Tentamen
TSEA22 Digitalteknik

Datum	2022-06-10
Lokal	T1, U4
Tid	08-12
Kurskod	TSEA22
Kursnamn	Digitalteknik
Provkod	TEN1
Antal uppgifter	6
Kursansvarig	Anders Nilsson
Lärare som besöker skrivsalen	Anders Nilsson
Telefon under skrivtiden	013-28 2635
Besöker skrivsalen	Ca 09 och 11
Kursadministratör	Maria Hamnér, 013-28 5715
Tillåtna hjälpmedel	Inga
Preliminära betygsgränser	Poäng Betyg 41-50 5 31-40 4 21-30 3 0-20 U
Visning	Kl 10.00-11.00 20/6 på Anders Nilssons kontor (3B:512) på ISY/DA

Viktig information

- Tänk igenom din lösning NOGGRANT och använd de lösningsprinciper som kursen förevisar. Okonventionella eller tvetydiga lösningar ger poängavdrag.
- Om inget annat sägs innebär ”konstruera” att nätet ska ritas upp, samt att hela lösningsgången måste redovisas.
- AND-, OR-, NAND-, NOR-grindar får ha godtyckligt antal ingångar. XOR- och XNOR-grinder har alltid 2 ingångar, inverterare alltid en ingång.
- ”Minimal” tillståndskodning gäller alltid med avseende på vald kodning. Dvs om inget annat sägs behöver inte flera tillståndskodningar provas för att hitta en minimal tillståndskodning. Starttillstånd kan väljas fritt, men ange alltid vad som är starttillstånd.
- Om inget annat sägs kan räknare, D-vippor och dylikt förutsättas vara nollställda vid uppstart.
- Skriv läsbart! Oläsbar text kan inte bedömas och ger därmed inga poäng.

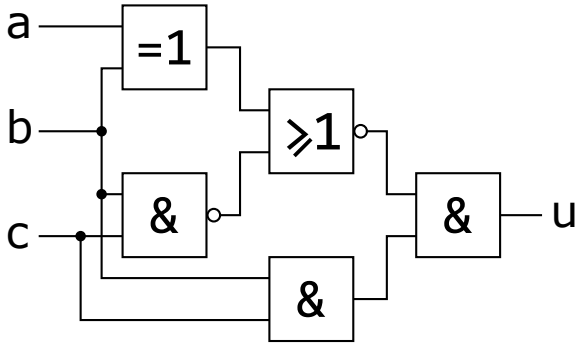
Lycka till!

Uppgift 1. Blandade uppgifter (5p)

a) Omvandla det decimala talet 610 till binärt. (1p)

b) Omvandla det hexadecimala talet 37 till decimalt. (1p)

c) Förenkla funktionen för följande kombinatoriska nät så mycket som möjligt, och rita upp det förenklade grindnätet (3p):



Lösning.
a) $610_{10} = 1001100010_2$

b) $37_{16} = 55_{10}$

c)

Tabell-lösning:

a	b	c	u
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

$u = abc$

Algebraisk lösning:
$u = ((ab' + a'b) + (bc)')'bc =$
$= ((ab' + a'b)'bc)bc =$
$= (ab')'(a'b)'bc =$
$= (a' + b)(a + b')bc =$
$= (a'a + a'b' + ab + bb')bc =$
$= a'b'bc + abbc =$
$= abc$

Uppgift 2. Bara xor-grindar(5p)

En fulladderare tar två insignalsbitar `a` och `b` samt en carry-in-signal `c`, och genererar en summa-bit `s` och även en carry-out-bit (som vi inte bryr oss om i den här uppgiften). Uttrycket för summa-biten `s` fås via följande karnaugh-diagram:

		bc			
		00	01	11	10
a	0	0	1	0	1
	1	1	0	1	0

som ger uttrycket: $s = \bar{a}\bar{b}c + \bar{a}b\bar{c} + a\bar{b}\bar{c} + abc$

Uttrycket för `s` ovan kräver fyra tre-ingångars and-grindar, en fyra-ingångars or-grind och tre inverterare.

Utgå från uttrycket för `s` ovan, använd de booleska räknelagarna och skriv om uttrycket så att bara två två-ingångars xor-grindar behövs. Ledning: $x \oplus y = \bar{x}y + x\bar{y}$

Redovisa hela lösningsgången och hoppa inte över steg i omskrivningen.

Lösning.

```
s = a'b'c + a'bc' + ab'c' + abc =
= a(b'c' + bc) + a'(b'c + bc') =
= a((b'c' + bc)'')) + a'(b'c + bc') =
= a(((b'c')'(bc)')') + a'(b'c + bc') =
= a(((b + c)(b' + c'))') + a'(b'c + b c') =
= a(bb' + bc' + b'c + cc')' + a'(b'c bc') =
= a(b'c + bc')' + a'(b'c + bc') =
= a(b⊕c)' + a'(b⊕c) =
= a⊕(b⊕c)
```

Alternativt skrivsätt:
$s = \bar{a}\bar{b}c + \bar{a}b\bar{c} + a\bar{b}\bar{c} + abc =$
$= a(\bar{b}\bar{c} + bc) + \bar{a}(\bar{b}c + b\bar{c}) =$
$= a\overline{\overline{(\bar{b}\bar{c} + bc)}} + \bar{a}(\bar{b}c + b\bar{c}) =$
$= a\overline{(\bar{b}\bar{c})(\overline{bc})} + \bar{a}(\bar{b}c + b\bar{c}) =$
$= a\overline{(b + c)(\bar{b} + \bar{c})} + \bar{a}(\bar{b}c + b\bar{c}) =$
$= a\overline{(b\bar{b} + b\bar{c} + \bar{b}c + c\bar{c})} + \bar{a}(\bar{b}c + b\bar{c}) =$

$= a\overline{(\bar{b}c + b\bar{c})} + \bar{a}(\bar{b}c + b\bar{c}) =$
$= a(\overline{b \oplus c}) + \bar{a}(b \oplus c) =$
$= a \oplus (b \oplus c)$

Uppgift 3. Kombinatorik (10p)

Skapa en minimal kombinatorisk krets som utför beräkningen

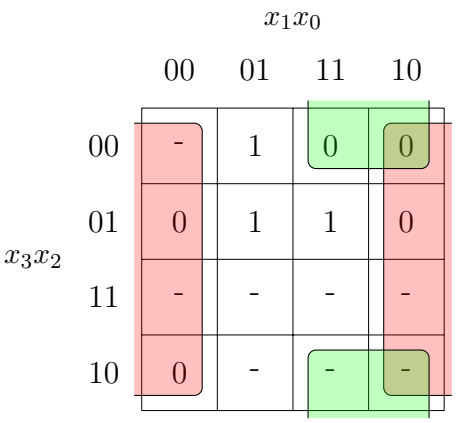
$$Y = (y_2, y_1, y_0) = (X - 3)^2 \bmod 6$$

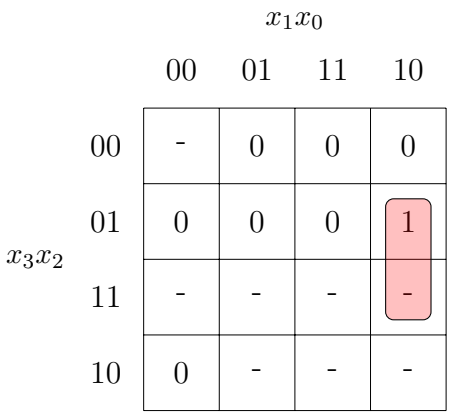
där X är ett fyrbitars binärt tal, $X = (x_3, x_2, x_1, x_0)$, $1 \leq X \leq 8$. mod 6 är modulo 6, dvs resten vid heltalsdivision med 6. Som ett exempel är $17 \bmod 6 = 5$ eftersom $17 - 2 \times 6 = 5$.

Använd enbart NOR-grindar och inverterare vid realiseringen. Tänk på att redovisa hela lösningsgången.

Lösning.

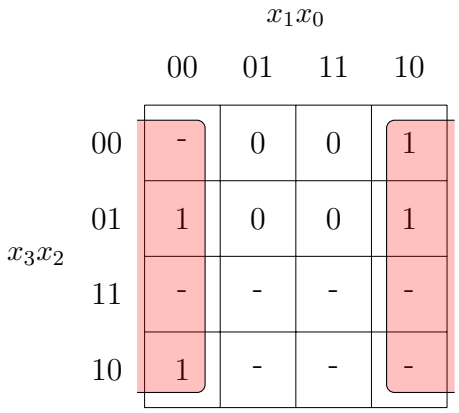
x_3	x_2	x_1	x_0	X	Y	y_2	y_1	y_0
0	0	0	0	0	-	-	-	-
0	0	0	1	1	4	1	0	0
0	0	1	0	2	1	0	0	1
0	0	1	1	3	0	0	0	0
0	1	0	0	4	1	0	0	1
0	1	0	1	5	4	1	0	0
0	1	1	0	6	3	0	1	1
0	1	1	1	7	4	1	0	0
1	0	0	0	8	1	0	0	1
1	0	0	1	9	-	-	-	-
1	0	1	0	10	-	-	-	-
1	0	1	1	11	-	-	-	-
1	1	0	0	12	-	-	-	-
1	1	0	1	13	-	-	-	-
1	1	1	0	14	-	-	-	-
1	1	1	1	15	-	-	-	-





$y_2 = (x_2'x_1 + x_0')' =$
$= ((x_2 + x_1')' + x_0')'$

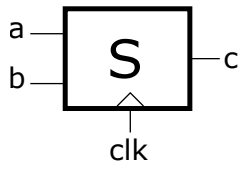
$y_1 = x_2x_1x_0' = (x_2' + x_1' + x_0)'$



$y_0 = x_0'$

Uppgift 4. Carry-generator (10p)

Konstruera en minimal sekvenskrets `S` som tar två binära två-bitars tal, `A=[`a_1a_0`]` och `B=[`b_1b_0`]`, och räknar ut utgående carry `c` när talen summeras. Summan ska dock inte beräknas, bara utgående carry `c`.



De båda talen kommer seriellt på ingångarna `a` respektive `b` med minst signifikat bit (bit a_0 resp b_0) under en klockcykel, därefter mest signifikat bit (bit a_1 resp b_1) under nästa klockcykel och samtidigt ska utgående carry `c` genereras. Därefter ska utgående carry för summan av två nya tal kunna genereras, osv. Se följande exempelsekvens:

omgång :	1	2	3	4	5	6	...
a:	1 1	1 0	1 1	0 1	1 1	1 1	...
b:	0 1	0 1	1 1	0 1	0 0	1 0	...
c:	0 1	0 0	0 1	0 1	0 0	0 1	...

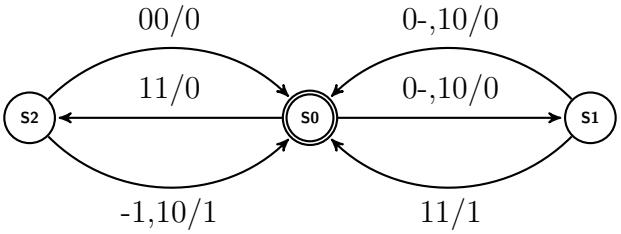
För varje omgång gäller alltså att *under första klockcykeln är utgående carry `c=0`*, men under andra klockcykeln ska utgående carry `c` genereras som 0 eller 1.

I exempelsekvensen under omgång 1 adderas talen $11_2 + 10_2 = 101_2$, vilket alltså ger carry `c=1` eftersom summan inte ryms inom två bitar, och under omgång 2 adderas talen $01_2 + 10_2 = 011_2$, vilket alltså ger carry `c=0` eftersom summan då ryms inom två bitar, osv.

Tänk på att redovisa hela lösningsgången med tillståndsdiagram med minimalt antal tillstånd, tillståndstabell, minimerade uttryck för förekommande signaler samt kretsschema.

Konstruktionen får endast använda D-vippor, NAND-grindar och inverterare.

Lösning.
Man kan föreställa sig att varje bitvis addition görs med en fulladderare. Den första additionen ger alltså två möjliga fall, dvs antingen så blir carry=0 (gå till `S1`) eller så blir carry=1 (gå till `S2`). Dock så ska utgående carry än så länge vara 0.
Vid additionen av nästa bitpar gäller samma sak, men nu finns även en tidigare ingående carry att ta hänsyn till (som då är `0` eller `1`), samt att denna gång ska utgående carry genereras, på bågen när man går tillbaka till tillstånd `S0`.
Starttillstånd är `S0`, och på bågarna i följande tillståndsdiagram gäller `ab/c`:



Kodning: Binär
Starttillståndet är $q = 00$.

		$q_1^+ q_0^+ / c$			
S	$q_1 q_0$	$ab = 00$	$ab = 01$	$ab = 10$	$ab = 11$
S0	00	01/0	01/0	01/0	10/0
S1	01	00/0	00/0	00/0	00/1
S2	10	00/0	00/1	00/1	00/1
--	--	--/-	--/-	--/-	--/-

$q_1 q_0$

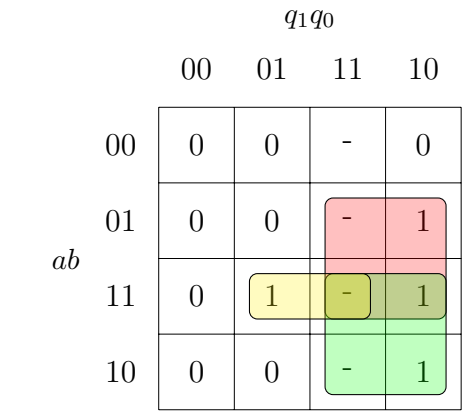
ab	00	01	11	10
00	0	0	-	0
01	0	0	-	0
11	1	0	-	0
10	0	0	-	0

$q_1^+ = abq_1'q_0'$

$q_1 q_0$

ab	00	01	11	10
00	1	0	-	0
01	1	0	-	0
11	0	0	-	0
10	1	0	-	0

$q_0^+ = a'q_1'q_0' + b'q_1'q_0' = ((a'q_1'q_0')'(b'q_1'q_0')')'$



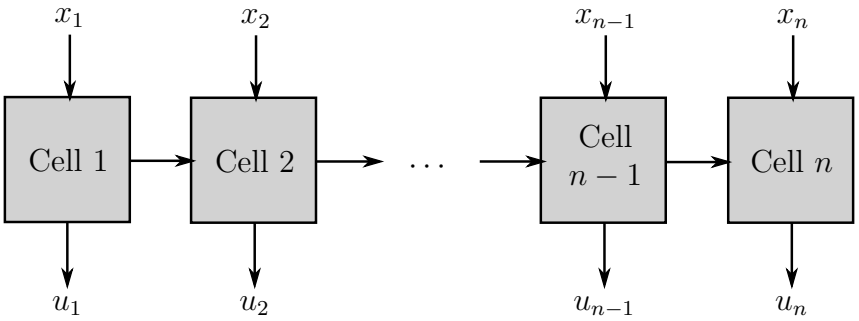
$$c = bq_1 + aq_1 + abq_0 = ((bq_1)'(aq_1)'(abq_0)')'$$

Uppgift 5. Släpp igenom varannan grupp av ettor (10p)

Konstruera en iterativ kombinatorisk krets med struktur enligt figur 1, ingångar $x_1, x_2, \ldots, x_n$ och utgångar $u_1, u_2, \ldots, u_n$. Kretsen ska släppa igenom varannan grupp av ettor med början från den andra gruppen av ettor. För att förtydliga specifikationen ges några exempel för $n = 19$, dvs $x = (x_1, x_2, \ldots, x_{19})$:

i :	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
x_i :	**1**	0	**1**	0	0	**1**	**1**	**1**	0	**1**	**1**	**1**	**1**	0	0	**1**	**1**	0	**1**
u_i :	0	0	**1**	0	0	0	0	0	0	**1**	**1**	**1**	**1**	0	0	0	0	0	**1**

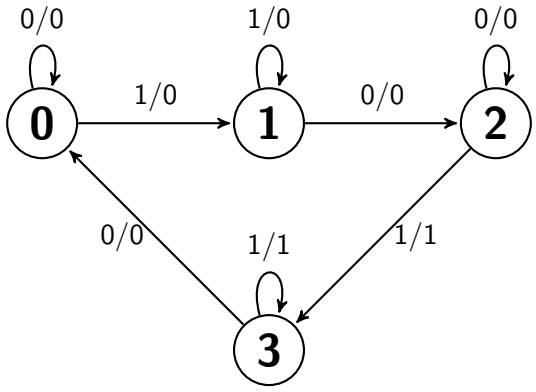
I exemplet släpps 1:orna i grupp nummer 2, 4 och 6 igenom medan 1:orna i grupp nummer 1, 3 och 5 blockeras.



Figur 1: Kretsens struktur, insignaler och utsignal.

Ni har tillgång till AND-, OR-grindar och inverterare och kan anta att $n \geq 5$. För full poäng krävs tillståndsdiagram med minimalt antal tillstånd, tillståndstabell, minimerade uttryck för alla celler och kretsschema med minimerade celler.

Lösning. Tillståndsdiagram med bågmarkeringar x/u.



Kodning alternativ 1: Binär
Starttillståndet är $q = 00$.

	$q_1^+q_0^+/u$	
q_1q_0	$x=0$	$x=1$
00	00/0	01/0
01	10/0	01/0
11	00/0	11/1
10	10/0	11/1

Cell 1: $(q_1, q_0) = (0, 0)$

$$q_1^+ = 0$$
$$q_0^+ = x_1$$
$$u_1 = 0$$

Cell 2: $(q_1, q_0) \in \{(0, 0), (0, 1)\}$

$$q_1^+ = q_0 x_2'$$
$$q_0^+ = x_2$$
$$u_2 = 0$$

Cell 3: $(q_1, q_0) \in \{(0, 0), (0, 1), (1, 0)\}$

$$q_1^+ = q_1 + q_0 x_3'$$
$$q_0^+ = x_3$$
$$u_3 = q_1 x_3$$

Cell $k \in \{4, \ldots, n-1\}$:

$$q_1^+ = q_1' q_0 x_k' + q_1 q_0' + \underline{q_1 x_k}$$
$$q_0^+ = x_k (behövs\ ej\ för\ cell\ n-1)$$
$$u_k = \underline{q_1 x_k}$$

Cell n:

$$u_k = q_1 x_k$$

Grinddelning av understruken term ger $3n-10$ inverterare, $3n-8$ AND-grindar och $n-3$ OR-grindar.

Kodning alternativ 2: Gray
Starttillståndet är $q = 00$.

	$q_1^+q_0^+/u$	
q_1q_0	$x=0$	$x=1$
00	00/0	01/0
01	11/0	01/0
11	11/0	10/1
10	00/0	10/1

Cell 1: $(q_1, q_0) = (0, 0)$

$$\begin{aligned} q_1^+ &= 0 \\ q_0^+ &= x_1 \\ u_1 &= 0 \end{aligned}$$

Cell 2: $(q_1, q_0) \in \{(0, 0), (0, 1)\}$

$$\begin{aligned} q_1^+ &= q_0 x_2' \\ q_0^+ &= x_2 + q_0 \\ u_2 &= 0 \end{aligned}$$

Cell 3: $(q_1, q_0) \in \{(0, 0), (0, 1), (1, 1)\}$

$$\begin{aligned} q_1^+ &= \underline{q_0 x_3'} + q_1 \\ q_0^+ &= \underline{q_0 x_3'} + q_1' x_3 \\ u_3 &= q_1 x_3 \end{aligned}$$

Cell $k \in \{4, \ldots, n-1\}$:

$$\begin{aligned} q_1^+ &= \underline{\underline{q_0 x_k'}} + \underline{q_1 x_k} \\ q_0^+ &= \underline{\underline{q_0 x_k'}} + q_1' x_k (\textit{behövs ej för cell } n-1) \\ u_k &= \underline{q_1 x_k} \end{aligned}$$

Cell n:

$$u_n = q_1 x_n$$

Grinddelning av understrukna termer ger $2n - 5$ inverterare, $3n - 7$ AND-grindar och $2n - 5$ OR-grindar.

Uppgift 6. Fibonaccigenerator (10p)

Konstruera en synkron sekvenskrets enligt figur 2 som ska visa entalsdelen på Fibonaccitalen i tur och ordning på en sjusegmentsdisplay.

Figur 2: Krets för att generera entalssiffran i Fibonaccitalen.

Fibonaccitalen definieras rekursivt enligt

$$a_0 = 0$$
$$a_1 = 1$$
$$a_i = a_{i-1} + a_{i-2} \text{ för } i > 1$$

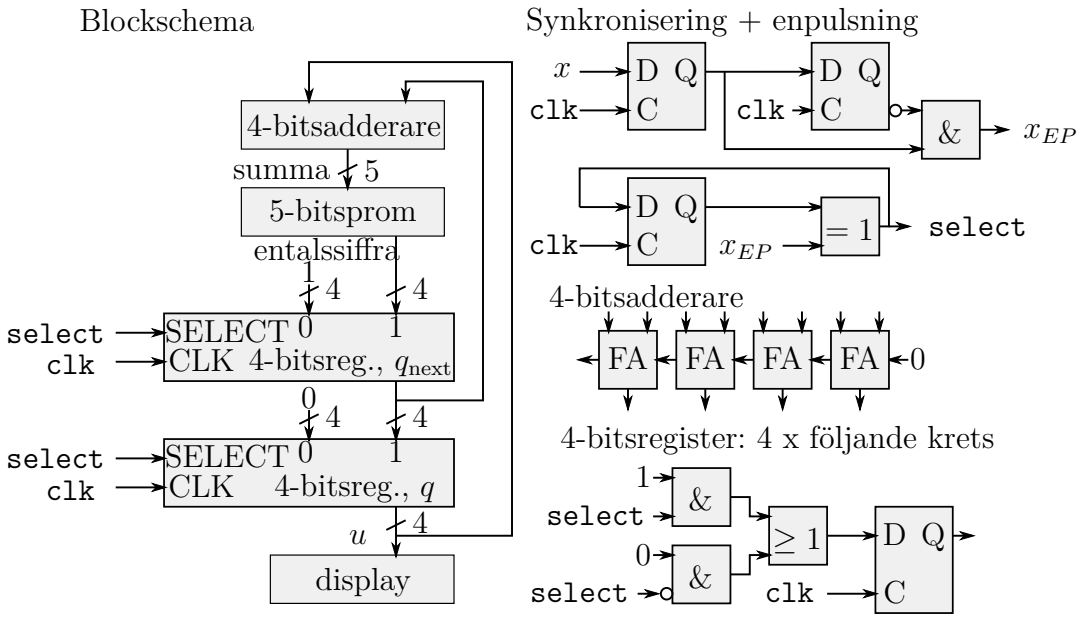
Detta betyder att början på sekvensen av Fibonaccital blir $0, 1, 1, 2, 3, 5, 8, 13, 21, \ldots$. Början på följden av entalssiffror som kretsen ska visa blir alltså $0, 1, 1, 2, 3, 5, 8, 3, 1, \ldots$. **Tips:** Det räcker med att använda entalssiffran av föregående två Fibonaccital för att räkna ut entalssiffran på nästa Fibonaccital.

Kretsen har två lägen. Antingen visas Fibonaccitalens entalssiffra upp en efter en i takt med klockan eller så är kretsen i vila och visar Fibonaccitalet $a_0 = 0$ hela tiden (dvs Fibonaccisekvensen startas om från början). Bytet mellan de två lägena sker genom att trycka på knappen kopplad till x. Längden på knapptrycket ska inte påverka kretsens funktion och x kommer in asynkront i kretsen. Kretsen ska starta i viloläget.

Till er konstruktion har ni heladderare, valfria grindar och inverterare, binärräknare med upp till 4-bitar, D-vippor och ett eller flera PROM med ordlängd på 4 bitar. Om $u = (u_3, u_2, u_1, u_0)$ är ett binärkodat tal där mest signifilkant bit är u_3 kommer sjusegmentsdisplayen visa motsvarande hexadecimala siffra. Kretsens initialtillstånd ska anges men den asynkrona kretsen för initiering behöver inte redovisas. Onödigt komplicerade konstruktioner ger poängavdrag, asynkrona lösningar ger kraftigt reducerad poäng.

Lösning. Figur 3 visar ett exempel på en lösning. Insignalen x synkroniseras och enpulsas x_{EP}. D-vippan tillsammans med XOR-grinden styr vilket läge kretsen är i `select`, `select` = 0 betyder viloläge och 1 att kretsen visar Fibonaccitalen i tur och ordning.

Det finns två 4-bitsregister, ett sparar nuvarande Fibonaccital q och det andra nästa Fibonaccital q_{next}. Displayen kopplas alltså direkt till utgången på ena 4-



Figur 3: Ett exempel på krets som löser uppgiften.

bitsregistret, dvs $u = q$. Vid klockning görs följande uppdatering

$$\begin{aligned} q^+ &= q_{\text{next}} \\ q^+_{\text{next}} &= \quad \bmod (q + q_{\text{next}}, 10) \end{aligned}$$

Additionen i den andra raden realiseras med en 4-bitsadderare där eventuellt överspill kan kastas. Resultatet av additionen kan bli ett godtyckligt 5-bitstal. 5-bitspromet tar ut entalsdelen i det det binära talet enligt följande tabell:

summa	entalssiffra
0 (00000)	0 (0000)
1 (00001)	1 (0001)
⋮	⋮
9 (01001)	9 (1001)
10 (01010)	0 (0000)
11 (01011)	1 (0001)
⋮	⋮
18 (10010)	8 (1000)
19-31	don't care (0000)

När select = 0 laddas det första Fibonaccitalet in i q, dvs $q^+ = 0$ och det andra Fibonaccitalet in i $q^+_{\text{next}} = 1$, vilket gör att $u = 0$ så länge krestsen är i viloläget.

Alla register ska ha initialtillstånd 0 utom synkroniseringsvippan som inte bör resettas alls.