Efficient BMC for Multi-Clock Systems with Clocked Specifications

Malay K. Ganai

(joint work with Aarti Gupta)

NEC Laboratories America,
Princeton, NJ

ASPDAC

Jan 24, 2007

Outline

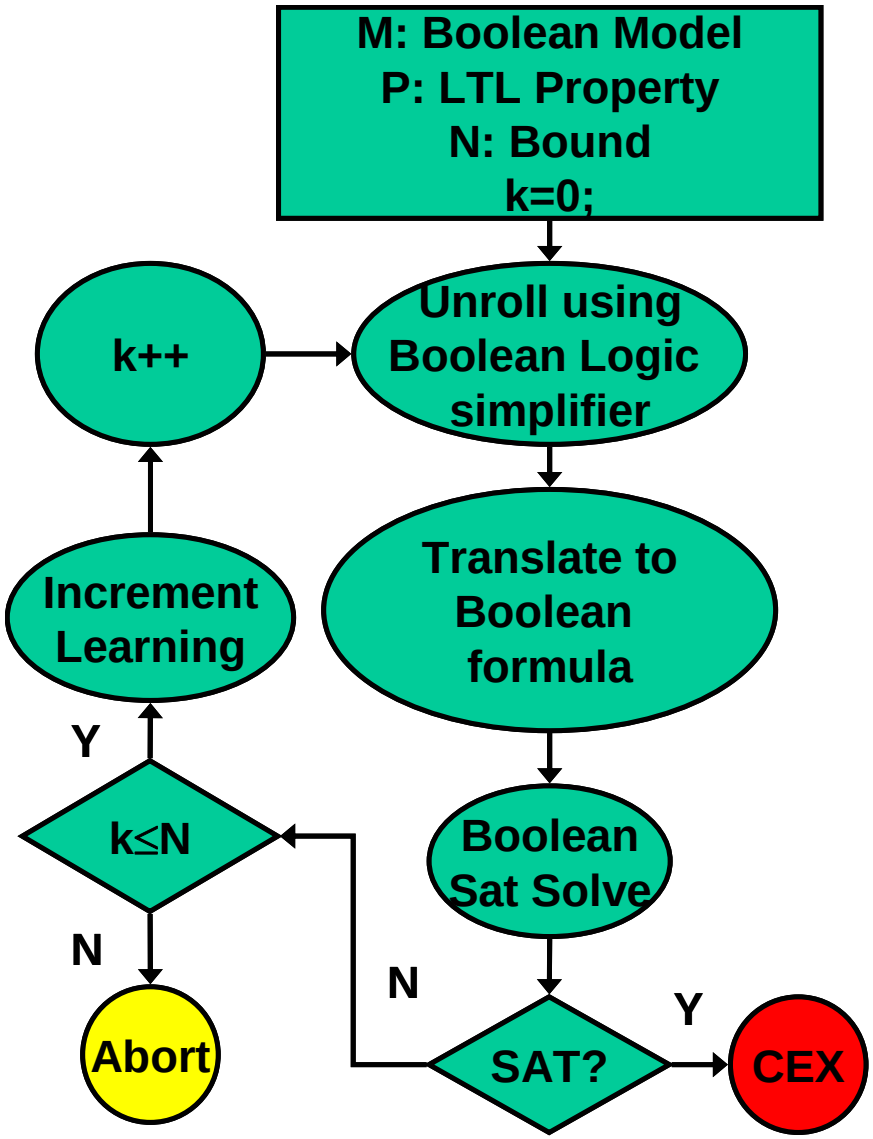
- Introduction and Motivation
- Background and Related Work
  - Bounded Model Checking (BMC)
  - Clocked Specifications and translation rules
- Our Contributions
  - Uniform Clock Modeling
  - Improving BMC for multi-clock systems
    - ☆ Reducing Unrolling and Loop-checks
    - ☆ Dynamic Simplifications of unrolled model
  - Customizing BMC for clocked specifications
- Experiments
  - Evaluation on *Opencore* multi-clock systems
- Summary

Multi-Clock Systems and Specifications

- Multi-clock systems
  - Multiple clock domains (MCD)
  - Clock with arbitrary frequency ratio
  - Gated and phase clocks

  - Synchronous: One clock generator, but with MCD
  - Advantages over single clock system
    - ☆ Reduce switching activity using gated clock for low power
    - ☆ Integration of IPs of multiple clocks
    - ☆ Reduce clock skews, global clock routing not required
- Clocked Specifications
  - Sub-formulas using various clocks
  - Express complex interaction between MCD

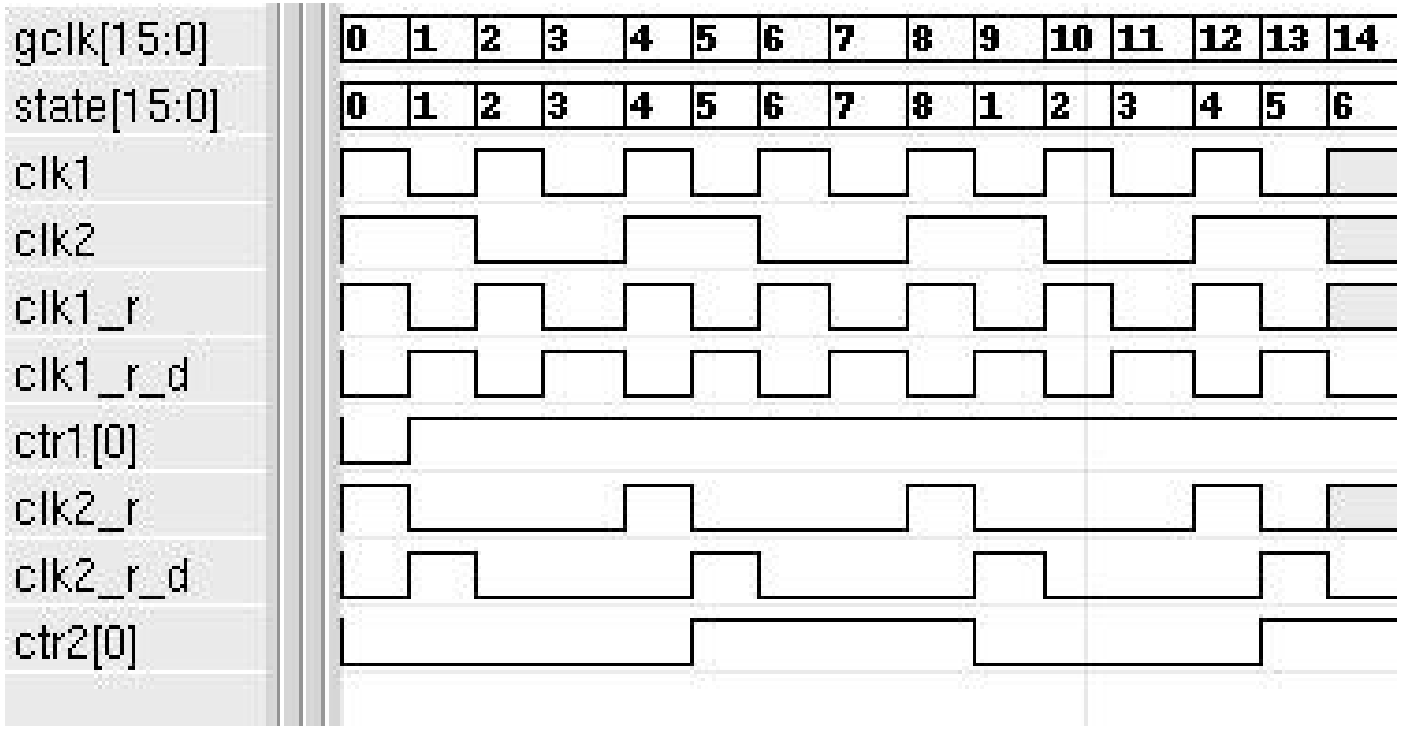
SAT-based BMC



LTL property, $\varphi :=$
$true \mid false \mid p \mid \neg p \mid$
$\varphi_1 \wedge \varphi_2 \mid \boldsymbol{X}\ \varphi \mid \boldsymbol{F}\ \varphi \mid \boldsymbol{G}\ \varphi \mid$
$[\varphi_1\ \boldsymbol{U}\ \varphi_2]$

- SAT: Improved engineering
- Dynamic circuit simplifications
- Property-specific customizations
- Incremental BMC
- Distributed BMC
- BMC for embedded memories

Clocked Specifications using “@”



- **Clocked LTL formulas:**
  - **P1:** ***F***(ctr2[0] * (***X***(ctr2[0]))@clk2_r_d)
  - **P2:** ***F***((ctr2[0] * ***X***(ctr2[0]))@clk2_r_d)

General Translation Rules

— Eisner *et al* , ICLAP 2003

Recursive Rules: $T^{clk}(f) \equiv (f)@clk$

- **Atomic proposition**
  - $T^{clk}(p) = \neg clk\ \boldsymbol{U}\ (clk \wedge p)$ // f is AP
- **Negation**
  - $T^{clk}(\neg f) = \neg T^{clk}(f)$
- **Conjunction**
  - $T^{clk}(f_1 \wedge f_2) = T^{clk}(f_1) \wedge T^{clk}(f_2)$
- **Next operator**
  - $T^{clk}(\boldsymbol{X}\ f) = \neg clk\ U\ (clk \wedge \boldsymbol{X}(\neg clk\ \boldsymbol{U}\ (clk \wedge T^{clk}(f)))$
- **Until operator**
  - $T^{clk}(\ f_1\ \boldsymbol{U}\ f_2\ ) = (clk \rightarrow T^{clk}(f_1))\ \boldsymbol{U}\ (clk \wedge T^{clk}(f_2))$
- **Nested Rule**
  - $T^{clk}((f)@clk1) = T^{clk1}(f)$

Translate clocked property to unclocked property and use standard BMC!

Translation Example

- **P1:** $F(ctr2[0] \wedge (\underline{X(ctr2[0])@C})$

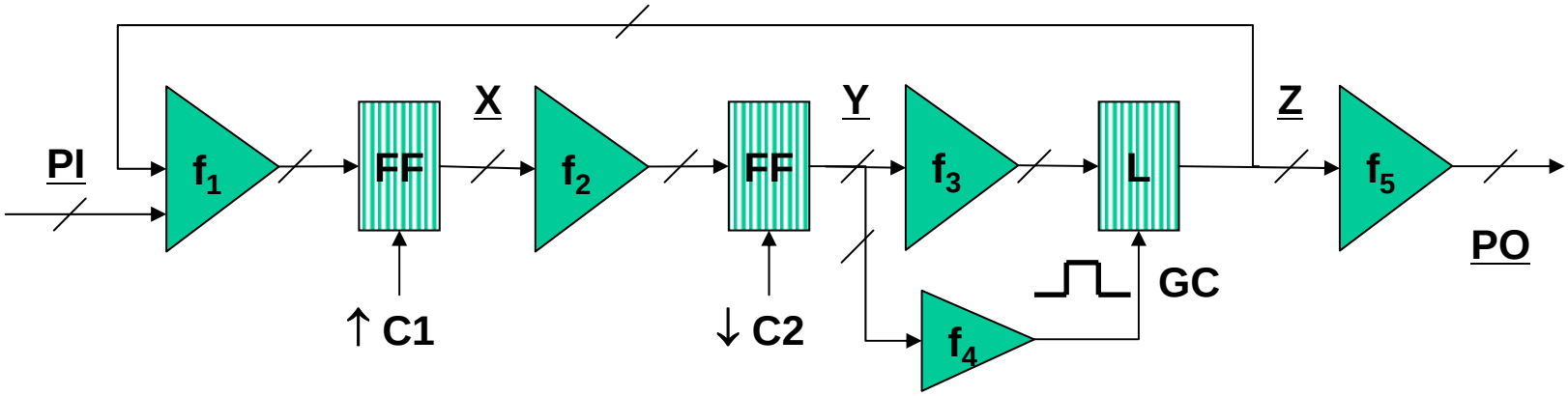
$F(ctr[0] \wedge (\neg C\ U\ (C \wedge$
$X\ (\neg C\ U\ (C \wedge$
$ctr2[0])))));$

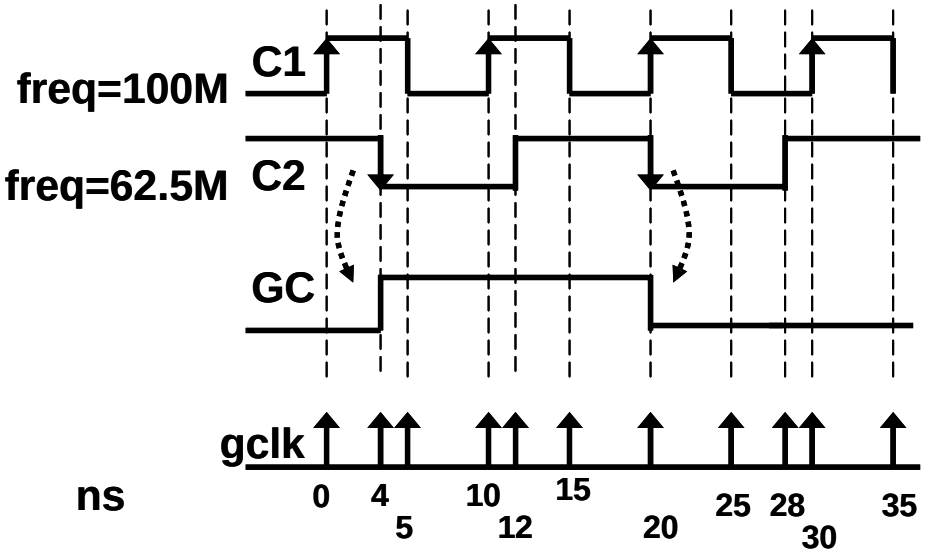
- **P2:** $F(\ \underline{(ctr2[0]\ *\ X(ctr2[0]))@C})$

$F(\neg C\ U\ (C \wedge$
$ctr2[0] \wedge X\ (\neg C\ U\ (C \wedge$
$ctr2[0])))));$

Translation leads to large nested un-clocked LTL formula !

Multi-clock System (synchronous)

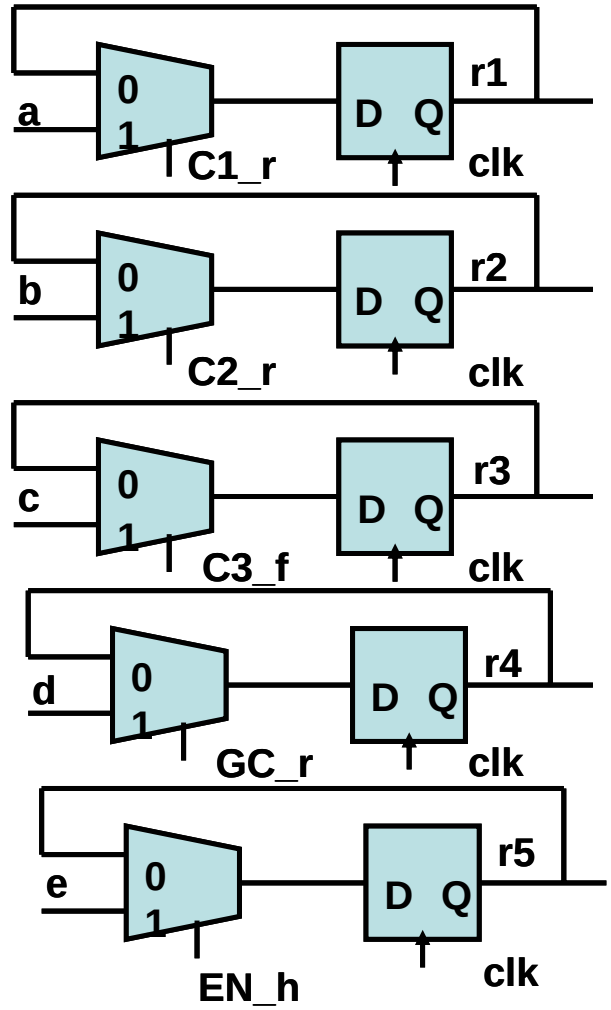




- **PI:** Primary Inputs
- **L:** Latches (Z)
- **FF:** Flip-flops (X, Y)
- **PO:** Primary Outputs
- **f_1-f_5:** Combination blocks
- **C1,C2:** Clocks
- **GC:** Gated Clock

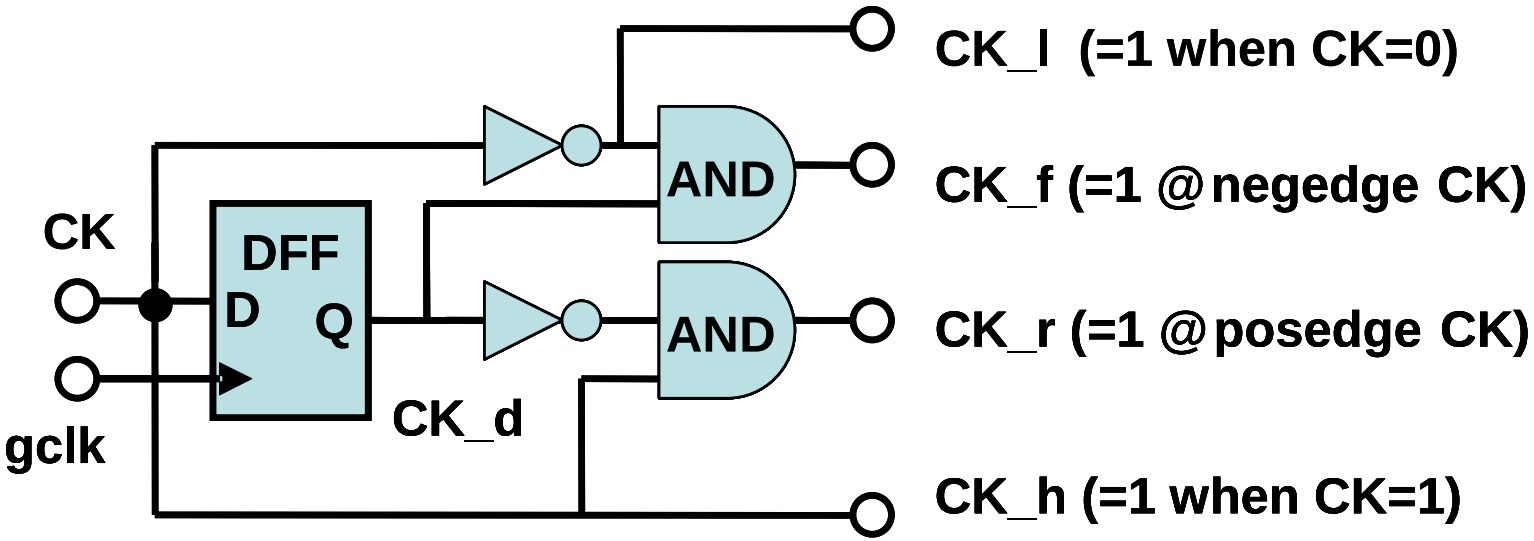
Clock Modeling

```
/* Multiple clocks */
always @(posedge C1) r1 <= a ;
always @(posedge C2)  r2 <= b ;
/* negative phase clock */
always @(negedge C3)  r3 <= c ;
/* Gated clock */
always @(posedge GC) r4 <= d ;
/* level-sensitive latch */
always @(EN_h) if (EN_h) r5 <= e;
....
```

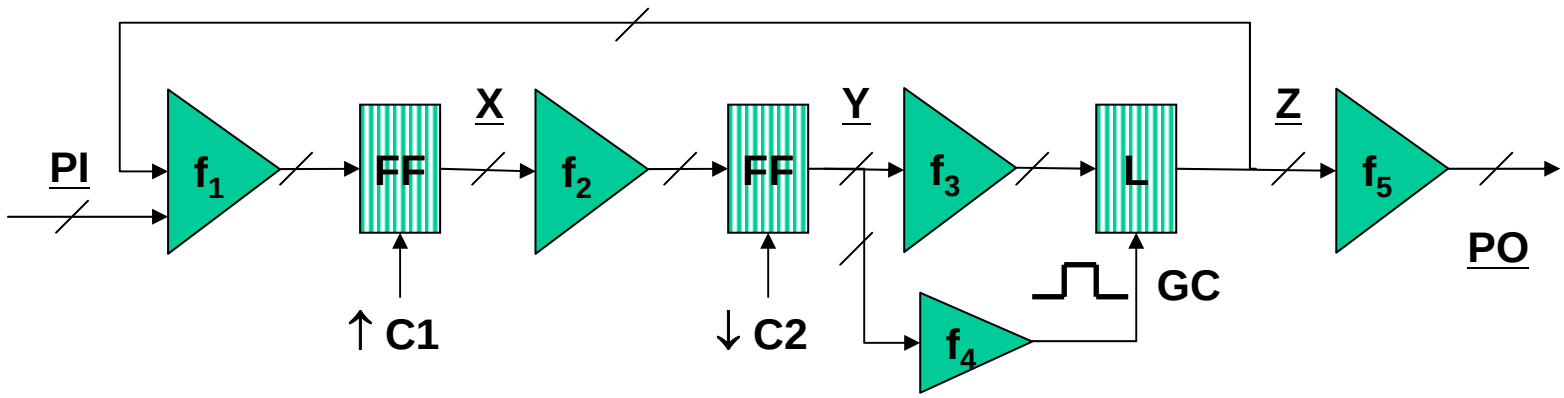


Translation is precise and cycle accurate

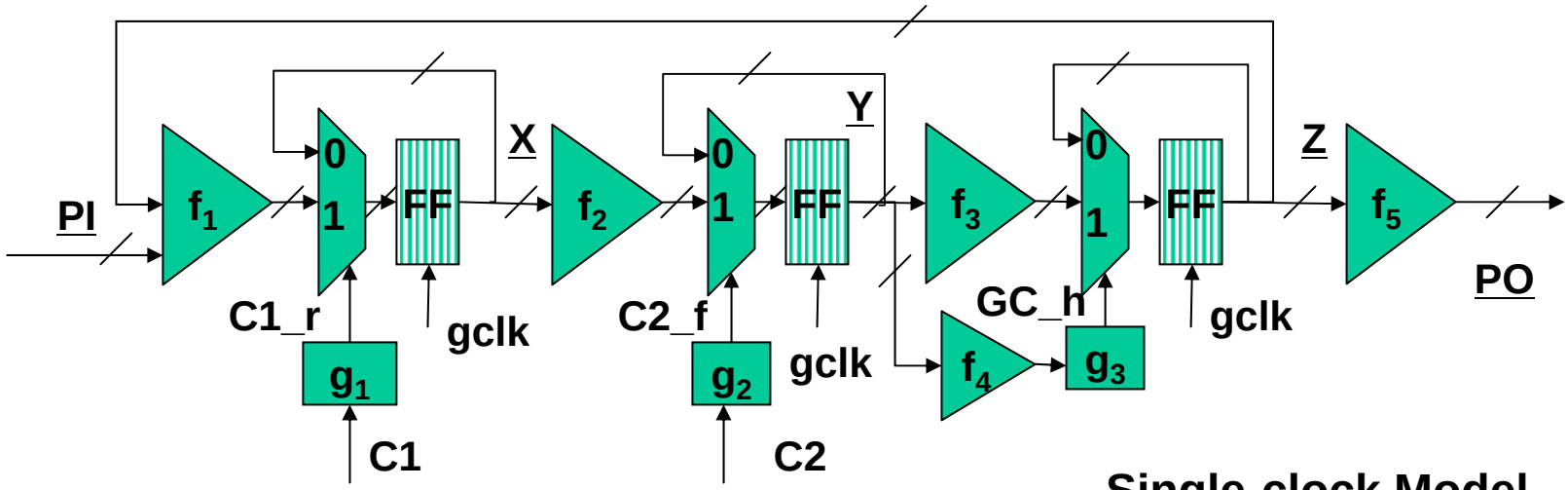
Clock-enable Signal Generator



Multi-clock to Single-clock Model



Multiple-clock Model



Single-clock Model

NEXT(X) = (C1_r) ? f_1(Z,PI) : X;
NEXT(Y) = (C2_f) ? f_2(X) : Y;
NEXT(Z) = (GC_r) ? f_3(Y) : Z;
PO = f_5(Z);

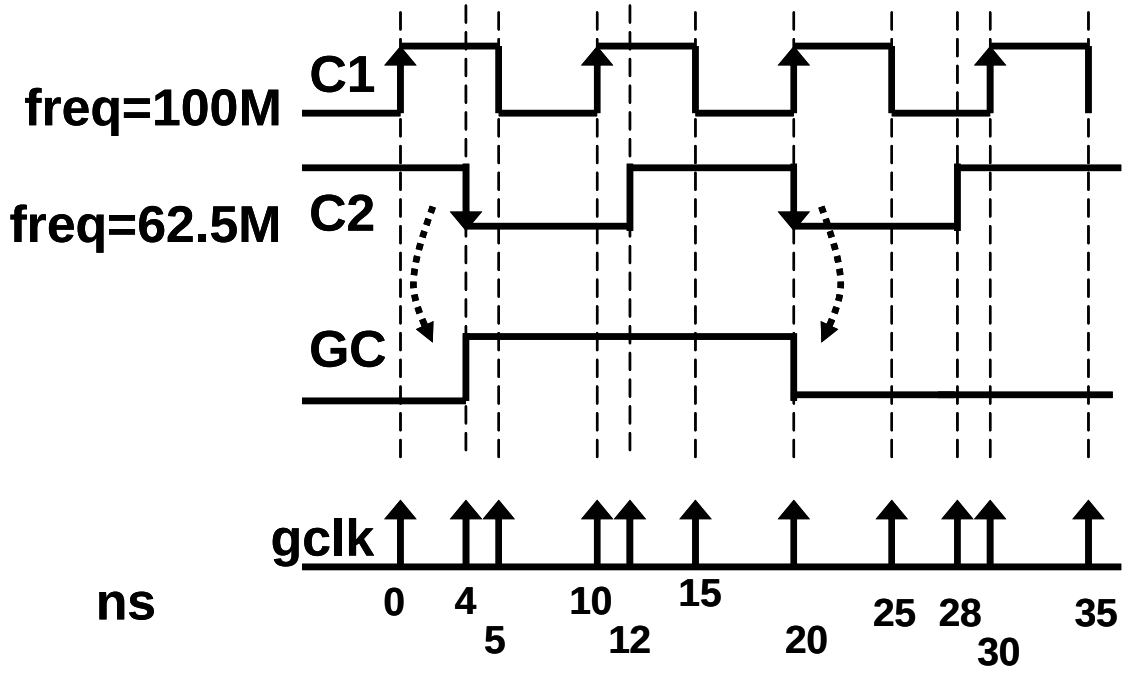
gclk frequency equals LCM of frequencies!

Clock schedules: Reduce BMC Unrolling

Configuration,

$S^i \equiv \langle t^i, c_1^i, c_2^i \rangle$

- $S^0 = \langle t^0 = 0, 1, 1 \rangle$
- $S^1 = \langle t^1 = 4, 1, 0 \rangle$
- $S^2 = \langle t^2 = 5, 0, 0 \rangle$
- $S^3 = \langle t^3 = 10, 1, 0 \rangle$
- $S^4 = \langle t^4 = 12, 1, 1 \rangle$
- $S^5 = \langle t^5 = 15, 0, 1 \rangle$
- $S^6 = \langle t^6 = 20, 1, 0 \rangle$
- $S^7 = \langle t^7 = 25, 0, 0 \rangle$
- $S^8 = \langle t^8 = 28, 0, 1 \rangle$
- $S^9 = \langle t^9 = 30, 1, 1 \rangle$



- Global state unchanged between S^i and S^{i+1}
- gclk ticks correspond to S^i
- BMC unrolling at gclk ticks
- Clock C_j is constrained to c_j^i value

Recurring Configurations: BMC loop checks

$S^i \equiv \langle t^i, c_1^i, \ldots, c_n^i \rangle$ is **equivalent** to $S^j \equiv \langle t^j, c_1^j, \ldots, c_n^j \rangle$
(i.e., recur after every R ticks of gclk)

$\forall \quad 0 \le k < R$

$\forall\ 0 \le m < n\ (c_m^{i+k} = c_m^{j+k})$
$t^{i+k} - t^{j+k} = T$

R: Recurrence Length
T: Repetition Period (=LCM of time periods)

	i			i+k							j			j+k					
c_1	0	1	0	1	0	1	.	.			0	1	0	1	0	1	.	.	
c_2	0	1	1	1	0	0	.	.			0	1	1	1	0	0	.	.	

"Global Clock states S^i and S^{i+R} are equivalent"

BMC loop-check using SAT solver between unrolling i and j if (i-j) % R = 0

Unrolling using Dynamic Simplification

k	PI^k	$C1\_r^k$	$C2\_f^k$	GC_h^k	$\underline{X}^k$	$\underline{Y}^k$	$\underline{Z}^k$	$\underline{PO}^k$
0	PI^0	1	0	0	$\underline{X}^0$	$\underline{Y}^0$	$\underline{Z}^0$	$PO^0 = f_5(\underline{Z}^0)$
1	PI^1	0	1	0	$\underline{X}^1 = f_1(\underline{Z}^0, \underline{PI}^0)$	$\underline{Y}^0$	$\underline{Z}^0$	$\underline{PO}^0$
2	PI^2	0	0	1	$\underline{X}^1$	$\underline{Y}^2 = f_2(\underline{X}^1)$	$\underline{Z}^0$	$\underline{PO}^0$
3	PI^3	1	0	1	$\underline{X}^1$	$\underline{Y}^2$	$\underline{Z}^3 = f_3(\underline{Y}^2)$	$\underline{PO}^3 = f_5(\underline{Z}^3)$
4	PI^4	0	0	1	$\underline{X}^4 = f_1(\underline{Z}^3, \underline{PI}^3)$	$\underline{Y}^2$	$\underline{Z}^3$	$\underline{PO}^3$
5	PI^5	0	0	1	$\underline{X}^4$	$\underline{Y}^2$	$\underline{Z}^3$	$\underline{PO}^3$
6	PI^6	1	1	1	$\underline{X}^4$	$\underline{Y}^2$	$\underline{Z}^3$	$\underline{PO}^3$
7	PI^7	0	0	0	$\underline{X}^7 = f_1(\underline{Z}^6, \underline{PI}^6)$	$\underline{Y}^7 = f_2(\underline{X}^6)$	$\underline{Z}^3$	$\underline{PO}^3$
8	PI^7	0	0	0	$\underline{X}^7$	$\underline{Y}^7$	$\underline{Z}^3$	$\underline{PO}^3$

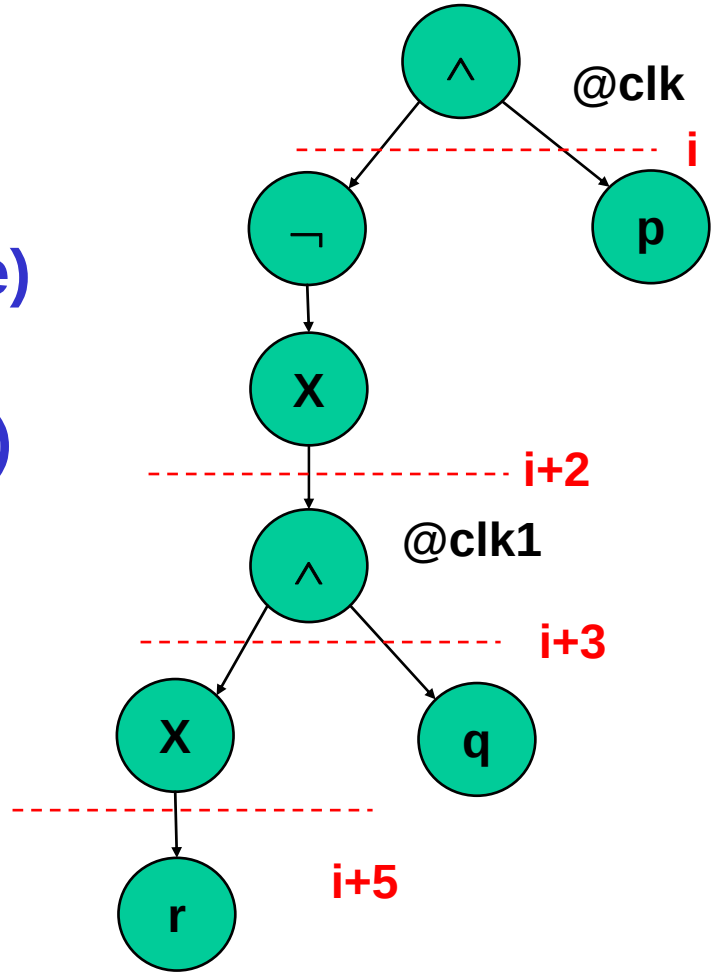
Simplification of unrolling using clock schedules:
Re-use of combinational blocks in unrolling !

Customization of Clocked Property: (F(f))@clk

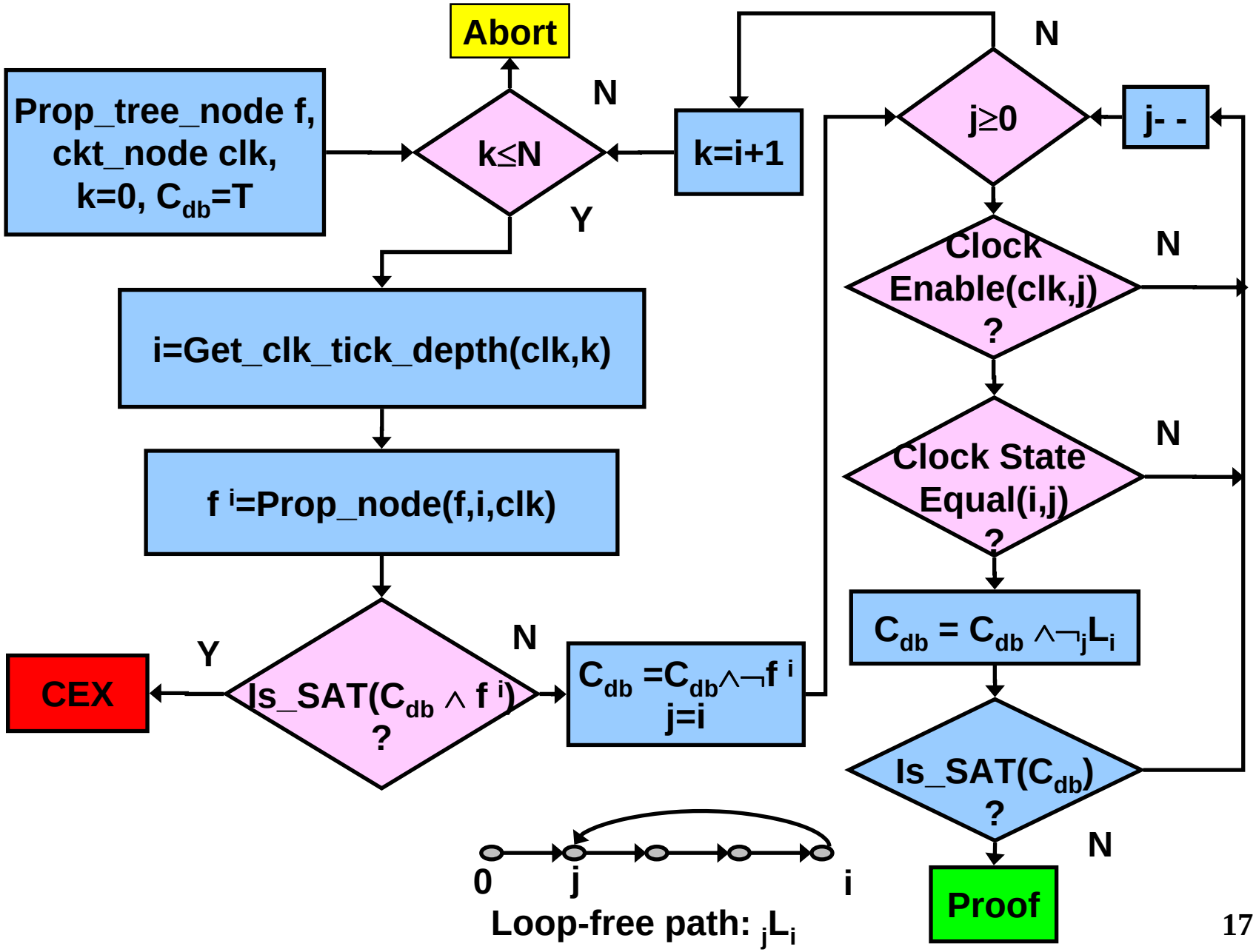
- **(*F*(f))@clk**
  - **Prop_tree node f**
    - ☆ **Boolean combination of nested *X* operators with propositional atoms**
  - **Sub expression clocks C (specified with @)**
    - ☆ **Boolean combination of input clocks**
- **Nested rules**
  - **(¬*X* f)@clk ≡ ¬ ((*X* f)@clk)**
  - ***X*(f ∧ *X*(g))@clk1)@clk ≡ (*X*f@clk1)@clk ∧(*X*(*X*g)@clk1)@clk**

Example: Clocked Property

- Example: $F(p \wedge \neg X(q \wedge X(r))@clk1)@clk$
  - Prop_tree_node f@clk
- ckt_node
  - p, q, r
- uckt_node (i^{th} unroll node)
  - p^i, q^i, r^i
- Clk ticks (i: unroll depths)
  - clk: i, i+2, i+4
  - clk1: i+1,i+3,i+5
- Prop_node (f,i,clk)
  - $p^i \wedge \neg (q^{i+3} \wedge r^{i+5})$



BMC_solve_F@: (F(f))@clk



Experiment: VGA/LCD Controller

- OpenCore Design: Provides VGA capabilities
  - Wishbone clock (Freq=416.66 Mhz) (wb_clk)
  - Pixel clock (Freq = 33.33Mhz) (px_clk)
  - Recurrence length=55 (Repeatition period = 83.33ns)
  - 162 FFs on pixel clock, 2340 FFs on wishbone clock
  - 87 primary input, 44K 2-input
- 13 clocked properties P1-P13
  - P1-P12: $(F(p \wedge X(q)))@px\_clk\_r$
  - P13: $F(p@wb\_clk\_r \wedge X(q)@px\_clk\_r)$
- BMC@ (Our method)
  - Customization, Dynamic simplification, clock schedules
- BestBMC (previous best so far)
  - Dynamic simplification, clock schedules
- Platform
  - 2.8 Ghz intel processor Xeon, 4 GB, Linux 2.4.21-27

Result: VGA/LCD Controller

Prp	WIT #D	#U	BMC@		BestBMC	
			F?	sec	F?(U*)	Sec
P1	2	1	Y	<1	Y	<1
P2	50	27	Y	1	Y	19
P3	101	55	Y	3	Y	186
P4	151	82	Y	5	Y	694
P5	351	190	Y	16	N(160)	TO
P6	101	55	Y	3	Y	186
P7	401	217	Y	18	N(161)	TO
P8	600	324	Y	32	N(162)	TO
P9	800	432	Y	52	N(162)	TO
P10	1000	540	Y	78	N(162)	TO
P11	800	432	Y	54	N(162)	TO
P12	850	459	Y	61	N(61)	TO
P13	906	489	Y	2.1k	N(81)	TO

Time limit: 2 hours

Experiment: Ethernet MAC Controller

- OpenCore Design: Tri-mode Ethernet MAC Controller
  - Clk_125M (F=125Mhz), Clk_user(F=100Mhz), Clk_reg(F=50Mhz)
  - Rx_clk(F=125/25/2.5Mhz), Tx_clk(F=125/25/12.5Mhz)
  - Recurrence length=19
  - 815 FFs on Clk_reg, 835 FFs @ Clk_user, 764 FFs @ Rx_clk
  - 775 FFs @ Tx_clk, 1772 FFs @ gated clocks
  - 142 primary input, 33K 2-input
- 16 clocked properties E1-E16
  - E2,E11: $(F(p))$@Clk_user_r
  - E9: $(F(p \wedge X(q))$@Clk_user_r))@Rx_clk_gated_r
  - Rest: $(F(p \wedge X(q)))$@clk_user_r
- BMC@ (Our method)
  - Customization, Dynamic simplification, clock schedules
- BestBMC (best so far)
  - Dynmaic simplification, clock schedules
- Platform
  - 2.8 Ghz Xeon, 4 GB, Linux 2.4.21-27

Result: Ethernet MAC Controller

Prp	WIT #D	#U	BMC@		BestBMC	
			F?	sec	F ?(U*)	sec
E1	299	149	Y	41	N(143)	TO
E2	269	134	Y	16	Y	4.4k
E3	279	139	Y	14	Y	5.3k
E4	463	232	Y	1.6k	N(145)	TO
E5	289	144	Y	21	144	5.9k
E6	309	154	Y	25	N(148)	TO
E7	299	149	Y	19	Y	6.7k
E8	319	159	Y	48	N(149)	TO
E9	434	216	Y	126	N(127)	TO
E10	299	159	Y	2	Y	3.1k
E11	2110	1235	Y	202	N(224)	TO
E12	2120	1240	Y	261	N(221)	TO
E13	2130	1247	Y	314	N(221)	TO
E14	2150	1259	Y	277	N(213)	TO
E15	2140	1252	Y	240	N(221)	TO
E16	2160	1264	Y	268	N(220)	TO

Summary

- Integrated scalable solution for verifying multi-clock system with clock specification
  - Uniform modeling scheme for multi-clock design with multiple, non-integral frequency rations, gated clocks, latches, multiple phases
  - Reducing BMC unrolling using event queue semantics
  - Dynamic simplification of BMC instances using clock constraints
  - Customization of clock specifications
- Experimental results show 1-2 orders of magnitude improvement due to customization