COMPUTER SCIENCE TRIPOS Part IB – 2017 – Paper 3

7 Prolog (ARB)

(*a*) Define `and/3` to model an AND gate, using `0` and `1` to mean "false" and "true" respectively. [2 marks]

(*b*) A definition of `or/3` is shown on the left, and a query's result on the right:

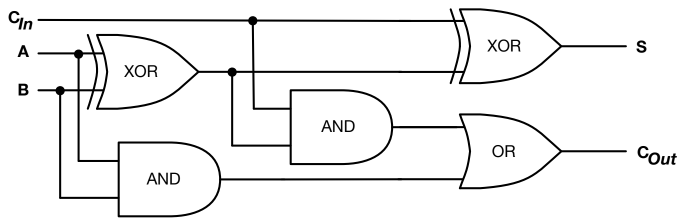
```
or(0,0,0).
or(_,_,1).
```

```
?- or(0,0,Result).
Result = 0 ;
Result = 1.
```

Explain this output, and correct `or/3` without increasing the number of clauses. [3 marks]

(*c*) Define `xor/3` using two rules. [2 marks]

(*d*) Define `full_adder(A, B, CIn, COut, S)` to implement the following circuit.



[5 marks]

(*e*) Define `zip(InList1, InList2, OutList)` such that if `InList1` is [1,3,...] and `InList2` is [2,4,...] then `OutList` must be [[1,2],[3,4],...]. [2 marks]

(*f*) Define `ripple_carry_adder(N, Inputs, CIn, COut, Result)` to cascade `N` calls to `full_adder/5`. Assume that we obtain parameter `Inputs` through `zip(`X`, `Y`, Inputs)` to add two `N`-bit values X and Y. In your answer take the most significant bit to be on the *right*. Thus you should expect to see:

```
?- ripple_carry_adder(2, [[1,1], [0,0]], 0, Cout, S).
Cout = 0, S = [0, 1].
?- ripple_carry_adder(2, [[0,0], [1,0]], 0, Cout, S).
Cout = 0, S = [0, 1].
?- ripple_carry_adder(2, [[0,0], [1,1]], 0, Cout, S).
Cout = 1, S = [0, 0].
```

[2 marks]

(*g*) Define the predicate `test(X,Y,N)` which tests `ripple_carry_adder/5` against Prolog's built-in addition function for up to `N` bits of precision, that is, fixed width `test(X,Y,N)` fails if overflow occurs (i.e., the carry bit is set). You may assume you are given predicates `dec2bin(Dec,BinList)` and `bin2dec(BinList,Dec)` for converting between integers and lists of bits, and `length(List,N)` which relates lists with their lengths. [4 marks]