```cpp

#include <thread>
#include "main.hpp"

FloorLights::FloorLights()
: X20{ 0 } {
    mThread = std::thread{&start, this}; // invokes the callable
}

FloorLights::~FloorLights() {
    mThread.join();
}

void FloorLights::start() {
    while (!gStop) { // So while the global variable hasnt told us to stop
        unsigned char newValue = 0;

        /// get the state from the elevator
        ElevState eleState = gLift.mState;

        /// get current floor from the elevator
        FloorNum eleFloor = gLift.mFloor;

        switch (eleState) {
            case ES_WAIT: continue; // if we are waiting then dont shine any lights

            case ES_UP: {
                int shift = static_cast<int>(eleFloor);
                newValue |= (0x80 >> shift);
            } break;

            case ES_DOWN: {
                int shift = static_cast<int>(eleFloor);
                newValue |= (0x01 << (3 - shift));
            } break;

            default: break;
        }

        X20 = newValue;

        std::this_thread::sleep_for(5s); // SOOO
        // My testing computer has 2 threads -- hyperthreaded
        // thos thread is the least significant so Im adding this
        // in a higher core machine, remove this
    }
}

unsigned char FloorLights::getLights() {
    return X20;
}
```