FIT 1047

Introduction to computer systems, networks and security

MONASH University

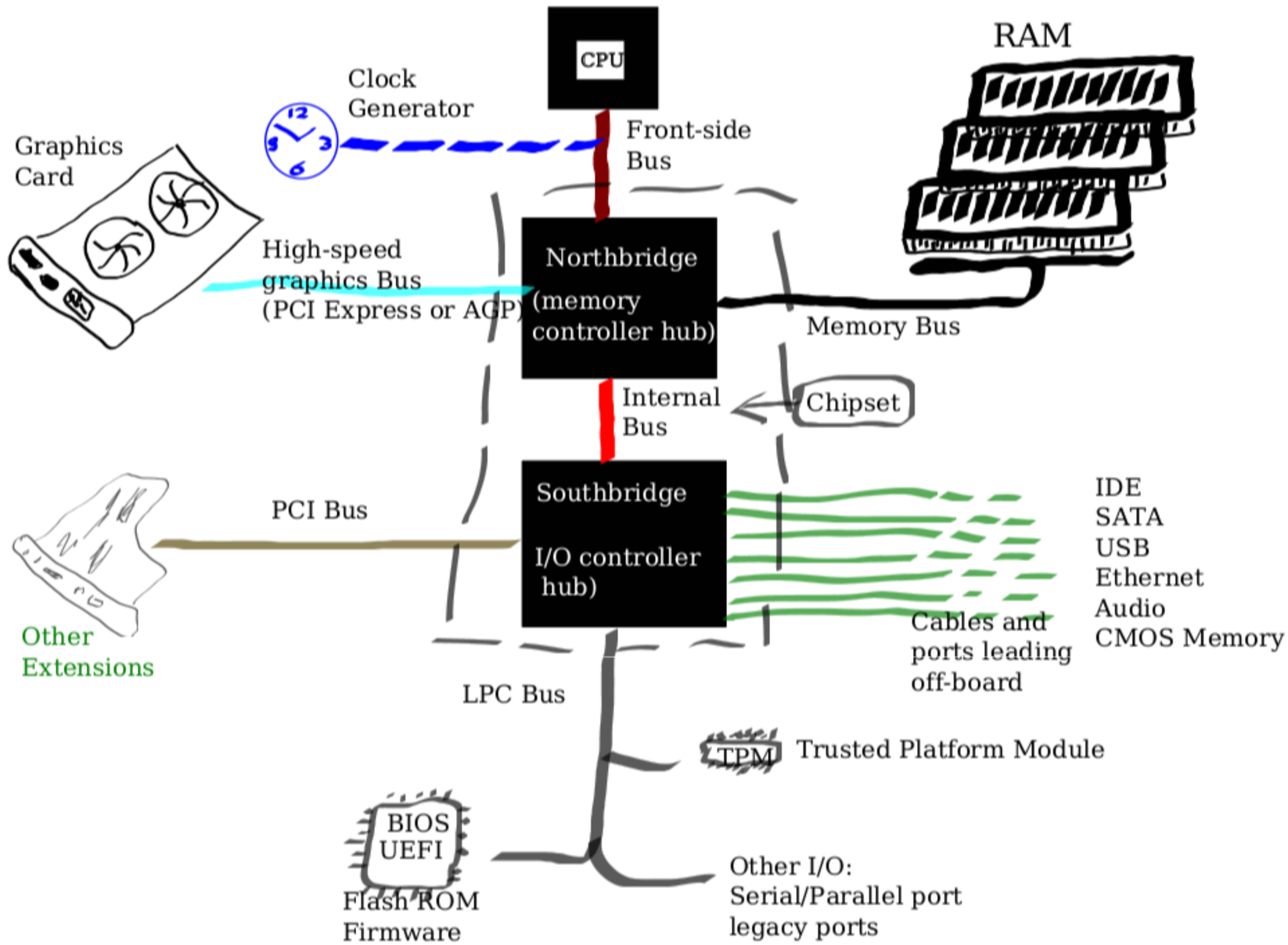
Overview

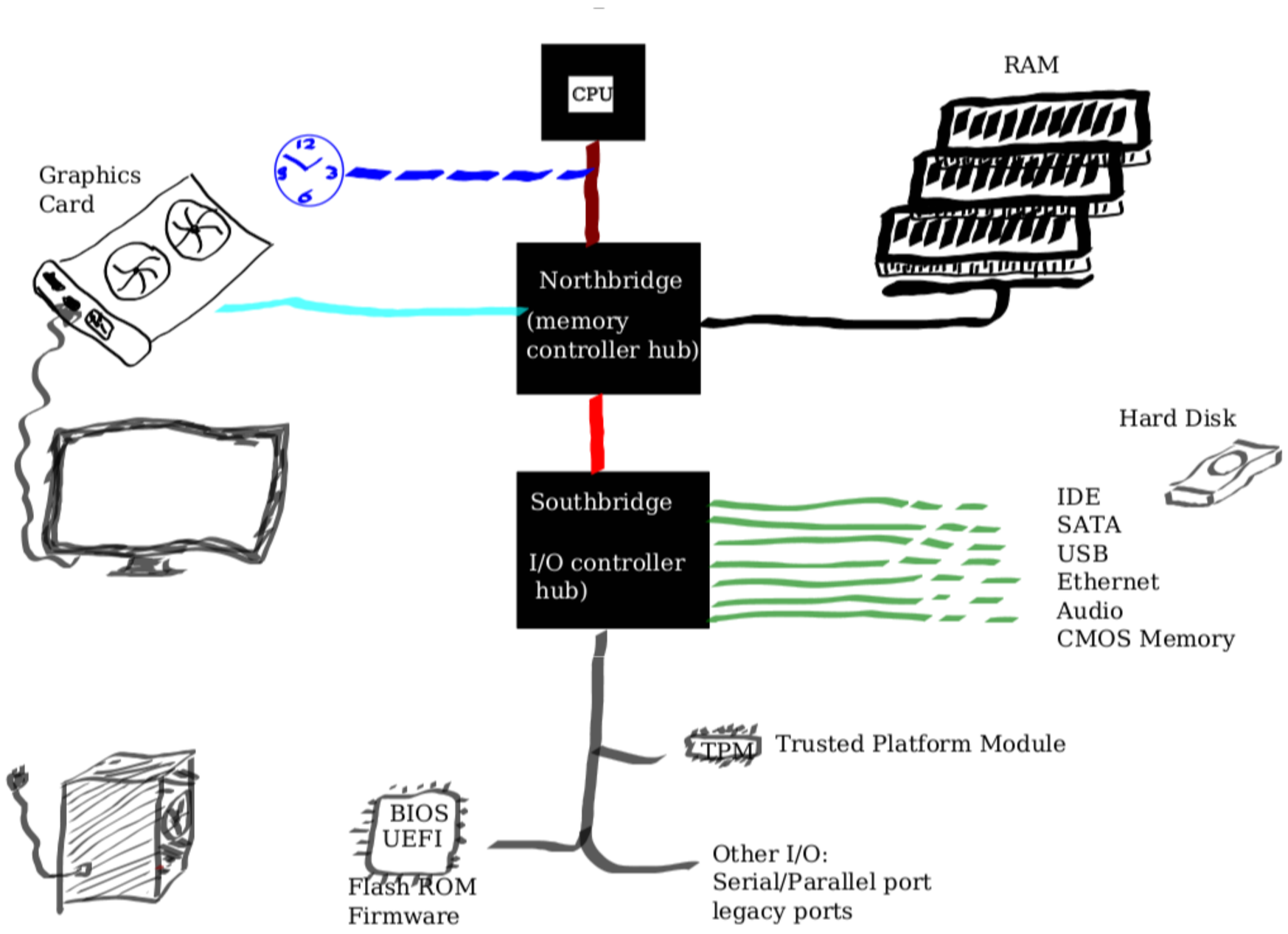
- PC boot sequence
- BIOS / UEFI

Compare JumpI and Jump

```
MAR <- PC
IR <- M[MAR]
PC <- PC+1
Decode
MAR <- IR[11-0]
MBR <- M[MAR]
PC <- MBR
```

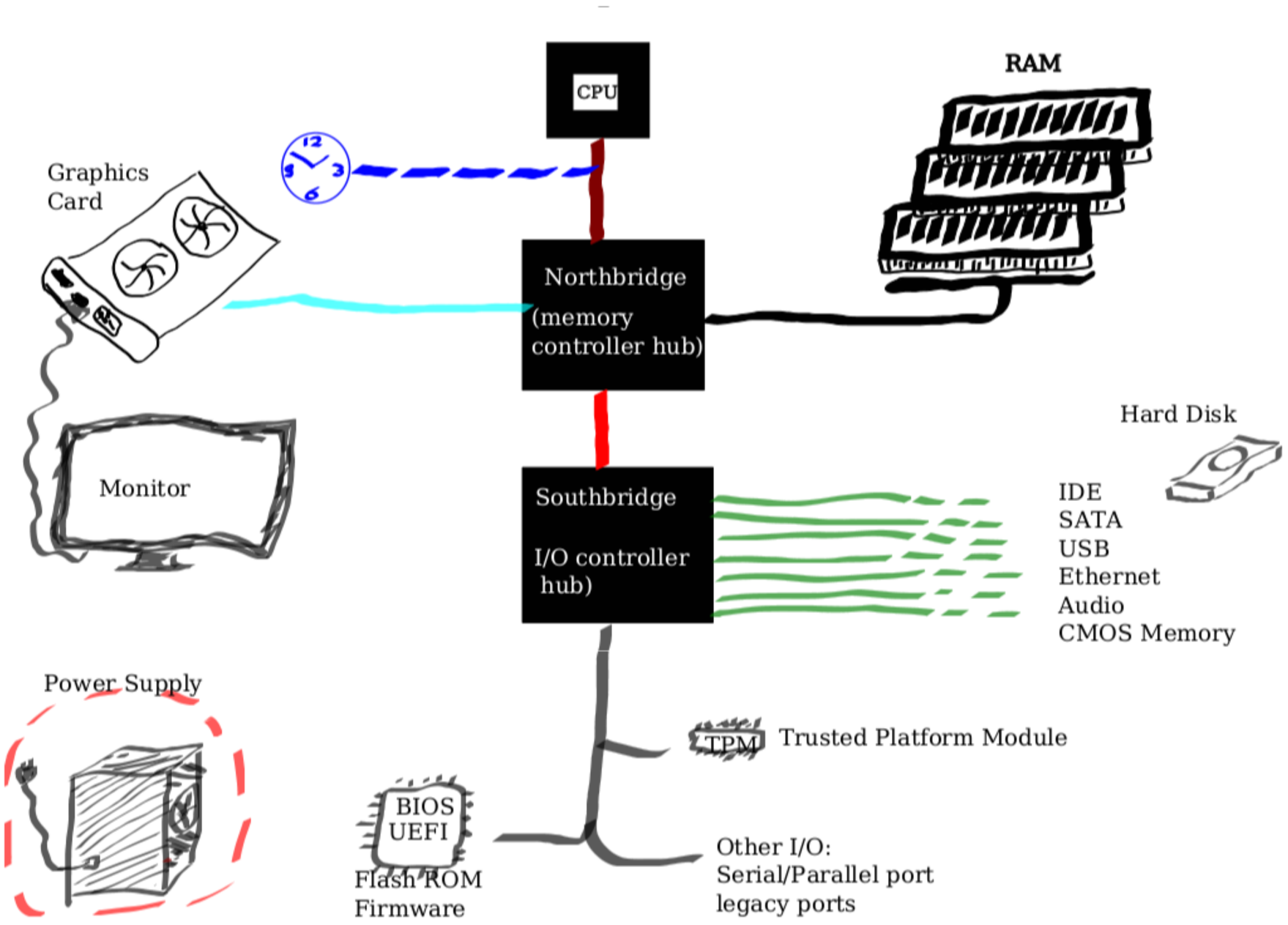
```
MAR <- PC
IR  <- M[MAR]
PC <- PC+1
Decode
PC <- IR[11-0]
```

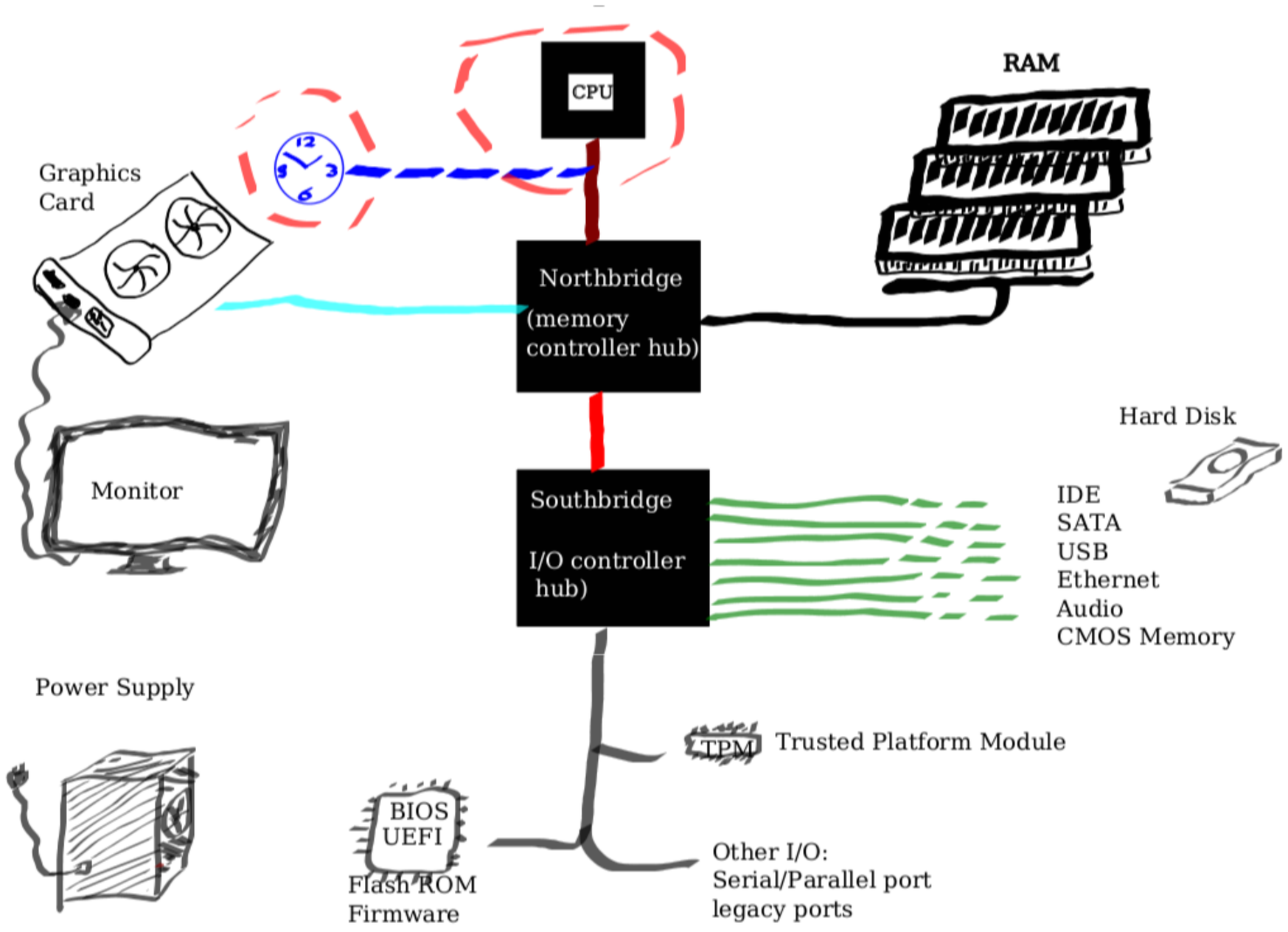

CPU
RAM
Clock
Generator
Front-side
Bus
Graphics
Card
High-speed
graphics Bus
(PCI Express or AGP)
Northbridge
(memory
controller hub)
Memory Bus
Internal
Bus
Chipset
Southbridge
I/O controller
hub)
PCI Bus
IDE
SATA
USB
Ethernet
Audio
CMOS Memory
Cables and
ports leading
off-board
Other
Extensions
LPC Bus
TPM
Trusted Platform Module
BIOS
UEFI
Flash ROM
Firmware
Other I/O:
Serial/Parallel port
legacy ports


CPU
RAM
Graphics
Card
Northbridge
(memory
controller hub)
Hard Disk
Southbridge
I/O controller
hub)
IDE
SATA
USB
Ethernet
Audio
CMOS Memory
TPM
Trusted Platform Module
BIOS
UEFI
Flash ROM
Firmware
Other I/O:
Serial/Parallel port
legacy ports

Boot process: Turn power on

- Power supply starts and provides energy to the motherboard and other components in the computer. Components should only really start to work after a stable power level is established.
- A power good signal can be sent to the motherboard which triggers the timer chip to reset the processor and start clock ticks.


CPU
RAM
Graphics
Card
Northbridge
(memory
controller hub)
Monitor
Hard Disk
Southbridge
I/O controller
hub)
IDE
SATA
USB
Ethernet
Audio
CMOS Memory
Power Supply
TPM
Trusted Platform Module
BIOS
UEFI
Flash ROM
Firmware
Other I/O:
Serial/Parallel port
legacy ports

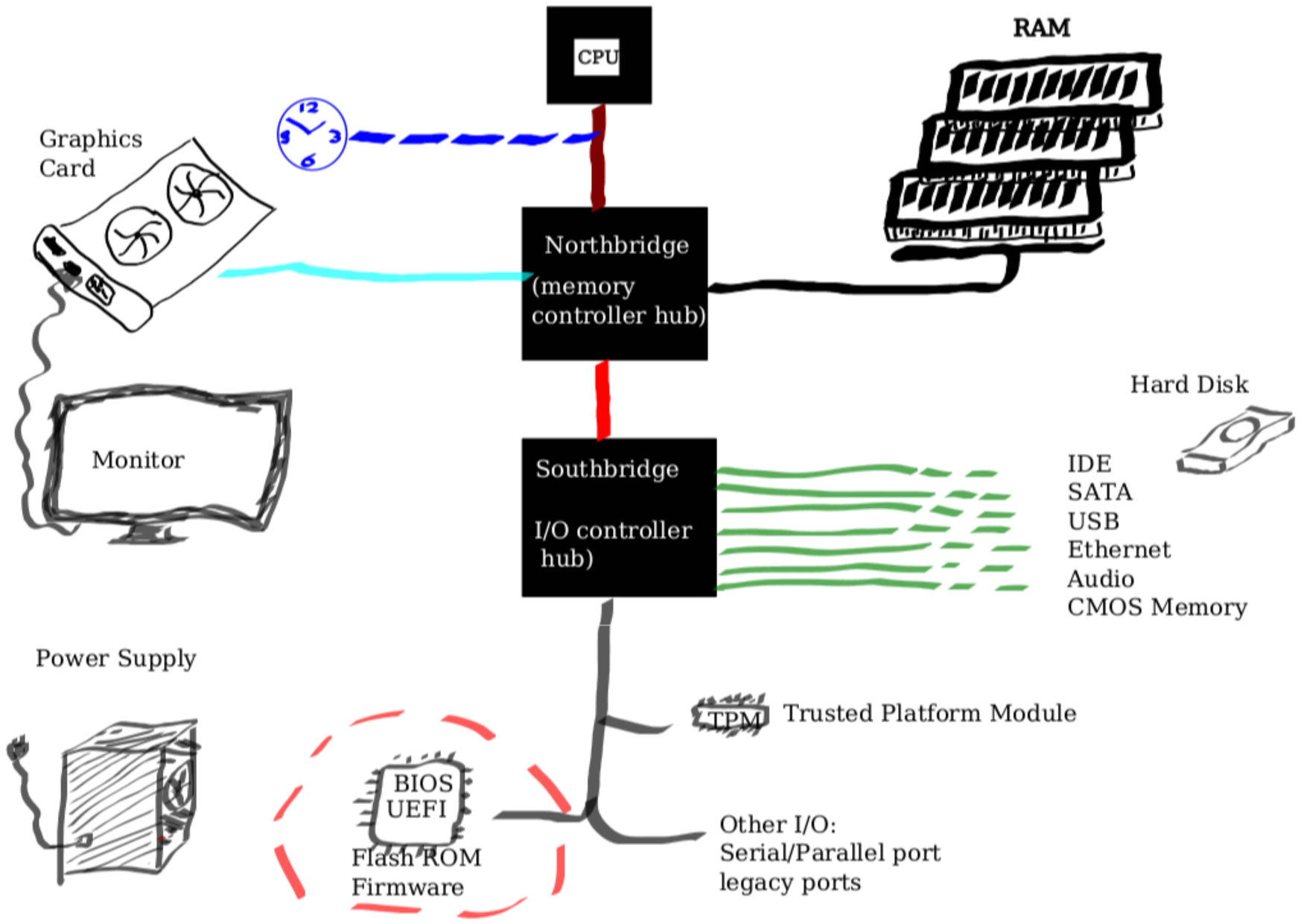

CPU
RAM
Graphics
Card
Northbridge
(memory
controller hub)
Hard Disk
Monitor
Southbridge
I/O controller
hub)
IDE
SATA
USB
Ethernet
Audio
CMOS Memory
Power Supply
TPM
Trusted Platform Module
BIOS
UEFI
Flash ROM
Firmware
Other I/O:
Serial/Parallel port
legacy ports

Boot process: Processor starts

- First the main fan starts (or other cooling) and motherboards clock-cycle starts.
- CPU gets power and starts working.
- CPU cannot do much without software.

When a computer starts it is really not able to do much. The CPU does not know about a hard-disk an operating system might sit on or any peripherals connected to the computer (mouse, keyboard, monitor, hardware, DVD-drive, printer, etc.).

Booting means to load software step-by-step and activate all components one step after the other.


CPU
RAM
Graphics
Card
Northbridge
(memory
controller hub)
Monitor
Hard Disk
Southbridge
I/O controller
hub)
IDE
SATA
USB
Ethernet
Audio
CMOS Memory
Power Supply
TPM
Trusted Platform Module
BIOS
UEFI
Flash ROM
Firmware
Other I/O:
Serial/Parallel port
legacy ports

Boot process: Initial software

- BIOS (Basic Input Output System or first steps of UEFI Unified Extensible Firmware Interface in modern PCs) is stored in non-volatile memory (ROM - read only memory) on the motherboard.
- It controls the start-up steps, provides initial system configuration (power saving, security, etc.), and initially configures accessible hardware.

Boot process: Initial software

- The reset command in the CPU triggers the execution of an instruction at a specific location in the BIOS chip.
- Location contains a Jump instruction that points to the actual BIOS start-up program in the chip.
- Booting really starts with the execution of this start-up program.

Boot process: POST

BIOS starts with a power-on-self-test POST:

- System memory is OK
- System clock / timer is running
- Processor is OK
- Keyboard is present
- Screen display memory is working
- BIOS is not corrupted

Results of POST can only be communicated through system beep.

Boot process: Video card

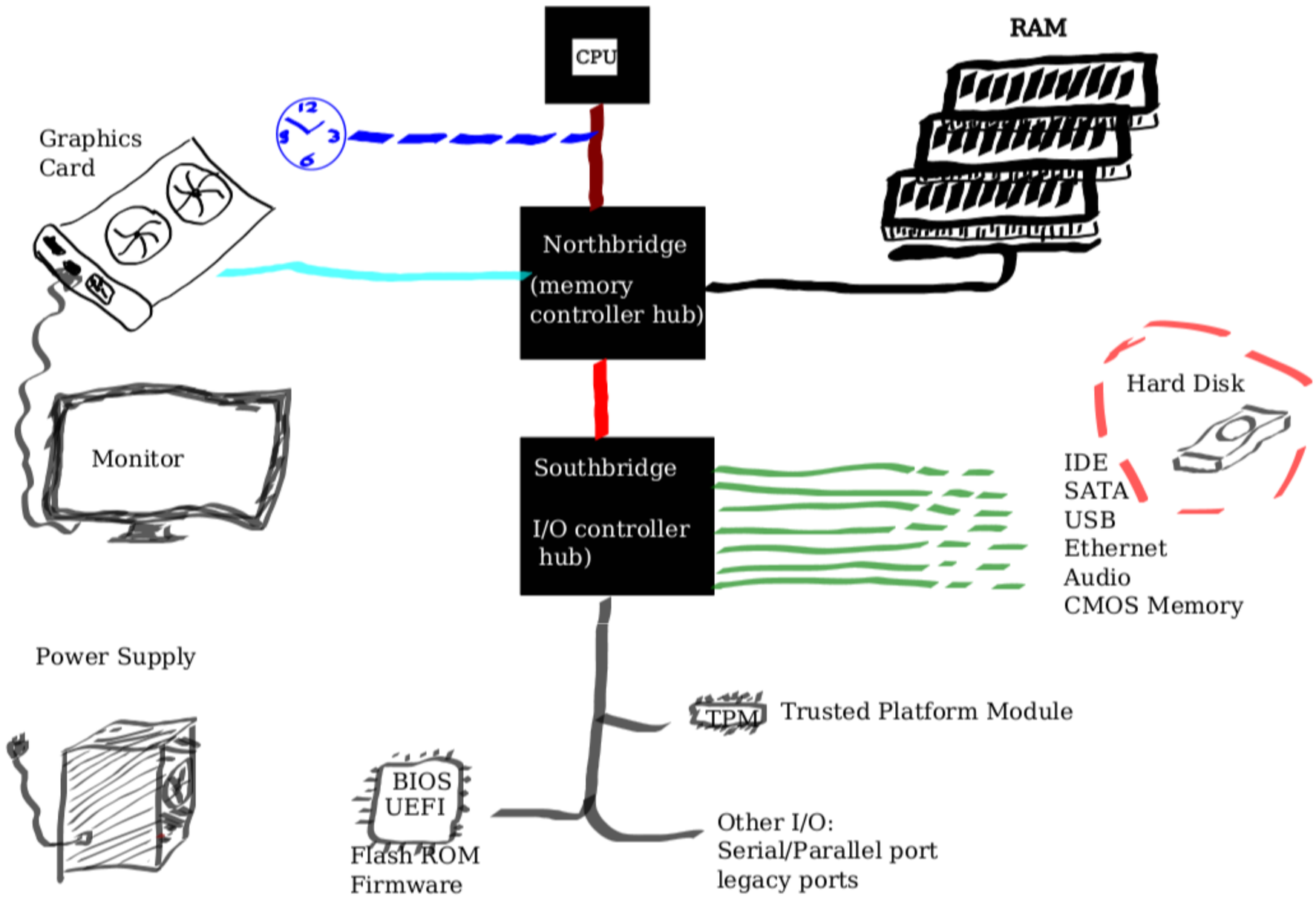
- The first thing after a successful POST is to initialize the video card and show some initial message on the screen.
- Note that the BIOS can only do a rudimentary initialization. Use of 3D, fancy graphics, etc. needs additional software, the so-called driver.

Boot process: Other hardware

- Then, the BIOS goes through all available hardware and initializes as far as possible without more complex driver software (UEFI has more options)
- Examples are type and size of hard-disk, DVD drive, timing of RAM (random access memory) chips, networking, sound, etc.

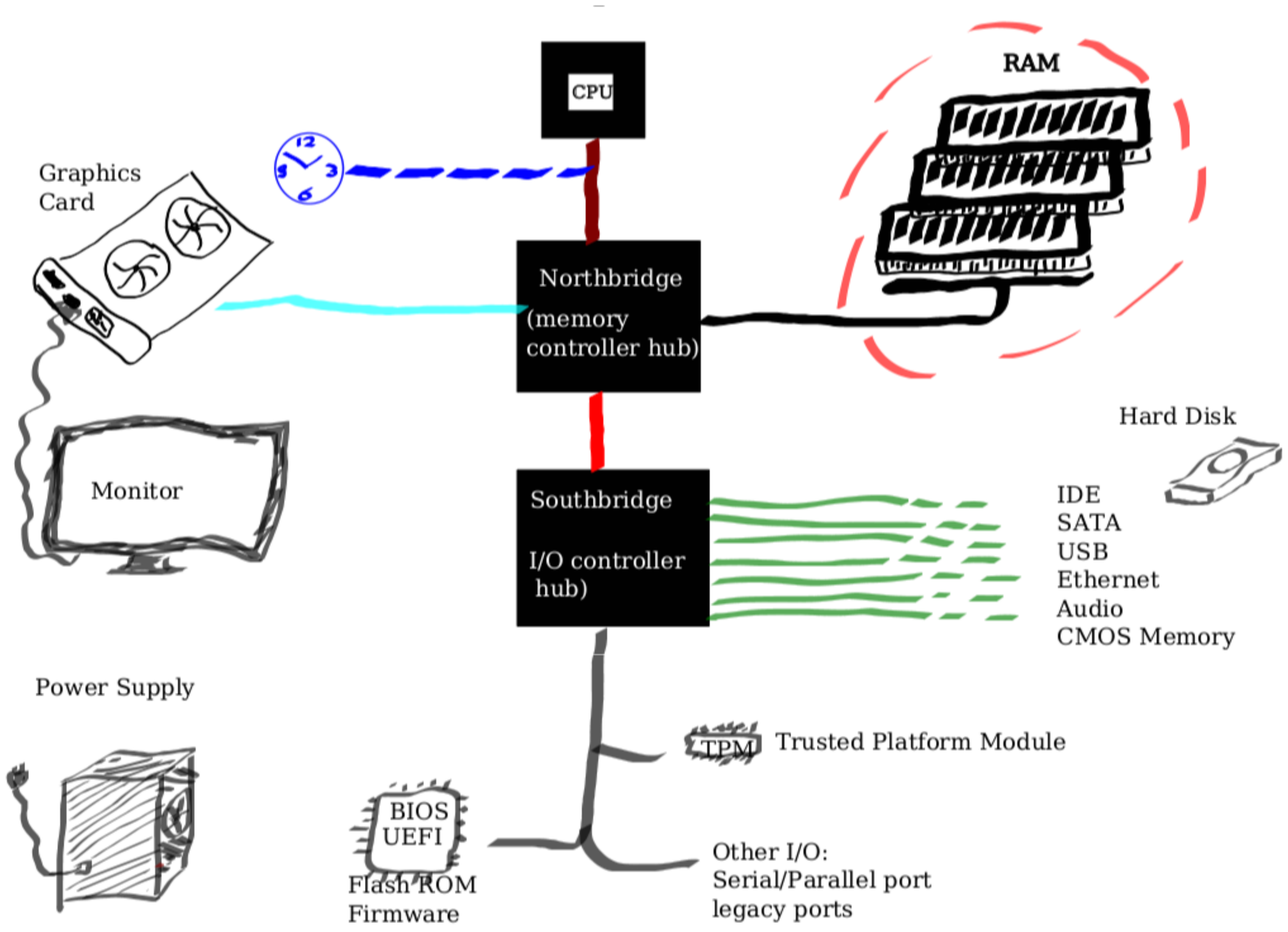
Boot process: Find Operating System

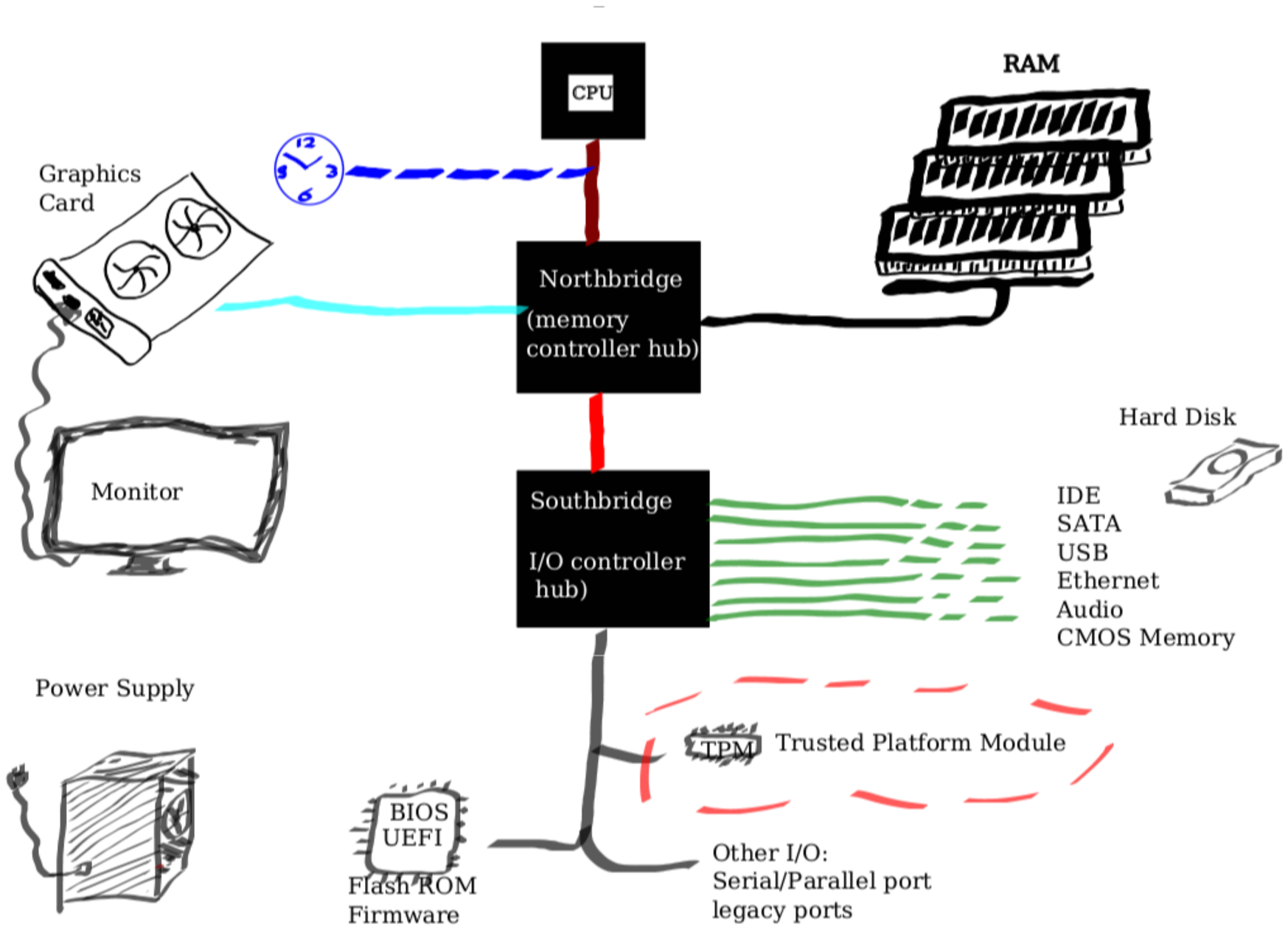
- BIOS needs to look for a bootable drive
- Can be on a hard-disk, USB Stick, DVD, floppy disk,...
- Order is defined in BIOS configuration (usually accessible by holding a particular key while start-up screen is shown).


CPU
RAM
Graphics
Card
Northbridge
(memory
controller hub)
Monitor
Hard Disk
Southbridge
I/O controller
hub)
IDE
SATA
USB
Ethernet
Audio
CMOS Memory
Power Supply
TPM
Trusted Platform Module
BIOS
UEFI
Flash ROM
Firmware
Other I/O:
Serial/Parallel port
legacy ports

Boot process: Boot sector

- On a bootable drive, there needs to be a boot sector with code to be executed (boot loader). On a hard disk, this information is in the Master Boot Record (MBR).
- The boot loader first loads the core part of the operating system, the kernel. Then it loads various device drivers (e.g. for the graphics card).
- Once all drivers are loaded, the Graphical User interface is started and personal settings are loaded.
- The computer is ready to use.


CPU
RAM
Graphics
Card
Northbridge
(memory
controller hub)
Hard Disk
Monitor
Southbridge
I/O controller
hub)
IDE
SATA
USB
Ethernet
Audio
CMOS Memory
Power Supply
TPM
Trusted Platform Module
BIOS
UEFI
Flash ROM
Firmware
Other I/O:
Serial/Parallel port
legacy ports


CPU
RAM
Graphics
Card
Northbridge
(memory
controller hub)
Hard Disk
Monitor
Southbridge
I/O controller
hub)
IDE
SATA
USB
Ethernet
Audio
CMOS Memory
Power Supply
TPM
Trusted Platform Module
BIOS
UEFI
Flash ROM
Firmware
Other I/O:
Serial/Parallel port
legacy ports

BIOS versus UEFI

- BIOS is outdated. Was intended to be an abstraction layer to access I/O, but it is no longer used for this.
- Restricted to 1,024 kilobytes of space, only works with hard-drives up to 2.2 terabyte space, cannot work with lots of current technology (and future technology)
- UEFI does not really replace the firmware. It sits between firmware and operating system. Like a mini-operating system.

UEFI

- Can address hard-disks up to 9.4 zettabytes (1 zettabyte is about a billion terabytes).
- Provides access to all hardware. Faster hardware initialization.
- Security and authentication features before the OS has started.
- Network access before the OS has started.

UEFI criticism

- Boot restrictions (i.e. secure boot) can prevent users from installing the operating system of their choice.
- Additional complexity provides additional possibilities for errors and new attack vectors.

Most new PCs now use UEFI.

Some more on secure boot in the security part of the unit.

A little bit on values/sizes/measurements

- What is all this Tera, Giga, Kilo, nano, pico, …?
- And sometimes its powers of 10 and sometimes powers of 2?

Prefix	Symbol	Factor
	Base	$10^0 = 1$
deka-	da	$10^1 = 10$
hecto-	h	$10^2 = 100$
kilo-	k	$10^3 = 1{,}000$
mega-	M	$10^6 = 1{,}000{,}000$
giga-	G	$10^9 = 1{,}000{,}000{,}000$
tera-	T	$10^{12} = 1{,}000{,}000{,}000{,}000$
peta-	P	$10^{15} = 1{,}000{,}000{,}000{,}000{,}000$

Prefix	Symbol	Factor
	Base	$10^0 = 1$
deci-	d	$10^{-1} = 0.1$
centi-	c	$10^{-2} = 0.01$
milli-	m	$10^{-3} = 0.001$
micro-	□	$10^{-6} = 0.000,001$
nano-	n	$10^{-9} = 0.000,000,001$
pico-	p	$10^{-12} = 0.000,000,000,001$
femto-	f	$10^{-15} = 0,000,000,000,000,001$

For Bytes (bits) in powers of 2:

Prefix	Symbol	Factor
Bytes (Bits)	B (bit)	$2^0 = 1$
Kibi-	KiB (Kibit)	$2^{10} = 1024$
Mebi-	MiB (Mibit)	$2^{20} = 1,048,576$
Gibi-	GiB (Gibit)	$2^{30} = 1,073,741,824$
Tebi-	TiB (Tibit)	$2^{40} = 1,099,511,627,776$
Pebi-	PiB (Pibit)	$2^{50} = 1,125,899,906,842,624$

This continues with Exbi-, Zebi, and Yobi,...

A few things the exam will not ask for:

- ASCII table (you need to understand it, but not memorize)
- Unicode
- MARIE programming (you need to be able to understand a MARIE program, but there will be no programming task)
- Names of Boolean Laws (again, you need to understand them)
- Dates and numbers in computer history