My personal introduction to computer gaming started in 1985, when my parents bought a MSX-1 Home Computer. I was fascinated by games such as Konami's *Knightmare* and *Nemesis 2*. It was not only the gameplay that interested me, but also how such games are developed. That's how I started my interest into programming and game development.

The same year I had my first home computer, Nintendo released a game called *Super Mario Bros.* on the Nintendo Entertainment System (NES). It was an instant blockbuster; it combined great graphics with smooth side scrolling. Side scrolling games from that time period, like Knightmare and Nemesis 2, moved at constant and "choppy" speed.

Super Mario Bros. was different, as the player dictates the scrolling speed. You could smoothly accelerate from walk to run or jump, and the screen would smoothly follow your actions. Super Mario Bros. was immensely successful, both commercially and critically. It helped popularize the side-scrolling platform game genre, and served as a killer app for the NES[1].

Super Mario Bros. demonstrated the real power of the NES, which was hardware-supported scrolling. Most computers of that era, such as the MSX and Commodore 64 systems, lacked hardware support for scrolling. To achieve side-scrolling on these platforms, one needed to move all the characters (typically represented by 8x8 pixel), resulting in the super choppy scrolling effect often seen. The only way to produce smooth pixel scrolling was to redraw the entire screen, offset by the desired number of pixels. This approach was incredibly performance intensive and beyond the capabilities of most hardware at the time.

[1]Upon release in Japan, 1.2 million copies were sold during its September 1985 release month. Within four months, about 3 million copies were sold in Japan

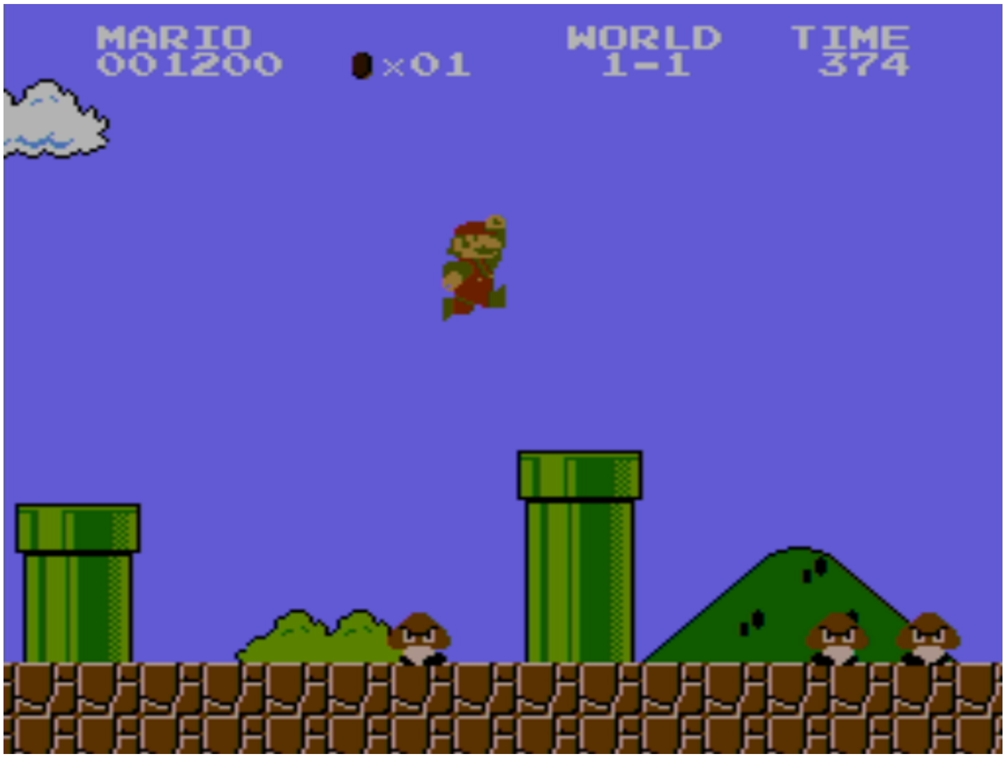

Figure 1: Super Mario Bros. on Nintendo Entertainment System

The NES was one of the very first home computers that supported smooth scrolling. Essentially, the hardware had a register you could just write to set the fine (pixel) scroll. If you want your background to be displayed scrolled 120 pixels in from the right, and 22 pixels from the top, you just write "120" and then "22" in order, to the same register. Done deal! The video chip takes care of the rest, running at the same constant speed as it always done.

The IBM PC was by late 80s far behind the gaming power of the NES. It was designed for office work rather than gaming. It was meant to crunch integers and display static images for word processing and spreadsheet applications. Most PC games around that time are graphic adventure games (*King's Quest*), static platform games (*Prince of Persia*) and simulation games (*Sim City*). Basically, the PC lacked all hardware support for sprites and smooth scrolling.

Then, on December 14th, 1990, a small unknown software company called "Ideas from the Deep" released *Commander Keen in Invasion of the Vorticons* for the IBM PC. It was the first smooth side-scrolling game on a PC, similar like Super Mario Bros on the NES.

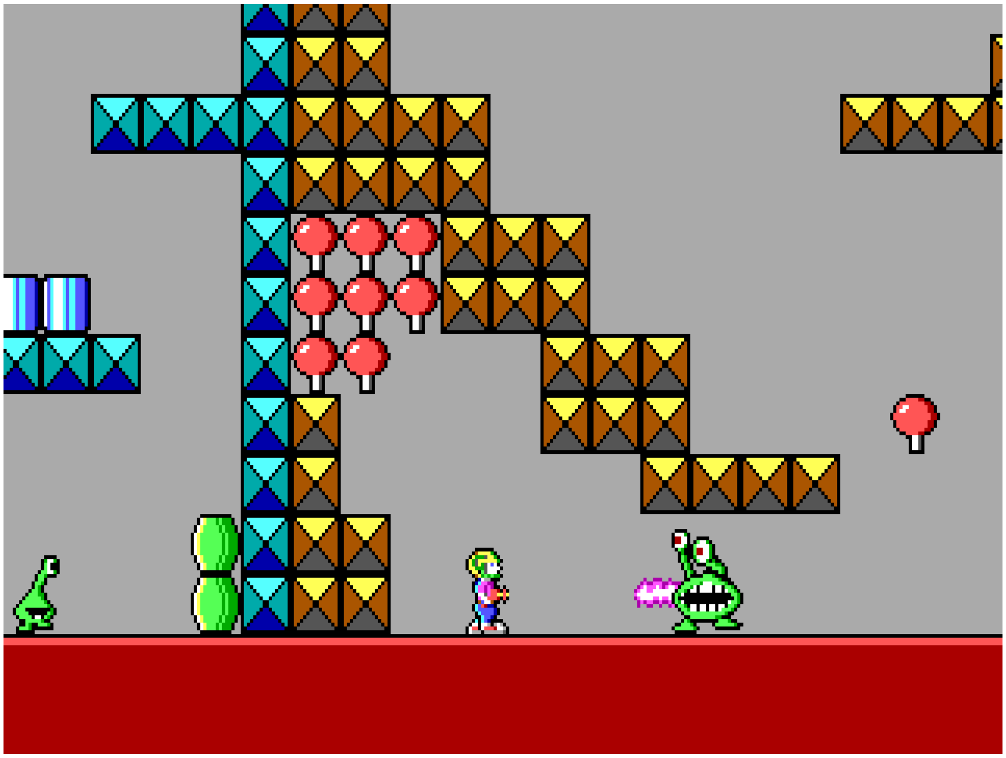

Figure 2: Commander Keen in Invasion of the Vorticons

How was this possible on the IBM PC? Many obstacles had to be overcome:

- The first 8086 CPU did not outperform the average home computer in terms of raw power. Only with the release of the 286 CPU the PC started to outperform the market in terms of raw power.
- As stated before, the video system (called EGA) did not support any form of scrolling. It did not even support any form of sprites, which allowed movement of something on the screen by simply updating its `(x,y)` coordinates.
- The video system could not double buffer. It was not possible to have smooth scrolling without ugly artifacts called "tears" on the screen.
- The PC Speaker, the default sound device, could only produce square waves resulting in a bunch of "beeps" which were more annoying than anything else.
- The audio ecosystem was fragmented. Each of the various sound systems had different capabilities and expectations

- The RAM addressing mode was not flat but segmented, resulting in complex and error prone pointer arithmetic.
- The bus was slow and I/O with the VRAM was a bottleneck. It was next to impossible to write a full framebuffer at 70 frames per second

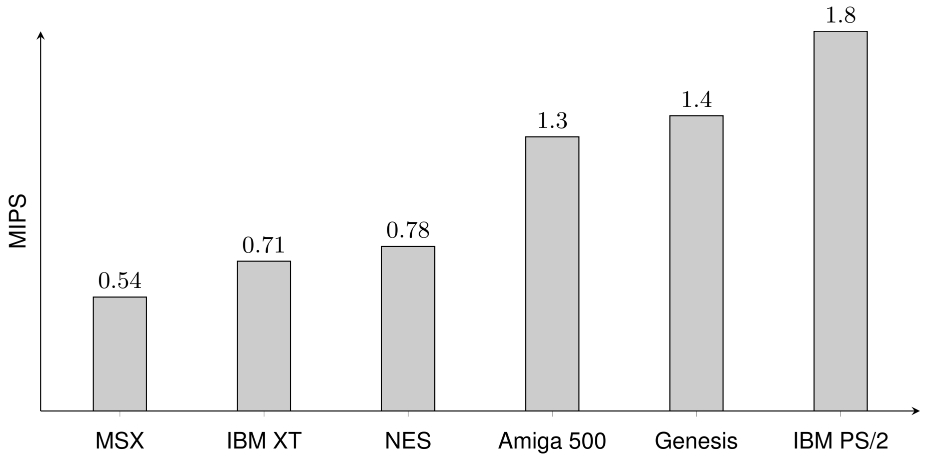

Figure 3: Consoles[2]vs PC, CPU comparison with MIPS[3,4].

Overall, it seemed impossible to create any reasonable side-scrolling game on the PC platform. But many around the world did not accept that and tinkered with the hardware to achieve unexpected results. Their ingenuity is what inspired me to write this book.

I've chosen to divide this book into three chapters:

- Chapter 2: The Hardware. An exploration of PC components from 1990.
- Chapter 3: The tools and assets. Which tools are used for game development and how are assets created and structured.
- Chapter 4: The Software. A deep dive into the Commander Keen game engine.

By first showing the hardware constraints, I hope programmers will develop an appreciation for the software and how it navigated obstacles, sometimes turning limitations into advantages.

[2]The MSX uses a Zilog Z80 running at 3.6MHz. The Amiga 500 and Genesis have a Motorola 68000 CPU respectively running at 7.16 MHz and 7.6 MHz. The NES uses a Ricoh 2A03 CPU running at 1.8 MHz.

[3]Million Instructions Per Second.

[4]Gamicus Fandom: https://gamicus.fandom.com/wiki/Instructions_per_second.

This book focuses on *Commander Keen in Keen Dreams*, which is developed after the first three releases in the series. The reason for this choice is straightforward: it is the only version with publicly released source code. Where relevant, I will also highlight technological differences across the versions of Commander Keen. However, code examples will be limited to Keen Dreams, due to the availability of the source material.

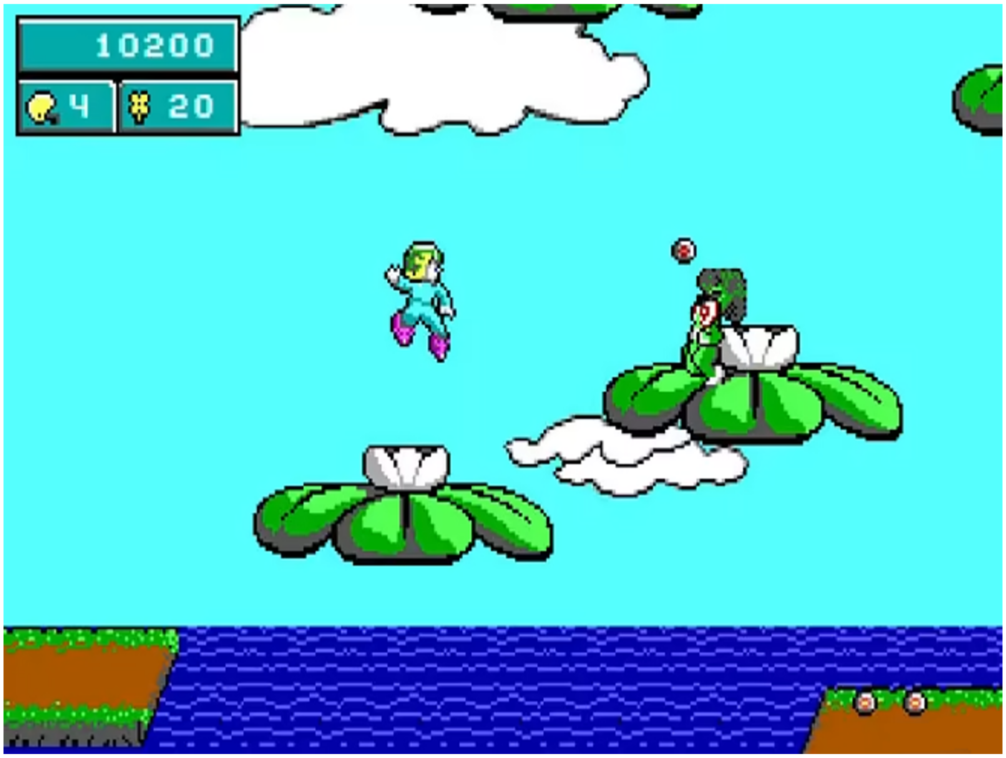

Figure 4: Commander Keen in Keen Dreams