**American Journal of Engineering and Technology Management**
2021; 6(3): 41-46
http://www.sciencepublishinggroup.com/j/ajetm
doi: 10.11648/j.ajetm.20210603.13
ISSN: 2575-1948 (Print); ISSN: 2575-1441 (Online)

# An Analysis on Clock Speeds in Raspberry Pi Pico and Arduino Uno Microcontrollers

**Madhavan Thothadri**

Department of Electrical and Electronics Engineering, Rajalakshmi Engineering College, Chennai, India

**Email address:**
madhavanthothadri@gmail.com

**To cite this article:**
Madhavan Thothadri. An Analysis on Clock Speeds in Raspberry Pi Pico and Arduino Uno Microcontrollers. *American Journal of Engineering and Technology Management.* Vol. 6, No. 3, 2021, pp. 41-46. doi: 10.11648/j.ajetm.20210603.13

**Received**: June 8, 2021; **Accepted**: June 19, 2021; **Published**: June 25, 2021

**Abstract:** Choosing an application-centric microcontroller development board undisputedly increases the efficiency of the system. It impedes on-field failures and improves the quality of research. This paper analyses the Clock speeds of Arduino Uno and Raspberry Pi Pico microcontrollers to test their computation speeds using Mandelbrot Set, a familiar self-recurring fractal object. Arduino Uno is one of the popularly-used microcontrollers in the field of development. Pi Pico is the first and latest Microcontroller from Raspberry Pi family. Though the boards are economic, the latter tends to be very powerful. Hence these microcontrollers are chosen for analysis. The Mandelbrot Set is formed by the microcontrollers on an OLED display using Escape Time (ET) Algorithm. ET Algorithm takes a position (x, y) and recursively calculates the pixels that have to be turned on to render the Mandelbrot set on the display. Initially the boards are tested at their standard clock speeds. Further they are decelerated to under-rated levels to find the deviation in the rate of change of computation with the raise in their core frequencies. The Arduino Uno requires complex on-board hardware modifications with an intensive monitoring setup to work at overclocked frequencies. Hence this board is not tested at overdriven clock speeds. But Pi Pico effortlessly adjusts its core frequency to work at desired computation speeds using its phase-controlled loop. With this parameter, benchmarks and results, one of the two boards is regarded ideal for applications requiring cumulative calculations.

**Keywords:** Raspberry Pi Pico, Arduino Uno, Mandelbrot Set, Escape Time Algorithm, Clock Speed

## 1. Introduction

Application-based testing on microcontroller development-boards enhances the way of selecting them for stream-specific developments. Arduino Uno is a single core microcontroller with ATmega328P chip. It has an operating voltage of 5 volts with a recommended input voltage between 7 and 12 volts. Uno comes with 1KB EEPROM, 2KB SRAM, 32 KB Flash memory and has 14 digital Input-Output pins. It has a 16MHz ceramic oscillator for its functioning [7, 17]. With Atmel's low power 8-bit CMOS microcontroller, Uno has an enhanced RISC architecture. It computes 30,000 lines of code rationally. The galore tailor libraries and compatibility of primo sensors enables smooth collection of data. [1, 11, 12]. The sketch is compiled and loaded to the board using the open-source Arduino Integrated Development Environment.

From the family of Raspberry Pi Single Board Computers, Raspberry Pi Pico is the first high performance, low-cost microcontroller board built with RP2040 chip. Pico has a Dual-core cortex M0+ processor with on-chip phase-controlled loop for adjusting the core frequency. With 133MHz clock speed, Pico has a 264byte SRAM and 16byte on-chip cache. Sixteen PWM channels, Real Time Counter, 12-bit Analog to Digital Converter, thirty multi-function IOs and an on-board USB1.1 add to the features of Raspberry Pi Pico. Pi Pico runs on MicroPython, an implementation of Python 3 Programming [10].

In this paper, the core frequencies of Arduino Uno and Raspberry Pi Pico microcontrollers are overdriven and underclocked to analyze the deviation characteristics of computation time. Overdriving hardware is an accepted process of accelerating the internal clocks of a processing unit beyond the boundaries defined by the manufacturer. Though wild and unstable, it is carried out to push to the limits of the processor for testing its maximum efficiency at the expense of higher current consumption. Underclocking is a process of decelerating the clock speeds of a microcontroller for reducing its power consumption. This comparison enables effective choosing of an efficient microcontroller for applications

requiring rapid computations. Using the microcontrollers, The Mandelbrot set is generated to compare their computation speeds. The Mandelbrot set, a perpetuation of the Julia set, is one of the remarkable complex fractal objects [21]. It is defined using the Peano-Picard iterations, known as the one step-feedback machine [2, 4, 16]. The Mandelbrot set is given by a complex quadratic function: $f_c(Z) = Z^2 + c$. Every value of c that corresponds to a connected Julia Set is represented by each point in the Mandelbrot Set. Similarly, each point in Mandelbrot's complement represents the value of c that denotes the unconnected Set of Julia [15]. This set is rendered using Escape Time Algorithm, an uncomplicated depiction approach.

# 2. Related Work

Microcontrollers are widely used testing tools for analysis and research [22]. Ricke W. Clark [19] has patented an automated scrutinizer for electrical devices with microcontrollers. It is a trouble-shooter that tests the analog and digital components with electrical parameters internally, to predict single or a set of faulty components on the circuit board. With this algorithm set on one of the internal microcontrollers, the device with unsound components is separated from the factory with fault isolation procedure. Lucia Faravelli [14] has carried out numeric testing on microcontrollers using the fuzzy logic control to tune the parameters for a stable run. The parameters are adjusted to be robust during sudden collapse of the microcontroller, primely caused in the actuators and sensors. Holm, Håvard H. [8] has examined the efficacy and power efficiency of Graphics Processing Unit with Python programming. The GPU of the system is accessed using Python through OpenCL and CUDA. As a GPU Load, The Mandelbrot set at different ranges has been generated and colored with continuous coloring scale to record the execution time. With the benchmarks, the author has stated the proportionality in the performance of GPU with a proper code calibration.

S. A. Zulkifli [20] has simulated and tested the Arduino Uno microcontroller on its Pulse Width Modulation signals. Arduino has been used as a motor controller to regulate the inverter outputs for generating various output signals. The MATLAB results show the ability of Arduino to change the patterns of PWM based on the current sensor input. D A Pankov and L A Denisova [5] have automated a testing for diagnosing the faults in microcontroller using SATM (System Architecture for the Automated testing for Microcontrollers). Using this system of diagnosis, stable and intermittent failures are detected. This system uses fault injectors that shoot up pseudo faults to the test unit. Heuristic Algorithm, Data Algorithm and Mutation Algorithm based injectors are assumed as real threats to the microcontroller. The fault detection rate of the automated diagnosis system is found with the created pseudo-threats. D'Ausilio, A. [6] has tested all the configurable input-output pins of Arduino Uno microcontroller for their signal accuracies. Six tests have been carried out in total with increasing complexities in each test. An external high precision I/O board has recorded the signals from the microcontroller. The results have been analyzed with MATLAB to check if the hardware meets the expected levels of accuracies in signal generation for usage at neuropsychological experiments. Results indicate that Arduino Uno performs accurately for its cost. Further, Uno board is regarded to be reliable for controlling the TTL length and delays in single and quad-channel experiments.

# 3. Methodology

Tests on microcontroller development-boards are performed either during the time of their failure or on a regular basis while the microcontroller is powered. In this paper, the microcontrollers are not tested with a failsafe checklist, but it tests the extent of controllers for their stability at tag end and tail end clock speeds. For a benchmark, Mandelbrot set is plotted on a display using each microcontroller at its standard clock speed. Then the test is carried out to find the rate of change of execution time at a scale between 0 and $Max_{freq}$ MHz in the microcontroller. $Max_{freq}$ is the highest standard frequency given by the manufacturer. Further, a mathematical model is formulated to analyse the working of the controllers at overclocked frequencies. With the performances and results, one of the two microcontrollers is regarded ideal for low-power concurrent-computations.

### *3.1. The Mandelbrot Set*

Mandelbrot set is a self-similar mathematical pattern that is rendered in the complex plane by plotting the complex number of the quadratic equation $f_c(Z) = Z^2 + c$ where $Z_0 = 0 \; and \; Z_{k+1} = Z_k^2 + c$. It defines a set of values of c for which the function $f_c(Z)$ does not jump to infinity. This set of 'c' values is defined such that the absolute value $Z_k$ is less than or equal to 2. In the complex plane, measuring the real part of the complex number in the X axis and the imaginary part in the Y axis, the Mandelbrot set is generated. This set represents the points of completely connected and disconnected sets of Julia. Since the set is self-similar on magnification in the vicinity of the Misiurewicz points, generating the set requires sustained iterative calculations. Hence this set is chosen to be tested on the microcontrollers. The Escape Time (ET) algorithm is one of the straight-forward algorithms to generate and plot the Mandelbrot set [3, 9, 13].

This set is plotted on a 0.96inch Optical Light Emitting Diode Display with 128 × 64 pixels resolution. Escape Time Algorithm recursively calculates the position (x, y) and checks it for the orbit trap condition before addressing a pixel [18]. Hence every pixel of the display is addressed cumulatively. With 128 × 64 resolution, the set claims about 8192 iterations as a conservative estimate. To plot the Mandelbrot set with Escape Time Algorithm, the minimum values of real and imaginary parts are -2.5 and -1 respectively. Similarly, the maximum value is 1 for both real and imaginary parts of the complex number.

*Algorithm 1: Escape Time Algorithm*

*Input: Re_Min, Im_Min, Re_Max, Im_Max, Disp_Height, Disp_Width, Max-Itr*

*Output: Pixels to be lit for Mandelbrot set formation*
*Turn off all the pixels*
Compute Fact_real as $\frac{(Re\_Max - Re\_Min)}{Disp\_Width - 1}$
Compute Fact_imag as $\frac{(Im\_Max - Im\_Min)}{Disp\_Height - 1}$
*Initialize x, y and n to Zero*
While $y \leq Disp\_Height$: {
Compute Const_Img as $(Im\_Max - (y \times Fact\_imag))$
While $x \leq Disp\_Width$: {
Compute Const_Real as $(Re\_Min + (x \times Fact\_real))$
*Assign value of Const_Real to CmpxRe*
*Assign value of Const_Img to CmpxIm*
*Assign True to Conditionbrot*
While $n \leq Max\_Itr$: {
If $((CmpxRe^2 + CmpxIm^2) > 4)$ {
*Assign False to Conditionbrot*
*Break}*
Compute CmpxRe as $((2 \times CmpxRe \times CmpxIm) + Const\_Real)$
Compute CmpxIm as $((CmpxRe^2 - CmpxIm^2) + Const\_Img)\}$
If (Conditionbrot==True): {
Turn on pixel at position (x, y)}}}

### *3.2. Experimental Setup*

The Microcontrollers are initially tested separately at their standard frequencies for obtaining the benchmarks. Arduino Uno is evaluated at 16MHz frequency. The Pi Pico is clocked to work at its standard 133MHz frequency. The microcontrollers are further under clocked and are assessed on the basis of their working for every $2^n$ MHz raise in their clock speeds. For the Raspberry Pi Pico Microcontroller, its standard is approximated to $2^7$MHz (128MHz) and it is set as an additional benchmark for analysis.

#### *3.2.1. Arduino Uno*

The microcontroller is interfaced with 128 × 64 OLED Display using SSD1306 driver. Escape Time algorithm-based Mandelbrot plotter code is loaded into the board. Despite Uno being a power supply, the display is powered using an external 5V power source as at lower frequencies, the board draws too small current to power the display. Figure 1 shows the circuit diagram of Arduino-interfaced OLED display. A decoupling capacitor is added to the output of the 5 Volt regulator. Experimental setup with Arduino microcontroller to obtain the benchmark is shown in Figure 2.

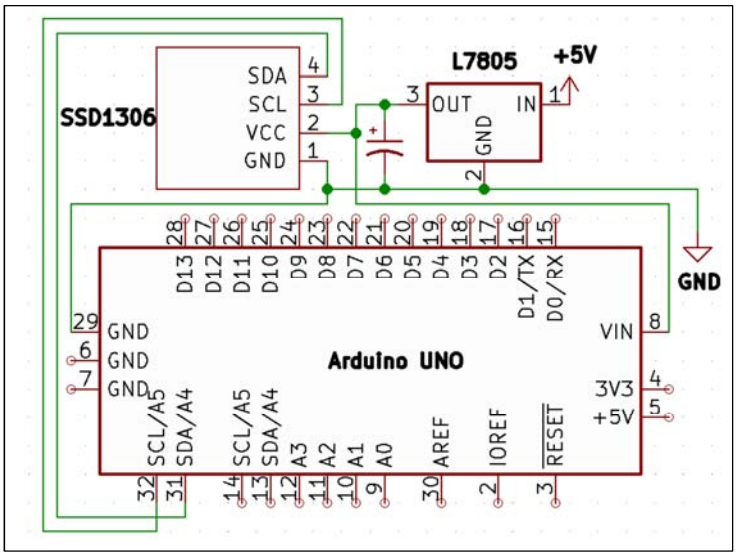


***Figure 1.*** *Circuit Diagram for interfacing OLED display with Arduino Uno.*

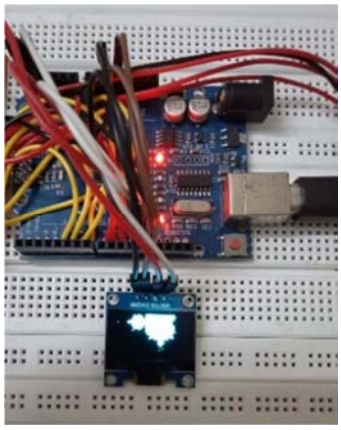

***Figure 2.*** *Experimental setup with Arduino Uno.*

#### *3.2.2. Raspberry Pi Pico*

Using MicroPython, the Mandelbrot code is loaded into Pico board. Using SSD1306 driver, the OLED display is connected with Pico board and is interfaced. The display is connected to one of the pairs of SDA and SCL pins. SCL is connected to GPIO 5 and SDA is connected to GPIO 4. As the pins of Pico board work only at 3V3, the board is not powered using the external 5V supply (using $V_{in}$). The external 5V input drives the OLED display as shown in Figure 3. Hence this board is powered using the Micro-USB port that connects to the on-board 3.3V regulator. Figure 4 shows the testing setup of the Raspberry Pi Pico microcontroller for getting the benchmark.

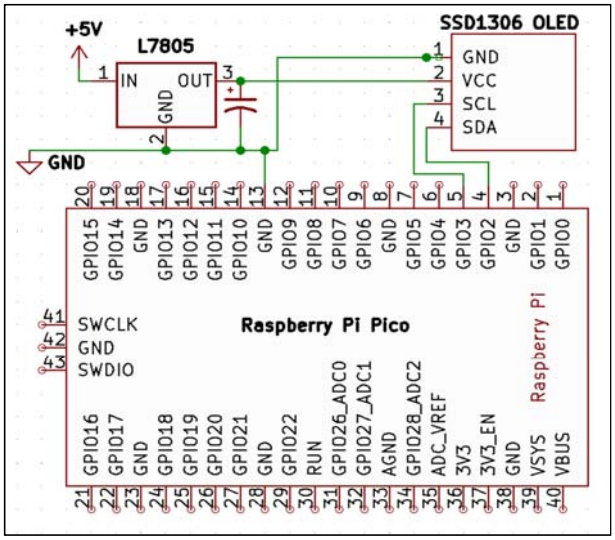


***Figure 3.*** *Circuit Diagram for interfacing OLED display with Raspberry Pi Pico.*

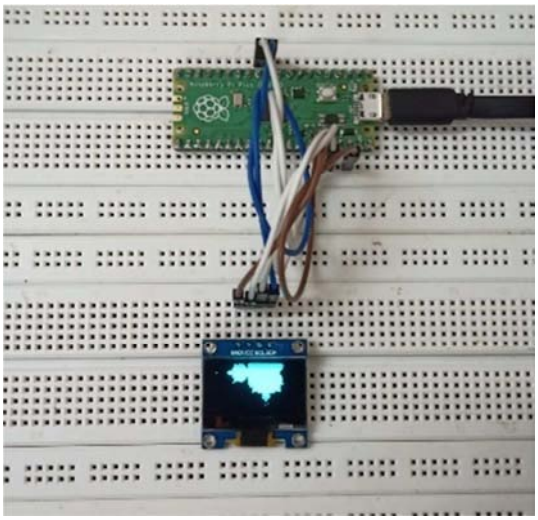

***Figure 4.*** *Experimental setup with Raspberry Pi Pico.*

### *3.3. Benchmarks*

The total time taken by a microcontroller in milli-seconds to plot the Mandelbrot set at its standard core frequency is taken as the basis. The Arduino Uno microcontroller at 16 MHz frequency takes 92927 milliseconds for plotting the Mandelbrot Set on a 0.96 inch 128 × 64 OLED display using the Escape Time Algorithm. Raspberry Pi Pico with 133 MHz frequency takes 61200 milliseconds for generating the ET algorithm-based Mandelbrot set on the display. At 128 MHz, Pico takes 61841 Milliseconds.

### *3.4. Mathematical Model*

The execution time is memoized by testing the microcontroller for every $2^n$ MHz raise in the clock speed, where n is such that $2^n \leq Max_{freq}$. The generalized form is given as,

$$2^u f = 2^{Q-u} t_{ms}, u \in [0, Q] \quad (1)$$

In the above equation,

$f$ represents Core frequency in MHz

$t_{ms}$ represents the Time taken to plot the Mandelbrot set in milliseconds.

Q denotes the maximum value of $n$ for which $2^n$ is closer or equal to $Max_{freq}$

#### *3.4.1. Arduino Uno*

From the above generalized equation, for the Arduino Uno microcontroller, Q = 4. The Arduino's architecture does not facilitate overclocking without an external Frequency Source. Hence the Uno microcontroller is analyzed only at underclocking conditions. The Underclocking is not flexibly facilitated as well. It is achieved only by adjusting the Pre-scaler. This is one of the major drawbacks of Arduino Uno microcontroller.

#### *3.4.2. Raspberry Pi Pico*

For the Pico Microcontroller, $Q \cong 7$. Since the Standard frequency of Pico is 133MHz, using the generalized equation, testing is approximated to 128 MHz frequency. This microcontroller has an edge over the Uno as it can be either overclocked or underclocked at ease. This makes it adaptable to adjust its scale between Best Performance and Best Power Efficiency. Pico is additionally analyzed at overdriven frequencies.

## 4. Results

### *4.1. Arduino Uno*

Arduino Uno deviates from the theoretical values during its execution time. Table 1 shows the calculated time and the experimental time taken (in milliseconds) by the Arduino Uno microcontroller to plot the Mandelbrot set on the display.

***Table 1.*** *Calculated time vs Observed time of computation speeds in Arduino Uno.*

| Core Frequency (MHZ) | Calculated Time (Millis) | Observed Time (Millis) |
|---|---|---|
| 16 | 92927 | 92927 |
| 8 | 185854 | 185636 |
| 4 | 371708 | 371275 |
| 2 | 743416 | 742564 |
| 1 | 1486832 | 1485010 |

Hence, using the experimental values the generalized equation is modified as,

$$2^u f = 2^{Q-u} t_{ms} + \Delta v, u \in [0, Q] \quad (2)$$

$\Delta v$ is the variation ( $Calculated\ time_{ms} - Observed\ time_{ms}$) in the Microcontroller.

This leads to a general implication that a negative value of $\Delta v$ represents a lag in the execution time. It becomes an undesired event in the selection of a particular microcontroller. It is evident from Table 1 that there is a deviation in the execution time. Though there is a deviation, as values of $\Delta v$ are positive for Uno microcontroller, it has a quicker

computation and it tends to work effectively at underclocked frequencies.

Figure 5 graphs the $\Delta v$ values of Arduino Uno microcontroller. This graph tends to become an antithesis in the aspect of computation speed for underclocked microcontrollers. From the definition of underclocking, the performance is reduced with the decrease in the core frequency. But from the graph, as the clock speed is brought down to 1 MHz, the set is plotted 1.822 seconds quicker than the calculated time.

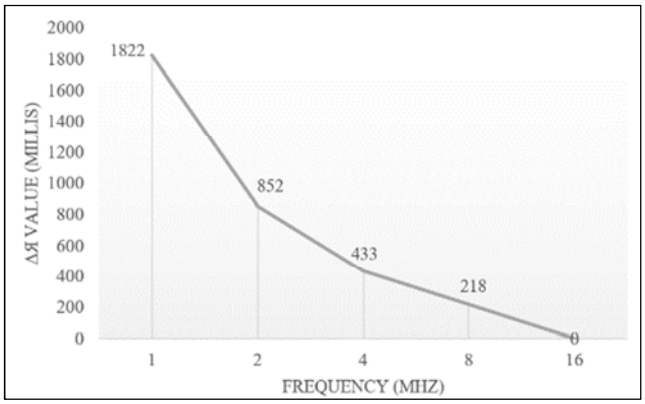


***Figure 5.*** *$\Delta v$ characteristics in Arduino Uno.*

### *4.2. Raspberry Pi Pico*

In Pi Pico microcontroller, there is a non-linearity in the execution time. Taking 128MHz as a standard, at underclocked frequencies, the execution time is quicker than expected. But the microcontroller's clock speed could not be brought down to a frequency less than 10MHz. It works only at frequencies between 10MHz and 270MHz. At underclocked frequencies, the rate of change in the computation time becomes unpredictable. There is a variation of ~10 milliseconds in the execution time when the set is plotted iteratively at any core frequency lesser than 64MHz. This variation becomes null only at 64MHz.

As the microcontroller is overclocked to 256MHz, the observed execution time lags behind the calculated value. Table 2 shows the calculated time and observed time taken by the Pi Pico microcontroller to plot the Mandelbrot set as the core frequency is raised in steps of $2^n$.

***Table 2.*** *Calculated time vs Observed time of computation speeds in Raspberry Pi Pico.*

| Core Frequency (MHZ) | Calculated Time (Millis) | Observed Time (Millis) |
|---|---|---|
| 256 | 30920.5 | 54332 |
| 128 | 61841 | 61841 |
| 64 | 123682 | 77635 |
| 32 | 115270 | 108265 |
| 16 | 216512 | 171630 |

Since there is a high variation in the execution time,

$$\Delta v = v \; as \; \Delta v \to 2000,$$

Therefore, from the above table, equation (2) is further modified as

$$2^u f = 2^{Q-u} t_{ms} + v, u \in [0, Q] \quad (3)$$

Figure 6 graphs the $v$ values of Raspberry Pi Pico. There is a vast variation in the values of $v$ as the core frequency is reduced in steps to 16MHz. The value initially increases at 64MHz, decreases to around 7005 milliseconds at 32MHz frequency and it again raises to 44882 milliseconds at 16MHz. The value at 16MHz is very much closer to that of its value at 64MHz. Significantly, the $v$ values are positive at lower clock speeds. The graph notably shows the negative value of $v$ at 256MHz core frequency. As the clock speed is accelerated above the standard frequency, the execution becomes slower than the expected time.

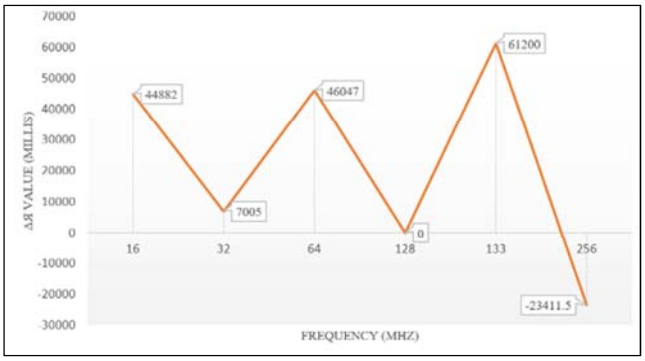


***Figure 6.*** *Variation (v) characteristics in Raspberry Pi Pico.*

## 5. Conclusion

With clock speed as a parameter to test the microcontrollers, neither of the boards tends to work as expected. An ideal board must have least deviation in its characteristics when its core frequency is adjusted. But with the market-available economic boards, it is inferred that Arduino Uno has a lesser deviation in its computation time at under-clocked frequencies. Though its maximum clock speed is 16MHz, it works efficiently as expected. But, the inability of Uno board to tailor its frequency adds to the drawback of the microcontroller. Furthermore, adjusting the pre-scaler affects the time related functions in Arduino. The functions like delay (), millis () do not work correctly when the pre-scaler is adjusted.

The powerful Pico board surprisingly has a distorted deviation in its rate of calculation. In spite of its concurrent computing, it works slower than expected at over-clocked frequencies. Additionally, the Pico board becomes unstable at frequencies lesser than 10MHz. It is evident that Pico microcontroller works faster than Uno board. However, at 16MHz, the Uno microcontroller computes quicker than the Pico board.

With the following implications, it can be concluded that Arduino has a good stability for developments that require low speed computations. Despite having large deviations in its characteristics, due to the ability of trouble-free adjusting of the core frequency, its dual core and its low current

consumption, the Raspberry Pi Pico Microcontroller is a fresh and good candidate for developments involving cumulative calculations at desired levels of speed. As a future work the tests can further be extended to assess the zooming factor of the Mandelbrot set for analyzing the float precision in microcontrollers.

## References


[1] Akram Syed Ali, Zachary Zanzinger, Deion Debose, Brent Stephens. (2016). Open-Source Building Science Sensors (OSBSS): A low-cost Arduino-based platform for long-term indoor environmental data Collection. Elsevier Building and Environment Volume 100, 114-126.

[2] Berinde, V. (2004). Picard iteration converges faster than Mann iteration for a class of quasi-contractive operators. Fixed Point Theory Appl 2004, 716359.

[3] Burns, A. (2002). Plotting the escape: an animation of parabolic bifurcations in the Mandelbrot set. Mathematics Magazine, 75 (2). 104-116.

[4] C. Zou, A. A. Shahid, A. Tassaddiq, A. Khan and M. Ahmad. (2020). Mandelbrot Sets and Julia Sets in Picard-Mann Orbit. In IEEE Access, vol. 8, pp. 64411-644.

[5] D A Pankov and L A Denisova. (2019). Automated testing and fault diagnosis of the microcontroller system. IOP Conf. Series: Materials Science and Engineering 537. 022072.

[6] D'Ausilio, A. (2012). Arduino: A low-cost multipurpose lab equipment. Behav Res 44, 305–313.

[7] Hamid Hussain Hadwan, Y. P. Reddy. (2016). Smart Home Control by using Raspberry Pi & Arduino UNO. International Journal of Advanced Research in Computer and Communication Engineering (IJARCCE) Vol. 5, Issue 4.

[8] Holm, Håvard H.; Brodtkorb, André R.; Sætra, Martin L. (2020). GPU Computing with Python: Performance, Energy Efficiency and Usability. Computation 8. 1-4.

[9] http://warp.povusers.org/Mandelbrot/ - 20th April 2021.

[10] https://datasheets.raspberrypi.org/pico/pico-datasheet.pdf - 28th April 2021.

[11] Kuldeep Singh Kaswan, Santar Pal Singh, Shrddha Sagar. (2020). Role Of Arduino in Real World Applications. International Journal of Scientific & Technology Research (IJSTR) Volume 9, Issue 01.

[12] Kumuda B, Promod Bariker, R Chandrashekhar, Rakesh kumar S, Ravishankar P. (2020). Autofarming using Arduino. International Research Journal of Engineering and Technology (IRJET) Volume: 07 Issue: 06.

[13] LIU Shuai, CHE Xiangjiu, WANG Zhengxuan. (2011). Improvement of Escape Time Algorithm by No Escape-Point. JOURNAL OF COMPUTERS, VOL. 6, NO. 8.

[14] Lucia Faravelli, Roberto Rossi, Guido Torelli. (2003). Numerical testing of a programmable microcontroller with fuzzy and adaptive features. Simulation Modelling Practice and Theory 11. 421–431.

[15] Mamta Rani, Vinod Kumar. (2004). Superior Mandelbrot Set. Journal of Korea Society of Mathematical Education Series D. Research in Mathematical Education Vol 8, No. 4. 279-291.

[16] Mandelbrot, Benoit B. (1982). The fractal geometry of nature. Rev. ed. of "Fractals", 1977. (English) Zbl 0504.28001 San Francisco: W. H. Freeman and Company.

[17] Mr. Jagtap Soham, Mr. Nikam Pramod, Mr. Ghatke Charudatta, Mr. Jadhav Jivan. (2019). Home Automation using Arduino and IoT. International Research Journal of Engineering and Technology (IRJET). Volume: 06 Issue: 03.

[18] P. D. Sisson. (2007). Fractal art using variations on escape time algorithms in the complex plane. Journal of Mathematics and the art, Volume – 1, Issue – 1.

[19] Ricke W. Clark, Irvine; Sean O'Leary, Aliso Viejo; David M. Shaheen, Corona; Earl C. Cox, La Crescenta. (2000). Self-Test and Status Reporting System for Microcontroller-Controlled Devices. (Patent No.: 6,104,304). United States Patent.

[20] S. A. Zulkifli, M. N. Hussin and A. S. Saad. (2014). MATLAB-Arduino as a low cost microcontroller for 3 phase inverter. 2014 IEEE Student Conference on Research and Development, pp. 1-5, doi: 10.1109/SCORED.2014.7072980.

[21] S. L. Singh, S. N. Mishra, W. Sinkala. (2012). A new iterative approach to fractal models. Communications in Nonlinear Science and Numerical Simulation. Volume 17, Issue 2, 521-529.

[22] Tsekoura, Ioanna, Rebel, Gregor, Berekovic, Mladen, Glösekötter, Peter. (2014). An evaluation of energy efficient microcontrollers. 9th International Symposium on Reconfigurable and Communication-Centric Systems-on-Chip, ReCoSoC 2014.