Experiment Realtime-Clock über I2C

A.Schultze, DK4AQ/23.05.2013

Aufgabe:

Das Uhrenchip DS1307 soll per I2C auf feste Startwerte initialisiert werden. Danach soll per seriellem Monitor fortlaufend
- Datum (Jahr / Monat / Tag)
- Uhrzeit (Std. / Min. / Sek.)

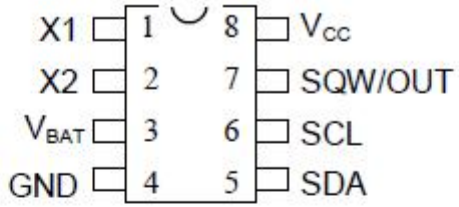
angezeigt werden.

Variante a:
Die Initialisierung auf feste Startwerte findet nur statt, wenn zwischwendurch die Batteriespannung für die Uhr ausgefallen ist. Das kann durch Nutzung der RAM-Zellen in der Uhr festgestellt werden.

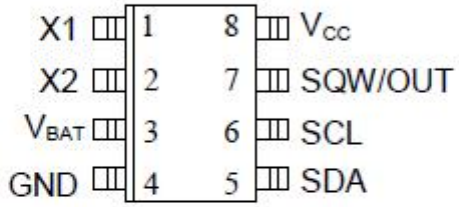
Variante b:
Es soll eine Stelleinrichtung eingebaut werden (über 2 Tasten oder Einlesen des Seriellen Monitors)

Variante c:
Weckerfunktion hinzufügen (Ton aus Piezo-Geber mit tone() oder mit Serial.write(0x07) an serielles Terminal-Programm (geht nicht mit Arduinos Serial Monitor, aber mit Hyperterminal oder puTTY-Terminal)

PIN ASSIGNMENT



DS1307 8-Pin DIP (300 mil)



DS1307Z 8-Pin SOIC (150 mil)

PIN DESCRIPTION

V_{CC}	- Primary Power Supply
X1, X2	- 32.768 kHz Crystal Connection
V_{BAT}	- +3V Battery Input
GND	- Ground
SDA	- Serial Data
SCL	- Serial Clock
SQW/OUT	- Square wave/Output Driver

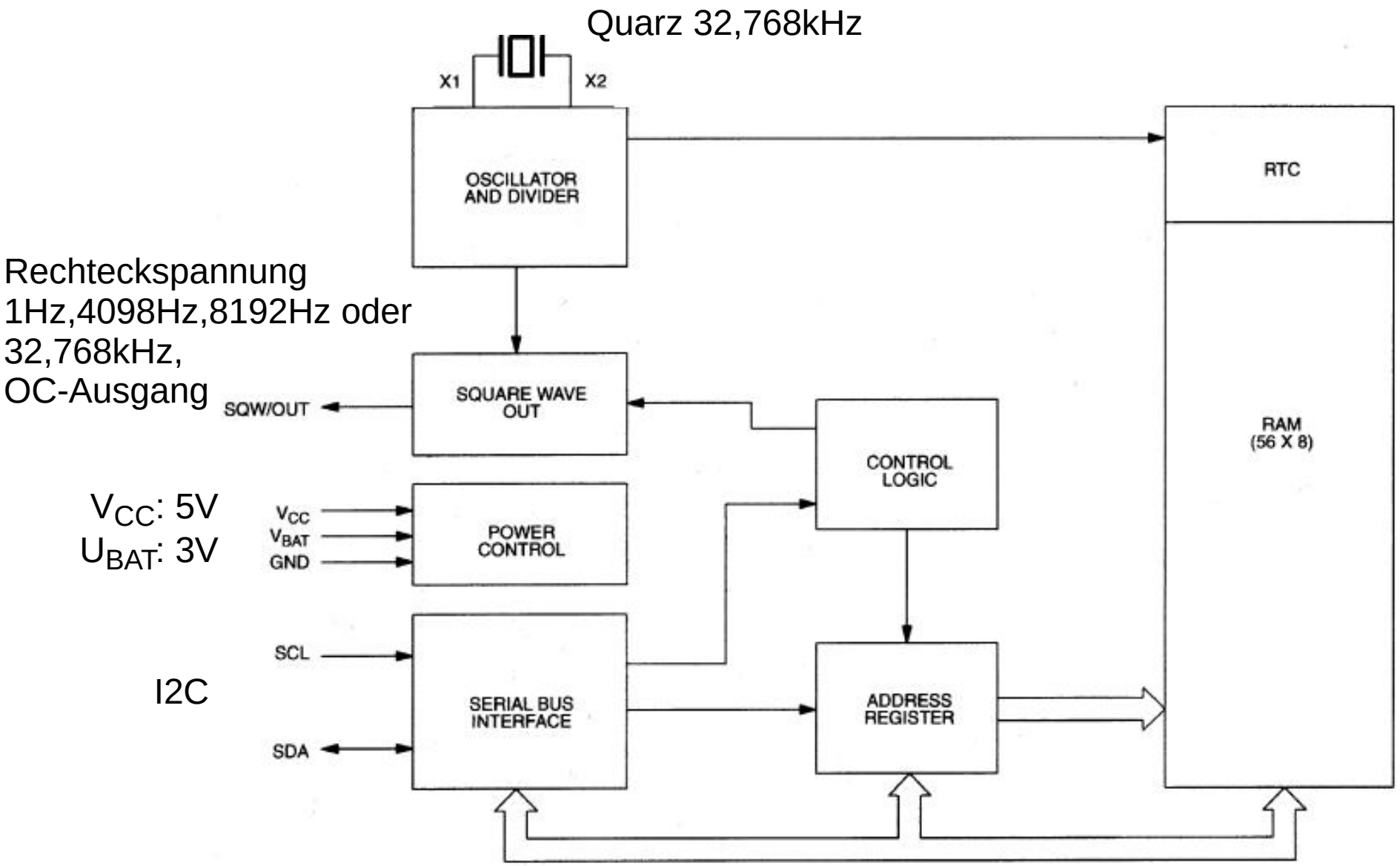
DS 1307

FEATURES

- Real time clock counts seconds, minutes, hours, date of the month, month, day of the week, and year with leap year compensation valid up to 2100
- 56 byte nonvolatile RAM for data storage
- 2-wire serial interface
- Programmable squarewave output signal
- Automatic power-fail detect and switch circuitry
- Consumes less than 500 nA in battery backup mode with oscillator running
- Optional industrial temperature range -40°C to +85°C
- Available in 8-pin DIP or SOIC
- Recognized by Underwriters Laboratory

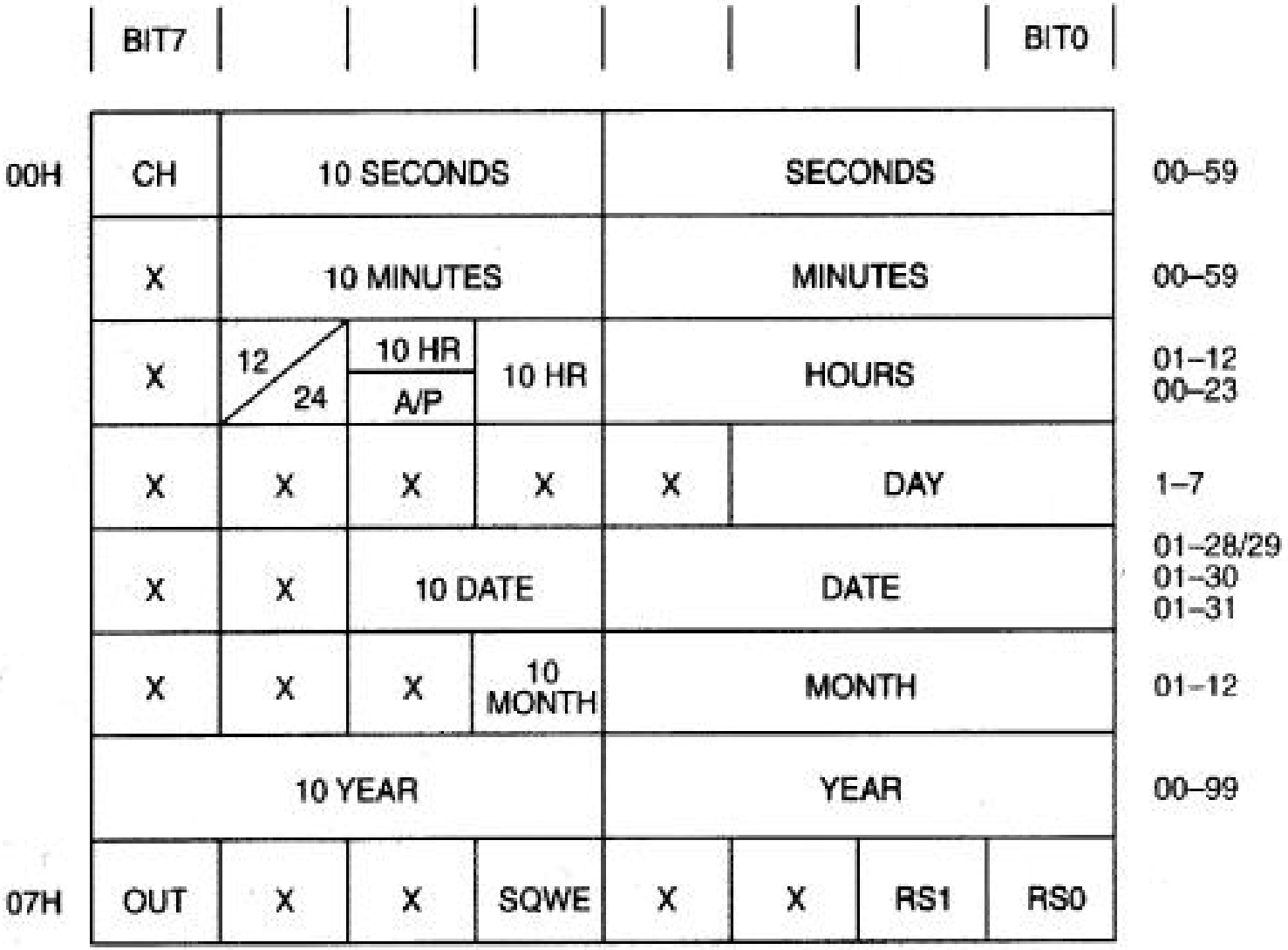
V_{CC}: 5V, U_{BAT}: 3V, I_{BAT} = 500nA
Externer Quarz: 32,768kHz
I2C-Geschwindigkeit: max. 100kb/s

Hardware-Struktur DS1307



DS1703 Timekeeper-Register

A.Schultze, DK4AQ



	BIT7						BIT0	
00H	CH	10 SECONDS			SECONDS			00–59
	X	10 MINUTES			MINUTES			00–59
	X	12/24	10 HR A/P	10 HR	HOURS			01–12 00–23
	X	X	X	X	X	DAY		1–7
	X	X	10 DATE		DATE			01–28/29 01–30 01–31
	X	X	X	10 MONTH	MONTH			01–12
	10 YEAR				YEAR			00–99
07H	OUT	X	X	SQWE	X	X	RS1	RS0

Aus diesen Registern wird Datum und Uhrzeit gelesen bzw. beim Einstellen geschrieben.
Register 07 enthält das Control-Register.

DS1703 Control-Register

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
OUT	X	X	SQWE	X	X	RS1	RS0

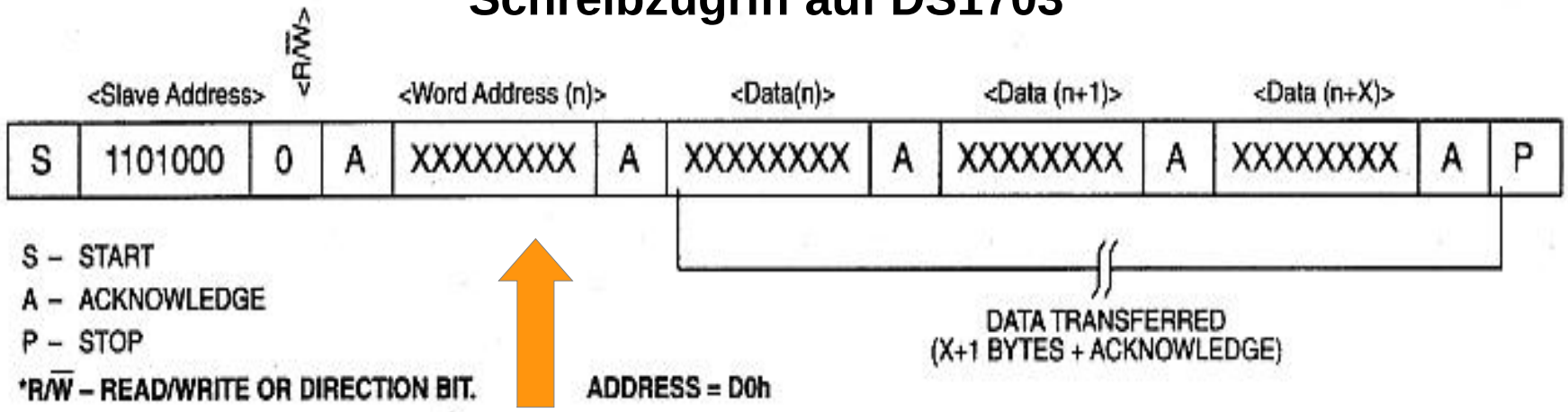
OUT: Output Control, gibt den Wert an Pin SQW/OUT beim Disable durch SQWE vor..

SQWE: Square Wave Enable-Signal für die Rechteckausgangsspannung an Pin SQW/OUT.

RS1, RS0: Rate Select, Einstellung der Frequenz am Pin SQW/OUT.

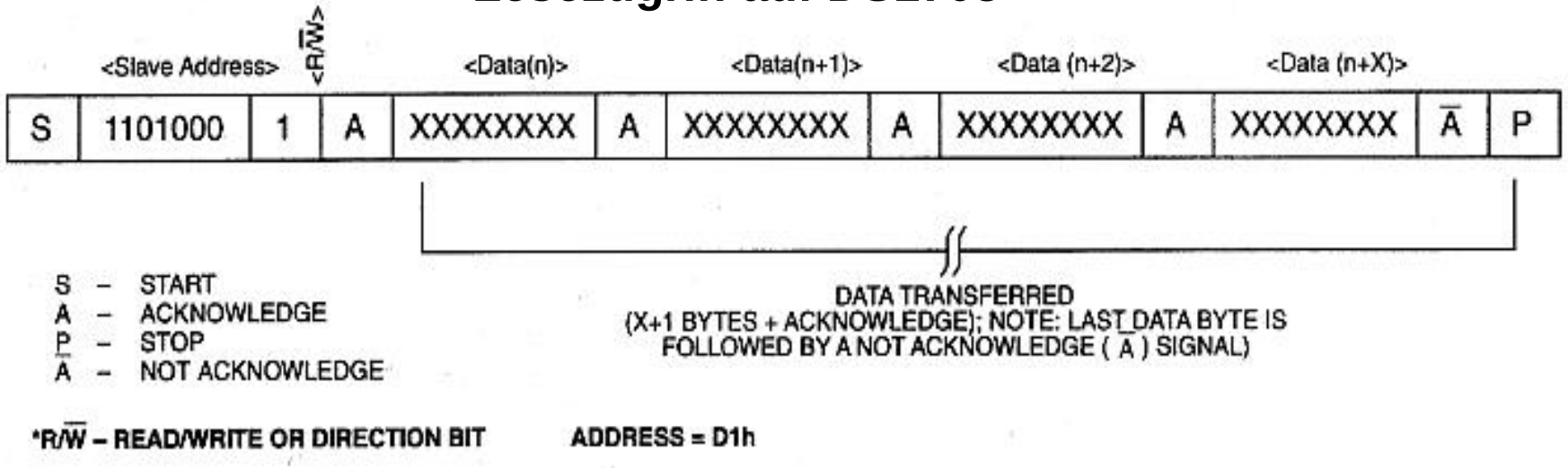
RS1	RS0	SQW OUTPUT FREQUENCY
0	0	1 Hz
0	1	4.096 kHz
1	0	8.192 kHz
1	1	32.768 kHz

Schreibzugriff auf DS1703

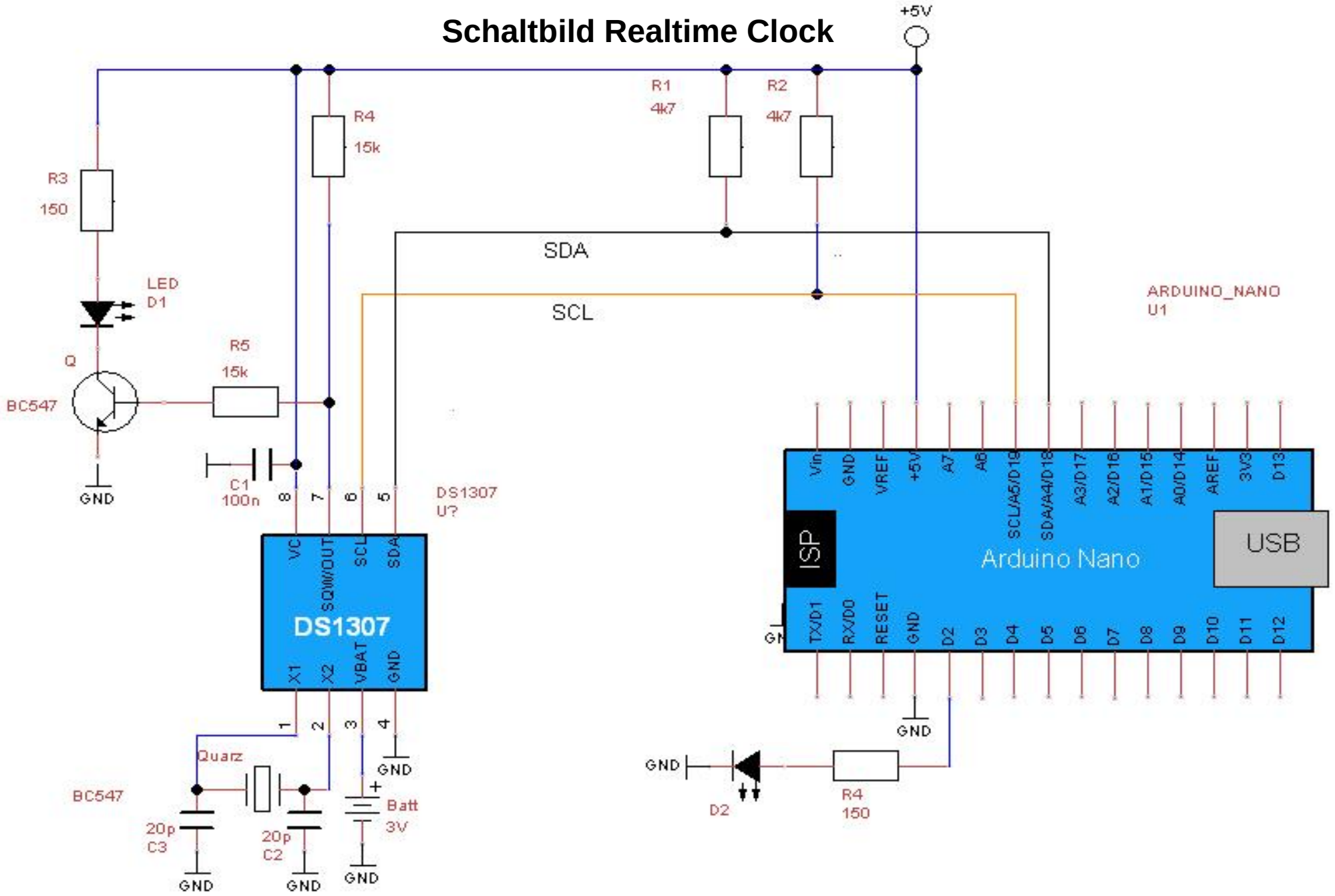


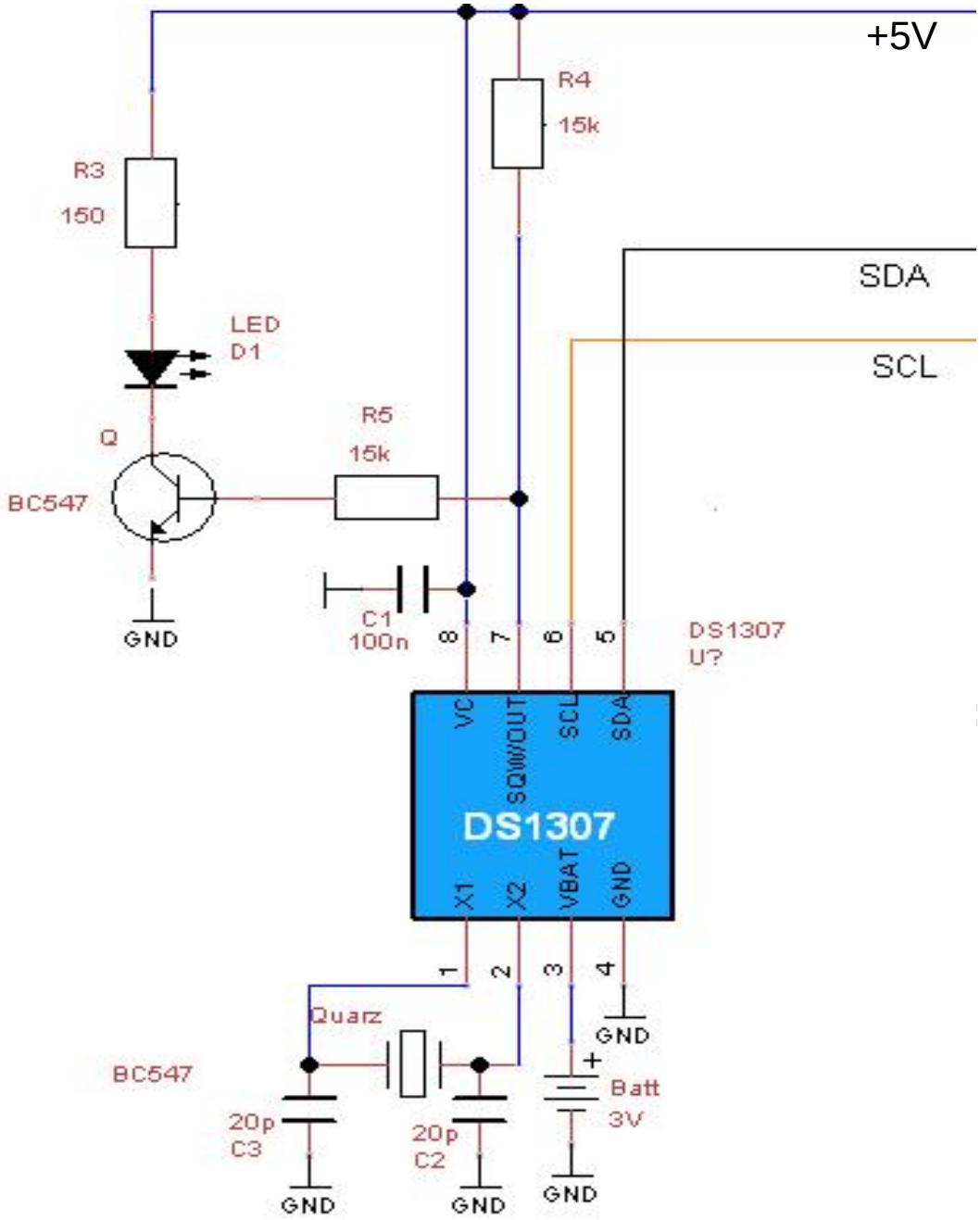
Register-Pointer, Beginn der Übertragung bei dieser internen Adresse, wird auch mit einem Schreibbefehl vor Lesen angewendet!

Lesezugriff auf DS1703

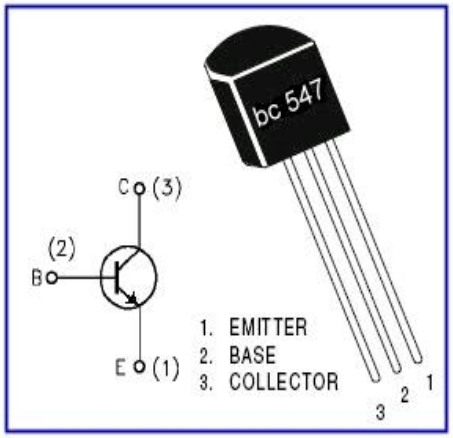


Schaltbild Realtime Clock

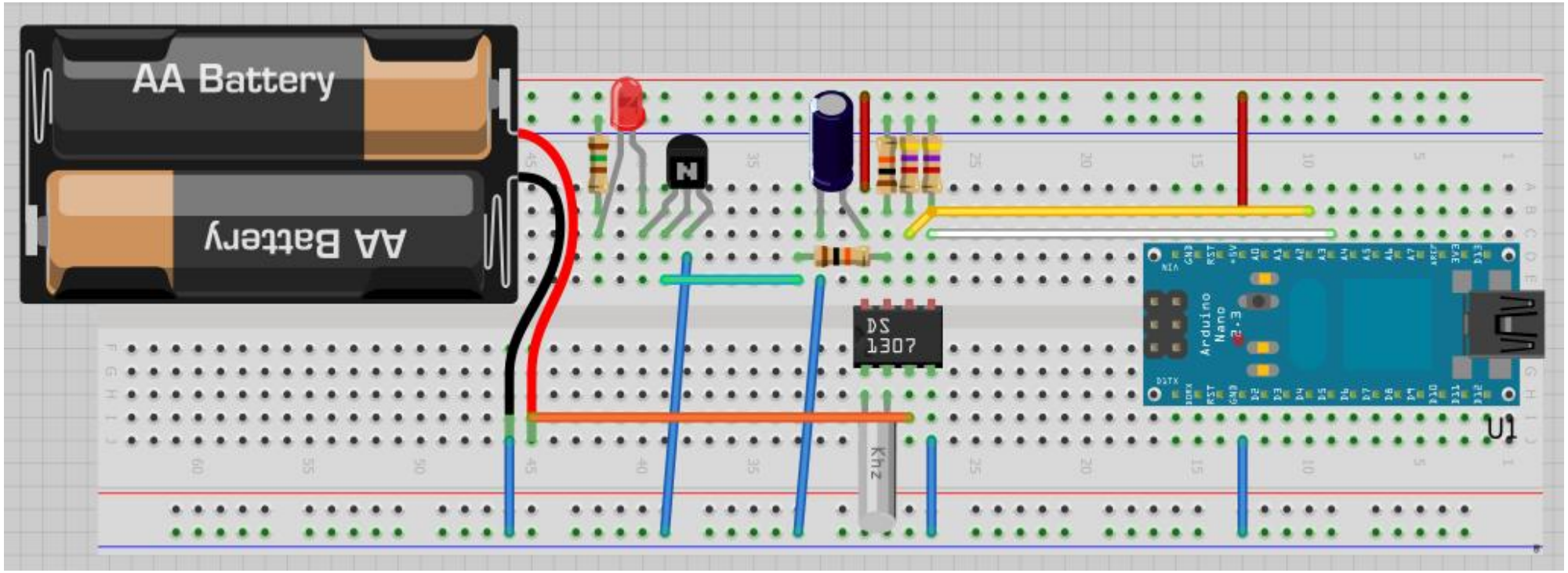




Beschaltung des DS1307



Aufbauvorschlag Realttime Clock



I2C_RealtClock_EXP4,
Definition Konstanten und Variablen

```
//I2C_RT_Clock_Exp4a mit DS1307, A.Schultze, DK4AQ
#include <Wire.h>

#define FALSE false
#define TRUE true

#define CLOCK_ADR 0x68
#define ledPin 2

//Startwert Uhr (einstellen)
#define SECONDS_START 0
#define MINUTES_START 0
#define HOURS_START   13
#define DAY_START 6
#define DATE_START 25
#define MONTH_START 05
#define YEAR_START 13
#define ANZ_MODE B01000000
#define CONTR_REG 0x10
```

```
byte seconds = 0;
byte minutes = 0;
byte hours_mode = 0;
byte hours = 0;
byte day = 0;
byte date = 0;
byte month = 0;
byte year = 0;
byte control_reg = 0;

boolean fehler = FALSE;
```

I2C_RealtClock_EXP4,
Setup() Teil 1

```
void setup()
{
  // Verwendete Bibliotheken
  Wire.begin();
  Serial.begin(9600);

  Serial.println("Start Uhr");
  // Ausgang für LED deklarieren
  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin,LOW);

  // Umsetzung der numerischen Werte in BCD-Werte
  byte seconds = decToBcd(SECONDS_START);
  byte minutes = decToBcd(MINUTES_START);
  byte hours_mode = decToBcd(HOURS_START)| ANZ_MODE;
  byte day = decToBcd(DAY_START);
  byte date = decToBcd(DATE_START);
  byte month = decToBcd(MONTH_START);
  byte year = decToBcd(YEAR_START);
  byte control_reg = CONTR_REG; // SQW-Ausgang aktiv auf 1Hz
```

I2C_RealtClock_EXP4, Setup() Teil 2

```
  byte control_reg = CONTR_REG; // SQW-Ausgang aktiv auf 1Hz

  Wire.beginTransmission(CLOCK_ADR); // Vorbereiten Schreiben

  Wire.write(0);              //Register-Pointer auf Reg.0
  Wire.write(seconds);        // Sende-Buffer 0= Sekunden Reg.adr 0
  Wire.write(minutes);        // Sende-Buffer 1= Minuten Reg.adr 1
  Wire.write(hours_mode);     // Sende-Buffer 2= Stunden+Modus Reg.adr 2
  Wire.write(day);            // Sende-Buffer 3= Wochentag Reg.adr 3
  Wire.write(date);           // Sende-Buffer 4= Datum Tag Reg.adr 4
  Wire.write(month);          // Sende-Buffer 5= Datum Monat Reg.adr 5
  Wire.write(year);           // Sende-Buffer 6= Datum Jahr Reg.adr 6
  Wire.write(control_reg);// Sende-Buffer 7= Control RFegister Reg.adr 7

  Wire.endTransmission(); // Start, Telegramm absenden, Stop

  Serial.println("Uhr gesetzt");
  Serial.write(0x07);

}
```

I2C_RealtClock_EXP4, loop() Teil 1

```
void loop()
{

  // Reset Register-Pointer
  Wire.beginTransmission(CLOCK_ADR);
  Wire.write(0);
  Wire.endTransmission();

  // Satz Werte einlesen
  fehler = FALSE;
  Wire.requestFrom(CLOCK_ADR, 7);

  // Einlesen mit Ausmaskierung der Control-Bits

  seconds = bcdToDec(Wire.read() & 0x7f);
  minutes = bcdToDec(Wire.read());
  hours = bcdToDec(Wire.read() & 0x3f);
  day = bcdToDec(Wire.read());
  date = bcdToDec(Wire.read());
  month = bcdToDec(Wire.read());
```

I2C_RealtClock_EXP4, loop() Teil 2

```
if(Wire.available())
{
    year = bcdToDec(Wire.read());
}
else
{
  fehler = TRUE;
}

if(fehler == TRUE)
{
  digitalWrite(ledPin,HIGH);
  Serial.println("keine Verbindung!");
}
else
{
  digitalWrite(ledPin,LOW);
  Serial.print("Datum: ");
  Serial.print(date);
  Serial.print(".");
```

I2C_RealtClock_EXP4, loop() Teil 3

```
    Serial.print(month);
    Serial.print(".20");
    Serial.print(year);

    Serial.print(" Zeit: ");
    Serial.print(hours);
    Serial.print(":");
    Serial.print(minutes);
    Serial.print(":");
    Serial.print(seconds);
    Serial.println(" ");

  }

  delay(500);
}
```

I2C_RealtClock_EXP4, loop() Teil 3

```
    Serial.print(month);
    Serial.print(".20");
    Serial.print(year);

    Serial.print(" Zeit: ");
    Serial.print(hours);
    Serial.print(":");
    Serial.print(minutes);
    Serial.print(":");
    Serial.print(seconds);
    Serial.println(" ");

  }

  delay(500);
}
```

I2C_RealtClock_EXP4, BcdToDec() und decToBcd()

```
// Convert binary coded decimal to normal decimal numbers
byte bcdToDec(byte val)
{
return ( (val/16*10) + (val%16));
}
```

```
// Convert normal decimal numbers to binary coded decimal
byte decToBcd(byte val)
{
return ( (val/10*16) + (val%10));
}
```