**CS310 PROJECT REPORT**

**Note:** To assist with the understanding of the project, it has been split into 4 incrementing parts. This means that each part will contain all the work from the previous along with certain additions. The title of each part is an indication of the addition it entails. For concision, this report will not repeat information among parts, and rather focus only on the additions.

**PROJECT:**

**Truth table**:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C | D | E | Af | Bf | Cf | Df | U |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | x | x | x | x | 1 |
| 1 | 0 | 1 | 0 | 1 | x | x | x | x | 1 |
| 1 | 0 | 1 | 1 | 0 | x | x | x | x | 1 |
| 1 | 0 | 1 | 1 | 1 | x | x | x | x | 1 |
| 1 | 1 | 0 | 0 | 0 | x | x | x | x | 1 |
| 1 | 1 | 0 | 0 | 1 | x | x | x | x | 1 |
| 1 | 1 | 0 | 1 | 0 | x | x | x | x | 1 |
| 1 | 1 | 0 | 1 | 1 | x | x | x | x | 1 |
| 1 | 1 | 1 | 0 | 0 | x | x | x | x | 1 |
| 1 | 1 | 1 | 0 | 1 | x | x | x | x | 1 |
| 1 | 1 | 1 | 1 | 0 | x | x | x | x | 1 |
| 1 | 1 | 1 | 1 | 1 | x | x | x | x | 1 |

**Note:** Decided to name the outputs Af, Bf, Cf, and Df instead of Q, R, S, T, in order to increase readability when looking at the circuit and to make the association between specific input and output more understandable. Also U is annotated as ERROR in the circuit and is a LED instead of a pin for clarity.

**Karnaugh Maps:**

Af:

E=0 E=1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CD\AB | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | X | 1 |
| 01 | 0 | 1 | X | 1 |
| 11 | 0 | 1 | X | X |
| 10 | 0 | 1 | X | X |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CD\AB | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | X | 1 |
| 01 | 0 | 1 | X | 1 |
| 11 | 0 | 1 | X | X |
| 10 | 0 | 1 | X | X |

**Note**: Since an entire row (or column), when inside a loop, has all projections simplified, these simplifications will not be shown, instead notated as 1.

Blue Loop: E, E̅, A, B, A, B̅, 1 🡪 A

Orange Loop: E, E̅, A̅, B, A, B, C̅, D, C, D🡪 BD

Red Loop: E, E̅, A̅, B, A, B, C, D, C, D̅ 🡪 BC

SOP Expression: Af = A + BD + BC

Bf:

E=0 E=1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CD\AB | 00 | 01 | 11 | 10 |
| 00 | 0 | 1 | X | 1 |
| 01 | 0 | 0 | X | 1 |
| 11 | 1 | 1 | X | X |
| 10 | 0 | 0 | X | X |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CD\AB | 00 | 01 | 11 | 10 |
| 00 | 0 | 1 | X | 0 |
| 01 | 1 | 0 | X | 1 |
| 11 | 1 | 0 | X | X |
| 10 | 1 | 0 | X | X |

Blue Loop: E̅, A, B, A, B̅, 1 🡪 AE̅

Orange Loop: E, E̅, A̅, B, A, B, C̅, D̅🡪 BC̅D̅

Red Loop: E, A̅, B̅, A, B̅, C̅, D, C, D 🡪 B̅DE

Purple Loop: E, A̅, B̅, A, B̅, C, D, C, D̅ 🡪 B̅CE

Green Loop: E̅, 1, C, D 🡪 E̅CD

SOP Expression: Bf: AE̅+BC̅D̅+ B̅DE+B̅CE+E̅CD

Cf:

E=0 E=1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CD\AB | 00 | 01 | 11 | 10 |
| 00 | 0 | 1 | X | 1 |
| 01 | 0 | 0 | X | 1 |
| 11 | 0 | 0 | X | X |
| 10 | 1 | 1 | X | X |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CD\AB | 00 | 01 | 11 | 10 |
| 00 | 1 | 1 | X | 1 |
| 01 | 0 | 0 | X | 0 |
| 11 | 1 | 1 | X | X |
| 10 | 0 | 0 | X | X |

Blue Loop: E̅, A, B, A, B̅, 1 🡪 AE̅

Orange Loop: E, E̅, A̅, B, A, B, C̅, D̅🡪 BC̅D̅

Green Loop: E̅, 1, C, D̅ 🡪 CD̅E̅

Red Loop: E, 1, C, D 🡪 CDE

Purple loop: E, 1, C̅, D̅ 🡪 C̅D̅E

SOP Expression: Cf = AE̅+BC̅D̅+CD̅E̅+CDE+C̅D̅E

Df:

E=0 E=1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CD\AB | 00 | 01 | 11 | 10 |
| 00 | 0 | 1 | X | 0 |
| 01 | 1 | 0 | X | 1 |
| 11 | 1 | 0 | X | X |
| 10 | 1 | 0 | X | X |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CD\AB | 00 | 01 | 11 | 10 |
| 00 | 1 | 1 | X | 1 |
| 01 | 0 | 0 | X | 0 |
| 11 | 0 | 0 | X | X |
| 10 | 1 | 1 | X | X |

Blue Loop: E, 1, C̅, D̅, C, D̅ 🡪 D̅E

Orange Loop: E, E̅, A̅, B, A, B, C̅, D̅ 🡪 BC̅D̅

Purple Loop: E̅, A̅, B̅, A, B̅, C̅, D, C, D 🡪 B̅DE̅

Red Loop: E̅, A̅, B̅, A, B̅, C, D, C, D̅ 🡪 B̅CE̅

SOP Expression: Df = D̅E+BC̅D̅+B̅DE̅+B̅CE̅

U:

E=0 E=1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CD\AB | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | 1 | 0 |
| 01 | 0 | 0 | 1 | 0 |
| 11 | 0 | 0 | 1 | 1 |
| 10 | 0 | 0 | 1 | 1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CD\AB | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | 1 | 0 |
| 01 | 0 | 0 | 1 | 0 |
| 11 | 0 | 0 | 1 | 1 |
| 10 | 0 | 0 | 1 | 1 |

Blue Loop: E, E̅, A, B, A, B̅, C, D, C, D̅ 🡪 AC

Orange Loop: E, E̅, A, B, 1 🡪 AB

SOP Expression: U = AC + AB

**Total transistor count: 288**

Counting:

AND/OR gates for 7 transistors

Inverters for 2 transistors

**PROJECT-NAND:**

**Note:** Fairly simple circuit to bubble push since the final OR gates all lead to AND gates, leading to the need of a miniscule amount of changes to convert to NAND only.

**Total transistor count: 319**

Counting:

Double NAND gates for 5 transistors

Triple NAND gates for 16 (2 AND gates and 1 inverter) etc.

**PROJECT-REGISTERS:**

**Purpose:**

For this part, registers were used to allow for the encoding of information up to 16 bits long. The user inputs the desired 4-bit combination and the selected format he wants to convert it to. He then presses the clock and the result is stored on the leftmost batch of outputs. He repeats the same process until he gets the desired number that will be displayed as a row of outputs.

**Limitations:**

This circuit was built as a display of knowledge and is not practical. Since the encoder operates in batches of 4 bits, it is safe to assume that the resulting 16-bit value will not be the correct result. This circuit would require a more intricate relation between the parts to manage to convert 4-bit batches of inputs into the correct outputs.

**How it was made:**

Several steps were required to create this circuit.

Step 1: The outputs of the original circuit are led into a splitter that encases them in a single wire. This is necessary since the register components of Logisim accept only multibit wires and not separate inputs.

Step 2: The outputs are fed in 4 separate registers.

Step 3: A clock and a clear button are implemented.

Step 4: Through the use of a counter, a simple logic sub-circuit is created that only “enables” the registers sequentially, one per clock pulse, while it “disables” the rest.

Step 5: The clear button resets all registers as well as the counter.

Step 6: A simple sub-circuit was built that identifies if there is erogenous input from the user and does not allow for the counter to increment while also resetting the register that is enabled without affecting the rest.

Step 7: A sub-circuit was created that stores the value of E and does not allow the user to change it after he pushes the clock and until he pushes the clear button. This is to prevent the change of the output encoding at any time, since that would lead to even more non-sensical results than we already have.

Step 8: Do not even attempt to calculate the transistor count.

**PROJECT-DISPLAY:**

**Purpose:**

For this part, the goal was to encode the binary input to a base 10 number and then display it using a 7-Segment display. For this purpose, the paper of Electronics Hub (2020) was used to extract the simplified expressions for the circuit. The completed circuit was also available in the paper but was not used since it was extremely unoptimized. Also, there was an error in the paper’s truth table so another gate was added to fix the issue (gate: BC).

**Limitations:**

The display cannot be used for the output since a truth table based on the encoded formats would need to be created, considering all inputs, and it would create an extremely complicated circuit. For the 16 bit output, knowledge way outside the scope of this class would have to be used, since all 16 outputs of the original circuit would be considered inputs for the display circuit.

**Bibliography:**

* Electronics Hub. 2020. *BCD To 7 Segment LED Display Decoder Circuit Diagram And Working*. [online] Available at: <https://www.electronicshub.org/bcd-7-segment-led-display-decoder-circuit/> [Accessed 22 November 2020].
* Harris, D. and Harris, S., 2013. *Digital Design And Computer Architecture*. Waltham, MA: Morgan Kaufmann.
* Maxfield, C., n.d. *Bebop To The Boolean Boogie, 3Rd Edition*.