

## Homework IV

CS 8803 : Languages and Compilers for Embedded Systems

Topic : Register Allocation and Instruction Scheduling

Due Date : July 21<sup>st</sup>, Tuesday, 6 pm

Total points : 100

### Important Instructions:

1. The homeworks are non-collaborative. You are allowed to discuss the homework amongst yourself for clarification purposes only. We strongly encourage you to clarify things with the TA however. Any collaboration about the solution will be treated as plagiarism.
2. Keep your answers precise and try not lengthy.
3. When necessary make suitable assumptions, clearly stating and highlighting them.
4. Please turn in the homework on time – late home-works may not be accepted.

**Question I [20 points]** Show an example interference graph where Briggs' algorithm generates non-optimal spill decisions - your example can either show that Briggs's comes up less than optimal spill code (ie, lower cost spill code can be found by a better algorithm) (acceptable solution) or second possibility could be that Briggs' algorithm generates a spill when there exists a zero spill coloring solution (a solution that is really dear to my heart but is a difficult one). Assume a suitable number of available registers for your example.

**Question II [25 points]** Devise an algorithm to generate live ranges and webs given LIVEIN [B] and LIVEOUT [B] sets being already computed for each basic block by the liveness analysis pass.

**Question III [25 points]** Show an interference graph example along with a SSA based control flow graph where : (a) coalescing will be useful – that is it will eliminate register-to-register moves without causing extra spills, (b) coalescing will not be useful where extra spill will be generated but register-to-register moves will be removed.

**Question IV [30 points] Part A:** Modify the list scheduling algorithm in the course slide to account for variable latency of an instruction. For example, consider a LD (load) instruction, depending on whether it leads to cache hit or miss it can take either 2 or 5 cycles. Your schedule should benefit from low latency but should also not cause extra stall cycles due to longer latency. **Part B:** After instruction schedule is generated, it is decided to move some LD/STs to occupy empty slots but this will have an effect on unsafe code due to register occupancy at the point of motion of LD/ST. Thus, one must perform selective register re-allocation. First show an example of why and how this will happen and then sketch an approach to selectively recolor the register allocation based on the new LD/ST placement points.

