Resurrecting FPGA Intrinsic Analog Evolvable Hardware

Derek Whitley[1,2], Jason Yoder[3], and Nicklas Carpenter[3]

[1]Indiana University, Bloomington, IN 47405
[2]Thinker Labs, Bloomington, IN 47404
[3]Rose-Hulman Institute of Technology, Terre Haute, IN 47803
dcwhitle@iu.edu

Abstract

In the spirit of past century evolvable hardware, we explore the application of evolutionary algorithms to field programmable gate arrays and provide an open-source platform for performing intrinsic analog evolvable hardware experiments. We target the reproduction of seminal field experiments that generated complex analog dynamics of unclocked FPGAs which were evolved through genetic manipulation of their binary circuit representation: the bitstream. Further, we demonstrate the intrinsic evolution of two non-trivial analog circuits with intriguing properties: amplitude maximization and pulse oscillation. Lastly, we explore the robustness of evolved circuits to temperature variation and across-chip circuit translation.

Introduction

In 1991, Hugo de Garis postulated that evolutionary algorithms such as genetic programming "will probably lead to electronic circuits being 'grown' in special hardware". Contextually, he was referring to a research domain known as embryological electronics (de Garis, 1991); however, only two years later, in conjunction with Tetsuya Higuchi (Higuchi et al., 1993), they conceptualized the field of *evolvable hardware* (EHW): the application of evolutionary algorithms to hardware systems during design, operation, or both. It is a technique that has demonstrated the capability of producing unique and often optimal solutions to many types of scientific and industry problems (Higuchi et al., 1999; Thompson, 1997; Haddow and Tyrrell, 2018; Miller et al., 2014). And the tool of choice was the field programmable gate array (FPGA).

FPGAs are one of the primary research tools used in EHW research because their physically reprogrammable architecture can emulate candidate circuits. When combined with an evolutionary algorithm (EA) running on a host CPU, circuits can be systematically selected from a population, loaded on the FPGA, evaluated for their performance according to a fitness function, selected for progenation, then mutated and recombined for the subsequent population, gradually improving the performance of a population of circuits.

As the field was forming, Adrian Thompson at the University of Sussex evolved a series of bitstream-evolution circuits that would canonize the evolutionary approach to circuit design, specifically for analog EHW. The first among these circuits was a completely analog millisecond oscillator (Thompson, 1995). Ultimately, this was meant to be a tool for future robotics experiments, providing a temporal bridge for signals that operate at biological timescales. Later experiments demonstrated the impressive search power provided by artificial evolution, most notably: Thompson's evolved tone-discriminator circuit (Thompson, 1997).

The End of an Era

Unfortunately for the budding research field, shortly after these cornerstone achievements, the Xilinx Corporation discontinued the electronic tool of choice for evolvable hardware: the Xilinx XC6200 series FPGA. To comprehend the severity of this loss, it is imperative to understand what a bitstream is and how it was used in evolvable hardware experiments. The bitstream of an FPGA is a binary configuration file used to define and program a circuit architecture for emulation. It is the lowest level of programmable instruction akin to assembly language, though instead of executing a series of instructions, it configures a circuit using the reprogrammable interconnects and logic resources on an FPGA. In essence, the bitstream is the genome of physically realizable circuits on an FPGA. The XC6200 series was unique in that its bitstream format was made available, ergo 1:1 relationships between configuration file entries and *in silico* resources were unambiguously documented for the end user. However, for cost and security reasons, FPGA manufacturers moved away from open bitstream documentation and began employing strongly encrypted bitstreams.

While prudent on behalf of industry concerns, the discontinuation of openly documented bitstreams temporarily halted research on intrinsic analog evolvable hardware. Without complete knowledge of a bitstream's format, EHW researchers were unable to perform analog experiments intrinsic to the FPGA. And although other forms of dynamically reconfigurable hardware have been available to practi-

tioners of EHW, such as field programmable analog arrays (FPAAs) and transistor arrays (FPTAs), none of these technologies are as thoroughly developed, widely accessible, or employed as the FPGA.

On Evolvable Hardware

The broader field of evolvable hardware is taxonomized into several subdomains and practices: digital or analog, intrinsic, extrinsic, or mixtrinsic (Stoica et al., 2000), adaptive hardware (AH) or evolvable hardware design (EHD) to name a few. For a complete overview of the field's structure, please refer to (Haddow and Tyrrell, 2018). The focus of the work presented here is formally categorized as intrinsic analog evolvable hardware design, meaning populations of circuits are evolved intrinsic to a hardware substrate and do not actively adapt to changes in their environment.

At its origin, EHW aimed to revolutionize the way electronics were designed, both digital and analog. Unlike its digital counterpart, though, analog circuit design must inherently contend with the unique semiconductor physics at each timescale relevant to its operational states, an obstacle that digital circuit design overcomes by abstracting away continuous state values in the time domain through the employment of clock signals and in the voltage domain by logic levels (Johnson and Graham, 1993). Analog EHW offered a way to simplify the creation of these complex circuits by delegating the search and design process to artificial evolution.

Further, one of the hallmark characteristics of intrinsic analog EHW is the ability for evolution to employ physical properties of the target hardware that would otherwise be abstracted away during digital operation or simulation due to the complex nature of real-world high-speed semiconductor physics. Simulations generally perform this abstraction to account for the slight error introduced during fabrication (i.e. transistor doping levels, variation in copper tracing thickness, etc.). The effects that such manufacturing imperfections have on the operation of a given circuit are non-linear and can interact in unpredictable ways. Such attempts to simulate analog circuit operation prior to embedding in physical hardware must contend with the reality gap — the abstract distance between a simulated system's behavior and a real system's behavior (Rieffel and Sayles, 2010). This is where the intrinsic approach to hardware evolution shines. Famously, Thompson's tone discriminator exploited just such physical properties of the FPGA, using only 100 logic gates of the available 24,000, to accomplish a task that was thought to be impossible under the resource constraints he placed on the experiment (Thompson, 1997).

As mentioned in the previous section, the termination of the Xilinx XC6200 series FPGA halted bitstream-level intrinsic analog EHW experiments. However, recent reverse-engineering efforts by (Wolf and Lasser, 2021) have paved a way using a different FPGA technology stack altogether. The Lattice iCE40 — an ultra-low power, economy-grade FPGA package whose fully documented bitstream was exposed in work demonstrated at the Chaos Communication Congress in Hamburg, Germany, in 2015 — now enables further research into FPGA-intrinsic analog EHW. Although the purpose of reverse-engineering the iCE40 bitstream was not motivated by EHW research, a fully documented bitstream is now available for just that.

According to the literature, retrospective justifications for the discontinued research into FPGA-intrinsic analog EHW focused on the difficulties of scaling experiments, lengthy circuit evaluation periods, and other technical minutiae rather than the discontinuation of the XC6200 (Haddow and Tyrrell, 2018). Such justifications were used to motivate more abstract methods of EHW such as the use of virtually reconfigurable circuits (Torresen, 2000), the employment of other reconfigurable arrays, and the creation of alternate experimental platforms altogether (Miller et al., 2014), all while avoiding one of the research subdomains that generated such excitement in the first place. To onlookers of EHW, it appeared that the whole field went silent (Clarke, 2012), when in fact it was blossoming in areas that didn't draw as much attention (Cancare et al., 2011). And while there is no evidence indicating a dead-end for this specific vein of research, many assumed it reached an untimely end. On the contrary, we argue the opposite; that FPGA-intrinsic analog evolvable hardware is still fertile with potential.

Methods

In classical evolvable hardware fashion, the experimental setup is as follows (see Figure 1):

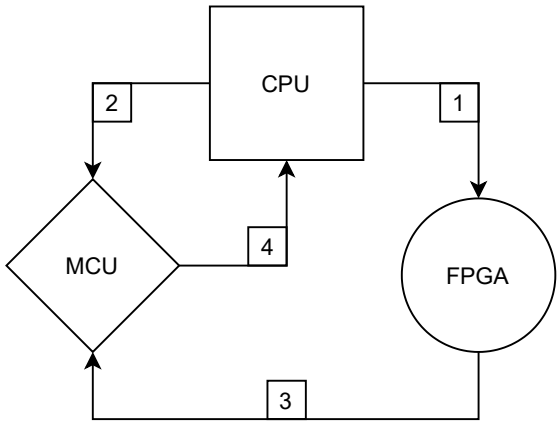

Figure 1: Experiment Setup

1. A host CPU running an evolutionary algorithm generates a population of pseudo-random bitstreams and systematically uploads each one to the FPGA for evaluation.

2. The host CPU prompts a microcontroller (MCU) to begin

capturing samples from the analog signal emitted from the FPGA during circuit operation.

3. The MCU reads the output from the FPGA and performs 12-bit analog-to-digital conversion (ADC).

4. The MCU transmits the ADC buffer from the sample set back to the host CPU where fitness is calculated by the evolutionary algorithm.

For all tasks that require an output from the FPGA, there is a high probability that a random circuit has a fitness value of zero. This is largely due to the probability that an output route is disconnected from the output pin or that connectivity between blocks is extremely sparse. To overcome this obstacle, prior to generating the initial population of bitstreams, a random search loop is executed to identify a candidate circuit that yields a non-zero fitness (further defined in the next section). Generally this search requires only sampling a few dozen circuits before generating a viable candidate with a very-low, yet positive fitness. Once found, the non-zero fitness circuit becomes the seed from which the initial random population is generated. It should also be explicitly noted that all circuits discussed in the present work operate with no clock and are fundamentally non-digital.

Concerning the Bitstream

Upload Delay Uploading a bitstream to an FPGA is a time consuming process that depends on bitstream complexity, on-chip resources, SRAM bandwidth, and whether or not the FPGA is outfitted for dynamic reconfiguration among other factors. Similar to the experiments performed by Thompson in the mid-90s, the FPGAs used here have an average upload speed of approximately 3.5 seconds. This time delay is by far the most limiting obstacle when performing experiments, making even the simplest questions a painstaking endeavor to ask. This obstacle is further exacerbated by the inability to parallelize across multiple identical FPGAs — a problem also discussed in subsequent sections.

Combinatorial Complexity The complete, unconstrained bitstream of an iCE40 FPGA configurable logic block (CLB) is 864 bits long. When multiplied by the number of CLBs included in an evolutionary run (96 CLBs in the work performed here) the total genotypic length yields a combinatorial space far too large to search in reasonable time, especially when combined with the lengthy upload and evaluation periods. To reduce the combinatorial complexity of the search space, we constrained the bitstream in three important ways:

1. Reduced the signal routing capabilities by excluding all spanning wires and allowing connectivity only to the 8 nearest neighbors (Moore's neighborhood).

2. Limited the number of configurable inputs to allow incoming signals from a maximum of two neighboring CLBs at a time.

3. Restricted the number of potential output operations to a single Boolean logic function (e.g. no carry-in or cascade chains).

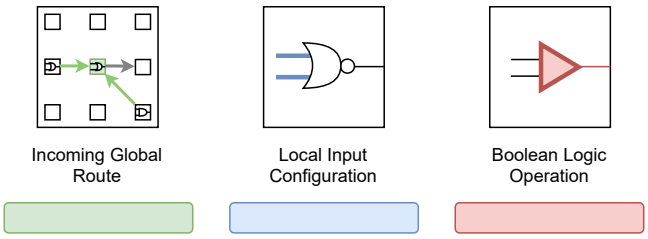

Figure 2: Phenotype Map

The above constraints provide the evolutionary algorithm access to three basic circuit-building capabilities: selection of incoming signals from neighboring CLBs, configuration of incoming signals into the CLB, and the Boolean logic operation to perform.

Relevant Technological Differences

Although the high-level concepts of operation for FPGA-intrinsic EHW remain the same, certain technological concessions were made for the work performed here due to fundamental differences between the discontinued Xilinx XC6216 and the Lattice iCE40HX1K. The first and most important difference between these devices is that the routing capabilities of modern FPGAs are sufficiently more complex than those thirty years ago; as such, we have substituted the simplistic north, east, west, south (NEWS) neighborhood routing capability with a Moore's neighborhood routing schema (all 8 surrounding neighbors). The second difference is that, as a result of modern routing complexity, CLB pass-through routing necessitates complicated pairwise relationships within the bitstream to pass signals between neighbors when not performing a logical operation; because of this, we have clamped pass-through signals off. The third difference is that the density of logic resources is drastically larger in modern FPGAs, such that a single CLB on the Lattice board has 8 logic elements in comparison to the single logic element found in an XC6216 CLB. Though we have disabled all but one logic element per CLB, the additional 'passive' resources may help or hinder evolution during search. The last difference is the number of CLBs accessible to evolution — Thompson's experiments used only a small portion of the total FPGA resources — a 10x10 array of CLBs at the farthest top-left location of the XC6216 — whereas the largest contiguous block of CLBs on the modern iCE40HX1K is 6x16 (96 CLBs) and occupies the entire central column of the FPGA. Though the total number of

CLBs is fairly close, the varied dimensions of the matrix are substantially disparate and may lead to potential geometric constraints during the course of evolution.

Establishing a Baseline

Early research into FPGA-intrinsic EHW showed not only that evolution may employ device-specific imperfections, but also that the coupling between device and environment during evolution was critical for the device's operation following evolution. This was demonstrated in two crucial ways: first, that translating a circuit that was evolved in one location of the FPGA to another location of the FPGA exhibits a degradation in performance; and second, that features of the external environment (temperature, humidity, proximity to FM radio stations, etc.) during circuit evolution became prerequisite for circuit operation after evolution (Thompson, 1995; Thompson et al., 1999). This coupling between device and environment inspired Adrian Thompson to create a controlled 'operational envelope' which he termed 'the Evolvatron' — an artificial conditioning environment designed to expose candidate circuits during evolution to a battery of controlled environmental pressures in an attempt to increase the robustness of the final circuit (Thompson, 1998).

To explore whether these environmental coupling phenomena are still present in new FPGAs, we devised a minimal litmus test for each type of coupling. First, to determine whether identical FPGA models perform differently under fixed environmental conditions, we instantiated a hand-crafted analog oscillator circuit and uploaded it to multiple FPGAs. The oscillator was created by networking all CLBs in the central column sequentially and fixing each CLB input pair to accept from only one location as well as fixing each logic function to an [AND] operation; doing so ensured that the output from one CLB was the input to another and that every CLB would fire in sequence. We then uploaded the bitstream representing the hand-crafted analog oscillator to 4 separate FPGAs and sampled their output. Each of the FPGAs were given time to reach standard operating temperature and ran in a typical indoor environment — room temperature (20C/68F) and 45% humidity.

FPGA #	Base Frequency
1	9.503 MHz
2	9.912 MHz
3	9.538 MHz
4	9.365 MHz

Table 1: Base Frequencies for FPGA Ring Oscillators

The base frequency for each FPGA generating the oscillator varied, though attention should be paid to the magnitude of variance. FPGAs 1 and 3 behaved most similarly, yet a remarkable 35 kHz apart, whereas FPGAs 2 and 4 differed as much as 148 kHz and 409 kHz from FPGA 1, respectively. Although this test is a coarse representation of device specific circuit behavior, it suggests that circuits evolved on device may display degraded performance in another.

Experimental Results

Each experiment was performed using a standard genetic algorithm (Holland, 1992) with tournament selection. As anticipated, the algorithm was able to sufficiently evolve a population of candidate circuits to conform to the pressures imposed by each fitness function. However, the output trace of each generation's best circuit behavior did not always yield an expected waveform.

Amplitude Maximization The first task we undertook was to select the simplest behavior we could identify for unambiguously determining that the intrinsic evolution of bitstreams works on a different platform than those used in the past. Given that the FPGA output is an analog signal, sufficiently modulating the changes in amplitude would do just that. Using the following fitness function, we evolved a population of 50 circuits over 100 generations.

$$\Psi = \frac{\sum_i^{|I|}(|x_i - x_{i+1}|)}{|I|} \quad (1)$$

This fitness function maximizes the amplitude of an output signal (Ψ) by summing the absolute differentials for all piecewise sample-pairs (x_i and x_{i+1}). The analog samples taken from the FPGA are received as an ordered set (I) of ADC-normalized, time-series voltage readings. The length of (I) is determined by the maximum buffer size for the digital sampling of the FPGA's output signal and is set within the MCU (*I=500* for the work performed here, but varies depending on sampling resolution).

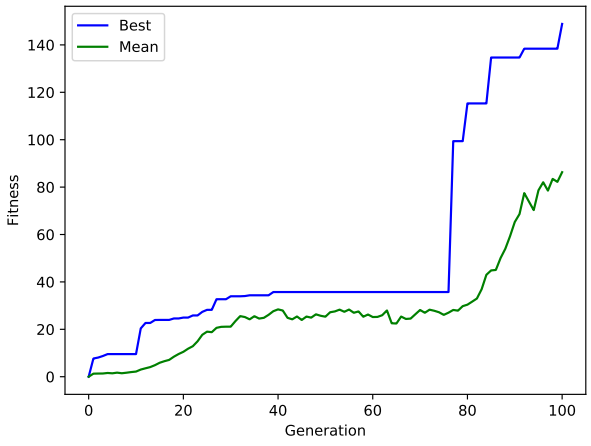

Figure 3: Evolution of Amplitude Maximization

As a result of the variable ADC buffer size, the maximum fitness is achieved when the FPGA's output matches the phase and frequency of the MCU's sampling period at maximum amplitude (3.3Vpp).

As discussed in the previous section, the initial population of candidate circuits was seeded following a random search for a circuit that yielded a non-zero fitness. The fitness function used during the initial random search is based on Equation 1.

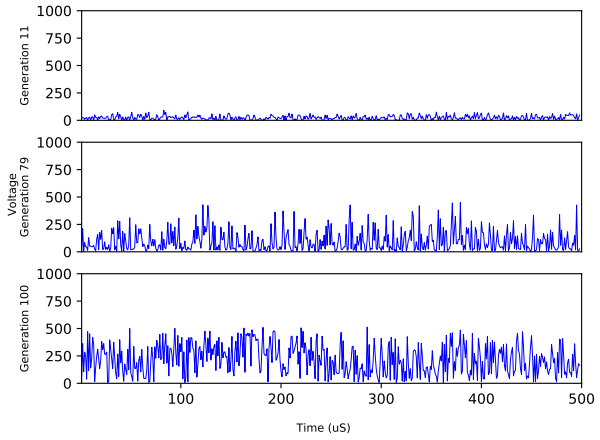

Figure 4: Generational Sample Outputs

Figure 3 depicts the evolutionary trajectory of the population of 50 circuits across 100 generations. Total (wall clock) time for this evolutionary run was 4 hours and 51 minutes. It is important to note that there are periods of dramatic increase in fitness separated by longer periods of more gradual, incremental increase; this is due to the nature of the binary connectivity between logic blocks. A sub-network of CLBs may be well developed but not attached to the critical path leading to the output until a single connection binds it to the global network, resulting in a dramatic increase or decrease in fitness. Reducing this coarseness in the fitness landscape is one of the targets for future research.

The trace depicted in Figure 4 highlights the three most dramatic leaps made during the course of evolution. Initially, the best circuit's output signal appears to exhibit a floor of low amplitude, while successive generations exhibit considerable gain. And although the maximum fitness for this particular function should display a very-high frequency triangle wave, the results presented conclusively demonstrate that the FPGA-intrinsic bitstream-evolution method does in fact work using modern hardware.

Pulse Oscillation Having verified the approach with a simple task, we turned next to one of Thompson's more advanced tasks — pulse oscillation — the goal of which was to evolve an analog oscillator whose output matched a predetermined frequency.

The experimental setup for this task required a different approach to measuring the output of the FPGA. Instead of performing ADC on an analog output signal, the MCU was programmed to measure rising edge impulses that crossed the MCU's trigger voltage (2.0 V). In this manner, pulses from the FPGA could be counted over a defined period (1 second) and transmitted back to the host CPU for fitness calculation. Because of the longer fitness evaluation period, the total (wall clock) time for this evolutionary run was 7 hours and 22 minutes — more than two hours longer than the average duration for the amplitude maximization experiments. The fitness function for pulse counting is as follows:

$$\Psi = \frac{1}{|f - n|} \tag{2}$$

Identical to the millisecond oscillator fitness function by (Thompson, 1995), this function attributes fitness by taking the inverse absolute difference of the desired frequency (f) from the measured frequency (n). Although simple in concept and implementation, this fitness function has some important caveats: most importantly, fitness assignments are exponentially graded as (n) approaches (f), leading to a disproportionate population fitness distribution.

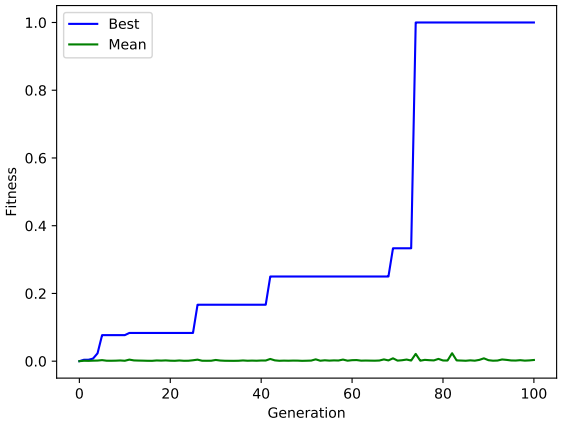

Figure 5: Evolution of Pulse Oscillator

This phenomenon is illustrated in Figure 5 where the best fitness climbs steadily for 70 generations reaching 25% of the maximum fitness, then 5 generations later suddenly jumps the remaining 75% to the maximum; where in stark contrast, the mean fitness appears to remain very low. However, the actual *mean of measured frequencies* (as opposed to *mean of fitness*) for each generation linearly approaches the target frequency when plotted in log-scale, as is shown in Figure 6. This indicates that the mean of the population is indeed evolving towards the target.

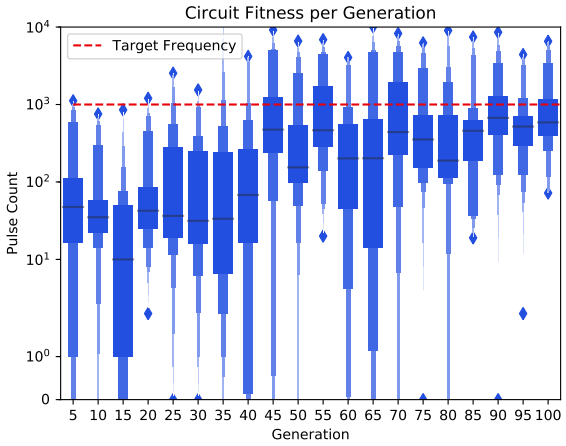

Figure 6: Circuit Population Performance

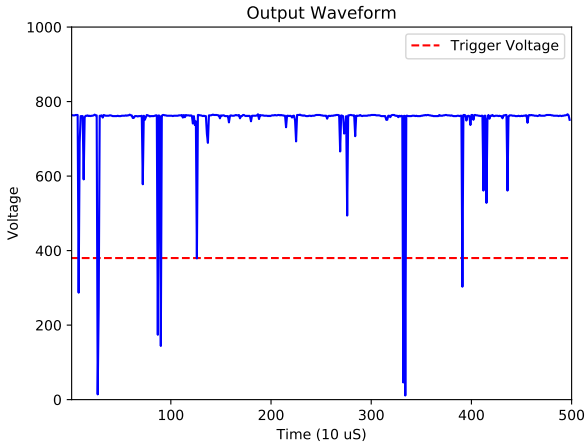

Figure 7: Pulse Oscillator Circuit Output Trace

When inspecting the behavior of the best circuit in the final generation using an oscilloscope, the output waveform and frequency displayed unusual characteristics as depicted in Figure 7. First, the circuit outputs a consistently high voltage which is aperiodically perturbed by a dip below the MCU's trigger voltage. As the MCU is programmed to detect rising edges, a pulse is only counted when the voltage returns back to the top. In essence, the waveform is upside-down yet still completes the task; though in subsequent evolutionary runs, the waveform would appear upside-up as was described in (Thompson and Rijsbergen, 2001). Second, the measured frequency has unstable periodicity. When sampling the output over time, the measured frequency would exceed or fall short of the target frequency by as much as 100 Hz.

There are several possible explanations for the observed instability. One possible explanation worth exploring is the relationship between the desired frequency and the FPGA's fabrication size (process-node) — lower-frequency oscillators may be more difficult to evolve on higher-density chips than on lower-density ones and vice versa. Another explanation may be that a single second evaluation period is not enough to enforce stability of the period. And finally, one explanation may be that the non-linear fitness assignment is the cause for such unstable final performance. Each question is a target for future research.

Concerning Robustness Having demonstrated the capability of conducting FPGA-intrinsic analog EHW, we turned finally to the question of robustness. How do evolved circuits perform under varying conditions, such as: uploading an evolved bitstream to a new FPGA of the same model, or operating at higher temperatures, or operating at lower temperatures?

We turned our attention first toward across-chip translation: how does a circuit evolved on one FPGA perform when uploaded to another of identical make and model? Using the same approach described in the previous section where a hand-designed analog oscillator bitstream was uploaded to multiple FPGAs, we likewise uploaded the best evolved pulse oscillator bitstream to multiple FPGAs. Although we previously determined that the best evolved pulse oscillator circuit displayed output signal variance during operation, we were interested in whether or not the circuit would operate at all after being uploaded to a different FPGA than the one it was evolved on.

Just as Thompson concluded (Thompson, 1998), the conditions under which an analog solution is intrinsically evolved play a critical role in the continued operation of the final circuit. Upon upload, we measured each FPGA's output using an oscilloscope. For 2 of the different FPGAs, the circuit was dead on arrival, yielding no measurable output. Yet the 3rd FPGA exhibited a distinctly different output — one with higher gain and a completely different frequency (30 kHz $\pm$ 5 kHz). When evaluating the effect of temperature on its operation, applying heat (145F/62C) to the original FPGA on which the pulse oscillator circuit was evolved resulted in an increase in frequency by several orders of magnitude until it dropped to 0 Volts. When the FPGA cooled back to standard operating temperature, the output would resume its typical, evolved behavior. Similarly, when the same FPGA was placed under refrigeration (45F/7C), its output was delayed for several minutes until it approached standard operating temperature, then would produce the evolved output behavior.

These observed environmental coupling phenomena are minimally sufficient for confirming the post-evolution behavior as previously described. However, more work is needed to establish a theoretical understanding of how and why these phenomena are produced.

Future Work

The methods demonstrated in these experiments present a clear path forward not only for addressing some of the unanswered questions surrounding analog evolvable hardware but also for questions that have yet to be asked. What is the relationship between FPGA fabrication size and signal frequency to evolution? How many CLBs are required to generate a solution of arbitrary complexity? Can the fitness landscape be smoothed? Does smoothing of the fitness landscape yield more stable circuits?

Currently, the authors are in the process of recreating the evolution of Thompson's tone discriminator, a task that requires an exceptionally longer amount of time per experiment in comparison to the tasks completed here.

A particular area of interest we intend to investigate is the application of FPGA-intrinsic EHW to the domain of reservoir computing. The conceptual parallels between reservoir computing and analog EHW suggest a unique and interdisciplinary vein of research. Conceptually speaking, a reservoir is a sparsely connected network of input-driven, time-dependent activation functions — a description not too far from the operation of analog EHW platforms.

A final goal for our future efforts is to assist in establishing an underlying theory for EHW as called for by (Haddow and Tyrrell, 2018) as the field lacks critical theory of the application of evolutionary techniques to analog circuit design.

Conclusion

In this work, we have demonstrated a method that until recently had not been possible for nearly two decades — since the discontinuation of the Xilinx XC6200 FPGA. We evolved populations of analog circuits intrinsic to FPGAs by the genetic manipulation of their bitstreams. We presented proof of concept results by evolving circuits that maximize the variation in amplitude of their output signal. Furthermore, we reproduced results of a millisecond oscillator demonstrated by one of EHW's most influential pioneers, Adrian Thompson. Finally, we established that the operation of the evolved circuits is tightly coupled to their environment, including the specific device on which the solution was evolved. This work constitutes a milestone for the continuation of analog evolvable hardware research. Though the field continues progress in many different directions, the vein of analog FPGA evolution has been virtually nonexistent since the turn of the century. By carving a path back into FPGA-intrinsic analog evolution, the field stands to make further progress as a whole.

The application of evolutionary techniques to analog and digital circuit design is an ongoing research track in both science and industry. In general, the extrinsic approach (simulation) dominates over the intrinsic approach due to the need for device-independent manufacturability and reproducibility. However, discarding the imperfections generated during manufacturing through abstraction is wasteful with regard to resources, especially when evolution may be able to exapt the existing resource toward a functional behavior. This device-dependent evolutionary approach to manufacturing was successfully employed by (Takahashi et al., 2006), whereby VLSI components driven by multi-clock systems that suffered from clock skewing were able to synchronize post fabrication. Device-specific imperfections are ubiquitous throughout hardware manufacturing. Researching the complex relationships these imperfections have with the underlying function of the hardware is imperative and may yield knowledge leading to better design. This is the motivation for the intrinsic approach to EHW.

Unlike the experimental platform used in the 90s (the Xilinx XC6216), the platform described here is inexpensive, widely available, and replete with resources, documentation, and examples for bootstrapping researchers. In order to spur interest, participation, and future collaboration, the authors have open-sourced each experiment and included tutorials for getting up and running swiftly. See (Whitley et al., 2021) for details as well as the associated GitHub page for source code and examples.

While the specific technology employed at the origin of evolvable hardware was rendered obsolete (and nearly taking a portion of the field with it), the experimental results detailed here aim to prevent that loss by clearing a path forward. Similarly to the work performed through the 90s, we have demonstrated that the intrinsic evolution of an FPGA bitstream can yield complex circuits capable of non-trivial analog behavior. And like most basic research that is performed, we have generated more questions than answers, though with one exception: we have established that FPGA-intrinsic analog evolvable hardware is once again possible.

References

Cancare, F., Bhandari, S., Bartolini, D. B., Carminati, M., and Santambrogio, M. D. (2011). A bird's eye view of FPGA-based evolvable hardware. *2011 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*, pages 169–175.

Clarke, P. (2012). Whatever happened to evolvable hardware? https://www.eetimes.com/whatever-happened-to-evolvable-hardware/.

de Garis, H. (1991). Genetic programming artificial nervous systems artificial embryos and embryological electronics. In *Schwefel HP., Männer R. (eds) Parallel Problem Solving from Nature. PPSN 1990. Lecture Notes in Computer Science*, volume 496, pages 117–123.

Haddow, P. C. and Tyrrell, A. M. (2018). Evolvable hardware challenges: Past, present and the path to a promising future. In *Inspired by Nature*, pages 3–37.

Higuchi, T., Iwata, M., Keymeulen, D., Sakanashi, H., Murakawa, M., Kajitani, I., Takahashi, E., Toda, K., Salami, M., Kajihara, N., and Otsu, N. (1999). Real-world applications of analog and digital evolvable hardware. *IEEE Transactions on Evolutionary Computation*, 3:220–234.

Higuchi, T., Niwa, T., Tanaka, T., Iba, H., de Garis, H., and Furuya, T. (1993). Evolving hardware with genetic learning: A first step towards building a darwin machine. In *From Animals to Animats 2*. The MIT Press.

Holland, J. H. (1992). *Adaptation in Natural and Artificial Systems*. The MIT Press.

Johnson, H. W. and Graham, M. (1993). *High-Speed Digital Design: A Handbook of Black Magic*. Prentice-Hall, Inc., Usa.

Miller, J. F., Harding, S. L., and Tufte, G. (2014). Evolution-in-materio: Evolving computation in materials. *Evolutionary Intelligence*, 7:49–67.

Rieffel, J. and Sayles, D. (2010). Evofab: A fully embodied evolutionary fabricator. In Tempesti, G., Tyrrell, A. M., and Miller, J. F., editors, *Evolvable Systems: From Biology to Hardware*, pages 372–380, Berlin, Heidelberg. Springer Berlin Heidelberg.

Stoica, A., Zubulum, R., and Keymeulen, D. (2000). Mixtrinsic evolution. In *Lecture notes in computer science, Springer, 1801:208–217*.

Takahashi, E., Kasai, Y., Murakawa, M., and Higuchi, T. (2006). *Post-Fabrication Clock-Timing Adjustment Using Genetic Algorithms*, pages 65–84. Springer US, Boston, MA.

Thompson, A. (1995). Evolving electronic robot controllers that exploit hardware resources. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 929, pages 640–656.

Thompson, A. (1997). An evolved circuit, intrinsic in silicon, entwined with physics. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 1259:390–405.

Thompson, A. (1998). On the automatic design of robust electronics through artificial evolution. pages 13–24.

Thompson, A., Layzell, P., and Zebulum, R. S. (1999). Explorations in design space: Unconventional electronics design through artificial evolution. *IEEE Transactions on Evolutionary Computation*, 3(3):167–195.

Thompson, A. and Rijsbergen, C. J. V. (2001). *Hardware Evolution: Automatic Design of Electronic Circuits in Reconfigurable Hardware by Artificial Evolution*. Springer-Verlag, Berlin, Heidelberg.

Torresen, J. (2000). Possibilities and limitations of applying evolvable hardware to real-world applications. *Field-Programmable Logic and Applications: The ...*, pages 230–239.

Whitley, D., Yoder, J., and Carpenter, N. (2021). Evolvable hardware website. http://evolvablehardware.org.

Wolf, C. and Lasser, M. (2021). Project icestorm. http://bygone.clairexen.net/icestorm/.