Clock

The Clock uses the Lego MindStorms Robotics Invention System to keep accurate time on a conventional analog clock face.

More photos of the Clock:




The clock uses a fiber-optic component as a rotation sensor, to step the motor by a fixed increment of 1/16 of a turn. This is geared down by a factor of three to drive the minute hand, and by a further factor of twelve to drive the hour hand. The hands are mounted coaxially, with the hour hand driven directly by the main axle. The minute hand is attached to a gear which is free to spin independently of the main axle, driven by a neighbouring gear on a seperate axle.

The simple gearing combined with the resolution of the rotation sensor results in the minute hand requiring 3 x 16 = 48 steps per hour to keep correct time. This is accomplished by stepping on four minutes out of every five. A simple NQC program monitors the RCX's internal clock and steps the motor on minutes that are not multiples of five. Each step toggles the state of the light/rotation sensor.

The accuracy of the clock is derived from that of the RCX's oscillator. My RCX seems to gain about a second each hour, or roughly twelve minutes each month. Compensating for this drift in software is left as an exercise for the reader. :-)

Building instructions

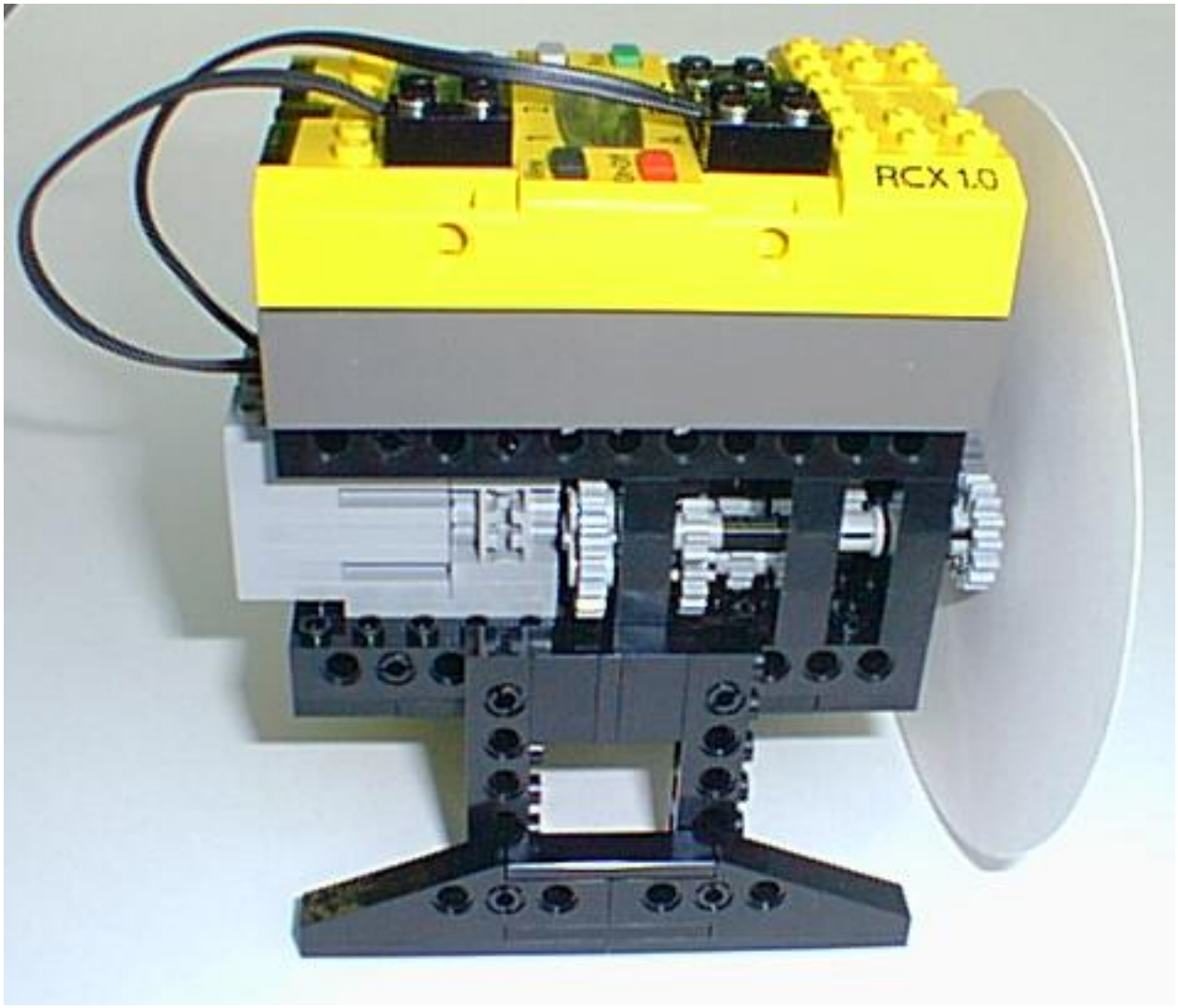
Building instructions for the clock are available here as a series of 35 images in five parts.

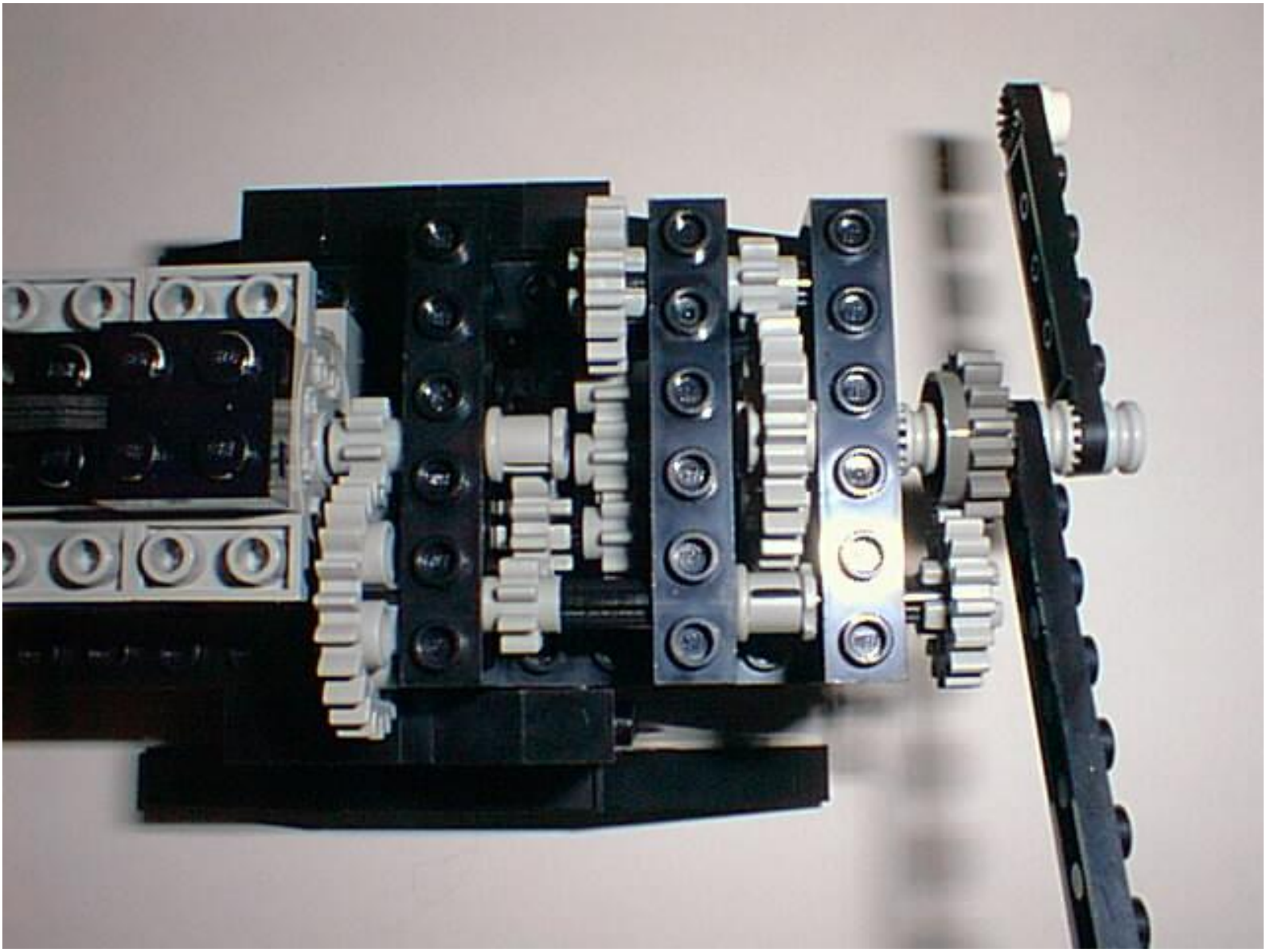
The Clock files:

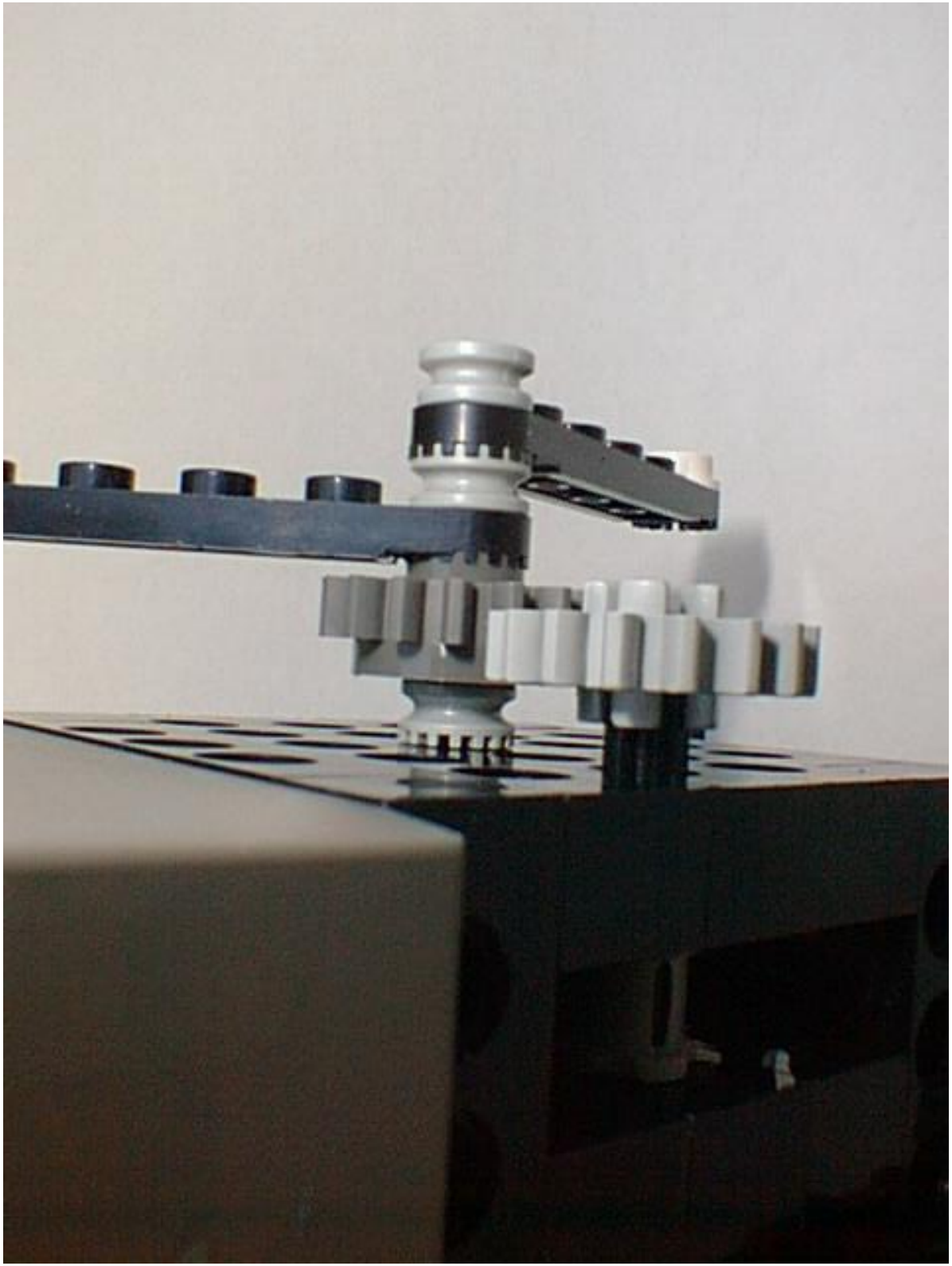
- clock.nqc source, for use with NQC

Up


NETWINDER Z75
SONY


RCX 1.0





Clock

Building Instructions

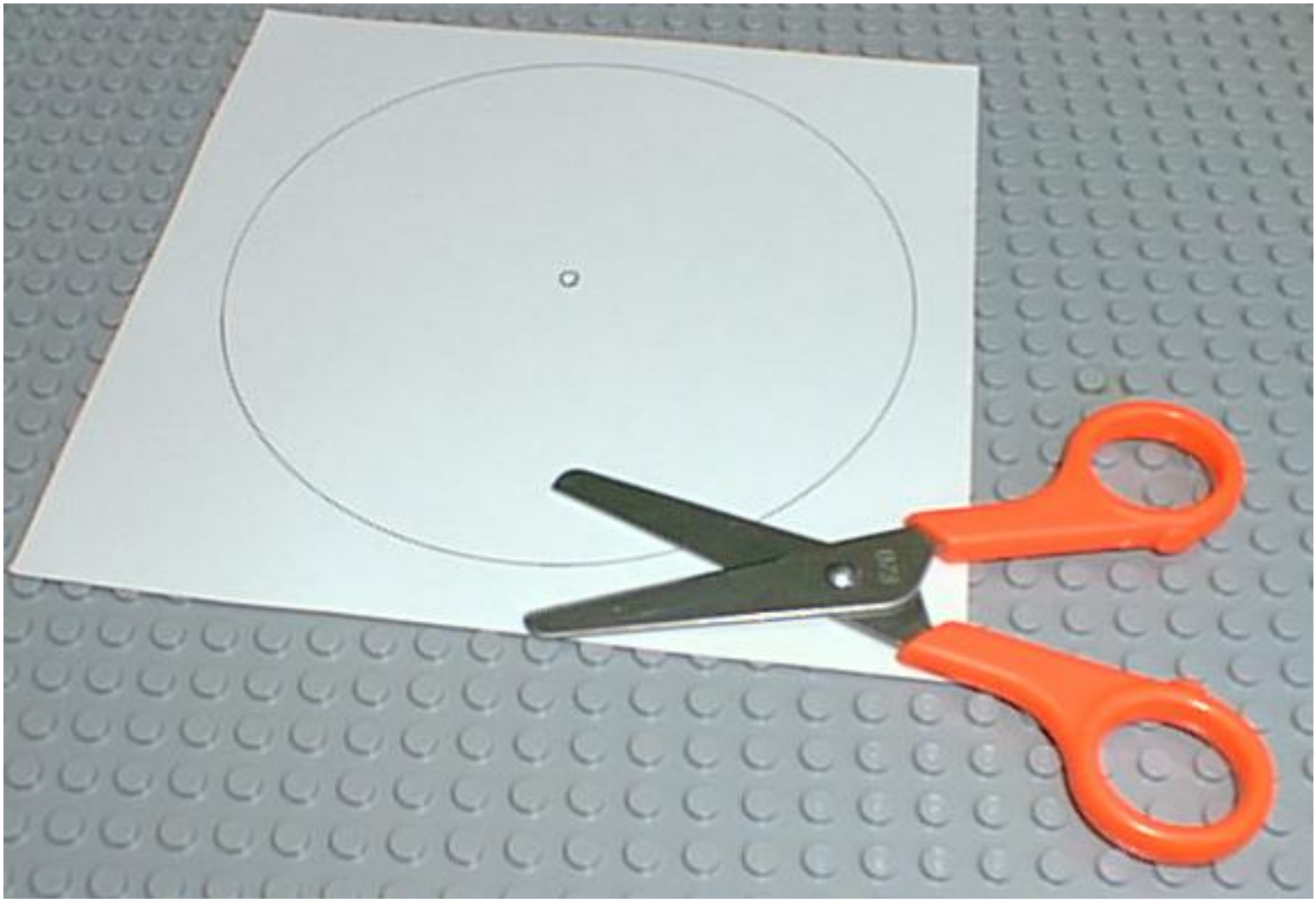
- Part 1 - Cut a face from poster board
- Part 2 - Build the base
- Part 3 - Assemble the gearing
- Part 4 - Add the drive and some legs
- Part 5 - Wire up the RCX

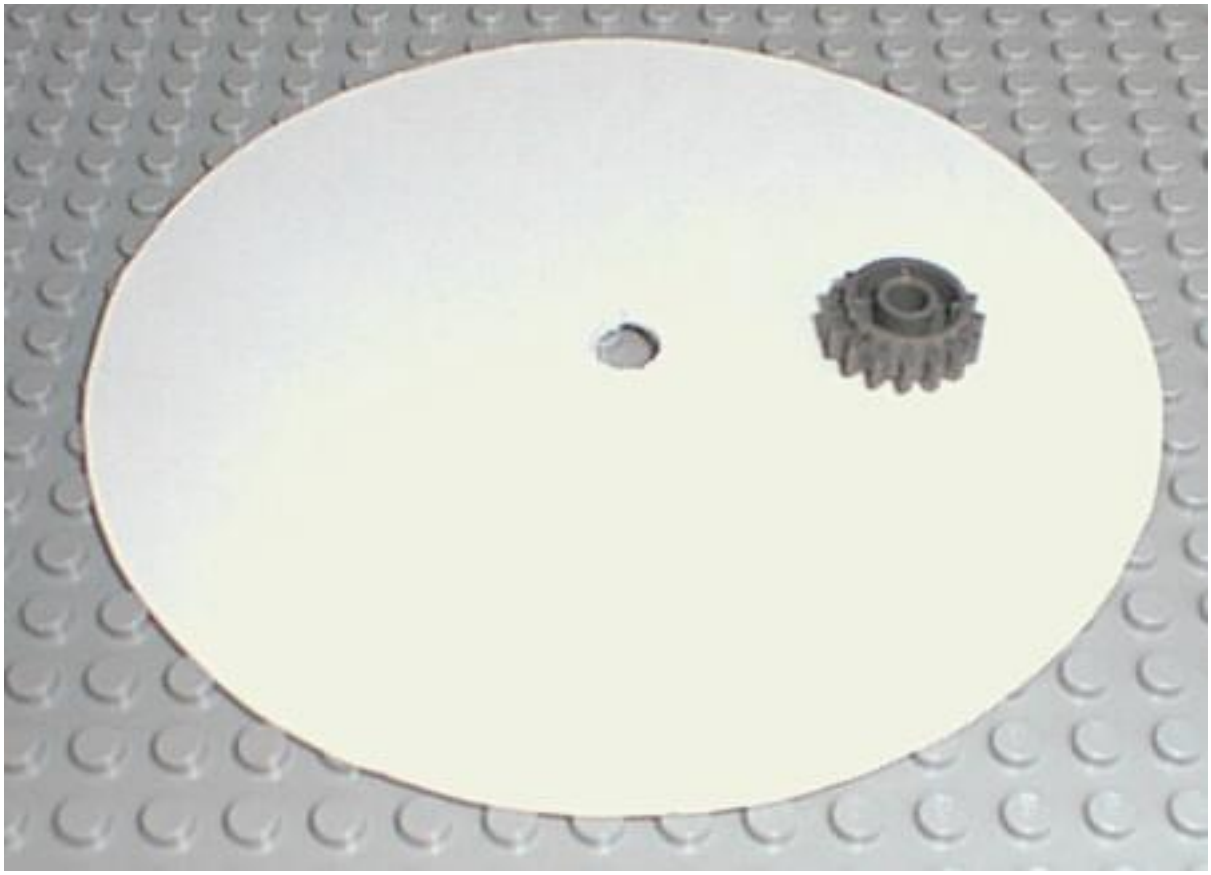
Up

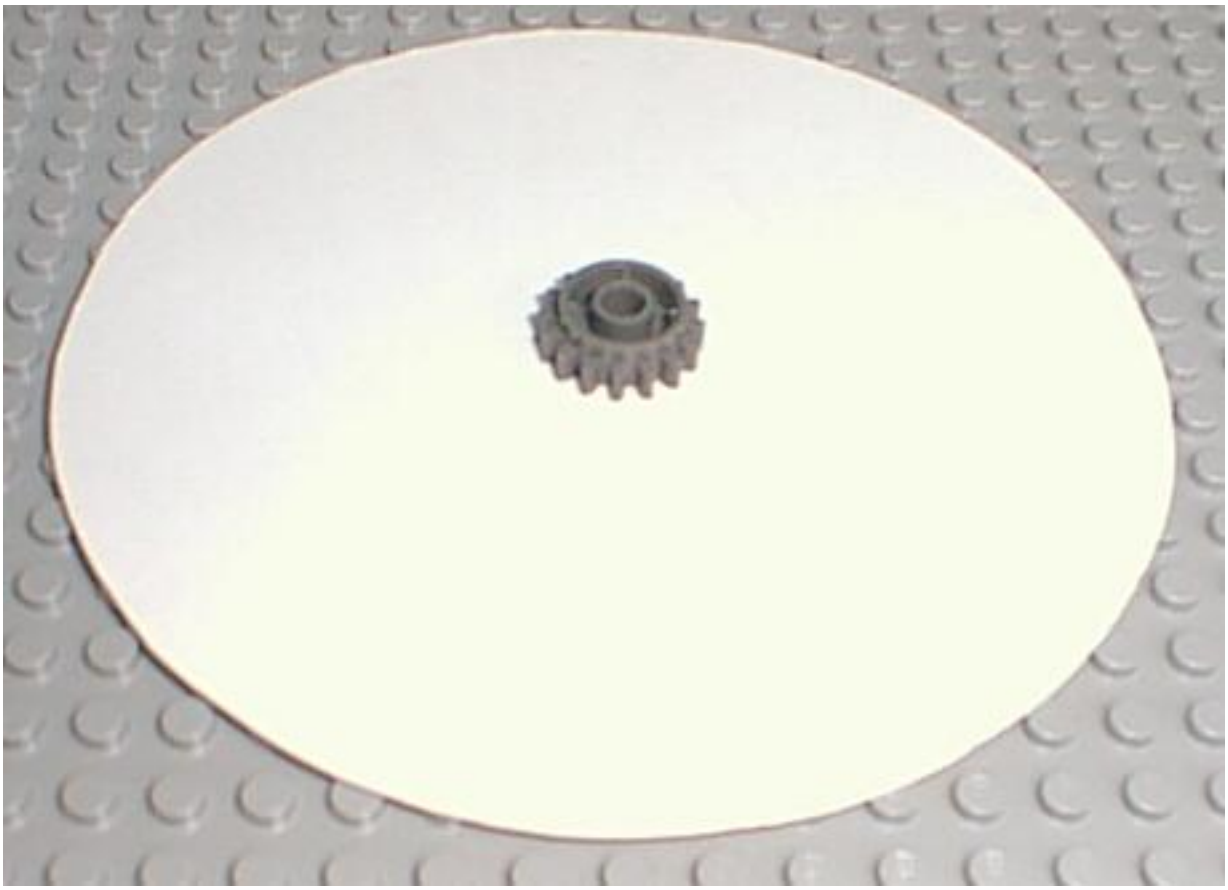
```
#define LIGHT           IN_1
#define MOTOR           OUT_A

#define THRESHOLD       50

#define PowerDownDelay(minutes)         asm { 0xb1, (minutes) }

int curr_watch;
int old_watch;
int tmp;
int state;

sub step
{
        if (state == 0) {
                if (LIGHT < THRESHOLD) {
                        Rev(MOTOR, 1);
                        while (LIGHT < THRESHOLD);
                        Off(MOTOR);
                }
                state = 1;
        } else {
                if (LIGHT > THRESHOLD) {
                        Rev(MOTOR, 1);
                        while (LIGHT > THRESHOLD);
                        Off(MOTOR);
                }
                state = 0;
        }
}

task main
{
        Sensor(LIGHT, IN_LIGHT);

        /* Uncomment the next line to prevent the RCX from
           falling asleep after the default 15 minutes.
           Supplying power from an AC adapter is recommended. */
        /* PowerDownDelay(0); */
        Display(0);
        old_watch = Watch();
        state = 0;

        while(1 == 1) {
                curr_watch = Watch();
                if( curr_watch != old_watch ) {
                        old_watch = curr_watch;
                        /* We only want to step four times each five
                           minutes, for 48 steps per hour.  Don't
                           step on minutes that are multiples of 5. */
                        tmp = (curr_watch / 5) * 5;
                        if (tmp != curr_watch) {
                                step();
                        }
                }
        }
}
```

Clock

Building Instructions







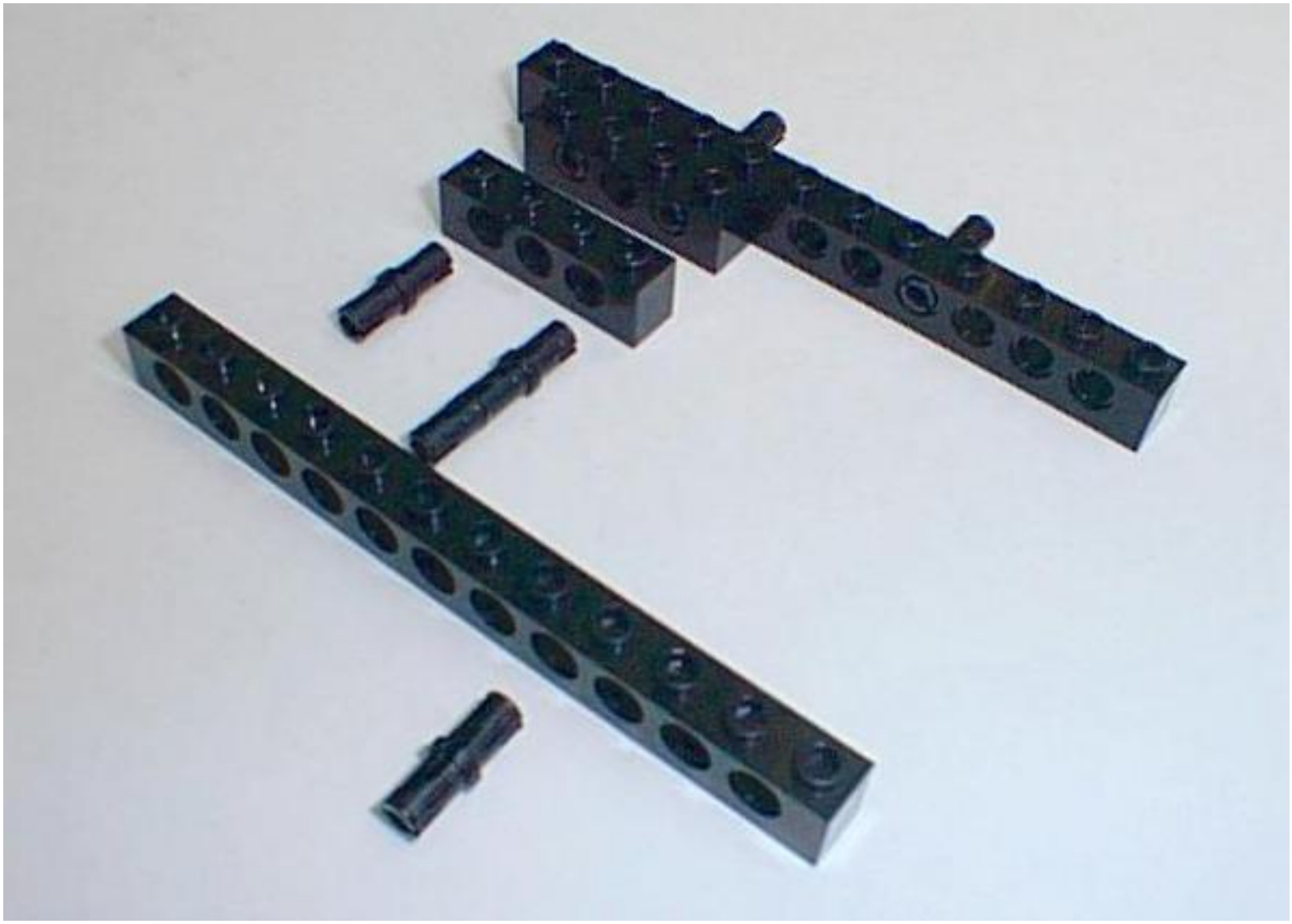
Next

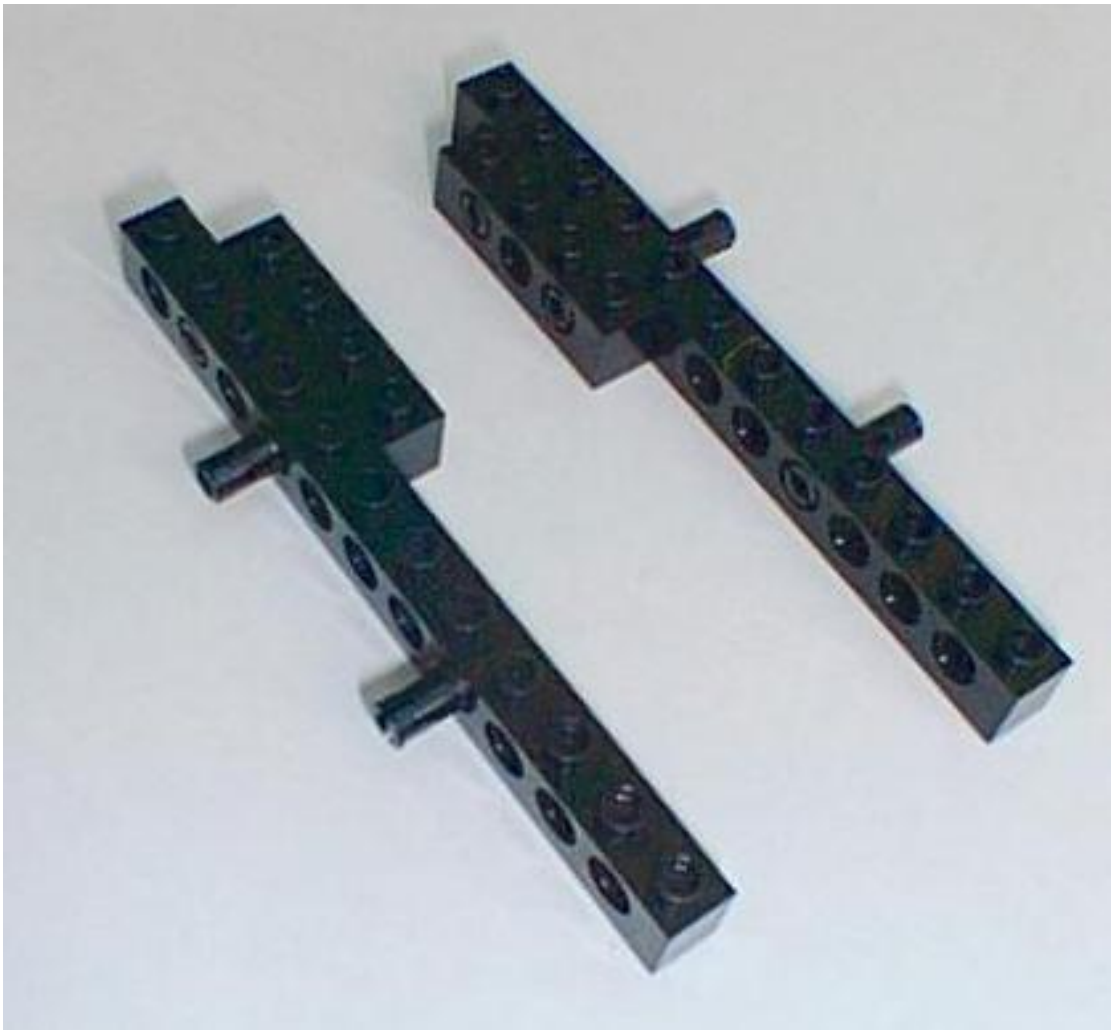
Up

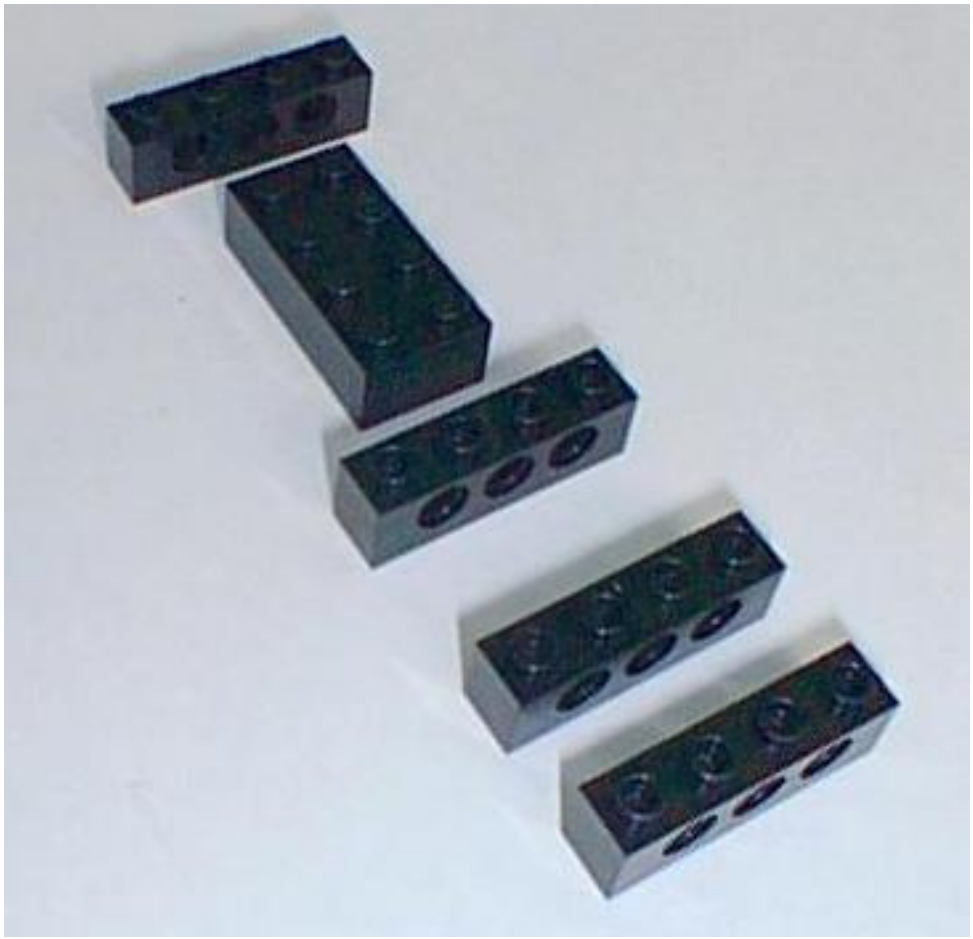
Clock

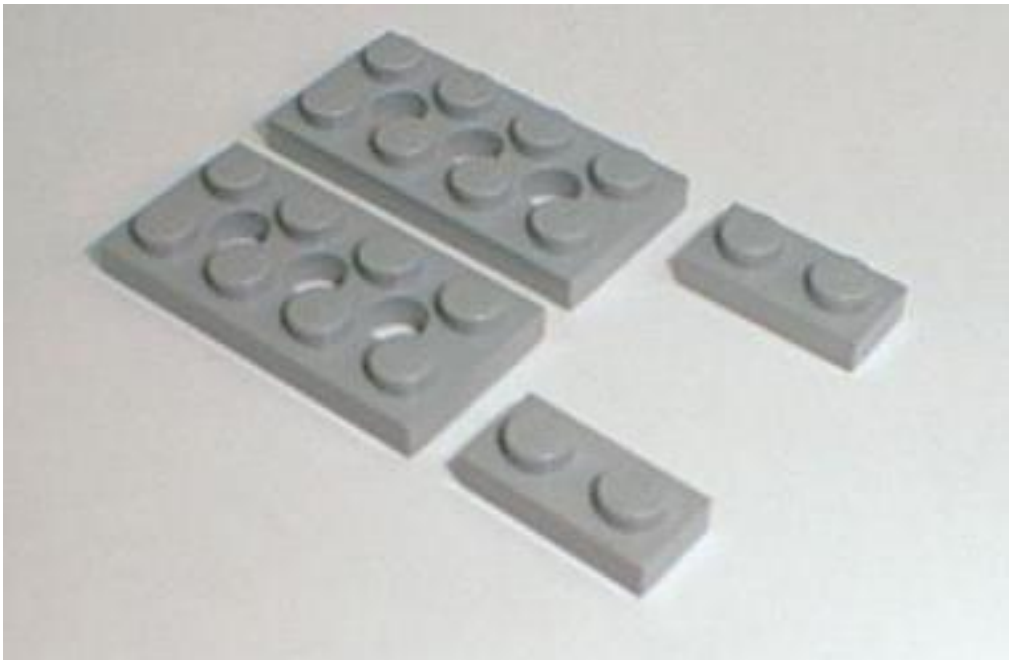
Building Instructions

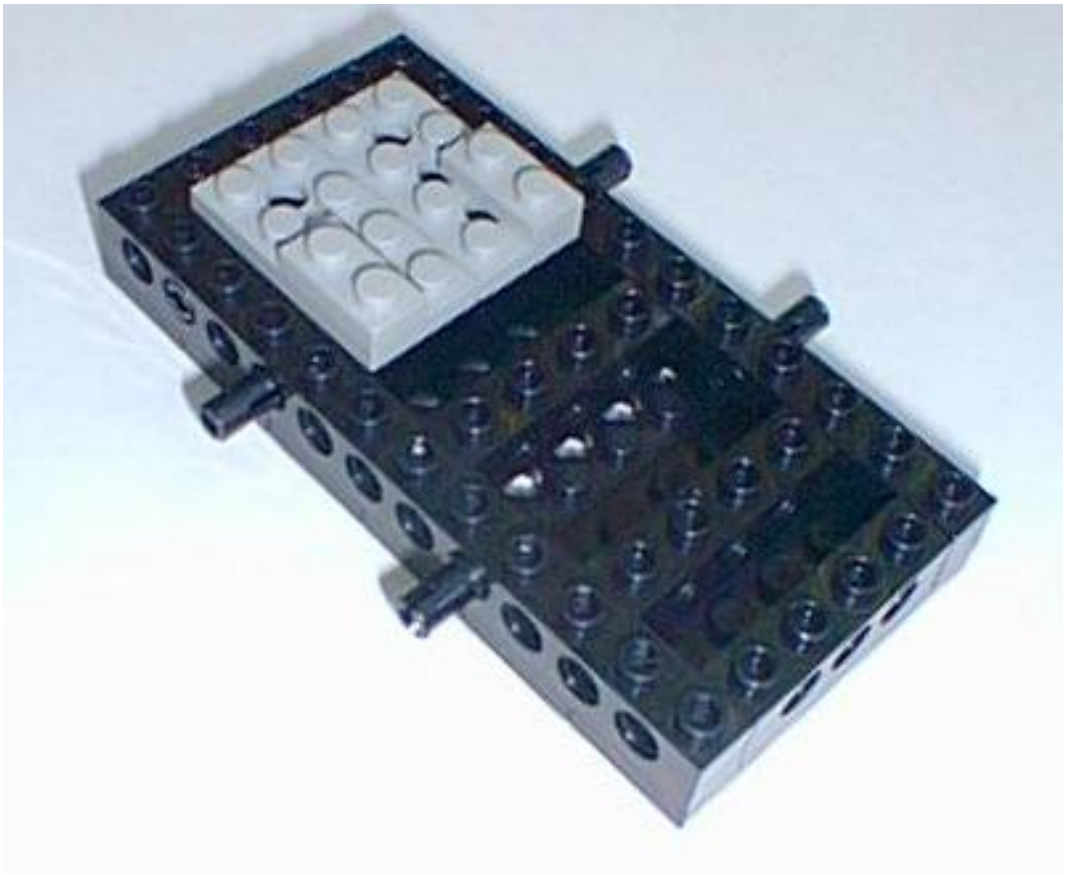
Back











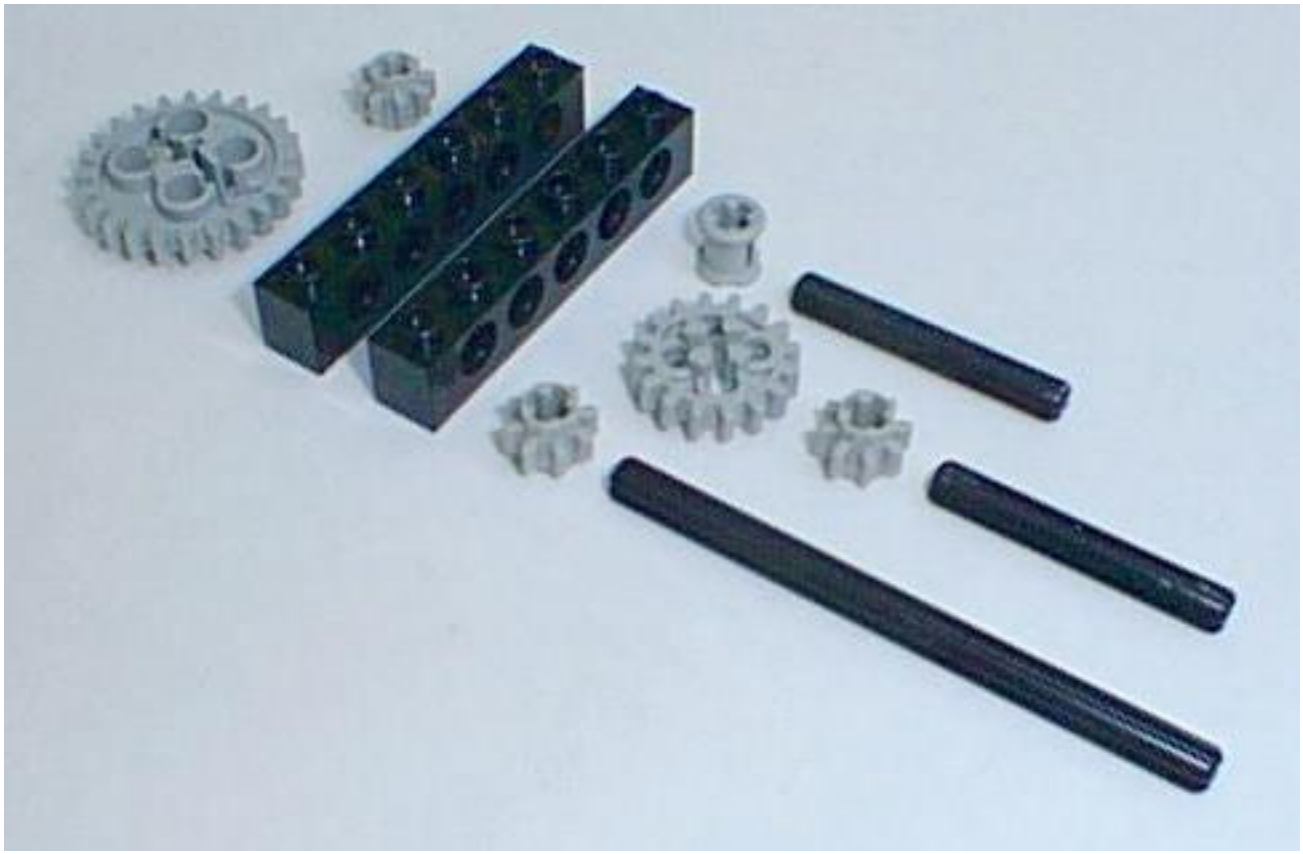
Next

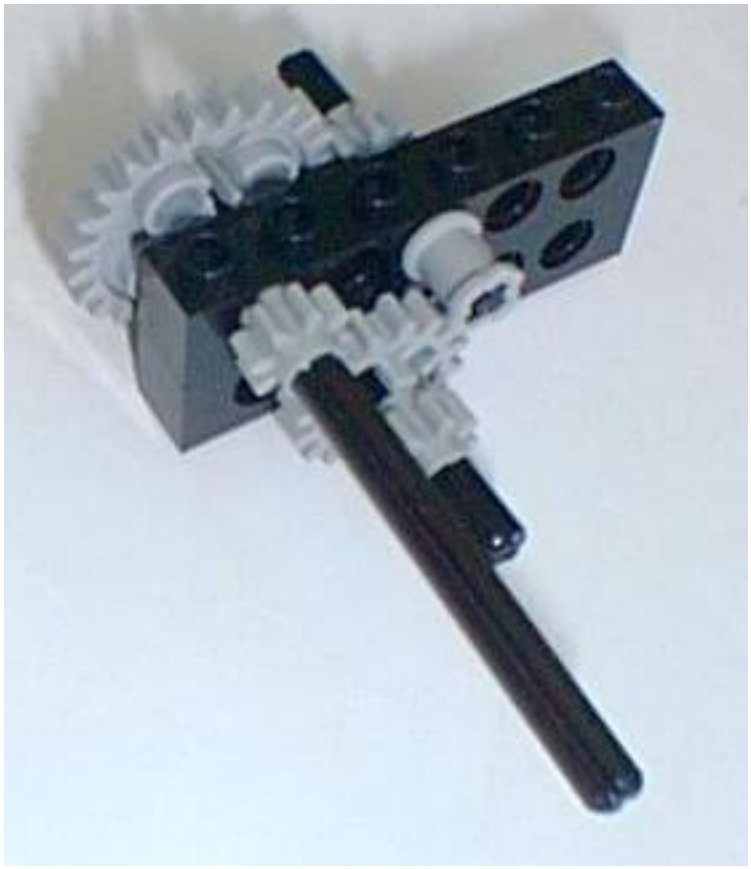
Up

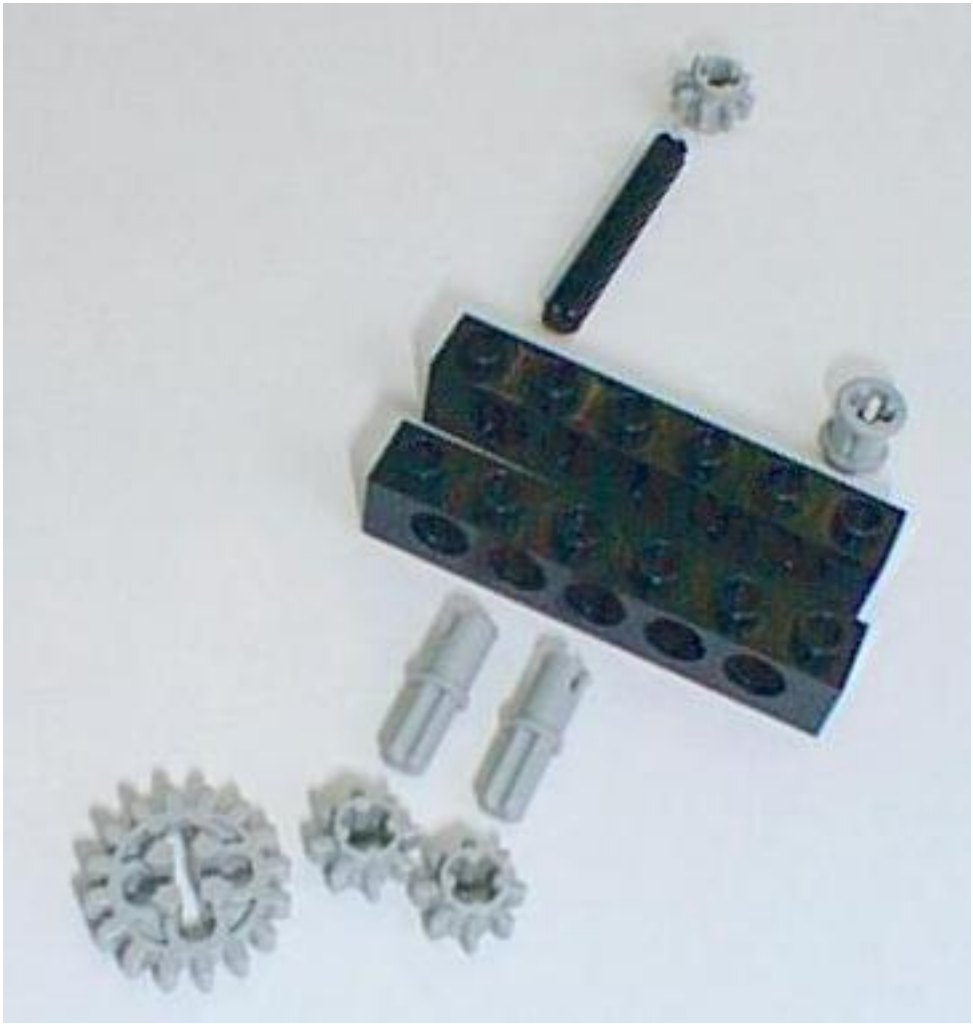
Clock

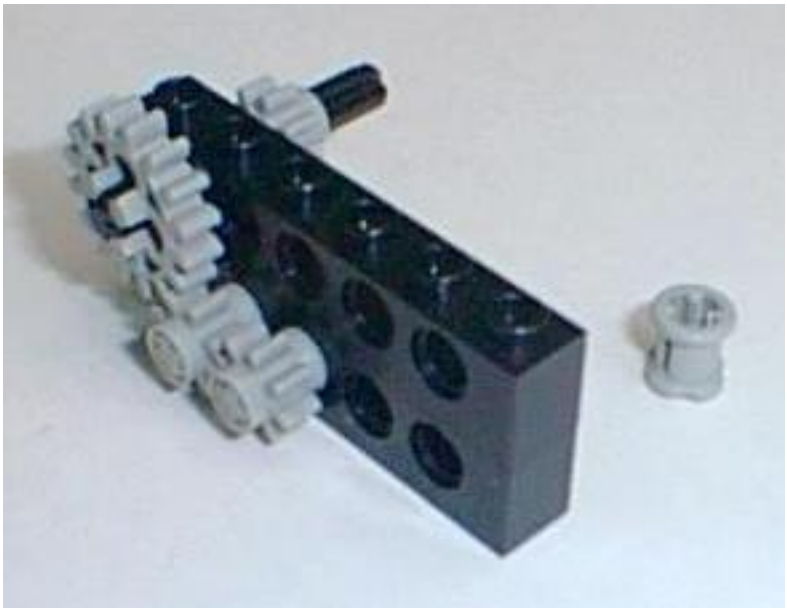
Building Instructions

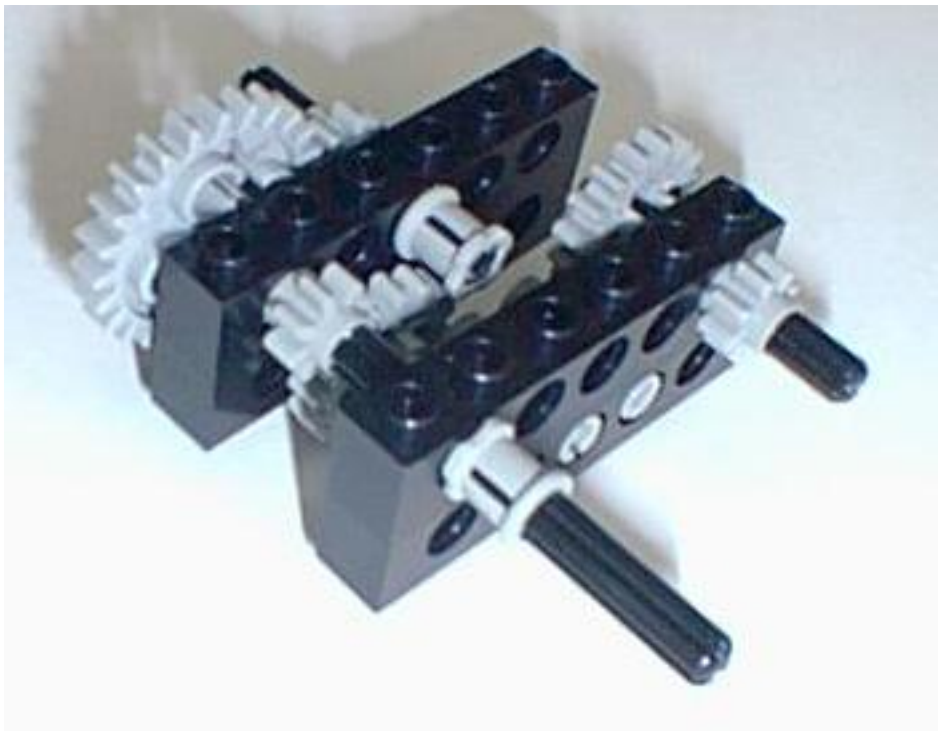
Back

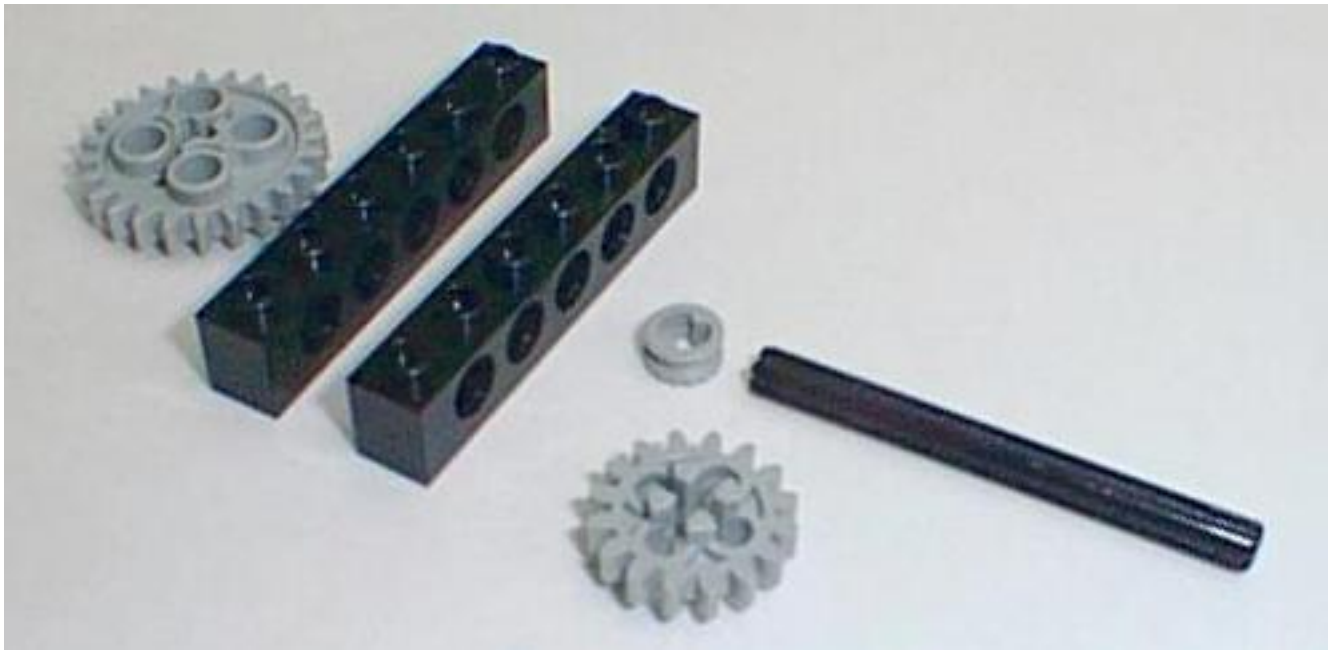


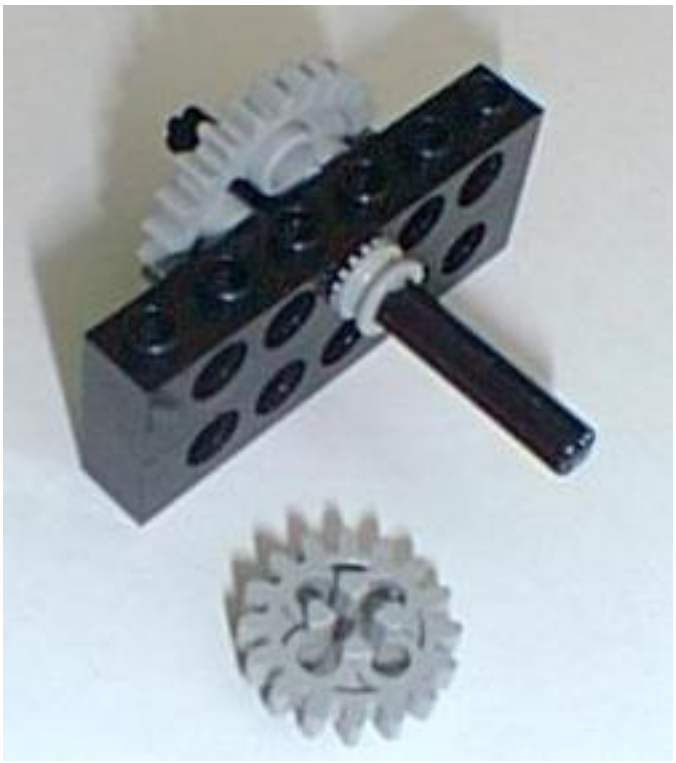


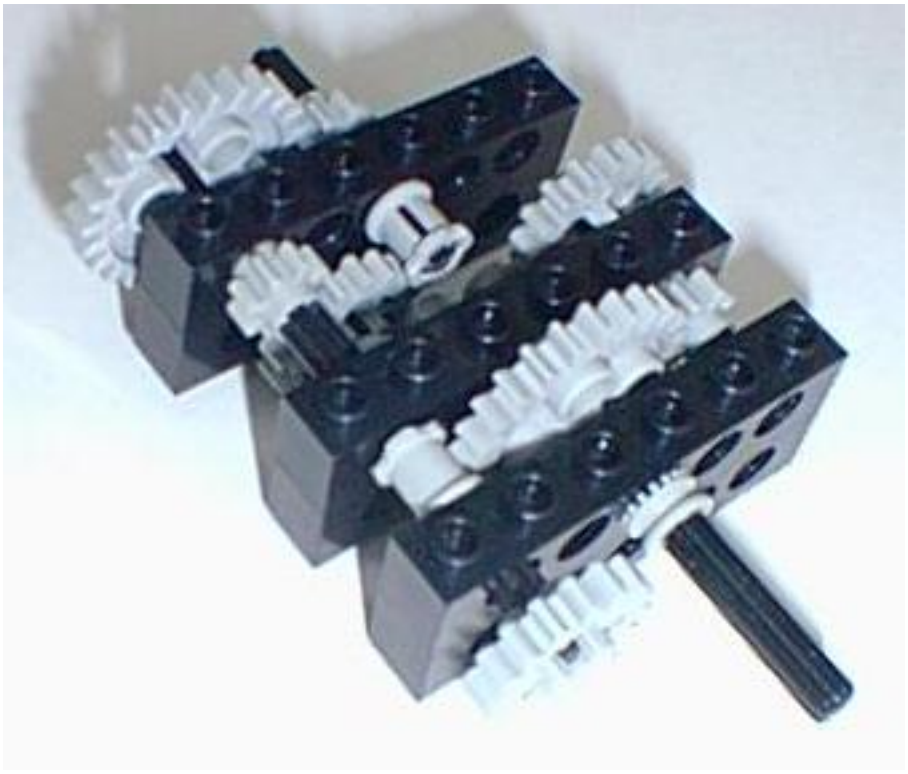


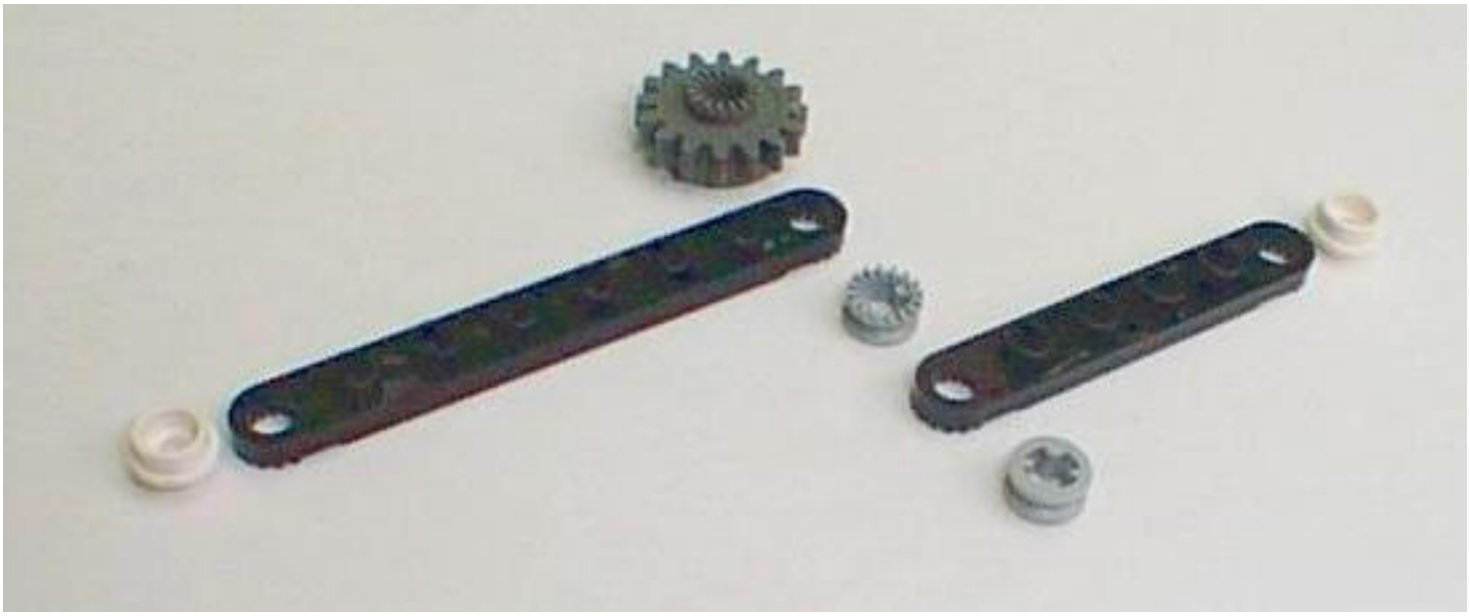


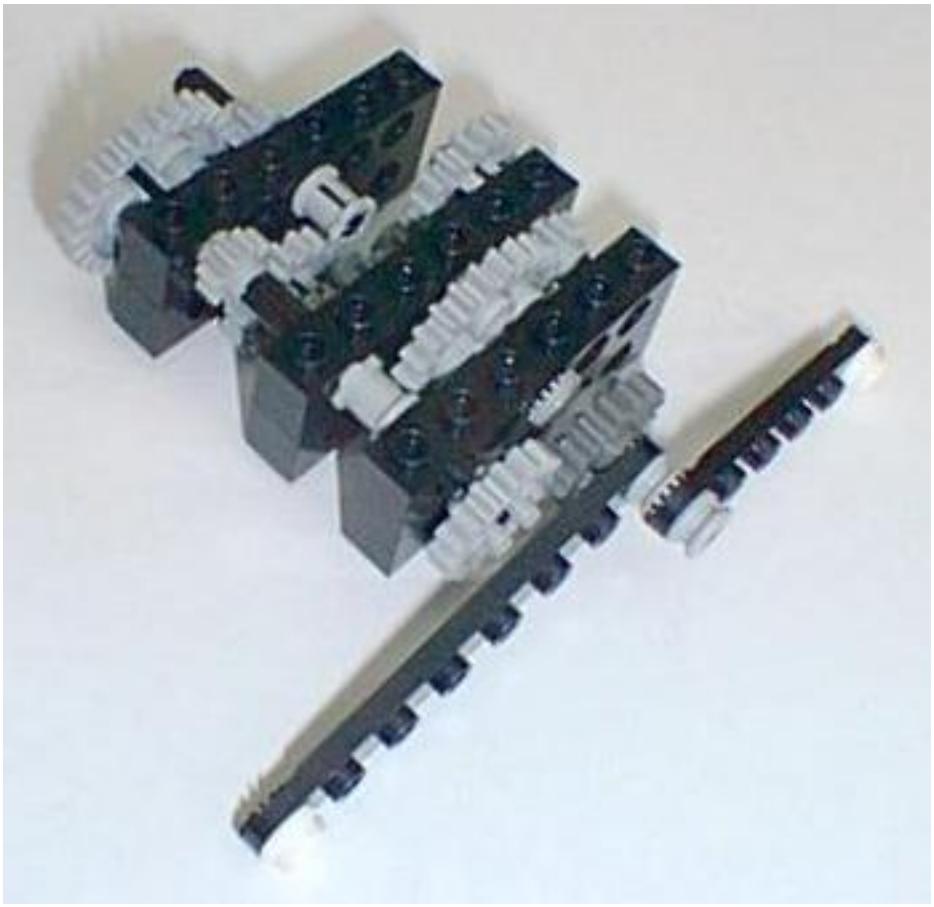












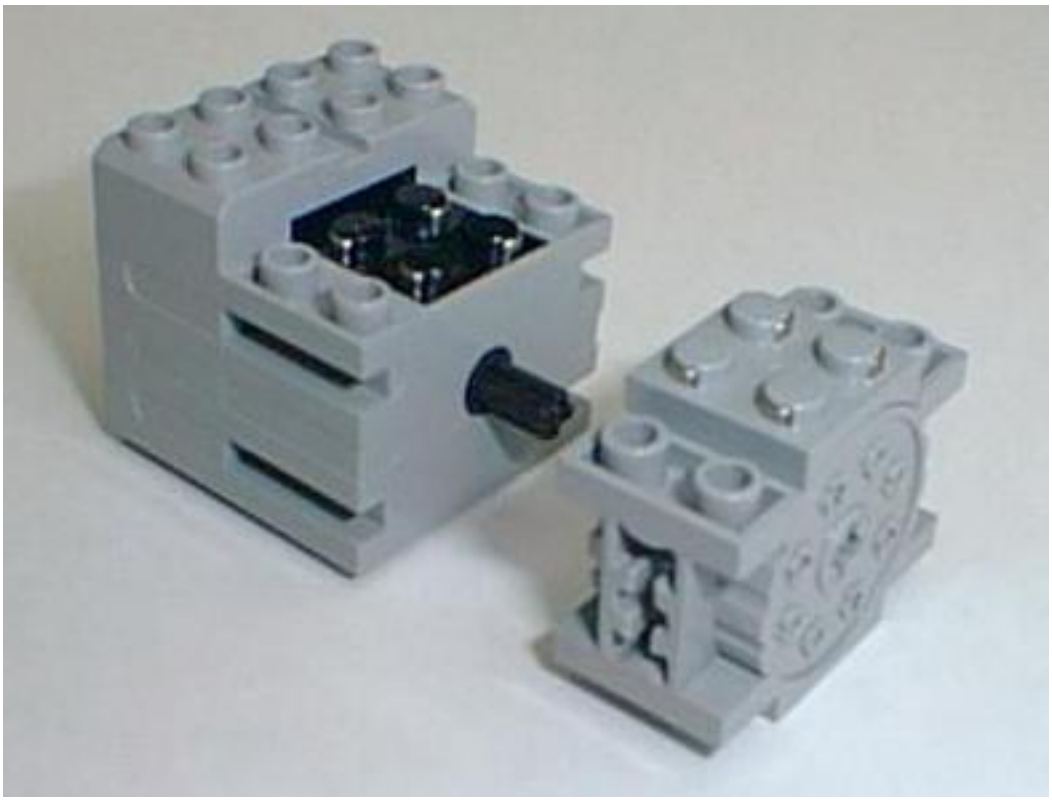
Next

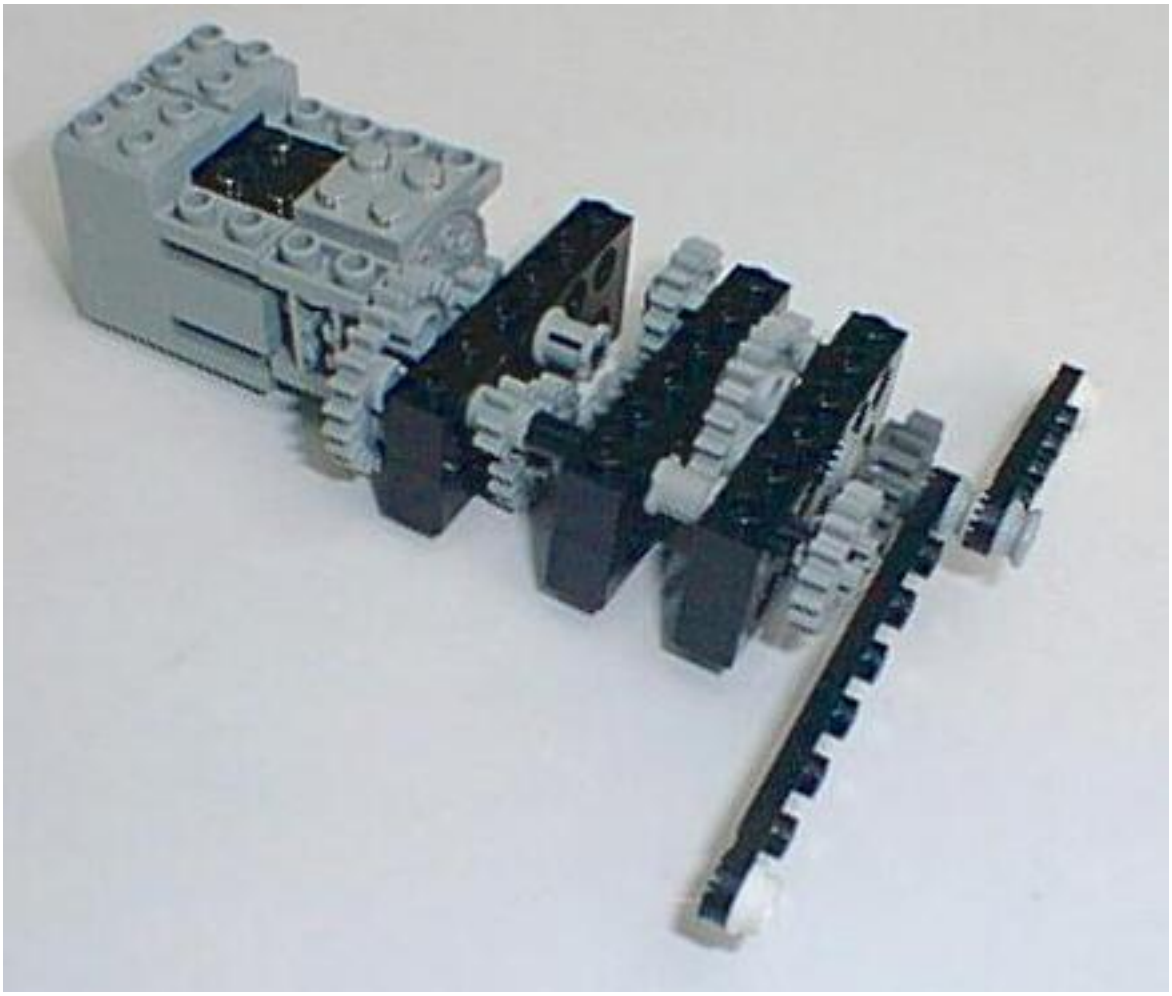
Up

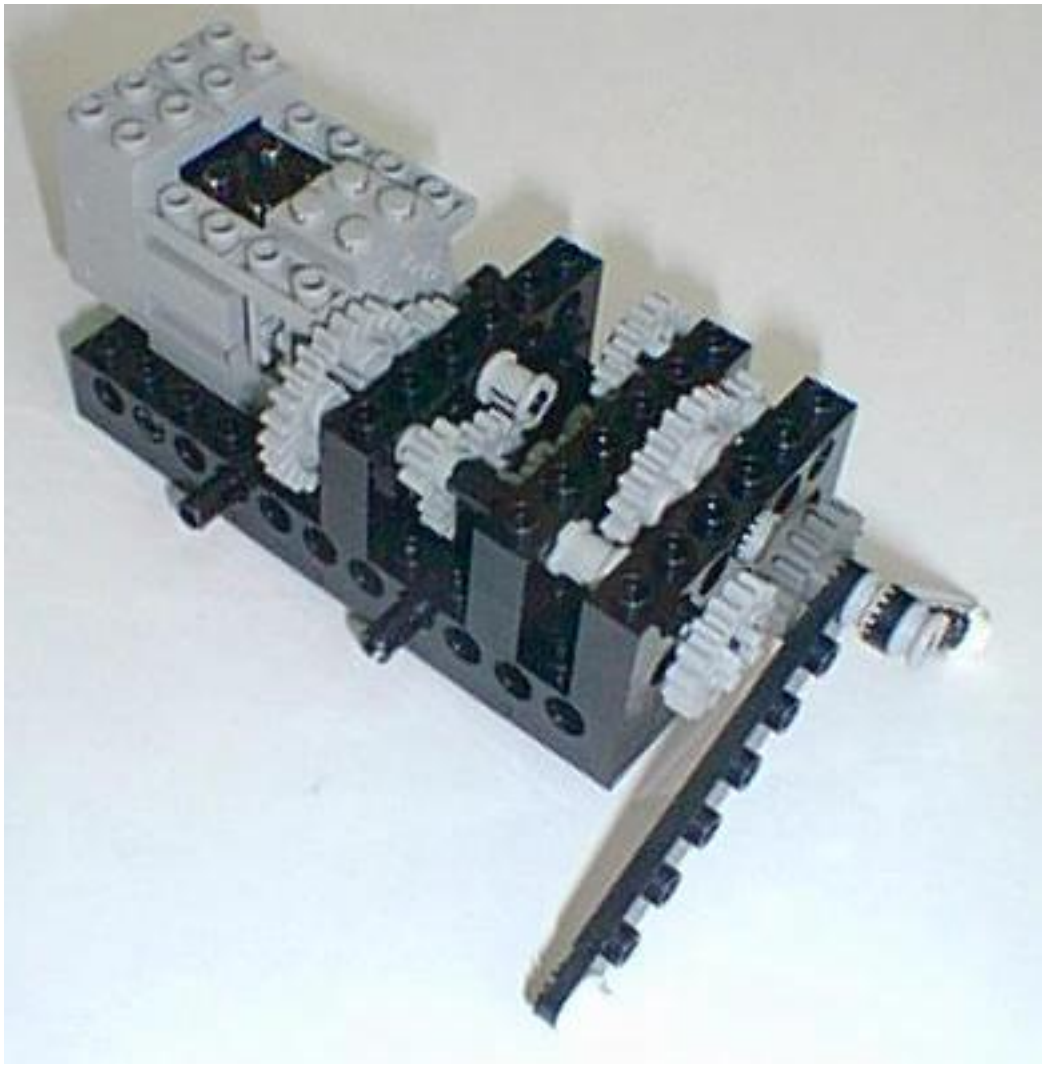
Clock

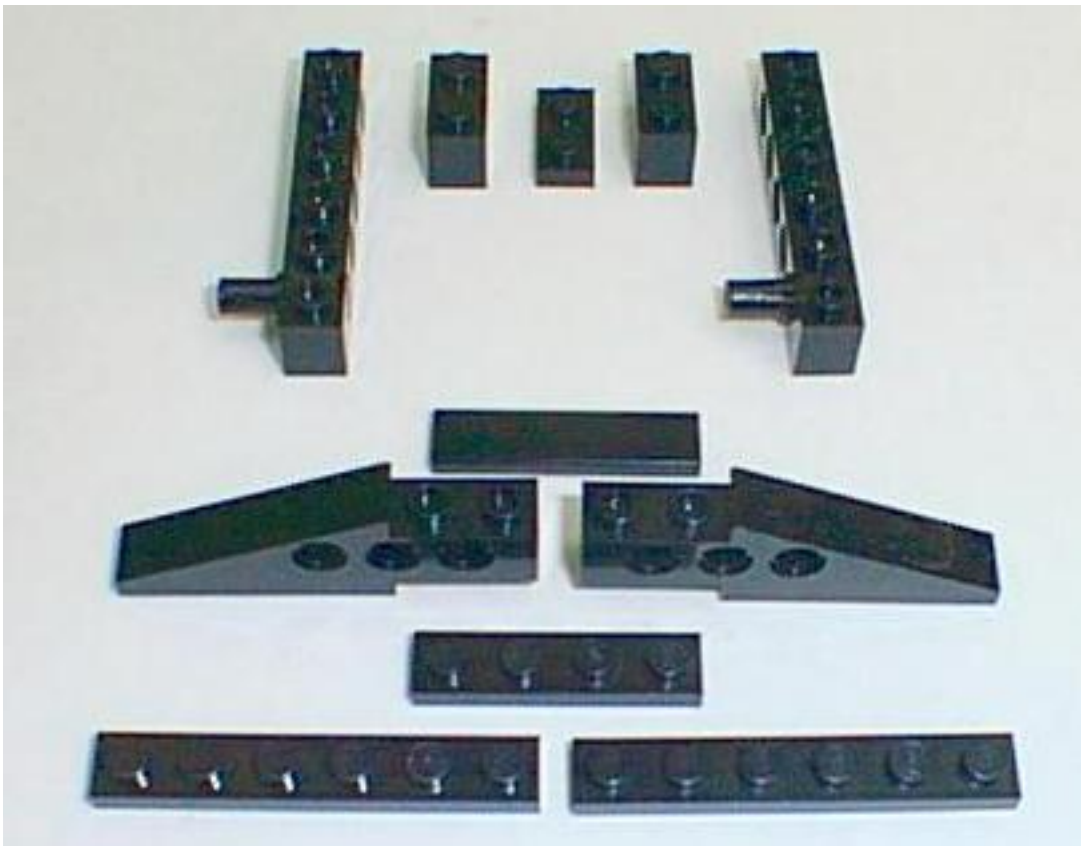
Building Instructions

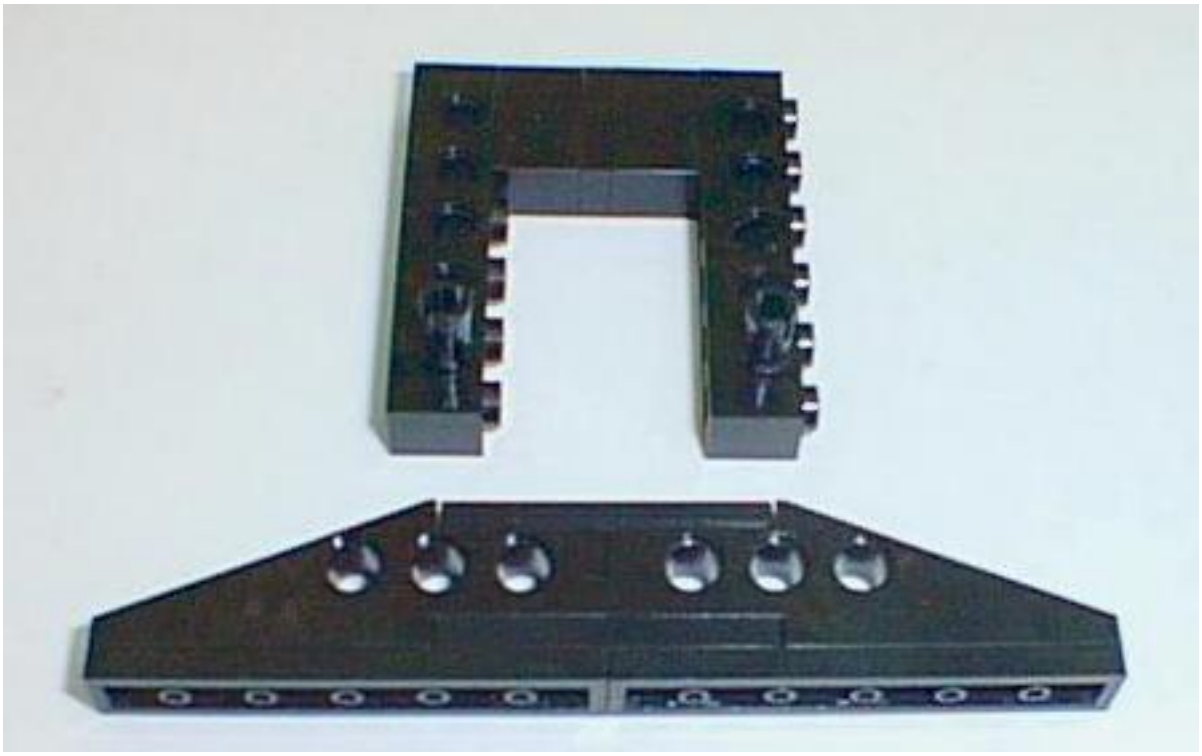
Back

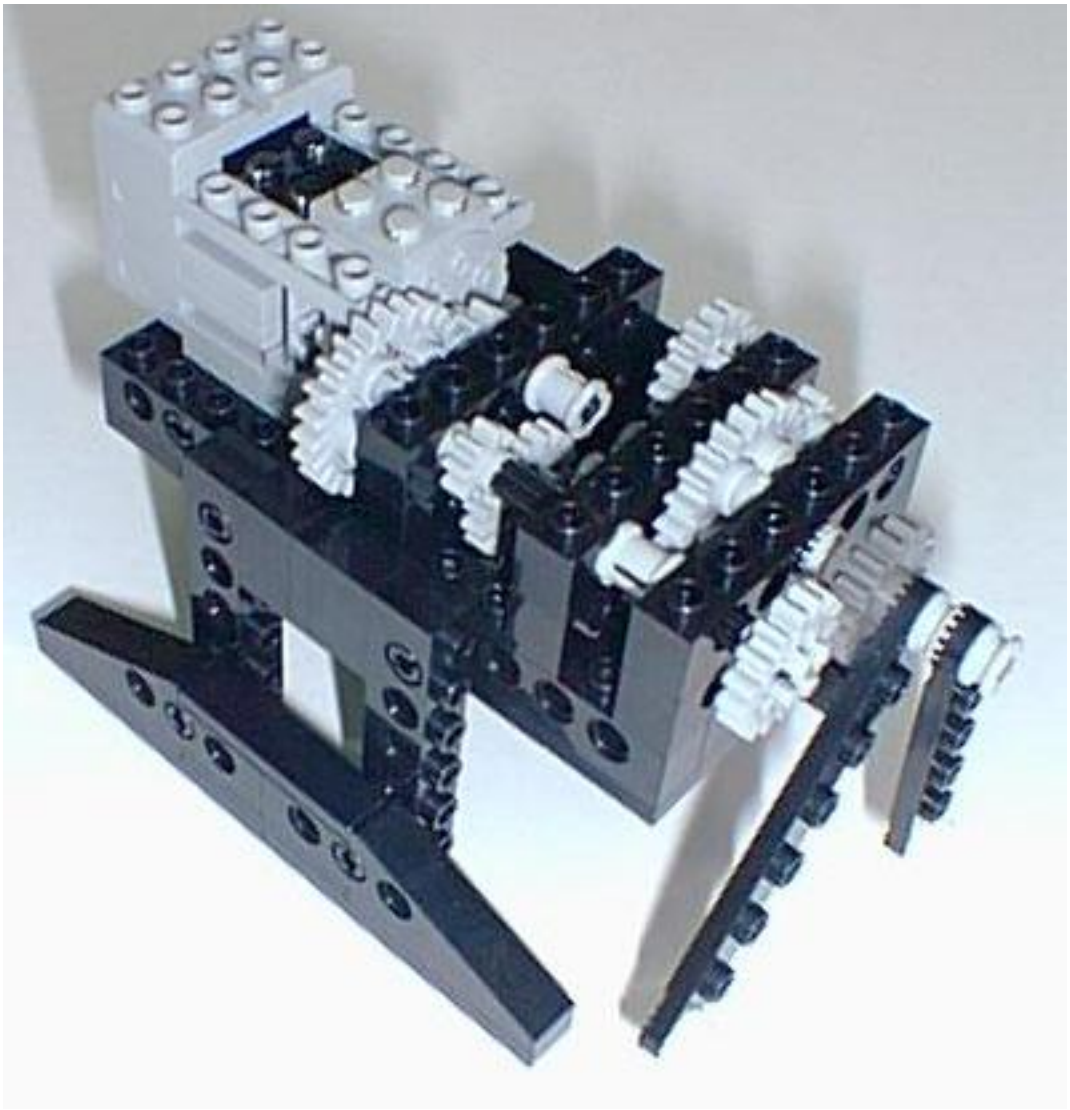












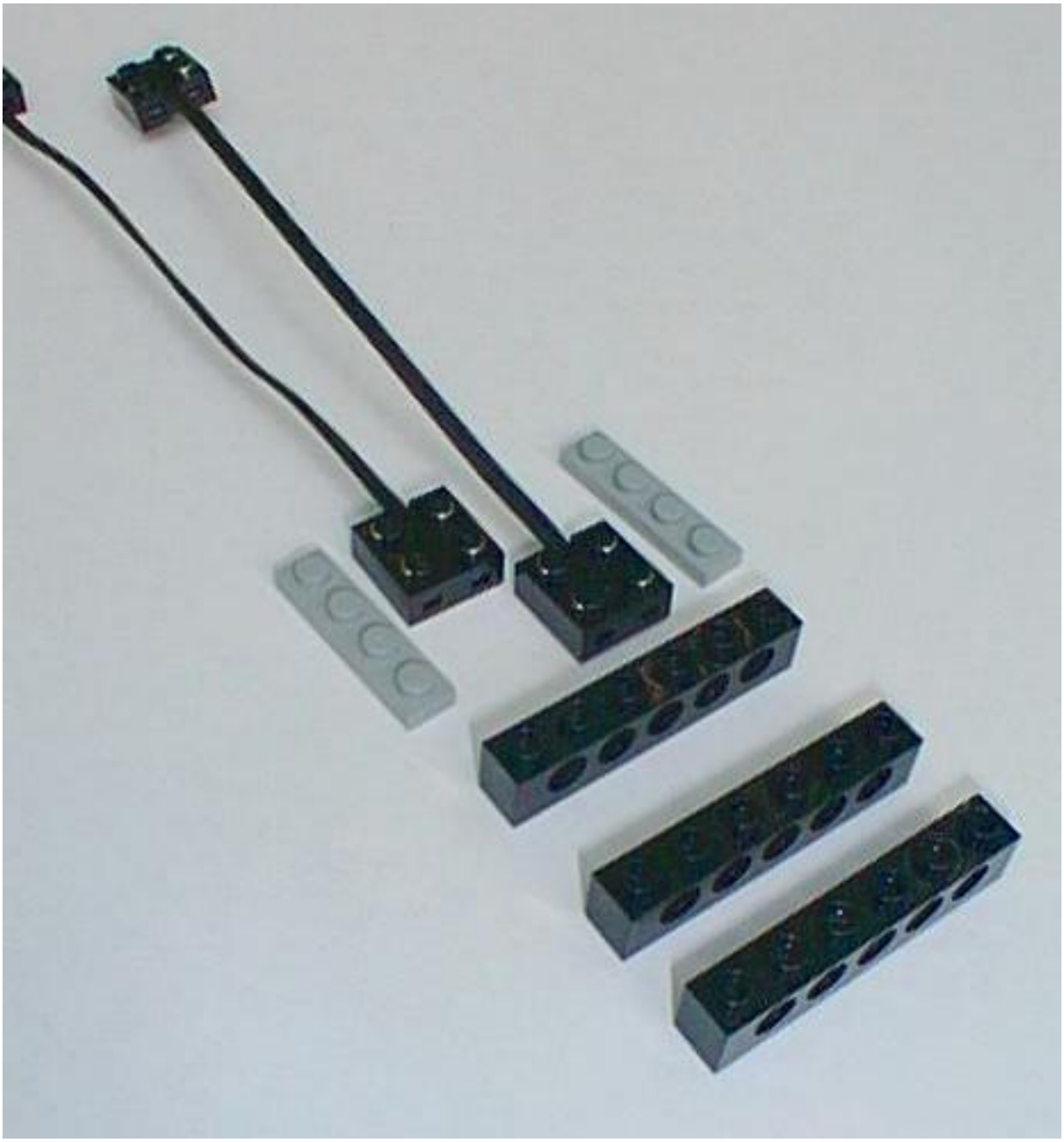
Next

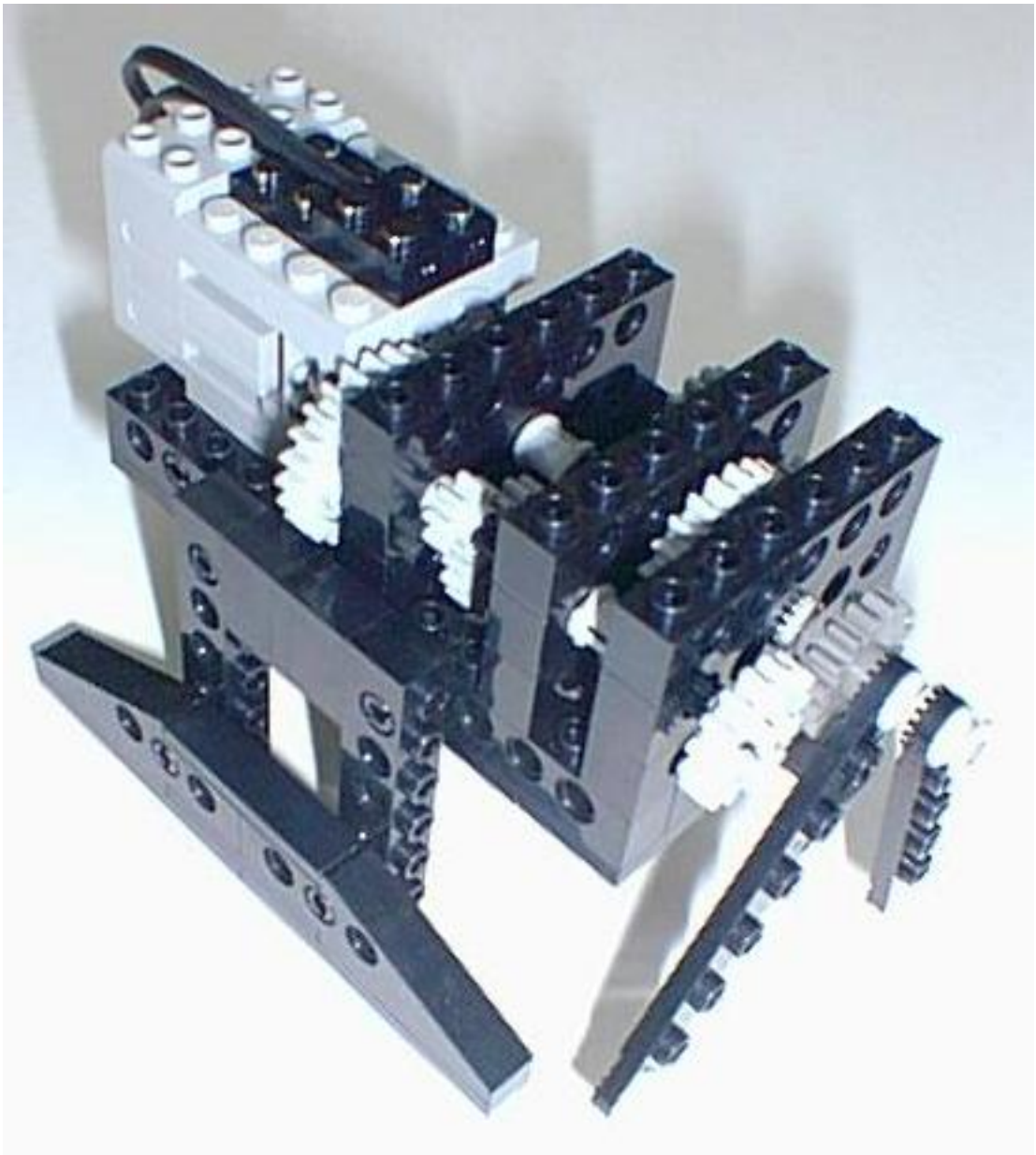
Up

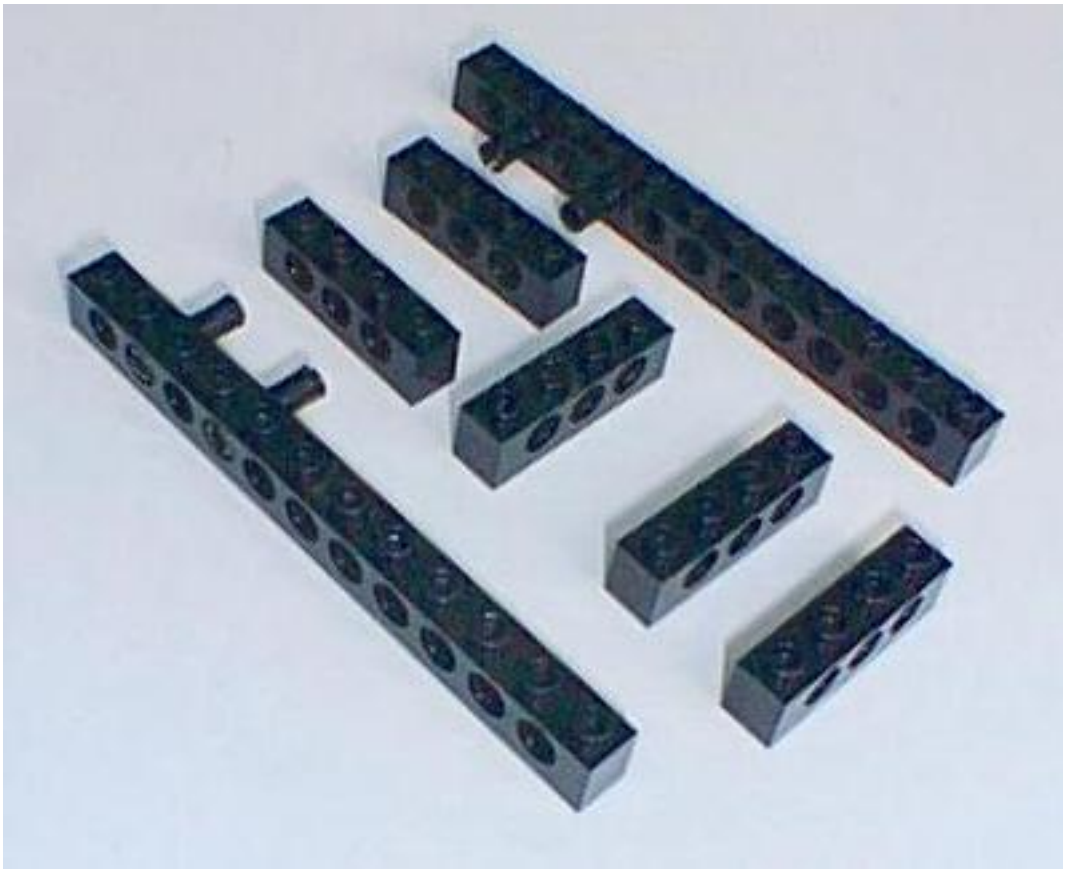
Clock

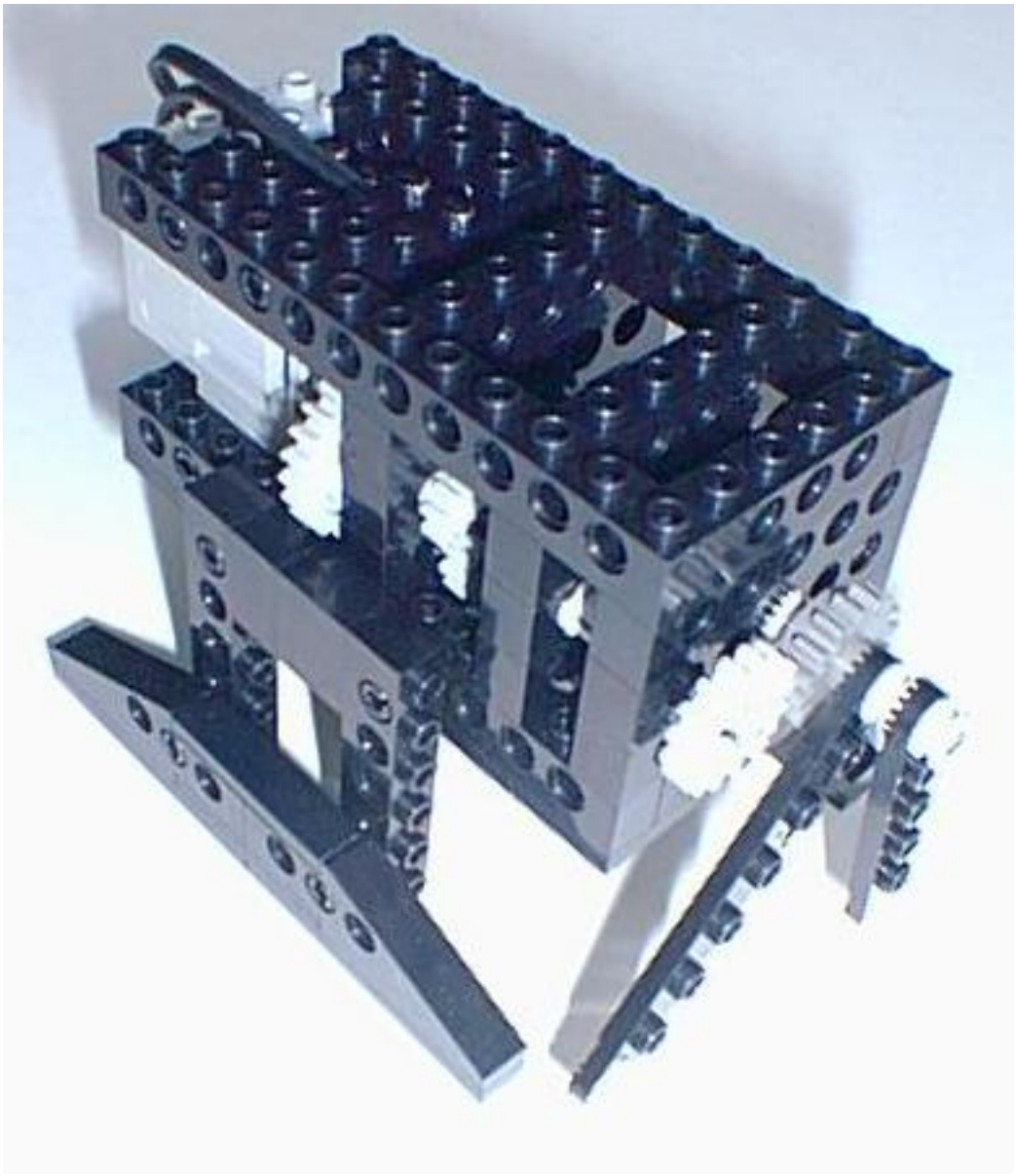
Building Instructions

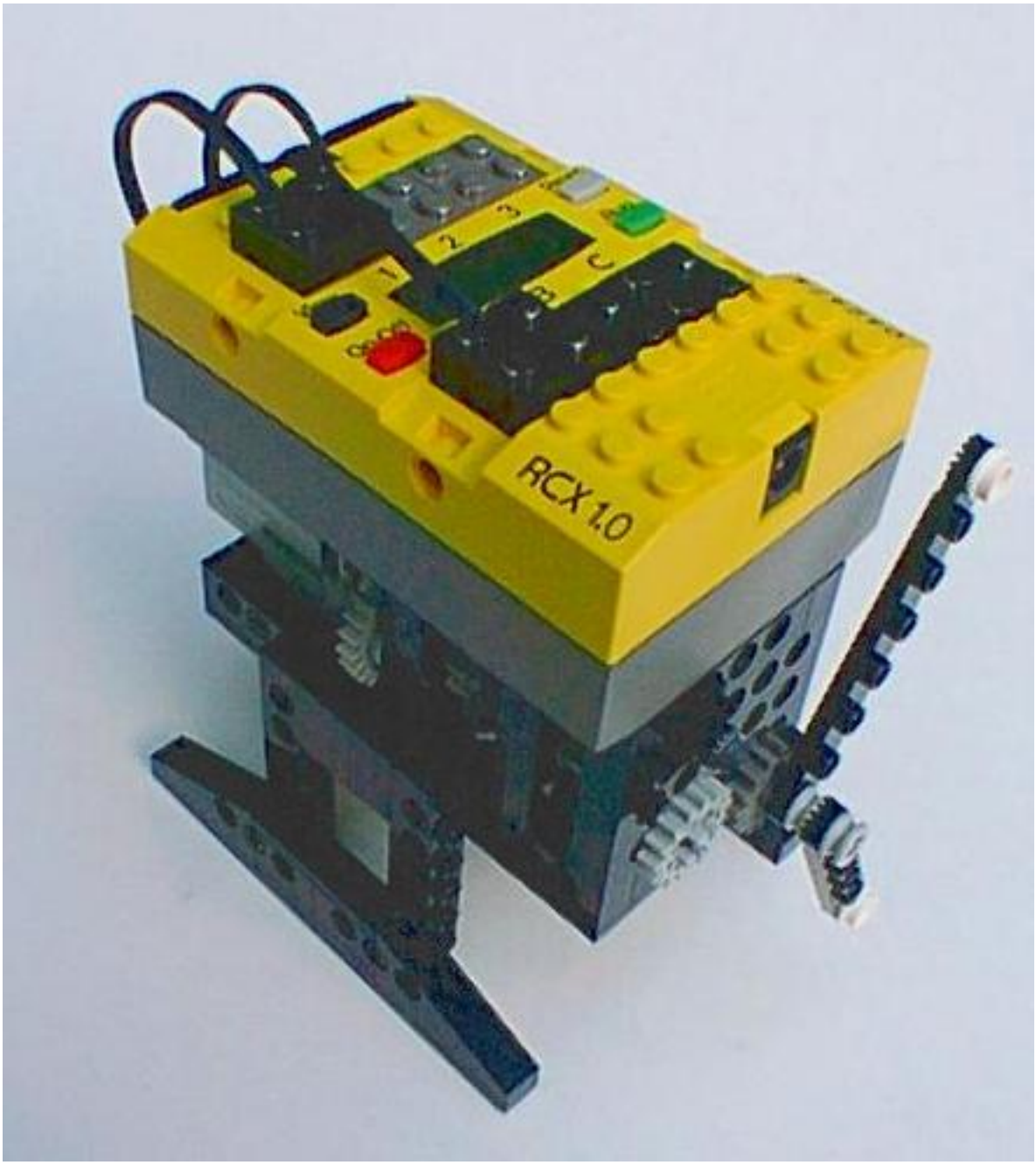
Back











Up