**Margiris Antanas Malakauskas Bioinformatika II kursas.**

**ARMv4 vs. Motorola 68HC05**

**Techninis ir istorinis kontekstas:**

**ARMv4:**

ARMv4 procesoriai pasirodė 1997 metais ir naudojo naujas VLSI ir CMOS technologijas: VLSI - tai tranzistorių integravimas į vieną silicio lustą, kas stipriai sumažino procesoriaus dydį nepaveikiant jo greičio ir efektyvumo. CMOS – komplimentarūs metalo oksido puslaidininkiai, kurie leidžiantys procesoriui naudoti mažus energijos kiekius (aktyviai vykdant procesus energijos sąnaudos vos siekė kelis šimtus mW). Šios technologijos leido ARMv4 procesoriui būti tų laikų mikroprocesorių dydžio (vos keli kvadratiniai milimetrai), svorio (vos kelių gramai), tačiau daug spartesniu ir efektyvesniu energijos sąnaudų prasme. Dėl šių savybių, jie tapo populiarūs tarp nešiojamuose įrenginiuose.

**Motorola 68HC05:**

Motorola 68HC05 pasirodė 1996 metais. Tai LSI mikrovaldiklis, kurio paskirtis buvo būti naudojamam įterptinėse sistemose, dažnai kaip užprogramuojamam ROM. Kaip ir ARMv4, šis utilizavo CMOS silicio pagrindą, tačiau neturėjo tiek daug tranzistorių (tūkstančius, tuo tarpu ARMv4 turėjo milijonus), kas silpnino jo sugebėjimus. Dėl šiek tiek ankstyvesnių technologijų, palyginus su ARMv4, šis mikrovaldiklis buvo keliais milimetrais didesnis (tačiau vis vien kompaktiškas), bei naudojo daugiau energijos – aktyvių procesų metu siekė iki 500 mW.

**Registrai:**

**ARMv4:**

Naudoja IR bendrosios paskirties registrus IR specializuotus registrus. Dažniausiai turėdavo 16 bendrosios paskirties registrus ir 2 specializuotus registrus. Iš bendrų registrų keli turėjo specifinęs paskirtis – atsarginei atminčiai, funkcijų grįžimo adreso saugyklai, programų counterį. Specializuoti registrai saugojo procesoriaus būsenos informaciją ir vykdomų programų būseną. Registrai turėjo 32 bitų plotį.

**Motorola 68HC05:**

68HC05 irgi naudojo ir bendros paskirties ir specializuotus registrus. Turėjo 3 o kartais 4 bendrus registrus ir 3 specializuotus registrus – valdymo, funkcijų gryžimo adresų ir atminties adresų generavimo. Procesoriaus greitį lyginant su ARMv4 stabdė ne tik registrų kiekis bet ir tai, kad registraiturėjo tik 8 bitų plotį.

**Architektūros Tipai:**

**ARMv4:**

ARMv4 naudojo registrinę architektūrą – procesoriuje esantys registrai apdoroja duomenis, kurie siunčiami tarp registrų. Toks duomenų apdorojimo būdas leido pakankamai greitai manipuliuoti duomenis ir neapkrauti atminties.

**Motorola 68HC05:**

68HC05 mikrovaldiklis naudojo akumuliatorinę architektūrą – egzistuoja pagrindinis darbinis registras („accumulator“) kuris tiesiogiai ima duomenis iš kitų menkesnių registrų ar net atminties. Tai leidžia greičiau apdirbti duomenis bet greičiau apkrauna atmintį/leidžia mažiau procesų. Palyginus su ARMv4 tai dažniausiai reiškė, kad programų dydžiai buvo mažesni, o laikas įvykdyti kiekvieną programą ilgesnis.

**Komandinė sistema:**

**ARMv4:**

ARMv4 architektūra naudoja RISC procesorių – bando visas instrukcijas įvykdyti per vieną ciklą. Instrukcijoms naudojamas 32 arba 16 bitų formatas.

Instrukcijų pavyzdžiai:

Duomenų tvarkymo - ADD, SUB(tract), MOV(e), CMP (compare), MUL(tiply)

Duomenų saugojimo – LDR (load register), STR (store register)

Kontrolės – MRS (Move to register from status register), MSR (Move status register

Ši architektūra taip pat buvo sukurta su įvairiausiais įrankiais Input/Output procesams – turi GPIO jungtis sensoriams ir LED šviesoms, turi galimybę stabdyti procesus pagal inputą iš kitų prietaisų, turi DMA (direct memory access) ir t.t.

**Motorola 68HC05:**

68HC05 mikrokontroleris naudoja CISC procesorių – ne taip kaip ARMv4, šis procesorius gali skirti daugiau ciklų instrukcijoms jeigu to prireikia, tačiau laiko mažesnį kiekį galimų instrukcijų. Instrukcijoms naudojamas 8 bitų formatas.

Instrukcijų pavyzdžiai:

Duomenų tvarkymo: ADD, SUB, AND, OR, EOR

Kontrolės – JMP (jump), JSR (Jump to subroutine), RTS (Return from subroutine)

Duomenų saugojimo: STA (store accumulator), LDA (Load accumulator)

Bitų manipuliacijos: BSET (Set specific bit), BCLR (clear specific bit)

Kaip ir ARMv4, geba naudoti daugelį I/O įrankių, tačiau neturi galimybės naudoti DMA ir dėl atminties trūkumo, bei energijos sąnaudų, pagrinde pritaikomas tik paprastiems procesams – pvz. ARMv4 galėjo turėti nustatytus operacijų nutraukimo atvėjus, priotizuoti pertraukimo operacijas. Tuo tarpu 68HC05 negalėjo suteikti prioriteto pertraukimo operacijosmis, jas pagrinde vykdė išoriniais veiksniais arba laikmačio principu.

**Duomenų tipai/atminties struktūra:**

**ARMv4:**

ARMv4 naudoja statuso registrą su požymių bitais, kurie nurodo procesoriaus būseną ir naudojami informacijos apdorojime ir srauto konrolėje. Požymių bitai: N(egative) – Nustato rezultato neigiamą/teigiamą būseną, Z(ero) – Nustato ar rezultatas lygus nuliui, C(arry) – Nustato sėkmingus nešimus ir pasiskirstymus, V (Overflow) – nustato ar ivyko „overflow“ (kai duomenys viršija registro plotą), Q (Saturation) Nurodo arimetinių operacijų „saturated“ būseną, T(humb state) – nurodo ar ARM procesorius perjungtas į „nykščio“ rėžimą, kuriame naudojama 16 bitų kodavimo sistema (vietoj normalios 32 bitų).

ARMv4 Naudojo ištisinę atminties erdvę – adresai yra tiesiogiai susiję su atminties vietomis. Tipiškas atminties kiekis buvo apie 32MB o maksimalus - 4 GB. Taip pat, ARMv4 procesorius geba palaikyti virtualią atmintį JEIGU tam buvo pritaikyta operacinė sistema – galimas buvo puslapiavimas jeigu kartu buvo naudojamas MMU.

ARMv4 palaiko įvairių rūšių duomenis (tam laikotarpiui) – sveikuosius skaičius, fiksuotojo ir slankiojo kablelio aritmetinius duomenis, dešimtainius skaičius ir pritaikius tam tikras modifikacijas, net apskaičiuoti kompleksinius skaičius

**Motorola 68HC05:**

68HC05 Taip pat naudojo požumių bitus, tačiau turėjo jų daug mažiau – N(egative), Z(ero), C(arry), V (Overflow), H(alf carry) tikrina ar įvyko pernešmas per pusę bitų (4 bitus 68HC08 atvėju).

68HC05 kaip ir ARMv4 Naudojo ištisinę atminties erdvę, tačiau jo tipiškas atminties kiekis buvo tik apie 4KB o maksimalus – 64KB. Ne taip kaip ARMv4, 68HC05 nesugebėdavo palaikyti virtualios atminties be specialių modifikacijų.

68HC05 negali palaikyti tiek daug duomenų rūšių kaip ARMv4, tačiau vis vien palaiko: sveikuosius skaičius ir, su paprastu modifikavimu, fiksuoto kablelio aritmetinis duomenis.

**Adresavimo režimai:**

**ARMv4:**

ARMv4 procesorius yra dviejų adresų mašina. Tai reiškia, kad šio procesoriaus instrukcijose gali būti iki dviejų adresų – tai leidžia imti duomenis iš vieno registro ir tuo pat metu vykdyti procesus kitame registre.

**Motorola 68HC05:**

68HC05 mikrokontroleris yra vieno adreso mašina. Tai reiškia kad akumuliatorius yra pagrindinis darbo registras ir komandos apdoroja tik vieną adresą, tad dažniausiai yra apdorojamos akumuliatoriuje. Lyginant su ARMv4 procesoriu, šis modelis lėtesnis dėl informacijos nuolatinio keliavimo iki akumuliatoriaus.

**Mašinos kodo/asemblerio pavyzdys:**

**ARMv4:**

**Motorola 68HC05:**

**Aukšto Lygio programavimo kalbų palaikymas:**

**ARMv4:**

**Motorola 68HC05:**

**Įvestys-Išvestys ir pertraukimai:**

**ARMv4:**

**Motorola 68HC05:**

**Literatūra:**

**ARMv4:**

[ARM Architecture Refrence Manual](https://www.cse.iitd.ac.in/~srsarangi/courses/2011/cs211/arm_ref_manual_book.pdf)

[ARM architektūros „šeima“ WIKIPEDIJA](https://en.wikipedia.org/wiki/ARM_architecture_family)

ARM System-on-Chip Architecture Steve Furber

**Motorola 68HC05:**

[EE|times najai paskelbto Motorola 68HC05 galimybės](https://www.eetimes.com/motorola-announces-new-68hc05-microcontroller-for-computer-peripheral-applications/)

[Motorola 68HC08 WIKIPEDIJA](https://en.wikipedia.org/wiki/Motorola_68HC05)

An introduction to Motorola‘s 68HC05 familly of 8-bit microcontrollers

**Apipavidalinimas/Metaduomenys:**

**ARMv4:**

**Motorola 68HC05:**