Digital Clock Implementation using Arduino with Multiplexing and Editing Features

Dhawal
Department of Electrical Engineering
Indian Institute of Technology Hyderabad
Email: ee24btech11015@iith.ac.in

I. Introduction

The digital clock system described here implements a feature-rich clock with editing capabilities using an Arduino microcontroller. The system utilizes a multiplexing technique to display time on six seven-segment displays using minimal I/O pins. This implementation includes pause/play functionality and digit-by-digit editing with increment and decrement buttons.

II. Components

Component	Value	Quantity
Arduino Uno		1
USB Cable	Type B	1
Seven Segment Display	Common Cathode	6
Push Buttons		4
IC 7447		1
Jumper Wires	M-M	16
Breadboard		1
Resistors	220Ω	7
Resistors	$10k\Omega$ (pull-down)	4

Table 1.0: Components List

III. Circuit Connections

A. Connections to Arduino

Make the button connections and IC 7447 connections to the Arduino as per the table below.

Item	Arduino Pin	Function
Button 1	D10	Edit Mode Toggle
Button 2	D11	Next Digit Selection
Button 3	D12	Increment Digit
Button 4	D13	Decrement Digit
IC 7447 Pin 7	D0	BCD Bit 0 (A)
IC 7447 Pin 1	D1	BCD Bit 1 (B)
IC 7447 Pin 2	D2	BCD Bit 2 (C)
IC 7447 Pin 6	D3	BCD Bit 3 (D)
Display 1	D4	Hours Tens Digit
Display 2	D5	Hours Units Digit
Display 3	D6	Minutes Tens Digit
Display 4	D7	Minutes Units Digit
Display 5	D8	Seconds Tens Digit
Display 6	D9	Seconds Units Digit

B. Connections from Seven Segment to BCD

Make the seven-segment connections identical for all seven segments. In total, there should only be 7 wires of output coming from the seven-segment display array.

IC 7447	Seven Segment (All)	Name
Pin 13	a	Controls segment a
Pin 12	b	Controls segment b
Pin 11	c	Controls segment c
Pin 10	d	Controls segment d
Pin 9	e	Controls segment e
Pin 15	f	Controls segment f
Pin 14	g	Controls segment g
Pin 8	Ground	Ground Supply
Pin 16	5V	Power Supply

Table 3.0: BCD to 7-Segment Connections

IV. Multiplexing Technique

All BCD inputs (A-D) are shared among six seven-segment displays. Displays are enabled one at a time using EN[0..5] = D4-D9. Each digit is displayed for 1ms, creating a fast alternating effect that appears continuous. This saves I/O pins and allows full six-digit display.

V. Digit Editing Logic

The clock allows pausing and digit-by-digit editing:

1) Press PAUSE (D10) to toggle run/edit mode. In edit mode, the clock stops.
2) Press NEXT (D11) to select the digit to edit (cycles 0-5: sec1, sec10, min1, min10, hr1, hr10).
3) Press INC (D12) to increment the selected digit with rollovers.
4) Press DEC (D13) to decrement the selected digit with rollunders.
5) Selected digit blinks every 500ms to indicate focus.

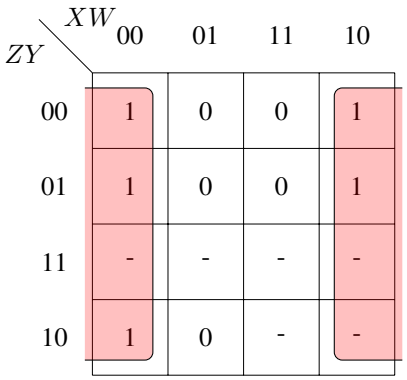
VI. Constraints Explanation

- **Seconds and Minutes Ones:** 0–9, standard BCD.
- **Seconds and Minutes Tens:** 0–5, to match 0–59 range.
- **Hours Ones:** 0–9 if hours tens = 0 or 1, but 0–3 if hours tens = 2, ensuring 24-hour format.
- **Hours Tens:** 0–2.

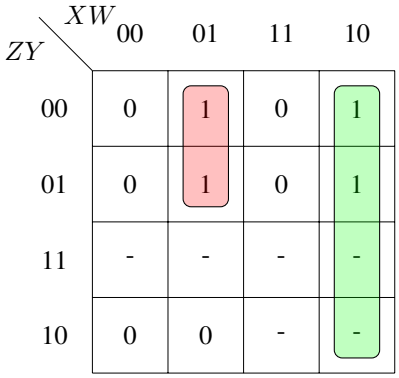
VII. Increment Logic and Truth Tables

A. Seconds Ones (0-9)

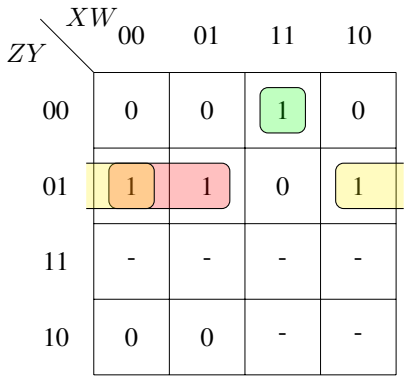
Z	Y	X	W	D	C	B	A
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	1
0	0	1	1	0	1	0	0
0	1	0	0	0	1	0	1
0	1	0	1	0	1	1	0
0	1	1	0	0	1	1	1
0	1	1	1	1	0	0	0
1	0	0	0	1	0	0	1
1	0	0	1	0	0	0	0



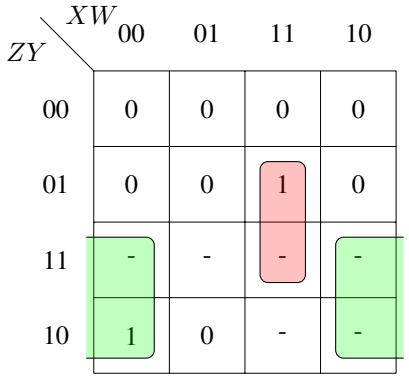
$$A = W_1'$$



$$B = (W_1 X_1' Z_1') + (W_1' X_1)$$



$$C = (X_1' Y_1) + (W_1' Y_1) + (W_1 X_1 Y_1')$$



$$D = (W_1' Z_1) + (W_1 X_1 Y_1)$$

B. Seconds Tens (0-5)

Z	Y	X	W	D	C	B	A
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	1
0	0	1	1	0	1	0	0
0	1	0	0	0	1	0	1
0	1	0	1	0	0	0	0

ZY \ XW	00	01	11	10
00	1	0	0	1
01	1	0	-	-
11	-	-	-	-
10	-	-	-	-

$$A = W_2'$$

ZY \ XW	00	01	11	10
00	0	1	0	1
01	0	0	-	-
11	-	-	-	-
10	-	-	-	-

$$B = (W_2 X_2' Y_2') + (W_2' X_2)$$

ZY \ XW	00	01	11	10
00	0	0	1	0
01	1	0	-	-
11	-	-	-	-
10	-	-	-	-

$$C = (W_2 X_2) + (W_2' X_2' Y_2)$$

$$D = 0$$

C. Minutes Ones (0-9)

Same as Seconds Ones with W3/X3/Y3/Z3.

D. Minutes Tens (0-5)

Same as Seconds Tens with W4/X4/Y4/Z4.

E. Hours Ones

I. Tens = 0/1 → 0-9

Same as Seconds Ones with W5/X5/Y5/Z5.

II. Tens = 2 → 0-3

X	W	D	C	B	A
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1
1	1	0	0	0	0

X \ W	0	1
0	1	0
1	1	0

$$A = W_5'$$

X \ W	0	1
0	0	1
1	1	0

$$B = (W_5 X_5') + (W_5' X_5)$$

$$C = 0$$

$$D = 0$$

F. Hours Tens (0-2)

X	W	D	C	B	A
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	0	0	0

X \ W	0	1
0	1	0
1	0	-

$$A = W_6' X_6'$$

X \ W	0	1
0	0	1
1	0	-

$$B = W_6 X_6'$$

$$C = 0$$
$$D = 0$$

VIII. Decrement Logic

A. Seconds Ones (0-9)

Z	Y	X	W	D	C	B	A
0	0	0	0	1	0	0	1
0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	1
0	0	1	1	0	0	1	0
0	1	0	0	0	0	1	1
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	1
0	1	1	1	0	1	1	0
1	0	0	0	0	1	1	1
1	0	0	1	1	0	0	0

ZY \ XW	00	01	11	10
00	1	0	0	1
01	1	0	0	1
11	-	-	-	-
10	1	0	-	-

$$A = W_1'$$

ZY \ XW	00	01	11	10
00	0	0	1	0
01	1	0	1	0
11	-	-	-	-
10	1	0	-	-

$$B = (X_1' W_1' ((Z_1' Y_1) + (Z_1 Y_1'))) + (Z_1' W_1 X_1)$$

ZY \ XW	00	01	11	10
00	0	0	0	0
01	0	1	1	1
11	-	-	-	-
10	1	0	-	-

$$C = (Z_1' Y_1 (X_1 + W_1)) + (Z_1 X_1' W_1' Y_1')$$

ZY \ XW	00	01	11	10
00	1	0	0	0
01	0	0	0	0
11	-	-	-	-
10	0	1	-	-

$$D = X_1'Y_1'((Z_1W_1) + (Z_1'W_1'))$$

B. Seconds Tens (0-5)

Z	Y	X	W	D	C	B	A
0	0	0	0	0	1	0	1
0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	1
0	0	1	1	0	0	1	0
0	1	0	0	0	0	1	1
0	1	0	1	0	1	0	0

ZY \ XW	00	01	11	10
00	1	0	0	1
01	1	0	-	-
11	-	-	-	-
10	-	-	-	-

$$A = W_2'$$

ZY \ XW	00	01	11	10
00	0	0	1	0
01	1	0	-	-
11	-	-	-	-
10	-	-	-	-

$$B = (Y_2X_2'W_2') + (Y_2'X_2W_2)$$

ZY \ XW	00	01	11	10
00	1	0	0	0
01	0	1	-	-
11	-	-	-	-
10	-	-	-	-

$$C = X_2'((Y_2W_2) + (Y_2'W_2'))$$

$$D = 0$$

C. Minutes Ones (0-9)

Same as Seconds Ones with W3/X3/Y3/Z3.

D. Minutes Tens (0-5)

Same as Seconds Tens with W4/X4/Y4/Z4.

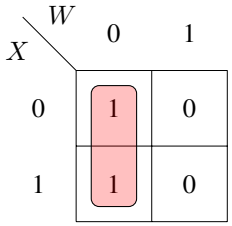
E. Hours Ones

I. Tens = 0/1 → 0-9

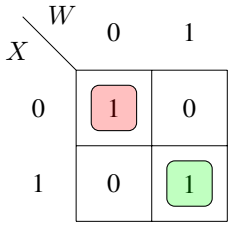
Same as Seconds Ones with W5/X5/Y5/Z5.

II. Tens = 2 → 0-3

X	W	D	C	B	A
0	0	0	0	1	1
0	1	0	0	0	0
1	0	0	0	0	1
1	1	0	0	1	0



$$A = W_5'$$



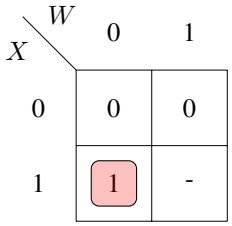
$$B = (X_5 W_5) + (X_5' W_5')$$

$$C = 0$$

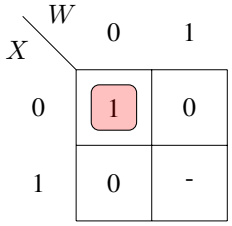
$$D = 0$$

F. Hours Tens (0-2)

X	W	D	C	B	A
0	0	0	0	1	0
0	1	0	0	0	0
1	0	0	0	0	1



$$A = X_6 W_6'$$



$$B = X_6' W_6'$$

$$C = 0$$

$$D = 0$$

IX. Control Implementation

1) Pressing Button 1 toggles between run mode and edit mode. In edit mode, the clock pauses.
2) In edit mode, pressing Button 2 selects the next digit for editing (cycles through all six digits).
3) In edit mode, pressing Button 3 increments the currently selected digit using the increment logic tables.
4) In edit mode, pressing Button 4 decrements the currently selected digit using the decrement logic tables.
5) The selected digit blinks at 5Hz (200ms on, 200ms off) for visual feedback.

X. Software Implementation

The Arduino code implements:

- Timer interrupt for clock ticking (10Hz interrupt rate)
- Button debouncing with software delays
- Multiplexed display refresh
- Editing mode with digit selection and value modification using the Boolean logic from the tables
- Proper constraints on time values (hours 0-23, minutes 0-59, seconds 0-59)

XI. Execution

A. Upload Code to Arduino

1) Connect Arduino to computer via USB
2) Open Arduino IDE
3) Copy the provided code into a new sketch
4) Select the correct board and port
5) Upload the code

B. Hardware Build

- Connect the seven-segment displays to the breadboard
- Connect all segment outputs together (through resistors)
- Make connections to the IC7447 according to Table 3.0
- Connect the IC7447 and the buttons to the Arduino according to Table 2.0
- Add appropriate current-limiting resistors for LEDs and pull-down resistors for buttons

Acknowledgment

The complete source code and documentation can be found at: https://github.com/Dhawal24112006/projects.git

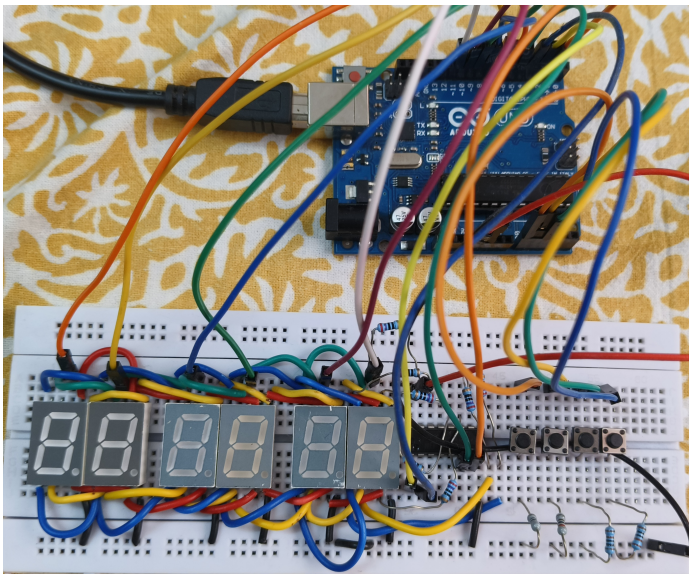


Fig. 1. Final Arduino-based Clock Implementation

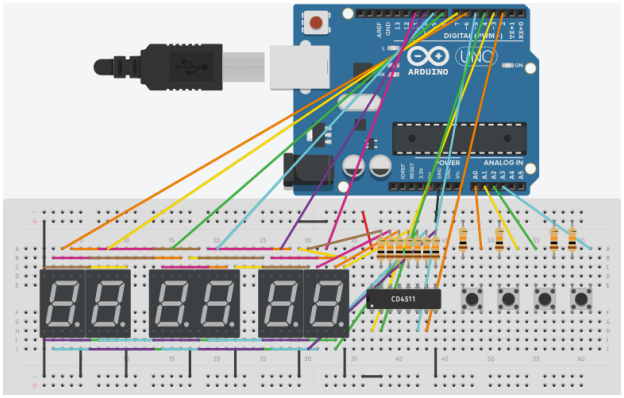


Fig. 2. Tinkercad Simulation of the Digital Clock