CS1Q Computer Systems

Solutions Week 17 (Lab) *Distributed* **Week 18**

Solutions Week 17 (Lab)

Programs

This is Program1.

```
% This is the program from Lecture 3.
% It uses memory locations for variables x, y, z
% and does some simple operations, corresponding
% to the following Ada code:
%
% x := 5;
% y := 3;
% z := x * 2 + y;

LDVAL R6, $0005
STORE R6, x[R0]
LDVAL R6, $0003
STORE R6, y[R0]
LOAD R1, x[R0]
LOAD R2, y[R0]
LDVAL R4, $0002
MUL R5, R1, R4
ADD R3, R5, R2
STORE R3, z[R0]
CALL exit[R0]

x DATA $0000
y DATA $0000
z DATA $0000
```

This is Program2, with the desired modification (assignment to register R1).

```
% This is the program from Lecture 3.
% It corresponds to the following Ada code,
% calculating the sum of the integers up to n.
%
%       s := 0;
%       while n > 0 loop
%         s := s + n;
%               n := n - 1;
%       end loop;
%
% The registers are used as follows:
%
%       R1 is n
%       R2 is s
%       R3 is 0
%       R4 is a temporary value
%       R5 is 1

% The exercise asks for R1 to be set, like this:
        LDVAL   R1,$0005

        LDVAL   R2,$0000
loop    LDVAL   R3,$0000
        CMPGT   R4,R1,R3
        JUMPF   R4,end[R0]
        ADD     R2,R2,R1
        LDVAL   R5,$0001
        SUB     R1,R1,R5
        JUMP    loop[R0]
end     CALL    exit[R0]
```

This is Program3, with the desired modification.

```
% Now we are modifying Program 2 so that it
% calculates the product from 1 up to n
% (i.e. the factorial function).
%
%       s := 1;
%       while n > 0 loop
%         s := s * n;
%         n := n - 1;
%       end loop;
%
% The registers are used as follows:
%
%       R1 is n
%       R2 is s
%       R3 is 0
%       R4 is a temporary value
%       R5 is 1

% Again we need to initialise n

        LDVAL   R1,$0005
        LDVAL   R2,$0001  % was $0000
loop    LDVAL   R3,$0000
        CMPGT   R4,R1,R3
        JUMPF   R4,end[R0]
        MUL     R2,R2,R1  % was ADD
        LDVAL   R5,$0001
        SUB     R1,R1,R5
        JUMP    loop[R0]
end     CALL    exit[R0]
```

This is one possible implementation of Program4. Variations are possible, for example storing the variables in memory instead of in registers.

```
% Calculating the sum of the elements of an array.
% Based on the program from Lecture 4
% which finds the largest element of an array.

% Ada code:

%     sum := a[0];
%     i := 1;
%     while a[i] <> -1 loop
%       sum := sum + a[i];
%       i := i + 1;
%     end loop;

% Registers:

%    R1 = sum
%    R2 = i
%    R3 = -1
%    R4 = 1
%    R5 = a[i]

        LDVAL   R3, $ffff   % R3 := -1
        LDVAL   R4, $0001   % R4 := 1
        LOAD    R1, a[R0]   % sum := a[0]
        LDVAL   R2, $0001   % i := 1
loop    LOAD    R5, a[R2]   % R5 := a[i]
        CMPEQ   R6,R5,R3    % R6 := (a[i] = -1)
        JUMPT   R6,end[R0]  % if a[i] = -1 then exit loop
        ADD     R1,R1,R5    % sum := sum + a[i]
        ADD     R2,R2,R4    % i := i + 1
        JUMP    loop[R0]    % go to top of while loop
end     CALL    exit[R0]    % stop

a       DATA    $0002       % values in array a
        DATA    $0005
        DATA    $0001
        DATA    $0007
        DATA    $0003
        DATA    $ffff       % indicates end of array a
```