

**LAPORAN PROYEK FINAL
JAM DOT MATRIX**



Dibuat Oleh:
Reza Ali Nirwansyah 5024211001

Dosen Pengampu:
Eko Pramunanto, S.T., M.T. 19661203199412 1 001

**DEPARTEMEN TEKNIK KOMPUTER
FAKULTAS TEKNOLOGI ELEKTRO DAN INFORMATIKA CERDAS
INSTITUT TEKNOLOGI SEPULUH NOPEMBER SURABAYA**

2023

A. GAMBARAN UMUM

Jam merupakan sebuah alat yang memiliki fungsi utama yaitu untuk menunjukkan waktu. Selain memiliki fungsi utama, pada umumnya jam memiliki fungsi tambahan. Diantaranya adalah Jam yang dipergunakan sebagai alarm, jam yang dapat difungsikan sebagai hiasan, kemudahan akses, pengingat dan sebagainya. Semua fungsi yang disebutkan sebelumnya dapat diterapkan pada lingkungan yang sesuai, agar fungsi dari jam tersebut lebih bermanfaat.

Perangkat yang akan dibuat merupakan jam dot matrix. Jam dot matrix merupakan sebuah perangkat yang menampilkan waktu dalam bentuk dot matrix, di mana waktu ditampilkan menggunakan titik-titik (dot) yang membentuk angka dan karakter pada suatu matriks. Matriks tersebut biasanya terdiri dari beberapa baris dan kolom titik yang dapat diatur untuk membentuk karakter dan angka tertentu.

Selain dapat menunjukkan waktu, Jam tersebut dapat menunjukkan suhu ruangan, tanggal, dan juga alarm. Pengguna dapat mengatur jam, tanggal dan alarm cukup dengan menggunakan 3 tombol saja.

B. SPESIFIKASI SISTEM

Adapun spesifikasi dari sistem yang disusun, spesifikasi yang disusun mencakup spesifikasi operasional, fungsional, dan perangkat keras. Ketiga poin tersebut akan dijelaskan pada penjelasan dibawah ini:

a. Spesifikasi Fungsional

Berikut ini merupakan spesifikasi fungsional dari jam dot matrix:

i. Antarmuka pengguna:

Jam dot matrix dirancang dengan antarmuka pengguna yang mudah digunakan, memanfaatkan tombol-tombol navigasi seperti tombol set, tombol up, dan tombol down, serta dot matrix display untuk menyajikan informasi waktu, tanggal, dan suhu secara intuitif. Pengguna Dapat berinteraksi dengan jam dan mengakses fungsi-fungsi utama dengan mudah

ii. Dot Matrix Display

Fungsi utama dari dot matrix display adalah menampilkan informasi waktu, tanggal, dan suhu secara jelas dan terbaca. Display ini memberikan tampilan yang informatif, memungkinkan pengguna untuk dengan cepat membaca dan memahami kondisi jam serta parameter lainnya.

iii. Buzzer

Buzzer pada jam dot matrix berperan penting dalam mendukung fungsi alarm. Dengan adanya buzzer, pengguna

dapat mengatur dan mengaktifkan alarm, dan saat alarm berbunyi, buzzer memberikan output berupa suara yang jelas untuk memberikan peringatan kepada pengguna.

iv. **Tombol dan Navigasi**

Tombol-tombol navigasi, termasuk tombol set, tombol up, dan tombol down, memberikan pengguna kemampuan untuk berpindah antara mode jam, menyesuaikan waktu, tanggal, dan alarm. Tombol-tombol ini dirancang untuk memudahkan pengguna dalam mengatur dan menavigasi fungsi-fungsi jam.

v. **Mode Jam**

Jam dot matrix menyediakan beberapa mode operasional, seperti mode waktu, mode penyetelan (adjust), dan mode alarm. Pengguna dapat beralih antara mode-mode ini sesuai kebutuhan mereka, memungkinkan fungsionalitas yang lebih luas dan adaptasi jam sesuai preferensi pengguna.

vi. **Penyetelan Waktu, Tanggal, dan Alarm**

Proses penyetelan waktu, tanggal, dan alarm dijelaskan dengan langkah-langkah yang jelas. Petunjuk yang diberikan memberikan panduan yang mudah diikuti, memastikan pengguna dapat dengan cepat dan akurat mengatur parameter-parameter tersebut.

b. **Spesifikasi Operasional**

Berikut adalah spesifikasi operasional dari jam dot matrix:

i. **Antarmuka Pengguna**

Jam dot matrix akan menyediakan antarmuka pengguna yang memungkinkan pengguna berinteraksi dengan jam. Antarmuka yang disediakan pada jam ini adalah

1. **Dot Matrix Display**

Dot matrix digunakan untuk menampilkan berbagai macam output. Output yang diberikan tergantung kondisi state yang sedang terjadi saat itu. State yang dimiliki pada jam ini adalah, RUN, ADJUST_TIME, ADJUST_DATE, ADJUST_ALARM, TRIGGER_ALARM

2. **Tombol dan Navigasi**

Pengguna dapat memberikan Input kepada jam menggunakan tombol yang disediakan. yaitu set, up, dan down. masing masing tombol memiliki fungsinya masing-masing

a. **Tombol Set**

Tombol ini adalah tombol yang banyak guna. dengan tombol ini, pengguna dapat berpindah-pindah state, ganti menu, dan

pilihan tergantung kombinasi penekanan tombol. Kombinasi itu diantaranya adalah

i. **Ditekan dan tahan (Hold)**

Ketika aksi ini dilakukan, maka secara umum akan mengganti tampilan, dari kondisi RUN yang menampilkan jam, menuju menu ADJUST.

ii. **Ditekan dua kali (Double Click)**

Tombol ini akan beraksi ketika posisi jam pada menu adjust. Aksi yang terjadi setelah melakukan itu adalah mengganti menu adjust, dari ADJUST_TIME menuju ADJUST_DATE, kemudian ADJUST_ALARM

iii. **Ditekan sekali (Single Click)**

Tombol ini memberikan aksi untuk berpindah posisi dalam suatu menu adjust. Ketika posisi sedang mengatur jam, pengguna ingin mengganti menit di menu adjust yang sama, pengguna cukup menekan tombol set sekali.

Selain untuk mengatur pilihan, tombol ini digunakan untuk mematikan alarm, ketika alarm sedang menyala.

b. **Tombol Up/Down**

Tombol ini memiliki fungsi yang mirip, hanya berbeda arah saja. Fungsi dari tombol ini adalah mengatur, mengubah pilihan pada menu adjust, seperti ingin mengatur jam, menit, dan detik. pada tombol up/down bisa dilakukan secara sekali tekan dan tahan. ketika single click, pergeseran hanya 1 kali, sedangkan ketika ditahan, pergeseran akan terus dilakukan hingga tombol itu dilepas.

ii. **Buzzer**

Perangkat ini diperlukan untuk menjalankan fungsi alarm. ketika alarm terpanggil, maka buzzer akan menyala terus hingga tombol set ditekan baru buzzer akan berhenti menyala

iii. **Pengaturan kecerahan layar otomatis**

Sensor LDR (Light Dependent Resistor) bekerja dengan

mendeteksi perubahan intensitas cahaya di sekitarnya. Ketika pencahayaan berubah, nilai resistansi pada sensor LDR berubah pula. Sistem menggunakan nilai resistansi ini untuk mengukur intensitas cahaya, dan berdasarkan perubahan tersebut, secara otomatis mengatur kecerahan layar agar sesuai dengan kondisi pencahayaan yang baru. Dengan menggunakan nilai yang diperoleh dari sensor LDR, sistem dapat menentukan tingkat kecerahan layar yang optimal, menciptakan pengalaman pengguna yang nyaman dan efisien, serta mengoptimalkan penggunaan energi dengan menyesuaikan kecerahan sesuai dengan lingkungan sekitar.

iv. Penerimaan Suhu

Operasional penerimaan suhu melibatkan proses pengukuran dan penerimaan data suhu oleh sistem. Sensor suhu, seperti LM35, digunakan untuk mengukur suhu dalam lingkungan tertentu. Data suhu yang diukur akan diteruskan ke sistem, di mana nilai tersebut kemudian dapat digunakan untuk menampilkan suhu ruang pada tampilan dot matrix

v. RTC

RTC digunakan untuk menyimpan data waktu, bahkan ketika sistem tidak memiliki daya, karena RTC memiliki baterai internal yang dapat menjaga data waktu.

vi. Status Mesin dan Kondisi

Berikut ini merupakan Status Mesin dan/atau Kondisi yang terjadi pada jam dot matrix

1. RUN

Status RUN merupakan status utama yang ada pada jam ini. Pada posisi ini, Jam dot matrix akan menampilkan waktu dengan format jj:mm:dd (hh:mm:ss) dalam 24 jam. Kemudian pada detik ke 10-12 dan detik 40-42 dot matrix akan menampilkan suhu ruangan. dan pada detik 13-15 dan 43-45 dot matrix akan menampilkan tanggal hari tersebut.

Selain berfungsi untuk menampilkan waktu, suhu, dan tanggal. Status RUN akan selalu memeriksa alarm. apakah alarmnya kondisi aktif? apakah jamnya sama? apakah menitnya juga sama? jika ketiga kondisi tersebut terpenuhi, maka akan berpindah ke status TRIGGER_ALARM

2. ADJUST_TIME

Kondisi ADJUST_TIME memungkinkan pengguna untuk mengatur jam, menit, dan detik yang

ditampilkan dalam format jj:mm:dd. Pada kondisi ini, posisi yang dipilih untuk penyesuaian akan ditandai dengan garis bawah, memudahkan pengguna untuk menentukan elemen waktu. Untuk berganti posisi bisa dilakukan dengan menekan tombol set sekali. dan untuk mengatur elemen, bisa dengan menekan tombol up/down baik itu dengan cara sekali klik (pergeseran 1 kali) atau juga ditahan (pergeseran secara terus menerus selama ditahan). Untuk berpindah ke menu berikutnya, bisa menekan tombol set dua kali (double click) dan jika balik ke RUN bisa menekan dan menahan tombol SET.

3. ADJUST_DATE

Kondisi ADJUST_DATE memungkinkan pengguna untuk mengatur tanggal, bulan, dan tahun yang ditampilkan dalam format dd:mm:yy. Pada kondisi ini, posisi yang dipilih untuk penyesuaian akan ditandai dengan garis bawah, memudahkan pengguna untuk menentukan elemen tanggal. Untuk berganti posisi bisa dilakukan dengan menekan tombol set sekali. dan untuk mengatur elemen, bisa dengan menekan tombol up/down baik itu dengan cara sekali klik (pergeseran 1 kali) atau juga ditahan (pergeseran secara terus menerus selama ditahan). Untuk berpindah ke menu berikutnya, bisa menekan tombol set dua kali (double click) dan jika balik ke RUN bisa menekan dan menahan tombol SET.

4. ADJUST_ALARM

Kondisi ADJUST_ALARM memungkinkan pengguna untuk mengatur alarm yang ditampilkan dalam format

[no alarm][kondisi alarm (on/off)][jam][menit]. Pada kondisi ini, posisi yang dipilih untuk penyesuaian akan ditandai dengan garis bawah, memudahkan pengguna untuk menentukan elemen alarm. Untuk berganti posisi bisa dilakukan dengan menekan tombol set sekali. dan untuk mengatur elemen, bisa dengan menekan tombol up/down baik itu dengan cara sekali klik (pergeseran 1 kali) atau juga ditahan (pergeseran secara terus menerus selama ditahan). Untuk berpindah ke menu berikutnya, bisa menekan tombol set dua kali (double click) dan jika balik ke RUN bisa menekan dan menahan tombol SET.

5. TRIGGER_ALARM

Kondisi Trigger Alarm terjadi ketika kondisi alarm yang telah diatur oleh pengguna terpenuhi. Pada saat ini, sistem akan memberikan respons dengan menampilkan pesan alarm, mengaktifkan buzzer untuk memberikan pemberitahuan suara yang jelas, dan mengizinkan keluar dari posisi Trigger Alarm saat tombol set ditekan. Proses ini memungkinkan pengguna untuk dengan cepat merespons alarm, memberikan informasi yang diperlukan, dan mematikan alarm

c. Spesifikasi Perangkat Keras

Berikut adalah spesifikasi perangkat keras yang digunakan pada alat ini:

i. Dot Matrix

Dot matrix adalah jenis tampilan atau layar yang terdiri dari sekumpulan titik-titik (dot) kecil yang tersusun dalam bentuk matriks. Setiap titik pada dot matrix dapat diatur secara independen, memungkinkan pembentukan karakter, gambar, atau teks dengan mengendalikan keadaan tiap titik. Umumnya, dot matrix digunakan dalam berbagai perangkat elektronik, seperti layar pada printer dot matrix, panel pesan berjalan (scrolling message boards), atau tampilan pada jam digital. Kelebihan dot matrix adalah kemampuannya untuk menampilkan karakter dan grafis yang dapat dikonfigurasi, meskipun resolusi visualnya mungkin lebih rendah dibandingkan dengan teknologi tampilan modern seperti LCD atau LED.

ii. Real Time Clock (RTC)

RTC adalah singkatan dari Real-Time Clock, yang dalam bahasa Indonesia berarti Jam Pemantau Waktu Nyata. RTC merupakan suatu perangkat keras pada komputer atau sistem tertentu yang bertugas untuk menyediakan informasi waktu yang akurat secara terus-menerus, bahkan ketika sistem dimatikan. RTC biasanya dilengkapi dengan baterai kecil agar dapat terus beroperasi dan menyimpan informasi waktu meskipun daya utama sistem dimatikan. Fungsi RTC sangat penting dalam berbagai aplikasi, termasuk sistem operasi, log waktu, penjadwalan tugas, dan aplikasi yang memerlukan presisi waktu yang tinggi.

iii. Buzzer

Buzzer adalah suatu perangkat elektronik yang dirancang untuk menghasilkan suara atau bunyi tertentu. Buzzer umumnya terdiri dari membran getaran atau elemen piezoelektrik yang dapat menghasilkan getaran saat diberi tegangan listrik. Ketika tegangan diberikan pada buzzer, getaran yang dihasilkan oleh elemen tersebut menciptakan gelombang suara, menghasilkan bunyi yang bisa didengar. Buzzer digunakan dalam berbagai konteks, seperti perangkat alarm, sinyal notifikasi, permainan, dan proyek elektronika lainnya. Keberagaman jenis dan ukuran buzzer membuatnya sangat fleksibel untuk digunakan dalam berbagai aplikasi.

iv. **Light Dependent Resistor (LDR)**

LDR (Light Dependent Resistor) adalah jenis resistor yang resistansinya berubah berdasarkan intensitas cahaya yang diterimanya. LDR terbuat dari bahan semikonduktor khusus yang memiliki resistivitas yang berkurang ketika terkena cahaya. Ketika cahaya mengenai LDR, resistansinya menurun, dan sebaliknya, ketika dalam kegelapan, resistansinya meningkat. Hal ini membuat LDR sangat berguna dalam berbagai aplikasi, seperti sensor cahaya otomatis pada lampu malam, pengatur kecerahan layar pada perangkat elektronik, dan sistem otomatis lainnya yang merespons perubahan intensitas cahaya.

v. **LM35**

LM35 adalah sensor suhu analog yang dirancang untuk memberikan keluaran tegangan yang linear berdasarkan suhu dalam skala Celsius. Sensor ini memberikan presisi yang baik dan biasanya digunakan dalam berbagai aplikasi elektronika, termasuk sistem kontrol suhu, peralatan medis, dan pengukuran suhu industri. LM35 memberikan keluaran tegangan yang proporsional terhadap suhu dengan koefisien 10 mV per derajat Celsius. Keunggulan LM35 meliputi kemudahan penggunaan, akurasi yang tinggi, dan respons yang cepat terhadap perubahan suhu. Hal ini membuatnya populer di berbagai proyek yang melibatkan pemantauan atau kontrol suhu.

vi. **Button**

Button atau tombol adalah perangkat input yang digunakan untuk memasukkan sinyal atau perintah ke dalam suatu sistem. Tombol biasanya berbentuk kecil dan dapat ditekan atau diklik oleh pengguna. Ketika tombol ditekan, dapat menghasilkan sinyal elektrik atau sinyal lainnya yang

kemudian diinterpretasikan oleh perangkat atau sistem yang terhubung.

vii. Arduino Nano

Arduino Nano merupakan komponen mikrokontroler yang dapat digunakan untuk membaca input dari suatu komponen dan menghasilkan output untuk mengatur komponen lain. Arduino Nano menggunakan chip mikrokontroler ATmega328P, memiliki spesifikasi tegangan input sebesar 5V. Konfigurasi yang tertanam pada bootloader ATmega328P adalah sebagai berikut:

Fuse Bit	LOW		HIGH		EXTENDED	
	Value	Name	Value	Name	Value	Name
7	1	CKDIV8 Divide clock by 8 bit	1	RSTDISBL External reset disable		
6	1	CKOUT Clock output	1	DWEN debugWIRE Enable		
5	1	SUT1 Select start-up time	0	SPIEN Enable Serial programming and Data Downloading		
4	1	SUT0 Select start-up time	1	WDTON Watchdog Timer Always On		
3	1	CKSEL3 Select Clock Source	1	EESAVE EEPROM memory is preserved through chip erase		
2	1	CKSEL2 Select Clock Source	1	BOOTSZ1 Select boot size	1	BODLEVEL2 Brown-out Detector trigger level
1	1	CKSEL1 Select Clock Source	1	BOOTSZ0 Select boot size	0	BODLEVEL1 Brown-out Detector trigger level

0	1	CKSEL0 Select Clock Source	0	BOOTRST Select reset vector	1	BODLEVE L0 Brown-out Detector trigger level
---	---	--------------------------------------	---	---------------------------------------	---	---

* Value 0: programmed (aktif), 1 unprogrammed (tidak aktif)

Konfigurasi diatas mengaktifkan beberapa fitur seperti berikut ini:

- Menggunakan External Crystal Oscillator
- Frequency 8.0 MHz - 16.0 MHz
- Start up time PWRDWN/RESET: 16K CK/14 CK + 65 ms
- Boot Flash section size 256 words
- Boot reset vector enabled
- Boot start address 0x3F00
- Serial Program Downloading (SPI) enabled
- Brown Out Detection level at 2.7V

viii. Casing Box

Jam dot matrix memiliki pelindung berupa 3d print. sehingga memiliki bentuk yang indah, dan dapat melindungi komponen didalamnya

C. RANCANGAN

a. State Machine

i. RUN

kondisi untuk menampilkan jam, tanggal, suhu, dan check alarm

ii. ADJUST_TIME

State machine dimana pengguna dapat mengatur jam

iii. ADJUST_DATE

State machine dimana pengguna dapat mengatur tanggal

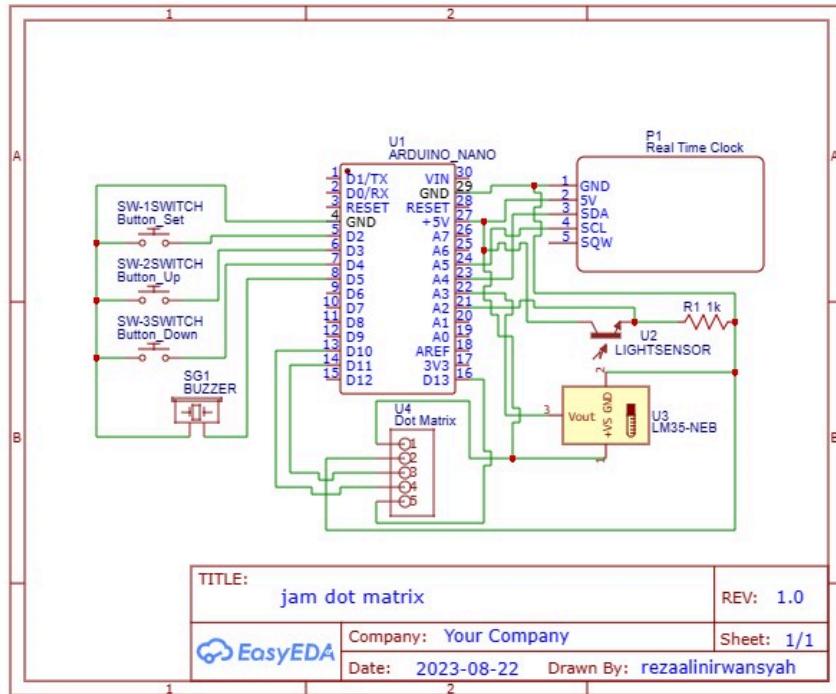
iv. ADJUST_ALARM

State machine dimana pengguna dapat mengatur alarm

v. TRIGGER_ALARM_X

State machine, ketika alarm aktif, maka akan membunyikan buzzer. dan menyala terus hingga pengguna mematikannya

b. Gambar Rangkaian



D. RENCANA OPERASI UJI

- Memeriksa semua komponen yang tersedia, memastikan bahwa semua komponen dapat berjalan dengan baik. apakah dot matrix bisa memberikan output, apakah sensor ldr bisa menerima perubahan cahaya, apakah sensor lm35 bisa bisa menerima suhu, apakah button dapat menerima input dengan baik, apakah rtc dapat memberikan waktu yang terus berjalan meski tidak diberi daya secara eksternal.
- Mencoba menampilkan Waktu dan membuat state RUN. mulai memeriksa sensor lm35, ldr, dan RTC.
- Mulai menambahkan State ADJUST_DATE. memeriksa apakah tombol bisa digunakan dengan baik, bisa berpindah state, dapat mengubah tanggal, dan mengganti pilihan.
- memastikan tombol bisa menerima input baik secara double, single, hold.
- Mulai menambahkan State ADJUST_TIME. memeriksa apakah tombol bisa digunakan dengan baik, bisa berpindah state, dapat mengubah waktu, dan mengganti pilihan.
- Menguji buzzer dan tampilan geser yang ditampilkan secara bersamaan
- Mulai menambahkan TRIGGER_ALARM, ketika kondisi terpenuhi, maka masuk ke state ini.
- Mulai menambahkan ADJUST_ALARM
- Memastikan semua fungsi berjalan dengan baik.

E. KESIMPULAN

Jam dot matrix yang akan dibuat memiliki fungsi utama sebagai penunjuk waktu, dengan tambahan kemampuan untuk menampilkan suhu ruangan, tanggal, dan alarm. Perangkat ini dapat diatur melalui penggunaan tiga tombol, memudahkan akses dan pengaturan oleh pengguna. Fungsionalitas yang beragam, seperti kemampuan menampilkan informasi suhu dan tanggal, menjadikan jam dot matrix ini sebagai perangkat yang multifungsi. Selain itu, desain tampilannya yang menggunakan titik-titik (dot) pada matriks memberikan fleksibilitas untuk membentuk karakter dan angka. Dengan demikian, jam ini tidak hanya berperan sebagai alat penunjuk waktu, tetapi juga memberikan nilai tambah dalam hal manajemen waktu dan informasi sehari-hari.

F. SARAN

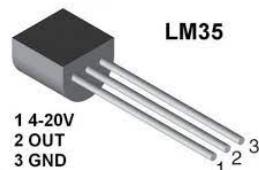
Saran untuk pengembangan dot matrix kedepannya adalah mempertimbangkan penambahan fitur-fitur inovatif. Secara hardware, pertimbangkan untuk menambahkan LED indikator sebagai penanda alarm mana yang aktif atau fungsi lainnya, memungkinkan pengguna mendapatkan pemberitahuan visual secara langsung. Integrasi dengan smartphone melalui WiFi atau Bluetooth akan memberikan fleksibilitas kontrol yang lebih besar. Selain itu, tambahkan kemampuan pengendalian speaker melalui Bluetooth untuk meningkatkan fungsionalitas audio. Di sisi software, pertimbangkan untuk menambahkan fitur Pomodoro yang dapat membantu pengguna dalam sesi belajar atau produktivitas. Saran terakhir adalah memilih mikrokontroler seperti ESP32 dengan memori yang lebih besar untuk mendukung pengembangan fitur-fitur canggih ini tanpa kendala ruang penyimpanan. Dengan demikian, dot matrix dapat menjadi lebih fungsional, interaktif, dan sesuai dengan kebutuhan pengguna modern.

G. DAFTAR PUSTAKA

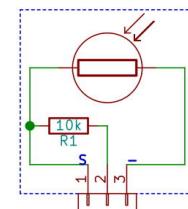
arduino.cc. Arduino Nano. Diakses pada 15 Desember 2023, dari
<https://docs.arduino.cc/hardware/nano>

H. LAMPIRAN

a. Datasheets atau Pinout



LM35



Light Depent Resistor



Real Time Clock

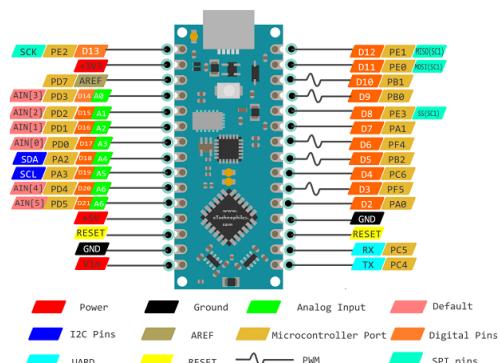


ArduinoCircuit.com

Buzzer



DOT MATRIX



Arduino Nano

b. Bentuk Fisik



Tampilan rangkaian

Bentuk Jadi



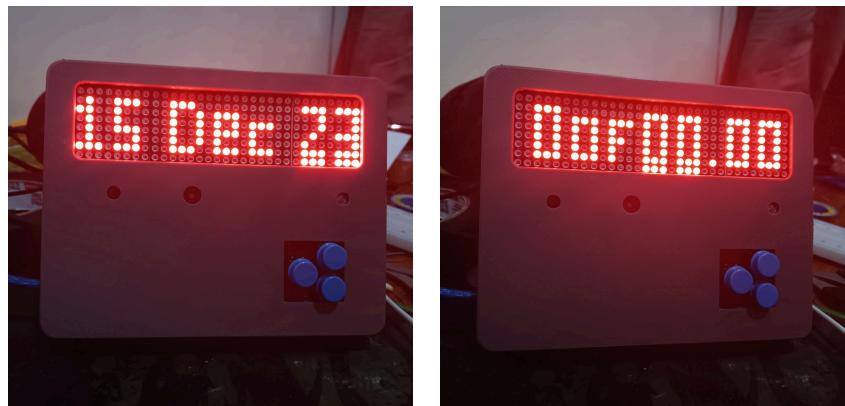
Tampilan Tanggal

Tampilan Suhu



Tampilan Jam

Tampilan ADJUST_JAM



Tampilan ADJUST_DATE

ADJUST_ALARM_0



ADJUST_ALARM_1

ADJUST_ALARM_2

c. Source Code

```
1. #include <RTCLib.h>
2. #include <SPI.h>
3. #include <MD_Parola.h>
4. #include <MD_MAX72xx.h>
5. #include <ezButton.h>
6.
7. #define HARDWARE_TYPE MD_MAX72XX::FC16_HW
8. #define MAX_DEVICES 4
9. #define CLK_PIN 13
10. #define Data_PIN 11
11. #define CS_PIN 10
12.
13. #define D_Suhu A3
```

```
14. #define bp_set 2
15. #define bp_up 4
16. #define bp_down 3
17. #define buzzerPin 5
18. #define LDRPin A2
19. const float BETA = 3950;
20. RTC_DS1307 rtc;
21.
22. ezButton bt_up(bp_up);
23. ezButton bt_set(bp_set);
24. ezButton bt_down(bp_down);
25.
26. MD_Parola myDisplay = MD_Parola(HARDWARE_TYPE, CS_PIN,
MAX_DEVICES);
27.
28. enum State {
29.     RUN,
30.     ADJUST_TIME,
31.     ADJUST_DATE,
32.     TRIGGER_ALARM_1,
33.     TRIGGER_ALARM_2,
34.     TRIGGER_ALARM_0,
35.     ADJUST_ALARM
36. };
37. State currentState = RUN;
38.
39. const char* shortMonths[] = {"Jan", "Feb", "Mar", "Apr",
"May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"};
40. struct ButtonHoldStatus {
41.     bool is_hold;
42.     unsigned long holdPressedTime;
43.     unsigned long lastRepeatTime;
44. };
45.
46. struct Alarm {
47.     uint8_t hour;
```

```
48.     uint8_t minute;
49.     bool enabled;
50.     const char* pesan;
51.     bool flag;
52. };
53.
54. ButtonHoldStatus holdStatusUp = {false, 0, 0};
55. ButtonHoldStatus holdStatusDown = {false, 0, 0};
56. ButtonHoldStatus holdStatusSet = {false, 0, 0};
57.
58. bool single_click(ezButton &button){
59.     if(button.isPressed()){
60.         return true;
61.     }
62.     return false;
63. }
64.
65. bool double_click(ezButton &button){
66.     static unsigned long lastPressTime = 0;
67.     static bool buttonWasReleased = true;
68.     const unsigned long double_click_interval = 250;
69.
70.     if(button.isPressed() && buttonWasReleased){
71.         unsigned long currentTime = millis();
72.         if((currentTime - lastPressTime) < double_click_interval){
73.             buttonWasReleased = false; // Hindari deteksi double
74.             click berulang
75.         }
76.         lastPressTime = currentTime;
77.     }
78.     if(button.isReleased()){
79.         buttonWasReleased = true;
80.     }
81.     return false;
82. }
```

```
83.  
84. bool hold_click(ezButton &button, ButtonHoldStatus &status){  
85.     const unsigned long hold_click_interval = 1500;  
86.     const unsigned long repeat_click_interval = 100;  
87.  
88.     if (button.isPressed() && !status.is_hold){  
89.         status.holdPressedTime = millis();  
90.         status.lastRepeatTime = status.holdPressedTime;  
91.         status.is_hold = true;  
92.     } else if (button.isReleased()){  
93.         status.is_hold = false;  
94.     }  
95.     if(status.is_hold){  
96.         unsigned long currentTime = millis();  
97.         if(currentTime - status.holdPressedTime >=  
hold_click_interval){  
98.             if(currentTime - status.lastRepeatTime >=  
repeat_click_interval){  
99.                 status.lastRepeatTime = currentTime;  
100.            return true;  
101.        }  
102.    }  
103. }  
104. return false;  
105. }  
106.  
107. bool hold_click_once(ezButton &button, ButtonHoldStatus  
&status){  
108.     const unsigned long hold_click_interval = 1500;  
109.  
110.     if(button.isPressed() && !status.is_hold){  
111.         status.holdPressedTime = millis();  
112.         status.is_hold = true;  
113.     } else if(button.isReleased()){  
114.         status.is_hold = false;  
115.     }  
116.     if(status.is_hold){
```

```
117.     unsigned long currentTime = millis();
118.     if(currentTime - status.holdPressedTime >=
    hold_click_interval){
119.         status.is_hold = false;
120.         return true;
121.     }
122. }
123. return false;
124. }
125.
126. MD_MAX72XX::fontType_t newFont[] PROGMEM =
127. {
128.     0, // 0
129.     0, // 1
130.     0, // 2
131.     0, // 3
132.     0, // 4
133.     0, // 5
134.     0, // 6
135.     0, // 7
136.     0, // 8
137.     0, // 9
138.     0, // 10
139.     0, // 11
140.     0, // 12
141.     0, // 13
142.     0, // 14
143.     0, // 15
144.     0, // 16
145.     0, // 17
146.     0, // 18
147.     0, // 19
148.     0, // 20
149.     0, // 21
150.     0, // 22
151.     0, // 23
```

```
152. 0, // 24
153. 0, // 25
154. 0, // 26
155. 0, // 27
156. 0, // 28
157. 0, // 29
158. 0, // 30
159. 0, // 31
160. 1, 0, // 32
161. 0, // 33
162. 0, // 34
163. 0, // 35
164. 0, // 36
165. 0, // 37
166. 0, // 38
167. 0, // 39
168. 0, // 40
169. 0, // 41
170. 0, // 42
171. 0, // 43
172. 0, // 44
173. 1, 0, // 45
174. 1, 64, // 46
175. 3, 96, 24, 6, // 47
176. 3, 126, 66, 126, // 48
177. 3, 68, 126, 64, // 49
178. 3, 98, 82, 78, // 50
179. 3, 82, 82, 126, // 51
180. 3, 30, 16, 126, // 52
181. 3, 94, 82, 114, // 53
182. 3, 126, 82, 114, // 54
183. 3, 2, 114, 14, // 55
184. 3, 126, 82, 126, // 56
185. 3, 94, 82, 126, // 57
186. 1, 36, // 58
187. 2, 64, 36, // 59
```

188. 1, 255, // 60
189. 0, // 61
190. 1, 0, // 62
191. 4, 0, 0, 0, 0, // 63
192. 3, 0, 0, 0, // 64
193. 4, 124, 18, 18, 124, // 65
194. 4, 126, 74, 74, 52, // 66
195. 4, 60, 66, 66, 66, // 67
196. 4, 126, 66, 66, 60, // 68
197. 4, 126, 82, 82, 82, // 69
198. 4, 126, 18, 18, 2, // 70
199. 4, 60, 66, 82, 116, // 71
200. 4, 126, 16, 16, 126, // 72
201. 3, 66, 126, 66, // 73
202. 4, 34, 66, 66, 62, // 74
203. 4, 126, 24, 36, 66, // 75
204. 4, 126, 64, 64, 64, // 76
205. 4, 126, 4, 4, 126, // 77
206. 4, 126, 4, 8, 126, // 78
207. 4, 60, 66, 66, 60, // 79
208. 4, 126, 18, 18, 12, // 80
209. 4, 60, 66, 98, 124, // 81
210. 4, 126, 18, 18, 108, // 82
211. 4, 76, 82, 82, 98, // 83
212. 3, 2, 126, 2, // 84
213. 4, 62, 64, 64, 62, // 85
214. 4, 30, 32, 64, 62, // 86
215. 5, 62, 64, 48, 64, 62, // 87
216. 4, 110, 16, 16, 110, // 88
217. 4, 14, 16, 112, 14, // 89
218. 4, 98, 82, 74, 70, // 90
219. 0, // 91
220. 0, // 92
221. 0, // 93
222. 0, // 94
223. 0, // 95

```
224. 0, // 96
225. 3, 116, 84, 124, // 97
226. 3, 124, 80, 112, // 98
227. 3, 120, 72, 72, // 99
228. 3, 112, 80, 124, // 100
229. 3, 124, 84, 92, // 101
230. 3, 124, 20, 4, // 102
231. 3, 92, 84, 124, // 103
232. 3, 124, 16, 112, // 104
233. 3, 0, 116, 0, // 105
234. 3, 0, 64, 60, // 106
235. 3, 124, 48, 72, // 107
236. 3, 0, 126, 0, // 108
237. 3, 124, 8, 124, // 109
238. 3, 124, 4, 124, // 110
239. 3, 124, 68, 124, // 111
240. 3, 124, 20, 28, // 112
241. 3, 28, 20, 124, // 113
242. 3, 124, 4, 4, // 114
243. 3, 92, 84, 116, // 115
244. 3, 8, 60, 72, // 116
245. 3, 124, 64, 124, // 117
246. 3, 60, 64, 60, // 118
247. 4, 60, 96, 96, 60, // 119
248. 3, 108, 16, 108, // 120
249. 3, 92, 80, 124, // 121
250. 3, 100, 84, 92, // 122
251. 0, // 123
252. 1, 126, // 124
253. 0, // 125
254. 0, // 126
255. 0, // 127
256. 3, 254, 194, 254, // 128
257. 3, 196, 254, 192, // 129
258. 3, 226, 210, 206, // 130
259. 3, 210, 210, 254, // 131
```

260. 3, 158, 144, 254, // 132
261. 3, 222, 210, 242, // 133
262. 3, 254, 210, 242, // 134
263. 3, 130, 242, 142, // 135
264. 3, 254, 210, 254, // 136
265. 3, 222, 210, 254, // 137
266. 0, // 138
267. 0, // 139
268. 0, // 140
269. 0, // 141
270. 0, // 142
271. 0, // 143
272. 0, // 144
273. 4, 252, 146, 146, 252, // 145
274. 0, // 146
275. 0, // 147
276. 4, 254, 194, 194, 188, // 148
277. 0, // 149
278. 4, 254, 146, 146, 130, // 150
279. 0, // 151
280. 0, // 152
281. 0, // 153
282. 4, 162, 194, 194, 190, // 154
283. 0, // 155
284. 0, // 156
285. 4, 254, 132, 132, 254, // 157
286. 0, // 158
287. 4, 188, 194, 194, 188, // 159
288. 0, // 160
289. 0, // 161
290. 0, // 162
291. 4, 204, 210, 210, 226, // 163
292. 0, // 164
293. 0, // 165
294. 0, // 166
295. 0, // 167

296. 0, // 168
297. 0, // 169
298. 0, // 170
299. 0, // 171
300. 0, // 172
301. 0, // 173
302. 0, // 174
303. 0, // 175
304. 0, // 176
305. 3, 244, 212, 252, // 177
306. 3, 252, 208, 240, // 178
307. 3, 248, 200, 200, // 179
308. 0, // 180
309. 3, 252, 212, 220, // 181
310. 3, 252, 148, 132, // 182
311. 3, 220, 212, 252, // 183
312. 0, // 184
313. 0, // 185
314. 0, // 186
315. 0, // 187
316. 3, 128, 254, 128, // 188
317. 0, // 189
318. 3, 252, 132, 252, // 190
319. 3, 252, 196, 252, // 191
320. 3, 252, 148, 156, // 192
321. 0, // 193
322. 3, 252, 132, 132, // 194
323. 3, 92, 84, 116, // 195
324. 3, 136, 188, 200, // 196
325. 3, 252, 192, 252, // 197
326. 3, 188, 192, 188, // 198
327. 0, // 199
328. 0, // 200
329. 3, 220, 208, 252, // 201
330. 0, // 202
331. 0, // 203

332. 0, // 204
333. 0, // 205
334. 0, // 206
335. 0, // 207
336. 0, // 208
337. 0, // 209
338. 0, // 210
339. 0, // 211
340. 0, // 212
341. 0, // 213
342. 0, // 214
343. 0, // 215
344. 0, // 216
345. 0, // 217
346. 0, // 218
347. 0, // 219
348. 0, // 220
349. 0, // 221
350. 0, // 222
351. 0, // 223
352. 0, // 224
353. 0, // 225
354. 0, // 226
355. 0, // 227
356. 0, // 228
357. 0, // 229
358. 0, // 230
359. 0, // 231
360. 0, // 232
361. 0, // 233
362. 0, // 234
363. 0, // 235
364. 0, // 236
365. 0, // 237
366. 0, // 238
367. 0, // 239

```
368.    0, // 240
369.    0, // 241
370.    0, // 242
371.    0, // 243
372.    0, // 244
373.    0, // 245
374.    0, // 246
375.    0, // 247
376.    0, // 248
377.    0, // 249
378.    0, // 250
379.    0, // 251
380.    0, // 252
381.    0, // 253
382.    0, // 254
383.    0, // 255
384. };
385.
386. void getSuhu(char suhuStr[]) {
387.     static unsigned long lastUpdateTime = 0; // Waktu terakhir
            pembacaan diperbarui
388.     static int lastStableReading = 0;           // Pembacaan stabil
            terakhir
389.     const long updateInterval = 1000;          // Interval waktu
            untuk memperbarui pembacaan (1 detik)
390.
391.     int analogValue = analogRead(D_Suhu);
392.     float voltage = analogValue * (5.0 / 1023.0);
393.     int currentReading = static_cast<int>(voltage * 100.0); //
            Pembacaan saat ini
394.
395.     // Periksa apakah sudah waktunya untuk memperbarui pembacaan
396.     if (millis() - lastUpdateTime >= updateInterval) {
397.         // Perbarui pembacaan stabil dan waktu terakhir pembacaan
            diperbarui
398.         lastStableReading = currentReading;
399.         lastUpdateTime = millis();
```

```
400. }
401.
402. // Gunakan pembacaan stabil terakhir untuk output
403. snprintf(suhuStr, 4, "%d", lastStableReading); // Gunakan
     buffer yang cukup untuk tiga digit
404. }
405.
406. void getTime(char timeStr[]) {
407.   DateTime now = rtc.now();
408.
409.   // Mengonversi nilai waktu menjadi string dalam format
     HH:MM:SS
410.   snprintf(timeStr, 9, "%02d:%02d:%02d", now.hour(),
     now.minute(), now.second());
411. }
412.
413. void getDate(char dateStr[]) {
414.   DateTime now = rtc.now();
415.   // Mendapatkan tiga huruf pertama dari nama bulan
416.   const char* monthChar = shortMonths[now.month() - 1];
417.   // Mengonversi nilai tanggal menjadi string dalam format
     YYYY-MMM-DD
418.   snprintf(dateStr, 12, "%02d-%s-%02d", now.day(), monthChar,
     now.year() % 100);
419. }
420.
421. void tampil(const char* data) {
422.   // myDisplay.displayClear();
423.   myDisplay.displayText(data, PA_CENTER, 0, 0, PA_NO_EFFECT,
     PA_NO_EFFECT);
424.   myDisplay.displayAnimate();
425. }
426.
427. Alarm alarms[3] = {
428.   {0, 0, false, "5024211001"},
429.   {0, 0, false, "5024211001 - Reza Ali Nirwansyah"},
430.   {0, 0, false, "Alarm aktif"}
```

```
431. };
432.
433. bool check_alarm(Alarm& alarm, DateTime now) {
434.     if (alarm.enabled && alarm.flag) {
435.         if (alarm.hour == now.hour() && alarm.minute ==
436.             now.minute()) {
437.             alarm.flag = false; // Set flag menjadi false setelah
438.             return true;
439.         } else if (alarm.minute != now.minute()) {
440.             alarm.flag = true; // Reset flag menjadi true jika menit
441.             berubah
442.         }
443.     }
444.
445.     bool isBuzzerPlay = true;
446.     unsigned long prevMillis_buzzer = 0;
447.     int currentNote = 0;
448.
449.     const int alarmSong[] = {HIGH, 200, LOW, 200, HIGH, 200, LOW,
450.     800};
451.     const int alarmSongSize = sizeof(alarmSong) /
452.         sizeof(alarmSong[0]);
453.     void playBuzzer() {
454.         if (millis() - prevMillis_buzzer >= alarmSong[currentNote
455.             + 1]) {
456.             // If it's time to change the note
457.             if (alarmSong[currentNote] == HIGH) {
458.                 tone(buzzerPin, 1000); // Ganti frekuensi sesuai
459.                 kebutuhan
460.             } else {
461.                 noTone(buzzerPin);
462.             }
463.         }
464.     }
465. }
```

```
461.  
462.     prevMillis_buzzer = millis();  
463.     currentNote += 2; // Menggeser indeks dua kali karena  
        setiap nada memiliki dua elemen (nada dan durasi)  
464.  
465.     if (currentNote >= alarmSongSize) {  
466.         // Reset to the beginning of the melody  
467.         currentNote = 0;  
468.     }  
469. }  
470. } else {  
471.     noTone(buzzerPin);  
472. }  
473. }  
474.  
475. void TriggerAlarm(Alarm alarm) {  
476.     bool isButtonPressed = false; // Variabel untuk menandai  
        apakah tombol sudah ditekan  
477.  
478.     myDisplay.displayText(alarm.pesan, PA_LEFT, 80, 0,  
        PA_SCROLL_LEFT, PA_SCROLL_LEFT);  
479.     while(!myDisplay.displayAnimate()) {  
480.         if (isBuzzerPlay) {  
481.             playBuzzer(); // Pastikan playBuzzer hanya dipanggil  
                jika isBuzzerPlay true  
482.         }  
483.         if(digitalRead(bp_set) == LOW) { // Periksa apakah tombol  
            ditekan  
484.             Serial.println("Tombol sudah ditekan");  
485.             isBuzzerPlay = false;  
486.             noTone(buzzerPin); // Matikan buzzer jika tombol ditekan  
487.             myDisplay.displayClear(); // Bersihkan display  
488.             isButtonPressed = true; // Tandai bahwa tombol sudah  
                ditekan  
489.             break; // Keluar dari loop while  
490.         }  
491.     }
```

```
492.     myDisplay.displayReset(); // Reset display setelah loop
493.
494.     if(isButtonPressed) {
495.         isBuzzerPlay = true;
496.         currentState = RUN;
497.     }
498. }
499.
500. void setup() {
501.     // put your setup code here, to run once:
502.     Serial.begin(9600);
503.
504.     if (!rtc.begin()){
505.         Serial.println("rtc ndak nemu");
506.         while(1);
507.     }
508.
509.     bt_set.setDebounceTime(50);
510.     bt_up.setDebounceTime(50);
511.     bt_down.setDebounceTime(50);
512.
513.     myDisplay.begin();
514.     myDisplay.setFont(newFont);
515.     myDisplay.setIntensity(5);
516.     myDisplay.displayClear();
517.     myDisplay.setCharSpacing(1);
518.
519. }
520.
521. void loop() {
522.     DateTime now = rtc.now();
523.     bt_up.loop();
524.     bt_set.loop();
525.     bt_down.loop();
526.
527.     int brightness;
```

```
528.     int ldrValue = analogRead(LDRPin);
529.     if (ldrValue < 341) { // 0 to 340
530.         brightness = 5;
531.     } else if (ldrValue < 682) { // 341 to 681
532.         brightness = 10;
533.     } else { // 682 to 1023
534.         brightness = 15;
535.     }
536.     myDisplay.setIntensity(brightness);
537.
538.     switch(currentState){
539.         case RUN:
540.             run(now);
541.             break;
542.         case ADJUST_TIME:
543.             adjust_time();
544.             break;
545.         case ADJUST_DATE:
546.             adjust_date();
547.             break;
548.         case TRIGGER_ALARM_1:
549.             TriggerAlarm(alarms[1]);
550.             break;
551.         case TRIGGER_ALARM_2:
552.             TriggerAlarm(alarms[2]);
553.             break;
554.         case TRIGGER_ALARM_0:
555.             TriggerAlarm(alarms[0]);
556.             break;
557.         case ADJUST_ALARM:
558.             adjust_alarm();
559.             break;
560.     }
561. }
562.
563. void run(DateTime now){
```

```
564.  
565.     char suhu[3];  
566.     char time[9];  
567.     char date[12];  
568.     getTime(time);  
569.     getSuhu(suhu);  
570.     getDate(date);  
571.  
572.     int sekon = now.second();  
573.  
574. if ((10 <= sekon && sekon <= 12) || (40 <= sekon && sekon <=  
        42)) {  
575.     tampil(suhu);  
576. } else if ((13 <= sekon && sekon <= 15) || (43 <= sekon &&  
        sekon <= 45)) {  
577.     tampil(date);  
578. } else {  
579.     tampil(time);  
580. }  
581.     if(hold_click_once(bt_set, holdStatusSet)){  
582.         currentState = ADJUST_TIME;  
583.     }  
584.  
585.     if(check_alarm(alarms[0], now)){  
586.         myDisplay.displayReset();  
587.         currentState = TRIGGER_ALARM_0;  
588.     }  
589.     if(check_alarm(alarms[1], now)){  
590.         myDisplay.displayReset();  
591.         currentState = TRIGGER_ALARM_1;  
592.     }  
593.     if(check_alarm(alarms[2], now)){  
594.         myDisplay.displayReset();  
595.         currentState = TRIGGER_ALARM_2;  
596.     }  
597. }
```

```
598.  
599. void adjust_time(){  
600.     static uint8_t jam = 0;  
601.     static uint8_t menit = 0;  
602.     static uint8_t detik = 0;  
603.     static uint8_t posisi = 0;  
604.     static bool adjust_time_init = true;  
605.  
606.     if(adjust_time_init){  
607.         DateTime now = rtc.now();  
608.         jam = now.hour();  
609.         menit = now.minute();  
610.         detik = now.second();  
611.         adjust_time_init = false;  
612.     }  
613.  
614.     char waktu[11];  
615.  
616.     if (single_click(bt_up)){  
617.         if(posisi == 0){  
618.             jam = (jam == 23) ? 23 : jam + 1; // Mengurangi jam dan  
memastikan tidak kurang dari 0  
619.         } else if(posisi == 1){  
620.             menit = (menit == 59) ? 59 : menit + 1; // Mengurangi  
menit dan memastikan tidak kurang dari 0  
621.         } else if(posisi == 2){  
622.             detik = (detik == 59) ? 59 : detik + 1; // Mengurangi  
detik dan memastikan tidak kurang dari 0  
623.         }  
624.     }  
625.  
626.     if (hold_click(bt_up, holdStatusUp)){  
627.         if(posisi == 0){  
628.             jam = (jam == 23) ? 23 : jam + 1; // Mengurangi jam dan  
memastikan tidak kurang dari 0  
629.         } else if(posisi == 1){
```

```
630.     menit = (menit == 59) ? 59 : menit + 1; // Mengurangi  
menit dan memastikan tidak kurang dari  
631. } else if(posisi == 2){  
632.     detik = (detik == 59) ? 59 : detik + 1; // Mengurangi  
detik dan memastikan tidak kurang dari 0  
633. }  
634. }  
635.  
636. if (single_click(bt_down)){  
637.     if(posisi == 0){  
638.         jam = (jam == 0) ? 0 : jam - 1; // Mengurangi jam dan  
memastikan tidak kurang dari 0  
639.     } else if(posisi == 1){  
640.         menit = (menit == 0) ? 0 : menit - 1; // Mengurangi menit  
dan memastikan tidak kurang dari 0  
641.     } else if(posisi == 2){  
642.         detik = (detik == 0) ? 0 : detik - 1; // Mengurangi detik  
dan memastikan tidak kurang dari 0  
643.     }  
644. }  
645.  
646. if (hold_click(bt_down, holdStatusDown)){  
647.     if(posisi == 0){  
648.         jam = (jam == 0) ? 0 : jam - 1; // Mengurangi jam dan  
memastikan tidak kurang dari 0  
649.     } else if(posisi == 1){  
650.         menit = (menit == 0) ? 0 : menit - 1; // Mengurangi menit  
dan memastikan tidak kurang dari 0  
651.     } else if(posisi == 2){  
652.         detik = (detik == 0) ? 0 : detik - 1; // Mengurangi detik  
dan memastikan tidak kurang dari 0  
653.     }  
654. }  
655.  
656. if (single_click(bt_set)){  
657.     posisi = (posisi + 1) % 3; // Mengubah posisi antara jam,  
menit, dan detik
```

```
658. }
659.
660. if(double_click(bt_set)){
661.
662.     DateTime now = rtc.now();
663.     currentState =ADJUST_DATE;
664.     rtc.adjust(DateTime(now.year(), now.month(), now.day(),
665.         jam, menit, detik));
666.     adjust_time_init = true;
667. }
668. if (hold_click_once(bt_set, holdStatusSet)){
669.     currentState = RUN;
670. // sprintf(waktu, "%02d:%02d:%02d", jam, menit, detik);
671. if(posisi == 0){
672.     sprintf(waktu, "%c%c:%02d:%02d", jam / 10 + '0' + 80, jam
673. % 10 + '0' + 80, menit, detik);
674. } else if(posisi == 1){
675.     sprintf(waktu, "%02d:%c%c:%02d", jam, menit / 10 + '0' +
676.     80, menit % 10 + '0' + 80, detik);
677. } else if(posisi == 2){
678.     sprintf(waktu, "%02d:%02d:%c%c", jam, menit, detik / 10
679. + '0' + 80, detik % 10 + '0' + 80);
680. }
681. void adjust_date() {
682.     static uint8_t tahun = 0;
683.     static uint8_t bulan = 0;
684.     static uint8_t tanggal = 0;
685.     static uint8_t posisi = 0; // 0 untuk tahun, 1 untuk bulan,
686.     2 untuk tanggal
687.     static bool adjust_date_init = true;
688.     myDisplay.setCharSpacing(1);
```

```
689.     if (adjust_date_init) {
690.         DateTime now = rtc.now();
691.         tahun = now.year();
692.         bulan = now.month();
693.         tanggal = now.day();
694.         adjust_date_init = false;
695.     }
696.
697.     char dateStr[11];
698.
699.     if (single_click(bt_up)) {
700.         if (posisi == 0) {
701.             tahun++;
702.         } else if (posisi == 1) {
703.             bulan = (bulan == 12) ? 1 : bulan + 1;
704.         } else if (posisi == 2) {
705.             tanggal = (tanggal == 31) ? 1 : tanggal + 1;
706.         }
707.     }
708.
709.     if (hold_click(bt_up, holdStatusUp)) {
710.         if (posisi == 0) {
711.             tahun++;
712.         } else if (posisi == 1) {
713.             bulan = (bulan == 12) ? 1 : bulan + 1;
714.         } else if (posisi == 2) {
715.             tanggal = (tanggal == 31) ? 1 : tanggal + 1;
716.         }
717.     }
718.
719.     if (single_click(bt_down)) {
720.         if (posisi == 0) {
721.             tahun--;
722.         } else if (posisi == 1) {
723.             bulan = (bulan == 1) ? 12 : bulan - 1;
724.         } else if (posisi == 2) {
```

```
725.         tanggal = (tanggal == 1) ? 31 : tanggal - 1;
726.     }
727. }
728.
729. if (hold_click(bt_down, holdStatusDown)) {
730.     if (posisi == 0) {
731.         tahun--;
732.     } else if (posisi == 1) {
733.         bulan = (bulan == 1) ? 12 : bulan - 1;
734.     } else if (posisi == 2) {
735.         tanggal = (tanggal == 1) ? 31 : tanggal - 1;
736.     }
737. }
738.
739. if (single_click(bt_set)) {
740.     posisi = (posisi + 1) % 3;
741. }
742.
743. if (double_click(bt_set)) {
744.     DateTime now = rtc.now();
745.     rtc.adjust(DateTime(tahun, bulan, tanggal, now.hour(),
746.                     now.minute(), now.second()));
747.     currentState = ADJUST_ALARM;
748.     adjust_date_init = true;
749. }
750. if (hold_click_once(bt_set, holdStatusSet)) {
751.     currentState = RUN;
752. }
753. char* monthChar = shortMonths[bulan - 1];
754. // sprintf(dateStr, sizeof(dateStr), "%02d-%s-%02d",
755.           tanggal, monthChar, tahun%100);
756. if (posisi == 0) {
757.     sprintf(dateStr, sizeof(dateStr), "%02d-%s-%c%c", tanggal,
758.             monthChar, (tahun % 100) / 10 + '0' + 80, tahun % 10 + '0' +
80);
```

```
758.     sprintf(dateStr, sizeof(dateStr), "%02d-%c%c%c-%02d",
    tanggal, monthChar[0] + 80, monthChar[1] + 80, monthChar[2] +
    80, tahun % 100);
759. } else if(posisi == 2){
760.     sprintf(dateStr, sizeof(dateStr), "%c%c-%s-%02d",
    tanggal / 10 + '0' + 80, tanggal % 10 + '0' + 80, monthChar,
    tahun%100);
761. }
762. // sprintf(dateStr, "p%sp", monthChar);
763. tampil(dateStr); // Asumsi bahwa fungsi tampil() ada dan
    dapat menampilkan tanggal
764. }
765.
766. void adjust_alarm(){
767.     static bool adjust_alarm_init = true;
768.     static uint8_t no_alarm = 0;
769.     static uint8_t posisi = 0;
770.     static uint8_t jam = alarms[no_alarm].hour;
771.     static uint8_t menit = alarms[no_alarm].minute;
772.     static bool aktif = alarms[no_alarm].enabled;
773.
774.     if(adjust_alarm_init){
775.         no_alarm = 0;
776.         adjust_alarm_init = false;
777.         jam = alarms[no_alarm].hour;
778.         menit = alarms[no_alarm].minute;
779.         aktif = alarms[no_alarm].enabled;
780.     }
781.
782.     if (single_click(bt_up)){
783.         if(posisi == 0){
784.             aktif = true;
785.         } else if(posisi == 1){
786.             jam = (jam == 23) ? 23 : jam + 1; // Mengurangi jam dan
                memastikan tidak kurang dari 0
787.         } else if(posisi == 2){
```

```
788.     menit = (menit == 59) ? 59 : menit + 1; // Mengurangi  
    menit dan memastikan tidak kurang dari 0  
789.     }  
790. }  
791.  
792. if (hold_click(bt_up, holdStatusUp)){  
793.     if(posisi == 1){  
794.         jam = (jam == 23) ? 23 : jam + 1; // Mengurangi jam dan  
        memastikan tidak kurang dari 0  
795.     } else if(posisi == 2){  
796.         menit = (menit == 59) ? 59 : menit + 1; // Mengurangi  
        menit dan memastikan tidak kurang dari 0  
797.     }  
798. }  
799.  
800. if (single_click(bt_down)){  
801.     if(posisi == 0){  
802.         aktif = false;  
803.     } else if(posisi == 1){  
804.         jam = (jam == 0) ? 0 : jam - 1; // Mengurangi jam dan  
        memastikan tidak kurang dari 0  
805.     } else if(posisi == 2){  
806.         menit = (menit == 0) ? 0 : menit - 1; // Mengurangi  
        menit dan memastikan tidak kurang dari 0  
807.     }  
808. }  
809.  
810. if (hold_click(bt_down, holdStatusDown)){  
811.     if(posisi == 1){  
812.         jam = (jam == 0) ? 0 : jam - 1; // Mengurangi jam dan  
        memastikan tidak kurang dari 0  
813.     } else if(posisi == 2){  
814.         menit = (menit == 0) ? 0 : menit - 1; // Mengurangi  
        menit dan memastikan tidak kurang dari 0  
815.     }  
816. }  
817.
```

```
818.     if (single_click(bt_set)){
819.         posisi = (posisi + 1) % 3;
820.     }
821.
822.     if (double_click(bt_set)){
823.         alarms[no_alarm].hour = jam;
824.         alarms[no_alarm].minute = menit;
825.         alarms[no_alarm].enabled = aktif;
826.         no_alarm++;
827.         jam = alarms[no_alarm].hour;
828.         menit = alarms[no_alarm].minute;
829.         aktif = alarms[no_alarm].enabled;
830.         if(no_alarm == 3){
831.             adjust_alarm_init = true;
832.             currentState = RUN;
833.         }
834.     }
835.
836.     if (hold_click_once(bt_set, holdStatusSet)){
837.         currentState = RUN;
838.     }
839.
840.     char* tanda;
841.     if(aktif){
842.         tanda = "on";
843.     } else {
844.         tanda = "of";
845.     }
846.
847.     char jamString[3];
848.     sprintf(jamString, "%02d", jam);
849.     Serial.println(jamString);
850.
851.     char menitString[3];
852.     sprintf(menitString, "%02d", menit);
853.     Serial.println(menitString);
```

```
854.  
855.     char alarmTime[11];  
856.     if (posisi == 0){  
857.         sprintf(alarmTime,"%d%c%c%s.%02d",no_alarm,tanda[0] + 80,  
             tanda[1] + 80,jamString,menit);  
858.     } else if(posisi == 1){  
859.         sprintf(alarmTime,"%d%c%c%c.%02d",no_alarm,tanda[0],  
             tanda[1],jamString[0] + 80, jamString[1] + 80,menit);  
860.     } else if(posisi == 2){  
861.         sprintf(alarmTime,"%d%c%c%c%c.%c%c",no_alarm,tanda[0],  
             tanda[1],jamString[0], jamString[1],menitString[0] + 80,  
             menitString[1] + 80);  
862.     }  
863.     tampil(alarmTime);  
864. }
```