ZAMMAZAZZER-A CRYPTOCURRENCY WHOSE PROOF-OF-WORK PROBLEM WILL BUILD THE REVERSIBLE COMPUTER

JOSEPH VAN NAME

Abstract. The proposed cryptocurrency Zammazazzer is designed to incentivize the development of the reversible computer. Reversible computers will eventually outperform and replace conventional computers everywhere. By Landauer's principle, reversible computers will eventually become many times more energy efficient than the possible the maximum possible efficiency of an irreversible computer. Since reversible computers will do calculations without producing the immense amount of heat required by conventional computers, reversible computers can potentially run many times faster than conventional computers with much denser parallelism. However, while reversible computers will eventually replace conventional computers, these energy efficient reversible computers do not exist in the free market, and hardware manufacturers are unwilling to make reversible computers in part due to the computational overhead incurred from using reversible algorithms instead of conventional algorithms. The proof-of-work problem for Zammazazzer is designed to incentivize the initial development of the energy efficient reversible computer, and the Zammazazzer Foundation, supported through cryptocurrency coins, shall fund the research, development, and standardization of cryptographic algorithms suitable for reversible computers through the use of smart contracts.

1. Introduction

The most prominent issue with cryptocurrencies today is that the proof-of-work (abbreviated POW) problems require a large investment in computing machinery and energy but these problems do not provide anything of value besides securing the blockchain. Some have attempted to resolve this issue by replacing the computationally intensive POW problem with proof-of-stake (abbreviated POS) which does not require intensive computation. However, POS cannot be used as a fair method of distributing newly minted coins, so one cannot expect POS to completely overtake POW for cryptocurrencies. Others have attempted to select a POW problem where the solution or the process of obtaining the solution advances science or mathematics in some way (for example, the POW problems for Primecoin and Gapcoin amount to finding peculiar prime numbers). The value however of the solution of these POW problems for Primecoin and Gapcoin is debated since no new theorems, conjectures, or platforms for cryptosystems have been obtained through mining these cryptocurrencies. Since POW problems for cryptocurrencies must satisfy some stringent requirements and since there is a movement away from POW as a consensus mechanism for cryptocurrencies, so far there are no cryptocurrencies with a POW problem which has undisputed real-world applications. On the other hand, a useful POW problem is the only way for a cryptocurrency

to have a secure consensus mechanism that gives people a justified perception that the cryptocurrency has intrinsic value.

Cryptocurrencies need to implement useful POW problems to secure a strong public image. If cryptocurrencies do not have a strong public image, then various organizations will be more likely to attack the security of the cryptocurrencies, and governments will pass laws restricting cryptocurrencies; furthermore, as the market for cryptocurrencies grows, environmental advocates will more intensely condemn and lobby against cryptocurrencies for having wasteful POW problems unless their POW problems have some other practical use besides achieving decentralization. Therefore, the best way for a cryptocurrency to maintain a strong public image and be in a good standing with the world's governments will be to select a suitable POW problem that advances science, technology, or mathematics and which gives cryptocurrencies a reputation for being innovative.

Landauer's principle gives a theoretical limit to the energy efficiency of conventional computers. However, Landauer's principle only applies to irreversible operations, and reversible operations are not restricted by Landauer's principle. Reversible computers, which only implement reversible operations, can therefore perform many times more operations per unit of energy than a conventional computer can ever perform. Since reversible computers are not restricted by Landauer's principle, reversible computers can potentially perform a larger quantity of operations than conventional computers since heat production will not limit the performance of reversible computers as much as it does for conventional computers. Unfortunately, all of today's computers are irreversible, and there are currently no reversible computers in the free market which are more efficient than a conventional computer.

Any function which can be computed by a conventional computer could also be computed by a reversible computer. However, today there is little demand for and little motivation to develop reversible computers since reversible computers typically need to take more steps to compute a function than is needed for an conventional computer to compute the same function. The increased computational complexity of reversible computation stems from the fact that purely reversible computers are not allowed to delete any information since deleting information is an irreversible process. On the other hand, in the future, since reversible computation is not subject to Landauer's limit, computational machinery will consist mainly of reversible components since the efficiency gained by using reversible computation will eventually compensate for the need for the additional complexity which is required in reversible computation. It is therefore necessary for people to research and develop reversible computers now so that we can continue to advance our computing technology beyond what is possible with conventional computers.

Today, it is clear that the exponential improvements of integrated circuits are slowing down; this slow down can only be overcome by using reversible computation instead of conventional computation. Today, people often state "Moore's law is dead" since conventional integrated circuits have much less room for improvement than in the past. In fact, the exponential growth in integrated circuit technologies has been slowing down since 2005. Before 2005, the clock speeds on CPUs have increased at an exponential rate, but since around 2005, the clock speeds on CPUs have ceased to increase beyond a couple of GHZ. Since 2005, chip manufacturers, instead of increasing the clock speeds of their integrated circuit, have simply

produced multi-core CPUs in order to increase performance. However, multi-core CPUs only have produced a limited increase in the performance of integrated circuits because today, most portions of multi-core CPUs must remain off at any given time and because Amdahl's law limits the performance gains that one obtains from parallel computation.

When one analyzes the energy usage of a modern integrated circuit, the reason that CPUs have not improved very much in recent years is clearly evident. Today, a completely flat integrated circuit can produce about $100W/\text{cm}^2$ power per unit area. This amount of power per surface area is equal to the blackbody radiation of an object with temperature over 2000 K (at 1000 K, all objects glow red). Today, the power usage of integrated circuits is only limited by limited ability of cooling devices to remove heat from the integrated circuits; if overheating and energy usage were not a problem for integrated circuits, today's integrated circuits would produce much more power per unit area than $100W/\text{cm}^2$. The energy usage and heat production of these integrated circuits have limited their performance and this energy usage is the main obstacle that people need to overcome in order to continue to produce better integrated circuits, and the only way to continue to decrease the energy usage of integrated circuits in the long term is to use reversible computation. Without reversible computation, we have at most 10 years improvements at the transistor level of computation. Since conventional computation will cease to improve in at most 10 years, we expect for reversible computation which is competitive to conventional computation to start to appear in the free market within 10 years. On the other hand, it will probably take at least a couple of decades for reversible computation to completely replace conventional computation because of the computational overhead incurred from reversible computation.

A reversible computation optimized POW problem (abbreviated RCO-POW problem) is a cryptocurrency proof-of-work problem which is designed to be efficiently solved by a reversible computer and to incentivize the development of the reversible computer. Since reversible computation is more complex and requires more operations than conventional computation, there will be little commercial interest in reversible computation until reversible computational processes are several times as efficient as conventional computational processes. After all, there is no reason to perform three times as many operations to run an algorithm on a reversible computer if the reversible computer only consumes half as much energy per reversible operation. On the other hand, an reversible computer will take just as many steps in each attempt to solve an RCO-POW problem as a conventional computer, and the reversible computer produces a minimal amount of garbage information that must be erased using conventional computing methods after every solution attempt. Therefore since RCO-POW problems are computationally intensive but are designed to be solved using a reversible computer as easily as they are solved using a conventional computer, computational machinery manufacturers will have a greater incentive to create for the first time reversible computers which are competitive with conventional computers on the free market in order to mine these crytpocurrencies. Cryptocurrencies with RCO-POW problems will therefore accelerate the initial development of the free market reversible computer.

The usefulness of an RCO-POW problem in a cryptocurrency depends on the time in history when the miners are mining such a coin. If people are mining a

cryptocurrency with an RCO-POW problem too early in the history of computation, then such an RCO-POW problem would not be useful since the efficiency of integrated circuits will be nowhere near Landauer's limit and because it would be much more profitable for manufacturers to improve their conventional computers without resorting to reversibility since Landauer's limit is far away and because most reversible computation requires space/time overheads. On the other hand, if such a cryptocurrency is launched too late in history when energy efficient reversible computers are already in the mainstream, then there would be little point in making a cryptocurrency to incentivize the development of these reversible computers which are already in the mainstream. The good news is that this year 2018 is an optimal year for developing the reversible computer since it is neither too early nor too late in history to incentivize the development of these reversible computers. This however means that the usefulness of RCO-POW problems has an expiration date.

The cryptocurrency Zammazazzer shall employ a RCO-POW problem in order to incentivize the development of the reversible computer. The RCO-POW problems shall be the very first free market application of energy efficient reversible computing. After RCO-POW problems, reversible symmetric cryptosystems such as block ciphers, hash functions, pseudorandom number generators, and stream ciphers will be the next free market application of reversible computers. The Zammazazzer Foundation, which shall be funded through Zammazazzer, will research, develop, and standardize reversible symmetric cryptosystems and other algorithms in order to accelerate a market for reversible computation outside of just cryptocurrency mining. Only after specialized reversible computers are developed for symmetric cryptography and for RCO-POW mining, people will start to use reversible computers for other purposes.

2. Useful POW problems

Cryptocurrency POW problems must satisfy several strict requirements. These requirements are quite obvious to anyone who is familiar with the original Bitcoin whitepaper [7]. These requirements are typically difficult to fulfill simultaneously, and therefore most computational problems do not satisfy the requirements for cryptocurrency POW problems. Since POW problems must satisfy such stringent conditions, today most POW problems in use today for cryptocurrencies are variants of cryptographic hash functions and key derivation functions (which are memory hard for ASIC resistance). The RCO-POW for Zammazazzer satisfies all of these requirements just as well as a cryptographic hash function satisfies these requirements. The following characteristics are needed for any cryptocurrency POW problem.

Easy to verify but difficult to solve-The POW problem needs to be difficult to solve, but it must be easy to verify the correctness of the solution to the POW problem.

Solutions tied to solver-The solutions of the POW problem need to be valid only for the solver of the problem for the current block in the blockchain. This condition is necessary so that if a miner Alice finds a block in the blockchain, then another miner Bob cannot claim Alice's solution as his own. We also need these conditions so that Alice cannot use a pre-computed solution to the current block in the blockchain.

Automatic generation-The POW problems need to be generated automatically by the cryptocurrency using random data from the blockchain. If miners, stakeholders, or a central authority were in charge of generating the POW problems, then those entities would most likely be able to manipulate the consensus algorithm in order to maximize their own personal profits and hence put the decentralization, fair distribution, and usefulness of the POW problem in jeopardy.

Fine tunable difficulty-The difficulty of the POW problem needs to be periodically adjusted to account for improvements in technology, the move from CPU based mining to ASIC based mining, improved algorithms, and an increasing market cap. Therefore, the difficulty of the problem needs to be algorithmically finely tunable so that new blocks are produced at a constant rate.

Pre-computation resistance-It should be very difficult or nearly impossible for an entity to obtain a significant advantage in solving a POW problem simply by performing calculations in advance.

Progress freeness-The amount of time for an entity or group of entities it takes to solve the POW problem at any difficulty level should follow approximately an exponential distribution. This property is needed to ensure that the probability of an entity solving any given block POW is directly proportional to his computing power in order to ensure decentralized consensus and a fair distribution of coins.

Unbreakability-The POW problem needs to be and remain cryptographically secure. There should be little to no risk that an entity will obtain a secret algorithm which will break the security of the POW problem. If an entity has a secret and quick algorithm that quickly solves a POW problem, then such an entity will have a strong chance of launching a 51 percent attack against the cryptocurrency. Even if an entity with a such an algorithm does not launch a 51 percent attack, that entity will likely win a disproportional amount of newly minted coins, and this phenomenon will ruin the fair distribution of the cryptocurrency. If a quick algorithm for solving the POW problem becomes public, then the problem will no longer be much more difficult to solve than to verify, and hence the problem will no longer be viable as a POW problem for cryptocurrencies.

Usefulness−The solutions to the POW problem or the process of obtaining the solutions should have some use outside the cryptocurrency. The POW problem should advance science, mathematics, or technology in some way. The usefulness of an ideal POW problem should always be equal to or greater than the cost of solving the POW problem in order to justify the use of the POW problem.

It is impossible to predict the future market cap of a cryptocurrency, so the usefulness of the POW problem should exceed the cost of solving the POW problem regardless of the market cap of the cryptocurrency and regardless of the amount of time in which the POW problem has been active in the cryptocurrency. It is more important for a POW problem to be useful at a high block reward than at a low block reward. For some potentially useful POW problems, the usefulness of the POW problem may follow a law of diminishing returns with respect to the block reward of the cryptocurrency. For other potentially useful POW problems, the usefulness will decrease over time since the POW will eventually be thoroughly researched or the problem may eventually become obsolete. For example, the POW for a cryptocurrency may be very useful when the block reward for the cryptocurrency is small but when the block reward is large, the usefulness of the POW problem will not be enough to justify the resources spent on solving the problem.

As another example, a POW problem may immediately lose its usefulness after a new scientific discovery or invention or the usefulness of the POW problem may diminish over time as the problem becomes obsolete. Most useful computational problems will not qualify as useful POW problems even if they do qualify as cryptocurrency POW problems since the usefulness of these problems will not justify the cost of solving these problems at all market caps for many years.

A useful POW problem should ideally be friction-free in the sense that the POW problem will not be any more useful if an uncorrupted central authority had the ability to spend the block rewards as it wishes in order to maximize the usefulness of the POW problem. RCO-POW problems are not friction-free since not all of the resources spent on solving these RCO-POW problems will be spent on producing reversible hardware; a portion of these resources will instead be spent on energy and on other costs which do not directly advance reversible computing technologies.

All POW problems incentivize the development of better computing technologies, so one should not consider a POW problem to be useful simply because it incentivizes the development of better computers since such a use will not justify the amount of resources spent on solving the problem; furthermore, such a problem cannot be friction free. We however consider RCO-POW problems to be useful problems since reversible computers will replace conventional computing technologies in the far future but there is little short term incentive to develop reversible computers.

RCO-POW problems are not too useful when the block rewards are low, but RCO-POW problems become very useful as the block rewards for cryptocurrencies with RCO-POW problems becomes very large. The RCO-POW for Zammazazzer will remain useful until reversible computers or partially reversible computers become commonplace.

2.1. **The solution lottery technique.** There are techniques which one could use to turn useful computational problems into useful cryptocurrency POW problems. One of these techniques is to use multiple problems instead of one; unfortunately, it is much more difficult to make a cryptocurrency using many POW problems (this is mainly due to the fact that it is easier to find one useful POW problem than it is to find hundreds) than it is to make a cryptocurrency that only has one POW problem, and the security of a cryptocurrency with multiple POW problems has not been explored. The solution lottery technique (abbreviated SLT) is another technique which one can use to produce useful POW problems from computational challenges; unlike the use of multiple problems, the SLT is easy to implement. We shall now give a description of a POW problem which we call Problem A with and without the SLT.

Suppose that D is a set of ordered pairs of the form (k,x) such that for random k, it is difficult to find a string x such that $(k,x) \in D$. The set D is possibly adjustable in order to account for an increase in difficulty. Suppose that $\mathrm{Data}(k,x)$ is a string which is easy to produce if one has verified that $(k,x) \in D$ but which cannot otherwise be produced. Suppose that C is an adjustable constant.

Problem A without SLT: Find a suitable hash k along with a string x such that $(k,x) \in D$.

Problem A with SLT: Find a suitable hash k along with an input x such that $(k,x) \in D$ and where $H(k||x||\mathrm{Data}(k,x)) < C$.

The idea behind the SLT is that a miner for a POW with the SLT will spend very little resources computing hashes and producing the data $\text{Data}(k, x)$; instead, a miner will spend almost all resources finding values x such that $(k, x) \in D$ for a pre-computed hash k; on the other hand, a POW with the SLT could gain many of the cryptographic benefits that comes from using hash functions as POW problems.

An appropriate use of the SLT can help turn useful computational challenges into useful POW problems by ameliorating some of the strict requirements of POW problems including fine-tunability, unbreakability, and progress freeness among other conditions. The SLT could also take a computational challenge with low or possibly low cryptographic security and turn it into a viable cryptocurrency POW problem.

There are currently no cryptocurrencies on the market which employ the SLT in their POW problems. It is currently an open avenue of research to investigate and develop new consensus algorithms based upon POW problems with the SLT. We shall use the SLT in the POW for Zammazazzer.

3. Reversible computation

A computation is said to be reversible if the computation does not erase or delete any information in any part of the computational process. Said differently, a computation is reversible if for every portion of the computation, the input can be recovered from the output by running the computation in reverse. For example, the AND gate is irreversible since it is impossible to determine the input of the AND gate when the output is FALSE. The OR gate is irreversible for the same reason. However, the NOT gate is reversible since one can recover the output from the input. Since a reversible gate is not permitted to erase any information, the output of a reversible gate must have the same number of bits as the input (the AND gate is irreversible since it has 2 inputs yet only one output). A logic gate is said to be reversible if it is bijective, and a combinatorial circuit is said to be reversible if all of its gates are reversible gates. Reversible sequential logic can be defined in a similar manner.

The gate $T : \{0, 1\}^3 \to \{0, 1\}^3$ defined by

$$T(x, y, z) = (x, y, (x \wedge y) \oplus z)$$

is known as the Toffoli gate. The gate $F : \{0, 1\}^3 \to \{0, 1\}^3$ defined by

$$F(0, y, z) = (0, y, z), F(1, y, z) = (1, z, y)$$

is known as the Fredkin gate. The gate

$$C(x, y) = (x, x \oplus y)$$

is known as the CNOT gate. The Toffoli, Fredkin, and CNOT gates are all reversible. No reversible gate on 2 bits is capable of universal computation, but the Toffoli gate is sufficient for universal computation and the Fredkin gate is also sufficient for universal computation (AND and OR gates can be easily simulated using either Toffoli gates or Fredkin gates, but such a simulation in general produces garbage information).

Landauer's principle states that erasing a bit always costs a minimum of $k \cdot T \cdot \ln(2)$ energy where k is Boltzmann's constant ($k = 1.38064852 \cdot 10^{-23} J/K$) and where T is the temperature. Landauer's principle is a consequence of the second law of thermodynamics since any computation which violates Landauer's principle is capable

of reducing the amount of entropy in a closed system. At around room temperature of $300K$, by Landauer's principle, erasing a bit costs at least $2.8 \cdot 10^{-21} J$. In practice, a *reliable* irreversible gate such as an AND gate must cost at least $100kT$ per bit deleted in order to overcome the thermal fluctuations and produce the desired output with almost no errors. Irreversible computation requires one to constantly erase information, so irreversible computation always costs a minimum amount of energy. On the other hand, reversible computation does not allow any erasure of information, so the energy efficiency of reversible computation is not limited by Landauer's principle. There is theoretically no limit to the energy efficiency of reversible computation. Since reversible computation in-principle is more energy efficient than irreversible computation, reversible computers can potentially run at much higher frequencies and use much denser parallelism than is possible with an irreversible computer.

Since it is necessary for all computers to eventually erase information, all computers must incorporate some form of irreversibility. The goal of reversible computation is therefore not to delete any information whatsoever, but instead to limit the amount of information deleted as much as possible in order to save energy and reduce the accumulation of heat. From now on, when we mention reversible computers, we shall assume that such computers will always delete a small amount of information in order to maximize efficiency, and we shall also assume that the reversible computers use physical reversibility in order to improve performance and energy efficiency.

It is a fairly straightforward and intuitive exercise to emulate any form of conventional computation (such as a cellular automaton, Turing machine, or combinatorial circuit) with a model of reversible computation at the expense that every intermediate state of the conventional computation is saved in memory. However, since memory is a limited resource, this approach alone cannot improve the performance of computational technology since one will eventually need to erase this garbage information produced in the intermediate steps of the computation. Fortunately, much of the garbage information produced by a reversible computer can be removed by a process called uncomputation. Suppose that $f : \{0,1\}^n \to \{0,1\}^n$ is a function that one wishes to compute using a reversible circuit without producing too much garbage information. Then we shall be able to compute the mapping where $(x, \mathbf{0}, \mathbf{0}) \mapsto (x, \mathbf{0}, f(x))$ using a reversible circuit. Observe that there exists some bijective

$$C : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n \times \{0,1\}^m$$

easily computable by a reversible circuit along with some $f_g : \{0,1\}^n \to \{0,1\}^m$ such that $C(x, \mathbf{0}) = (f(x), f_g(x))$ for all $x \in \{0,1\}^n$. While the reversible function C in-essence computes the function f, the function C also produces garbage information $f_g(x)$. Fortunately, the following trick known as uncomputation [2] allows one to remove the garbage information $f_g(x)$.

Define a mapping

$$L : \{0,1\}^n \times \{0,1\}^m \times \{0,1\}^n \to \{0,1\}^n \times \{0,1\}^m \times \{0,1\}^n$$

by letting $L(x, y, z) = (x, y, x \oplus z)$ (L is computable using n CNOT gates). Let $\mathrm{Id} : \{0,1\}^n \to \{0,1\}^n$ denote the identity function. Define

$$H = (C \times \mathrm{Id})^{-1} \circ L \circ (C \times \mathrm{Id}).$$

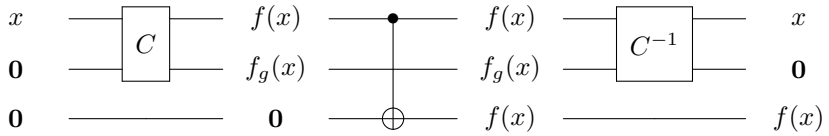
Then H is easily computable by a reversible circuit, but

$$H(x, \mathbf{0}, \mathbf{0}) = (x, \mathbf{0}, f(x))$$

for all $x \in \{0,1\}^n$ (the middle m "scratch work" bits which start off and end as zeroes after computing and uncomputing the function f are known as ancilla bits).

While the function $C \times \mathrm{Id}$ computes the function f and outputs the garbage $f_g(x)$, the inverse $(C \times \mathrm{Id})^{-1}$ uncomputes the function f and removes the garbage $f_g(x)$ leaving just the initial input x along with the copied output $f(x)$. Since the process of uncomputation is reversible, uncomputation removes the garbage $f_g(x)$ in a completely reversible manner.

The following circuit diagram illustrates how uncomputing can be used to reset bits back to $\mathbf{0}$ without incurring the thermodynamic cost from Landauer's principle.



Through the careful use of uncomputation, all calculations could be carried out reversibly with only a modest increase in the amount of space and time needed for such a calculation; an irreversible circuit of width w and depth d can be simulated using a reversible circuit of width $O(w \cdot \log(d))$ and depth $O(d^{1+\epsilon})$ [3],[1] for all $\epsilon > 0$.

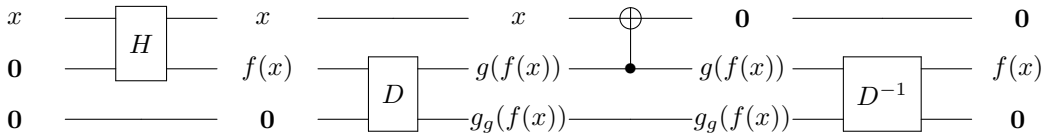
Uncomputation could also be used to compute bijective functions without producing any garbage information and without storing the input the bijective function. Suppose that $f, g : \{0,1\}^n \to \{0,1\}^n$ are inverse functions. If both f, g can be computed using a conventional circuit that uses r gates, then f, g can be computed on a reversible circuit that uses r gates and produces no garbage information using the following technique.

Let m be a natural number. Let $H : \{0,1\}^n \times \{0,1\}^n \times \{0,1\}^m$ be a a reversible circuit such that $H(\mathbf{x}, \mathbf{0}^n, \mathbf{0}^m) = (\mathbf{x}, f(\mathbf{x}), \mathbf{0}^m)$ for all $\mathbf{x} \in \{0,1\}^n$. Let $M : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n \times \{0,1\}^n$ be the function defined by $M(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \oplus \mathbf{y}, \mathbf{y})$. Let $g_g : \{0,1\}^m \to \{0,1\}^m$ be a function and let $D : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n \times \{0,1\}^m$ be a reversible circuit such that $D(\mathbf{y}, \mathbf{0}) = (g(\mathbf{y}), g_g(\mathbf{y}))$. Then let

$$K = (\mathbf{Id}_n \times D^{-1}) \circ (M \times \mathbf{Id}_m) \circ (\mathbf{Id}_n \times D) \circ H.$$

Then $K(\mathbf{x}, \mathbf{0}, \mathbf{0}) = (\mathbf{0}, f(\mathbf{x}), \mathbf{0})$ for all inputs $\mathbf{x}$, but K is efficiently computable using a reversible circuit.

The following circuit diagram illustrates how reversible computation could be used to compute any efficiently computable bijective function with an efficiently computable inverse.



Today there are no commercially available reversible computers, and today most researchers are only interested in reversible computation since reversible computation is a major component of quantum computation. While reversible computers are

not commercially available, there are currently reversible programming languages and reversible compilers[1].

Since reversible computation requires a space and time overhead, the free market is currently hesitant to invest in the research and development of reversible computing devices. However, since the space and time overhead for reversible computation is quite modest, we believe that in the future, the most efficient method of computing will be nearly reversible computation. It will therefore be beneficial to humanity for an entity to incentivize the development of the reversible computer.

3.1. **Evidence for the dominance of reversible computing.** There is already plenty of experimental evidence that reversible computers whose energy efficiency goes beyond Landauer's limit are possible and that such computers will be constructed shortly. For example, in [5], the authors have calculated that a mechanical computer can compute a logical gate operation while spending about 10^{-26} joules which is well below Landauer's limit for any reasonable temperature. Such a mechanical computer could outperform today's computers in by a factor of 100 billion while consuming the same amount of energy. While such a reversible computer is physically possible, it is not feasible to manufacture such a reversible computer any time soon.

There are other proposed technologies for reversible computers which could be developed in the near future since these technologies could be fabricated in similar ways that current integrated circuits are fabricated . Furthermore, people have experimentally shown that some of these reversible technologies can be more energy efficient than conventional computers and also than Landauer's limit. For example, experiments have demonstrated that some reversible superconducting circuits dissipate less energy than Landauer's limit [8].

4. Hashspin and RCO-POW problems

4.1. **Todays POW problems are not designed for reversible circuits.** There are currently no RCO-POW problems implemented for any cryptocurrency nor have people strived to make cryptocurrency POW problems useful outside the cryptocurrency. We shall now give an overview of the current POW problems implemented in cryptocurrencies.

Cryptographic hash functions-Many cryptocurrency POW-problems require one to find peculiar hashes simply by evaluating as many inputs as possible, but today's cryptographic hash functions are not designed to be efficiently computed using a reversible computer. Since a cryptographic hash function maps data of an arbitrary length to data of a fixed length, cryptographic hash functions are by their very definition at least partially irreversible. Nevertheless, cryptographic hash functions today employ a significant amount of reversibility since a certain amount of bijectivity is needed for collision resistance and other security properties of hash functions. Furthermore, these bijective components are often computable just as easily using a reversible computer since cryptographic hash functions need to be computed as efficiently as possible.

Even if a cryptographic hash function were designed to be solved by a reversible computer, such a function would not make a good RCO-POW problem for several

[1]The programming language Janus, created in 1982, is the first time-reversible programming language.

reasons. For a hash-based POW problems with an n bit hash, one must delete all n bits of information after every hash attempt or uncompute the hash function. More importantly, cryptographic hash functions do not employ the SLT which means that they require many more bits of security than is required for a POW problem and therefore cryptographic hash functions need to be designed for cryptographic security and efficiency instead of for accelerating the development of reversible computers. This means that cryptographic hash functions are poor RCO-POW problems even if they are designed to be computed using reversible computers.

The move from hash functions to ASIC resistance- In the cryptocurrency ecosystem, there is a move away from cryptographic hash function based POW problems such as SHA-256 and towards ASIC resistant POW algorithms. Right now over 80 percent of all Bitcoins have already been mined, so there is little chance that one would consider constructing a reversible computer for mining Bitcoins even if SHA-256 were designed to be computed by a reversible computer. On the other hand, ASIC resistance is currently incompatible with reversible computation optimization. Even though ASIC resistant POW problems remain a popular type of consensus algorithm, the arguments in favor of ASIC resistance are weak.

4.2. **A template for RCO-POW problems with SLT.** In this section, we shall give a template for RCO-POW problems using the SLT which we shall call Template FP (FP stands for "fixed-point"). Before we discuss Template FP, we shall introduce a less sophisticated template for RCO-POW problems which we shall call Template HL (HL stands for "hash-like") in order to have something to compare Template FP with.

Input for Template HL: Let m, n be positive integers with $n > m$, and let $f : \{0,1\}^n \to \{0,1\}^n$ be a function designed to be computed using a reversible circuit. Let C be an adjustable constant.

Problem for Template HL: Find an appropriate m bit hash k along with an $m - n$ bit string x such that $f(k||x) < C$. k will be the hash of the Merkle root of the current block, the previous block hash and other information.

The optimal algorithm for solving the problem for Template HL needs to be to simply a brute force approach, so the function f needs to be nearly as cryptographically secure as a cryptographic hash function or an encryption function.

Input for Template FP: For each m bit hash k, let $f_k : \{0,1\}^n \to \{0,1\}^n$ be a bijective function easily computable by reversible circuit without ancilla and without garbage bits (the function f_k is known as the base function). Let r be a natural number which is preferrably near $n/2$. Let $\pi : \{0,1\}^n \to \{0,1\}^r$ be a projection function. We are also given a cryptographic hash function H along with a number C.

Problem for Template FP: Find a suitable m bit long hash k along with an n bit string x such that $\pi(x) = \pi(f_k(x))$ and $H(k||x||f_k(x)) < C$.

There are several clear advantages of using Template FP for an RCO-POW problem. Since Template FP uses the SLT, nearly all of the resources spent on solving the RCO-POW problem shall be spent on computing the function f_k as opposed to hashes. After every computation of the function f_k, one only needs to delete at most r bits of information. Here r could be very low (for example, $r = 16$ or $r = 32$). On the other hand, for Template HL, one would need to delete all m bits. Today, for a cryptographic hash function, 256 bit hashes are required for

collision resistance, so one would typically need to delete 256 bits of information after every solution attempt using Template HL.

A more important reason to use Template FP over Template HL is that the functions f in Template HL require a much higher level of cryptographic security than the functions f_k in Template FP. The cryptographic security of a POW problem with Template FP is based upon the difficulty of finding a previously undiscovered string x such that $\pi(f_k(x)) = \pi(x)$. However, the problem of finding a new x with $\pi(f_k(x)) = \pi(x)$ only requires r bits of security. Furthermore, since the functions f_k are designed to be computed quickly using specialized hardware, in practice, the problem of finding a new x with $\pi(f_k(x)) = \pi(x)$ only requires much less than r bits of security.

The low security requirement of the functions f_k under Template FP give cryptographers more freedom in designing the function f_k in order to accelerate the development of the reversible computer. Today RCO-POW problems must employ new cryptographic functions which have not stood the test of time and the scrutiny of the cryptographic community. Therefore, the low security requirements of the functions f_k are necessary for the cryptographic security of f_k and for the public to quickly gain confidence in the security of the the function f_k.

Non-commuting factors condition for Template FP: The function f_k must satisfy the non-commuting factors condition which states that there cannot be two non-trivial functions g_k and h_k such that $f_k = g_k \circ h_k = h_k \circ g_k$ and where $\mathrm{RDF}(f_k) \approx \mathrm{RDF}(g_k) + \mathrm{RDF}(h_k)$ and where $\mathrm{RDF}(f)$ is a measure of the difficulty of computing the function f by a reversible circuit without garbage bits. The non-commuting factors condition is violated by a function f_k such that $f_k = g_k^v$ for some function g_k with $\mathrm{RDF}(f_k) \approx v \cdot \mathrm{RDF}(g_k)$ and where g_k^v denotes the v-fold iteration of the function g_k. The non-commuting factors condition is desired for RCO-POW problems that satisfy Template FP because the optimal reversible algorithm for solving an RCO-POW problem that does not satisfy the non-computing factors condition for Template FP requires one to delete more than r bits of information for every time the function f_k is computed. For example, if $f_k = g_k^v$ and $\mathrm{RDF}(f_k) \approx v \cdot \mathrm{RDF}(g_k)$, then the optimal algorithm for solving the RCO-POW problem with Template FP requires one to delete $v \cdot r$ bits of information for each time the function f_k is computed (and the optimal algorithm requires one to store the $v \cdot r$ bits of information for each instance the computer is computing f_k in parallel).

4.3. **The RCO-POW problem Hashspin for Zammazazzer.** The RCO-POW Hashspin shall use Template FP where the base function f_k is a function designed to be computed using two linear feedback shift registers (abbreviated LFSRs).

We have chosen to use LFSRs for Hashspin for several reasons. First of all, LFSRs are reversible, so there will be no algorithmic overhead from running an LFSR reversibly. LFSRs have already been used in symmetric cryptography in stream ciphers and pseudo-random number generators, so one can be more confident in the cryptographic security of LFSR based cryptosystems. LFSRs are exceedingly simple so reversible computing manufacturers will be able to quickly construct reversible devices that compute these LFSRs. Lastly, LFSRs are energy intensive in the sense that in order to maximize mining profits, each register of these LFSRs must be running at all times and therefore hardware manufacturers will need to use reversibility in order to minimize this energy usage.

We shall now define the base function and projection function for Hashspin.

Definition 4.1. Define

$$S : \{0,1\}^{15} \times \{0,1\}^{17} \to \{0,1\}^{15} \times \{0,1\}^{17}$$

to be the function where if

$$S((x_n)_{n=0}^{14}, (y_n)_n^{16}) = ((x'_n)_{n=0}^{14}, (y'_n)_{n=0}^{16}),$$

then

$$x'_n = x_{n-1 \mod 15}, y'_n = y_{n-1 \mod 17}.$$

Definition 4.2. Let

$$T : \{0,1\}^{15} \times \{0,1\}^{17} \to \{0,1\}^{15} \times \{0,1\}^{17}$$

be the function where if

$$T((x_n)_{n=0}^{14}, (y_n)_n^{16}) = ((x'_n)_{n=0}^{14}, (y'_n)_{n=0}^{16}),$$

then

(1) $x'_n = x_n, y'_n = y_n$ for $n \notin \{1, 3\}$,
(2) $x'_1 = x_0 \oplus x_1$,
(3) $x'_3 = x_3 \oplus (x_0 \wedge y_0)$,
(4) $y'_3 = y_0 \oplus y_3$, and
(5) $y'_1 = x_0 \oplus y_1$.

Definition 4.3. Suppose that $L_a((x_n)_{n=0}^{14}, (y_n)_n^{16}) = ((x'_n)_{n=0}^{14}, (y'_n)_{n=0}^{16})$. Then $y'_n = y_n$ for all n, $x'_n = x_n$ whenever $n \neq 2$, and $x'_2 = x_2 \oplus a$.

Definition 4.4. Suppose that $k = a_0 \dots a_{255}$ is a 256 bit hash. Define

$$f_k = (L_{a_{255}} \circ (T \circ S)^4) \circ \cdots \circ (L_{a_0} \circ (T \circ S)^4)$$

Definition 4.5. Define

$$\pi : \{0,1\}^{15} \times \{0,1\}^{17} \to \{0,1\}^8 \times \{0,1\}^8$$

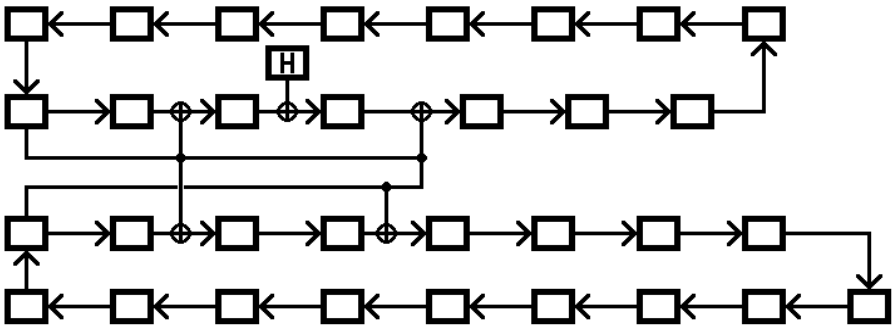
to be the function where if

$$\pi((x_n)_{n=0}^{14}, (y_n)_{n=0}^{16}) = ((x'_n)_{n=0}^{7}, (y'_n)_{n=0}^{7}),$$

then $x'_n = x_{n+7}$ and $y'_n = y_{n+9}$ for $n \in \{0, \dots, 7\}$.

The POW problem Hashspin is the POW problem under Template FP with base function f_k and projection function π.

The following image is the circuit diagram for a single round of the base function f_k of Hashspin. The register with the letter H inscribed denotes the location in which the data from the hash is added onto the LFSR. Observe that Hashspin requires the only shifts, 3 CNOT gates, one Toffoli gate, and another CNOT gate for inputting the data from the 256 bit hash onto the shift register.



The following text is a program in the reversible programming language Janus that mines the function Hashspin with some obvious modifications. One can test this code using an online interpreter for Janus at [6]. Since Janus is a completely reversible programming language, the garbage information produced after every solution attempt is kept in the array "garbage."

```
procedure roundmap(int x[15],int y[17])

local int i=14
local int j=16

from (i=14) loop
i-=1
x[i]<=>x[(i+1)%15]
until (i=0)

from (j=16) loop
j-=1
y[j]<=>y[(j+1)%17]
until (j=0)

x[1]^=x[0]
x[3]^=x[0]&y[0]
y[3]^=y[0]
y[1]^=x[0]

delocal int j=0
delocal int i=0

procedure basefunction(int hash[256],int x[15],int y[17])

local int i=0
local int j=0

from (i=0) loop
i+=1

from (j=0) loop
```

```
j+=1
call roundmap(x,y)
until (j=4)
j-=4

x[2]^=hash[i-1]
until (i=256)

delocal int j=0
delocal int i=256

procedure readoff(int x[15],int y[17],int w[16])
local int i=0

from (i=0) loop
w[i]^=x[i+7]
i+=1
until (i=8)
i-=8
from (i=0) loop
w[i+8]^=y[i+9]
i+=1
until (i=8)

delocal int i=8

procedure main()
int garbage[10][16]
int i=0
int j=0
int hash[256]

int x[15]={1,0,1,0,1,1,0,1,1,0,0,1,0,1,0}
int y[17]={1,1,0,0,0,0,0,1,0,0,1,0,1,0,1,0,1}

from (i=0) loop
hash[i]+=((i%3)+(i%5)+(i%2)+(i%17))%2
i+=1
until i=256
i-=256

j+=1
call readoff(x,y,garbage[0])
call basefunction(hash,x,y)
call readoff(x,y,garbage[1])

from (i=0) loop
i+=1
```

```
j+=1
call basefunction(hash,x,y)
call readoff(x,y,garbage[j])

until (i=10)||(garbage[j][0]=garbage[i][0]&&garbage[j][1]=garbage[i][1])
```

While Hashspin relies solely upon the use of one Toffoli gate per round for its non-linearity, this one Toffoli gate in some sense produces a maximal amount of non-linearity in the following sense.

For $0 \leq r < 10$, there is a linear operator A where

$$A((x_n)_{n=0}^{14}, (y_n)_{n=0}^{16}, (S \circ T)^r (x_n)_{n=0}^{14}, (y_n)_{n=0}^{16}) = x_0 \cdot y_0 \oplus \cdots \oplus x_{r-1} \cdot y_{r-1}.$$

In other words, the Toffoli gates essentially add the inner product onto the data in the LFSRs. In Section 6.2, we shall see that the inner product is a maximally non-linear function and this maximal non-linearity will help produce a great amount of cryptographic security for Hashspin.

5. The Zammazazzer Foundation

The Zammazazzer Foundation shall be an organization funded through the initial distribution of the Zammazazzer cryptocurrency. The primary goal of the Zammazazzer Foundation is to promote the development and lead the direction of the cryptocurrency Zammazazzer. The secondary goal of the Zammazazzer foundation will be to advance reversible computation mainly be researching, developing, and evaluating reversible cryptosystems. Since it is difficult to predict the future market cap of a cryptocurrency, the Zammazazzer Foundation may have a large quantity of funds leftover after maintaining Zammazazzer and researching reversible cryptosystems. These leftover funds shall be spent on projects which will further advance reversible computation and these projects will likely include developing open source reversible software and on educating talented computer programmers on reversible computation. The Zammazazzer Foundation has no plans on spending funds to directly research or develop reversible hardware since the process of developing hardware is best suited for a for-profit entity. The Zammazazzer Foundation will not spend funds nor participate in any activities directly related to Zammazazzer, reversible computation, cryptocurrencies, or cryptography.

5.1. Some Zammazazzer specifications.

The specifications for Zammazazzer are tentative and will only be set in stone when Zammazazzer is launched.

ICO: We will not use an ICO to distribute any coins. An ICO goes against the principles of using a fair initial distribution mechanism and of incentivizing the development of the reversible computer.

Smart contracts: The initial cryptocurrency that we shall launch shall not admit Turing complete smart contracts. However, we eventually want to launch another cryptocurrency which will admit smart contracts so more people in the cryptocurrency community will take part in the development of the reversible computer. We would also like to launch a smart contract based cryptocurrency since smart contracts can be used to evaluate new cryptosystems and since these smart contracts should be run on a cryptocurrency which is secured using an RCO-POW problem instead of a wasteful POW problem.

Halving time: There will not be a halving time the block reward for Zammazazzer. The block reward will remain the same for all time unless there is an unquestionable supermajority consensus among miners, stakeholders, developers, and other parties to change the block reward.

Average block time: A new block for Zammazazzer will be formed on average every 2 minutes.

What will the coins be called? Each coin shall be called a CIRC which stands for Certificate of Innovation in Reversible Computation. The symbol for Zammazazzer shall be CIRC.

Block reward: For every block, 56 CIRCs will be distributed to the miner and 8 CIRCs shall be distributed internally in order to fund Zammazazzer. This will amount to 64 CIRCs distributed in total for every block.

Future Zammazazzer Foundation funding: The Zammazazzer Foundation will never increase its funding beyond 12.5% of all newly minted coins. In order to promote a decentralized ecosystem, the Zammazazzer Foundation may reduce or completely eliminate its funding. The Zammazazzer Foundation may also eventually upgrade to a more democratic governance and funding mechanism.

5.2. **Funding symmetric cryptography research and standardization.** We plan for a portion of the funds to the Zammazazzer Foundation to be spent on researching and standardizing reversible symmetric cryptosystems. This expenditure is in line with the goal of promoting the development of the reversible computer, and this expenditure will help give Zammazazzer its intrinsic value. We shall use cryptocurrency smart contracts as much as possible in the process of standardizing these reversible cryptosystems in order to promote Zammazazzer and in order to more fairly standardize reversible cryptosystems.

The second real-world application of reversible computation after the application of reversible computers to mine Zammazazzer will be in computing symmetric cryptosystems in hardware. Furthermore, since Zammazazzer is a cryptocurrency, the Zammazazzer Foundation will be able to employ cryptographers who are able to research, evaluate, and standardize these reversible cryptosystems. The symmetric cryptosystems which we shall investigate shall be evaluated mainly in terms of their reversible hardware performance and security. Since Zammazazzer will incentivize the development of the reversible computation through its RCO-POW problem, it makes sense to further accelerate the development of reversible computation by researching and standardizing reversible cryptosystems so that people may safely use reversible cryptosystems in reversible hardware instead of the cryptosystems that we have today.

Currently, symmetric cryptosystems such as cryptographic hash functions, encryption systems, and pseudo-random number generators are somewhat reversible in the sense that these cryptosystems employ many bijective functions and some of these functions can be computed reversibly without any need for uncomputation, any extra steps, or any less parallelism. Furthermore, it is possible to construct efficient and secure symmetric cryptosystems which can be computed in a reversible computer in just as many steps as they can be computed using a conventional computer. Reversible computation will become crucial in low-power devices which need to spend energy performing cryptographic functions. Reversible computation will also improve the security of these symmetric cryptosystems because energy

efficient reversible computation may be used to thwart side-channel attacks such as power-analysis attacks and attacks from analyzing electromagnetic radiation.

The goal in our investigations into symmetric cryptography will be to standardize reversible cryptosystems so that those cryptosystems are ready to use as soon as hardware manufacturers are able to make reversible hardware that can perform those reversible algorithms better than conventional hardware is able to perform conventional cryptographic algorithms.

Most public key and advanced cryptosystems use functions which are not as naturally suited for reversible computers, so the Zammazazzer Foundation will only fund the development of these algorithms if they could be used in the cryptocurrency sector. The hash-based signature algorithms are a notable exception.

We shall now elaborate on the types of reversible cryptosystems which the Zammazazzer Foundation plans on researching and standardizing.

Reversible block ciphers-Since encryption and decryption are inverse functions of one another and because no particular kind of bijective function is required in a block cipher, block ciphers are the most natural candidate for the first free market use of reversible computation outside of mining cryptocurrencies. However, today's block ciphers are not designed to be computed using reversible computers and therefore one will incur some computational overhead if one wants to compute one of today's block ciphers using a reversible computer. The Zammazazzer Foundation will therefore research and standardize at least one block cipher which is designed to be computed using reversible hardware so that one does not incur a computational overhead simply from reversible computation of block ciphers.

Reversible hash functions-While cryptographic hash functions are never bijective, cryptographic hash functions always contain mostly bijective components in order to ensure cryptographic properties such as collision resistance. As with block ciphers, today's cryptographic hash functions are not designed to be computed using reversible computers. However, hash functions together with message authentication codes and one-way functions can be designed to be efficiently computed in reversible hardware.

Reversible pseudorandom number generation-Suppose that m and n are natural numbers. Let $f : \{0,1\}^m \times \{0,1\}^n \to \{0,1\}^m \times \{0,1\}^n$ be a function designed to be computed by a reversible circuit on $m+n$ bits without any ancilla or garbage bits. Then suppose that $(x_0, y_0) \in \{0,1\}^m \times \{0,1\}^n$ is the seed state. Suppose now that $f(x_k, y_k) = (x_{k+1}, y_{k+1})$ for each k. Then the sequence of numbers $x_0, \ldots, x_k, \ldots$ is a pseudorandom number sequence optimized for computation using a reversible computer.

People have already constructed pseudorandom number generators for stream ciphers which do not incur any overhead when one computes these stream ciphers reversibly. LFSRs are commonly used to construct stream ciphers and the algorithms for computing these LFSRs are already completely reversible. Since non-linearity is required for cryptographic security of all cryptosystems, LFSRs by themselves are unable to achieve cryptographic security for pseudorandom number generators, so these LFSRs must be modified or combined together in order to become suitable for cryptographic purposes, and these LFSRs are often combined in an irreversible way.

Reversibility resistant key derivation functions-When reversible computers come about, one may need key derivation functions and password hashing functions which are not only memory hard but are also inherently irreversible so that brute force attacks must either use an exceedingly great amount of memory or use a large amount of irreversibility. While reversibility resistant key derivation functions will not accelarate the development of reversible computers, reversibility resistant key derivation functions will help prepare people for the advent of reversible computers.

Advanced reversible cryptosystems-We plan on researching advanced cryptosystems which will incentivize the research and development of abstract reversible combinatorial circuits with low computational complexity since the efficiency of these cryptosystems will depend on the computational complexity of the reversible combinatorial circuits in which these cryptosystems act on.

Cryptosystems for cryptocurrencies-The Circlefish shall spend

5.3. **Tokenized cryptanalysis.** The Zammazazzer Foundation shall use smart contracts in order to research, evaluate, and to help standardize cryptosystems.

Smart contracts can to a large extent be used to help evaluate reversible symmetric cryptosystems. Advanced cryptographic algorithms based upon reversible circuits are more difficult to analyze using smart contracts, but these advanced algorithms are closely related to other cryptographic algorithms which can easily be analyzed using smart contracts. In the past, symmetric cryptosystems such as the DES,AES,SHA-2, and SHA-3 have been standardized by government agencies instead of the private sector. Smart contracts within cryptocurrencies however make it feasible and profitable for private organizations to produce, fairly and objectively evaluate, and standardize future symmetric cryptosystems. The basic idea behind these smart contracts is to award entities with tokens or CIRCs [2] for producing secure cryptosystems or for finding cryptographic weaknesses in cryptosystems. Since these cryptosystems will be evaluated using smart contracts, there is relatively little risk of a biased evaluation of these cryptosystems.

These tokens will be valuable for several reasons:

(1) these tokens are scarce items which cannot be copied and can only be produced by helping solve real-world cryptography problems,
(2) the owners of these tokens will receive dividends simply from owning these tokens, and
(3) these tokens, through a technique called "proof-of-burn," may be traded in for other tokens or coins in the initial distribution phase of a new cryptocurrency or token.

The purpose of using a smart contract for a cryptocurrency token to standardize cryptographic algorithms is multifold:

(1) A cryptocurrency token and CIRCs are a good incentivization for cryptographers to develop secure and efficient cryptosystems. A smart contract may be used to reward cryptographers who develop secure and efficient cryptosystems.

[2]The problems that one must solve in order to obtain these tokens cannot be POW problems for their own blockchains for several reasons. For instance, the problems which one must solve to obtain these tokens are highly interactive problems while POW problems are required to be non-interactive.

(2) A cryptocurrency token and CIRCs also incentivizes a thorough cryptanalysis of a cryptosystem by rewarding an entity who finds a cryptographic weakness in a cryptocurrency.
(3) A smart contract is naturally an objective, transparent, and honest way to measure the security and efficiency of a cryptosystem.
(4) A smart contract gives cryptographers objective standards to achieve.

The efficiency of a reversible cryptosystem in the hardware can easily be measured through a smart contract. The following security properties among other security properties of symmetric cryptosystems can be analyzed using a smart contract:

(1) Collision resistance-An entity will receive a reward for creating a collision resistant cryptographic hash function. Another entity will receive a reward for finding a collision or near collision in a weakened or modified version of the cryptographic hash function. Similar properties of hash functions such as preimage resistance, and second preimage resistance can also be analyzed in a similar manner using cryptocurrency tokens.
(2) Reversibility-A smart contract can be designed so that only reversible cryptosystems may be submitted for cryptanalysis. To maximize efficiency, cryptographers will need to avoid using irreversible components, ancilla bits, garbage bits, and uncomputation in their cryptosystem design.
(3) The effectiveness of symmetric encryption in producing a cryptographic hash function-There are several ways of producing a cryptographic hash function from a symmetric encryption algorithm, so one can measure the strength of a symmetric encryption algorithm by measuring how well it behaves as a cryptographic hash function.
(4) Other security properties can be analyzed using a cryptocurrency token.

While some of the security properties of cryptocurrencies can be analyzed algorithmically through a decentralized token, not all of the security properties of cryptocurrencies can be easily measured algorithmically through a cryptocurrency token including the following characteristics:

(1) A smart contract cannot completely take into consideration a cryptographer's 'gut feeling' about the security of a cryptosystem based upon knowledge of previous cryptosystems.
(2) Sometimes theoretical attacks are too computationally costly to carry out in practice. These attacks are difficult for a smart contract to take into consideration unless the smart contract has access to an automatic theorem verification system.
(3) The developers of a cryptographic algorithm may have a secret attack against a cryptosystem. A smart contract may be able to thwart some attacks based upon secret knowledge by rewarding the development of simple cryptosystems and to only use a designated set of nothing-up-my-sleeve numbers, but it still may be possible to construct a cryptosystem based upon secret knowledge even if the token has these automatic protective measures.
(4) A token for evaluating cryptosystems incentivizes organizations to keep information secret. For example, an attacker is incentivized to keep his algorithm for attacking a cryptosystem a secret in order to maximize his own profit (on the other hand, some of this secret information may recoverable through reverse engineering). Therefore, the token distribution period

should end before the cryptosystem is standardized and all design decisions for the cryptosystems should be explained during the standardization process.

(5) Some attacks against a symmetric cryptosystem require a large quantity of plaintexts and/or ciphertexts to be analyzed. Such attacks would be too costly to analyze on the blockchain.
(6) A token may not give a completely accurate reading of the security of a cryptosystem since there may be specialized hardware for breaking one cryptosystem while the specialized hardware for analyzing another cryptosystem may not yet be available. Furthermore, people may expend more effort at attacking one cryptosystem rather than another.

We shall therefore use the results obtained by cryptocurrency tokens as simply objective guidelines for the security of the underlying cryptosystems rather than as the final say in the security of cryptosystems.

Sample token: We shall now describe a proof-of-concept design for a token for evaluating a public key encryption system using a cryptocurrency token. While we formulate this proof-of-concept design for a smart contract for evaluating public key encryption systems, this proof-of-concept could be modified to evaluate all sorts of cryptosystems. This proof-of-concept works best in a public key cryptosystem that satisfies the following conditions:

(1) The public key cryptosystem is relatively new and needs to be researched,
(2) The security of the public key cryptosystem heavily depends on the choice of the private key and the problem of determining a short and secure private key for the public key cryptosystem is a cryptographic challenge,
(3) There are an endless number of strategies for producing a secure private and public key pair, and
(4) The security level of the proposed cryptosystem is finely tunable by trading security for efficiency.

For the proof-of-concept, suppose that for each private key k, $\mathrm{pub}(k)$ is the public key corresponding to k, $E_{\mathrm{pub}(k)}$ is our public key encryption function, and D_k is our private key decryption function. Let ℓ be a function where $\ell(k)$ is a measure of the inefficiency of the public key encryption/decryption with k as the private key. Let $n_{\max}$ be a number. The number $n_{\max}$ shall be used in order to control the difficulty of an attack against the public key cryptosystem. The number $n_{\max}$ shall be periodically adjusted so that approximately 50 percent of the time, an entity is able to break the public key cryptosystem during the evaluation period.

The smart contract for this proof-of-concept shall run in cycles consisting of two time periods which we shall call the bidding period and the evaluation period.

Bidding period: During the bidding period, entities bid for the right to submit their encrypted message and public key for cryptanalysis. All entities are allowed to bid and all bids are posted on the blockchain. All bidders shall be unable to spend the bidded amount until the bidding period is over. Suppose that the highest bidder which we shall call Alice bids r CIRCs. Then Alice will be unable to spend those r CIRCs during the evaluation period.

Evaluation period: At the beginning of the evaluation period, Alice selects a private key k with $\ell(k) \leq n_{\max}$ and she posts $\mathrm{pub}(k)$ along with $y = E_{\mathrm{pub}(k)}(x)$ for some message x.

If Bob is able to determine the message x with $y = E_{\mathrm{pub}(k)}(x)$ or the private key k, then Bob commits the message x or the private key k. After Bob's commitment has been on the blockchain for several blocks, Bob reveals the message x or the private key k. Alice will then lose her r CIRCs. Alice's r CIRCs shall be distributed to the Zammazazzer Foundation, Bob as a reward for breaking Alice's public key encryption, and as dividends to the owners of the tokens in order to help give the tokens value.

If nobody is able to determine the message x or the private key k by the end of the evaluation period, then Alice will reveal her private key k. If Alice is unable to produce a private key k, then Alice's r CIRCs shall be given to the Zammazazzer Foundation and to the token owners. If Alice produces her private key k, then Alice will be refunded her r CIRCs and she shall be given a token.

In this proof-of-concept, Alice is incentivized to bid a high amount so that she is the winning bidder. Alice is also incentivized to find a secure private key k since Alice does not want to lose her r CIRCs and she wants to obtain a token after the evaluation period. An attacker Bob is incentivized to obtain Alice's private key k since Bob would like to obtain a portion of Alice's r CIRCs. Alice is disincentivized from breaking her own cryptosystem since if Alice breaks her own cryptosystem, then Alice cannot receive a token, and Alice would have to give some of her CIRCs away.

6. A justification for the Hashspin design and reversibility in cryptography

In Section 6.1, we shall explain the mathematical background behind LFSRs and why they are useful in symmetric cryptography, and we shall justify the design of Hashspin. In the following sections, we shall further justify how reversible computing can build just as secure and efficient symmetric cryptosystems as conventional computation by looking at the specific cases of key scheduling and the use of Toffoli gates in symmetric cryptography.

6.1. LFSRs and a justification for Hashspin.

We shall give an exposition of LFSRs and we shall explain the design for Hashspin.

Suppose that F is a field. Then let F^* denote the set of all non-zero elements of F. Then F^* is a group under multiplication.

Theorem 6.1. *If F is a field and K is a finite subgroup of F^*, then K is cyclic. In particular, if F is a finite field, then F^* is cyclic.*

Suppose that F is a finite field and $a, a_0, \ldots, a_{k-1} \in F$. Then the sequence $(s_n)_n$ is said to be a k-th order linear recurring sequence if it satisfies the recurrence relation

$$s_{n+k} = a_{k-1}s_{n+k-1} + \cdots + a_0 s_n + a. \tag{1}$$

Define $\mathbf{s}_n = (s_n, \ldots, s_{n+k-1})$ for each n. Then $\mathbf{s}_n$ is said to be the n-th state vector and the vector $\mathbf{s}_0$ is said to be the initial state vector of the sequence $(s_n)_n$.

If $a = 0$, then we say that $(s_n)_n$ is a k-th order linear homogeneous recurring sequence and the relation 1 is said to be a homogeneous recurrence relation. The polynomial $f(x) = x^k - a_{k-1}x^{k-1} - a_{k-2}x^{k-2} - \cdots - a_0 \in F[X]$ is said to be the characteristic polynomial of the sequence $(s_n)_n$.

If F is a finite field, then a generator a of the cyclic group F^* is said to be a primitive element of F.

For a finite field F, a polynomial $f(x) \in F[X]$ is said to be a primitive polynomial if it is the minimal polynomial of some primitive element $a \in K$ for some finite field extension K of F.

A homogeneous linear recurring sequence in a field F with a primitive characteristic polynomial and a non-zero initial state vector shall be called a maximal period sequence in F.

Theorem 6.2. *Suppose that F is a field of order q. Every k-th order maximal period sequence in a field F is periodic with period $q^k - 1$, the largest possible period of a k-th order period sequence.*

A linear feedback shift register (abbreviated LFSR) for a primitive polynomial f is a piece of hardware which will have the n-th state vector $\mathbf{s}_n$ as its state during the n-th cycle and which will output the bit s_n at the n-th cycle.

Suppose that $p(x) = x^k \oplus a_{k-1}x^{k-1} \oplus a^{k-2}x^{k-2} \oplus \cdots \oplus a_0$ is a primitive polynomial of degree k over the field F_2. Then an LFSR for the polynomial p could be constructed using only $|\{i \mid i \in \{1, \ldots, k-1\}, a_i \neq 0\}|$ XOR gates along with machinery for shifting bits. In particular, if $p(x)$ is a trinomial over F_2, then an LFSR for $p(x)$ only needs one CNOT gate.

While LFSRs over F_2 require exceedingly simple hardware, they are often used in symmetric cryptography to produce stream ciphers since the linear homogeneous recurring sequence produced as an output by the LFSR is a decent pseudo-random bit sequence.

Hashspin is constructed using two LFSR's, namely the 15 bit LFSR over the field F_2 with primitive feedback polynomial $x^{15} \oplus x \oplus 1$, and the 17 bit LFSR over the field F_2 with primitive feedback polynomial $x^{17} \oplus x^3 \oplus 1$. We chose these LFSRs since they are of relatively prime lengths (relatively prime LFSR lengths are desirable for cryptographic security since these LFSRs will be unrelated to each other), and since the sum of their lengths is $15 + 17 = 32$ which is a power of 2. We also chose LFSRs associated with primitive trinomials since an LFSR associated with a primitive trinomial could be built using simpler hardware than a general primitive polynomial. The lengths 15 and 17 of the LFSRs are also relatively prime to 4, the number of cycles the LFSRs must run before another bit from the 256 bit hash is added to the LFSRs; since $4, 15, 17$ are pairwise relatively prime, there is very little chance that a cryptanalyst could exploit some hidden synchronization between the LFSRs and the hash bits being inputted into the LFSRs in order to break Hashspin.

In Hashspin the 15 bit and the 17 bit LFSRs are interconnected using a CNOT gate and a Toffoli gate so that these LFSRs influence each other and because the Toffoli gate provides the non-linearity which is required in any cryptosystem. The reversible component of the function Hashspin is as simple as possible; there does not exist a CNOT or Toffoli gate that can be removed from the reversible component Hashspin without rendering Hashspin completely insecure.

6.2. **Toffoli gates and CNOT gates for symmetric cryptography.** Reversible gates such as the Toffoli gate together with the linear CNOT gate are ideal logic gates for producing secure symmetric cryptosystems even if there were no possible performance gains stemming from reversible computation.

Through Gaussian elimination, every invertible linear transformation $L : F_2^n \to F_2^n$ can be computed through using a composition of at most $O(n^2)$ CNOT gates. An improved algorithm in [4] decomposes every linear transformation $L : F_2^n \to F_2^n$ into $n^2/\log_2(n)$ CNOT gates. The algorithm in [4] is optimal since almost every linear transformation $L : F_2^n \to F_2^n$ needs $n^2/(2 \cdot \log_2(n))$ CNOT gates. CNOT gates are therefore ideal for producing the linear transformations which are needed for symmetric cryptography without the need for ancilla bits nor garbage bits.

Toffoli gates provide the necessary non-linearity for symmetric cryptosystems. To illustrate how Toffoli gates can be used to effectively construct cryptosystems, we shall illustrate how Toffoli gates can construct bent functions which are useful in symmetric cryptography to thwart potential differential attacks and linear algebraic attacks.

Suppose that $f, g : F_2^n \to F_2$ are functions. Then define the Hadamard distance $d(f, g)$ from f to g by $d(f, g) = |\{x \in F_2^n | f(x) \neq g(x)\}|$.

Define the non-linearity $\mathrm{nl}(f)$ of a function $f : F_2^n \to F_2$ to be $\min\{d(f, g) : g \text{ is linear}\}$ where d denotes the Hadamard distance function.

We say that a function $f : F_2^n \to F_2$ is a bent function if n is even and $\mathrm{nl}(f) = 2^{n-1} - 2^{n/2-1}$.

Bent functions are precisely the functions $f : F_2^n \to F_2$ of maximal non-linearity with n even.

Since bent functions have a maximal degree of non-linearity and bent functions cannot be approximated in any way by linear functions, bent functions are used in symmetric cryptography in order to ensure resistance to linear algebraic attacks.

A balanced function is a function $f : F_2^n \to F_2$ where $\{x \in F_2^n | f(x) = 0\} = 2^{n-1}$ (i.e., the value $f(x)$ is 1 precisely $1/2$ of the time).

If a is a non-zero vector and f is a bent function, then the derivative $\Delta_a f$ defined by $\Delta_a f(x) = f(x) \oplus f(x \oplus a)$ is a balanced function. Since the derivative $\Delta_a f$ of a bent function f is always a balanced function, bent functions are used in symmetric cryptography to make the symmetric cryptosystems resistant to differential attacks.

The inner product function $Ip : F_2^n \times F_2^n \to F_2$ defined by $Ip(x_1, \ldots, x_n; y_1, \ldots, y_n) = x_1 \cdot y_1 \oplus \cdots \oplus x_n \cdot y_n$ is a bent function.

Toffoli gates can easily be used to construct bent functions. The function $f : F_2^n \times F_2^n \times F_2 \to F_2^n \times F_2^n \times F_2$ defined by

$$f(x_1, \ldots, x_n; y_1, \ldots, y_n; z)$$

$$= (x_1, \ldots, x_n; y_1, \ldots, y_n; z \oplus x_1 \cdot y_1 \oplus \cdots \oplus x_n \cdot y_n)$$

$$= (x_1, \ldots, x_n; y_1, \ldots, y_n; z \oplus Ip(x_1, \ldots, x_n; y_1, \ldots, y_n))$$

is computable by a reversible circuit consisting of solely n Toffoli gates. Toffoli gates together with CNOT gates could therefore be used to effectively incorporate many bent functions into a symmetric cryptosystem.

6.3. **Reversible key schedules.** The only possible source of irreversibility in a symmetric encryption-decryption system is in the key schedule algorithm. However, with slight care, we shall show that even key schedule algorithms can be made completely reversible even without requiring an unacceptable amount of memory and without uncomputation.

The key schedule is a component of a symmetric cryptosystem which may be a potential source of irreversibility or of an unacceptable memory usage.

Some level of reversibility is required in order to produce a secure key schedule. Suppose that our block cipher has secret key k and round key k_i for $i \in \{0, \ldots, n-1\}$, and suppose that $L_i(k) = k_i$ for $i \in \{0, \ldots, n-1\}$ (i.e. L_i is the function mapping the secret key k to the i-th round key).

Each of the round keys k_i should contain all of the information in the secret key k. In other words, each of the functions L_i should be bijective. Therefore, one should expect for there to exist good functions mapping secret keys to their key schedules which can be computed in just as many steps using a reversible computer as they can be computed on a conventional computer.

A minor problem: Suppose that $F_0, \ldots, F_{n-2}$ are functions such that $L_{i+1} = F_i \circ L_i$ for all i and $L_0 = \mathrm{Id}$ where each F_i is optimized for reversible computation. Then, for most functions F_i, one would need to uncompute $F_0, \ldots, F_{n-2}$ or delete the final round key k_{n-1} in order to $k = k_0$. However, if $F_0, \ldots, F_{n-1}$ are functions optimized for reversible computation such that $F_{n-1} \circ \cdots \circ F_0 = \mathrm{Id}$, then there is no need for uncomputation, deletion, or holding the expanded key $(k_0, \ldots, k_{n-1})$ in memory when constructing the key schedule. The following methods could be used to ensure that $F_{n-1} \circ \cdots \circ F_0 = \mathrm{Id}$.

Method 1: Let $n = 2m$ and select functions $F_0, \ldots, F_{n-1}$ such that $F_{m+i} = F_{m-i-1}^{-1}$ for $i \in \{0, \ldots, m-1\}$. With this method, we have $k_{m+i} = k_{m-i}$ for $i \in \{0, \ldots, m-1\}$. One should think of this method as a key schedule where the round keys are reused during the process of uncomputation and therefore the uncomputation has more use than simply resetting the round key back to k_0. The fact that each round key is used twice is only a minor security weakness since the round key k_{m+i} and the round key k_{m-i} could be applied to different portions of the data being encrypted and in different ways.

Method 2: There exists bijective linear transformations F over a vector space over F_2 such that $F^n = \mathrm{Id}$, and such low period linear transformations could be constructed using reversible linear cellular automata over F_2 of dimensions 1 or 2. The following conjecture has been empirically verified.

Conjecture 6.3. Suppose that A is a vector space over the field F_2. Let C be the collection of all cellular automata $f : A^{\mathbb{Z}_{2^n}^2} \to A^{\mathbb{Z}_{2^n}^2}$. If $a \in A$, then let $c_a = (a)_{i \in A^{\mathbb{Z}_{2^n}^2}}$ denote the identity function. Define a homomorphism $\phi : C \to \mathrm{End}(A)$ by letting $f(c_a) = c_{\phi(f)(a)}$. If $\phi(f)^p$ is the identity function, then $f^{p \cdot 2^{n+1}}$ is the identity function.

Since the key schedule is derived from the secret key using a linear function, one needs to take additional care in order to ensure the security of the symmetric cryptosystem; in particular, one needs to incorporate a larger quantity of non-linearity in components of the symmetric cryptosystem which are not a part of the key schedule. Nevertheless, the linearity of a key schedule is a small price to pay in order to produce a secure, efficient, and completely reversible symmetric cryptosystem.

7. Miscellaneous sections and conclusion

7.1. **Other cryptocurrencies.** We expect that in the future, other cryptocurrencies will use RCO-POW problems to secure their blockchains. While Zammazazzer may not give enough incentive to initially develop the reversible computer, these other cryptocurrencies together with Zammazazzer should provide enough incentive

for manufacturers to develop the reversible computer. In the near future, we recommend for other cryptocurrencies to use Hashspin or some minor variant thereof as their RCO-POW problem to make it easier for computational machinery manufacturers to produce reversible computers that can efficiently solve these cryptocurrency problems. If there are too many RCO-POW problems in the cryptocurrency ecosystem, then reversible machinery manufacturers must develop many different kinds of reversible devices in order to solve these RCO-POW problems which may be difficult if reversible computers have barely been on the free market. On the other hand, once reversible computers that can compute Hashspin arise on the free market, it may be worthwhile for other cryptocurrencies to use other RCO-POW problems besides Hashspin.

While other cryptocurrencies may employ an RCO-POW problem in order to incentivize the development of the reversible computer, other cryptocurrencies will probably not be effective in standardizing and researching cryptosystems related to reversible computation. After all, if there are several organizations standardizing reversible cryptosystems, then it will probably be difficult to determine which of these standards should be used in practice.

7.2. **Zammazazzer in a post-reversible world.** After reversible computers are commercially available and the demand for reversible computers can be easily sustained without cryptocurrency POW problems, there would be little need for Zammazazzer to use POW problems designed to be solved by a reversible computer. At this point, which will likely be many years in the future, the POW problem for Zammazazzer should be replaced by some other problems with scientific value. While there are no good useful POW problems found in other cryptocurrencies, since cryptocurrencies are still quite new, people will develop many useful POW problems for cryptocurrencies in the future. In the far future, after the POW for Zammazazzer is no longer needed in order to accelerate the development of the reversible computer and after a sufficient number of CIRCs have been distributed already, Zammazazzer may switch to a consensus algorithm that does not require mining.

7.3. **Decentralization and security.** The POW problem for Zammazazzer satisfies all of the desired properties that a hash-based POW problem satisfies except for the fact that the mining hardware manufacturing for Zammazazzer may be more centralized than it would be if Zammazazzer used a hash-based POW problem. Since Zammazazzer is designed to be mined by very specialized cutting-edge hardware, it is plausible that a single hardware manufacturer will initially hold a monopoly over producing all the hardware for mining Zammazazzer which will be a point of centralization. However, since Zammazazzer and similar cryptocurrencies will be the first market for reversible computing machinery, reversible computing machinery manufacturers will have a strong incentive to ensure that these cryptocurrencies remain secure. These manufacturers will therefore refrain from launching attacks against these cryptocurrencies. These businesses will likely also make a strong effort to ensure that no particular party obtains an exceedingly large share of the Zammazazzer mining power. In particular, these businesses will likely limit the amount of Zammazazzer mining they do themselves and instead sell the hardware they produce to other miners. These businesses will also promote and

use Zammazazzer in order to increase the market cap for Zammazazzer in order to more increase their profits from reversible computing devices.

7.4. **Hashspin probationary period:** Since little is known about the cryptographic security of the function Hashspin, when Zammazazzer is initially launched, the POW-problem Hashspin shall use extra rounds in order to achieve cryptographic security. For the first year, the function f_k will require for the hash k to be 1024 bits long and for the LFSRs to run for 4096 rounds. For the period between the first year and the third year, Zammazazzer will automatically and linearly decrease the amount of rounds and the length of the hash k until the hash k is 256 bits long and the LFSRs run for 1024 rounds. If someone finds a significant cryptographic weakness with Hashspin, then the Zammazazzer Foundation will modify Hashspin or replace Hashspin with another RCO-POW problem. Since the Zammazazzer Foundation will spend many resources developing reversible cryptosystems, the Zammazazzer Foundation will be the best organization to make these decisions about the cryptographic security of Hashspin.

7.5. **Misconceptions about RCO-POW problems.** There are many misconceptions about reversible computation and how reversible computation relates to cryptocurrencies, and these misconceptions have greatly hampered the acceptance of RCO-POW problems. Here are a few of these misconceptions.

The energy usage of cryptocurrency mining is justified.- Some state that the expense is justified since the expense is required to secure the cryptocurrency and to distribute newly minted coins. Others have stated that gold and fiat currencies require much more energy and resources than cryptocurrencies. These arguments poorly attempt to justify the use of a POW problem even if it expends much energy, and these arguments do not justify the use of a hash-based POW problem over a useful POW problem.

The wasted resource problem is solved by POS.-Some have focused on using POS as a solution to this problem. However, with the influx of recent ICO scams, the cryptocurrency community will need to continue to rely on POW for a fair initial distribution regardless of the success of POS as a consensus mechanism. In any case, the cryptocurrency community should look towards multiple potential solutions to the wasted energy problem instead of just POS.

I don't care about wasted resources. I just want to make money.-If you think this way, then I have some bad news for you. You are a person who is likely to fall for a scam or a bubble. A cryptocurrency that does not waste resources on a useless POW problem and which does not scam people through an ICO will be a better investment.

Conventional irreverible computation can be made arbitrarily efficient.-Some otherwise educated people believe that conventional computation can be made arbitrarily efficient without the need of reversible computation and they continue in this erroneous belief even when presented with conflicting factual information such as Landauer's principle, the laws of thermodynamics, and the fact that the laws of physics are time reversible.

All computational operations cost $k \cdot T \cdot \ln(2)$ energy.-The $k \cdot T \cdot \ln(2)$ energy does not apply to reversible computation.

Reversible computation is incapable of doing the computations that I want it to do better than conventional computation.-Only someone unfamilar with Bennett's pebble game would have this thought.

The energy efficiency of Bitcoin mining will soon approach Landauer's limit. This will produced decentralized consensus since nearly all of the resources used for mining Bitcoin will be energy costs for the mining hardware.-SHA-256 mining can be made completely reversible with a manageable computational overhead. Therefore, by Landauer's limit, the energy efficiency of Bitcoin mining is zero. Furthermore, if you believe in quantum supremacy, you will realize that Grover's algorithm can be used to mine Bitcoin without being subject to Landauer's limit either. In fact, it is impossible to construct a mining problem that cannot be solved by sufficiently advanced reversible computers using less energy than by using conventional computers. Furthermore, if one were to construct a cryptocurrency mining problem which is difficult to mine on a reversible computer, then such a mining problem would require one to compute many non-invertible functions. However, even in this scenario, one could still mine such a cryptocurrency reversibly by uncomputing all of the garbage data generated after every every solution attempt.

Other cryptocurrency mining problems are useful too or are more useful than RCO-POW problems.-Many otherwise educated people have promoted certain computational problems as useful POW problems. The issue with these POW problems is that they are either unsuitable as POW problems or they have a very little practical value, and there is no way to convince a population that it is worthwhile to spend billions of dollars to work on these POW problems. These problems include computing irreversible cellular automata, stumbling upon interesting prime numbers, performing elliptic curve calculations, unknotting knots, or some other obscure problem which would not increase the wealth of society. Since all computers will eventually become reversible computers in the future, it is exceedingly difficult to produce a useful POW problem which is nearly as useful as an RCO-POW problem.

SHA-256 is an RCO-POW problem.-I have to admit that SHA-256 is by accident more amenable to reversible computation than most algorithms. The Zammazazzer foundation will take advantage of the amenability of symmetric computation to reversibility by researching and developing reversible symmetric cryptosystems. The RCO-POW problem Hashspin was designed with the reversibility friendliness of symmetric cryptography in mind, and Hashspin was designed so that it would be much easier to manufacture a free market reversible computer for mining Zammazazzer than it would be to construct a free market reversible computer to calculate SHA-256 or mine Bitcoin.

RCO-POW problems will not ease or accelerate the development of reversible computation. We should therefore not use RCO-POW problems.-In order to justify the use of a mining algorithm such as SHA-256 over an RCO-POW problem, one needs to be 100% sure that RCO-POW problems will not effectively accelerate the development of the reversible computer.

An RCO-POW problem will produce centralized mining.-Anyone who claims that RCO-POW problems will have more centralized mining than POW problems such as SHA-256 is usually just speculating. RCO-POW problems will help make cryptocurrencies more popular and obtain a higher market cap which would then produce more decentralized mining.

7.6. **Final remarks.** Many people favor cryptocurrencies because they do not want centralized organizations to unfairly control the currency. However, these same people should not tolerate nor support coins distributed in an unfair or careless manner, and so far all other cryptocurrencies have been distributed unfairly through an ICO or through a POW problem which rewards the entity which causes the most environmental destruction. Instead, people should only invest in coins that are distributed through useful POW problems. Not only is it financially irresponsible to invest in a coin distributed unfairly, but it is also ecologically irresponsible to invest in a cryptocurrency which wastes energy solving a useless problem.

There is no reason to use a useless POW problem instead of a useful POW problem, and it is unwise to invest in an unfairly distributed POS cryptocurrency. The only good cryptocurrency is a cryptocurrency that employs an RCO-POW problem or some equally useful mining problem. Cryptocurrencies such as Bitcoin or Ethereum employ a useless POW problem instead of an RCO-POW problem or other useful POW problem simply because nearly everyone in the cryptocurrency community is completely uneducated about reversible computation.

References

1. An Analysis of Bennett's Pebble Game. LANL report LAUR–95–2258. E. Knill, knill@lanl.gov. May 1995.
2. C.H. Bennett. Logical reversibility of computation. IBM J. Res. Develop., 17:525-532, 1973.
3. C.H. Bennett. Time-space trade-offs for reversible computation. SIAM J. Comput., 18(1989), 766–776.
4. The asymptotic complexity of matrix reduction over finite fields. Demetres Christofides. 2014
5. Molecular Mechanical Computing Systems. Ralph C. Merkle. Robert A. Freitas Jr. Tad Hogg. Thomas E. Moore. Matthew S. Moses. James Ryley. IMM Report No. 46. April 2016. http://www.imm.org/Reports/rep046.pdf
6. https://topps.diku.dk/pirc/?id=janusP
7. Bitcoin: A Peer-to-Peer Electronic Cash System. Satoshi Nakamoto. 2008
8. Progress with Physically and Logically Reversible Superconducting Digital Circuits. Jie Ren and Vasili K. Semenov. 2011

E-mail address: jvanname@mail.usf.edu