Investigating Microprocessor Heat Dissipation Through Computational Methods

Aran Truslove

***Abstract*—In this investigation, computational methods were used to investigate the heat dissipation of a 2D microprocessor model with a thermal power output of 5 $\times$ 10^8 Wm^{-3}. A finite difference method was used to solve Poisson's equation for heat conduction in a microprocessor system to find an estimate of the average operating temperature of the microprocessor. Various scenarios were analysed including a microprocessor attached to a ceramic case, a heat sink and when experiencing either natural or forced convection. The dimensions of the heat sink were varied to determine the smallest footprint that resulted in an operating temperature below 80 °C when a 20 ms^{-1} wind speed was applied. A microprocessor system with a footprint of $1495\ mm^2$ was found to satisfy this criterion, resulting in a mean operating temperature of 72 ± 7 °C.**

I. Introduction

Microprocessors have had a profound impact on society over the last century. They can be found in a variety of systems, from consumer computers to international space stations. It is estimated that 2.1 billion microprocessor units were shipped in 2021 alone [1]. This exemplifies the integral role that microprocessors play in the modern world and highlights the importance of designing them to be safe and efficient.

A side effect of the computational power of a microprocessor is the vast amount of heat it emits [2]. This arises from the many millions of transistors that modern microprocessors contain [3]. Each time a transistor switches between 0 and 1, a small amount of heat is produced due to electrical resistance. For a single transistor switch, this heat output is negligible but modern microprocessors are capable of executing billions of cycles per second. This results in a substantial heat accumulation, that if not dissipated efficiently, can cause the microprocessor components to break down [4]. For context, Intel processors' internal thermal controls typically activate between 100 °C and 105 °C to prevent damage to components [5].

Computational simulations can be used to test the operating temperature of a microprocessor in a range of scenarios. This involves creating a model of the microprocessor and its cooling components to investigate the temperature of the combined system through numerical methods. An advantage of this approach is a reduction in costs since physical components do not have to be tested. Additionally, various configurations can be rapidly compared to determine a theoretically ideal setup for reducing the microprocessor temperature. In this investigation, the Jacobi method was used to solve Poisson's equation for thermal conduction. Multiple cooling scenarios were studied to find a configuration that allowed the microprocessor to operate at a mean temperature below 80 °C.

II. Theory

Two primary physical scenarios were analysed in this investigation, the microprocessor attached to a ceramic case and the microprocessor and ceramic case attached to a heat sink. These setups are shown in Figure 1 and Figure 2. Details of the properties of the objects in the system are given in Table I.

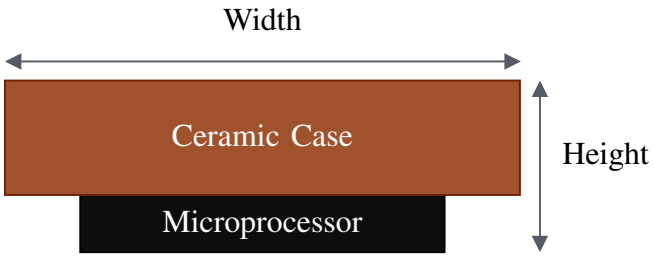

Fig. 1: Block diagram of the microprocessor attached to a ceramic case. The width and height arrows indicate the directions that these measurements are taken for the objects.

For all scenarios investigated, the microprocessor had a width of 14 mm and height of 1 mm while the ceramic case had a width of 20 mm and height of 2

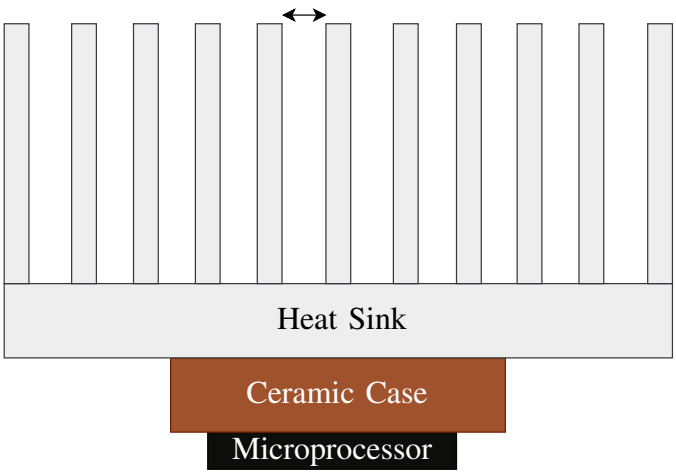

Fig. 2: Diagram of the microprocessor and ceramic case attached to a heat sink containing fins. The fin separation measurement is indicated.

Object	Material	Thermal Conductivity ($Wm^{-1}K^{-1}$)	Thermal Power Output (Wm^{-3})
Microprocessor	Silicon	150	5×10^8
Ceramic Case	Copper Tungsten	230	0
Heat Sink	Aluminium	250	0

TABLE I: Outlines the object properties that were used in the computational model.

mm. The heat transport equation in a homogeneous medium,

$$-k\left(\frac{\partial^2 T}{\partial x^2}+\frac{\partial^2 T}{\partial y^2}+\frac{\partial^2 T}{\partial z^2}\right)=q(x,y,z), \quad (1)$$

relates the temperature at a position to the thermal power output. k is the thermal conductivity of the material (units: $Wm^{-1}k^{-1}$), T is temperature (units: $^{\circ}C$), x, y, z are coordinate positions (units: m) and q is the thermal power output density (units: Wm^{-3}). This is an elliptical PDE that can be solved numerically through finite difference methods. To greatly reduce the execution time of solving this PDE, the z-component was ignored resulting in a 2D heat equation.

When considering interfaces between different materials, (1) no longer applies since k varies across the interface. To handle this case, conservation of heat flux is applied for a system in a steady state. The equation relating heat flux to temperature is

$$\boldsymbol{\phi} = -k\nabla T, \quad (2)$$

where ϕ is the heat flux (units: Wm^{-2}). For the boundary between an object and the surrounding air the heat flux equation is given by

$$\boldsymbol{\phi_s} = h(T_{surf} - T_a), \quad (3)$$

where $\phi_{\mathbf{s}}$ is the heat flux from the object to the air as a result of convection, h is the heat transfer coefficient of the interface (units: $Wm^{-2}K^{-1}$), T_{surf} is the temperature at the surface of the object and T_a is the ambient temperature of the air. For the model in this investigation, T_a was set to 20 °C (room temperature).

For the convection between the material and the air, two different scenarios were considered. The first is natural convection, in which no external fan is used. For this case, the heat transfer coefficient, $h_{natural}$, is modelled by the equation

$$h_{natural} = 1.31(T_{surf} - T_a)^{\frac{4}{3}}. \quad (4)$$

For the case of a fan providing additional cooling, the heat transfer coefficient, h_{forced}, is given by

$$h_{forced} = 11.4 + 5.7v, \quad (5)$$

where v is the wind speed produced by the fan (units: ms^{-1}) [2].

III. Method

A. Discretising the Heat Equations

The previous equations provide a theoretical framework that characterises the heat behaviour within the microprocessor system. For a simple Poisson's equation with well-defined boundaries, a solution may be found analytically. However, for the more complex microprocessor systems analysed in this investigation, numerical methods need to be implemented to determine an approximate solution for temperatures.

The domain was discretised into a set of grid points to ensure compatibility with numerical methods. The central difference scheme, used to approximate the second derivatives at each grid point, $T''(x_i)$, is

$$T''(x_i) \approx \frac{T(x_{i+1}) - 2T(x_i) + T(x_{i-1})}{\Delta x^2}, \quad (6)$$

where Δx is the step size between adjacent grid points. The truncation error of this approximation is $O(\Delta x^2)$. Applying (6) to (1) for a 2D domain leads to

$$T_{i+1,j} + T_{i,j+1} + T_{i-1,j} + T_{i,j-1} - 4T_{i,j} = -\frac{q\Delta x^2}{k}, \quad (7)$$

the indices i, j represent the x and y positions of the grid points. The Jacobi iterative method was used, to solve the system of linear equations described in (7). The equation for this method is

$$\vec{T}^{(n+1)} = \mathbf{D}^{-1}\vec{b} - \mathbf{D}^{-1}(\mathbf{M}_L + \mathbf{M}_U)\vec{T}^{(n)}, \quad (8)$$

where $\vec{T}^{(n+1)}$ is the temperature solutions of the next iteration, $\vec{T}^{(n)}$ is the solutions of the current iteration and $\vec{b}$ is a vector of constant terms corresponding to the right-hand side of (7). $\mathbf{D}$, $\mathbf{M}_L$ and $\mathbf{M}_U$ are related to the matrix, $\mathbf{M}$, representing the left-hand side of (7) by $\mathbf{M} = \mathbf{D} + \mathbf{M}_L + \mathbf{M}_U$. $\mathbf{D}$ is a matrix of diagonal terms of $\mathbf{M}$, while $\mathbf{M}_L$ and $\mathbf{M}_U$ are lower and upper diagonal matrices made up of the non-diagonal terms. When applying (8) to (7), an equation for the next iteration for the temperature at each grid point is obtained

$${T_{i,j}}^{(n+1)} = \frac{1}{4}(T^{(n)}_{i+1,j} + T^{(n)}_{i,j+1} + T^{(n)}_{i-1,j} + T^{(n)}_{i,j-1} + \frac{q\Delta x^2}{k}). \quad (9)$$

This equation is valid for the case that the grid point spacing in the x-direction is the same as the spacing in the y-direction (Δx = Δy).

For points on the material boundaries, (2) is discretised with the forward difference scheme, leading to

$$T_I = \frac{k_A T_A + k_B T_B}{k_A + k_B}. \quad (10)$$

T_I is the temperature of the point on the interface, k_A, k_B are the thermal conductivities of the adjacent points above and below the interface while T_A and T_B are the temperatures of the points above and below the boundaries.

When solving for grid points on the boundary between an object and air, one of the grid points for a non-corner boundary and two grid points for a corner will lie outside the bounds of the object when solving (9). To handle this case, (3) is discretised using the central difference scheme:

$$\frac{T_{i+1,j} - T_{i-1,j}}{2\Delta x} = -\frac{\phi_s}{k}. \quad (11)$$

(11) is specifically for boundaries normal to the x-direction but an analogous equation is used for boundaries normal to the y-direction. (11) can be substituted into (9) to remove the ghost points.

B. Algorithm Implementation

To discretise the spatial domain, a rectangular grid was overlaid on all the objects in the microprocessor system. The distance between adjacent grid points was set to the step size. Three masks were created with the same shape as the rectangular grid of points. The first two masks were for thermal conductivity and thermal power output, with each point populated with values according to Table I. The third mask specified the type of operation required to be performed on each point. These were numbered from 1 to 10, and distinguished between interior points, left, right, bottom and top boundaries, the four corner types and material interfaces.

Once these masks were generated, an initial guess for the temperature of the whole system was used to populate a rectangular grid of the same shape as the masks. (9) was then used to solve the spatial temperature for each grid point in the microprocessor system, by applying the necessary boundary/interface modifications as determined by the operation mask. After each iteration, the boundary condition, (3), was updated with the most current surface temperature values.

A stopping condition was used to determine when the temperature solutions had converged. The equation that was used to determine convergence is given by

$$\epsilon^{(n)} = \frac{\|\vec{T}^{(n+1)} - \vec{T}^{(n)}\|}{\|\vec{T}^{(n)}\|}, \quad (12)$$

where $\|\vec{T}^{(n)}\|$ is the Euclidean norm of the n^{th} iteration of microprocessor temperature solutions. $\epsilon^{(n)}$ was monitored for each iteration and once $\epsilon^{(n)}$ decreased to less than the stopping condition, the solution was considered to have converged. For this investigation, the stopping condition was chosen to be 1×10^{-7}. Stopping conditions greater than this resulted in unstable solutions, often converging after a single iteration, while stricter stopping conditions led to significant execution times.

C. Code Validation

To validate that the PDE solver was working correctly, $u(x, y) = x^2+y^2+5$, was tested. The PDE

solver was compared to the analytic solutions over a 10x10 domain, with a step size of 1 in arbitrary units. When discrepancies between the numerical PDE solution and the analytic solution were encountered, the algorithm was stepped through, one iteration at a time to troubleshoot the error. Once the PDE solver solutions matched the analytic solution, testing on the microprocessor system commenced, using the same technique to troubleshoot discrepancies.

D. Uncertainty Analysis

Two main forms of uncertainties were considered for the numerical solution to the PDE. Convergence uncertainty was calculated for each point by the equation $|\vec{T}^{(n+1)} - \vec{T}^{(n)}|$. This uncertainty was subsequently propagated when determining the mean temperature of the microprocessor.

Richardson extrapolation was used to determine an estimate for the exact solution with its associated discretisation uncertainty. This was done by determining the mean temperature at one step size and then solving for the mean temperature again with half the original step size. For these two solutions, an exact solution, T_{exact}, can be extrapolated by the equation

$$T_{exact} = \frac{4T(\frac{\Delta x}{2}) - T(\Delta x)}{3}. \tag{13}$$

A conservative estimate for uncertainty of T_{exact} is given by $|T_{exact} - T(\frac{\Delta x}{2})|$. For all the scenarios investigated, the discretisation uncertainty was magnitudes greater than the convergence uncertainty so the convergence uncertainty was neglected for the investigation [6].

IV. Results

A. Microprocessor and Ceramic Case

The first scenario analysed was that of the microprocessor attached to a ceramic case under natural convection. The solution of temperatures was determined for a 0.5 mm and 0.25 mm step size. (13) was then used to determine an estimate for the exact value of mean temperature. The mean temperature of the microprocessor for this scenario was found to be 5800 ± 200 °C. This temperature is substantially higher than the 1400 °C melting point of silicon [7].

B. Microprocessor, Ceramic Case and Heat Sink

1) Number of Fins: A heat sink was added on top of the ceramic case as in Figure 2. The heat sink fins each had a width of 1 mm, separation of 2 mm and height of 30 mm. Solutions of the mean microprocessor temperature were found with a varying number of fins. The number of fins tested was varied from 10 to 20. To reduce the execution time, solutions were found with a step size of 1 mm and 0.5 mm. This resulted in larger uncertainties but more practical execution times. A graph of the mean temperatures is shown in Figure 4.

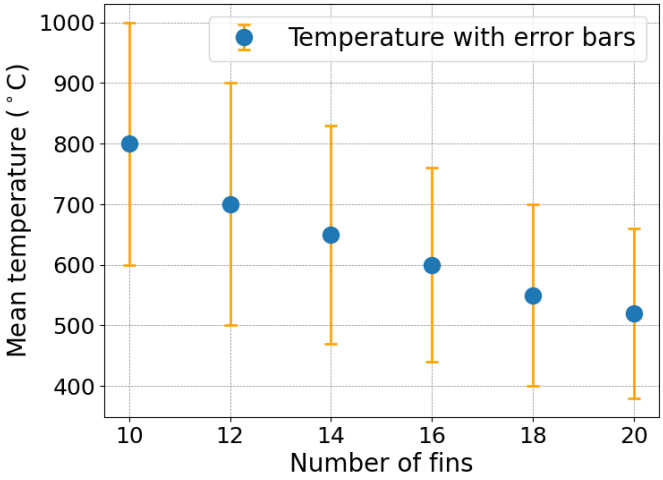

Fig. 3: Mean operating microprocessor temperature as a function of the number of fins on the heat sink with natural convection.

Figure 3 demonstrates a decreasing mean temperature of the microprocessor as the number of fins increased. This result is expected since a greater number of fins will lead to a larger surface length of the system in contact with the surrounding air, hence leading to increased heat dissipation from convection. The change in temperature with an increasing number of fins appears to fall at a diminishing rate. This makes sense in the context of the physical model since the additional surface length with each new fin as a fraction of the total surface length decreases with an increasing number of fins. Additionally, each set of new fins is situated further away from the microprocessor than the current set of fins, hence contributing less to the heat dissipation of the microprocessor.

2) Fin Height: For 14 fins with a fin width of 1 mm and fin separation of 2 mm, the mean operating temperature of the microprocessor was

investigated for a varying fin height. The fin height was varied from 0 mm to 60 mm. The results of this investigation are shown in Figure 4.

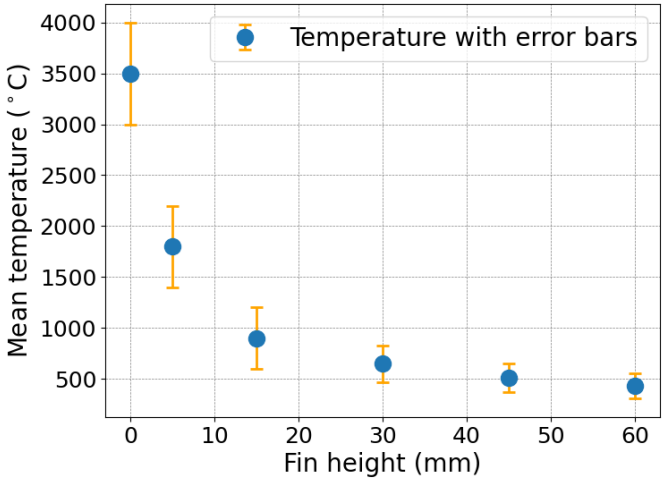

Fig. 4: Mean operating microprocessor temperature as a function of the fin height.

Figure 4 demonstrates a decrease in mean microprocessor temperature with a diminishing rate as the height of the fins increases. This relationship is expected for the same reasons as those that explained the change in microprocessor temperature with an increasing number of fins.

3) Fin Spacing: For 14 fins, each with a width of 1 mm and height of 30 mm, the fin separation was varied from 1 mm to 7 mm in 1 mm steps. The heat sink base width was increased to accommodate the increasing separation. For all the separations measured, the mean operating temperature of the microprocessor was determined to have a nominal value of 650 $°C$. However, the uncertainty varied between 70 $°C$ and 160 $°C$. This implies that for a constant number of fins, varying the heat sink separation makes no significant impact on the mean temperature of the microprocessor. Given this finding, to reduce the footprint of the microprocessor system, it is likely beneficial to increase the density of heat fins.

None of the configurations investigated for natural convection resulted in temperatures below 100 °C. Although, much larger heat sinks may have reduced the mean microprocessor temperature to below this threshold. The dimensions of the heat sink would likely have been significantly greater than what could feasibly be used in a real-world system. This emphasises the importance of implementing additional methods to enhance the cooling of microprocessor systems with high thermal outputs.

C. Forced Convection

To overcome the excessively high temperatures experienced by the microprocessor system with natural convection, a 20 ms^{-1} wind speed is incorporated in the model. This alters the boundary conditions by changing the heat transfer coefficient from $h_{natural}$ to h_{forced}, given by (5). In consideration of the previous findings, to minimise the temperature of the microprocessor, a 1 mm fin separation was used and the fin width was held at 1 mm. To address diminishing returns in microprocessor temperature reduction with disproportionate dimension increases, the heat sink width and fin height were equally set for each measurement. The heat sink width was varied from 35 mm to 59 mm and for each measurement, 0.005 mm and 0.0025 mm step size solutions were used. The results of this investigation are demonstrated in Figure 5.

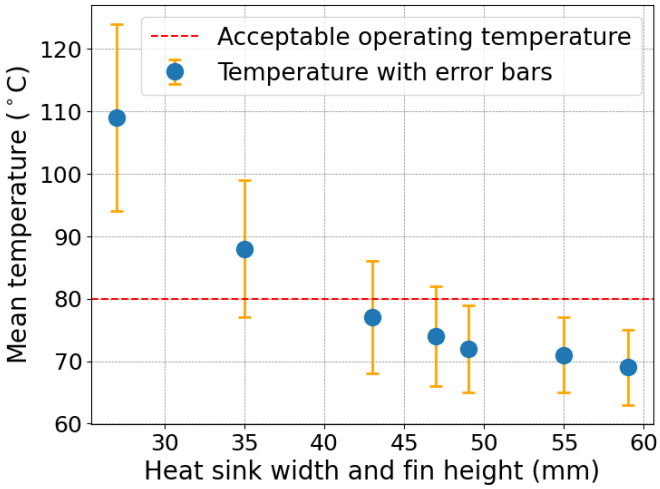

Fig. 5: Mean operating microprocessor temperature as a function of heat sink width and fin height. The red dashed line indicates the 80 °C temperature below which the microprocessor can operate safely.

The smallest heat sink width and fin height that permitted the microprocessor to operate below 80 °C and within the margin of its uncertainty, was determined to be 49 mm (53 mm heat sink height including the base) with 25 fins. This resulted in a mean microprocessor temperature of 72 ± 7 °C with a combined system footprint of 1495 mm^2. The size of this system suggests it could be suitable for integration with typical desktop computers.

V. Discussion

A. Execution Time

The Python `NumPy` module was utilised for vector additions in each iteration of the Jacobi method. `NumPy` executes vectorised operations with precompiled C code which can result in significantly faster performances compared to equivalent operations with Python loops [8]. Regardless of this implementation, the execution of the PDE solver took several minutes for many of the scenarios analysed. This made it impractical to achieve lower uncertainties as it limited further decreases in step size.

Analysis with the `cProfile` module showed that the Jacobi iteration accounted for 90% of the execution time. To address this, successive over-relaxation could have been used in place of the Jacobi method as it often converges with fewer iterations when an optimal weighting factor is chosen [9]. Variable step sizes could be implemented, using smaller step sizes for more intricate objects such as the microprocessor and fins while using larger step sizes for the heat sink base. This would work to reduce the number of calculations while not significantly compromising on accuracy.

B. Limitations of the Model

Although solving a 2D model is less computationally demanding, it comes with drawbacks when accurately representing the physical situation. In practice, microprocessor systems would be non-uniform in the z-direction which makes it challenging to justify using a 2D simulation for a 3D system.

Another limitation of the model is that the temperature of the air is considered to be fixed at 20 °C and its temperature increase as a result of being in contact with the microprocessor system is neglected. This assumption can be considered more accurate for the forced convection case since this involves replacing air in the immediate surroundings of the microprocessor system with room-temperature air delivered by the fan. The heating up of air can play a more prominent role when the microprocessor system has a high surface area to volume ratio, particularly for systems with a large number of heat sink fins.

VI. Conclusion

In this computational investigation, the Jacobi method was used to solve Poisson's heat equation for a 2D microprocessor system. Various scenarios were analysed by varying the dimensions of the heat sink and considering both natural and forced convection. For natural convection cases, no microprocessor system was found that led to viable operating temperatures, highlighting the importance of incorporating active cooling methods. In the case of forced convection, a heat sink configuration was found that resulted in a mean operating microprocessor temperature below 80 °C. This resulted in a microprocessor system that had a footprint of $1495\ mm^2$, representing a system that hypothetically could be integrated into a desktop computer. The limitations of the model were discussed, including the use of a 2D model for a 3D system and the heating up of the surrounding air being neglected.

References

[1] "Total microprocessor market history and forecast," https://hardwarebee.com/total-microprocessor-market-history-and-forecast/, 2021, accessed: 13/12/2023.

[2] J. Owen and M. Scott, "Heat dissipation in microprocessors," 2023, Imperial College London.

[3] Intel. (2008) Intel microprocessor quick reference guide. Accessed: 13/12/2023. [Online]. Available: https://www.intel.com/pressroom/kits/quickreffam.htm

[4] M. (https://electronics.stackexchange.com/users/4245/majenko), "Why does a processor get hot?" Electrical Engineering Stack Exchange. [Online]. Available: https://electronics.stackexchange.com/q/160251

[5] Intel. (2023) Information about temperature for intel® processors. Accessed: 13/12/2023. [Online]. Available: https://www.intel.com/content/www/us/en/support/articles/000005597/processors.html

[6] C. Chicone, "5.4.9 test approximation order using richardson extrapolation," in *An Invitation to Applied Mathematics*, C. Chicone, Ed. Academic Press, 2017, pp. 85–193. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128041536500051

[7] The Editors of Encyclopaedia Britannica, "silicon," https://www.britannica.com/science/silicon, 2023, accessed: 13/12/2023.

[8] NumPy Developers, "What is numpy?" https://numpy.org/devdocs/user/whatisnumpy.html, 2023, accessed: 14/12/2023.

[9] J. Owen and M. Scott, "Computational physics lecture notes: 3.5.2 gauss-seidel method / 3.5.3 successive over-relaxation method," 2023, imperial College London.