Using Python to Model Heat Diffusion Inside a Microprocessor

Jack Carlin
jack.carlin17@imperial.ac.uk
Blackett Laboratory
Imperial College London, SW7 2AZ

***Abstract*—This reports documents the methods used to analyse how heat diffuses in a microprocessor and any key accessories which may be added on. To do so, methods such as discretisation and numerical differentiation were implemented through Python to simulate the motion of heat in a system. The physics was then put to the test by altering the system and optimising parameters to find relationships between heat flow and the shape of the system. All of this was done through the scope of trying to minimise the temperature of said microprocessor so that it would be suitable for operation, which was ascertained by completing key tasks.**

I. Introduction

THE aim of the investigation was to create a simulation in Python, to observe how heat diffuses in electronic components; in this case, specifically a microprocessor. To do so, a program was written to solve an elliptic partial differential equation (PDE) in the steady state, which modelled the iterative change of the system until convergence.
This is a necessary approach because despite heat flow being described by a differential equation, it is not always possible to solve it analytically. Therefore it is required to simplify the problem by dicretizing and solve numerically.

The model itself is valuable because it gives a benchmark of predicting what temperature a central processing unit in an electronic device may reach given certain conditions, and as a result manufacturers can use that information to set fail-safes prior to physical testing.

II. Theoretical Background

The fundamental equation we looked to solve was:

$$\vec{\nabla} \cdot \vec{\phi} = q(\vec{r}). \tag{1}$$

where $\vec{\phi}$ is the flux of thermal power which can be described as the gradient of temperature $\vec{\phi} = -k\vec{\nabla}T$, and $q(\vec{r})$ is the thermal power density in units of W/m^2. By combining the definition of the thermal flux with *Equation 1*, we find a form of Poisson's Equation:

$$-k\vec{\nabla}^2 T = q(\vec{r}), \tag{2}$$

where k is the thermal conductivity in units of W/mK, and T is the temperature.

In order to solve the differential equation we have to look at the boundary conditions. The boundaries of the system will be losing heat to the surroundings, which means that the boundaries are described by a temperature difference. As a result the boundary conditions would be categorised as Von Neumann boundaries, and their behaviour would be more closely described by *Equation 1*. This surface flux, ϕ_{surf}, is explicitly shown as,

$$\phi_{surf} = h(T_{surf} - T_{amb}), \tag{3}$$

where T_{surf} and T_{amb} are the surface and ambient temperatures, and h is the heat transfer coefficient at the boundary. The heat transfer coefficient is a proportionality constant which changes depending on how the energy is being removed from the system at the boundaries. Throughout this investigation, we only consider natural convection and forced convection which invoke approximate heat transfer coefficients of,

$$h_{natural} = 1.31(T_{surf} - T_{amb})^{\frac{1}{3}}, \tag{4}$$

$$h_{forced} = 11.4 + 5.7\nu, \tag{5}$$

where ν is the windspeed that is passing over the surfaces of the system.

III. Implementation and Validation

The implementation was split into two avenues: considering the interior points of the system, and considering the boundary points; however all of these points were manifested on a meshgrid. In order to implement the differential equation into the interior points, *Equation 2* had to be discretized. To do so $\vec{\nabla}^2 T_{ij}$ had to be approximated using a finite difference method, this Laplacian can be displayed in the form of a pictorial operator,

$$\vec{\nabla}^2 T_{ij} = \frac{1}{\delta^2}\begin{bmatrix} & 1 & \\ 1 & -4 & 1 \\ & 1 & \end{bmatrix} T_{ij} \tag{6}$$

where δ is the spacing between the grid points, which was realised as a resolution in the program. Consequently, using the Jacobi Method [1], the Poisson Equation can be approximated and put in the form of an iterative equation:

$$T_{ij}^{n+1} = \frac{T_{i+1,j}^n + T_{i-1,j}^n + T_{i,j+1}^n + T_{i,j-1}^n}{4} + \frac{\delta^2 q_{ij}}{4k_{ij}}, \tag{7}$$

where n is the iteration number. Essentially, this equation means that for the interior points, the new iterated temperature at a certain point is the average of its neighbouring points on the previous iteration.

After enforcing the iterative technique for the interior points, the boundary points were considered. Given that the form of the system does not change, the boundary points could be found and stored so that on every iteration they can be accessed and updated. The updates on the boundary are influenced by the Von Neumann boundary conditions which are outlined in the previous section, and to enforce them a directional central difference scheme is used.
The central difference method was used to apply the Von Neumann boundary conditions because it is a numerical approximation to the gradient, however this also had to be attributed to a certain direction according to the direction that the flux is leaving the system at that point. As a result the program was written such that the when the points along a certain boundary were updated, there would be an input into the function which defined the direction of the flux and thus the direction in which the central difference method would be applied across. Taking the upward facing boundary the CDM would be in the form,

$$\left.\frac{\partial T}{\partial y}\right|_{i,j} = \frac{T_{i,j+1} - T_{i,j-1}}{2\delta} = -\frac{\phi_{i,j}}{k_{i,j}}, \tag{8}$$

where $\phi_{i,j}$ is the ***surface*** flux at point i, j, given by *Equation 3*. In the case of the upward facing boundary, the temperature above the boundary, $T_{i,j+1}$, is fictitious; the point itself is non-physical but is used solely for the purpose of applying the boundary condition.

These boundaries surround the material, which will be set up two-dimensionally in the format of the microprocessor with a ceramic case mounted on top. Later on a finned heatsink was added on top of that for increased heat diffusion. The chip, case and heatsink base were all generated in similar fashions as they are simple rectangular components in two dimensions, however the fins proved to be more difficult. They were initialised recursively, meaning that the first one was outlined including the boundary in between the fins, and then repeated along the width of the heatsink base. All subsections of the total material were then used as index ranges for the rest of the program, with a parameter for resolution which defined how many mesh-grid points would be contained within one square millimetre.

Once both the frameworks for updating the interior and boundary points were set up, the iterative process would begin, with the interior points being updated first and then the boundary points. As a result a steady-state solution could be found, showing how heat generated by the chip will distribute itself across the whole system. Naturally, as the program iterated to find the temperature solution at every point, each time the value got closer and closer to the solution. This can be realised by setting up a convergence criteria, ϵ, which gives an idea of how much the temperature is changing every iteration,

$$\epsilon = \frac{\sum_{i,j} T_{i,j}^{n+1} - \sum_{i,j} T_{i,j}^{n}}{\sum_{i,j} T_{i,j}^{n}}. \tag{9}$$

The ϵ sums both the temperatures on the grid and the next grid iteration, then finds the proportional change between the two iterations to find how close the current iteration is to the theoretical solution. Subsequently the number of iterations the program will run through can be defined by the convergence parameter; once ϵ reaches a certain value the process will end, and in general ϵ was set to 1×10^{-7}. A convergence criteria using summation was chosen to represent the whole grid; if a singular point was chosen to compare, a certain risk would be opened up if program was initialised with a mistake.

At this point in the method, the simulation was complete and could be run with different parameters such as fin height or fin spacing, which would result in varying final temperatures (discussed in more detail in the next section). However, varying parameters proved to be very computationally intensive, so in order to alleviate such computational stress, parameters were varied according to a method akin to Newton-Raphson's Method. A binary search algorithm was used to find the midpoint of bounds where the lowest temperature would result, so that the behaviour close to the optimal parameter was well sampled.

Finally in order to do a quick check on whether or not the heat is diffusing in the way we would expect, an easy check to carry out would be to remove the source term within the microprocessor and let the program run. In theory this would lead to the whole system averaging out at the ambient temperature, but in practice this showed to take many iterations as the temperature would only tend to T_{amb}. However it was clear that the shape of the temperature would imply that it was converging to a value close to 20°C, which was the set ambient temperature for the investigation.

IV. Results and Discussion

A. No Heat Sink

The first task that we set out to complete was to solve for the temperature of just the microprocessor and its ceramic case, with only natural convection to take heat from the system. The system in question includes a microprocessor of width 14mm and thickness 1mm, below a case of width 20mm and thickness 2mm. The processor is also assumed to produce 0.5W of thermal power density per 1mm $\times$ 1mm grid cell. Given these conditions, it can be seen in *Figure 1* that a system with just the microchip and case alone is not viable as the temperature is far above a working temperature range of 60°C - 80°C, with an average temperature of 6600K (2 s.f.). For this simulation in particular the temperature of the system was initialised at 500K but has clearly converged as shown in *Figure 2* where the epsilon can be seen to level off after 5 million iterations. The general trend of the convergence parameter, ϵ, is that it follows some form of exponential decline.

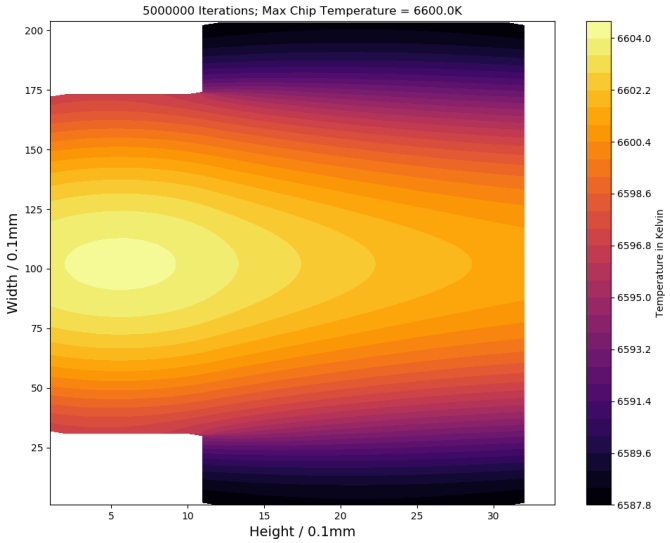

Fig. 1. *Filled contour map of the temperature distribution in the first case. Given the orientation of the graph is side-wards the reader should consider it rotated 90°anticlockwise. Furthermore, the proportionality of the plot is not to scale, the microprocessor and case are far thinner than they are wide.*

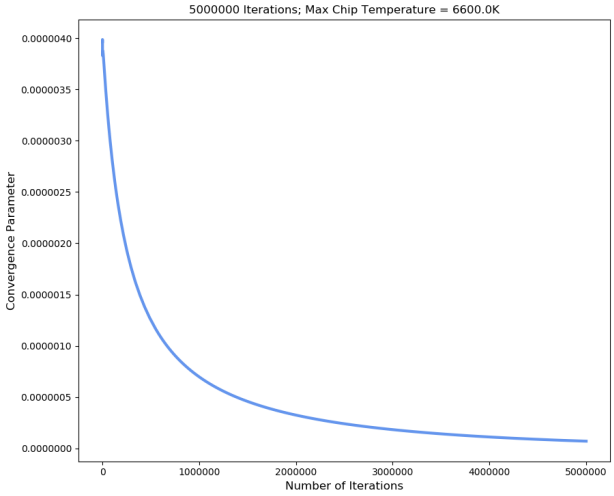

Fig. 2. *The relationship between ϵ and the number of iterations is shown here to be an exponential decline. At 2,000,000 iterations the convergence begins to level off rapidly. It is also worth noting to the reader that there is a small blip right at the beginning where there is a brief inflection due to the instability at the start of the run.*

B. With Heat Sink

The prior task was reasonably non-physical as for the last two decades, central processing units have been powerful enough to require at least a heatsink in order to avoid overheating. As a result it was imperative to implement a heat sink component into the system.

The heatsink was initialised emulating that of an aluminium base of thickness 4mm; with periodic fins along the top with a width of 1mm, spacing of 5mm, and height of 30mm. The base width was not given as part of the task, however it was set to 61mm which fits with common dimensions of heatsinks and fan systems that are manufactured [2]. As shown in *Figure 3*, the heat now dissipates into the fins, which helps with a cooling effect as now there is a larger surface area to volume ratio in comparison to the original case in *Section A*. After

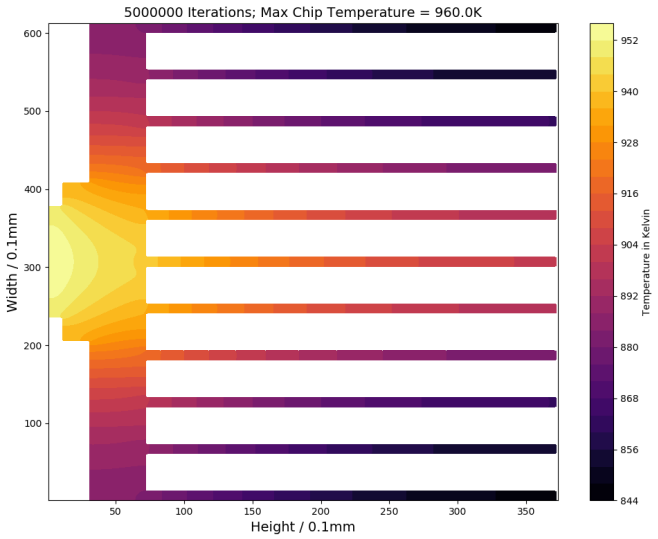

Fig. 3. *Filled contour map of the temperature distribution of the initially given dimensions of the system with a heatsink. Similarly to the first contour map, the resolution is set to 0.1, thus there are 10 points per millimetre. The colours may be misleading, but the darker areas of the contours are still very warm in this case; the temperature difference is only 100K*

the heatsink was added, the mean temperature of the the chip went from 6600K to around 955K (with a max temperature of 960K). This is still far beyond the working the temperature and would not be suitable for use however it does show that adding a heatsink can cause drastic cooling in the system. Furthermore the program converged in much the same way, as shown in *Figure 4*, however this time the plot has changed into a semi-log format to explore whether the behaviour stays exponential; the logarithm of the convergence at base 10 decreases linear with every iteration which would imply that this exponential behavior holds.

Despite the vast improvement in the chip temperature, it still would not be suitable for use in electronic devices, however the heatsink is a more versatile component of the system in comparison to the case and chip because it is manufactured with the intention to cool down the other components and thus is not confined to a certain size based on other variables like the chip. Subsequently, we were able to vary two key parameters of the heatsink and investigate how they affect the temperature. These variables were the height of the fin and the spacing between each fin. Naturally both will increase the surface area and thus the amount of energy that can be removed from the system is larger, yet it is worth noting which will have the larger effect whilst also considering whether or not the parameters are physical enough to actually be used within an electronic device.

The first parameter that was varied was the number of fins within a set base width, essentially varying the spacing. *Figure 5* shows the relationship between the maximum chip temperature and the number of fins that can fit with integer

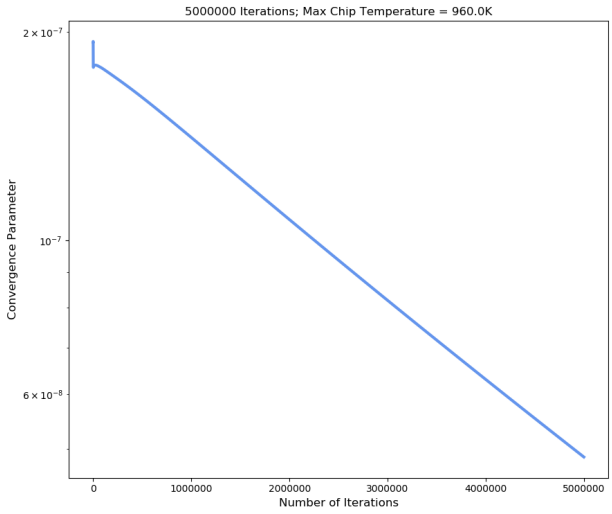

Fig. 4. *Now the relationship between convergence and the number of iterations is confirmed to be mostly exponential as it varies linearly on a semi-log plot. The vertical line at the start the is a more pronounced version of the aforementioned inflection.*

spacing within a 61mm wide base, this in turn means that the fin spacing was varied from 5mm (displaying in *Figure 3*) down to 1mm over 1mm steps. The relationship is seemingly

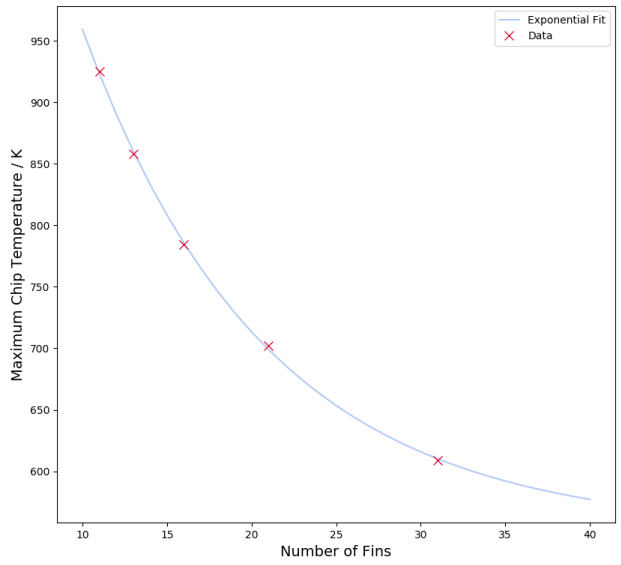

Fig. 5. *The program was run multiple five different times, each time with a different number of fins. The relationship seems to be in the form of an exponential again, however with only five points to work from the fit is likely flawed to a certain extent.*

exponential so having a smaller fin spacing would lead to a lower CPU temperature, however it also means that after a certain point the gain is very small due to diminishing returns; a clear conclusion is that 5mm spacing is inadequate so the initial form of the heatsink had to be changed. Although 0.5mm spacing would lead to more cooling, for the sake of the task, 1mm was set as the minimum and going forward with analysis that was the spacing used.

Lastly, the height of the fins were also varied however they showed almost identical behaviour. *Figure 6* shows how the maximum chip temperature exponentially decreases as the height of the fins increase. Similarly with the conclusions

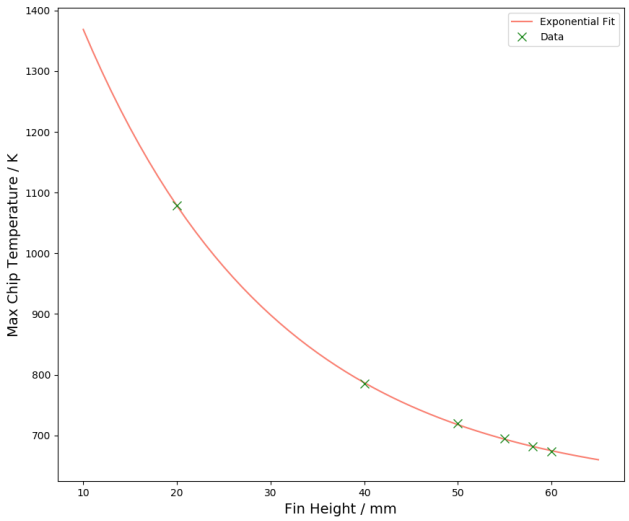

Fig. 6. *Here the relationship between the maximum temperature of the chip and the height of the fins on the heatsink to be exponential. A binary search algorithm was used to sample more of the points near the minimum, however in this case the temperature is always declining within this range of fin heights. Therefore it is most well sampled closest to the maximum value analysed, 60mm.*

drawn with *Figure 5*, smaller fin heights lead to very little cooling whatsoever but very large fin heights are unrealistic in terms of manufacturing and any height change amongst them will cause a very small amount of change. Anything past 40mm fin height is reasonable as afterwards the maximum heat starts to level off however it will depend a lot on what kind of electronic device it will be used in. Moving forward onto the next task, 50mm was used as the fin height.

C. With Heat Sink and Forced Convection

As aforementioned, it was deemed logical to keep the base dimensions at 61mm $\times$ 4mm, reducing the fin spacing to the minimum of 1mm, and increasing fin height to 50mm, from what was displaying in *Figure 3*. These changes would increase the surface area to volume ratio drastically and thus would maximise the heat loss from the system whilst staying within a space that may be allocated in real devices such as a personal computer.
Using these dimensions, the final part of the task was to add a variable of forced convection that would emulate a small fan drawing the warm air out of the system. The only change here would be to shift the heat transfer coefficient to the one described *Equation 5*, and in turn increasing the flux out of the material where the fast moving air will draw more heat from the surface.

This change proved to be very successful in removing heat from the system when applied in tandem with more optimal parameters. The resulting system is shown in *Figure 7*, with a surface-passing windspeed of 20m/s. The average temperature

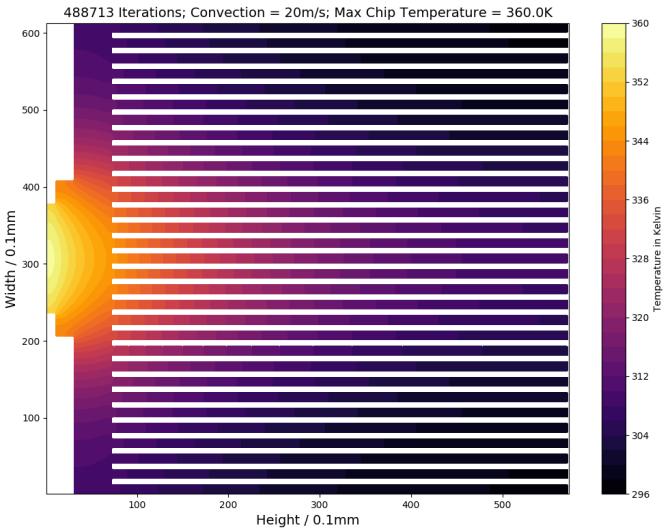

Fig. 7. *Final contour map of the report. The map shows a visualisation of more optimal parameters in the previous map with 31, 50mm tall fins. It is worth noting to the the reader that the number of iterations is far less than in previous analysis, however as shown below the temperature was close to convergence, with an* ϵ *on the order of* 10^{-8}.

of the microprocessor in this case is **355K**, with the maximum being 360K. The operating temperature range is 60°C - 80°C, and when including forced convection into the simulation, the mean chip temperature will converge onto **83.3°C** for the specific setup derived from the *Section B*. *Figure 8* allows us to reinforce that the temperature is indeed converging onto a value close to the operating range by looking at how temperature changes per iteration. Clearly by the last iteration the temperature change is levelling off quite fast.

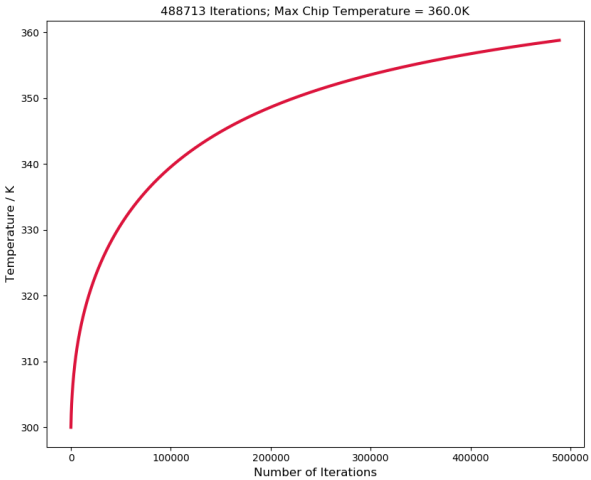

Fig. 8. *This relationship provides much the same information as the previous convergence parameter versus number of iterations plots seen in Figures 2 and 4. Instead the temperature is visualised for the final plot to show that the relationship between temperature and a given iteration is an exponential tending to the final temperature.*

So with the final set up which is close to optimal parameters, the final temperature reached was 83.3°C. This would continue to decrease with a wider base and taller fins, however given the exponential nature of the relationship, probably only by a little. This final temperature however may still be valid given the limitations of the simulation. Of course the simulation wasn't able to entirely predict a temperature for a given set of parameters, one of the biggest limitations of the model is that it is in two-dimensions. Subsequently the surface area to volume ratio is underestimated in our model, thus less heat will be leaving the system; put simply, the model we use for analysis will overestimate the temperature of the system. Furthermore, a slightly less important mechanism which was not considered was the loss of heat through radiative effects. The microchip is very small so the amount of radiation leaving the system would not cause a large shift in the model but an assumption could be made that it would lower the final temperature by a couple of °C, however it was out of the scope of the investigation.

V. Conclusion

To conclude, the goal was to investigate how heat diffusion can be realised in the steady-state by solving a form of Poisson's Equation under the context of the studying microprocessors. In turn, a model was developed to predict a temperature that the system would converge to, and thus the system was then altered with physical context to get the microprocessor temperature within operating range.
This goal was achieved to the extent that a program was written that was capable of testing multiple different parameters for a microprocessor cooling system and thus able to explore the relationship between heat content of the microchip and the shape of the system.

The investigation was not without its limitations however. These constrictions can be categorised into computational and physical limitations. Regarding the former, the program had to update many points per iteration given that the lowest resolution used was 10 grid points per mm, and as a result data analysis and certain checks took a lot of time; stagnating progress. Whereas on the physical side, the simulation was not able to encapsulate a true representation of the system primarily because it was analysed in two dimensions, as well ignoring the thermodynamics of the surroundings and radiative effects. Despite this, the program still mostly fulfilled what it set out to do in a simplified form.

References

[1] S. A. T. William H. Press, *Numerical Recipes: The Art of Scientific Computing*, Third Edition. Cambridge University Press, 1986.

[2] M. I. Suzana Prstic, "Bypass effect in high performance heat sinks," *International Thermal Sciences Conference*, 2000.