Week 1

Hamming distance : The hamming distance between two n-bit number determined by checking the same corresponding bits.

1010 - 0100 → distance : 3

Decimal to binary : We divide the decimal number by 2 up to 0. After each operation the remainder is written to LSB to MSB.

$15 \xrightarrow[1]{} 7 \xrightarrow[1]{} 3 \xrightarrow[1]{} 1 \xrightarrow[1]{} 0$: 1111

Negative numbers : First we take 1's complement of the unsigned number that corresponds to the negative one, then add 1 to this number. The whole operation is 2's complement.

$$\begin{array}{ll} 0100 \longrightarrow & 1011 \\ (4) & \underline{+ \quad\;\; 1} \\ & 1100 : -4 \end{array}$$

Addition (Unsigned) :

$$\begin{array}{rl} 01110101 & : 117 \\ \underline{+01100011} & : 99 \\ \textcircled{0}\,10011000 & : 216 \end{array}$$

No carry

$$\begin{array}{rl} 11111111 & : 255 \\ \underline{+\;00000001} & : 1 \\ \textcircled{1}00000000 & : 256 \end{array}$$

carry

we use the carry number in unsigned numbers.

Addition (Signed) :

$$\begin{array}{rl} 11111111 & : -1 \\ \underline{+00000001} & : 1 \\ \textcircled{1}00000000 & : 0 \end{array}$$

carry

$$\begin{array}{rl} 11111111 & : -1 \\ \underline{+11111111} & : -1 \\ \textcircled{1}11111110 & \end{array}$$

carry

we ignore the carry numbers, the nsf number determines the sign.

Subtraction (Unsigned):

$$\begin{array}{r} 00000101 : 5 \\ -\underline{00000001 : 1} \end{array} \xrightarrow{\text{2's comp.}} \begin{array}{r} 00000101 : 5 \\ -\underline{11111111 : -1} \\ \textcircled{1}00000100 : 4 \end{array}$$

↳carry (no borrow)

There must be carry number when we carry out a subtractio operation with unsigned int, but we ignore the carry number.

$$\begin{array}{r} 00000001 \\ -\underline{00000101} \end{array} \xrightarrow{\text{2's comp.}} \begin{array}{r} 00000001 : 1 \\ -\underline{11111011 : -5} \\ 011111100 \end{array}$$

→cannot be represented

the carry number is "0", so we cannot represent this operation.

Overflow:

Addition → pos. + pos. → neg. | neg. + neg. → pos.

Subtraction → pos. − neg. → neg. | neg. − pos. → pos.

WEEK 2

Axioms

any $a, b \in B$

closure: $a+b \in B$, $a \cdot b \in B$

commutative: $a+b = b+a$, $a \cdot b = b \cdot a$

associative: $a+(b+c) = (a+b)+c$, $a.(b \cdot c) = (a.b) \cdot c$

identity: $a+0 = a$, $a \cdot 1 = a$

distributive: $a+(b \cdot c) = (a+b) \cdot (a+c)$, $a.(b+c) = a.b + a.c$

inverse: $a+\bar{a} = 1$, $a \cdot \bar{a} = 0$

- Duals of all proven theorems are also theorems.

Theorems

annihilator (dominance): $a+1 = 1$, $a \cdot 0 = 0$

involution: $\bar{\bar{a}} = a$

idempotency: $a+a+a \cdots = a$, $a.a.a \ldots = a$

absorption: $a+ab = a$, $a.(a+b) = a$

de morgan:

- $a \cdot b + \bar{a} \cdot c$ has 3 variables and 4 literals.

Disjunctive Normal Form (DNF): Sum of Products (SOP). OR of ANDs. → $a \cdot b + c \cdot \bar{d} + \bar{e} \cdot f$

Conjunctive Normal Form (CNF): Product of Sums (POS). AND of ORS. → $(a+b) \cdot (\bar{a}+c+d)$

Order Relation

x_1: 1011 If each component of x_1 is smaller than or equal to x_2, $x_1 \nless x_2$. But there is no order relation between
x_2: 1101 x_1 and x_2.

If $E(x) \leq F(x)$ then $E(x)$ implies $F(x)$, $E(x) \Rightarrow F(x)$, $F(x)$ covers $E(x)$.

- $E + E \cdot F = E$
- $E \cdot (E+F) = E$

proof: $E \cdot 1 + E \cdot F \rightarrow E(1+F) \rightarrow E$

- $E + \bar{E} \cdot F = E + F$
- $E \cdot (\bar{E}+F) = E \cdot F$

proof: $|E+\bar{E}| \cdot (E+F) \rightarrow 1 \cdot (E+F) \rightarrow E+F$

The Consensus Theorem (SOP Form)

$E_1(x_2 \cdot \ldots \, x_m)$, $E_2(x_2, \ldots \, x_n)$

$E = x_1 \cdot E_1 + \bar{x}_1 \cdot E_2$ (x_1 is biform variable)

- $E_1 \cdot E_2$ is consensus term
- The consensus term is redundant and can be eliminated.

$x_1E_1 + \bar{x}_1E_2 + E_1 \cdot E_2 = x_1E_1 + \bar{x}_1 \cdot E_2$

proof: $x_1E_1 + \bar{x}_1E_2 + E_1E_2(x_1 + \bar{x}_1) \Rightarrow x_1E_1 + \bar{x}_1E_2 + x_1E_1E_2 + \bar{x}_1E_1E_2$ the pair ones can absorb the other one.

$$\downarrow$$

$y_1E_1 + \bar{X}_1 \cdot E_2$

The Consensus Theorem (POS Form)

$E_1(x_2,\dots x_n)$, $E_2(x_2,\dots\dots,x_m)$

$E = (x_1+E_1)\cdot(\bar{x}_1+E_2)$

- E_1+E_2 is consensus term

$(x_1+E_1)\cdot(\bar{x}_1+E_2)\cdot(E_1+E_2) = (x_1+E_1)(\bar{x}_1+E_2)$

Boolean Functions

1) Basic Functions : Multiple inputs, single output.

- There are $2^{(2^n)}$ possible basic functions for n binary variables.

2) General Functions: Multiple inputs, multiple outputs.

3) Incompletely Specified Functions: Some of the outputs won't be generated. These outputs are don't care terms.

A	B	F
0	0	1
0	1	X
1	0	0
1	1	1

$\bar{A}B$ is don't care term.

X means it can be 0 or 1, we cannot demonstrate it.

Indexed Representations

Basic Function

Row Num.	Input x_1, x_2	Output y
0	0 0	1
1	0 1	0
2	1 0	1
3	1 1	0

representation

$y = f(\underbrace{x_1, x_2}_{\text{order is important}}) = U_{\underbrace{1}_{\text{Set of 1-generating points}}}(\overbrace{0,2}^{\text{the row numbers of output 1}})$

$y = f(x_2,x_1) = U_1(0,1)$

$y = f(x_1,x_2) = U_0(1,3)$

All three represents the same function.

Incompletely Specified Function

N	x_1	x_2	y_1	y_2
0	0	0	1	1
1	0	1	0	Φ
2	1	0	Φ	0
3	1	1	0	Φ

- We have to write at least 2 of the three different groups (1-generating, 0-gene., don't care)

$y_1 = f(x_1,x_2) = U_1(0) + U_0(1,3)$ there are 2 more repr., but enough

$y_2 = f(x_1,x_2) = U_{\Phi}(1,3) + U_1(0)$

1st Canonical Form (SOP)

- a function with 2 variables has 4 minterms.

ab, $\bar{a}b$, $a\bar{b}$, $\bar{a}\bar{b}$

A	B	C	F	$\bar{F}$	
0	0	0	0	1	m0
0	0	1	1	0	m1
0	1	0	0	1	m2
0	1	1	1	0	m3
1	0	0	0	1	m4
1	0	1	1	0	m5
1	1	0	1	0	m6
1	1	1	1	0	m7

$$F(A,B,C) = \bar{A}\bar{B}C + \bar{A}BC + A\bar{B}C + AB\bar{C} + ABC \xrightarrow{\text{simplify}} AB + C$$

$$\overline{F(A,B,C)} = \bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + A\bar{B}\bar{C}$$

$$F(A,B,C) = \sum m(1,3,5,6,7)$$

$$F = \sum_{A,B,C}(1,3,5,6,7)$$

2nd Canonical Form (POS)

- a function with 2 variables has 4 maxterms.

$a+b$, $\bar{a}+b$, $a+\bar{b}$, $\bar{a}+\bar{b}$

A	B	C	F	$\bar{F}$	
0	0	0	0	1	M0
0	0	1	1	0	M1
0	1	0	0	1	M2
0	1	1	1	0	M3
1	0	0	0	1	M4
1	0	1	1	0	M5
1	1	0	1	0	M6
1	1	1	1	0	M7

$$F(A,B,C) = (A+B+C)\cdot(A+\bar{B}+C)\cdot(\bar{A}+B+C) \xrightarrow{\text{simplify}} (A+C)\cdot(B+C)$$

$$\overline{F(A,B,C)} = (A+B+\bar{C})\cdot(A+\bar{B}+\bar{C})\cdot(\bar{A}+B+\bar{C})\cdot(\bar{A}+\bar{B}+C)\cdot(\bar{A}+\bar{B}+\bar{C})$$

$$F(A,B,C) = \Pi M(0,2,4)$$

$$F = \Pi_{A,B,C}(0,2,4)$$

Conversions Between Canonical Forms

Complement

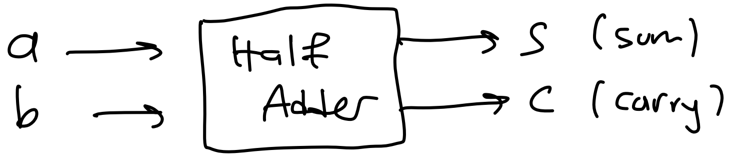
$$F(A,B,C) = \Sigma m(1,3,5,6,7) \longrightarrow \overline{F(A,B,C)} = \Sigma m(0,2,4) \rightarrow \bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + A\bar{B}\bar{C}$$

↓ De Morgan

$$F(A,B,C) = \Pi M(0,2,4) \rightarrow (A+B+C)\cdot(A+\bar{B}+C)\cdot(\bar{A}+B+C)$$

W5

Half Adder →

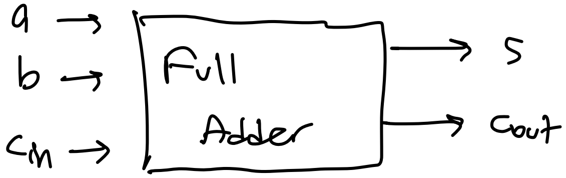


Doesn't contain carry input.

$s = a \oplus b$ a, b are 1-bit nums.

$c = a . b$

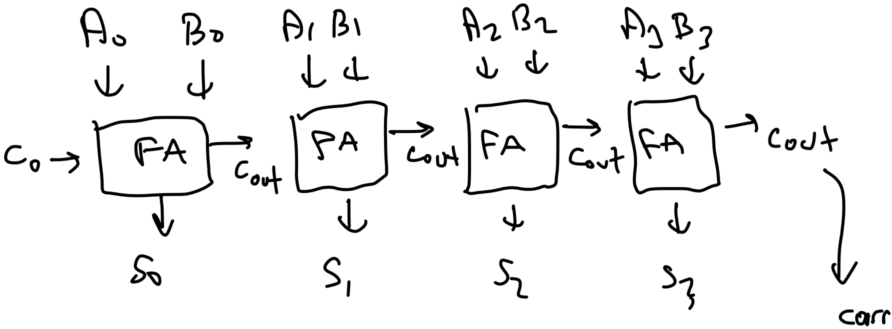
Full Adder →



$S = a \oplus b \oplus c$ a, b are 1-bit nums.

$Cout = ab + bc + ac$

Parallel Adder →



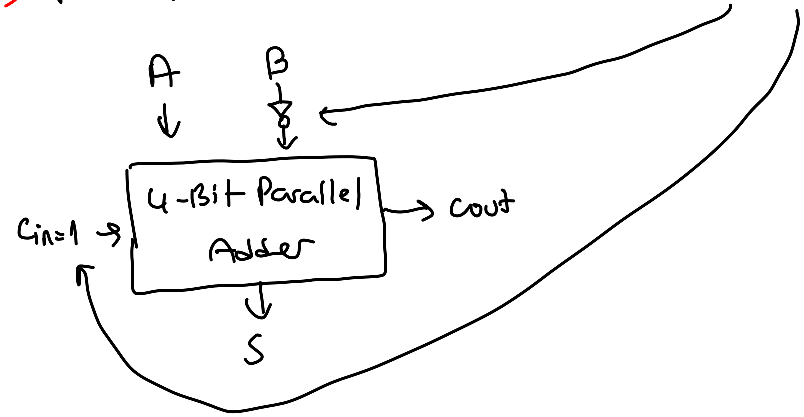
4-Bit PA consist of 4 FA.

A, B are 4-bit nums.

$$\begin{array}{r} A_3A_2A_1A_0 \\ + \; B_3B_2B_1B_0 \\ \hline \text{carry} \; S_3S_2S_1S_0 \end{array}$$

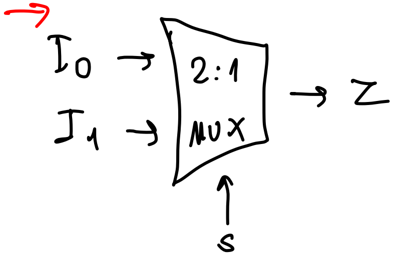
Parallel Adder (Subtraction) → A - B (4-bit)

$A - B = A + \overline{B} + 1$



Multiplexer (MUX) (m:1) →

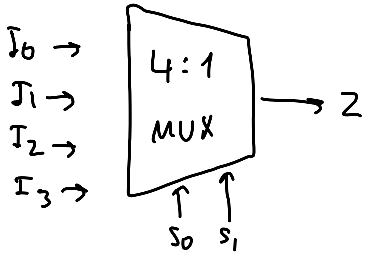
It selects one input according to selector input.



$S=0 \rightarrow I_0$

$S=1 \rightarrow I_1$

$Z = \bar{S} I_0 + S I_1$



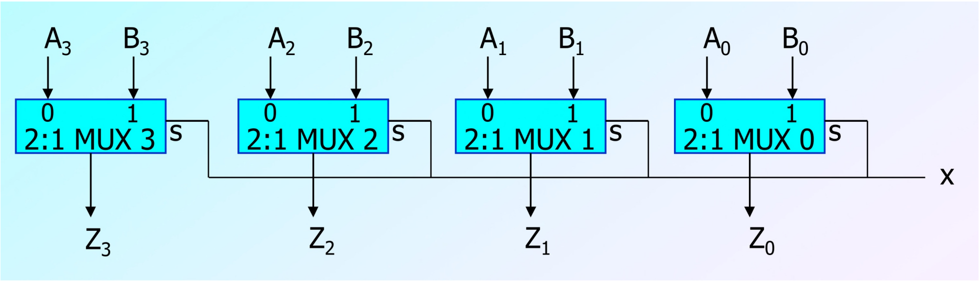
$S_1=0,\ S_0=0 \rightarrow I_0$

$S_1=0,\ S_0=1 \rightarrow I_1$

$S_1=1,\ S_0=0 \rightarrow I_2$

$S_1=1,\ S_0=1 \rightarrow I_3$

Parallel MUX →

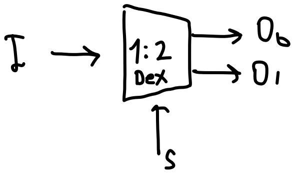


$x=0 \rightarrow Z=A$

$x=1 \rightarrow Z=B$

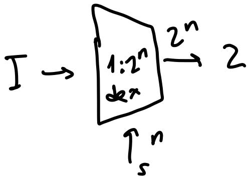
Demultiplexer (1:m) →

It drives the input to the output according to the selector.



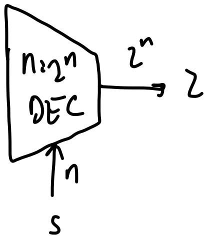
$S=0 \rightarrow I=O_0$

$S=1 \rightarrow I=O_1$



Decoder →

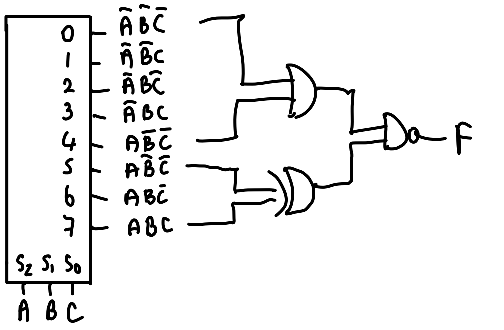
Decoder is a demultiplexer with constant 1 input.

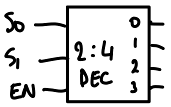


n number of selector

2^n number of output

- We can use decoder as a minterm generator. → Then implement a function using the minterms.





If EN: 1 → outputs are valid.

EN: 0 → all outputs are zero.

It creates a new case "three-state outputs"

w6

PLA → input (and) → output (or) SOP / AND and OR gates are programmable.

PAL → Same structure. / Only AND gates are programmable.

w7

- static 0 hazard → POS form
- static 1 hazard → SOP form
- $Z = AB + B'C$ → there can be static 1 hazard, to avoid that; the consensus term of Z should be added to the circuit, but it's not efficient due to the increasing of cost.

$Z = AB + B'C + AC$

w8

- Flip-flop controlled by clock signal. Latch don't.
- Clock → Edge Triggered → (Positive logic) (0→1) → rising edge → register time = setup t. + hold t.

  (negative logic) (1→0) → falling edge

  → Input changes in the settling time.

  setup t.: (minimum time the data signal should be steady before the transition) — hold t.: (→ " " " " after the transition)

- A memory unit must have 1) Two stable states 2) Control input

- S-R Latch

S sets 1 to output when its value 1. R sets 0 to output when its value 1. (S → set, R → reset)

case 1: S → 1, R → 0 → Q = 1, Q' = 0
S → 0, R → 0 → Q = 1, Q' = 0

case 2: S → 0, R → 1 → Q = 0, Q' = 1
S → 0, R → 0 → Q = 0, Q' = 1

} If both inputs are zero, latch preserves its state

case 3: S → 1, R → 1 → Q = 0, Q' = 0 → invalid

CHARACTERISTIC EQN: $\boxed{Q(t+1) = S + Q(t)\cdot\overline{R} \quad (S\cdot R = 0)}$

S-R with Enable → It preserves its state when the latch is not enabled.

$\bar{S}$-$\bar{R}$ Latch → Designed with NAND gates. $S \rightarrow$ (Reset), $R \rightarrow$ (Set)

• D Latch

It is designed to avoid undesirable condition of S-R Latch.

$E=1 \rightarrow D=0 \rightarrow S=0,\ R=1 \rightarrow Q(t+1)=0$

$D=1 \rightarrow S=1,\ R=0 \rightarrow Q(t+1)=1$

CHARACTERISTIC EQN: $\boxed{Q(t+1)=D}$

$E=0 \rightarrow D=\emptyset \rightarrow S=0,\ R=0 \rightarrow Q(t+1)=Q(t)$

- D-Latch can be designed like flip-flop. (It has rising edge and falling edge versions.)

T Flip-Flop (Toggle)

. $T=0 \rightarrow Q(t+1)=Q(t)$

$T=1 \rightarrow Q(t+1)=Q'(t)$

CHARACTERISTIC EQN: $\boxed{Q(t+1)=T \oplus Q(t)}$

J-K Flip-Flop

. It is combine of both S-R and T flip-flops.

. $J=1,\ K=0$ } S-R Latch $(S:J,\ R:K)$ $\rightarrow Q(t+1)=1$

$J=0,\ K=1$ } $\rightarrow Q(t+1)=0$

$J=0,\ K=0$ } Toggle FF $\rightarrow Q(t+1)=Q(t)$

$J=1,\ K=1$ } $\rightarrow Q(t+1)=Q'(t)$

CHARACTERISTIC EQN: $\boxed{Q(t+1)=J\cdot\overline{Q(t)}+\bar{K}\cdot Q(t)}$

w9

Mealy: Output depends on both current state and input values.

Moore: Output depends only on current state.

Note: Change in input ; immediately (propagation delay is considered) changes the output in mealy model because output depends on input values.

; doesn't change the output value if the clock signal is not in the setup time.

Analysis of FF

- Determine F functions that enters ff as an input.
- Find the next state exp. using ff char. eqn. and Fs. (H
- Create state table using next state exp.
- Determine the exp. of output.
- Construct state table for G.
- Draw state diagram (optional)

w10

How to design clocked synchronous sequential circuits

- Determine the model (Mealy or Moore)
- Determine the states
  - Determine state transitions based on inputs
  - Construct the state transition and output table.
- There are $\lceil \log_2 m \rceil$ flip-flops in the circuit (m: number of states)
- Determine which ff is used.
- Determine the F function enters in ff's.

Transition Tables

S-R

symbol	S	R
0	0	∅
α	1	0
β	0	1
1	∅	0

J-K

	symbol	J	K
00	0	0	∅
01	α	1	∅
10	β	∅	1
11	1	∅	0

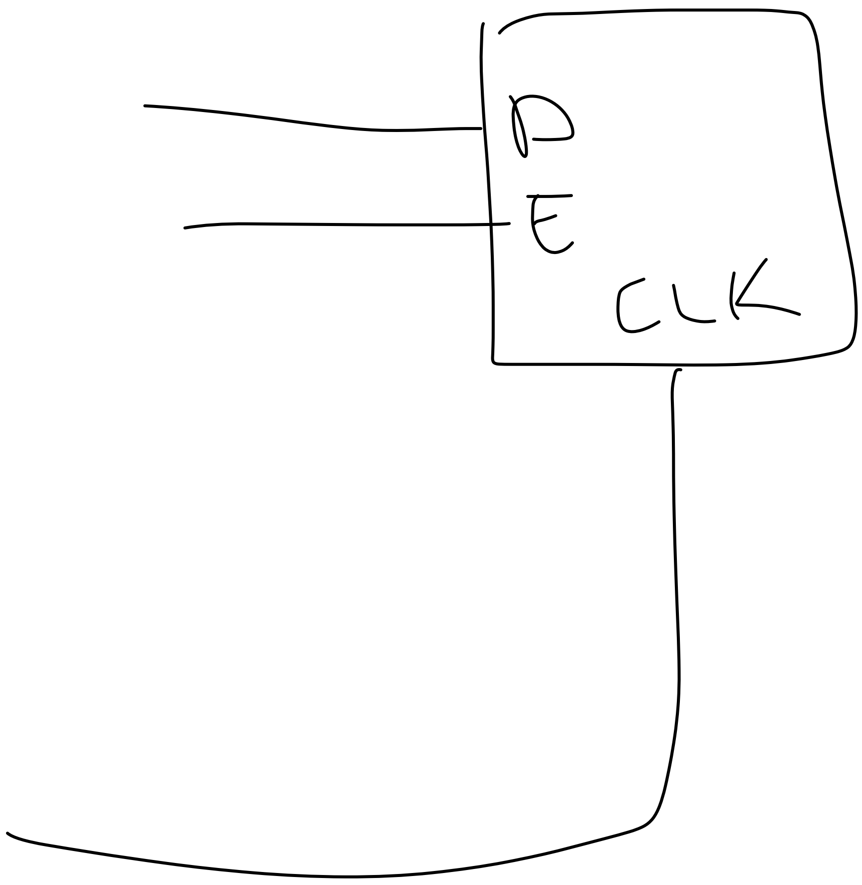
D

symbol	D
0	0
α	1
β	0
1	1

T

symbol	T
0	0
α	1
β	1
1	0

All Circuits


D
E
CLK