SIT111 - Task 3.2C The ALU Truth Table

Overview

The Hack ALU discussed in the unit computes a fixed set of functions out = $f_i(x,y)$; where x and y are the chip's two 16-bit inputs, out is the chip's 16-bit output, and f_i is an arithmetic or logical function selected from a list of eighteen possible functions. We instruct the ALU which function to compute by setting six input bits, called control bits, to selected binary values.

Each one of the six control bits instructs the ALU to carry out a certain elementary operation. Taken together, the combined effects of these operations cause the ALU to compute a variety of useful functions. Since the overall operation is driven by six control bits, the ALU can potentially compute $2^6 = 64$ different functions. Eighteen of these functions are documented in Listing 1 and Figure 1 given below at the end of this task sheet.

In this task, you are required to work out how the ALU computes a given function, in the tables a-e.

Task requirements

a. Go through week 3 class materials on Google Classroom & complete the practice problems in week 3

b. Task 3.1 P

c. Read the task instructions

Task Instructions

1. Fill in the tables a-e below by procedurally working through the implementations of each function in the ALU. The problems are applied to 4-bit numbers for simplicity. Write the decimal equivalent of the arbitrarily provided binary inputs and outputs. Show all steps. You may manipulate the tables by adding, deleting, merging cells etc as you see fit.

   The first one is done for you as an example.

OUT	x	y
f(x,y) = 0 **0000**	1010 **-6**	0001 **1**
set x to zero	**0000**	
don't negate x	**0000**	
set y to zero		**0000**
don't negate y		**0000**
add x and y	**0000**	
don't negate result	**0000**	

a.

OUT	x	y
f(x,y) = -1	0001	0011

b.

OUT	x	y
f(x,y) = !x	1010	0011

c.

OUT	x	y
f(x,y) = x+1	1011	0001

d.

OUT	x	y
f(x,y) = x+y	1011	0101

e.

OUT	x	y
f(x,y) = x&y	1011	1001

Note: The truth table and the pseudo code of the ALU (taken from *The Elements of Computing Systems : Building a Modern Computer from First Principles)* are given in Listing 1 & Figure 2 for your reference.

Listing 1: Pseudo code for the ALU

```
Chip name: ALU
Inputs: x[16], y[16], // Two 16-bit data inputs
        zx, // Zero the x input
        nx, // Negate the x input
        zy, // Zero the y input
        ny, // Negate the y input
        f, // Function code: 1 for Add, 0 for And
        no // Negate the out output
Outputs: out[16], // 16-bit output
        zr, // True iff out=0
        ng // True iff out<0
Function: if zx then x = 0 // 16-bit zero constant
        if nx then x = !x // Bit-wise negation
        if zy then y = 0 // 16-bit zero constant
        if ny then y = !y // Bit-wise negation
        if f then out = x + y // Integer 2's complement addition
        else out = x & y // Bit-wise And
        if no then out = !out // Bit-wise negation
        if out=0 then zr = 1 else zr = 0 // 16-bit eq. comparison
        if out<0 then ng = 1 else ng = 0 // 16-bit neg. comparison
Comment: Overflow is neither detected nor handled.
```

Reference

Nisan, Noam, and Shimon Schocken. *The Elements of Computing Systems : Building a Modern Computer from First Principles* MIT Press, 2005

http://nand2tetris-questions-and-answers forum.32033.n3.nabble.com/le/n95834/alu worksheet.pdf

These bits instruct how to preset the x input		These bits instruct how to preset the y input		This bit selects between +/And	This bit inst. how to postset out	Resulting ALU output
zx	nx	zy	ny	f	no	out=
if zx then x=0	if nx then x=!x	if zy then y=0	if ny then y=!y	if f then out=x+y else out=x&y	if no then out=!out	f(x,y)=
1	0	1	0	1	0	0
1	1	1	1	1	1	1
1	1	1	0	1	0	−1
0	0	1	1	0	0	x
1	1	0	0	0	0	y
0	0	1	1	0	1	!x
1	1	0	0	0	1	!y
0	0	1	1	1	1	−x
1	1	0	0	1	1	−y
0	1	1	1	1	1	x+1
1	1	0	1	1	1	y+1
0	0	1	1	1	0	x−1
1	1	0	0	1	0	y−1
0	0	0	0	1	0	x+y
0	1	0	0	1	1	x−y
0	0	0	1	1	1	y−x
0	0	0	0	0	0	x&y
0	1	0	1	0	1	x|y

Figure 1: The ALU Truth Table