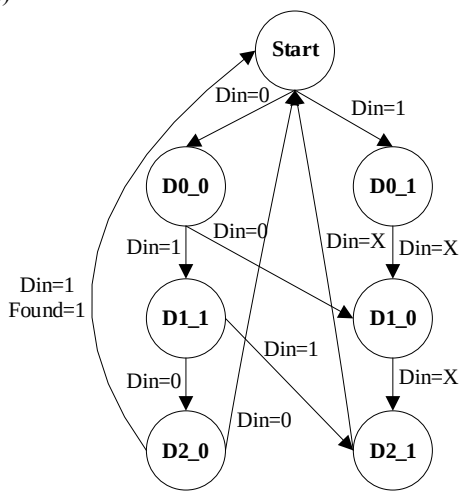
3. Секвенцијалне мреже

Задатак 7.4.13
Пројектовати 4-битни серијски бит детектор секвенце сличан оном описаном у примеру 7.9. Улаз у детектор секвенце је означен са Din, а излаз са Found. Детектор ће поставити Found на 1, савки пут када се на улазу појави 4-битна секвенца “0101”. За све друге улазне секвенце излаз има вредност 0.
а) Наћи дијаграм стања коначног аутомата.
б) Бинарно кодирати стања. Колико D-флип-флопова је потребно за имплементацију меморије стања за овај коначни аутомат.
ц) Наћи таблицу прелаза стања за овај коначни аутомат.
д) Извршити синтезу логичких израза за следеће стање.
е) Извршити синтезу логичких израза за излаз.
ф) Да ли је овај аутомат Милијев или Муров?
г) Нацртати логичку мрежу за овај коначни аутомат.

Решење задатка 7.4.13
а)



б)

стање	код
Start	000
D0_0	001
D1_1	010
D2_0	011
D0_1	100
D1_0	101
D2_1	110

3 D-флип-флопа

ц)

тренутно стање				улаз	следеће стање				излаз
	Q2_cur	Q1_cur	Q0_cur	Din		Q2_nxt	Q1_nxt	Q0_nxt	Found
Start	0	0	0	0	D0_0	0	0	1	0
Start	0	0	0	1	D0_1	1	0	0	0
D0_0	0	0	1	0	D1_0	1	0	1	0
D0_0	0	0	1	1	D1_1	0	1	0	0
D1_1	0	1	0	0	D2_0	0	1	1	0
D1_1	0	1	0	1	D2_1	1	1	0	0
D2_0	0	1	1	0	Start	0	0	0	0
D2_0	0	1	1	1	Start	0	0	0	1
D0_1	1	0	0	0	D1_0	1	0	1	0
D0_1	1	0	0	1	D1_0	1	0	1	0
D1_0	1	0	1	0	D2_1	1	1	0	0
D1_0	1	0	1	1	D2_1	1	1	0	0
D2_1	1	1	0	0	Start	0	0	0	0
D2_1	1	1	0	1	Start	0	0	0	0

д)

Q2_nxt

Q2_cur Q1_cur \ Q0_cur Din	00	01	11	10
00	0	0	0	1
01	1	1	0	1
11	0	0	X	1
10	1	0	X	1

Q1_nxt

Q2_cur Q1_cur \ Q0_cur Din	00	01	11	10
00	0	1	0	0
01	0	1	0	0
11	1	0	X	1
10	0	0	X	1

Q0_nxt

Q2_cur Q1_cur \ Q0_cur Din	00	01	11	10
00	1	1	0	1
01	0	0	0	1
11	0	0	X	0
10	1	0	X	0

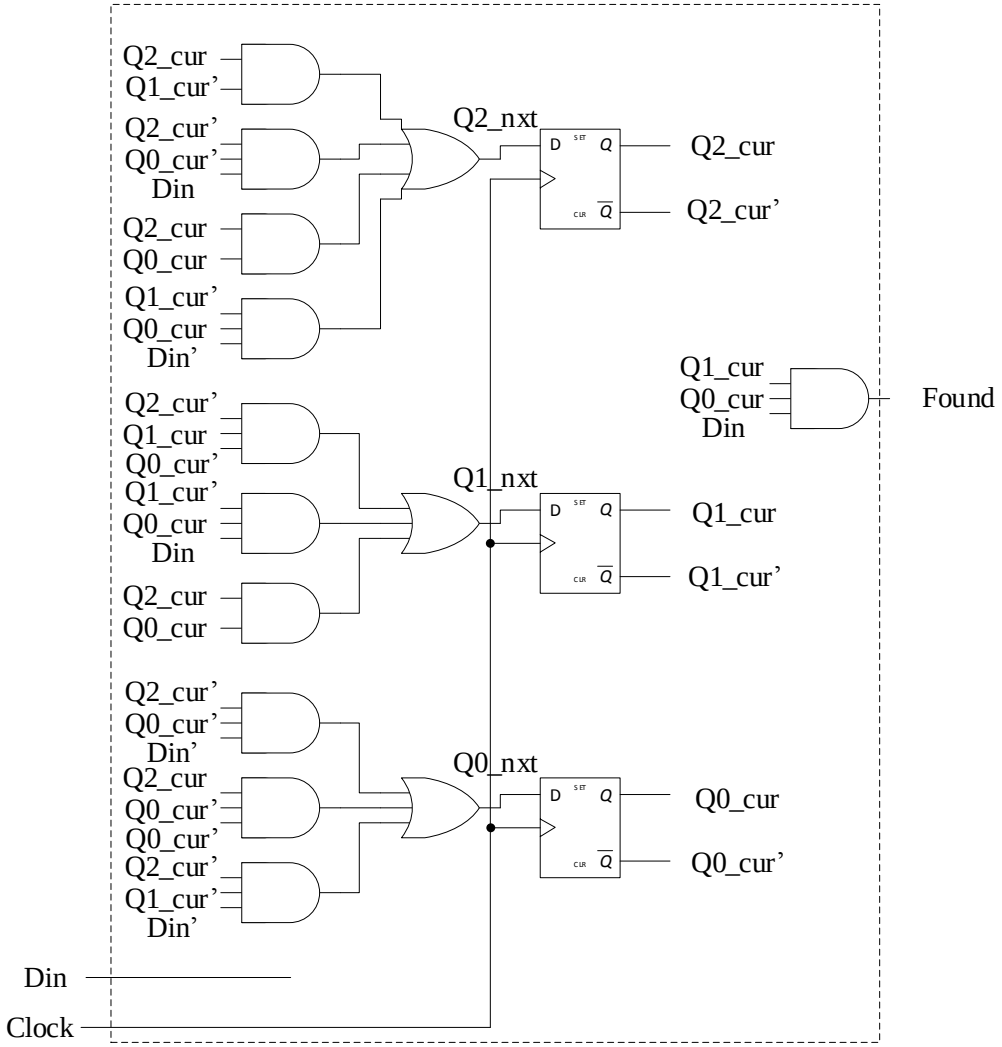
Found

Q2_cur Q1_cur \ Q0_cur Din	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	0	1	X	0
10	0	0	X	0

Q2_nxt = (Q2_cur * Q1_cur') + (Q2_cur' * Q0_cur' * Din) + (Q2_cur * Q0_cur) + (Q1_cur' * Q0_cur * Din')
Q1_nxt = (Q2_cur' * Q1_cur * Q0_cur') + (Q1_cur' * Q0_cur * Din) + (Q2_cur * Q0_cur)
Q0_nxt = (Q2_cur' * Q0_cur' * Din') + (Q2_cur * Q1_cur‘ * Q0_cur') + (Q2_cur' * Q1_cur' * Din')
Found = Q1_cur * Q0_cur * Din

ф) Милијев

г)



Задатак 7.4.14

Пројектовати контролер за 20-центни аутомат за слаткише сличан оном описаном у примеру 7.12. Улаз у контролер су кованице од 5 и 10 центи, а слаткиш се издаје сваки пут када корисник унесе 20 центи. Аутомат има два улаза Nin и Din. Nin се поставља на 1 када корисник убаци кованицу од 5 центи, док се улаз Din поставља на 1 када је убачено 10 центи. Аутомат има 2 излаза: Dispense и Change. Dispense се поставља на 1 сваки пут када корисник убаци укупно најмање 20 центи, док се Change поставља на 1 ако је убачено више од 20 центи и потребно је да се врати кусур од 5 центи.

а) Наћи дијаграм стања коначног аутомата.

б) Бинарно кодирати стања. Колико D-флип-флопова је потребно за имплементацију меморије стања за овај коначни аутомат.

ц) Наћи таблицу прелаза стања за овај коначни аутомат.

д) Извршити синтезу логичких израза за следеће стање.

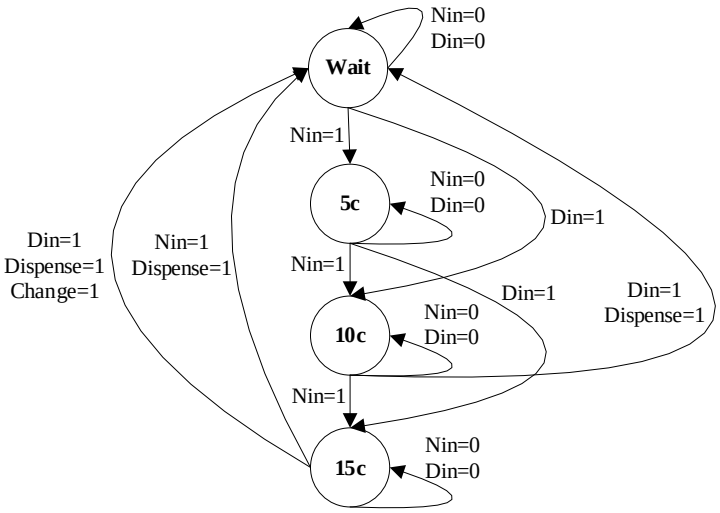
е) Извршити синтезу логичких израза за излаз.

ф) Да ли је овај аутомат Милијев или Муров?

г) Нацртати логичку мрежу за овај коначни аутомат.

Решење задатка 7.4.14

а)



б)

стање	код
Wait	00
5c	01
10c	10
15c	11

2 D-флип-флопа

ц)

тренутно стање			улаз		следеће стање			излаз	
	Q1_cur	Q0_cur	Nin	Din		Q1_nxt	Q0_nxt	Dispense	Change
Wait	0	0	0	0	Wait	0	0	0	0
Wait	0	0	0	1	10c	1	0	0	0
Wait	0	0	1	0	5c	0	1	0	0
Wait	0	0	1	1	Wait	0	0	0	0
5c	0	1	0	0	5c	0	1	0	0
5c	0	1	0	1	15c	1	1	0	0
5c	0	1	1	0	10c	1	0	0	0
5c	0	1	1	1	5c	0	1	0	0
10c	1	0	0	0	10c	1	0	0	0
10c	1	0	0	1	15c	1	1	0	0
10c	1	0	1	0	Wait	0	0	1	0
10c	1	0	1	1	10c	1	0	0	0
15c	1	1	0	0	15c	1	1	0	0
15c	1	1	0	1	Wait	0	0	1	0
15c	1	1	1	0	Wait	0	0	1	1
15c	1	1	1	1	15c	1	1	0	0

д)

Q1_nxt

Q1_cur Q0_cur / Nin Din	00	01	11	10
00	0	0	1	1
01	1	1	0	1
11	0	0	1	1
10	0	1	0	0

Q0_nxt

Q1_cur Q0_cur / Nin Din	00	01	11	10
00	0	1	1	0
01	0	1	0	1
11	0	1	1	0
10	1	0	0	0

Dispense

Q1_cur Q0_cur / Nin Din	00	01	11	10
00	0	0	0	0
01	0	0	1	0
11	0	0	0	0
10	0	0	1	1

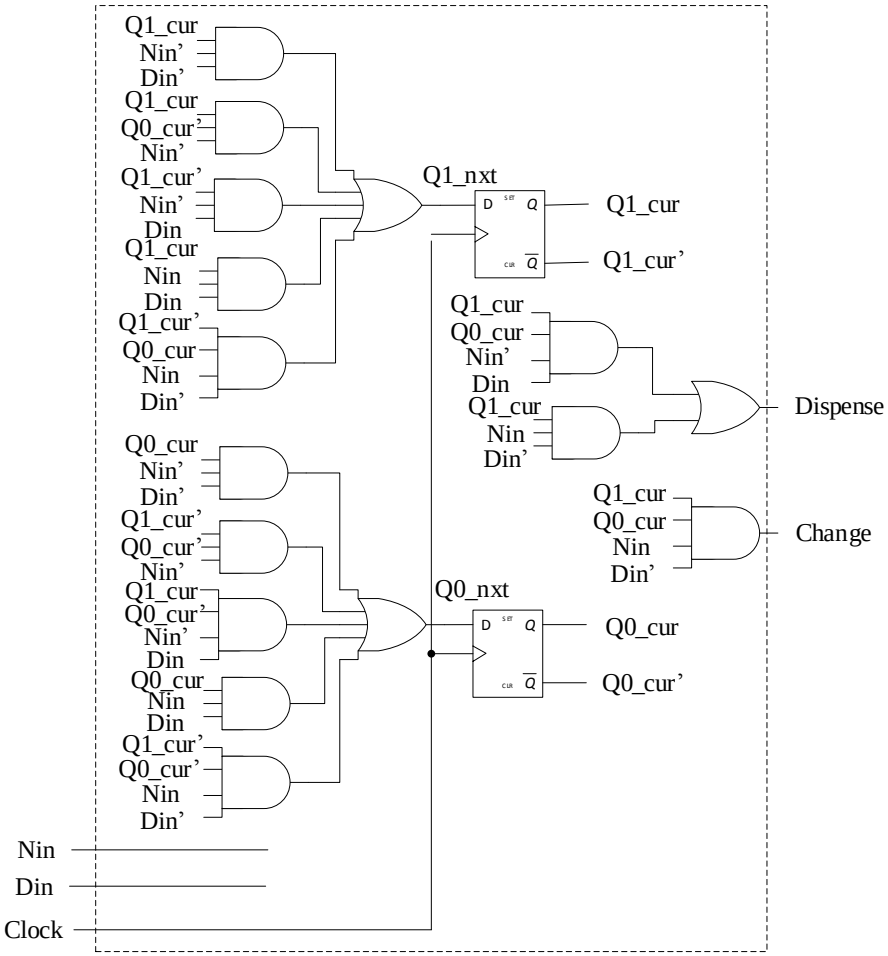
Change

Q1_cur Q0_cur / Nin Din	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	0	0	0	0
10	0	0	1	0

Q1_nxt = (Q1_cur * Nin’ * Din’) + (Q1_cur * Q0_cur’ * Nin’) + (Q1_cur’ * Nin’ * Din) + (Q1_cur * Nin * Din) + (Q1_cur’ * Q0_cur * Nin * Din’)
Q0_nxt = (Q0_cur * Nin’ * Din’) + (Q1_cur’ * Q0_cur‘ * Nin’) + (Q1_cur * Q0_cur’ * Nin’ * Din) + (Q0_cur * Nin * Din) + (Q1_cur’ * Q0_cur’ * Nin * Din’)
Dispose = (Q1_cur * Q0_cur * Nin’ * Din) + (Q1_cur * Nin * Din’)
Change = (Q1_cur * Q0_cur * Nin * Din’)

ф) Милијев

г)



Задатак 7.5.1

Пројектовати 3-битни бинарни кружни бројач навише сличан оном описаном у примеру 7.15. Овај коначни аутомат има 8 стања и захтева 3 бита за кодирање стања. Промељиве за тренутно стање назвати: Q2_cur, Q1_cur и Q0_cur, а за следеће стање Q2_nxt, Q1_nxt и Q0_nxt. Излаз бројача дефинисати као 3-битни вектор Count.

а) Наћи логички израз за следеће стање Q2_nxt?

б) Наћи логички израз за следеће стање Q1_nxt?

ц) Наћи логички израз за следеће стање Q0_nxt?

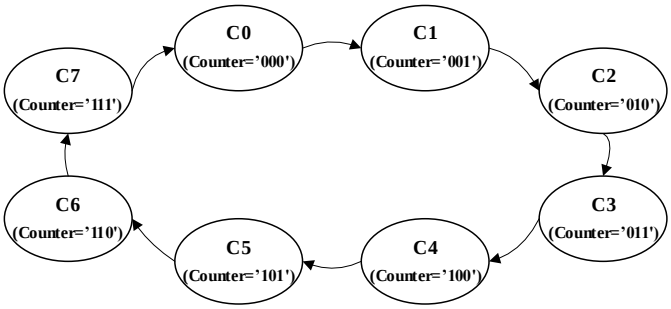
д) Наћи логички израз за излаз Count(2)?

е) Наћи логички израз за излаз Count(1)?

ф) Наћи логички израз за излаз Count(0)?

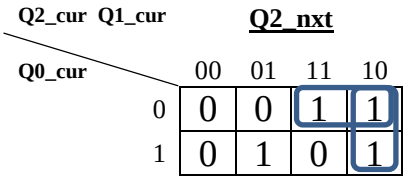
г) Нацртати логичку мрежу за овај бројач.

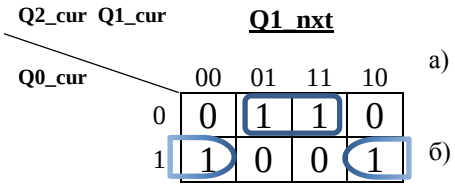
Решење задатка 7.5.1



стање	код	излаз
C0	000	000
C1	001	001
C2	010	010
C3	011	011
C4	100	100
C5	101	101
C6	110	110
C7	111	111

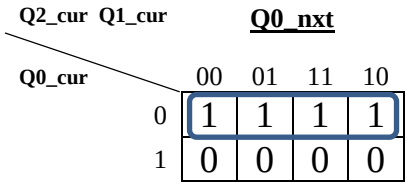
	тренутно стање				следеће стање			излаз		
	Q2_cur	Q1_cur	Q0_cur		Q2_nxt	Q1_nxt	Q0_nxt	Count(2)	Count(1)	Count(0)
C0	0	0	0	C1	0	0	1	0	0	1
C1	0	0	1	C2	0	1	0	0	1	0
C2	0	1	0	C3	0	1	1	0	1	1
C3	0	1	1	C4	1	0	0	1	0	0
C4	1	0	0	C5	1	0	1	1	0	1
C5	1	0	1	C6	1	1	0	1	1	0
C6	1	1	0	C7	1	1	1	1	1	1
C7	1	1	1	C0	0	0	0	0	0	0



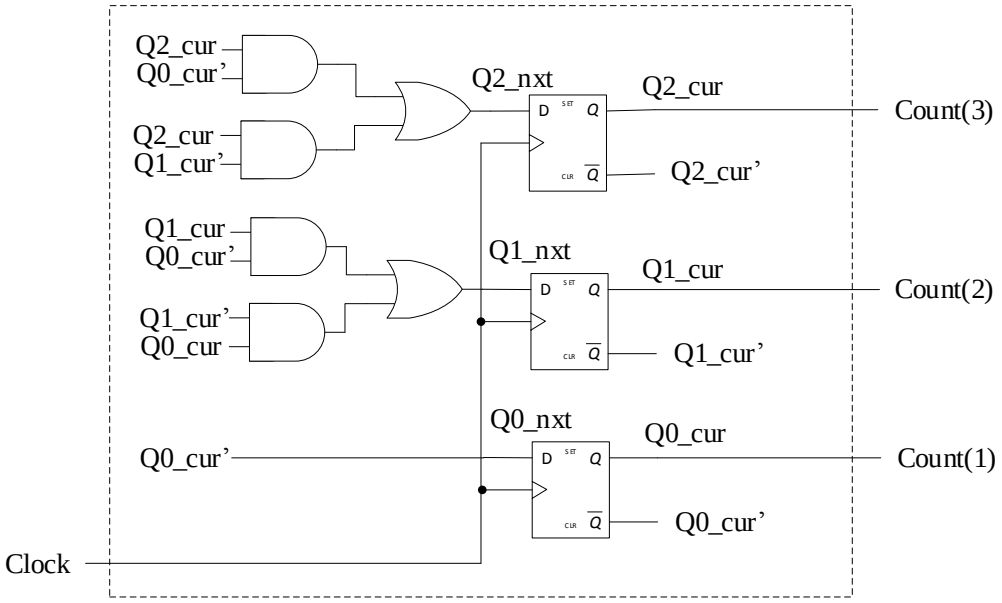


а)

б)

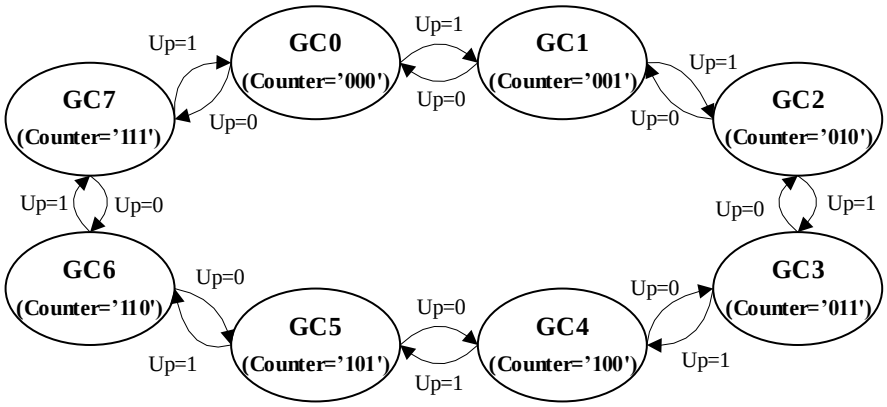


а) Q2_nxt = (Q2_cur * Q0_cur’) + (Q2_cur * Q1_cur’)
б) Q1_nxt = (Q1_cur * Q0_cur’) + (Q1_cur’ * Q0_cur)
ц) Q0_nxt = Q0_cur’
д) Count(2) = Q2_nxt
е) Count(1) = Q1_nxt
ф) Count(0) = Q0_nxt
г)



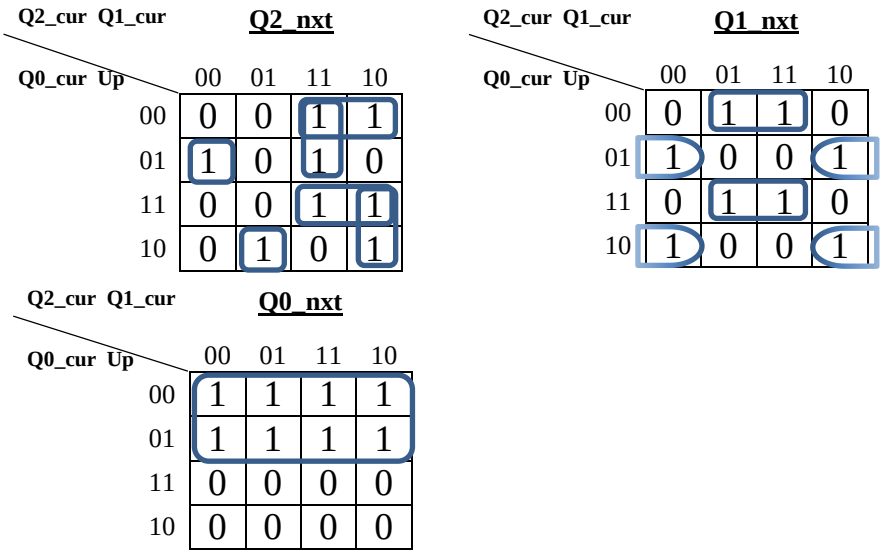
Задатак 7.5.7
Пројектовати 3-битни кружни бинарни бројач у грејовом коду навише/наниже сличан оном описаном у примеру 7.21. Бројач има улаз “Up” који одрешује смер бројања. Када је Up=1 бројач се инкрементира, а када је Up=0 бројач се декрементира. Овај аутомат има 8 стања и захтева 3 бита за кодирање стања. Промељиве за тренутно стање назвати: Q2_cur, Q1_cur и Q0_cur, а за следеће стање Q2_nxt, Q1_nxt и Q0_nxt. Излаз бројача дефинисати као 3-битни вектор Count.
а) Наћи логички израз за следеће стање Q2_nxt?
б) Наћи логички израз за следеће стање Q1_nxt?
ц) Наћи логички израз за следеће стање Q0_nxt?
д) Наћи логички израз за излаз Count(2)?
е) Наћи логички израз за излаз Count(1)?
ф) Наћи логички израз за излаз Count(0)?
г) Нацртати логичку мрежу за овај бројач.

Решење задатка 7.5.7



стање	код
GC0	000
GC1	001
GC2	010
GC3	011
GC4	100
GC5	101
GC6	110
GC7	111

тренутно стање				улаз	следеће стање				излаз		
	Q2_cur	Q1_cur	Q0_cur	Up		Q2_nxt	Q1_nxt	Q0_nxt	Count(2)	Count(1)	Count(0)
GC0	0	0	0	0	GC1	0	0	1	0	0	1
GC0	0	0	0	1	GC7	1	1	1	1	1	1
GC1	0	0	1	0	GC2	0	1	0	0	1	0
GC1	0	0	1	1	GC0	0	0	0	0	0	0
GC2	0	1	0	0	GC3	0	1	1	0	1	1
GC2	0	1	0	1	GC1	0	0	1	0	0	1
GC3	0	1	1	0	GC4	1	0	0	1	0	0
GC3	0	1	1	1	GC2	0	1	0	0	1	0
GC4	1	0	0	0	GC5	1	0	1	1	0	1
GC4	1	0	0	1	GC3	0	1	1	0	1	1
GC5	1	0	1	0	GC6	1	1	0	1	1	0
GC5	1	0	1	1	GC4	1	0	0	1	0	0
GC6	1	1	0	0	GC7	1	1	1	1	1	1
GC6	1	1	0	1	GC5	1	0	1	1	0	1
GC7	1	1	1	0	GC0	0	0	0	0	0	0
GC7	1	1	1	1	GC6	1	1	0	1	1	0



а) Q2_nxt = (Q2_cur * Q0_cur’ * Up’) + (Q2_cur * Q1_cur * Q0_cur’) + (Q2_cur’ * Q1_cur’ * Q0_cur’ * Up) +
(Q2_cur * Q0_cur * Up) + (Q2_cur * Q1_cur’ *Q0_cur) + (Q2_cur’ * Q1_cur * Q0_cur * Up’)

б) Q1_nxt = (Q1_cur * Q0_cur’ * Up’) + (Q1_cur’ * Q0_cur’ * Up) + (Q1_cur * Q0_cur * Up) + (Q1_cur’ * Q0_cur * Up’)

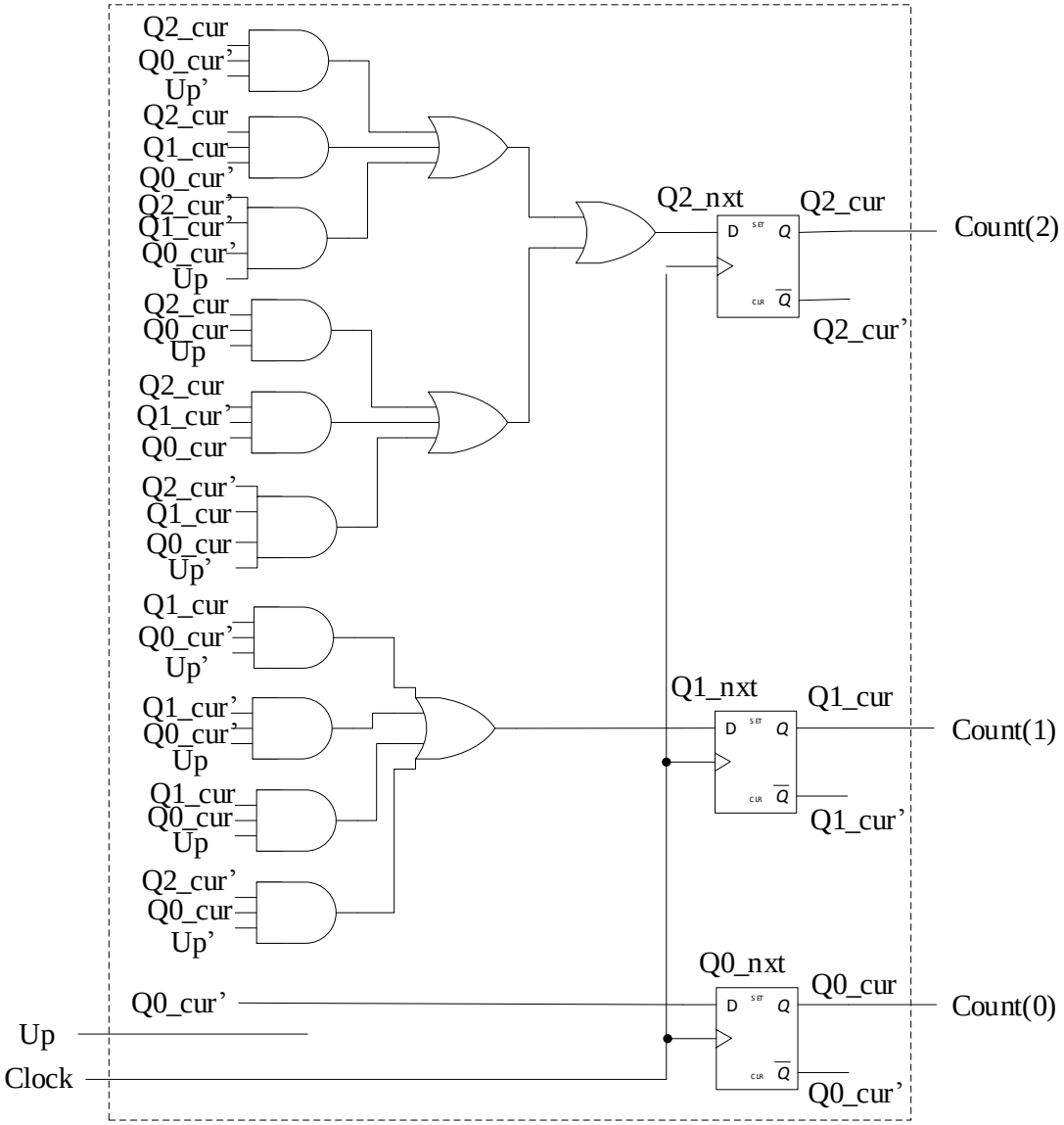
ц) Q0_nxt = Q0_cur’

д) Count(2) = Q2_nxt

е) Count(1) = Q1_nxt

ф) Count(0) = Q0_nxt

r)


Q2_cur
Q0_cur'
Up'
Q2_cur
Q1_cur
Q0_cur'
Q2_cur'
Q1_cur'
Q0_cur'
Up
Q2_cur
Q0_cur
Up
Q2_cur
Q1_cur'
Q0_cur
Q2_cur'
Q1_cur
Q0_cur
Up'
Q1_cur
Q0_cur'
Up'
Q1_cur'
Q0_cur'
Up
Q1_cur
Q0_cur
Up
Q2_cur'
Q0_cur
Up'
Q2_nxt
Q2_cur
D
SET
Q
CLR
Q
Count(2)
Q2_cur'
Q1_nxt
Q1_cur
Count(1)
Q1_cur'
Q0_nxt
Q0_cur
Count(0)
Q0_cur'
Q0_cur'
Up
Clock