

# Laporan Proyek Jam Digital

## Versi Final: Jam Digital Lengkap

Nama : Abraham Pratomo & Kevin Reagen S.

NIM : 13221051 & 13221087

## Contents

<b>Spesifikasi</b>	<b>1</b>
<b>Perancangan</b>	<b>2</b>
Perancangan Perangkat Keras	3
Perancangan Perangkat Lunak	4
<b>Implementasi</b>	<b>12</b>
Implementasi Perangkat Keras	12
Implementasi Perangkat Lunak	13
<b>Pengujian / Testing</b>	<b>19</b>
<b>Kesimpulan</b>	<b>22</b>

## Daftar Gambar

Gambar 1 - Data Flow Diagram Level 0	3
Gambar 2 - Skema Rangkaian Sistem Jam Digital	4
Gambar 3 - Flowchart utama jam digital	5
Gambar 4 - Flowchart untuk mode Startup	6
Gambar 5 - Flowchart fungsi loop	7
Gambar 6 - Flowchart untuk mode Jam	8
Gambar 7 - Flowchart untuk setting mode Jam	9
Gambar 8 - Flowchart untuk mode Stopwatch	10
Gambar 9 - Flowchart untuk mode Timer	11
Gambar 10 - Flowchart untuk setting mode Timer	12
Gambar 11 – Flowchart untuk mode Buzzer	13
Gambar 12 – Flowchart untuk interrupt 1	14
Gambar 13 – Flowchart untuk interrupt 2	15
Gambar 14 – Flowchart untuk interrupt 3	16
Gambar 15 - Finite State Machine Mealy	18
Gambar 16 - Foto rangkaian perangkat yang sudah disusun	20
Gambar 17 - Tampilan Startup	23
Gambar 18 - Clock Setting	24
Gambar 19 - Tampilan Mode Clock	24

Gambar 20 - Tampilan Mode Stopwatch	25
Gambar 21 - Tampilan Mode Timer	26

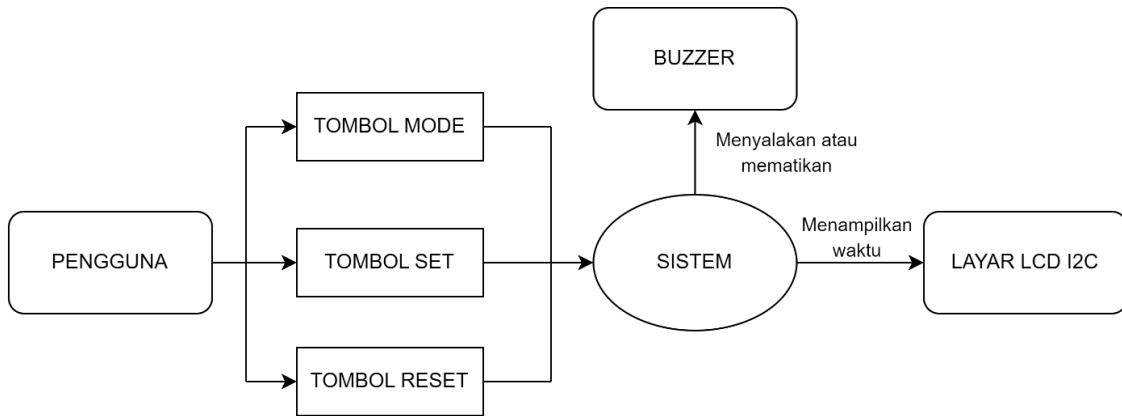
## Spesifikasi

Alat yang dirancang terdiri dari komponen-komponen berikut:

- |                                       |   |
|---------------------------------------|---|
| 1. Arduino Nano                       | x 1   |
| 2. Breadboard                         | x 1   |
| 3. Kabel jumper                       | seperlunya                                  |
| 4. LCD I2C                            | x 1   |
| 5. Buzzer                             | x 1   |
| 6. Push-Button                        | x 3   |
| 7. Laptop + Kabel USB-to-UART<br>kode | untuk catu daya Arduino Nano & pengunggahan |
| 8. Resistor 1kΩ                       | x3  |

Alat yang dirancang merupakan suatu jam digital yang memiliki tiga mode: **Waktu** biasa, **Stopwatch**, dan **Timer**. Alat ini akan memiliki tiga tombol dengan fungsinya masing-masing. Tombol pertama bernama **MODE** dipakai secara umum untuk mengubah ketiga mode milik jam, tetapi selain itu dipakai untuk mengubah elemen pengaturan waktu yang ingin diubah pada mode **Waktu** dan **Timer**. Tombol kedua bernama **SET** dipakai untuk melakukan afirmasi waktu yang sudah diatur, umumnya pada mode **Waktu** dan **Timer**, serta memulai timer setelah selesai dikonfigurasi. Selain itu, tombol ini juga dapat dipakai untuk memulai/memberhentikan stopwatch, sehingga memiliki nama lain **START**. Tombol terakhir bernama **RESET** dipakai untuk utamanya mengembalikan nilai waktu yang sudah terlewati oleh stopwatch ke nilai ‘0’. Namun, selain itu ada fungsi kedua yaitu untuk inkrement elemen waktu yang sudah ditunjuk pada mode **Waktu** dan **Timer**, sehingga memiliki nama kedua **INCREMENT/INCREASE**.

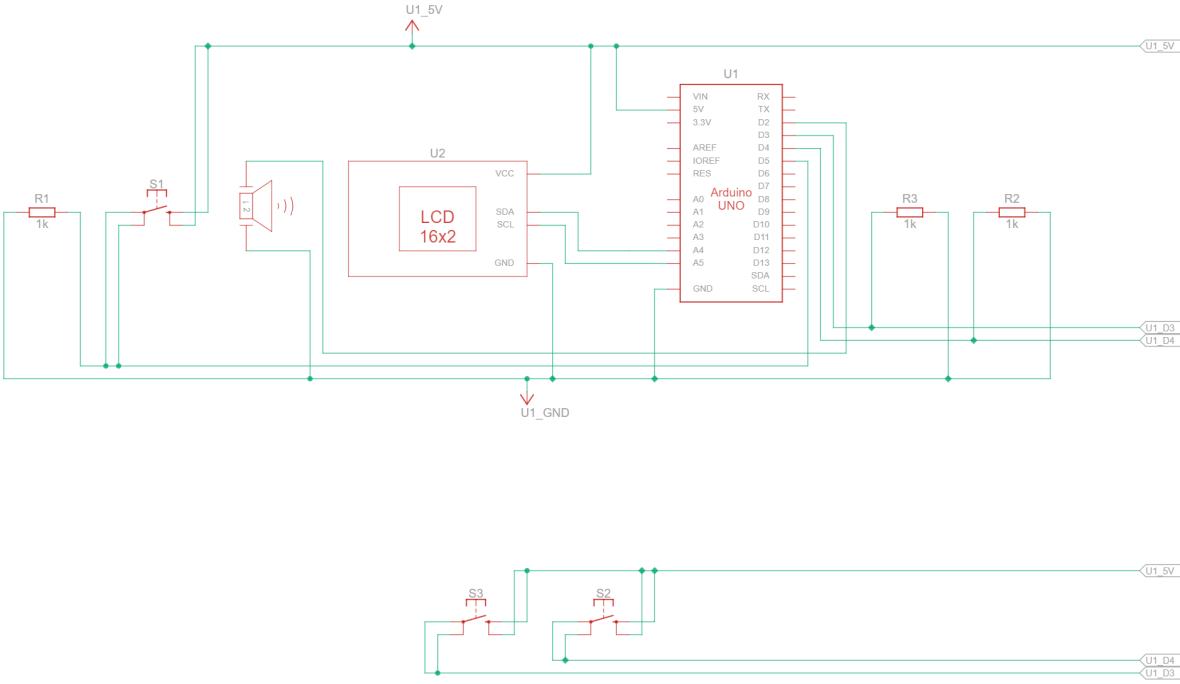
# Perancangan



Gambar 1 - Data Flow Diagram Level 0

Cara kerja dari sistem ini adalah pengguna berinteraksi dengan sistem jam digital melalui tombol yang memberikan input (terdapat tiga buah push-button). Sistem kemudian memproses masukan ini dan menampilkan waktu pada jam digital yang merupakan keluarannya. Selain menampilkan waktu pada layar LCD, sistem juga mampu mengirimkan sinyal pada buzzer untuk menyala atau mati.

## Perancangan Perangkat Keras



Gambar 2 - Skema Rangkaian Sistem Jam Digital

Ketika pertama dinyalakan, jam digital menunjukkan display angka 00:00:00. Lalu, jam digital otomatis melakukan count-up. Untuk melakukan konfigurasi waktu, pengguna perlu menekan push-button S2 yang terhubung ke pin D5 (tombol “Set”) untuk masuk ke dalam Clock Setting. Di dalam Clock Setting, pengguna meningkatkan angka-angka jam ataupun menit menggunakan push-button S3 yang terhubung ke pin D4 (tombol “Reset”). Konfirmasi pengaturan waktu dapat dilakukan dengan menekan push-button S2 yang terhubung ke pin D3 (tombol “Set”).

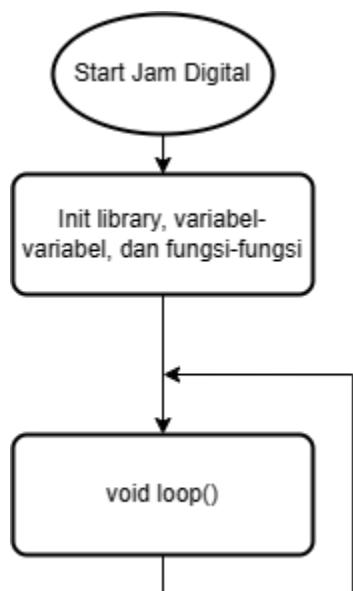
Jika ingin menggunakan fitur stopwatch, pengguna perlu menekan push-button S1 yang terhubung ke pin D5 (tombol “Mode”0). Tampilan awal stopwatch adalah 00:00:00. Untuk melakukan Start dan Stop, pengguna perlu menekan push-button S2. Ketika pengguna menekan push-button S3 ketika stopwatch berada dalam posisi Stop, angka stopwatch akan kembali ke tampilan awal 00:00:00.

Jika ingin menggunakan fitur timer, pengguna perlu menekan kembali push-button S1. Untuk melakukan konfigurasi waktu timer, pengguna perlu menekan push-button S2 untuk masuk ke Timer Setting. Di Timer Setting, pengguna bisa meningkatkan angka-angka detik, menit, dan jam menggunakan push-button S3. Untuk memulai countdown dari timer, pengguna harus menekan push-button S2. Timer juga dipasangkan dengan alarm atau buzzer. Jadi, ketika count dari timer sudah mencapai ‘0’, buzzer otomatis berbunyi menandakan waktu habis sesuai fitur jam digital pada umumnya. Buzzer dihubungkan dengan pin D2 Arduino Nano dikarenakan Arduino Nano hanya mendukung fungsi interrupt pada pin D2 dan D3.

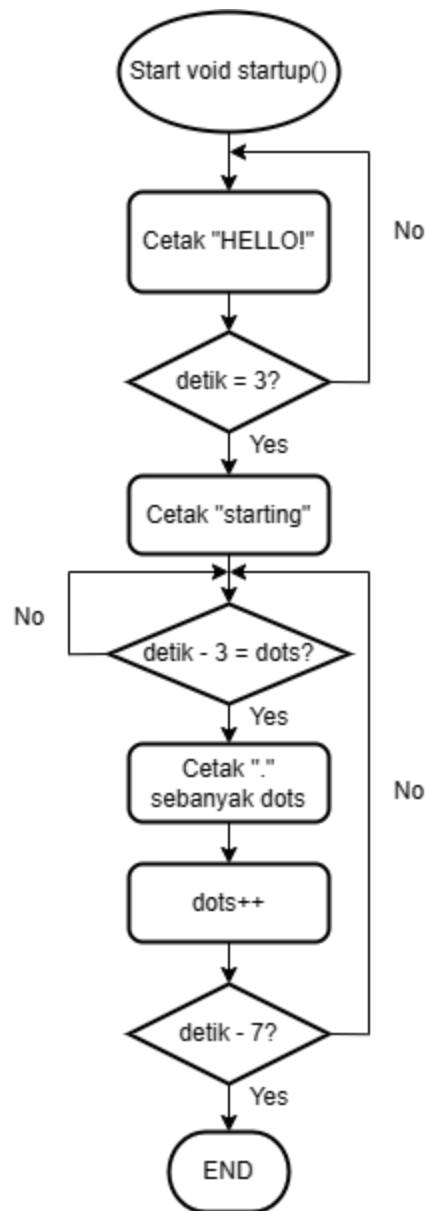
Push-button dikonfigurasikan menggunakan resistor pull-down eksternal supaya push-button memiliki kondisi awal “LOW”. Fungsi dari resistor pull-down adalah menghubungkan pin input ke 5V ketika switch terhubung. Saat tidak ada sinyal eksternal yang diberikan ke pin, pin input tidak

mendapat arus dan tegangan sama sekali karena tidak tercipta jalur dari pin 5V ke pin input, sehingga kondisinya adalah “LOW”. Sinyal eksternal yang dimaksudkan di sini adalah sinyal hasil dari ditekannya push-button.

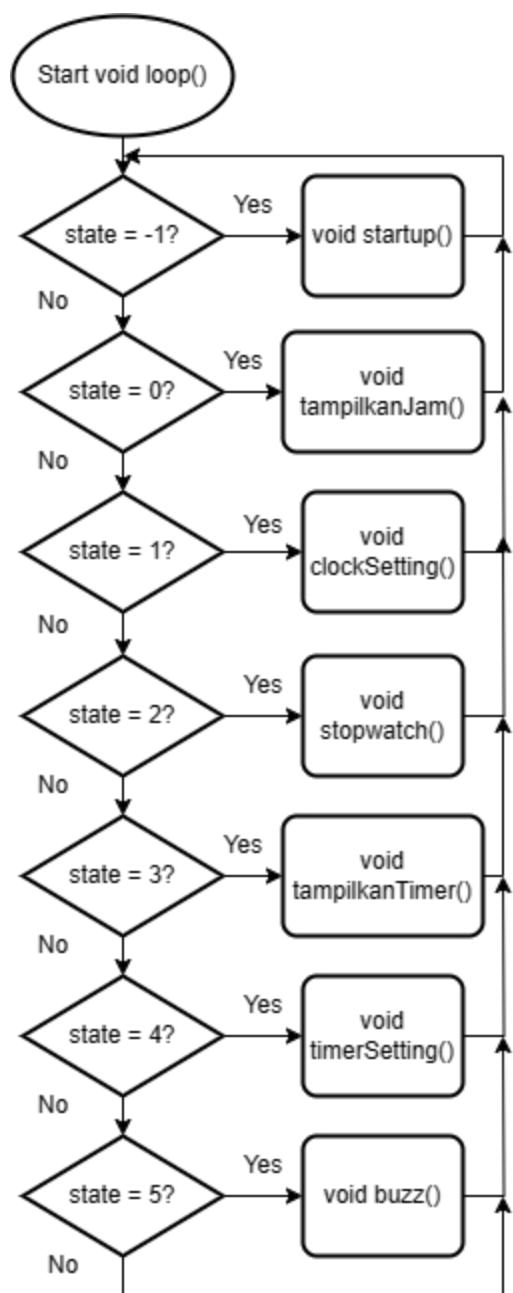
## Perancangan Perangkat Lunak



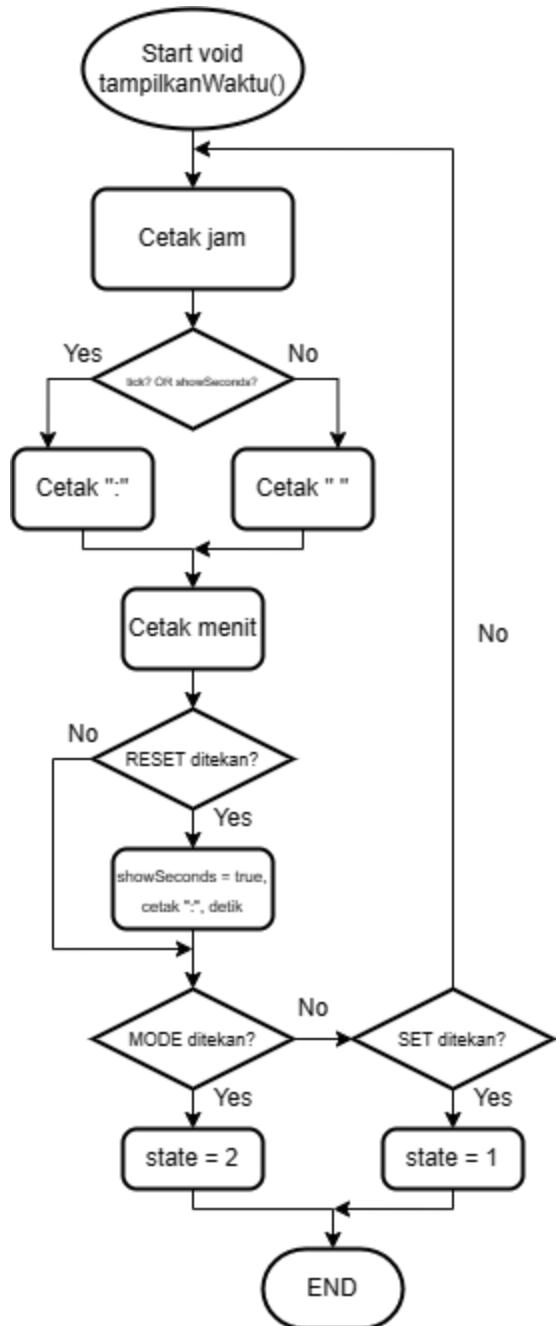
Gambar 3 - Flowchart utama jam digital



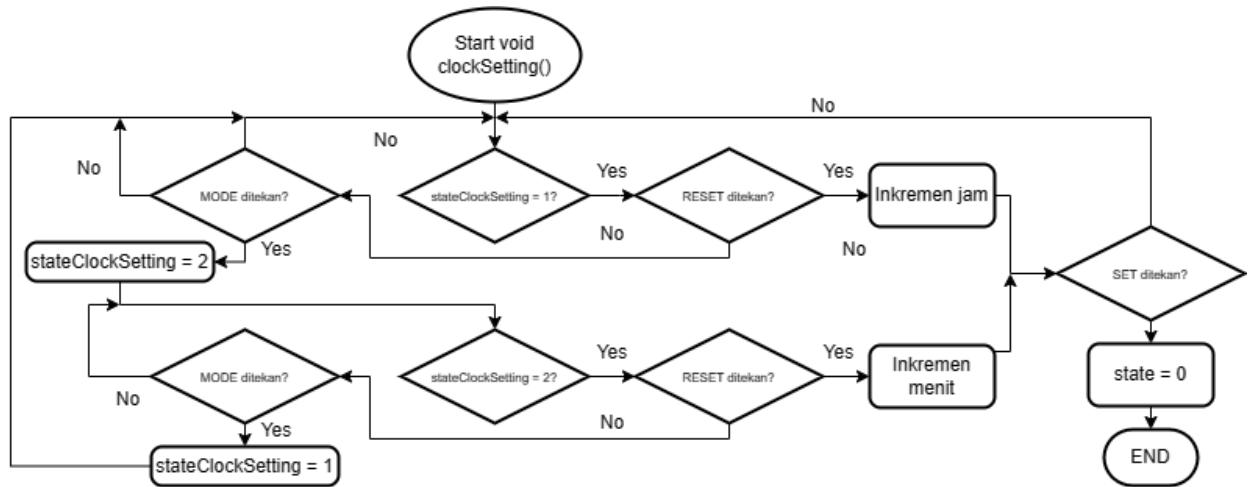
Gambar 4 - Flowchart untuk mode Startup



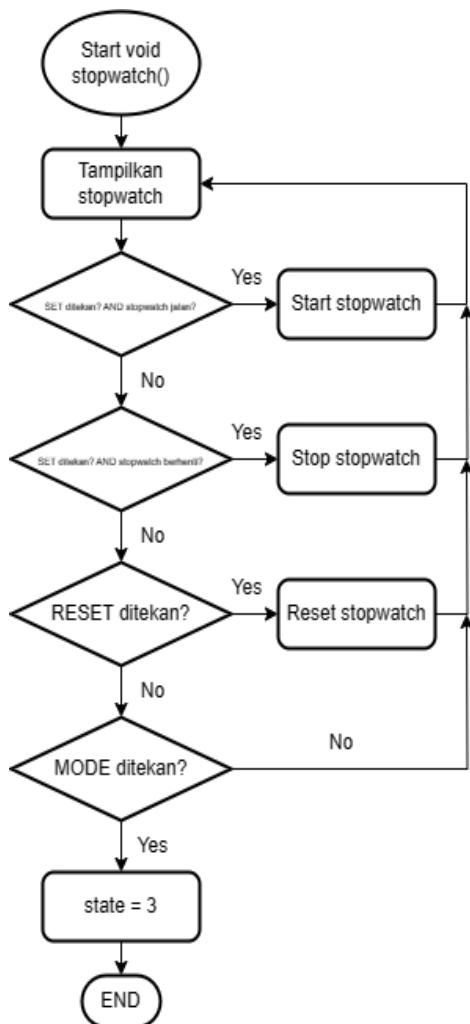
Gambar 5 - Flowchart fungsi loop



Gambar 6 - Flowchart untuk mode Jam

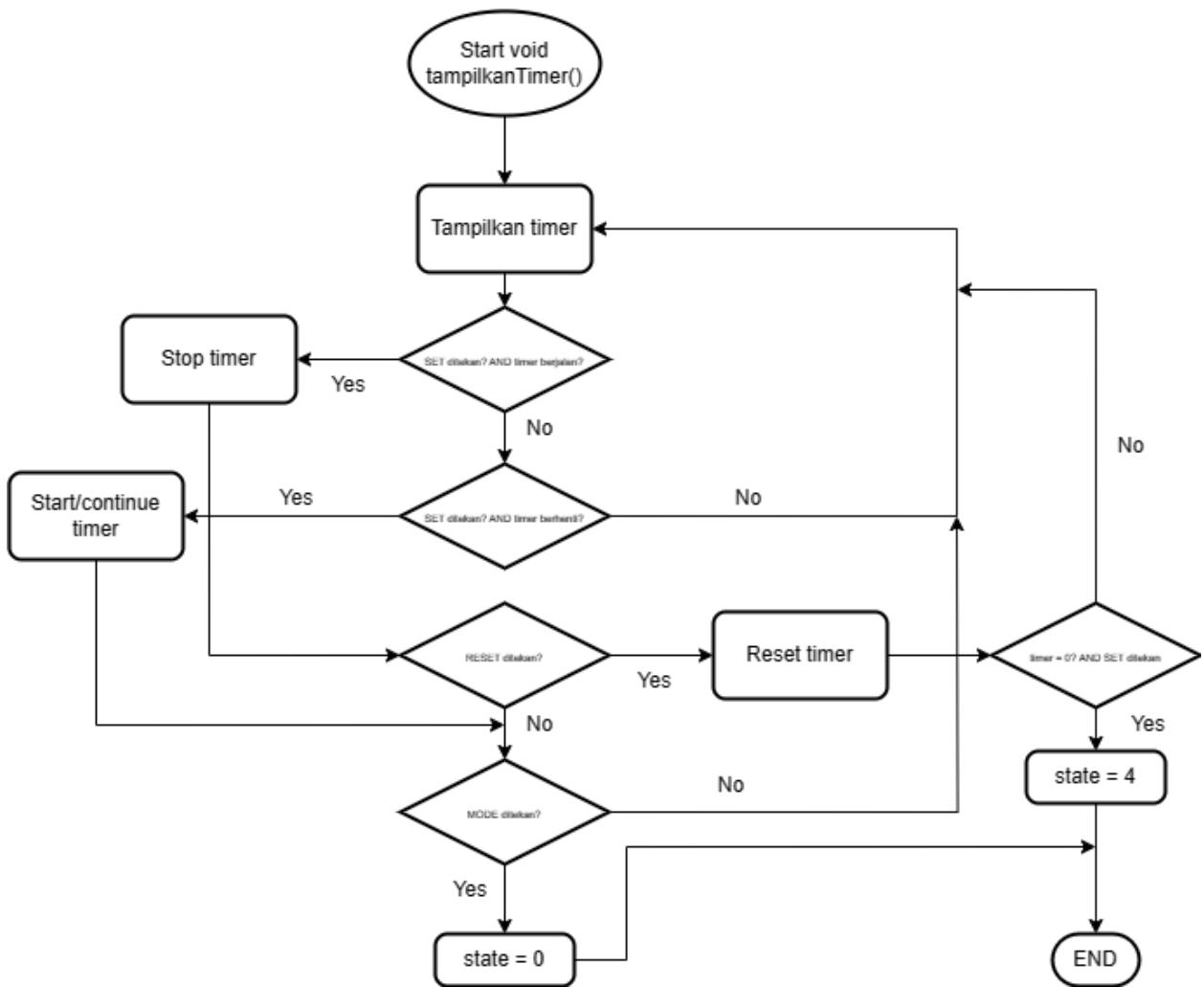


Gambar 7 - Flowchart untuk setting mode Jam

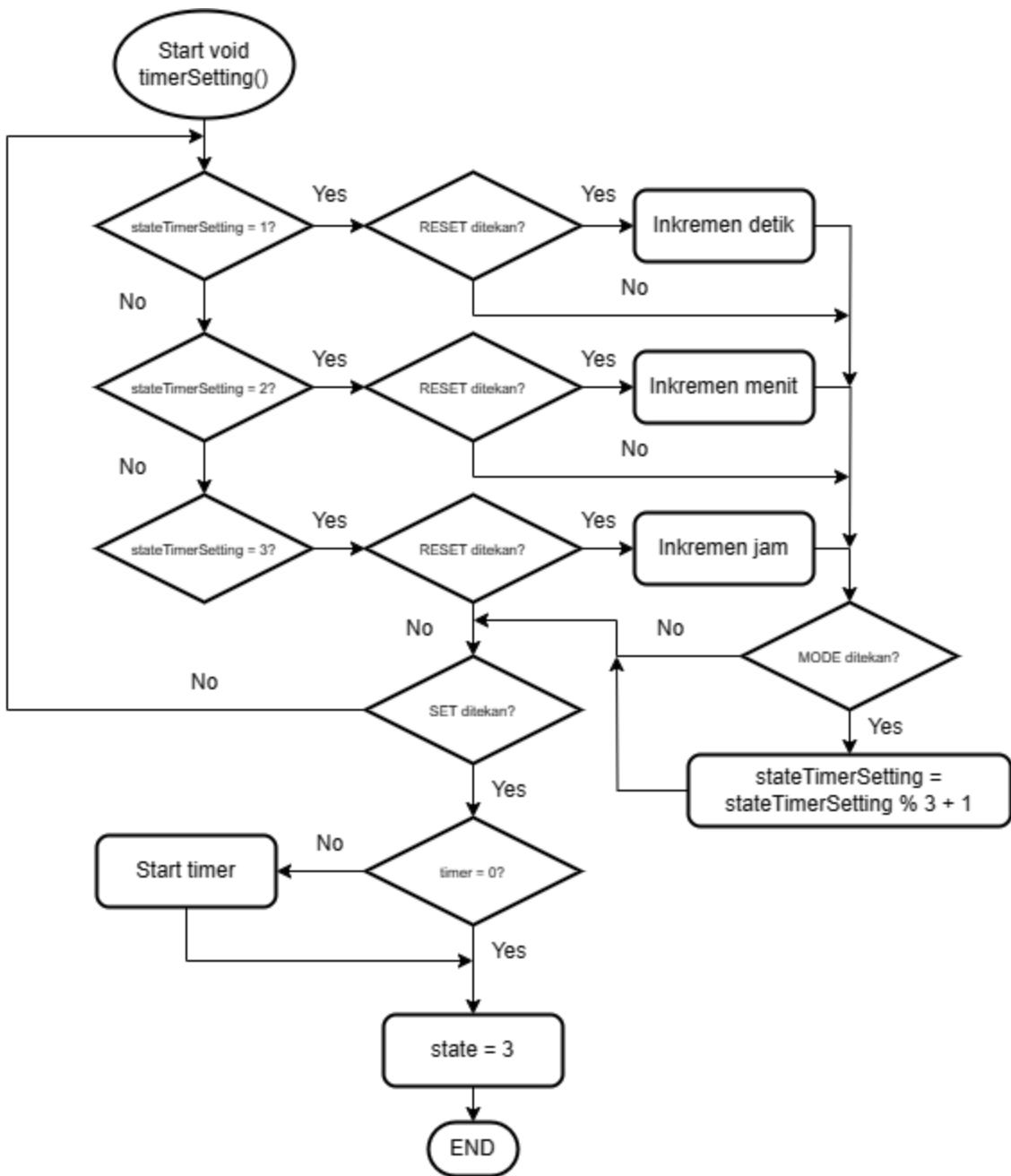


Gambar 8 - Flowchart untuk mode Stopwatch

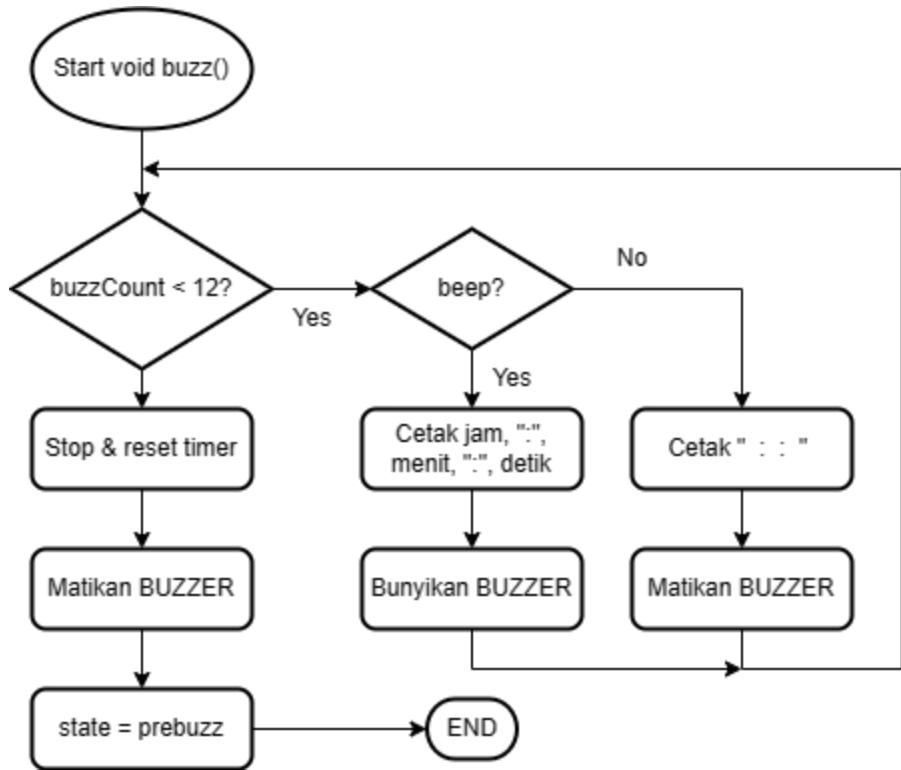




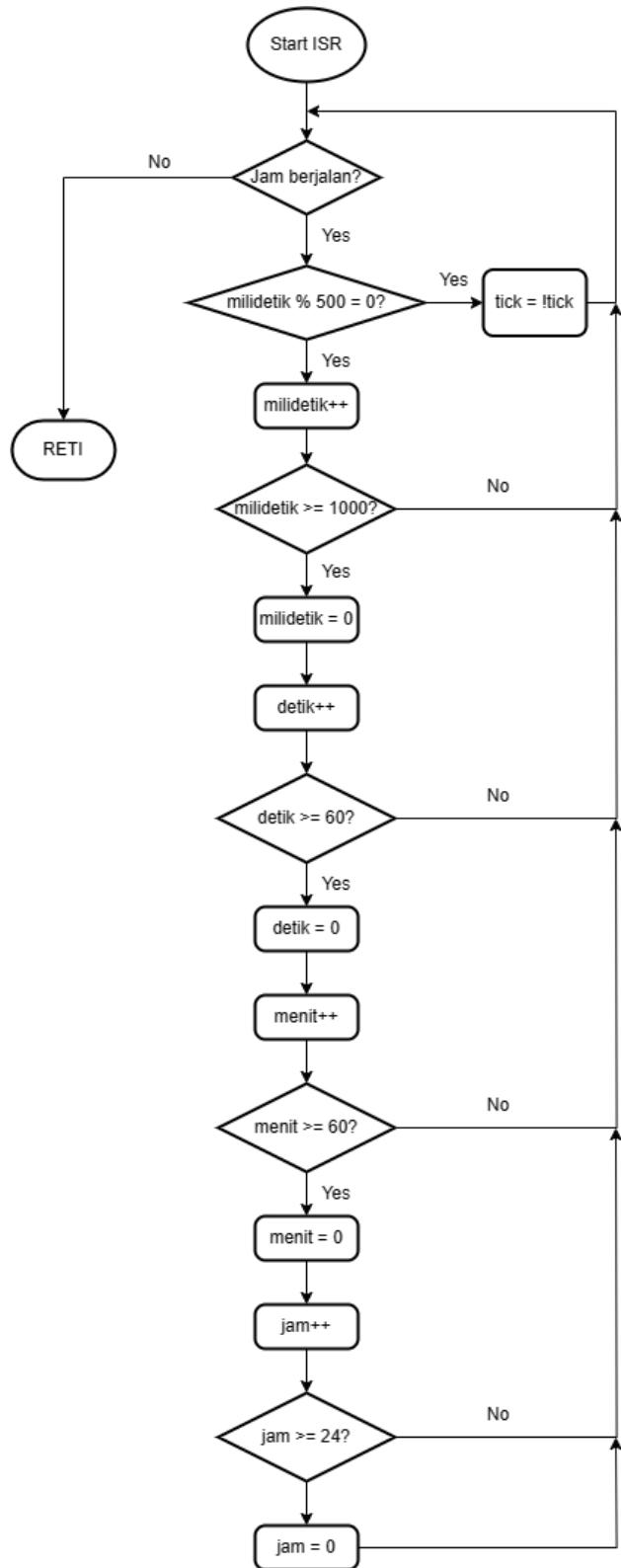
Gambar 9 - Flowchart untuk mode Timer



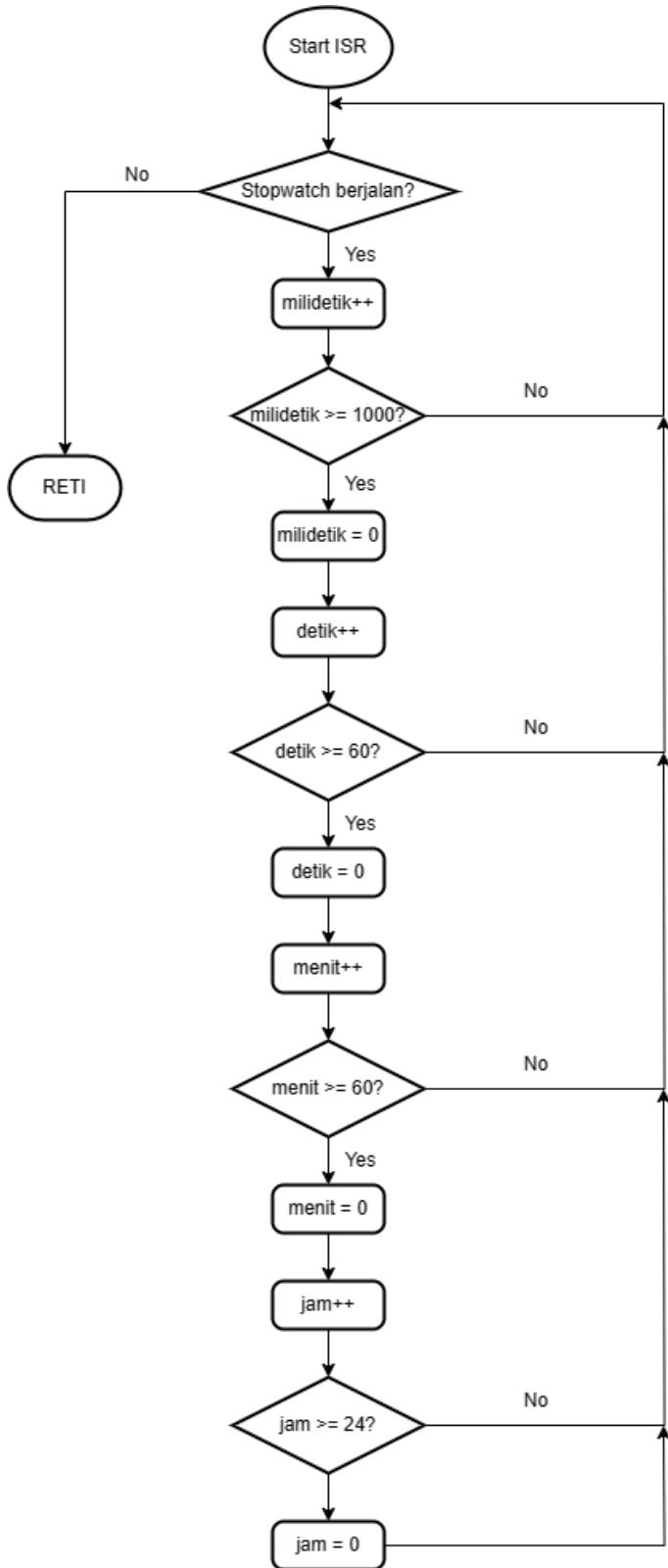
Gambar 10 - Flowchart untuk setting mode Timer



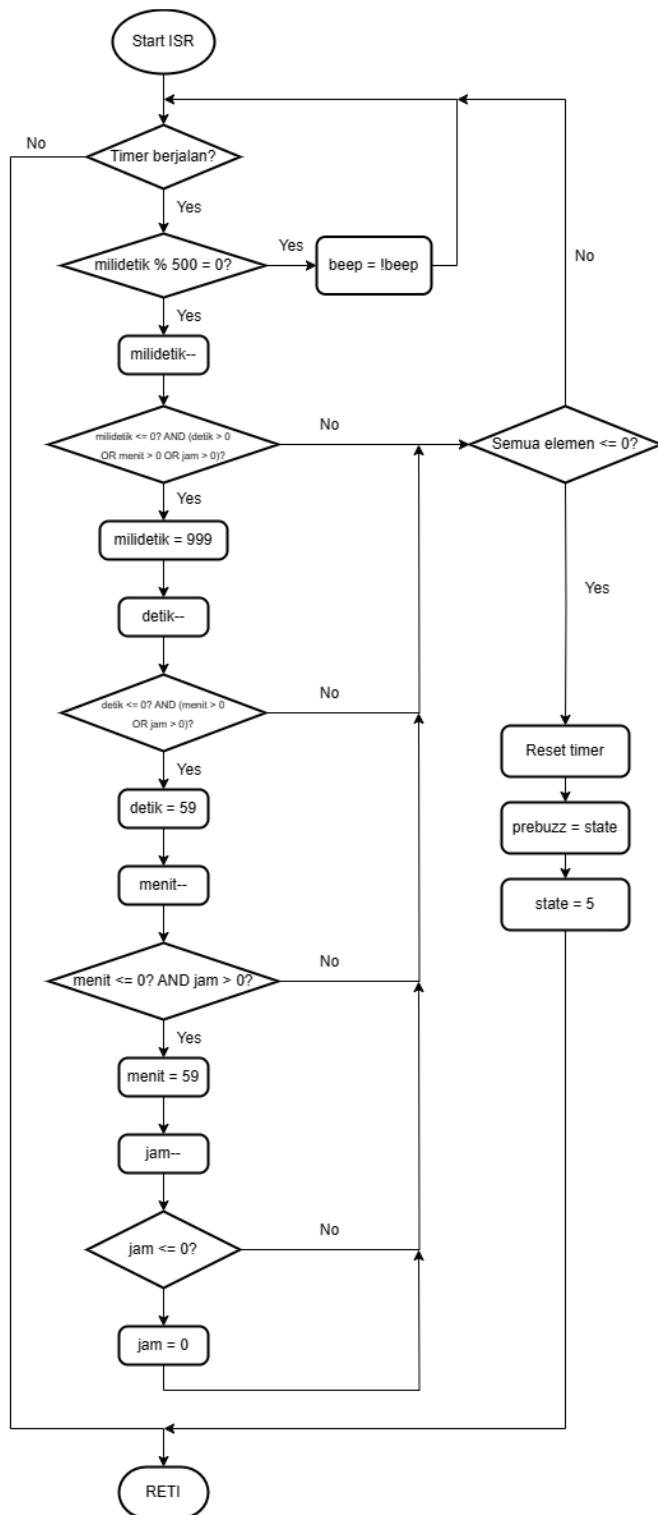
Gambar 11 – Flowchart untuk mode Buzzer



Gambar 12 – Flowchart untuk interrupt 1



Gambar 13 – Flowchart untuk interrupt 2



Gambar 14 – Flowchart untuk interrupt 3

Pada diagram alur utama, terdiri tujuh mode utama, dengan dua merupakan startup dan interrupt alarm: tampilan jam beserta pengaturannya, tampilan stopwatch, dan tampilan timer beserta pengaturannya, masing-masing dapat diakses secara sekuensial dengan penekanan

tombol **MODE** (untuk pengaturan jam dan timer lewat menekan tombol SET dari tampilan normalnya).

Pada diagram alur startup, Arduino akan menampilkan suatu pesan “HELLO!” selama 3 detik berbarengan dengan buzzer yang menyala selama 750 ms, selain untuk menandakan bahwa jam sudah menyala, juga sebagai pengecek fungsionalitas buzzer. Lalu, suatu pesan di bawahnya bertulisan “starting” akan muncul diikuti dengan titik yang bertambah setiap detik sebanyak satu buah. Apabila jumlah titik setelah “starting” sudah berjumlah tiga buah, artinya waktu yang dilewati proses startup sudah berlangsung selama tujuh detik, Arduino akan berpindah ke state pengaturan jam dahulu untuk meminta pengguna waktu yang akan di-set.

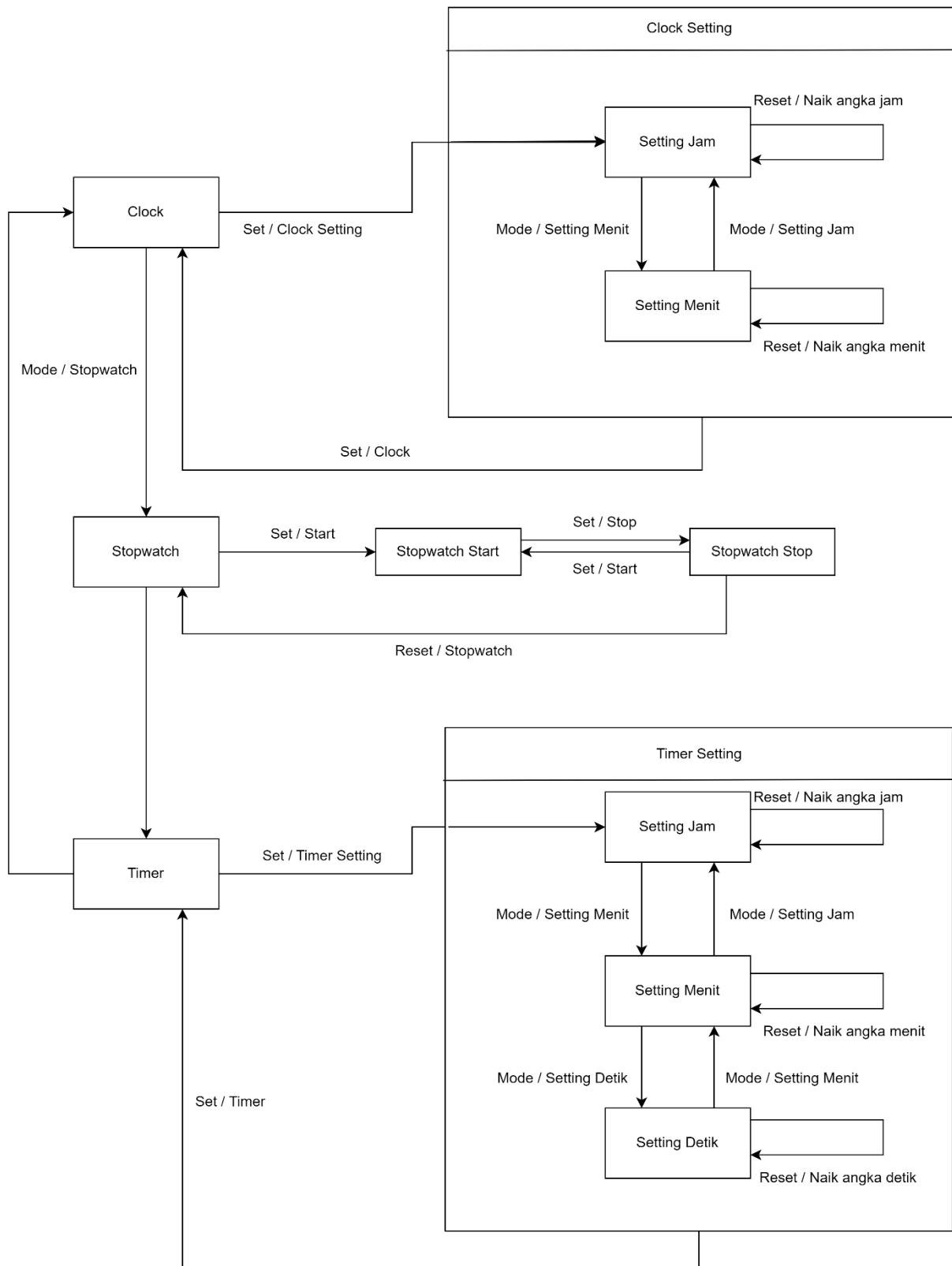
Pada diagram alur tampilan jam, pengguna dapat mengatur waktu dengan menekan tombol **SET**, lalu mengubah elemen yang ingin diubah (jam atau menit) dengan tombol **MODE**. Hal ini dideskripsikan melalui diagram alur setting jam. Saat tombol **SET** ditekan untuk kedua kalinya, program akan kembali ke tampilan jam awal alias waktu sudah selesai di-set. Inkremen nilai elemen waktu dilakukan dengan menekan tombol **RESET**.

Pada diagram alur tampilan stopwatch, pengguna dapat memulai atau memberhentikan stopwatch dengan menekan tombol **SET**. Lalu, penekanan tombol **RESET** akan me-reset stopwatch, kembali ke tampilan stopwatch siap untuk diubah modenya.

Pada diagram alur tampilan timer, saat pengguna menekan tombol **SET**, pengguna dapat langsung mengubah jam dengan tombol reset sebagai inkrementer. Penekanan tombol **MODE** akan mengubah elemen waktu yang ingin diubah (jam, menit, atau detik). Saat tombol **SET** ditekan kedua kalinya, akan ada dua perilaku. Apabila timer yang di-set masih bernilai ‘0’, maka timer tidak akan berjalan tapi langsung kembali ke tampilan awal, apabila tidak, maka timer akan kembali ke tampilan awal sekaligus memulai timer. Saat timer belum habis, penekanan **SET** dapat menghentikan atau melanjutkan timer. Saat timer dihentikan dan belum habis, menekan tombol **RESET** dapat membuat timer kembali ke 0 tanpa memicu alarm buzzer. Inkremen elemen waktu timer dilakukan dengan menekan tombol **RESET**.

Pada diagram alur buzzer, buzzer akan dipicu apabila timer yang sedang berjalan sudah mencapai nilai 0. Saat hal ini terjadi, akan dilakukan suatu rutin di mana buzzer akan dimati-nyalakan selama 5 detik, dengan interval mati dan nyala selama 500 ms. Beriringan dengan buzzer yang mati-nyala, LCD akan menampilkan “00:00:00” saat buzzer menyala dan “: : :” saat tidak. Hal ini untuk menandakan bahwa timer sudah habis. Apabila rutin tersebut selesai dilakukan, Arduino akan kembali ke state sebelum buzzer yang disimpan oleh variabel prebuzz.

Dalam Arduino, terdapat fungsi ISR atau Interrupt Service Routine yang tujuannya adalah untuk memanggil fungsi interupsi yang hanya terjadi ketika syarat tertentu sudah dipenuhi. Program ini menggunakan tiga buah ISR dari masing-masing grup timer. ISR Timer0 dan Timer1 berfungsi untuk melakukan count-up waktu untuk mode “Clock” dan mode “Stopwatch”. Sedangkan, ISR Timer2 kebalikannya, yaitu untuk melakukan count-down waktu untuk mode “Timer”.



Gambar 15 - Finite State Machine Mealy

Pada FSM, terdapat tiga buah push-button yaitu:

1) Mode

Tombol **MODE** digunakan untuk melakukan transisi state antara Clock, Stopwatch, dan Timer. Selain itu, tombol **MODE** juga digunakan untuk melakukan perubahan mode pengaturan jam, menit, dan detik pada Clock Setting dan Timer Setting (dapat dilihat pada Gambar 4).

2) Set

Pada mode Clock dan Timer, tombol **SET** digunakan untuk bertransisi ke state setting bersangkutan (Clock ke Clock Setting dan Timer ke Timer Setting). Selain itu, ketika berada pada mode Clock Setting dan Timer Setting, tombol **SET** berfungsi untuk melakukan penyimpanan pengaturan yang sudah diterapkan dan kembali ke mode Clock atau Timer sesuai dengan keadaan.

Pada mode Stopwatch, tombol **SET** digunakan untuk memulai count-up dari stopwatch sekaligus memberhentikan waktu stopwatch. Pada mode Timer Setting, tombol **SET** digunakan untuk memulai countdown timer dan pada mode Timer tombol **SET** digunakan untuk melakukan pause waktu timer.

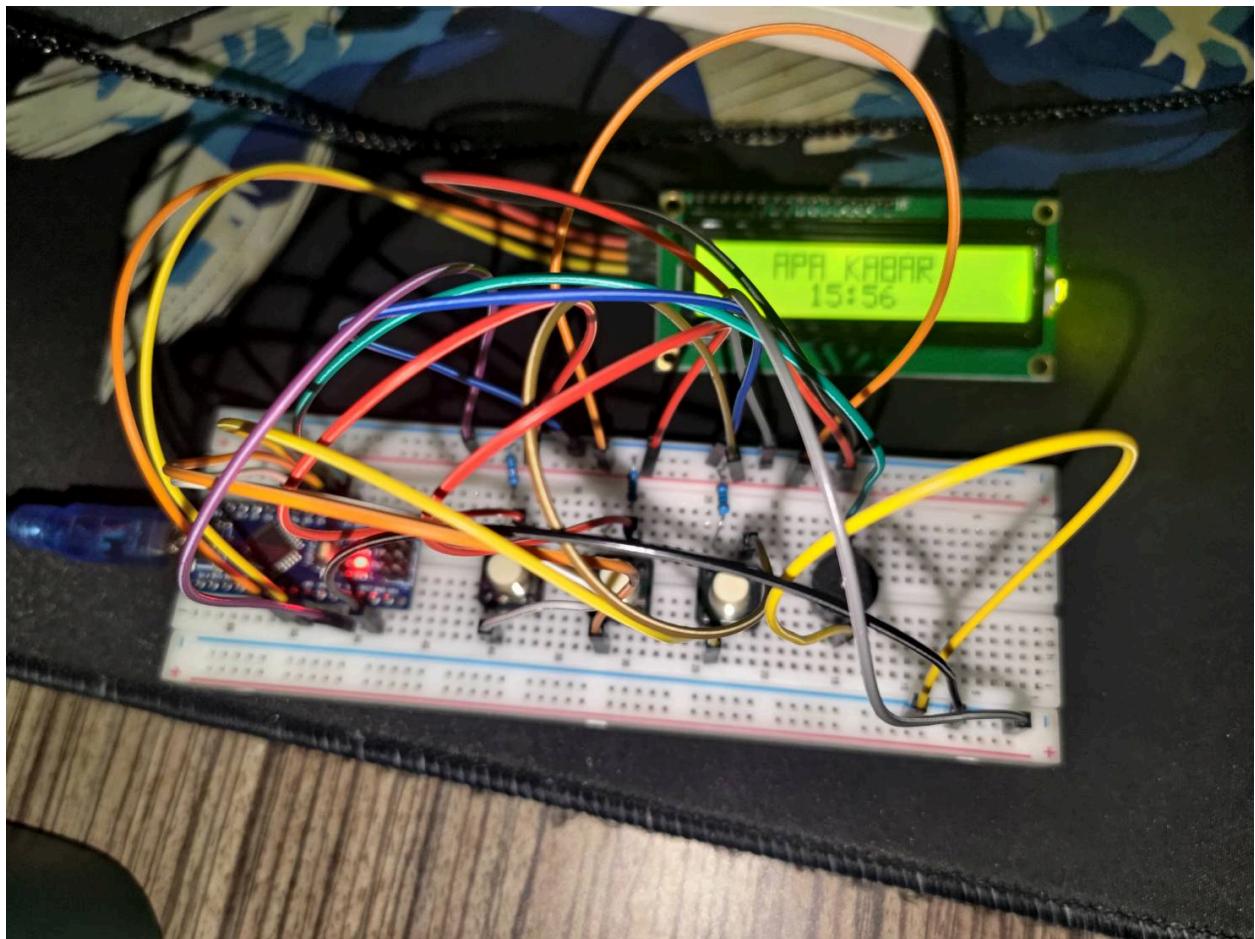
3) Reset

Tombol **RESET** memiliki fungsi sebagai berikut: meningkatkan angka jam atau menit pada Clock Setting atau Timer Setting sesuai dengan elemen yang sedang ingin diatur, melakukan reset waktu stopwatch, dan reset waktu timer.

## Implementasi

### Implementasi Perangkat Keras

Komponen yang digunakan pada tugas besar ini sudah dituliskan pada bagian “Spesifikasi”. Penyusunan rangkaian di atas breadboard dibuat sesuai dengan skematik yang dibuat untuk perancangan perangkat keras. Didapat tampilan aktualnya sebagai berikut.



Gambar 16 - Foto rangkaian perangkat yang sudah disusun

## Implementasi Perangkat Lunak

Terdapat tiga mode pada sistem jam digital, yaitu mode “Clock”, mode “Stopwatch”, dan mode “Timer”. Setiap mode tersebut diimplementasikan memiliki grup timernya masing-masing, mode “Clock” diatur oleh grup Timer0, mode “Stopwatch” diatur oleh grup Timer1, dan mode “Timer” diatur oleh grup Timer2. Pada perangkat Arduino, Timer merupakan unit perangkat keras yang terintegrasi dengan mikrokontroler yang dapat menghitung waktu atau menghasilkan pulsa. Penggunaan ketiga Timer secara bersamaan adalah dengan tujuan supaya ketiga mode dapat berjalan secara bersamaan.

Timer yang digunakan dalam perangkat lunak diatur dengan ketelitian 1000 Hz atau 1 milidetik. Sebenarnya, acuan internasional dari 1 detik adalah durasi  $9.192.631.770$  kali osilasi resonansi siklotron cesium-133 (Cs) yang diukur pada suhu nol Kelvin (0 K). Ini berarti bahwa 1 milidetik adalah durasi  $9.192.631$  kali osilasi resonansinya saja. Alasan digunakannya ketelitian 1 milidetik adalah untuk memberikan tingkat akurasi yang tinggi ketika pada mode “Stopwatch”. Namun, kedua mode lainnya juga menggunakan ketelitian 1 milidetik untuk menyajikan tingkat akurasi yang tinggi.

Pengaturan Timer dilakukan dengan cara mengkonfigurasikan register-registernya. Register-register utama yang penting adalah sebagai berikut:

#### ✓ TCCR<sub>x</sub> (TCCRxA dan TCCRxB)

TCCR merupakan kependekan Timer/Counter Control Register. Pada Timer0, nilai dari register TCCR0A ini digunakan untuk mengatur mode operasi (PWM, CTC, atau fast PWM) sedangkan nilai dari register TCCR0B digunakan untuk mengatur prescaler. Register TCCR pada Timer0 dan Timer2 memiliki fungsi yang sama. Pada Timer1, terdapat fungsi tambahan dari register TCCR1B yaitu mengontrol pengaturan capture dan mode compare.

#### ✓ TCNT<sub>x</sub>

TCNT merupakan kependekan dari Timer/Counter. Register ini digunakan untuk menyimpan nilai hitungan waktu atau hitungan yang dihasilkan oleh grup Timer tertentu. Nilai dari register ini meningkat seiring berjalannya waktu sesuai dengan konfigurasi clock yang sudah diatur sebelumnya. Nilai dari register ini nantinya dibandingkan dengan nilai dari register OCRx.

#### ✓ OCRxA

OCR merupakan kependekan dari Output Compare Register. Fungsi utama dari register ini adalah untuk menyimpan nilai yang akan dibandingkan dengan nilai TCNT saat counter berjalan. Ketika nilai TCNT mencapai nilai yang sama dengan nilai yang tersimpan dalam OCR, terjadi operasi perbandingan (compare match), yang kemudian melakukan instruksi sesuai yang sudah diatur pengguna.

Tabel 1 - Konfigurasi Register Timer0

```
TCCR0A = 0;  
TCCR0B = 0;  
TCNT0 = 0;  
OCR0A = 249;  
TCCR0A |= (1 << WGM01);  
TCCR0B |= (1 << CS01) | (1 << CS00);  
TIMSK0 |= (1 << OCIE0A);
```

Awalnya, nilai TCCR0A dan TCCR0B diatur ke nilai ‘0’ terlebih dahulu sebagai langkah inisialisasi. Tujuan dilakukan inisialisasi tersebut adalah untuk menghindari pengaturan tidak terduga akibat konflik dengan pengaturan lain. Kemudian, nilai TCNT0A diatur ke nilai ‘0’ karena nilai awal sebelum dimulainya counter yang diinginkan adalah benar-benar dari ‘0’. Nilai OCR0A sebesar “249” digunakan untuk mengatur frekuensi clock Timer0 menjadi 1000 Hz. Persamaan frekuensi clock pada Timer0 adalah sebagai berikut:

$$f_{clock} = \frac{clock\ speed}{(OCR0A+1) \times prescaler}$$

Pada potongan kode `TCCR0B |= (1 << CS01) | (1 << CS00)`, perintah tersebut digunakan untuk mengatur bit CS01 dan CS00 pada register TCCR0B agar nilai prescaler adalah sebesar “64”. Dengan demikian, frekuensi clock internal mikrokontroler akan dibagi oleh 64 sebelum digunakan untuk menghitung waktu pada Timer0. Frekuensi clock dari ATMega328 adalah 16 MHz. Maka, nilai OCR0A dapat dicari dengan persamaan:

$$OCR0A = \frac{clock\ speed}{f_{clock} \times prescaler} - 1$$

$$OCR0A = \frac{16 \times 10^6}{1000 \times 64} - 1$$

$$OCR0A = 250 - 1 = 249$$

Nilai OCR sebesar “249” inilah yang menyebabkan Timer0 beroperasi dengan frekuensi 1000 Hz atau setiap 1 milidetik. Terakhir, pada potongan kode `TIMSK0 |= (1 << OCIE0A)`, perintah tersebut digunakan untuk mengatur bit OCIE0A (Output Compare A Match Interrupt Enable) pada register TIMSK0 (Timer/Counter Interrupt Mask Register) untuk mengaktifkan interrupt saat nilai TCNT0 mencapai nilai yang sama dengan OCR0A. Interrupt yang dilakukan adalah memanggil fungsi ISR(TIMER0\_COMPA\_vect). Fungsi ISR untuk Timer0 dan Timer1 yang digunakan pada mode “Clock” dan “Stopwatch” memiliki isi yang mirip yaitu untuk melakukan *increment* waktu mulai dari milidetik hingga jam. Satu perbedaan adalah pada Timer0, terdapat variabel tick yang setiap 500 ms mengalami negasi. Variabel tick ini digunakan untuk efek berkedip-kedip angka jam atau menit ketika pengguna sedang melakukan pengaturan jam atau menit. Juga, pada tampilan mode “Clock”, setiap 500 ms titik dua pemisah antara jam dan menit berkedip-kedip.

Tabel 2 - Konfigurasi Register Timer1

```
TCCR1A = 0;
TCCR1B = 0;
TCNT1 = 0;
OCR1A = 249;
TCCR1B |= (1 << WGM12);
TCCR1B |= (1 << CS11) | (1 << CS10);
TIMSK1 |= (1 << OCIE1A);
```

Awalnya, nilai TCCR1A dan TCCR1B diatur ke nilai ‘0’ terlebih dahulu sebagai langkah inisialisasi dengan tujuan yang sama dengan inisialisasi Timer0 yaitu menghindari konflik pengaturan. Kemudian, nilai TCNT1 diatur ke nilai ‘0’ karena nilai awal sebelum dimulainya counter yang diinginkan adalah benar-benar dari ‘0’. Nilai OCR0A sebesar “249” digunakan untuk mengatur frekuensi clock Timer0 menjadi 1000 Hz. Persamaan frekuensi clock dan prescaler yang digunakan antara Timer0 dan Timer1 adalah sama sehingga nilai OCRnya pun juga sama yaitu sebesar “249”.

Tabel 3 - Konfigurasi Register Timer2

```
TCCR2A = 0;
TCCR2B = 0;
TCNT2 = 0;
OCR2A = 124;
TCCR2A |= (1 << WGM21);
TCCR2B |= (1 << CS22) | (1 << CS20);
TIMSK2 |= (1 << OCIE2A);
```

Perbedaan konfigurasi register Timer2 dengan konfigurasi Timer0 dan Timer1 adalah nilai dari OCR dan prescaler yang digunakan. Frekuensi yang diinginkan adalah tetap 1000 Hz atau tingkat ketelitian 1 milidetik. Namun, prescaler pada Timer2 bukan lagi 64 melainkan 128, sehingga didapatkan persamaan:

$$OCR0A = \frac{\text{clock speed}}{f_{\text{clock}} \times \text{prescaler}} - 1$$

$$OCR0A = \frac{16 \times 10^6}{1000 \times 128} - 1$$

$$OCR0A = 125 - 1 = 124$$

Perbedaan nilai prescaler ini disebabkan karena kami ingin melakukan perbandingan ketelitian atau galat antara keduanya. Perbandingan ini nanti akan disertakan pada bagian Pengujian.

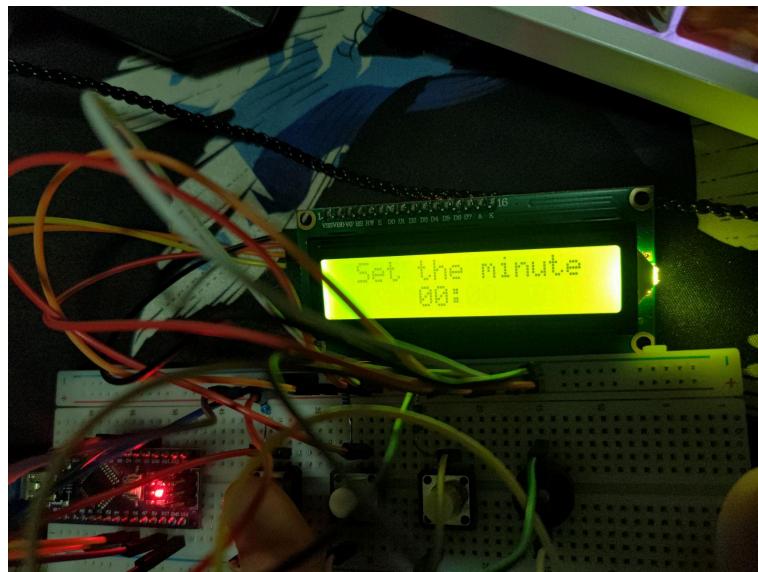
Pada implementasinya, tidak digunakan interupsi tombol sama sekali karena kami ingin mempertahankan fitur-fitur yang ada pada jam digital selayaknya. Hal ini disebabkan karena ketika satu buah push-button sudah dideklarasikan sebagai pemanggil fungsi interrupt, push-button tersebut tidak dapat berfungsi untuk melakukan tindakan lain. Berbeda halnya dengan jam digital yang diimplementasikan kali ini, satu tombol bisa digunakan untuk beberapa fungsi, seperti tombol **MODE** yang bisa berfungsi untuk melakukan pergantian state sekaligus untuk pergantian elemen waktu yang ingin diubah, tombol **SET** yang bisa berfungsi untuk menerapkan pengaturan waktu pada mode “Clock” dan “Timer” sekaligus untuk melakukan start dan stop pada mode “Stopwatch”, dan tombol **RESET** yang bisa berfungsi untuk melakukan reset waktu stopwatch dan timer sekaligus meningkatkan angka jam atau menit pada Clock Setting atau Timer Setting. Masing-masing timer memiliki variabel “waktu” yang berbeda-beda karena menggunakan grup Timer yang berbeda. Satuan waktu pada mode “Clock” memiliki imbuhan Clock (jamClock, menitClock, detikClock, dan milidetikClock). Satuan waktu pada mode “Stopwatch” memiliki imbuhan SW (jamSW, menitSW, detikSW, dan milidetikSW). Terakhir, satuan waktu pada mode “Timer” mempunyai imbuhan Timer (jamTimer, menitTimer, detikTimer, dan milidetikTimer). Sama halnya dengan satuan waktu, syarat pemanggilan interupsi timer ISR() ditandai dengan aktifnya variabel interruptxAktif dengan “x” adalah nomor grup timer. Tidak digunakan fungsi delay() dan millis() sama sekali pada implementasi program ini, debouncing tidak menggunakan delay() melainkan menggunakan variabel <nama button>Pressed. Nilai inisial variabel tersebut bernilai “false” karena tombol awalnya berada dalam kondisi “LOW”. Ketika push-button berada dalam kondisi “HIGH” atau sudah ditekan, variabel ini langsung diubah ke nilai “true” dan ketika push-button dilepas nilai dikembalikan ke “false”.

Pada awal program dimulai, terdapat fungsi startup yang menampilkan tulisan “HELLO” selama 3 detik. Juga, selama 750 milidetik pertama, buzzer berbunyi bebarengan dengan tulisan “HELLO” tersebut. Setelah itu, muncul tulisan “starting . . .” dengan jeda antara “starting” dan antar “.” adalah 1 detik. Jadi, jam digital ini mempunyai intro selama 7 detik hingga akhirnya masuk ke mode “Clock Setting” untuk melakukan pengaturan awal jam.

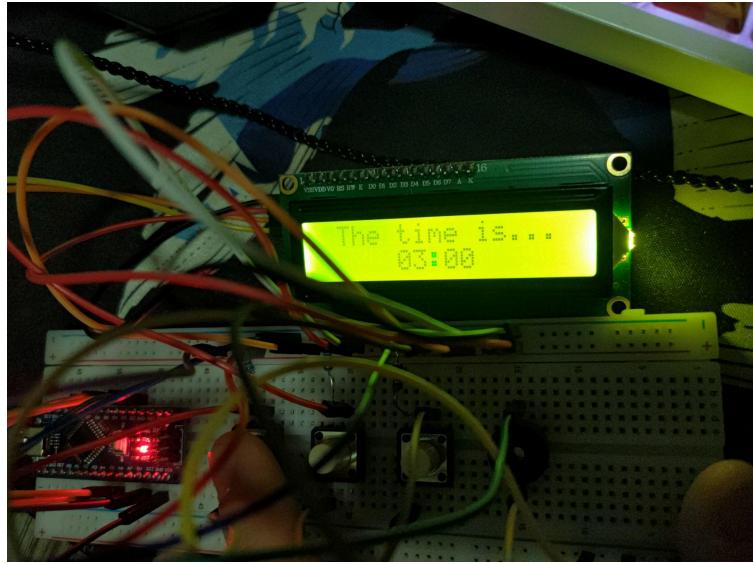


Gambar 17 - Tampilan Startup

Ketika masuk ke mode “Clock Setting”, pengguna langsung diminta untuk meminta mengatur waktu yang diinginkan. Untuk mengganti elemen waktu antara jam atau menit yang ingin diatur, pengguna perlu menekan tombol **MODE**. Untuk meningkatkan angka menit atau jam, pengguna perlu menekan tombol **RESET**. Jika pengguna sudah selesai mengatur waktu, pengguna perlu menekan tombol **SET** untuk memulai counter Timer dari waktu tersebut. Ketika dalam mode “Clock Setting”, detik terus-menerus dibuat bernilai ‘0’ untuk memudahkan kalibrasi waktu.



Gambar 18 - Clock Setting



Gambar 19 - Tampilan Mode Clock

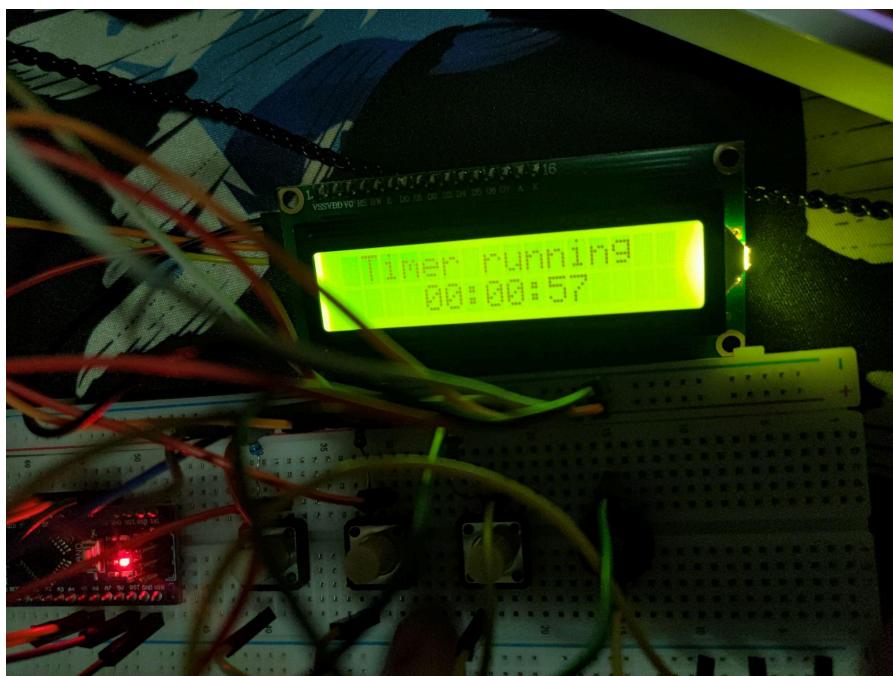
Tampilan *default* dari mode “Clock” hanya menampilkan format HH:MM atau jam dan menitnya saja. Jika pengguna ingin melihat tampilan detik, pengguna bisa menahan tombol **RESET**.

Mode selanjutnya adalah mode “Stopwatch”. Jika pengguna ingin berpindah dari mode “Clock” ke mode “Stopwatch”, pengguna perlu menekan tombol **MODE** dan LCD menunjukkan tampilan 00:00:00.000 dalam format HH:MM:SS.MS. Saat pertama kali pengguna masuk ke mode “Stopwatch”, waktu tidak berjalan hingga ketika pengguna menekan tombol “SET”. Ketika waktu sedang counting-up, pengguna bisa menekan tombol **SET** kembali untuk memberhentikan counter. Reset waktu kembali ke nilai ‘0’ bisa dilakukan dengan menekan tombol **RESET** ketika program dalam mode “Stopwatch” dengan keadaan waktu berhenti.



Gambar 20 - Tampilan Mode Stopwatch

Mode terakhir adalah mode “Timer”, jika pengguna ingin berpindah ke mode ini pengguna perlu menekan tombol **MODE**. Perpindahan mode bisa dilakukan meskipun stopwatch dalam keadaan berjalan dan waktu dari stopwatch akan terus berjalan hingga diberhentikan pengguna. Ketika pertama kali masuk mode “Timer”, pengguna dihadapkan dengan tampilan 00:00:00 terlebih dahulu. Untuk melakukan pengaturan dan masuk ke mode “Timer Setting”, pengguna perlu menekan tombol **SET**. Sama seperti “Clock Setting”, pengaturan elemen waktu yang ingin diatur (jam/menit/detik) dapat dipilih menggunakan tombol **MODE**. Peningkatan angka waktu dapat dilakukan dengan menekan tombol **RESET**. Setelah selesai melakukan pengaturan waktu dan ingin memulai timer, pengguna perlu menekan tombol **SET**.



Gambar 21 - Tampilan Mode Timer

Ketika pengguna menekan tombol **MODE** lagi, program kembali ke mode “Clock” dengan tampilan waktu saat ini karena fungsi ISR Timer0 tidak pernah dimatikan. Juga, ketika timer sudah mencapai nilai ‘0’ ketika pengguna sedang tidak berada di dalam mode Timer, maka setelah buzzer berbunyi selama 5 detik dan berkedip-kedip, jam digital akan kembali ke tampilan sebelum buzzer berbunyi.

## Pengujian / Testing

Kode penuh program jam digital dilampirkan pada bagian LAMPIRAN dan juga di dalam link Drive yang dikumpulkan bersamaan dengan file laporan ini dikarenakan jumlah baris yang mencapai 999.

Pada bagian ini, dilakukan pengujian terhadap implementasi perangkat keras dan lunak sistem jam digital masing-masing mode “Clock” yang menggunakan Timer0, mode “Stopwatch” yang menggunakan Timer1, dan mode “Timer” yang menggunakan Timer2.

Pengujian pertama yang dilakukan adalah mode "Clock". Nilai waktu pada jam digital dibandingkan dengan nilai waktu pada website time.is.

Tabel 4 - Data Pengukuran Akurasi Mode "Clock"

Range Waktu	Pengukuran Ke-	Website	Jam Digital	Galat ( $\epsilon$ )
0 - 60 detik	1	15:56:05	15:56:05	0
	2	15:56:10	15:56:10	0
	3	15:56:20	15:56:20	0
	4	15:56:40	15:56:40	0
	5	15:56:55	15:56:55	0
240 - 300 detik	6	16:00:05	16:00:05	0
	7	16:00:10	16:00:10	0
	8	16:00:20	16:00:20	0
	9	16:00:40	16:00:40	0
	10	16:00:55	16:00:55	0
900 – 2100 detik	11	16:11:35	16:11:34	0.10695 %
	12	16:11:55	16:11:54	0.10471 %
	13	16:13:40	16:13:39	0.09434 %
	14	16:16:20	16:16:19	0.08197 %
	15	16:25:00	16:24:58	0.11494 %

Perlu diketahui bahwa pada pengukuran ke-15 yaitu pada tepat 29 menit adalah waktu jam digital tepat tertinggal 2 detik di belakang website. Dikarenakan website digunakan sebagai acuan, maka dari itu perhitungan galat menggunakan persamaan berikut:

$$\epsilon = \frac{|\Delta t|}{\text{Total Waktu Website}} \times 100\%$$

$$\epsilon_{average} = \frac{\sum_{i=1}^{15} \epsilon_i}{15} = 0.0335\%$$

Namun, galat sebesar 0.0335% ini hanya relevan dan dalam jangka waktu Tabel – 4 saja alias hanya berlaku pada 29 menit awal. Dikarenakan jam digital hanya menggunakan ketelitian 1 detik (tidak sampai 1 milidetik), maka tidak dapat diprediksi secara akurat berapa galatnya pada pengukuran ke-1 hingga ke-10. Jika ingin mendapatkan prediksi paling dekat terhadap galat

masa depan, hanya perlu dilakukan pengamatan terhadap selisih waktu pada pengukuran ke-15 yaitu sebesar **0.11494%**. Nilai galat sebesar 0.11494% ekuivalen dengan apabila waktu sebenarnya sudah berjalan 1 jam, maka waktu di jam digital ini akan tertinggal sebesar 4.1378 detik. Apabila waktu sebenarnya sudah berjalan 24 jam atau 1 hari, waktu di jam digital akan tertinggal 99.308 detik.

Pengujian kedua yang dilakukan adalah mode "Stopwatch". Perbandingan nilai waktu tidak lagi menggunakan website melainkan menggunakan *smartphone*. Pengujian dilakukan dengan penekanan bersamaan antara tombol pause pada *smartphone* dan tombol "SET" yang digunakan untuk memberhentikan waktu stopwatch.

Tabel 5 - Data Pengukuran Akurasi Mode "Stopwatch"

Waktu	Pengukuran Ke-	Smartphone	Jam Digital	Galat ( $\epsilon$ )
60 detik	1	11.85 s	11.834 s	0.13502%
	2	24.66 s	24.628 s	0.12976%
	3	38.78 s	38.727 s	0.13667%
	4	49.93 s	49.880 s	0,12356%
	5	59.08 s	59.007 s	0,11399%
300 - 600 detik	6	338.63 s	338.244 s	0,12308%
	7	357.49 s	357.050 s	0,10929%
	8	449.25 s	448.759 s	0,09438%
	9	550.98 s	550.460 s	0,09407%
	10	604.88 s	604.311 s	0,09403%
600 - 900 detik	11	660.46 s	659.839 s	0,10132%
	12	783.63 s	782.836 s	0,09483%
	13	874.15 s	873.321 s	0,08933%
	14	900.02 s	899.216 s	0,08973%
	15	945.01 s	944.162 s	0,12356%

Pada pengujian ini, acuan yang digunakan adalah *smartphone*, lalu didapatkan galat dari perhitungan:

$$\epsilon = \frac{|\Delta t|}{\text{Total Waktu Smartphone}} \times 100\%$$

$$\epsilon_{average} = \frac{\sum_{i=1}^{15} \epsilon_i}{15} = 0.10861\%$$

Dapat dilihat bahwa nilai galat yang didapat pada pengujian mode “Stopwatch” memiliki nilai yang sangat mendekati dengan galat pada pengujian mode “Clock”. Hal ini dikarenakan ketelitian hingga 1 milidetik yang tidak ditampilkan pada mode “Clock” sehingga pengujian ke-1 hingga ke-10 pada mode “Clock” yang terlihat sama sebenarnya terdapat selisih dalam orde milidetik. Pada mode stopwatch, terdapat galat tambahan berupa selisih waktu antara penekanan antara tombol stopwatch *smartphone* dan tombol “SET” jam digital.

Pada pengujian mode “Timer”, dilakukan perbandingan hitung mundur antara waktu di *smartphone* dengan jam digital. Mulanya, keduanya diatur mulai dari 1 jam 45 menit (6300 detik) lalu di-start secara bersamaan. Ketika *smartphone* sudah terlebih dahulu mencapai ‘0’, di saat itu juga dinyalakan stopwatch pada device komputer dan stopwatch komputer diberhentikan di saat timer pada jam digital mencapai ‘0’. Didapatkan selisih waktu sebesar 7.355 s. Galat bisa dihitung menggunakan persamaan:

$$\epsilon = \frac{|\Delta t|}{6300 \text{ s}} \times 100\% = \frac{7.355}{6300 \text{ s}} \times 100\% = 0.11674\%$$

Terlihat, galat pada mode “Timer” sangat mendekati galat dari mode “Clock” namun lebih jauh dari mode “Stopwatch” akibat selisih waktu penekanan tombol. Bisa disimpulkan bahwa prescaler 64 dengan 128 tidak memiliki signifikansi yang terlalu besar karena galat yang didapatkan pun sangat mendekati.

Galat dengan nilai kurang lebih 0.11% ini sebenarnya tidak terlalu signifikan apabila jam digital hanya digunakan untuk penggunaan sehari-hari saja. Namun, untuk hal-hal seperti penelitian ilmiah, industri, dll, galat ini sangat signifikan.

## Kesimpulan

Pembuatan jam digital dilakukan menggunakan Arduino IDE dengan perangkat Arduino Nano bagian dari mikrokontroler ATMega328P. Jam digital sepenuhnya memanfaatkan perangkat Timer pada Arduino dengan cara mengkonfigurasikan nilai register-registernya sesuai dengan frekuensi yang diinginkan. Ketika register TCNT sudah mencapai nilai OCR, terjadi compare match dan fungsi ISR() dipanggil. Timer0 dan ISR(TIMER0\_COMPA\_vect) mengatur mode “Clock”, Timer1 dan ISR(TIMER1\_COMPA\_vect) mengatur mode “Stopwatch”, dan Timer2 dan ISR(TIMER2\_COMPA\_vect) mengatur mode “Timer”. Seluruh fitur pada spesifikasi dan perancangan jam digital sudah berhasil diimplementasikan 100%. Namun, terdapat galat perhitungan waktu pada Timer pada jam digital ini sebesar  $\pm 0.11\%$  dari acuan waktu seharusnya.