High Level Synthesis

Computer Aided Design for Reconfigurable Computer Systems

By: Garrett Reynolds
Contents

• What is HLS and Why use HLS
• Synthesis
  – Scheduling & Allocation
• Problems
• HLS with Reconfigurable Datapath Components
• Resources
What is HLS

- HLS is the process of taking a behavioral description and automatically translating it to a structural description at the register transfer level.

- Process must take user constraints and hardware constraints into consideration.

- Structural description will consist of functional units, memory elements and interconnections.
Why use HLS?

- **Shorter design cycle**
  - Get designs out the door quicker, lower costs
- **Less Errors**
- **Large variety of solutions**
  - Designer can choose design depending on different trade offs
- **Self documenting**
- **Make technology available to more people**
Synthesis

• Is broken down into 2 main areas, synthesis and allocation

• Before synthesis can take place an internal representation must be made from the programmed language

• The internal representation is usually in the form of a graph and takes data flow and control into consideration
Synthesis

• A data flow graph (DFG) is a common way to represent the internal structure.

• Operations are shown in nodes and are linked together if there is a data dependency between nodes.

• $Y = \text{max}((\text{Ashr1}) + (B - (B \text{shr} 3)), B)$
Synthesis

- The DFG does not include control for loops and branches.
- The DFG can be augmented with control nodes to form a control data flow graph CDFG.
Scheduling

• The aim is to reduce the number of control steps to complete the program

• The first figure would require 23 control steps, where the second figure would only require 10
Scheduling

• To ensure efficient scheduling a designer has to consider the interaction with allocation as well as the type of scheduling algorithm that will be used

  – To schedule in the same c-step the designer needs to know if they use the same FU
  – Need to find efficient schedule by knowing the delays of the operations used and this is only done when both the FU’s and interconnections are defined
  – Need to know which operations can be done in parallel to know how many FU’s should be used
Scheduling

• To overcome this problem there have been many different solutions that either limit the number of FU’s, schedule and allocate at the same time or allocate first.

• When selecting a scheduling algorithm there are 2 general categories that most can fit into:
  – Transformational and iterative/constructive

• Transformational can be very computationally expensive but usually comes to a more optimal solution.
Scheduling

- Very common constructive algorithms are the ASAP and the ALAP configurations.
- For ASAP, each operation is taken from the graph and placed in the earliest control step possible.
- These approaches are very general and sometimes end up giving a longer than necessary solution.
Scheduling

• To solve this problem a solution such as list scheduling can be used where each operation is given a general criterion that is used to judge when it should be scheduled

• This criterion can be a number of different options depending on the solution required
Allocation

- 3 main goals
  - Map operations to FU's
  - Assign values to registers
  - Provide the interconnections from the registers to the FU's via buses or multiplexers

- Concentrate on optimizing some area while considering the user constraints
  - Interconnect length
  - Register, bus or multiplexer cost
  - Critical path delay

- Operations can use the same FU as long as the operations are mutually exclusive, applies to memory and interconnect as well
Allocation

- For allocation the methods fall into 2 categories being iterative/constructive and global
- The iterative technique will choose an operation, value or interconnect then make the assignment and repeat until the graph has been entirely covered
- The way that it allocates specific choices is decided by a set of rules from the designer
• Global allocation mainly uses graphs to formulate an optimal solution
• These graphs consist of the nodes which represent operations, values or connections that are connected using an arc if they are mutually exclusive to other nodes
Allocation

- Once the graph is created, cliques can be found.
- If the objective was to minimize the hardware used, then the algorithm would be to find the least amount of cliques.
- These algorithms are usually greedy and are very costly to find the optimal solution.
Problems

- HLS is capable of finding good solutions, but given specialized goals, manual optimization is necessary.
- A lack of interactivity exists where the designer has limited control on the outcome of the design process.
- Wide varieties of libraries need to be accepted by the synthesis tools.
- HLS will effect the layout due to architecture choices and the tools that can communicate to both levels are necessary.
HLS with Reconfig Datapath Components

- They propose a solution that utilizes runtime reconfigurable components during HLS.
- Using a resource constrained schedule they used the list algorithm to schedule their RTR.
- Using a priority list relating to the difference in ASAP and ALAP values they were able to schedule operations appropriately.
HLS with Reconfig Datapath Components

- It is possible that using this heuristic can cause the control step period to double in the worst case

INSERT READY OPS(V,PList_1, PList_2, ..., PList_m);
Cstep=0;
while ((PList_1 ≠ ∅) or ... or (PList_1 ≠ ∅)) do
  Cstep=Cstep+1;
  for k=1 to m do
    for funit=1 to N_k do
      if (PList_k ≠ ∅) then
        S_current=SC_OP(S_current, FIRST(PList_k, Cstep));
        PList_k=DELETE(PList_k, FIRST(PList_k));
      endif
    endfor
  endfor
insert READY OPS(V, PList_1, PList_2, ..., PList_m);
endwhile
HLS with Reconfig Datapath Components

2 multipliers, 1 Add

1 multiplier, 1 reconfig multiplier, 1 adder
HLS with Reconfig Datapath Components

- Results showed using 2 reconfigurable components they were able to achieve an average speed increase of 53% which can account for the worst case scenario when the algorithm will double the control step size.
Resources

Project Update

- Neural Network with back propagation
- In the process of programming a simple neural network and getting ready for profiling