A. We can see that result must be in register `%edi`, since this value gets copied to `%eax` at the end of the function as the return value (line 13). We can see that `%esi` and `%ebx` get loaded with the values of `x` and `n` (lines 1 and 2), leaving `%edx` as the one holding variable `mask` (line 4.)
B. Register `%edi(result)` is initialized to −1 and `%edx(mask)` to 1.
C. The condition for continuing the loop (line 12) is that mask is nonzero.
D. The shift instruction on line 10 updates `mask` to be `mask<<n`.
E. Lines 6–8 update `result` to be `result^(x&mask)`.
F. Here is the original code:

```
1 int loop(int x, int n)
2 {
3     int result = -1;
4     int mask;
5     for (mask = 0x1; mask != 0; mask = mask << n) {
6           result ^= (x & mask);
7     }
8     return result;
9 }
```