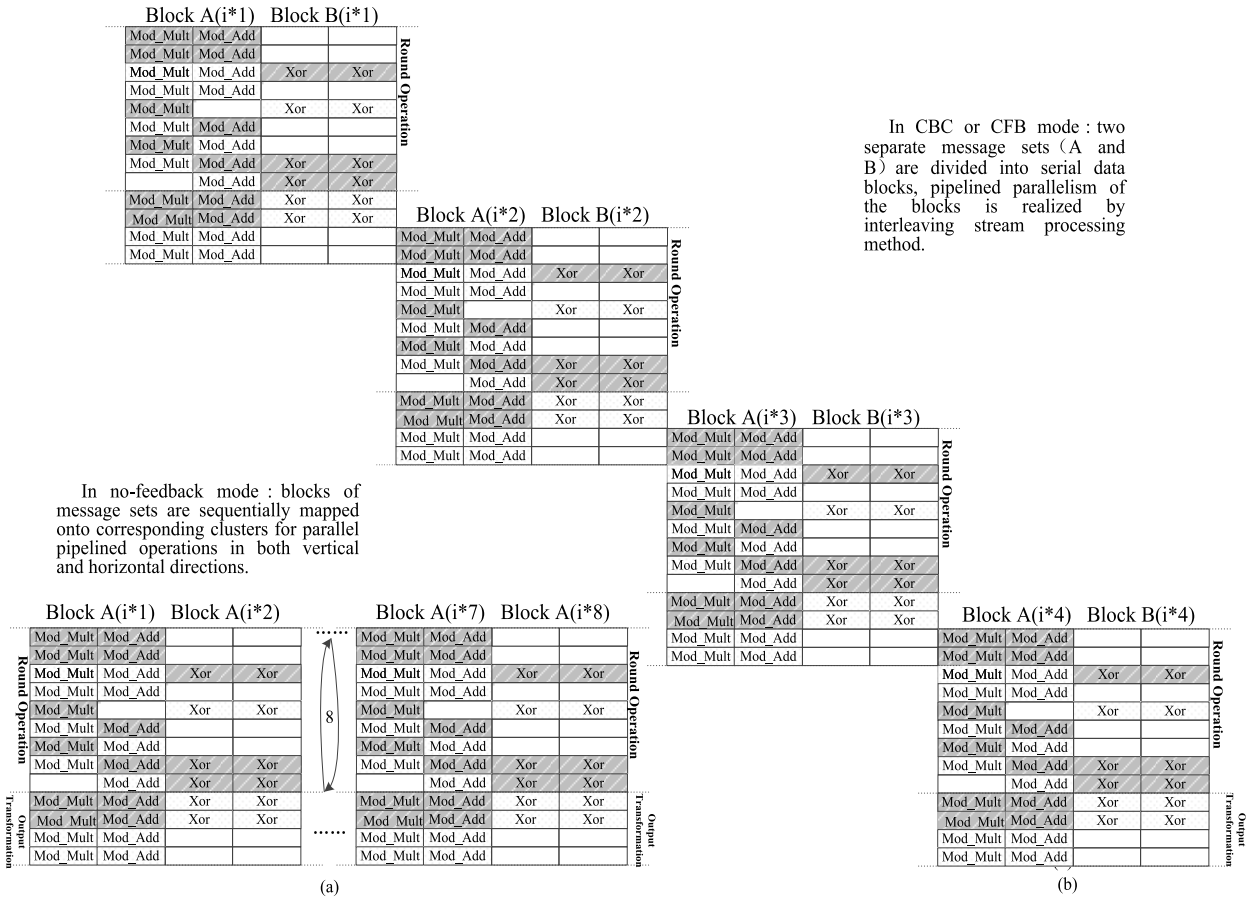


FIGURE 9. IDEA algorithm.

8 to 15, 16 to 23, and 24 to 31. The data bits of one sub-key byte participating in a round operation have a one-to-one correspondence to the data bits of one X_i byte. Four blocks are mapped in order onto four cipher operation clusters, and their algorithm round operations are performed in parallel. The resulting information is output following a reverse sequence transformation that is realised by the permutation unit in the final round.

D. BRIEF SUMMARY

We can configure hardware circuits into a specific structure according to the algorithm operations, round operation flow, and operation width of a cipher algorithm. The operation units necessary for the target algorithm are also configured and used. These units are also combined to implement the required computation pipeline. Operation clusters are reconfigured into an optimised arithmetic mapping circuit for the target cipher, which ensures that the processor can achieve optimal performance.

Based on our novel architecture design, HARBSP can realise longitudinal stream parallelism and horizontal block parallelism for cipher tasks using cipher operation clusters. Developing parallelism can significantly improve the data throughput and resource utilisation of the processor. For different cipher algorithms, appropriate mapping can be implemented to develop optimal parallelism and achieve high performance.

VII. EXPERIMENTS AND COMPARISONS

The RTL code of HARBSP was synthesised by a 65 nm CMOS process unit library and corresponding load models. The ASIC post synthesis implementation results are shown in Table 2. A typical power of HARBSP is 160 mW.

For the comprehensive evaluation of the HARBSP's algorithm-adaptive performance, typical algorithms were mapped onto HARBSP under the no-feedback mode (e.g., ECB mode) and feedback mode (e.g., CBC mode) in this work, such as AES, IDEA, SMS4, Camellia, DES, KASUMI, RC6. For the no-feedback mode, Fig. 11 presents the algorithm throughput of HARBSP.

We compared HARBSP to other cipher processors in terms of performance, area, frequency, and performance/area ratio. Table 3 presents the comparison information. Compared with the other cipher processors [29]–[35], HARBSP provides higher performance and a smaller area, and it obviously has multiple times advantage in performance/area ratio.

By comparing the data in Table 3, it is easy to find that in no-feedback mode, the cipher performance

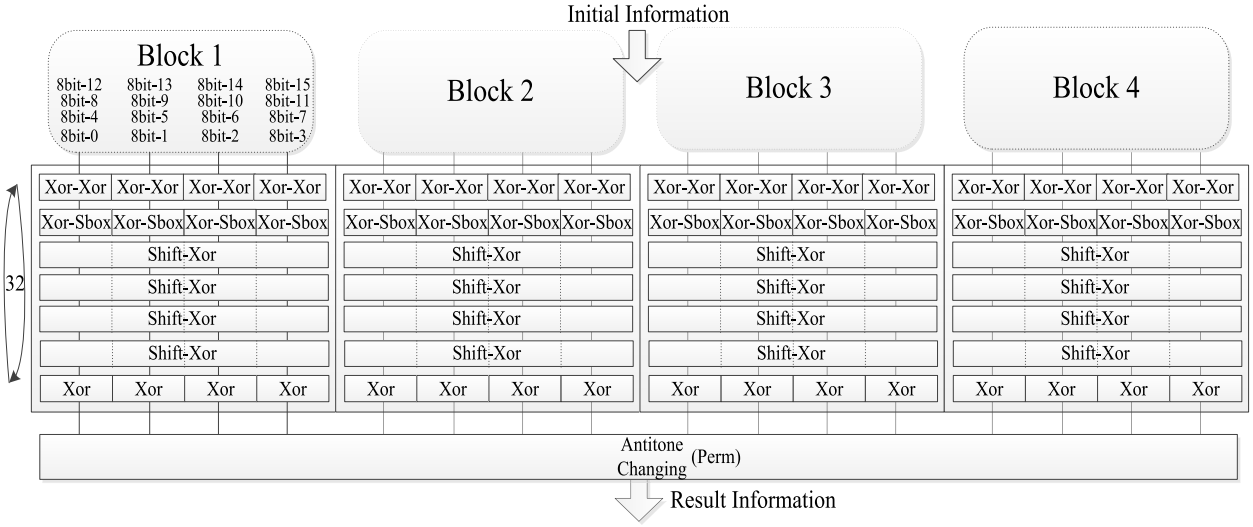


FIGURE 10. SMS4 algorithm.

TABLE 2. Result based on ASIC implementation.

Delay Time /ns	Area /mm^2	Slack /ns	Frequency /MHz
1.75	1.26	0.02	570

(throughput) of HARBSP is in the range of 1.22 Gbps to 7.19 Gbps. Its area-efficiency (performance/area ratio) is in the range of 0.97 Gbps/mm^2 to 5.71 Gbps/mm^2. In feedback mode, the throughput of HARBSP ranges from 0.39 Gbps to 1.93 Gbps, and its area efficiency is in the range of 0.31 Gbps/mm^2 to 1.53 Gbps/mm^2.

The area efficiency of HARBSP is compared to those of other state-of-the-art cipher processors in Table 3. We normalised the implementation processes of all processors to 65 nm. Based on this process size, the areas, frequencies, and performance are scaled to new values in brackets. Ref. [35] presented a reconfigurable ASIP cipher processor with a typical VLIW processor architecture. The cipher operation and data transmission of this processor are closely tied; the structure of its cipher operation module makes it difficult to leverage the characteristics of algorithm data to realise pipeline parallel computing. Compared to this processor, HARBSP has a clear advantage in terms of area efficiency (3.1 to 4.1 times greater). Ref. [32] presented an array structure cipher processor. Compared with this design, HARBSP has approximately 38 times greater area efficiency when implementing the AES algorithm. Ref. [33] presented an ASIP secure core used for the encryption and decryption of data in a secure processor. When mapping the AES algorithm, HARBSP has a significant area efficiency advantage over this processor. Ref. [34] presented an ASIP cipher processor, and we found that the area efficiency of HARBSP ranges from 1.6 to 2.3 times than that of this processor. The cipher processor presented in Ref. [29] is mainly composed of different ASIC cores, and each core corresponds to a specific algorithm. Its total cost was not provided, and the verification methods for different algorithms are different, making it difficult to evaluate its implementation area. However, the verified performance of HARBSP is significantly better than that of the related verification algorithms for this processor (MILENAGE is based on AES, meaning they are similar algorithms).

The array structure cipher processors [30], [31] have better performance in the SMS4 algorithm. However, for the target cipher algorithms, HARBSP still achieves approximately 1.2 to 11.4 times area efficiency than these processors. New array processors may provide enhanced performance, but for mobile terminals, the hardware resource overhead of array processors is enormous. Additionally, their application cost is very high, which makes it difficult to apply these processors in information terminals with strict requirements.

We also considered the AEGIS and ASCON algorithms, which were the final winning algorithms of the CAESAR competition, as examples of new authentication encryption algorithms. These algorithms were verified on the HARBSP. The verification tests for AEGIS and ASCON revealed the applicable performance (throughput) and area efficiency (performance/area). To the best of our knowledge, this is a new attempt to perform hardware verification of the two algorithms on this type cipher processors.

HARBSP fully optimises the utilisation of hardware resources. Its area efficiency is obviously higher than those of other cipher processors. Additionally, previous papers on relevant processors have considered fewer verification algorithms than our work. In general, when compared to other cipher processors, HARBSP has significant advantages in terms of flexible block cipher adaptation ability. Moreover, it achieves higher throughput and excellent performance/area ratio under different cipher work modes, and its area efficiency is about 2 to 38 times than those of other cipher processors.

TABLE 3. Comparison information.

Paper	Year	Process (nm)	Frequency (MHz)	Area (mm^2)	Mapping Algorithm	No-Feedback Mode Performance (Gbps)	Performance/ Area (Gbps/mm^2)	Feedback Mode Performance (Gbps)	Performance /Area (Gbps/mm^2)	Type
35	2016	180	350 (969*)	18 (2.35*)	AES IDEA DES	1.55(4.29*) 0.79(2.19*) 0.45(1.25*)	0.09(1.83*) 0.04(0.93*) 0.03(0.53*)	--- --- ---	--- --- ---	ASIP
30 31	2016	65	400	4.28	AES SMS4 DES Camellia	2.16 6.30 2.72 3.59	0.50 1.47 0.64 0.84	--- --- ---	--- --- ---	Array
32	2013	65	1210	6.63	AES	1.02	0.15	---	---	Array
33	2019	32	500 (246*)	1 (4.13*)	AES	0.20(0.10*)	0.20(0.02*)	---	---	ASIP
34	2014	130	238 (476*)	8.75 (2.19*)	AES DES IDEA	2.87(5.74*) 1.44(2.88*) 1.46(2.92*)	0.33(2.62*) 0.17(1.31*) 0.16(1.33*)	0.40(0.80*) --- ---	0.05(0.37*) --- ---	ASIP
29♦	2018	FPGA	150 1096	--- ---	KASUMI MILENAGE (Based on AES)	1.16 0.29	--- ---	--- ---	--- ---	Algorithm Cores
This Work	2020	65	570	1.26	AES IDEA SMS4 Camellia DES KASUMI RC6	7.19 3.80 2.28 3.51 2.75 1.22 1.89	5.71 3.02 1.81 2.79 2.18 0.97 1.50	1.93 0.95 0.57 0.75 0.68 0.39 0.65	1.53 0.75 0.45 0.60 0.54 0.31 0.52	Cipher Stream Processor
					AEGIS-128L	0.61 Gbps		0.48 Gbps/mm^2		
					ASCON-128	0.53 Gbps		0.42 Gbps/mm^2		

*Areas are scaled as P^2, frequencies are scaled as 1/P, where P is the process-size ratio for 65 nm.
♦ Area is not provided.

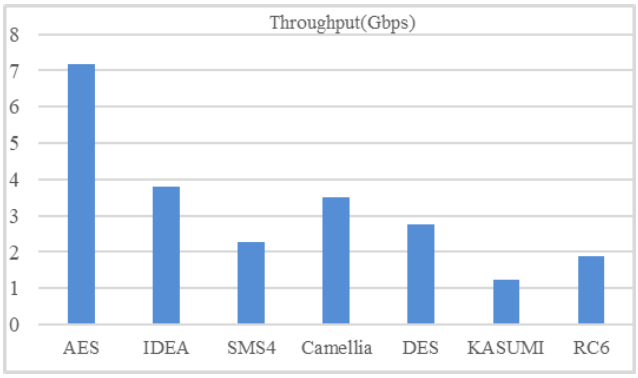


FIGURE 11. Algorithm throughput.

VIII. CONCLUSION

In this paper, we proposed the PR^2SP architecture and presented the HARBSP. HARBSP was implemented as a reconfigurable cipher stream processor with high area efficiency. HARBSP has two main advantages for cipher processing. (1) It can effectively realise the stream-state parallel computing advantages of block ciphers based on the PR^2SP architecture. It also decouples the stream scheduling process from the cipher computing process of task streams. It also extensively excavates the parallel processing potential of cipher algorithms to improve hardware utilisation and the degree of parallelism significantly. (2) HARBSP adopts reconfigurable technology that can flexibly execute a wide range of symmetric cipher algorithms. It avoids the issues faced by other cipher processors with fixed granularity in terms of matching multi-granularity operations from different algorithms, which effectively reduces the number of data operation cycles.

HARBSP has a hierarchical circuit structure. Its cipher computation region and data scheduling region cooperate to fully develop the three-dimensional parallelism of cipher tasks in both the vertical and horizontal directions. Experimental results demonstrated that HARBSP achieves excellent cipher processing performance with flexible algorithm adaptation and superior area efficiency. It can be widely used in resource-constrained application scenarios, such as sensitive mobile terminals. Therefore, under increasingly critical information security conditions, it can effectively protect users' important information with high security intensity.

REFERENCES

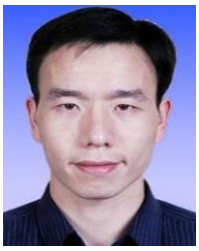
[1] *5G Network Security Requirements And Architecture*, Standard IMT-2020 (5G) Promotion Group, 2017.
[2] *5G Security-Scenarios and Solutions*, Ericsson, Stockholm, Sweden, 2017.
[3] *3G Security, Security Architecture*, document TS 33.102 V12.1.0, 3GPP, 2014.
[4] *3Gpp System Architecture Evolution; Security Architecture*, document TS 33.401 v15.0.0, 3GPP, 2017.
[5] *5G Network Technology Architecture*, Standard IMT-2020 (5G) Promotion Group, 2015.

[6] *View on 5G Architecture*, 5G PPP, V 1.0, Eur. Union, Brussels, Belgium, 2016.
[7] *Study on the Security Aspects of the Next Generation System*, document TR 33.899 V1.1.0, 3GPP, 2017.
[8] Q. Wu, B. Qin, L. Zhang, J. Domingo-Ferrer, O. Farras, and J. A. Manjon, "Contributory broadcast encryption with efficient encryption and short ciphertexts," *IEEE Trans. Comput.*, vol. 65, no. 2, pp. 466–479, Feb. 2016.
[9] M. Iwamoto, K. Ohta, and J. Shikata, "Security formalizations and their relationships for encryption and key agreement in information-theoretic cryptography," *IEEE Trans. Inf. Theory*, vol. 64, no. 1, pp. 654–685, Jan. 2018.
[10] S. Wang, J. Zhou, J. K. Liu, J. Yu, J. Chen, and W. Xie, "An efficient file hierarchy attribute-based encryption scheme in cloud computing," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 6, pp. 1265–1277, Jun. 2016.
[11] C. Debrup and R. Francisco, "Block cipher modes of operation from a hardware implementation perspective," in *Cryptographic Engineering*. Boston, MA, USA: Springer, 2009, pp. 321–363.
[12] L. Bossuet, M. Grand, and L. Gaspar, "Architectures of flexible symmetric key crypto engines—A survey: From hardware coprocessor to multi-crypto-processor system on chip," *ACM Comput. Surv.*, vol. 45, no. 4, pp. 115–123, 2013.
[13] N. Anastasios and S. Nicolas, "LTE/SAE security issues on 4G wireless networks," *IEEE Security Privacy*, vol. 45, no. 4, pp. 115–123, Apr. 2013.
[14] S. Dang, O. Amin, B. Shihada, and M. S. Alouini MS, "What should 6G be?" *Nature Electron.*, vol. 3, no. 1, pp. 20–29, 2020.
[15] R. Khan, P. Kumar, D. N. K. Jayakody, and M. Liyanage, "A survey on security and privacy of 5G technologies: Potential solutions, recent advancements, and future directions," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 1, pp. 196–248, 1st Quart., 2020.
[16] S. Hessel, D. Szczesny, N. Lohmann, A. Bilgic, and J. Hausner, "Implementation and benchmarking of hardware accelerators for ciphering in LTE terminals," in *Proc. GLOBECOM-IEEE Global Telecommun. Conf.*, Honolulu, HI, USA, Nov. 2009, pp. 2316–2322.
[17] Y. Dai, I. Kouichi, and Y. Jun, "A very compact hardware implementation of the KASUMI block cipher," in *Information Security Theory and Practices. Security and Privacy of Pervasive Systems and Smart Devices*. Berlin, Germany: Springer-Verlag, 2010, pp. 293–307.
[18] L. Steinfeld, M. Ritt, F. Silveira, and L. Carro, "Low-power processors require effective memory partitioning," in *Embedded Systems: Design, Analysis and Verification*. Berlin, Germany: Springer, 2013, pp. 52–80.
[19] G. Tim, L. Gregor, and M. Amir, *Lightweight Cryptography for Security and Privacy*. Berlin, Germany: Springer, 2015, pp. 56–129.
[20] I. Algredo-Badillo, F. R. Castillo-Soria, K. A. Ramírez-Gutiérrez, L. Morales-Rosales, A. Medina-Santiago, and C. Feregrino-Uribe, "Lightweight security hardware architecture using DWT and AES algorithms," *IEICE Trans. Inf. Syst.*, vol. 101, no. 11, pp. 2754–2761, 2018.
[21] J. Ax, G. Sievers, J. Daberkow, M. Flasskamp, M. Vohrmann, T. Jungeblut, W. Kelly, M. Porrmann, and U. Ruckert, "CoreVA-MPSoC: A many-core architecture with tightly coupled shared and local data memories," *IEEE Trans. Parallel Distrib. Syst.*, vol. 29, no. 5, pp. 1030–1043, May 2018.
[22] J. L. Hennessy and D. A. Patterson, *Computer Architecture: The Quantitative Approach*. Beijing, China: China Posts Telecom Press, 2013, pp. 53–78.
[23] Q. Zhao, Z. Gu, and H. Zeng, "Design optimization for AUTOSAR models with preemption thresholds and mixed-criticality scheduling," *J. Syst. Archit.*, vol. 72, pp. 61–68, Jan. 2017.
[24] J. Hu, C. J. Xue, W.-C. Tseng, Q. Zhuge, Y. Zhao, and E. H.-M. Sha, "Memory access schedule minimization for embedded systems," *J. Syst. Archit.*, vol. 58, no. 1, pp. 48–59, Jan. 2012.
[25] A. Sahi, D. Lai, and Y. Li, "An efficient hash based parallel block cipher mode of operation," in *Proc. 3rd Int. Conf. Comput. Commun. Syst. (ICCCS)*, Nagoya, Japan, Apr. 2018, pp. 33–40.
[26] B. K. Khailany, T. Williams, J. Lin, E. P. Long, M. Rygh, D. W. Tovey, and W. J. Dally, "A programmable 512 GOPS stream processor for signal, image, and video processing," *IEEE J. Solid-State Circuits*, vol. 43, no. 1, pp. 202–213, Jan. 2008.
[27] S.-K. Chen, C.-Y. Hung, C.-C. Chen, and C.-W. Liu, "Parallelizing complex streaming applications on distributed scratchpad memory multicore architecture," *Int. J. Parallel Program.*, vol. 42, no. 6, pp. 875–899, Dec. 2014.
[28] Y. Wang, Z. Shao, H. C. B. Chan, D. Liu, and Y. Guan, "Memory-aware task scheduling with communication overhead minimization for streaming applications on bus-based multiprocessor System-on-Chips," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 7, pp. 1797–1807, Jul. 2014.
[29] A. N. Bikos and N. Sklavos, "Architecture design of an area efficient high speed crypto processor for 4G LTE," *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 5, pp. 729–741, Sep. 2018.
[30] B. Wang and L. Liu, "Dynamically reconfigurable architecture for symmetric ciphers," *Sci. China Inf. Sci.*, vol. 59, no. 4, pp. 1–16, Apr. 2016.
[31] B. Wang and L. Liu, "A flexible and energy-efficient reconfigurable architecture for symmetric cipher processing," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Piscataway, NJ, USA, May 2015, pp. 1182–1185.
[32] B. Liu and B. M. Baas, "Parallel AES encryption engines for many-core processor arrays," *IEEE Trans. Comput.*, vol. 62, no. 3, pp. 536–547, Mar. 2013.
[33] L. Ren, C. W. Fletcher, A. Kwon, M. van Dijk, and S. Devadas, "Design and implementation of the ascend secure processor," *IEEE Trans. Dependable Secure Comput.*, vol. 16, no. 2, pp. 204–216, Mar. 2019.
[34] X. Li and R. Wang, "Reconfigurable cluster block cipher processing architecture," *Comput. Appl. Softw. Chin.*, vol. 31, no. 1, pp. 52–60, 2014.
[35] L. Wei, Z. Xiaoyang, N. Longmei, C. Tao, and D. Zibin, "A reconfigurable block cryptographic processor based on VLIW architecture," *China Commun.*, vol. 13, no. 1, pp. 91–99, Jan. 2016.
[36] P. Wang and J. McAllister, "Streaming elements for FPGA signal and image processing accelerators," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 24, no. 6, pp. 2262–2274, Jun. 2016.
[37] P. S. Vaidya, J. J. Lee, V. S. Pai, M. Lee, and S. Hur, "Symbiote coprocessor unit—A streaming coprocessor for data stream acceleration," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 24, no. 3, pp. 813–826, Mar. 2016.
[38] J. Kong, P. Liu, and Y. Zhang, "Atomic streaming: A framework of on-chip data supply system for task-parallel MPSoCs," *IEEE Comput. Archit. Lett.*, vol. 11, no. 1, pp. 202–213, Jan./Jun. 2012.

YUFEI ZHU (Graduate Student Member, IEEE) received the B.S. degree in electronics science and technology from Southeast University, Nanjing, China, in 2013, and the M.S. degree in electronics science and technology from the Institute of Information Science and Technology, China, in 2017. He is currently pursuing the Ph.D. degree with the High-Performance Microprocessor Research Group, National University of Defense Technology, Changsha, China.

He is currently a member of the National Natural Science Foundation projects. His current research interests include microprocessor architecture design, hardware security, reconfigurable computing, and VLSI design.

ZUOCHENG XING (Member, IEEE) received the B.S. degree from the Guilin University of Electronic and Technology, Guilin, China, in 1987, and the M.S. and Ph.D. degrees from the National University of Defense Technology, Changsha, China, in 1990 and 2001, respectively. He was a Professor with the School of Computer, National University of Defense Technology, one of the Academic Leader of high-performance computer architecture and microelectronics and solid electronics, and a Doctoral Tutor. He has been engaged in teaching and research on computer science for over 20 years, and responsible or take part in over 20 important projects, including Galaxy and TH-1/1A/2 series high-performance supercomputers design and FT series high-performance general-purpose CPU design, the National Natural Science Foundation, and the National Defense Pre-Research Funds. His current research interests include microprocessor architecture design, information security, 5G wireless communications, and VLSI architecture design for communication. He is a Senior Member of the China Computer Society, a member of the ACM, and the Executive Director of the Hunan Computer Users Association and the Editorial Board of *Journal of Computer Science and Technology* and *International Journal of Advanced Research in Computer Science*.



JINHUI XUE received the B.S. degree in computer science and technology and the M.S. and Ph.D. degrees from the National University of Defense Technology, Changsha, China.

His research areas include reconfigurable computing architecture and compiling. He has been a researcher who devotes to explore reconfigurable computing, chip architecture, and information security. His current research interests include reconfigurable computing and information security, microprocessor architecture design, and SoC system design.

ZERUN LI received the bachelor's degree from Beijing Jiaotong University, in 2016, and the M.S. degree in electronic science and technology from the National University of Defense Technology, in 2018, where he is currently pursuing the Ph.D. degree in electrical science and technology. His research interests include microprocessor architecture, hardware security, and artificial intelligence.

YIFAN HU received the B.S. degree in communication engineering from the Ocean University of China, in 2014, and the M.S. degree in communication engineering in 2017. He is currently pursuing the Ph.D. degree in information security with the National University of Defense Technology (NUDT). His research interests include hardware security and software-defined networking (SDN) in the areas of cyber-physical systems.

YANG ZHANG received the B.S. degree in electrical engineering from Xi'an Jiaotong University, Xi'an, China, in 2011, and the M.S. and Ph.D. degrees in electrical engineering from the National University of Defense Technology, in 2013 and 2018, respectively. He is currently an Assistant Professor with the Institute of Microelectronics and Microprocessors, National University of Defense Technology. His research interests include next generation architecture design, information security, and optimization of parallel programs on processors.

YONGZHONG LI received the B.S. degree in radio technology from the Huazhong University of Science and Technology, China, and the M.S. degree in software engineering from the National University of Defense Technology, Changsha, China. He has been engaged in teaching and research on computer science and technology for over 20 years. His research areas include information security and communication and network security.