CompOrg HW2:

3.13. 62×12 ⟶ 0011 1110 × 1100

Iter	Step	Multiplier	Multiplicand	Product
0	Initial values	1100	0011 1110	0000 0000
1	1a: 0 ⇒ no operation	1100	0011 1110	0000 0000
	2: sl multpcand	1100	0111 1100	0000 0000
	3: sr multiplier	0110	0111 1100	0000 0000
2	1a: 0 ⇒ no op.	0110	0111 1100	0000 0000
	2: "	0110	1111 1000	0000 0000
	3: "	0011	1111 1000	0000 0000
3	1a: 1 ⇒ prod += mcand	0011	1111 1000	1111 1000
	2: "	0011	1111 0000	1111 1000
	3: "	0001	1111 0000	1111 1000
4.	1a: 1 ⇒ prod += mcand	0001	1111 0000	1111 1000
	2: "	0001	1110 0000	1111 1000
	3: "	0000	1110 0000	1111 1000

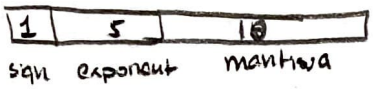
Final product → 1111 1000 = 744_{10}

3.18. $74 \div 21 \longrightarrow 01001010 \div 00010101$ (8-bit)

iter	Step	Quotient	Divisor	Remainder
0	initial values	000000	01010100	01001010
1	1: rem = rem - div.	000000	01010100	(1)1110110
	2b: rem<0 => +Div, sll Q, Q0=0	000000	01010100	01001010
	3: Shift div right	000000	00101010	01001010
2	1: rem = rem - div	000000	00101010	00100000
	2b: rem ≥ 0 => sll Q, Q0=1	000001	00101010	00100000
	3: shift div right	000001	00010101	00100000
3	1: rem = rem - div	000001	00010101	00001011
	2b: rem ≥ 0 => sll Q, Q0=1	000011	00010101	00001011
	3: shift div right	000011	00001010	00001011
4	1: rem = rem - div	000011	00001010	00000001
	2b: rem ≥ 0 => sll Q, Q0=1	000111	00001010	00000001
	3: shift div right	000111	00000101	00000001
5	1: rem = rem - div	000111	00000101	11111100
	2b: rem<0 => +Div, Q sll, Q0=0	001110	00000101	00000001
	3: shift div right	001110	00000010	00000001
6	...			

but we must end at iter. 3 because we have correct values for Q = 3 and remainder = 11 then.

3.27. half precision of 16-bits



bias = 15 → $(-1)^n \; 2^{\text{exp-bias}} \; (1 + \text{mantissa})$

-1.5625×10^{-1} → n = 1,

1.5625×10^{-1} => denormalized:

0.15625	×2	0.3125	0	⇒ 0.00101
0.3125	×2	0.625	0	
0.625	×2	1.25	1	=> normalized
0.25	×2	0.5	0	1.01×2^{-3}
0.5	×2	1.0	1	

→ $(-1)^S \times 2^{\text{exp-bias}} \times (1+\text{mant}) = (-1) \times 2^{-3} \times (1.01)$

so, S = 1, exp-bias = -3, mantissa = 0100000000

exp - 15 = -3,

exp = 12 → 1100

→ | 1 | 01100 | 0100000000 |, the range is smaller than of single precision bcs numbers through 2^{-14} to 2^{15}, whereas single precision has 2^{-126} to 2^{127} range.

Accuracy is 10 bits while single prec. has 23 bits.

3.29. sum of floating point numbers

A = 26.125 → 11010 + 0.125 × 2 = 0.250 0 → 11010.001_2 ⎫ exp=4 +bias = 19

0.125 × 2 = 0.5 0 ⎪ 19 → 10011_2

0.5 × 2 = 1.0 1 → $1.1010001_2 \times 2^4$ ⎭ mant → 1010001...

A = 0 10011 1010001000

B: 0.4150390625 → 0.0110101001 using above method → 1.10101001×2^{-2}

→ exp = -2 +bias = 13 → 01101_2

B = 0 01101 1010100100

---

19 - 13 = 6 bits alignment needed → align to higher num. → shift B's mant. right by 6

B => 0.0000011010 1 → with 3 GRS bits → 0.0000011010 | 101

Add mantissas → 1.1010001000
+ 0.0000011010
= 1.1010100010 1000

Hence:

0 10011 1010100010 □