Topic 4

Processor Architectures

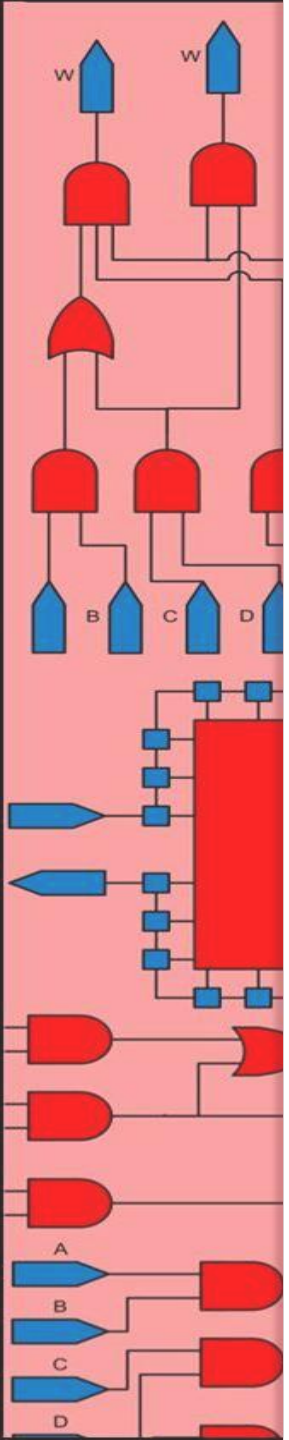
Zain Navabi

Slides prepared by Katayoon Basharkhah

CYBERSECURITY NATIONAL LABORATORY
CYBER CHALLENGE.IT

Course Roadmap

	Topic 1	Topic2	Topic 3	Topic 4
Goals	A Processing System	Memory Model	Processor Devices	Processor Architecture
	Embedded System Introduction	Peripheral Hardware		Show Case ISA
Contents	•CPU •Instructions •Memory-Map •Memory •I/O device	Different memory structures Memory mapping in the bussing and devices' communications.		• SAYAC Instructions • Datapath Design • Controller Design • Simulation and FPGA Synthesis
	Introduction to processor-based system, general concepts and different components of the system	•SDRAM •ROM •SRAM	• Address Logic • Data Registers • Handshaking • Interrupt	A sample processor named SAYAC. Processor internal architecture starting form the processor instruction set and coming down to the all RTL components of the processor datapath and controller.

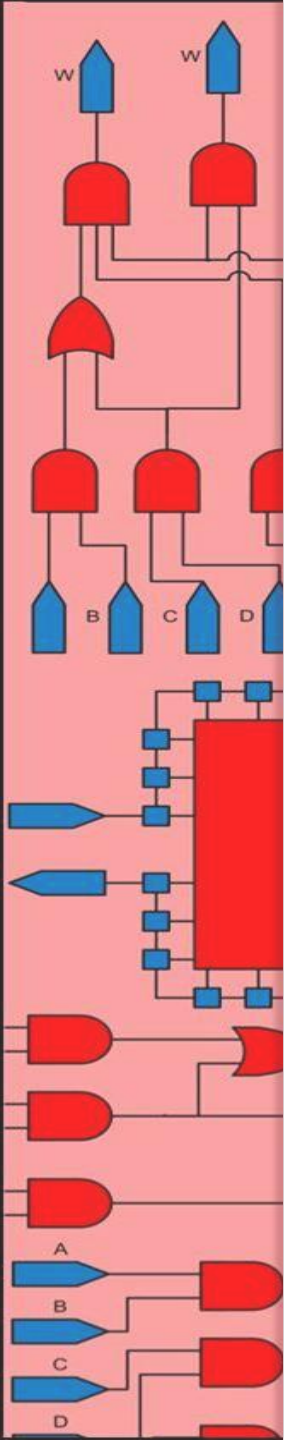


Processor Architecture

Learning Outcomes:

- Processor – Memory interface
- Instruction Execution Flow
- Processor Registers
- Instruction Format and Addressing Modes
- SAYAC Instructions
- Design of SAYAC
- Datapath and Controller
- Description in VHDL
- Memory Model and Testbench

Simple Architecture Yet Ample Circuitry



Processor Architecture

Outline:

- **Processor and Memory Model**
- Processor Model Specification
- Instruction Execution Cycle
- Processor Registers
- Instruction Format
- SAYAC Instructions

D1

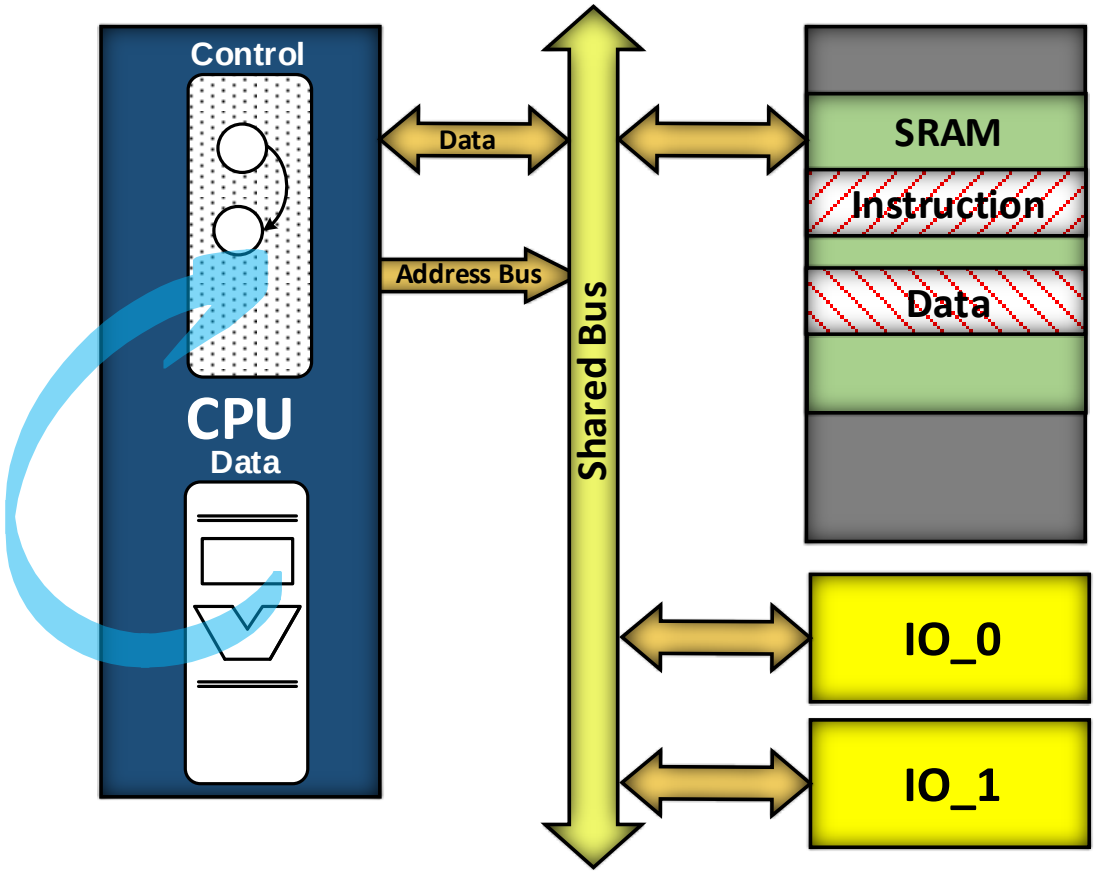
- Design of SAYAC
  - SAYAC Datapath and Instruction flow
  - Controller
  - Datapath VHDL Description
    - Control Signals
    - Components of Datapath
    - Bussing
  - Controller VHDL Description
  - Putting SAYAC Together
  - Memory Model
  - Instruction execution and Testing
  - Conclusions

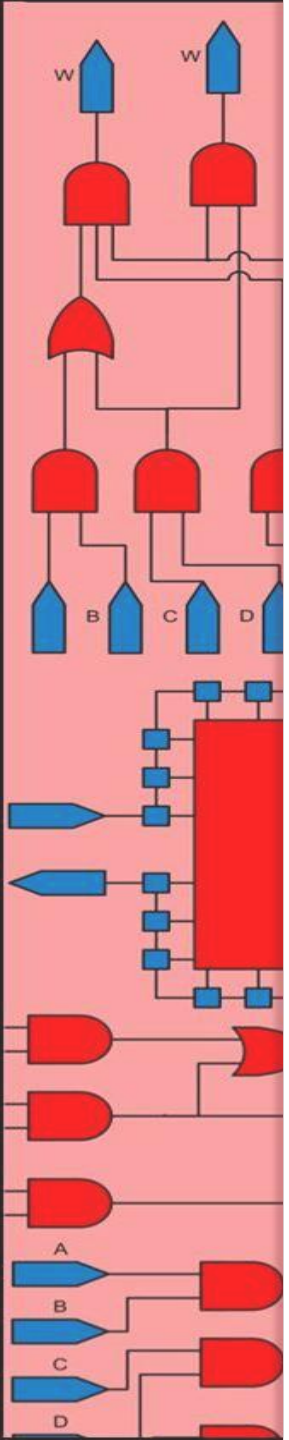
D2

Processor and Memory Model

Von Neumann Process Model

- A memory, containing instructions and data
- A data unit, for performing arithmetic and logical operations
- A control unit, for interpreting instructions



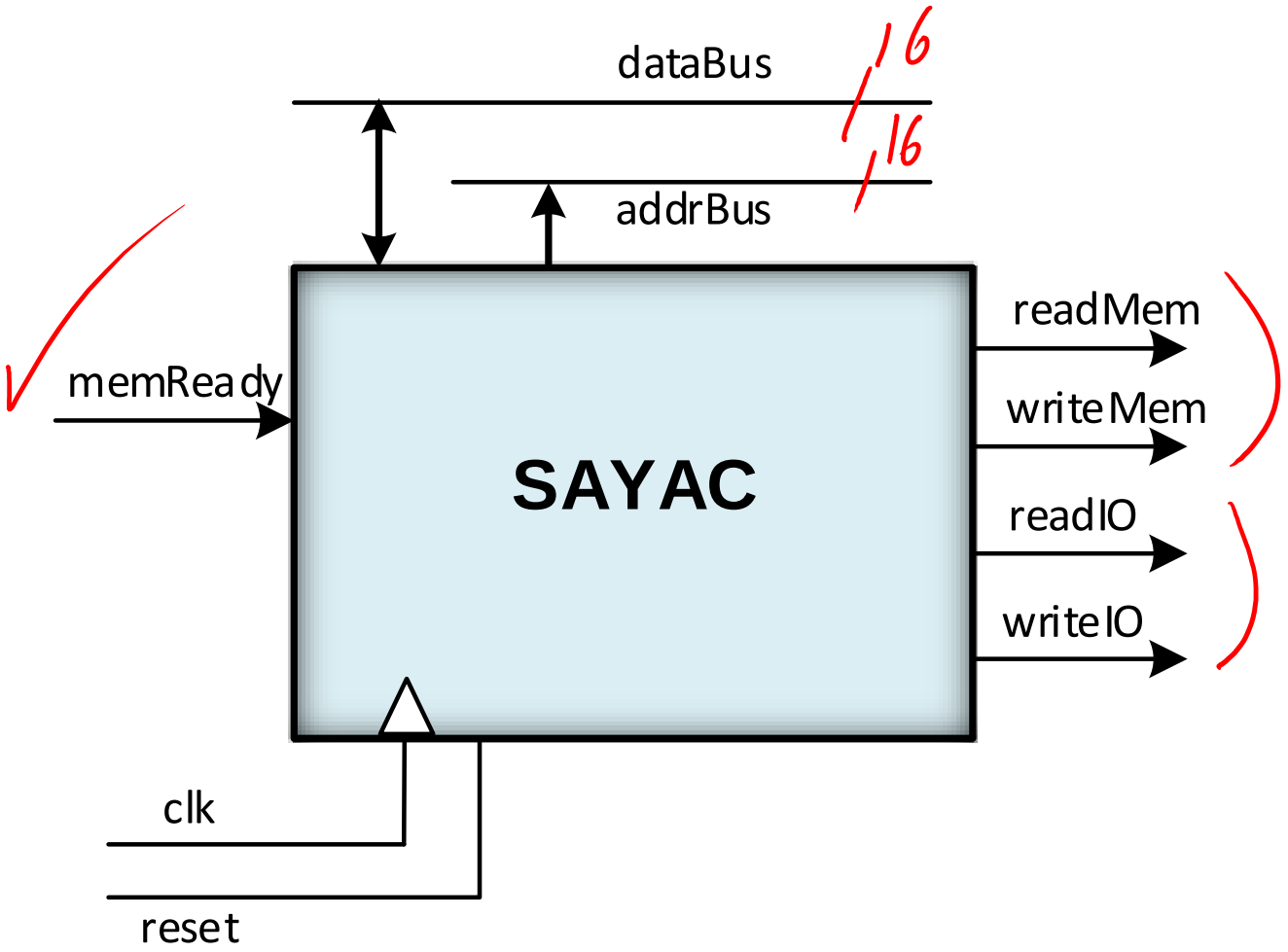


Processor Architecture

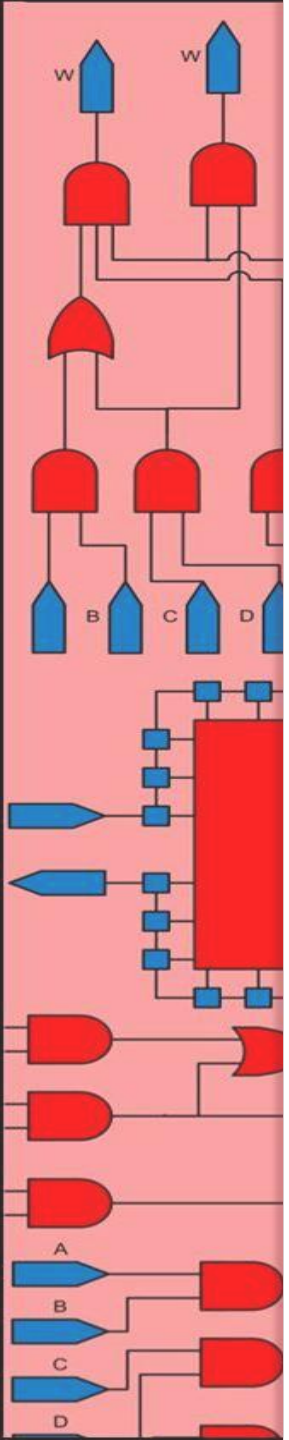
Outline:

- Processor and Memory Model
- **Processor Model Specification**
- Instruction Execution Cycle
- Processor Registers
- Instruction Format
- SAYAC Instructions

Processor Model Specification



Simple **A**rchitecture **Y**et **A**mple **C**ircuitry

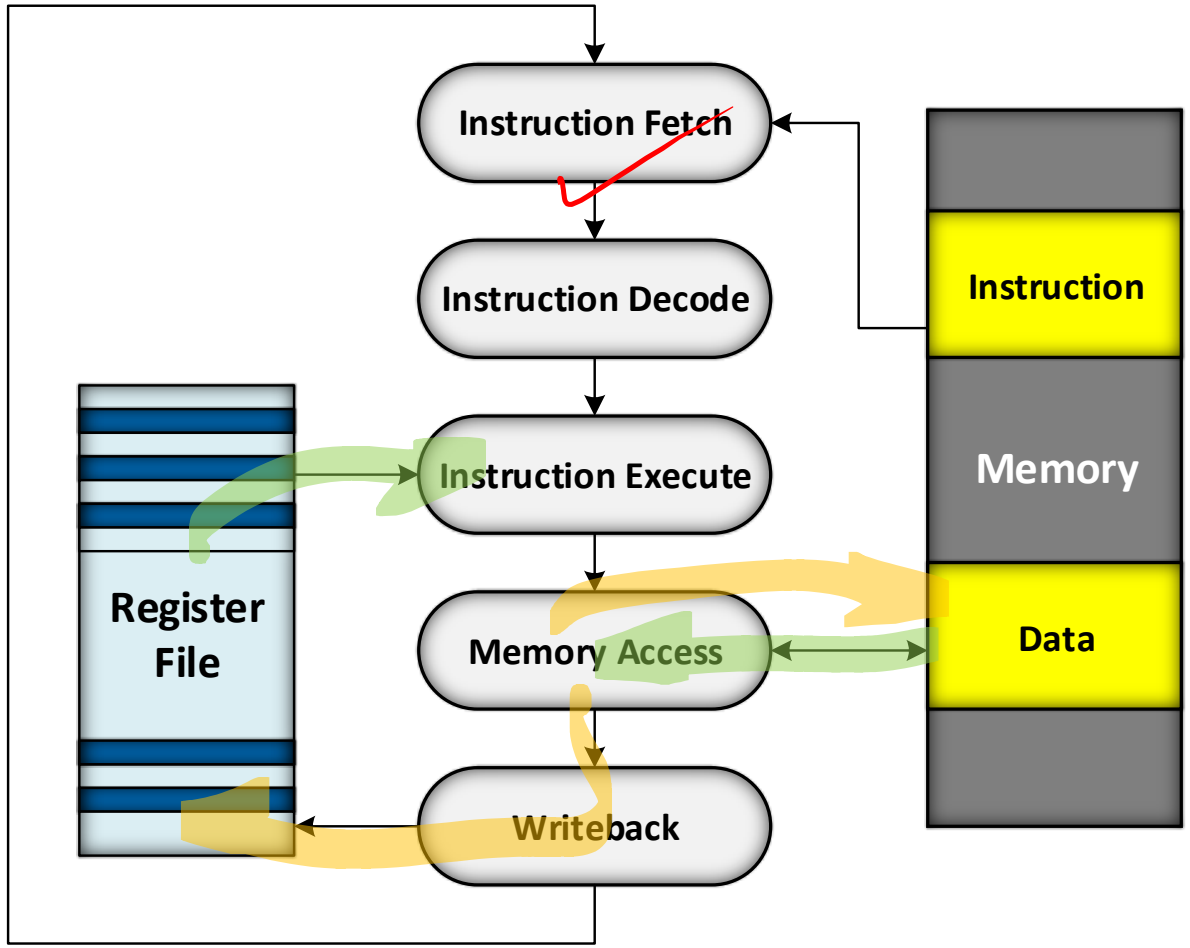


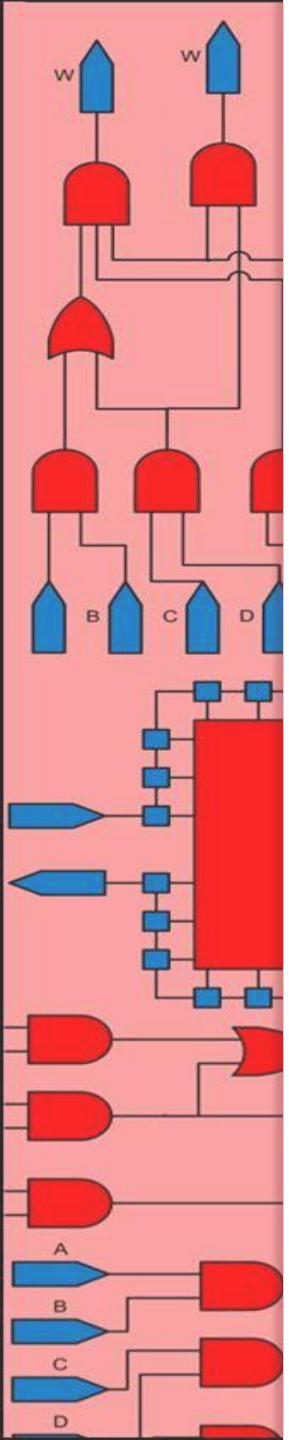
Processor Architecture

Outline:

- Processor and Memory Model
- Processor Model Specification
- **Instruction Execution Cycle**
- Processor Registers
- Instruction Format
- SAYAC Instructions
- Design of SAYAC

Instruction Execution Cycle



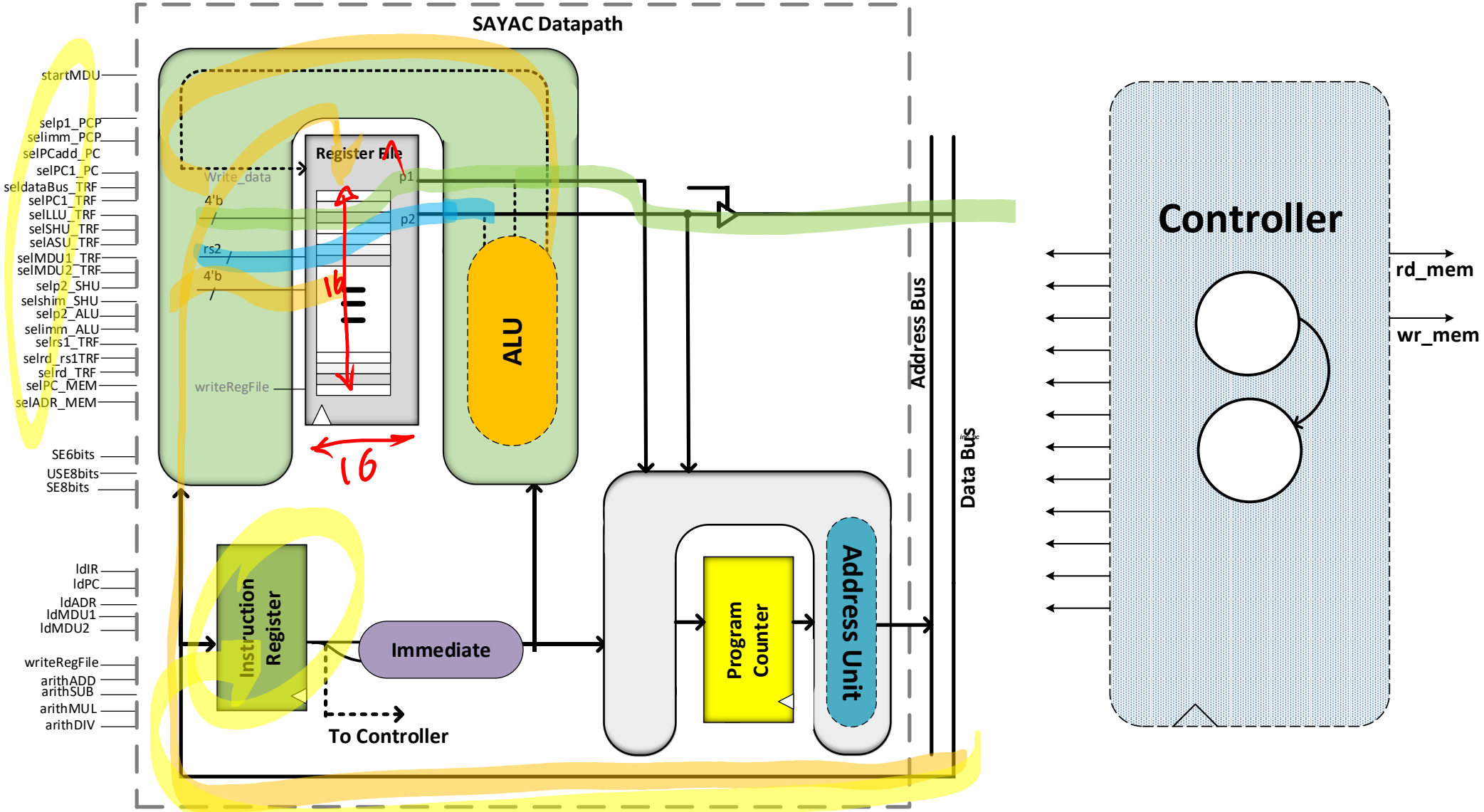


Processor Architecture

Outline:

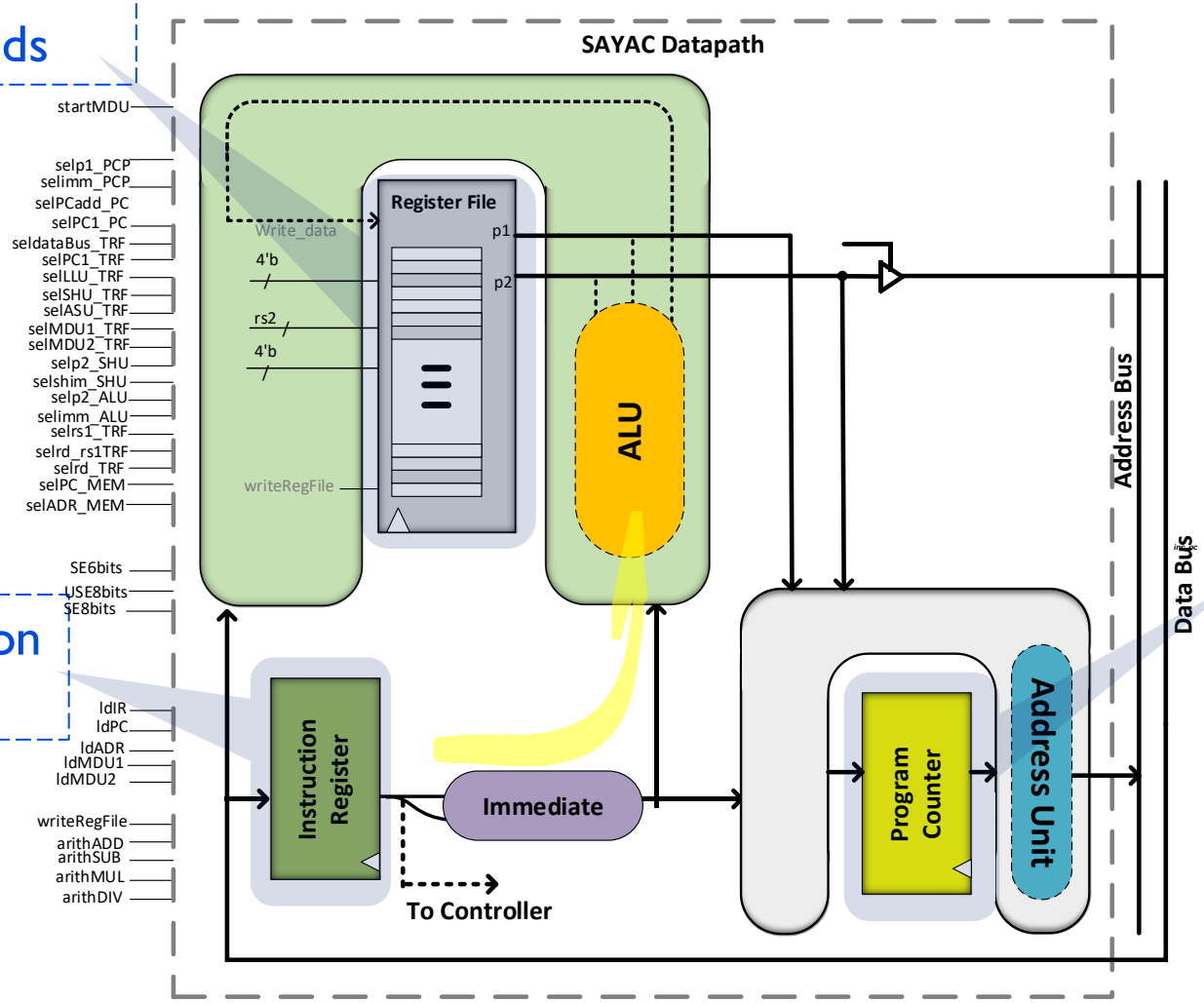
- Processor and Memory Model
- Processor Model Specification
- Instruction Execution Cycle
- **Processor Registers**
- Instruction Format
- SAYAC Instructions
- Design of SAYAC

SAYAC Architecture



Processor Registers

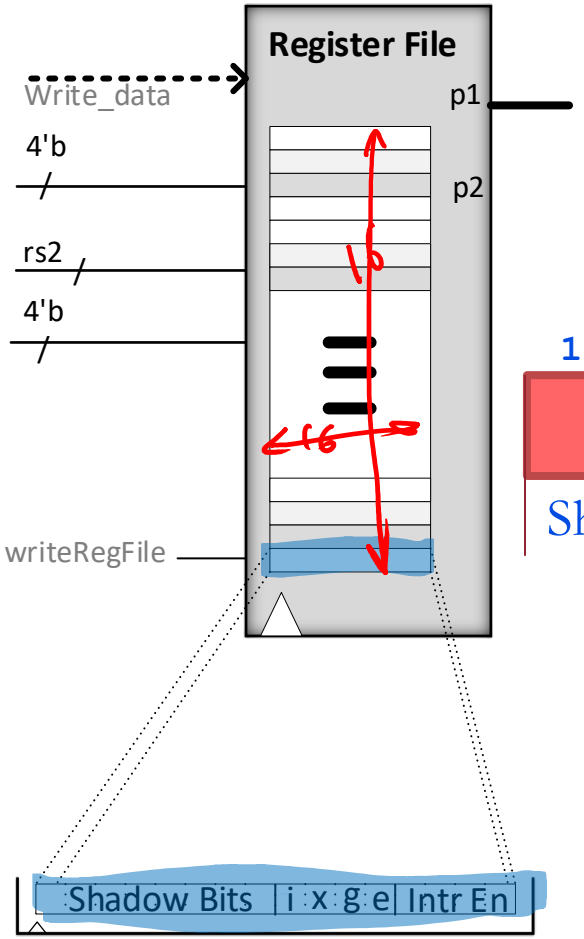
General purpose registers containing operands

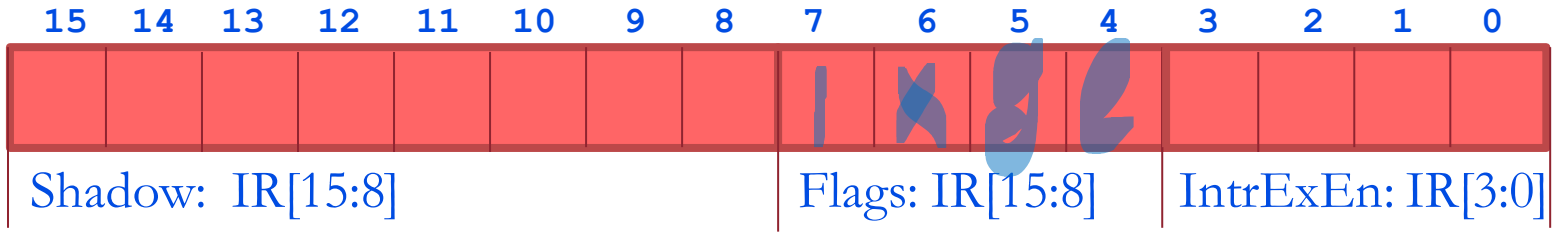


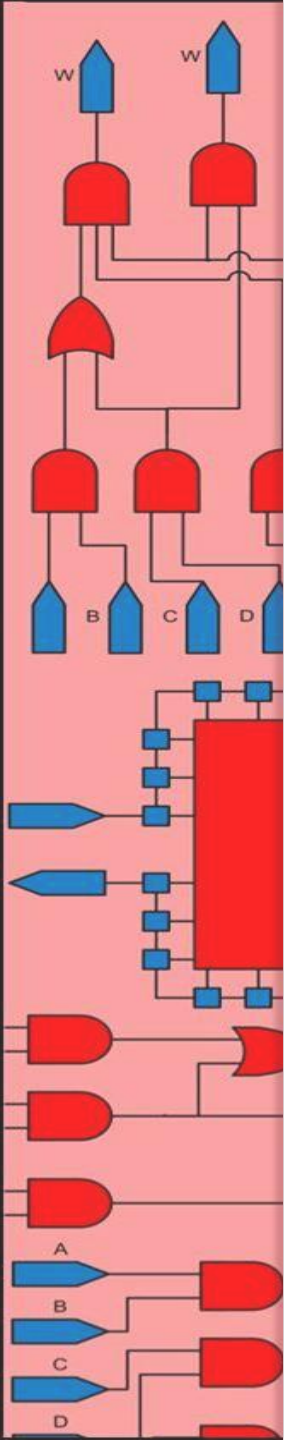
Contains the instruction being executed

Contains next instruction address

Shadow, Flags, and Interrupt and Exception







Processor Architecture

Outline:

- Processor and Memory Model
- Processor Model Specification
- Instruction Execution Cycle
- Processor Registers
- **Instruction Format**
- SAYAC Instructions

Instruction Format

Instruction Classes:

- Data Transfer Instructions
- Arithmetic / Logical Instructions
- Control Flow Instructions

Instruction	Notation
LDR	.rd. <= [.rs1.]
	.rd. <= { .rs1. }
STR	[.rd.] <= .rs1.
	{ .rd. } <= .rs1.
JMR	PC <= PC + .rs1. .rd. <= PC + 1 if s = 1
JMI	PC <= PC + "imm" .rd. <= PC + 1
ANR	.rd. <= .rs1. AND .rs2.
ANI	.rd. <= .rd. AND USE"imm"
MSI	.rd. <= SE"imm"
MHL	.rd. [15:8] <= "imm"
SIR	.rd. <= .rs1. LS± .rs2.
SAR	.rd. <= .rs1. AS± .rs2.
ADR	.rd. <= .rs1. + .rs2.
SUR	.rd. <= .rs1. – .rs2.
ADI	.rd. <= .rd. + SE"imm"
SUI	.rd. <= .rd. - SE"imm"
MUL	.rd. <= .rs1. × .rs2. 'LSB .rd+1. <= .rs1. × .rs2. 'MSB
DIV	.rd. <= .rs1. ÷ .rs2. 'Quo .rd+1. <= .rs1. ÷ .rs2. 'Rem
CMR	flags <= Cmp(.rs1. , .rd.)
CMI	flags <= Cmp(.rd. , SE"imm")
BRC	PC <= .rd. if flag
BRR	PC <= PC + .rd. if flag
SHI	.rd. <= .rd. LS± "shim"
	.rd. <= .rd. AS± "shim"
NTR	.rd. <= 1sComp(.rs1.)
	.rd. <= 2sComp(.rs1.)
NTD	.rd. <= 1sComp(.rd.)
	.rd. <= 2sComp(.rd.)

Nomenclature:

.rsd. → R_f content pointed by *rsd*

[*adr*] → Memory addressed by *adr*

{ *loc* } → IO device addressed by *loc*

SE"imm, USE"imm" → Signed, Unsigned Extension of "imm"

.rsd.
16 bit
of regfile
pointed
by 4 bit
rsd

Instruction Format

Addressing Modes (In General)

1- Direct Addressing

2- Indirect Addressing

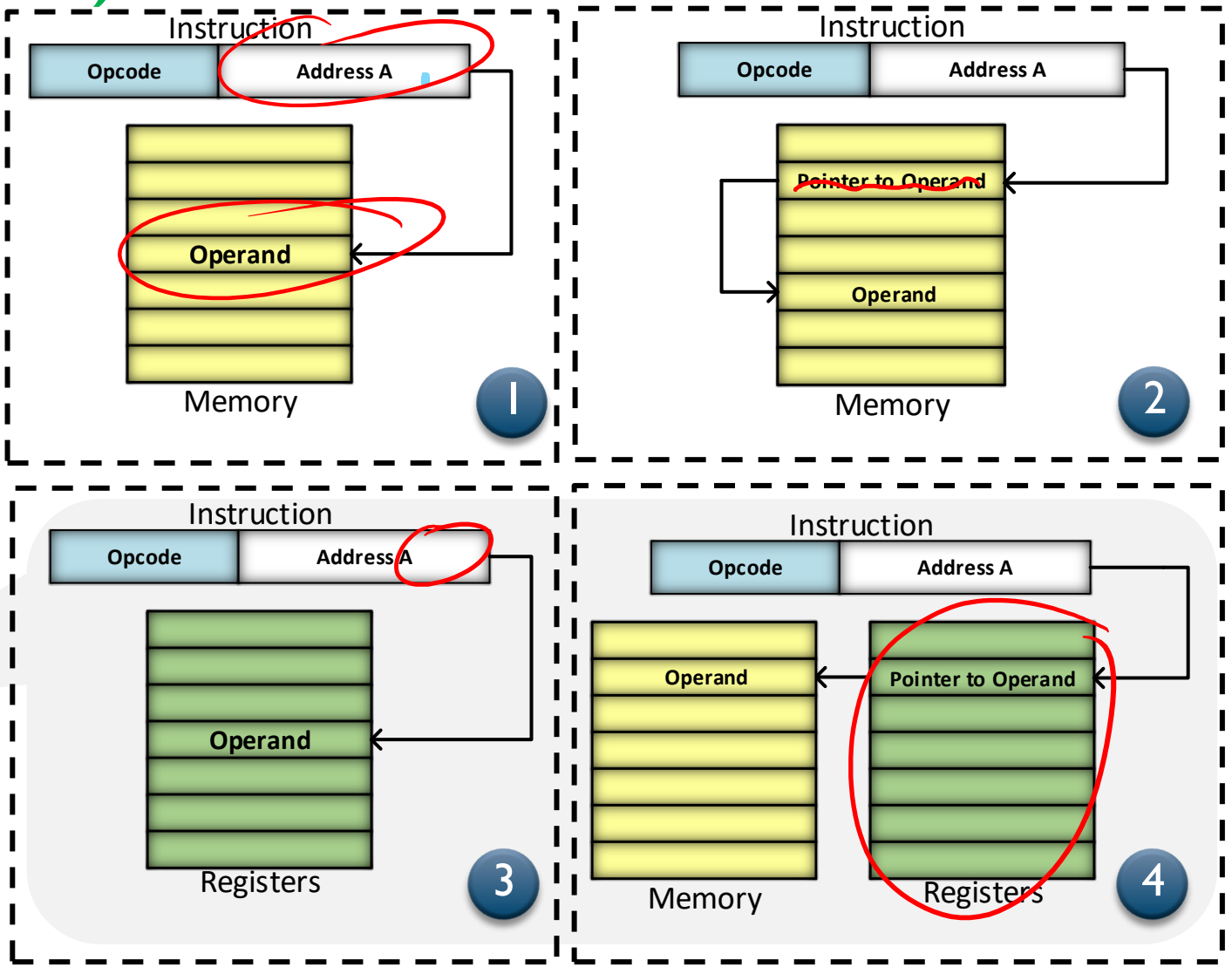
3- Register Addressing

4- Register Indirect Addressing

No or less Memory Access

Faster Execution

Limited Address Space

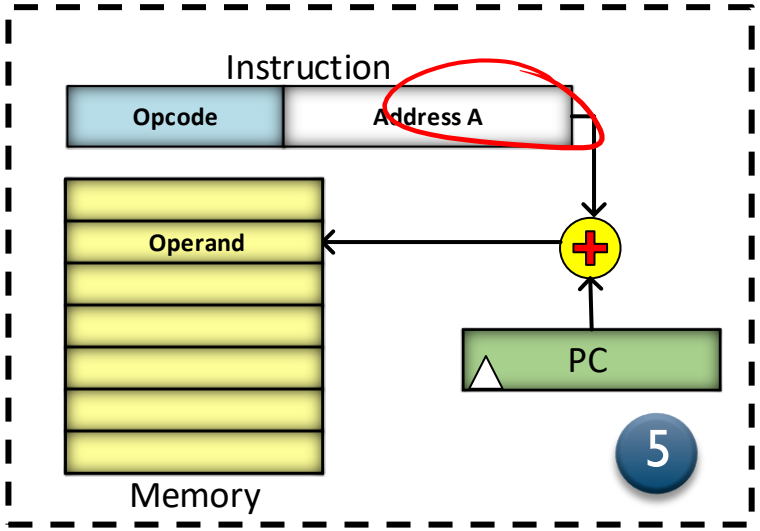


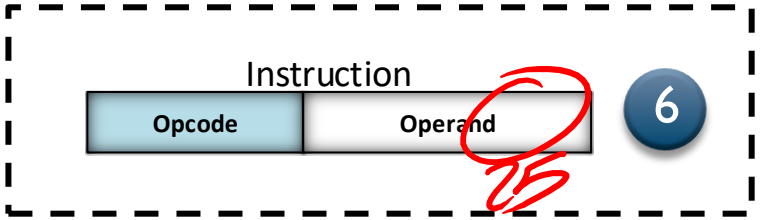
Instruction Format

Addressing Modes (In General)

5- PC Relative Addressing

6- Immediate Addressing





Instruction Format

Instruction Types:

- R-Type Instruction
  - Contains registers for operands and addressing
- I-Type Instruction
  - Contains an immediate value embedded within the instruction word

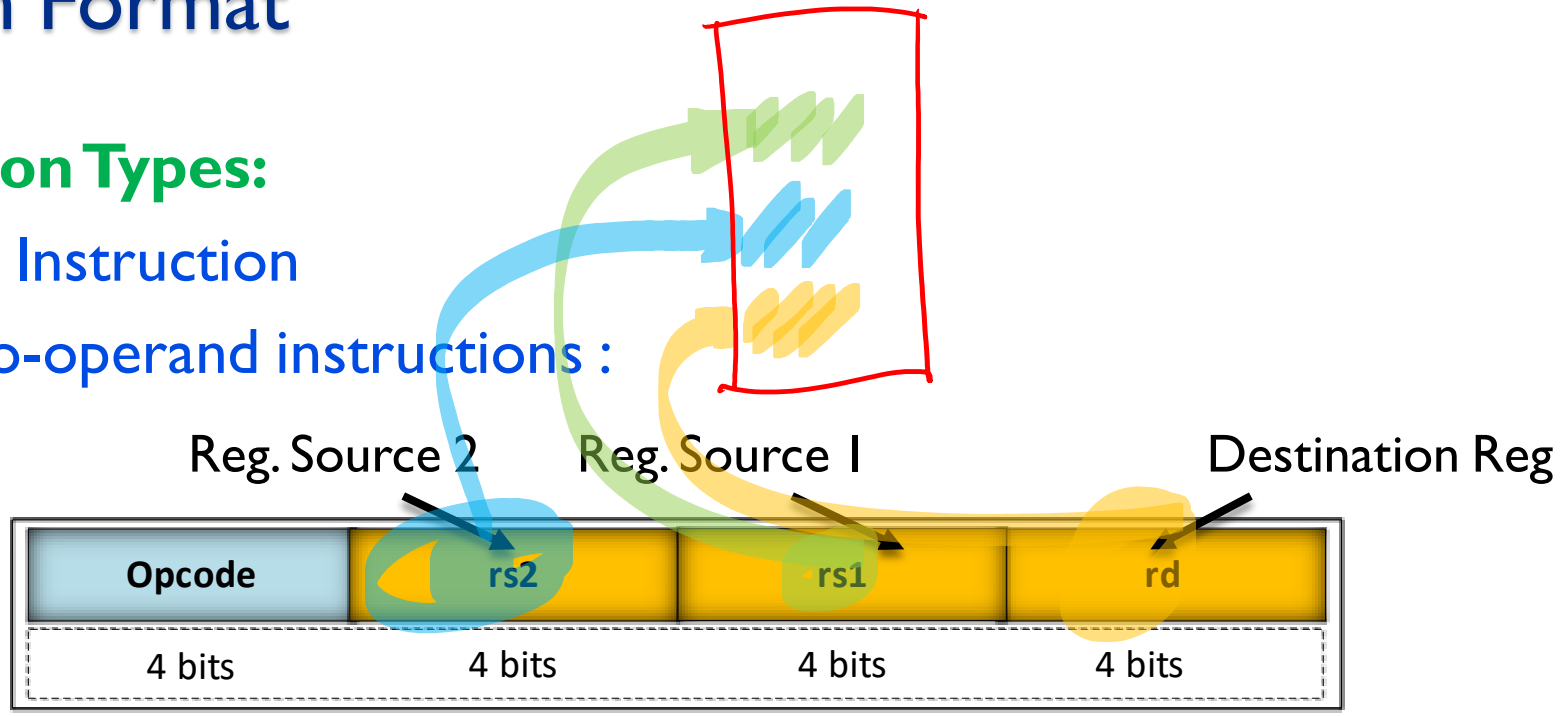
Opcode	Immediate or register
4 bits (Up to 8-bits)	12-8 bits

For instructions with alternatives opcode can be extended up-to 8 bits

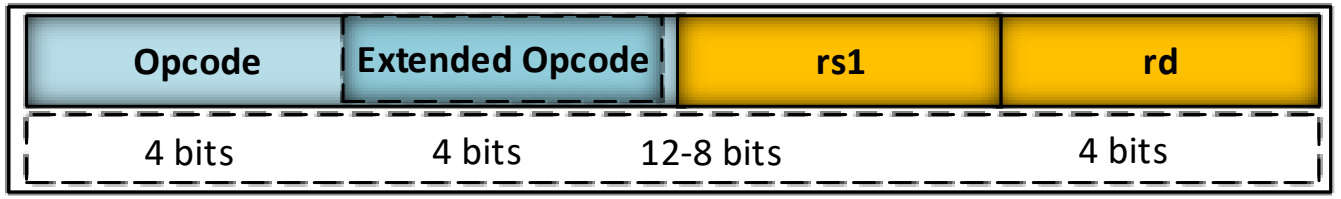
Instruction Format

Instruction Types:

- R-Type Instruction
  - Two-operand instructions :



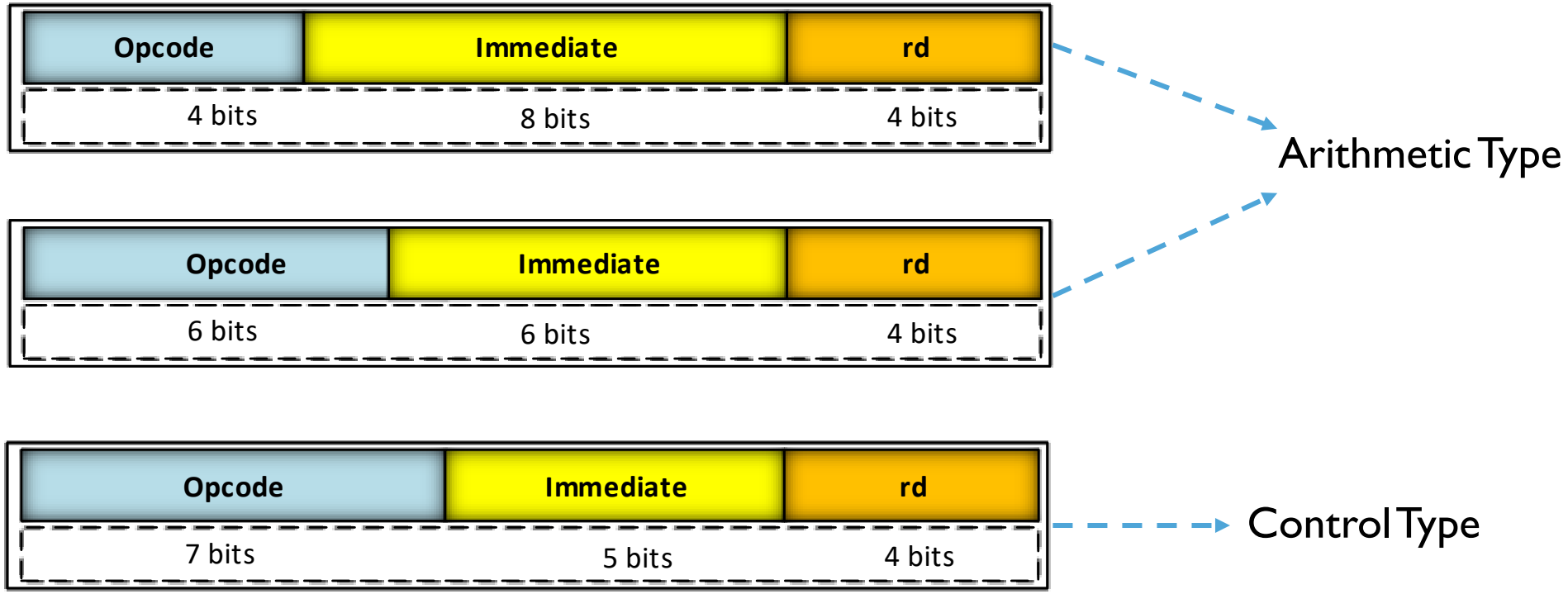
  - One-operand instructions :

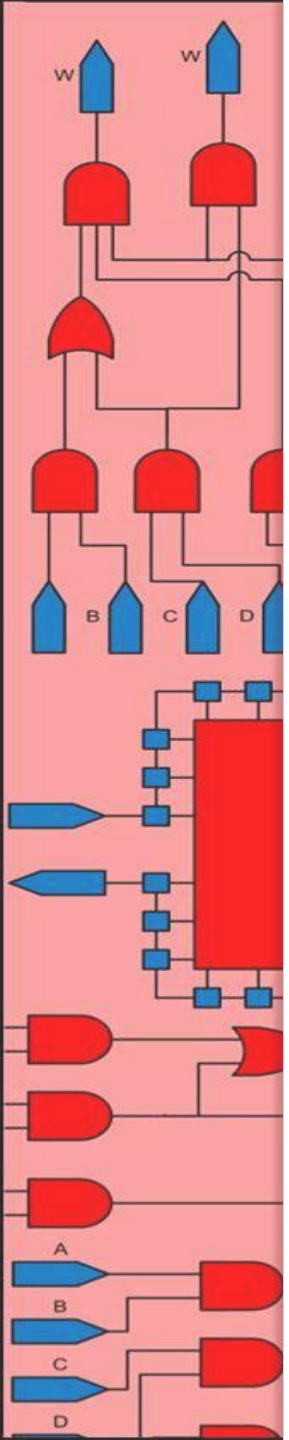


Instruction Format

Instruction Types:

- I-Type Instruction





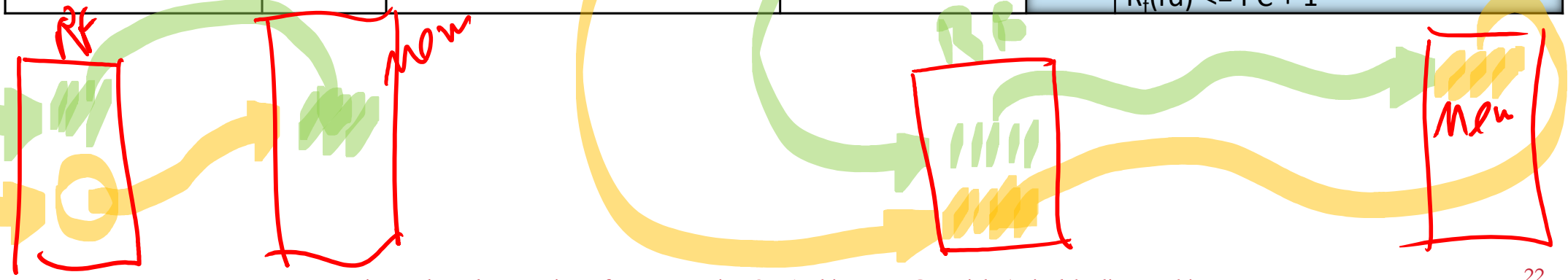
Processor Registers

Outline:

- Processor and Memory Model
- Processor Model Specification
- Instruction Execution Cycle
- Processor Registers
- Instruction Format
- **SAYAC Instructions**
  - **Basic**
  - **Shadow**

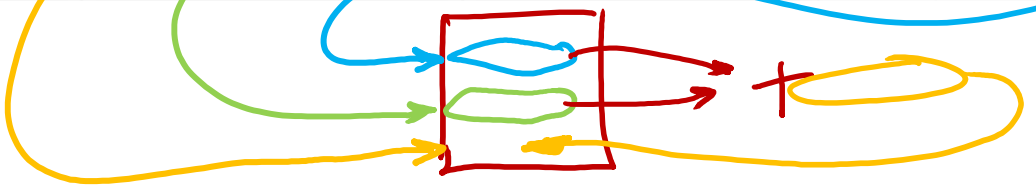
SAYAC Instructions - Basic

[15:12]	[11:10]	[9]	[8]	[7:4]	[3:0]	Instruction	Notation
0000	Reserved						
0001	Reserved						
0010	00	0		rs1	rd	LDR	R_f(rd) <= MEM(R_f(rs1))
		1					R_f(rd) <= IO(R_f(rs1))
	01	0		rs1	rd	STR	MEM(R_f(rd)) <= R_f(rs1)
		1					IO(R_f(rd)) <= R_f(rs1)
	10	s		rs1	rd	JMR	PC <= PC + R_f(rs1) R_f(rd) <= PC + 1, IF s=1
	11	imm			rd	JMI	PC <= PC + "imm" R_f(rd) <= PC + 1



SAYAC Instructions - Basic

[15:12]	[11:8]	[7:4]	[3:0]	Instruction	Notation
0011	rs2	rs1	rd	ANR	$R_f(rd) <= R_f(rs1)$ AND $R_f(rs2)$
0100	imm		rd	ANI	$R_f(rd) <= R_f(rd)$ AND USE"imm"
0101	imm		rd	MSI	$R_f(rd) <=$ SE"imm"
0110	imm		rd	MHI	$R_f(rd)$ 'MSB <= "imm"
0111	rs2	rs1	rd	SLR	$R_f(rd) <= R_f(rs1)$ LS$\pm$ $R_f(rs2)$
1000	rs2	rs1	rd	SAR	$R_f(rd) <= R_f(rs1)$ AS$\pm$ $R_f(rs2)$
1001	rs2	rs1	rd	ADR	$R_f(rd) <= R_f(rs1) + R_f(rs2)$
1010	rs2	rs1	rd	SUR	$R_f(rd) <= R_f(rs1) - R_f(rs2)$
1011	imm		rd	ADI	$R_f(rd) <= R_f(rd)$ + SE"imm"
1100	imm		rd	SUI	$R_f(rd) <= R_f(rd)$ – SE"imm"
1101	rs2	rs1	rd	MUL	$R_f(rd) <= R_f(rs1) \times R_f(rs2)$ 'LSB $R_f(rd+1) <= R_f(rs1) \times R_f(rs2)$ 'MSB
1110	rs2	rs1	rd	DIV	$R_f(rd) <= R_f(rs1) \div R_f(rs2)$ 'Quo $R_f(rd+1) <= R_f(rs1) \div R_f(rs2)$ 'Rem

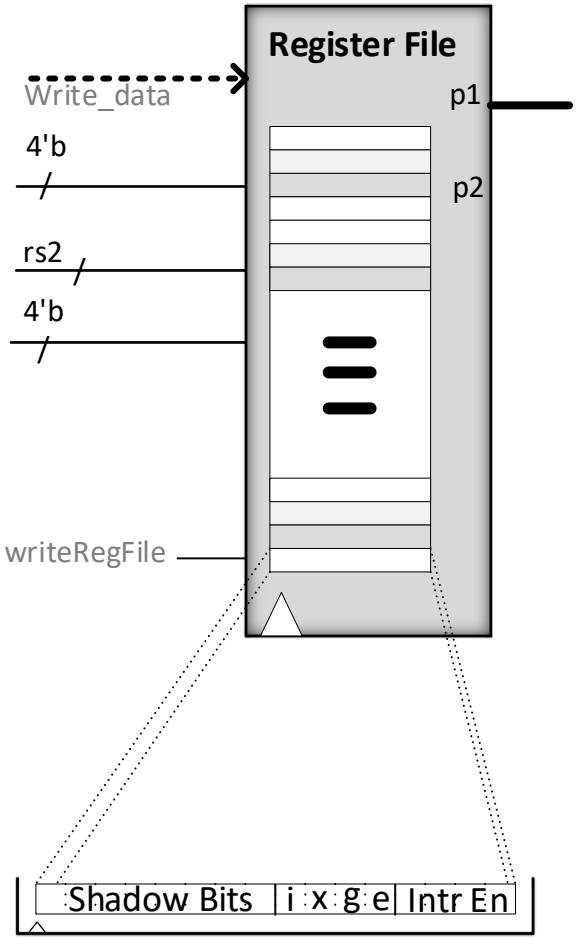


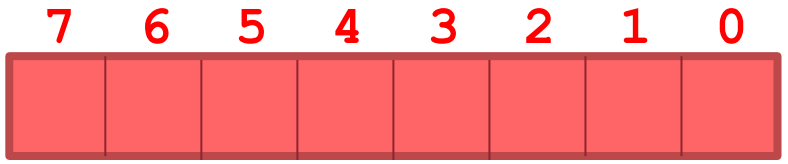
SAYAC Instructions - Basic

ixgl

[15:12]	[11:10]	[9]	[8]	[7:4]	[3:0]	Instruction	Notation
1111	00	0		rs1	rd	CMR	flags <= Cmp(R_f(rs1), R_f(rd))
		1	imm		rd	CMI	flags <= Cmp(R_f(rd), SE"imm")
	01	0	flag interpretation bits		rd	BRC	PC <= R_f(rd), IF flag
		1	flag interpretation bits		rd	BRR	PC <= PC + R_f(rd), IF flag
	10	0	shim		rd	SHI	R_f(rd) <= R_f(rd) LS± "shim"
		1	shim		rd		R_f(rd) <= R_f(rd) AS± "shim"
	11	0	0	rs1	rd	NTR	R_f(rd) <= 1sComp(R_f(rs1))
			1				R_f(rd) <= 2sComp(R_f(rs1))
		1	0		rd	NTD	R_f(rd) <= 1sComp(R_f(rd))
			1				R_f(rd) <= 2sComp(R_f(rd))

SAYAC Instructions - Shadow

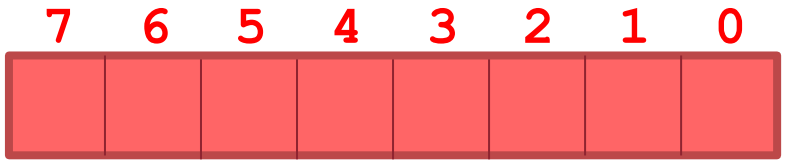




Shadow [7:0] which is IR[15:8]

SAYAC Instructions

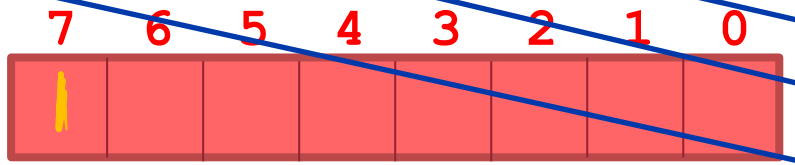
[15:12]	[11:10]	[9]	[8]	[7:4]	[3:0]	Instruction	Notation
0000	Reserved						
0001	Reserved						
0010	00	0		rs1	rd	LDR	Rf(rd) <= MEM(Rf(rs1))
		1					Rf(rd) <= IO(Rf(rs1))
	01	0		rs1	rd	STR	MEM(Rf(rd)) <= Rf(rs1)
		1					IO(Rf(rd)) <= Rf(rs1)
	10	s		rs1	rd	JMR	PC <= PC + Rf(rs1) Rf(rd) <= PC + 1, IF s=1
	11	imm			rd	JMI	PC <= PC + "imm" Rf(rd) <= PC + 1



Shadow [7:0] which is IR[15:8]

SAYAC Instructions - Shadow

	[15:12]	[11:8]	[7:4]	[3:0]	Instruction	Notation
	0011	rs2	rs1	rd	ANR	Rf(rd) <= Rf(rs1) AND Rf(rs2)
7 1	0100	**imm**	**rs1**	rd	**ANI**	**R_f(rd) <= R_f(rs1) AND USE"imm"**
6 1	0101	imm		rd	**MSI**	**Rf(rd) <= SE"imm"**
	0110	imm		rd	MHI	Rf(rd) 'MSB <= "imm"
	0111	rs2	rs1	rd	SLR	Rf(rd) <= Rf(rs1) LS± Rf(rs2)
	1000	rs2	rs1	rd	SAR	Rf(rd) <= Rf(rs1) AS± Rf(rs2)
	1001	rs2	rs1	rd	ADR	Rf(rd) <= Rf(rs1) + Rf(rs2)
	1010	rs2	rs1	rd	SUR	Rf(rd) <= Rf(rs1) – Rf(rs2)
7 1	1011	**imm**	**rs1**	rd	**ADI**	**R_f(rd) <= R_f(rs1) + SE"imm"**
7 1	1100	**imm**	**rs1**	rd	**SUI**	**R_f(rd) <= R_f(rs1) – SE"imm"**
5 1	1101	rs2	rs1	rd	**MUL**	**R_f(rd) <= R_f(rs1) × R_f(rs2) 'LSB** R_f(rd+1) <= R_f(rs1) × R_f(rs2) 'MSB
4 1 3 1	1110	rs2	rs1	rd	**DIV**	**R_f(rd) <= R_f(rs1) ÷ R_f(rs2) 'Quo** R_f(rd+1) <= R_f(rs1) ÷ R_f(rs2) 'Rem



Shadow [7:0] which is IR[15:8]

Flip extended bits

Right-most bits only

Signed Mul Div

Quotient only

SAYAC Instructions - Shadow

[15:12]	[11:10]	[9]	[8]	[7:4]	[3:0]	Instruction	Notation
1111	00	0		rs1	rd	**CMR**	**flags <= Cmp(R_f(rs1), R_f(rd))**
		1	imm		rd	**CMI**	**flags <= Cmp(R_f(rd), SE"imm")**
	01	0	flag interpretation bits		rd	BRC	PC <= R_f(rd), IF flag
		1	flag interpretation bits		rd	BRR	PC <= PC + R_f(rd), IF flag
	10	0	shim		rd	SHI	Rf(rd) <= Rf(rd) LS± "shim"
		1	shim		rd		Rf(rd) <= Rf(rd) AS± "shim"
	11	0	0	rs1	rd	NTR	Rf(rd) <= 1sComp(Rf(rs1))
			1				Rf(rd) <= 2sComp(Rf(rs1))
		1	0		rd	NTD	Rf(rd) <= 1sComp(Rf(rd))
			1				Rf(rd) <= 2sComp(Rf(rd))

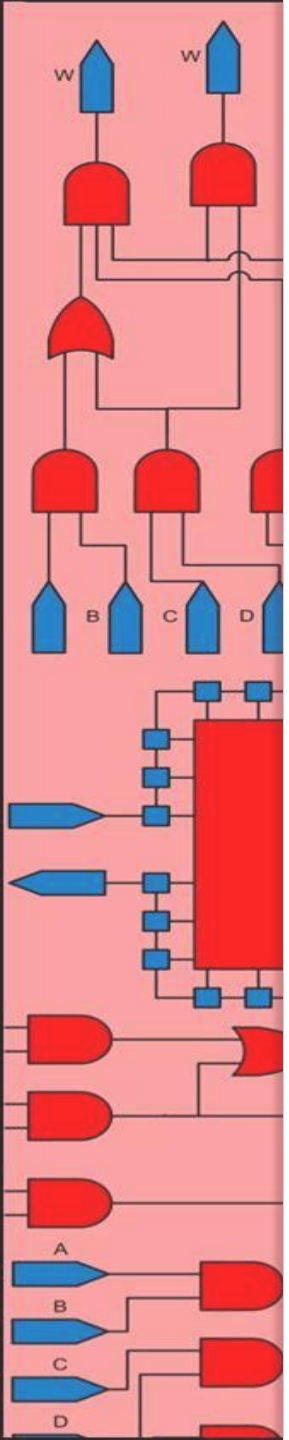
1 2

1 2

7	6	5	4	3	2	1	0

Signed Compare

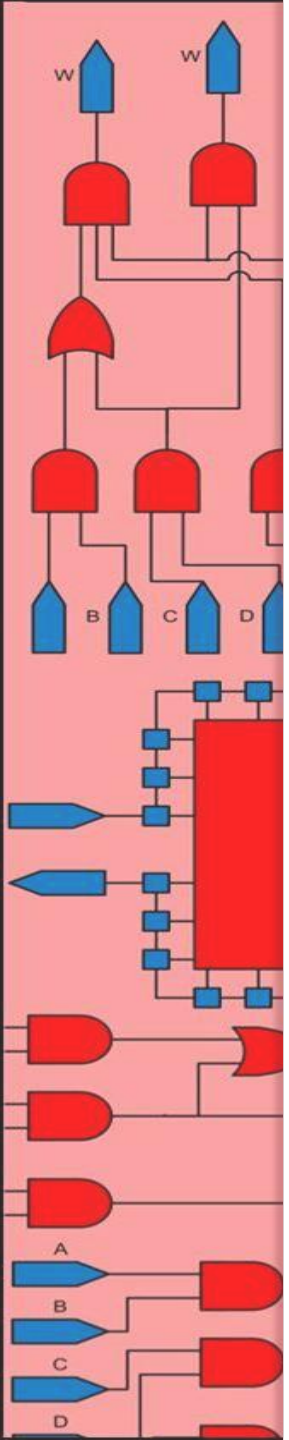
Shadow [7:0] which is IR[15:8]



Processor Architecture

Outline:

- Processor and Memory Model
- Processor Model Specification
- Instruction Execution Cycle
- Processor Registers
- Instruction Format
- SAYAC Instructions
- **SAYAC Hardware Implementation**
  - SAYAC Datapath and Instruction flow
  - Controller
  - Datapath VHDL Description
    - Control Signals
    - Components of Datapath
    - Bussing
  - Controller VHDL Description
  - Putting SAYAC Together
  - Memory Model
  - Instruction execution and Testing
  - Conclusions

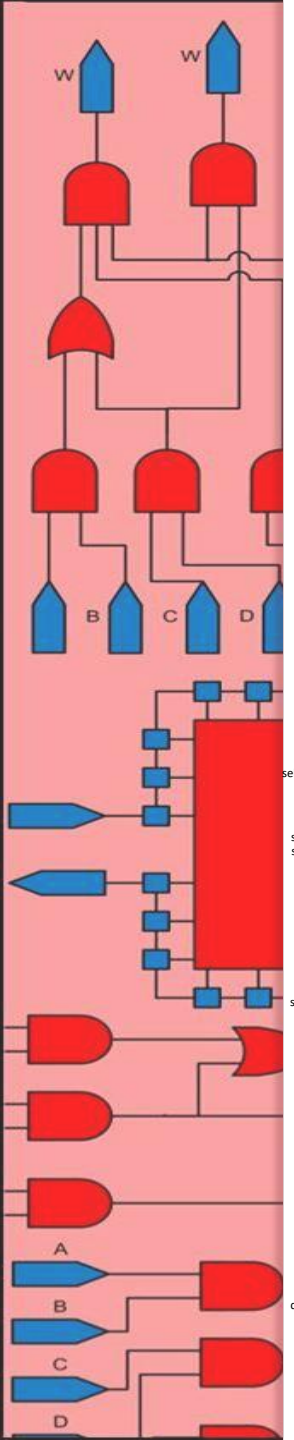


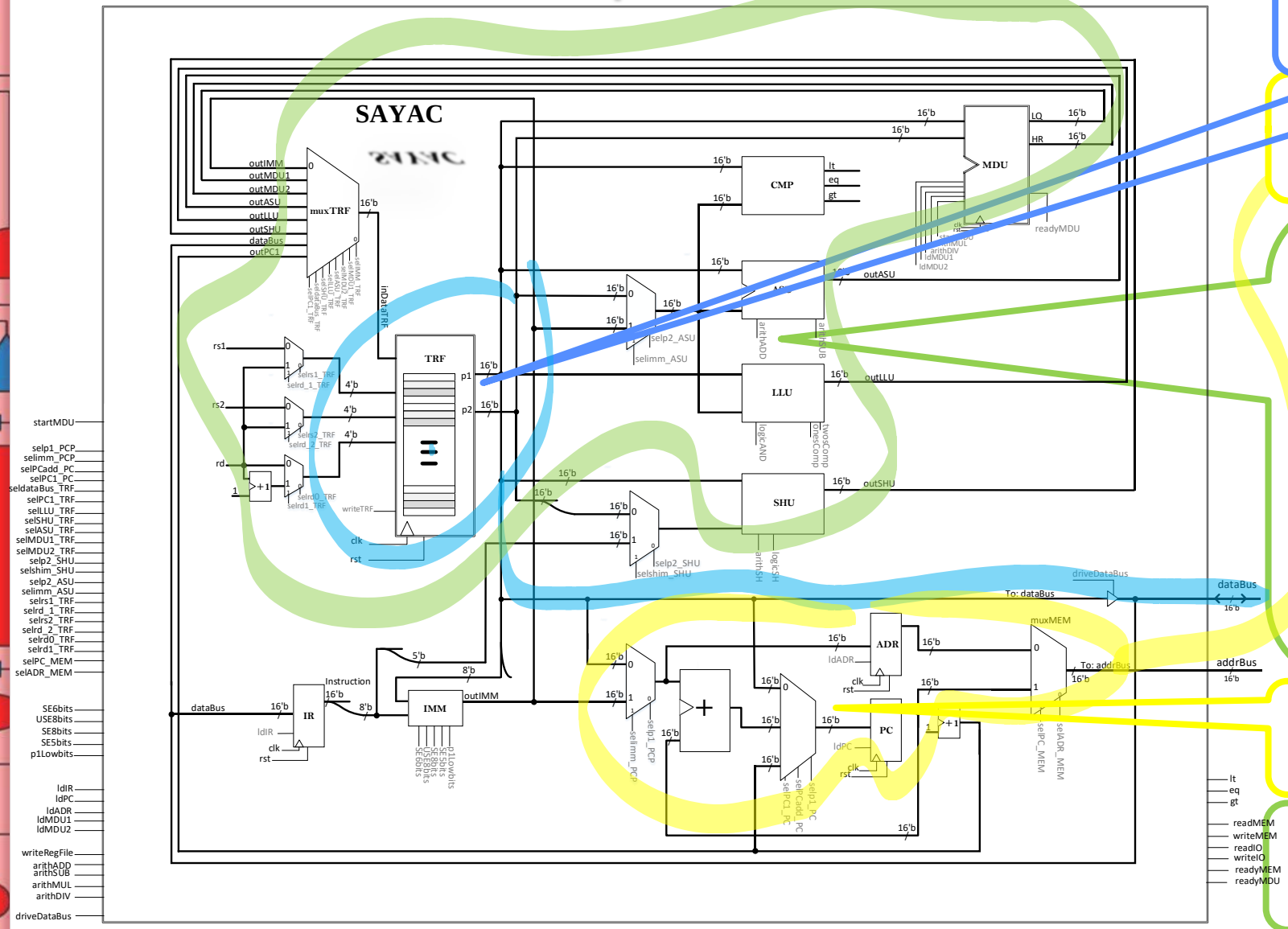
SAYAC Hardware Implementation

Outline:

- Processor and Memory Model
- Processor Model Specification
- Instruction Execution Cycle
- Processor Registers
- Instruction Format
- SAYAC Instructions
- **SAYAC Hardware Implementation**
  - **SAYAC Datapath and Instruction flow**
  - Controller
  - Datapath VHDL Description
    - Control Signals
    - Components of Datapath
    - Bussing
  - Controller VHDL Description
  - Putting SAYAC Together
  - Memory Model
  - Instruction execution and Testing
  - Conclusions

SAYAC Hardware Implementation





Instruction	Notation
LDR	.rd. <= [.rs1.]
	.rd. <= { .rs1. }
STR	[.rd.] <= .rs1.
	{ .rd. } <= .rs1.
JMR	PC <= PC + .rs1.
	.rd. <= PC + 1 if s = 1
JMI	PC <= PC + "imm"
	.rd. <= PC + 1
ANR	.rd. <= .rs1. AND .rs2.
ANI	.rd. <= .rd. AND USE"imm"
MSI	.rd. <= SE"imm"
MHL	.rd. [15:8] <= "imm"
SIR	.rd. <= .rs1. LS± .rs2.
SAR	.rd. <= .rs1. AS± .rs2.
ADR	.rd. <= .rs1. + .rs2.
SUR	.rd. <= .rs1. – .rs2.
ADI	.rd. <= .rd. + SE"imm"
SUI	.rd. <= .rd. - SE"imm"
MUL	.rd. <= .rs1. × .rs2. 'LSB
	.rd+1. <= .rs1. × .rs2. 'MSB
DIV	.rd. <= .rs1. ÷ .rs2. 'Quo
	.rd+1. <= .rs1. ÷ .rs2. 'Rem
CMR	flags <= Cmp(.rs1. , .rd.)
CMI	flags <= Cmp(.rd. , SE"imm")
BRC	PC <= .rd. if flag
BRR	PC <= PC + .rd. if flag
SHI	.rd. <= .rd. LS± "shim"
	.rd. <= .rd. AS± "shim"
NTR	.rd. <= 1sComp(.rs1.)
	.rd. <= 2sComp(.rs1.)
NTD	.rd. <= 1sComp(.rd.)
	.rd. <= 2sComp(.rd.)

Nomenclature:

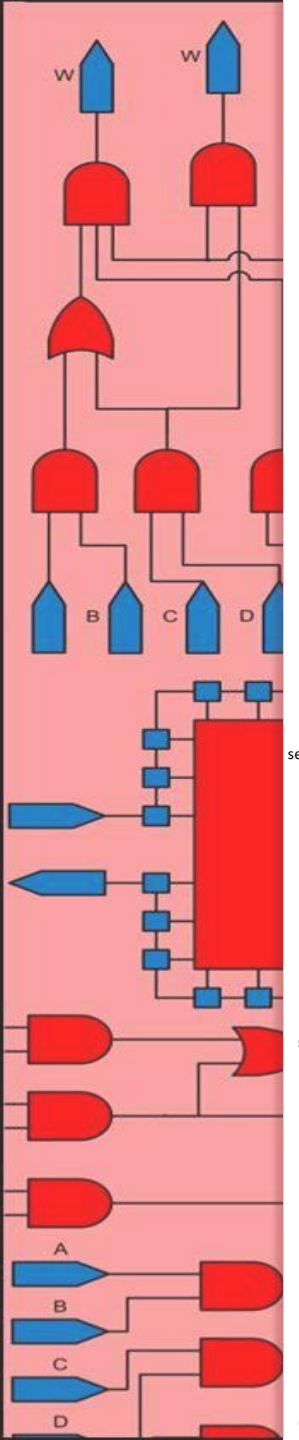
.rsd. → R_f content pointed by *rsd*

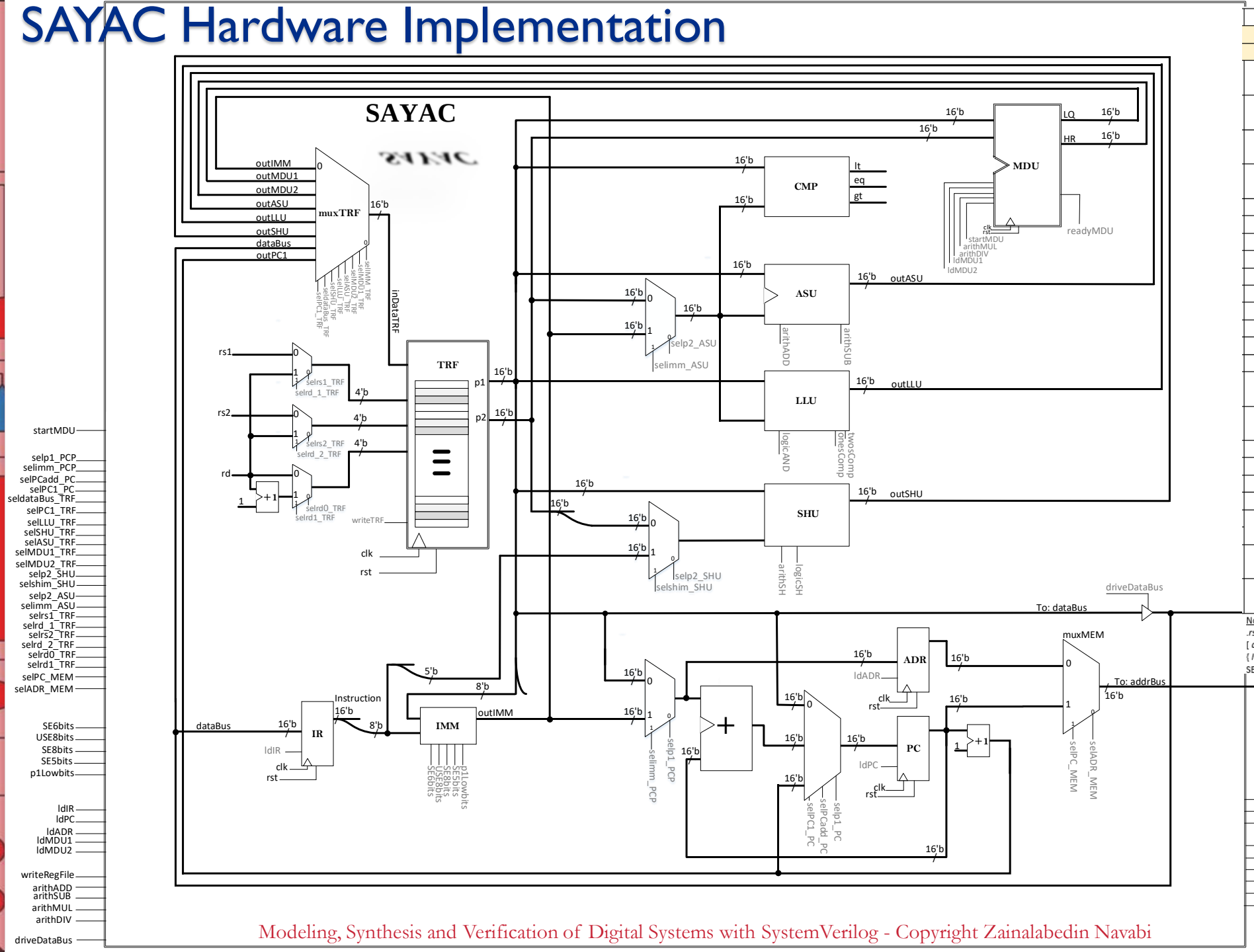
[*adr*] → Memory addressed by *adr*

{ *loc* } → IO device addressed by *loc*

SE"imm, USE"imm" → Signed, Unsigned Extension of "imm"

SAYAC Hardware Implementation

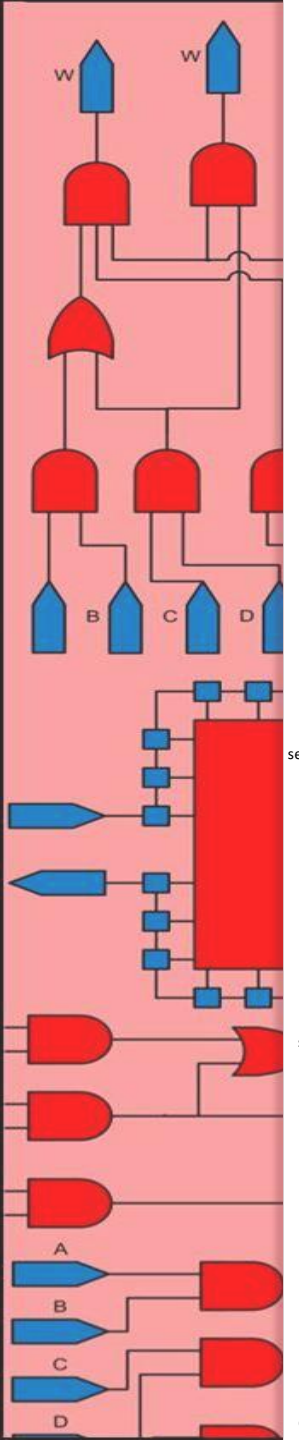


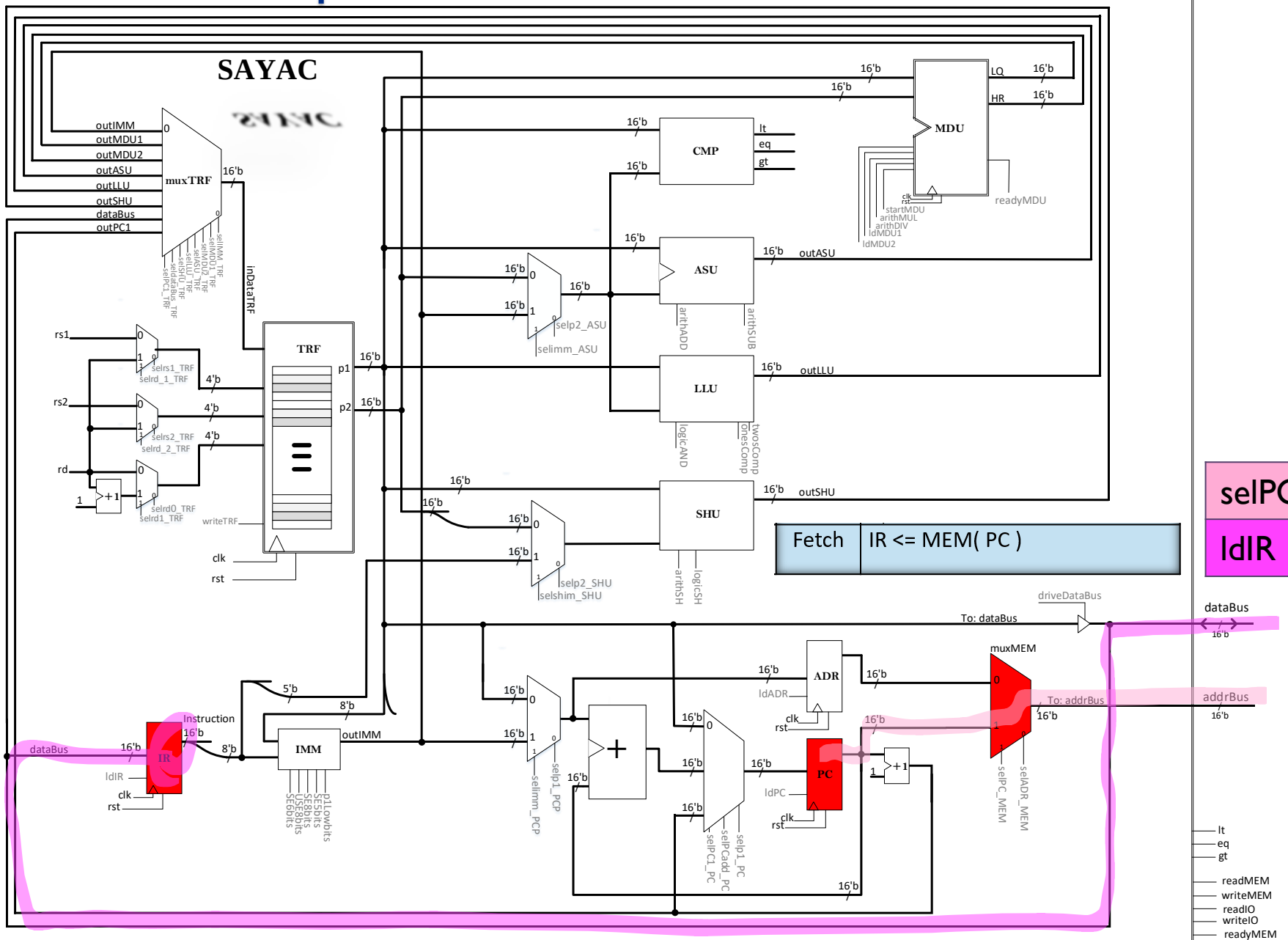


Instruction	Notation
LDR	.rd. <= [.rs1.]
	.rd. <= { .rs1. }
STR	[.rd.] <= .rs1.
	{ .rd. } <= .rs1.
JMR	PC <= PC + .rs1.
	.rd. <= PC + 1 if s = 1
JMI	PC <= PC + "imm"
	.rd. <= PC + 1
ANR	.rd. <= .rs1. AND .rs2.
ANI	.rd. <= .rd. AND USE"imm"
MSI	.rd. <= SE"imm"
MHL	.rd. [15:8] <= "imm"
SIR	.rd. <= .rs1. LS± .rs2.
SAR	.rd. <= .rs1. AS± .rs2.
ADR	.rd. <= .rs1. + .rs2.
SUR	.rd. <= .rs1. – .rs2.
ADI	.rd. <= .rd. + SE"imm"
SUI	.rd. <= .rd. - SE"imm"
MUL	.rd. <= .rs1. × .rs2. 'LSB
	.rd+1. <= .rs1. × .rs2. 'MSB
DIV	.rd. <= .rs1. ÷ .rs2. 'Quo
	.rd+1. <= .rs1. ÷ .rs2. 'Rem
CMR	flags <= Cmp(.rs1. , .rd.)
CMI	flags <= Cmp(.rd. , SE"imm")
BRC	PC <= .rd. if flag
BRR	PC <= PC + .rd. if flag
SHI	.rd. <= .rd. LS± "shim"
	.rd. <= .rd. AS± "shim"
NTR	.rd. <= 1sComp(.rs1.)
	.rd. <= 2sComp(.rs1.)
NTD	.rd. <= 1sComp(.rd.)
	.rd. <= 2sComp(.rd.)

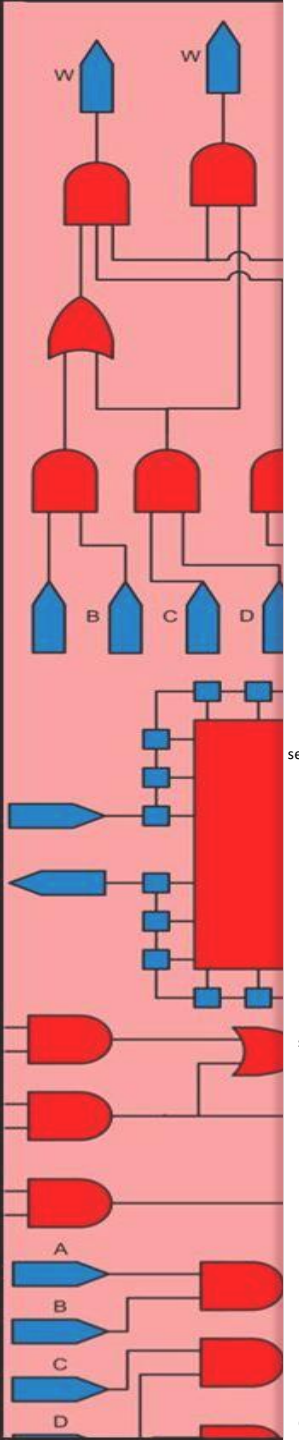
Nomenclature:
.rsd. → R_i content pointed by *rsd*
[*adr*] → Memory addressed by *adr*
{ *loc* } → IO device addressed by *loc*
SE"imm, USE"imm" → Signed, Unsigned Extension of "imm"

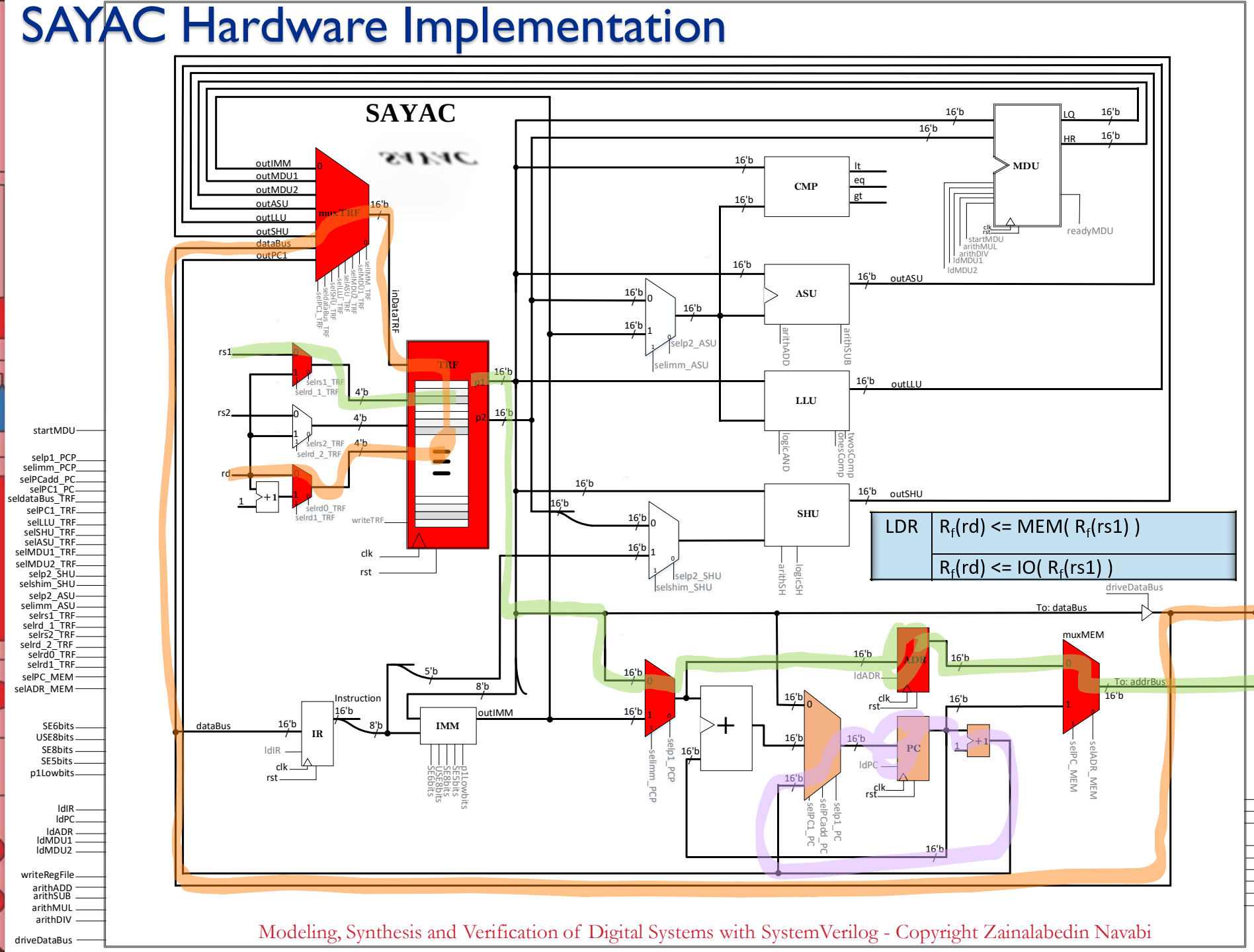
SAYAC Hardware Implementation





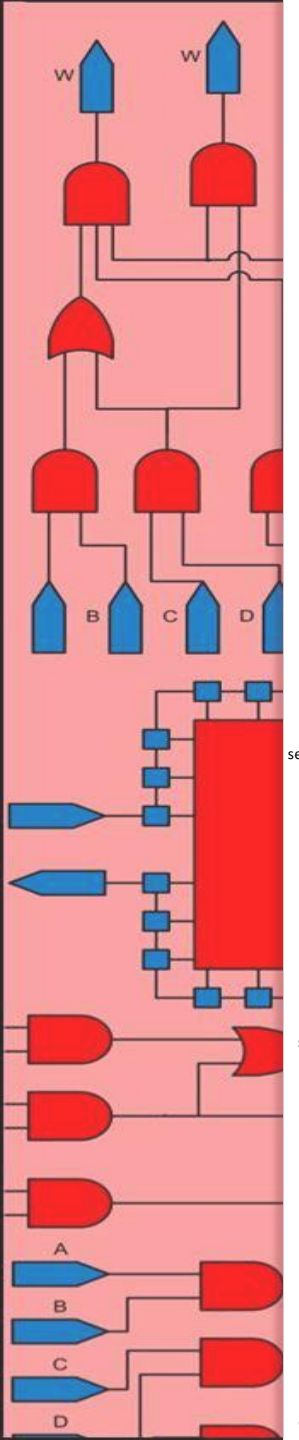
SAYAC Hardware Implementation

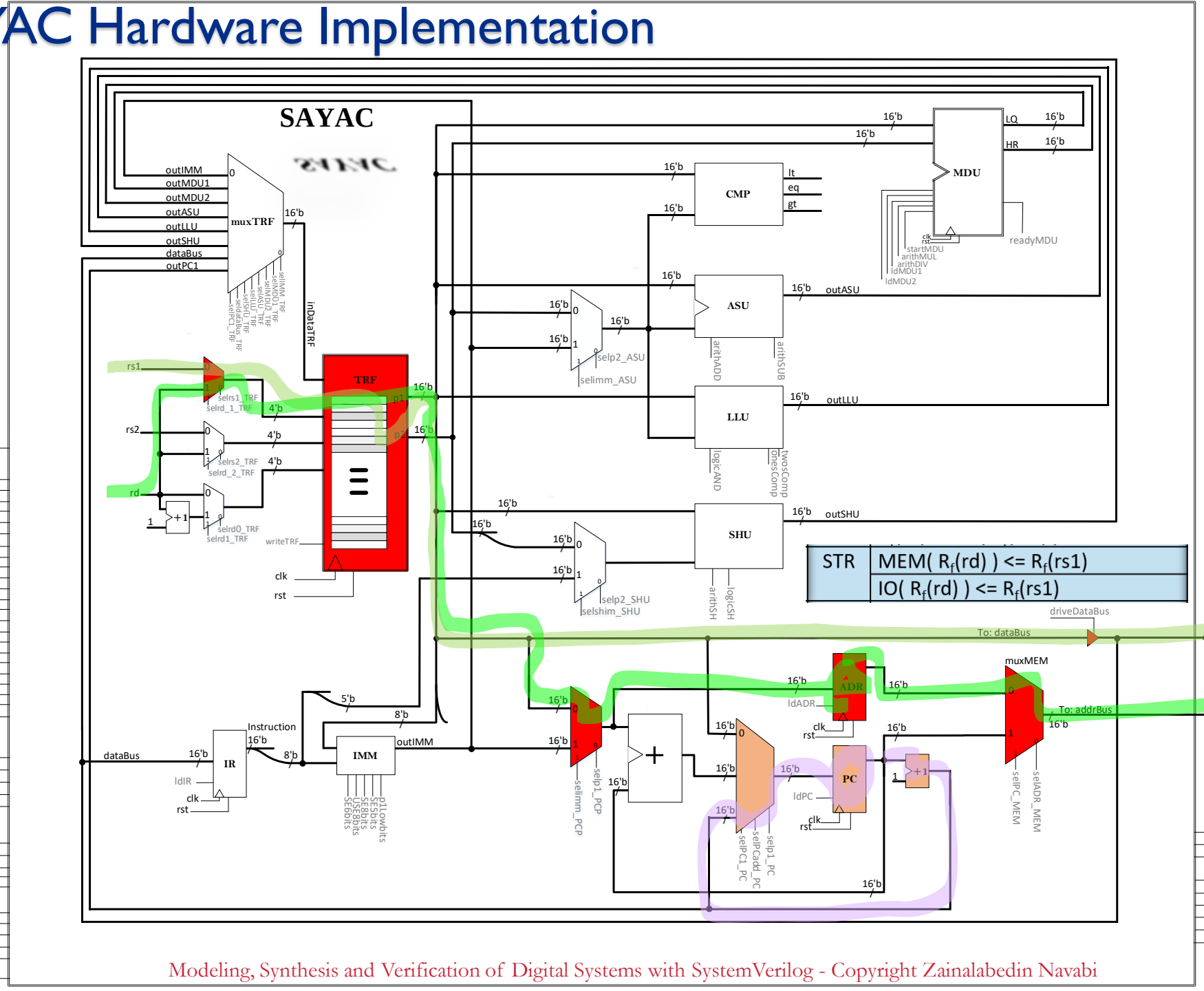




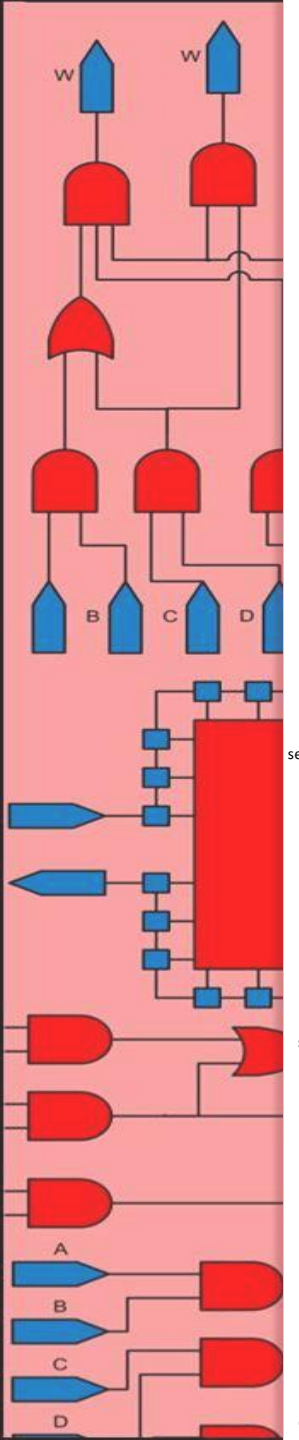
selrs1_TRF
selrd0_TRF
selp1_PCP
ldADR
Clk ⎍
selADR_MEM
seldatabus_TRF
writeRegFile
readIO/MEM
selPC1_PC
ldPC

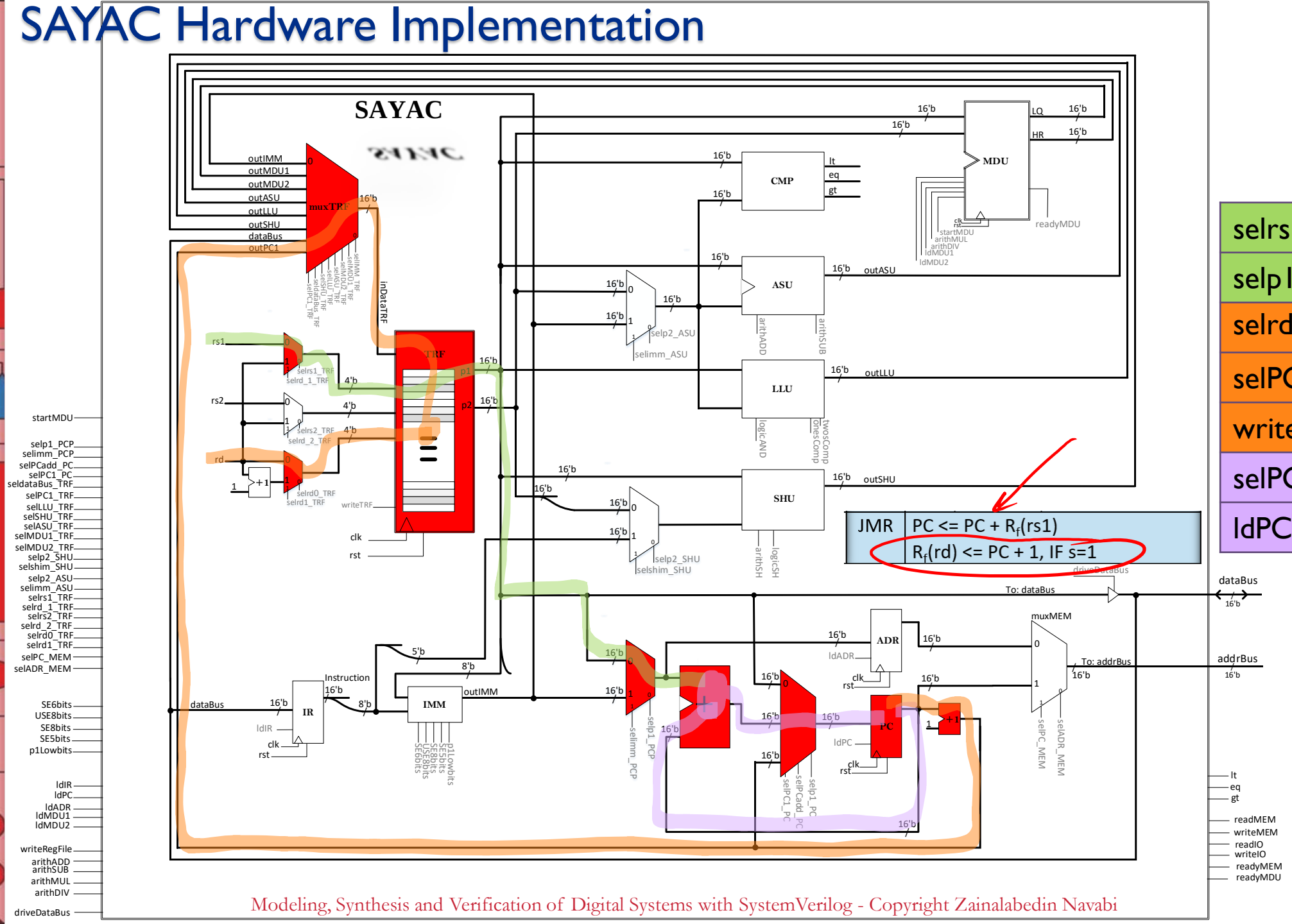
SAYAC Hardware Implementation





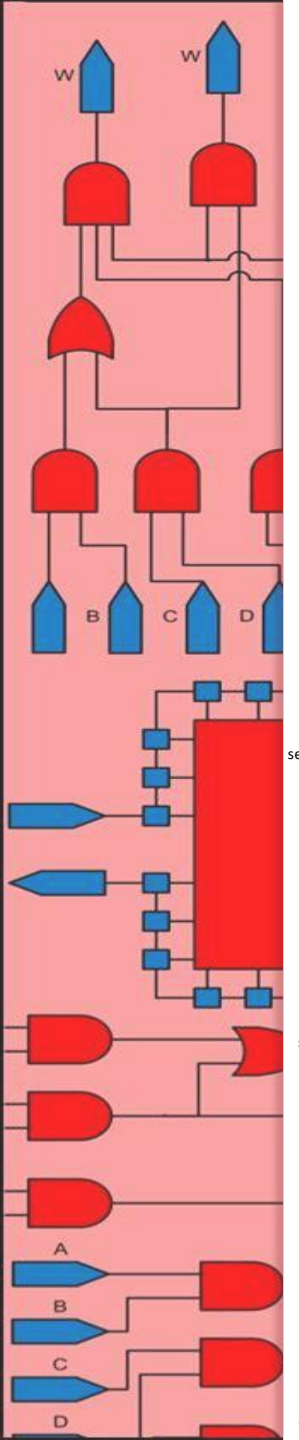
SAYAC Hardware Implementation

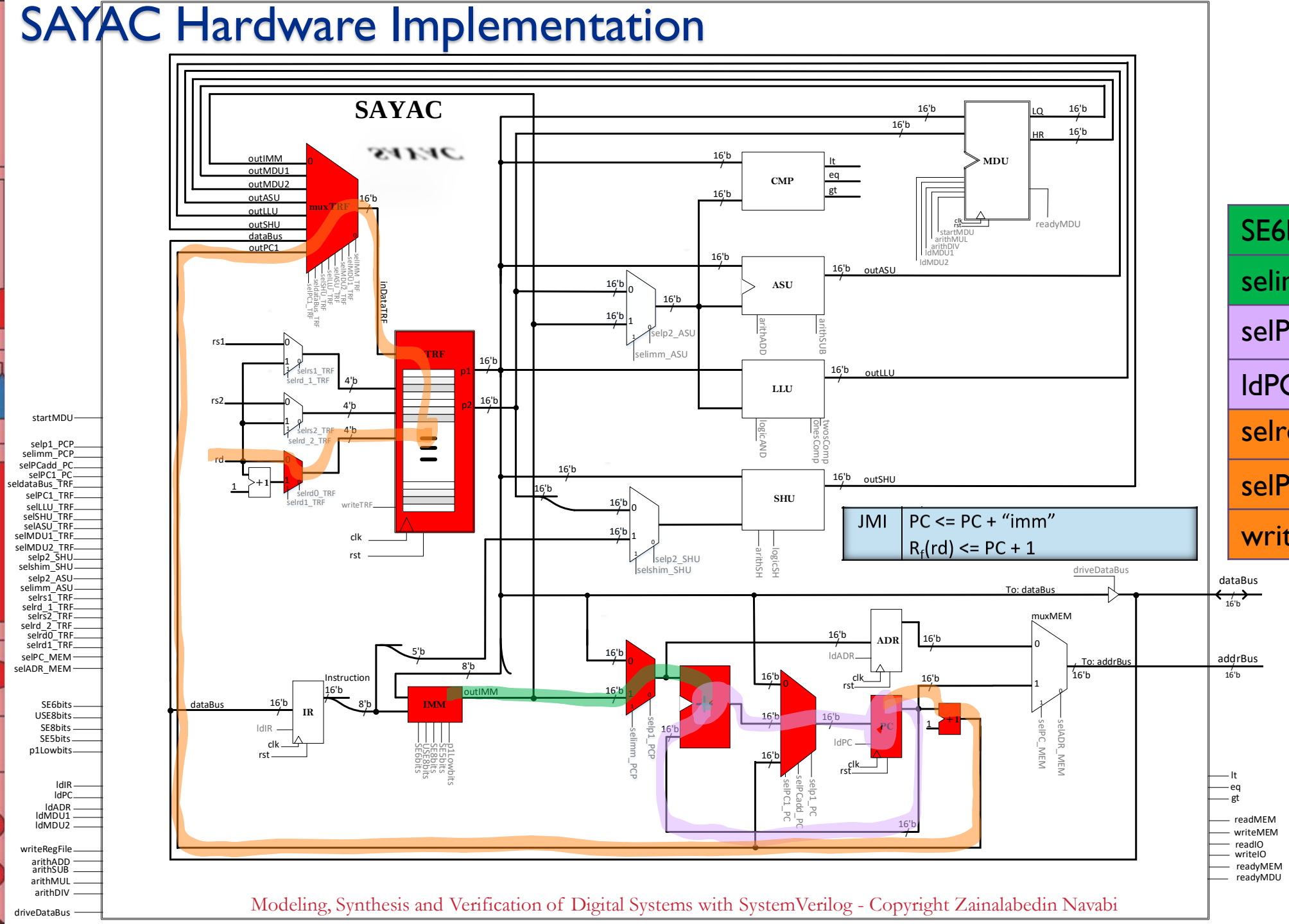




selrs1_TRF
selp1_PCP
selrd0_TRF
selPC1_TRF
writeRegFile
selPCadd_PC
ldPC

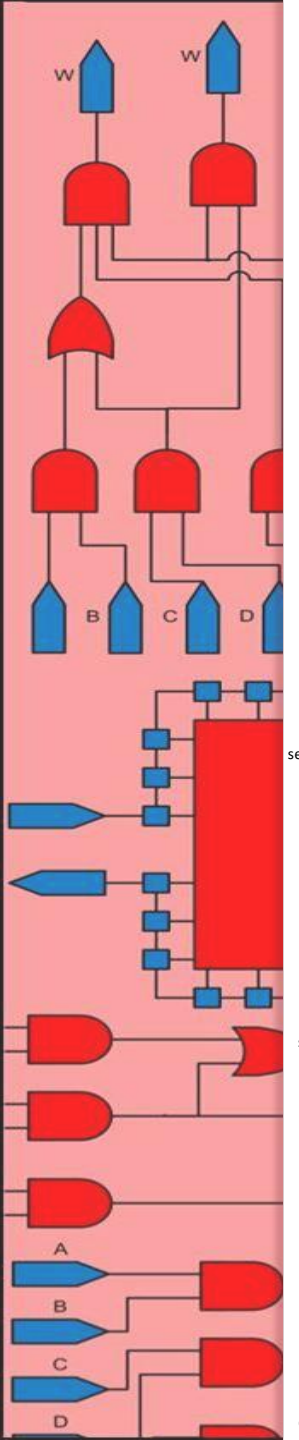
SAYAC Hardware Implementation

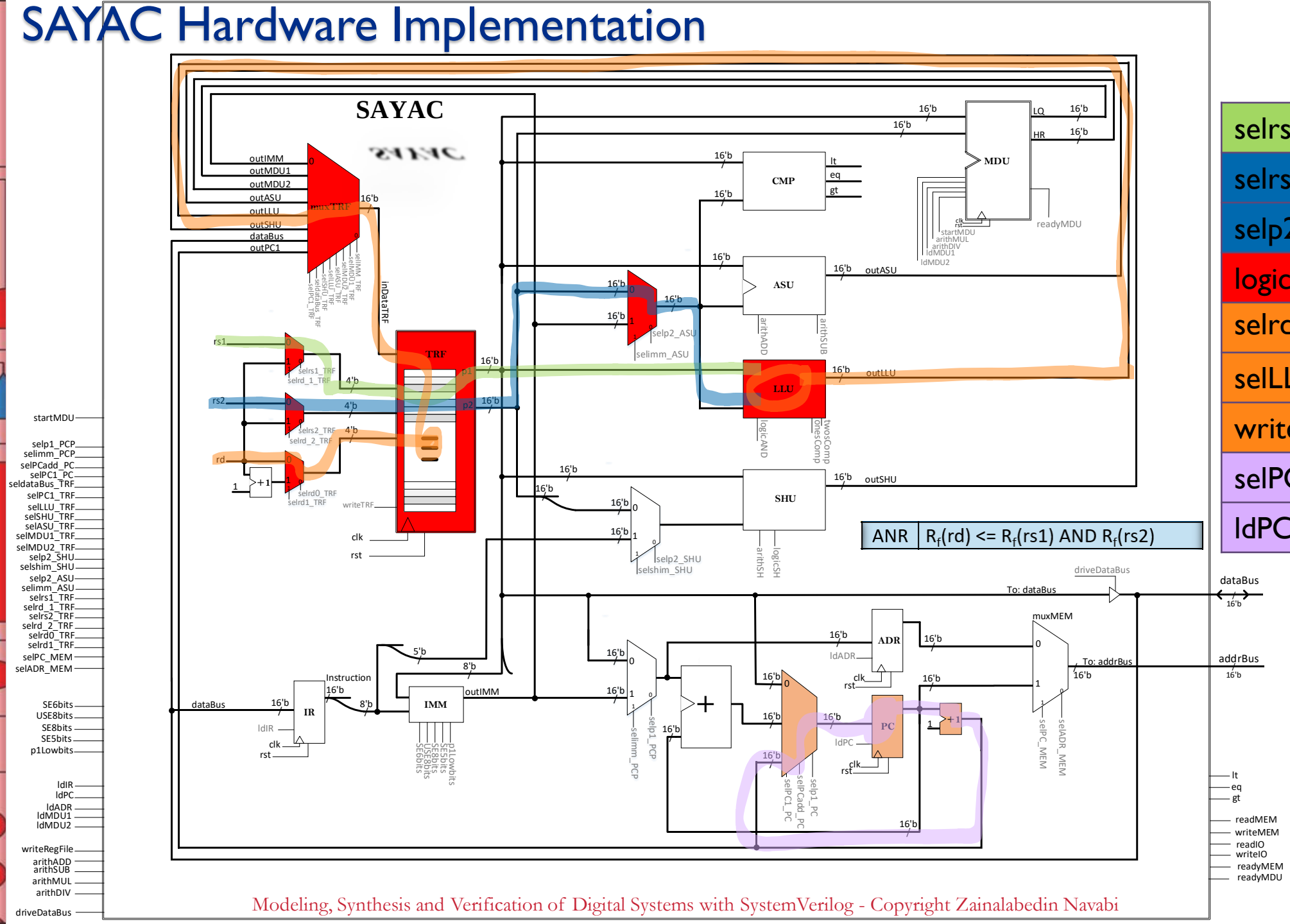




SE6bits
selimm_PCP
selPCadd_PC
ldPC
selrd0_TRF
selPC1_TRF
writeRegFile

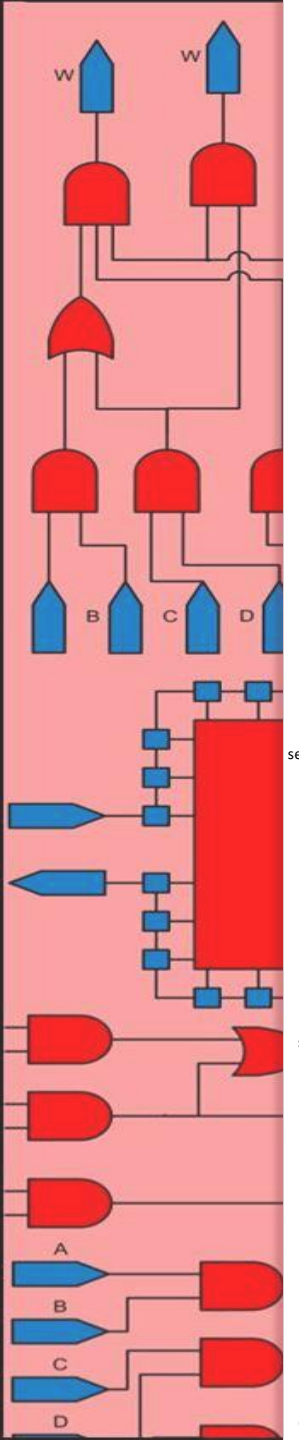
SAYAC Hardware Implementation

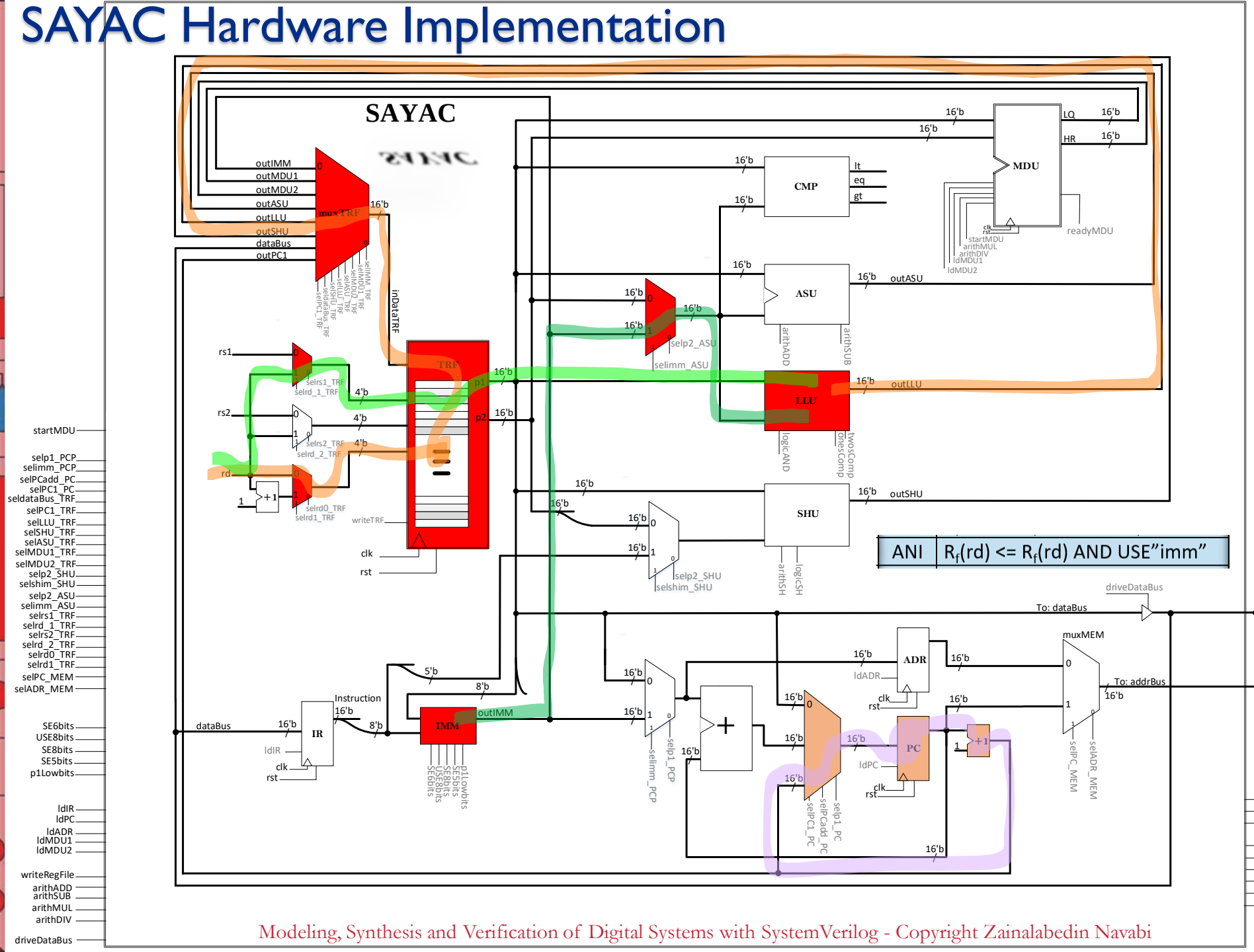




selrs1_TRF
selrs2_TRF
selp2_ASU
logicAND
selrd0_TRF
selLLU_TRF
writeRegFile
selPC1_PC
ldPC

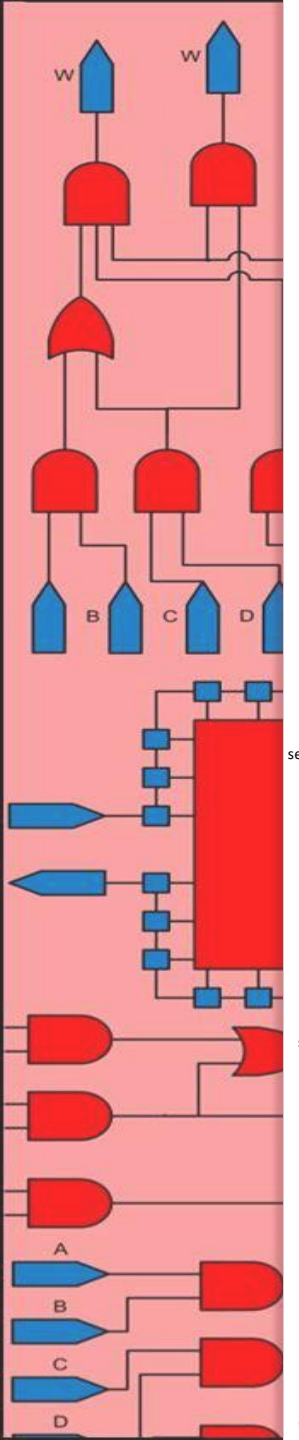
SAYAC Hardware Implementation

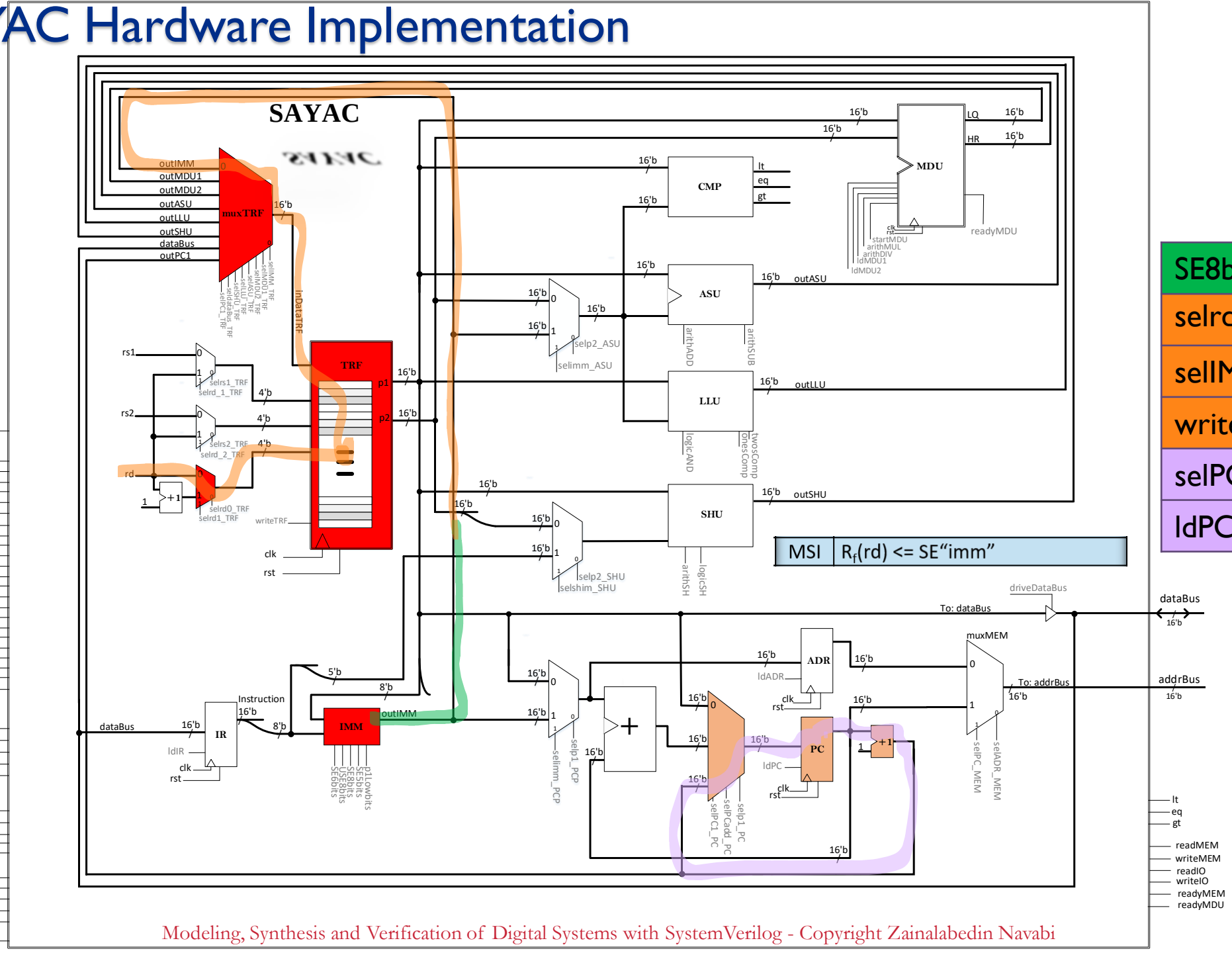




selrd_1_TRF
USE8bits
selimm_ASU
logicAND
selrd0_TRF
selLLU_TRF
writeRegFile
selPC1_PC
ldPC

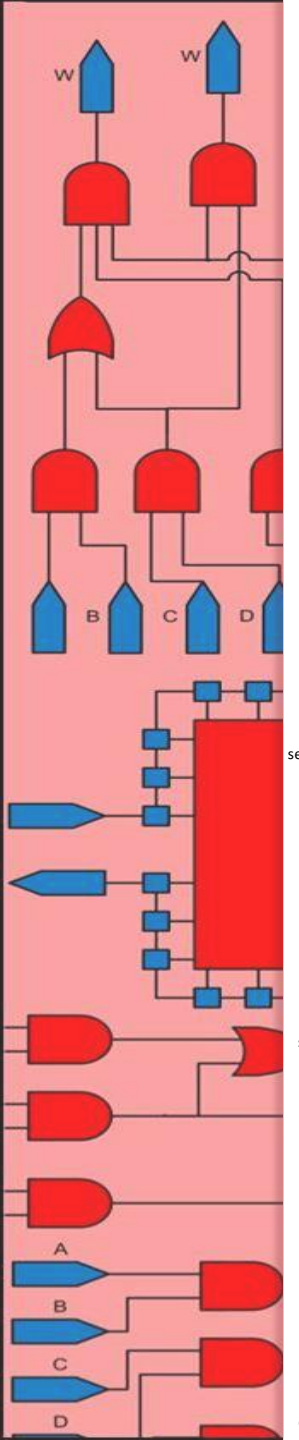
SAYAC Hardware Implementation

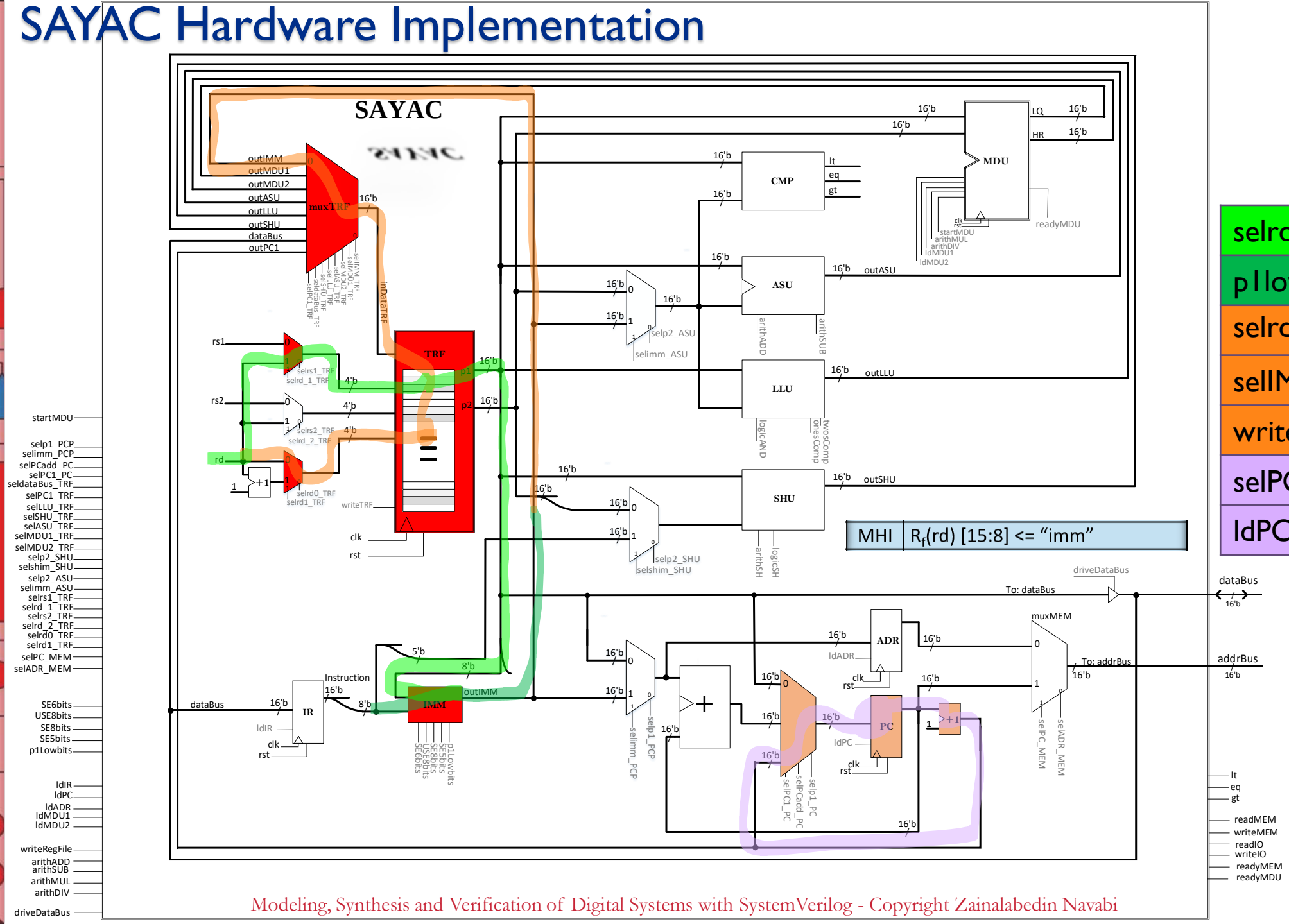




SE8bits
selrd0_TRF
selIMM_TRF
writeRegFile
selPC1_PC
ldPC

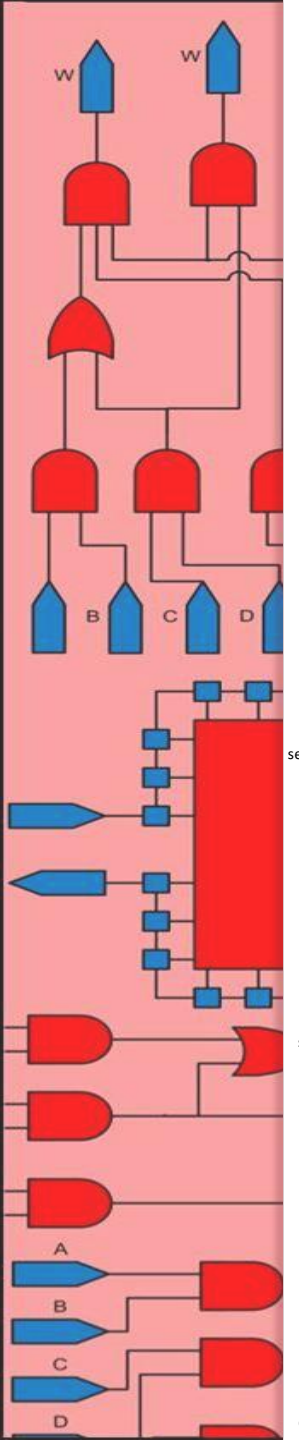
SAYAC Hardware Implementation

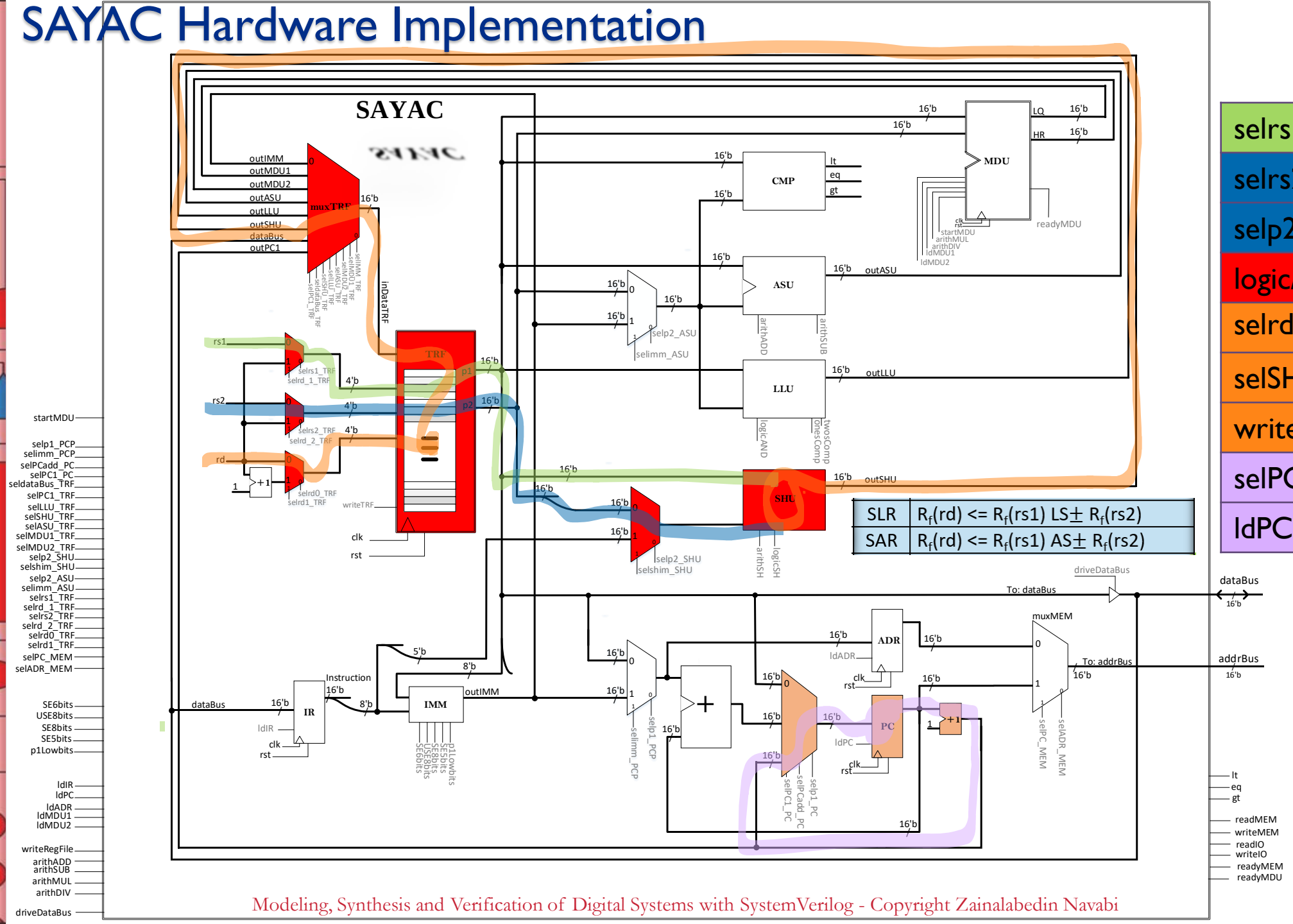




selrd_1_TRF
p1lowbits
selrd0_TRF
selIMM_TRF
writeRegFile
selPC1_PC
ldPC

SAYAC Hardware Implementation

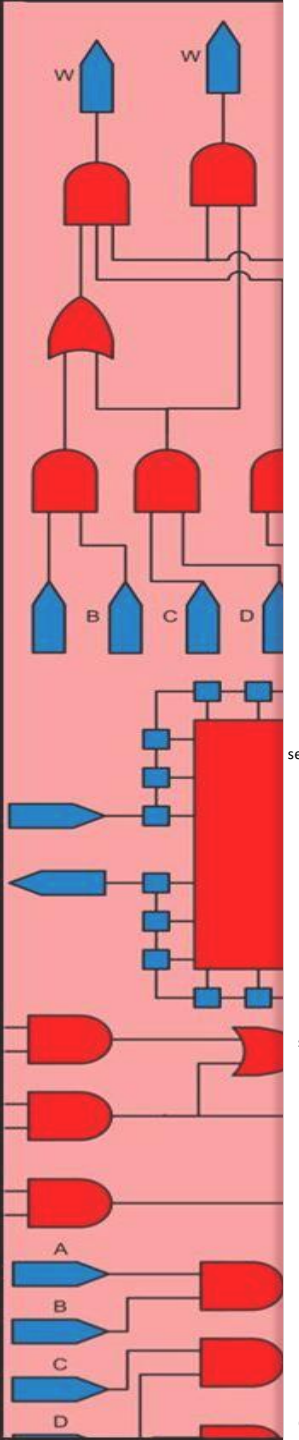


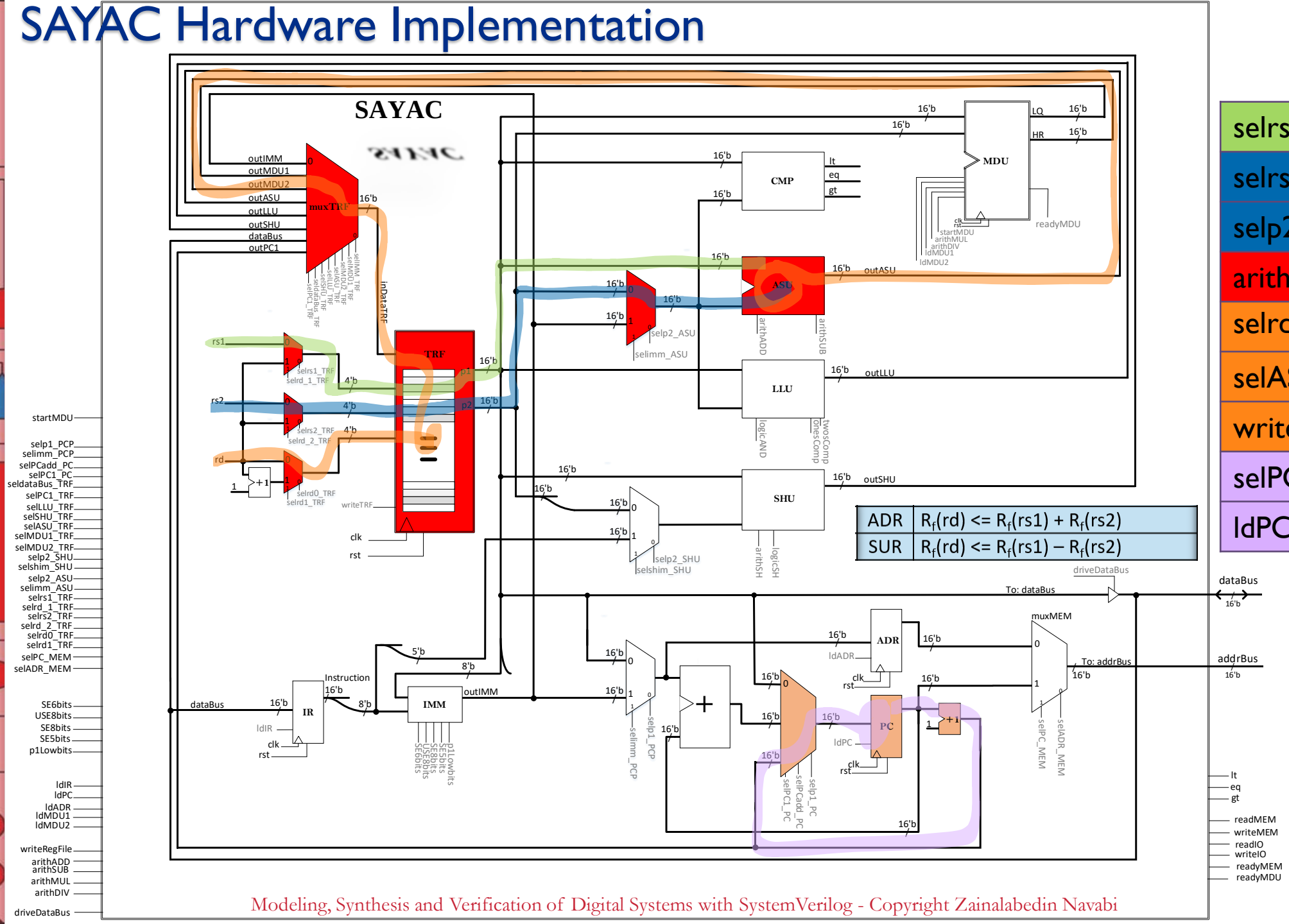


SLR	$R_f(rd) <= R_f(rs1)\ LS\pm\ R_f(rs2)$
SAR	$R_f(rd) <= R_f(rs1)\ AS\pm\ R_f(rs2)$

selrs1_TRF
selrs2_TRF
selp2_SHU
logic/arithSH
selrd0_TRF
selSHU_TRF
writeRegFile
selPC1_PC
ldPC

SAYAC Hardware Implementation

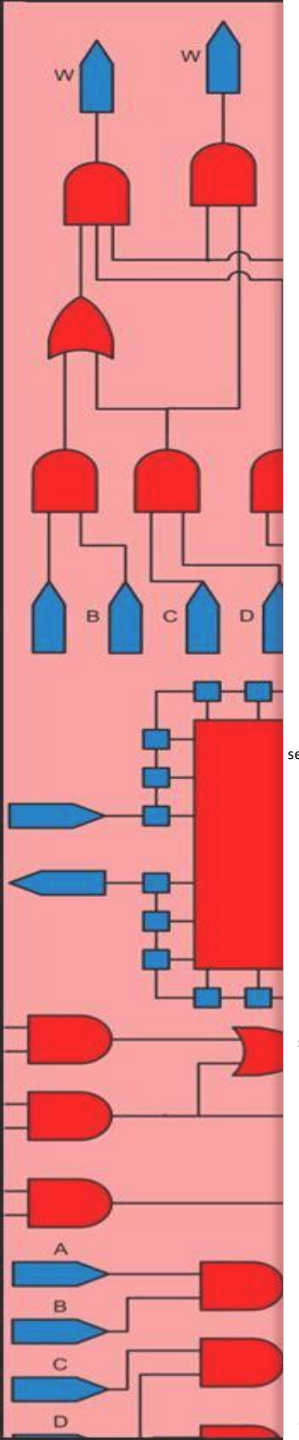


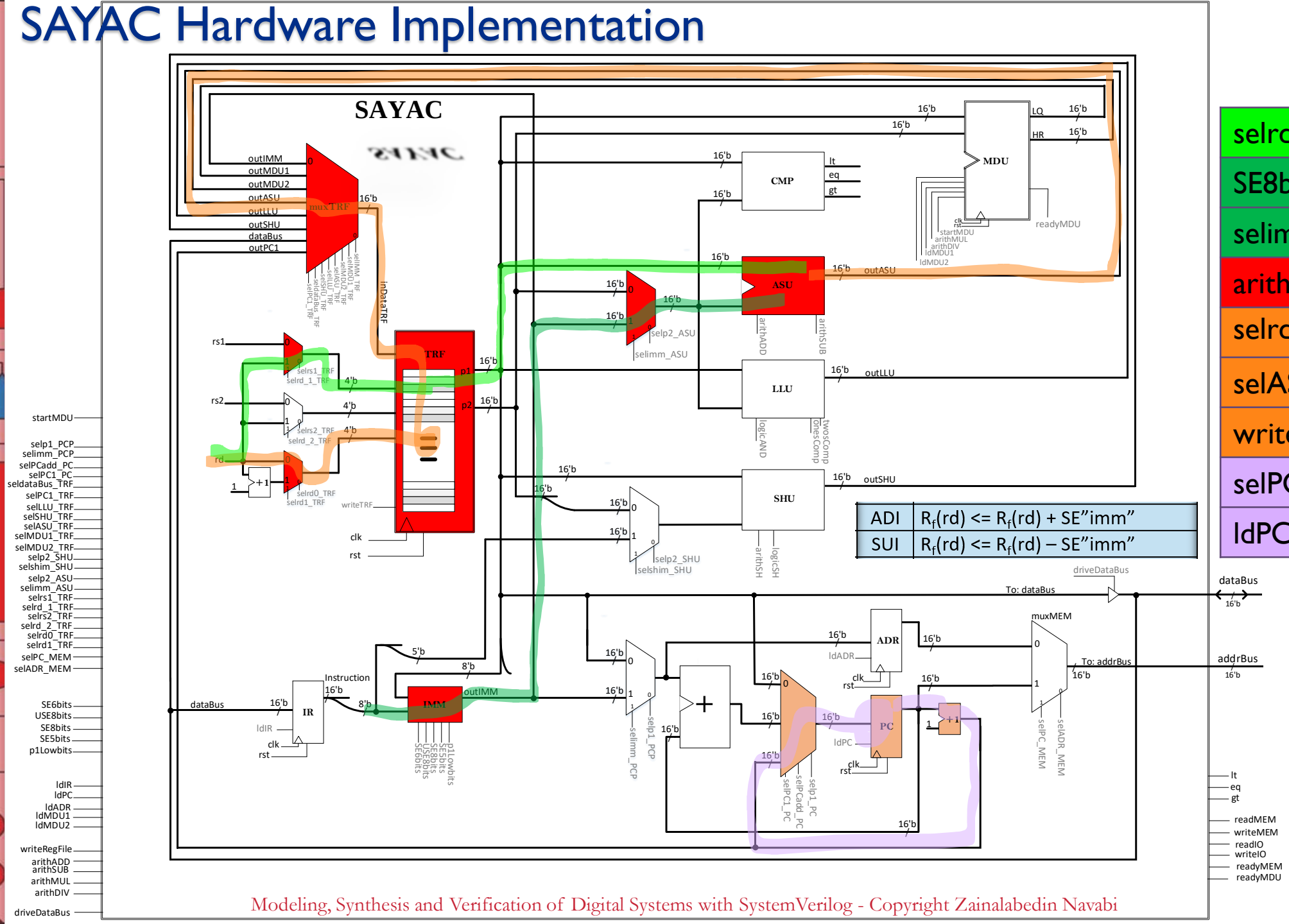


ADR	$R_f(rd) <= R_f(rs1) + R_f(rs2)$
SUR	$R_f(rd) <= R_f(rs1) - R_f(rs2)$

selrs1_TRF
selrs2_TRF
selp2_ASU
arithADD/SUB
selrd0_TRF
selASU_TRF
writeRegFile
selPC1_PC
ldPC

SAYAC Hardware Implementation

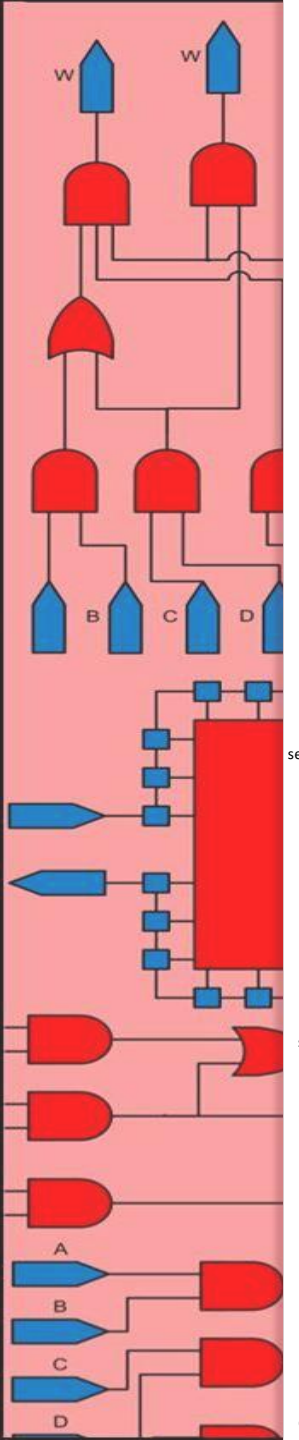


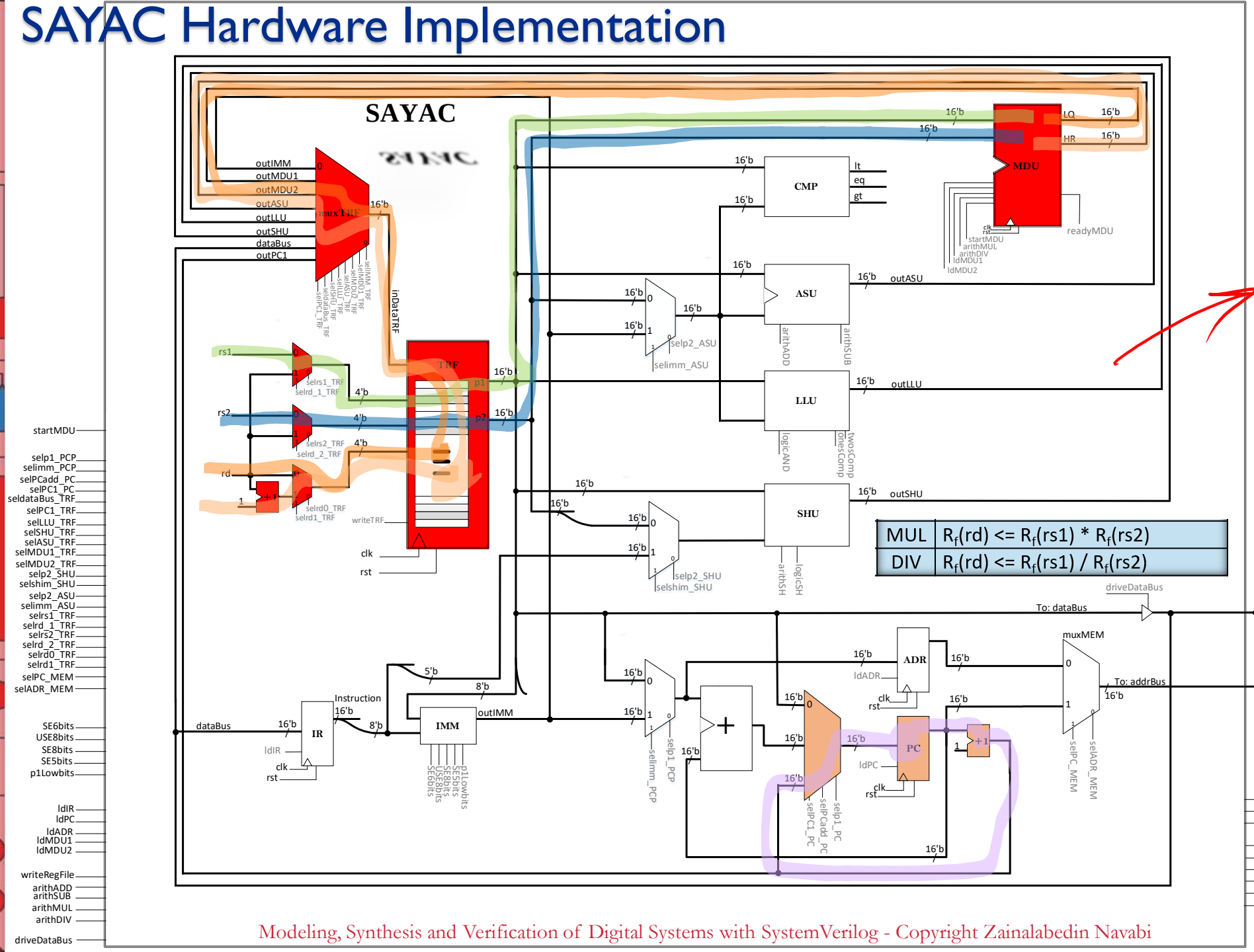


ADI	$R_f(rd) <= R_f(rd) + SE"imm"$
SUI	$R_f(rd) <= R_f(rd) - SE"imm"$

selrd_1_TRF
SE8bits
selimm_ASU
arithADD/SUB
selrd0_TRF
selASU_TRF
writeRegFile
selPC1_PC
ldPC

SAYAC Hardware Implementation

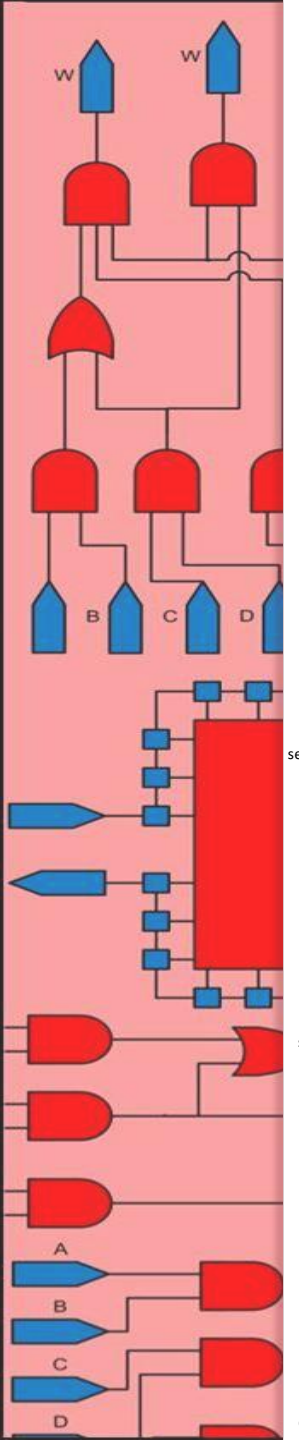


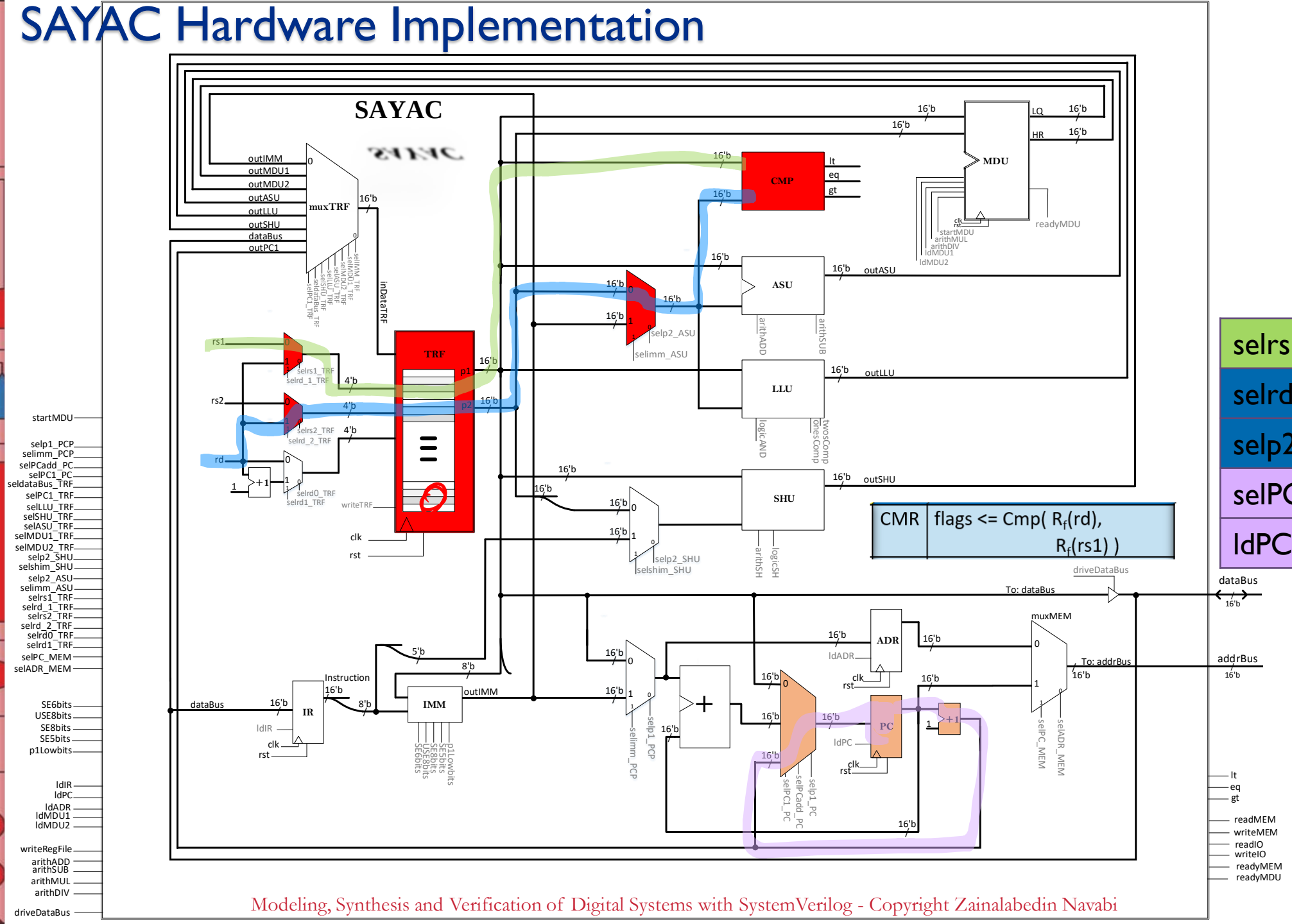


selrs1_TRF
selrs2_TRF
startMDU
arithMUL/DIV
ldMDU1
Clk
selrd0_TRF
selMDU1_TRF
writeRegFile
ldMDU2
Clk
selrd1_TRF
selMDU2_TRF
writeRegFile
selPC1_PC
ldPC

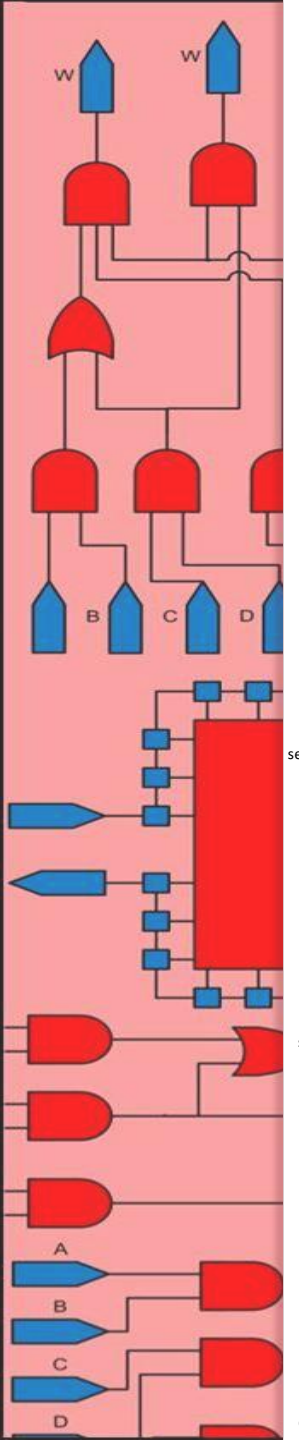
readMEM
writeMEM
readIO
writeIO
readyMEM
readyMDU

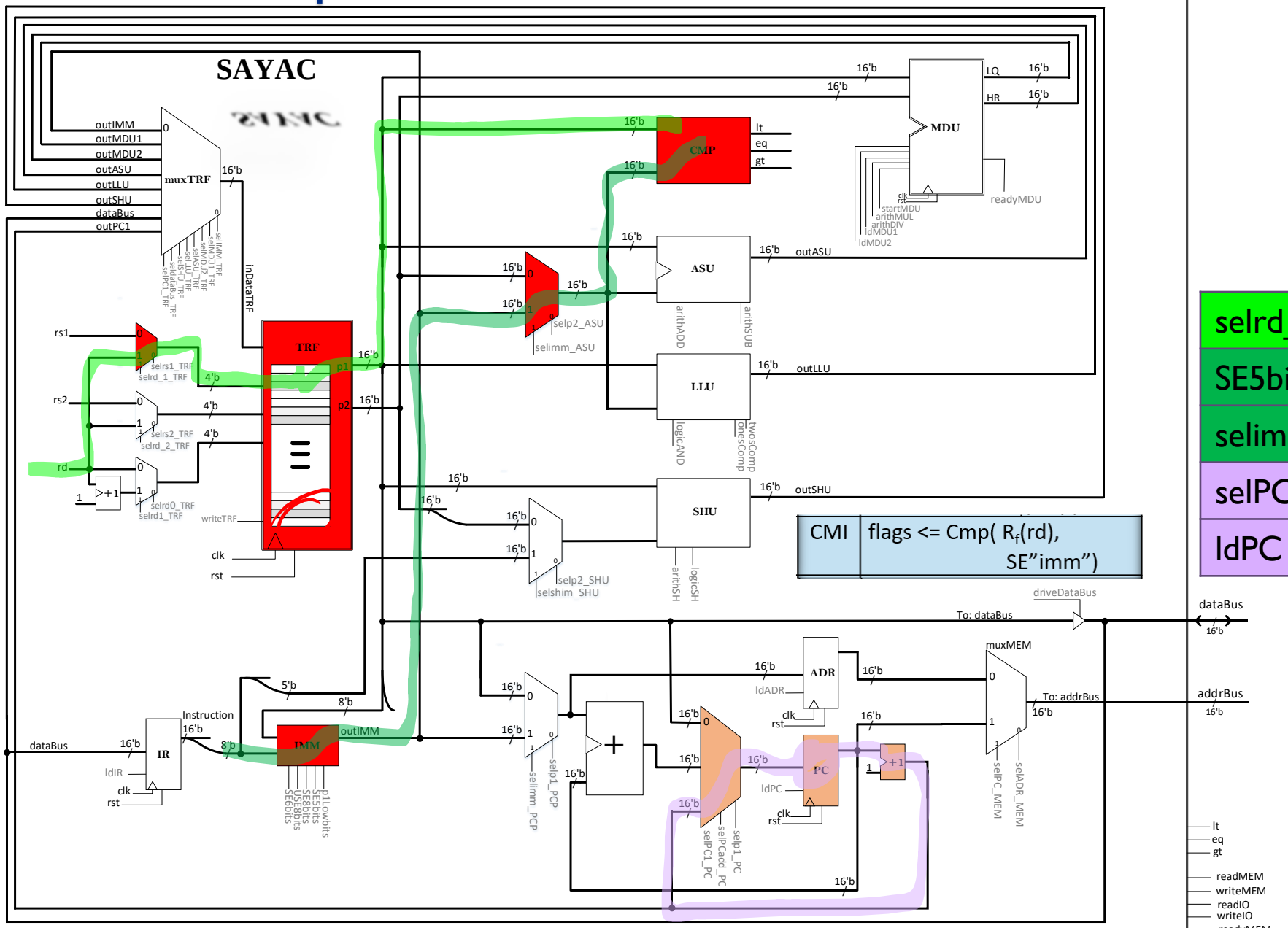
SAYAC Hardware Implementation





SAYAC Hardware Implementation

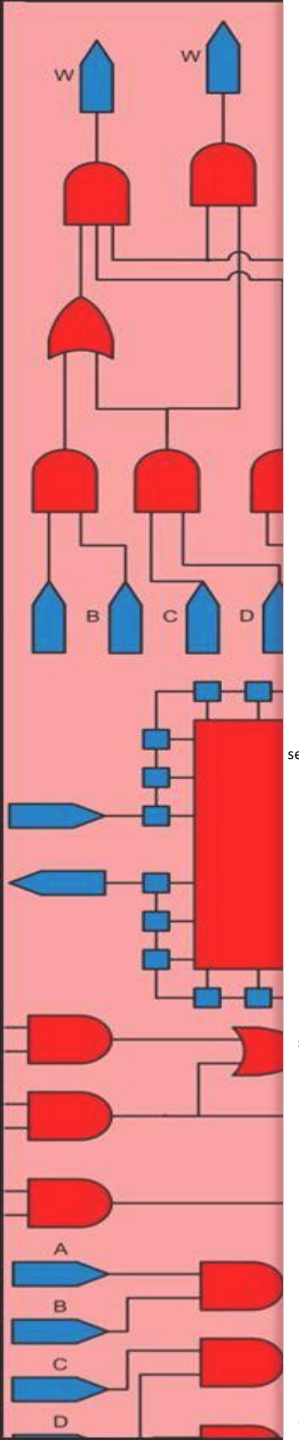


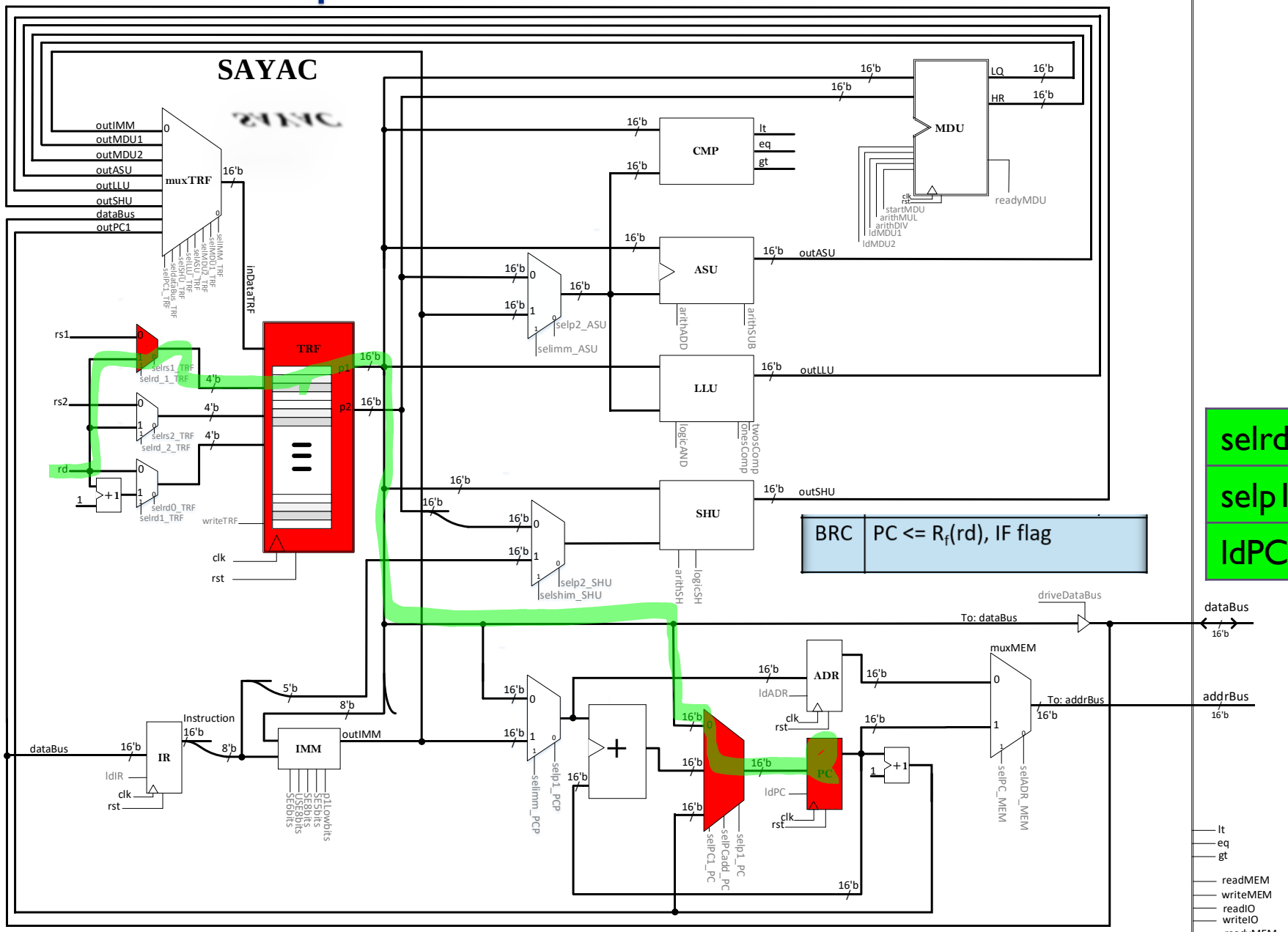


CMI	flags <= Cmp(R_f(rd), SE"imm")

selrd_1_TRF
SE5bits
selimm_ASU
selPC1_PC
ldPC

SAYAC Hardware Implementation

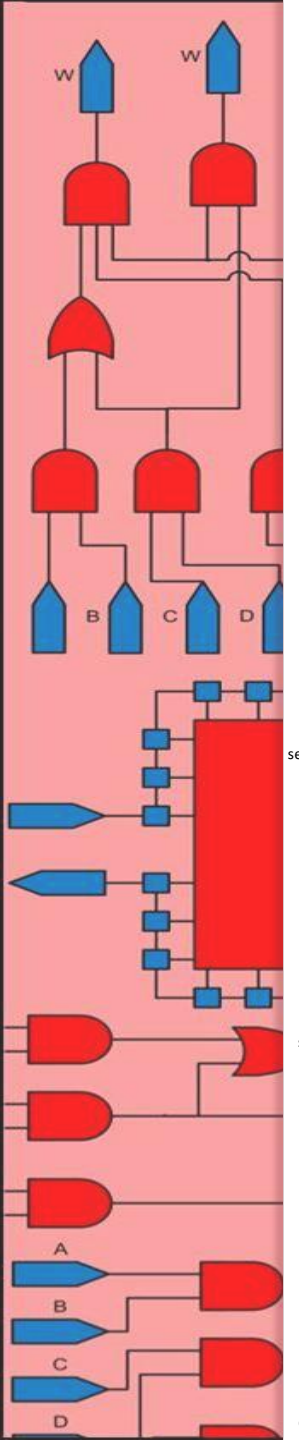


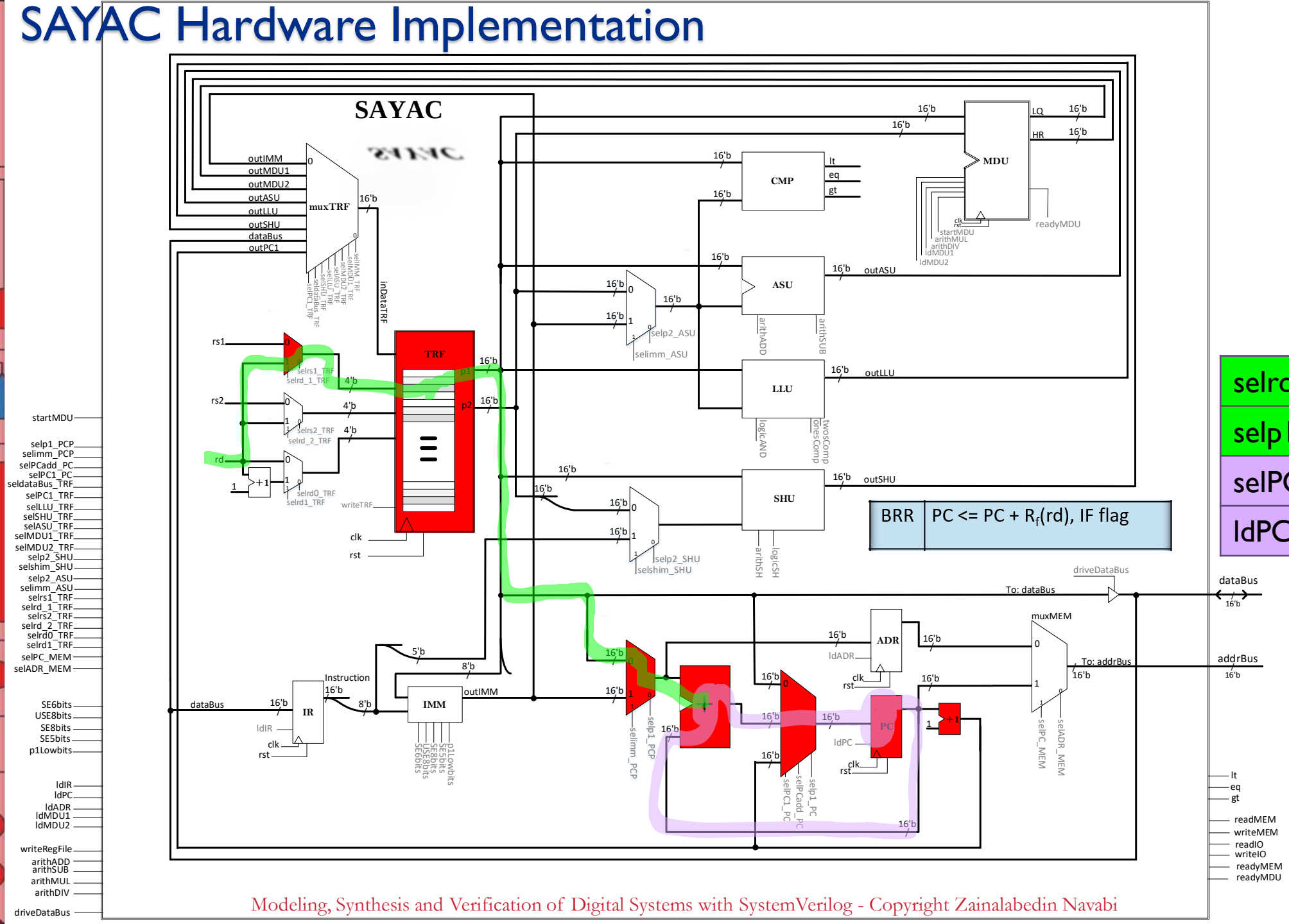


BRC	PC <= R_f(rd), IF flag

selrd_1_TRF
selp1_PC
ldPC

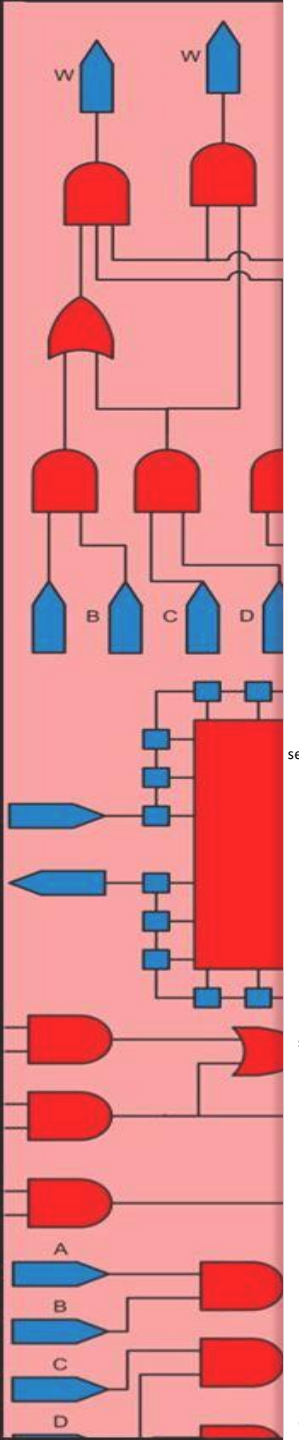
SAYAC Hardware Implementation

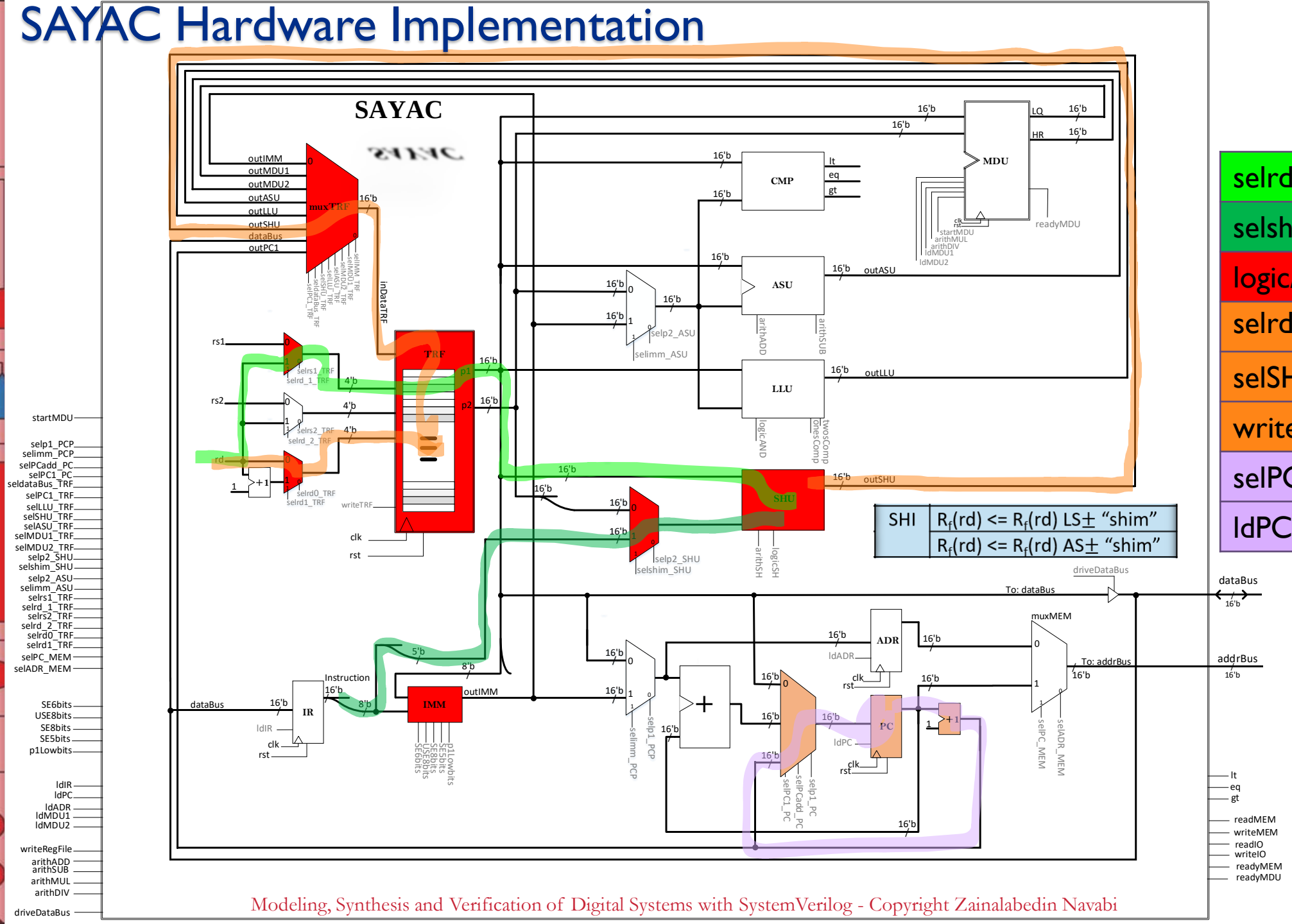




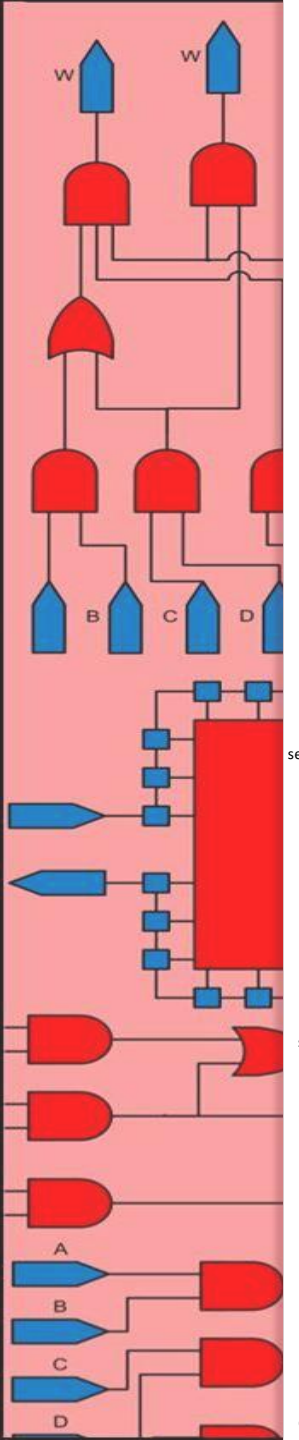
selrd_1_TRF
selp1_PCP
selPCadd_PC
ldPC

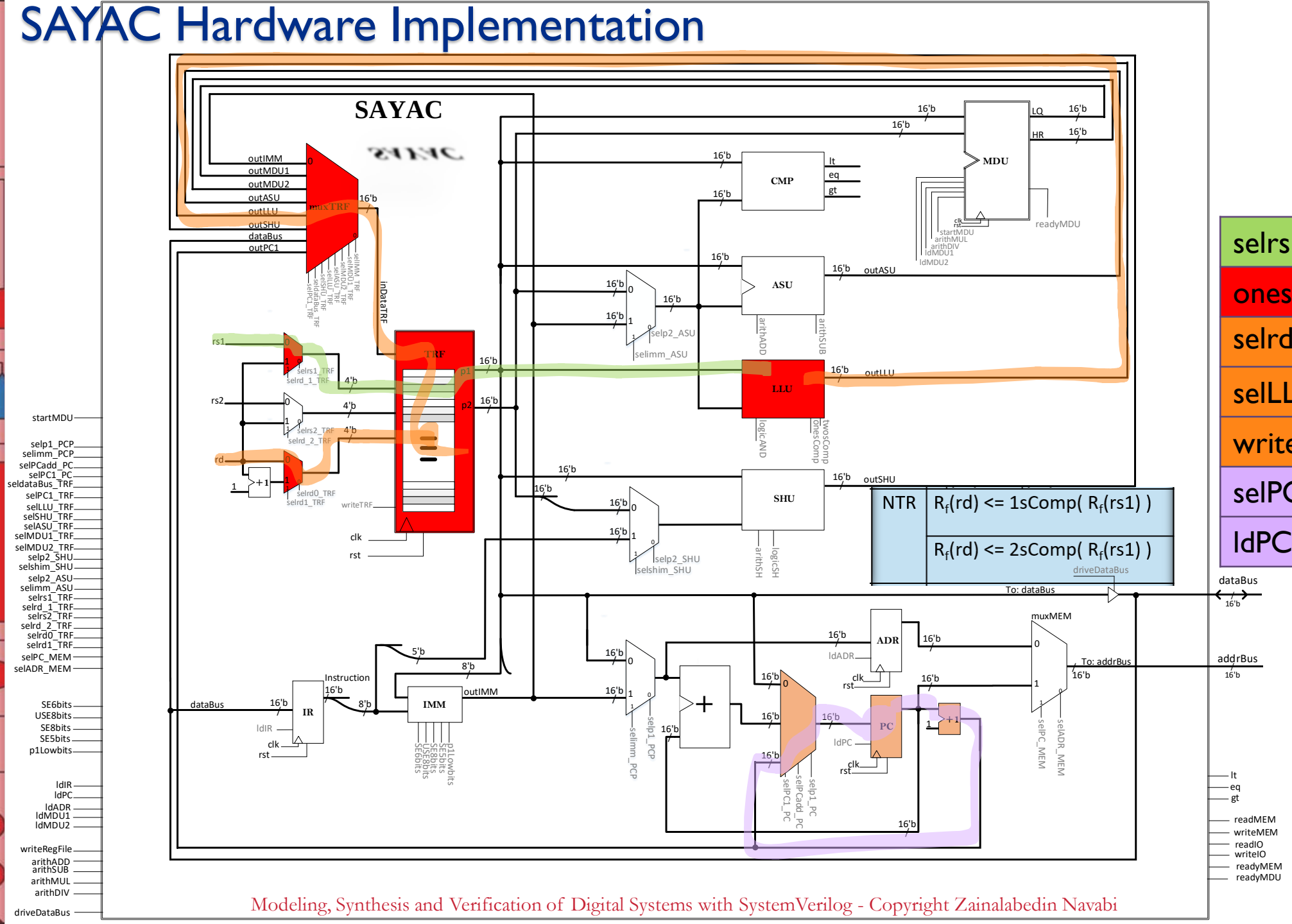
SAYAC Hardware Implementation





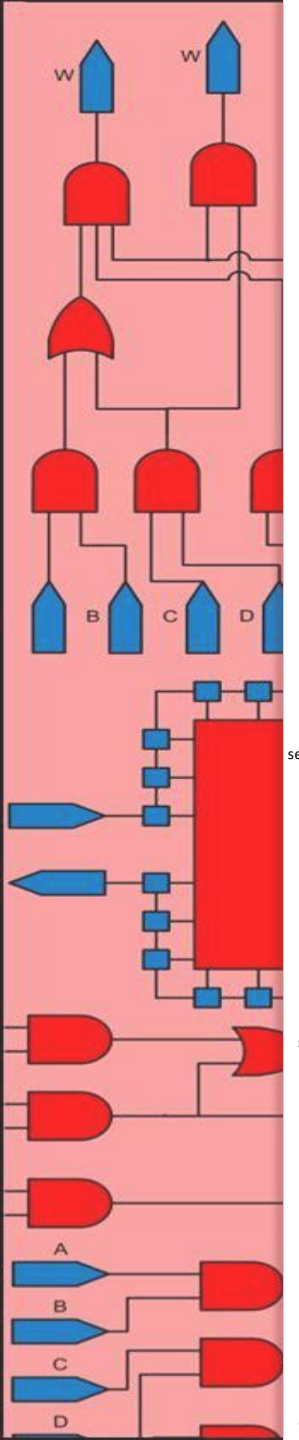
SAYAC Hardware Implementation

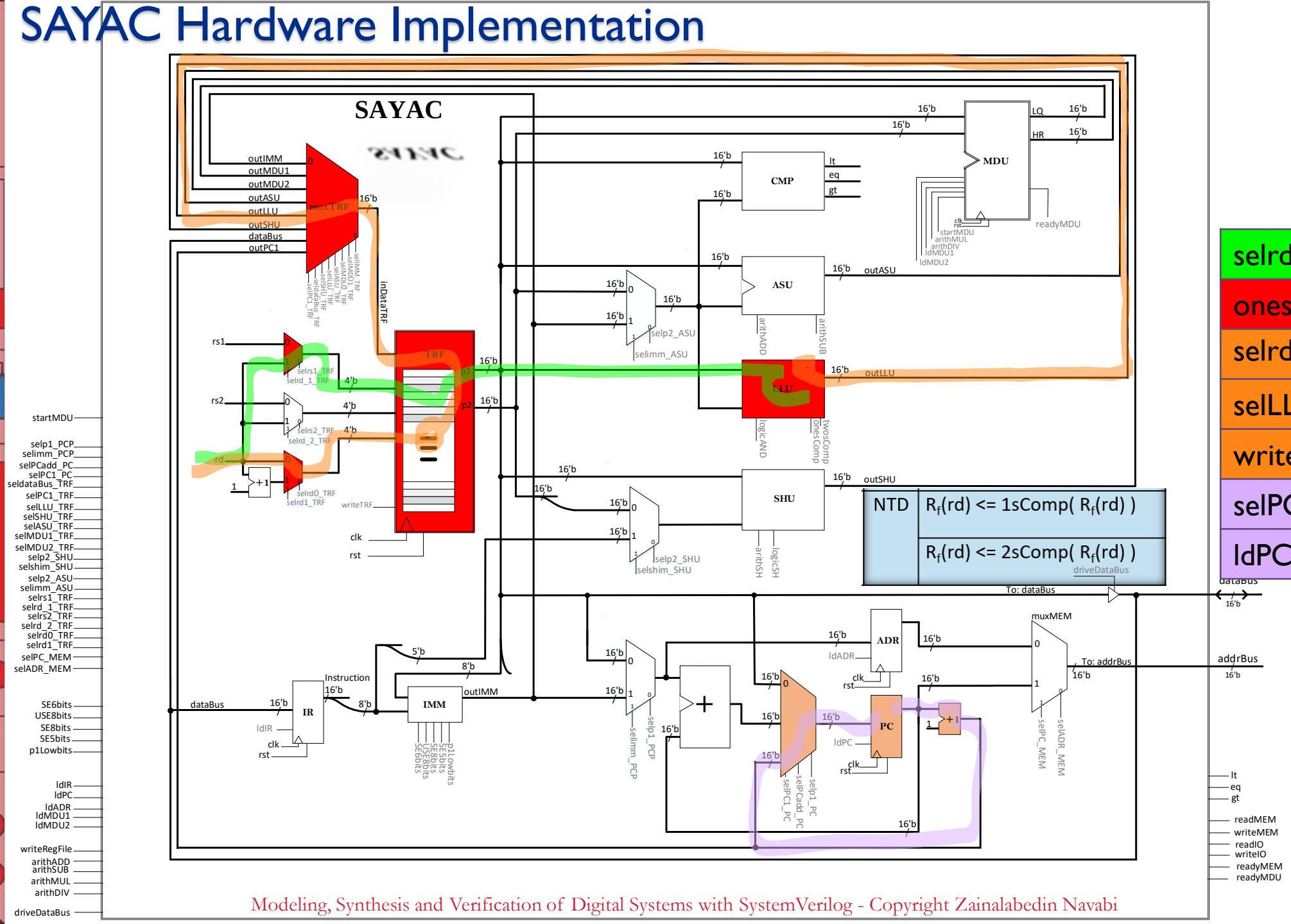




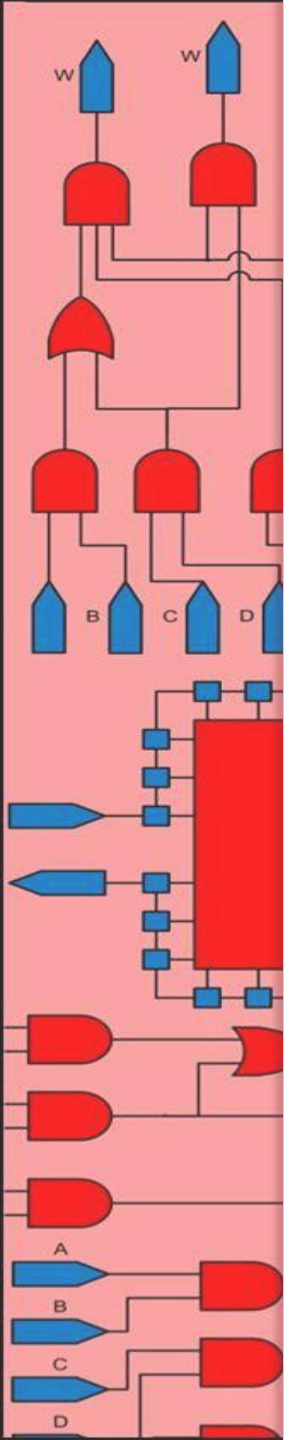
selrs1_TRF
ones/twoscomp
selrd0_TRF
selLLU_TRF
writeRegFile
selPC1_PC
ldPC

SAYAC Hardware Implementation





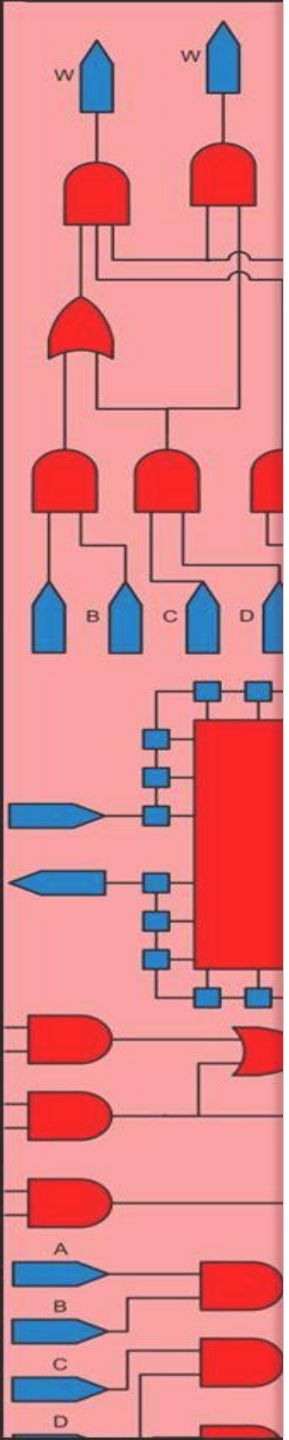
selrd_1_TRF
ones/twoscomp
selrd0_TRF
selLLU_TRF
writeRegFile
selPC1_PC
ldPC



SAYAC Hardware Implementation

Outline:

- Processor and Memory Model
- Processor Model Specification
- Instruction Execution Cycle
- Processor Registers
- Instruction Format
- SAYAC Instructions
- **SAYAC Hardware Implementation**
  - SAYAC Datapath and Instruction flow
  - **Controller**
  - Datapath VHDL Description
    - Control Signals
    - Components of Datapath
    - Bussing
  - Controller VHDL Description
  - Putting SAYAC Together
  - Memory Model
  - Instruction execution and Testing
  - Conclusions



Conclusion

In this topic we have covered: