Double-Operand Math Instructions Examples

```
Line nr.  Assembly                      Machine Code
                                        aoooooddddiiiijjjjxxxxxxxxxxxxxx
1.        FADD r1, r2, r3               00010100000001000110000000000000
2.        FSUB r1, r2, #5               10011000000010000000000000000101
```

From left to right, bit 'a' defines the addressing mode. In line 1, the second source `r3` is a register. For register addressing a=0. The next five 'o' bits are the FADD opcode.

Bits 'dddd', 'iiii', 'jjjj', are the addresses for the `r1`, `r2`, and `r3` registers respectively. The rest are don't cares in this case, so I just made them zero.

In line 2, the `#` symbol indicates to the assembler that immediate addressing should be used instead, so the 'a' bit is set to 1. Also, the encoding format therefore changes, the last 18 bits after when `r1` and `r2` are encoded are used to represent the 18 bit binary value of 5.

On the tables below, line 1 of the above code corresponds to an r-type Double-Operand ALU instruction, and line 2 corresponds to an i-type instruction.

			30	25	21	17	13 0
Double-Operand ALU	r-type:	0	opcode	rd	rs1	rs2	0
	i-type:	1	opcode	rd	rs	18-bit immediate	
Single-Operand ALU	EXP:	0	opcode	rd	22-bit immediate		
		0	opcode	rd	0		
Load, Store, Move		0	opcode	rd	rs	0	
Branching	B:	0	opcode	rd	22-bit immediate		
		0	opcode	rd	0		
No-Operation and Halt		0	opcode	0			