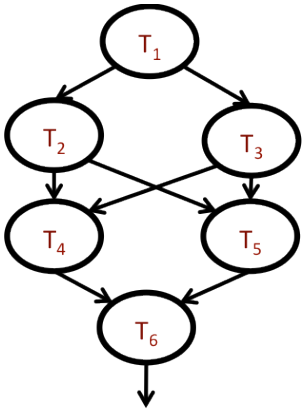
Brown University
Division of Engineering
EN2911X Reconfigurable Computing.
Prof. Sherief Reda
HW/LAB 04. [35 points – No team work allowed]
Due Date: Friday, Oct 10th (in class).

BROWN

Q1 [35 points]

A. [3 points] For the given task graph, report the execution time, total LEs and total memory for a pure SW implementation and for a pure HW implementation.



It is desired to identify a SW/HW partitioning such that the execution time is minimized such that the total number of LEs is 60. For all of the following questions, the requirements are to report the HW partition, SW partition, the execution time, and the number of LEs and the total memory size.

B. [5 points] Starting with a SW solution, apply the greedy heuristic to identify a SW/HW solution and report the requirements.

C. [7 points] Write down the linear equations that capture the objective and constraints. Use the LPSOLVE package (http://lpsolve.sourceforge.net) to solve the integer linear program and report the requirements. Contrast the optimal solution to that of the greedy heuristic in (B).

D. [8 points] If the total number of LEs are varied from 0 to 160 LEs in steps of 20, re-solve the part (c) and report the execution time versus the number of LEs in a plot (x-axis is number of LEs allowed and y-axis is the execution time)

E. [7 points] Given a budget of 60 LEs and a memory budget of 65 KB, resolve the ILP and report the requirements. Contrast the solution to that of question (C).

F. [5 points] Re-solve the linear equations, but this time without enforcing the binary constraint on the task variables (x_i). Instead, enforce only lower bounds (0) and upper bounds (1) on the variables x_i that represent the tasks implementation. Note that in this case we are solving a linear program. Does the resultant fractional solution make sense? Can we make the fractional solution sensible by rounding it to integers? Comment on the optimality of this approach.

	HW execution time (ns)	**HW area (LEs)**	**SW execution time (ns)**	**SW area (KB)**
T_1	4	21	16	10
T_2	3	29	18	40
T_3	9	17	36	13
T_4	5	48	25	16
T_5	10	45	60	19
T_6	2	17	14	12

LPSOLVE is really easy to use. For instance, here is the code for the LP example of slide 13 in the lecture

```
max: x1 + x2;
2x1 + x2 <= 4;
3x1 + 4x2 <= 12;
x1 >= 0;
x2 >= 0;
```

and the code for slide 14 is

```
max: x1 + x2;
2x1 + x2 <= 4;
3x1 + 4x2 <= 12;
int x1, x2;
```

To enforce binary integer constraints on variables, say x1 and x2, you would use “bin x1, x2” instead of “int x1, x2”.

Note: you can use “min:” instead of “max:” and you can use other relational operators for the constraints (e.g., "<" "<=" "=" ">" ">=")