1. Slt instruction in system is Slt value, a, b.
It will set value to 1 if a<b, otherwise 0.
For comparing if a is less than b, computer more prefer using Subtraction: a-b. If a-b<0 that means a<b.
So in the "32-bit Alu", only the very first bit will be set to compare a and b, for the rest will be set to false.
So in the very first bit, if a<b (a-b<0) then less returns true (1) otherwise less returns false (0). Therefore, the result in the "32-bit Alu" will be either "0000....00" or "0000....01"

"0000....00" — very first bit (1 bit) + 32 bits

"0000....01" — very first bit (1 bit) + 32 bits

For the control signal, because Alu wants to calculate the Subtraction. So binvert is setted to 1. Therefore "01".
For the output of the result, it will be setted to less which is "11". Therefore, ainvert, binvert: "01", operation "11".
The signal values for Slt instruction are "0111".

2. 0101 * 1110 = 0100 0110

MD	AC	MQ
0101	0000	1110
	0000	1110
	0000	0111
	0101	
	0101	0111
	0010	1011
	0101	
	0111	1011
	0011	1101
	0101	
	1000	1101
	0100	0110

3. (a) $\underbrace{1}_{-}\underbrace{0}_{+}\underbrace{00}_{0}\underbrace{111}_{7}\underbrace{0}_{0}\underbrace{1}_{-1}\underbrace{0}_{0} = 413_{10}$

$1\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 0$

$256+128+32+(-4)+2+(-1) = 413_{10}$

(b) $[1010 * 0101] = 1110\ 0010$

MD	AC	MQ	Q-1
1010	0000	0101	0
0101	+ 0110		
+ 1	0110	0101	
0110	0011	0010	1
2's complement	+ 1010		
	1101	0010	
	1110	1001	0
	+ 0110		
	0100	1001	
	0010	0100	1
	1010		
	1100	0100	1
	1110	0010	0

Product: 1110 0010

Q R

11/4 = 2 3

4. 1011 / 0100 = R: 0011, Q: 0010

MD	AC	MQ
0100	0000	1011
1	0001	011□
is 1100	+1100	
	1101 <0	011□
	1101	011 0
	+0100	
	0001	0110
	0010	110□
	1100	
	1110 <0	110□
	1110	110 0
	+0100	
	0010	1100
	0101	100□
	+1100	
	0001 >0	100□
	0001	1001
	0011	001□
	+1100	
	1111 <0	001□
	1111 <0	0010
	+0100	
	0011	0010
	R	Q

11 4 2 3

Q R

5. 1011 / 0100 = 0010 0011

MD	AC	MQ
0100	0000	1011
/	0001 >0	0110
1100 2s	+1100	
	1101	0110
	1101	0110
	1010 <0	1100
	+0100	1100
	1110	1000
	1101 <0	1000
	+0100	
	0001 >0	1001
	0011 >0	0010
	+1100	
	1111 <0	0010
	+0100	
	0011	0010
	R	Q

6. The reason why non-restoring division algorithm achieves higher efficiency is because the non-restoring division only executes if (AC<0) then AC ← AC+MD on it's last step while restoring division execute that on every steps of it's caculation.