COL719: Lab assignment 2
Development of High Level Synthesizer
(Controller stage)

Naman Jain (2020anz8848), SIT

Problem statement

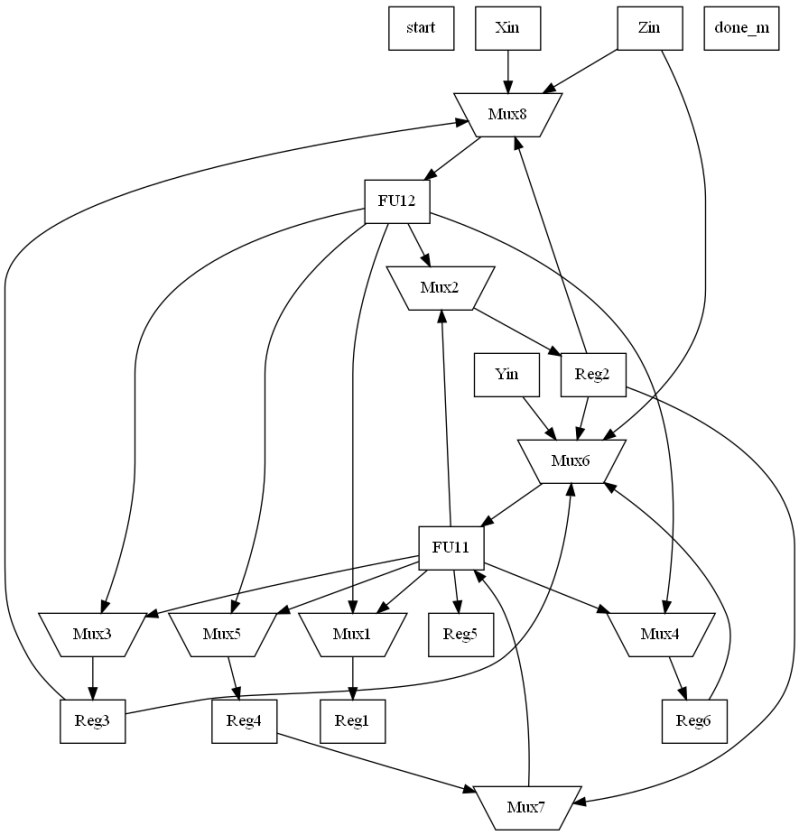
Aim for part 5 of the assignment is to synthesis FSM for the created datapath.

1 Assumption

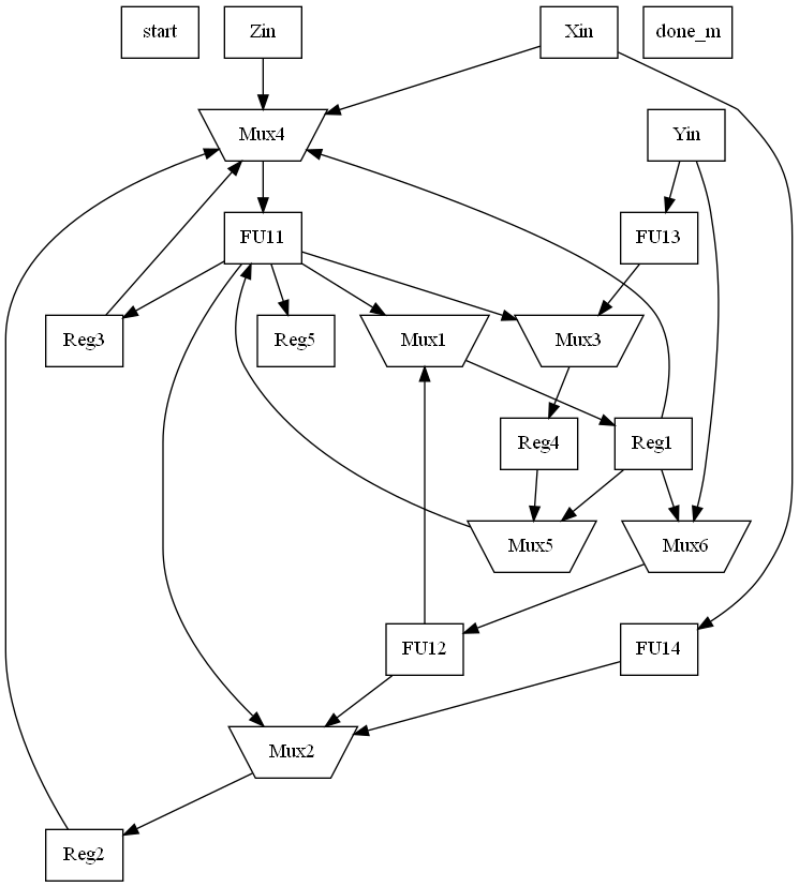
- Restricted HDL language as given in the document "HDL for Lab Assignment 2"
- Resource list will be given in format **k:([set of ops], bit width, delay)**, where **k** is the type of resource
- **No chaining of operations is done in scheduler and no multiple functional units for one operation,** that is supplied resource has sufficient bit width to accomodate largest operand.
- All operands and destination node will be declared before start of behavioural description. In few cases, not declaring destination node will not thrown any error.
- Graphviz module is only used to visualize the generated graph in PNG format
- Resulting timing schedule is stored as **T_list (minimum latency)** and **T_list_minr, num_r (minimum resource)**
- Global variable information is passed manually
- Resulting binding is stored as **res_binding (resource binding)** and **reg_binding (register binding)**
- Datapath information is stored in **Registers, Muxs and FUs** dictionary.
- Controller information is stored in **fsm_s and fsm_dot (visualiztion purpose)**

2 Results

In this part, improvements have been done to datapath generation stage. Figure 1 and 2 shows the improved datapath for GCD and modulo module. Figure 3, 4 and 5 show the generated FSM. Each block state contains the output signal value for multiplexer of registers & FUs as well as control for operation selection in a FU.



(a) min latency



(b) min resource

Figure 1: Generated datapath for GCD

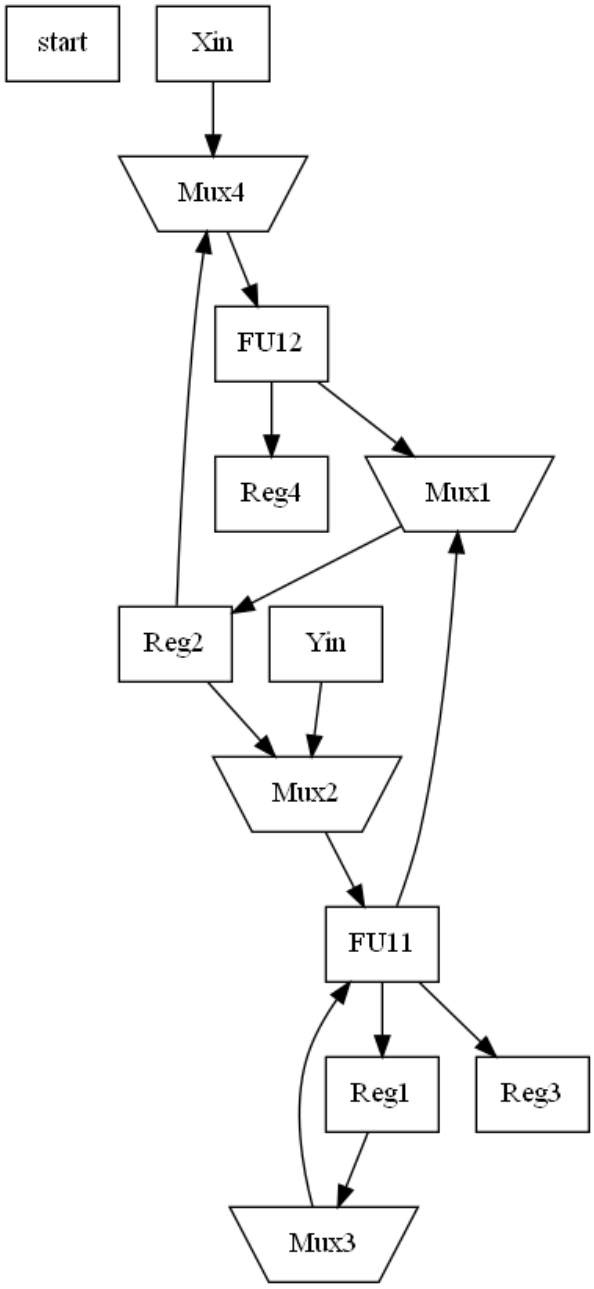
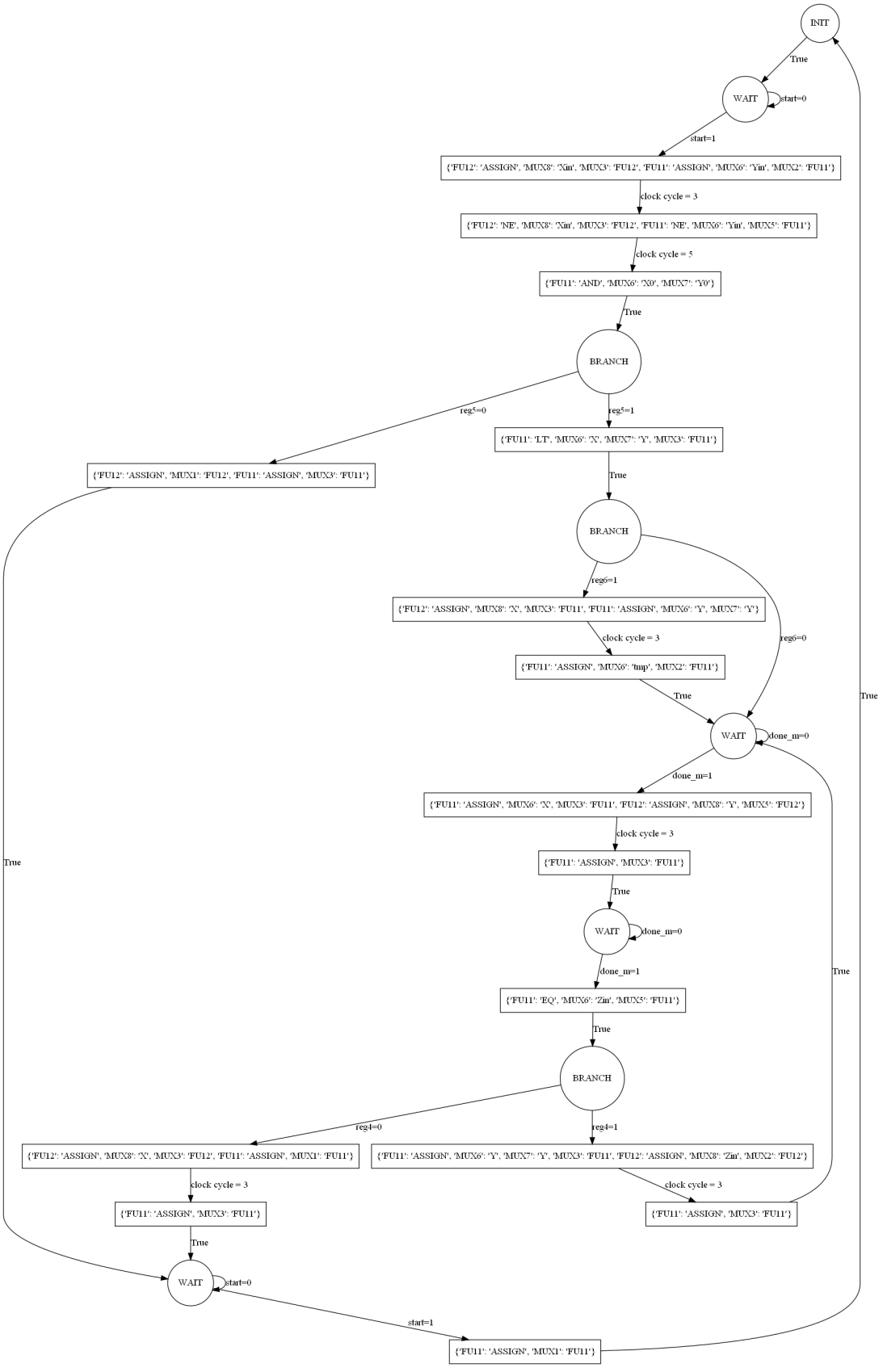
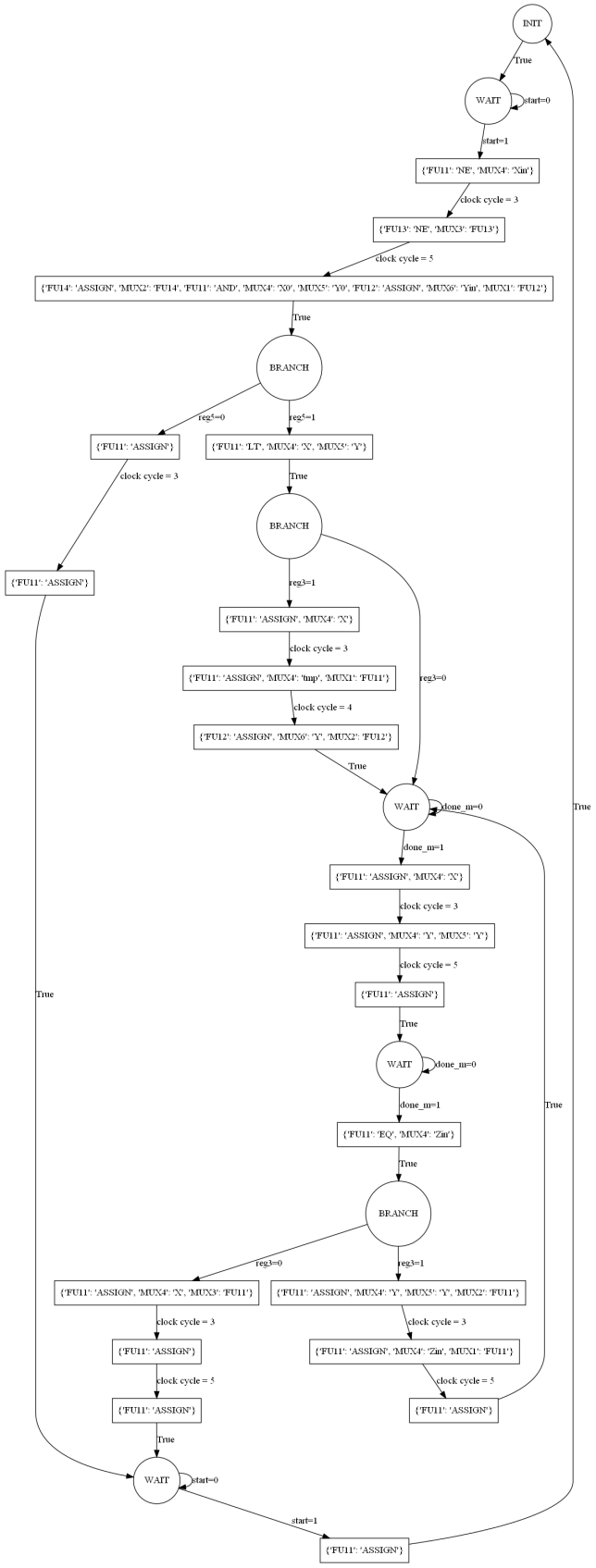


Figure 2: Generated datapath for modulo min latency



(a) min latency

Figure 3: Generated FSM controller for GCD



(a) min resource

Figure 4: Generated FSM controller for GCD

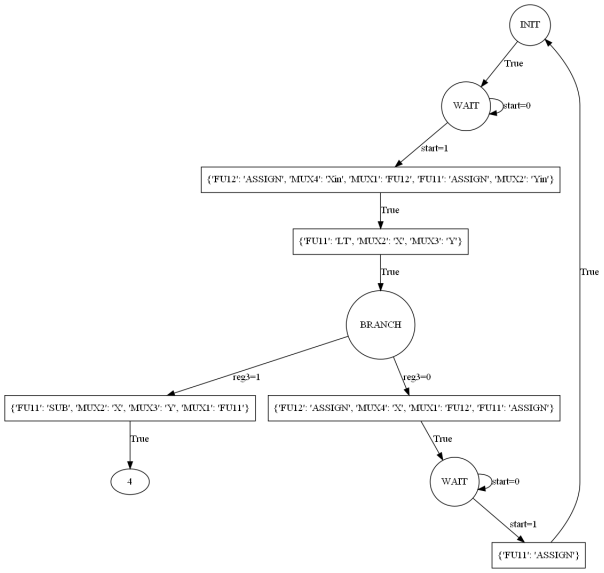


Figure 5: Generated FSM controller for modulo min latency

3 Instructions to run python code

- To run the python file use ***python front_ end.py***
  - When prompted enter the filename with full path (eg: ***Enter the HDL filename:C:/Users/nsahu/Documents/S II/COL719/Lab_ Assignment_ 2/gcd_ HDL***)
  - Then choose option to generate CFG ot CDFG (eg: ***Enter option for CDFG visualization (1)BLOCK expanded in DFG or (2)BLOCK not expanded:1***)
  - Output graph image in PNG format will stored in the given path with suffix either '_cfg' or '_cdfg'. Graph will also be shown in the console.
  - Choose the scheduling options (eg ***Enter option for type of schedule (1) minimum latency (2) minimum resource:2)***
  - For minimum resources need to enter latency for each block node (eg ***Enter option latency constraint for BLOCK:2 (if no latency then enter 0):10)***
  - Then to account for global variables prompt will be given as follows - **(For variable X == start time:3 and end time:3 Is this a Global variable (Y/N):Y)**
- Library modules used are graphviz and copy. **Testing is done using python** 3.8.5**.**
- To install graphviz - ***pip install graphviz*** and install it into Windows [1]/Linux machine (***sudo apt install graphviz***)

References

[1] https://graphviz.org/download/