Progresivní technologie v informatice II

Vestavné systémy

vlastnosti, využití, požadavky, způsoby návrhu

Fakulta informačních technologií
České vysoké učení technické v Praze

Financováno Evropskou unií
NextGenerationEU
Národní plán obnovy

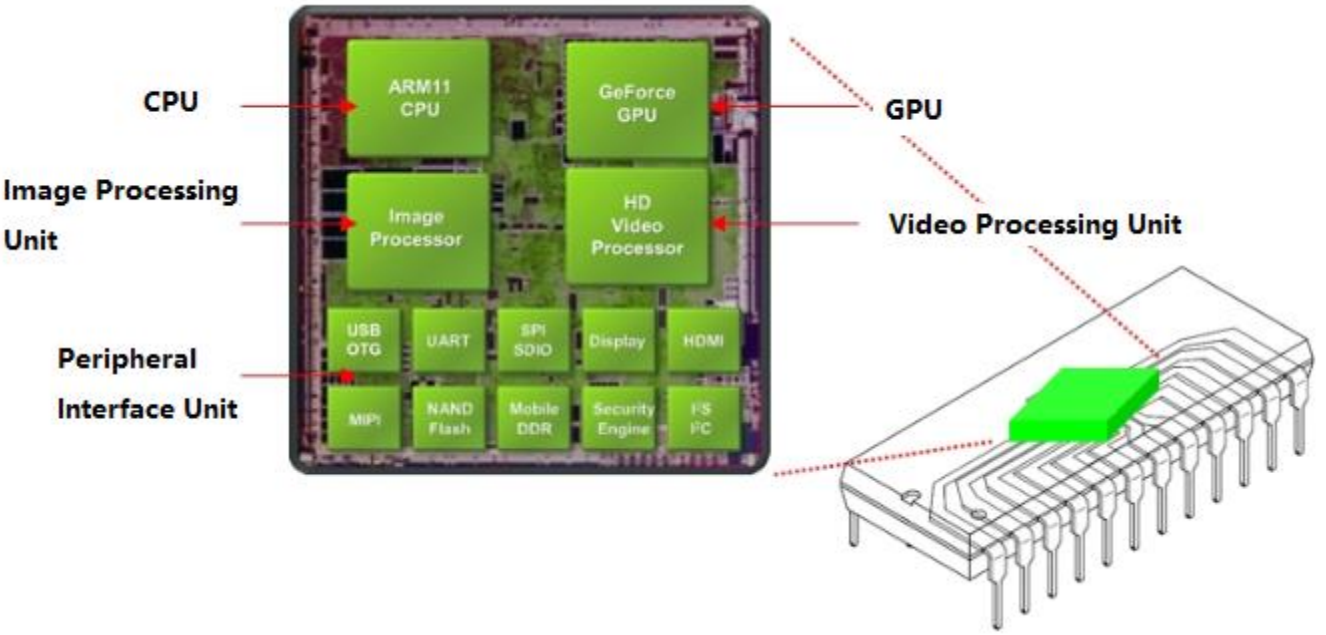
Vestavné/zabudované (angl. „embedded“) systémy

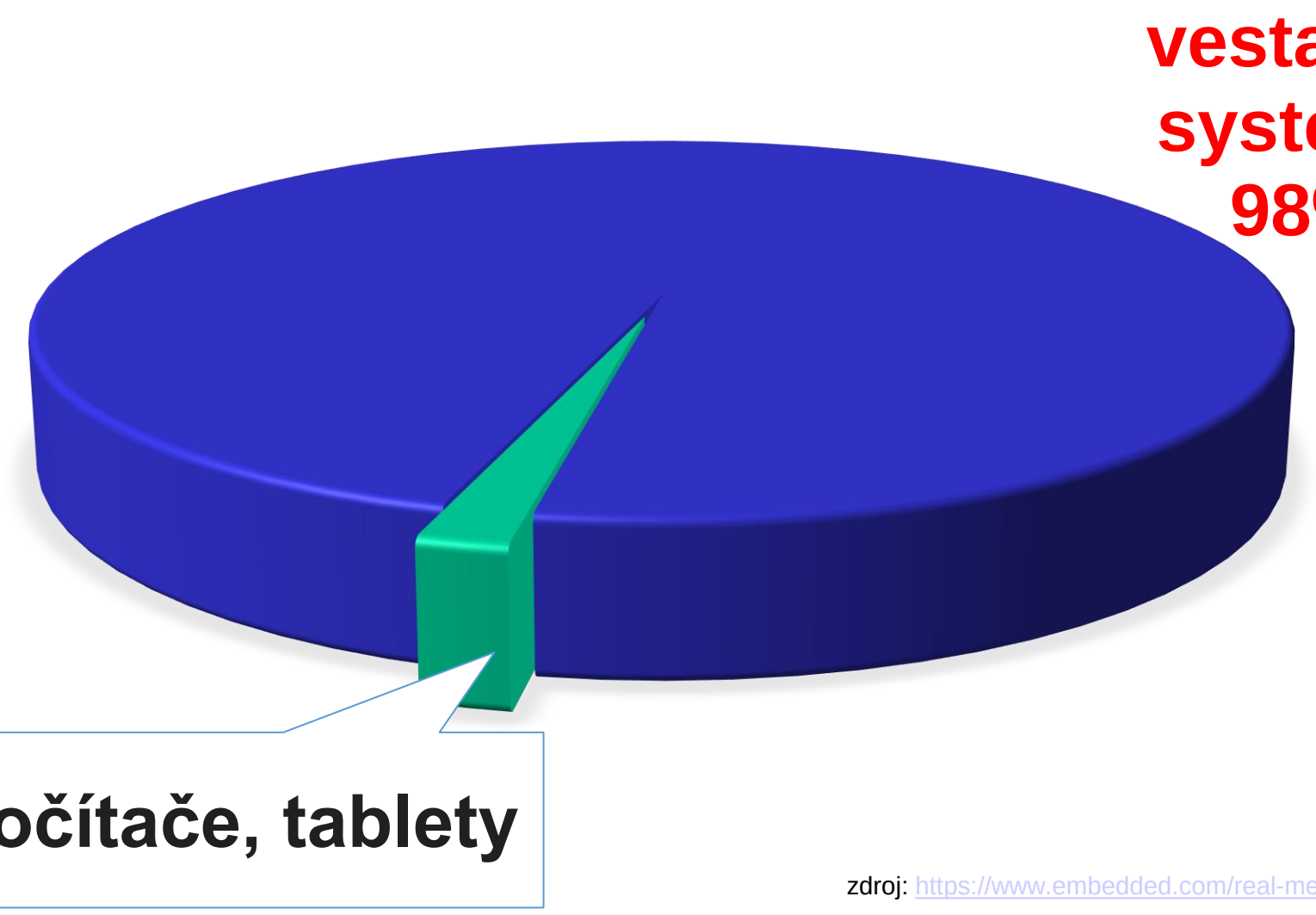
- jsou jednoúčelové počítače, ve kterém je řídicí systém zcela zabudován do zařízení, které ovládá
- na rozdíl od univerzálních počítačů (např. osobní počítače), jsou vestavné systémy specializované na předem definované činnosti

Příklady:

- kalkulačky
- bankomaty
- řídící jednotky v automobilech (pro řízení motoru, brzd, atd.)
- vybavení domácnosti (pračky, myčky, televizory, atd.)
- počítačové periferie (tiskárny, myši, routery, atd.)

- Systém na čipu (angl. System on Chip - SoC)
  - integruje všechny součásti počítače (elektronického systému) do jednoho “integrovaného” obvodu (tzv. čipu)
  - většinou obsahuje procesor (CPU), paměti (ROM, RAM), číslicové, analogové, mixed-signal bloky, digitálně signálové procesory (DSP), periferie
  - nízká spotřeba elektrické energie
  - využívá se především ve vestavných systémech





zdroj: https://www.embedded.com/real-men-program-in-c/















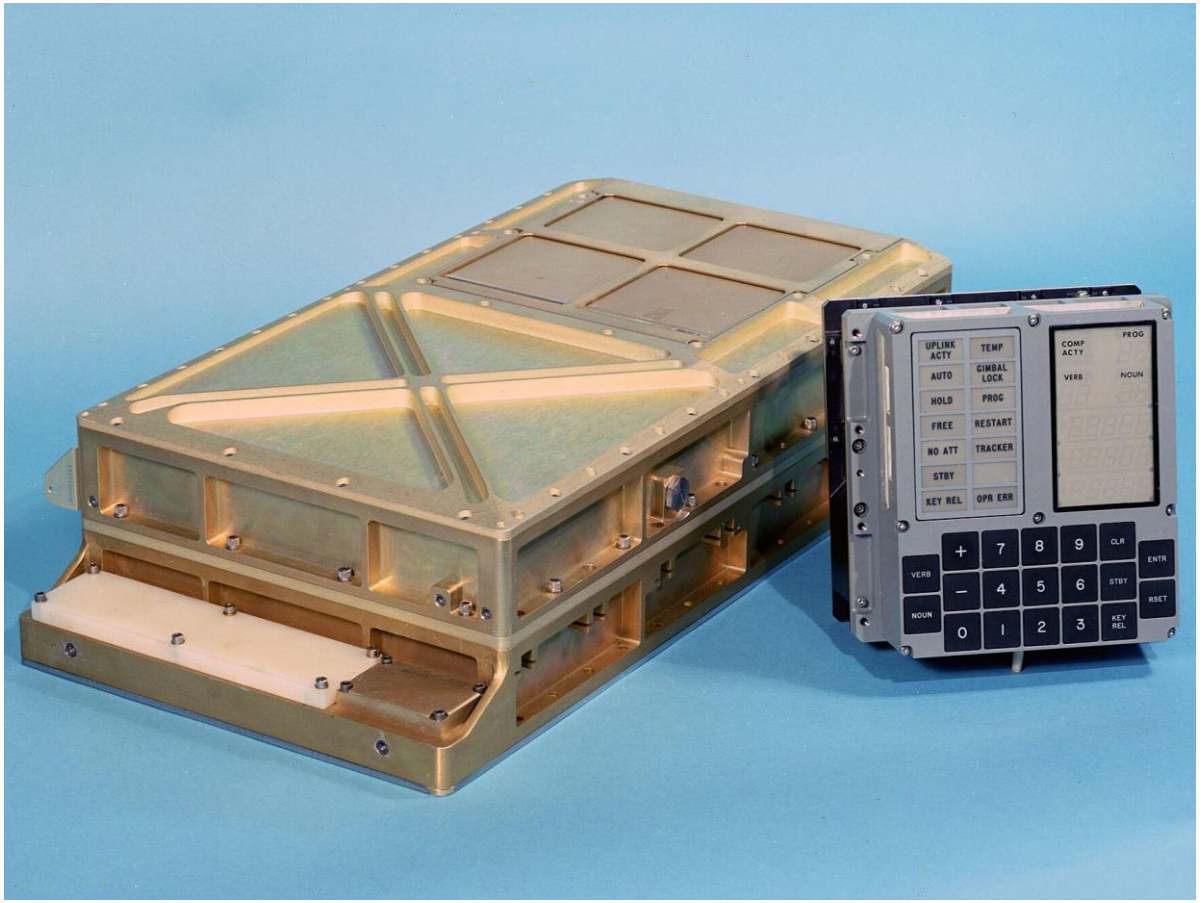


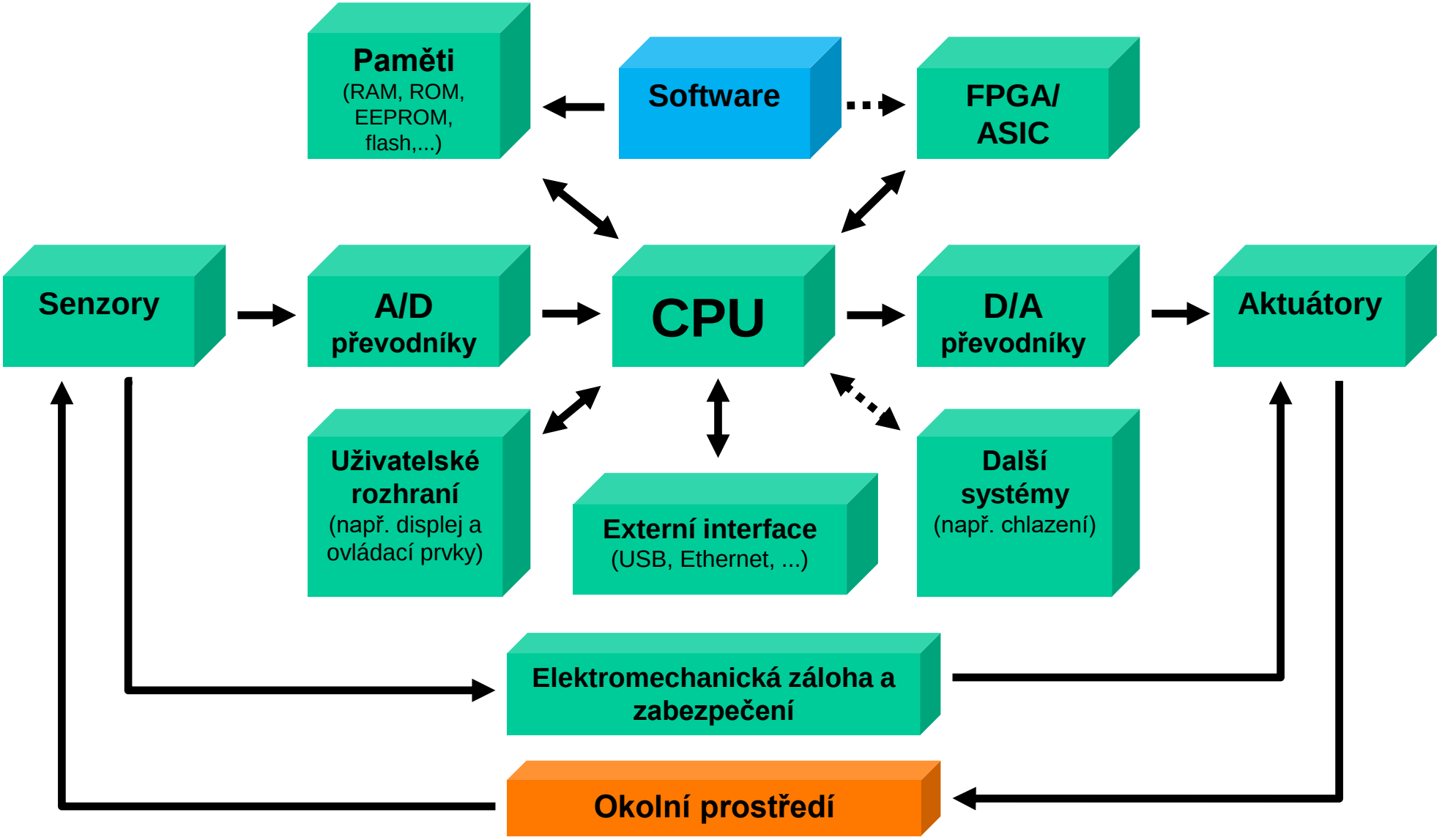

všechna tato zařízení obsahují zabudované (= vestavěné) počítače

Apollo Guidance Computer, 1966-1975

Za první vestavný systém je často považován naváděcí počítač Apollo (AGC), který byl vyvinut na MIT a použit v kosmické lodi Apollo 11. AGC byl zodpovědný za řízení naváděcích, navigačních a řídicích systémů kosmické lodi a lunárního modulu.

Procesor	diskrétní křemíkové integrované obvody
Frekvence	2.048 MHz
Feritové paměti	15-bitová délka slova + 1-bit parita 2048 slov RAM 36864 slov ROM
Porty	DSKY, IMU, Hand Controller, Rendezvous Radar (CM), Landing Radar (LM), Telemetry Receiver, Engine Command, Reaction Control System
Spotřeba	55 W
Programovací jayzk	AGC assembler
Hmotnost	32 kg
Rozměry	61 cm × 32 cm × 17 cm




Paměti
(RAM, ROM, EEPROM, flash,...)
Software
FPGA/ ASIC
Senzory
A/D převodníky
CPU
D/A převodníky
Aktuátory
Uživatelské rozhraní
(např. displej a ovládací prvky)
Externí interface
(USB, Ethernet, ...)
Další systémy
(např. chlazení)
Elektromechanická záloha a zabezpečení
Okolní prostředí

- **Procesor** (nebo více procesorů)
- **ASIC** (application specific integrated circuit / zákaznický integrovaný obvod), **DSP** (digital signal processor)
- **Paměťové bloky** (RAM, ROM, EEPROM, flash, ...)
- **Zdroje časování** (oscilátory, fázové smyčky)
- **Čítače-časovače, real-timové časovače, power-on reset generátory**
- **Analogový interface** (A/D-D/A převodníky)
- **Regulátory napětí a příkonu**
- **Propojení**
- **Externí interface** (UART, USART, USB, Ethernet, FireWire, ...)

- Je nutné navrhnout hardware i software (+ interakce s okolím)
- SW … řídí procesorová jádra, periferie a interface

- Paralelně HW i SW (tzv. hardware-software codesign)
- Často skládání HW bloků, jader, driverů
  → problém CAD (Computer Aided Design) nástrojů
- Blok programovatelného/rekonfigurovatelného hardwaru
  → FPGA (Field Programmable Gate Array)
- Verifikace návrhu
  - funkční (testování pomocí simulací):
    HVL (Hardware Verification Language), SystemVerilog, SystemC, OpenVera
  - formální (verifikace formálními matematickými důkazy):
    CTL (Computation Tree Logic), LTL (Linear Temporal Logic)

Podle požadavků použití/aplikace:

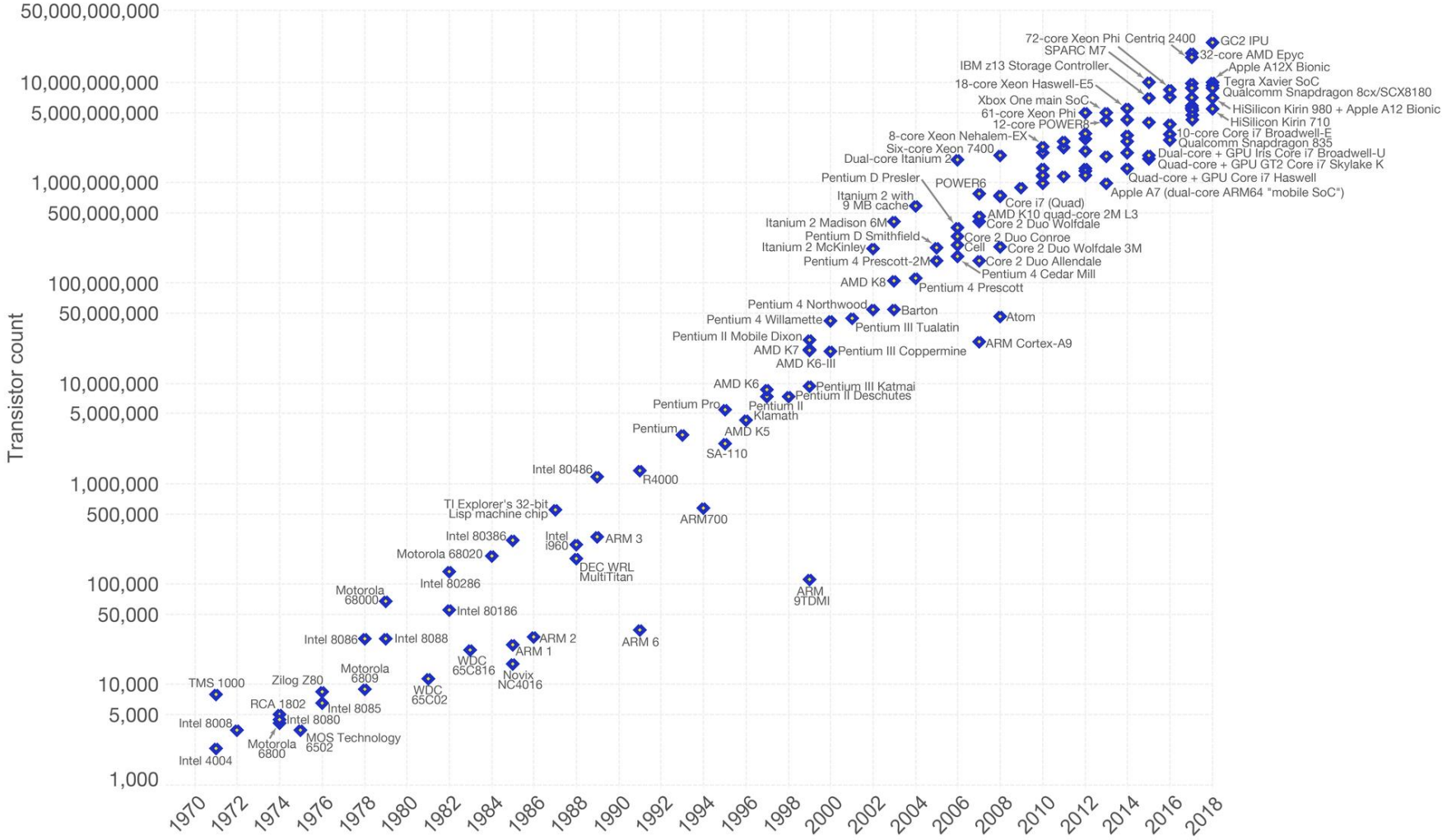
- Technologie:
  - Standardní bloky – jádra, mikrokontroléry, programy v C
  - Speciální HW
    - FPGA (Field Programmable Gate Array) – programovatelný návrh
    - ASIC (Application Specific Integrated Circuit) – plně zákaznický návrh
- Omezení
  - Velikost (area-overhead)
  - Spotřeba ... low-power design
  - Rychlost ... Real-time systémy
  - Spolehlivost ... lepší konkrétní parametr podle aplikace
  - Cena
  - „time-to-market“

- Rychlý rozvoj technologií ... Moorův zákon
- Možnost realizovat a prakticky používat složitější systémy
- Možnost implementovat adaptivní systémy
  (= složitější algoritmy, strojové učení)
- Nové technologie, budoucnost, vize
- Open source software i hardware

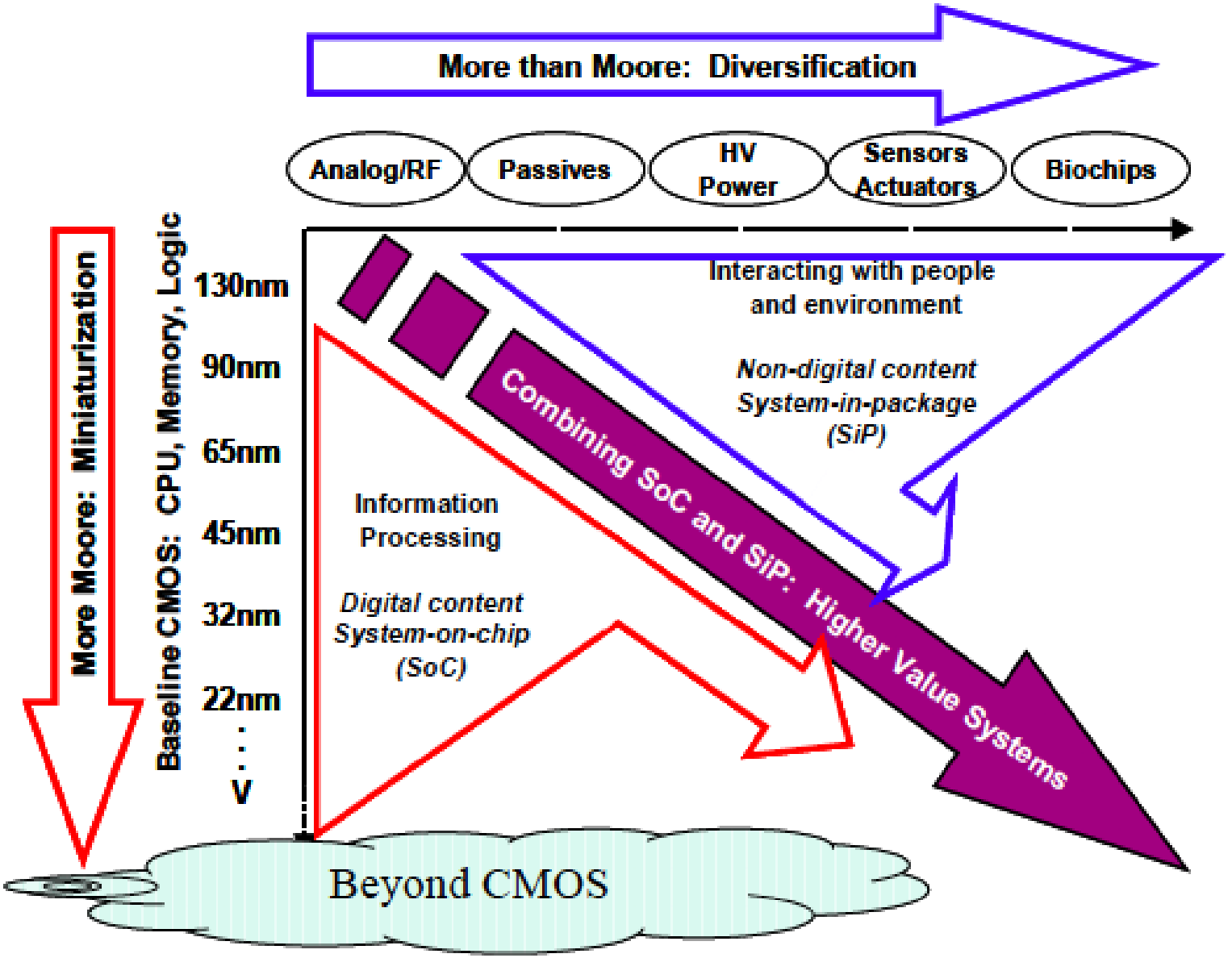
- Vhodné modely a návrhové prostředky

Moorův zákon - počet tranzistorů v čipech integrovaných obvodů (1971-2018)

Moorův zákon popisuje empirickou zákonitost, že počet tranzistorů v integrovaných obvodech se zdvojnásobuje přibližně každé dva roky. Tento pokrok je důležitý, protože s Moorovým zákonem souvisí i další aspekty technologického pokroku - například rychlost zpracování dat nebo cena elektronických výrobků.



Data source: Wikipedia (https://en.wikipedia.org/wiki/Transistor_count)
The data visualization is available at OurWorldinData.org. There you find more visualizations and research on this topic.

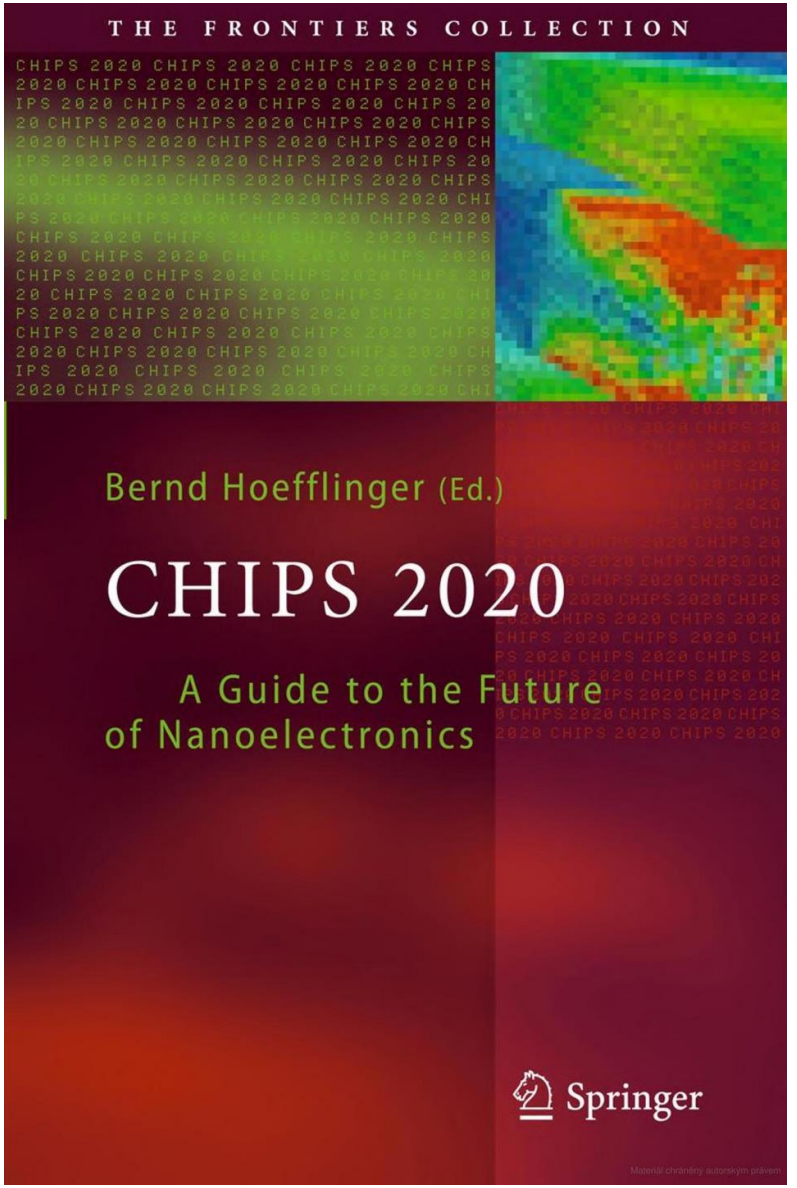

More than Moore: Diversification
Analog/RF
Passives
HV Power
Sensors Actuators
Biochips
More Moore: Miniaturization
Baseline CMOS: CPU, Memory, Logic
130nm
90nm
65nm
45nm
32nm
22nm
V
Interacting with people and environment
Non-digital content System-in-package (SiP)
Combining SoC and SiP: Higher Value Systems
Information Processing
Digital content System-on-chip (SoC)
Beyond CMOS

kniha

CHIPS 2020

A Guide to the Future of Nanoelectronics

https://www.chips2020.net/list-of-chapters



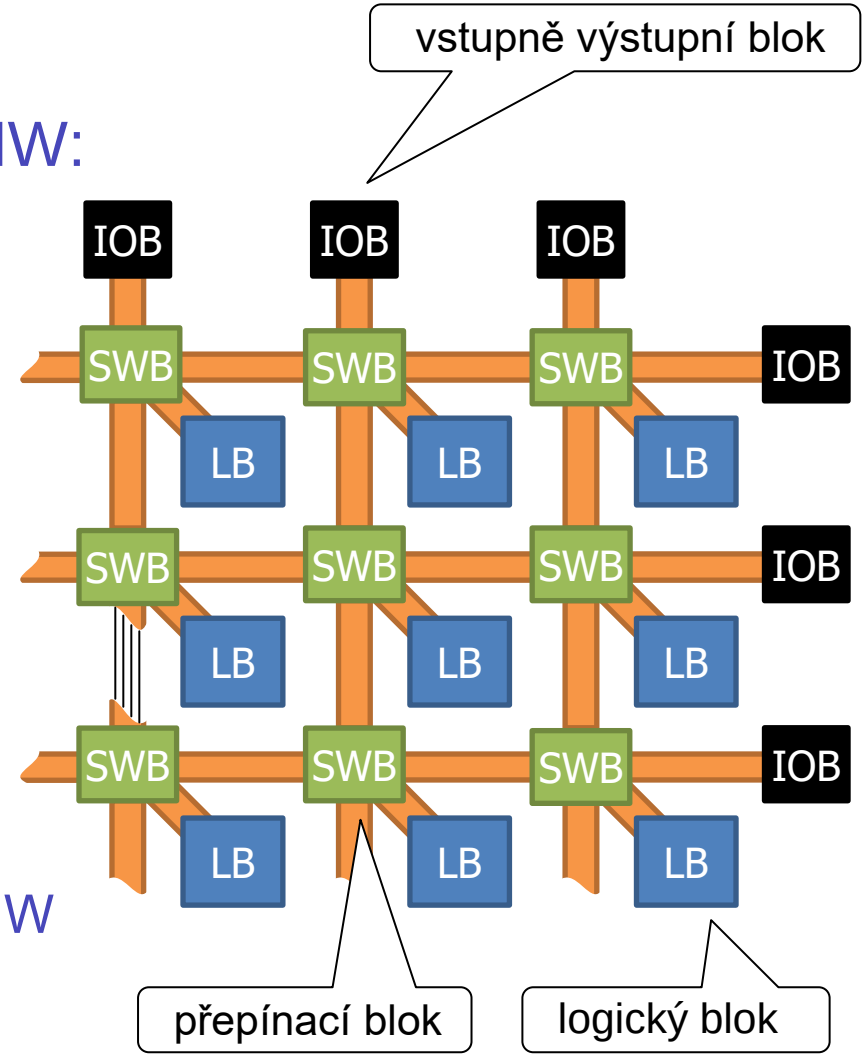
- **C**
  - Efektivní využití systémových prostředků
  - Výkon / rychlost
- **C++**
  - Podpora objektově orientovaného programování
- **Java**
  - Nezávislost na platformě: Kód lze spustit na jakémkoli zařízení s nainstalovaným virtuálním strojem Java (JVM)
  - Bezpečnost běhu: automatická správa paměti a kontrola typů, které pomáhají předcházet běžným bezpečnostním zranitelnostem známým z C/C++
- **Python**
  - Snadné používání
  - Rozsáhlé knihovny s efektivními implementacemi např. pro zpracování obrazu, strojového učení, atd.
  - V současnosti jeden z nejpopulárnějších programovacích jazyků vůbec ➔ velká komunita a podpora řešení na nejrůznějších internetových fórech
- **Verilog**
  - Verilog je jazyk (vzdáleně podobný jazku C) pro popis hardware, slouží pro modelování a návrh elektronických systémů. Jazyk (někdy nazývaný Verilog HDL) podporuje design, verifikaci a realizaci analogových, digitálních a smíšených signálových obvodů s různou úrovní abstrakce.

- Pracuje paralelně, stále
  vyšší výkon, ale i komplikace návrhu
- Cílové prostředí není lineární sekvence instrukcí, ale „2,5D“ prostor (více 2D ploch navrstvených)
  vyšší výkon, ale i komplikace návrhu
- Prostorové vztahy hrají důležitou roli
- Předmět počítačové podpory
  - EDA: Electronic Design Automation
  - architektury SW
  - algoritmy

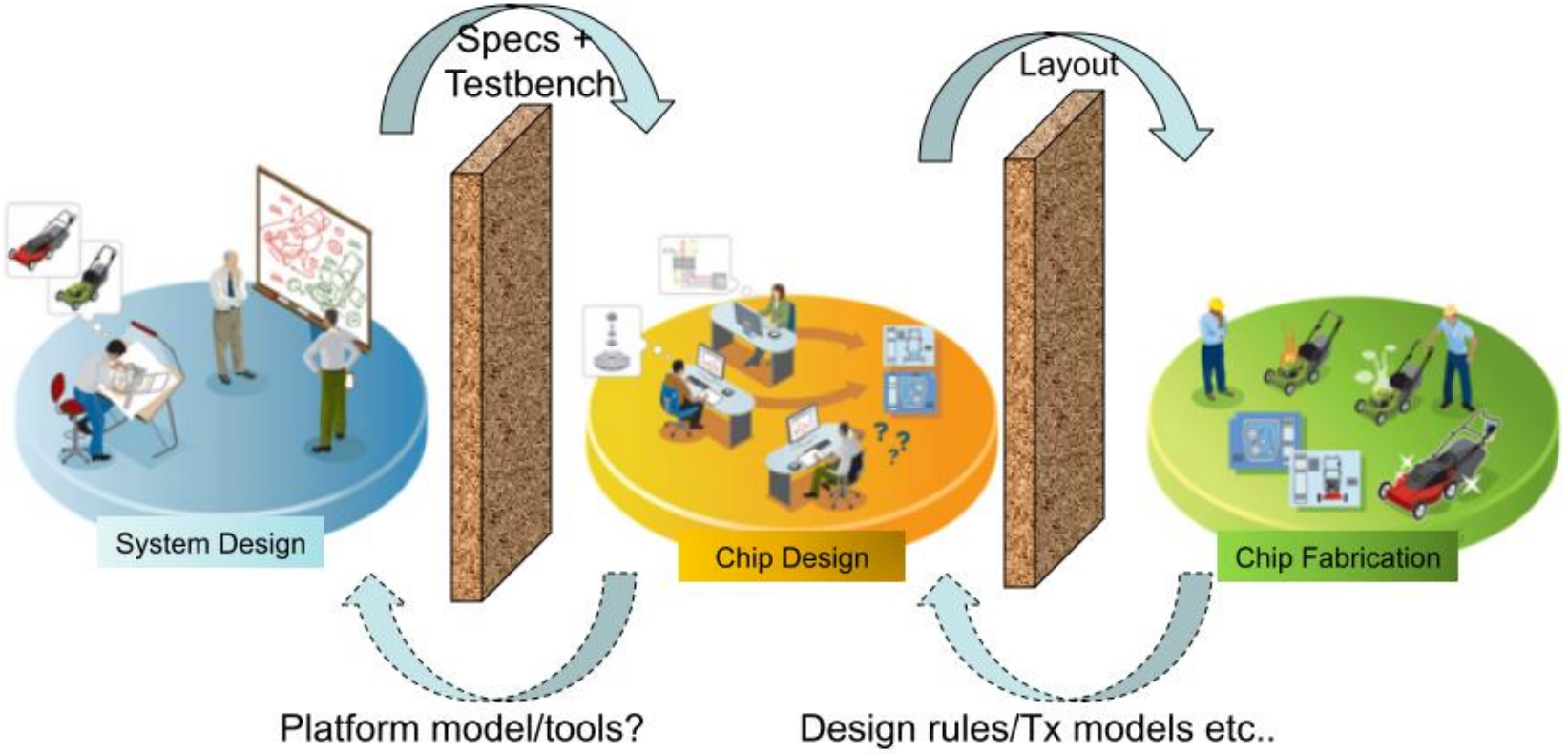
- FPGA … pole programovatelných „hradel“
- FPGA … „RAM/flash based“
- Rekonfigurovatelná implementace HW:
  - Konfigurační data – bitstream
    - Definuje implementovanou funkci
    - Definuje strukturu obvodu
- Garance úrovně spolehlivosti
- Poruchy a jejich modely ….

  → jiné modely než ASIC

  → FPGA je další úroveň mezi SW a HW



Challenge: Today's design process– The Walls



- Specs in Matlab or C
- Chip output must match testbench ***bit-exactly***
- However, those testbenches reflect a small subset of use cases

- Jak se bude chovat vestavná paměť při agresivním snížení napětí?
- Jak kompenzovat chyby paměti na systémové úrovni?
- Kolik ušetřím energie? Co mě to bude stát?
- Snížením V_{dd} se zvyšuje pravděpodobnost poruchy ...
- Tolerance chyb za cenu redundance
- S nespolehlivostí je třeba počítat již na **systémové, tzn. softwarové úrovni**

- *Počítej s (ne)spolehlivostí*: systémový návrhář by měl předpokládat určitou úroveň spolehlivosti

a

- *Vyjádři (ne)spolehlivost*: návrhář SoC by měl umět modelovat spolehlivost na systémové/softwarové úrovni

- Propojení na čipu je limitujícím faktorem pro zvýšení výkonu a snížení spotřeby
- Moorův zákon – stále se zvyšující hustota integrace
- Vede ale i k vyšší pravděpodobnosti chyb
- Nemožnost použít jeden rozvod CLK – problémy synchronizace ...
  - Globálně asynchronní / lokálně synchronní (GALS)
  - Několik různých hodinových domén
- Možné řešení … využití síťových přincipů i na čipu = NoC (Network on Chip) ale s ohledem na omezení (spolehlivost, velikost, spotřebu, výkon .... podle aplikace)

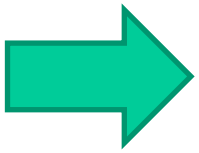
Fyzická úroveň

- Komunikační kanály jsou realizovány na čipu pomocí propojovacích vodičů
- Bezchybnou komunikaci ale není možné garantovat (šum, SEU (Single Event Upset), …)

Úroveň architektury a řízení

- Architektura určí topologii NoC a fyzickou organizaci
- Protokoly specifikují jak využít síťové zdroje ve funkčním režimu systému
- Rozdíl mezi mikro-sítí (NoC) a obecnou sítí: nutnost splnit omezující energetické požadavky
- Využití různých architektur propojení (sdílené prostředky, přímé, nepřímé i hybridní sítě)

- Jaký použít nástroj pro dorozumění mezi návrhářem a zadavatelem?
- Jak splnit požadavky pro různá omezení (velikost, spotřebu, spolehlivostní parametry a které to jsou, cenu, ...)?
- Jak optimálně realizovat systém obsahující HW, SW, a další bloky?
- Jak tyto požadavky garantovat současně ve vývoji, výrobě a při používání systému?



Je třeba vhodný model

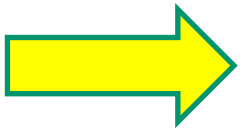
.... ale každý model je nějakým způsobem zjednodušený

Hierarchický popis

- Celek je složen z částí, části z menších částí... → úrovně hierarchie
- Složitost popisu na každé úrovni dána schopností lidí (a strojů) vnímat ..... (žádná funkce nemá mít víc jak 500 řádek, každé schéma se musí vejít na papír A2, ...)
- Odlišení domén chování, struktury a fyzické implementace
- Dekompozice obvodu *(paralelně)*
- Dekompozice algoritmů *(sériově)*
- Analytické a syntetické kroky
- Důkladnější algoritmy *(jinak to prostě nejde)*
- CAD systémy, EDA (Electronic Design Automation) tools

- Von Neumann model (Sekvenční provádění, paměť programu, etc.)
- Discrete-event model, časový diagram ... Verilog, VHDL
- Konečný automat (Finite state machine FSM) ...
  - klasický (jednoduchý)
  - „state charts“ [Harel, 1987] s hierarchií
- Diferenciální rovnice

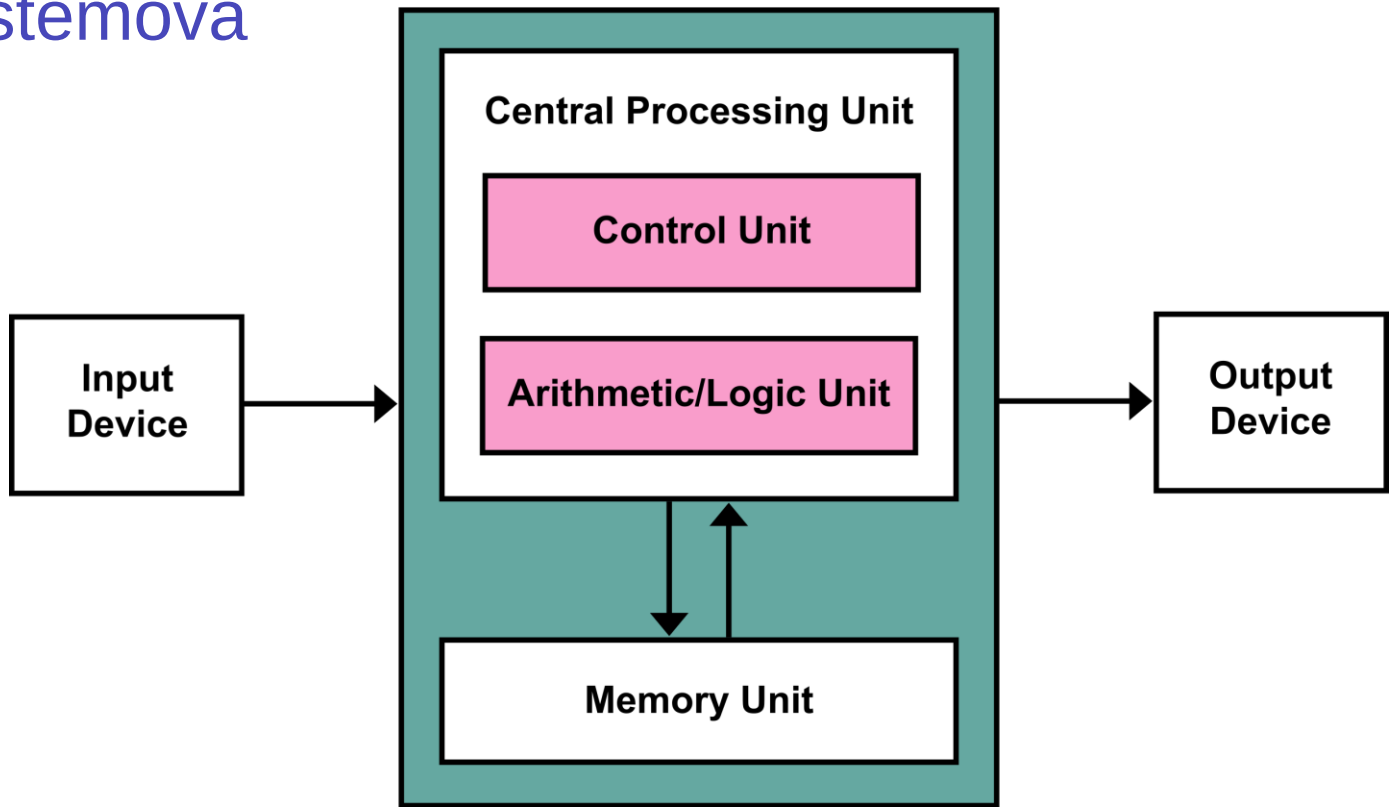
Takový jazyk, který toto všechno splňuje neexistuje



je třeba kompromis

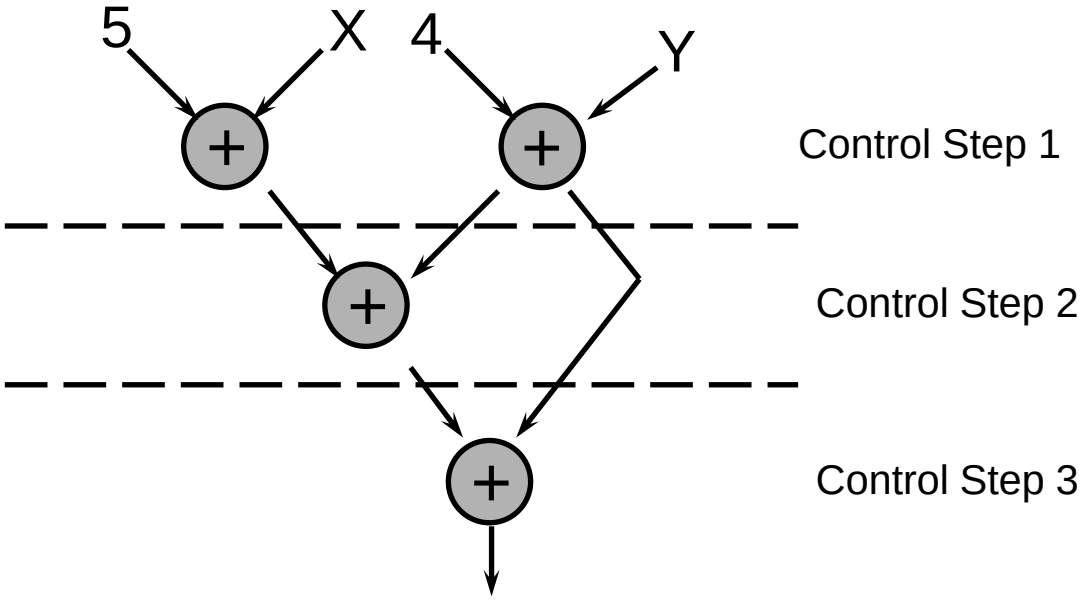
- Specification and Description Language (SDL) i pro distribuované systémy
- Dataflow models:
  - zcela odlišný způsob náhledu na výpočet : imperativní jazykové styly, pohyb dat je prioritou, systém (ne programátor) zodpovídá za plánování
- Kahn Process Network
- Objektově-orientované reprezentace
- Petri Nets a jejich rozšíření (Timing PN, CPN, GPN, ...)

- Von Neumannova architektura (výpočetní model)
  - společná paměť pro program i data
  - instrukce se vykonávají sekvenčně
  - sdílená systémová sběrnice



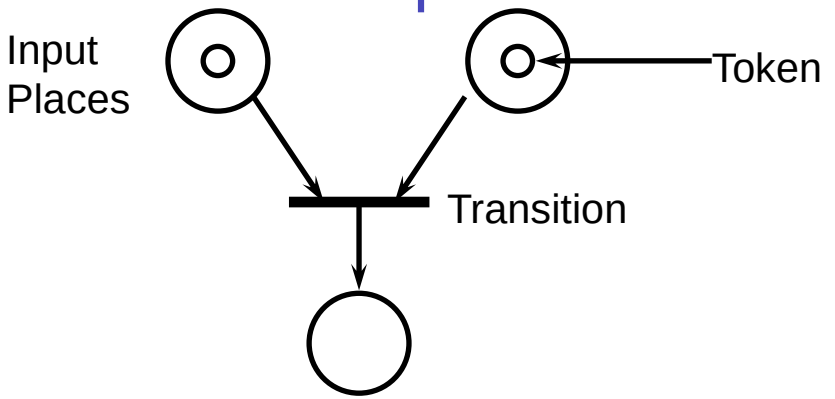
- Grafy obsahující uzly odpovídající operacím … reprezentace v HW nebo v SW
- Obvykle využívané v high-level HW syntéze
- Modelují data flow, řídící kroky, souběžné operace
- Využití: neuronové sítě, zpracování signálu/obrazu, ...

Příklad:



- Používají techniky dříve aplikované jen na SW i pro správu HW
- Využití jazyka C++ pro popis HW a vyjádření OO vlastností
- Využití konceptů OO:
  - Datová abstrakce
  - Zapouzdření (skrývání informace, nemožnost přímého přístupu k datům)
  - Dědičnost
- Využití stavebních bloků pro využití OO podpory
  - Opakované využíváni komponent
  - Nižší cena návrhu
  - Rychlejší návrhový proces
  - Zvýšená spolehlivost

- Petri Nets: matematický a grafický model obsahující dva typy uzlů (místa a přechody), značení (marking) míst a pravidla pro změnu (firing)
  - Místa (Places) – vyjadřují podmínky a nesou tečky (tokens)
  - Tečky – reprezentují tok informace po síti
  - Přechody (Transitions) – vyjadřují události (změnu stavu), spuštěním (firing) přechodu nastane událost
  - Značení (Marking) – rozmístění teček po Petriho síti, reprezentuje stav



- Systems Modelling Language (SySML)
- je určen pro modelování SW částí (nejen) vestavných systémů
- vznikl jako varianta UML (Unified Modeling Language), ale od verze 2.0 je nezávislý
- využívá 7 ze 14 UML diagramů + přidává dva nové

https://sysml.org/

- Vestavné systémy se vyskytují všude kolem nás
- Klíčový problém: optimalizace metrik návrhu
  - Návrhové metriky si navzájem konkurují
- Pro zvýšení produktivity je nutný jednotný pohled na hardware i software
  - modely ... hierarchické, podrobné, ...
  - dedikované procesy a metodické postupy
  - universální konstrukční principy (propojující modelovací principy s realitou vývoje a implementací)
  - automatizovatelné kroky zajišťující rychlost návrhu
- Klíčové technologie
  - Procesor: univerzální, aplikačně specifický, jednoúčelový
  - Návrh: kompilace/syntéza, knihovny, testování/verifikace