Digital Design Kap 4

Karnaugh Map – Härleder min/max. Alternativ till sanningstabell.   
Ringa in så stora block som möjligt för billigare implentation.

Gray Code – endast en siffra som skiljer mellan 2 grannar.

0 1 3 2

4 5 7 6

12 13 15 14

8 9 11 10

Literal (Bokstavlig?), antalet variabler i en produkt term,  
Ex, x1,’x2, x3 has three literals .

Implicant (Implikant?), som indikerar att inmatningsvärdet är 1.   
Ex, ’x1,’x2,x3 \* x1,’x2,x3 för f two implicant for f

Prime Implicant, när man har nått minsta antalet varialer I en “literals”  
Ex, ’x , x1x2

Cost, antalet grindar + ingångar. En ingång kan vara x eller ’x  
Ex, g = ’(x1’x2 + x3) \* (’x4 + x5)  
5 grindar, 2 AND 2 OR 1 NOT  
9 ingångar, 2 \* 2 + 2 \* 2 + 1   
Cost 14 ( 9 + 5)

Essential Prome Implicant, Om prime implicant endast finns i en minterm.

**Hitta min kostnad:**  
1. Hitta alla prime implicants Karnaugh  
2. Hitta essential prime implicants (“Förenkla Karnaugh”)  
3. Korrigera ev. ”Don’t care”

SOP Sum-of-product min 1:or  
POS Product-of-sum max 0:or

Don’t care, “incomplelty specified”   
Ett tillstånd som aldrig kommer att inträffa, så det spelar ingne roll om det är 0 / 1.  
d i karnaugh-map.

Ex. ∑m(2,4,5,6,10) + d(12,13,14,15)

Detta gör att man få ner kostnaden, svårt att hitta optimala lösningen, CAD hjälper till i verkligheten.

Kan använda ”multiple-output circuits”, 2∫ som delar grinder för att minska kostnaden.

Tvånivås grindnät är bara effektivt id få variabler, vid många variabler får man lätt fan-in (ingångs problem). Detta löser man med ”multilevel logic expression”. Två viktiga tekniker är factoring and functional decomposition.

Förenkla uttrycken och få ner antalet grindar.

BILD 4.21

Detta blev ett 3 nivås grindnät.  
Fler logiska nivåer ökar propagtion-time(tiden att slå om), men behöver färre ingångar på grindarna.

AND-OR SOP 🡪 NAND-NAND  
OR-AND POS 🡪 NOR-NOR

NAND = invers utmatning på AND  
NAND = invers inmatning på OR

Analysera flerlager kretsar är svårt, analysera subfunktioner.  
Antingen, från input -> output eller output -> input