Numerical Modeling ?

- Why do we need need numerical modelling in the first place?
- Usually some complex or not so complex phenomena needs to studied.

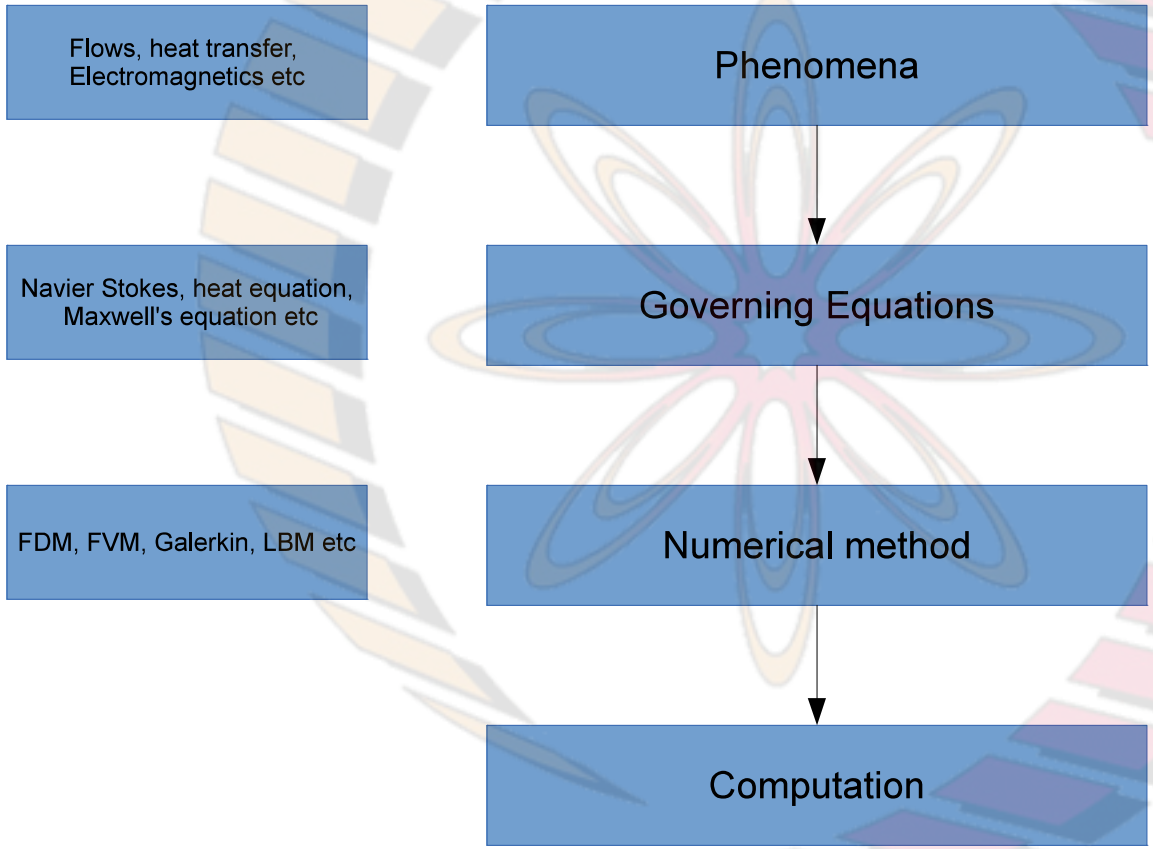
Numerical Modeling ?

- Why do we need need numerical modelling in the first place?
- Usually some complex or not so complex phenomena needs to studied.
- Analytical solutions are not easily available or in most cases not achievable at all.

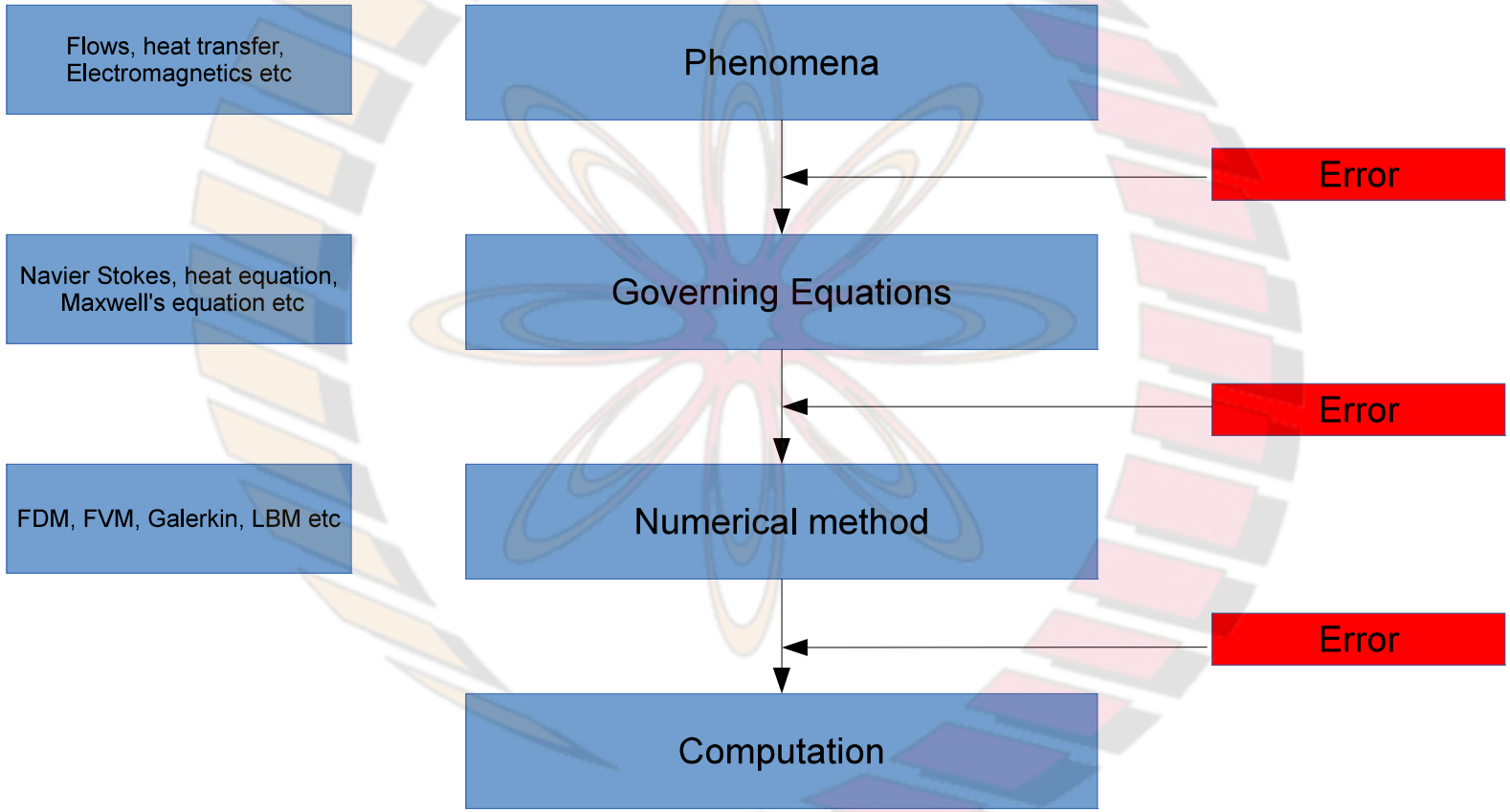
Numerical Modeling ?

- Why do we need need numerical modelling in the first place?
- Usually some complex or not so complex phenomena needs to studied.
- Analytical solutions are not easily available or in most cases not achievable at all.
- What are the steps involved in numerical modelling?

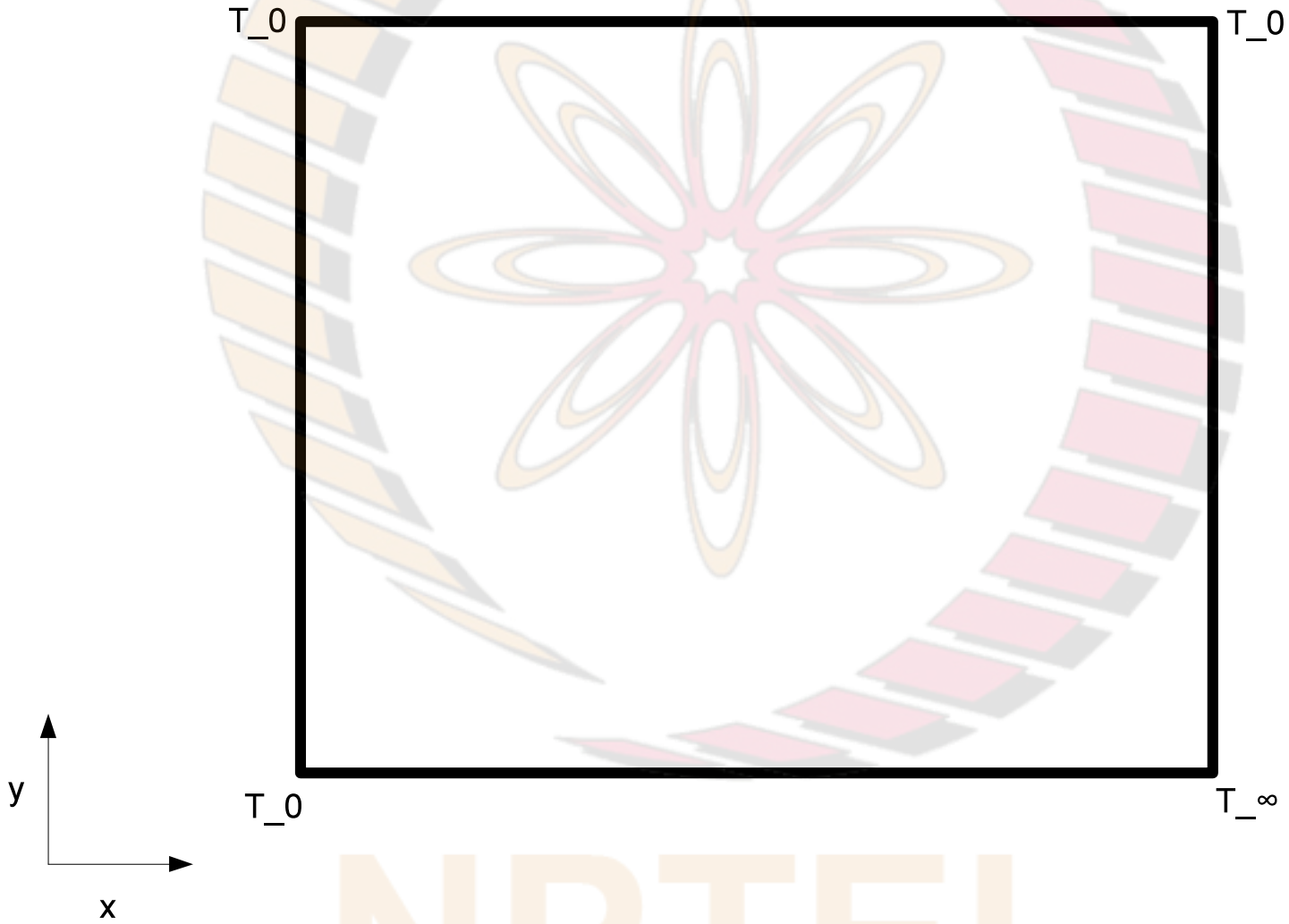
Numerical Modeling



Numerical Modeling



Steady State Heat Conduction

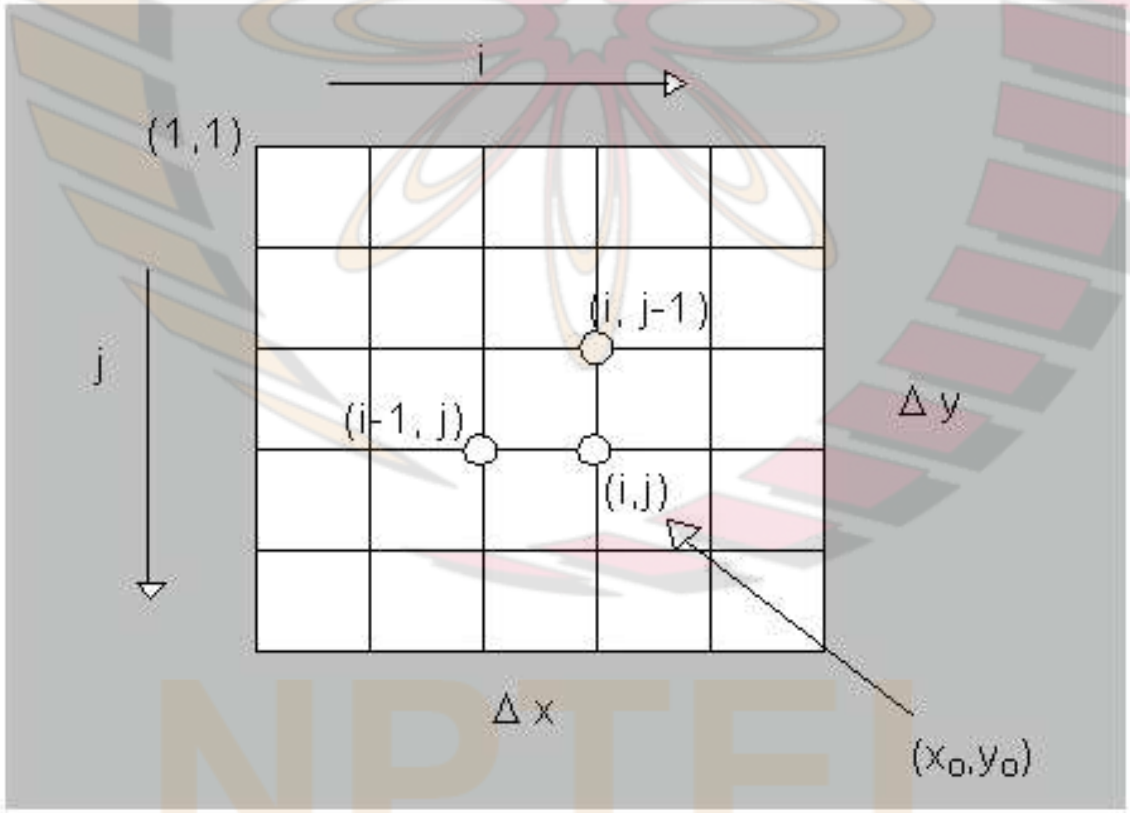


Steady State Heat Conduction

Phenomenon is modelled using Laplace's equation

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = \nabla^2 T = 0$$

which can be discretised on a grid as,



Steady State Heat Conduction

The discretised equation at a single point (i,j) is

$$\frac{T_{i-1,j} - T_{i,j} + T_{i+1,j}}{(\Delta x)^2} + \frac{T_{i,j-1} - T_{i,j} + T_{i,j+1}}{(\Delta y)^2} = 0$$

Assemble all the equations for all unknown points in the Matrix form and then solve

$$Ax = B$$

You can choose any Linear Algebra Solver (iterative or Direct). Iterative is more efficient.

So what are the steps involved ?

One writes a computer code,

```
#define NC       1000                  /* Number of Cols        */
#define NR       1000                   /* Number of Rows        */
#define NITER    1000                   /* Max num of Iterations */
#define MAX(x,y) ( ((x) > (y)) ? x : y )

#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <mpi.h> /* only for timing */
#include <sys/time.h>

void initialize( float t[NR+2][NC+2] );
void set_bcs   ( float t[NR+2][NC+2]);

int main( int argc, char **argv ){

  int        niter;                    /* iter counter  */

  float      t[NR+2][NC+2];           /*temperature */
  float      told[NR+2][NC+2];        /* previous temperature*/
  float      dt;                       /* Delta t       */

.................................
```

Compile and Execute!!!

Steady State Heat Conduction

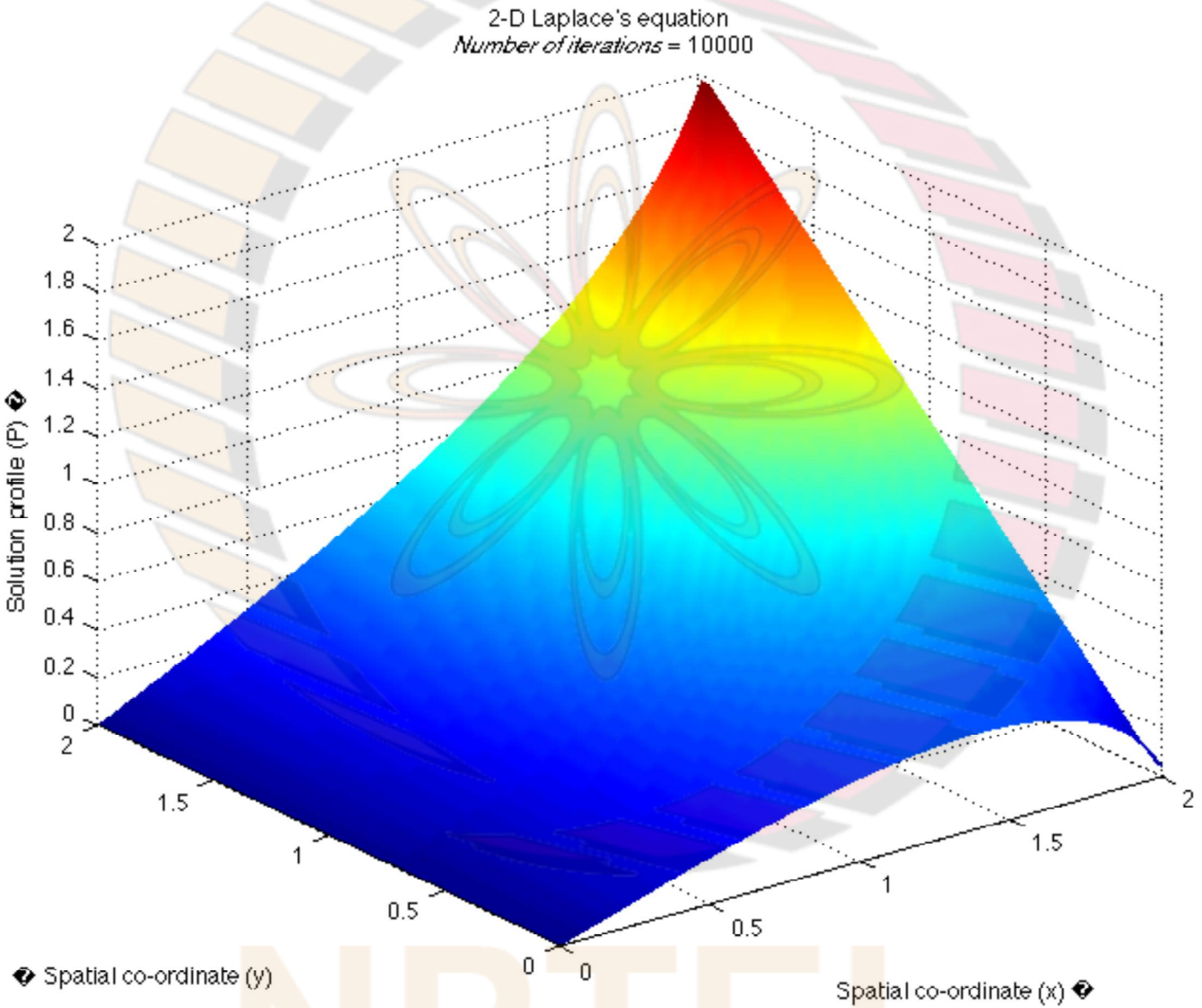
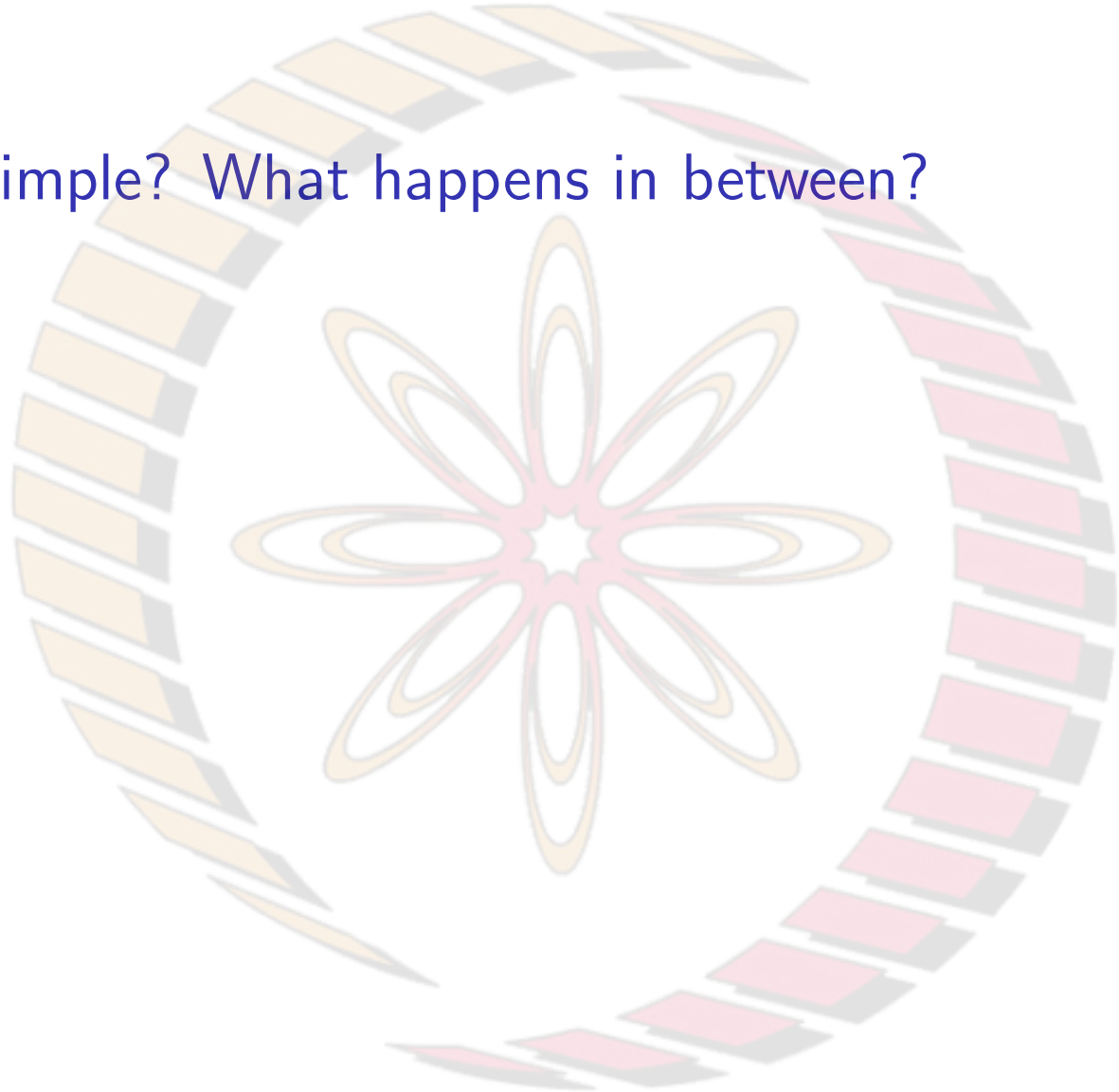


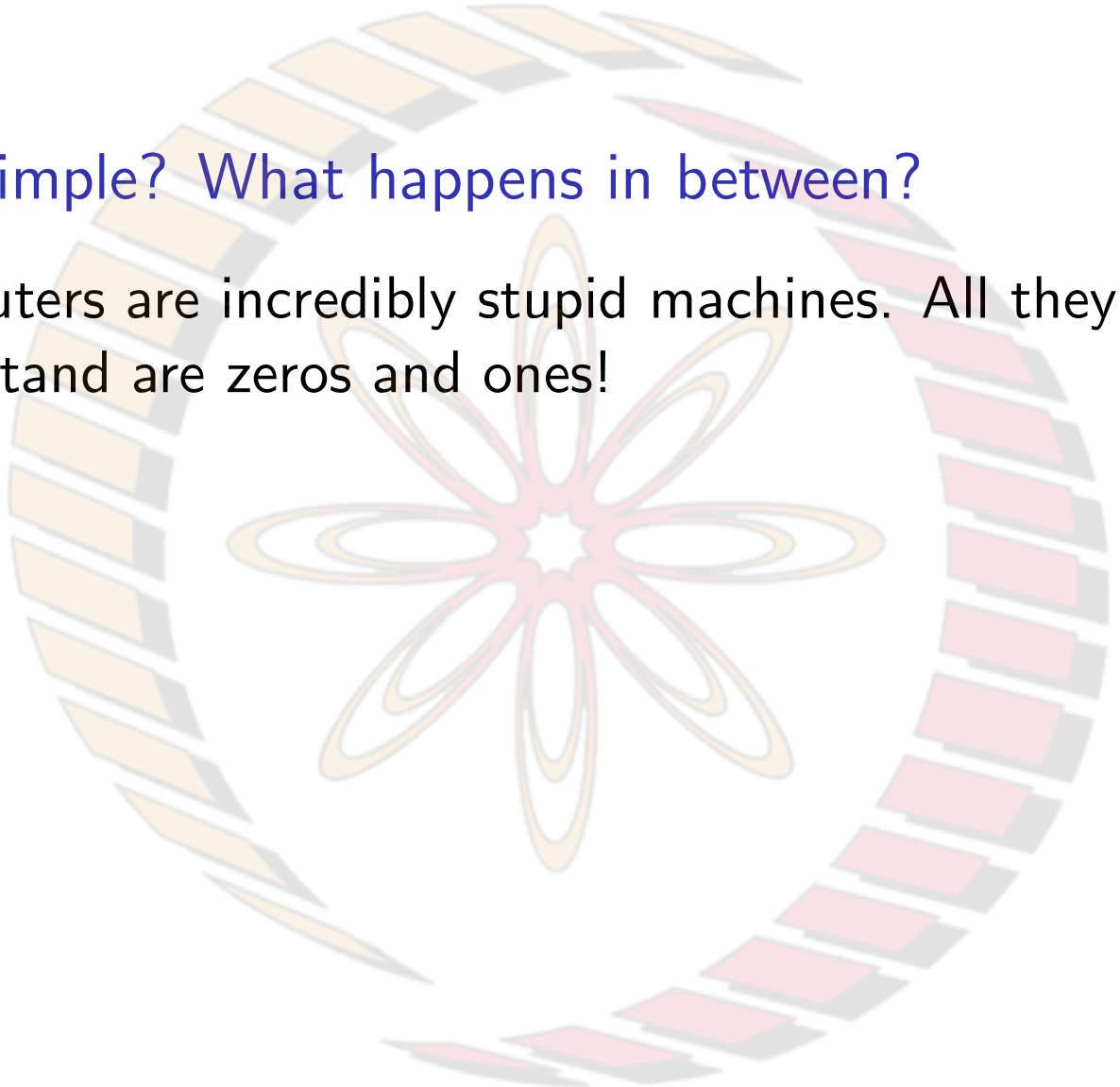
Image: Mathworks

Is it that simple? What happens in between?



Is it that simple? What happens in between?

- Computers are incredibly stupid machines. All they understand are zeros and ones!



Is it that simple? What happens in between?

- Computers are incredibly stupid machines. All they understand are zeros and ones!
- Microprocessors have logic circuits to perform basic mathematical operations such as (ADD, MUL, SUB etc) on binary numbers.

Is it that simple? What happens in between?

- Computers are incredibly stupid machines. All they understand are zeros and ones!
- Microprocessors have logic circuits to perform basic mathematical operations such as (ADD, MUL, SUB etc) on binary numbers.
- Each operation consists a set of "instructions" which the processor performs

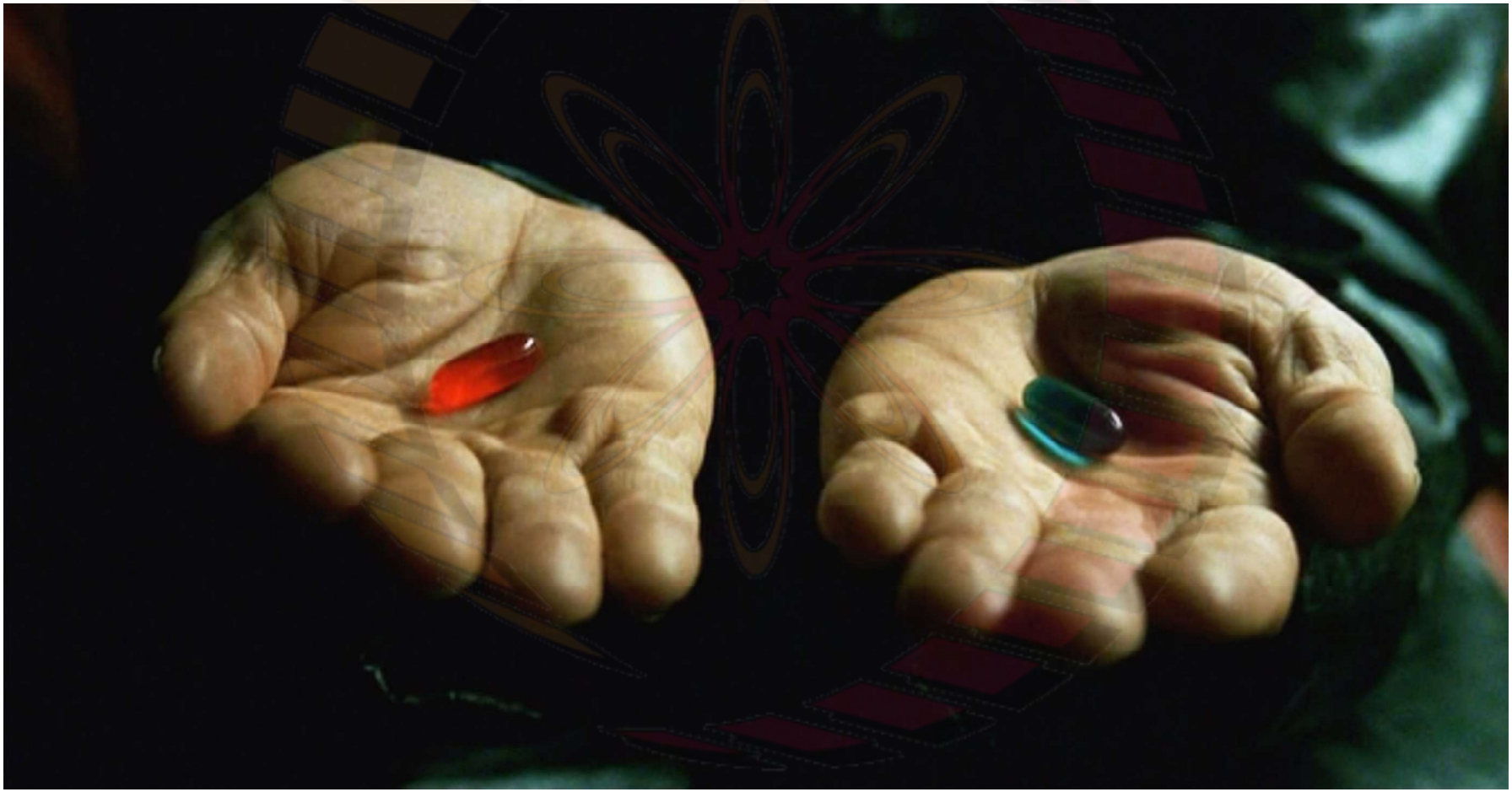
Is it that simple? What happens in between?

- Computers are incredibly stupid machines. All they understand are zeros and ones!
- Microprocessors have logic circuits to perform basic mathematical operations such as (ADD, MUL, SUB etc) on binary numbers.
- Each operation consists a set of "instructions" which the processor performs
- Our calculations are broken down into thousands (or even millions) of such instructions which are then executed.

Is it that simple? What happens in between?

- Computers are incredibly stupid machines. All they understand are zeros and ones!
- Microprocessors have logic circuits to perform basic mathematical operations such as (ADD, MUL, SUB etc) on binary numbers.
- Each operation consists a set of "instructions" which the processor performs
- Our calculations are broken down into thousands (or even millions) of such instructions which are then executed.
- All these operations are basically hidden from the User.

So let's dive into the Rabbit hole: Blue pill or Red Pill?



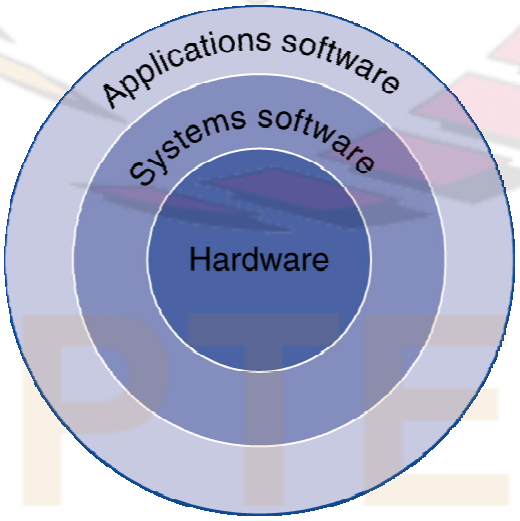
How many of you recognise this?

How many of you recognise this?

A Punch card !! Image: Wikipedia

Computer System Layers

- Application software :Written in high-level language
- System software
- Compiler: translates HLL code to machine code
- Operating System: service code
  - Handling input/output
  - Managing memory and storage
  - Scheduling tasks & sharing resources
- Hardware
  - Processor, memory, I/O controllers



Abstractions

- Abstraction helps us deal with complexity
- Hide lower-level detail.
- Instruction set architecture - ISA: An abstract interface between the hardware and the lowest level software of a machine
  - Encompasses all the information necessary to write a machine language program that will run correctly, including instructions, registers, memory access, I/O
  - Eg. x86, x86_64, IA-32, ARM, SPARC, Motorola 68K, MIPS
- ABI (application binary interface): The user portion of the instruction set plus the operating system interfaces used by application programmers
  - Defines a standard for binary portability across computers

Credit: Based on notes from EC232 UMass – Amherst