Hardware Acceleration of Dash Mining Using Dynamic Partial Reconfiguration on the ZYNQ Board

Mohamed H. Abdulmonem[3], Jihad EssamEddeen[2], Michael H. Zakhari[1], Sayed Hanafi[3] and Hassan Mostafa[1,3,4]

[1]Electronics and Communication Department, Faculty of Engineering, Cairo University, Egypt.

[2]Electronics and Communication Department, Faculty of Engineering, Alexandria University, Egypt

[3]Nanotechnology and Nanoelectronics Department, University of Science and technology, Zewail City, Giza, Egypt.

[4]Electrical & Computer Engineering Department, University of Toronto, Ontario, Canada.

{*s-mohamead.hosni@zewailcity.edu.eg, gehad.essamedeen@gmail.com,*
michael.hany.mofeed@onelab-eg.com, s-sayedhanafy@zewailcity.edu.eg, hmostafa@uwaterloo.ca}

***Abstract*— Dynamic Partial Reconfiguration (DPR) enables reconfiguration of FPGA parts at runtime to provide flexible hardware accelerators with advantages in area, power, reconfiguration time, and memory utilization. In this paper, a design employing DPR technology is proposed to accelerate the mining of the cryptocurrency DASH using the PCAP controller on the ZYNQ 702 Evaluation Board. The DPR design remarkably reduces the area needed for implementing the hash of the mining process of dash on the ZYNQ 702 board.**

Keywords*— Dynamic Partial Reconfiguration, PCAP, software-controlled (S-C) DPR, Dash mining, Cryptocurrencies, FPGA mining, Blockchain technology

I. INTRODUCTION

Blockchain technology provides a distributed ledger of transactions to allow a democratic economy to control financial systems. Instead of using third party verification, each transaction is verified by the majority consensus of the network. To build such a system, the Blockchain Technology systems consist of five main components.

A. Blockchain components

1) Cryptographic hash functions: Cryptocurrencies are based on cryptographic proof instead of trust of a financial authority [1]. A cryptographic hash function calculates a unique output for an input of any size to allow different individuals to have the same output of a given input [2]. Since the hash functions are non-reversible one-way functions, the method of obtaining the input of a certain output is trial and error, this method requires certain computational power to make the consensus of the network output. This computational power depends on the mining difficulty set by the community.

2) Transactions: In cryptocurrencies, a coin is defined by the users who owned it. Owners transfers their coins by signing a hash of the previous transaction that they received the coin from with the public key of the next owner that he is transferring the coin to. Adding these to the end of the coin makes the new coin owner able to easily verify the chain of ownership of the coin [1].

3) Asymmetric-key cryptography: In Asymmetric-key cryptography, two keys are used: public keys and private keys. For encryption, the public key is used as it is known for the public. For decryption, the private key is used, and it is only known by the user [3]. In Cryptocurrencies such as Bitcoin, using Asymmetric-key cryptography, users use private keys to sign transactions, while public keys are used to verify the signatures of the transactions and to derive the addresses of the users that are used in signing the transactions.

4) Ledgers: Ledgers are databases in a distributed fashion that all the users have access to them. Ledgers consist of timestamped blocks of transactions published and verified by the miners of the network.

5) Mining: Mining is the process of assembling a candidate block of transactions after getting an up-to-date copy of the blockchain. A miner builds his blocks by grouping valid transactions into a new block to extend the latest block of the blockchain. Then the miner validates his blocks by finding a nonce. This nonce is supposed to give a hash with a particular initial number of zeros after being added to the transactions Merkle tree's hash and the previous blocks' hash. The number of zeros is given by the mining difficulty. The step of finding the nonce requires the most exhaustive work as the nonce is found by trial and error. So, the higher the hash rate, the higher the probability of finding the nonce, then broadcasting the block and then earning an incentive. In Bitcoin, the mining hash algorithm is SHA256. Every cryptocurrency uses its own hashing algorithm. Mining can be done by CPUs, GPUs, FPGAs, and ASICs. ASICs are the most efficient mining hardware. Due to the high performance of calculating resulting from dedicated hardware, ASICs contribute to the Bitcoin network with more than 99% of the hash power with the remaining 1% shared between CPUs GPUs and FPGAs.

B. Dash cryptocurrency

As the inefficiency of Bitcoin emerges due to this ASICs dominance, high transaction fees, privacy, and scalability of Bitcoin [4], Bitcoin is now competing with many currencies. One of the solid contenders is Dash. Dash is a self-governing open source cryptocurrency that was launched in 2014. While both Dash and Bitcoin are mind by proof of work, dash is mined by a new hashing algorithm, X11. Dash was launched to compete with the other cryptocurrencies by solving the main issues facing Bitcoin. These issues are mainly the slow acceptance of transactions, the anonymity of transactions in the network, and ASICs dominance.

1) Slow acceptance & anonymity of transactions: To overcome these issues, Dash introduced the concept of Masternodes. A Masternode is a server that has a complete copy of the blockchain, performs specific tasks and takes an

incentive in return (45% of the block rewards). Masternodes do block validation related tasks and two other significant tasks. Firstly, known as InstantSend, masternodes are called to vote on whether a transaction is valid or not. If it is valid, the Masternodes instantly broadcast the transaction to the network without waiting for all the users to reach consensus [5]. Secondly, known as PrivateSend, breaks every transaction input into multiple standard denominations and mixes it with other transactions inputs to assure the anonymity of the transactions [5].

2) ASICs dominance: In Bitcoin, the entire hash rate is dominated by ASICs. Accordingly, small miners with CPUs, GPUs and FPGAs cannot compete in broadcasting blocks. To overcome this issue and to have a fair competition between hardware in the early stages of mining, Dash introduced a new hashing algorithm (X11) to cryptocurrencies. X11, known as algorithm chaining, consists of 11 hashing algorithms from SHA3, which are BLAKE, BLUE MIDNIGHT WISH (BMW), Grøstl, JH, Keccak, Skein, Luffa, CubeHash, SHAvite-3, SIMD, and ECHO. Each hash is calculated then submitted to the next. So, it is minimal that an ASIC is created for the currency in the early stages of mining [7]. Now, X11 has been implemented on CPUs, GPUs and ASICs; however, due to the large size of implementation are needed for it, Dash is not being mined by FPGAs.

II. Literature Review

The complete mining process of Dash was not implemented on FPGAs before. Dries Truyens [8] profiled the 11 algorithms of X11 and chose the slowest 3 consecutive algorithms to be accelerated on FPGA, which are Skein, JH and Keccak. Dries Truyens has accelerated only three hashing algorithms out of the 11 hashing algorithms in the X11 using Zedboard. Consequently, the CPU-miner achieved a higher hash rate with the acceleration of the three slowest algorithms. Moreover, the main issue for not implementing the 11 algorithms was in the area needed for the 11 hashes to be implemented together.

III. The Main Idea of The Proposed Method

DPR improves the FPGA design by modifying the logic on the FPGA dynamically. To do so, the design should be divided into static part and dynamic part. The dynamic part will have partial bit streams of different logic circuits [9]. This dynamic part can consist of multiple reconfigurable modules that can swapped at runtime. The basic premise of partial reconfiguration is shown in Fig 1 by Xilinx. Since the X11 RTL design has the following specifications: Firstly, the major issue of implementing the X11 hashing algorithm on FPGAs is in the area needed for the implementation. Secondly, all the X11 hashes are needed to be calculated in a series architecture not in parallel. Finally, reducing the area/power is a major advantage for the hashing hardware application. A software-controlled (S-C) dynamic partial reconfigurable design is proposed to accelerate the process of calculating the X11 hash and reduce the power needed for the hash in this paper. The ZYNQ 702 evaluation board is chosen as it has enough resources to implement every module of the 11 hashes at once. The partial reconfiguration controller is the PCAP controller as it has a 32-bit wide data interface that operates at a clock frequency of 100 MHZ and it also achieves a theoretical reconfiguration throughput of 400 MB/s [10, 11]. The

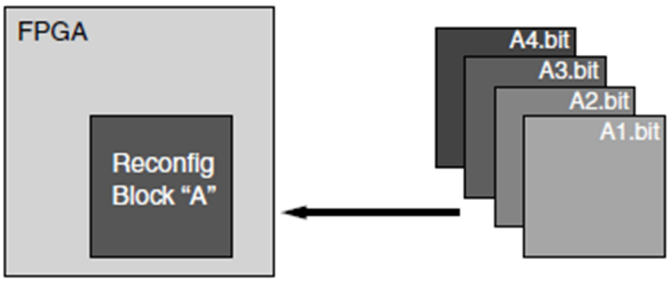

Fig 1. Partial reconfiguration design concept.

disadvantage of the PCAP is preventing the ARM processor from doing software tasks during reconfiguration. However, this drawback does not affect this implementation as the ARM processor is not required to do any software task during the reconfiguration. Consequently, the design proposed has every hash function as a reconfigurable module and every hash output will be submitted to the ARM processor then given as an input to the next hash function after reconfiguration.

IV. Methodology

Using Vivado IPI and Software Development Kit, the steps executed are:

1. Developing the Keccak hashing algorithm core RTL based on the implementation of [12], Blake and Skein core RTLs based on [13].
2. Modifying the remaining 8 hash functions RTLs taken from Mike Xia's cryptography library [14] to have the same interface of Clock, Reset, 512-bit input, and 512-bit output as the previous developed hashes.
3. Developing behavioral testbenches for the hashing modules then testing them separately to assure the right results of every hash function.
4. Creation and synthesis of the block design for the static and reconfigurable modules with instantiating the ZYNQ PS with SD 0 and UART 1 interfaces. Also, enabling the GP0 interface with FCLK0 and RESET0_N ports. Fig 2.
5. Determining the largest module in area by the resource utilization and to quantify the benefits of the dynamic reconfigurable design over the static design.
6. Defining the floorplan for the configurable region SLICE_X26Y0:SLICE_X112Y149 as shown in Fig 3. Then testing if it contain the largest reconfigurable module SIMD with less than 20% of the utilization unused.
7. Loading the largest reconfigurable module SIMD then Placing and routing it as in Fig 4 then generating the static route design as shown in Fig 5.
8. Placing and routing all the other reconfigurable modules. Then writing checkpoints for every implementation.
9. Generating the partial & full bitstreams for every configuration of the 11 hashes.
10. Creating the application in C that takes the output from every hash and enter it as input to the next one through the AXI interface between the PS and the FPGA.
11. Testing the complete system of the DPR hashing functions.

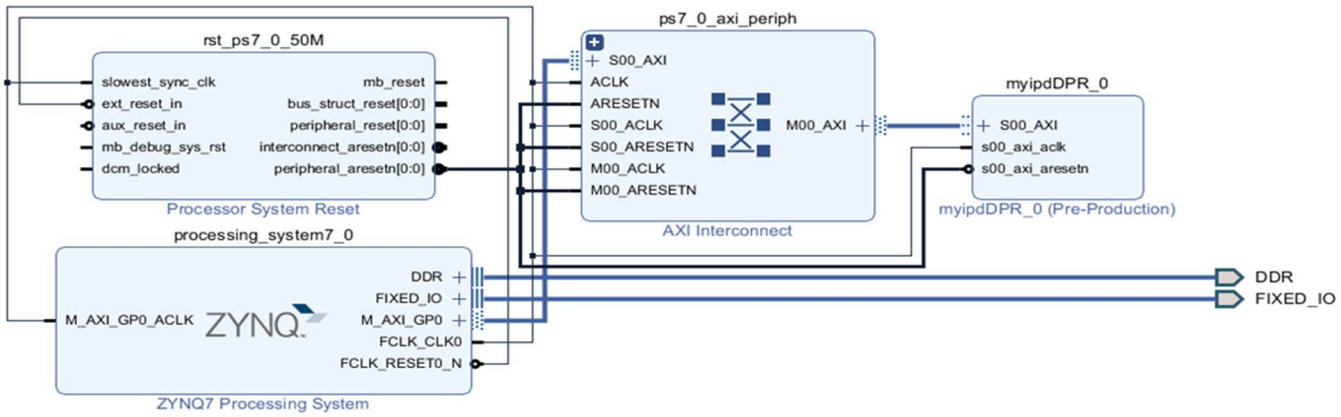

Fig 2. The block diagram of the Zynq processor with the required AXI peripherals, UART 1 and SD 0 interfaces and the reconfigurable module (UltraP_0) that will be configured as 11 different hash functions.

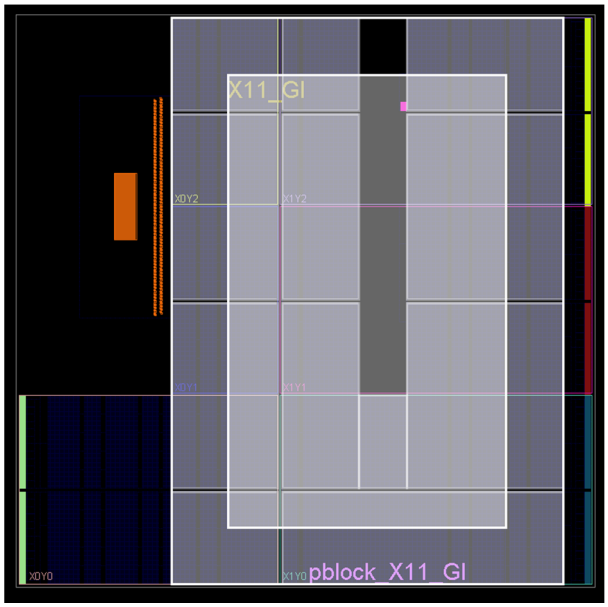

Fig 3. The floorplan of the reconfigurable pBlock of SLICE_X26Y0: SLICE_X112Y149.

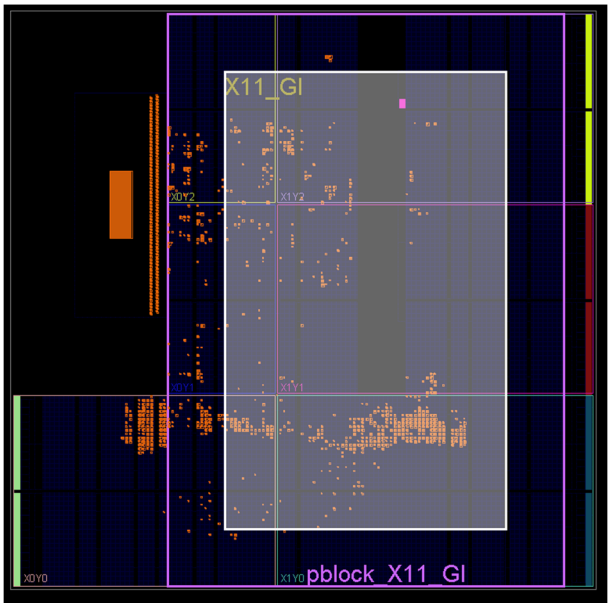

Fig 5. The static route design.

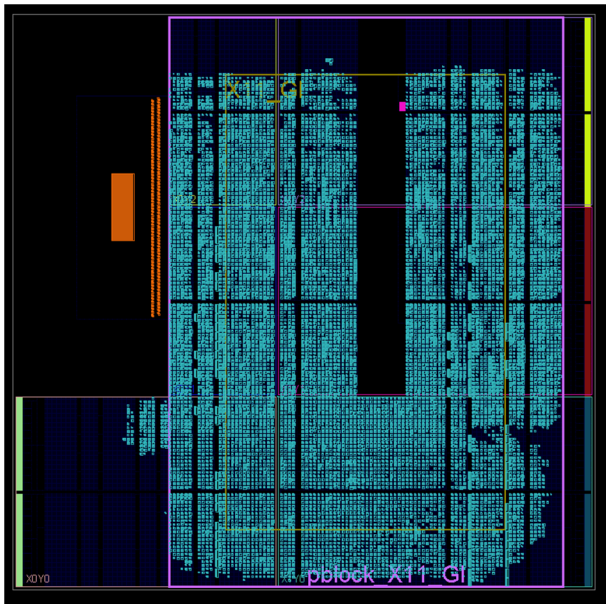

Fig 4. The placed & routed design for SIMD.

V. RESULTS

Table I shows the resource utilization of the 11 different modules where SIMD was the largest one to utilize the partially reconfigurable area. The required Look Up Tables (LUTs) for the reconfigurable modules to be implemented on the FPGA simultaneously were over 17,000 LUT which utilizes over 320% of the available LUT in the ZYNQ 702 FPGA. Using DPR, only 67% of ZYNQ 702 available LUT were utilized for implementation as shown in Table II. Moreover, after generating the partial bitstream files, the size of each file was 2,831,895 bytes. Hence, the corresponding theoretical reconfiguration time by the ZYNQ processor is 7.08 ms for each configuration which delays the process of calculating the hash; however, it makes it achievable to mine the Dash cryptocurrency using a small size FPGA such as the ZYNQ 702. Also, it lowers the static power needed for the process. Moreover, the total clock cycles taken by the 11 modules without the reconfiguration time are 9,713 clock cycles on a Frequency of 800 MHz.

TABLE I. THE RESOURCE UTILIZATION OF THE 11 RECONFIGURABLE MODULES.

Resources	Blake	BMW	Groestl	JH	Keccak	Skein
LUT	8,431 (15.9%)	30,645 (57.6%)	9,289 (17.5%)	4,965 (9.33%)	21,938 (41.2%)	5,205 (9.78%)
FF	2,199 (2.07%)	3,787 (3.56%)	5,144 (4.83%)	2,615 (2.45%)	10,000 (9.40%)	3210 (3.02%)
Block RAM Tile	4 (2.85%)	4 (2.85%)	4 (2.85%)	4 (2.85%)	4 (2.85%)	4 (2.85%)
DSP	72 (32.7%)	72 (32.7%)	72 (32.7%)	72 (32.7%)	72 (32.7%)	72 (32.7%)
	Luffa	**CubeHash**	**SHAvite-3**	**SIMD**	**ECHO**	
LUT	14,988 (28.2%)	6,116 (11.5%)	21,028 (39.5%)	35,393 (66.5%)	9,889 (18.6%)	
FF	5,734 (5.39%)	2,073 (1.95%)	1,871 (1.76%)	14,246 (13.4%)	3,142 (2.95%)	
Block RAM Tile	4 (2.85%)	4 (2.85%)	4 (2.85%)	4 (2.85%)	4 (2.85%)	
DSP	72 (32.7%)	72 (32.7%)	72 (32.7%)	72 (32.7%)	72 (32.7%)	

TABLE II. BASIC AND DPR DESIGNS COMPARISON AS PERCENTAGES OF THE ZYNQ 702 RESOURCES

Design	LUT	FF	DSP
Basic	>320%	>52%	359.7%
DPR	67%	33%	32.7%

VI. CONCLUSION

Implementation of X11 hashing algorithm on the ZYNQ 702 is not possible without the DPR based design suggested in this paper. Using DPR, the ZYNQ 702 FPGA was able to accelerate the process of calculating the hash of the X11 algorithm that is being used in Dash mining. As the reconfiguration time is high for the application of mining a cryptocurrency which requires high hash rates, the 11 reconfigurable modules for the hashes can be optimized to decrease the area needed for the reconfiguration. Also, it will decrease the reconfiguration time and the total power needed. Moreover, it will decrease the total clock cycles required for the process. A further modification to the design is optimizing the hashing algorithms to produce a FPGA based Dash miner with a hash rate per power, that can compete in hashing with other hardware in the network of Dash such as GPUs and ASICs.

ACKNOWLEDGMENT

This work was partially funded by ONE Lab at Zewail City of Science and Technology and at Cairo University, NTRA, ITIDA, and ASRT.

REFERENCES

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," November 2008.

[2] D. Yaga, P. Mell, N. Roby, and K. Scarfone, "Blockchain technology overview," *NISTIR 8202*, Jan. 2018.

[3] A. Gupta and N. Kaur Walia, "Cryptography Algorithms: A Review", *International Journal of Engineering Development and Research*, 2014, Volume 2, Issue 2, pp. 1667-1672.

[4] J. Herrera - Joancomartí, C. Pérez-Solà, "Privacy in Bitcoin Transactions: New Challenges from Blockchain Scalability Solutions," *Modeling Decisions for Artificial Intelligence*, Lecture Notes in Computer Science, 2016, vol. 9880, pp. 26-44.

[5] Dash Core Group, "Features," 2019.

[6] Dash Core Group, "Understanding Masternodes," 2018.

[7] E. Duffield, and D. Diaz, "Dash: A Privacy-Centric Crypto-Currency," April 2014.

[8] D. Truyens. "FPGA based hardware accelerator for Dash Mining". M. S. thesis. Faculty of Engineering Science. KU LEUVEN. 2016.

[9] E.Youssef, H. A. Elsemary, M. A. El-Moursy, A. Khattab, and H. Mostafa. "Energy Adaptive Convolution Neural Network Using Dynamic Partial Reconfiguration." *2020 IEEE 63rd International Midwest Symposium on Circuits and Systems (MWSCAS)*, pp. 325-328, 2020.

[10] A. K. ELdin, A. Mohamed, A. Nagy, Y. Gamal, A. Shalash, Y. Ismail, and H. Mostafa, "Design Guidelines for the High-Speed Dynamic Partial Reconfiguration Based Software Defined Radio Implementations on Xilinx Zynq FPGA," *International Symposium on Circuits and Systems (ISCAS 2017)*, Baltimore, USA, IEEE, May 2017.

[11] K. Khatib, M. Ahmed, A. K. Eldin, M. Abdelghany and H. Mostafa, "Dynamically reconfigurable power efficient security for internet of things devices," *2018 7th International Conference on Modern Circuits and Systems Technologies (MOCAST)*, pp. 1-4, 2018.

[12] G. Provelengios, P. Kitsos, N. Sklavos and C. Koulamas, "FPGA-based Design Approaches of Keccak Hash Function," *2012 15th Euromicro Conference on Digital System Design*, Izmir, 2012, pp. 648-653.

[13] N. At, J. Beuchat, E. Okamoto, İ. San and T. Yamazaki, "Compact Hardware Implementations of ChaCha, BLAKE, Threefish, and Skein on FPGA," in *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 2, pp. 485-498, Feb. 2014.

[14] M. Xia, "Cryptography library," GitHub repository, 2018.