```
pub fn shift_rows(state: &mut [[u8; 4]; 4]) {
	let mut row = [0u8; 4];
	for i in 1..4 {
		for j in 0..4 {
			row[j] = state[i][(j + i) % 4];
		}
		for j in 0..4 {
			state[i][j] = row[j];
		}
	}
}

pub fn mix_columns(state: &mut [[u8; 4]; 4]) {
	let mut column = [0u8; 4];
	for c in 0..4 {
		for r in 0..4 {
			column[r] = xtime(state[r][c])
			^ state[(r + 1) % 4][c] ^ xtime(state[(r + 1) % 4][c])
			^ state[(r + 2) % 4][c]
			^ state[(r + 3) % 4][c];
		}
		for r in 0..4 {
			state[r][c] = column[r]
		}
	}
}

fn ff_mult(mut a: u8, mut b: u8) -> u8 {
	let mut sum: u8 = 0;
	for i in 0..8 {
		if b & 1 != 0 {
			sum ^= a;
		}
		b >>= 1;
		a = xtime(a);
	}
	sum
}

fn xtime(a: u8) -> u8 {
	if a & 0x80 != 0 {
		(a << 1) ^ 0x1b
	} else {
		a << 1
	}
}
```