1. 7주차 결과보고서

|  |
| --- |
| * 1. - Even Parity bit Generator & Checker   2. - Odd Parity bit Generator & Checker   - 2 bit binary Comparator  - 결과 검토 |

**20141196 김성희**

1. 1. Even Parity bit Generator & Checker
   1. **1-1. Even Parity bit Generator**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| - 원본 데이터의 1의 개수가 홀수일 때, parity bit 1을 추가한다. 즉 (원본데이터 + parity bit)의 1의 개수가 짝수가 되게 끔 bit를 추가한다.  - 구현은 입력 4bit로 해보자.  **< Truth Table >**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **A** | **B** | **C** | **D** | **Pb** | | **0** | **0** | **0** | **0** | **0** | | **0** | **0** | **0** | **1** | **1** | | **0** | **0** | **1** | **0** | **1** | | **0** | **0** | **1** | **1** | **0** | | **0** | **1** | **0** | **0** | **1** | | **0** | **1** | **0** | **1** | **0** | | **0** | **1** | **1** | **0** | **0** | | **0** | **1** | **1** | **1** | **1** | | **1** | **0** | **0** | **0** | **1** | | **1** | **0** | **0** | **1** | **0** | | **1** | **0** | **1** | **0** | **0** | | **1** | **0** | **1** | **1** | **1** | | **1** | **1** | **0** | **0** | **0** | | **1** | **1** | **0** | **1** | **1** | | **1** | **1** | **1** | **0** | **1** | | **1** | **1** | **1** | **1** | **0** |   **< Karnaugh Map > for Pb**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **AB**  **CD** | **00** | **01** | **11** | **10** | | **00** | **0** | **1** | **0** | **1** | | **01** | **1** | **0** | **1** | **0** | | **11** | **0** | **1** | **0** | **1** | | **10** | **1** | **0** | **1** | **0** |   Pb = A’B’C’D+A’B’CD’+A’BC’D’+A’BCD+ABC’D+ABCD’+AB’C’D’+AB’CD  = A’B’ (C’D+CD’)+AB(C’D+CD’)+A’B(C’D’+CD)+AB’ (C’D’+CD)  = (A’B+AB’)(C’D’+CD)+(A’B’+AB)(C’D+CD’)  = (A’B+AB’)ⓧ(C’D+CD’)  = **AⓧBⓧCⓧD**  **< Code >**    - 왼쪽은 Design source, 오른쪽은 Simulation source다.  **< Simulation >**    - A,B,C,D 중에서 1인 입력 값이 홀수 개인 경우 Pb가 1이 되는 것을 볼 수 있다.  **< Schematic >**    **< FPGA >**  -Design Sources경로에 code 짜기 -> Run Synthesis & Run Implementation -> Constraints경로에 .xdc 파일 추가하기 -> Open Implemented Design 클릭 후 Constraints Wizard 클릭(Define Target, .xdc파일 타겟 설정) -> Window 탭에서 I/O ports 클릭 -> 원하는 pin 선택 및 LVCMOS18(I/O Std) 선택 -> 저장 후 .xdc reload(아래 그림처럼 코드가 자동으로 짜서 나온다.) -> Generate Bitstream 클릭(Synthesis, Implementation도 자동으로 실행) 후에 Open Hardware Manager 클릭 -> Open Target 클릭 후 Auto Connect 클릭 -> Program Device 클릭 -> FPGA에 업로드 끝  <- 입력, 출력 pin 설정  - UP switch, RIGHT switch, DOWN switch, LEFT switch 각각 A, B, C, D 입력과 맵핑  - LED1은 Pb에 맵핑  - 각 switch를 홀수 개만큼 누르면 LED1에 불이 켜진다. |

* 1. **1-2. Even Parity bit Checker**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| - (원본 데이터 + parity bit)의 1의 개수가 홀수면 1, 짝수면 0을 출력하는 회로  - 구현은 입력 5bit(3bit+1bit)로 해보자.  **< Truth Table >**   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **A** | **B** | **C** | **D** | **Pb** | **Ch** | | **0** | **0** | **0** | **0** | **0** | **0** | | **0** | **0** | **0** | **0** | **1** | **1** | | **0** | **0** | **0** | **1** | **0** | **1** | | **0** | **0** | **0** | **1** | **1** | **0** | | **0** | **0** | **1** | **0** | **0** | **1** | | **0** | **0** | **1** | **0** | **1** | **0** | | **0** | **0** | **1** | **1** | **0** | **0** | | **0** | **0** | **1** | **1** | **1** | **1** | | **0** | **1** | **0** | **0** | **0** | **1** | | **0** | **1** | **0** | **0** | **1** | **0** | | **0** | **1** | **0** | **1** | **0** | **0** | | **0** | **1** | **0** | **1** | **1** | **1** | | **0** | **1** | **1** | **0** | **0** | **0** | | **0** | **1** | **1** | **0** | **1** | **1** | | **0** | **1** | **1** | **1** | **0** | **1** | | **0** | **1** | **1** | **1** | **1** | **0** | | **1** | **0** | **0** | **0** | **0** | **1** | | **1** | **0** | **0** | **0** | **1** | **0** | | **1** | **0** | **0** | **1** | **0** | **0** | | **1** | **0** | **0** | **1** | **1** | **1** | | **1** | **0** | **1** | **0** | **0** | **0** | | **1** | **0** | **1** | **0** | **1** | **1** | | **1** | **0** | **1** | **1** | **0** | **1** | | **1** | **0** | **1** | **1** | **1** | **0** | | **1** | **1** | **0** | **0** | **0** | **0** | | **1** | **1** | **0** | **0** | **1** | **1** | | **1** | **1** | **0** | **1** | **0** | **0** | | **1** | **1** | **0** | **1** | **1** | **0** | | **1** | **1** | **1** | **0** | **0** | **1** | | **1** | **1** | **1** | **0** | **1** | **0** | | **1** | **1** | **1** | **1** | **0** | **0** | | **1** | **1** | **1** | **1** | **1** | **1** |   **< Karnaugh Map > for Ch**  A = 0   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **BC**  **D Pb** | **00** | **01** | **11** | **10** | | **00** | **0** | **1** | **0** | **1** | | **01** | **1** | **0** | **1** | **0** | | **11** | **0** | **1** | **0** | **1** | | **10** | **1** | **0** | **1** | **0** |   A = 1   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **BC**  **D Pb** | **00** | **01** | **11** | **10** | | **00** | **1** | **0** | **1** | **0** | | **01** | **0** | **1** | **0** | **1** | | **11** | **1** | **0** | **1** | **0** | | **10** | **0** | **1** | **0** | **1** |   Ch = A’(B’C’D’Pb+B’C’DPb’+B’C’D’Pb’+B’CDPb+BCD’Pb+BCDPb’+BC’D’Pb’+BC’DPb)  + A(B’C’D’Pb’+B’C’DPb+B’C’D’Pb+B’CDPb’+BCD’Pb’+BCDPb+BC’D’Pb+BC’DPb’)  = A’(B’C’(D’Pb+DPb’)+BC(D’Pb+DPb’)+B’C(D’Pb’+DPb)+BC’(D’Pb’+DPb))  + A(B’C’(D’Pb’+DPb)+BC(D’Pb’+DPb)+B’C(D’Pb+DPb’)+BC’(D’Pb+DPb’))  = A’((B’C+BC’)(D’Pb’+DPb)+(B’C’+BC)(D’Pb+DPb’))  + A((B’C+BC’)(D’Pb+DPb’)+(B’C’+BC)(D’Pb’+DPb))  = A’((B’C+BC’)ⓧ(D’Pb+DPb’))  + A((B’C+BC’)ⓧ(D’Pb+DPb’))’  = A’(BⓧCⓧDⓧPb) + A(BⓧCⓧDⓧPb)’  = **AⓧBⓧCⓧDⓧPb**  **< Code >**    - 왼쪽은 Design source, 오른쪽은 Simulation source다.  **< Simulation >**    - A,B,C,D,Pb 중에서 1인 입력 값이 홀수 개인 경우Ch가 1이 되는 것을 볼 수 있다.  **< Schematic >**    **< FPGA >**  -Design Sources경로에 code 짜기 -> Run Synthesis & Run Implementation -> Constraints경로에 .xdc 파일 추가하기 -> Open Implemented Design 클릭 후 Constraints Wizard 클릭(Define Target, .xdc파일 타겟 설정) -> Window 탭에서 I/O ports 클릭 -> 원하는 pin 선택 및 LVCMOS18(I/O Std) 선택 -> 저장 후 .xdc reload(아래 그림처럼 코드가 자동으로 짜서 나온다.) -> Generate Bitstream 클릭(Synthesis, Implementation도 자동으로 실행) 후에 Open Hardware Manager 클릭 -> Open Target 클릭 후 Auto Connect 클릭 -> Program Device 클릭 -> FPGA에 업로드 끝  <- 입력, 출력 pin 설정  - UP switch, RIGHT switch, DOWN switch, LEFT switch, MID switch 각각 A, B, C, D, Pb 입력과 맵핑  - LED1은 Ch에 맵핑  - 각 switch를 홀수 개만큼 누르면 LED1에 불이 켜진다. |

1. 2. Odd Parity bit Generator & Checker
   1. **2-1. Odd Parity bit Generator**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| - 원본 데이터의 1의 개수가 짝수일 때, parity bit 1을 추가한다. 즉 (원본데이터 + parity bit)의 1의 개수가 홀수가 되게 끔 bit를 추가한다.  - 구현은 입력 4bit로 해보자.  **< Truth Table >**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **A** | **B** | **C** | **D** | **Pb** | | **0** | **0** | **0** | **0** | **1** | | **0** | **0** | **0** | **1** | **0** | | **0** | **0** | **1** | **0** | **0** | | **0** | **0** | **1** | **1** | **1** | | **0** | **1** | **0** | **0** | **0** | | **0** | **1** | **0** | **1** | **1** | | **0** | **1** | **1** | **0** | **1** | | **0** | **1** | **1** | **1** | **0** | | **1** | **0** | **0** | **0** | **0** | | **1** | **0** | **0** | **1** | **1** | | **1** | **0** | **1** | **0** | **1** | | **1** | **0** | **1** | **1** | **0** | | **1** | **1** | **0** | **0** | **1** | | **1** | **1** | **0** | **1** | **0** | | **1** | **1** | **1** | **0** | **0** | | **1** | **1** | **1** | **1** | **1** |   **< Karnaugh Map > for Pb**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **AB**  **CD** | **00** | **01** | **11** | **10** | | **00** | **1** | **0** | **1** | **0** | | **01** | **0** | **1** | **0** | **1** | | **11** | **1** | **0** | **1** | **0** | | **10** | **0** | **1** | **0** | **1** |   Pb = A’B’C’D’+A’B’CD+A’BC’D+A’BCD’+ABC’D’+ABCD+AB’C’D+AB’CD’  = (A’B’+AB)(C’D’+CD)+ (A’B+AB’)(C’D+CD’)  = (**AⓧBⓧCⓧD)**’  **< Code >**    - 왼쪽은 Design source, 오른쪽은 Simulation source다.  **< Simulation >**    - A,B,C,D 중에서 1인 입력 값이 짝수(또는0)개인 경우 Pb가 1이 되는 것을 볼 수 있다.  **< Schematic >**    **< FPGA >**  -Design Sources경로에 code 짜기 -> Run Synthesis & Run Implementation -> Constraints경로에 .xdc 파일 추가하기 -> Open Implemented Design 클릭 후 Constraints Wizard 클릭(Define Target, .xdc파일 타겟 설정) -> Window 탭에서 I/O ports 클릭 -> 원하는 pin 선택 및 LVCMOS18(I/O Std) 선택 -> 저장 후 .xdc reload(아래 그림처럼 코드가 자동으로 짜서 나온다.) -> Generate Bitstream 클릭(Synthesis, Implementation도 자동으로 실행) 후에 Open Hardware Manager 클릭 -> Open Target 클릭 후 Auto Connect 클릭 -> Program Device 클릭 -> FPGA에 업로드 끝  <- 입력, 출력 pin 설정  - UP switch, RIGHT switch, DOWN switch, LEFT switch각각 A, B, C, D 입력과 맵핑  - LED1은 Pb에 맵핑  - 각 switch를 짝수 개만큼 누르면 LED1에 불이 켜진다. |

* 1. **2-2. Odd Parity bit Checker**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| - (원본 데이터 + parity bit)의 1의 개수 짝수 개면 1을, 홀수 개면 0을 출력하는 회로.  - 구현은 입력 4bit(3bit+1bit)로 해보자.  **< Truth Table >**   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **A** | **B** | **C** | **D** | **Pb** | **Ch** | | **0** | **0** | **0** | **0** | **0** | **1** | | **0** | **0** | **0** | **0** | **1** | **0** | | **0** | **0** | **0** | **1** | **0** | **0** | | **0** | **0** | **0** | **1** | **1** | **1** | | **0** | **0** | **1** | **0** | **0** | **0** | | **0** | **0** | **1** | **0** | **1** | **1** | | **0** | **0** | **1** | **1** | **0** | **1** | | **0** | **0** | **1** | **1** | **1** | **0** | | **0** | **1** | **0** | **0** | **0** | **0** | | **0** | **1** | **0** | **0** | **1** | **1** | | **0** | **1** | **0** | **1** | **0** | **1** | | **0** | **1** | **0** | **1** | **1** | **0** | | **0** | **1** | **1** | **0** | **0** | **1** | | **0** | **1** | **1** | **0** | **1** | **0** | | **0** | **1** | **1** | **1** | **0** | **0** | | **0** | **1** | **1** | **1** | **1** | **1** | | **1** | **0** | **0** | **0** | **0** | **0** | | **1** | **0** | **0** | **0** | **1** | **1** | | **1** | **0** | **0** | **1** | **0** | **1** | | **1** | **0** | **0** | **1** | **1** | **0** | | **1** | **0** | **1** | **0** | **0** | **1** | | **1** | **0** | **1** | **0** | **1** | **0** | | **1** | **0** | **1** | **1** | **0** | **0** | | **1** | **0** | **1** | **1** | **1** | **1** | | **1** | **1** | **0** | **0** | **0** | **1** | | **1** | **1** | **0** | **0** | **1** | **0** | | **1** | **1** | **0** | **1** | **0** | **0** | | **1** | **1** | **0** | **1** | **1** | **1** | | **1** | **1** | **1** | **0** | **0** | **0** | | **1** | **1** | **1** | **0** | **1** | **1** | | **1** | **1** | **1** | **1** | **0** | **1** | | **1** | **1** | **1** | **1** | **1** | **0** |   **< Karnaugh Map > for Ch**  A = 0   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **BC**  **D Pb** | **00** | **01** | **11** | **10** | | **00** | **1** | **0** | **1** | **0** | | **01** | **0** | **1** | **0** | **1** | | **11** | **1** | **0** | **1** | **0** | | **10** | **0** | **1** | **0** | **1** |   A = 1   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **BC**  **D Pb** | **00** | **01** | **11** | **10** | | **00** | **0** | **1** | **0** | **1** | | **01** | **1** | **0** | **1** | **0** | | **11** | **0** | **1** | **0** | **1** | | **10** | **1** | **0** | **1** | **0** |   Ch = A’(B’C’D’Pb’+B’C’DPb+B’C’D’Pb+B’CDPb’+BCD’Pb’+BCDPb+BC’D’Pb+BC’DPb’)  + A(B’C’D’Pb+B’C’DPb’+B’C’D’Pb’+B’CDPb+BCD’Pb+BCDPb’+BC’D’Pb’+BC’DPb)  = A’(B’C’(D’Pb’+DPb)+BC(D’Pb’+DPb)+B’C(D’Pb+DPb’)+BC’(D’Pb+DPb’))  + A(B’C’(D’Pb+DPb’)+BC(D’Pb+DPb’)+B’C(D’Pb’+DPb)+BC’(D’Pb’+DPb))  = A’((B’C’+BC)(D’Pb’+DPb)+(B’C+BC’)(D’Pb+DPb’))  + A((B’C’+BC)(D’Pb+DPb’)+(B’C+BC’)(D’Pb’+DPb))  = A’((B’C+BC’)ⓧ(D’Pb+DPb’))’  + A((B’C+BC’)ⓧ(D’Pb+DPb’))  = A’(BⓧCⓧDⓧPb)’ + A(BⓧCⓧDⓧPb)  = (**AⓧBⓧCⓧDⓧPb)’**  **< Code >**    - 왼쪽은 Design source, 오른쪽은 Simulation source다.  **< Simulation >**    - A,B,C,D,Pb 중에서 1인 입력 값이 짝수(또는 0)개인 경우 Ch가 1이 되는 것을 볼 수 있다.  **< Schematic >**    **< FPGA >**  -Design Sources경로에 code 짜기 -> Run Synthesis & Run Implementation -> Constraints경로에 .xdc 파일 추가하기 -> Open Implemented Design 클릭 후 Constraints Wizard 클릭(Define Target, .xdc파일 타겟 설정) -> Window 탭에서 I/O ports 클릭 -> 원하는 pin 선택 및 LVCMOS18(I/O Std) 선택 -> 저장 후 .xdc reload(아래 그림처럼 코드가 자동으로 짜서 나온다.) -> Generate Bitstream 클릭(Synthesis, Implementation도 자동으로 실행) 후에 Open Hardware Manager 클릭 -> Open Target 클릭 후 Auto Connect 클릭 -> Program Device 클릭 -> FPGA에 업로드 끝  <- 입력, 출력 pin 설정  - UP switch, RIGHT switch, DOWN switch, MID switch각각 A, B, C, D, Pb 입력과 맵핑  - LED1은 Ch에 맵핑  - 각 switch를 짝수 개만큼 누르면 LED1에 불이 켜진다. |

1. 3. 2-bit binary Comparator

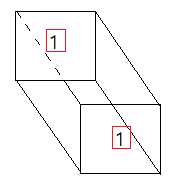
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| - 2bit 입력 2개의 크기를 비교하는 회로  **- (**A1 A2), (B1 B2) 2bit 입력 2개와 F1, F2, F3 1bit 출력 3개  **< Truth Table >**   |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | **A1** | **A2** | **B1** | **B2** | **F1(A>B)** | **F2(A=B)** | **F3(A<B)** | | **0** | **0** | **0** | **0** | **0** | **1** | **0** | | **0** | **0** | **0** | **1** | **0** | **0** | **1** | | **0** | **0** | **1** | **0** | **0** | **0** | **1** | | **0** | **0** | **1** | **1** | **0** | **0** | **1** | | **0** | **1** | **0** | **0** | **1** | **0** | **0** | | **0** | **1** | **0** | **1** | **0** | **1** | **0** | | **0** | **1** | **1** | **0** | **0** | **0** | **1** | | **0** | **1** | **1** | **1** | **0** | **0** | **1** | | **1** | **0** | **0** | **0** | **1** | **0** | **0** | | **1** | **0** | **0** | **1** | **1** | **0** | **0** | | **1** | **0** | **1** | **0** | **0** | **1** | **0** | | **1** | **0** | **1** | **1** | **0** | **0** | **1** | | **1** | **1** | **0** | **0** | **1** | **0** | **0** | | **1** | **1** | **0** | **1** | **1** | **0** | **0** | | **1** | **1** | **1** | **0** | **1** | **0** | **0** | | **1** | **1** | **1** | **1** | **0** | **1** | **0** |   **< Karnaugh Map >**  F1(A>B)   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **A1 A2**  **B1 B2** | **00** | **01** | **11** | **10** | | **00** | **0** | **1** | **1** | **1** | | **01** | **0** | **0** | **1** | **1** | | **11** | **0** | **0** | **0** | **0** | | **10** | **0** | **0** | **1** | **0** |   F1 = A2B1’B2’+A1B1’+A1A2B2’  = A2B2’(A1+ B1’)+A1B1’  F2(A=B)   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **A1 A2**  **B1 B2** | **00** | **01** | **11** | **10** | | **00** | **1** | **0** | **0** | **0** | | **01** | **0** | **1** | **0** | **0** | | **11** | **0** | **0** | **1** | **0** | | **10** | **0** | **0** | **0** | **1** |   F2 = A1’A2’B1’B2’+A1’A2B1’B2+A1A2B1B2+A1A2’B1B2’  = A1’B1’(A2’B2’+A2B2)+A1B1(A2’B2’+A2B2)  = (A1’B1’+A1B1)(A2’B2’+A2B2)  = (A1ⓧB1)’(A2ⓧB2)’  F3(A<B)   |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **A1 A2**  **B1 B2** | **00** | **01** | **11** | **10** |  |  |  |  |  | | **00** | **0** | **0** | **0** | **0** |  |  |  |  |  | | **01** | **1** | **0** | **0** | **0** |  |  |  |  |  | | **11** | **1** | **1** | **0** | **1** |  |  |  |  |  | | **10** | **1** | **1** | **0** | **0** |  |  |  |  |  |   F3 = A1’A2’B2+A2’B1B2+A1’B1  = A2’B2(A1’+B1)+A1’B1  **< Code >**    - 왼쪽은 Design source, 오른쪽은 Simulation source다.  **< Simulation >**    - A = (A1 A2), B = (B1 B2)일 때 F1은 A>B, F2는 A=B, F3은 A<B일 때, 1의 값을 가진다.  **< Schematic >**    **< FPGA >**  -Design Sources경로에 code 짜기 -> Run Synthesis & Run Implementation -> Constraints경로에 .xdc 파일 추가하기 -> Open Implemented Design 클릭 후 Constraints Wizard 클릭(Define Target, .xdc파일 타겟 설정) -> Window 탭에서 I/O ports 클릭 -> 원하는 pin 선택 및 LVCMOS18(I/O Std) 선택 -> 저장 후 .xdc reload(아래 그림처럼 코드가 자동으로 짜서 나온다.) -> Generate Bitstream 클릭(Synthesis, Implementation도 자동으로 실행) 후에 Open Hardware Manager 클릭 -> Open Target 클릭 후 Auto Connect 클릭 -> Program Device 클릭 -> FPGA에 업로드 끝  <- 입력, 출력 pin 설정  - UP switch, RIGHT switch, LEFT switch, DOWN switch 각각 A1, A2, B1, B2 입력과 맵핑  - LED1, LED2, LED3 각각 F1, F2, F3에 맵핑  - 위simualtion에 맞춰서 switch를 누르면 LED1, LED2, LED3을 끄고 켤 수 있다. |

1. 4. 결과 검토 및 논의 사항

- 2-bit binary comparator에 2input gate가 아닌 3input gate를 사용 가능했다면 훨씬 보기 좋고 효율적인 shcematic을 만들 수 있었을 것이다.

1. 5. 기타 이론

- 입력 변수가 5개인 Karnaugh Map은 1-2와 2-2 처럼 표를 두 개 그리고 3차원적으로 상상해서 표시하면 된다. 예를 들어 A=0 B’C’D’E’와 A=1 B’C’D’E’가 둘 다 1인경우 아래 그림처럼 각각 위 아래로 z축을 추가하여 묶으면 된다.



1. 6. FPGA pin No.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **LED1** | **LED2** | **LED3** | **LED4** | **LED5** |
| F15 | F13 | F14 | F16 | F17 |
| **UP switch** | **RIGHT switch** | **DOWN switch** | **LEFT switch** | **MID switch** |
| E21 | G22 | F21 | D21 | G21 |