1994 Paper 8 Question 3

Comparative Architectures

A naïve user translates the following C code:

```
extern int h(int *x, int flag);
int f(int *x) { return h(x, x[0]); }
int g(int *x) { return h(x + 2, (x[5]+x[6]) | 1); }
```

into the following assembly code for the MIPS R2000:

```
   .set    noreorder
   .globl  f,g       ; export f,g (implicitly import 'h')
f: lw      $5,0($4)  ; r4 holds 'x', load *x as 2nd argument
   j       h         ; tail-call to h
g: add     $4,$4,8   ; r4 holds 'x', load &x[2] as 1st argument
   lw      $5,12($4) ; r4 holds 'x+2', load x[5] as 2nd argument
   lw      $6,16($4) ; r4 holds 'x+2', load x[6] to a temporary
   add     $5,$5,$6  ; do the '+' ...
   or      $5,$5,1   ; ... and the '|'.
   j       h         ; tail-call to h
```

Each line of the above assembler program is individually a legal instruction or pseudo-instruction with valid comment. Explain in detail the effect of calling the assembler version of `f` with a suitable argument including a C function which exactly corresponds to the effect of the assembler version of `f`. [5 marks]

State the purpose of the `.set noreorder` directive. [2 marks]

Explain how the programmer has failed to understand the R2000 instructions and give a correct translation of the C code into assembly code (do not suggest removing the `.set noreorder` as part of the answer). [5 marks]

Explain briefly the origins of the errors (in both `f` and `g`) in terms of MIPS R2000 architecture. [5 marks]

How might knowing the first instruction of the compilation of `h()` affect your answer? [3 marks]