ECE 274 - Digital Logic

Lecture 15

- Lecture 15 – RTL Design
  - Four Step RTL Design Method
  - RTL Examples

Step 2: Create a Datapath

- Datapath must
  - Implement data storage
  - Implement data computations
- Look at high-level state machine, do three substeps
  - (a) Make data inputs/outputs be datapath inputs/outputs
  - (b) Instantiate declared registers into the datapath (also instantiate a register for each data output)
  - (c) Examine every state and transition, and instantiate datapath components and connections to implement any data computations

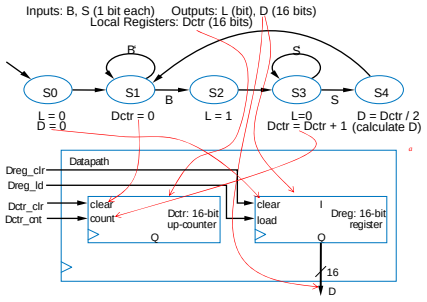
Instantiate: to introduce a new component into a design.

Step 2 Example: Laser-Based Distance Measurer

(a) Make data inputs/outputs be datapath inputs/outputs

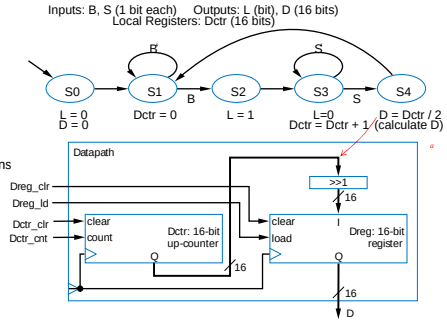
(b) Instantiate declared registers into the datapath (also instantiate a register for each data output)

(c) Examine every state and transition, and instantiate datapath components and connections to implement any data computations

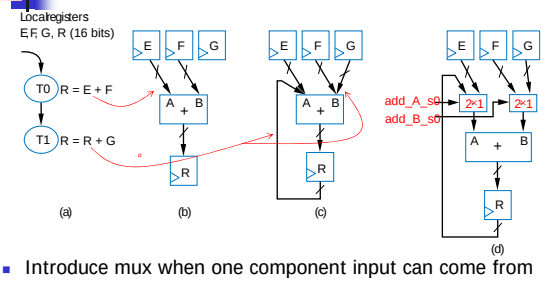


Step 2 Example: Laser-Based Distance Measurer

(c) (continued) Examine every state and transition, and instantiate datapath components and connections to implement any data computations

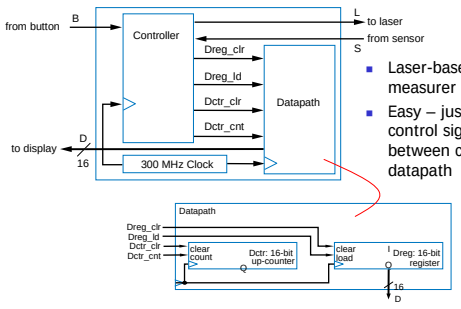


Step 2 Example Showing Mux Use



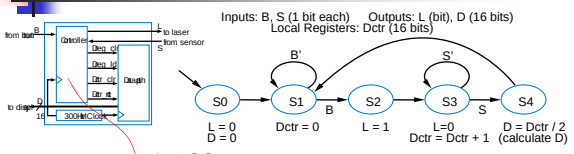
- Introduce mux when one component input can come from more than one source

Step 3: Connecting the Datapath to a Controller

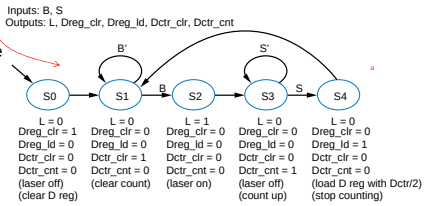


- Laser-based distance measurer example
- Easy – just connect all control signals between controller and datapath

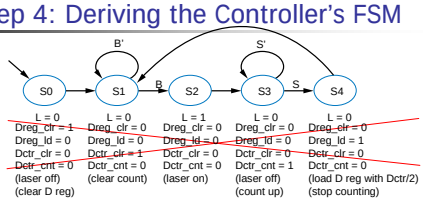
Step 4: Deriving the Controller's FSM



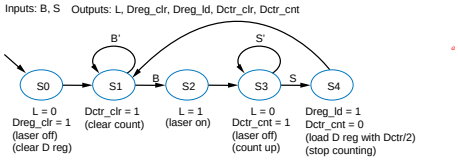
- FSM has same structure as high-level state machine
  - Inputs/outputs all bits now
  - Replace operations by bit operations using datapath



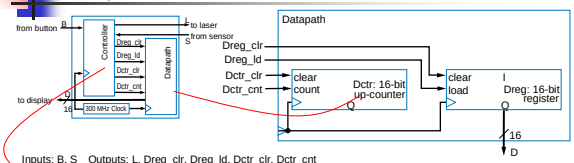
Step 4: Deriving the Controller's FSM

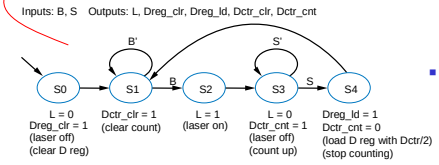


- Using shorthand of outputs not assigned implicitly assigned 0



Step 4

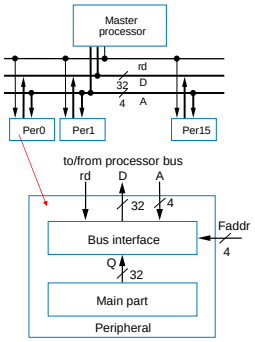




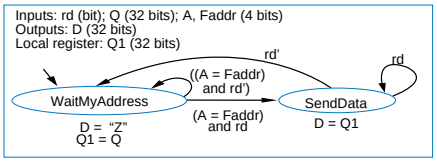
- Implement FSM as state register and logic (Ch3) to complete the design

RTL Design Examples and Issues

- We'll use several more examples to illustrate RTL design
- Example: **Bus interface**
  - Master processor can read register from any peripheral
    - Each register has unique 4-bit address
    - Assume 1 register/periph.
  - Sets *rd=1*, *A=address*
  - Appropriate peripheral places register data on 32-bit *D* lines
    - Periph's address provided on *Faddr* inputs (maybe from DIP switches, or another register)

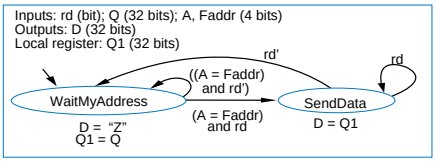


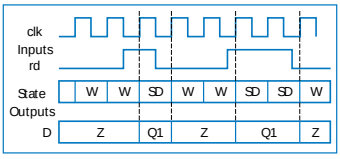
RTL Example: Bus Interface



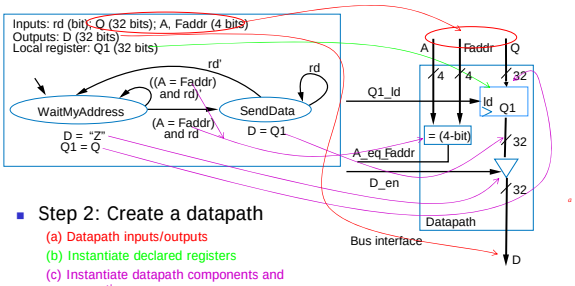
- Step 1: Create high-level state machine
  - State **WaitMyAddress**
    - Output "nothing" ("Z") on *D*, store peripheral's register value *Q* into local register *Q1*
    - Wait until this peripheral's address is seen (*A=Faddr*) and *rd=1*
  - State **SendData**
    - Output *Q1* onto *D*, wait for *rd=0* (meaning main processor is done reading the *D* lines)

RTL Example: Bus Interface



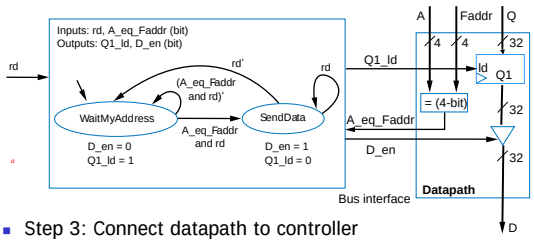


RTL Example: Bus Interface



- Step 2: Create a datapath
  - (a) Datapath inputs/outputs
  - (b) Instantiate declared registers
  - (c) Instantiate datapath components and connections

RTL Example: Bus Interface



- Step 3: Connect datapath to controller
- Step 4: Derive controller's FSM

Design Challenge

- Design Challenge
  - Create a high-level state machine that describes the following system behavior.
    - The system has an 8-bit input A, a single-bit input d, and a 32-bit output S.
    - On every clock cycle, if d=1, the system should add A to a running sum and output that sum on S.
    - If d=0, the system should instead subtract. Ignore issues of overflow and underflow. Don't forget to include an initialization state.
    - Hint: declare and use an internal register to keep the sum.
    - Add a 1-bit input rst to the system. When rst=1, the system should clear its sum back to 0.
  - Using the RTL design method, convert the high-level state machine to a datapath and controller
  - Due:
    - Next Lecture (Wednesday, October 26)
  - Extra Credit *(Homework)*
    - 2 points