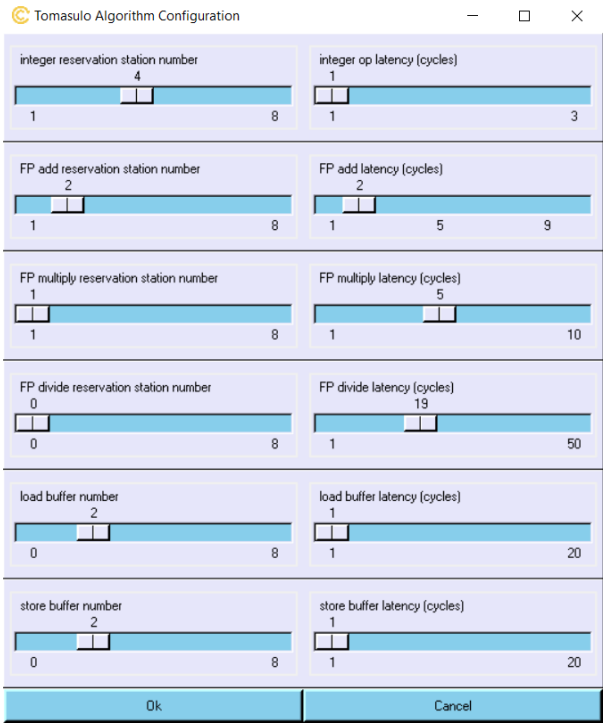
Assignment 1 - Tomasulo

1. By using DLXview (run dlxview.exe file), make the following exercise:

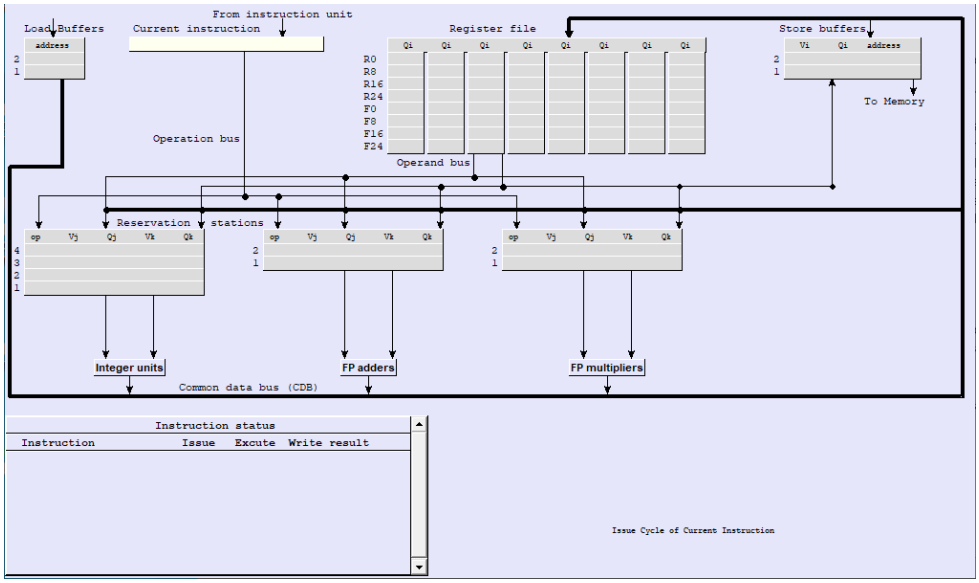
Let us imagine a processor that implements the Tomasulo algorithm without speculation with the following features:

- Integer (INT) instructions are processed in a similar way than floating point (FP) ones, so:
  - INT units have a number of reservation stations (RS), where instructions wait until their RAW dependencies are solved.
  - INT instructions use the CDB to write their results in the INT register file.
- This processor features the following hardware (which has to be configured in DLXview through the "configure" button that you can see in the DLXview program once you open it, then you select "Tomasulo"):
  - INT: latency=1 cycle
  - FPadd: latency=4 cycles
  - FPmul: latency=6 cycles
  - Load: latency=1 cycle
  - Store: latency=1 cycle



- For now, you are free to select the number of reservation stations of each type that you consider. DLXView will assume that there is one functional unit associated to each RS of each type, and the FP adders and multipliers are pipelined. In addition, the delay slot of branches is 1 instruction. These features cannot be changed in DLXview.

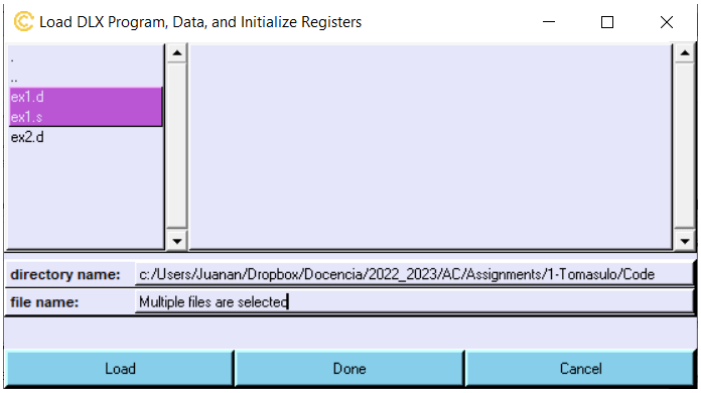
- In RAW dependencies, the consumer instruction cannot be executed in the same clock cycle as the producer instruction puts the involved data in the CDB.



Determine the minimum number of resources (number of RSs, load buffers and store buffers) that are needed so that the issue of instructions is never stopped at any moment. Let us assume that the following code is executed:

```
      ld    f2, a
      add   r1, r0, xtop
loop: ld    f0, 0(r1)
      sub   r1, r1, #8
      multd f4, f0, f2
      bnez  r1, loop
      sd    8(r1), f4
      trap  #0
```

This code is provided in the `ex1.s` file. This file, alongside `ex1.d`, must be selected in DLXview and loaded by using the "load" button of DLXview, which is now unblocked once you have configured the processor:



2. Assuming the same processor as in Exercise 1, write a DLX code for a neural network with 5 entries composed of 2 neurons of type perceptron. Let us assume the following pseudo-code:

```
// matrix with weights
float W[2][5]={0.3,0.45,1.2,6.8,3.2,1.1,0.8,2.2,1.5,0.68};
float B[2]={0.3,1.1}; // bias vector
float U[2]={2.3,3.8}; // threshold vector
float X[5]={ 0.1,0.1,0.2,0.3,0.4}; // input vector
float Y[2]; // output vector

int main() {
int i,j;
float tmp;

for (i=0;i<2;i++) {
      tmp=B[i];
      for (j=0;j<5;j++) {
            tmp=tmp+W[i][j]*X[j];
      }
      if (tmp> U[i]) Y[i] = tmp;
      else Y[i] = 0;
}
return 0;
}
```

Note: For the assignment, it is enough to provide the two files with your code, which can be named `ex2.s` and `ex2.d`.