Laboratorinis darbas. Registrai

Turinys

---

1. Tikslas

Susipažinti su postūmio registrais, jų struktūra, veikimu, taikymo galimybėmis ir realizavimu naudojant trigerius. Išsiaiškinti postūmių operacijas ir jų atlikimo būdus.

2. Teorija

Registras – tai įtaisas, skirtas informacijai įrašyti ir saugoti, taip pat kitoms operacijoms su informacija atlikti. Registrus sudaro atminties ląstelės – trigeriai – ir juos valdančiosios schemos. Pagal informacijos įvedimo ir išvedimo būdą registras gali būti:

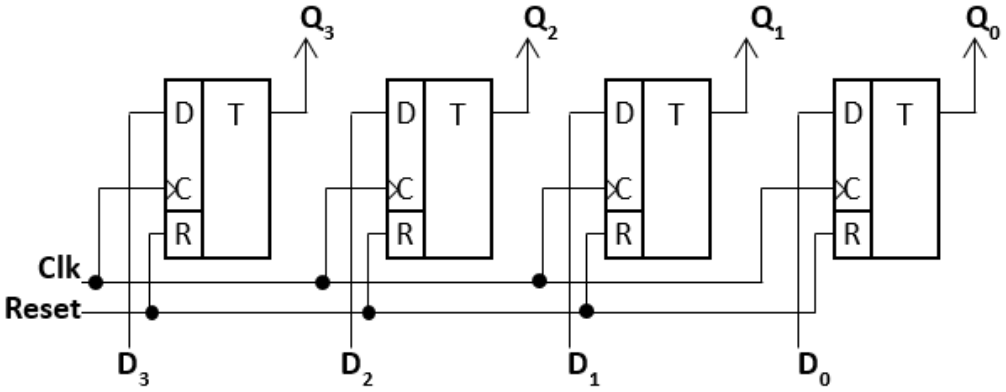
- saugojimo (lygiagretusis);
- postūmio (nuoseklusis);
- universalusis.

Registras gali būti vientaktis (dažniausiai toks registras sudaromas iš dinaminio valdymo trigerių) ar dvitaktis, kai informacija į jį įrašoma per du taktus (kaip ir MS

trigeriuose). Kadangi žemiausiąją informacijos žodžio ar baito skiltį (nulinį indeksą) įprasta rodyti dešinėje, registrų skiltys numeruojamos nuo dešinės pusės, pradedant 0 (literatūroje galima sutikti ir priešingą numeraciją). Tokiu atveju registre įrašytą informaciją interpretuojant kaip skaičių, jo reikšmė nustatoma sumuojant atitinkamas reikšmes, gaunamas dvejetą pakėlus indekso laipsniu.

2.1 Lygiagretieji registrai

Lygiagretųjį registrą galima sudaryti iš ***n*** sinchroninių D trigerių su įvesties ir išvesties valdymo grandinėmis, kaip parodyta 1 paveiksle.

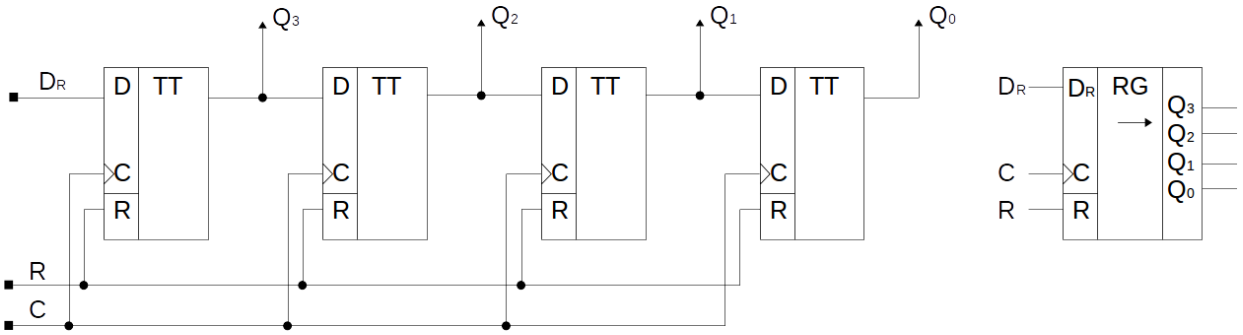


1 pav. Keturių skilčių lygiagrečiojo registro su valdymo grandinėmis schema

Šioje schemoje panaudoti D trigeriai, kuriuose įvesties signalas ***Reset*** ištrina įrašytą informaciją ir visus trigerius nustato į nulinę būseną. Signalas ***Clk*** nustato momentą, kada informacija įrašoma: kol ***Clk*** = 0, informacija išlieka nepasikeitusi (informacijos saugojimas), jei ***Clk*** = 1, įrašoma informacija, paduodama į įvestis D_i.

2.2 Postūmio (nuoseklusis) registras

Postūmio registras gaunamas sujungus sinchroninius dinaminio valdymo D trigerius nuosekliai (grandinėle). Tokio registro schema pateikta 2 paveiksle. Šis registras vykdo loginį postūmį į dešinę. Informacija į jį įrašoma postūmio metu, paduodant ją nuosekliu kodu į įvestį D_R. Įvesties signalas ***R*** visus trigerius nustato į nulinę būseną. Signalas ***C*** nustato postūmio momentą.



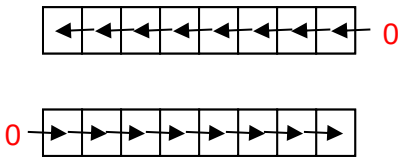
2 pav. Keturių skilčių postūmio į dešinę registro schema (a) ir grafinis žymėjimas (b)

Universalus postūmio registras vykdo ne tik postūmį į dešinę (LR, tiesioginis), bet ir į kairę (LL, reversinis). Vykdant postūmį, kiekvienu sinchroimpulso taktu įvedama informacija nuosekliai stumiama išilgai registro, kol pasiekia registro išvestis.

2.3 Loginiai ir cikliniai postūmiai

Kompiuteryje postūmio operacijos reikalingos atliekant daugybą ir dalybą, veiksmus su slankaus kablelio skaičiais. Dvejetainiai kodai gali būti pastumti dviem būdais:

1) Paprastai vykdant ***loginį postūmį*** į kairę ar į dešinę kiekviena skiltis perstumiama į gretimos vietą:



Iki postūmio: **0 1 1 0 0 0 1 1**

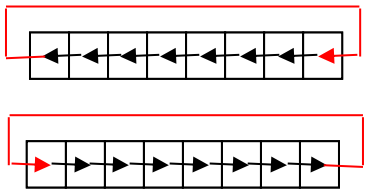
Pastūmus į kairę: **1 1 0 0 0 1 1 0**

Iki postūmio: **0 1 1 0 0 0 1 1**

Pastūmus į dešinę: **0 0 1 1 0 0 0 1**

Kaip matome, į atlaisvinamą skiltį (paprastai) įrašomas **0** (čia jis išskirtas raudona spalva)**.**

2) ***Cikliškai*** stumiant į kairę ar į dešinę išstumiama skiltis (čia ji išskirta raudona spalva) įrašoma į kitame gale atlaisvinamą skiltį:



Iki postūmio: **1 1 1 0 0 0 1 0**

Pastūmus į kairę: **1 1 0 0 0 1 0 1**

Iki postūmio: **0 1 1 0 0 0 1 1**

Pastūmus į dešinę: **1 0 1 1 0 0 0 1**

2.4 Aritmetiniai postūmiai

Aritmetinis postūmis skiriasi nuo paprastojo postūmio tik tuo, kad stumiami dvejetainiai kodai traktuojami kaip skaičiai, turintys ženklą (tai kairioji skiltis); be to, ženklo skiltis nėra stumiama.

Priminsime galimus skaičių su ženklu kodavimo būdus.

Dažniausiai taikomi trys skaičių kodavimo būdai:

1) tiesioginis kodas (angl. *Sign-and-magnitude)*;
2) atvirkštinis kodas (angl. *Ones' complement* arba 1*'s complement)*;
3) papildomasis kodas (angl. *Two's complement* arba 2*'s complement)*.

Ženklui saugoti skiriamas vienas bitas. Dažniausiai tai yra aukščiausiasis bitas. Jei šis bitas lygus 0, skaičius laikomas teigiamu, jei lygus 1 – neigiamu. Likę žodžio bitai nurodo skaičiaus absoliučiąją vertę (dydį). Taigi, jei registras yra 8 bitų ilgio, vienas jų skiriamas ženklui, o vertei saugoti lieka 7.

Tiesioginis kodas. Naudojant šį kodą, aukščiausias bitas rodo skaičiaus ženklą, o likę – skaičiaus modulio vertę. Skaičiaus modulio vertė gali kisti nuo 0000000 (0) iki 1111111 (127). Vadinasi, tokiu žodžiu galima atvaizduoti skaičius nuo –127 iki +127. Šioje sistemoje galimi du skaičiaus 0 kodavimo variantai: 00000000 (+0) ir 10000000 (–0). Kai kurie ankstyvieji kompiuteriai (pvz., IBM 7090) naudojo tiesioginį kodą dėl jo intuityvumo.

Pavyzdžiui, tiesioginiame kode skaičiai +26 ir -26 koduojami taip:

+26 : 00011010,

–26 : 10011010.

Atvirkštinis kodas. Naudojant šį kodą, neigiamieji skaičiai atvaizduojami kaip teigiamų skaičių bitų inversija. Kaip ir tiesioginis kodas, atvirkštinis turi du skaičiaus 0 kodavimo variantus: 00000000 (+0) ir 11111111 (–0). Pavyzdžiui, atvirkštiniame kode skaičiai +26 ir -26 koduojami taip:

+26 : 00011010,

–26 : 11100101.

N bitų ilgio registre galima išsaugoti skaičius nuo $-(2^{N-1}-1)$ iki $+(2^{N-1}-1)$ ir ±0.

Papildomasis kodas. Norint išspręsti skaičiaus 0 atvaizdavimo ir supaprastinti sudėties operacijos problemas, buvo pasiūlyta sistema, vadinama papildomuoju kodu. Papildomajame kode neigiamieji skaičiai atvaizduojami bitų seka, didesne vienetu nei

atvirkštinis skaičiaus kodas. Papildomajame kode yra tik vienas būdas atvaizduoti nulį (00000000). Priešingas skaičius (nesvarbu, teigiamas ar neigiamas) randamas invertuojant visus bitus ir prie rezultato pridedant vienetą. Šiame kode neigiamų ir teigiamų skaičių sudėtis atliekama pagal tą pačią taisyklę. Atimtis atliekama pridedant atėminį su priešingu ženklu.

Neigiamas skaičius verčiamas priešingu (teigiamu) taip:

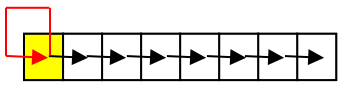
1) invertuojami visi duotojo skaičiaus bitai;
2) pridedamas vienetas.

Pavyzdžiui, dvejetainėje sistemoje +26 = 00011010. Tuomet $\overline{00011010}$ + 1= 11100101 + 1 = 11100110 (–26 papildomajame kode)

Dažniausiai ir beveik visuose šiuo metu naudojamuose bendrosios paskirties procesoriuose vartojamas papildomasis kodas.

Kadangi kompiuteriuose naudojamas papildomasis kodas, aritmetinio postūmio papildomajame kode taisykles galima formuluoti taip:

1. Atliekant aritmetinį postūmį į dešinę, ženklo skiltis nesikeičia ir įrašoma atlaisvinamą gretimą skiltį:



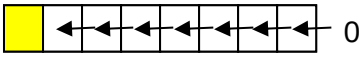
Iki postūmio: **0 1 1 0 0 0 1 1**

Pastūmus į dešinę: **0 0 1 1 0 0 0 1**

Iki postūmio: **1 0 1 0 0 0 1 1**

Pastūmus į dešinę: **1 1 0 1 0 0 0 1**

2. Atliekant aritmetinį postūmį į kairę, ženklo skiltis nesikeičia, o į atlaisvinamą žemiausiąją skiltį įrašomas 0:



Iki postūmio: **0 0 1 0 0 0 1 1**

Pastūmus į kairę: **0 1 0 0 0 1 1 0**

Iki postūmio: **1 1 1 0 0 0 1 1**

Pastūmus į kairę: **1 1 0 0 0 1 1 0**

Aritmetinis postūmis atitinka skaičiaus modulio dvigubinimą (stumiant į kairę) arba jo mažinimą 2 kartus (stumiant į dešinę).

Atliekant informacijos aritmetinius postūmius registruose, į laisvas skiltis įrašoma informacija, nurodyta 1 lentelėje.

1 lentelė. Laisvų skilčių užpildymas atliekant aritmetinius postūmius

Ženklas (skaičius)	***Kodas***	***Į laisvą skiltį įrašoma***	
		AR stumiant į dešinę	***AL stumiant į kairę***
+ (teigiamas)	bet koks	0	0
– (neigiamas)	tiesioginis	0	0
	atvirkštinis	1	1
	papildomasis	1	0

Aritmetinio postūmio taisyklės gali būi suformuluotos taip:

- jei skaičiai pateikiami ***papildomuoju*** kodu, **stumiant į dešinę į atlaisvinamą skiltį įrašoma ženklo skilties reikšmė, o stumiant į kairę – 0**;
- jei skaičiai pateikiami ***tiesioginiu*** kodu, **stumiant abiem atvejais į atlaisvinamą skiltį įrašomas 0**;
- jei skaičiai pateikiami ***atvirkštiniu*** kodu, **stumiant abiem atvejais į *atlaisvinamą skiltį įrašoma ženklo skilties reikšmė.***

2.5 Universalusis registras

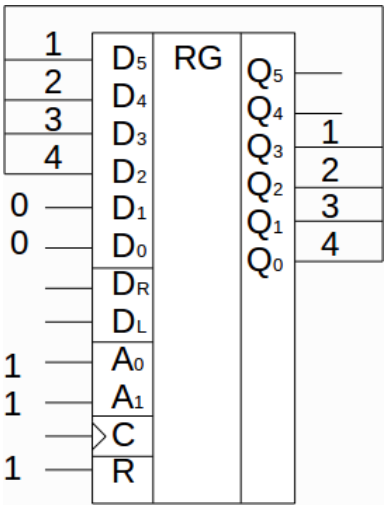
Postūmio registras paprastai vykdo loginį postūmį per vieną skiltį į dešinę (LR_1) ar į kairę (LL_1) ir lygiagretų informacijos įrašymą. Informacijai įrašyti į atsilaisvinusią skiltį naudojama atskira įvestis D_R (vykdant LR_1) ir D_L (vykdant LL_1). Toks registras vadinamas universaliuoju. Šie registrai gali realizuoti ir ciklinį postūmį į dešinę ar į kairę. Tada pakanka išstumiamosios skilties išvestį sujungti su atitinkama D_R ar D_L įvestimi. Kitas postūmio operacijas (pavyzdžiui, postūmį per dvi skiltis) galima realizuoti naudojant lygiagretų įrašymą, tada registro išvestys sujungiamos su atitinkamomis įvestimis.

Pavyzdžiui, turime 6 skilčių universalųjį postūmio registrą, vykdantį lygiagretų informacijos įrašymą bei loginius postūmius per 1 skiltį į dešinę ir į kairę. Šio registro darbą aprašanti lentelė pateikta žemiau (2 lentelė).

2 lentelė. Universaliojo registro veikimo lentelė

R	A_0A_1	$D_5...D_0$	Q_5 Q_4 Q_3 Q_2 Q_1 Q_0	Paaiškinimai
0	*x* *x*	*x...x*	0 0 0 0 0 0	Nulio nustatymas
1	0 0	*x...x*	Q_5 Q_4 Q_3 Q_2 Q_1 Q_0	Saugojimas
1	1 0	*x...x*	D_R Q_5 Q_4 Q_3 Q_2 Q_1	Loginis postūmis į dešinę, įrašant D_R (LR_1, D_R)
1	0 1	*x...x*	Q_4 Q_3 Q_2 Q_1 Q_0 D_L	Loginis postūmis į kairę, įrašant D_L (LL_1, D_L)
1	1 1	$B_5...B_0$	B_5 B_4 B_3 B_2 B_1 B_0	Lygiagretus informacijos įrašymas

Universaliuoju registru galima realizuoti bet kokį informacijos postūmį, pavyzdžiui, loginį postūmį per dvi skiltis įrašant nulius (LL_2, 0). Tokį postūmį realizuos schema, pavaizduota 3 paveiksle.



3 pav. Loginis postūmis universaliuoju registru į kairę per dvi skiltis įrašant 0 (LL_2, 0)

2.6 Specializuotas postūmio registras

Specializuotas postūmio registras vykdo lygiagretų informacijos įrašymą ir reikiamą postūmių (tai gali būti loginiai, cikliniai ar aritmetiniai per vieną ar kelias skiltis į dešinę ar į kairę).

Pavyzdžiui, turime 6 skilčių specializuotą postūmio registrą, vykdantį lygiagretų informacijos įrašymą bei tokį postūmių rinkinį: loginis postūmis per 2 skiltis į kairę (LL_2) į atsilaisvinusią skiltį įrašant 0, ciklinis postūmis per 1 skiltį į dešinę (CR_1) ir aritmetinis

postūmis per 1 skiltį į dešinę (AR_1) papildomajame kode. Šio registro darbą aprašanti lentelė pateikta žemiau (3 lentelė).

3 lentelė. Specializuoto registro veikimo lentelė

A_1A_0	Q_5 Q_4 Q_3 Q_2 Q_1 Q_0	Paaiškinimai
x *x*	0 0 0 0 0 0	Nulio nustatymas
0 0	B_5 B_4 B_3 B_2 B_1 B_0	Lygiagretus informacijos įrašymas
0 1	Q_3 Q_2 Q_1 Q_0 0 0	Loginis postūmis per 2 skiltis į kairę, įrašant *0* (LL_2)
1 0	Q_0 Q_5 Q_4 Q_3 Q_2 Q_1	Ciklinis postūmis per 1 skiltį į dešinę (CR_1)
1 1	Q_5 Q_5 Q_4 Q_3 Q_2 Q_1	Aritmetinis postūmis per 1 skiltį į dešinę (AR_1)

3. `Laboratorinio darbo užduotis

1. Naudojant schemų redaktorių, sudaryti užduotyje nurodyto ilgio universalaus postūmio registro schemą (panašią į pateiktą 2 lentelėje). Sudaryti testinius rinkinius ir patikrinti, kaip veikia schema.

2. Naudojant multiplekserius ir lygiagretųjį registrą, suprojektuoti specializuotą postūmio registrą, realizuojantį užduotyje nurodytas postūmio mikrooperacijas. Sudaryti testinius rinkinius ir patikrinti, kaip veikia schema.

3. Parengti laboratorinio darbo ataskaitą. Joje pateikti suprojektuotas schemas ir laiko diagramas.

4. Pavyzdys

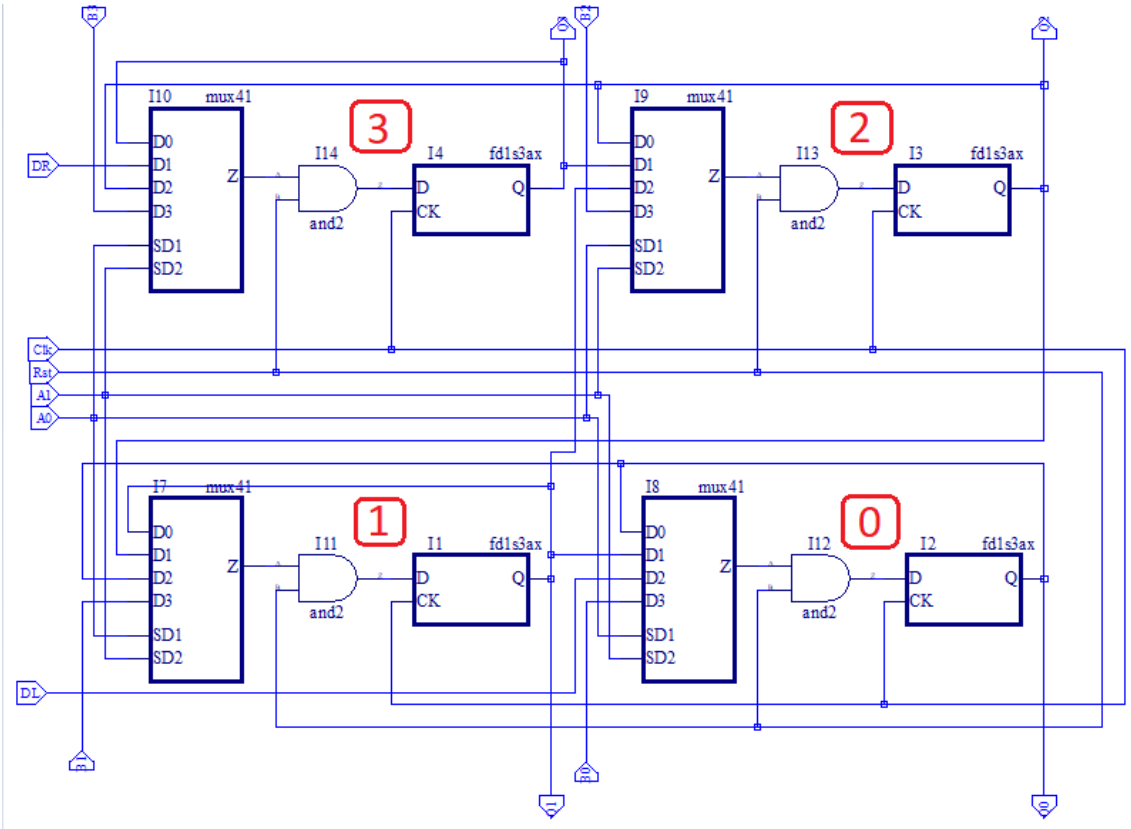
Suprojektuokime du 4 skilčių registrus – universalųjį ir specializuotą, vykdantį tokias postūmio mikrooperacijas: LL_2 (į atsilaivinančias skiltis įrašoma informacija – 1), CR_1, AL_1; nulio nustatymas – sinchroninis; skaičių kodas – papildomasis. Pirmojoje schemoje naudosime bibliotekoje esančius D trigerius **fd1s3ax** (*D Flip-Flop GSR Clear*), o nustatymą į nulinę būseną atliksime naudodami įrašymo mikrooperaciją, o antrojoje schemoje – D trigerius (*D Flip-Flop Asynchronuos clear*) su asinchronine *RESET* įvestimi – **fd1s3dx**.

1. Sudarome universaliojo registro veikimo lentelę:

R	A_0A_1	$Q_3\ Q_2\ Q_1\ Q_0$	*Mikrooperacija*
0	$x\ \ x$	0 0 0 0	Nulio nustatymas
1	0 0	$Q_3\ Q_2\ Q_1\ Q_0$	Saugojimas
1	1 0	$D_R\ Q_3\ Q_2\ Q_1$	Loginis postūmis į dešinę, įrašant D_R (LR_1, D_R)
1	0 1	$Q_2\ Q_1\ Q_0\ D_L$	Loginis postūmis į kairę, įrašant D_L (LL_1, D_L)
1	1 1	$B_3\ B_2\ B_1\ B_0$	Lygiagretus informacijos įrašymas

Registras vykdo keturias mikrooperacijas, todėl joms valdyti naudosime multiplekserius. Signalus A_1, A_0 jungsime į multiplekserių adresines įvestis SD2 ir SD1, o duomenų įvestis sujungsime su signalais pagal sudarytąją registro veikimo lentelę.

Sinchroniniam nulio nustatymui prieš trigerių *D* įvestis įterpkime IR elementus, valdomus signalo *Res*. Universaliojo registro schema parodyta 5 paveiksle.



5 pav. Universaliojo registro schema

Sudarome registro veikimą tikrinantį testą, kuris patikrintų visas registro vykdomas mikrooperacijas. Testo scenarijus bus toks:

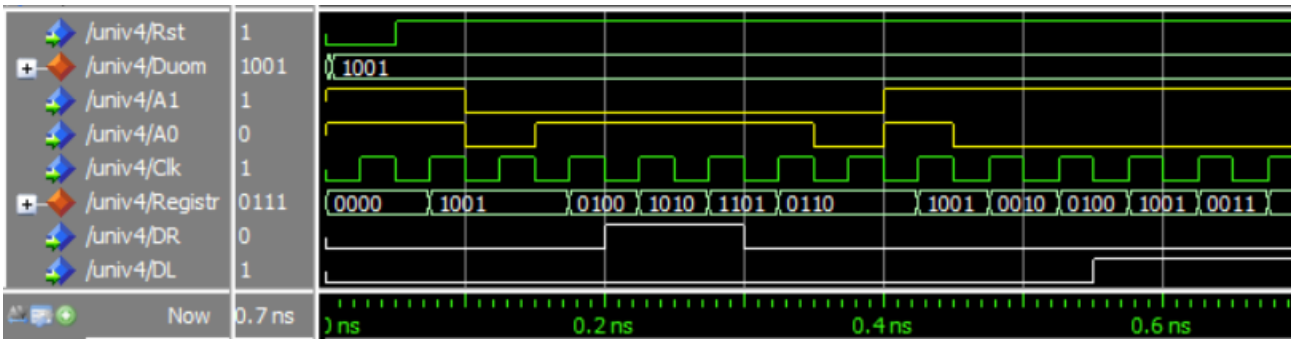
1. 1 periodas – registrą nustatysime į pradinę būseną – 0000.

2. 2 periodas – į registrą įrašysime duomenis – 1001.
3. 3 periodas – saugosime registro turinį.
4. 4-7 periodai – 4 kartus vykdysime registro turinio postūmį į dešinę, postūmių metu keisdami DR bito reikšmę.
5. 8 periodas – saugosime registro turinį.
6. 9 periodas – į registrą įrašysime pradinius duomenis – 1001.
7. 10-13 periodai – 4 kartus vykdysime registro turinio postūmį į kairę, postūmių metu keisdami DL bito reikšmę.

Šį scenarijų atitiks tokios modeliavimo direktyvos:

```
restart –f
force -freeze sim:/Univ/Rst 0 0, 1 {50 ps}
force -freeze sim:/Univ/Clk 0 0, 1 {25 ps}  -r 50
force -freeze sim:/Univ/A1 1 0, 0 {100 ps}
force -freeze sim:/Univ/A1 1 400
force -freeze sim:/Univ/A0 1 0, 0 {100 ps}
force -freeze sim:/Univ/A0 1 150, 0 {350 ps}
force -freeze sim:/Univ/A0 1 400, 0 {450 ps}
force -freeze sim:/Univ/DR 0 0, 1 {200 ps}
force -freeze sim:/Univ/DR 0 300
force -freeze sim:/Univ/DL 0 0, 1 {550 ps}
force -freeze sim:/Univ/B3 1 0
force -freeze sim:/Univ/B2 0 0
force -freeze sim:/Univ/B1 0 0
force -freeze sim:/Univ/B0 1 0
run 700
```

Modeliuodami gauname laiko diagramas, pavaizduotas 6 paveiksle. Vaizdumo ir analizės supaprastinimo dėlei įvestys B3-B0 ir registro išvestys Q3-Q0 apjungtos į universaliąsias magistrales (atitinkamai – Duom ir Registr), o jų turinys rodomas dvejetainėje sistemoje.



6 pav. Universaliojo registro laiko diagrama

Kai įvedami duomenys apjungti (*Combine*) į **Duom**, tuomet direktyvas

```
force -freeze sim:/Univ/B3 1 0
force -freeze sim:/Univ/B2 0 0
force -freeze sim:/Univ/B1 0 0
force -freeze sim:/Univ/B0 1 0
```

galima pakeisti viena:

```
force -freeze sim:/Univ/Duom  1001  0
```

Jei operaciją išrenkantys signalai **A1** ir **A0** būtų apjungti (*Combine*), pavyzdžiui, į **Oper**, tuomet direktyvas

```
force -freeze sim:/Univ/A1 1 0, 0 {100 ps}
force -freeze sim:/Univ/A1 1 400
force -freeze sim:/Univ/A0 1 0, 0 {100 ps}
force -freeze sim:/Univ/A0 1 150, 0 {350 ps}
force -freeze sim:/Univ/A0 1 400, 0 {450 ps}
```

galima būtų pakeisti viena:

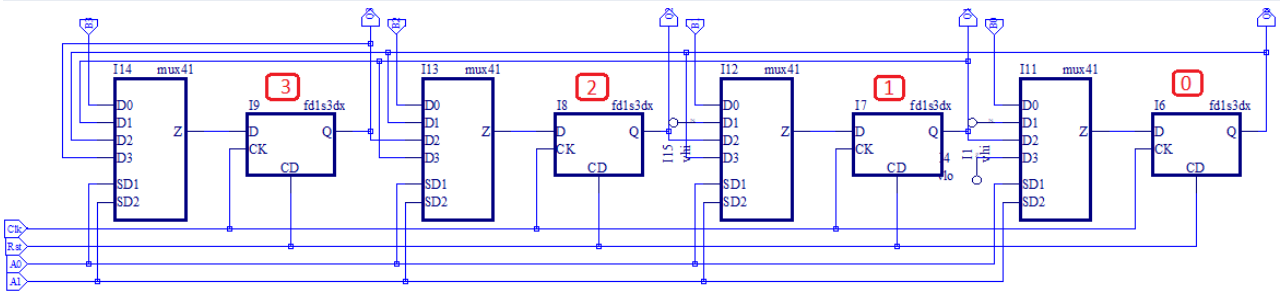
```
force -freeze sim:/Univ/Oper  11  0, 00 100, 01 150, 00 350, 11 400, 10 450
```

Toks apjungimas pageidautinas, nes esant reikalui būtų paprasčiau keisti operacijų seką ar jų vykdymo trukmę.

2. Projektuodami specializuotą registrą, vykdantį postūmio mikrooperacijos LL2 įrašant 1, CR_1 ir AL_1 naudosime multiplekserį su keturiomis įvestimis ir trigerius. Schemai sudaryti reikalingą informaciją surašome į lentelę:

A_0A_1	$Q_3\ Q_2\ Q_1\ Q_0$	*Mikrooperacija*
0 0	$B_3\ B_2\ B_1\ B_0$	Lygiagretus informacijos įrašymas
1 0	$Q_1\ Q_0\ 1\ 1$	Loginis postūmis į kairę – LL_2
0 1	$Q_0\ Q_3\ Q_2\ Q_1$	Ciklinis postūmis į dešinę – CR_1
1 1	$Q_3\ Q_1 Q_0\ 0$	Aritmetinis postūmis į kairę – AL_1

Šią sistemą patogu realizuoti multiplekseriais, kaip pavaizduota 7 paveiksle.

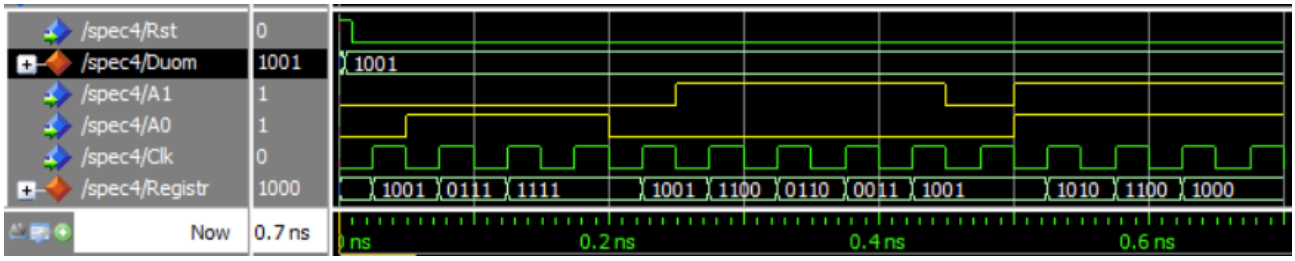


7 pav. Specializuoto registro su asinchroniniu nulio nustatymu schema

Sudarome registro veikimą tikrinantį testą, kuris patikrintų visas registro vykdomas mikrooperacijas. Paruošiame modeliavimo direktyvas:

```
restart –f
force -freeze sim:/Spec/Rst 1 0, 0 {10 ps}
force -freeze sim:/Spec/Clk 0 0, 1 {25 ps}  -r 50
force -freeze sim:/Spec/A1 0 0, 1 {250 ps}
force -freeze sim:/Spec/A1 0 450, 1 {500 ps}
force -freeze sim:/Spec/A0 0 0, 1 {50 ps}
force -freeze sim:/Spec/A0 0 200, 1 {500 ps}
force -freeze sim:/Spec/B3 0 0, 1 {5 ps}
force -freeze sim:/Spec/B2 0 0
force -freeze sim:/Spec/B1 0 0
force -freeze sim:/Spec/B0 0 0, 1 {5 ps}
run 700
```

Modeliuodami gauname 8 paveiksle parodytas laiko diagramas.



8 pav. Specializuoto registro laiko diagrama

Nagrinėdami laiko diagramas matome, kad specializuotas registras vykdo:

- 0-50 ps – informacijos įrašymą (įrašyti duomenys – 1001);
- 50-200 ps – LL_2 mikrooperaciją;
- 200-250 ps – informacijos įrašymą iš naujo;
- 250-450 ps – CR_1 mikrooperaciją;
- 450-500 ps – informacijos įrašymą iš naujo;
- 500-700 ps – AL_1 mikrooperaciją.

Aritmetinio postūmio rezultatai priklauso nuo skaičiaus ženklo, todėl pabaigoje reikėtų pakeisti B3 reikšmę priešinga ir vėl modeliuoti aritmetinį postūmį (3-4 taktus).