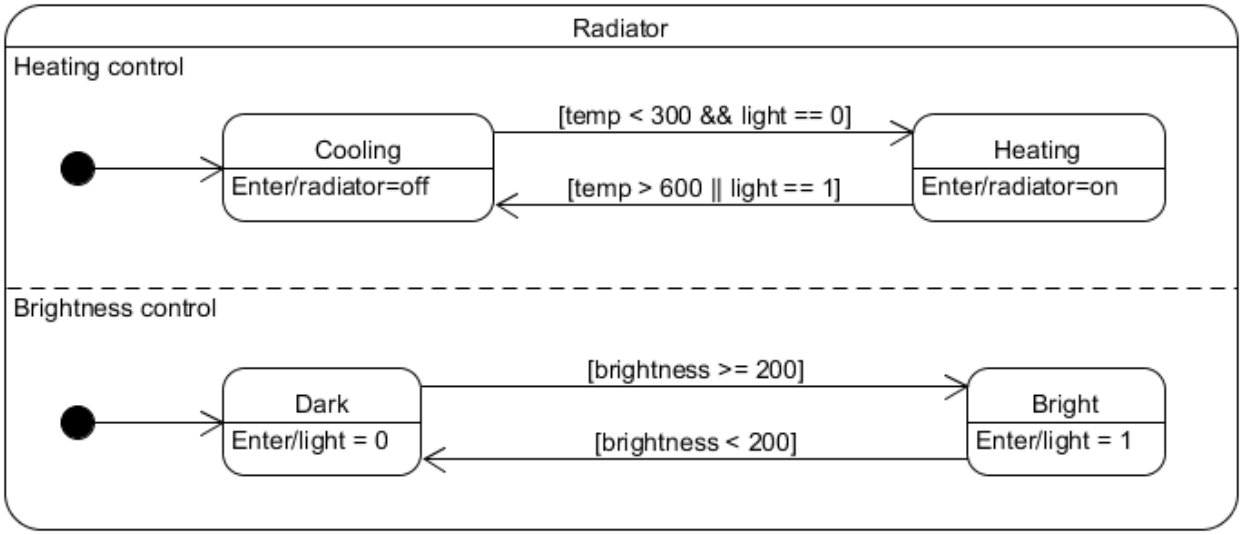
Embedded Systems - Praktikum 10

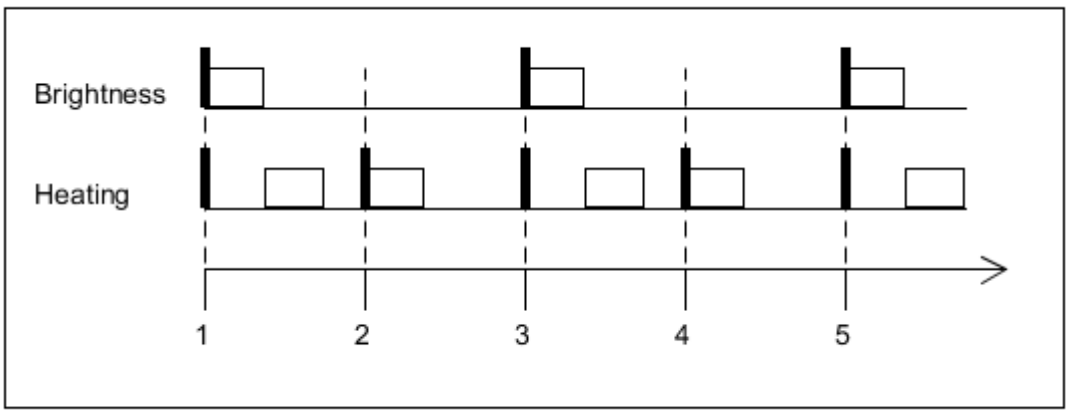
Malte Riechmann, André Kirsch

Aufgabe 1

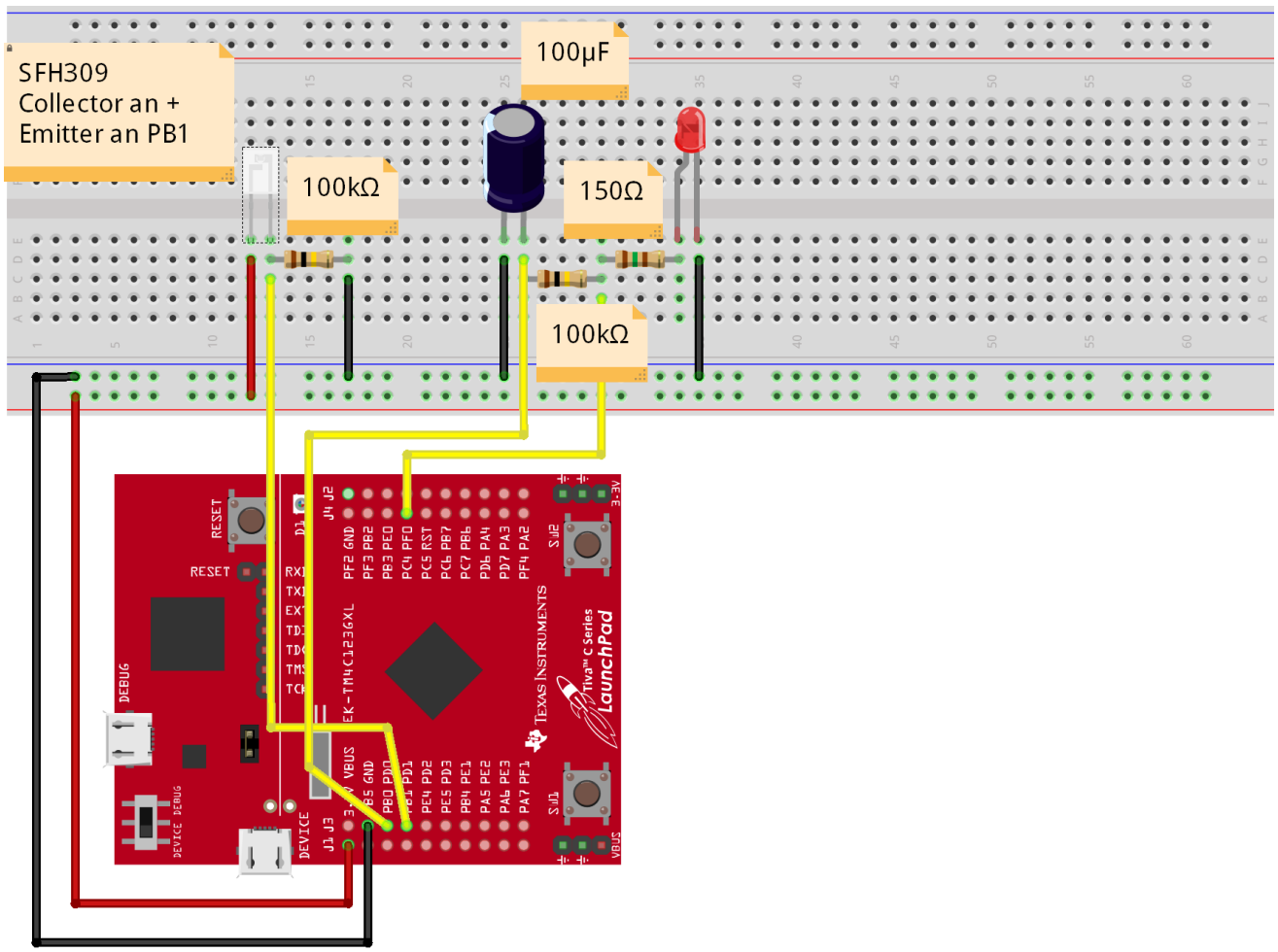
Sequenzdiagramm



Zeitverlaufsdiagramm



Aufgabe 2



Die zusätzliche LED haben wir parallel zum Kondensator angeschlossen, sodass sie leuchtet, wenn der Kondensator aufgeladen wird.

Aufgabe 3

Wir haben uns dazu entschieden, den Code wie in Energia aufzubauen mit einer *setup* und einer *loop* Funktion für jede Task. In den *setup* Funktionen setzen wir die Pin Modi und aktivieren die serielle Ausgabe:

```
void setupHeatingControl() {
    Serial.begin(9600);
    pinMode(tempPin, INPUT);
    pinMode(heatingPin, OUTPUT);
}

void setupBrightnessControl() {
    Serial.begin(9600);
    pinMode(brightnessPin, INPUT);
}
```

In den *loop* Funktionen wechseln wir zwischen den States, die in einem *enum* dargestellt werden, wenn die Bedingungen, wie sie auch im Statechart aus Aufgabe 1 zu sehen sind, erfüllt sind.

```
void loopHeatingControl() {
    switch(stateH) {
    case COOLING:
      if(analogRead(tempPin) < lowTemp && !light) {
          digitalWrite(heatingPin, HIGH);
          stateH = HEATING;
      }
      break;
    case HEATING:
      if(analogRead(tempPin) > highTemp || light) {
          digitalWrite(heatingPin, LOW);
          stateH = COOLING;
      }
      break;
    }
    Serial.print("HeatingPin: ");
    Serial.println(analogRead(tempPin));
}

void loopBrightnessControl() {
    switch(stateB) {
    case DARK:
      if(analogRead(brightnessPin) >= changeBrightness) {
          light = true;
          stateB = BRIGHT;
      }
      break;
    case BRIGHT:
      if(analogRead(brightnessPin) < changeBrightness) {
          light = false;
          stateB = DARK;
      }
      break;
    }
    Serial.print("BrightnessPin: ");
    Serial.println(analogRead(brightnessPin));
}
```

Die *setup* Funktionen rufen wir direkt in der *main* Funktion auf. Die *loop* Funktionen werden in einer Endlosschleife in einer *"task"* Funktion aufgerufen. Dort rufen wir auch die Delay Funktion aus *freeRTOS* auf. Diese Funktionen übergeben wir dann in der *main* Funktion den *xTaskCreate* Funktionen, in denen wir auch die Priorität setzen.

```
void taskBrightnessControl(void *pvParameters) {
    for (;;) {
        loopBrightnessControl();
        if (serialEventRun)
            serialEventRun();
        vTaskDelay(brightnessWaitTime);
    }
}

void taskHeatingControl(void *pvParameters) {
    for (;;) {
        loopHeatingControl();
        if (serialEventRun)
            serialEventRun();
        vTaskDelay(heatingWaitTime);
    }
}


int main(void) {
    setupBrightnessControl();
    setupHeatingControl();

    xTaskCreate(taskHeatingControl, "Heating Control", configMINIMAL_STACK_SIZE + 100, NULL,
tskIDLE_PRIORITY + 1UL, NULL);
    xTaskCreate(taskBrightnessControl, "Brightness Control", configMINIMAL_STACK_SIZE + 100,
NULL, tskIDLE_PRIORITY + 2UL, NULL);
    vTaskStartScheduler();
}
```