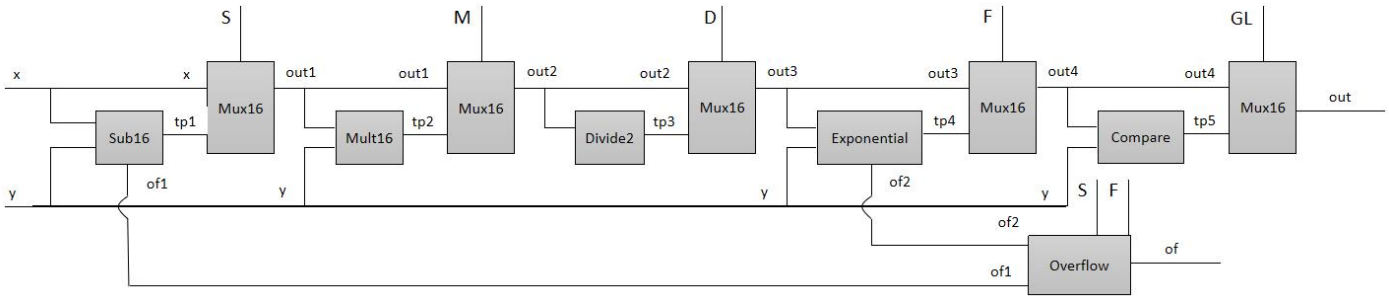
Structure:



Instruction:

The chip input is two 16-bits value and output is one 16-bits value.

Following is how to use different gate to accomplishment different function.

1. **Mention: the below operation execute correctly only if the specific switch is true(open).**

   S: Subtract y from x.

   M: Multiply x by y. **[note, both x and y can only be between -100 and +100]**

   D: Divide x by 2. **[note, only for positive values of x and fractional results should be rounded down]**

   F: Calculate an exponential expression. **[note, x or y cannot be negative, maximum values is x = 9 and y = 5]**

   GL: Decide if x is bigger than y, output = 1 if true, -1 in binary if false.

   of: if a number is greater than 32767 (overflow) of is true; otherwise not.

2. mention:the below operation execute correctly when many switches are true(open) at the same time.

   S, D: Divide (the result of subtract y from x) by 2. (x – y) / 2

   [note, only for x greater than y and fractional result should be rounded down]

   M, D: Divide (the result of the multiply x by y) by 2. x * y / 2

   [note, only for x and y both positive or negative and fractional result should be round down]

   [Also, both x and y can only be between -100 and +100]

   D, F: Calculate (divide x by 2) ^y. (x / 2) ^ y

   [note, x or y cannot be negative, maximum values of x= 19 and y = 5][if overflow it will be mentioned]

   [note, if the result of divide x by 2 is fractional result, it will be rounded down]

   S, M: x * y – y * y

   [note, both x and y can only be between -100 and +100]

3. Also, other way of switch can do something but not efficient and useful(one reason is the range of x and y is very small).

Efficiency:

1. In multiply chip, there are two ways to accomplish it: according to the different value of each bits of the second input to add the value several times, eventually, add them together to get the answer.(for instance, when see the second bits in the second input, the value need to add to itself, because when the second bits is 1, that mean in the decimal it is 2) or normal left shift and add together at last.

I choose the second way because it may **call less chip**, it do not need to call the add16 chip so many times and therefore reduce the times to call nand.

2. Inside the exponential and sub chip add two extra out (of1 and of2) which can **be directly used in judge overflow**, it avoid to implement complex judgement in the overflow chip.

3. For the whole structure I use x to be the main line and each out combine with y, I think it can also improve the utilization, **no more time to use x**, also, **more calculation** can be accomplished in this way.