6주차 결과보고서

전공: 컴퓨터공학과 학년: 2학년 학번: 20211522 이름: 김정환

**1.**

.................

Adder와 Subtractor, Code converter의 개념을 이해하고, Full Adder, Half Adder, Full Subtractor, Half Subtractor, BCD 8421 – 2421 converter를 Verilog를 사용하여 구현한다. Verilog로 구현한 회로를 FPGA로 동작을 확인한다.

................

**2.**

.......................

우선 Half Adder의 구현은 아래와 같은 코드로 구현하였다.

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

이는 강의자료에 주어진 구조도를 바탕으로 하였고 이를 이용하여 합인 s=a^b, 즉 XOR Gate를 지나도록 하였고, 자리 올림 c=a&b로 AND Gate를 지나도록 하였다. 이를 simulation한 결과는 아래와 같다.

스크린샷, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

이를 바탕으로 진리표를 작성하면 아래와 같다.

|  |  |  |  |
| --- | --- | --- | --- |
| In A | In B | Out S | Out C |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

다음으로 Full Adder의 구현은 아래와 같은 코드로 구현하였다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

이 역시 강의 자료에 주어진 자료도를 바탕으로 하였고, 해당 구조도와 차이점으로는 input인 C는 c로 나타내고, output인 C는 cout으로 나타냈다. 합인 s=(a^b)^c로 XOR Gate를 2번 거치는 형태를 보이고, 자리 올림의 출력인 cout=(a&b) | ((a^b)&c)로 이는 Half Adder일 때 자리 올림과 입력으로 들어온 자리 올림과 a, b의 합의 자리 올림을 모두 고려하는 구조이다. 이를 simulation한 결과는 아래와 같다.

스크린샷, 멀티미디어 소프트웨어, 그래픽 소프트웨어이(가) 표시된 사진

자동 생성된 설명

이를 바탕으로 진리표를 작성하면 아래와 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| In C | In B | In A | Out S | Out C |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

.........................

**3.**

.......................

우선 Half Subtractor의 구현은 아래와 같은 코드로 구현하였다.

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

강의자료에는 D와 b의 출력을 가지지만 코드 상에서는 각각 s와 c로 사용하였고, 차인 s=a^b, c=(~a)&b로 구현하였다. 이를 simulation한 결과는 아래와 같다.

스크린샷, 멀티미디어 소프트웨어이(가) 표시된 사진

자동 생성된 설명

이를 바탕으로 진리표를 작성하면 아래와 같다.

|  |  |  |  |
| --- | --- | --- | --- |
| In A | In B | Out D | Out b |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |

다음으로 Full Subtractor의 구현은 아래와 같은 코드로 구현하였다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

강의 자료에 주어진 것 중 자리 내림의 입력은 c로 차인 D는 s로, 자리 내림의 출력은 cout으로 나타냈다. 각각은 Full Adder와 유사하나 cout에서 a에 NOT만 추가된 점에 차이가 있다. 이를 simulation한 결과는 아래와 같다.

스크린샷, 멀티미디어 소프트웨어, 그래픽 소프트웨어이(가) 표시된 사진

자동 생성된 설명

이를 바탕으로 진리표를 나타내면 아래와 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| In A | In B | In | Out D | Out |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 |

.........................

**4.**

.......................

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| In p | In q | In r | In s | Out a | Out b | Out c | Out d |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 0 | x | x | x | x |
| 1 | 0 | 1 | 1 | x | x | x | x |
| 1 | 1 | 0 | 0 | x | x | x | x |
| 1 | 1 | 0 | 1 | x | x | x | x |
| 1 | 1 | 1 | 0 | x | x | x | x |
| 1 | 1 | 1 | 1 | x | x | x | x |

강의 자료에 주어진 대로 BCD 8421-2421 Code converter의 진리표를 나태내면 위와 같이 나타낼 수 있다. 그 중에서 2421에서 사용하지 않는 값들에 대한 output은 x로 표기하였다. 이를 바탕으로 output a, b, c, d 각각에 대해서 카르노 맵을 작성하면, 아래와 같다.

우선 a에 대한 카르노 맵이다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| rs\pq | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | X | 1 |
| 01 | 0 | 1 | X | 1 |
| 11 | 0 | 1 | X | X |
| 10 | 0 | 1 | X | X |

이 경우 a에 대해서 간소화를 진행하면 a=p+qr+qs=(p+q)(p+r+s)로 SOP, POS form으로 나타낼 수 있다.

다음으로 b에 대한 카르노 맵을 작성하면,

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| rs\pq | 00 | 01 | 11 | 10 |
| 00 | 0 | 1 | X | 1 |
| 01 | 0 | 0 | X | 1 |
| 11 | 0 | 1 | X | X |
| 10 | 0 | 1 | X | X |

이 경우 b에 대해서 간소화를 진행하면 b=p+qr+qr’s’=(p+q)(q+r+s’)으로 SOP, POS form으로 나타낼 수 있다.

다음으로 c에 대한 카르노 맵을 작성하면,

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| rs\pq | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | X | 1 |
| 01 | 0 | 1 | X | 1 |
| 11 | 1 | 0 | X | X |
| 10 | 1 | 0 | X | X |

이 경우 c에 대해서 간소화를 진행하면 c=p+q’r+qr’s=(p+q+r)(p+r+s)(q’+r’)으로 SOP, POS form으로 나타낼 수 있다.

다음으로 d에 대한 카르노 맵을 작성하면,

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| rs\pq | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | X | 0 |
| 01 | 1 | 1 | X | 1 |
| 11 | 1 | 1 | X | X |
| 10 | 0 | 0 | X | X |

이 경우 d에 대해서 간소화를 진행하면 d=s로 나타낼 수 있다.

해당하는 4개의 식은 모두 AND, OR, NOT으로 구성되어 있으므로 NAND와 NOR로도 변환할 수 있는데, 우선 NAND로 나타내면 Verilog 식으로는 아래와 같다.

a=~(~p&~(q&r) & ~(q&s))

b=~(~p&~(q&~r&~s)&~(q&r))

c=~(~p&~(~q&r)&~(q&~r&s))

d=s

다음으로 NOR로 나타내면,

a=~(~(p|q)|~(p|r|s))

b=~(~(p|q)|~(q|r|s))

c=~(~(p|q|r)|~(p|r|s)|~(~q|~r))

d=s

로 나타낼 수 있다.

실습에서는 NAND로 나타낸 식을 이용하여 구현하였다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

이를 simulation한 결과는 아래와 같다.

스크린샷, 사각형이(가) 표시된 사진

자동 생성된 설명

진리표와 동일한 결과를 얻을 수 있는 것을 확인하여 식이 정확히 작성되었음을 확인할 수 있다.

.........................

**5.**

.......................

본 실험에서는 Full Adder, Full Subtractor, Half Adder, Half Subtractor 그리고 8421-2421 Code converter를 구현하였다. Half Adder와 Half Subtractor는 단순히 1bit의 두 수가 주어졌을 때의 덧셈, 뺄셈을 구현하였으나, Full Adder와 Full Subtractor에서 입력으로 각각 자리 올림과 빌림 수를 통하여 이전 자리의 수의 결과도 영향을 미치도록 하는 연산을 구현했다. 이는 1bit의 계산만이 아닌 n개의 bit 연산을 구현할 때 이전 자리 올림과 빌림이 제대로 일어나도록 할 것이다. 다음으로 8421-2421 Code converter는 주어진 진리표에 맞춰 진리표를 먼저 작성하고 회로를 구성하는 방식으로 구현하였다. 이 과정에서 K-map, 즉 카르노 맵을 사용하여 논리식의 형태를 SOP, POS form으로 단순화하는 과정을 거쳤고, 이 과정에서 이전 보고서 내용의 단순화에 더해서 무관항을 함께 엮는 방식으로 단순화할 수 있는 방식도 이용하여 정리하였다. 다음으로 AND, OR, NOT만으로 이루어진 식을 NAND와 NOR로 바꾸는 과정도 거쳤는데, 이는 Verilog 상에서의 식은 더 복잡해지지만 NAND와 NOR로도 모두 표현 가능함을 확인할 수 있었다. 또한 실제 회로에서는 더 적은 transistor를 사용함으로써 더 효율을 높일 가능성이 존재한다.

.........................

**6.**

.......................

게이트들로 만드는 회로 중에는 Flip-Flop 회로도 존재한다. 이는 1 bit의 정보를 보관 및 유지할 수 있는 회로이다. 이는 기본적으로 loop를 도는 구조 즉 순환되는 회로의 형태를 확장시킨 형태이다. 입력으로 들어오는 값은 S, R로 각각 set과 reset을 의미하고, CP로 컴퓨터 내부에서 규칙적으로 발생시키는 전기신호도 받는다. 여기서 각각 R과 S는 CP와 NAND 연산을 거친다. 각각의 출력은 서로 반대편의 또 하나의 NAND의 출력값과 NAND 연산을 하는 형태를 가진다. 이러한 방식으로 출력이 다시 입력으로 주어지는 구조를 가지기에 이전 값을 저장하는 기능이 구현된다. 이를 회로로 나타내면,

도표, 라인, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

위와 같이 나타낼 수 있는데, 해당 회로도에서는 최초 연산은 AND, 다음은 NOR를 사용한 형태의 Flip-Flop 회로이다.

.........................