컴퓨터 구성

(LogicWorks를 사용한 자판기 구현)

|  |  |  |
| --- | --- | --- |
| 2조 | | |
| 과목명 | | 컴퓨터 구성 |
| 담당 교수 | | 김양우 교수님 |
| 정보통신공학과 | 2017112173 | 박동근 |
| 정보통신공학과 | 2019112193 | 박상준 |
| 정보통신공학과 | 2019112102 | 이성현 |
| 정보통신공학과 | 2019112110 | 이정민 |
| 제출일 | | 2020 12월 16일 |

**< Contents >**

**Ⅰ. 프로젝트 목표 및 방향 ...........................................................**

1.1 추진 목표

1.2 프로젝트의 목적

1.3 프로젝트의 필요성

1.4 관련 연구

**Ⅱ. 자판기 구현............................................................................**

2.0 설계 개요 및 음료 가격 설명

2.1 심볼 소개

2.1.1 MUX

2.1.2 beverages 1

2.1.3 beverages 2

2.1.4 BCD Adder

2.1.5 BCD Subtractor

2.1.6 4 X D FF

2.1.7 BCD 7segments

2.1.8 PossibleCal

2.1.9 PossibleBlock

2.1.10 Return T

2.1.11 Return M

2.2 전체 회로도 및 파트 분배

2.2.1 데이터 처리 파트

2.2.2 배출 파트

2.2.3 반환 파트

2.2.4 출력 및 버튼 파트

**Ⅲ. 프로젝트 수행 보고.................................................................**

3.1 프로젝트 팀원 및 참여도

3.2 프로젝트 추진 단계

3.3 프로젝트 전체 추진 일정

3.4 프로젝트 팀 전략 및 성과

3.5 팀 회의 일지

**Ⅳ. 참고문헌..............................................................................**

**1. 프로젝트 목표 및 방향**

**1.1 추진 목표**

**-Logicworks를 통하여 자판기와 그 기능들을 구현한다.**

**-자판기의 기능으로는 현금 주입, 잔액 확인, 음료 선택, 음료 배출, 잔액회수가 있다.**

**-입/출력 방식은 Logicworks 내 기능을 사용한다.**

**-현금은 10원, 50원, 100원, 500원, 1,000원, 5,000원, 10,000원, 50,000원 단위로 주입한다.**

**-잔액 확인은 약 999,999원 까지 가능하다.**

**-음료의 개수는 8개(460원, 870원, 1,400원, 3,300원, 5,490원, 8,340원, 9,990원, 10,000원)로 구성한다**

**-음료 선택은 Logicworks 내 Binary Switch등을 이용한다.**

**-LogicWorks 내의 Binary Probe 혹은 LED 등을 이용하여 음료 배출 기능을 구현한다.**

**-잔액 부족 시 Binary Probe 혹은 LED 등을 통하여 확인한다.**

**-음료 배출 시 잔액에 음료 가격을 반영한다.**

**- 잔액 회수 시 50,000원, 10,000원, 5,000원, 1,000원, 500원, 100원, 50원, 10원 순으로 출력한다.**

**- 자판기 On/Off 버튼을 구현한다.**

**1.2 프로젝트의 목적**

**‘컴퓨터 구성’ 강의의 이론 강의와 실습 활동에서 배운 내용들을 참고하여 Logicworks를 통해 자판기를 구현한다. 프로젝트를 통하여, 자판기의 원리와 구성을 이해하고 설계 능력과, 문제해결능력 등의 개발 능력을 함양하는 데에 그 목적이 있다.**

**1.3 프로젝트의 필요성**

**자판기는 가산기와 감산기, BCD(이진화 십진수)를 활용하여 설계할 수 있는 기능적인 부분들을 가지고 있다.**

**본 프로젝트를 진행함에 있어서 이미 ‘컴퓨터 구성’ 강의를 수강한 학생들은 수업시간에 가산기와 감산기, BCD, 그리고 7-segment에 대하여 학습하였다.**

**이러한 지식들을 바탕으로, 자판기를 구현할 수 있다.**

**1.4 관련 연구**

**자판기의 설계에는 BCD Adder(가산기), BCD subtructor(감산기), 7-segment가 사용된다.**

**가산기와 감산기는 금액의 입력, 음료 선택에 따른 잔액 차감, 잔액 반환에 사용이 된다.**

**이러한 계산 결과를 사용자에게 시각적으로 보여주는 기능이 필요한데, 이것이 바로**

**7-segment이다. 7-segment는 a, b, c, d, e, f, g의 7개 입력을 통하여 숫자를 시각적으로 보여준다.**

**본 프로젝트에서는 계산된 4bit의 BCD값을 7-segment에 입력될 수 있도록 BCD-7segment 심볼을 구현하여 사용하였다.**

**이때 7-segment가 입력 받는 BCD의 값은 10, 100, 1,000, 10,000, 100,000원의 단위로 잔액을 받았다. 입력에서 어떤 자리의 값이 올림이 생긴다면 올림은 바로 다음의 상위 단위로 넘어간다. 마찬가지로 음료 선택 시 단위의 값이 내림이 발생한다면 내림은 바로 다음의 상위 단위로 전달된다.**

**2. 디지털 자판기 설계**

1. **설계 개요 및 기본 용어 설명**

**(0)설계 개요**

**LogicWorks를 사용한 디지털 자판기는 크게 4부분으로 나눌 수 있다.**

**먼저 돈을 넣는 입력부, 음료를 선택하는 음료 선택부, 음료를 배출하는 음료 배출부, 잔액을 반환하는 잔액반환부의 단계를 거친다.**

**(1)음료 가격**

**0. 포카리스웨트 460원**

**1. 오렌지 주스 870원**

**2. 포도 주스 1400원**

**3. 레쓰비 3300원**

**4. 콜라 5490원**

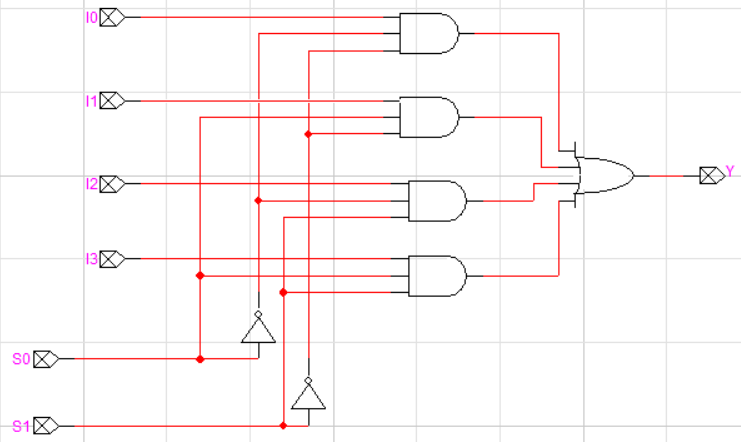
**5. 사이다 8340원**

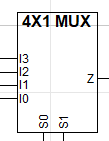
**6. 초코에몽 9990원**

**7. 생수 10000원**

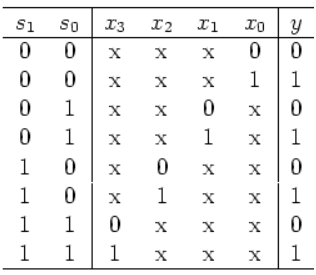
**2-1.심볼 소개**

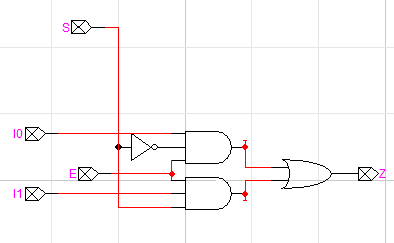
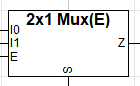
1. **MUX**





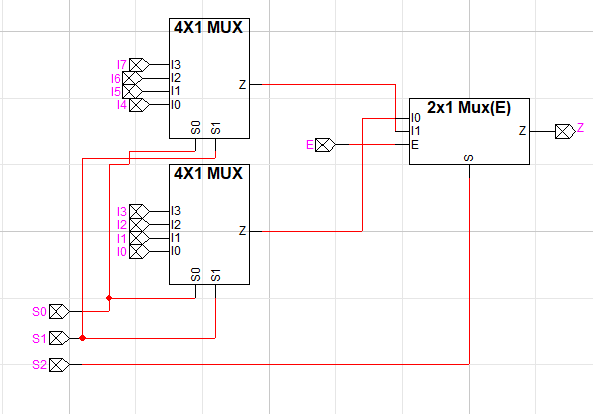
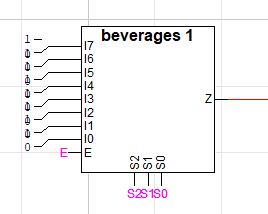
먼저 가장 기본이 되는 출력장치인 MUX는 Multiplexer의 약어이다. MUX는 N개의 선택 입력에 따라 출력을 달리하는 것이 특징인데, 우리 조가 사용한 4X1 MUX는 4개의 입력을 MUX에 넣어두고 S0 S1의 입력에 따라 I0..3 중 하나가 출력된다. 다음은 4X1 MUX의 진리표이다.





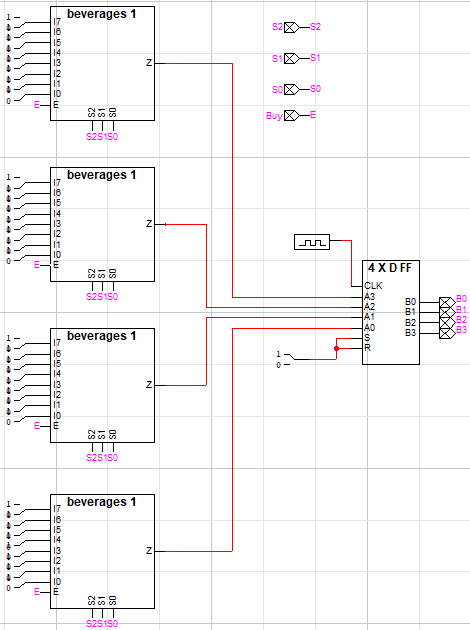
2X1 MUX는 기본적으로 2개의 입력을 넣어두고 S의 입력, E(Buy 버튼을 누를 시 1, 그 외에는 0)의 입력에 따라 Z의 값이 출력된다.

**2. beverages 1**



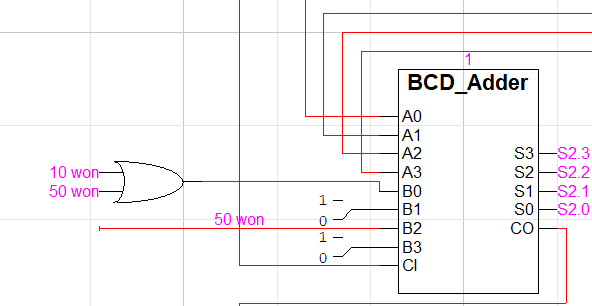
beverages 1심볼은 음료가격의 한자릿수를 출력해주는 기능을 한다. S0 S1 S2와 E의 입력에 따라 I0..7이 출력된다. 즉 음료 선택부분에 있어서 8X1 MUX기능을 하는 심볼이라고 볼 수 있다.

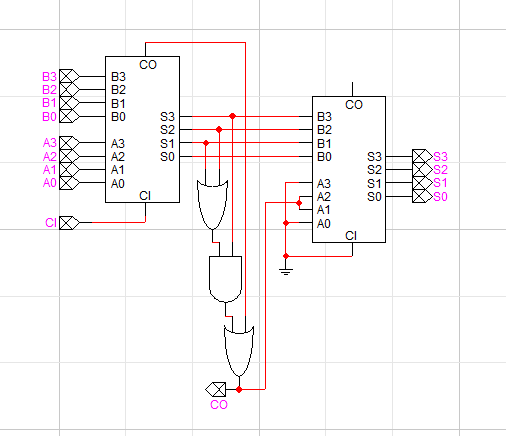
**3. beverages 2**

****

beverages 2 심볼은 Buy 버튼(E)과 S0 S1 S2의 입력을 받아 음료가격의 한 자릿수를 4비트로 출력해준다. 전체 회로도에서 이 심볼은 1의자리~100,000자리까지 총 6개가 사용된다. 6개가 모여 하나의 음료 가격을 나타내주고 완성된 가격을 통해 음료선택기능을 할 수 있다.

**4. BCD Adder**





BCD는 binary coded decimal로, 이진수부호화 된 십진수를 뜻한다.

이때, 주의해야 할 점은 이진수부호화 시켜줄 때 숫자를 통째로 이진수로 만드는 것이 아니라

각 자릿수를 구분해준다.

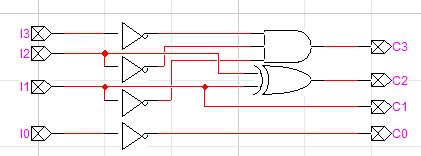
두 BCD A3 A2 A1 A0와 B3 B2 B1 B0의 합을 나타내주기 위한 4비트 가산기이다.

10원의 버튼만 입력된다면 or게이트를 통과해 0001로 1을 나타내주고, 50원 버튼을 입력 받는다면 0101로 5를 나타내준다. 따라서 10원과 50원, 100원과 500원 등을 계속 구분해서 입력시켜줄 수 있다. 입력 받은 돈(넣은 돈)은 BCD Adder를 통과해 나온 Selection값은 BCD Subtractor1(2번설명 참고)으로 들어가게 된다.

**5. BCD Subtractor**

1.BCD\_cpm

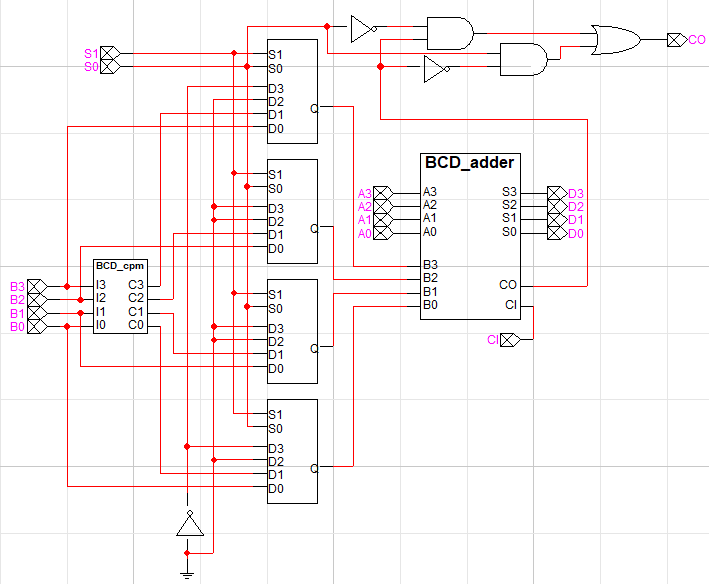




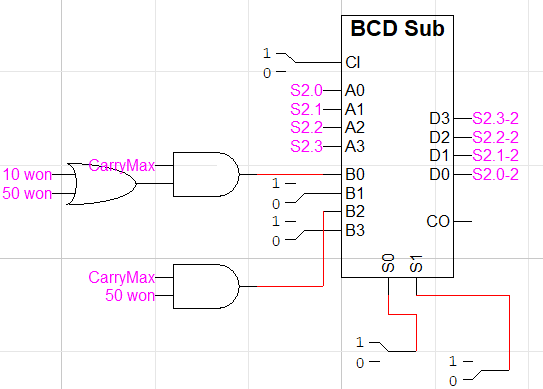
BCD\_cpm은 입력된 BCD값을 9의 보수로 만들어주는 기능을 한다.

2.BCD\_Adder

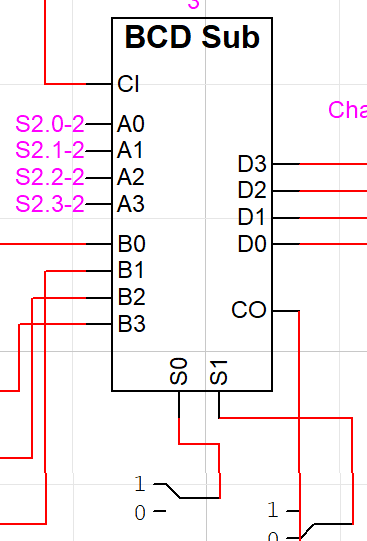
앞 설명을 참고한다.



BCD\_Sub의 전체 회로도이다. BCD\_Sub는 강의자료를 참고하여 만들었다. A-B를 연산한다 했을 때, 먼저 B의 값을 9의 보수로 바꾸어준다. 그 후 바꾸어준 값과 A값 그리고 1을 더해준다. 마지막으로 A-B값을 얻을 수 있다.

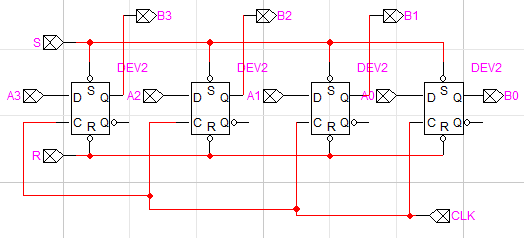
회로에서는 BCD Adder를 거친 후 두개의 BCD Subtractor를 거친다. 다음은 첫 번째로 통과하는 BCD Subtractor이다. 그림은 총 6개의 블록중 하나를 가져온 것이다 

BCD Subtractor1이라고 하겠다. CarryMax는 투입 금액 100,000자리에 해당하는 BCD Adder에서 나오는 Carry값이다. B0에는 ex)10won과 ex)50won을 or게이트로 연산, 그 출력을 다시 CarryMax와 AND게이트로 연산해준다. B1과 B3에는 0를 넣어준다. B2에는 CarryMax와 ex)50won을 AND게이트로 연산해준다. 연산을 하면 결과적으로 ex)10원이 들어올 경우 0001로 1을 나타내주고, 50원이 들어올 경우 0101로 5를 나타내준다. 따라서 1과 5를 구분해서 연산할 수 있다. BCD Subtractor1의 출력이 BCD Subtractor2의 A0..3의 입력으로 들어간다. BCD Subtractor 1는 100,000단위의 수가 9를 초과한다면 초과된 값을 다시 빼주어 되돌려주기 위해 만들었다. 즉, BCD Subtractor가 2개 존재하는 이유이다. 값이 초과되었는지 판단하는 기준은 CarryMax이다.



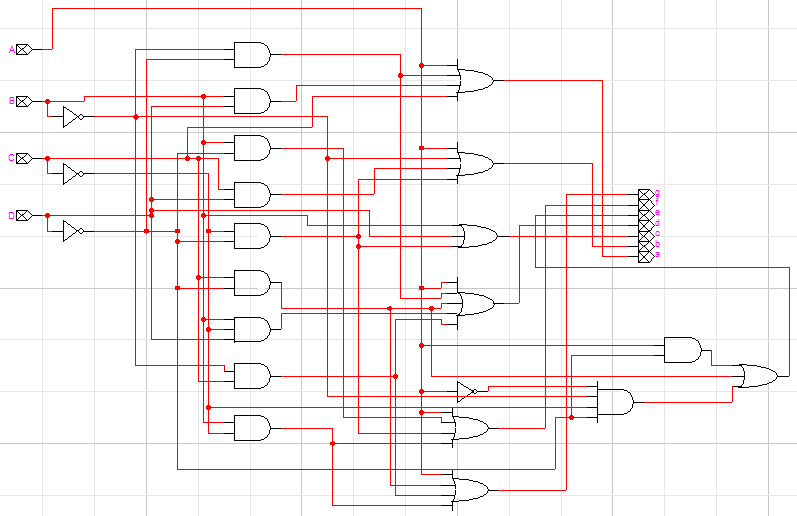
두번째로 통과하는 BCD Subtractor는 BCD Subtractor2라고 하겠다. BCD Subtractor1에서 들어오는 A0..3값(투입된 돈)과 beverages 2에서 들어오는 B0..3(음료가격)을 받아 A-B연산을 수행하고 그 결과값을 6번 설명에 나오는 4 X D FF에 저장한다. 음료를 구매하지 않아 beverages 2에서 입력이 들어오지 않는다면 BCD Adder -> BCD Subtractor1의 입력만 받아 저장장치인 4 X D FF에 저장해준다.

**6. 4 X D FF**



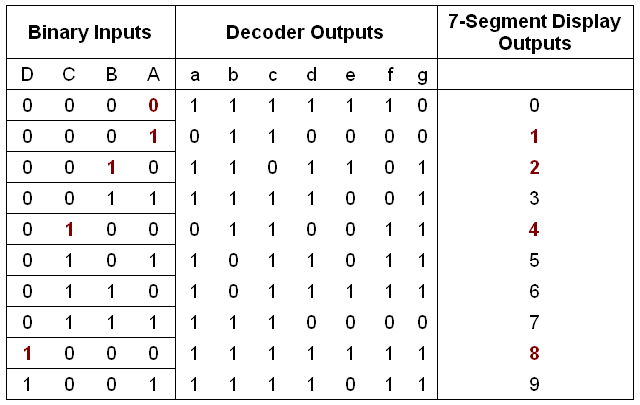
4 X D FF은 D flip-flops를 사용하여 구현한 4비트 레지스터이다. 감 가산을 모두 수행하고 나온 각 값들(투입된 돈에 관한 값, 음료에 대한 값)이 저장되는 곳이다.

**7. BCD 7segment**

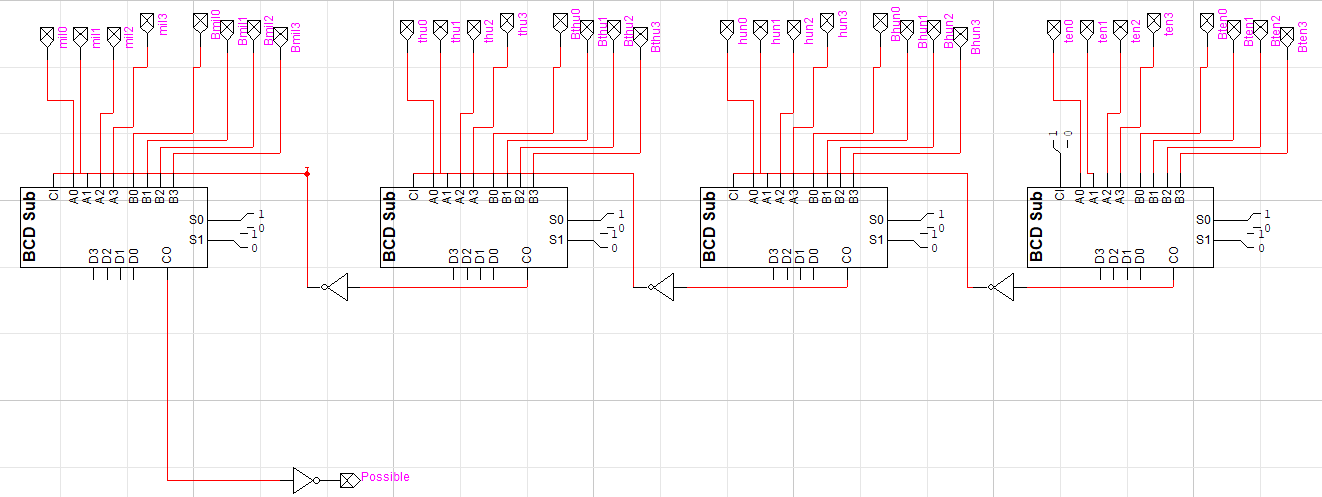
****

BCD(Binary Coded Decimum) 인코딩 체계에서 각 소수점(0-9)은 동등한 이진 패턴(일반적으로 4비트)으로 표시된다. Seven segment display는 일정한 패턴(공통 음극 또는 공통 양극 유형)으로 배열된 7개의 발광 다이오드(LED)로 구성된 전자 장치로서, 16진수 숫자(이 경우 입력은 BCD, 즉 0-9)를 표시하는데 사용된다. 그러나 7개 세그먼트 디스플레이는 다른 LED 세그먼트에 전압을 직접 공급하여 작동하지 않는다. 먼저, 우리의 십진수는 BCD 등가 신호로 변경되고 BCD 7 세그먼트 디코더는 그 신호를 7개 세그먼트 디스플레이로 공급되는 형태로 변환한다.

다음은 디코더의 진리표이다.

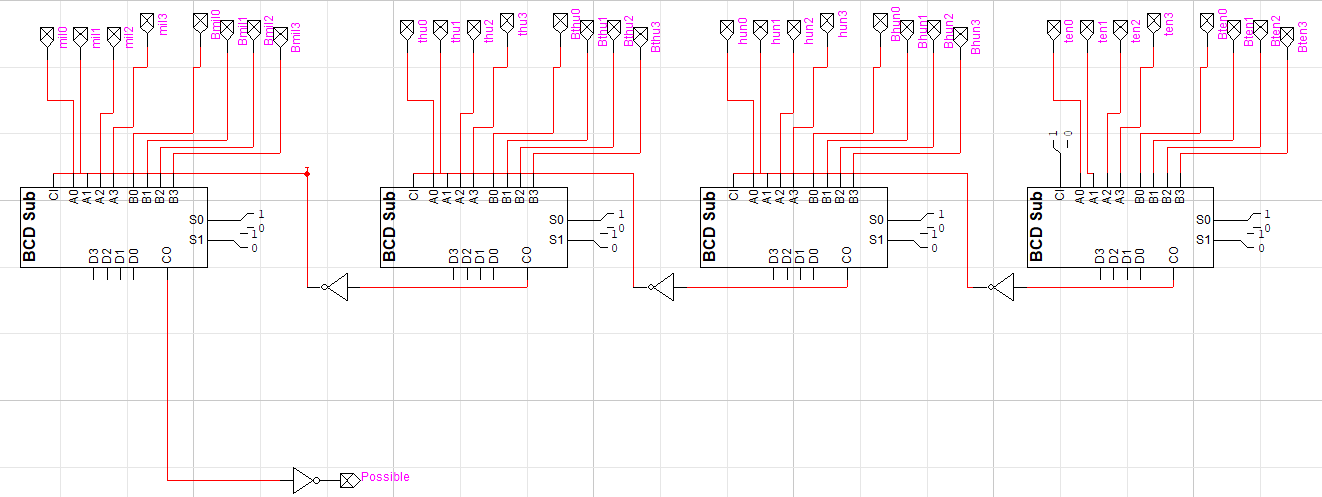


**8. PossibleCal**



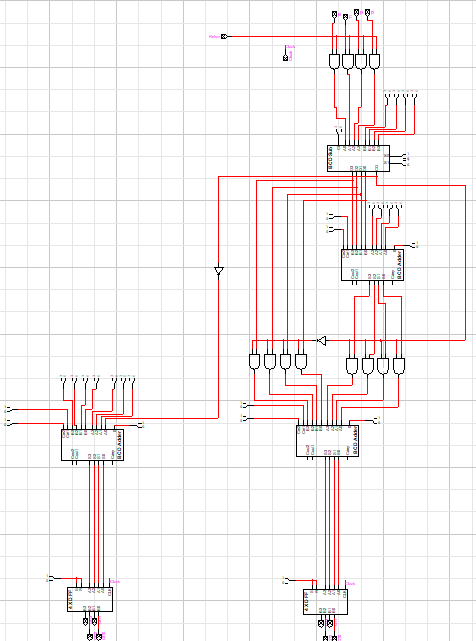
PossibleCal은 각 자릿수마다 작은 자릿수부터 투입된 돈에서 음료 가격을 빼는 작업을 하여 마지막으로 나온 Carry값이 1이면 음료가격이 투입된 돈보다 비싸므로 0(살 수 없다)을 반환해준다. 전체 회로도에서 PossibleCal의 출력 값인 Pcal이 beverages 2의 출력 B와 And게이트로 묶여 BCD Subtractor2의 입력 B로 들어가게 된다. PossibleCal은 음료가격의 한계값이 10,000이기 때문에 아래에 나오는 PossibleBlock보다 입력의 수가 4개 더 작다.

**9. PossibleBlock**



심볼의 구조와 기능은 앞에 설명된 PossibleCal과 동일하다. PossibleBlock과 PossibleCal을 구분하여 설정해준 것은 음료를 구입하고자 할 때 LED등의 입력을 출력 해주기 위해서이다. 즉 PossibleBlock의 출력이 음료 출력 LED의 input으로 들어가게 된다. 하지만 PossibleCal과 입력의 차이가 있다. 음료 가격은 10,000원이라는 한계 값이 있지만, 투입 금액은 999,999원까지 가능하다. 따라서 PossibleCal에 비해 입력이 4개 더 많다.

**10. ReturnT**

****

4 X D FF

BCD Adder

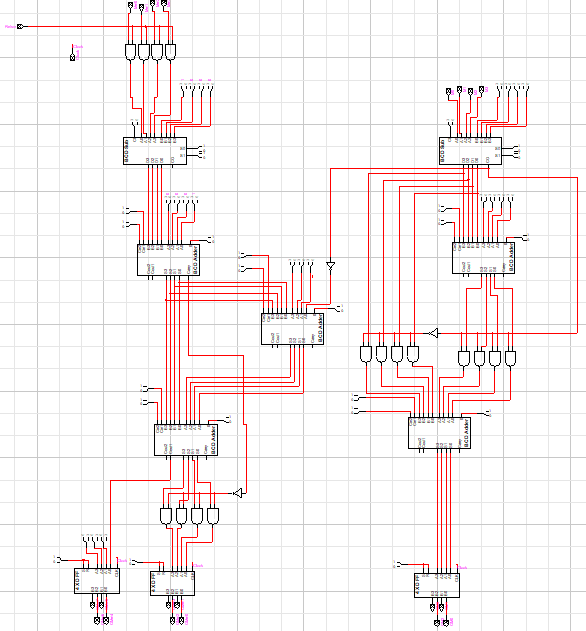
BCD Subtractor

10, 50, 100, 500, 1,000, 5,000원 단위의 반환에 사용되는 회로이다. 심볼의 구조는 BCD Adder와 BCD subtractor로 구성되어 있으며, 심볼을 작동 방식은 다음과 같다. 우선 BCD 값으로 받아온 단위에서 5를 감산한다. 이때, 데이터가 5보다 작다면 다시 5를 더해준 후, 그 수만큼 4 X D FF(10, 100, 1,000원)에 저장한다.

단, 데이터가 5보다 크다면 4 X D FF(50, 500, 5,000원)에 값을 1 증가시킨다. 그 후 남은 값은 그 수만큼 4 X D FF(10, 100, 1,000원)에 저장한다.

이러한 계산을 거쳐 나온 값들은 각 화폐단위의 4 X D FF에 저장된다. 저장된 값은 BCD 7 segment 심볼을 거쳐 변환되어 7segment의 입력에 연결된다.

**11. ReturnM**

****

4 X D FF

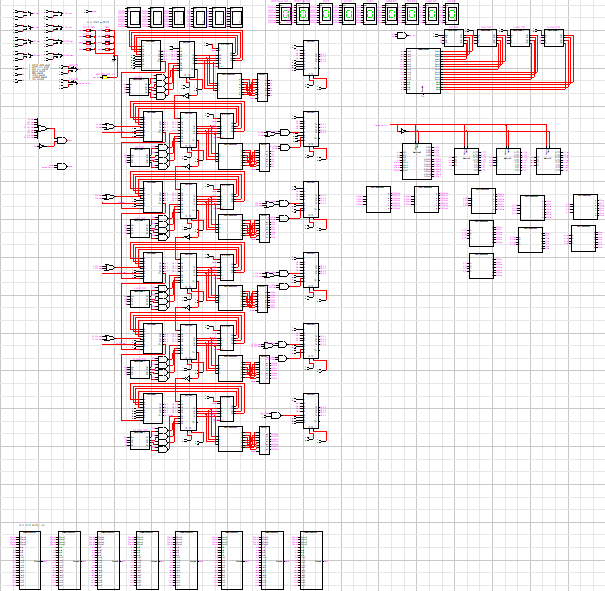
BCD Adder

BCD Subtractor

10,000, 50,000원 단위의 잔액을 반환하는 심볼이다. ReturnT와 따로 구성된 이유는 100,000의 존재 때문이다. 이 때 100,000원 단위는 개당 2개의 50,000원으로 처리할 수 있다. 심볼의 작동 원리는 Return T와 유사하다.

하지만 50,000의 개수 한도가 다른 단위에 비해 높기 때문에(잔액 한도가 999,999원) 50,000원의 개수가 10개가 넘어간다면 그 올림을 10장 단위의 50,000원 4 X D FF에 가산하여 저장한다.

**2-2. 전체 회로도 및 파트 분배**

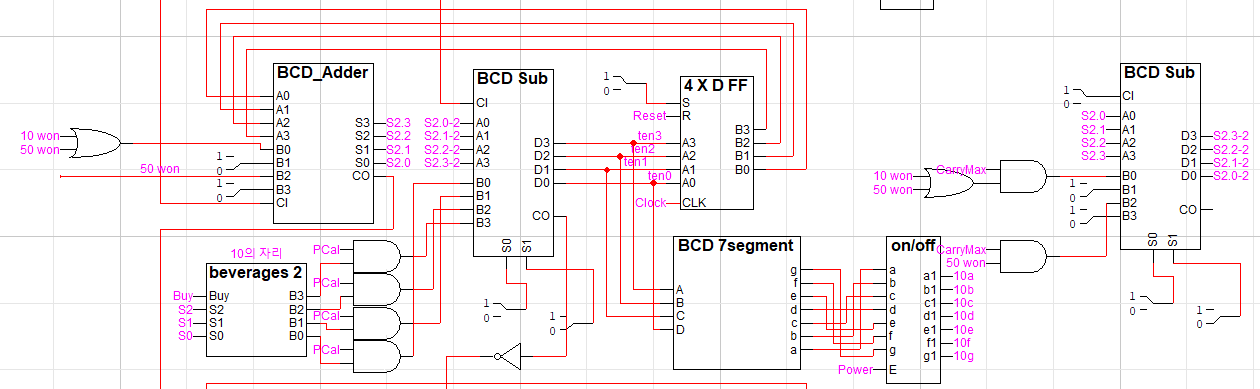
****

On/Off

Clock신호 출력

우리가 구현한 디지털 자판기의 전체 회로도이다. 색 별로 파란색은 데이터 처리 파트, 노란색은 배출 파트, 검은색은 반환 파트, 빨간색은 출력 파트, 녹색은 버튼 파트이다. 아래에 각 파트별로 기능들을 설명하겠다.

1. **데이터 처리 파트**

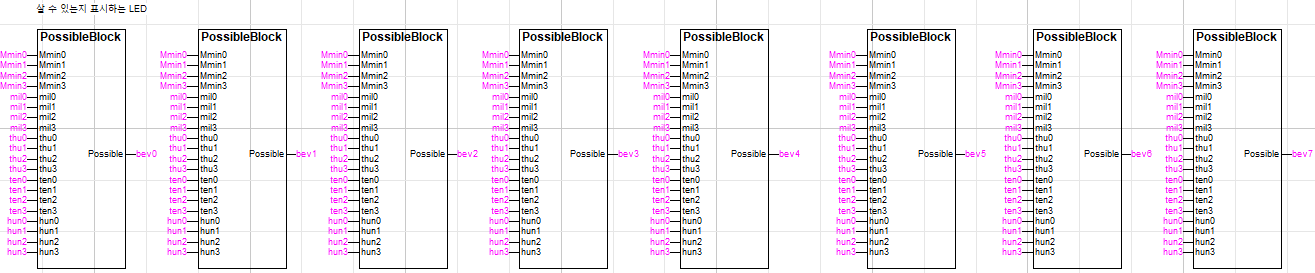


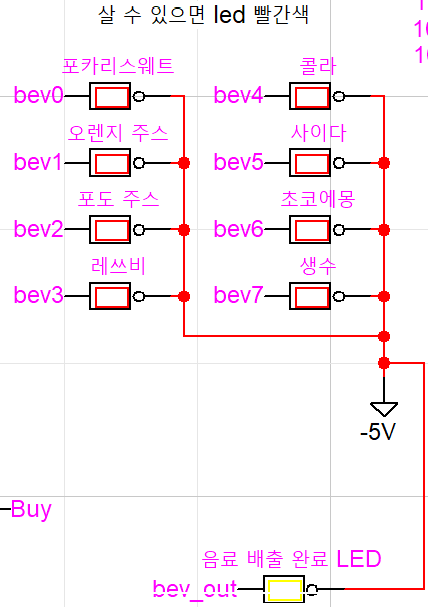
위 그림은 데이터 처리 파트 중 한 부분(10원, 50원 자리)을 잘라온 것이다. 돈을 넣을 때는 투입된 돈을 감가산하고 저장하는 기능을 수행하고 Buy버튼을 눌렀을 때에는 투입된 돈에서 음료 가격을 감산하고 저장하는 기능을 수행한다. 4 X D FF의 출력이 다시 BCD Adder의 입력 A0..3으로 들어가면서 돈의 값을 계속 갱신해 줄 수 있다, 실제로는 100,000자리까지 표현할 수 있게 총 6블록이 있다. 감가산한 결과를 BCD-7segment로 표시해준다. BCD Subtractor2의 CarryOut(CO)값은 다음 BCD Subtracctor2의 CarryIn(CI)으로 들어간다.

결과적으로, 6개의 블록들은 각각 총 가격의 한자릿수를 나타내게 된다.

Change\_Return은 잔돈 반환을 할 때, 원래 투입했던 돈을 0원으로 초기화 시켜주기 위해 사용한다.

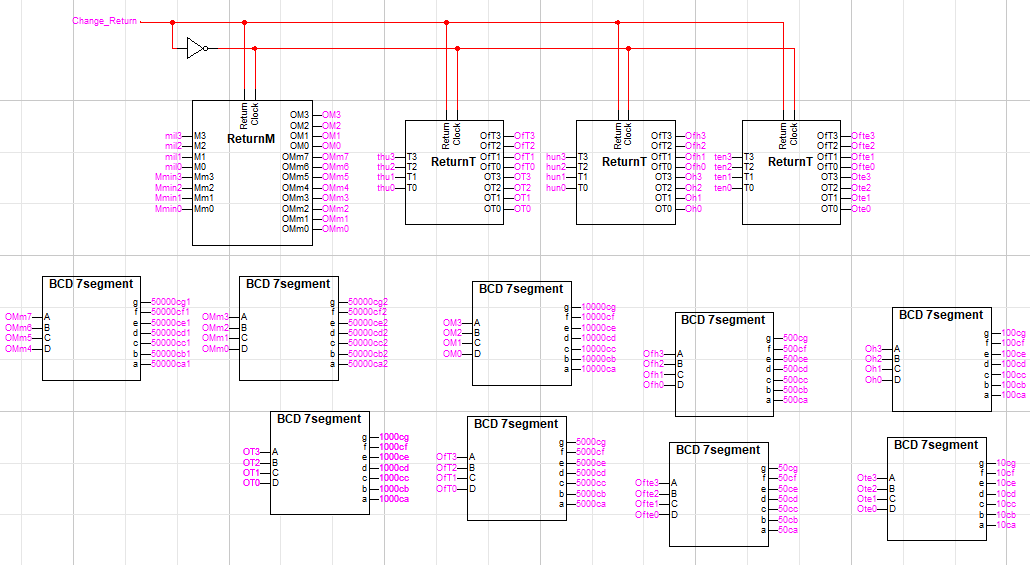
1. **배출 파트**





위 그림은 배출 파트와 LED출력 부분이다. 투입된 돈에서 음료 가격을 빼 CarryOut값을 산출한다. 출력 값인 CarryOut값이 1이면 (투입된 돈 – 음료 가격)이 음수이므로 음료를 뽑을 수 없다. 따라서 CarryOut값이 0일 때 살 수 있다. PossibleBlock의 입력으로 연산 파트 중 BCD Subtractor2의 출력을 가져온다. 각 자릿수를 조합하여 완성된 가격이 투입된 돈보다 작을 경우 LED(빨간색)의 등은 켜지게 된다. 음료 배출 완료 LED(노란색)는 Buy버튼을 누를 때 켜진다. 음료가 총 8개이므로 PossileBlock도 8개를 썼다.

1. **반환 파트**

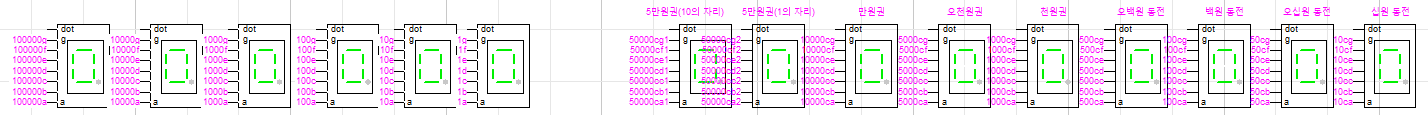




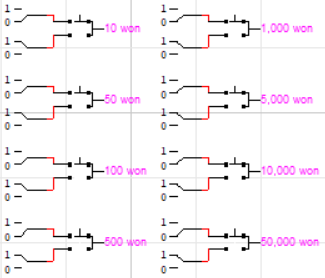
버튼 파트의 Change\_Return 버튼이 작동하면 실행되는 부분 기능 회로이다. 버튼이 작동한 시점의 데이터 처리 파트의 각 단위의 4 X D FF에 저장된 값을 가져와 잔액을 각 단위 별 개수로 7 segment를 통해 출력한다.

10~5,000원 까지는 ReturnT를 통하여 그 개수를 알아낼 수 있고, 10,000~50,000(1장 단위, 10장 단위)원은 ReturnM을 통하여 개수를 알아낼 수 있다. 이러한 계산으로 알아낸 각 단위의 개수는 각각의 4 X D FF에 저장되고, 이 값은 BCD 7 segment를 거쳐서 출력 파트의 잔액 표시 7-segment에 입력되어 출력된다.

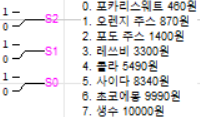
1. **출력 및 버튼 파트**

****

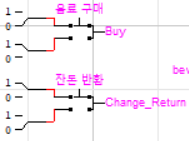
각각 연산 파트의 7-segment와 반환 파트의 7segment에서 입력을 받아 금액들을 7- segment display에 표시해준다



돈 투입 버튼이다. Binary Switch와 SPDT Pushbutton을 이용하여 평소에는 0을 가지게끔, 버튼을 누를 때는 1값을 가지게끔 만들어주었다.



음료 선택 스위치이다. Binary Switch로 입력 받아 0~7번의 음료를 선택한다.



위부터 순서대로 음료 구매 버튼, 잔돈 반환 버튼이다.

Binary Switch와 SPDT Pushbutton을 이용하여 평소에는 0을 가지게끔, 버튼을 누를 때는 1값을 가지게끔 만들어주었다.



On/Off 스위치(Power스위치, 전원)이다.

**3. 프로젝트 수행 보고서**

**3-1 프로젝트 팀원 및 참여도**

|  |  |  |
| --- | --- | --- |
|  | **맡은 역할** | **기여도** |
| **박동근** | 회로도 구현, PPT 작성, 보고서 작성 | 25% |
| **박상준** | 회로도 구현 및 종합, 회의 소집, 팀원 관리 | 25% |
| **이성현** | 회로도 구현 및 에러 조율, 프로젝트 발표 | 25% |
| **이정민** | 자료 수집, 회로도 구현, 보고서 작성 | 25% |

**3-2 프로젝트 추진 단계**

- 7segment 분석 및 BCD 7segment 심볼 구현

- BCD Adder 및 BCD subtractor 심볼 구현

- 금액 입력에 따른 잔액 증가 확인

- 음료 가격을 나타내기 위하여 beverages 1과 beverages 2 심볼 구현

- 음료의 구매에 따른 잔액 변화와 구매 가능 여부, 그에 따른 LED를 구현하기 위해서 PossibleCal, PossibleBlock 심볼 구현

- ON/OFF기능 구현

**3-3 프로젝트 전체 추진 일정**

|  |  |
| --- | --- |
| **날짜** | **추진 내용** |
| **2020. 11. 14** | 7segment 분석 및 BCD 7segment 심볼 구현 |
| **2020. 11. 21** | BCD Adder 및 BCD subtractor 심볼 구현,  회로 설계 및 금액 입력에 따른 잔액 증가 확인 |
| **2020. 11. 28** | 음료 가격을 나타내기 위하여 beverages 1과 beverages 2 심볼 구현  회로 설계 및 작동 확인 |
| **2020. 12. 05** | 음료의 구매에 따른 잔액 변화와 구매 가능 여부, 그에 따른 LED를 구현하기 위해서 PossibleCal, PossibleBlock 심볼 구현  회로 설계 및 작동 확인 |
| **2020. 12. 14** | ON/OFF기능 구현 및 작동 확인 |

**3-4 프로젝트 팀 전략 및 성과**

- BCD Adder, BCD subtrator, BCD 7segment 등을 수업 시간에 배운 내용과 자료 조사로 얻은 내용을 바탕으로 제작하여 자판기 기능들을 구현하는 데에 큰 도움이 되었다.

- 회로의 연산의 처음부터 끝까지 BCD 단위만을 사용하여 회로 제작이나 오류 발생시에 용이하였고 곱셈이나 나눗셈, 나머지 연산 대신 덧셈과 뺄셈만을 이용한 회로를 제작하여 구현 난이도가 보다 쉽고 계산 속도가 빠르게 하였다.

- 자판기의 실제 기능을 사용하여 판매가능 음료를 LED로 표현해 사용자가 먼저 구매가능한 음료를 인식할 수 있도록 하고 그럼에도 구매를 했을 경우, 그 행동을 저지한다

**3-5 팀 회의 일지**

|  |  |
| --- | --- |
|  | **1차 회의** |
| **일시** | 2020.11.14 |
| **장소** | 학교 앞 카페 |
| **조원**  **및**  **참석여부** | 전원 참석 |
| **내용** | 기본적인 회로 아이디어 토의  7segment 분석 및 BCD 7segment 심볼 구현 |

|  |  |
| --- | --- |
|  | **2차 회의** |
| **일시** | 2020.11.21 |
| **장소** | 학교 앞 카페 |
| **조원**  **및**  **참석여부** | 전원 참석 |
| **내용** | BCD Adder 및 BCD subtractor 심볼 구현,  회로 설계 및 금액 입력에 따른 잔액 증가 확인  추가적인 아이디어 토의 |

|  |  |
| --- | --- |
|  | **3차 회의** |
| **일시** | 2020.11.28 |
| **장소** | 신공학관 3165 |
| **조원**  **및**  **참석여부** | 전원 참석 |
| **내용** | beverages 1과 beverages 2 심볼 구현  회로 설계 및 오류 제거, 작동 확인 |

|  |  |
| --- | --- |
|  | **4차 회의** |
| **일시** | 2020.12.05 |
| **장소** | 신공학관 3165 |
| **조원**  **및**  **참석여부** | 전원 참석 |
| **내용** | PossibleCal, PossibleBlock 심볼 구현  회로 설계 및 작동 확인  ON/OFF 기능에 대한 토의 |

|  |  |
| --- | --- |
|  | **5차 회의** |
| **일시** | 2020.12.14 |
| **장소** | 신공학관 3165 |
| **조원**  **및**  **참석여부** | 전원 참석 |
| **내용** | ON/OFF 기능 구현  보고서 및 PPT에 관한 토의 |

**4.참고 문헌**

-한국기술교육대학교 정보기술공학부 강의자료

-수업 강의 자료