The Impact of Overclocking the CPU to the Genetic Algorithm

Ahmed Hikmat[†] and Azween Abdullah[††]
Computer and Information Sciences Department, Universiti Teknologi PETRONAS, Bandar Seri Iskandar, 31750 Tronoh, Perak, MALAYSIA

Summary

In this paper, an experiment to show the impact of overclocking the CPU to the Genetic algorithm has been conducted. For the CPU part, we have used Intel core 2 duo E6420, its original frequency is 2.13 GHz, and we have succeeded to hit 3.20 GHz which is about 50% boost. For the genetic algorithm part, we have used the Rastrigin's function, which is often used to test the genetic algorithm. The number of generation is fixed to 5000 generation. We implement the function by using MATLAB R2009a. In both cases, the CPU time is measured and plotted, and the difference has been shown.

Keywords:
CPU overclocking, genetic algorithm, Rastrigin's function, MATLAB R2009a.

1. Introduction

Many people probably do not know what overclocking is but have possibly heard the term used before. To put it in its simplest terms, overclocking is taking a computer component such as a processor and running at a specification higher than rated by the manufacturer [1]. Every part produced by companies such as Intel and AMD are rated for specific speeds. They have tested the capabilities of the part and certified it for that given speed. Of course, most parts are underrated for increased reliability. Overclocking a part simply takes advantage of the remaining potential out of a computer part that the manufacturer is unwilling to certify the part for but it is capable of [1]. The primary benefit of overclocking is additional computer performance without the increased cost. Most individuals who overclock their system either want to try and produce the fastest desktop system possible or to extend their computer power on a limited budget [1]. In some cases, individuals are able to boost their system performance 25% or more! For example, a person may buy something like an AMD 2500+ and through careful overclocking end up with a processor that runs at the equivalent processing power as a AMD 3000+, but at a greatly reduced cost [1]. There are drawbacks to overclocking a computer system. The biggest drawback to overclocking a computer part is voiding the warranty provided by the manufacturer because it is not running within its rated specification [1]. Overclocked parts that are pushed to their limits also tend to have a reduced functional lifespan or even worse, if improperly done, can be destroyed completely [1]. For that reason, all overclocking guides on the net will have a disclaimer warning individuals of these facts before stating the steps to overclocking [1]. In this experiment we will overclock an Intel processor (Intel Core 2 Duo E6420), its original frequency is 2.13GHz, we will try to hit 3.20GHz which is about 50% increase in CPU frequency, and see what is the impact of this increase of frequency to the genetic algorithm, more precisely we will apply Rastrigin's object function (which is often used to test the genetic algorithm [2]) into the genetic algorithm using (MATLAB-R2009a) [Released: 2009-03-06] and see how fast the CPU will finish 5000 generations in both cases.

1.1 Genetic Algorithm

This section is for giving a brief introduction to the genetic algorithm. The genetic algorithm is a method for solving both constrained and unconstrained optimization problems that is based on natural selection, the process that drives biological evolution [2]. The genetic algorithm repeatedly modifies a population of individual solutions. At each step, the genetic algorithm selects individuals at random from the current population to be parents and uses them to produce the children for the next generation [2]. Over successive generations, the population "evolves" toward an optimal solution [2]. The genetic algorithm can be applied to solve a variety of optimization problems that are not well suited for standard optimization algorithms, including problems in which the objective function is discontinuous, on differentiable, stochastic, or highly nonlinear [2].

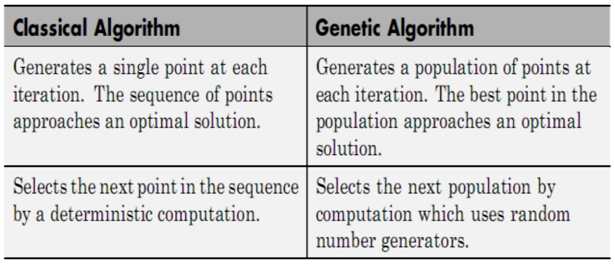
The genetic algorithm uses three main types of rules at each step to create the next generation from the current population [2]:

• Selection rules select the individuals, called parents, that contribute to the population at the next generation.

• Crossover rules combine two parents to form children for the next generation.

• Mutation rules apply random changes to individual parents to form children.

The genetic algorithm differs from a classical, derivative-based, optimization algorithm in two main ways, as summarized in the following table [2].



Classical Algorithm	Genetic Algorithm
Generates a single point at each iteration. The sequence of points approaches an optimal solution.	Generates a population of points at each iteration. The best point in the population approaches an optimal solution.
Selects the next point in the sequence by a deterministic computation.	Selects the next population by computation which uses random number generators.

Table 1[2]

2. Overclocking Procedure

There are many guides on how to overclock the CPU in the computer, we will take one of them and modify it a little bit to our experiment. For complete details refer to this website:

http://www.xbitlabs.com/articles/cpu/display/newbie-oc-guide.html

All the information stated down about the overclocking is taken the from above website.

2.1- List of useful software

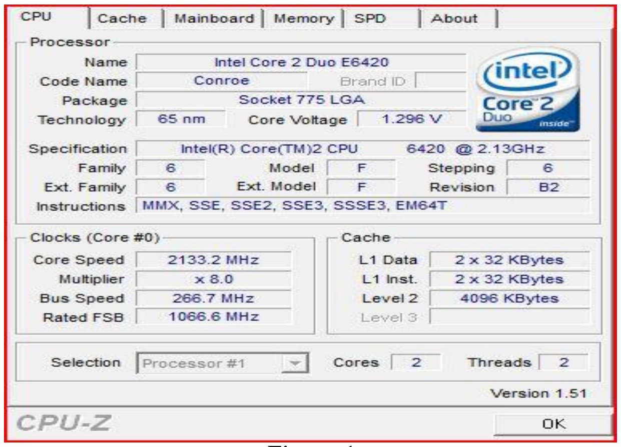
Every overclocker has a wide range of software tools that fall into a few groups:

- Informational/diagnostic
- Monitoring
- Overclocking
- Stability check
- Performance benchmark

There is no sharp separating line between the categories. Informational utilities can often benchmark performance while monitoring tools can overclock as well. Informational & diagnostic utilities can accurately identify the system configuration [3]. In their article they state many useful programs and utilities; we will use two of them the first one called **CPU-Z** which is a program that reports information about the CPU as well as mainboard and memory. The second one is **SpeedFan** which is software used to detect the temperature of computer components and detect and change computer fan speeds. It also has feature that changes the fan speed depending on the temperature of various components. Both of the programs are free of charge. According to [3] the CPU should be overclocked using BIOS options unless there are problems so other methods should be considered. Anyhow in our experiment we use the BIOS.

2.2- CPU Overclocking Basics

Overclocking means; clocking a hardware part at a frequency higher than the default one [3].There are some possible reason why overclocking is possible. It may be due to a high margin of safety provided by the manufacturer, due to marketing reasons that made the manufacturer set lower default characteristics than possible, or due to the use of faster components than necessary [3].In a PC, everything is standardized and synchronized. The standardization is obligatory for components from different makers to be able to work together at all [3]. The synchronization ensures that the components work together smoothly [3]. The frequency of the system bus (or Front Side Bus – FSB) is considered the basic frequency of the system. The rest of the buses the various devices and components are connected with usually work at lower frequencies that are generated from the FSB frequency by means of divisors [3]. The CPU frequency is currently much higher than the FSB frequency and is generated by means of a multiplier [3].In our experiment, the Intel Core 2 Duo E6420 processor works at a bus frequency of 266MHz. Its frequency multiplier is x8 and the multiplication of the two yields the resulting CPU frequency, which is 266x8=2.13GHz (see figure 1). It means that the CPU frequency can be increased by increasing the FSB frequency or the multiplier. We will try to increase the bus frequency to 400MHz so that the CPU frequency will be 3.2GHz which is about 50% boost.



Processor
Name: Intel Core 2 Duo E6420
Code Name: Conroe Brand ID
Package: Socket 775 LGA
Technology: 65 nm Core Voltage: 1.296 V
Specification: Intel(R) Core(TM)2 CPU 6420 @ 2.13GHz
Family: 6 Model: F Stepping: 6
Ext. Family: 6 Ext. Model: F Revision: B2
Instructions: MMX, SSE, SSE2, SSE3, SSSE3, EM64T

Clocks (Core #0)
Core Speed: 2133.2 MHz
Multiplier: x 8.0
Bus Speed: 266.7 MHz
Rated FSB: 1066.6 MHz

Cache
L1 Data: 2 x 32 KBytes
L1 Inst.: 2 x 32 KBytes
Level 2: 4096 KBytes
Level 3

Selection: Processor #1 Cores: 2 Threads: 2
Version 1.51

Figure 1

This is in fact all the information that needed to go into the mainboard's BIOS Setup, increase the FSB frequency and overclock the CPU, but there are some things that everyone should be aware of [3].

2.3- Getting Ready to Overclock

The first step: Check and download the latest version for the BIOS. Go to the mainboard manufacturer's website and download the latest version. Almost always the new version of BIOS has more capabilities than the old one. New BIOS versions not only correct errors, but sometimes add new parameters or expand the range of the available ones. We can read the version number of the current BIOS during the startup or sometimes there are programs to do this job for example CPU-Z [3].

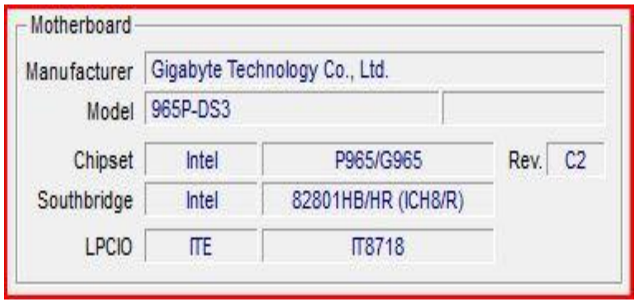
The second step: Reduce the memory frequency because overclocking and increasing the FSB frequency leads to increasing the memory frequency proportionally. And if the memory works at a high frequency by default, it may become the limit to further CPU overclocking. It's desirable to set the minimum possible memory frequency in the BIOS [3].

The third step: Set higher memory timings, at least the basic ones, for example 5-5-5-15-2T for the widespread DDR2 memory type. We should do this for the same reason as we reduced the memory frequency, i.e. to prevent the memory chips from interfering with the CPU overclocking [3].

2.4- Overclocking the CPU

Now after knowing the basic steps the rest is clear, increase the FSB frequency in the BIOS, save the settings, boot up the OS and test. keep an eye on the temperature reading [3]. That's all, now we will start the overclocking procedure. Here are some figures before and after the CPU overclocking.

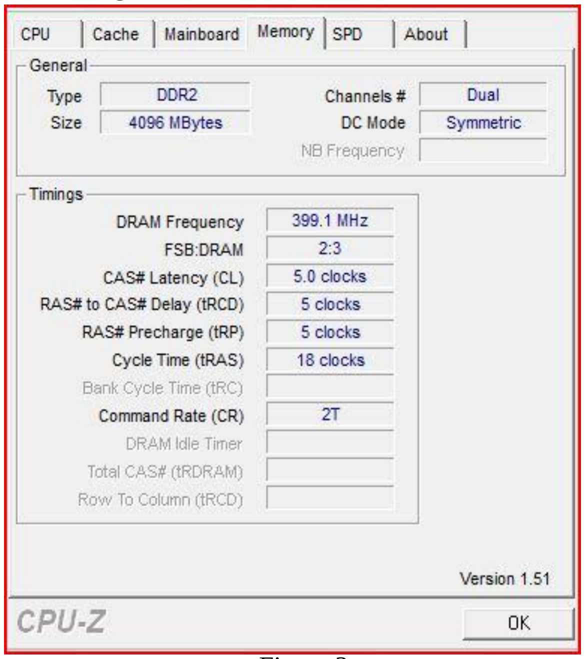
Figure 2: contains the mainboard information.



Manufacturer	Gigabyte Technology Co., Ltd.			
Model	965P-DS3			
Chipset	Intel	P965/G965	Rev.	C2
Southbridge	Intel	82801HB/HR (ICH8/R)		
LPCIO	ITE	IT8718		

Figure 2

Figure 3: contains the memory specification before the overclocking.



Type	DDR2	Channels #	Dual
Size	4096 MBytes	DC Mode	Symmetric
		NB Frequency	

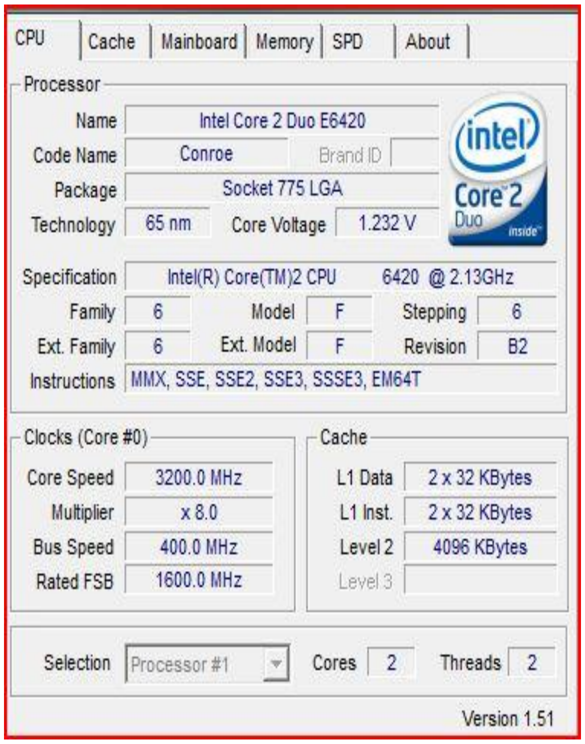
DRAM Frequency	399.1 MHz
FSB:DRAM	2:3
CAS# Latency (CL)	5.0 clocks
RAS# to CAS# Delay (tRCD)	5 clocks
RAS# Precharge (tRP)	5 clocks
Cycle Time (tRAS)	18 clocks
Bank Cycle Time (tRC)	
Command Rate (CR)	2T
DRAM Idle Timer	
Total CAS# (tRDRAM)	
Row To Column (tRCD)	

Version 1.51

Figure 3

Now we will overclock the CPU.

Figure 4, the CPU after overclocking it to 3.2GHz successfully.



Name	Intel Core 2 Duo E6420				
Code Name	Conroe	Brand ID			
Package	Socket 775 LGA				
Technology	65 nm	Core Voltage	1.232 V		
Specification	Intel(R) Core(TM)2 CPU 6420 @ 2.13GHz				
Family	6	Model	F	Stepping	6
Ext. Family	6	Ext. Model	F	Revision	B2
Instructions	MMX, SSE, SSE2, SSE3, SSSE3, EM64T				

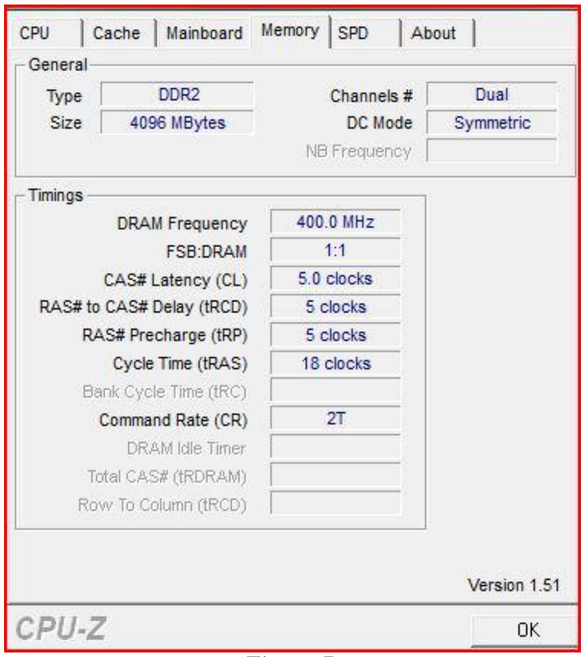
Core Speed	3200.0 MHz	L1 Data	2 x 32 KBytes
Multiplier	x 8.0	L1 Inst.	2 x 32 KBytes
Bus Speed	400.0 MHz	Level 2	4096 KBytes
Rated FSB	1600.0 MHz	Level 3	

Selection Processor #1 Cores 2 Threads 2

Version 1.51

Figure 4

Figure 5, the memory setting after the overclocking. We succeeded to make the FSB:Dram ratio to 1:1 that's will give as the best performance[3].



General

Type	DDR2	Channels #	Dual
Size	4096 MBytes	DC Mode	Symmetric
		NB Frequency	

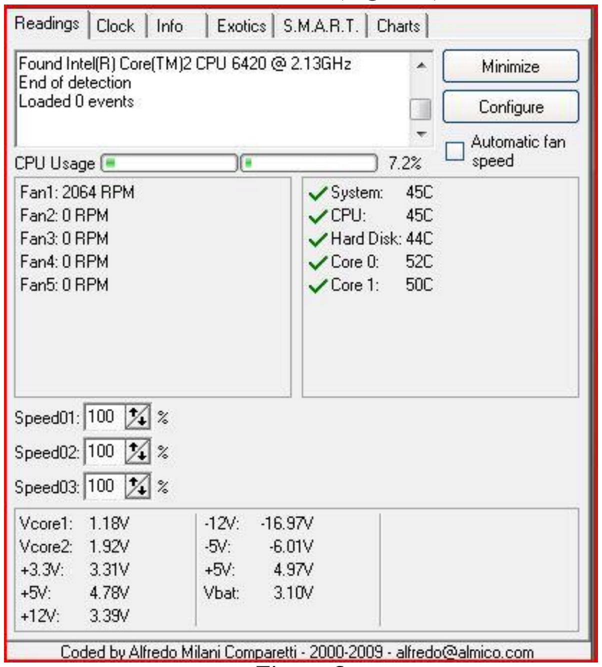
Timings

DRAM Frequency	400.0 MHz
FSB:DRAM	1:1
CAS# Latency (CL)	5.0 clocks
RAS# to CAS# Delay (tRCD)	5 clocks
RAS# Precharge (tRP)	5 clocks
Cycle Time (tRAS)	18 clocks
Bank Cycle Time (tRC)	
Command Rate (CR)	2T
DRAM Idle Timer	
Total CAS# (tRDRAM)	
Row To Column (tRCD)	

Version 1.51

Figure 5

And about the CPU temperature we use the Speedfan software and here are the results (Figure 6):



```
Found Intel(R) Core(TM)2 CPU 6420 @ 2.13GHz
End of detection
Loaded 0 events
```

CPU Usage 7.2%

Fan1: 2064 RPM	✓ System: 45C
Fan2: 0 RPM	✓ CPU: 45C
Fan3: 0 RPM	✓ Hard Disk: 44C
Fan4: 0 RPM	✓ Core 0: 52C
Fan5: 0 RPM	✓ Core 1: 50C

Speed01: 100 %
Speed02: 100 %
Speed03: 100 %

Vcore1:	1.18V	-12V:	-16.97V
Vcore2:	1.92V	-5V:	-6.01V
+3.3V:	3.31V	+5V:	4.97V
+5V:	4.78V	Vbat:	3.10V
+12V:	3.39V		

Coded by Alfredo Milani Comparetti - 2000-2009 - alfredo@almico.com

Figure 6

The CPU temp. is 45c which is considered very normal according to [3].Now let's discuss the genetic algorithm part.

3. Genetic algorithm implementation:

For the genetic algorithm we have used Matlab R2009a which is the latest version in the Matlab series. We will apply the Rastrigin's function to show the impact of overclocking the CPU to the genetic algorithm. We fix the number of generation to 5000 and then we will see how fast the CPU will finish the job in both cases.

The object function is called the Rastrigin's function we will try to find the minimum of this function which is a function that is often used to test the genetic algorithm [2].

For two independent variables, Rastrigin's function is defined as [2]:

$$Ras(x) = 20 + x_1^2 + x_2^2 - 10(\cos 2\pi x_1 + \cos 2\pi x_2).$$

Genetic Algorithm and Direct Search Toolbox software contains an M-file,rastriginsfcn.m, that computes the values of Rastrigin's function. The following figure shows a plot of Rastrigin's function [2].

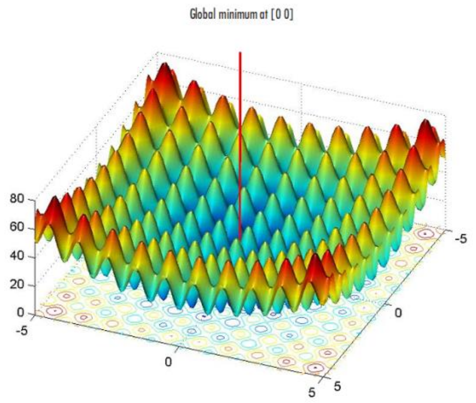


Figure 7[2]

As the plot shows, Rastrigin's function has many local minima—the "valleys" in the plot. However, the function has just one global minimum, which occurs at the point [0 0] in the x-y plane, as indicated by the vertical line in the plot, where the value of the function is 0. At any local minimum other than [0 0], the value of Rastrigin's function is greater than 0. The farther the local minimum is from the origin, the larger the value of the function is at that point. Rastrigin's function is often used to test the genetic algorithm, because its many local minima make it difficult for standard, gradient-based methods to find the global minimum [2].

We use the genetic algorithm GUI (gatool) inside the MATLAB to call for the Rastrigin function. The number of generation we use is 5000, we will plot some graphs before and after the overclocking process.

4. Results and discussion

First (figure 8) shows the number of generations vs. Fitness value, at each iteration we get a fitness value from the object function.

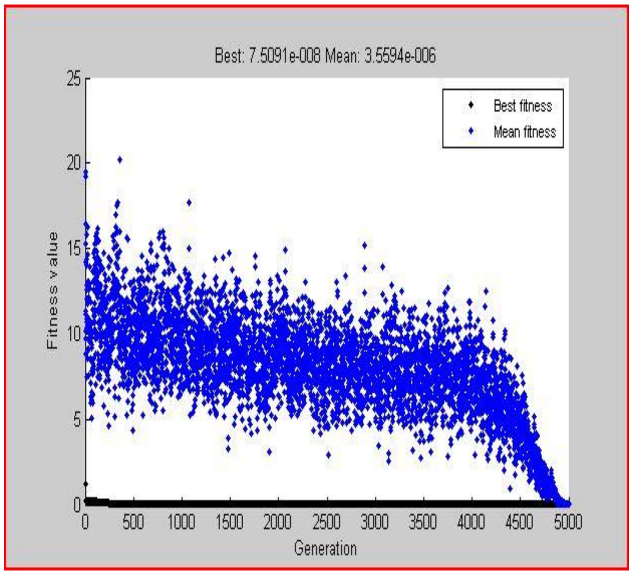


Figure 8

Before the overclocking we get the following result:

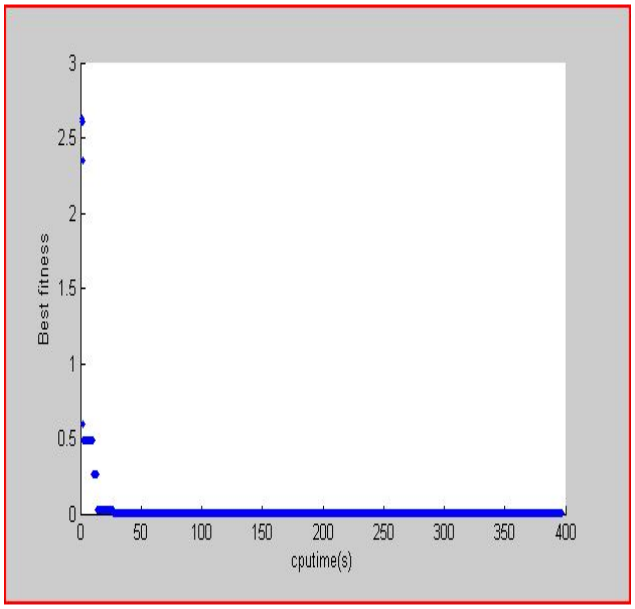


Figure 9

Figure 9 , is for (best fitness vs. CPU time) as we can see the CPU finish the 5000 generation by almost 400 second.

After the overclocking process (figure 10):

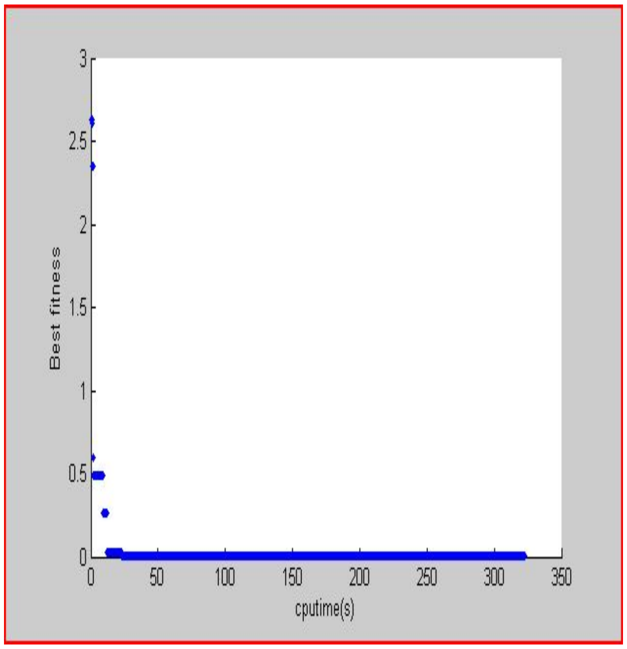


Figure 10

After the overclocking the CPU finish the job by around 320 seconds i.e. the difference is 80 second i.e. there is 20% boost in performance not bad for the genetic algorithm.

5. Conclusion and future work

In this experiment we show the impact of overclocking the CPU to the Genetic algorithm. For the CPU part, we have used Intel core 2 duo E6420, its original frequency is 2.13 GHz, and we have succeeded to hit 3.20 GHz which is about 50% boost. For the genetic part we have used the Rastrigin’s function which is often used to test the genetic algorithm. The number of generation is fixed to 5000 generation and in both cases the CPU time is measured and plotted. After the overclocking process we get 20% performance boost. This number gives us an idea about how the genetic algorithm performs with different kind of processors. Our future work will include testing the genetic algorithm with different object functions by using dual core processor and an overclocked dual core processor and quad core processor to see how much performance we can gain for the Rastrigin’s function.

References:

[1]http://compreviews.about.com/od/cpus/l/aaOverclock.htm

[2] Genetic algorithm and direct search toolbox user’s guide, Version 2.4.1, March 2009.

http://www.mathworks.com/access/helpdesk/help/pdf_doc/gads/gads_tb.pdf

[3]http://www.xbitlabs.com/articles/cpu/display/newbie-oc-guide.html

Ahmed Hikmat obtained his bachelor's degree in Control & Systems Engineering – Computer Branch in 2005 from University of Technology (Baghdad-Iraq).His area of research specialization includes network routing protocol (OSPF), genetic algorithm,parallel processing. Currently he is pursuing his Master's degree at Universiti Teknologi PETRONAS.

Azween Abdullah obtained his bachelor's degree in Computer Science in 1985, Master in Software Engineering in 1999 and his PhD in computer science in 2003. His work experiences includes twenty years in institutions of higher learning in both the management and academic capacities, and fifteen years in commercial companies as Software Developer and Engineer, Systems Analyst and IT/MIS and educational consultancy and training. He has spent more than a decade with leading technology firms and universities as a process analyst, senior systems analyst, project manager, and lecturer. He have participated in and managed several software development projects. These have included the development of management information systems, software process improvement initiatives design and implementation, and several business application projects. His area of research specialization includes computational biology, system survivability and security, autonomic computing and self-healing and regenerating systems, formal specifications and network modeling. His contributions include publishing several journal and refereed conference papers and in the development of programs to enhance minority involvement in bridging the ICT digital gap. Currently he is working on two projects funded by the Ministry of Science Technology and Innovation.