IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue 4, April 2022

DOI: 10.17148/IJARCCE.2022.114145

THE PHYSICAL LIMITS OF COMPUTING

Vidhi Divekar[1], Sheetal Wadhai[2]

Department of Computer Engineering, Universal College of Engineering, Pune

Abstraction: Over the last > half-century, computational performance per unit cost (including energy/cooling cost) has improved exponentially, thanks to transistor downscaling and accompanying energy savings. However, the physical boundaries of this scaling path are only 10 years away. MOSFET technology's inherent limitations

INTRODUCTION:

Physical computing is a creative framework or type of computing that involves the development of interactive physical systems controlled by software and hardware that can sense and respond to our surroundings without the direct intervention of people. This sort of computing is a method of learning how people connect with computers by observing how individuals express themselves physically (Tigoe, 2004). Physical computing is all around us; for example, it can be found in cars, microwaves, and washing machines.

There are several platforms available that make it simple to understand and construct physical objects without requiring a comprehensive understanding of electronics. Arduino and Raspberry Pi are two of the most well-known platforms.

During the first week of the module, we began to understand the fundamentals of the Arduino kit with a plethora of electronics. This week, we used the Arduino, a breadboard, coloured wires, resistors, and LEDs.

Arduino Leonardo

The first step was to download and install the Arduino IDE on our computers. The next step was to properly connect the Arduino to the computer using a micro-usb cable and to choose the correct board and port inside the software's options.

The first programme you write when learning a new framework, like all programming languages, is "Hello World." In this context, "Hello world" refers to programming an LED to flash at various time intervals using the delay function. However, I also added two extra LEDs to make the experiment more intriguing.

This experiment has a very easy procedure. First, we connect one wire from Arduino's 'ground' to the breadboard's negative side. Then we join two wires to two distinct Arduino pin positions, which in my case are numbers 8 and 13. We place the resistors on the breadboard and connect the two wires. All that remains is to connect the LEDs to the breadboard, one leg to the 'ground' and the other to the resistor.

Inside the Arduino program we start by declaring 4 integer variables:

- Ledpin
- Ledpin2
- Brightness
- FadeAmount

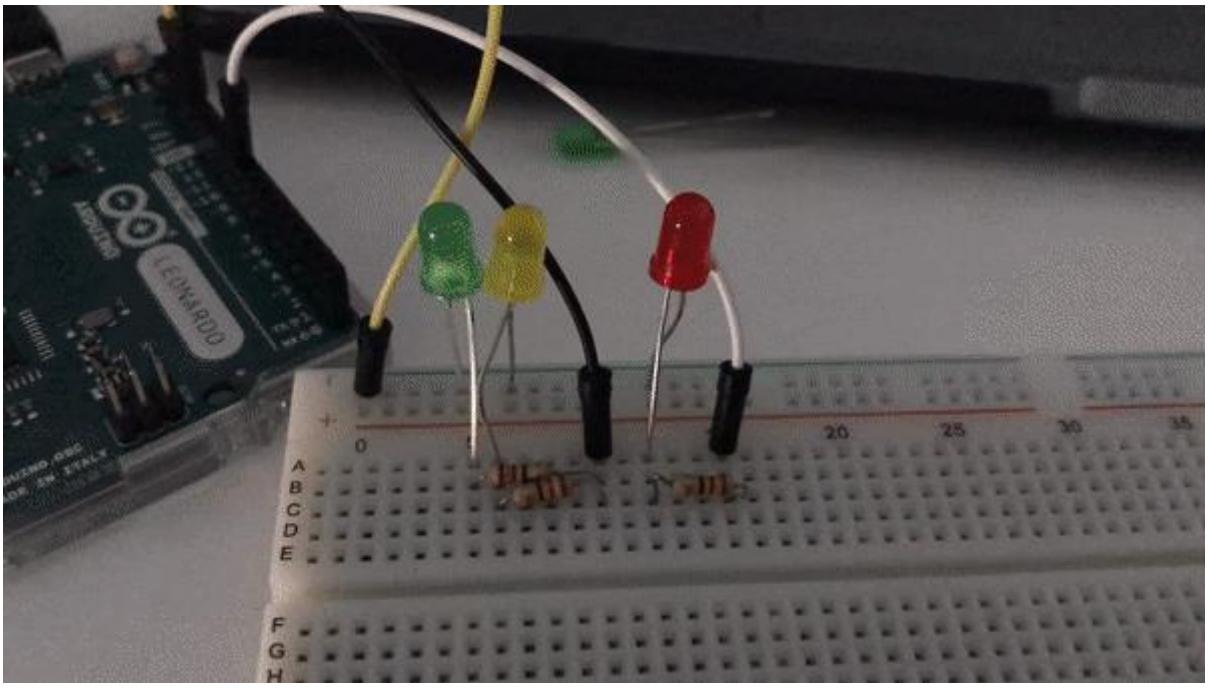
IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

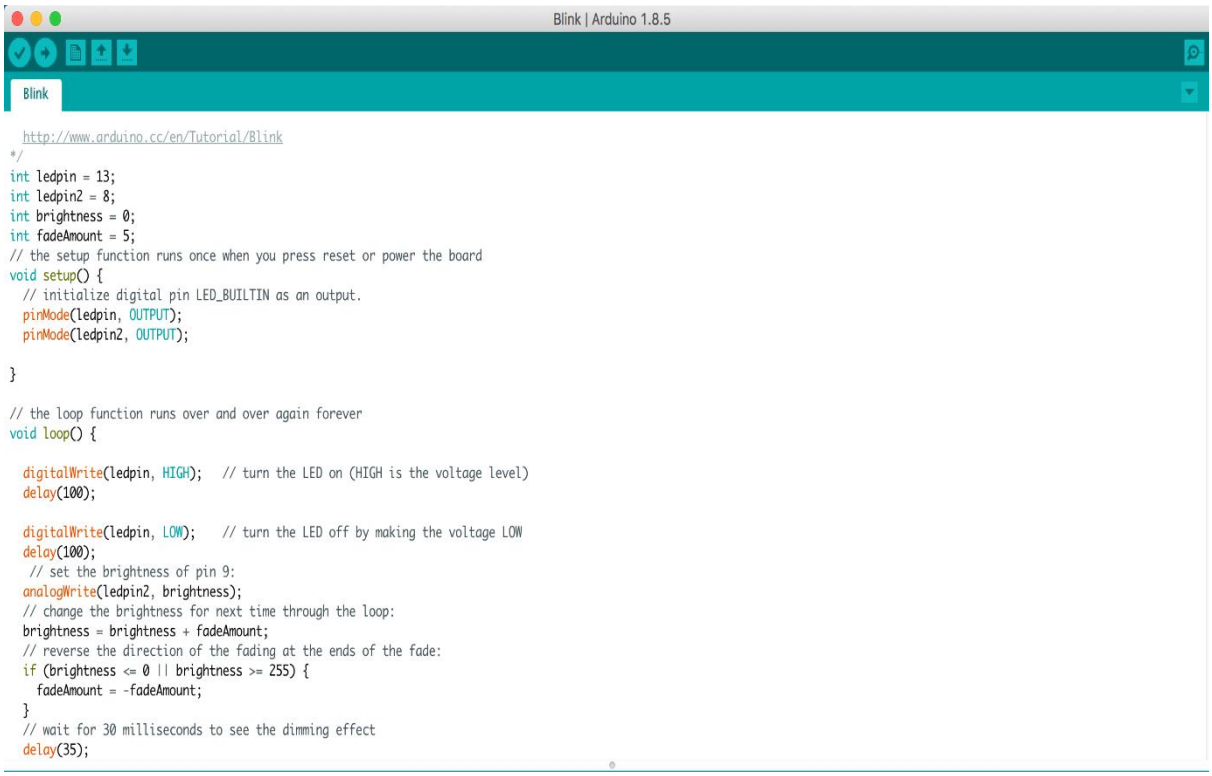
Impact Factor 7.39 ⁜ Vol. 11, Issue 4, April 2022

DOI: 10.17148/IJARCCE.2022.114145

Using the pin-Mode () function, we initialise the digital pins as outputs within the'setup' method. To turn on the LEDs, we use the digital-Write () function, which accepts two parameters: the pin number and the voltage level. In addition, I used the analog-Write () function, which takes the same parameters, but this time I modified the fade amount based on the brightness.



LEDs flashing in different time points



```
Blink | Arduino 1.8.5

Blink

  http://www.arduino.cc/en/Tutorial/Blink
*/
int ledpin = 13;
int ledpin2 = 8;
int brightness = 0;
int fadeAmount = 5;
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(ledpin, OUTPUT);
  pinMode(ledpin2, OUTPUT);

}

// the loop function runs over and over again forever
void loop() {

  digitalWrite(ledpin, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(100);

  digitalWrite(ledpin, LOW);    // turn the LED off by making the voltage LOW
  delay(100);
   // set the brightness of pin 9:
  analogWrite(ledpin2, brightness);
  // change the brightness for next time through the loop:
  brightness = brightness + fadeAmount;
  // reverse the direction of the fading at the ends of the fade:
  if (brightness <= 0 || brightness >= 255) {
    fadeAmount = -fadeAmount;
  }
  // wait for 30 milliseconds to see the dimming effect
  delay(35);
```

A variety of things influence computation limits. There are various physical and practical constraints to the amount of calculation or data storage that can be performed with a given amount of mass, volume, or energy.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

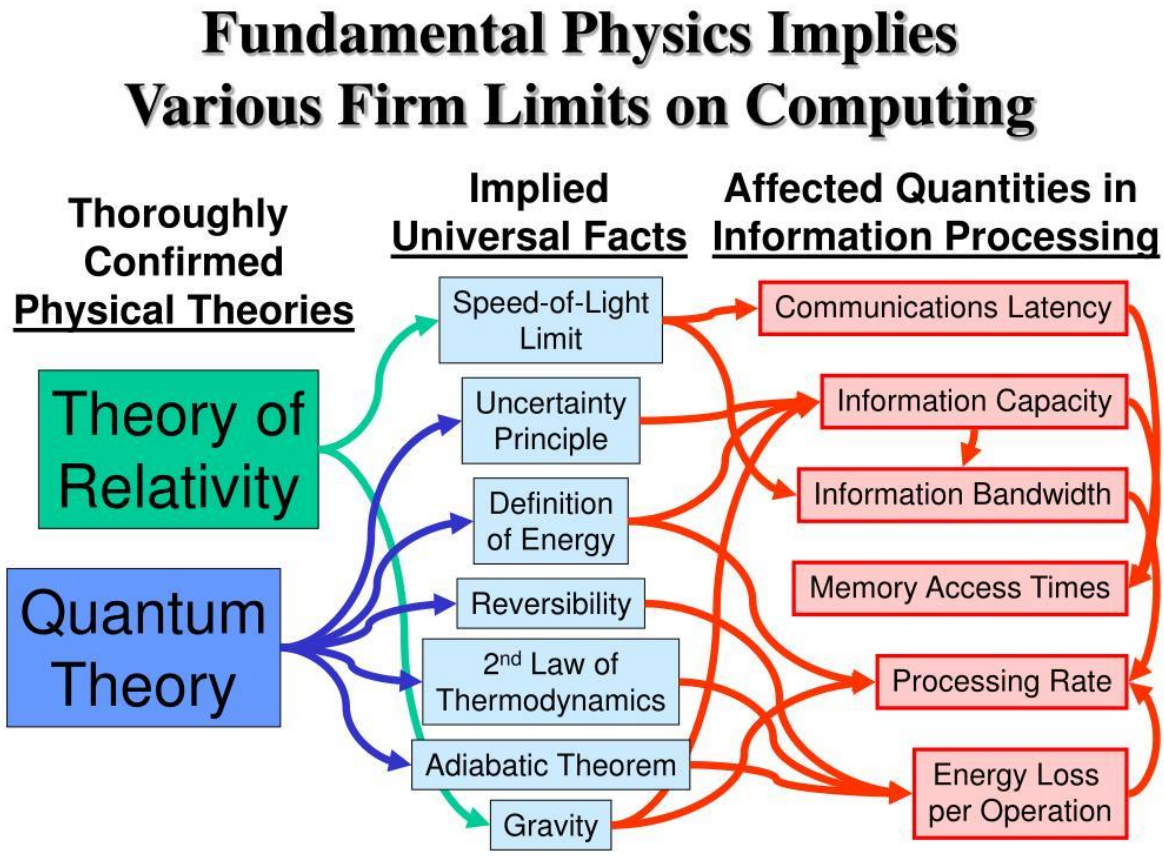
International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue 4, April 2022

DOI: 10.17148/IJARCCE.2022.114145

Computational science and engineering and computer science and engineering are inextricably linked. Scientific and engineering challenges have some of the most stringent computational power needs, which drives the development of new bit-device technologies and circuit architectures, as well as the scientific and mathematical study of better algorithms and more sophisticated computing theory. The ENIAC was inspired by the necessity for finite-difference artillery ballistics simulations during WWII, and huge calculations in every field of science and engineering inspire today's PetaFLOPSscale * supercomputers (cf. IBM's Blue Gene [1]). Meanwhile, computational processes themselves aid in the development of more efficient computing systems. Manufacturing processes, logic device physics, circuits, CPU designs, communications networks, and distributed systems all benefit from computational modelling and simulation. computing technology, resulting in ever-increasing densities of useful computational work that can be performed with a given amount of time, material, space, energy, and cost. Furthermore, long-term economic growth facilitated by scientific and industrial improvements in many domains makes larger total levels of societal computing expenditures more affordable. More affordable computing, in turn, enables entirely new applications in research, engineering, and other sectors, driving up demand even further. Computational efficiency has grown continuously and drastically since computing's beginnings, thanks in part to this positive feedback loop between increasing demand and better computer technology. Looking back over the last forty years (and the next ten or twenty), this empirical pattern is most usually described in terms of the Moore's Law [2] is well-known for describing the growing density of micro lithographed transistors in integrated semiconductor circuits.

Interestingly, despite the fact that Moore's Equation was originally expressed in terms specific to semiconductor technology, the patterns of rising processing density inherent in the law tend to remain true across numerous technologies. The history of computing technology can be traced back via discrete transistors, vacuum tubes, electromechanical relays, and gears, and amazingly, the same exponential curve can be seen extending over all of these major technical advances. Furthermore, when seen in the long term, the curve appears to be super-exponential; the frequency of doubling computing efficiency appears to rise over time ([5], pp. 20-25). Naturally, we ask how far we can realistically aspire to go This lucky tendency will lead us. Can we continue to construct ever more powerful and faster computers with our current economic resources and use them to address ever larger and more complicated scientific and engineering challenges indefinitely? What are the boundaries? Are there any restrictions? Can we hope to sustain the curve when * Peta = 10 15, FLOPS = Floating-point Operations Per Second semiconductor technology approaches its technology-specific limits by switching to some alternative technology, and then to another after that one runs out?



IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 7.39 ⁜ Vol. 11, Issue 4, April 2022

DOI: 10.17148/IJARCCE.2022.114145

Hardware limits or physical limits

Processing and memory density

- According to the Bekenstein bound, the amount of information that can be stored within a spherical volume is limited to the entropy of a black hole with the same surface area.
- Thermodynamics limits a system's data storage based on its energy, number of particles, and particle modes. It is a stronger bound than the Bekenstein bound in practise.

Processing speed

- Bremermann's limit, which is based on mass–energy vs quantum uncertainty limitations, is the highest processing speed of a self-contained system in the material world.

Communication delays

- The Margolus–Levitin theorem limits maximal processing speed perunit of energy to 6 1033 operations per second per joule. This constraint, however, can be removed if quantum memory is available. Computational algorithms that use arbitrarily little amounts of energy/time per elementary computing step can subsequently be created.

Energy supply

Landauer's principle establishes a lower theoretical limit for energy consumption: kT ln 2 consumed every irreversible state transition, where k denotes the Boltzmann constant and T denotes the computer's operating temperature. This lower bound does not apply to reversible computation. T cannot be reduced to less than 3 kelvins, the approximate temperature of cosmic microwave background radiation, without expending more energy on cooling than is saved in computing. However, on a period of 109 - 1010 years, the cosmic microwave background radiation will be dropping exponentially, allowing for 1030 times as many computations per unit of energy. Significant portions of this argument have been called into question. The Margolus–Levitin theorem limits the greatest computational speed per unit of time.

Building devices that approach physical limits

Several methods have been proposed for producing computing devices or data storage devices that approach physical and practical limits:

- A cold degenerate star may theoretically be used as a massive data storage device by carefully perturbing it to various excited states, much like an atom or quantum well. Because no naturally degenerate star can cool to this temperature for an incredibly long time, such a star would have to be created artificially. It is also feasible that nucleons on the surface of neutron stars might form complex "molecules," which some have proposed could be utilised for computing, resulting in a type of computronium based on femto-technology that is quicker and denser than computronium based on nanotechnology. Several strategies for developing computational machines or data storage devices that approach physical and practical constraints have been proposed:

- • If a suitable mechanism for extracting trapped information can be devised, a black hole might be used as a data storage or computer device. In principle, such extraction may be achievable (Stephen Hawking's suggested solution to the black hole information dilemma). This would result in storage density that is exactly equal to the Bekenstein bound. Seth Lloyd calculated the computational capabilities of a "ultimate laptop" formed by compressing a kilogramme of matter into a black hole with a radius of 1.485 1027 metres, concluding that it would only last about 1019 seconds before evaporating due to Hawking radiation, but that during this brief time it could compute at a rate of about 5 1050 operations per second, eventually performing about 1032 operations on 1016 bits (~1PB). "Interestingly, despite the fact that this hypothetical calculation is performed at ultra-high densities and speeds, the total number of bits available to be processed is not far from the number available to present computers operating in more familiar conditions," Lloyd writes.
- In The Singularity is Near, Ray Kurzweil cites Seth Lloyd's estimations that a universal-scale computer can perform 1090 operations per second. The universe's mass is calculated to be 3 1052 kg. If all matter in the universe was converted into a black hole, it would last 2.8 10139 seconds before vanishing owing to Hawking radiation. 2.8 10229 operations would be performed by such a universal-scale black hole computer during its lifespan.

Abstract limits in computer science

The computability and complexity of computing issues are frequently sought for in the field of theoretical computer science. The degree to which issues are computable is described by computability theory, but the asymptotic degree of resource consumption is described by complexity theory. As a result, computational issues are classified according to their difficulty. The arithmetic and polynomial hierarchies classify the degree to which problems are computable and computable in polynomial time, respectively. The level of the arithmetic hierarchy, for example, classifies computable,

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 7.39 ⁜ Vol. 11, Issue 4, April 2022
DOI: 10.17148/IJARCCE.2022.114145

partial functions. Furthermore, because this hierarchy is stringent, any other class in the arithmetic hierarchy classifies strictly un-computable functions.

Loose and tight limits

Many constraints in computer science are derived in terms of physical constants and abstract models of computation. While very few recognised limits actually hinder cutting-edge technology, many technical challenges cannot currently be explained by closed-form restrictions.

REFERENCES:

1. G.E. Moore, "Cramming More Components onto Integrated Circuits," Electronics, 19 Apr. 1965, pp. 114–117.
2. G.E. Moore, "An Update on Moore's Law," Intel Developer Forum Keynote Speech, San Francisco, 30 Sept. 1997, http:// developer.intel.com/pressrom/archive/speeches/gem93097.htm.
3. Semiconductor Industry Association, Int'l Technology Roadmap for Semiconductors: 2001 Edition; http://public.itrs.net.
4. D.K.K. Lee and A.J. Schofield, "Metals Without Electrons: The Physics of Exotic Quantum Fluids," Visions of the Future: Physics and Electronics, J.M.T. Thompson, ed., Cambridge Univ. Press, Cambridge, England, 2001, pp. 17–37.
5. D. Deutsch, The Fabric of Reality: The Science of Parallel Universes— And Its Implications, Penguin Books, New York, 1997.
6. R. Landauer, "Information is Physical," Physics Today, vol. 44, May 1991, pp. 23–29.