1 VON-NEUMANN MOTAKO ARKITEKTURA

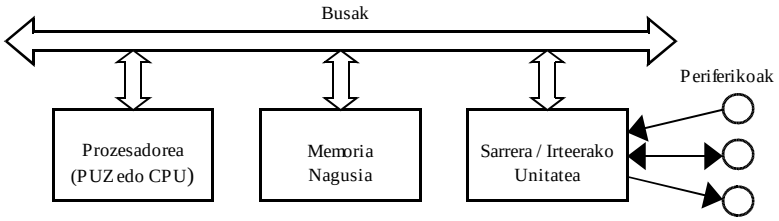
1 VON-NEUMANN MOTAKO ARKITEKTURA

1.1 SARRERA

Kapitulu honen helburua oinarrizko konputagailu baten funtzionamendua zehaztea da. Oro har, informazioa prozesatzen duen gailua da konputagailua; hartarako, hainbat agindu edo ekintza berezi exekutatzen ditu datu jakinen gainean. Hasteko, bada, konputagailuen oinarrizko egitura azalduko da, von Neumann izenekoa, haren funtsezko lau osagaiak aurkeztuz: prozesadorea, memoria nagusia, sarrera/irteerako gailuak eta busak. Memoria da konputagailuak erabiltzen duen informazioaren biltegia, eta haren egitura azaltzea izango da hirugarren atalaren helburua, non, adibidez, helbide, eduki eta edukiera kontzeptuak azalduko diren.

1.2 KONPUTAGAILUEN VON NEUMANN EGITURA

Konputagailu digitalen oinarrizko egitura 1945. urtean John von Neumann-ek ezarritakoa da, eta, aldaketa gutxi batzuekin, oraindik ere konputagailu gehienetan erabiltzen da. **Von Neumann** motako konputagailua hainbat unitatek osatuta dago; haren egitura orokorra 1.1 irudian ageri da.



1.1 irudia. Von Neumann motako konputagailuaren egitura orokorra.

Makina hori **aginduak**, hots, eragiketa bereziak, **exekutatzeko** gai da. Agindu horiek **datuak** prozesatzen dituzte, haiekin hainbat funtzio exekutatuz. Hau da, **programak —**aginduak eta datuak— prozesatzen edo exekutatzen ditu konputagailuak. Aginduak zein datuak konputagailuaren **memoria nagusian (MN)** gordetzen dira.

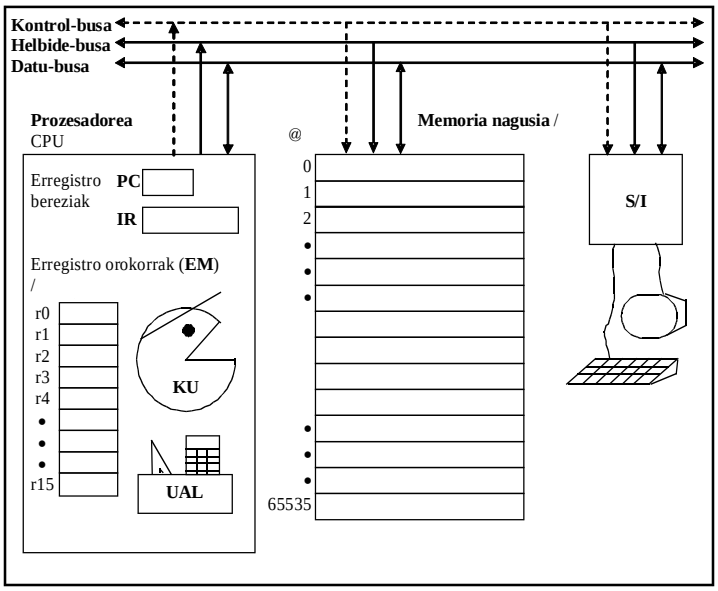
Prozesatzeko Unitate Zentrala (PUZ) —ingelesezko terminoa, **CPU** (*Central Processing Unit*), erabiliagoa da— edo soilik **prozesadorea**, aginduak exekutatzeaz arduratzen da, hau da, zereginak zehazten ditu eta egitekoak betearazten dizkie gailu guztiei. Exekutatu behar diren aginduak, neurri handi batean behintzat, oinarrizko funtzioak dira: aritmetikoak (batuketa, biderketa, ...) zein logikoak (AND, OR, ...). Datuak edo eragigaiak memoria nagusian daude; bertatik irakurriko dira, eta bertan utziko dira emaitzak.

Sarrera/irteerako (S/I-ko) unitatea konputagailuaren eta kanpo aldearen arteko informazio-transferentziaz arduratzen da. Informazioa kanporatzeko edota barneratzeko, **S/I-ko gailuak** edo **periferikoak** erabiltzen dira: pantaila, teklatua, inprimagailuak, diskoak, eta abar.

Prozesadorearen, memoria nagusiaren eta sarrera/irteerako unitateen arteko komunikazioa (osagaien arteko konexioak), **bus** izeneko bide espezifikoen bitartez egiten da.

1.2.1 Konputagailu baten ikuspegi funtzionala

1.2 irudian, konputagailu baten eskema logiko sinple bat ageri da bere oinarrizko osagai edo blokeekin.



1.2 irudia. Konputagailu baten ikuspegi funtzionala

- **Prozesadorea**. Aginduen exekuzioaz arduratzen da. Bertan, aginduetan adierazitakoa egingo da, hau da, aginduak exekutatuko dira. Horretarako, osagai hauek izango ditu:
  - **Kontrol-unitatea, KU.** Haren lana hau izango da: memoria nagusitik aginduak irakurri, aztertu, eta, azkenik, haiek exekutatzeko konputagailuak behar dituen **kontrol-seinaleak** sortu.
  - **Unitate aritmetiko/logikoa, UAL** (*Arithmetic and Logic Unit*, ALU). Oinarrizko eragiketak (batuketak, kenketak, AND, OR, eta abar) egiten ditu.
  - **Erregistro-multzoa, EM.** Barne-biltegia da, edukiera gutxikoa baina azkarra, eta behin behineko informazioa gordetzeko erabiltzen da. Oro har, memoria nagusia motela da konputagailuaren prozesu-abiadurarekin alderatuta (adib. memoriako **atzipen-denbora** 50 ns izan daiteke, baina aginduen exekuzioa kontrolatzen duen **erlojuaren periodoa** 1 ns izan daiteke). Hori dela eta, behin baino gehiagotan erabili behar den informazioa (tarteko emaitzak, indizeak, etab.) erregistroetan gordetzen da.

    Oro har, bi erregistro mota daude: (1) **helburu orokorreko erregistroak**, programatzaileak kalkuluak egiteko erabil ditzakeenak; eta (2) **helburu bereziko erregistroak**, prozesadoreak bakarrik erabil ditzakeenak aginduak zuzen exekutatzeko. Erregistro-multzoa lehenengoek osatzen dute. ARM prozesadoreak adibidez 16 erregistro orokor ditu. Horrez gain, helburu bereziko bi erregistro izango ditu: PC eta IR.
- **Memoria nagusia** (aurrerantzean, memoria soilik). Prozesadorerako aginduak zein datuak gordetzen ditu. Erregistroak baino **edukiera** askoz handiagoa du; **atzipena,** ordea, motelagoa da

(adibidez, 50 ns memoria atzitzeko eta 1 ns erregistro bat atzitzeko; edukiera aldetik, berriz, milioika aldiz handiagoa da memoria).

- **Sarrera/irteerako unitatea.** Esan bezala, konputagailuaren eta ingurunearen arteko komunikazioa ahalbidetzen du. Periferiko izena hartzen duten unitateetako datu-transferentziaz arduratzen da (pantaila, teklatua, diskoak, zintak, inprimagailua, etab.).
- **Busak**. Konputagailuaren osagai guztiak konektatzen dituzte. Busen bidez, osagaien arteko informazio-trukea gauzatzen da. Garraiatzen den informazioa hiru motakoa izan daiteke: (a) **helbideak**, aginduen zein datuen memoria-posizioak adierazteko —ohikoa da @ ikurra erabiltzea helbidea adierazteko—; (b) **kontrol-informazioa**, bete behar den ataza zehazteko —irakurketa, idazketa, etab.—; eta (c) **datuak**, memoriaren, erregistroen edo periferikoen edukia.

1.3 MEMORIAREN OINARRIZKO ANTOLAKETA

Lehen aipatu den moduan, memoriak gordetzen du CPUk prozesatuko duen informazioa: aginduak eta datuak. Memoriarekin oinarrizko bi eragiketa egiten dira: **irakurketa** eta **idazketa.** Memoria nagusia eta prozesadorea komunikatzeko, busak erabiltzen dira: helbide-busa, memoriako helbideak adierazteko; datu-busa, datuak garraiatzeko; eta kontrol-busa, eragiketa jakin bat adierazteko. Atal honetan, memoriaren oinarrizko egitura ikusiko dugu[1].

Digitu bitar (*binary digit*) edo **bit** bat gordetzeko gai diren osagai fisikoez eraikitzen dira memoriak. Bit bat, fisikoki adieraz daitekeen informazio-unitate txikiena da, eta 0 edo 1 balioak har ditzake. Memoria baten tamaina edo **edukiera** bit kopuru gisa adieraz daitekeen arren, unitate handiagoak erabili ohi dira. Erabiliena, hain zuzen, **byte**a da: 8 bit. Memoriaren edukiera neurtzeko, bytearen multiploak erabiltzen dira[2]:

1 kB (kilobyte):	2^{10} byte edo 1024 byte
1 MB (megabyte):	2^{20} byte edo 2^{10} kB
1 GB (gigabyte):	2^{30} byte edo 2^{10} MB
1 TB (terabyte):	2^{40} byte edo 2^{10} GB

Memoriako bitak hainbat **posiziotan** edo **gelaxkatan** elkartzen dira, guztiak tamaina edo bit kopuru berekoak (byte bat da ohikoena). Gelaxkako bitek adierazten duten informazioa memoria-posizio horren **edukia** da. Memoriako posizio bakoitzak **helbide** bat dauka, eta helbide hori erabili behar da gelaxka horretan eragiketa bat, irakurketa edo idazketa, egin ahal izateko. Gelaxka bat da **helbidera daiteken unitate minimoa.** Beraz, gelaxka baten edukia da memoriaren eta prozesadorearen artean trukatzen den informazio kantitate minimoa.

Adi, ez nahastu bi kontzeptu hauek: memoria-gelaxka baten helbidea —haren kokapena memorian, gelaxka bakoitzerako desberdina eta beti bera—, eta haren edukia —une horretan gordetzen duen informazioa, eta aldakorra—.

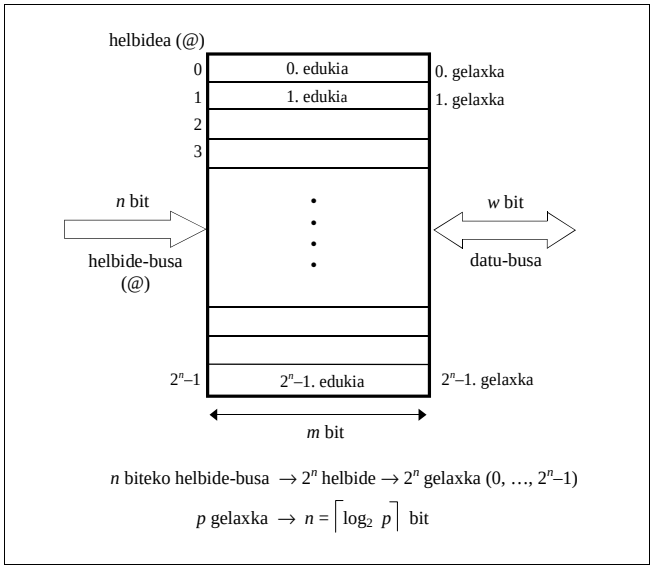
Helbideen zein datuen barne-adierazpena $r = 2$ oinarrian egiten da, hots, bitarrez.

---

1 Prozesadoreen kalkulu-abiadura gero eta altuagoei erantzun egokia emateko, memoria-sistemak ere gero eta konplexuagoak dira (cache memoria, busak, eta abar). Horiek azaltzea ikasgai honen helburutik kanpo dago. Hori dela eta, oinarrizko kontzeptuak eta egiturak besterik ez dira azalduko, eta kontzeptu horiek integratzen dituen eredu sinple bat proposatu.

2 Salbuespen gisa, memoriaren edukiera adierazten denean, k multiploak ez du ohiko 1000 adierazten, 1024 (2^{10}) baizik. Adi! kasu bakarra da hori; beste magnitude guztietan, betiko esanahia mantentzen da: k = 10^3, M = 10^6, eta abar.

Helbideak emateko, helbide-busa erabiltzen da; haren tamaina, zabalera edo bit kopurua memoriako gelaxka kopuruarekin zuzenki lotuta dago. Izan ere, helbide-buseko bit kopuruak nahikoa izan behar du memoria-posizio guztiak helbideratzeko. Gogoratu: n bitekin 2^n helbide desberdin adieraz daitezke; edo, alderantziz, p gelaxka dituen memoria bat helbideratzeko, $\lceil \log_2 p \rceil$ bit behar dira helbide-busean. 1.3 irudian, memoriaren egitura logikoa ageri da. Hurrengo hiru adibideetan, aurreko kontzeptuen arteko erlazioak lantzen dira.



1.3 irudia. Memoriaren antolaketa logikoa.

1.1 Adibidea

Memoria baten gelaxka bakoitza byte batekoa da, eta memoriaren edukiera, guztira, 2 kilobyte da. Adieraz itzazu memoria horrek duen posizio kopurua eta helbide-busaren zabalera. Errepikatu aurrekoa gelaxkak 2 eta 4 bytekoak izanik.

Tam_{mem} memoriaren tamaina osoa eta Tam_{pos} memoria-posizio bakoitzaren tamaina izanik, memoriaren posizio kopurua[3] honela lor daiteke:

$$Posizio\_kop = Tam_{mem} / Tam_{pos}$$

Beraz, helbide-busaren zabalera edo tamaina hau izango da:

$$Helbide\_bus\_tamaina = \lceil \log_2 (Tam_{mem} / Tam_{pos}) \rceil$$

Emaitzak hauek dira:

Edukiera (osoa)	Gelaxken tamaina	Posizio kopurua	Helbide-busaren tamaina ($\log_2 pos\_kop$)
2 kilobyte	1 byte	2048	11
2 kilobyte	2 byte	1024	10
2 kilobyte	4 byte	512	9

[3] "Kopuru" eta "helbide" hitzak hainbat aldiz erabili behar dira; askotan, laburdurak erabiltzen dira: # ikurra kopurua adierazteko, eta @ ikurra helbidea adierazteko.

1.2 Adibidea

10 biteko helbide-busa erabiltzen duen memoria bat izanik, kalkula ezazu memoriaren edukiera gelaxkak byte batekoak badira. Eta gelaxkak 2 bytekoak badira?

Helbide-busaren bit kopurua jakinik, *n*, posizio kopurua honako hau da:

$$Posizio\_kop = 2^n$$

Adibide honetan, helbide-busaren zabalera 10 bitekoa denez, orduan $2^{10} = 1024$ gelaxka edo posizio daude memorian.

Memoriaren tamaina kalkulatzeko, posizio kopurua eta haien tamaina biderkatu behar dira:

$$Tam_{mem}(byte) = Posizio\_kop \times Tam_{pos}(byte)$$

Proposatzen diren bi kasuetarako, honako emaitza hauek izango ditugu:

Posizio kopurua	Gelaxken tamaina	Memoriaren edukiera
1024	1 byte	1 kilobyte
1024	2 byte	2 kilobyte

1.3 Adibidea

Memoria batek 2 byteko posizioak ditu, eta haren helbide-busa 10 bitekoa da. Zein da memoriaren tamaina eta posizio kopurua? Eta helbide-busa 11 bitekoa balitz?

Aurreko adibidean esan den moduan, memoriako posizio kopurua eta helbide-busaren tamaina honela erlazionatuta daude:

$$posizio\_kop = 2^n$$

Memoriaren tamaina kalkulatzeko, hauxe besterik ez da egin behar: biderkatu posizio kopurua posizio bakoitzaren tamainarekin.

Proposatzen diren bi kasuetarako, honako emaitza hauek izango ditugu:

Posizio-tamaina	Helbide-busa	Posizio kopurua	Memoriaren edukiera
2 byte	10 bit	1024 (1 k)	2 kilobyte
2 byte	11 bit	2048 (2 k)	4 kilobyte

Prozesadorearen eta memoriaren arteko datu-transferentziak egiteko, datu-busa erabiltzen da. Konputagailu batek luzera askotariko **datuak** prozesatzen ditu (8, 16, 32, 64 bit, etab.), baina horietako bat tamaina “pribilegiatua” da. Oinarrizko tamaina horri **hitza** deritzo, eta prozesadorearen barne-egitura —erregistroen tamaina, datu-busaren zabalera, UALean egiten diren eragiketen tamaina, etab.— neurri horretarako dago bereziki diseinatuta. Gaur egun, hitzaren tamaina 64 bit da konputagailu komertzial gehienetan, baina badira 32 biteko hitzak erabiltzen dituzten konputagailuak ere. Gainerako datu-tamainak hitzaren arabera deskribatu ohi dira; adibidez, 32 biteko hitzen kasuan, hauek dira datu-tamaina posible batzuk: bytea (8 bit), hitz erdia (16 bit), hitza (32 bit), hitz bikoitza (64 bit).

Datu-busaren tamainak edo zabalerak zedarritzen du transferentzia batean garraia daiteken bit kopuru maximoa. Aipatu berri dugun moduan, datu-busaren zabalera eta hitzaren tamaina bat datoz konputagailu gehienetan. Hots, 64 biteko prozesadoreetan, datu-busa 64 bitekoa da (gutxienez).

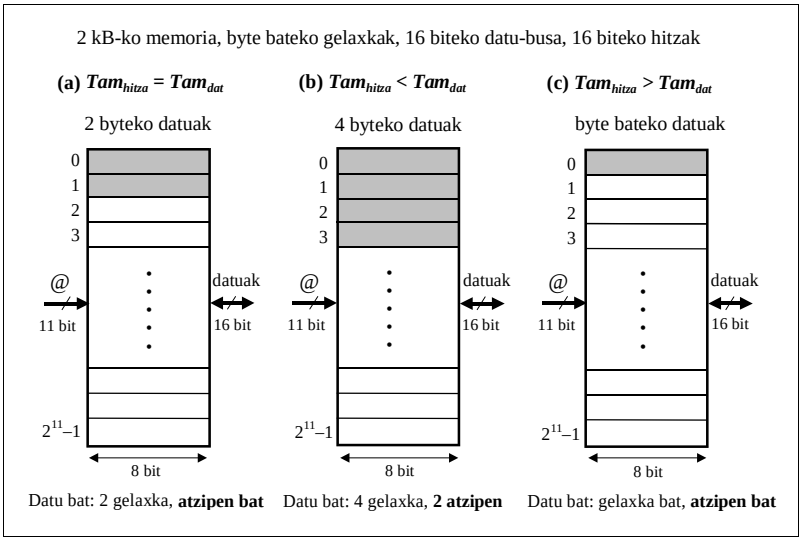
Datu bat idazteko edo irakurtzeko, memoria atzitu behar da, eta atzipen batean hitz bat (edo hitz zati bat) maneiatzen da, irakurtzeko zein idazteko. Oro har, memoriako eragiketa batean egin behar den **atzipen kopurua** idatzi/irakurri behar den **datuaren tamainaren arabera** da. Adi: atzipen kopuruak eragin zuzena du memoriako agindu baten exekuzio-denboran! Memoriako eragiketa baten atzipen kopurua kalkulatzeko, honako hau hartu behar dugu kontuan:

baldin ($Tam_{dat} \leq Tam_{hitza}$) → atzipen bakar bat

baldin ($Tam_{dat} > Tam_{hitza}$) → $\lceil Tam_{dat} / Tam_{hitza} \rceil$ atzipen

Gogoratu: konputagailu gehienetan, $Tam_{hitza} = Tam_{datu\text{-}busa}$ da.

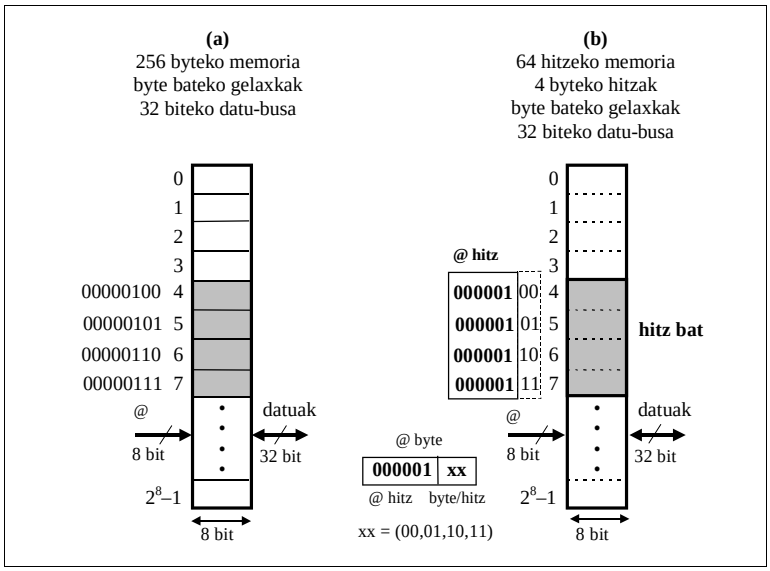
1.4 irudiak hiru egoera desberdin erakusten ditu, hitzaren eta datuaren tamaina erkatuz. (a) eta (c) kasuetan, nahikoa da atzipen batekin datua prozesatzeko; (b) kasuan, aldiz, bi atzipen behar dira.



1.4 irudia. Memoriako atzipen kopuruak datuaren eta memoriako hitzen tamainak kontuan harturik.

Lehen aipatu dugun moduan, memoriako gelaxka bat da helbidera daitekeen informazio kopururik txikiena. Kopuru hori byte bat denean (ohikoena), byteak helbideratzen direla esaten da (1.5a irudia). Hots, memoriako byte guztiek helbide desberdina dute memorian. Horrek ez du esan nahi, hala ere, memoriako atzipenak beti bytekoak izango direnik. Izan ere, informazio gehiago jaso ohi da (2 byte, 4 byte, etab.) memoriako atzipen bakoitzean.

Byteak helbideratzen dituzten memorietan, hitz batek posizio bat baino gehiago okupatuko du eskuarki, hitzaren byte kopuruaren arabera. Beraz, bi helbide mota bereiz litezke: batetik, hitzaren helbidea; eta, bestetik, hitza osatzen duten byteen helbideak, memoriak behar dituenak. Hitz jakin bat atzitzeko eman behar den helbidea haren lehenengo byteari dagokiona da. Adibidez, 1.5b irudian, 256 posizioko (byteko) memoria ageri da, 4 byteko 64 hitzek osatuta. 4 byteko hitzak ditugunez, hitzak honako helbide hauetan daude: 0–3, 4–7, 8–11, ...



1.5 irudia. Byteak helbideratzen dituen memoria.

Byte baten helbideak, bitetan jarrita, bi eremu hauetan bana daitezke: byte hori duen hitzaren helbidea, “logikoa”; eta bytearen posizioa hitz horren barnean:

@ byte	@ hitza	byte
n bit	*n* – *k* bit	*k* bit

n biteko helbide batetik abiatuta —$n = \log_2$ (*memoriako byte kop.*)—, helbideen eremu bakoitzaren bit kopurua hau da:

- bytearen posizioa hitz horretan → $k = \log_2$ (*hitzen byte kop.*)
- hitzaren helbidea memorian → $n - k = \log_2$ (*mem. hitz kop.*)

Adibidez, 12 helbidea honela interpreta daiteke 1.5b irudiko memorian:

helbidearen bit kopurua, $n = 8$ bit

hitz baten barruan, byte bat adierazteko, $k = 2$ bit

hitzen helbideak, beraz, $n - k = 6$ bit

(12) 0000 1100 →	0000 11	00
	3 hitza	0 bytea

Oro har, byte baten helbidea emanda —hamartarrez esaterako— honela kalkulatzen dira byte horri dagokion hitzaren helbidea eta bytearen posizioa hitzaren barnean:

@hitza = *@byte* ***div*** *hitzaren_tamaina (byte)*

byte = *@byte* ***mod*** *hitzaren_tamaina (byte)*

non *div* zatiketaren zatidura eta *mod* zatiketaren hondarra diren. Aurreko adibidean, @byte = 12 eta hitzaren tamaina = 4 byte; beraz,

@hitza = 12 *div* 4 = 3

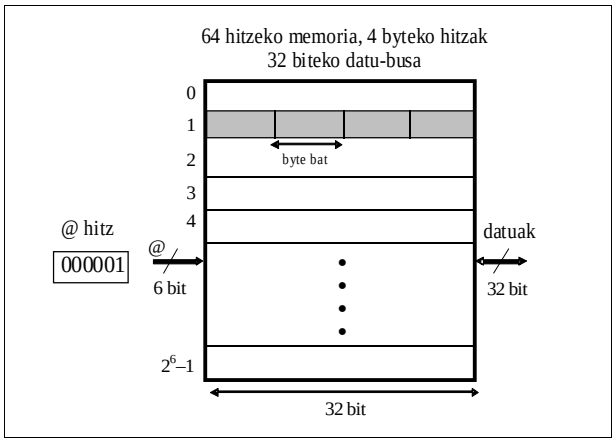
byte = 12 *mod* 4 = 0

Egungo konputagailu guztietan, **atzipen lerrokatuak** besterik ez dira onartzen; hots, hitzen hasierako byteak helbideratu behar dira (byte eremuan, bit guztiek 0koak izan behar dute). Hala egiten ez denean, helbideratzea ez-lerrokatua da. Atzipen ez-lerrokatuak ez dira erabiltzen, beste gauza batzuen artean, motelagoak direlako. Izan ere, memoriako bi hitz irakurri behar dira behar den hitza

eskuratzeko. Adibidez, 4 byteko hitz bat eskuratzeko, 5 helbidetik abiatuta (000001-01 helbidea, ez-lerrokatua), 1 hitzaren 3 byte (memoriako 5, 6 eta 7 helbideak) eta 2 hitzaren byte bat (memoriako 8 helbidea) irakurri beharko genituzke (ikus 1.5 irudia).

Arestian esan dugun moduan, tamaina askotariko datuak prozesatzen dituzte konputagailuen aginduek. Zenbait kasutan, ez da hitz oso bat idazten edo irakurtzen, baizik eta zati bat: byte bat, hitz erdia, ... Adibidez, 1.5 irudiaren eskemari jarraituz, 000001-00 helbidearekin hitz osoa, hitz erdia (2 byte) edo byte bakarra (hitzaren lehenengo bytea) atzi daitezke. Kasu horietan, aginduak berak adieraziko du irakurri behar den zatia, eta zati hori besterik ez da transferituko datu-busean.

Byteak helbideratzea da ohikoena; hala ere, zenbait sistematan hitza da helbidera daiteken informazio-unitate txikiena (1.6 irudia). Hitzen helbideetan ez dira bereiziko hitza osatzen duten byteak. Helbideragarria den unitate txikiena hitza denez, arazoa hitza baino txikiagoak diren datuak atzitzean datza; adibidez, byte bat. Hori egin ahal izateko, softwarearen laguntza behar da; prozesadoreak desplazamendu- eta maskara-eragiketak (2. kapituluan aztertuko ditugu) exekutatu behar ditu datu-busetik helduko den hitzaren gainean, behar duen hitz zatia eskuratzeko eta gainerakoa baztertzeko. Gaur egungo konputagailu komertzial gehienek byteak helbideratzen dituzte memorian, eta, hala, software-gainkargarik gabe (maskarak, desplazamenduak, etab.) atzi daitezke hitza baino txikiagoak diren datuak.



1.6 irudia. Hitzak helbideratzen dituen memoria.