ECE Computing Primer

Neil Banerjee

neil@nus.edu.sg

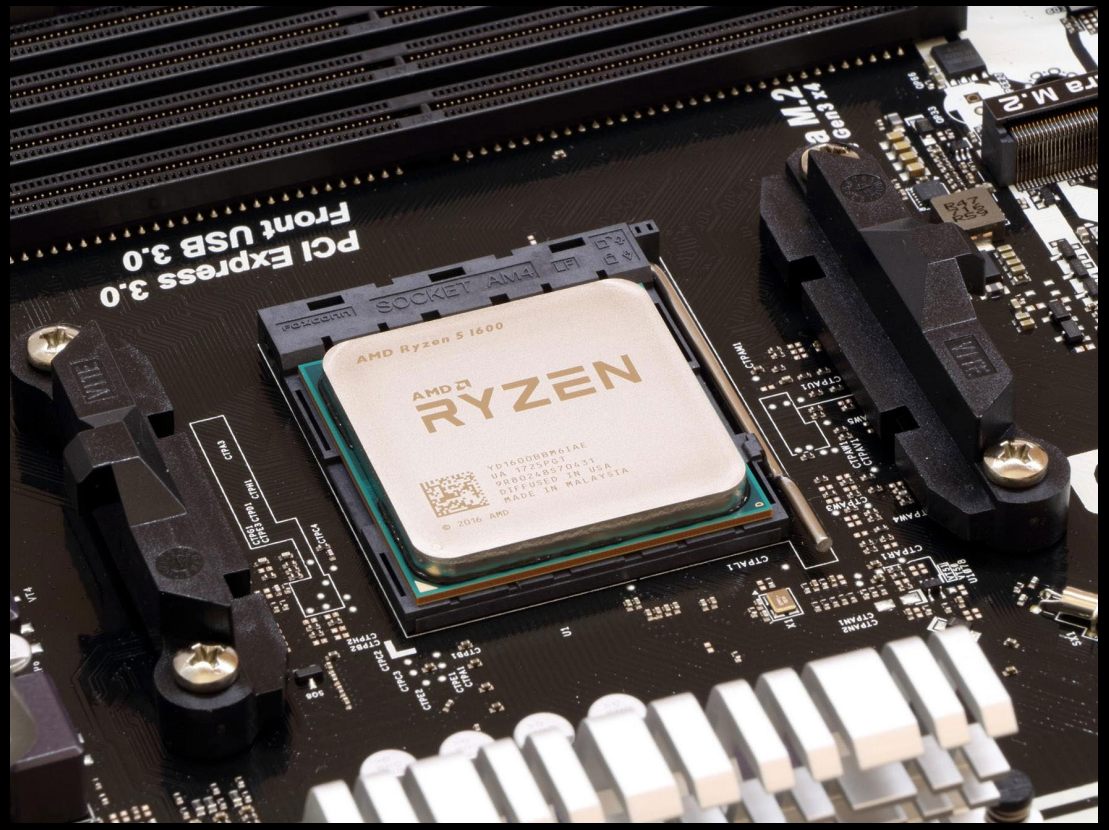
Topics to cover

- What's inside a computer?
- What's a file? A folder?
- What do I use to write code?
- How does code run on my computer?
- Why doesn't my code work?
- How can I use the terminal and command line?

What's inside a computer?

A quick overview of the innards of our favourite tool

CPU: quick mafs



- Main computational device
- Fetches and executes programs
- Very good at executing tasks serially (i.e. one after another)
- Can’t do too many things at once

Memory: remember me

- Used to store programs, files
- Primary memory: temporary working memory
- Secondary memory: permanent storage

GPU: pretty pictures

- Used to produce visuals
- Cannot execute programs without CPU
- Very good at computing many things at once
- Not very good at general computing

Operating System:

- Software that manages your computer and lets you use programs
- Windows, macOS, Android, Ubuntu, etc.

What's a file?

A brief look at how the file system works

What's a file?

- File: a single piece/unit of data
- E.g. one image, one song, one spreadsheet
- Filename extension (.png, .txt, .csv etc) used to indicate the type of data
  - Extension DOES NOT CHANGE the file itself
  - Simply tells the computer what to expect


CAT


DO
G

What's a folder?

- Folder: can hold multiple files
- E.g. "Music" folder holds many songs
- Folders can contain folders
- "Folders" also called "directories"

What's a path?

- Path: the "address" of a file or folder
- Different operating systems organise files and folders differently, have different types of paths

The Windows file system

- Every storage device is assigned a letter
  - The device with Windows is always C:
  - Other devices are lettered D:, E:, … - can be changed
- Storage devices contain files and folders
- C: contains:
  - Program Files, Program Files (x86)
  - Users
  - Windows
  - Others created by you/your programs

Paths in Windows

- Paths use '\' to indicate levels of hierarchy
  - A '\' indicates that we go inside the device/folder
  - E.g. C:\Users\neilb\Downloads
- Paths can lead to folders or files
  - C:\Users\neilb\Downloads – path to folder
  - C:\Users\neilb\Downloads\computing.pptx – path to file

The Mac file system

- UNIX-like file system
- ‘/’ is the root of the file system
- ‘/Volumes/’ is where all storage devices live
- Paths use ‘/’ to indicate levels of hierarchy

Demo

What do I use to write code?

An overview of the tools used to write, debug and run code

Text editors

- As name implies: create, edit text files
- All code is written as a text file
- E.g. Notepad, TextEdit, Notepad++, gedit etc.

Integrated Development Environments

- A set of tools to help you write, debug, run and deploy code
- A text editor may be part of an IDE
- E.g. IDLE, PyCharm, Arduino IDE, STM32Cube

How does code really run?

The basic ideas of how code is read and run

C/C++ code is compiled to run

- Compiler takes our code, turns it into machine language
- Arduino IDE – click “Verify”
- C, C++, Rust, Go are all compiled

Python code is interpreted

- Program is read by an interpreter line by line
- Interpreter performs the instructions in the program
- Same code can run on any computer
- Python, JS, BASIC are interpreted

Why doesn't my code work?

An ultra-quick tutorial on debugging

Where is the problem?

- Running a program is complex – lots of moving parts
- Different kinds of errors can result from different parts failing
- We will cover two kinds of errors:
  - Syntax errors
  - Runtime errors

Syntax errors

- Programming languages are very strict with grammar
- Small mistakes → program won't run
- Easiest to detect – IDE (or text editor) is your friend!
  - Compiled languages: code won't compile
  - Interpreted languages: interpreter will crash w/error
- Read the documentation to figure out what's wrong

Runtime errors

- Program runs, but wrong output/crash
- Harder to detect, much harder to solve :(
- Lots of tools in our arsenal, must use them all:
  - Flowcharts
  - Pseudocode
  - Test cases
  - Debugger

Flowcharts

- Represent programs visually
- Think about logic needed
- Drawing by hand easiest
- app.diagrams.net (draw.io), MS Visio

Pseudocode

- Write programs in somewhat plain English
- Think about logic, not language
- No single standard

Test cases

- Code should be tested for correctness
- Use many test cases to check your code
- Check corner, edge cases; make sure always correct
  - Extreme values, unexpected but possible inputs, etc.

Debuggers

- The most powerful tool in your arsenal
- Run code line by line, pinpoint where bugs lie
- Main idea: Breakpoints
  - Program pauses when a breakpoint is found
  - Choose how to proceed:
    - Step Over: Execute current line and proceed to the next
    - Step In: Enter the function called by the current line
    - Step Out: Exit the function, go back to where it was called
- Setting up debugger varies by language, IDE
  - Above ideas are universal
  - Use favourite search engine/AI to learn about specific IDEs

Demo