2022 Magnetuhr Version 2 «Geduldsuhr»

Uhr mit einer Kugel als Zeitanzeige. Die Kugel bewegt sich jede Minute um eine Minute weiter.

Martin Allemann, Aug 2022

1 KONZEPT 1

2 DESIGN 2

2.1 MAIN PARTS 3
2.2 BOARD CONNECTIONS 4
2.3 UNTERSETZUNG 5
2.4 ARDUINO MIT A4988 5

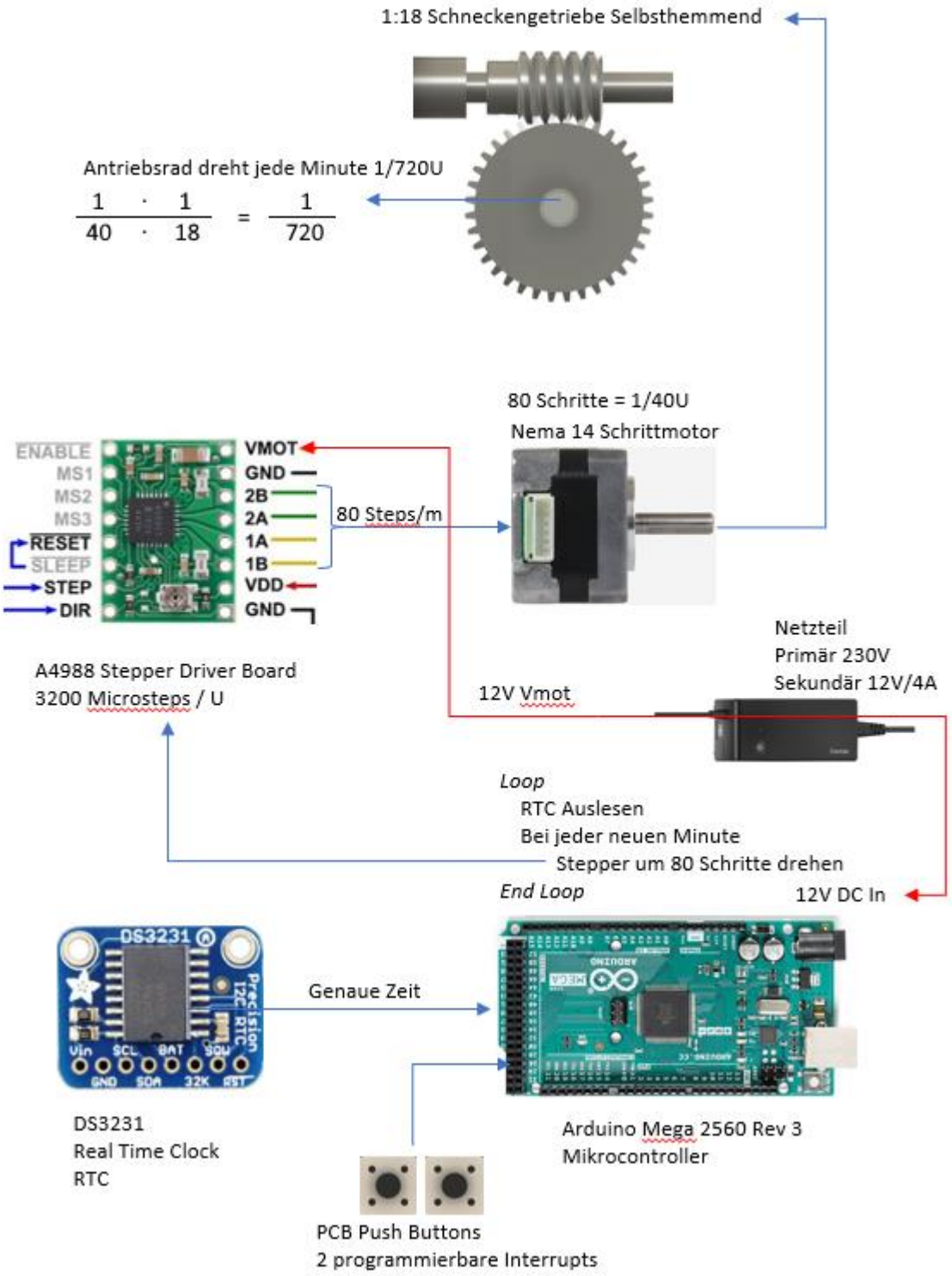
3 ARDUINO SKETCH 6

3.1 HEADER 6
3.2 VOID SETUP() 7
3.3 VOID LOOP() 7
3.4 SUBROUTINES 8
3.4.1 MOVE STEPPER 8
3.4.2 INITIAL CLOCK SEETINGS 8
3.4.3 INTERRUPT HANDLER 9

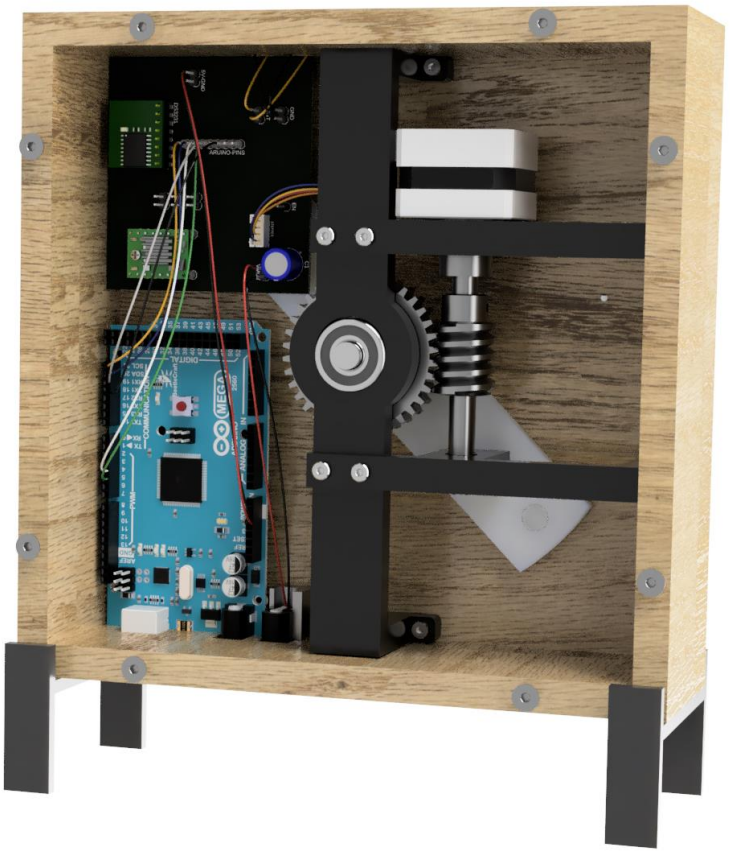
4 BOM 10

4.1 PRINT PARTS 11
4.2 CNC PARTS 12
4.3 DREH PARTS 13
4.4 KAUF PARTS 14

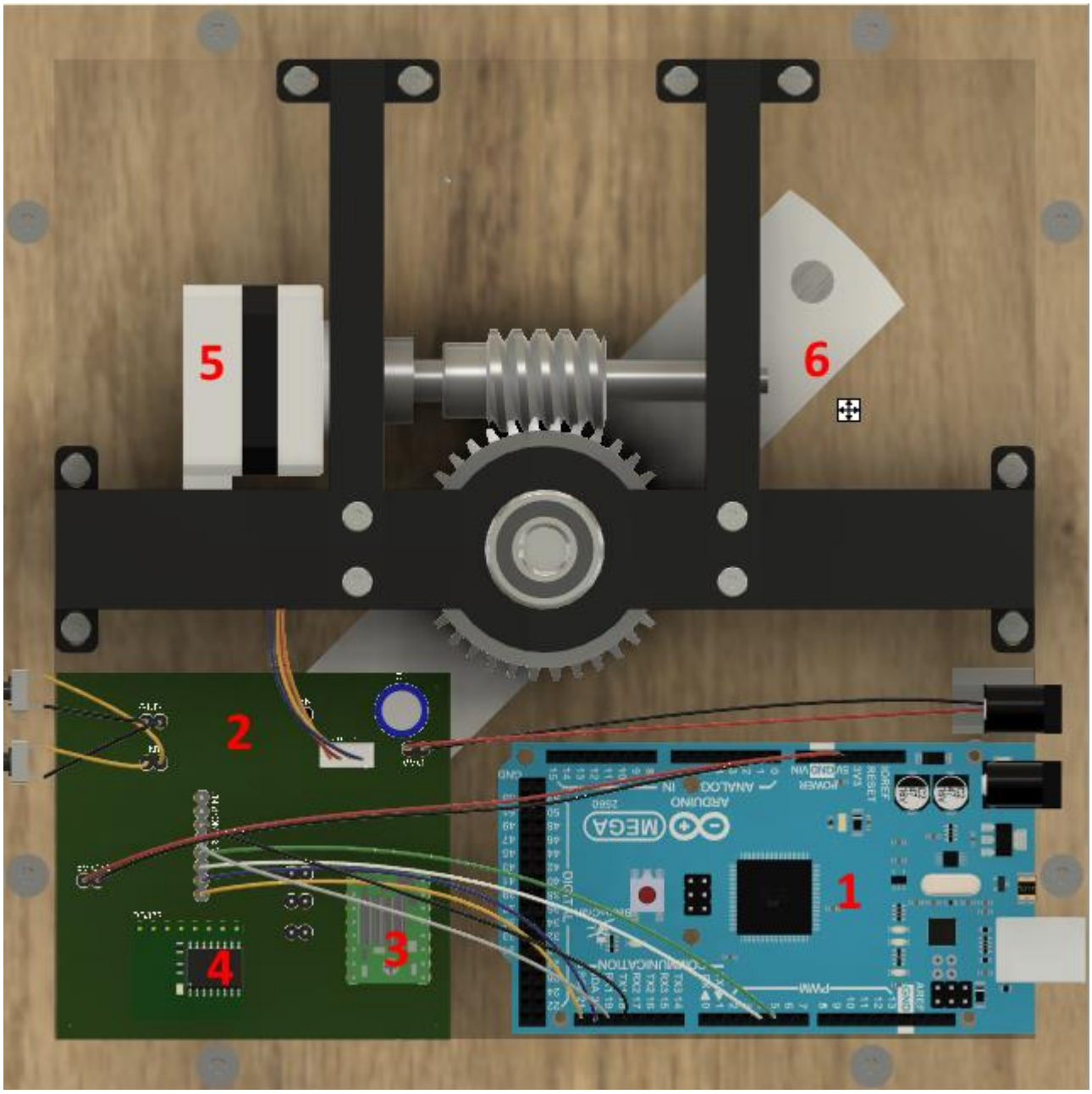
1 KONZEPT


1:18 Schneckengetriebe Selbsthemmend
Antriebsrad dreht jede Minute 1/720U
1 · 1 / 40 · 18 = 1 / 720
80 Schritte = 1/40U
Nema 14 Schrittmotor
ENABLE
MS1
MS2
MS3
RESET
SLEEP
STEP
DIR
VMOT
GND
2B
2A
1A
1B
VDD
GND
80 Steps/m
A4988 Stepper Driver Board
3200 Microsteps / U
12V Vmot
Netzteil
Primär 230V
Sekundär 12V/4A
Loop
RTC Auslesen
Bei jeder neuen Minute
Stepper um 80 Schritte drehen
End Loop
12V DC In
DS3231
Genaue Zeit
DS3231
Real Time Clock
RTC
Arduino Mega 2560 Rev 3
Mikrocontroller
PCB Push Buttons
2 programmierbare Interrupts

2 DESIGN



2.1 Main Parts



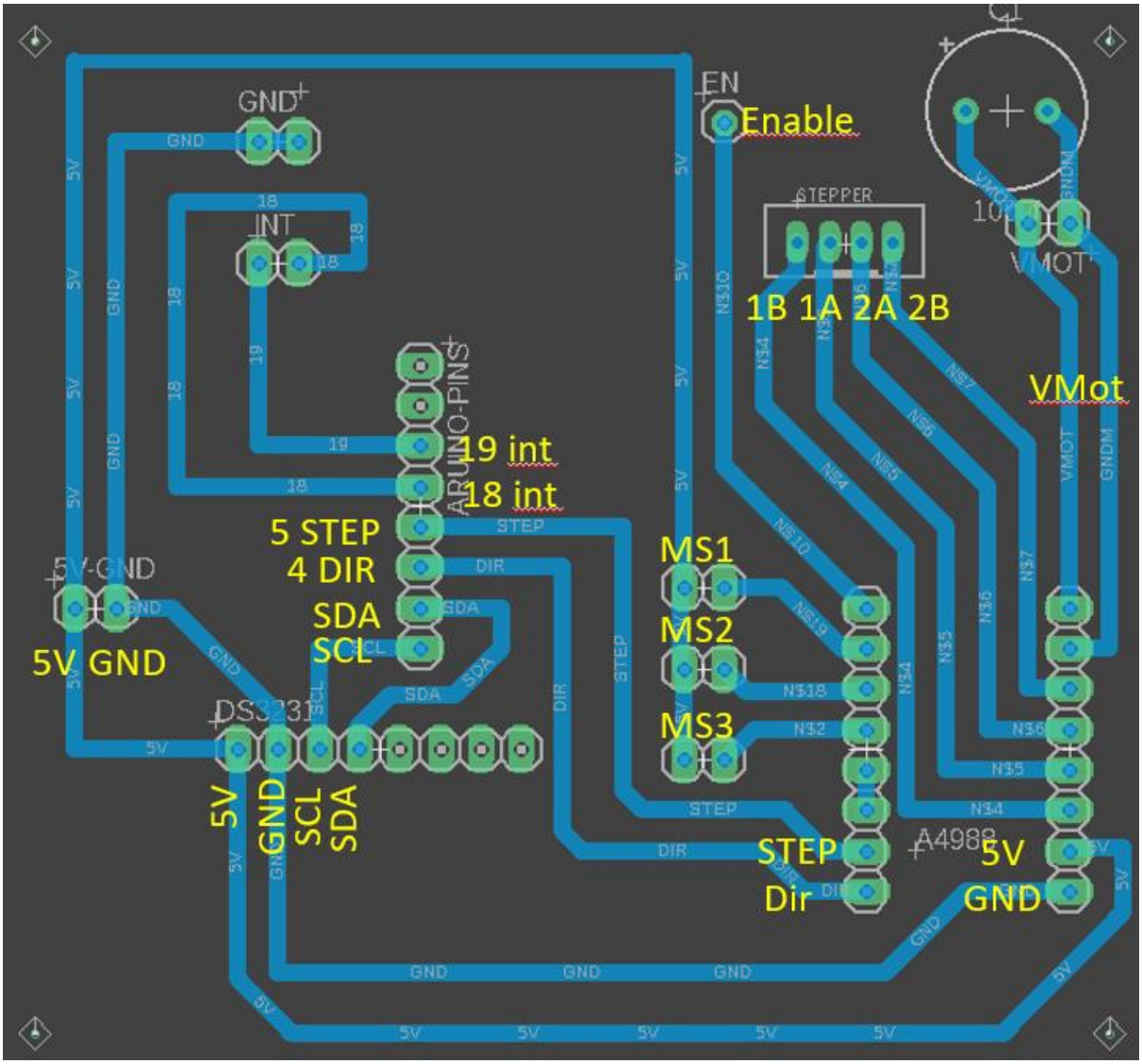
1: Arduino MEGA (1): Liest im Loop die Realtime Clock DS3231 (4) und schaltet den Magnetrotor (6) via Schrittmotor (5) und Schneckentrieb jede Minute um 1/720 Umdrehung (1 Minute bei 12 Stundenanzeige) weiter. Verarbeitet 2 programmierbare Interrupt Pins.

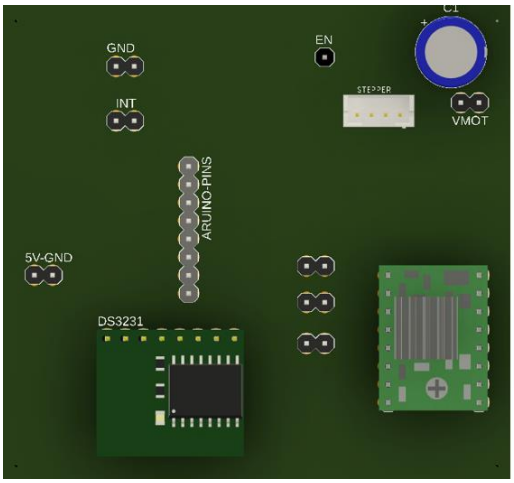
2: Board (2): verbindet die Arduino Pins mit dem Schrittmotortreiber A4988 (3) und der RTC DS3231 (4). Zusätzlicher DC Eingang für Schrittmotor (Vmot).

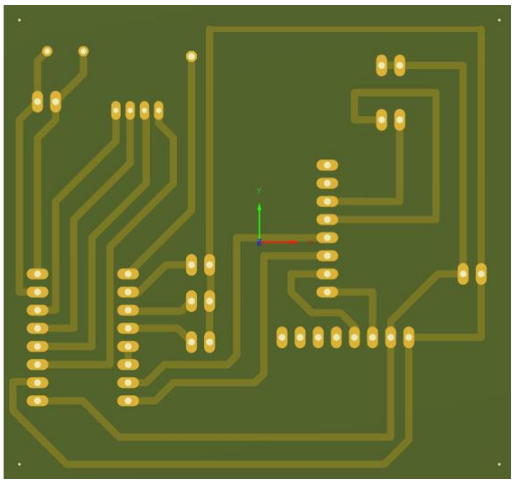
5: Nema 14 Schrittmotor (5) treibt mit Schnecke in einer 1:18 Untersetzung den Magnetrotor (6) an. Bei 3200 Microsteps entspricht eine Minute am Schrittmotor 80 Steps.

6: Magnetrotor (6): Mitnehmer mit eingebautem Permanentmagnet, nimmt die im Zifferblatt auf der Frontseite eingelassene Kugel mit.

2.2 Board Connections





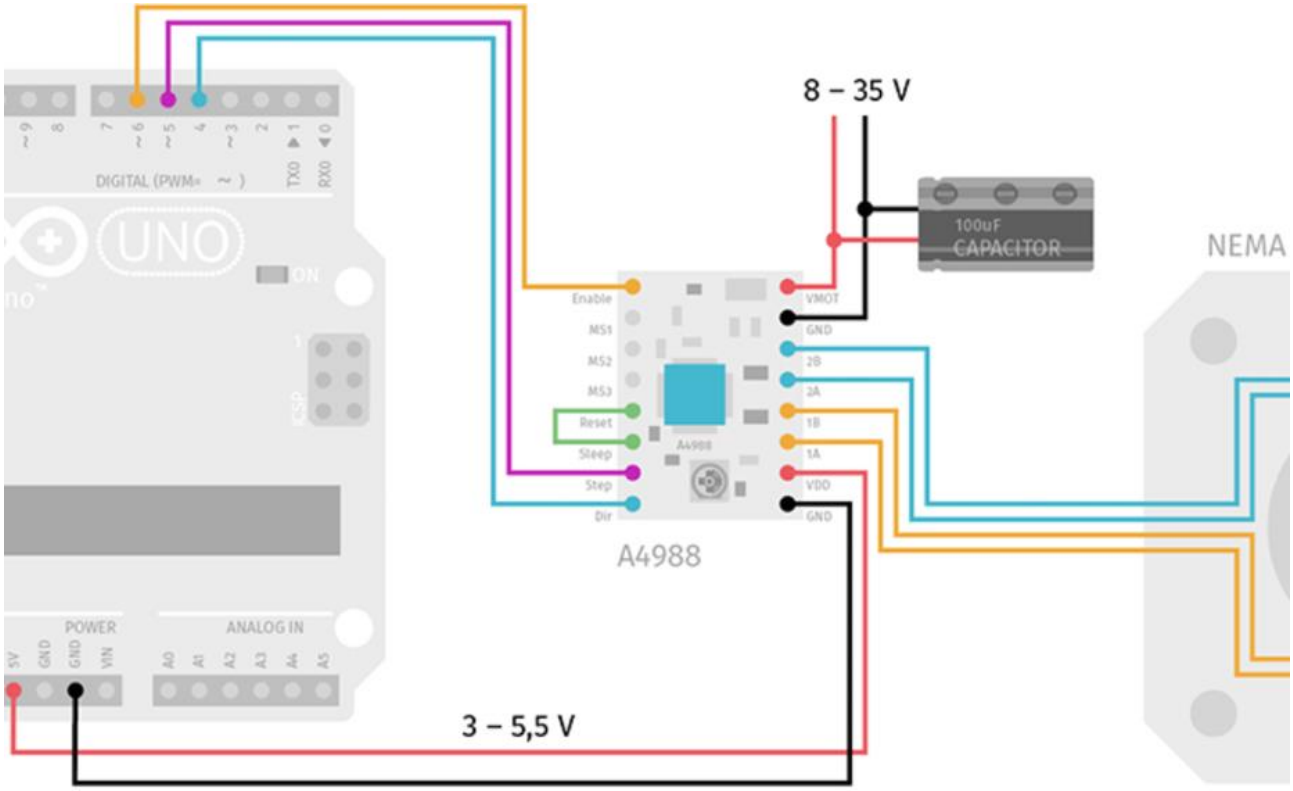


2.3 Untersetzung

Für eine Minute am Rotor muss der Schrittmotor 80 Schritte fahren:

Schnecke	1 zu	18			
1 Minute	1/720 Umdrehung =		0.0013889	Umdrehungen	
Bei 1:18	18x (1/720)		0.0250000	Umdrehungen	
Schritte	5	Schritte am Antrieb für 1 Minute			bei 200 Steps/U
Schritte	80	Schritte am Antrieb für 1 Minute			bei 3200 Steps/U

2.4 Arduino mit A4988



3 ARDUINO SKETCH

3.1 Header

```
//-------------------------------------------------------------------------------------
// Magnetuhr Code © Martin Allemann 2022
// Read Real Time Clock DS3231
// After every Minute, move Stepper by 'steps' (1 Minute at work gear output)
// Enable a4988 Stepper driver only if needed (avoid Stepper noise). The work gear
// holds the torque
// Set Clock at Startup (commented out in production code after initial clock setting)
// Interupt 18/19: move clock 1m clockwise or 1m counterclockwise. Set Seccond to 0 at
// 1m clockwise move
//-------------------------------------------------------------------------------------

// Include RTC and Serial/I2C Libraries
#include <DS3231.h>
#include <Wire.h>

// Stepper variables
int stepCounter;                // Count Steps moved
int steps = 80;                 // number of steps to move every 1 minute
int stepdelay = 1000;           // delay between steps

// Pins used as Interrupt pins connected to PCB-Switches
const byte interruptPin18 = 18; // Step 15min
const byte interruptPin19 = 19; // Step 1min
int int18stat = 1;              // Interrupt Pin Status (1:high, 0:low)
int int19stat = 1;              // Interrupt Pin Status (1:high, 0:low)
bool int18on = false;           // True if Interrupt was fired
bool int19on = false;           // True if Interrupt was fired

// Real Time Clock variables
byte LastMinute = 0;
byte year;
byte month;
byte date;
byte dOW;
byte hour;
byte minute;
byte second;
bool century = false;
bool h12Flag;
bool pmFlag;

// Instantiate RTC Board
DS3231 myRTC;
```

3.2 Void Setup()

```
void setup()
{
    // Start the I2C interface
    Wire.begin();

    // Set Clock and Start Serial Monitor. Comment out for production
    // SetClock();
    //Serial.begin(9600);
    //Serial.println("Start");


    // Set Stepper Output Pins
    pinMode(6, OUTPUT);           // Enable
    pinMode(5, OUTPUT);           // Step
    pinMode(4, OUTPUT);           // Direction

    // Disable Motor at Startup
    digitalWrite(6, HIGH);         // Disable Motor

    // define interrupt pins. Interrupts if button pressed
    // buttons between pin18/19 to GND, open if not pressed
    pinMode(interruptPin18, INPUT_PULLUP);
    pinMode(interruptPin19, INPUT_PULLUP);
}
```

3.3 Void Loop()

```
void loop()
{
    // Interrupt Handler Pin 18 and Pin 19
    int18stat = digitalRead(interruptPin18);
    int19stat = digitalRead(interruptPin19);
    if (int18stat == 0)
    {
        int18on = true;
        int18();
    }
    if (int19stat == 0)
    {
        int19on = true;
        int19();
    }

    // Get Minute and Move Stepper every 1 Minute by 1 Minute
    minute = myRTC.getMinute();
    if (minute != LastMinute)
    {
        Step1MinuteCW();

        // Control output, comment out in production
        // Serial.print(myRTC.getHour(h12Flag, pmFlag), DEC);
        // Serial.print(" ");
        // Serial.print(Minute);

        LastMinute = minute;
    }
}
```

3.4 Subroutines

3.4.1 Move Stepper

```
void Step1MinuteCW()
{
    // move stepper for 1 Minute clockwise
    digitalWrite(6, LOW);        // Enable
    digitalWrite(4, LOW);        // im Uhrzeigersinn
    for (stepCounter = 0; stepCounter < steps; stepCounter++)
    {
        digitalWrite(5, HIGH);
        delayMicroseconds(stepdelay);
        digitalWrite(5, LOW);
        delayMicroseconds(stepdelay);
    }
    digitalWrite(6, HIGH);       // Disable
}

void Step1MinuteCCW()
{
    // move stepper for 1 Minute counterclockwise
    digitalWrite(6, LOW);          // Enable
    digitalWrite(4, HIGH);         // im gegenUhrzeigersinn
    for (stepCounter = 0; stepCounter < steps; stepCounter++)
    {
        digitalWrite(5, HIGH);
        delayMicroseconds(stepdelay);
        digitalWrite(5, LOW);
        delayMicroseconds(stepdelay);
    }
    digitalWrite(6, HIGH);         // Disable
}
```

3.4.2 Initial Clock Seetings

```
void SetClock()
{
    // Set Clock to exact time
    // Only once, after the time is set, the RTC keeps the time through battery power
    // Set Clock
    hour = 14;
    minute = 54;
    second = 0;
    myRTC.setClockMode(true);     // set to 12h
    myRTC.setHour(hour);
    myRTC.setMinute(minute);
    myRTC.setSecond(second);

}
```

3.4.3 Interrupt Handler

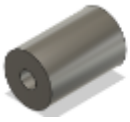
```
// Interrupt Handling to adjust Time
void int18()
{
    if (int18on)
    {
        // Let minute start at 0 Seconds
        myRTC.setSecond(0);
        Step1MinuteCW();
        delay(1000);
    }
    int18on = false;
}

void int19()
{
    if (int19on)
    {
        Step1MinuteCCW();
        delay(1000);
    }
    int19on = false;
}
```

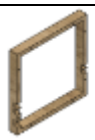
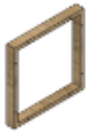
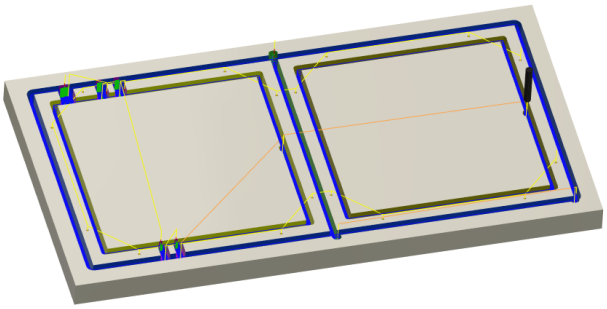
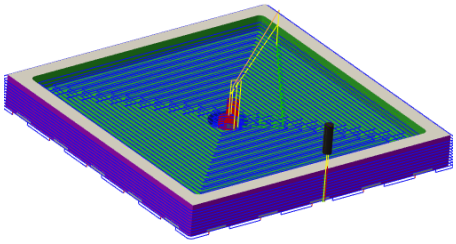
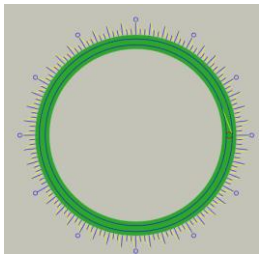
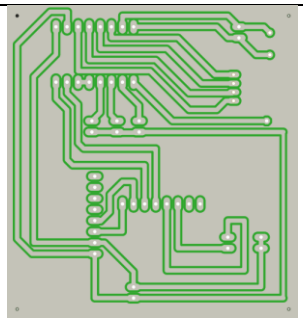
4 BOM

Level	Part Number	Q	Type
1	Zifferblatt	1	Part
1.1	Lager26x10h8	1	Part
2	MagnetRotor	1	Part
2.1	Magnet8x5	1	Part
3	Lager22x8h7	1	Part
4	Pololu A4899	1	Part
5	User Library-DS3231 Mini	1	Assembly
6	Board	1	Part
7	User Library-ARDUINO_MEGA_2560	1	Assembly
8	Kugel	1	Part
9	B4B-PH-K-S	1	Part
10	PushBtn	2	Part
11	DC socket	1	Part
12	Scheibe	1	Part
13	Magnet8x5	1	Part
14	AntriebSchneckeN14	1	Assembly
14.1	SchneckeRad	1	Assembly
14.1.1	32101800 Rad	1	Part
14.1.2	32111800 Schnecke	1	Part
14.2	ZeigerWelle	1	Part
14.3	WelleN14	1	Part
14.4	Lager10x5h4	2	Part
14.5	Nema 14	1	Part
15	Buegel14	1	Assembly
15.1	Platte1	2	Part
15.2	Nema14Halter	1	Part
15.3	Platte2	2	Part
15.4	NemaLagerHalter	1	Part
15.5	Steg	1	Part
15.6	Schrauben	1	Assembly
15.6.1	Stainless Steel Socket Head Screw	12	Part
16	RueckwandH2	1	Part
17	RueckwandH1	1	Part
18	PlatinenHalter	6	Part
19	DC Halter	1	Part

4.1 Print Parts

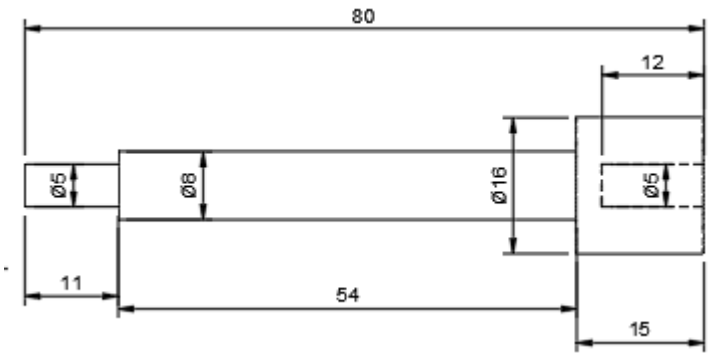
 Buegel14 8/13/22	Bügel für Schrittmotor und Getriebe Lagerung/Halterung
 Steg 8/9/22	
 Nema14Halter 8/13/22	
 NemaLagerHalter 8/1/22	
 Platte1 8/1/22	
 Platte2 8/9/22	
 DC Halter 8/17/22	Halterung für DC Jack Vmot
 PlatinenHalter 8/16/22	Halterung für Driver Board und Arduino

4.2 CNC Parts

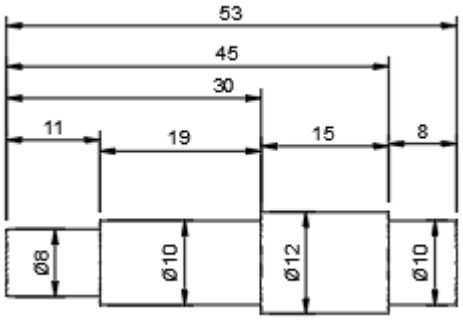
 Scheibe 8/18/22	Plexi Scheibe als Rückwand Abdeckung
 RueckwandH2 8/17/22  RueckwandH1 8/17/22	 Rückwand Gehäuseteile mit Eingängen für Arduino und Interrupt Switches
 Zifferblatt 8/15/22	  Zifferplatt mit Laserbeschriftung
 Board 8/8/22	 PCB Platine Kupfer einseitig beschichtet 35um, Leiterbahnen gefräst

4.3 Dreh Parts

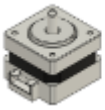
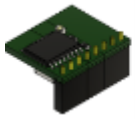
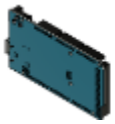
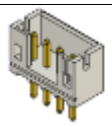
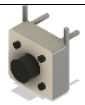
WelleN14
8/13/22



ZeigerWelle
8/1/22



4.4 Kauf Parts

 SchneckeRad 8/4/22	1:18 Schneckengetriebe selbsthemmend
 Nema 14 8/10/22	Nema 14 Schrittmotor
 Kugel 8/3/22  Magnet8x5 7/31/22	Kugel und Permanentmagnet
 Lager10x5h4 8/1/22  Lager22x8h7 8/1/22  Lager26x10h8 7/31/22	Diverse Lager
 Pololu A4899 7/31/22	Stepper Driver
 User Library-DS3231 8/2/22	Real Time Clock RTC DS3231
 User Library-ARDUINO_MEGA 8/3/22	Arduino Mega 2560 Rev 3
 B4B-PH-K-S 8/4/22	JST PH Stecker 2mm Pinweite
 DC socket 8/7/22	DC Socket für Vmot Eingang
 PushBtn 1:46:37 PM	Push Button für Interrupthandling