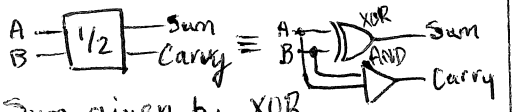
Jacob Hurst
4/7/19

CS 3602 : HW 3

1. Full Adder via 1/2 Adders:

½ Adder :

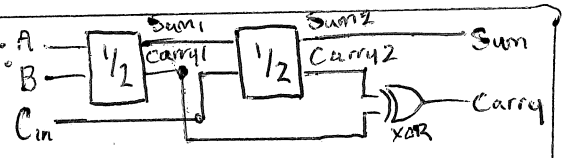
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



Sum given by XOR

Carry given by AND

Common Full Adder :



Truth Table :

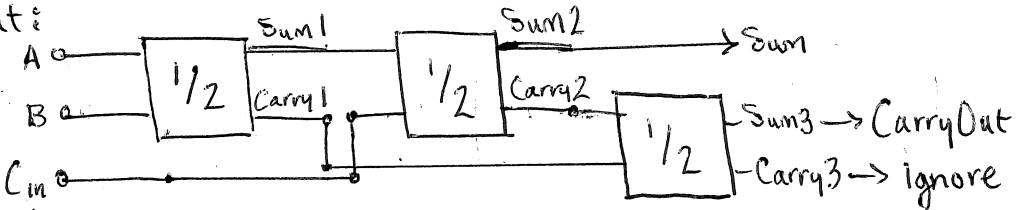
A	B	Cin	Sum1	Carry1	Sum2	Carry2	CarryOut
0	0	0	0	0	0	0	0
0	0	1	0	0	1	0	0
0	1	0	1	0	1	0	0
0	1	1	1	0	0	1	1
1	0	0	1	0	1	0	0
1	0	1	1	0	0	1	1
1	1	0	0	1	0	0	1
1	1	1	0	1	1	0	1

Carry1 XOR Carry2 ≡ Sum from ½ Adder between Carry1 & Carry 2.

⇓

- replace Carry1 XOR Carry2 w/ a ½ adder, keep Sum.

Result:



A	0	0	0	0	1	1	1	1
B	0	0	1	1	0	0	1	1
Cin	0	1	0	1	0	1	0	1
Sum	0	1	1	0	1	0	0	1
CarryOut	0	0	0	1	0	1	1	1

Sum1 : 00111100

Carry1 : 00000001

Carry2 : 00010100

$Sum = (A \oplus B) \oplus C_{in}$, $C_{out} = ((A \oplus B) \wedge C_{in}) \oplus (A \wedge B)$

$= (A\bar{B}C) + (A\bar{B}\bar{C}) + (\bar{A}B\bar{C}) + (\bar{A}\bar{B}C)$.

$= (A \wedge B) \vee (A \wedge C) \vee (B \wedge C)$.

Simplifying the two expressions for Sum & Cout result in a further Complicated circuit which would require more than 2 ½ Adders or an implementation using other gates.

CS362: HW3

2. Nim Neural Net Design (3 pile, sizes 0→63):

Algorithm: i) Compute $Z = A \oplus B \oplus C$ & feed forward A, B, C, Z to level 1.

ii) Compute $A_{new}, B_{new}, C_{new} = A \oplus Z, B \oplus Z, C \oplus Z$ & feed forward A, B, C, $A_{new}, B_{new}, C_{new}$ to level 2.

iii) Compute $Valid_A, Valid_B, Valid_C$ by $Valid_{pile} = \begin{cases} 1 & \text{if } pile - pile_{new} \geq 0 \\ 0 & \text{otherwise} \end{cases}$

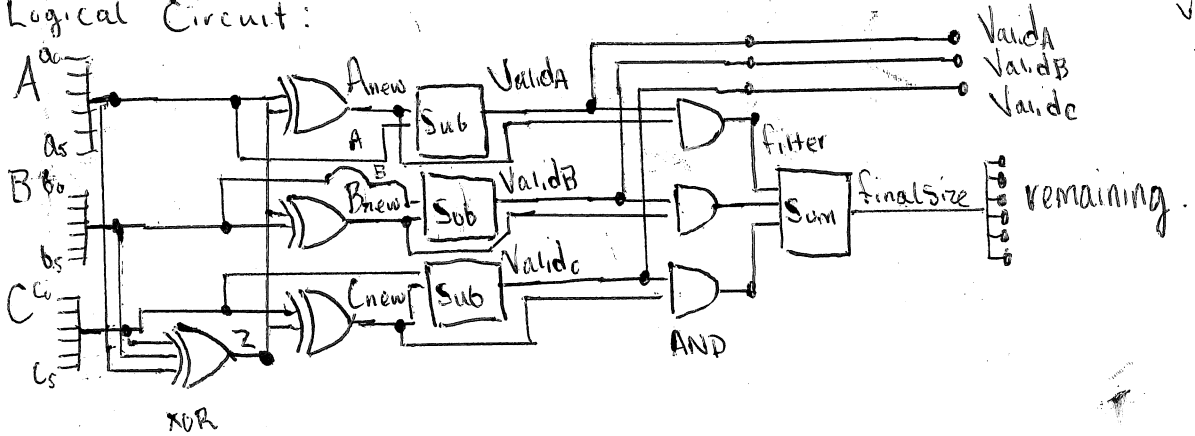
& feed forward $A_{new}, B_{new}, C_{new}, Valid_A, Valid_B, Valid_C$ into level 3.

iv) Filter negative $pile_{new}$ by anding w/ respective valid bits & feed forward filtered $pile_{news}$ (should have 2 piles = 0 now & 1 that is ≥ 0) & $Valid_{pile}$'s into level 4.

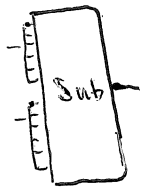
v) Sum the filtered piles to get the remaining pile size for the selected pile. & feed forward the $Valid_{piles}$ & Sum into level 5 (output).

Output is $(Valid_A, Valid_B, Valid_C, Remaining)$ where $Valid_{pile}$ is 1 if selected for that pile & remaining is bitwise (6 bits) in the case where 3 moves are valid, ref takes first move. when 0 moves valid, ref takes1.

Logical Circuit:

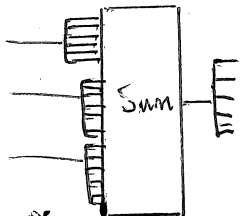


Layer 0 1 2 3 4 5 Output.



input: 2×6 bits

output: 1 bits



input: 3×6 bits

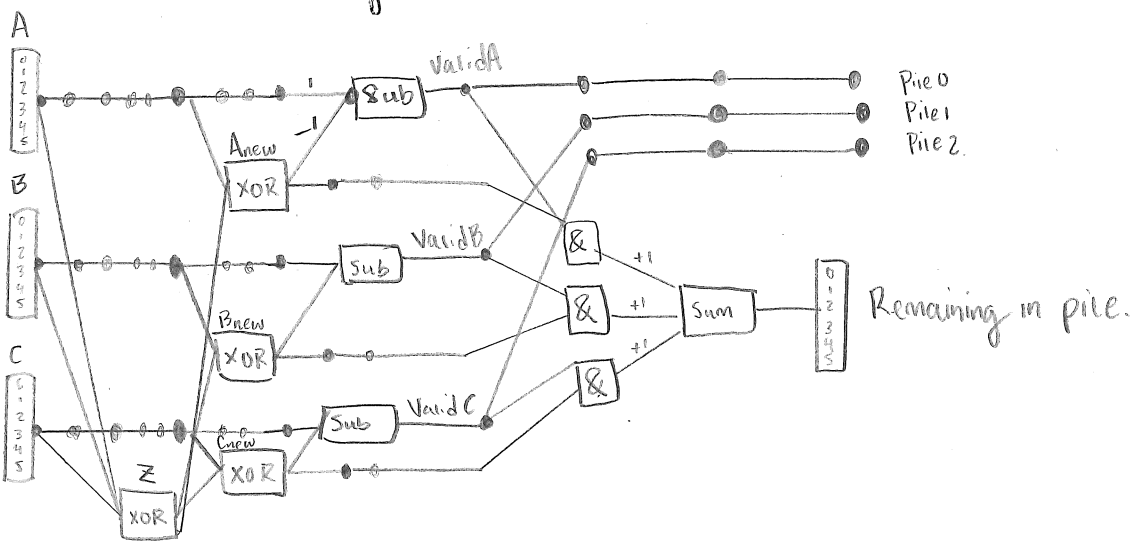
Output 6 bits

(3×6) input XOR for Z, output: 6

(2×6) input XOR for $pile_{new}$, output 6

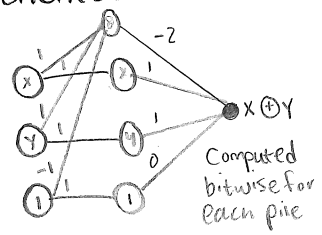
(2×6) input AND for filtering output 6.

Nim Neural Net Design

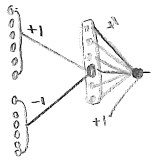


Components:

XOR



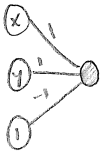
Sub



output is bit indicating if valid.

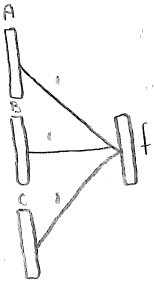
$\begin{cases} 0 & \text{if } A - Anew < 0 \\ 1 & \text{if } A - Anew \geq 0 \end{cases}$

AND



used to "zero out" nonvalid piles.

Sum



add up bits in each final pile, should be 0, or in most cases the single valid pile after subtracting. if all 3 piles are valid, the ref knows to select move of eliminating pile for NN.