

**Programming Assignment 5: Due May 28 by 5pm**

Write two C programs that generate the register allocation and functional unit allocation of a scheduled dataflow graph. Register allocation should use the greedy clique partitioning algorithm you developed for PA1. Functional unit allocation should use weighted clique partitioning with the weights developed by Tseng and presented in class. You only need to do clique partitioning on the functional unit compatibility graph once. Build your compatibility graph so that operations that do not share the same resource type are not compatible. You should take operations into consideration. Note that operations are different from resource types. Also, note that input and output operations have one variable and one operation.

The input to your program will be a graph and the maximum number of csteps. `main.c` will call force directed scheduling to schedule your graph, then call register allocation, and then call functional unit allocation. The output is a scheduled and bound graph represented by the reservation table.

Your program should make use of the high level synthesis library `libHLS`. `libHLS` includes several functions for user info that you should use to represent information shared between register allocation and functional unit allocation.

To submit your programming assignment, put your code in your COE account under the link `Courses/ECE3485/PA5/`. Submit the three files: `synthesize.h`, `reg_alloc.c` and `fu_alloc.c`. Make sure your files are group readable.

If you have questions on this assignment, send email to `mel@ece.neu.edu`.

We will run your code according to the instructions. You will be graded on:

1. How well your code runs (including whether you followed the directions)
2. how well your code works, and
3. the quality of the code and the comments.

Uncommented code will lose points.

## Getting Started

Instructions for this assignment can be found in `~libhls/PA5/README`.

A skeleton C file for your code can be found in `~ece3485/PA5/examples/main.c`. The program parses command-line arguments and calls: `reg_alloc.c` and `fu_alloc.c`. Your code goes into these two functions. Any user defined data structures go into `synthesize.h`.

All work in this class is expected to reflect your individual effort.