Project: Digits of Pi — Conceptual Design Specification

Student: Shravya Rekhi
Student ID: 169087129
Course: CP 220
Date: October 3rd 2025

Description:

The project aims to build a combinational logic circuit that selects a decimal position of π for a 4-bit binary input and outputs a 4-bit binary value equal to that location's decimal digit. The integer part 3 is output by input 0, the first decimal digit 1 is output by input 1, and so on. Selection up to decimal place 10 must be handled by the circuit.

Inputs:

The Pi Digit Circuit will use four binary inputs:

1. **A0** (Least Significant Bit)
   - Input: 1 if switch is ON, 0 if OFF.
2. **A1**
   - Input: 1 if switch is ON, 0 if OFF.
3. **A2**
   - Input: 1 if switch is ON, 0 if OFF.
4. **A3** (Most Significant Bit)
   - Input: 1 if switch is ON, 0 if OFF.

Input Specification:

- The circuit reads the inputs as a 4-bit binary number (A3 A2 A1 A0), representing decimal values from 0 up to 10. Unused inputs 11 through 15 could be regarded as insignificant values.

Outputs:

- The chosen π digit will be represented by a 4-bit binary output from the Pi Digit Circuit:
  O3 (Most Significant Bit)
  O2
  O1
  O0 (Least Significant Bit)

Output Specification:

- The circuit's four output lines, designated O3, O2, O1, and O0, add up to a 4-bit binary number. The 4-bit input specifies the position of the binary value, which represents the decimal digit of π .

Examples of input-output mappings are as follows:

Input = **0000** → Output = **0011** (digit 3)
Input = **0001** → Output = **0001** (digit 1)
Input = **0010** → Output = **0100** (digit 4)
Input = **0011** → Output = **0001** (digit 1)
Input = **0100** → Output = **0101** (digit 5)
Input = **0101** → Output = **1001** (digit 9)
Input = **0110** → Output = **0010** (digit 2)
Input = **0111** → Output = **0110** (digit 6)
Input = **1000** → Output = **0101** (digit 5)
Input = **1001** → Output = **0011** (digit 3)
Input = **1010** → Output = **0101** (digit 5)

It is suitable for digital logic circuits and displays since each input maps to its binary equivalent of the corresponding digit of Pi.

Handling Combinations

Valid Inputs (0–10): The circuit generates the appropriate 4-bit binary output that corresponds to the matching π digit for binary inputs 0000 through 1010.

Invalid Inputs (11–15): To signal an incorrect input condition, all outputs (O3..O0) for binary inputs 1011 through 1111 are set to 0.

Error Conditions and Response

Invalid Input Range:

- **Condition**: Inputs 1011–1111 (decimal 11–15) do not correlate to any π digit and are beyond the acceptable range (0–10).
- **Response**: All outputs (O3, O2, O1, and O0) will be set to 0 for these invalid inputs.

Non-binary Inputs:

- **Condition:** The circuit is unable to identify the intended digit position if any input line is left floating, unstable, or in a non-binary condition (not obviously 0 or 1).
- **Response:** All outputs (O3, O2, O1, and O0) will likewise be set to 0 in this scenario.

Ambiguous Possibilities:

Don't Care Conditions:

Condition: Inputs A3, A2, A1, and A0 in the decimal 11–15 range (1011–1111) do not correlate to any π digit in this design since they are outside the declared valid input set (0–10).

Response:By default these input combinations are automatically handled as don't care conditions for logic minimization.

Multiple Outputs:

Condition: For every valid address, the circuit outputs a 4-bit binary-coded decimal digit (O3..O0). Every acceptable input (0..10) must map to precisely one π decimal digit (3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5) in a unique way.

Response:

- Make sure that output encoding avoids ambiguity by requiring that the 4-bit output for each acceptable input match the binary encoding of the single matching decimal digit (for example, input 0000 → output 0011 for digit 3).
- Use a single 4-bit binary digit instead of a one-hot multi-output method so that just the encoded digit is displayed as an acceptable input.

Recommended Approach: Direct Mapping

Reasoning:

Precision Requirement: The circuit's objective is to produce particular π decimal digits. The circuit needs to transfer every input to the matching digit directly, without any approximation, because π is non-terminating and non-repeating (3.1415926535…).

Significant Digits: Every π digit has a mathematical meaning. The circuit outputs the precise binary value of the required digit rather than changing, rounding, or reduce it in order to preserve precision.

Output Representation: The circuit creates a distinct 4-bit binary output equal to the π digit at each valid 4-bit input (0–10) (for example, input 0010 → output 0100 for digit 4). By setting outputs to zero for invalid inputs (11–15), undefinable or misleading results are avoided.

Conclusion

The Digits of Pi circuit successfully implements a combinational logic design that maps a 4-bit binary input representing a decimal position to the corresponding 4-bit binary output representing the precise digit of π. The circuit treats inputs outside of this range as don't-care or error situations, with outputs set to zero to avoid ambiguity. It processes legitimate inputs between 0 and 10 by generating the precise binary-coded decimal value for each digit. Karnaugh map minimization-derived simplified logic expressions provide an efficient design that uses the fewest gates possible without sacrificing proper functionality. This method ensures that the necessary π digits are displayed with precision, clarity, and dependability. The concept offers a clear, accurate, and readily verifiable digital representation of the first eleven digits of π and may be implemented practically with common logic gates, DIP switches, and LED or 7-segment display outputs.