ECE2030h - HW-4 v.2 Due Thursday 10/11 before class.

Problem 1. Design a full-adder using NOR gates by the Mixed Logic Technique.

Show the each stage separately. Do not use XOR gates.

The procedure for performing mixed logic conversions is as follows:

1. Draw the logic network for the given Boolean equation using only AND and OR gates. Replace all complements with a bar (no bubbles or inverters yet!)

Once the initial Boolean equation is drawn with AND gates, OR gates and bars, the self-documenting redesign begins.



Sum = A(BC + B'C') + A'(BC' + B'C)

Cout = AB + BC + AC
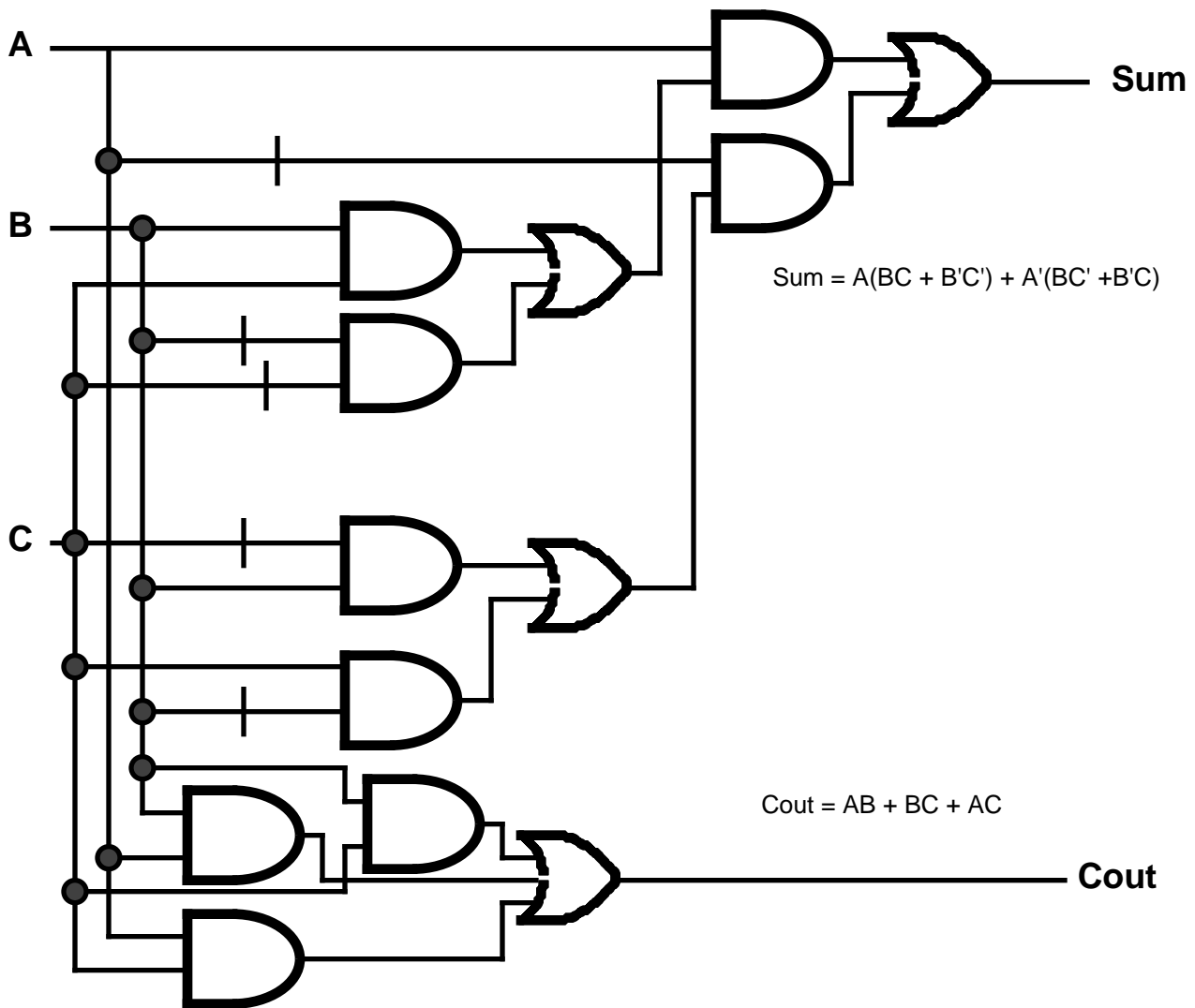
2. Add complement bubbles and NOT gates (inverters) within the network to appropriately convert logic gates to the desired gate sets.

The rules in adding complement bubbles and NOT gates

 a. All pairs of bubbles must cancel each other out.

 b. Exactly one and only one bubble needed on each bar (which then can disappear).

 c. Add a NOT gate and a bubble where needed to make bubbles cancel, or a bubble is needed to make a gate the desired type.



Sum = A(BC + B'C') + A'(BC' +B'C)

Cout = AB + BC + AC
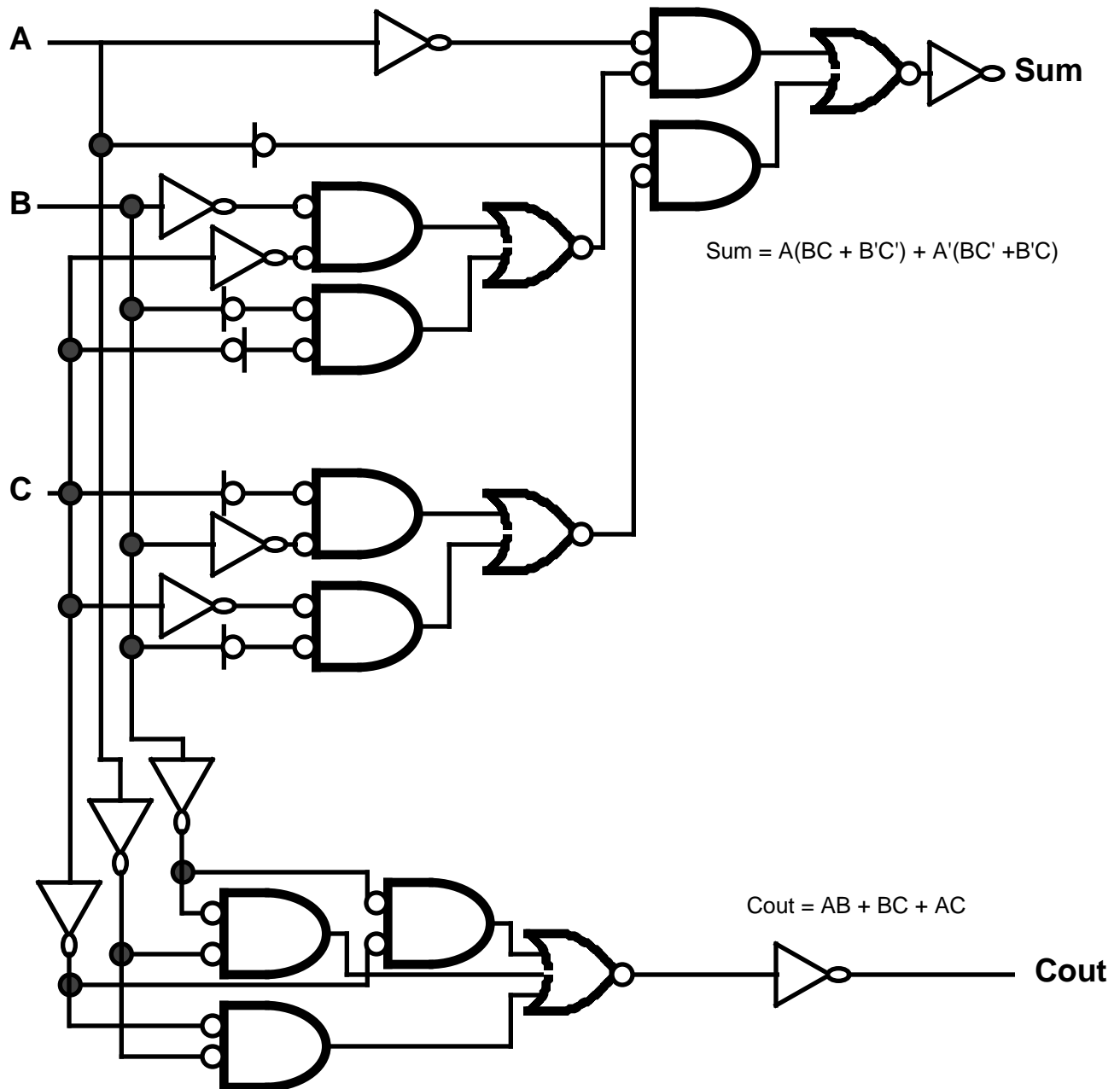
3. Replace DeMorgan equivalent gate symbols with standard gate symbols (e.g., NOR gates should be shown as an OR gate symbol with a "bubble" on the output instead of an AND gate symbol with bubbles on the inputs).
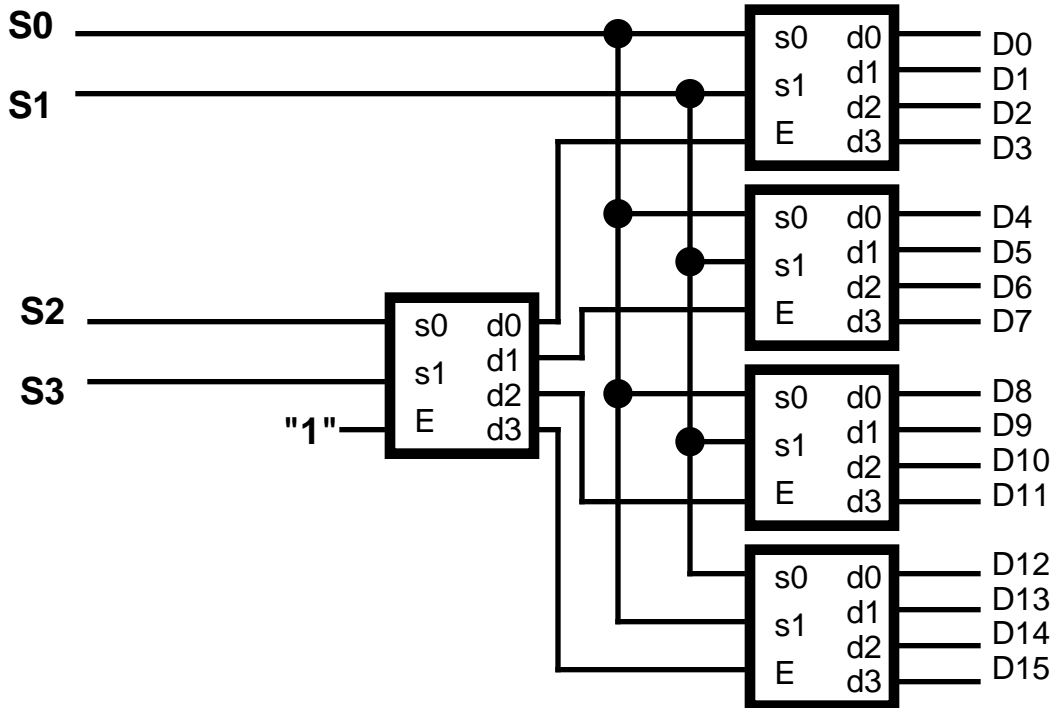
**A**

**B**

**C**

Sum = A(BC + B'C') + A'(BC' +B'C)

**Sum**

Cout = AB + BC + AC

**Cout**
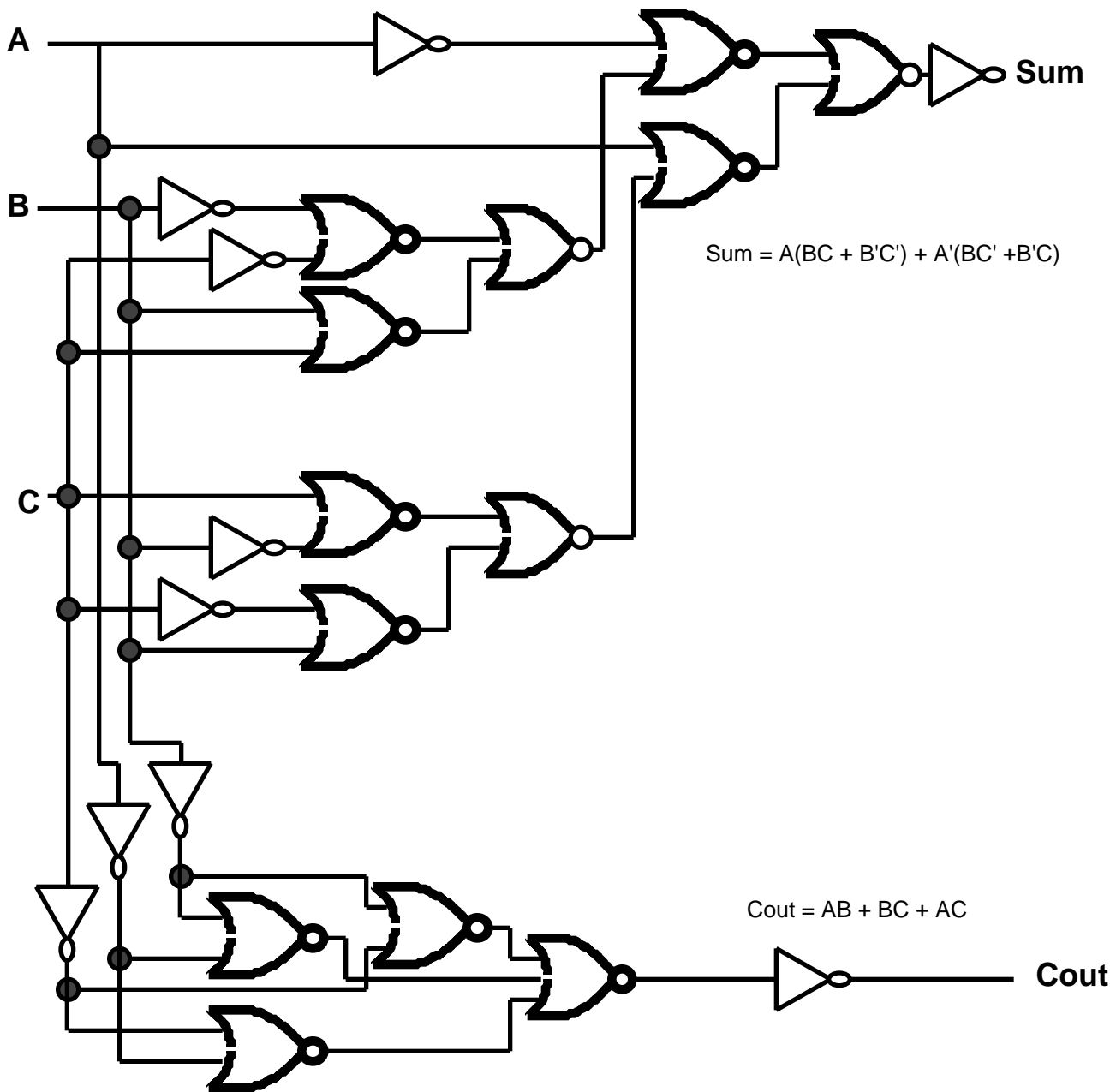
Problem 2. On a separate sheet of paper, use five 2-bit decoders to make a 4-bit decoder. Use the enable (E) inputs appropriately (connect to logic "1" when not used).

The circuit inputs are S0, S1, S2, S3 which form the four-bit binary number that set one of 16 output lines D0, D1, ..., D15 true ('1"). Each 2-bit decoder has inputs s0, s1, E and outputs d0, d1, d2, d3.

```
S0 ─────────────────────●────────┌──────────┐
                        │         │ s0    d0 │──── D0
S1 ───────────────────●─┼───────  │       d1 │──── D1
                      │ │         │ s1    d2 │──── D2
                      │ │         │ E     d3 │──── D3
                      │ │         └──────────┘
                    ●─┼─┼───      ┌──────────┐
                    │ │ │         │ s0    d0 │──── D4
                  ●─┼─┼─┼───      │       d1 │──── D5
                  │ │ │ │         │ s1    d2 │──── D6
                  │ │ │ │         │ E     d3 │──── D7
         ┌──────────┐  │          └──────────┘
S2 ──────│ s0    d0 │  │          ┌──────────┐
         │       d1 │  ●──────    │ s0    d0 │──── D8
S3 ──────│ s1    d2 │             │       d1 │──── D9
         │ E     d3 │    ●─────   │ s1    d2 │──── D10
    "1"──│          │    │         │ E     d3 │──── D11
         └──────────┘              └──────────┘
                                  ┌──────────┐
                                  │ s0    d0 │──── D12
                                  │       d1 │──── D13
                                  │ s1    d2 │──── D14
                                  │ E     d3 │──── D15
                                  └──────────┘
```

Problem 3. A keypad has 16 buttons that place a logic "1" signal on one of 16 output lines. Show how to use a single logic building block circuit that we discussed in class to generate a unique 4-bit binary number that indicates which key was pressed. Identify the type of building block circuit that is used.

```
┌──────────┐        ┌──────────┐
│          │════════│ 16-line  │────────── bit-0
│  Key Pad │        │ (4-bit)  │
│ (16 keys)│        │ Encoder  │────────── bit-1
│          │        │          │
│          │        │          │────────── bit-2
│          │        │          │
│          │        │          │────────── bit-3
└──────────┘        └──────────┘
```