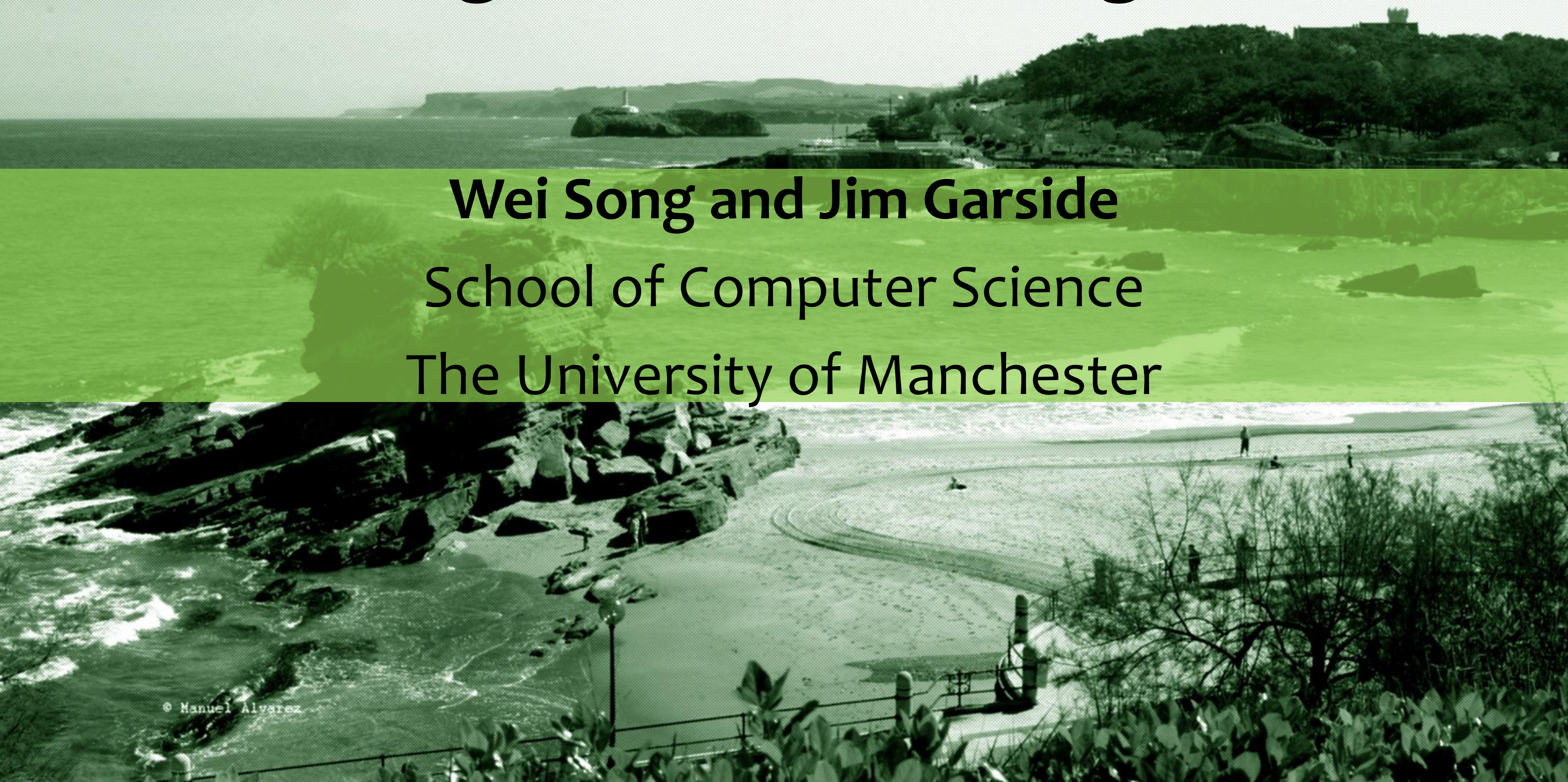
MANCHESTER
1824
The University of Manchester

EPSRC
Pioneering research and skills

Automatic Controller Detection for Large Scale RTL Designs

Wei Song and Jim Garside
School of Computer Science
The University of Manchester



© Manuel Alvarez

Motivation

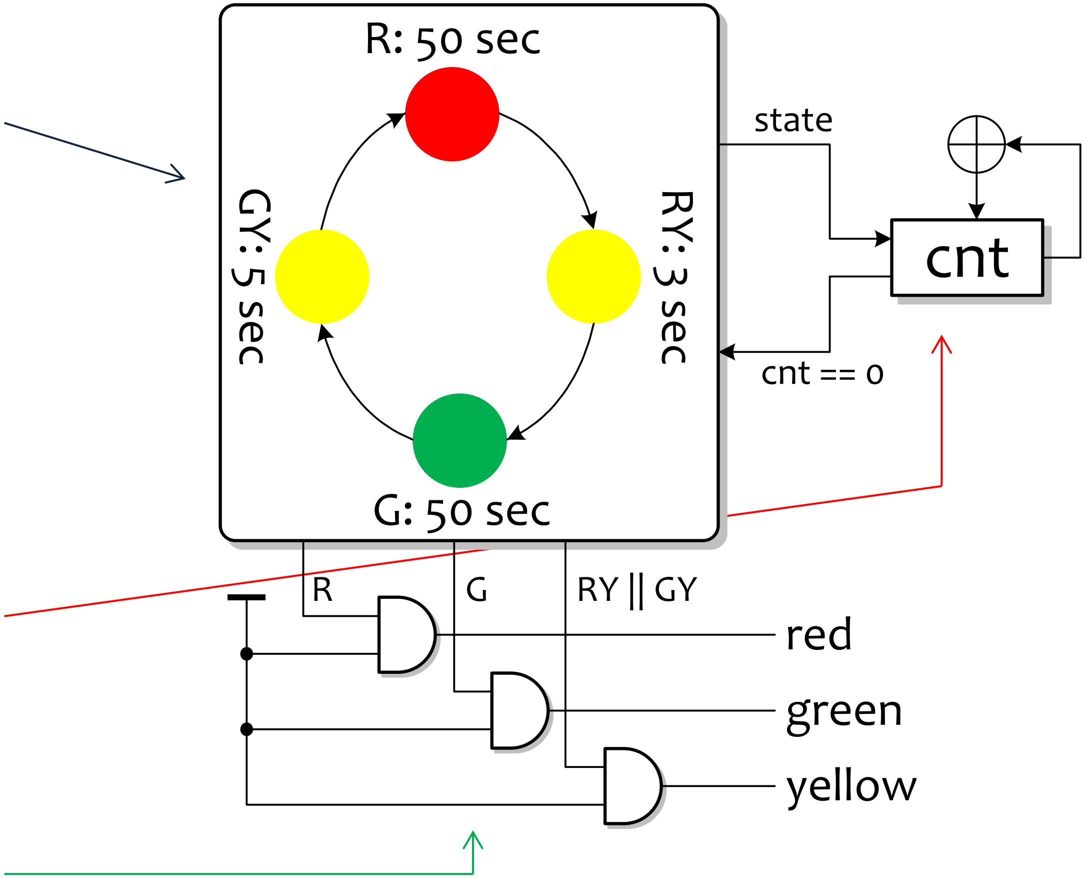
- **Problem**
  - Matching of coding styles (DC: Design Compiler).
  - Only Finite State Machines (FSMs)
- **Target**
  - An algorithm that can detect **ALL** controllers.
- **Solution**
  - Signal-level data flow graph
  - Common pattern of controllers

Traffic Light Controller

```
always @(posedge clk or negedge rstn)
  if(~rstn)
    state <= R;
  else
    state <= state_nxt;

always @(state or cnt) // next state
  if(cnt == 0)
    case(state)
      R:  state_nxt = YR;
      YR: state_nxt = G;
      G:  state_nxt = YG;
      default:
        state_nxt = R;
    endcase // case (state)
  else
    state_nxt = state;
```

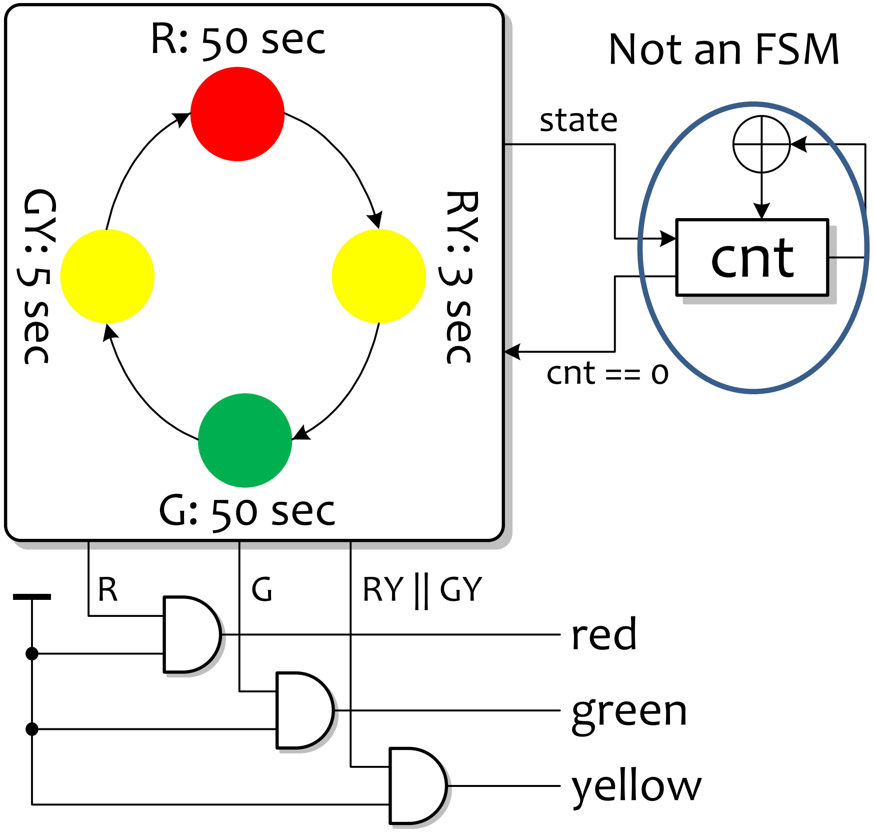
```
always @(posedge clk or negedge rstn)
  if(~rstn)
    cnt <= 0;
  else if(cnt == 0)
    case(state)
      R:  cnt <= 2;
      YR: cnt <= 49;
      G:  cnt <= 4;
      default:
        cnt <= 49;
    endcase // case (state)
  else
    cnt <= cnt - 1;
```

```
assign red = state == R ? 1 : 0;
assign green = state == G ? 1 : 0;
assign yellow =
   (state == YR || state == YG) ? 1 : 0;
```



DC: Matching of Coding Styles

- Never be assigned by a value other than predefined states.
- Can be used in only == and != statements
- Never be used as a port

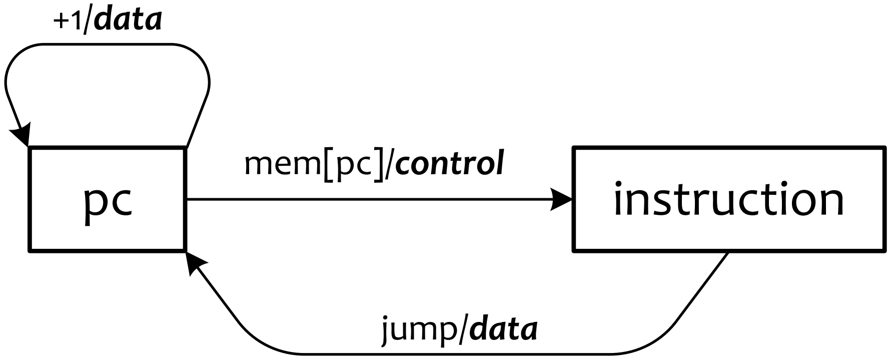


(Recognize all the counters used as controllers.)

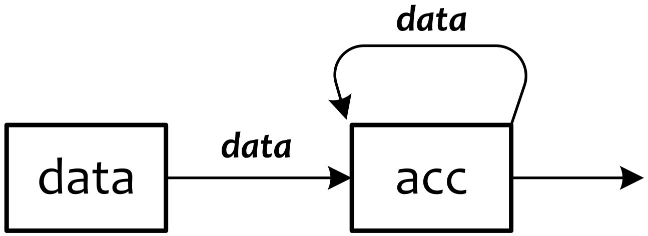
Criteria of Controllers

- **Requirement 1**
  - – A self-loop.
- **Requirement 2**
  - – A controlling output.
- **Requirement 3**
  - – No data input other than itself or constants.

Examples of Counters

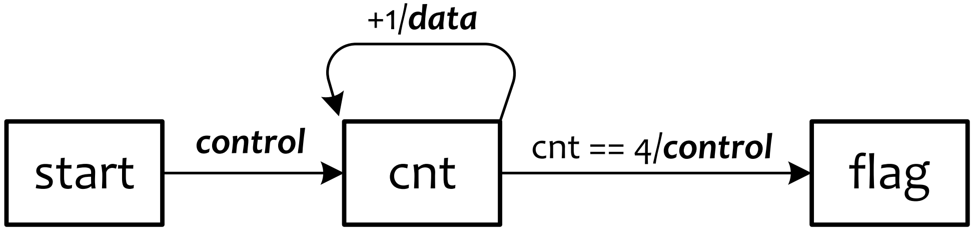


Not a controller.
Does not meet
Requirement 3.



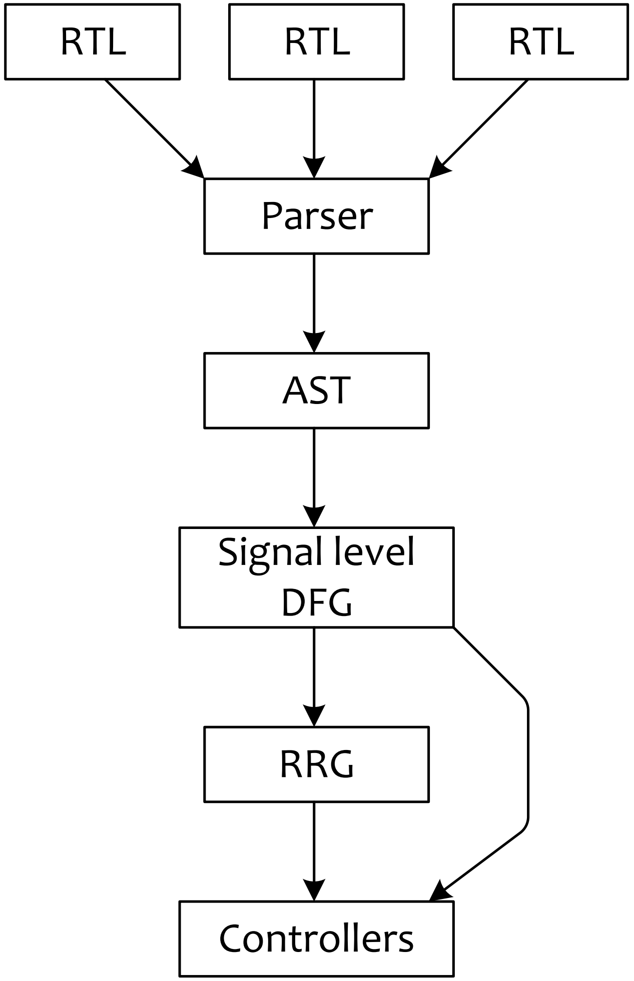
acc’ = acc + data

Not a controller.
Does not meet
Req. 2 & 3.



A controller.

Detection Flow



Multi-file Verilog RTL designs

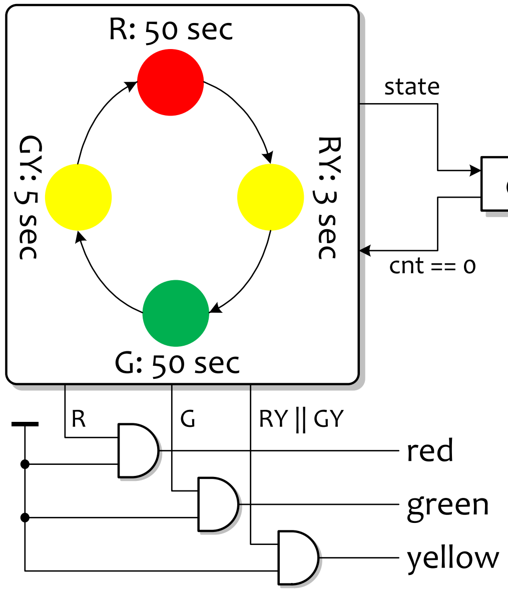
Hierarchical internal abstract sematic tree

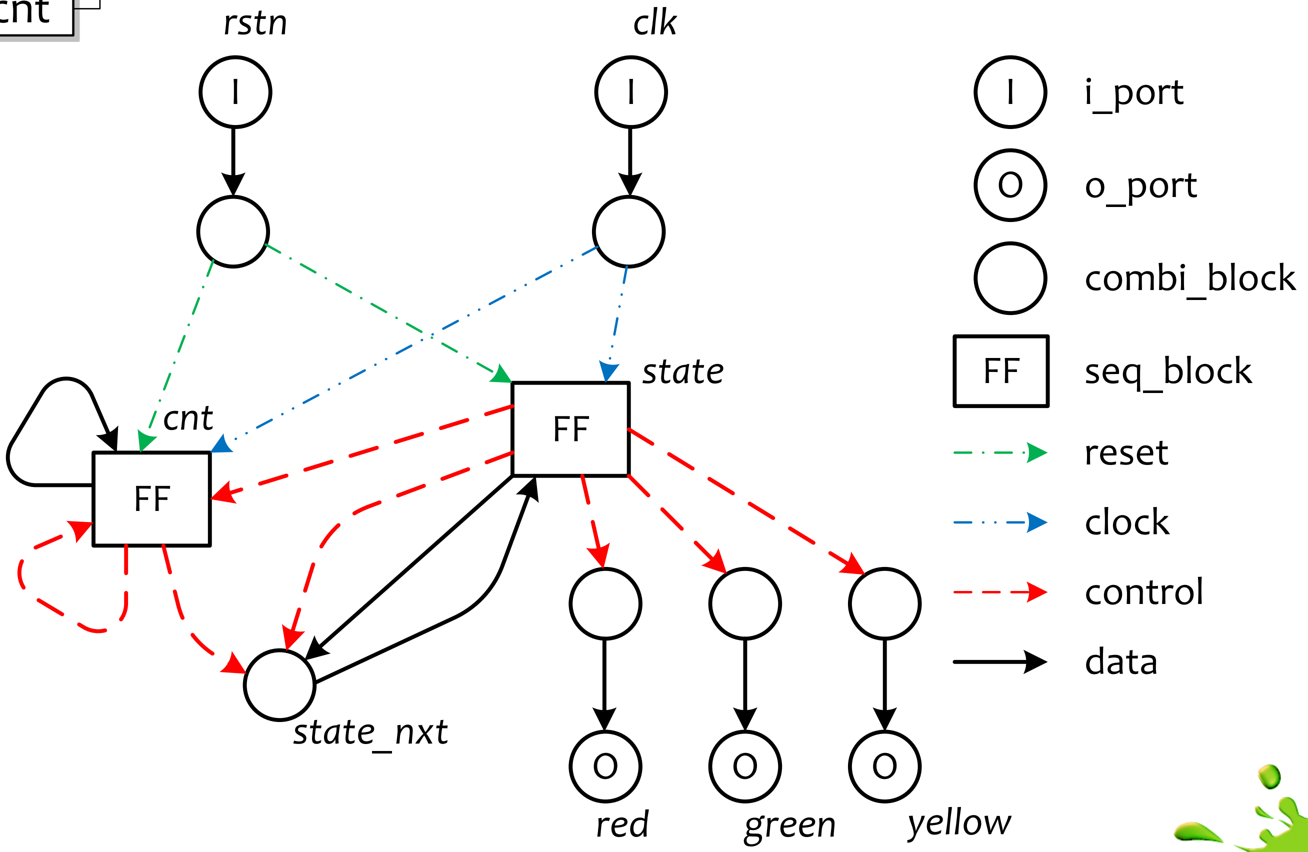
Hierarchical signal-level data flow graphs (DFGs) (Connections between **signals**)

Register relation graph
(Connections between **Flip-Flops**)

Controller detection and report

Signal-Level DFG

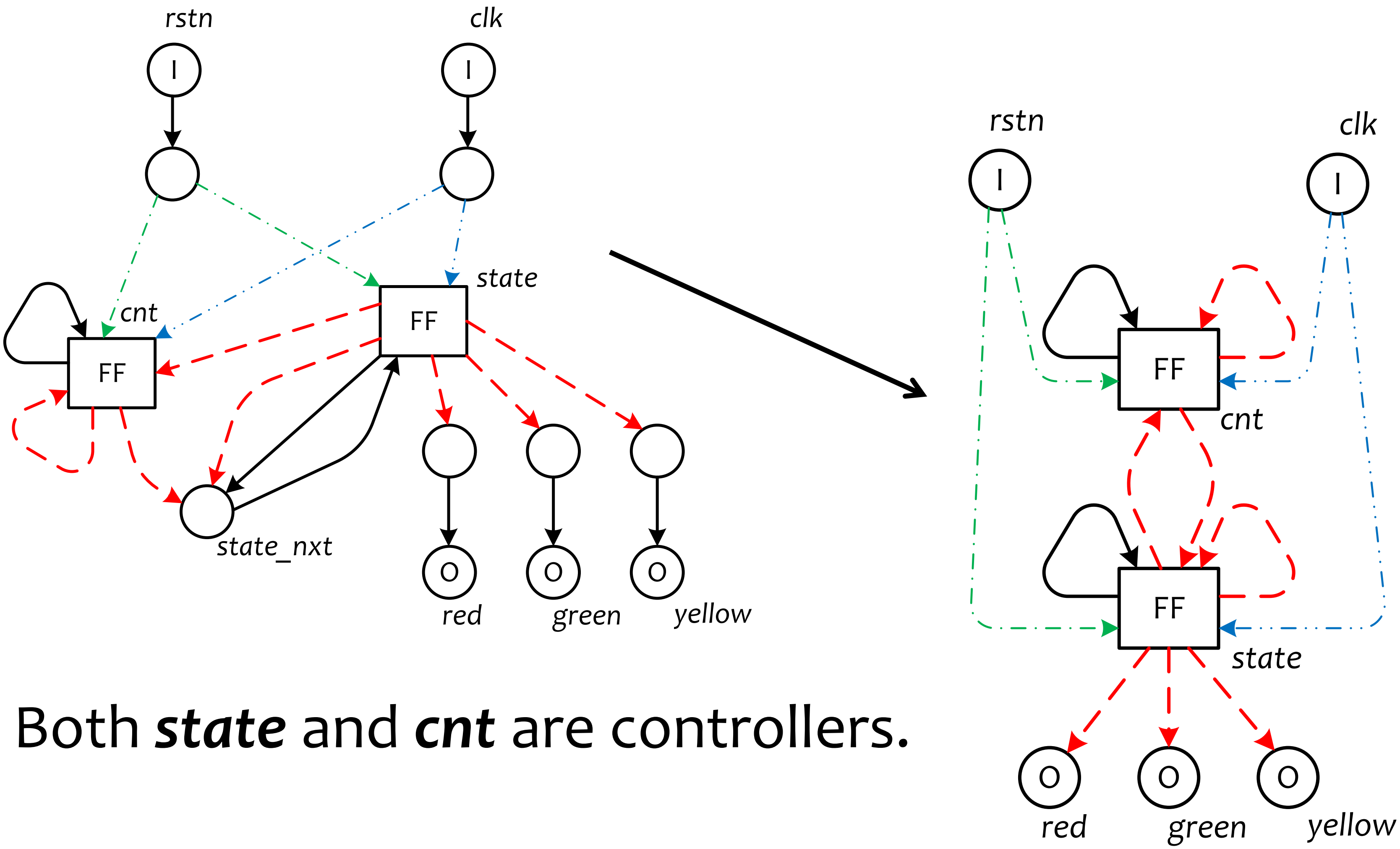




Type Estimation

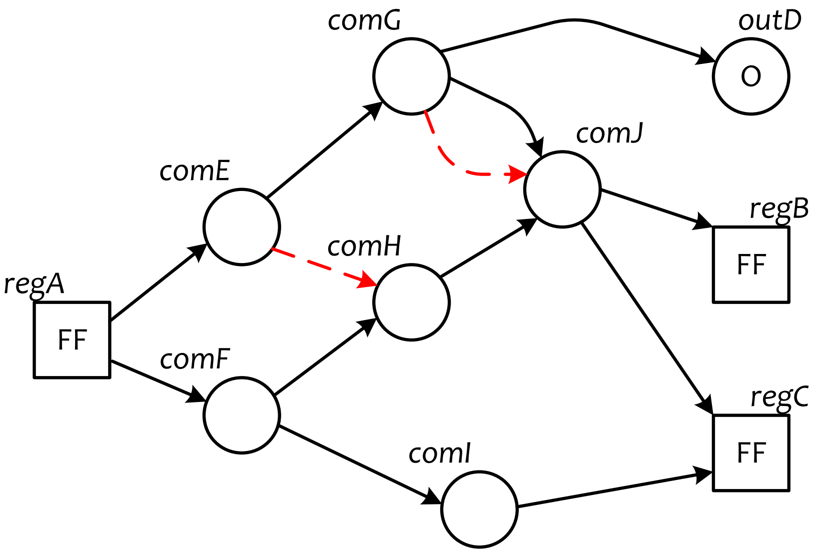
```
if (a == 0)
case (state)
dout = mem[addr];
sum = da * time + extra;

dout = b > 3 ? da : db;
if(b > 3 ? da : db)    // control has a higher priority
```

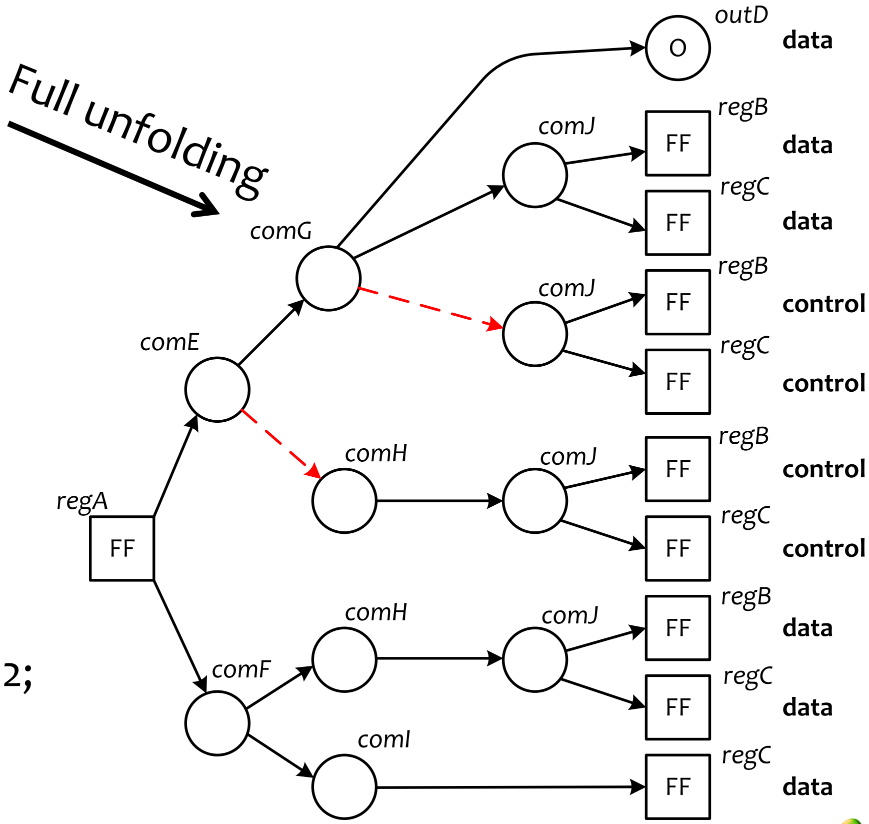
Register Relation Graph (RRG)



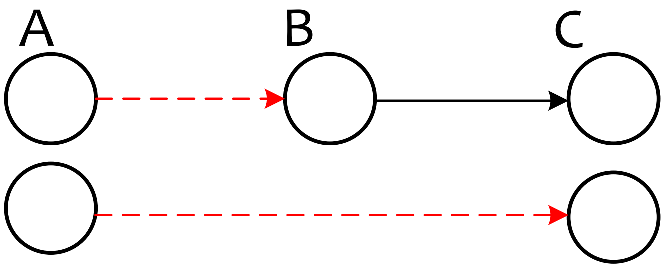
Both ***state*** and ***cnt*** are controllers.

Dynamic Programming





Partial path reduction



B = **A**==0 ? 5 : 12;
C = **B** + 10;

C = (**A**==0 ? 5 : 12) + 10;

Test Cases

Design Name	Description	Reason to choose	No. of Regs
OR1200	A 32-bit 5-stage OpenRISC microprocessor	Data path controlled PC Scan chain Combinational forward loop	124
Reed-Solomon	A claimed industrial standard Reed-Solomon decoder IP	A not so well-written design Single block FSMs with irrelevant signal assignments	325
H.264/AVC	A 196K gate H.264/AVC baseline decoder	A well-written and large-scale design	855

Test Results

Name	Time	Reported	Verified	FSM	Counter	Flag	Error	Rate
OR1200	1s	19	17	7	5	5	2	89%
Reed-Solomon	2.0s	56	54	6	36	12	2	96%
H.264/AVC	7.1s	55	49	13	30	6	6	89%

Environment: Intel Core^{TM} 2 Duo 3.00 GHz PC with 2GB memory

All FSMs are detected with a small number of false errors.

Limitations:
Combinational loops, separated assignment (a[0]=b; a[1]=c;),
Rom-style tables, etc.

Conclusion

- A pattern-matching algorithm has been proposed to detect all controllers including FSM, controlling counters and flags.
- Advantages:
  - Pattern matching
  - No restriction on coding styles
  - Recognize controlling counters
  - Automatic type estimation

Thanks for your listening!
Any questions?