605.611 - Foundations of Computer Architecture

Assignment 04

Sabbir Ahmed

February 22, 2021

1. Implement the table to show the steps in the multiplication using Figure 3.7. Use 8 bit numbers.

   (a) 3 * 5

Iteration	**Multiplicand**	**Product**	**Steps**
0	0000 0011	0000 0000 0000 0101	Start
1	0000 0011	0000 0001 1000 0010	LSB = 1 Add Shift right
2	0000 0011	0000 0000 1100 0001	LSB = 0 Shift right
3	0000 0011	0000 0001 1110 0000	LSB = 1 Add Shift right
4	0000 0011	0000 0000 1111 0000	LSB = 0 Shift right
5	0000 0011	0000 0000 0111 1000	LSB = 0 Shift right
6	0000 0011	0000 0000 0011 1100	LSB = 0 Shift right
7	0000 0011	0000 0000 0001 1110	LSB = 0 Shift right
8	0000 0011	0000 0000 0000 1111	LSB = 0 Shift right

(b) 4 * 6

Iteration	**Multiplicand**	**Product**	**Steps**
0	0000 0100	0000 0000 0000 0110	Start
1	0000 0100	0000 0000 0000 0011	LSB = 0 Shift right
2	0000 0100	0000 0010 0000 0001	LSB = 1 Add Shift right
3	0000 0100	0000 0011 0000 0000	LSB = 1 Add Shift right
4	0000 0100	0000 0001 1000 0000	LSB = 0 Shift right
5	0000 0100	0000 0000 1100 0000	LSB = 0 Shift right
6	0000 0100	0000 0000 0110 0000	LSB = 0 Shift right
7	0000 0100	0000 0000 0011 0000	LSB = 0 Shift right
8	0000 0100	0000 0000 0001 1000	LSB = 0 Shift right

2. Implement the table to show the steps in division using Figure 3.13

   (a) 15 / 6

Iteration	**Divisor**	**Complement**	**Remainder**	**Steps**
0	0000 0110	1111 1010	0000 0000 0000 1111	Start
1a	0000 0110	1111 1010	0000 0000 0001 1110	Shift left
1b	0000 0110	1111 1010	1111 1010 0001 1110	Add complement
1c	0000 0110	1111 1010	0000 0000 0001 1110	Add divisor
2a	0000 0110	1111 1010	0000 0000 0011 1100	Shift left
2b	0000 0110	1111 1010	1111 1010 0011 1100	Add complement
2c	0000 0110	1111 1010	0000 0000 0011 1100	Add divisor
3a	0000 0110	1111 1010	0000 0000 0111 1000	Shift left
3b	0000 0110	1111 1010	1111 1010 0111 1000	Add complement
3c	0000 0110	1111 1010	0000 0000 0111 1000	Add divisor
4a	0000 0110	1111 1010	0000 0000 1111 0000	Shift left
4b	0000 0110	1111 1010	1111 1010 1111 0000	Add complement
4c	0000 0110	1111 1010	0000 0000 1111 0000	Add divisor
5a	0000 0110	1111 1010	0000 0001 1110 0000	Shift left
5b	0000 0110	1111 1010	1111 1011 1110 0000	Add complement
5c	0000 0110	1111 1010	0000 0001 1110 0000	Add divisor
6a	0000 0110	1111 1010	0000 0011 1100 0000	Shift left
6b	0000 0110	1111 1010	1111 1101 1100 0000	Add complement
6c	0000 0110	1111 1010	0000 0011 1100 0000	Add divisor
7a	0000 0110	1111 1010	0000 0111 1000 0000	Shift left
7b	0000 0110	1111 1010	0000 0001 1000 0000	Add complement
7c	0000 0110	1111 1010	0000 0001 1000 0001	Invert LSB
8a	0000 0110	1111 1010	0000 0011 0000 0010	Shift left
8b	0000 0110	1111 1010	1111 1101 0000 0010	Add complement
8c	0000 0110	1111 1010	0000 0011 0000 0010	Add divisor

(b) 13 / 3

Iteration	Divisor	Complement	Remainder	Steps
0	0000 0011	1111 1101	0000 0000 0000 1101	Start
1a	0000 0011	1111 1101	0000 0000 0001 1010	Shift left
1b	0000 0011	1111 1101	1111 1101 0001 1010	Add complement
1c	0000 0011	1111 1101	0000 0000 0001 1010	Add divisor
2a	0000 0011	1111 1101	0000 0000 0011 0100	Shift left
2b	0000 0011	1111 1101	1111 1101 0011 0100	Add complement
2c	0000 0011	1111 1101	0000 0000 0011 0100	Add divisor
3a	0000 0011	1111 1101	0000 0000 0110 1000	Shift left
3b	0000 0011	1111 1101	1111 1101 0110 1000	Add complement
3c	0000 0011	1111 1101	0000 0000 0110 1000	Add divisor
4a	0000 0011	1111 1101	0000 0000 1101 0000	Shift left
4b	0000 0011	1111 1101	1111 1101 1101 0000	Add complement
4c	0000 0011	1111 1101	0000 0000 1101 0000	Add divisor
5a	0000 0011	1111 1101	0000 0001 1010 0000	Shift left
5b	0000 0011	1111 1101	1111 1110 1010 0000	Add complement
5c	0000 0011	1111 1101	0000 0001 1010 0000	Add divisor
6a	0000 0011	1111 1101	0000 0011 0100 0000	Shift left
6b	0000 0011	1111 1101	0000 0000 0100 0000	Add complement
6c	0000 0011	1111 1101	0000 0000 0100 0001	Invert LSB
7a	0000 0011	1111 1101	0000 0000 1000 0010	Shift left
7b	0000 0110	1111 1010	1111 1101 1000 0010	Add complement
7c	0000 0110	1111 1010	0000 0000 1000 0010	Add divisor
8a	0000 0110	1111 1010	0000 0001 0000 0100	Shift left
8b	0000 0110	1111 1010	1111 1101 0000 0100	Add complement
8c	0000 0110	1111 1010	0000 0001 0000 0100	Add divisor

3. Implement the hardware multiplication and division algorithms in software. You can use addition and subtraction and any bit-wise operations you need. You can do this in any language you choose.

   See the attached Python file.