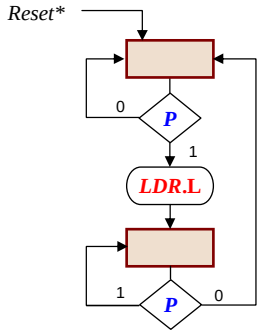
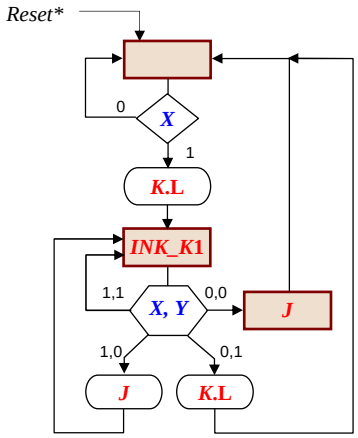
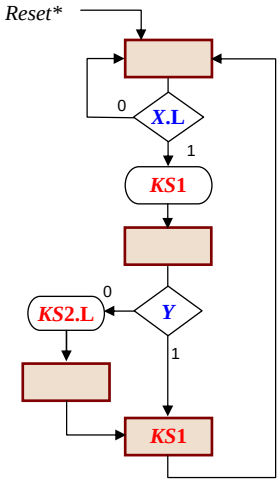
6 GAIA: ARIKETAK

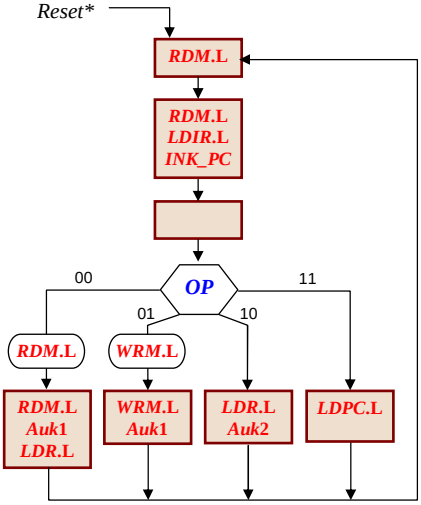
Liburuan proposatutako ariketak

6.1. Sistema digital batzuen kontrol-algoritmoak ageri dira irudietan. Kasu bakoitzerako: (a) kodetu egoerak eta sortu egoeren arteko trantsizio-taula; (b) eraiki kontrol-unitatea multiplexoreak eta D biegonkorrak erabiliz, eta sortu kontrol-seinaleak; eta (c) egin kontrol-unitatearen funtzionamendua islatuko duen kronograma bat. Kronogrametan, batetik, sarrera-seinaleak, eta, bestetik, automataren egoerak eta kontrol-seinaleak ageri behar dira; sarrera-seinaleak definitu beharko dituzu, nahi duzun moduan, automataren egoera-trantsizio guztiak analizatu ahal izateko (hartu, erreferentzia gisa, ariketa ebatzietan erabilitakoak). Kontrol-seinaleen eta sarreren logika (.H edo .L) kontrol-algoritmoetan bertan ageri da.



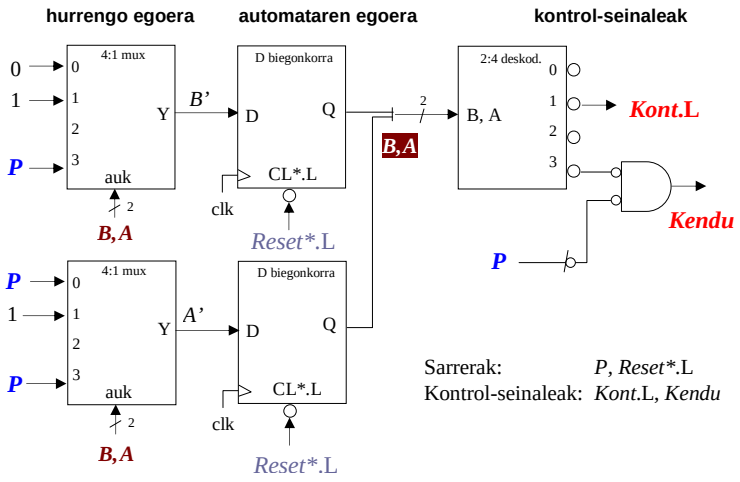






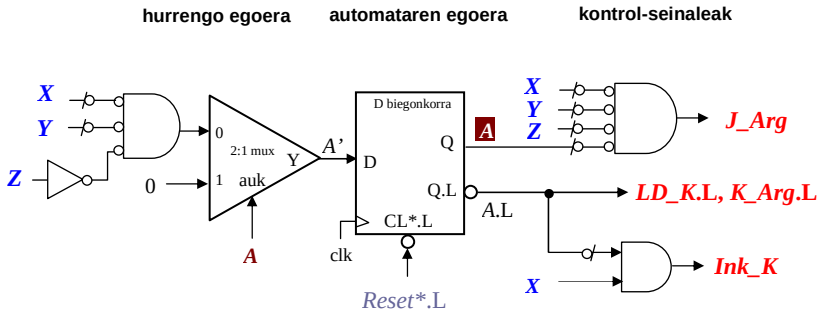
6.2. Bi sistema digital sinpleren kontrol-unitateak ageri dira irudietan. Kasu bakoitzerako: (a) egin kontrol-unitatearen funtzionamendua islatzen duen kronograma bat. Beharrezkoa da kronogrametan agertzea, batetik, sarrera-seinaleak eta, bestetik, automataren egoerak eta kontrol-seinaleak; sarrera-seinaleak definitu beharko dituzu, sistemaren portaera osoa analizatu ahal izateko; eta (b) idatzi kontrol-unitateari dagokion ASM grafoa.

1. kontrol-unitatea



Sarrerak: *P*, *Reset**.L
Kontrol-seinaleak: *Kont*.L, *Kendu*

2. kontrol-unitatea

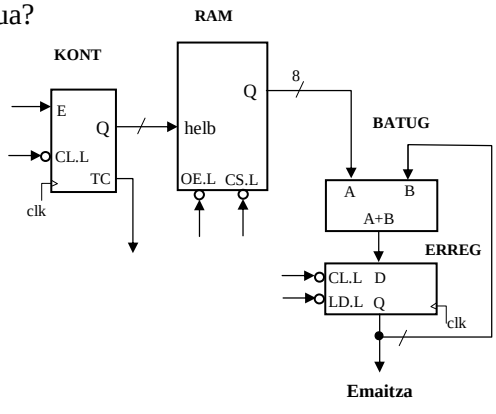


Sarrerak: *X*, *Y*, *Z*, *Reset**.L
Kontrol-seinaleak: *J_Arg*, *K_Arg*.L, *LD_K*.L, *Ink_K*

6.3. Sistema digital baten atal batean, RAM memoria bat erabiltzen da byte bateko datuak gordetzeko. *HASI* izeneko kontrol-seinalea aktibatzen denean, lehen 16 datuen batura kalkulatu behar da, eta horretarako irudiko zirkuitua antolatu da.

Egin ezazu kontrol-algoritmo bat zirkuitua kontrolatzeko, aipatutako eragiketa egin dezan. Gero, eraiki ezazu kontrol-unitatea, sortu kontrol-seinaleak eta analizatu sistema osoaren portaera kronograma baten bidez (memoriaren edukia, adibidez, honako hau da: 0, 1, 2, 3...).

Zenbat bitekoak izango dira batugailua eta erregistroa, ez badugu nahi gainezkatzea gertatzerik? eta kontagailua?

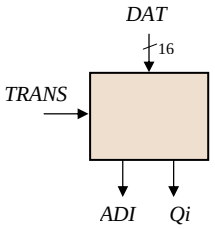


6.4. Sistema digital batek 16 biteko datu bat transmititu behar du seriean, bitez bit. Horretarako, desplazamendu-erregistro bat erabiltzen du. Hasieran, *TRANS* seinalea aktibatzen denean, transmititu behar den datua (sarrera bat) desplazamendu-erregistroan kargatzen da (bit guztiak batera), eta, gero, 16 desplazamendu egiten dira, eskuinera; hala, bit guztiak erregistroko Q_0 irteeran ageriko dira, zikloz ziklo; irteera hori da, hain zuzen ere, transmitituko den bita.

Adibide sinple batez (4 bit) ikus daiteke sistemaren portaera:

transmititu behar den datua:	0011
kargatu erregistroan (*TRANS* = 1):	$Q_3Q_2Q_1\mathbf{Q_0} = 001\mathbf{1}$
desplazatu bit bat eskuinera	$Q_3Q_2Q_1\mathbf{Q_0} = x00\mathbf{1}$
desplazatu bit bat eskuinera	$Q_3Q_2Q_1\mathbf{Q_0} = xx0\mathbf{0}$
desplazatu bit bat eskuinera	$Q_3Q_2Q_1\mathbf{Q_0} = xxx\mathbf{0}$
desplazatu bit bat eskuinera	$Q_3Q_2Q_1\mathbf{Q_0} = xxxx$

Beraz, $Q_0 \rightarrow 1, 1, 0, 0$

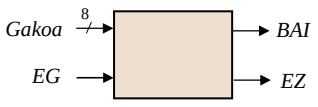


Bitak transmititzeaz gain, transmisioaren hasiera eta bukaera adierazi behar dira: transmisioa hasten denean, seinale batek (*ADI* esaterako) 1 balioa hartuko du, eta bukaeran 0a.

Diseinatu sistema hori honako zirkuitu hauek erabiliz: desplazamendu-erregistro bat datua hartzeko eta desplazatzeko, kontagailu bat bit guztiak transmititu direla egiaztatzeko, eta JK biegonkor bat (*ADI*) transmisioa egiten ari dela adierazteko. Antolatu prozesu-unitatea, zehaztu erabili behar diren kontrol-seinaleak eta, gero, proposa ezazu ASM kontrol-algoritmo bat eta dagokion kontrol-unitatea.

Diseinua ondo dagoela egiaztatzeko, egin ezazu kronograma bat eta aztertu, pausoz pauso, diseinatu duzun sistemaren portaera. Errore bat detektatzean, zuzendu eta errepikatu kronograma.

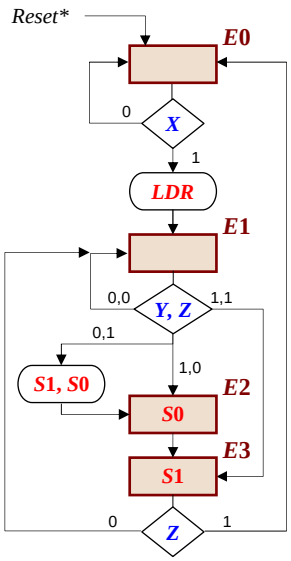
6.5. Sistema digital jakin batek gako-hitz batzuk erabiltzen ditu eragiketa jakin bat egiteko. Sistemaren azpiatal batek kontrolatzen ditu gakoak (8 bitekoak) eta adierazten du zuzenak direnetz. Onartzen diren gako guztiak 16 posizioko ROM memoria batean daude. Gako bat egiaztatu behar denean, *EG* seinalea aktibatuko da; une horretan, gakoa erregistro batean kargatu eta ROM memorian dauden gakoekin konparatuko da, banan-banan. Sistemak aurkitzen badu memorian berdina den gako bat, ontzat emango du sarrerako gakoa eta BAI seinalea aktibatuko du; aldiz, memoria osoa irakurrita, ez badu aurkitu gakoa, EZ seinalea sortuko du.



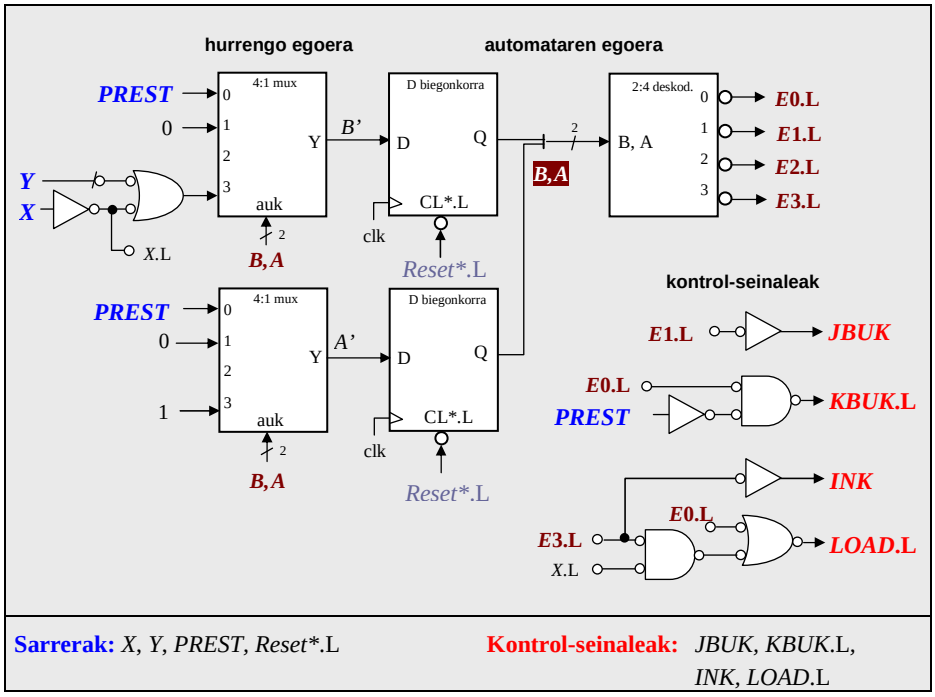
Diseina ezazu sistema horretarako prozesu-unitate bat zirkuitu hauek erabiliz: 16×8 biteko ROM memoria bat, 4 biteko kontagailu bat (ROM memoriaren helbideak emateko), 8 biteko erregistro bat (prozesatzen ari den gakoa gordetzeko), 8 biteko konparagailu bat (gakoa eta memoriako hitzak konparatzeko) eta bi JK biegonkor (*BAI* eta *EZ* adierazleetarako). Adierazi argi eta garbi kontrolatu behar dituzun kontrol-seinaleak. Gero, egin ezazu kontrol-algoritmoa, eta eraiki kontrol-unitatea eta kontrol-seinaleak.

Liburuan ebatzitako ariketak

6.1. Sistema digital baten kontrol-unitatearen portaera islatzen da irudiko ASM algoritmoan. Lau egoera ditu —*E*0, *E*1, *E*2 eta *E*3—, hiru sarrera prozesatzen ditu —*X*, *Y*.L eta Z—, eta hiru kontrol-seinale sortzen ditu —*LDR*.L, *S*1 eta *S*0—.
ASM algoritmo horri dagokion kontrol-unitatea sortu behar da, eta haren portaera irudikatu kronograma batean.



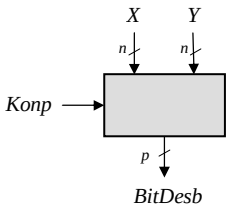
6.2. Sistema digital baten kontrol-unitatearen eskema logikoa ageri da ondoko irudian.. Zirkuituaren portaera analizatu behar da, eta "exekutatzen" duen kontrol-algoritmoa ondorioztatu.



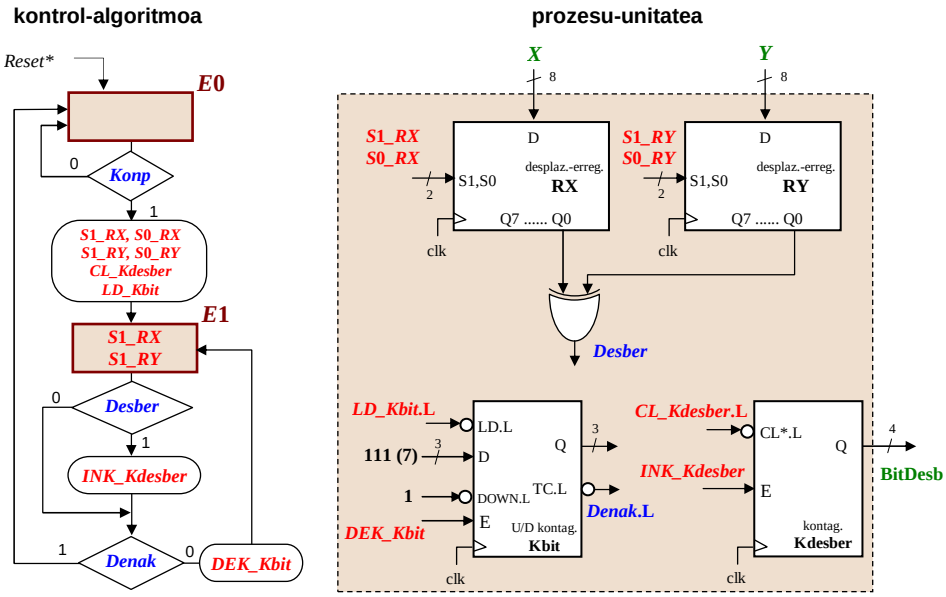
Sarrerak: *X*, *Y*, *PREST*, *Reset**.L

Kontrol-seinaleak: *JBUK*, *KBUK*.L, *INK*, *LOAD*.L

6.3. Sistema digital bat diseinatu da n biteko bi zenbaki erkatzeko; hain zuzen ere, kalkulatu behar da zenbat bitetan diren desberdinak bi zenbakiak. Esaterako, sarrerako datuak *X* = 1**10**1 **0**11**0** eta *Y* = 1**00**0 **1**11**1** badira, emaitza 4 izango da.



Sistema digital horren kontrol-algoritmoa eta prozesu-unitatea irudikoak dira (n = 8 bit kasurako).

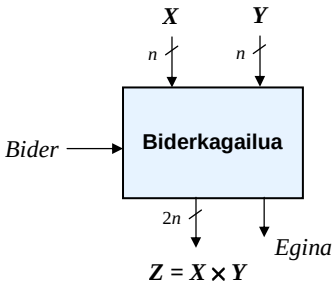


Datuak bitez bit desplazatzen dira eskuinera, eta pisu txikieneko bitak konparatzen dira, XOR ate baten bidez. Konparatu den bit kopurua kontrolatzeko, Kbit kontagailua erabiltzen da, eta Kdesber kontagailua, bit desberdinen kopurua adierazteko.

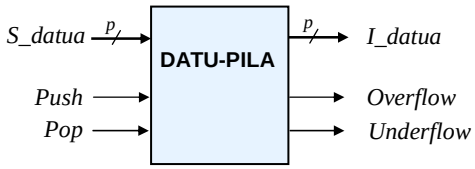
ASM algoritmoari dagokion kontrol-unitatea sortu behar da (multiplexore eta biegonkorren bidez) eta, sistemaren funtzionamendua egiaztatzeko, kronograma bat egin (kronogramarako datuak: *Reset*, *Konp*, *X* eta *Y*).

6.4. Konputagailuetan gehien exekutatzen diren eragiketa aritmetikoak batuketa eta biderketa dira, eta hainbat algoritmo daude eragiketa horiek modurik eraginkorrenean egiteko. Diseinu-ariketak lantzeko asmoz, biderkagailu sinpleena diseinatuko dugu adibide honetan. $X \times Y$ egiteko algoritmorik sinpleena oso ezaguna da: gehitu X zenbakia Y aldiz: $X + X + ... + X$ (Y aldiz). Jakina, biderketak egiteko modu hori ez da batere eraginkorra: eragiketaren erantzun-denbora (latentzia) handia da, eta, gainera, ez da konstantea, biderkatzen diren zenbakien araberakoa baita. Hala ere, aukera ematen du zirkuitu digital oso bat diseinatzeko.

Beraz, hau da proposamena: n biteko bi zenbakien biderkagailua diseinatu behar da. Irudian ageri dira zirkuituaren sarrerak eta irteerak. X eta Y biderkatu behar diren bi zenbakiak dira, n bitekoak. Emaitza Z da, 2n bitekoa. Bider seinaleak biderketari ekiteko agintzen du, eta Egina adierazleak biderketa bukatu dela adierazi behar du.



6.5. Ariketa honetan, konputazio-sistema baten datu-pila diseinatu nahi dugu. RAM memoria bat erabili behar da pila egiteko, eta bi eragiketa onartu behar ditu: *Push* (datu bat pilan sartu) eta *Pop* (datu bat pilatik atera). Gainezkatzeak (goitik —*overflow*— zein behetik —*underflow*—) detektatu eta adierazi behar dira.



6.6. n biteko bi zenbakien Zatitzaile Komunetan Handiena (ZKH) kalkulatzen duen sistema digitala sortu behar da. Hasi eta Egina seinaleek eragiketaren hasiera eta bukaera adierazten dute.

A eta *B* zenbakien ZKHa kalkulatzeko, honako algoritmo hau erabil daiteke:

```
X := A; Y := B;
bitartean (X ≠ Y)
   baldin (X > Y)
      orduan   X := X – Y;
      bestela  Y := Y – X;
ambitartean
ZKH := X;
```