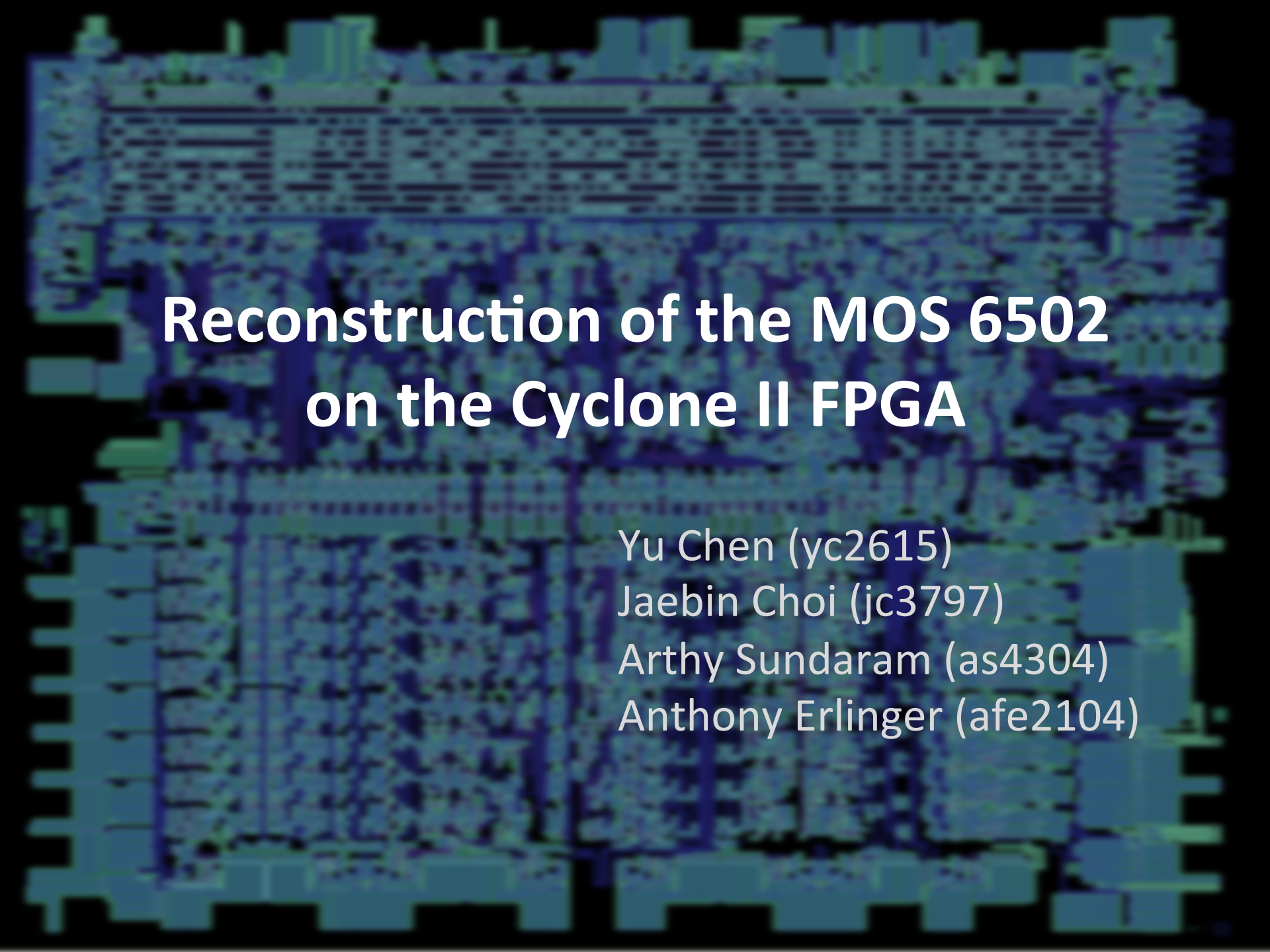
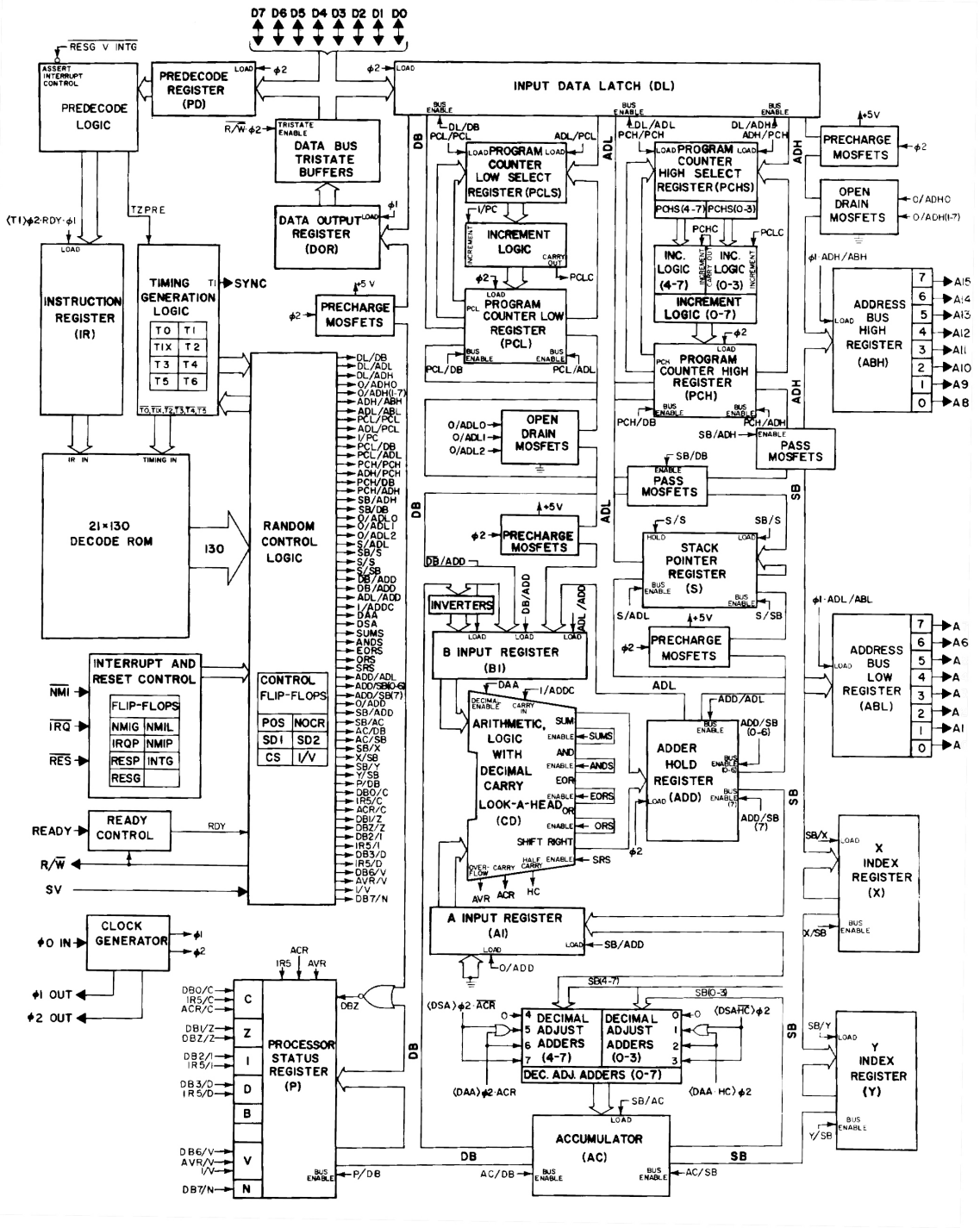


Reconstruction of the MOS 6502 on the Cyclone II FPGA

Yu Chen (yc2615)
Jaebin Choi (jc3797)
Arthy Sundaram (as4304)
Anthony Erlinger (afe2104)

Outline

- Redesign 6502
  - RTL Level structure
- 6502 opcodes summary
- 6502 on the DE2 board
  - Monitoring the internal registers
  - Bouncing Ball

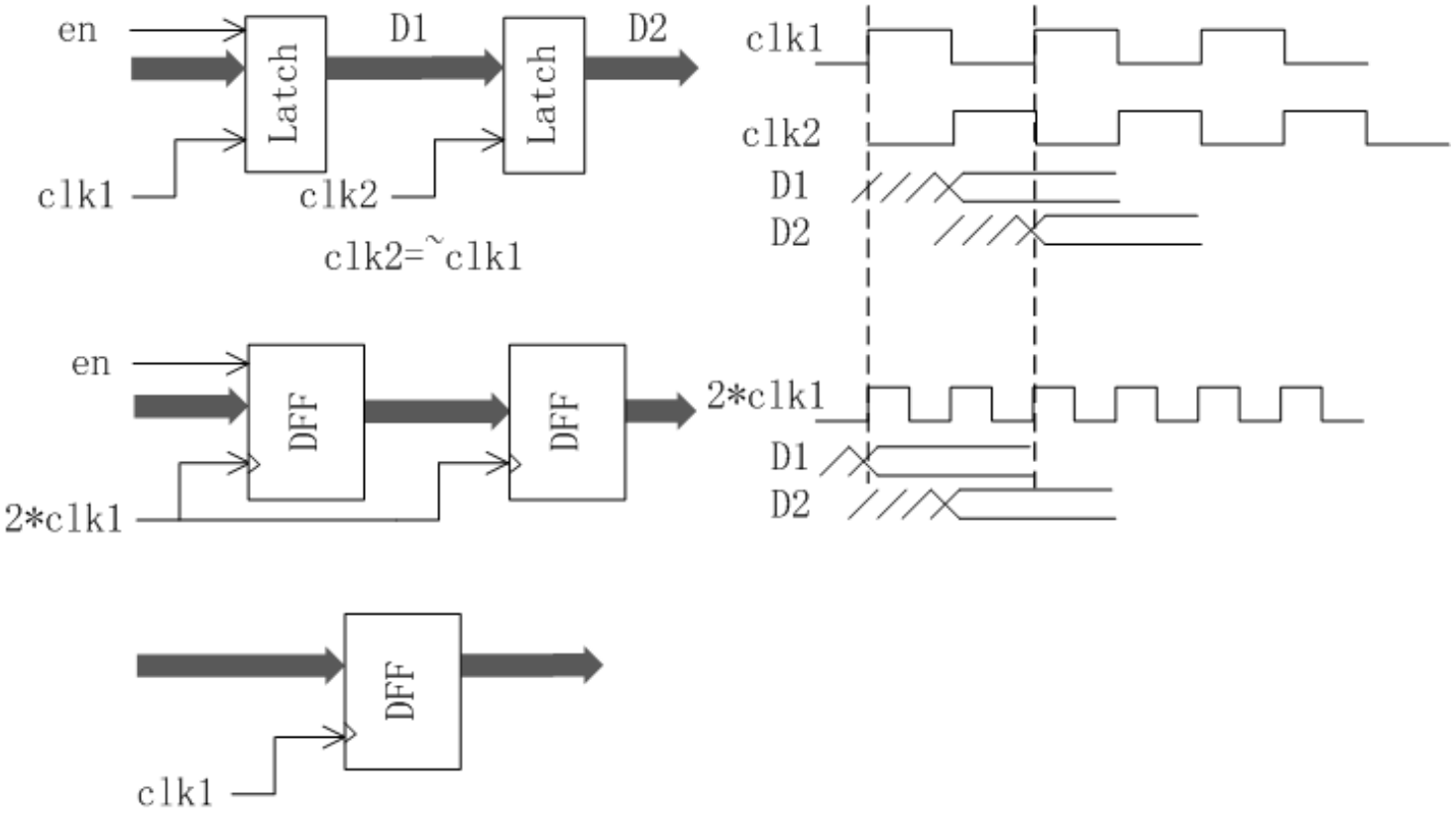


- Original Block Diagram

- Challenges:
  - Latch based
  - Two phases clk
  - High Impedance Bus

Redesign 6502

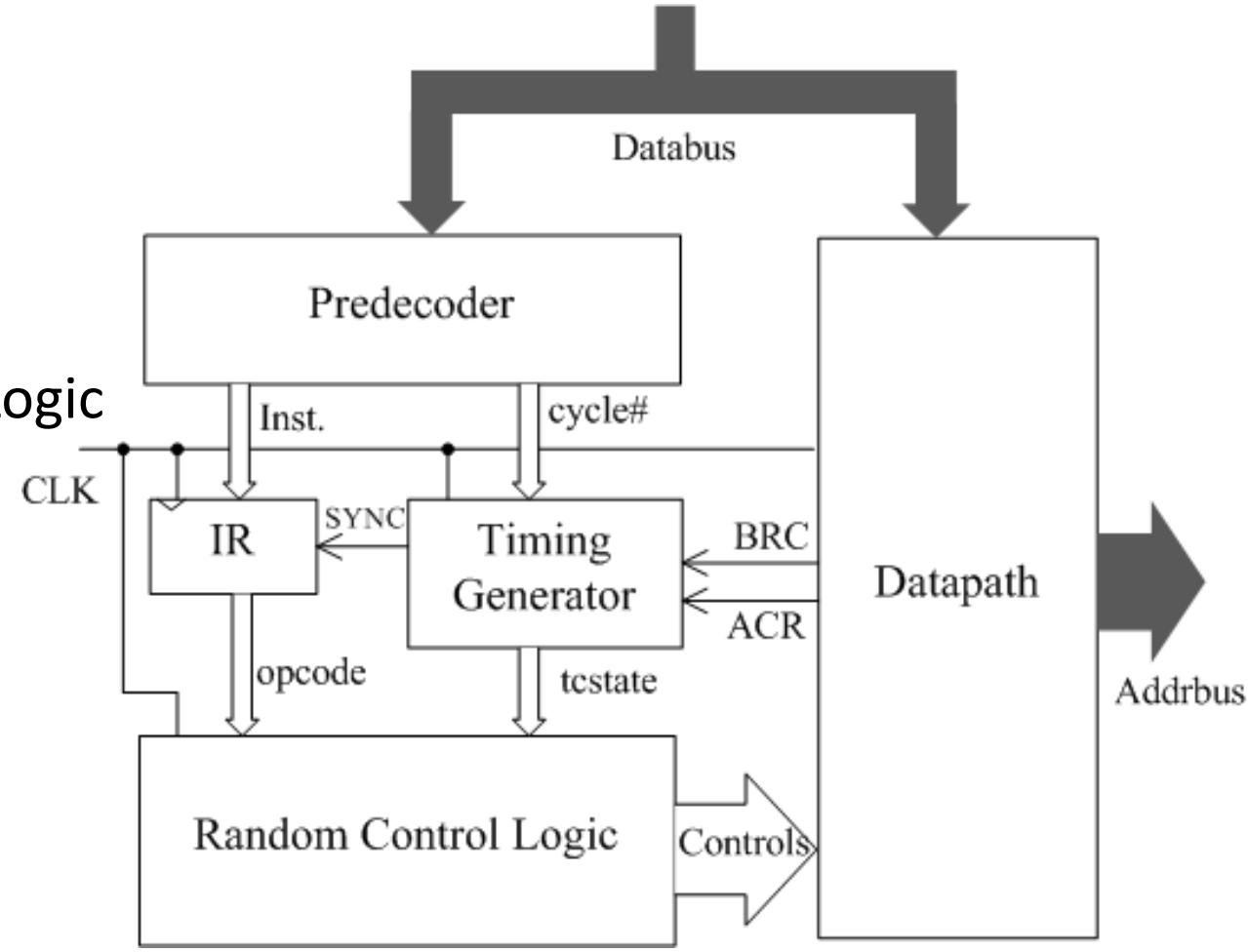
- Latch→DFF



- Single Clock
- Datapath→Mux

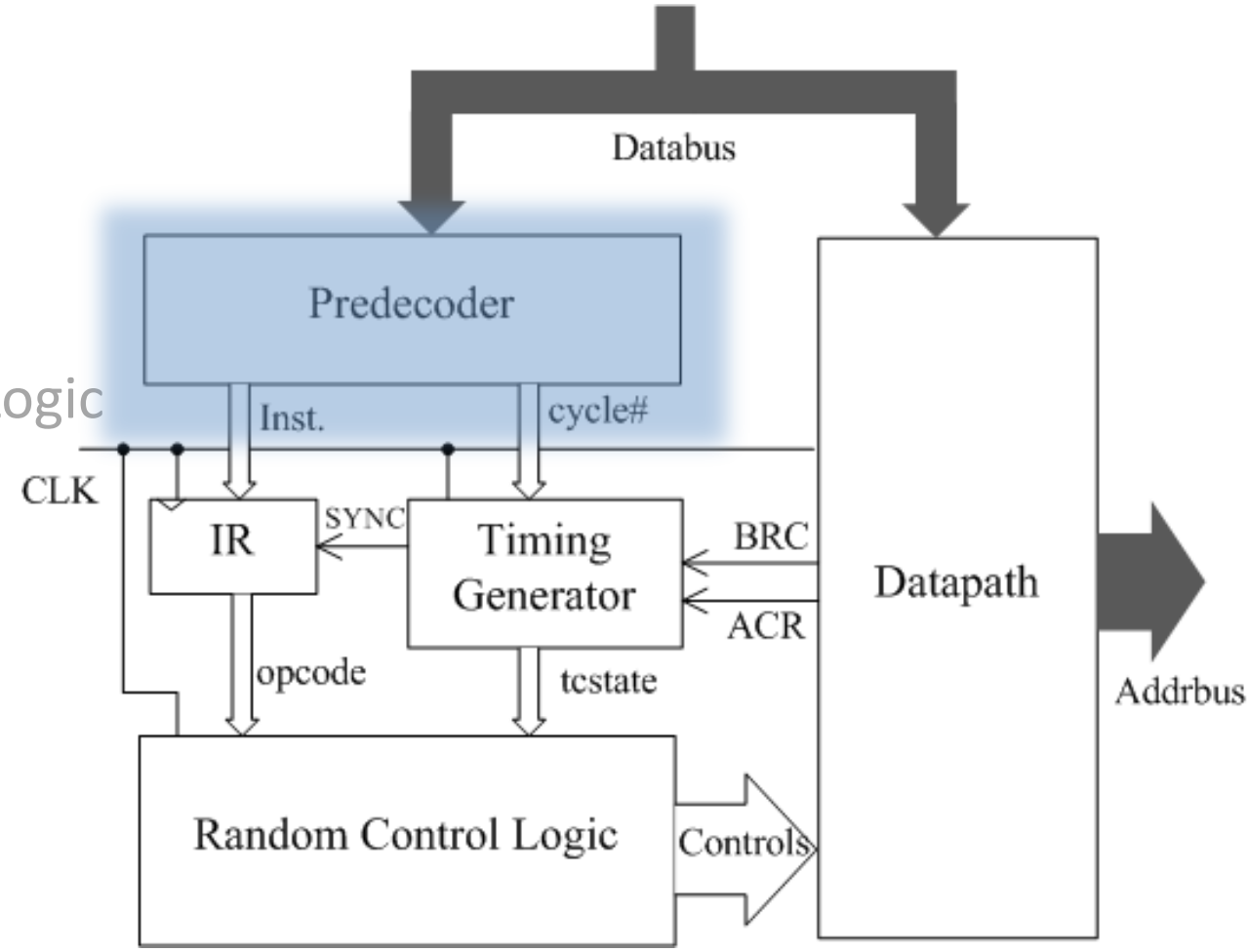
RTL level structure

- Control Path
  - Predecoder
  - Instruction Reg.
  - Timing Generator
  - Random Control Logic
- Datapath
  - Mux
  - Register File
  - ALU



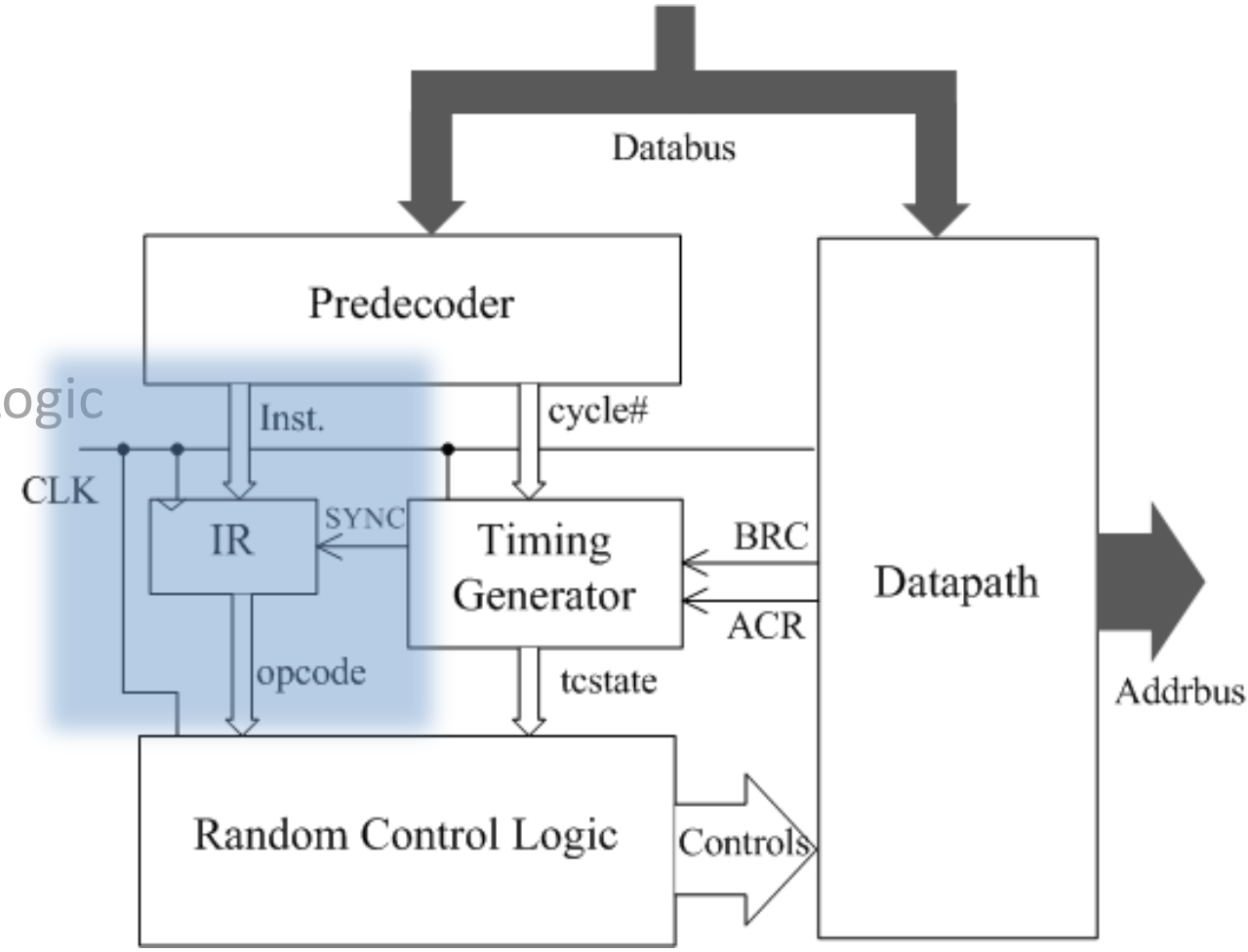
RTL level structure

- Control Path
  - Predecoder
  - Instruction Reg.
  - Timing Generator
  - Random Control Logic
- Datapath
  - Mux
  - Register File



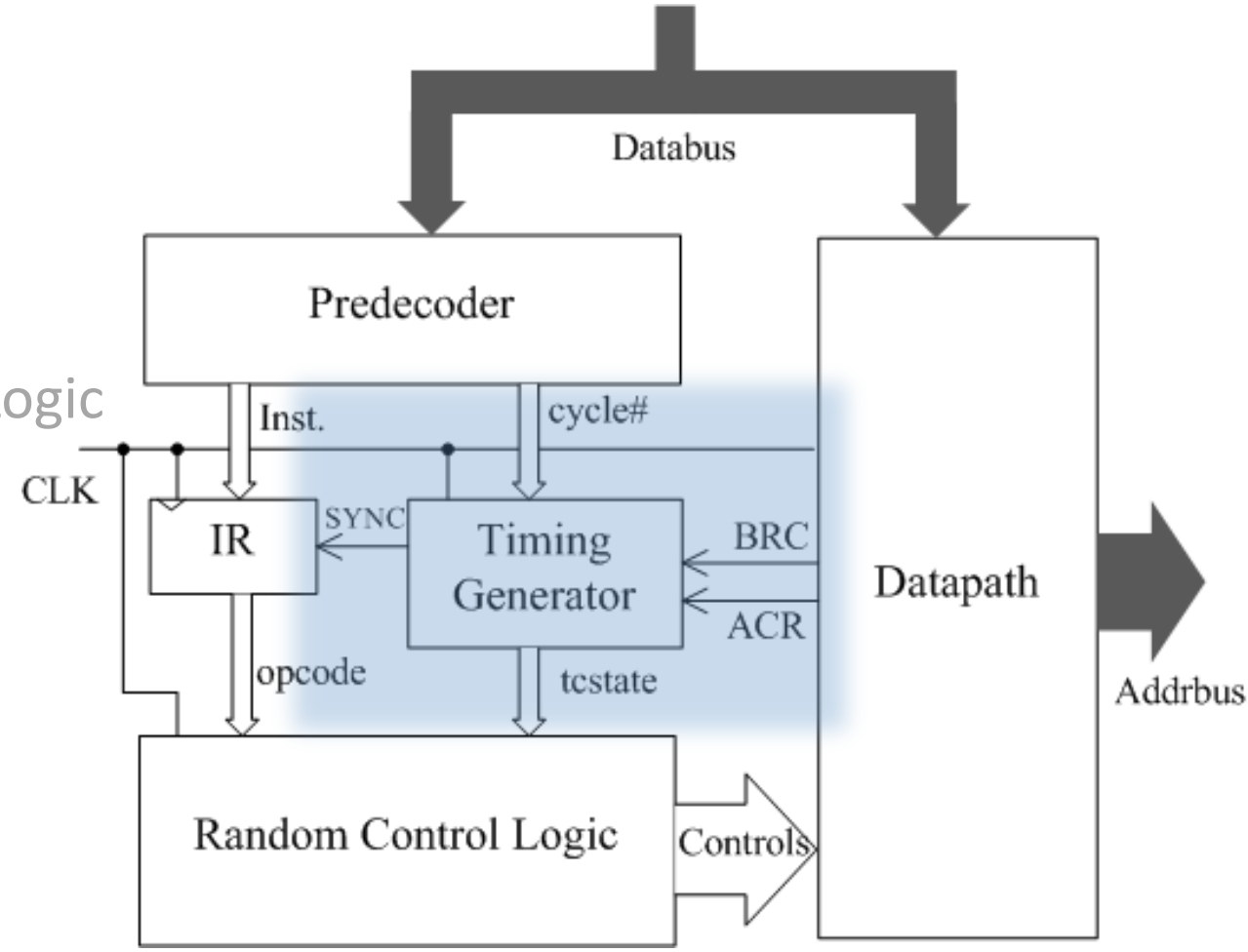
RTL level structure

- Control Path
  - Predecoder
  - Instruction Reg.
  - Timing Generator
  - Random Control Logic
- Datapath
  - Mux
  - Register File



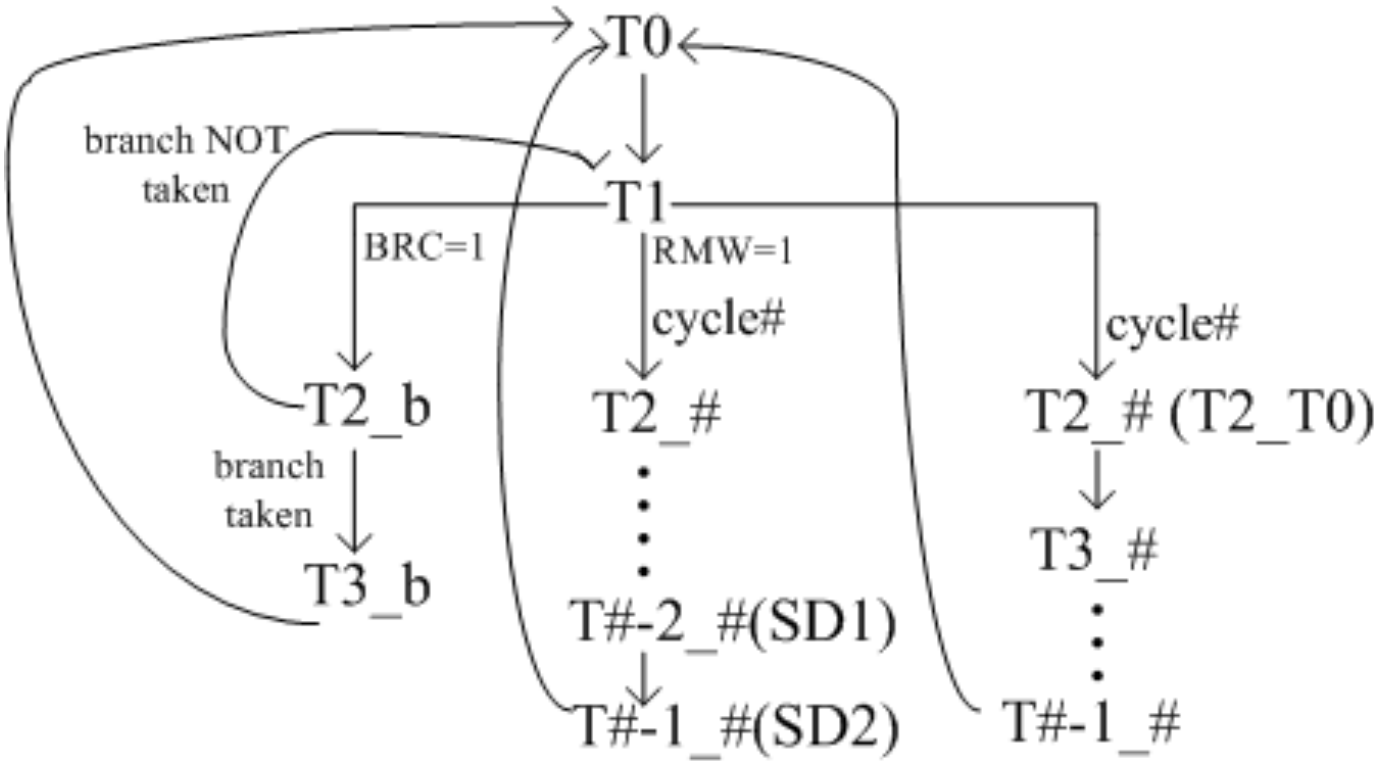
RTL level structure

- Control Path
  - Predecoder
  - Instruction Reg.
  - Timing Generator
  - Random Control Logic
- Datapath
  - Mux
  - Register File



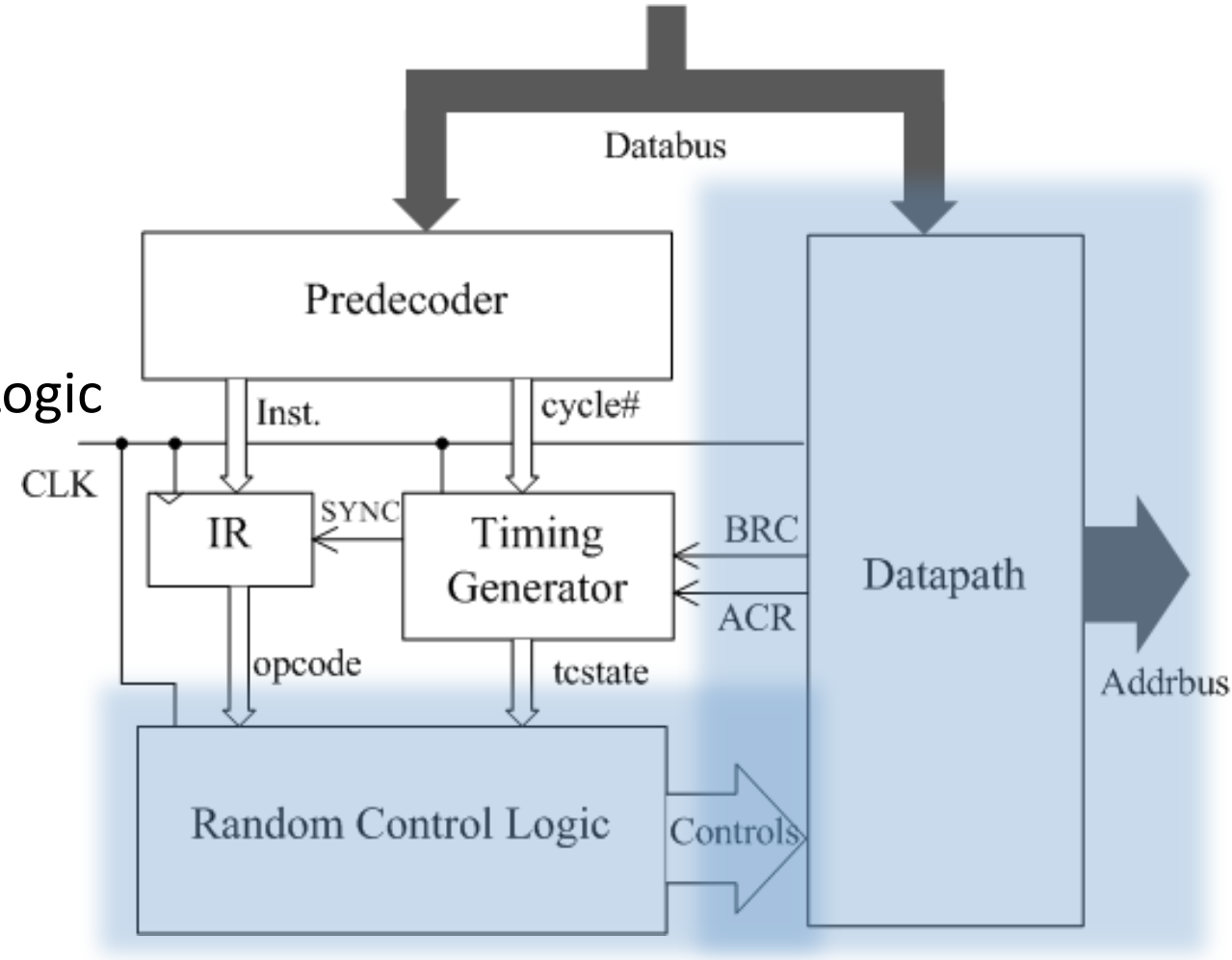
Timing Generator—Mealy Machine

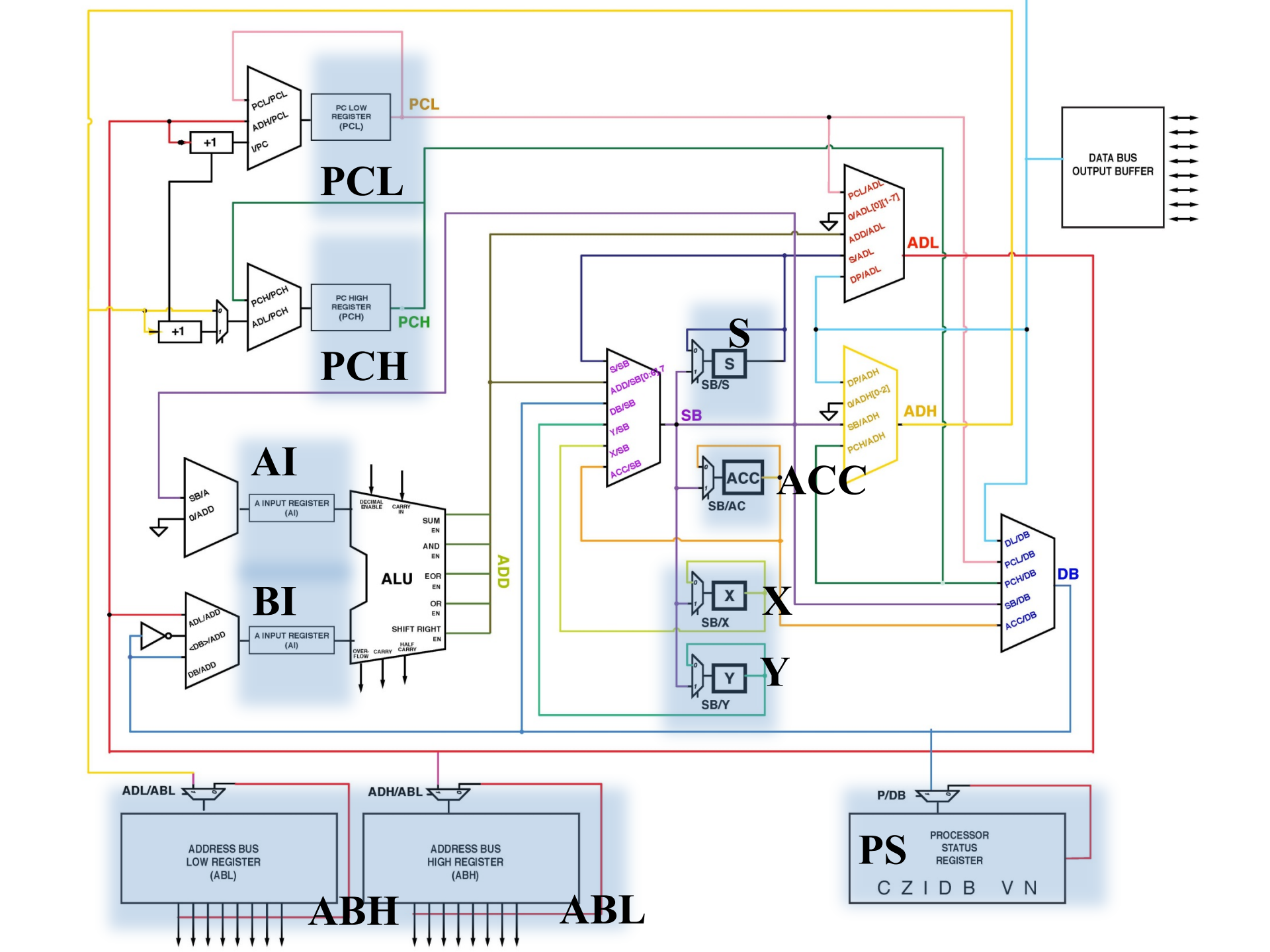
- Three types
  - Branch
  - RMW
  - Normal



RTL level structure

- Control Path
  - Predecoder
  - Instruction Reg.
  - Timing Generator
  - Random Control Logic
- Datapath
  - Mux
  - Register File




PCL
PCH
AI
BI
ALU
ADD
SB
ACC
X
Y
S
ADL
ADH
DB
ABH
ABL
PS
PCL/PCL
ADH/PCL
I/PC
PC LOW REGISTER (PCL)
PCH/PCH
ADL/PCH
PC HIGH REGISTER (PCH)
+1
SB/A
0/ADD
A INPUT REGISTER (AI)
ADL/ADD
<DB>/ADD
DB/ADD
DECIMAL ENABLE
CARRY IN
SUM
AND
EOR
OR
SHIFT RIGHT
OVER-FLOW
CARRY
HALF CARRY
S/SB
ADD/SB[0:6,7]
DB/SB
Y/SB
X/SB
ACC/SB
SB/S
SB/AC
SB/X
SB/Y
PCL/ADL
0/ADL[0][1-7]
ADD/ADL
S/ADL
DP/ADL
DP/ADH
0/ADH[0-2]
SB/ADH
PCH/ADH
DL/DB
PCL/DB
PCH/DB
SB/DB
ACC/DB
DATA BUS OUTPUT BUFFER
ADL/ABL
ADH/ABL
P/DB
ADDRESS BUS LOW REGISTER (ABL)
ADDRESS BUS HIGH REGISTER (ABH)
PROCESSOR STATUS REGISTER
C Z I D B V N

6502 Instruction Set Architecture

- Each instruction is encoded within 8 bits (62 total). Which uniquely defines the

  1. Operation to be performed (known as the '*opcode*').

  2. Addressing mode which defines how that operation handles and modifies memory.

Addressing modes

- 12 types of Addressing modes, but general addressing modes are:

000	(zero page,X)
001	zero page
010	#immediate
011	absolute
100	(zero page),Y
101	zero page,X
110	absolute,Y
111	absolute,X

- Each opcode can have multiple addressing modes. Taking this into account, there are 152 possible instructions.

General Instruction Format

Most Codes follow: AAABBBCC

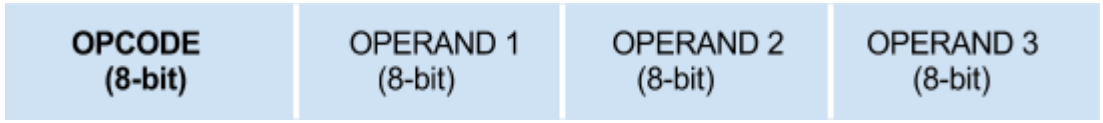
BBB: Defines the Addressing Mode

AAA---CC: Defines the Opcode

	00-Jan-00	01-Jan-00	10-Jan-00	11-Jan-00	09-Apr-00	10-Apr-00	19-Apr-00	20-Apr-00	26-Sep-02	27-Sep-02	06-Oct-02	07-Oct-02	04-Jan-03	05-Jan-03	14-Jan-03
0	BRK b	ORA (d,X)				ORA d	ASL d		PHP	ORA #	ASL A			ORA a	ASL a
1	BPL r	ORA (d),Y				ORA d,X	ASL d,X		CLC	ORA a,Y				ORA a,X	ASL a,X
10	JSR a	AND (d,X)			BIT d	AND d	ROL d		PLP	AND #	ROL A		BIT a	AND a	ROL a
11	BMI r	AND (d),Y				AND d,X	ROL d,X		SEC	AND a,Y				AND a,X	ROL a,X
100	RTI	EOR (d,X)				EOR d	LSR d		PHA	EOR #	LSR A		JMP a	EOR a	LSR a
101	BVC r	EOR (d),Y				EOR d,X	LSR d,X		CLI	EOR a,Y				EOR a,X	LSR a,X
110	RTS	ADC (d,X)				ADC d	ROR d		PLA	ADC #	ROR A		JMP (a)	ADC a	ROR a
111	BVS r	ADC (d),Y				ADC d,X	ROR d,X		SEI	ADC a,Y				ADC a,X	ROR a,X
1000		STA (d,X)			STY d	STA d	STX d		DEY		TXA		STY a	STA a	STX a
1001	BCC r	STA (d),Y			STY d,X	STA d,X	STX d,Y		TYA	STA a,Y	TXS			STA a,X	
1010	LDY #	LDA (d,X)	LDX #		LDY d	LDA d	LDX d		TAY	LDA #	TAX		LDY a	LDA a	LDX a
1011	BCS r	LDA (d),Y			LDY d,X	LDA d,X	LDX d,Y		CLV	LDA a,Y	TSX		LDY a,X	LDA a,X	LDX a,Y
1100	CPY #	CMP (d,X)			CPY d	CMP d	DEC d		INY	CMP #	DEX		CPY a	CMP a	DEC a
1101	BNE r	CMP (d),Y				CMP d,X	DEC d,X		CLD	CMP a,Y				CMP a,X	DEC a,X
1110	CPX #	SBC (d,X)			CPX d	SBC d	INC d		INX	SBC #	NOP		CPX a	SBC a	INC a
1111	BEQ r	SBC (d),Y				SBC d,X	INC d,X		SED	SBC a,Y				SBC a,X	INC a,X

Operands

Most operations (except for implied instructions such as CLC, DEX, INY, etc…) accept up to three operands which occupy the following adjacent bits in memory.



Access to these operands and to internal and external memory is defined by a Mealy State Machine.

Debugging

- 1. Software debug: Modelsim
  - Timed databus behavior

    Ex) Databus<=x”A9”; wait for 40ns;

    Databus<=x”07”; wait for 40ns;
  - Tested ~50% of opcodes, one by one
  - Does not test data fetch, or memory structure
  - Quick debug.

Debugging

- 2. Hardware debug
  - Real time
  - Tests full 6502 implementation
  - Limited number of output display routes
    - A, X, Y, and flags
  - SignalTap II Logic Analyzer
    - Real time.
    - Slower clock used for debug.

Debugging

- Common bugs
  - SUMS, I_ADDC not turned off after next cycle
  - PC incrementing at wrong cycle
  - Resetting the initialization ROM

  Surprisingly few errors regarding:
  - Mask overlap
  - Edge-triggered / level-sensitive misbehavior
  - Mealy machine

  => Our ~500 timing diagrams did pay off!

Bouncing Ball

- In code: 8 statements

```
Define: X, Y, sizeX, sizeY, dirX, dirY

If X = sizeX
    dirX = 0;
end
If X = 0
    dirX = 1;
end
while clock and dirX=0
    X = X-1;
end
while clock and dirX=1
    X = X+1;
end
```

```
If Y = sizeY
    dirY = 0;
end
If Y = 0
    dirY = 1;
end
while clock and dirY=0
    Y = Y-1;
end
while clock and dirY=1
    Y = Y+1;
end
```

Bouncing Ball

- In assembly language: 12 branches, 50 lines

```
--INIT
LDX #$00                A2
LDY #$00                A0
LDA #$06 --sizeX        A9
STA $0070               8D
LDA #$04 --sizeY        A9
STA $0071               8D
LDA #$01 --dirX         A9
STA $0072               8D
LDA #$01 --dirY         A9
STA $0073               8D

--BRC1
LDA $0072               AD
CMP #$01                C9
BEQ (BRC3)              F0
BNE (BRC4)              D0

--BRC2
LDA $0073               AD
CMP #$01                C9
BEQ (BRC5)              F0
BNE (BRC6)              D0

--BRC3
INX                     E8
CPX $0070               EC
BEQ (BRC7)              F0
BNE (BRC2)-(BRC12)      D0

--BRC4
DEX                     CA
CPX #$00                E0
BEQ (BRC8)              F0
BNE (BRC2)-(BRC12)      D0

--BRC5
INY                     C8
CPY $0071               CC
BEQ (BRC9)              F0
BNE (BRC1)-(BRC11)      D0

--BRC6
DEY                     88
CPY #$00                C0
BEQ (BRC10)             F0
BNE (BRC1)-(BRC11)      D0

--BRC7
LDA #$00                A9
STA $0072               8D
JMP (BRC2)              4C

--BRC8
LDA #$01                A9
STA $0072               8D
JMP (BRC2)              4C

--BRC9
LDA #$00                A9
STA $0073               8D
JMP (BRC1)              4C

--BRC10
LDA #$01                A9
STA $0073               8D
JMP (BRC1)              4C

--BRC11
JMP (BRC1)              4C

--BRC12
JMP (BRC2)              4C
```

Bouncing Ball

- In the ROM: 98 bytes of ROM

```
constant ROM : rom_type :=
(

    x"A2", x"00", x"A0", x"00", x"A9", x"ff", x"8D", x"70", x"00", x"A9", x"df", x"8D", x"71", x"00", --Init1/2 (24, x18)
    x"A9", x"01", x"8D", x"72", x"00", x"A9", x"01", x"8D", x"73", x"00", --Init2/2 (24, x18)
    x"AD", x"72", x"00", x"C9", x"01", x"F0", x"0B", x"D0", x"11", --B1: 0018 (9) --B3 and B4
    x"AD", x"73", x"00", x"C9", x"01", x"F0", x"11", x"D0", x"17", --B2: 0021 (9) --B5 and B6
    x"E8", x"EC", x"70", x"00",        x"F0", x"18", x"D0", x"39", --B3: 002A (8) --B7 and B12
    x"CA", x"E0", x"00",               x"F0", x"19", x"D0", x"32", --B4: 0032 (7) --B8 and B120
    x"C8", x"CC", x"71", x"00",        x"F0", x"19", x"D0", x"23", --B5: 0039 (8) --B9 and B11
    x"88", x"C0", x"00",               x"F0", x"1A", x"D0", x"20", --B6: 0041 (7) --B10 and B110
    x"A9", x"00", x"8D", x"72", x"00", x"4C", x"21", x"00", --B7: 0048 (8) --J2
    x"A9", x"01", x"8D", x"72", x"00", x"4C", x"21", x"00", --B8: 0050 (8) --J2
    x"A9", x"00", x"8D", x"73", x"00", x"4C", x"18", x"00", --B9: 0058 (8) --J1
    x"A9", x"01", x"8D", x"73", x"00", x"4C", x"18", x"00", --B10: 0060 (8) --J1
    x"4C", x"18", x"00", --B11: 0068 (3) --J1
    x"4C", x"21", x"00", --B12: 006B (3) --J2
    x"ff", x"ff", --6
    x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", --7
    x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", --8
    x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", --9
    x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", --10
    x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", --11
    x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", --12
    x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", --13
    x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", --14
    x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00", x"00" --15

    --B1: 0018
    --B2: 0021
    --B3: 002A
    --B4: 0032
    --B5: 0039
    --B6: 0041
    --B7: 0048
    --B8: 004F
    --B9: 0056
    --B10: 005D
    --B11: 0064
    --B12: 0066
```