MANCHESTER
1824
The University of Manchester

Utilizing signal-level data flow graph in analysing large-scale RTL circuits

Supported by the GAELS project:
Globally Asynchronous Elastic Logic Synthesis

Wei Song

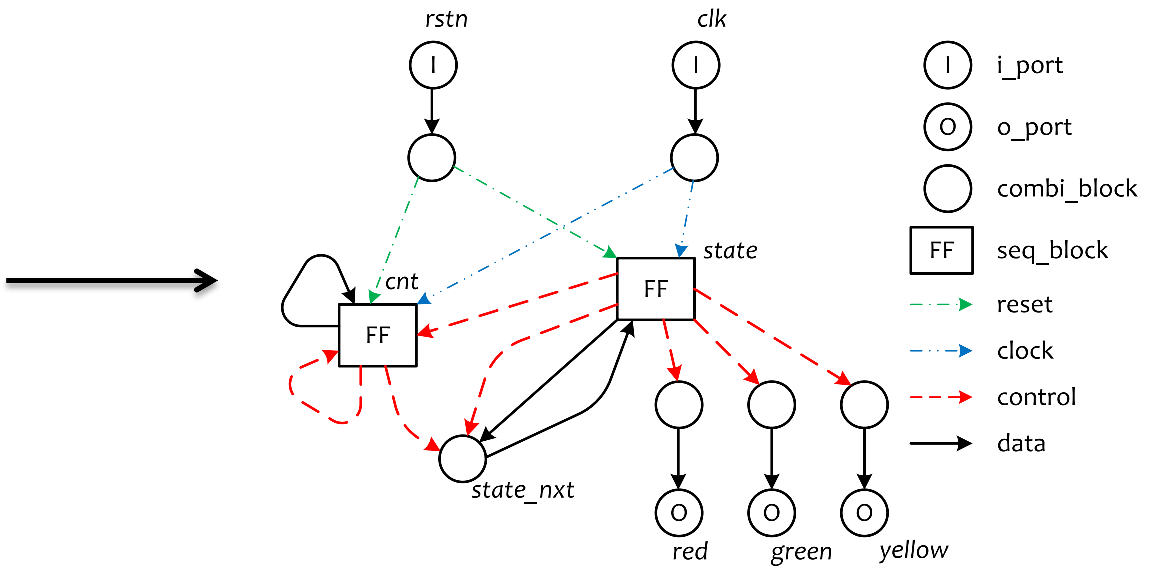
13/05/2014

Advanced Processor Technologies Group
The School of Computer Science

SDFG

- SDFG --- Signal-level Data Flow Graph

```
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;
```



Controller recognition
Data-path recognition

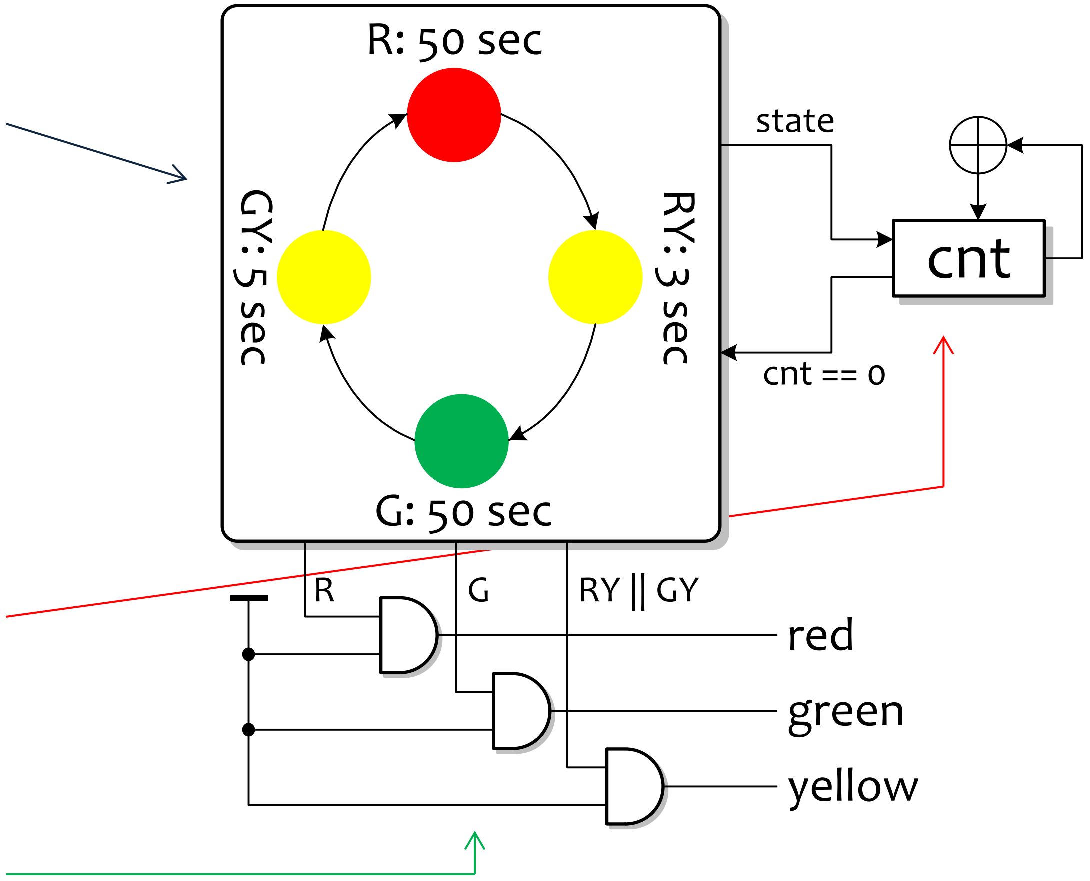
Index

A Traffic Light Controller

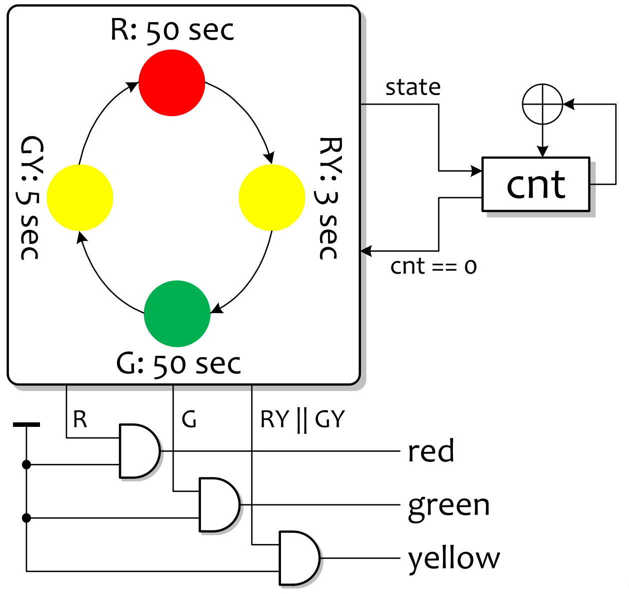
```
always @(posedge clk or negedge rstn)
  if(~rstn)
    state <= R;
  else
    state <= state_nxt;

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;
```

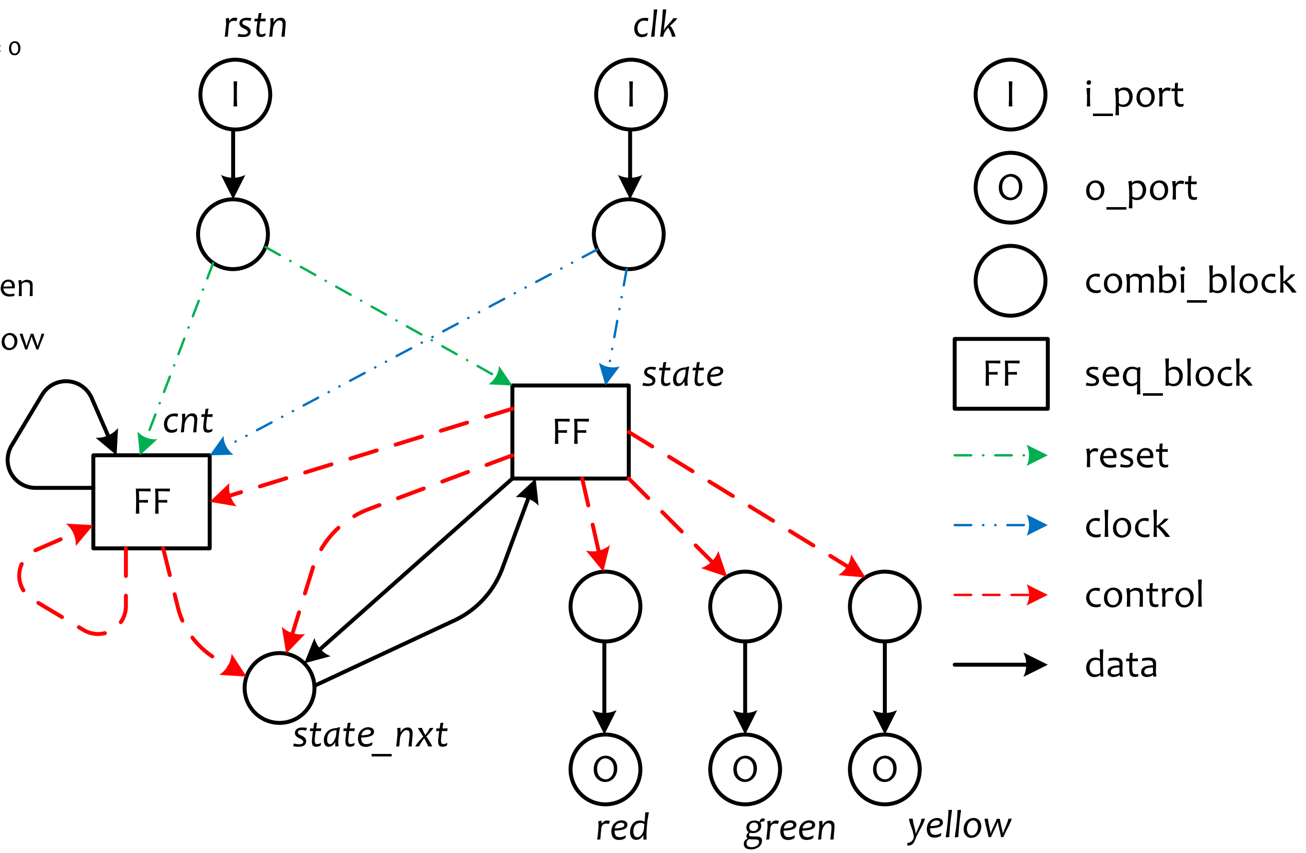
```
always @(posedge clk or negedge rstn)
  if(~rstn)
    cnt <= 0;
  else if(cnt == 0)
    case(state)
      R:  cnt <= 2;
      YR: cnt <= 49;
      G:  cnt <= 4;
      default:
        cnt <= 49;
    endcase // case (state)
  else
    cnt <= cnt - 1;
```

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



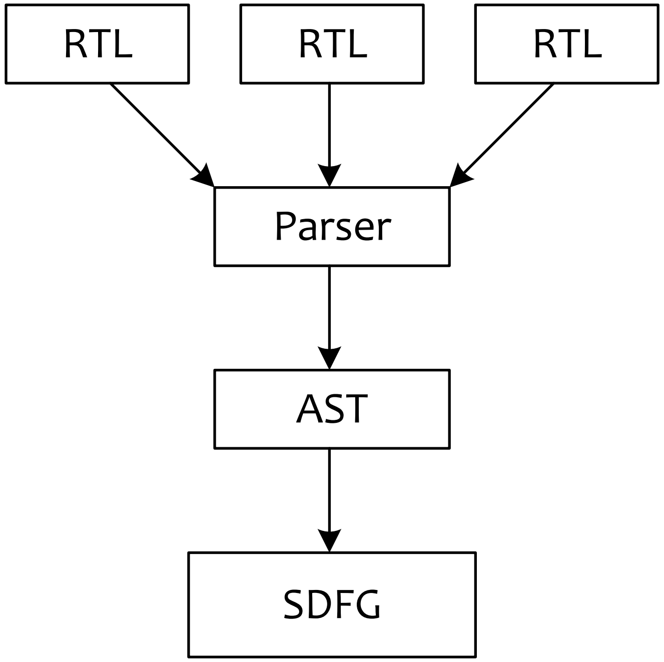
Signal-Level Data Flow Graph





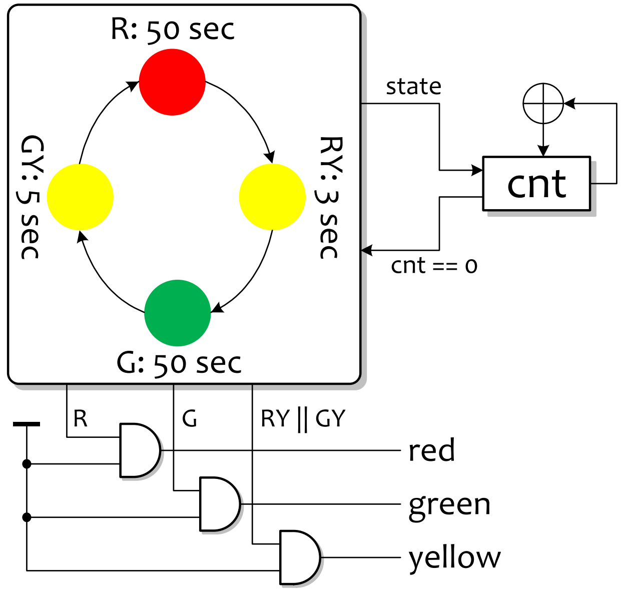
Relation between signals

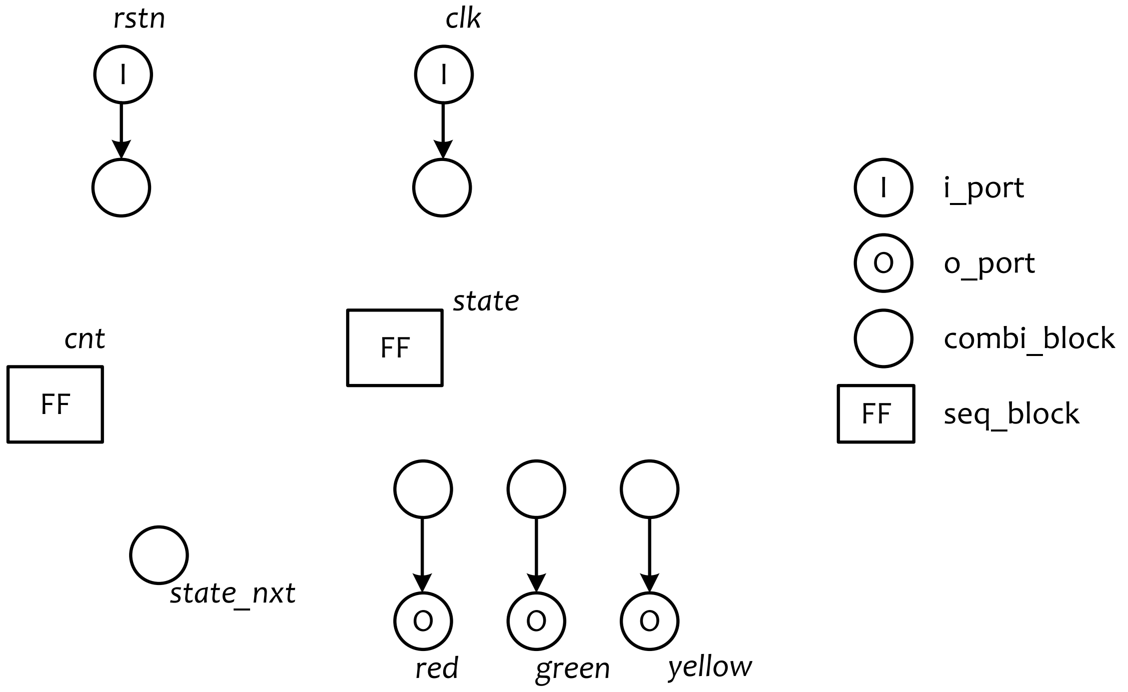
Tool Flow



1. Convert a multi-file Verilog RTL to abstract syntax trees (ASTs) using a Verilog Parser (Bison + Flex).
2. For each Module, generate an SDFG graph with all signals drawn as nodes with types (FF, combi, port, module).
3. Connect nodes by estimating the relations between signals (arc type estimation).

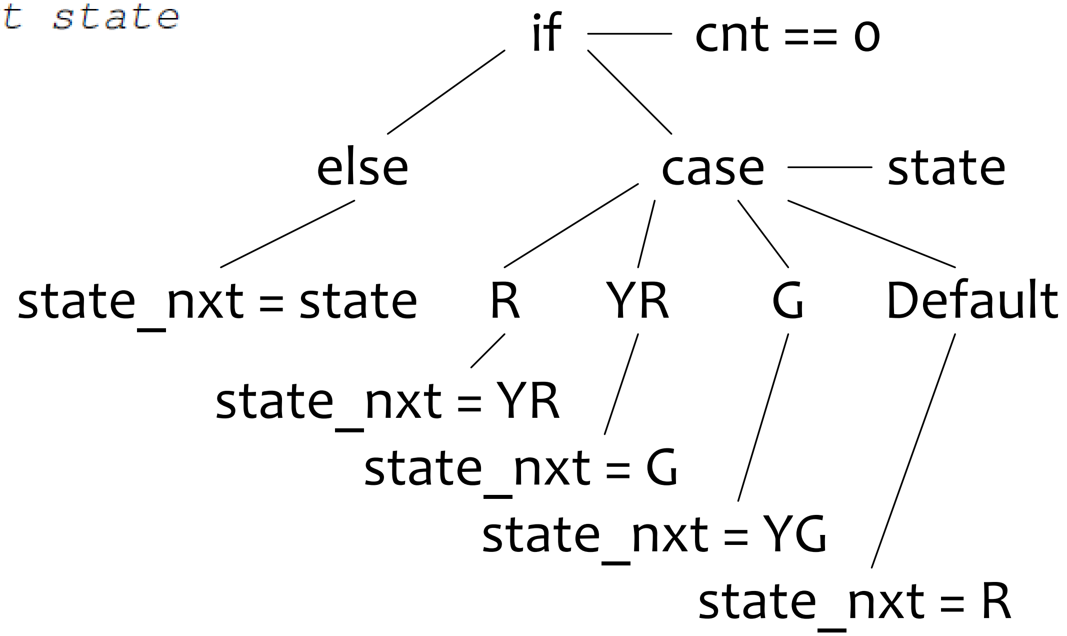
Node Insertion



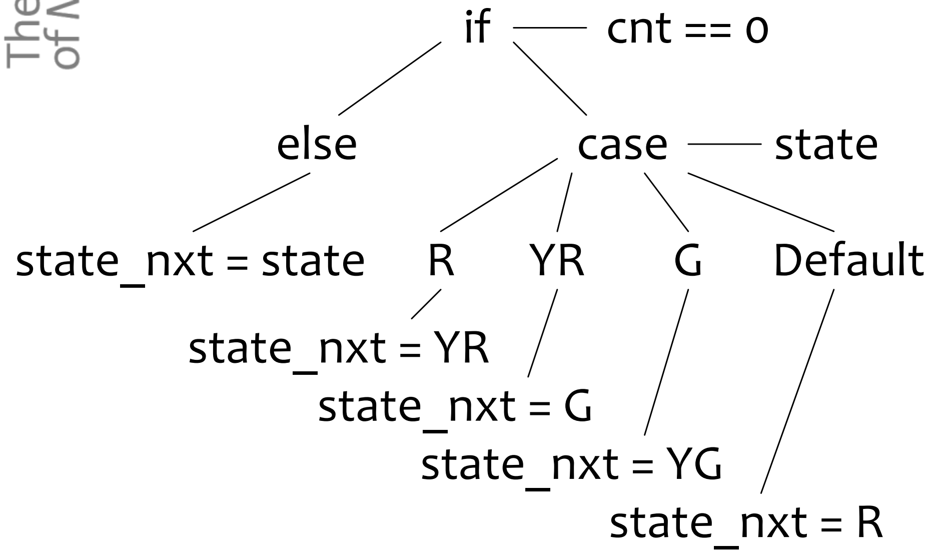


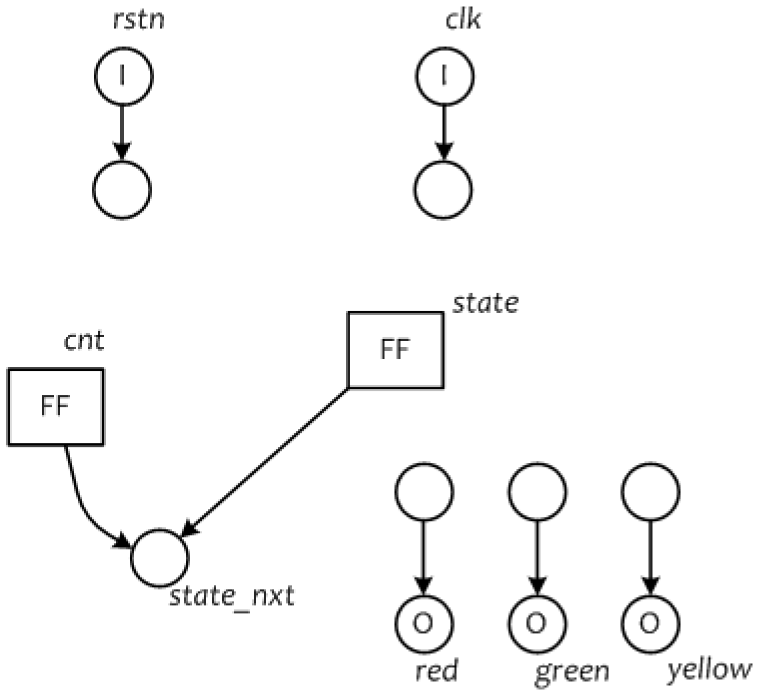
Abstract Syntax Tree (AST)

```
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;
```

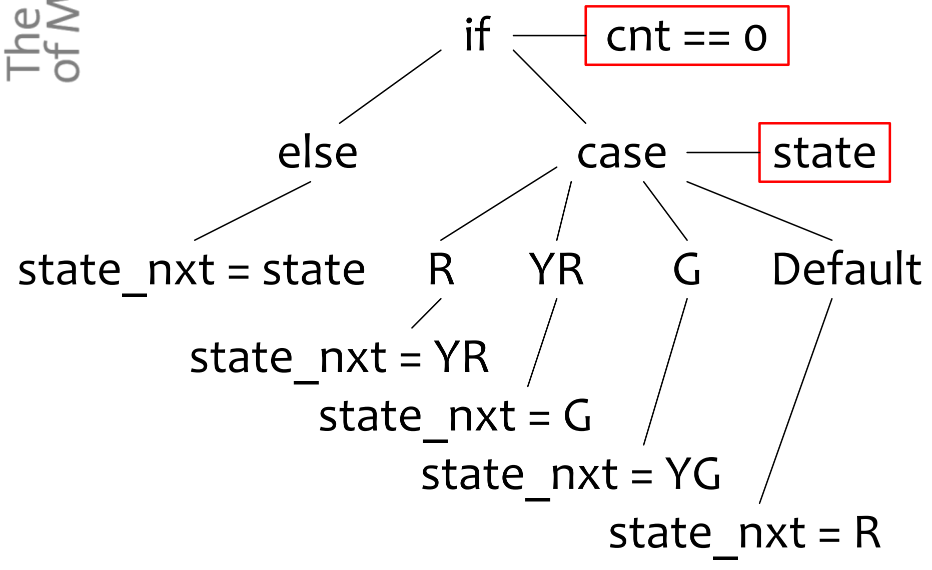


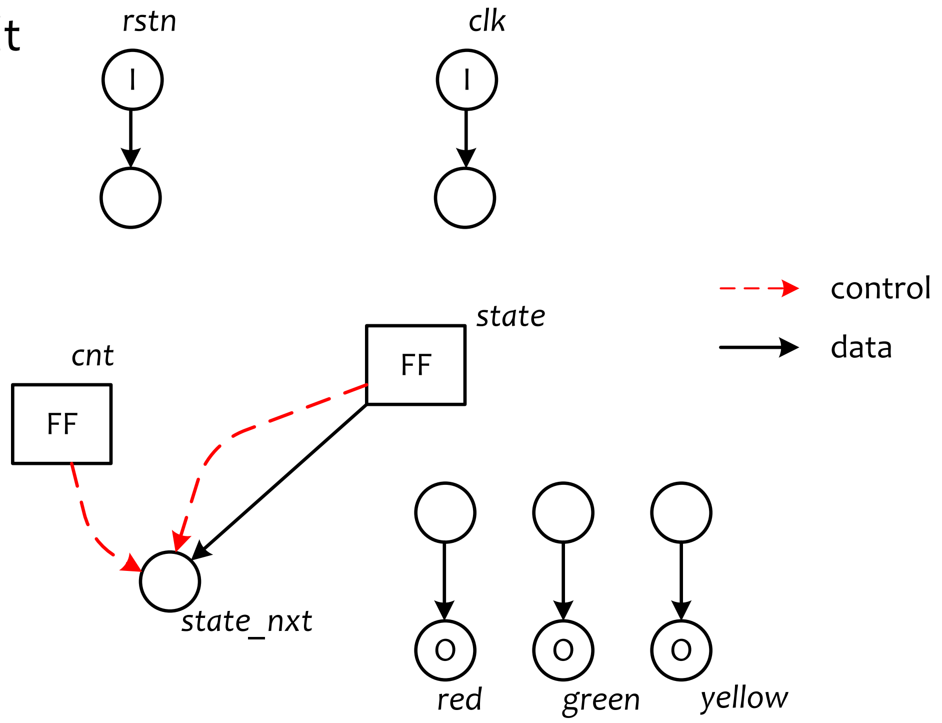
Arc Type Estimation



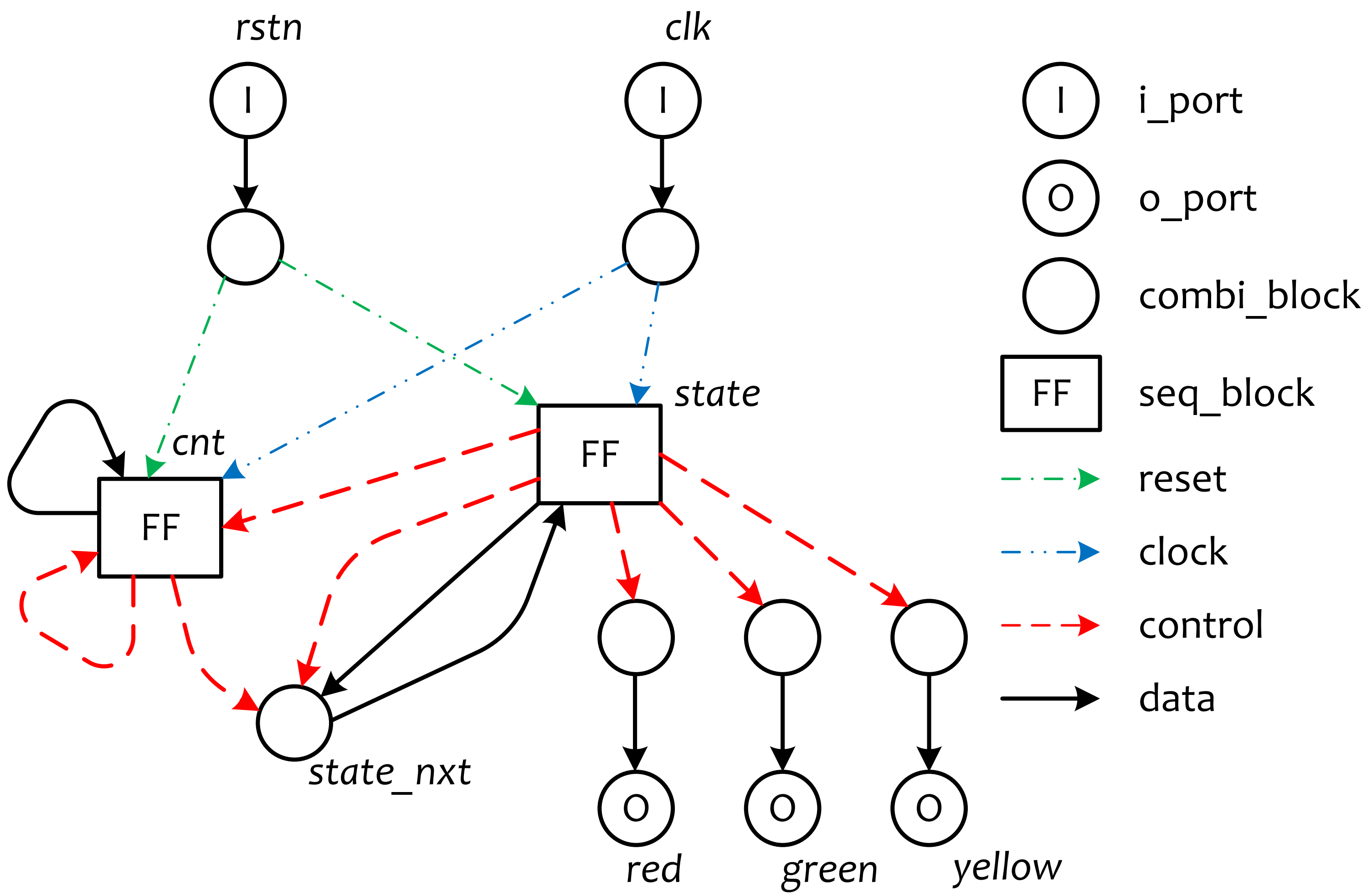


Arc Type Estimation





Arc Type Estimation



Available Types

- DATA
  - DDP: self loop
    - state = **state**;
  - CAL: calculation
    - sum = **a** + **b**;
  - ASS: assign
    - dout = **din**;
  - DAT: other
    - dout = **din** & **enable**;
- CONTROL
  - CMP: comparison
    - If(a > b)
  - EQU: equal
    - If(state == S_BEGIN)
  - LOG: logic calculation
    - If(valid && en)
  - ADR: address
    - dout = mem[adr]
  - CTL: other
    - If(a & b)

MANCHESTER 1824

The University of Manchester

Case 1: Controller Extraction

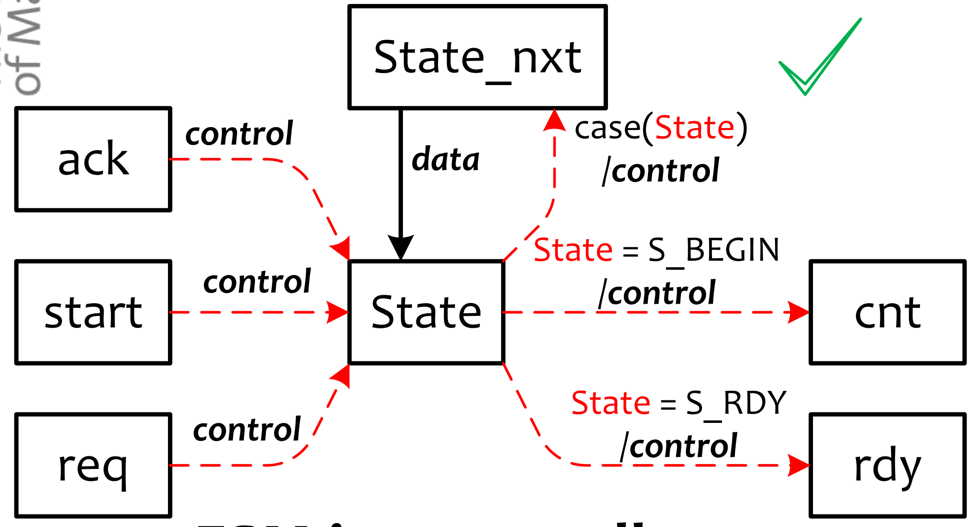
- Traditional: Synopsys Design Compiler
  - Coding style, only FSM
- SDFG:
  - Pattern matching in SDFG and RRG (register relation graph)
  - FSM, counter, flag

- Wei Song and Jim Garside. **Automatic controller detection for large scale RTL designs.** In *Proc. of EUROMICRO Conference on Digital System Design (DSD)*, Santander, Spain, pp. 884-851, September 2013

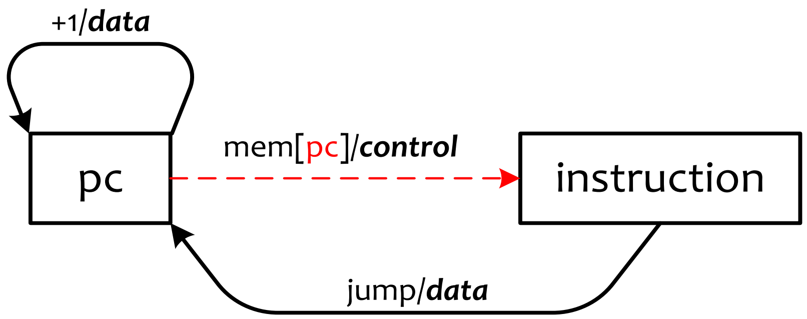
Define a Controller

- Controller: Finite State Machine
  - A register which stores states
  - Control the behaviour of other signals
  - A finite state space

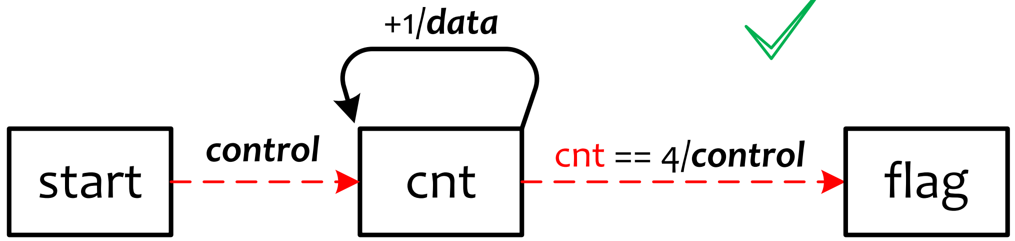
Examples



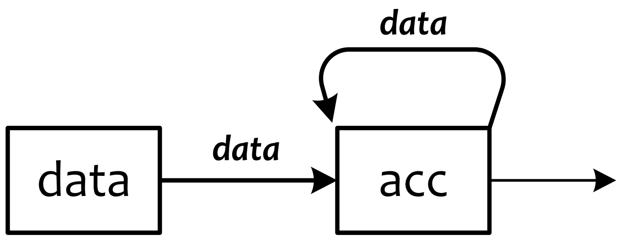
FSM is a controller



PC is NOT a controller



Counter can be a controller

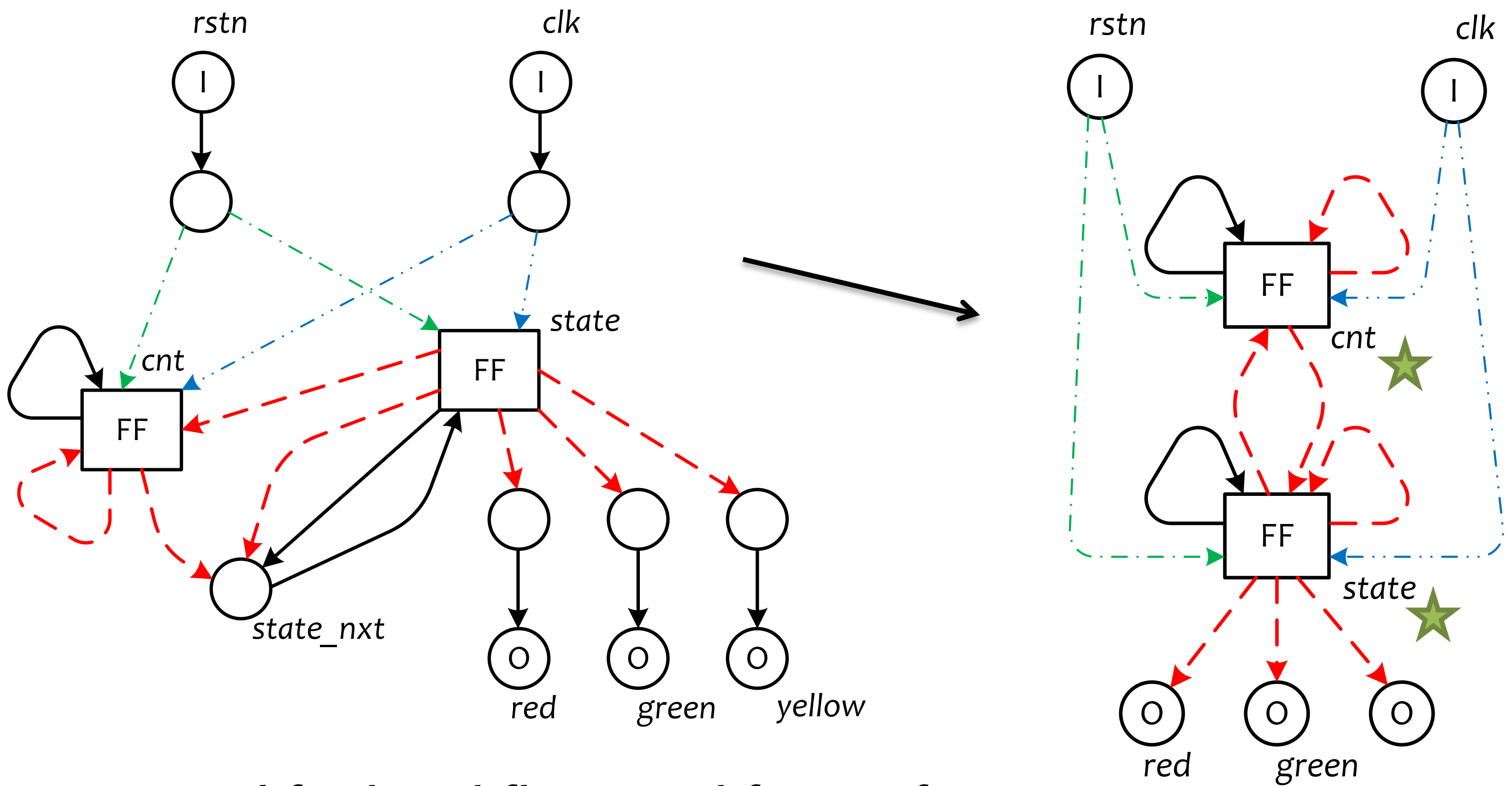


acc’ = acc + data

Pattern Criteria

- **Definition 1**
  - A controller is a register which has at least one self-loop not going through higher hierarchy.
    (*A register which stores states*)
- **Definition 2**
  - A controller is a register which has at least one controlling output path.
    (*A controlling register*)
- **Definition 3**
  - The inputs for a controller must be constant or from itself.
    (*Finite state space*)

Register Relation Graph (RRG)

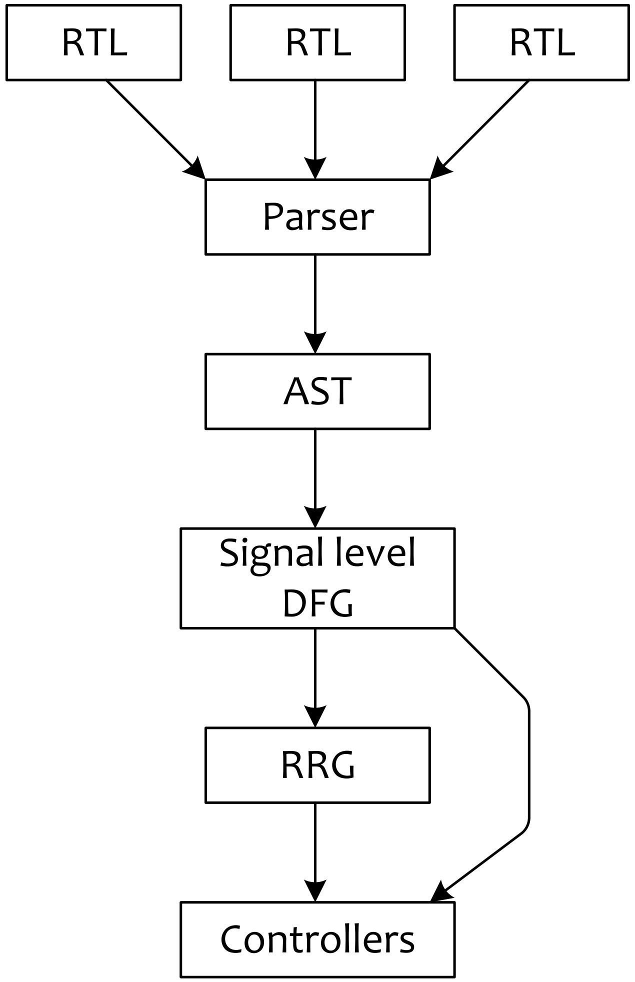


RRG is a simplified and flattened form of SDFG.

It has only registers and ports.

The pattern criteria can be directly applied on RRG.

Tool Flow



Multi-file Verilog RTL designs

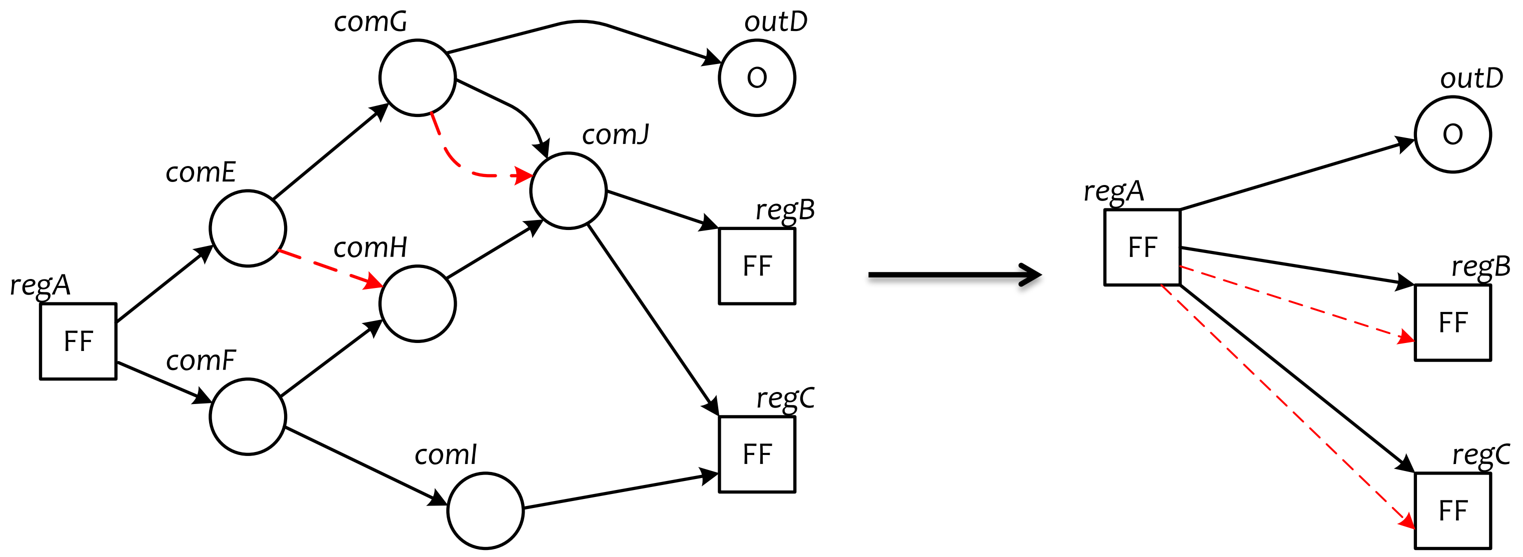
Hierarchical internal abstract sematic tree

Hierarchical signal-level data flow graphs (Connections between **regs** and **wires**)

Register relation graph (Connections between **regs**)

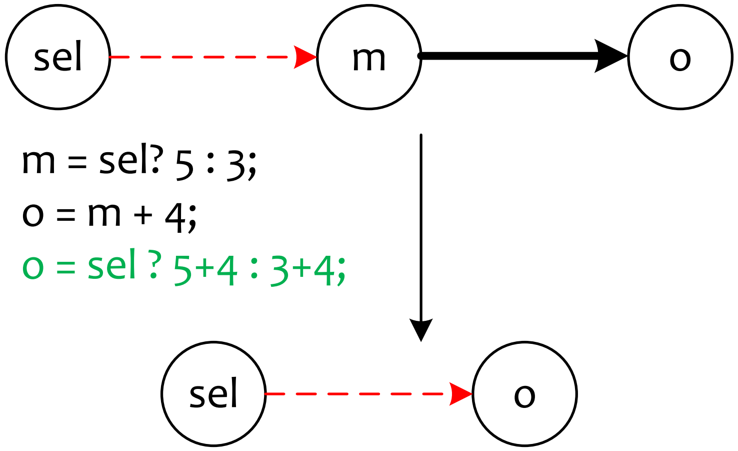
Controller detection and report

Reduction of Combi. Nodes

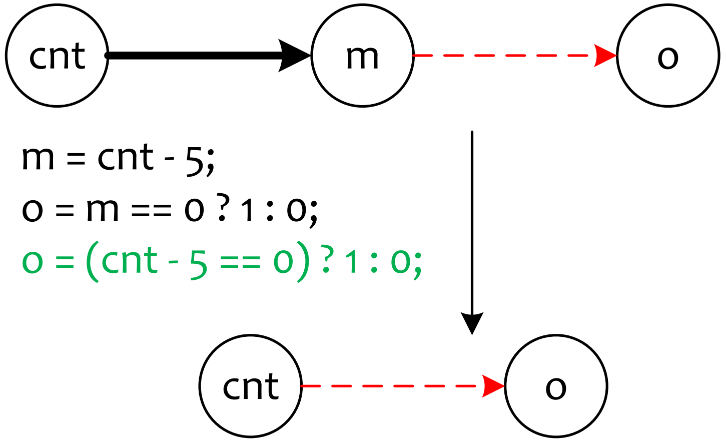


Path Type Calculation

Type reduction

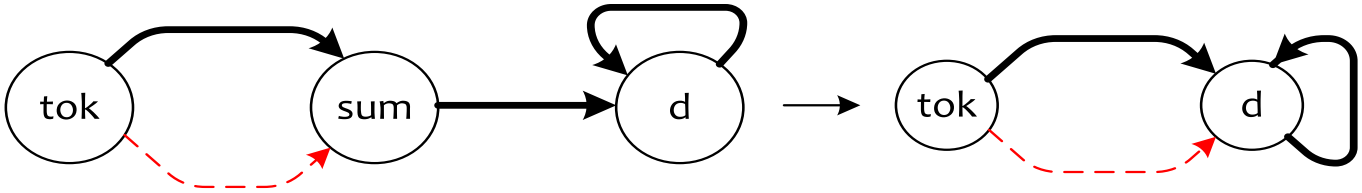


m = sel? 5 : 3;
o = m + 4;
o = sel ? 5+4 : 3+4;



m = cnt - 5;
o = m == 0 ? 1 : 0;
o = (cnt - 5 == 0) ? 1 : 0;

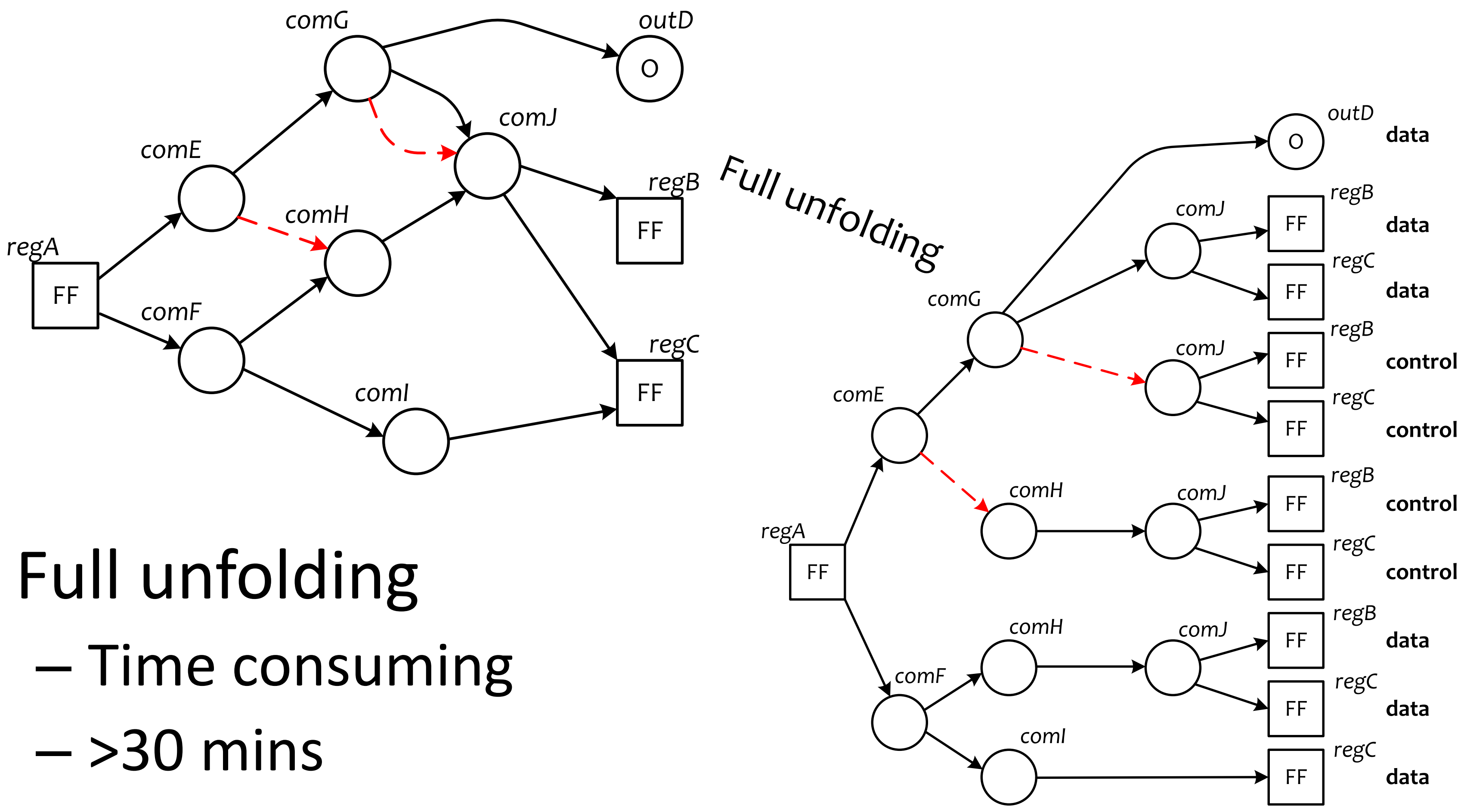
Parallel type preservation



sum = tok == 0? 0 : tok;
d = d + sum;

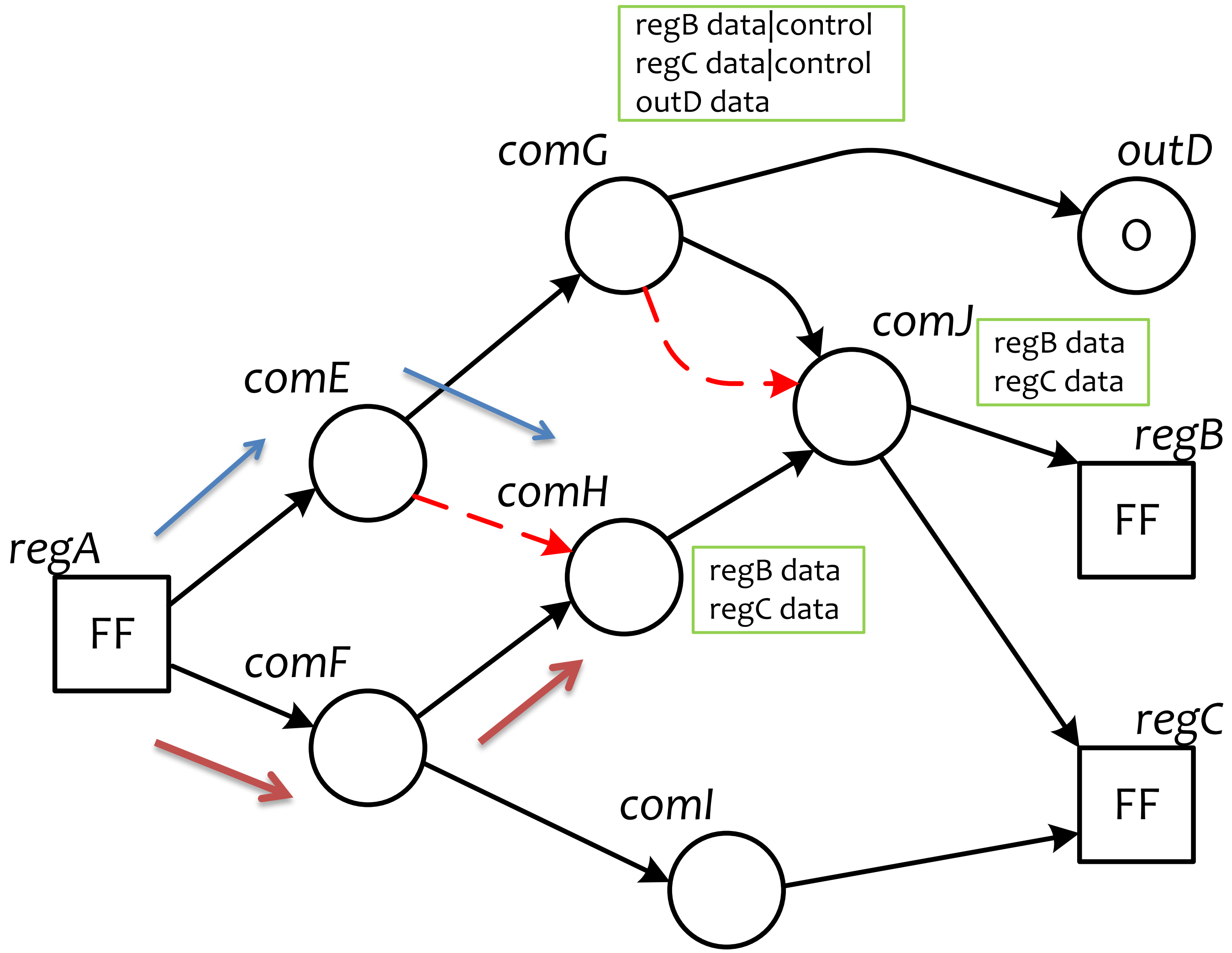
d = tok==0 ? d + 0 : d + tok;

Path Exploration



- Full unfolding
  - Time consuming
  - >30 mins

Software Cache (Dyn. Prog.)



MANCHESTER 1824

The University of Manchester

Results

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

MANCHESTER 1824

The University of Manchester

Results

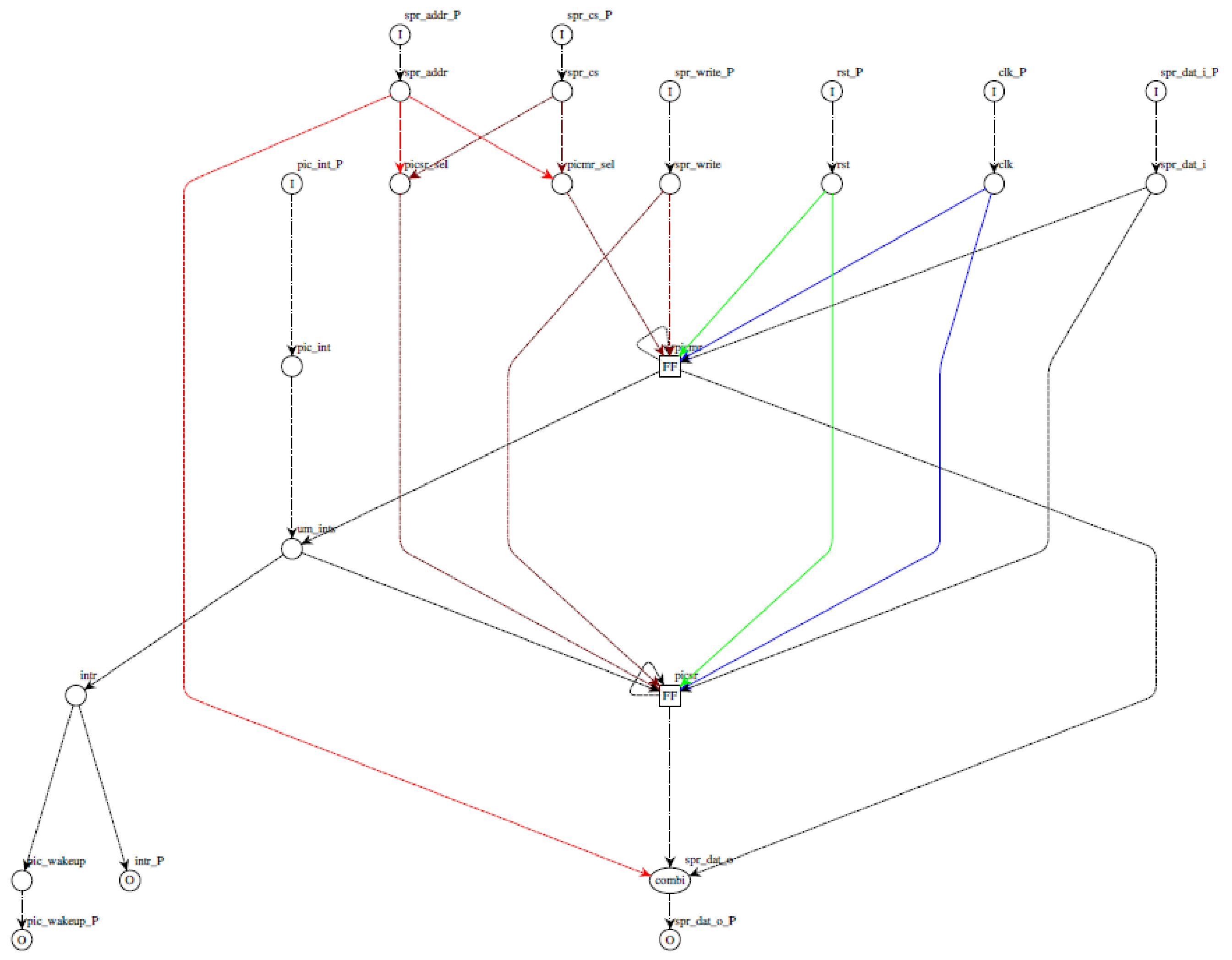
Name	Time	Reported	Verified	FSM	Counter	Flag	Error	Rate
OR1200	1s	19	17	7	5	5	2	89%
Reed-Solomon	2.0s	56	54	6	36	12	2	96%
H.264/AVC	7.1s	55	49	13	30	6	6	89%

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.

Computer Generated SDFG



MANCHESTER
1824

The University
of Manchester

Computer Generated Report

```
> report_fsm
SUMMARY:
In this extraction, 2074 nodes has been scanned, in which 120 nodes are
registers.
In total 30 FSM controllers has been found in 101 potential FSM registers.
The extracted FSMs are listed below:
.......
[10]  or1200_cpu/or1200_except/except_type FSM|ADR
[11]  or1200_cpu/or1200_except/extend_flush FSM|FLAG
[12]  or1200_cpu/or1200_except/state FSM|FLAG
[13]  or1200_cpu/or1200_if/saved FLAG
[14]  or1200_cpu/or1200_mult_mac/div_free FLAG
........
```

MANCHESTER
1824

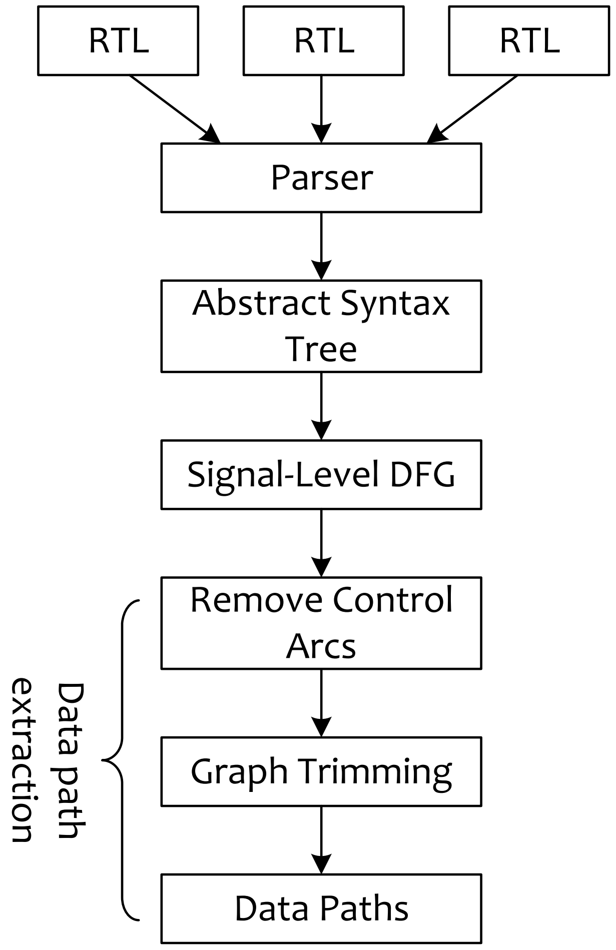
The University
of Manchester

Case 2: Data-path Extraction

- Traditional
  - State space analysis
  - Only feasible for small designs
- SDFG
  - Trimming control related nodes
  - Fast even for large scale designs

- Wei Song, Jim Garside and Doug Edwards. **Automatic data path extraction in large-scale register-transfer level designs.** In *Proc. of IEEE International Symposium on Circuits and Systems (ISCAS)*, Melbourne, Australia, June 2014

Tool Flow



Multi-file Verilog RTL designs

Hierarchical internal abstract sematic tree

Hierarchical signal-level data flow graphs (Connections between **regs** and **wires**)

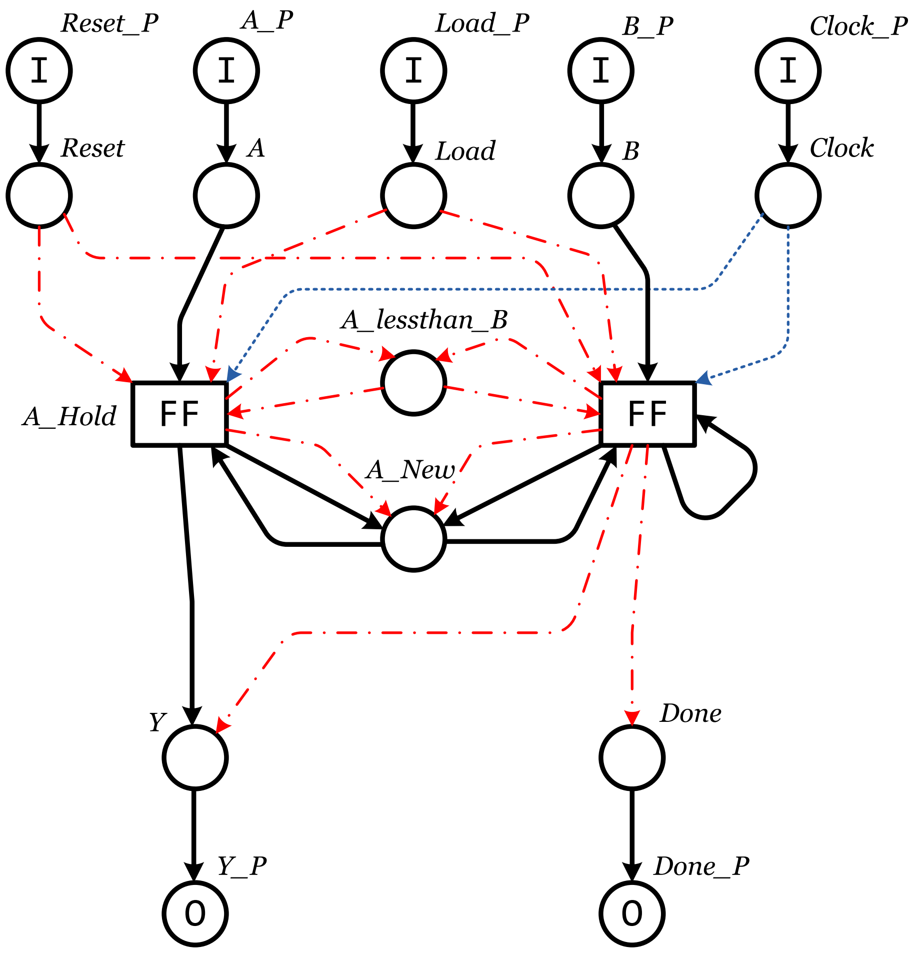
Remove control arcs

Remove dangling nodes

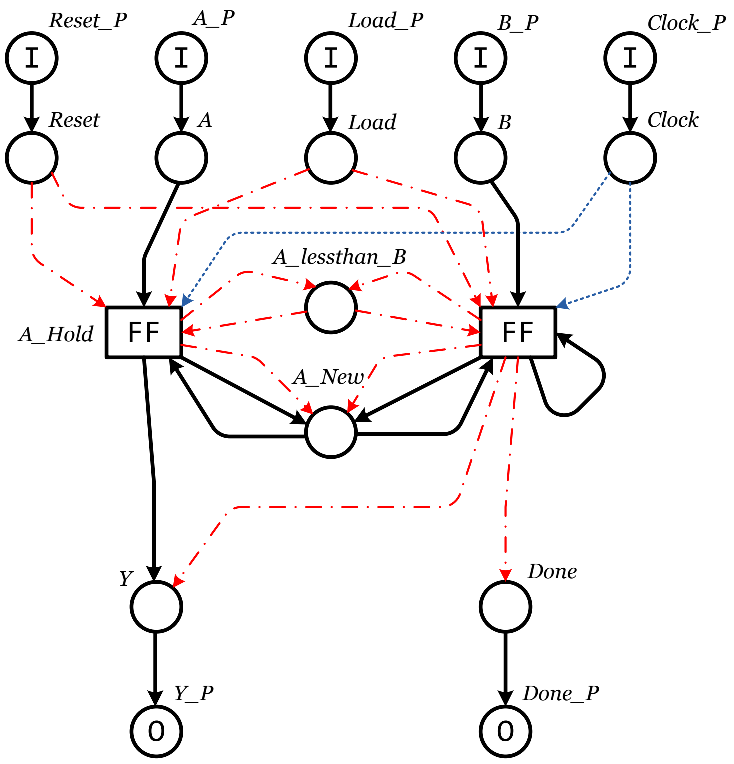
Report data path in a reduced SDFG

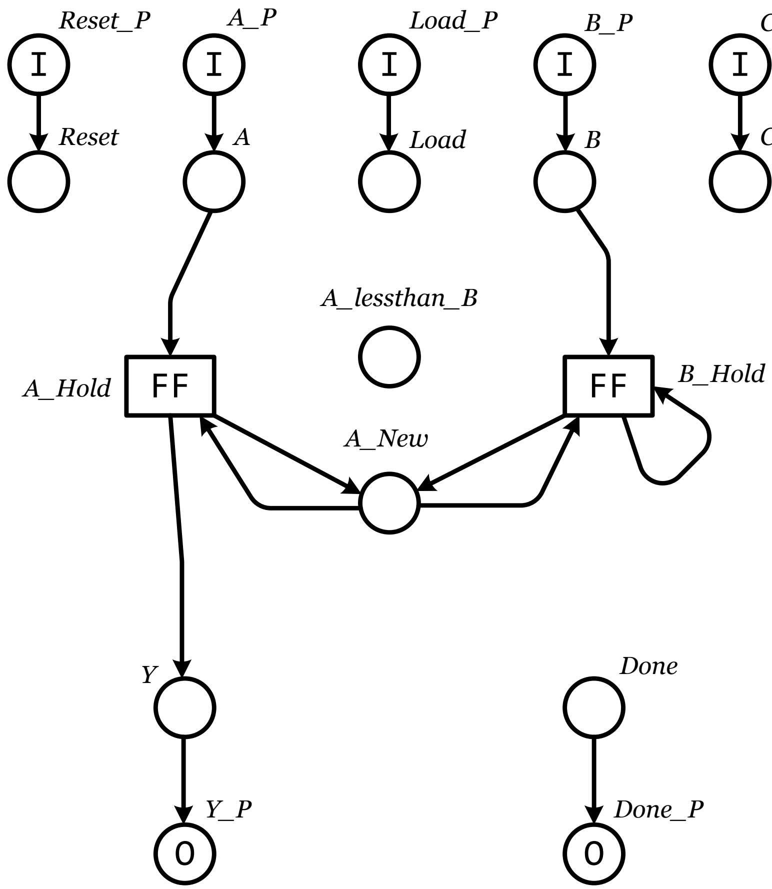
Greatest Common Divisor

```
module GCD (Clock,Reset,Load,A,B,Done,Y);
    input  Clock,Reset,Load;
    input  [7:0] A,B;
    output Done;
    output [7:0] Y;
    reg A_lessthan_B,Done;
    reg [7:0] A_New,A_Hold,B_Hold,Y;

    always @(posedge Clock)
      if(Reset) begin
         A_Hold = 0;
         B_Hold = 0;
      end else if(Load) begin
         A_Hold = A;
         B_Hold = B;
      end else if(A_lessthan_B) begin
         A_Hold = B_Hold;
         B_Hold = A_New;
      end else
         A_Hold = A_New;

    always @(A_Hold or B_Hold)
      if(A_Hold >= B_Hold) begin
         A_lessthan_B = 0;
         A_New = A_Hold - B_Hold;
      end else begin
         A_lessthan_B = 1;
         A_New = A_Hold;
      end

    always @(A_Hold or B_Hold)
      if(B_Hold == 0) begin
         Done = 1;
         Y = A_Hold;
      end else begin
         Done = 0; Y = 0;
      end
endmodule
```

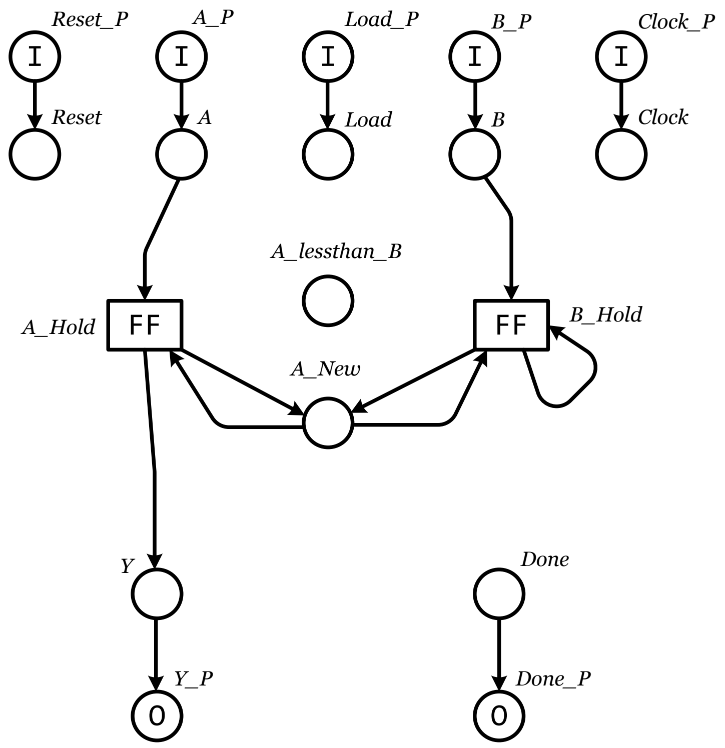


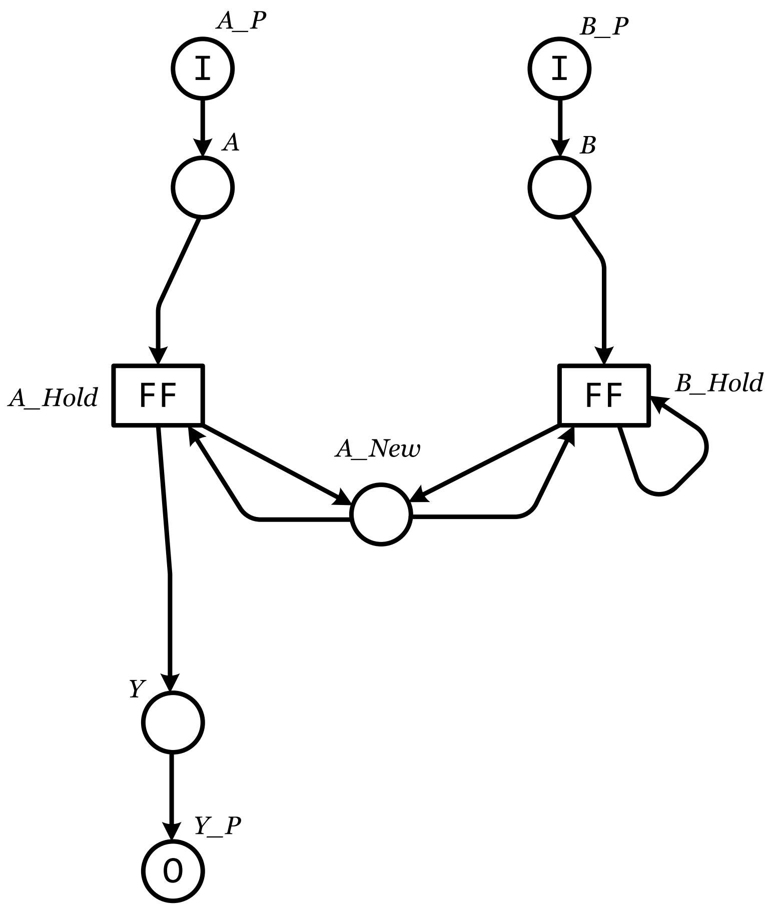
Remove Control Arcs



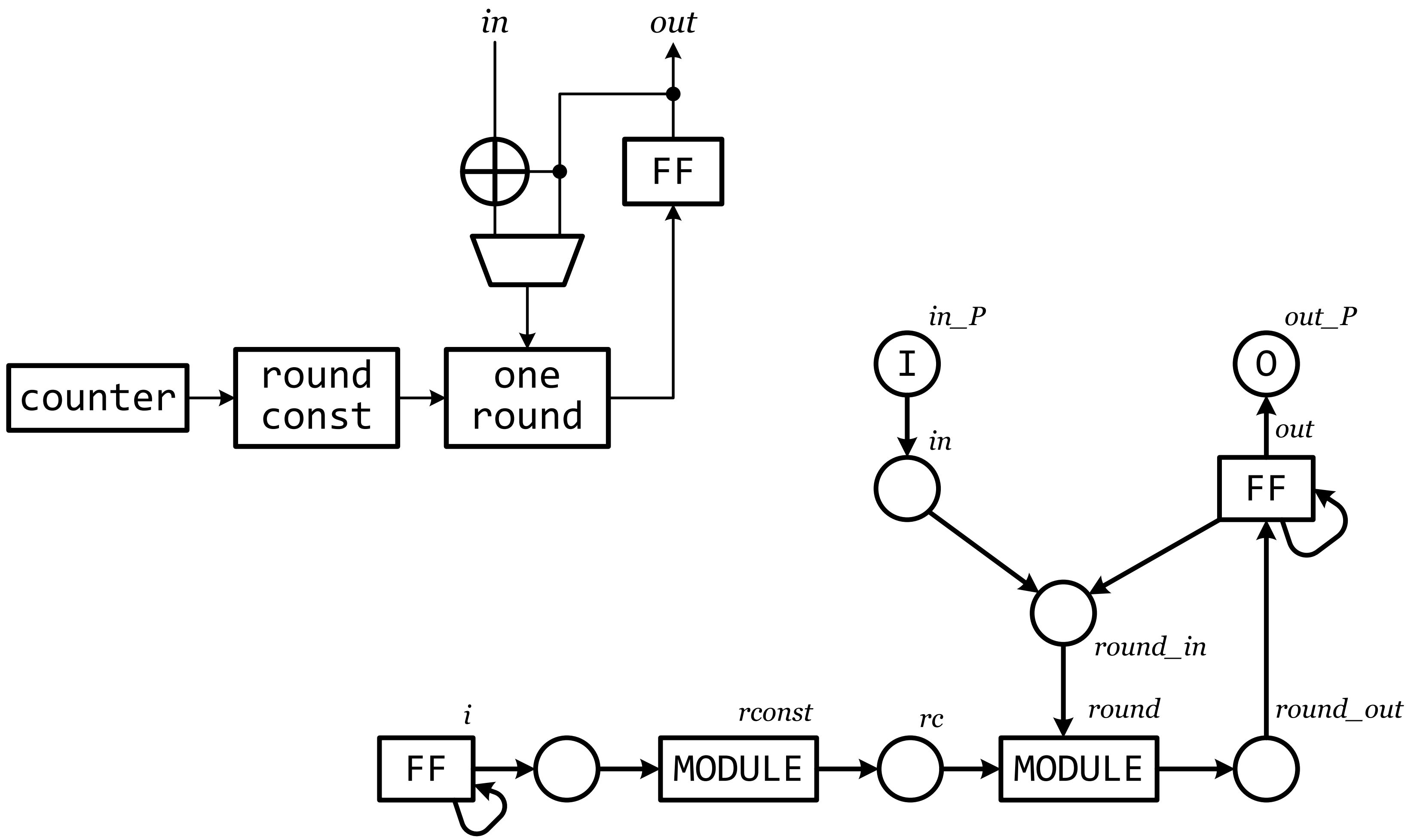


Trim the SDFG

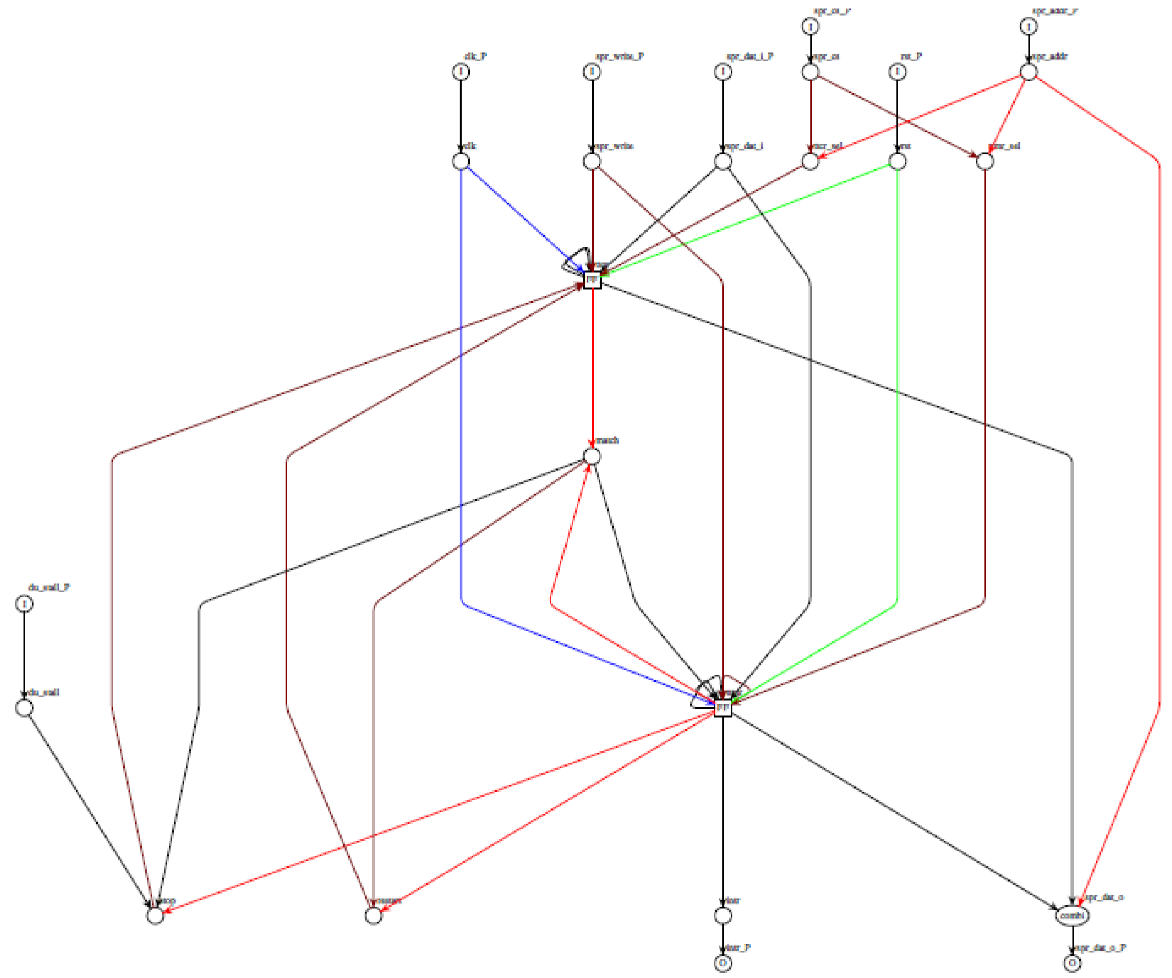


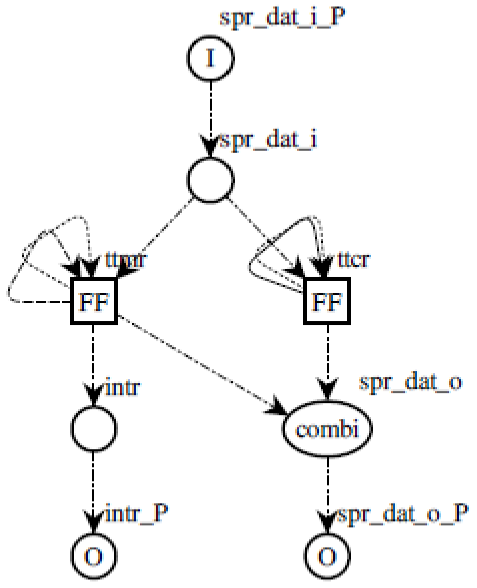


Permutation Module (SHA-3)

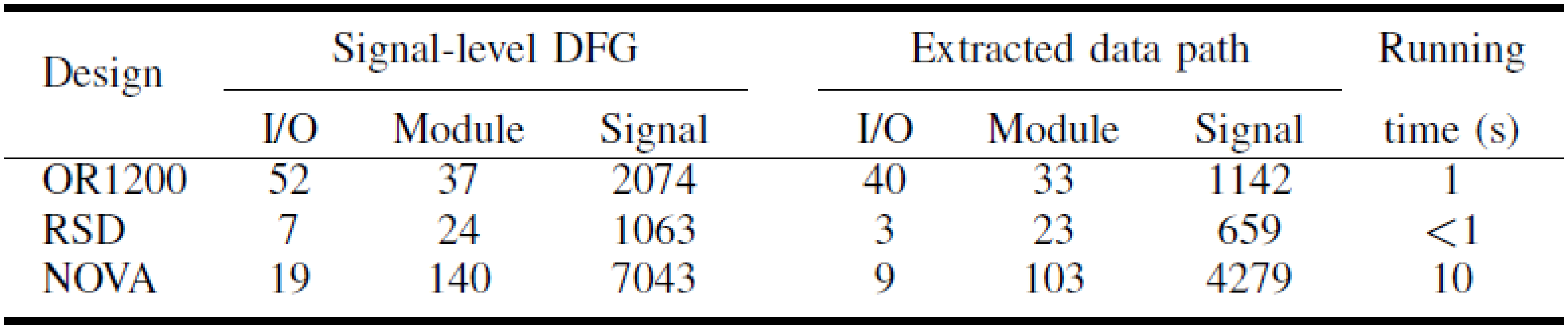


Large Scale Designs





Performance



Design	Signal-level DFG			Extracted data path			Running
	I/O	Module	Signal	I/O	Module	Signal	time (s)
OR1200	52	37	2074	40	33	1142	1
RSD	7	24	1063	3	23	659	<1
NOVA	19	140	7043	9	103	4279	10

Async Verilog Synthesisor (AVS)

- AVS
  - A C/C++ analysis shell system
  - https://github.com/wsong83/Asynchronous-Verilog-Synthesiser
- Libraries
  - Parser: Bison, Flex, Boost::Spirit
  - Function: C++0x, GNU Boost, GNU MP Bignum lib
  - Shell: cppTcl, Tcl/Tk, rlwrap
  - Graphic: Qt, OGDF
  - File format: pugixml

MANCHESTER
1824

The University of Manchester

Async Verilog Synthesisor (AVS)

- Shell commands

– analyze	read in the Verilog HDL design files.
– annotate_saif	annotate a saif file for a design.
– current_design	set or show the current target design.
– elaborate	build up a design from a Verilog module.
– extract_datapath	extract the datapaths from an SDFG.
– extract_rrg	extract a register relation graph (RRG) from.
– extract_sdfg	extract the SDFG of a module.
– partition	partition the current design.
– read_saif	read a saif file for a design.
– report_fsm	report the FSMs in a design.
– report_partition	report possible partitions of the current design.
– write_sdfg	write out the SDFG graph (SDFG/RRG/DataPath).

Summary

- SDFG
  - A new diagram which represents the relations between the signals in a large scale Verilog RTL design.
- Automatic Controller Extraction
  - Extracting all controllers using pattern matching
- Data-path extraction
  - Brutal but effective
- Future work
  - Port type estimation
  - Data rate estimation / data-flow extraction
  - System partition

THANK YOU!