**8주차 결과 보고서**

남민혁

**1. 실험 목적**

8주차 실험의 목적은 아래와 같다.

* 7-Segment Display 의 개념 이해
* Verilog를 사용하여 7-Segment Display 구현
* 입력 신호 생성 후 Simulation을 통하여 구현된 각 Gate 동작 확인
* FPGA 통해서 Verilog로 구현된 회로의 동작 확인

**2. 7-Segment Display의 결과 및 Simulation 과정에 대해서 설명하시오. (Truth table 작성 및 k-map 포함,0~F,DP)**

**-Truth Table**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | | | | **Output** | | | | | | | |
| **In A** | **In B** | **In C** | **In D** | **Out A** | **Out B** | **Out C** | **Out D** | **Out E** | **Out F** | **Out G** | **Out SEG** |
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

**- Karnaugh Map (A)**

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

**- Karnaugh Map (B)**

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

**- Karnaugh Map (C)**

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

**- Karnaugh Map (D)**

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

**- Karnaugh Map (E)**

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

**- Karnaugh Map (F)**

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

**- Karnaugh Map (G)**

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

**- Karnaugh Map (SEG)**

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

**-최소화된 Boolean 함수**

위의 Karnaugh map을 활용하여 각 output에 대하여 최소화한 식은 다음과 같다.

**-Verilog 코딩**

다음은 최소화한 식을 바탕으로 Verilog 코드를 작성한 결과이다. 각각은 논리회로 구성을 위한 Verilog 코드, 이를 바탕으로 모든 가능한 입력으로 부터 기대되는 출력이 나타나는지를 확인하기 위한 Test bench 코드, 그리고 FPGA에 직접 이진 프로그램 파일을 주입하여 물리적으로 입력과 출력을 주고 받기 위하여 입출력을 FPGA의 핀에 할당하는 Constraint에 대한 코드이다.

|  |  |
| --- | --- |
| Verilog | `timescale 1ns / 1ps  module seven\_segment\_display(  input ina,  input inb,  input inc,  input ind,  output outa,  output outb,  output outc,  output outd,  output oute,  output outf,  output outg,  output outseg  );    assign outa = (~inb \* ~ind) | (~ina \* inc) | (inb \* inc) | (ina \* ~ind) | (~ina \* inb \* ind) | (ina \* ~inb \* ~inc);  assign outb = (~ina \* ~inb) | (~inb \* ~ind) | (~ina \* ~inc \* ~ind) | (~ina \* inc \* ind) | (ina \* ~inc \* ind);  assign outc = (~ina \* ~inc) | (~ina \* ind) | (~inc \* ind) | (~ina \* inb) | (ina \* ~inb);  assign outd = (ina \* ~inc \* ~ind) | (~ina \* ~inb \* ~ind) | (~inb \* inc \* ind) | (inb \* ~inc \* ind) | (inb \* inc \* ~ind);  assign oute = (~inb \* ~ind) | (inc \* ~ind) | (ina \* inc) | (ina \* inb);  assign outf = (~inc \* ~ind) | (inb \* ~ind) | (ina \* ~inb) | (ina \* inc) | (~ina \* inb \* ~inc);  assign outg = (~inb \* inc) | (inc \* ~ind) | (ina \* ~inb) | (ina \* ind) | (~ina \* inb \* ~inc);  assign outseg = 1;  endmodule |
| Test Bench | `timescale 1ns / 1ps  module seven\_segment\_display\_tb;  reg aa;  reg bb;  reg cc;  reg dd;  wire a;  wire b;  wire c;  wire d;  wire e;  wire f;  wire g;  wire seg;  seven\_segment\_display u\_seven\_segment\_display(  .ina (aa),  .inb (bb),  .inc (cc),  .ind (dd),  .outa (a),  .outb (b),  .outc (c),  .outd (d),  .oute (e),  .outf (f),  .outg (g),  .outseg (seg)  );  initial aa = 1'b0;  initial bb = 1'b0;  initial cc = 1'b0;  initial dd = 1'b0;  always aa = #400 ~ aa;  always bb = #200 ~bb;  always cc = #100 ~cc;  always dd = #50 ~dd;  initial begin  #800  $finish;  End  endmodule |
| Constraint | set\_property IOSTANDARD LVCMOS18 [get\_ports ina]  set\_property IOSTANDARD LVCMOS18 [get\_ports inb]  set\_property IOSTANDARD LVCMOS18 [get\_ports inc]  set\_property IOSTANDARD LVCMOS18 [get\_ports ind]  set\_property IOSTANDARD LVCMOS18 [get\_ports outa]  set\_property IOSTANDARD LVCMOS18 [get\_ports outb]  set\_property IOSTANDARD LVCMOS18 [get\_ports outc]  set\_property IOSTANDARD LVCMOS18 [get\_ports outd]  set\_property IOSTANDARD LVCMOS18 [get\_ports oute]  set\_property IOSTANDARD LVCMOS18 [get\_ports outf]  set\_property IOSTANDARD LVCMOS18 [get\_ports outg]  set\_property IOSTANDARD LVCMOS18 [get\_ports outdp]  set\_property PACKAGE\_PIN J4 [get\_ports ina]  set\_property PACKAGE\_PIN L3 [get\_ports inb]  set\_property PACKAGE\_PIN K3 [get\_ports inc]  set\_property PACKAGE\_PIN M2 [get\_ports ind]  set\_property PACKAGE\_PIN D20 [get\_ports outa]  set\_property PACKAGE\_PIN C20 [get\_ports outb]  set\_property PACKAGE\_PIN C22 [get\_ports outc]  set\_property PACKAGE\_PIN B22 [get\_ports outd]  set\_property PACKAGE\_PIN B21 [get\_ports oute]  set\_property PACKAGE\_PIN A21 [get\_ports outf]  set\_property PACKAGE\_PIN E22 [get\_ports outg]  set\_property PACKAGE\_PIN E14 [get\_ports outseg] |

**-Simulation 출력 결과 비교**

**Chart

Description automatically generated**

모든 가능한 입력 값에 대한 출력값을 확인하기 위하여 Simulation을 시행한 결과 의도한 진리표와 모든 결과가 동일함을 확인하였다.

**-구현된 Schematic**

Diagram, schematic

Description automatically generated

**-이진 파일이 주입된 FPGA에서 구현된 모습**

A circuit board

Description automatically generated A circuit board

Description automatically generated

이진 파일을 FPGA에 주입하여 스위치로 0111(2)와 1001(2)를 입력으로 넣었을 때 FPGA의 seven-segment-display에서 시현되는 모습이다.

**3. 결과 검토 및 논의사항**

이번주 실험에서는 FPGA를 활용하여, 4개의 스위치로부터 0부터 15의 이진수를 받아, 0부터 F까지 16진수를 1자리 7-segment display를 구현하였다. 구현을 위하여 주어진 7-segment display의 결과물을 기반으로 진리표를 구성한 후, 이를 활용하여 Karnaugh Map를 이용하여 SOP 형태로 최소화 하였다. 이후 최소화한 식을 Verilog 코드로 작성하고 모든 가능한 입력에 대해 simulate 해보았다.

결과적으로 FPGA에 이진파일 프로그램을 주입하여 4개의 스위치로 0부터 15까지의 이진수 입력을 받으면 1자리 7-segment display에 주어진 결과와 동일한 결과를 출력하는지 확인하였다. 그 결과 모든 디자인이 의도한 진리표와 다르지 않고 동일하게 결과가 출력됨을 확인하였다.

**4. 추가 이론 조사 및 작성**

본 실험에서는 직사각형으로 이루어진 글자를 표현하기 위한 7-segment display를 사용하였으나, 실제로는 좀더 다양한 문자를 표현하기 위하여 세그먼트를 추가하거나 쪼개어 구현하기도 한다. 한 예로 16-segment display는 기존 7-segment display에서 각 사각형 가운데 사선 세그먼트 4개와 세로인 세그먼트 2개, 가로 세그먼트를 기존 1개에서 2개로 나누어 구성하였다. 그러나 이를 구현하기 위해서는 7-Segment Display에 비해 세그먼트의 수가 두배 이상 필요하여, 구현을 위해 Pin의 개수가 두배 이상으로 증가되어 실제로는 자주 사용되지 않고 있다. 그럼에도 일반 LCD에 비해 대비가 높아, 간단한 기능을 구현하면서 알파벳을 표현하고, 가독성이 높은 LED 패널을 구현해야하는 차량 오디오나 공학용 계산기 등에서 사용된다.

또한 본 실험에서 한 개의 7-segment display만 사용하여 구현해보았으나, 일반적으로는 FPGA에 있는 모든 7-segment display를 사용하는 것과 같이 여러 개의 7-segment display를 사용하여 결과를 출력하는 방식으로 확장시킬 수 있다. 이를 구현하는 방식으로는 Static Type과 Dynamic Type으로 나눠지며, 구체적으로는 다음과 같다.

먼저 Static Type의 7-Segment Display Array는 한번에 모든 7-Segment Display를 켜서 값을 나타내는 방식이다. 이를 위해서는 7-Segment Display의 각 구성요소들이 병렬로 연결되어서는 안되며 아래 그림과 같이 각 7-Segment Display에 개별적으로 입력 값이 들어와야 한다. 이 경우에는 MCU의 모든 포트들의 출력은 Latch 기능이 있어, 새로운 데이터가 출력되기 전에는 항상 출력된 값이 유지되어 안정적이다. 그러나 포트 핀의 수를 너무 많이 점유한다는 단점이 있다.

A picture containing diagram

Description automatically generated

한편 Dynamic Type의 7-Segment Display Array는 한번에 7개의 세그먼트를 키지 않고 7-Segment Display 각각을 하나의 input 통로를 통해서 순차적으로 빠르게 점등시킴으로써 LED의 잔상 효과를 이용해 모두 한번에 켜진 착시를 일으켜 동시에 켜진것 처럼 보이게 하는 방식이다. 하나의 Input으로 다수의 7-Segment Display를 점등시켜야 하므로, 각 디스플레이별로 아래 그림과 같이 병렬 구성이 필요하다.

Diagram

Description automatically generated

**5. 참고 문헌**

강석태, “Verilog HDL Summary”, <http://vlsi.hongik.ac.kr/lecture/%EC%8B%A4%ED%97%98/Verilog_Summary.pdf>.

위키피디아, “Seven-segment display”, <https://en.wikipedia.org/wiki/Seven-segment_display>.

위키피디아, “Fourteen-segment display”, <https://en.wikipedia.org/wiki/Fourteen-segment_display>.

위키피디아, “Sixteen-segment display”, <https://en.wikipedia.org/wiki/Sixteen-segment_display>.

한국기술교육대학교 전기전자통신공학부, “디지털시스템 설계 및 실습” Verilog HDL 문법 강의자료, <https://cms3.koreatech.ac.kr/sites/yjjang/down/dsys11/M01_VerilogHDL01.pdf>.