3) ostvariti muxom koji multiplekisira po B pa po A pa po C.

$$f = \bar{B} f_{B=0} + B \cdot f_{B=1}$$

$$= \bar{B}\left(\bar{A} f_{B=0,A=0} + A f_{B=0,A=1}\right) + B\left(\bar{A} f_{B=1,A=0} + A f_{B=1,A=1}\right)$$

$$= \bar{B}\left\{\bar{A}\left(\bar{C} f_{B=0,A=0,C=0} + C f_{B=0,A=0,C=1}\right) + A\left(\bar{C} f_{B=0,A=1,C=0} + C f_{B=0,A=1,C=1}\right)\right\}$$

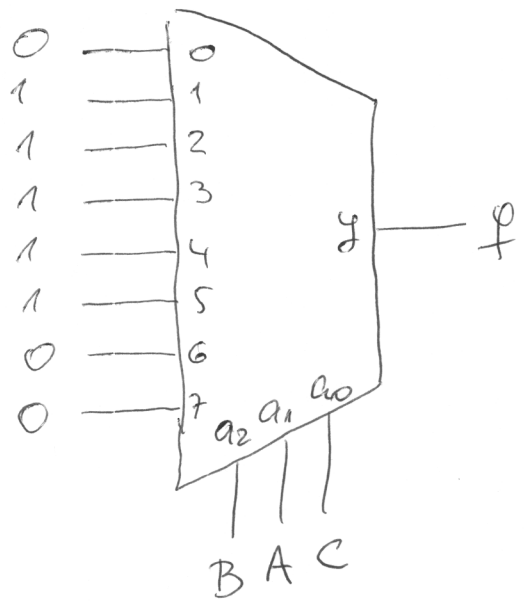
$$+ B\left\{\bar{A}\left(\bar{C} f_{B=1,A=0,C=0} + C f_{B=1,A=0,C=1}\right) + A\left(\bar{C} f_{B=1,A=1,C=0} + C f_{B=1,A=1,C=1}\right)\right\}$$

$$= \bar{B}\left\{\bar{A}\left(\bar{C}\cdot 0 + C\cdot 1\right) + A\left(\bar{C}\cdot 1 + C\cdot 1\right)\right\}$$

$$+ B\left\{\bar{A}\left(\bar{C}\cdot 1 + C\cdot 1\right) + A\left(\bar{C}\cdot 0 + C\cdot 0\right)\right\}$$

$$= \bar{B}\bar{A}\bar{C}\cdot 0 + \bar{B}\bar{A}C\cdot 1 + \bar{B}A\bar{C}\cdot 1 + \bar{B}AC\cdot 1 + B\bar{A}\bar{C}\cdot 1 + B\bar{A}C\cdot 1 + BA\bar{C}\cdot 0 + BAC\cdot 0$$

reziduale funkcije od ϕ varijabli (konstante)

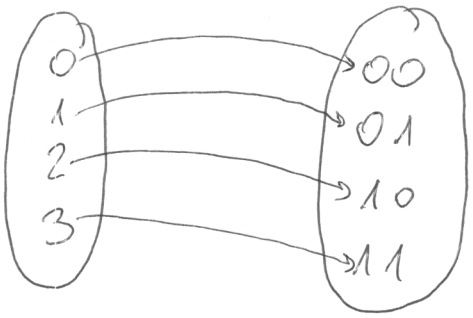


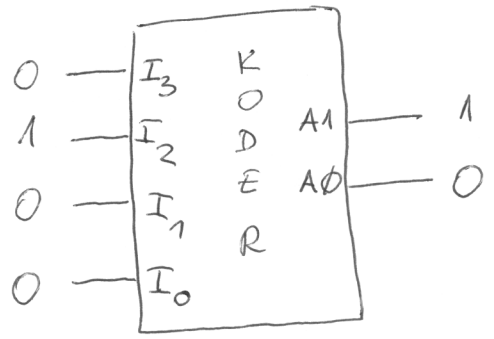
- ako funkciju ostvarujemo rezidualnim funkcijama od ϕ varijabli, kažemo da smo napravili "simulaciju rada permanentne memorije (ROM)" ⇒ neefikasno
- efikasnije rješenje je koristiti trivijalne reziduale funkcije ⇒ reziduale funkcije od jedne varijable

Koder

Obavlja kodiranje. Omogućava korisniku da odabere jedan simbol te na izlazu generira pridruženu kodnu riječ. Ima ulaza koliko ima simbola a izlaza koliko kodna riječ ima bitova.

Primjer:

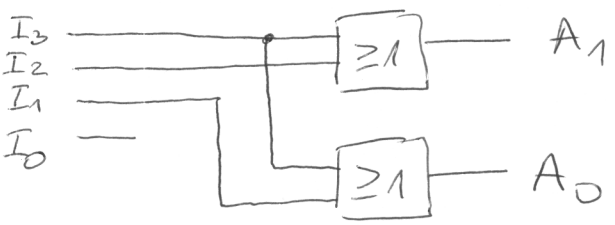




Ako korisnika natjeramo da točno na jednom ulazu smije (i mora) biti jedinica, sklop možemo projektirati trivijalno:

I_3	I_2	I_1	I_0	A_1	A_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

$$\Rightarrow \quad A_1 = I_3 + I_2$$
$$A_0 = I_3 + I_1$$



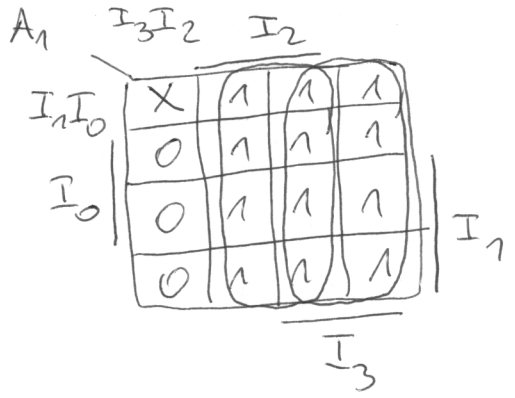
- ostaju neriješena dva problema
  1) kako korisnika natjerati da ne postavi jedinice na dva ili više ulaza
  2) kako ga natjerati da postavi jedinicu na barem jedan ulaz

Prioritetni koder

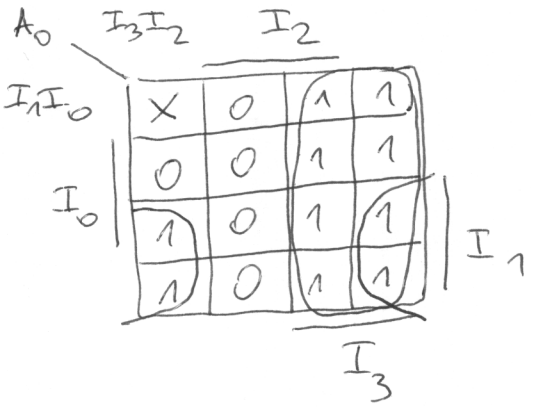
- rješava oba moćena problema
  1) uvodi prioritete između ulaza pa u slučaju da je korisnik postavio jedinice na više ulaza, na izlazu generira kôd ulaza najvišeg prioriteta
     (tipično, I_0 je najmanjeg prioriteta, I_{n-1} najvišeg)
  2) uvodi dodatni izlaz y koji govori je li na izlazima doista kodna riječ; ako su svi ulazi nula, nije odabran niti jedan simbol pa će y biti 0; ako je barem jedan simbol zatražen, na izlazu će biti kôd onog najvećeg prioriteta a y će biti 1

I_3	I_2	I_1	I_0	A_1	A_0	y
0	0	0	0	x	x	0
0	0	0	1	0	0	1
0	0	1	x	0	1	1
0	1	x	x	1	0	1
1	x	x	x	1	1	1

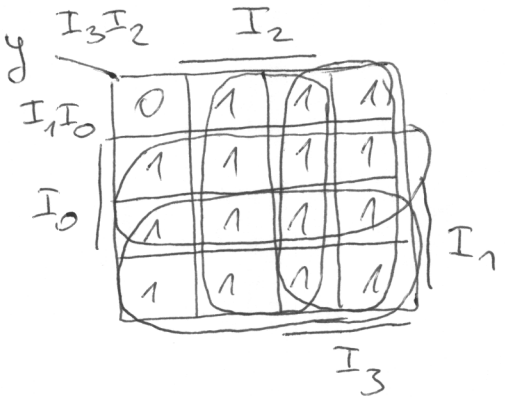
Možemo ga projektirati tablično:



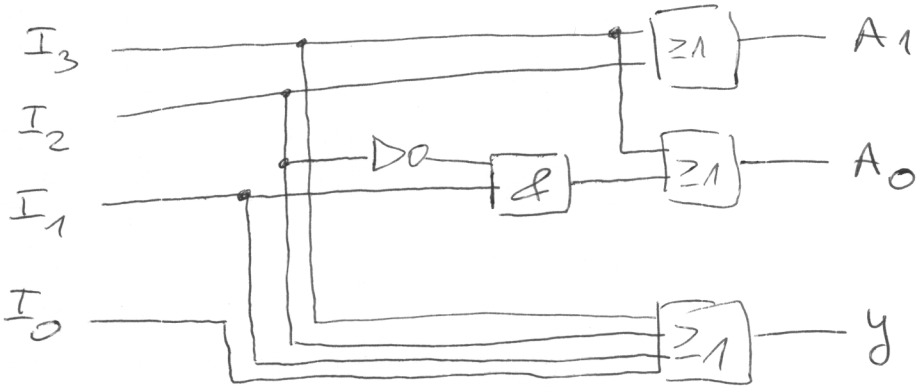
$A_1 = I_3 + I_2$



$A_0 = I_3 + \overline{I}_2 I_1$



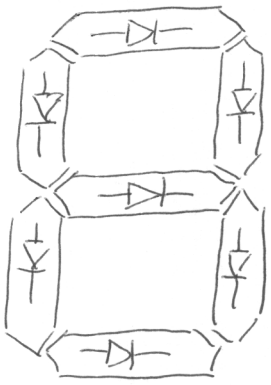
$y = I_3 + I_2 + I_1 + I_0$



- možemo ga projektirati i kako je prikazano u slideovima (vidi slide 49)

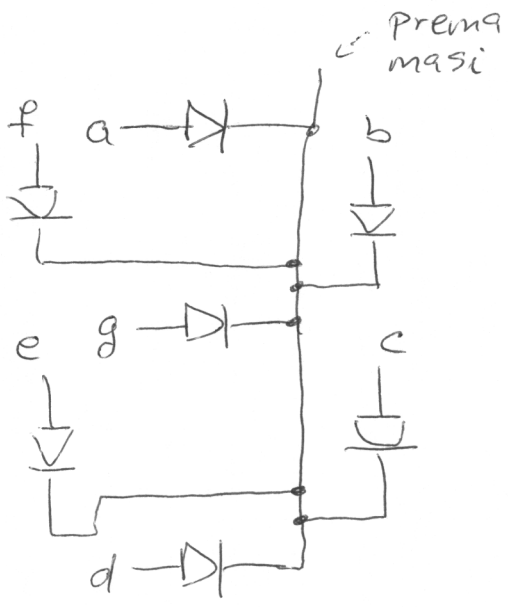
7-segmentni prikaznik

- 7 svjetlećih dioda

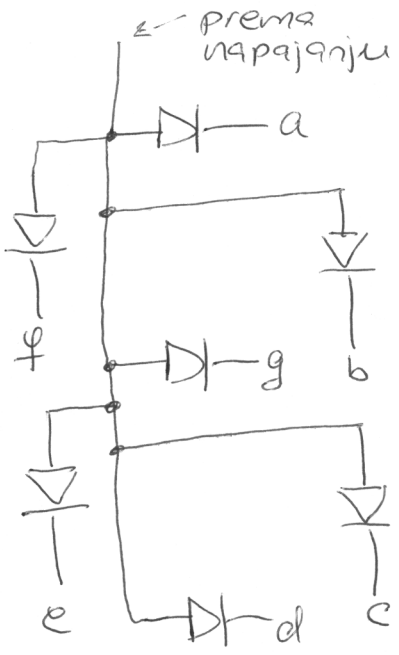


- zbog uštede na broju pinova, jedna strana svih svjetlećih dioda spaja se zajedno i izvodi na jedan pin

a) upravljanje jedinicom



b) upravljanje nulom



Pretvornik kôda BCD u 7-segmentni

Sklop koji na ulazu dobiva BCD kod znamenke a na izlazu generira bitove a,b,c,d,e,f,g uz koje će 7-segmentni prikaznik prikazati tu znamenku

- postoje čipovi koji podržavaju upravljanje nulom (google: 7447 datasheet), čipovi kojima se na jeden od ulaza dovodi informacija treba li upravljati nulom ili jedinicom (google: CD4543 datasheet, pin je PHASE, na izlazu je upravljivi invertor)

za konstrukciju vidi slideove 53 i 54.