The Unexplored Magic of Reverse Computing

Introduction

Reverse Computing (RC) is an elusive and almost alien computing concept since it defines a fully reversible, lose-less, zero energy consumption classical and quantum computing device. RC has been hailed as a next generation platform that circumvents the limitation of Moore’s law, enables low cost quantum computing and spawns a multitude of innovative new technologies and applications. RC as per its terminology, is facilitated by implementation of specialised Reversible Gates (RG) that can persevere states of input and output for an undefined period of time. RG maintains similar thermodynamic and information entropy or simply put, does not loose information or generate heat.

Heat generation is a major stumbling block to enhance processor speeds as heat melts silicon. As per Moore's law, a microprocessors transistor density doubles every 24 months achieved by packing more transistors in a finite amount of space. The higher the transistor density, the higher the heat produced. The density reaches a critical point where packing more transistors does not yield a higher processing throughput. This bottleneck is the limitation of Moore's law which, in theory, can be resolved by RG as they do not generate heat during processing. Hence, RG and RC are all slated to be a silver bullet to resolve processor manufacturing problems, enable high speed quantum computing and generate major breakthroughs in computing and sciences. However, we will empirically confirm that both RG and RC are grossly misunderstood, misinterpreted and incorrectly applied to classical and quantum computing concepts. In a series of upcoming papers and patents (due for release by mid 2022) we will clarify and argue our position by updating core RC terminology. Furthermore, we will also list updated operability guidelines via real world examples of RC applications.

Background

Origins of RG can be traced back to 1950 when an IBM researcher Rolf Landauer established a

relationship between amount of heat generated and bit processing in a logical gate. His studies confirmed that every bit lost on output by a gate, generates a measurable amount of energy. This problem of energy dissipation is resolvable by RG implementation. In 1973, Charles Bennett of IBM explored energy neutral computers and coined the term, ‘adiabatic computers’. These computers created by RG would not generate heat and could run indefinitely once they are fully initialised. In 1970s Richard Feynman theoretically proved that RG do not have an energy minimum and that RC is attainable. In 1982, Edward Fredkin and Tommaso Toffoli of Massachusetts Institute of Technology (MIT) designed the first universal reversible gate which could generate input from an output and were deployed in quantum computers. These gates were later used by researchers at Griffith University and University of Queensland, Australia to build a photon based quantum computer. This computer was the basis of a 2,000 qubit quantum computer that operates as an adiabatic machine at absolute zero.

The rationale behind RC is that a system is designed that does not generate heat and then would be:

- Energy saving device,
- Undertake high speed tasks as processing throughput is not restricted due to heat, *and*

- Enable quantum computing applications.

While heat generation is exhibited by hardware and logical gates, software also plays an important role. Tasks such as functions, calls, loops, lookups, file management, memory functions generate high levels of heat. To resolve this issue software was ported to being RC-friendly by development of a new programming language (led by Nirvan Tyagi of MIT in 2016) called Energy Efficient Language (EEL). This programming language limits the use of high intensity modules by replacing them with low overhead coding techniques. This idea of generating software code that could be reversed was led by a study in Newcastle University by Brian Randell in 1970. The study was to create software with several algorithms for same function. If one of the algorithm would fail to give the correct result then the system would go back to the pre-defined state and try a different algorithm. Another RC-friendly popular method was developed by Tom Anderson in 1976 that led to building of an operating system called ‘recovery blocks’. This was close to fuzzy logic as it would take an input, compute and run the output for acceptance or levels of acceptability. If the acceptance level would not achieve a minimal threshold value then the system would backup to a predefined state and restart its computational efforts.

RC limitations:

In this introductory document and for the scope of this paper, the following problems point the direction as to why a re-work on the fundamental definitions of RC is required:

Energy saving for generic classical architecture: RC has been evolving over the years on the principle of achieving zero energy output and eventually to create an adiabatic all-purpose generic computing system. This is simply not accurate and should not be used in future discussions. Let us assume that we have a system with multiple gates and each gate does not emit a resultant energy. We will assume that a system exists with interconnected logical gates and they have achieved principles of zero energy and entropy. These gates are magical and can defeat the laws of physics to achieve a perfect zero sum energy dissipation. Such an adiabatic system does not assume for heat loss due to customisation in circuit design, attached peripheral devices as keyboard, mouse, video, and other chips are operating on the motherboard, sister boards and chipsets attached to internal buses. All these secondary devices generate heat as the gates cannot operate independently. Furthermore, the adiabatic system also does not account for memory leaks, Operating System (OS) functions and other

custom or generic software routines that are running on the machine. Incase custom software is installed on the adiabatic machine, it would lead to additional complexities. Customised software designed for such a machine would be stripped and will have limited functionalities. Hence, a truly adiabatic system can only be made available for a single operation and a truly generic all purpose reversible computing device is not achievable.

Energy saving for generic quantum architecture: Software on quantum computers is not developed as a set of instructions. Instead, they are managed by a set of equations that accept input parameter(s) and computation commences when the system settles into a conducive state. This means that truly universal gates have not been created as generic instructions are not applicable to quantum computing.

Let us assume that a custom classical or quantum adiabatic system has been built for 64 bit operation to decrypt a PBKDF2 Wi-Fi password (WPA/WPA2). To ensure that this system is truly adiabatic, we would need to create gates that can pre-calculate and store all possible inputs and outputs for 4 way handshake and PMKs. Without counting any other complexities of WiFi password and handshakes, the creation of all possible values of a 256 bit PMK would require 10^{77} gates.

An Intel i9 processor is assumed have 12 billion transistors or 1.2^{10}.
Apart from cramming all the gates, memory, storage and many more bells and whistles will be required to make the system truly adiabatic. This leads to conclusion that such a system will be large, energy inefficient for both classical and quantum.
For a decryption deployment on classical machine, a parallel alignment of hundreds of gates will be required along with advanced hardware design. The size and design will in itself would be not energy efficient.
For a decryption deployment on quantum machine, the system would be have to cooled to absolute zero and maintained at such level for long duration of times to achieve the result. The requirement to keep such a system at absolute zero for a long duration would never be energy efficient.
The lack of energy efficiency in both cases defeats the purpose of RC.

Conclusion:

The above failure cases are intended to serve as a starting point to convince the reader that a change in RC definitions are required. Without updated definitions we cannot unlock RC to achieve the next generation of computing and a promised technical revolution. We need to determine how to achieve a

zero energy process in RC in classical and quantum realm by:

- Calculating all possible energy costs including tertiary sources and absolute zero cooling,
- Create soft and hard processes that may be stripped to create a universal series of instructions, and
- Achieve interoperability across platforms and not be limited to cracking passwords on quantum machines.

In our next paper we will discuss updated terminologies and solutions with practical examples in an attempt to re-define RC.