2.2 Atmiņa

Mikrokontrollera atmiņa var tikt sadalīta atbilstoši tās funkcijām:

Reģistru atmiņa: Relatīvi maza atmiņa, kura ir izvietota CPU. To izmanto kā pagaidu atmiņu vērtībām, ar ko strādā CPU. To var arī saukt par CPU īstermiņa atmiņu.

Datu atmiņa: atmiņa ilgākai datu glabāšanai. CPU parasti izmanto ārējo atmiņu, kas ir daudz ietilpīgāka nekā reģistru atmiņa. Dati, kas tiek glabāti datu atmiņā, var būt gan īslaicīgi, gan var būt derīgi tik ilgi, kamēr CPU darbojas. Ja pievieno ārējo atmiņu CPU, tas prasa aparatūras piepūli. Tāpēc šī iemesla dēļ mikrokontrolleri datus parasti izvieto atmiņā, kas atrodas uz pašas mikroshēmas.

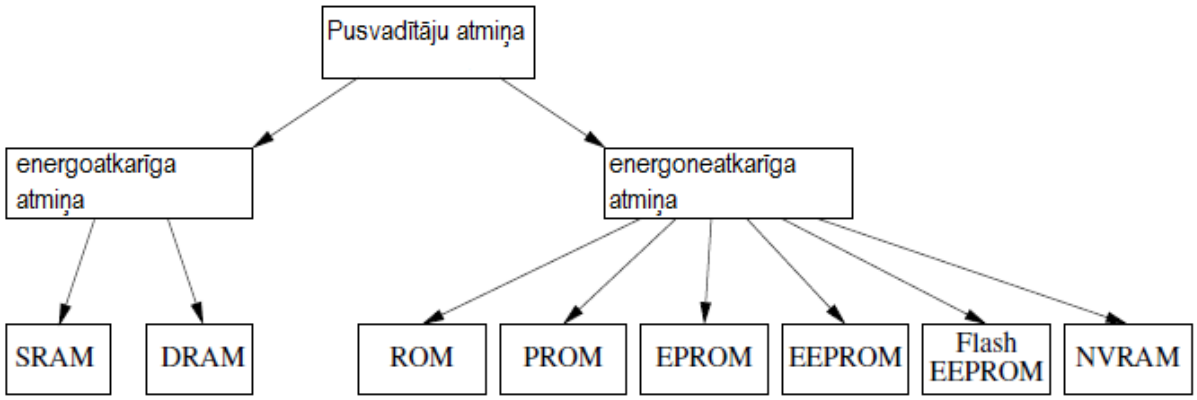
Instrukciju atmiņa: Tāpat kā datu atmiņa, instrukciju atmiņa parasti ir salīdzinoši liela ārējā atmiņa. Neimana arhitektūrā tā var būt tā pati fiziskā atmiņā kā datu atmiņa. Instrukciju atmiņa arī parasti ir integrēta MCU (microcontroller unit).

Runājot par atmiņu, kas iestrādāta MCU – protams, šādai uz mikroshēmas esošai atmiņai ietilpība ir ierobežota. Un bieži vien nav iespējams, nesarežģījot mikrokontrollera konstrukciju, paplašināt atmiņu, izmantojot ārējo atmiņu.

Tā kā mikrokontrolleri visbiežāk tiek izmantoti samērā vienkāršu uzdevumu izpildei, tad nav nepieciešams pārmērīgs atmiņas apjoms, ir ieteicams iekļaut nelielu datu un instrukciju atmiņas apjomu mikroshēmā. Tādā veidā, kopējās sistēmas izmaksas ievērojami samazinās.

Lietotājs vienmēr var izvēlēties sev atbilstošo starp daudzajiem mikrokontrolleriem. Dažādi mikrokontrolleri parasti nodrošina dažādus atmiņas apjomus, lai lietotāji varētu izvēlēties konkrētu mikrokontrolleri, kas var nodrošināt ar nepieciešamo atmiņas apjomu.

Atmiņu var iedalīt arī pēc izmantošanas veida. No programmētāju viedokļa šim iedalījumam ir liela jēga. Savukārt aparatūras projektētāji dod priekšroku atmiņas iedalīšanai pēc elektronisko detaļu, no kurām ir izveidota atmiņa, fizikālām īpašībām. Iedalot atmiņu šādi, ir divi galvenie atmiņas veidi: energoneatkarīga atmiņa un energoatkarīga atmiņa. Termins „energoatkarīga atmiņa" nozīmē – atmiņas saturs tiek nodzēsts, kad barošana tiek pārtraukta. Energoneatkarīga atmiņa saglabā datus, kas būs vajadzīgi arī pēc barošanas pārtraukšanas.



Attēlā var redzēt, ka energoatkarīga atmiņa iedalās statiskā un dinamiskā atmiņā. Energoneatkarīgās atmiņas tipi: ROM, PROM, EPROM, EEPROM, FLASH EEPROM, NVRAM.

Energoatkarīga atmiņa

Statiskais RAM. Statiskas brīvas pieejas atmiņa (Static Random Access Memory, SRAM) ir pirmā veida operatīva atmiņa, kas tika plaši izmantota. SRAM čips sastāv no šūnu masīva, kur katra šūna spēj uzglabāt vienu bitu informācijas. Lai uzglabātu nedaudz informācijas, pielieto trigeri, kas pamatā sastāv no sešiem tranzistoriem.

Parasti šīs šūnas izvieto matricas veidā, lai palielinātu glabāšanas datu apjomu (skat. att. 2.5). Kā redzams, visas DOUT līnijas ir saistītas kopā. Lai atlasītu vienu matricas šūnu un visas pārējas šūnas salikt kopā pielieto CS līniju. Līdz ar to viena bita piekļūšanas uzdevumam, SRAM ir vajadzīga papildus loģika.

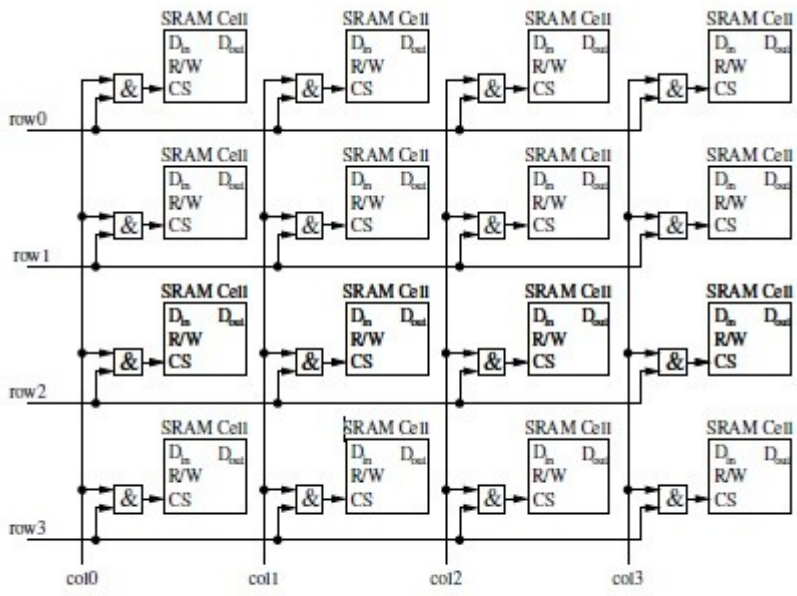


Figure 2.5: A matrix of memory cells in an SRAM.

Dinamiskais RAM (Dynamic Random Access Memory): tranzistoru skaits, kas nepieciešams uz vienu bitu informācijas ir samazināts uz vienu. Tātad vienāda izmēra mikroshēmām DRAM ir daudz lielāka uzglabāšanas jauda, salīdzinot ar SRAM.

Tā vietā, lai izmantojot daudz tranzistoru veidot trigerus, viens informācijas bits tiek glabāts kondensatoros. Informācija patiešām tiek glabāta kondensatoros, bet, lai to nolasītu ir vajadzīgs tranzistors.

Ja ir vajadzīgs saglabāt loģisko signālu, tad atmiņas šūnu, kurai ir jāpiekļūst, var adresēt ar tranzistora palīdzību. Lai saglabātu loģisko nulli, ir jāizvēlas šūna un jāizlādē kondensators. Lai nolasītu informāciju ir jāpārbauda, vai kondensators ir uzlādēts, vai nav.

Kad kondensators ir uzlādēts, tam teorētiski ir jāglabā lādiņš. Tomēr, ņemot vērā zudumu strāvas, kas plūst neideālas izolācijas dēļ, kondensatori zaudē savu lādiņu. Tas nozīmē, ka pēc uzlādēšanas, lādiņš pakāpeniski samazinās. Pēc neilga laika lādiņš tiks pazaudēts un informācija arī. Tādējādi DRAM ir jābūt pieejamam ik pēc pāris milisekundem, vai arī tā informācija tiks nozaudēta.

Lai novērstu atjaunošanas problēmu, ir pieejami DRAM ar iebūvētu atjaunošanas loģiku, kas savukārt aizņem mikroshēmas daļu, kas dažkārt nav nepieciešams: gadījumā kad CPU nav vajadzības piekļūt RAM katru ciklu, var pielietot iekšējo ciklu atjaunošanas uzdevumam. DRAM var atjaunot datus izmantojot ciklus starp CPU pieejam.

Energoneatkarīga atmiņa

Pretēji SRAM un DRAM, pastāvīgas atmiņas saglabā savu saturu pat tad, ja barošana ir atslēgta.

ROM (Read Only Memories, tikai nolasāma atmiņa) bija pirmā veida pastāvīgā pusvadītāju atmiņa. Ja jūs gribas izmantot ROM, jums ir jānodod dati, kas mikroshēmu ražotājam, kuri izstrādās specifisku mikročipu, kas saturēs jūsu datus.

PROM (Programmējama lasāmatmiņa). Tie ir galvenokārt atmiņas šūnu matricas, kur katra satur silīcija drošinātāju. Sākumā katrs drošinātājs ir neskarts, un katras šūnas loģiskais ieraksts ir 1. Izvēloties šūnu un piemērojot īsu, bet lielu strāvas impulsu, šūnas drošinātāju var iznīcināt, tādējādi ieprogrammējot loģisko 0 izvēlētajā šūnā.

EPROM pārprogrammējamā lasāmatmiņa. Atmiņa tiek būvēta uz FET tranzistoriem, vai drīzāk vienā no to izejam ko sauc par aizvaru. To dažreiz sauc par peldošiem aizvaru, jo tas ir pilnīgi izolēts no pārējās shēmas. Tomēr, piemērojot pietiekami augsto spriegumu, ir iespējams uzlādēt šo aizvaru. Šīs fizikālais process saucās par lavīnveida injekciju. Tātad, nevis sadedzinot drošinātāji, bet gan ievadot elektronus peldošā aizvarā, tādējādi aizverot tranzistora slēdzi.

Kad šūna ir ieprogrammēta, elektroniem jāpaliek aizvarā uz nenoteiktu laiku. Tomēr tāpat kā DRAM gadījumā, strāvas noplūde caur neideālo tranzistora izolatoru, izlādē aizvaru. Laika gaitā, peldošais aizvars zaudē pietiekami daudz elektronus lai informācija no tā tiktu dzesta. EPROM manuālā rokasgrāmatā, ražotājs norāda, cik ilgi atmiņas saturs paliek neskarts, parasti, tas ir apmēram desmit gadu.

EPROM gadījumā ierobežotais derīguma termiņš faktiski tiek izmantots ka priekšrocība: uzliekot silīcija mikroshēmu zem UV gaismas, procesu var paātrināt. Pēc aptuveni 30 minūtēm, UV gaisma ir atbrīvos peldošo aizvaru un EPROM dati tiek dzēsti. Šī iemesla dēļ EPROM ir neliels stikla logs paketē, caur kuru ir redzama mikroshēma. Parasti logs ir pārklāts ar gaismas

aizsargplēvi. Lai izdzēstu EPROM datus, jums ir noņemt aizsargplēvi un jāuzliek mikroshēmu zem intensīvas UV gaismas.

EEPROM

EPROM programmēšanas un jo īpaši dzēšanas process ir diezgan sarežģīts. Lai tos ieprogrammētu, pielieto īpašu programmēšanas spriegums, kas parasti ir augstāks nekā tranzistora darba spriegums. Lai izdzēstu EPROM ir nepieciešams UV gaismas avots.

EEPROM (elektriski pārprogrammējama un programmējamais ROM) ir visas EPROM priekšrocības, taču programmēšanas un dzēšanas process neprasa tik lielus resursus. Nav nepieciešams īpašais spriegums lai ieprogrammētu, un nav vajadzīgs UV gaismas avots lai nodzēstu informāciju. EEPROM darbojas ļoti līdzīgi EPROM, izņemot to, ka elektronus var novadīt no peldoša aizvara, piemērojot paaugstinātu spriegumu.

Protams, EEPROM ir savi trūkumi: tos ir iespējams ieprogrammēt un nozēst ierobežots skaits reižu un tie nesaglabā savu informācijas bezgalīgi.

Flash

Tagad, EEPROM, šķiet, ir lieliska izvēle priekš energoneatkarīgas atmiņas. Tomēr tai ir viens trūkums: tā ir diezgan dārgi. Kā kompromisu, ir pieejams Flash EEPROM. Flash ir EEPROM variants, kur katras adreses dzēšana nav iespējama, bet tikai lielākiem blokiem vai pat visas atmiņas uzreiz. Tādā veidā var vienkāršot iekšējo loģiku, kas savukārt ieverojami samazina cenas. Turklāt sakarā ar to, ka nav iespējams izdzēst vienu baitu, Flash EEPROM parasti izmanto programmas, nevis datu atmiņai. Tas savukārt nozīmē, ka var samazināt pieņemamo izturību (jo to neprogrammē tik daudz, ka datu EEPROM).Tāpēc, Flash EEPROM bieži vien ir zemāks ierakstu/dzesēšanas ciklu skaits, salīdzinot ar EEPROM. Parasti tas ir uz 1,000-10,000 cikliem. Tieši tas padara Flash EEPROM lētāk.

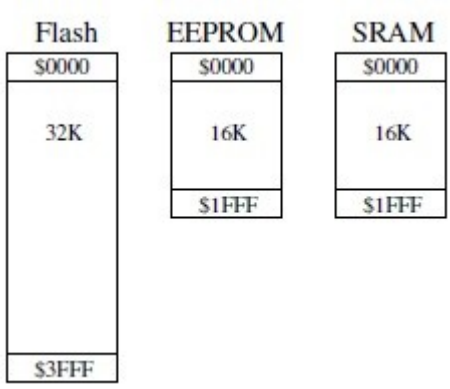
NVRAM

Visbeidzot, ir atmiņas veids, kas apvieno operatīvas un pastāvīgas atmiņas priekšrocības: pastavīgais RAM (NVRAM). To var panākt dažādos veidos. Viena no tām ir vienkārši pievienojiet nelielu iekšējo akumulatoru SRAM ierīcei, lai tad, kad ārēja jauda ir izslēgta, SRAM joprojām saglabātu savu saturu. Vēl viens variants ir apvienot SRAM ar EEPROM vienā iepakojumā. Pēc ieslēgšanas, dati tiek kopēti no EEPROM uz SRAM. Darbības laikā, dati ir nolasītu un ierakstīti SRAM. Ja strāvas padeves ir pārtraukšas, datus kopē uz EEPROM.

Piekļuve Atmiņai

Daudzi mikrokontrolleri nāk ar jau ierakstītu programmu un datu atmiņu. Parasti, programmas atmiņa būs uz Flash EEPROM pamata un datu atmiņā sastāvēs no SRAM un EEPROM. Konkrēto adresi pievērš atmiņas adresē divos veidos:

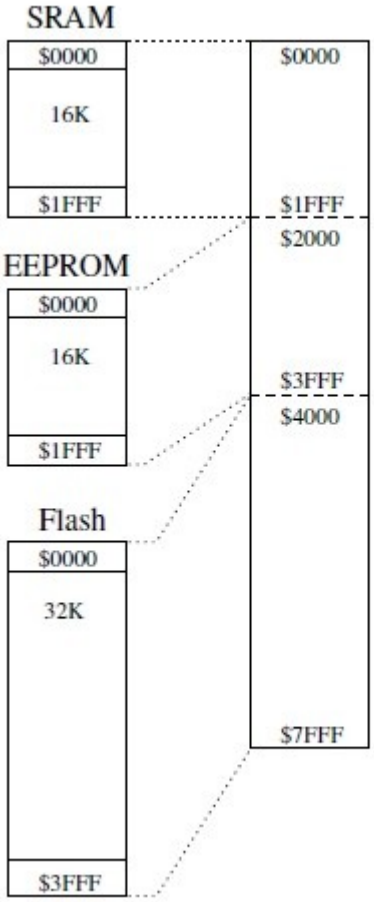
Katra atmiņa ir adresēta atsevišķi, skat. att. 2.7



Attēls 2.7 Atdalīta atmiņas adresācija.

Trīs dažādu atmiņas veidu adreses diapazoni var būt tāds vienādi. Programmētājs nosaka, kura atmiņa būs pieejama pielietojot konkrēto piekļuves metodi. Piemēram, lai piekļūtu EEPROM, pielieto īpašu EEPROM indeksu reģistru.

Visiem atmiņas veidiem ir kopīgais adrešu diapazons, skat. att. 2.8



Šeit programmētājs piekļūst EEPROM tādā pašā veidā kā SRAM. Mikrokontrolleru izmanto adresi lai noteiktu, kurai atmiņai te notiek piekļuve. Piemēram, EEPROM varētu piešķirt adrešu klāstu 0x1000 - 0x2000, bet SRAM diapazonu 0x2000 - 0x3000. Tagad, kad programmētājs piekļust adresei 0x1800, mikrovadības zina, ka tas ir EEPROM diapazonā, un tāpēc piekļūs EEPROM. Kaut arī šī metode ir ļoti vienkārša, tā ir arī pēc būtības vissnedrošāka: nepareiza adrese var novest pie nepareizas atmiņas veida nolasīšanas. Tas varētu būt īpaši bīstami, ja jus nejauši piekļuvat SRAM, nevis EEPROM - ar biežo piekļuvi EEPROM var nolietoties dažu minūšu laikā. Atsevišķa atmiņas adresācija, no otras puses, nāk ar netiešu aizsardzību pret nepareiza veida piekļuvi.

Piekļūstot baitu adresētai atmiņas ir viens moments, kas jāņem vērā: Pieņemsim, ka 16 bitu kontrolleris raksta vārdu (divi baiti) iekš SRAM, teiksim 0x0100 adresē. Vārds sastāv no zema un augsta baita. Kādā secībā baiti tiek ierakstīti? Ir divi varianti: zems baits varētu doties uz 0x0100 un augstāis baits uz nākamo adresi (0x0101), vai otrādi.

Tā ir problēma, ko sauc par „Big Endian”:

Big Endian arhitektūra glabā augsto baitu pirmo. Tātad, ja jūs rakstāt angļu valodas vārdu 0x1234 adresē 0x0100, tad augstais baits 0x12 aizietu uz adresi 0x0100, un zemais baits 0x34 uz adresi 0x0101. Nosaukums ir izveidots līdzīgi: Lielais vārda gals (BIG) glabājas pirmais - tātad, to sauc Big Endian.

Little Endian: Little Endian arhitektūras piekļūst atmiņai otrādi (mazais beigu vārds tiek ierakstīts pirmais). Šeit zemais baits tiek saglabāts pirmais. Rakstot 0x1234 adresē 0x0100 šadā veidā ieraksta 0x34 adresē 0x0100 un 0x12 adresē 0x0101.