Abstract:

This project presents a simple web-based calculator designed using HTML, CSS, and JavaScript. The calculator provides basic arithmetic functionality, including addition, subtraction, multiplication, and division. The user interface is created with HTML for structure and styled with CSS for a clean and intuitive layout. JavaScript is used to implement the calculator's logic, enabling interactive features such as real-time calculations and input validation. This project demonstrates the integration of front-end web technologies to create a functional and user-friendly application. It serves as a foundational exercise for beginners in web development and highlights essential programming concepts such as event handling, DOM manipulation, and data processing in a browser environment.

Introduction:

A simple calculator is a fundamental web application that allows users to perform basic arithmetic operations such as addition, subtraction, multiplication, and division. This project demonstrates how to build a calculator using core web technologies—HTML for the structure, CSS for styling, and JavaScript for the logic and interactivity.

The main goal of this calculator is to provide an easy-to-use interface where users can input numbers and operators to perform quick calculations directly in their web browser. By developing this project, beginners can gain practical experience in designing user interfaces, handling user inputs, and implementing logical operations, all of which are essential skills in front-end web development.

This calculator not only helps reinforce essential programming concepts but also lays the groundwork for more advanced web applications in the future.

Objective:

The objective of this project is to design and develop a simple web-based calculator using HTML, CSS, and JavaScript that can perform basic arithmetic operations including addition, subtraction, multiplication, and division. The calculator aims to provide a user-friendly interface that allows users to input numbers and mathematical operators to perform quick and accurate calculations. This project serves as a foundational exercise in web development, helping learners understand the structure of web pages, styling elements, and implementing interactivity through scripting.

Methodology:

The development of the simple calculator was carried out using a structured approach involving three main web technologies: HTML, CSS, and JavaScript.

1. **Planning and Design**:
   The project began with planning the layout of the calculator interface. The design was kept simple and minimal to focus on functionality. A standard calculator layout was chosen with number buttons (0–9), basic operators (+, −, ×, ÷), a clear (C) button, and an equals (=) button.
2. **HTML Structure**:
   HTML was used to create the basic structure of the calculator. This included a display screen for showing input and results, and a grid of buttons for numbers and operations. Semantic elements and proper organization ensured readability and ease of styling.
3. **CSS Styling**:
   CSS was used to enhance the visual appearance of the calculator. Styling included layout alignment, button design, hover effects, and overall responsiveness to make the calculator more user-friendly and visually appealing.

4. **JavaScript Functionality**:
   JavaScript was implemented to handle the calculator's logic. Event listeners were added to each button to capture user input. Functions were written to perform arithmetic operations, update the display, and manage edge cases such as division by zero or multiple operator inputs.
5. **Testing and Debugging**:
   The final step involved testing the calculator for accuracy and reliability. Common input patterns were tested, and any bugs or unexpected behaviours were corrected to ensure smooth functionality.

Source code:

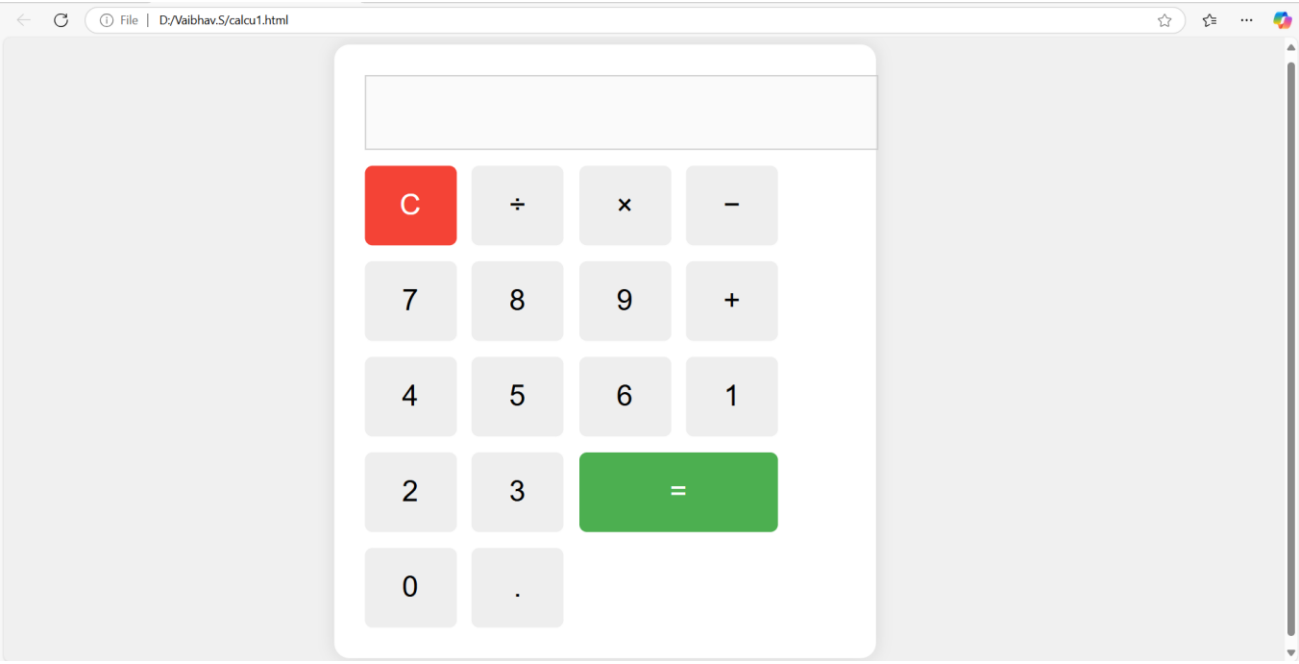
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Simple Calculator</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background: #f0f0f0;
      display: flex;
      height: 100vh;
      align-items: center;
      justify-content: center;
    }

    .calculator {
      background: #fff;
      padding: 20px;
      border-radius: 10px;
      box-shadow: 0 0 10px rgba(0,0,0,0.1);
    }

    #result {
      width: 100%;
      padding: 10px;
      font-size: 1.5em;
      text-align: right;
      margin-bottom: 10px;
      border: 1px solid #ccc;
    }

    .buttons {
      display: grid;
```

```
    grid-template-columns: repeat(4, 60px);
    gap: 10px;
   }

   button {
    padding: 15px;
    font-size: 1.2em;
    cursor: pointer;
    border: none;
    background: #eee;
    border-radius: 5px;
    transition: background 0.2s;
   }

   button:hover {
    background: #ddd;
   }

   .equal {
    grid-column: span 2;
    background-color: #4CAF50;
    color: white;
   }

   .clear {
    background-color: #f44336;
    color: white;
   }
  </style>
</head>
<body>

<div class="calculator">
  <input type="text" id="result" disabled>
  <div class="buttons">
    <button onclick="clearResult()" class="clear">C</button>
    <button onclick="appendValue('/')">÷</button>
```

```
    <button onclick="appendValue('*')">×</button>
    <button onclick="appendValue('-')">−</button>

    <button onclick="appendValue('7')">7</button>
    <button onclick="appendValue('8')">8</button>
    <button onclick="appendValue('9')">9</button>
    <button onclick="appendValue('+')">+</button>

    <button onclick="appendValue('4')">4</button>
    <button onclick="appendValue('5')">5</button>
    <button onclick="appendValue('6')">6</button>

    <button onclick="appendValue('1')">1</button>
    <button onclick="appendValue('2')">2</button>
    <button onclick="appendValue('3')">3</button>
    <button onclick="calculateResult()" class="equal">=</button>

    <button onclick="appendValue('0')">0</button>
    <button onclick="appendValue('.')">.</button>
  </div>
</div>

<script>
  function appendValue(value) {
    document.getElementById('result').value += value;
  }

  function clearResult() {
    document.getElementById('result').value = '';
  }

  function calculateResult() {
    try {
      let result = eval(document.getElementById('result').value);
      document.getElementById('result').value = result;
    } catch (error) {
      document.getElementById('result').value = 'Error';
```

```
    }
  }
</script>

</body>
</html>
```

Output:



Conclusion:

This simple calculator demonstrates the basic functionality of a web-based calculator using HTML, CSS, and JavaScript.

It allows users to perform basic arithmetic operations such as addition, subtraction, multiplication, and division.

This project is a great starting point for beginners to understand how web technologies work together to create interactive applications.

Reference:

```
<section style="text-align: center; margin-top: 40px;">
  <h2>References</h2>
  <ul style="list-style-type: none; padding: 0;">
    <li>
      <a href="https://developer.mozilla.org/en-
US/docs/Web/HTML" target="_blank">
        MDN Web Docs – HTML
      </a>
    </li>
    <li>
      <a href="https://developer.mozilla.org/en-
US/docs/Web/CSS" target="_blank">
        MDN Web Docs – CSS
      </a>
    </li>
    <li>
      <a href="https://developer.mozilla.org/en-
US/docs/Web/JavaScript" target="_blank">
        MDN Web Docs – JavaScript
      </a>
    </li>
    <li>
```

```
      <a
href="https://www.w3schools.com/js/js_calculator.asp"
target="_blank">
        W3Schools – JavaScript Calculator Tutorial
      </a>
    </li>
  </ul>
</section>
```

JSPM University Pune
Faculty of Science and Technology
School of Basic and Applied Sciences
Bachelor of Computer Applications

FULL STACK DEVELOPMENT PROJECT

SIMPLE CALCULATOR

NAME: SHUBHAM KAWADE
ROLL.NO: 61, DIV: B
PRN: 22356030138

Abstract:

This project presents a simple web-based calculator designed using HTML, CSS, and JavaScript. The calculator provides basic arithmetic functionality, including addition, subtraction, multiplication, and division. The user interface is created with HTML for structure and styled with CSS for a clean and intuitive layout. JavaScript is used to implement the calculator's logic, enabling interactive features such as real-time calculations and input validation. This project demonstrates the integration of front-end web technologies to create a functional and user-friendly application. It serves as a foundational exercise for beginners in web development and highlights essential programming concepts such as event handling, DOM manipulation, and data processing in a browser environment.

Introduction:

A simple calculator is a fundamental web application that allows users to perform basic arithmetic operations such as addition, subtraction, multiplication, and division. This project demonstrates how to build a calculator using core web technologies—HTML for the structure, CSS for styling, and JavaScript for the logic and interactivity.

The main goal of this calculator is to provide an easy-to-use interface where users can input numbers and operators to perform quick calculations directly in their web browser. By developing this project, beginners can gain practical experience in designing user interfaces, handling user inputs, and implementing logical operations, all of which are essential skills in front-end web development.

This calculator not only helps reinforce essential programming concepts but also lays the groundwork for more advanced web applications in the future.

Objective:

The objective of this project is to design and develop a simple web-based calculator using HTML, CSS, and JavaScript that can perform basic arithmetic operations including addition, subtraction, multiplication, and division. The calculator aims to provide a user-friendly interface that allows users to input numbers and mathematical operators to perform quick and accurate calculations. This project serves as a foundational exercise in web development, helping learners understand the structure of web pages, styling elements, and implementing interactivity through scripting.

Methodology:

The development of the simple calculator was carried out using a structured approach involving three main web technologies: HTML, CSS, and JavaScript.

1. **Planning and Design**:
   The project began with planning the layout of the calculator interface. The design was kept simple and minimal to focus on functionality. A standard calculator layout was chosen with number buttons (0–9), basic operators (+, −, ×, ÷), a clear (C) button, and an equals (=) button.
2. **HTML Structure**:
   HTML was used to create the basic structure of the calculator. This included a display screen for showing input and results, and a grid of buttons for numbers and operations. Semantic elements and proper organization ensured readability and ease of styling.
3. **CSS Styling**:
   CSS was used to enhance the visual appearance of the calculator. Styling included layout alignment, button design, hover effects, and overall responsiveness to make the calculator more user-friendly and visually appealing.

4. **JavaScript Functionality**:
   JavaScript was implemented to handle the calculator's logic. Event listeners were added to each button to capture user input. Functions were written to perform arithmetic operations, update the display, and manage edge cases such as division by zero or multiple operator inputs.
5. **Testing and Debugging**:
   The final step involved testing the calculator for accuracy and reliability. Common input patterns were tested, and any bugs or unexpected behaviours were corrected to ensure smooth functionality.

Source code:

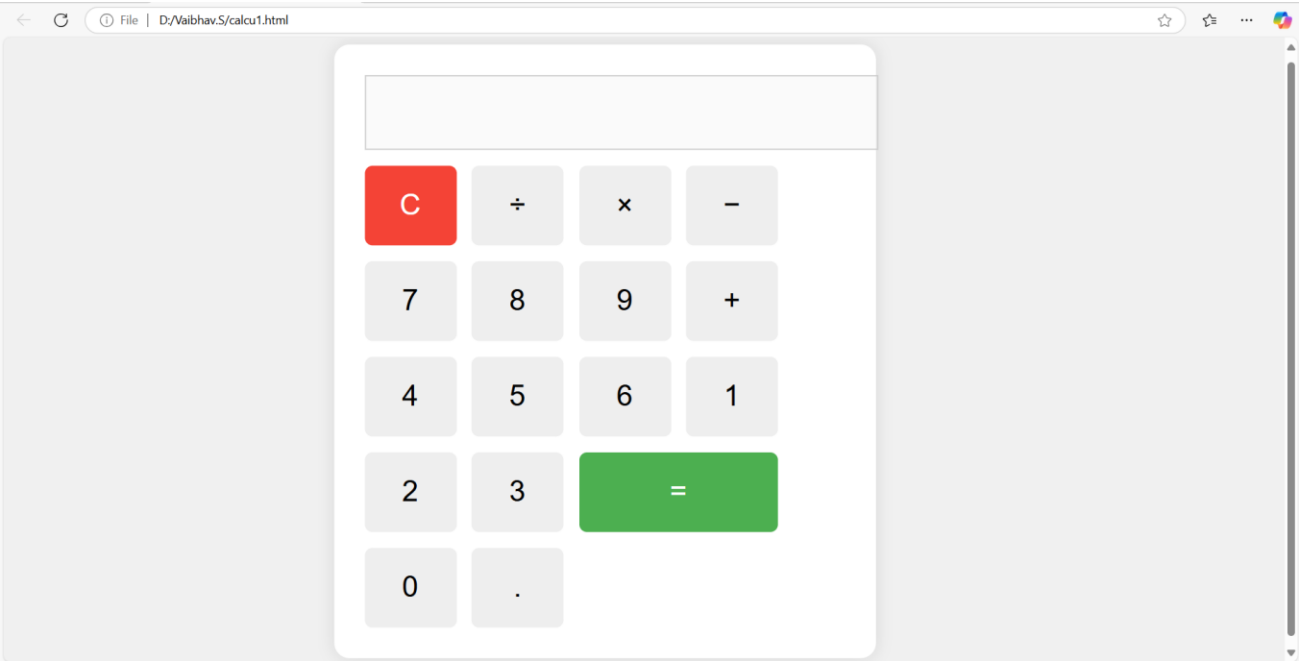
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Simple Calculator</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background: #f0f0f0;
      display: flex;
      height: 100vh;
      align-items: center;
      justify-content: center;
    }

    .calculator {
      background: #fff;
      padding: 20px;
      border-radius: 10px;
      box-shadow: 0 0 10px rgba(0,0,0,0.1);
    }

    #result {
      width: 100%;
      padding: 10px;
      font-size: 1.5em;
      text-align: right;
      margin-bottom: 10px;
      border: 1px solid #ccc;
    }

    .buttons {
      display: grid;
```

```
    grid-template-columns: repeat(4, 60px);
    gap: 10px;
   }

   button {
    padding: 15px;
    font-size: 1.2em;
    cursor: pointer;
    border: none;
    background: #eee;
    border-radius: 5px;
    transition: background 0.2s;
   }

   button:hover {
    background: #ddd;
   }

   .equal {
    grid-column: span 2;
    background-color: #4CAF50;
    color: white;
   }

   .clear {
    background-color: #f44336;
    color: white;
   }
  </style>
</head>
<body>

<div class="calculator">
  <input type="text" id="result" disabled>
  <div class="buttons">
    <button onclick="clearResult()" class="clear">C</button>
    <button onclick="appendValue('/')">÷</button>
```

```
    <button onclick="appendValue('*')">×</button>
    <button onclick="appendValue('-')">−</button>

    <button onclick="appendValue('7')">7</button>
    <button onclick="appendValue('8')">8</button>
    <button onclick="appendValue('9')">9</button>
    <button onclick="appendValue('+')">+</button>

    <button onclick="appendValue('4')">4</button>
    <button onclick="appendValue('5')">5</button>
    <button onclick="appendValue('6')">6</button>

    <button onclick="appendValue('1')">1</button>
    <button onclick="appendValue('2')">2</button>
    <button onclick="appendValue('3')">3</button>
    <button onclick="calculateResult()" class="equal">=</button>

    <button onclick="appendValue('0')">0</button>
    <button onclick="appendValue('.')">.</button>
  </div>
</div>

<script>
  function appendValue(value) {
    document.getElementById('result').value += value;
  }

  function clearResult() {
    document.getElementById('result').value = '';
  }

  function calculateResult() {
    try {
      let result = eval(document.getElementById('result').value);
      document.getElementById('result').value = result;
    } catch (error) {
      document.getElementById('result').value = 'Error';
```

```
    }
  }
</script>

</body>
</html>
```

Output:



Conclusion:

This simple calculator demonstrates the basic functionality of a web-based calculator using HTML, CSS, and JavaScript.

It allows users to perform basic arithmetic operations such as addition, subtraction, multiplication, and division.

This project is a great starting point for beginners to understand how web technologies work together to create interactive applications.

Reference:

```
<section style="text-align: center; margin-top: 40px;">
  <h2>References</h2>
  <ul style="list-style-type: none; padding: 0;">
    <li>
      <a href="https://developer.mozilla.org/en-
US/docs/Web/HTML" target="_blank">
        MDN Web Docs – HTML
      </a>
    </li>
    <li>
      <a href="https://developer.mozilla.org/en-
US/docs/Web/CSS" target="_blank">
        MDN Web Docs – CSS
      </a>
    </li>
    <li>
      <a href="https://developer.mozilla.org/en-
US/docs/Web/JavaScript" target="_blank">
        MDN Web Docs – JavaScript
      </a>
    </li>
    <li>
```

```
      <a
href="https://www.w3schools.com/js/js_calculator.asp"
target="_blank">
        W3Schools – JavaScript Calculator Tutorial
      </a>
    </li>
  </ul>
</section>
```

JSPM University Pune
Faculty of Science and Technology
School of Basic and Applied Sciences
Bachelor of Computer Applications

FULL STACK DEVELOPMENT PROJECT

SIMPLE CALCULATOR

NAME: SHUBHAM KAWADE
ROLL.NO: 61, DIV: B
PRN: 22356030138

Abstract:

This project presents a simple web-based calculator designed using HTML, CSS, and JavaScript. The calculator provides basic arithmetic functionality, including addition, subtraction, multiplication, and division. The user interface is created with HTML for structure and styled with CSS for a clean and intuitive layout. JavaScript is used to implement the calculator's logic, enabling interactive features such as real-time calculations and input validation. This project demonstrates the integration of front-end web technologies to create a functional and user-friendly application. It serves as a foundational exercise for beginners in web development and highlights essential programming concepts such as event handling, DOM manipulation, and data processing in a browser environment.

Introduction:

A simple calculator is a fundamental web application that allows users to perform basic arithmetic operations such as addition, subtraction, multiplication, and division. This project demonstrates how to build a calculator using core web technologies—HTML for the structure, CSS for styling, and JavaScript for the logic and interactivity.

The main goal of this calculator is to provide an easy-to-use interface where users can input numbers and operators to perform quick calculations directly in their web browser. By developing this project, beginners can gain practical experience in designing user interfaces, handling user inputs, and implementing logical operations, all of which are essential skills in front-end web development.

This calculator not only helps reinforce essential programming concepts but also lays the groundwork for more advanced web applications in the future.

Objective:

The objective of this project is to design and develop a simple web-based calculator using HTML, CSS, and JavaScript that can perform basic arithmetic operations including addition, subtraction, multiplication, and division. The calculator aims to provide a user-friendly interface that allows users to input numbers and mathematical operators to perform quick and accurate calculations. This project serves as a foundational exercise in web development, helping learners understand the structure of web pages, styling elements, and implementing interactivity through scripting.

Methodology:

The development of the simple calculator was carried out using a structured approach involving three main web technologies: HTML, CSS, and JavaScript.

1. **Planning and Design**:
   The project began with planning the layout of the calculator interface. The design was kept simple and minimal to focus on functionality. A standard calculator layout was chosen with number buttons (0–9), basic operators (+, −, ×, ÷), a clear (C) button, and an equals (=) button.
2. **HTML Structure**:
   HTML was used to create the basic structure of the calculator. This included a display screen for showing input and results, and a grid of buttons for numbers and operations. Semantic elements and proper organization ensured readability and ease of styling.
3. **CSS Styling**:
   CSS was used to enhance the visual appearance of the calculator. Styling included layout alignment, button design, hover effects, and overall responsiveness to make the calculator more user-friendly and visually appealing.

4. **JavaScript Functionality**:
   JavaScript was implemented to handle the calculator's logic. Event listeners were added to each button to capture user input. Functions were written to perform arithmetic operations, update the display, and manage edge cases such as division by zero or multiple operator inputs.
5. **Testing and Debugging**:
   The final step involved testing the calculator for accuracy and reliability. Common input patterns were tested, and any bugs or unexpected behaviours were corrected to ensure smooth functionality.

Source code:

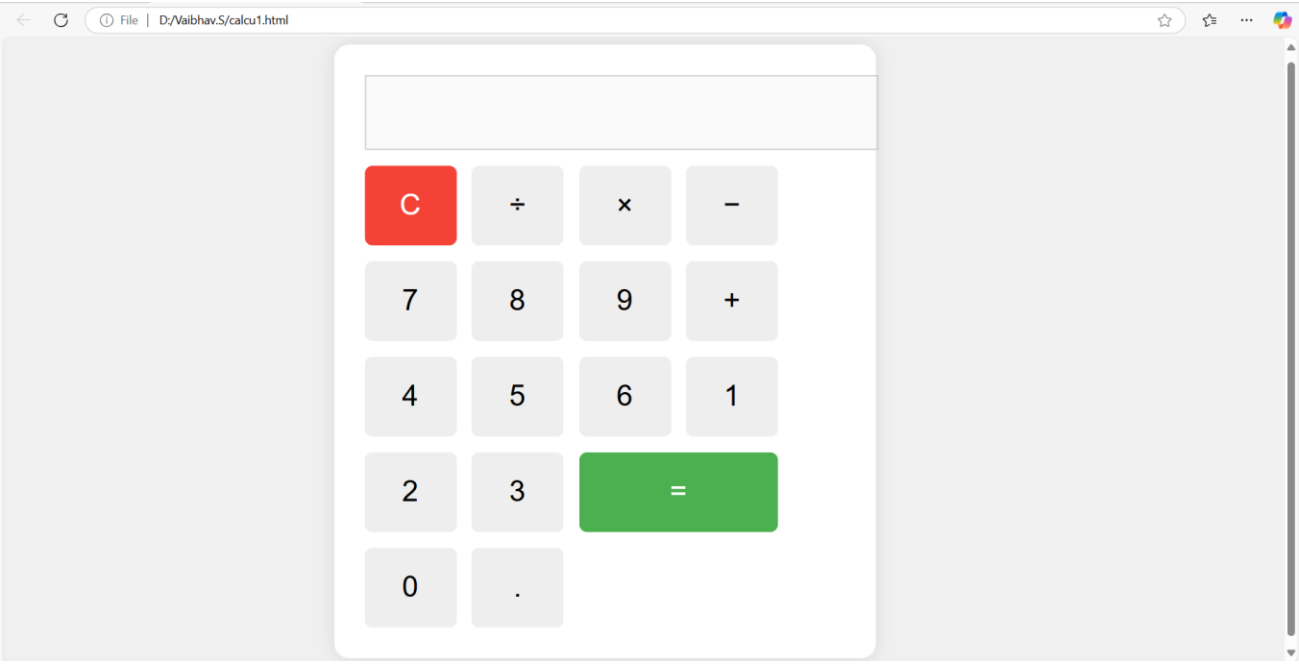
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Simple Calculator</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background: #f0f0f0;
      display: flex;
      height: 100vh;
      align-items: center;
      justify-content: center;
    }

    .calculator {
      background: #fff;
      padding: 20px;
      border-radius: 10px;
      box-shadow: 0 0 10px rgba(0,0,0,0.1);
    }

    #result {
      width: 100%;
      padding: 10px;
      font-size: 1.5em;
      text-align: right;
      margin-bottom: 10px;
      border: 1px solid #ccc;
    }

    .buttons {
      display: grid;
```

```
    grid-template-columns: repeat(4, 60px);
    gap: 10px;
   }

   button {
    padding: 15px;
    font-size: 1.2em;
    cursor: pointer;
    border: none;
    background: #eee;
    border-radius: 5px;
    transition: background 0.2s;
   }

   button:hover {
    background: #ddd;
   }

   .equal {
    grid-column: span 2;
    background-color: #4CAF50;
    color: white;
   }

   .clear {
    background-color: #f44336;
    color: white;
   }
  </style>
</head>
<body>

<div class="calculator">
  <input type="text" id="result" disabled>
  <div class="buttons">
    <button onclick="clearResult()" class="clear">C</button>
    <button onclick="appendValue('/')">÷</button>
```

```
    <button onclick="appendValue('*')">×</button>
    <button onclick="appendValue('-')">−</button>

    <button onclick="appendValue('7')">7</button>
    <button onclick="appendValue('8')">8</button>
    <button onclick="appendValue('9')">9</button>
    <button onclick="appendValue('+')">+</button>

    <button onclick="appendValue('4')">4</button>
    <button onclick="appendValue('5')">5</button>
    <button onclick="appendValue('6')">6</button>

    <button onclick="appendValue('1')">1</button>
    <button onclick="appendValue('2')">2</button>
    <button onclick="appendValue('3')">3</button>
    <button onclick="calculateResult()" class="equal">=</button>

    <button onclick="appendValue('0')">0</button>
    <button onclick="appendValue('.')">.</button>
  </div>
</div>

<script>
  function appendValue(value) {
    document.getElementById('result').value += value;
  }

  function clearResult() {
    document.getElementById('result').value = '';
  }

  function calculateResult() {
    try {
      let result = eval(document.getElementById('result').value);
      document.getElementById('result').value = result;
    } catch (error) {
      document.getElementById('result').value = 'Error';
```

```
    }
  }
</script>

</body>
</html>
```

Output:



Conclusion:

This simple calculator demonstrates the basic functionality of a web-based calculator using HTML, CSS, and JavaScript.

It allows users to perform basic arithmetic operations such as addition, subtraction, multiplication, and division.

This project is a great starting point for beginners to understand how web technologies work together to create interactive applications.

Reference:

```
<section style="text-align: center; margin-top: 40px;">
  <h2>References</h2>
  <ul style="list-style-type: none; padding: 0;">
    <li>
      <a href="https://developer.mozilla.org/en-
US/docs/Web/HTML" target="_blank">
        MDN Web Docs – HTML
      </a>
    </li>
    <li>
      <a href="https://developer.mozilla.org/en-
US/docs/Web/CSS" target="_blank">
        MDN Web Docs – CSS
      </a>
    </li>
    <li>
      <a href="https://developer.mozilla.org/en-
US/docs/Web/JavaScript" target="_blank">
        MDN Web Docs – JavaScript
      </a>
    </li>
    <li>
```

```
      <a
href="https://www.w3schools.com/js/js_calculator.asp"
target="_blank">
        W3Schools – JavaScript Calculator Tutorial
      </a>
    </li>
  </ul>
</section>
```

JSPM University Pune
Faculty of Science and Technology
School of Basic and Applied Sciences
Bachelor of Computer Applications

FULL STACK DEVELOPMENT PROJECT

SIMPLE CALCULATOR

NAME: SHUBHAM KAWADE
ROLL.NO: 61, DIV: B
PRN: 22356030138

PAGE NO.
DATE:

Assignment No:-2

Q.1) Name SFRs associated with each I/O part of PIC18F What is role of PORTx sfr?

→ SFRs associated with I/O port in PIC18F - In PIC18F series, each I/O port is controlled by several special function Register.

1) PORT → Register → stores actual i/o data of port.
2) TRISx Register → configures each pin as either input (1) or output (0).
3) LATx Register → used for latching o/p values to reduce read-modify write issue
4) ANSELx Register → configure pin as analog or digital.

- Role of PORTx SFR -
It act as buffer that stores current state of port. When reading, it show actual logic level at pin & when writing it changes output value.

Q.2) Calculate total delay generated by timer0 if (FFF1)H is loaded into it. Assume Crystal Frequency = 10 MHz.

→ Given,
Crystal freq = 10 MHz
Timer 0 prescaler = 1
Timer 0 is 16 bit
Timer load value = FFF1H (65521 in decimal)
- Timer clock freq =
$F_{timer} = \frac{F_{oscillator}}{4} = \frac{10}{4} MHz = 2.5 MHz$

PAGE NO.
DATE:

-Timer clock period

$$\text{Timer} = \frac{1}{F_{timer}} = \frac{1}{2.5MHz} = 0.4\mu s$$

Tcount = 65536 - initial value
= 65536 - 65521
= 15

Total delay = Tcount x timer
= 15 x 0.4
= 6 µs

Q.3) Explain Working of PIC18F timer1 with help of suitable diagram.

→ Timer is 16 bit timer & its 16 bit register is split into 2 bytes refer as TMR1L & TMR1H -Timer1 can be programmed 16 bit mode only & unlike timer 0 it does not support 8 bit mode. timer 0 it does not support 8-b' Timer1 also has T1CON register in addition to TRRIFE flag bit goes high. when TMR1H : TMR1L overflow from FFFF to 0000 timer1 also has prescaler option built. only supports factors of 1:1, 1:2, 1:4 & 1:8

PAGE NO.
DATE:

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
TMR1H								TMR1L							

set TMR1P flag bit

Timer High & low register.

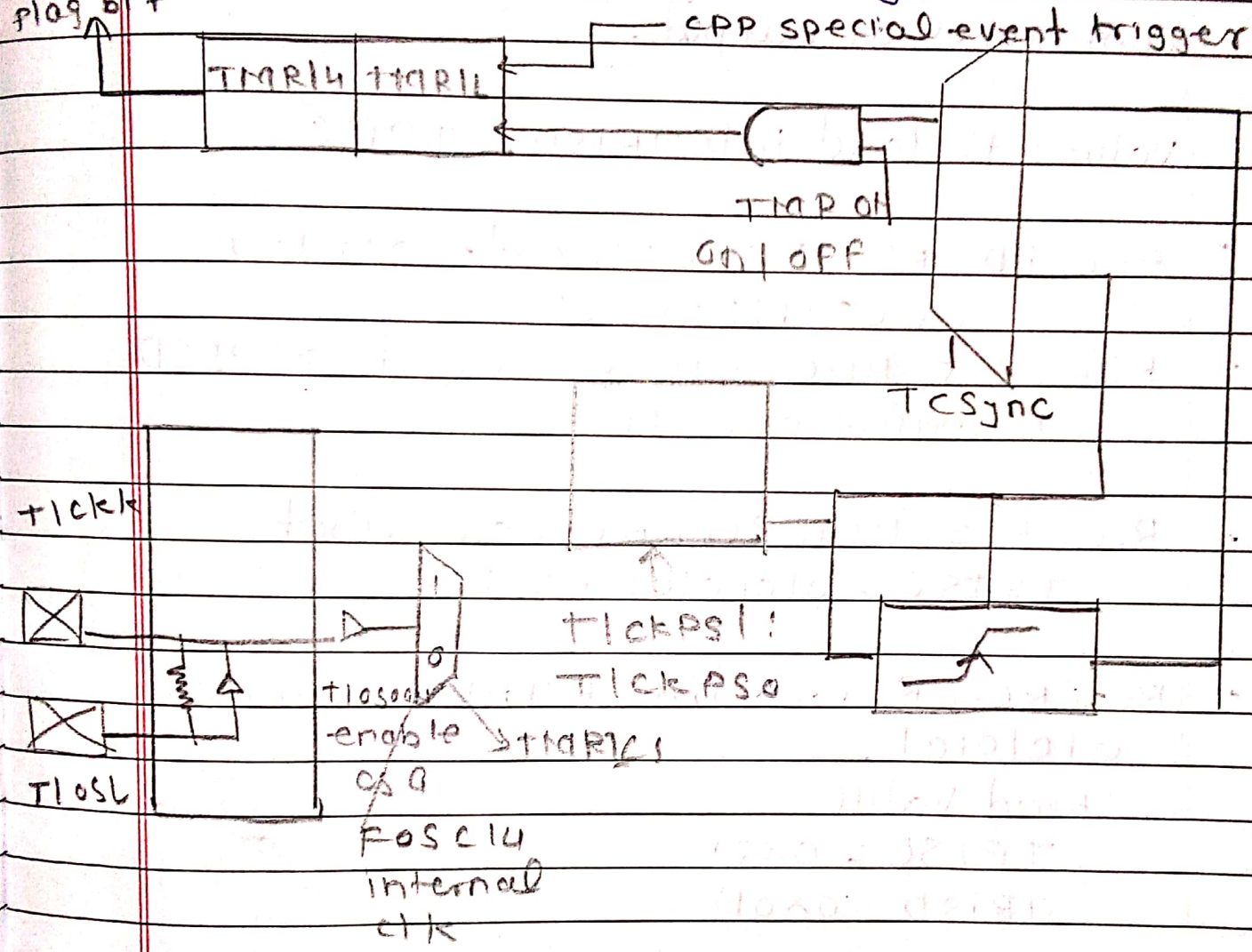


Fig Timer1 block Diagram.

Q.4) What is role of TRIS X SFR in PIC18F? Find value to be loaded in ~~TRISD~~ & TRISC register for following.

RD0, RD1, RD2, RD3, as input port
RD4, RD5, RD6, RD7, as output
RC0, RC2, RC4, RC6, RC7 as output
RC1, RC3, RC5, as input.

→ Role of TRIS X Register -
- TRIS X reg determines whether pin function as Input or output.
- Writing 1 configures pin as input & 0 configures as output.

Values to load into TRISD & TRISC -

- RD0, RD1, RD2, RD3 as input → TRISD:
  00001111 = 0x0F
- RD4, RD5, RD6, RD7 as output → TRISD
  Remains 00001111

- RC0, RC2, RC4, RC6, RC7, as output -
  TRISC = 01010101 = 0x55

- RC1, RC3, RC5 as input - TRISC remains 01010101,
  Find Value.
  TRISC = 0x55
  TRISD = 0x0F

Q.5) Explain in detail prescaling & postscaling of PIC18F timer.

→ • prescaler -
- It divides input clock before feeding it to timer.
- It helps extend timer range by reducing frequency
- Available prescaler Value - 1:2, 1:4, 1:8, 1:16 etc
- The division factor is programmable typically through specific bits in timer contrd

register.
- A higher prescaler value result in slower timer increment rate, which is useful for measuring longer duration but reduce timers resolution

* Postscaler.
- It operates after time counting register & control frequency of interrupt generation by dividing no of timer overflow required to trigger an interrupt.
- similar to prescaler, postscaler division factor is programmble via specific bit in timer control register.

- e.g - Timer in PIC18F4550 allow for a postscaler setting ranging from 1:1 to 1:16.

Q.6) Draw format of T0CON register & explain functionality of each bit.

Bit	7	6	5	4	3	2	1	0
Name	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0

Fig. Format of T0CON reg.

- Functionality
- TMR0ON (bit 7) - enable timer0 (1=ON 3=OFF)
- T08BIT (bit 6) - 1 = 8 bit mode, 0 = 16 bit
- T0CS (bit 5) = 1 = external clk, 0 = internal clk
- T0CS (bit 4) - select edge for exter clk

(1=high to low, 0=low to high)

- PSA (bit 3) : 1=no prescaler, 0=prescaler assigned.

- T0PS2 : T0PS0 (bit 2-0) - prescaler value selection (000 = 1:2, 111 = 1:256)

Q.7) Explain working of PIC18F timer 0 in 16 bit mode with suitable diagram.

→ Timer 0 -

It can be used as 8 bit or 16 bit timer
- The 16 bit register of timer 0 is accessed as low byte & high byte as shown
- The low byte reg. is called TMR0L & high register is called TMR0H.

eg - "MOVWF TMR0L" moves value of WREG into TMR0L the low byte of timer. These register can also be read like any other register.

eg - MOVFF TMR0L, PORTB
It copies TMR0L to PORTB

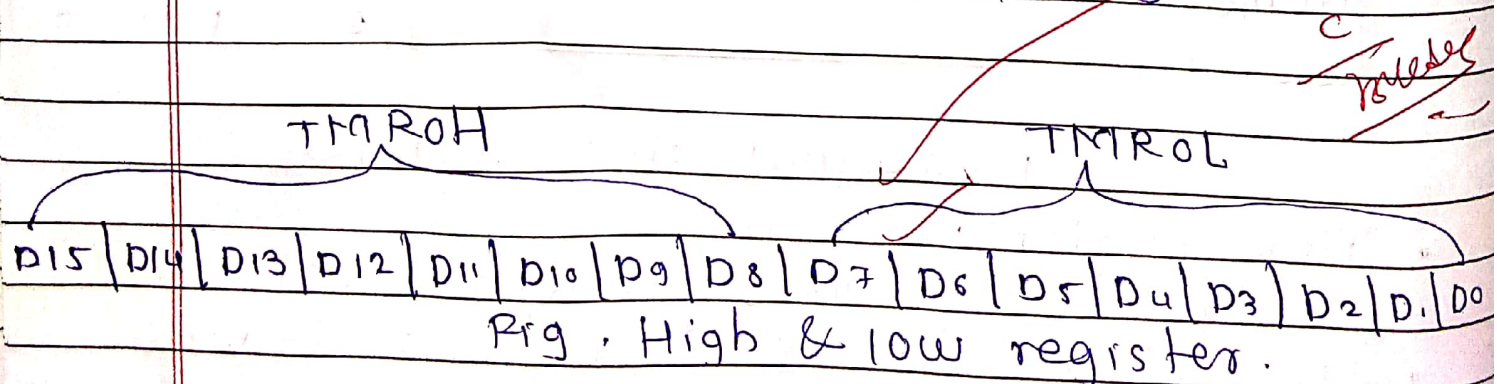


Fig. High & low register.

PAGE NO.
DATE:

Assignment No:-2

Q.1) Name SFRs associated with each I/O part of PIC18F What is role of PORTx sfr?

→ SFRs associated with I/O port in PIC18F - In PIC18F series, each I/O port is controlled by several special function Register.

1) PORT → Register → stores actual i/o data of port.
2) TRISx Register → configures each pin as either input (1) or output (0).
3) LATx Register → used for latching o/p values to reduce read-modify write issue
4) ANSELx Register → configure pin as analog or digital.

- Role of PORTx SFR -
It act as buffer that stores current state of port. When reading, it show actual logic level at pin & when writing it changes output value.

Q.2) Calculate total delay generated by timer0 if (FFF1)H is loaded into it. Assume Crystal Frequency = 10 MHz.

→ Given,
Crystal freq = 10 MHz
Timer 0 prescaler = 1
Timer 0 is 16 bit
Timer load value = FFF1H (65521 in decimal)
- Timer clock freq =
$F_{timer} = \frac{F_{oscillator}}{4} = \frac{10}{4} MHz = 2.5 MHz$

PAGE NO.
DATE:

-Timer clock period

$$\text{Timer} = \frac{1}{F_{timer}} = \frac{1}{2.5MHz} = 0.4\mu s$$

Tcount = 65536 - initial value
= 65536 - 65521
= 15

Total delay = Tcount x timer
= 15 x 0.4
= 6 µs

Q.3) Explain Working of PIC18F timer1 with help of suitable diagram.

→ Timer is 16 bit timer & its 16 bit register is split into 2 bytes refer as TMR1L & TMR1H -Timer1 can be programmed 16 bit mode only & unlike timer 0 it does not support 8 bit mode. timer 0 it does not support 8-b' Timer1 also has T1CON register in addition to TRRIFE flag bit goes high. when TMR1H : TMR1L overflow from FFFF to 0000 timer1 also has prescaler option built. only supports factors of 1:1, 1:2, 1:4 & 1:8

PAGE NO.
DATE:

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
TMR1H								TMR1L							

set TMR1P flag bit

Timer High & low register.

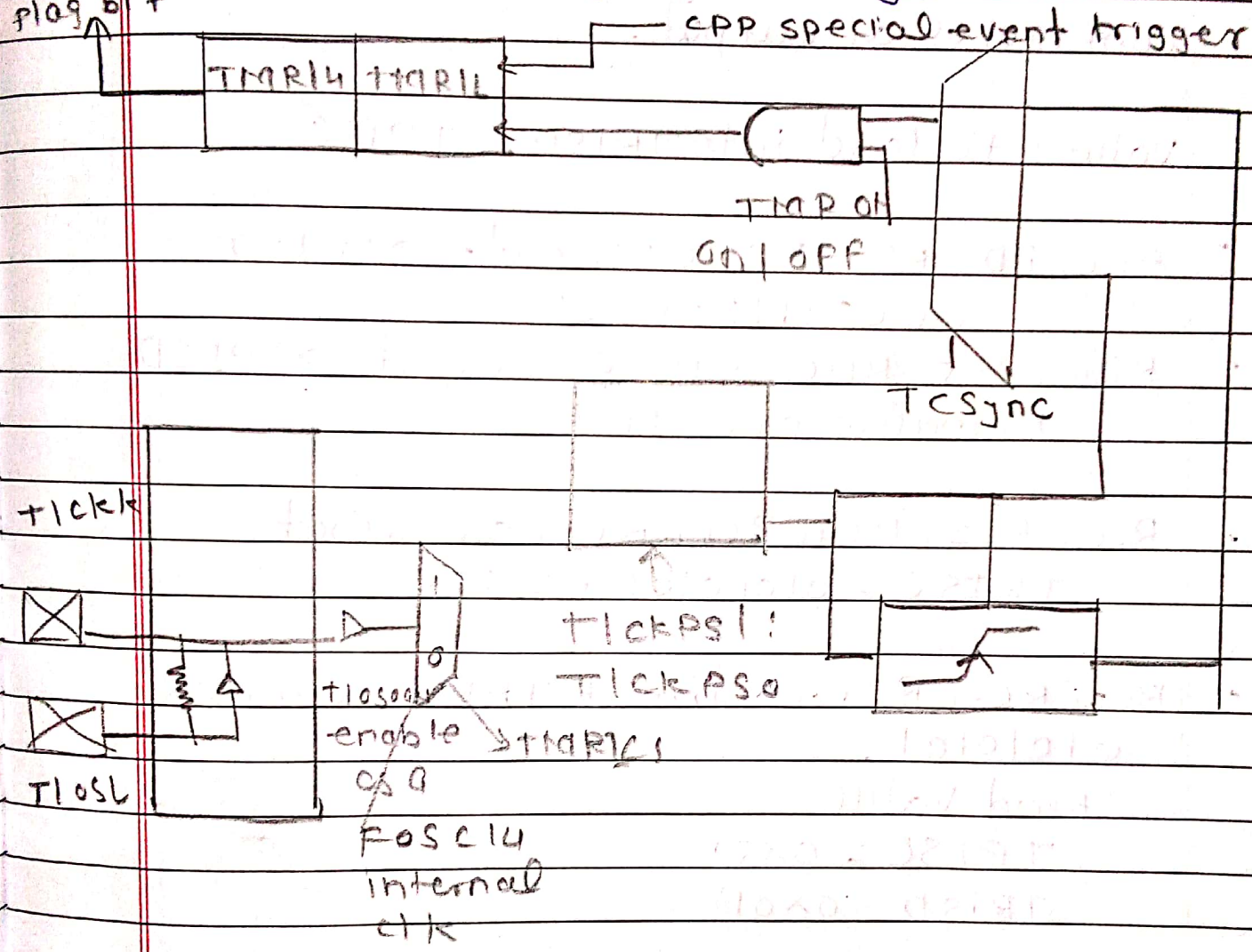


Fig Timer1 block Diagram.

Q.4) What is role of TRIS X SFR in PIC18F? Find value to be loaded in ~~TRISD~~ & TRISC register for following.

RD0, RD1, RD2, RD3, as input port
RD4, RD5, RD6, RD7, as output
RC0, RC2, RC4, RC6, RC7 as output
RC1, RC3, RC5, as input.

→ Role of TRIS X Register -
- TRIS X reg determines whether pin function as Input or output.
- Writing 1 configures pin as input & 0 configures as output.

Values to load into TRISD & TRISC -

- RD0, RD1, RD2, RD3 as input → TRISD:
  00001111 = 0x0F
- RD4, RD5, RD6, RD7 as output → TRISD
  Remains 00001111

- RC0, RC2, RC4, RC6, RC7, as output -
  TRISC = 01010101 = 0x55

- RC1, RC3, RC5 as input - TRISC remains 01010101,
  Find Value.
  TRISC = 0x55
  TRISD = 0x0F

Q.5) Explain in detail prescaling & postscaling of PIC18F timer.

→ • prescaler -
- It divides input clock before feeding it to timer.
- It helps extend timer range by reducing frequency
- Available prescaler Value - 1:2, 1:4, 1:8, 1:16 etc
- The division factor is programmable typically through specific bits in timer contrd

register.
- A higher prescaler value result in slower timer increment rate, which is useful for measuring longer duration but reduce timers resolution

* Postscaler.
- It operates after time counting register & control frequency of interrupt generation by dividing no of timer overflow required to trigger an interrupt.
- similar to prescaler, postscaler division factor is programmble via specific bit in timer control register.

- e.g - Timer in PIC18F4550 allow for a postscaler setting ranging from 1:1 to 1:16.

Q.6) Draw format of T0CON register & explain functionality of each bit.

Bit	7	6	5	4	3	2	1	0
Name	TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0

Fig. Format of T0CON reg.

- Functionality
- TMR0ON (bit 7) - enable timer0 (1=ON 3=OFF)
- T08BIT (bit 6) - 1 = 8 bit mode, 0 = 16 bit
- T0CS (bit 5) = 1 = external clk, 0 = internal clk
- T0CS (bit 4) - select edge for exter clk

(1=high to low, 0=low to high)

- PSA (bit 3) : 1=no prescaler, 0=prescaler assigned.

- T0PS2 : T0PS0 (bit 2-0) - prescaler value selection (000 = 1:2, 111 = 1:256)

Q.7) Explain working of PIC18F timer 0 in 16 bit mode with suitable diagram.

→ Timer 0 -

It can be used as 8 bit or 16 bit timer
- The 16 bit register of timer 0 is accessed as low byte & high byte as shown
- The low byte reg. is called TMR0L & high register is called TMR0H.

eg - "MOVWF TMR0L" moves value of WREG into TMR0L the low byte of timer. These register can also be read like any other register.

eg - MOVFF TMR0L, PORTB
It copies TMR0L to PORTB

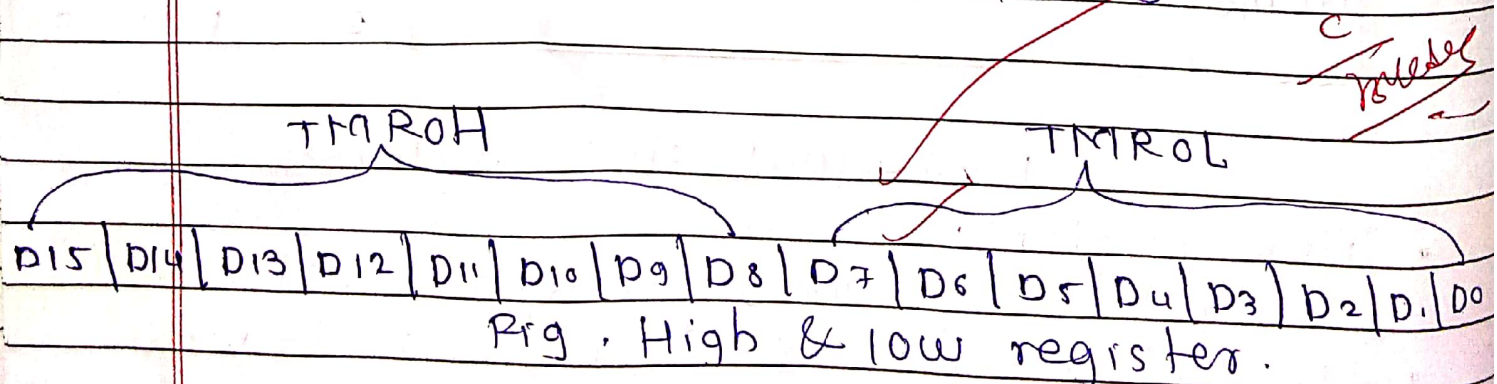


Fig. High & low register.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.53
Volume 4, Issue 2, January 2024

Automated Bus e-Ticketing Service

Prof. M. S. Malkar[1], Meet Mundada[2], Atharv Patil[3], Gaurav Phatak[4], Sairam Vaidya[5], Ajinkya Salunke[6]
Professor, Department of Computer Engineering[1]
Students, Department of Computer Engineering[2.3.4.5,6]
Pimpri Chinchwad Polytechnic, Pune, Maharashtra, India

Abstract: *Nowadays public transport systems like metro are well advanced. The need to improve passenger safety, convenience and performance of existing public transport is increasing the demand for intelligent transportation systems in the market. The paper- based ticketing system for collecting bus fares has been considered the source of major financial losses in India. It is difficult to assureevery passenger to buy a ticket. A paper ticket becomes useless for passengers upon reaching the destination. The number of untold tickets per day is very high. In the era of technology, India should focus on developing an automated system to collect bus fares. Therefore, this paper proposes an automated card operated system using RFID and GPS for bus travel in India.*

Keywords: Bus e-Ticketing

I. INTRODUCTION

Today, everything in the world is becoming smart and digital, with significant advances in the transport sector. However, public transport buses in India have not kept pace with these new developments. Intelligent vehicle research has been an active area of work, especially in the contextof public transport.

Traditionally, every bus has been controlled by a conductor who collects money from each passenger and issues a ticket. To address this, we are proposing an IoT-based ticketing system. Theprimary objective of this project is to use IR sensors to automatically count passengers and GPS sensors to calculate the distance travel by each passenger. The corresponding fare will then be debited from the RFID card.

II. LITERATURE SURVEY

In general, buses are typically managed by conductors who collect money from each passenger and issue tickets. Initially, printed papers or tokens were used as tickets. Nowadays, hand-operated machines are employed for ticket printing. However, this system has several disadvantages. Passengers must carry the ticket until they reach their destination, conductors need to ensure that everyone has a valid ticket, and the ticketing process takes a comparatively long time, requiring more paper.

For example, when a passenger wants to travel on a bus, they need to carry money. The conductorcollects the money and issues a ticket, a process that every passenger must repeat. This results in increased time, resource waste, and energy consumption.

The data pertains to an Automatic Fare Collection (AFC) system integrated with an Automated Vehicle Location (AVL) system that records each passenger's transaction when boarding the bus. This data includes information about the route, vehicle, travel card, as well as the time and place where the journey begins. While some of this data is recorded for on-board ticket inspection, it alsoenables innovative spatial verification features introduced by the methodology.

The integration of the Internet of Things (IoT) in Automatic Fare Collection is crucial. An AFCsystem consists of automated gate machines, ticket vending machines, and ticket checking machines. A stable and integrated platform is essential for smooth passenger flow during peakhours, ensuring that all data is collected and transmitted to the server.

In recent progress, RFID-based automatic bus ticketing has seen remarkable development in various technologies for public welfare, especially in the field of public transport. RF modules,particularly Radio Frequency Identification Devices (RFID), are gaining prominence for their potential in the near and distant future of public transport bus systems.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.53
Volume 4, Issue 2, January 2024

III. MODULE IDENTIFICATION

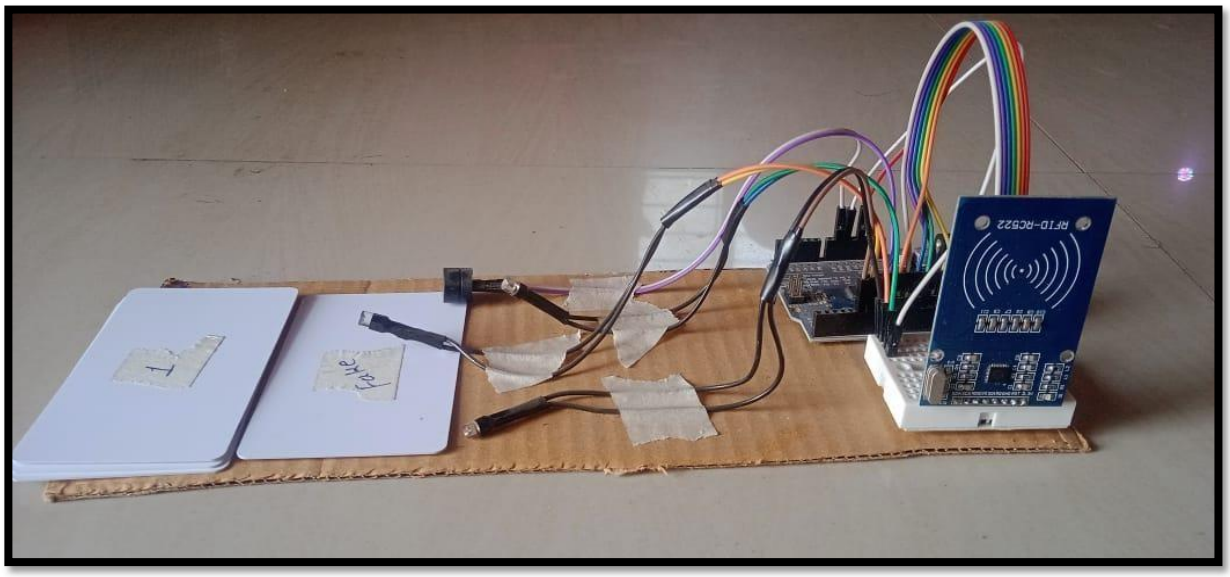
The project is designed for bus passengers traveling within the city. Users can view daily ticket transactions through the app. Each passenger is issued an RFID card, which they must swipe on theRFID reader upon boarding the bus. Additionally, they need to swipe the card again when reachingtheir destination.

The primary goal of this project is to utilize an IR sensor to count passengers and a GPS sensor toautomatically calculate the distance travelled. The corresponding fare is then debited from the RFID card, eliminating the need for a paper-based ticketing system. This approach ensures a seamless process with no conductor interference. Ticket history is stored in a database, and transaction counts are automated.

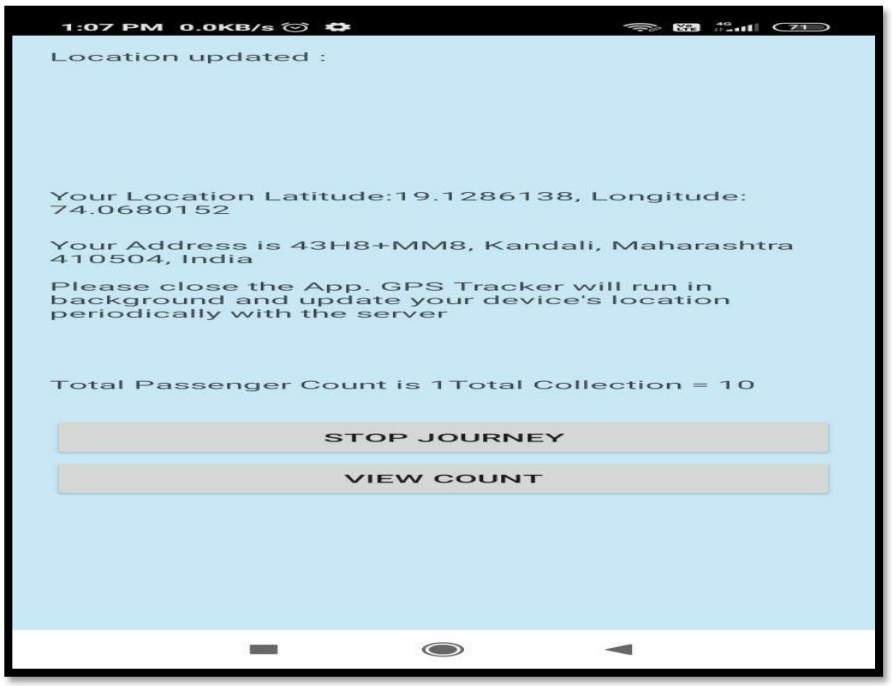
The operational steps are as follows:

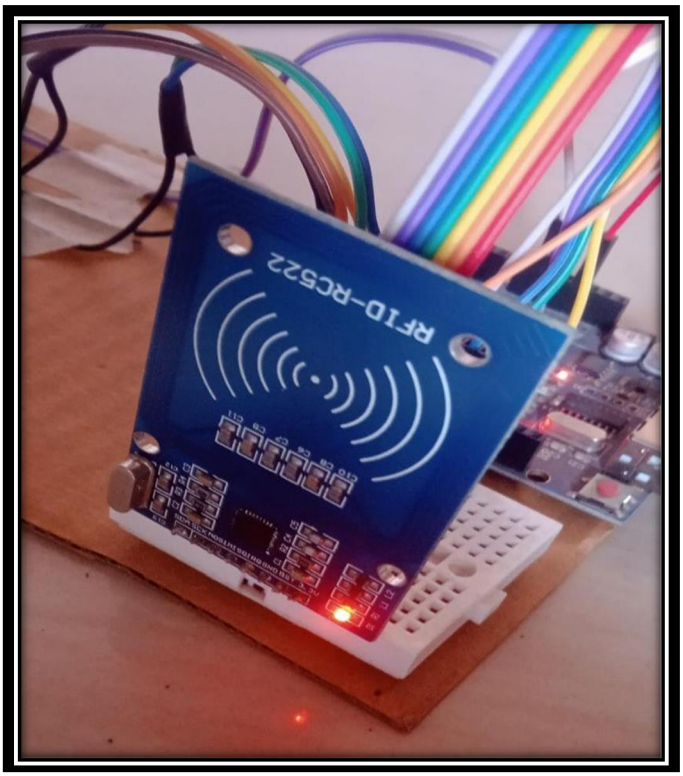
- Swipe the RFID card upon passenger boarding.
- Store GPS location during the swipe.
- Send an SMS to the traveler with the GPS location.
- The passenger swipes the RFID card upon reaching the destination.
- Calculate the distance from the stored GPS location (Step 2).
- Deduct the appropriate fare

IV. IMPLEMENTATION RESULT



IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.53
Volume 4, Issue 2, January 2024





V. MODULE DESCRIPTION

The bus is used by regular bus passengers to make frequent intercity travel trips at a profitablecost compared to daily bus fares.

Bus ticketing is done manually with no computerized user detail record.To overcome this, we have decided on our project topic

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.53
Volume 4, Issue 2, January 2024

Every bus is controlled by a conductor. The conductor will collect money from each passenger and issue a ticket. To overcome this, we will create IOT based ticketing system

VI. CONCLUSION

The project will be presented as a fully automated, reliable, transparent and convenient system for ticketing.
RFID cards can be reusable, much more convenient than paper-based ticketing systems.

REFERENCES

[1]. Dr. Vinit Kotak, "RFID-based bus ticketing system using android and GTFS", International Journal of Advanced Research in Computer and Communication Engineering (IJARCCE) Vol. 5, Issue 3, March 2016.
[2]. V. Apsara, "RFID based bus ticketing system for Public Transport System (PTS)", International Journal of Industrial Electronics and Electrical Engineering (IJIEEE) Vol. 4, Issue 5, May 2016.
[3]. Mr. Mohammad Osman , "Enhancement of Public Transportation services using wireless technologies like Zigbee, RFID ,GSM and GPS" ,International Journal of Engineering Trends and Technology (IJETT) Vol. 6, No. 7 ,December 2013.
[4]. T. Manikandan, "Conductor less bus ticketing system using RFID and accident information through GPS and GSM", International Journal of Innovative Science, Engineering and Technology (IJISET), Vol. 2, Issue 9, September 2015.
[5]. Paul Hamilton, "Intelligent agent based RFID system for demand bus Scheduling and T International Journal of Future Computer and Communication (IJFCC), Vol. 2, No. 5, October 2013.
[6]. Dr. Bos Jos, "RFID based bus ticketing system", International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering (IJAREEIE), Vol. 4, Issue 4, April 2015. [7]. Christian Oberli, "Performance Evaluation of UHF RFID technologies for real time passenger recognition in PTS", IEEE Transactions on Intelligent Transportation System, Vol. 11, Issue 3, September 2010.
[8]. Prof. K. T. Patil, "RFID Based Ticketing System For Local Trains", International Journal of Computer Science and Information Technologies (IJCSIT), Vol. 6, Issue 3, 2015.
[9]. Ana Aguiar, "Personal Navigator for a public transport system using RFID ticketing". [10]. Arul Das, "GPS Based Automated Public Transport Fare Collection Systems Based On Distance Travelled By Passenger Using Smart Card.
[11] Vinod Bharat et al. "Study of Detection of Various types of Cancers by using Deep Learning: A Survey", International Journal of Advanced Trends in Computer Science and Engineering, 2019, Volume 8 Issue 4,pp 1228-1233
[12] Vinod Bharat et al. "A review paper on data mining techniques", International Journal of Engineering Science and Computing (IJESC), 2016, Volume 6 Issue 5, pp 6268-6271.
[13] V Bharat, S Shubham, D Jagdish, P Amol and K Renuka, "Smart water management system in cities", 2017 International Conference on Big Data Analytics and Computational Intelligence (ICBDAC), 2017, March.
[14] Vinod Bharat, Sandeep Mali, KishorSawant and NileshThombare. Article: A Survey on Public Batch Auditing Protocol for Data Security. IJCA Proceedings on National Conference on Advances in Computing NCAC 2015(7):39-42, December 2015

Contact

7249599164 (Mobile)
patilatharv625@gmail.com

www.linkedin.com/in/atharv-patil-114143227 (LinkedIn)

Top Skills

java
Data Analytics
GitHub

Certifications

Introduction to Programming Using HTML
Introduction to Javascript
Google Cloud Computing Foundations Certificate
Data Analysis with Python
Prompt Design in Vertex AI

Publications

Automated Bus e-Ticketing Service

Atharv Patil

Aspiring Software Developer | SPPU'27 | BSITOR | Cloud Computing | AI/ML| DSA
Pune, Maharashtra, India

Summary

I have completed my Diploma in Computer Engineering and am currently pursuing a Bachelor’s degree in Information Technology. My journey in the tech field has been driven by a passion for problem-solving and innovation.

I have a strong foundation in programming languages such as Java and Python, with a growing interest in Artificial Intelligence (AI) and Cloud Computing. My experience includes exploring machine learning, natural language processing, and designing scalable solutions using cloud platforms like Google Cloud.

I am constantly learning and upgrading my skills through hands-on projects and certifications. My goal is to contribute to impactful projects in the fields of AI, Cloud Computing, and software development, while collaborating with like-minded professionals to solve real-world challenges.

With a combination of technical expertise and a dedication to growth, I aim to make a meaningful impact in the tech industry.

Education

Jspm BSIOTR

Bachelor of Engineering - BE, Information Technology · (August 2024 - June 2027)

Pimpri Chinchwad Education Trust'S. Pimpri Chinchwad Polytechnic

Diploma, Computer Engineering · (August 2021 - June 2024)

D.E.S High school Datala

Ssc