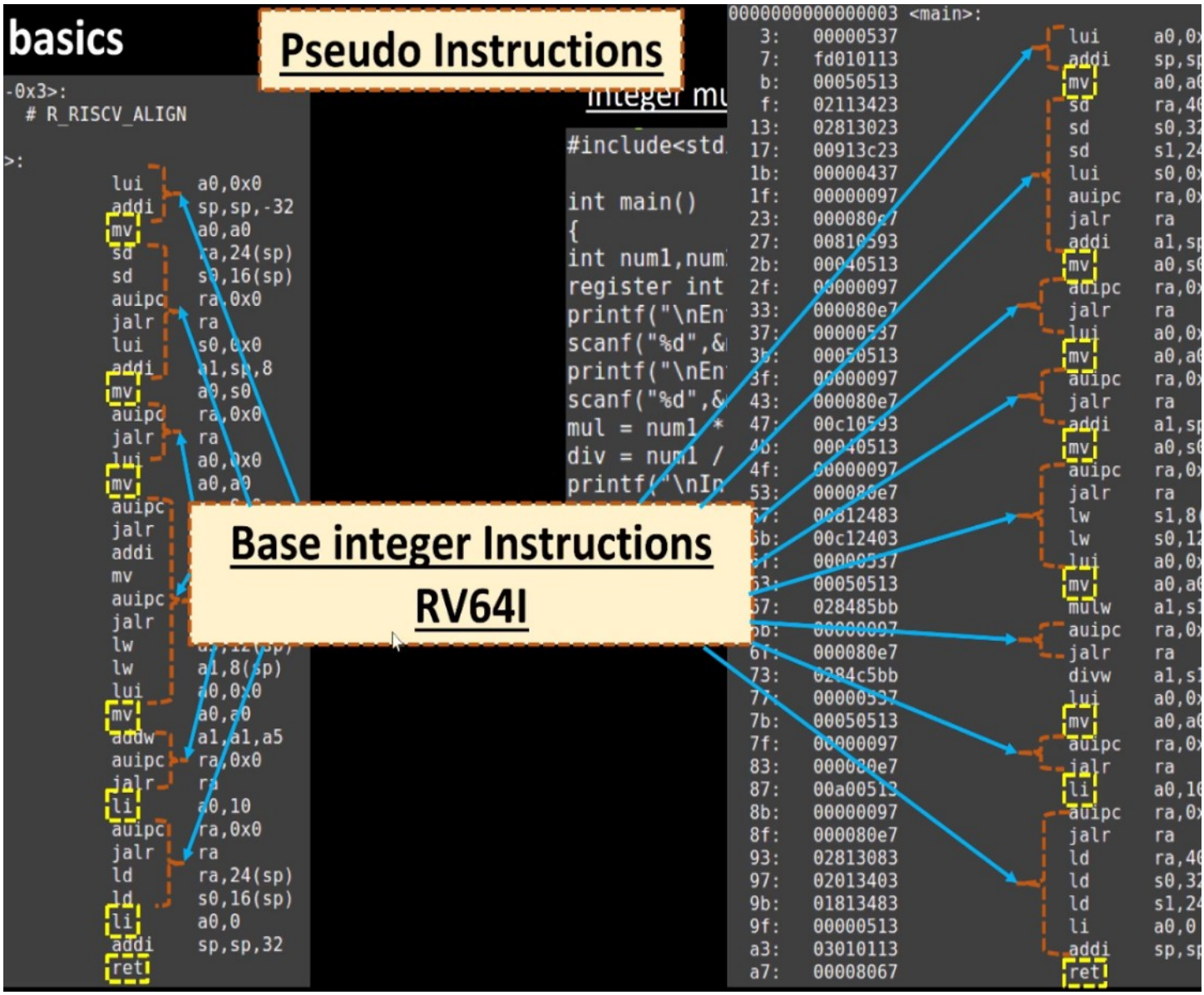
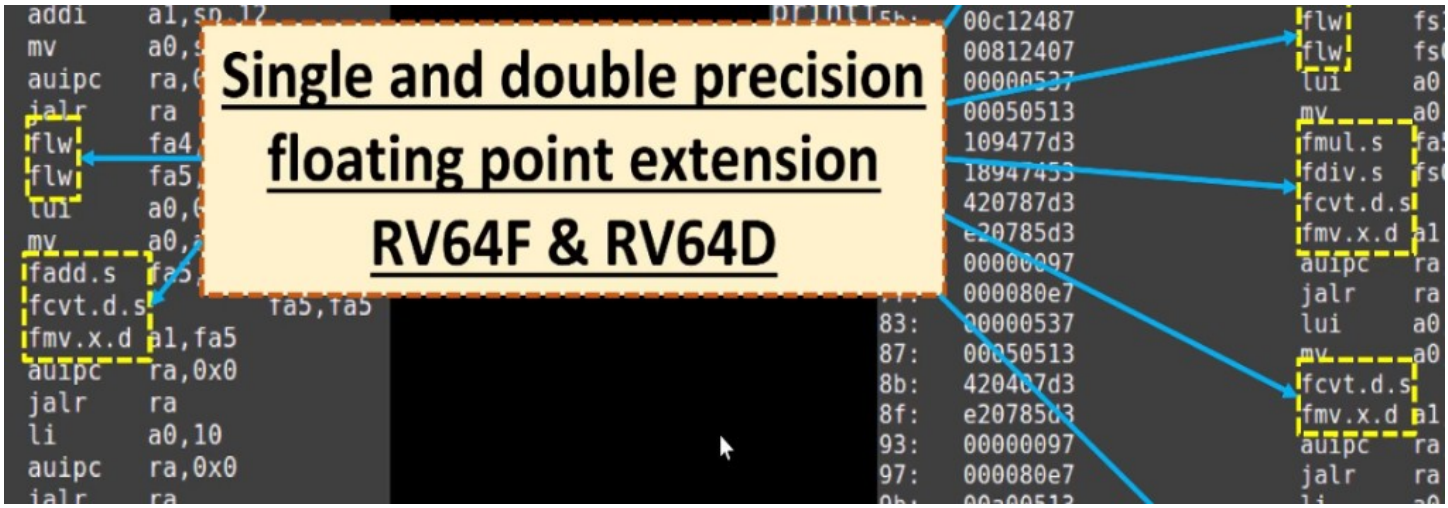
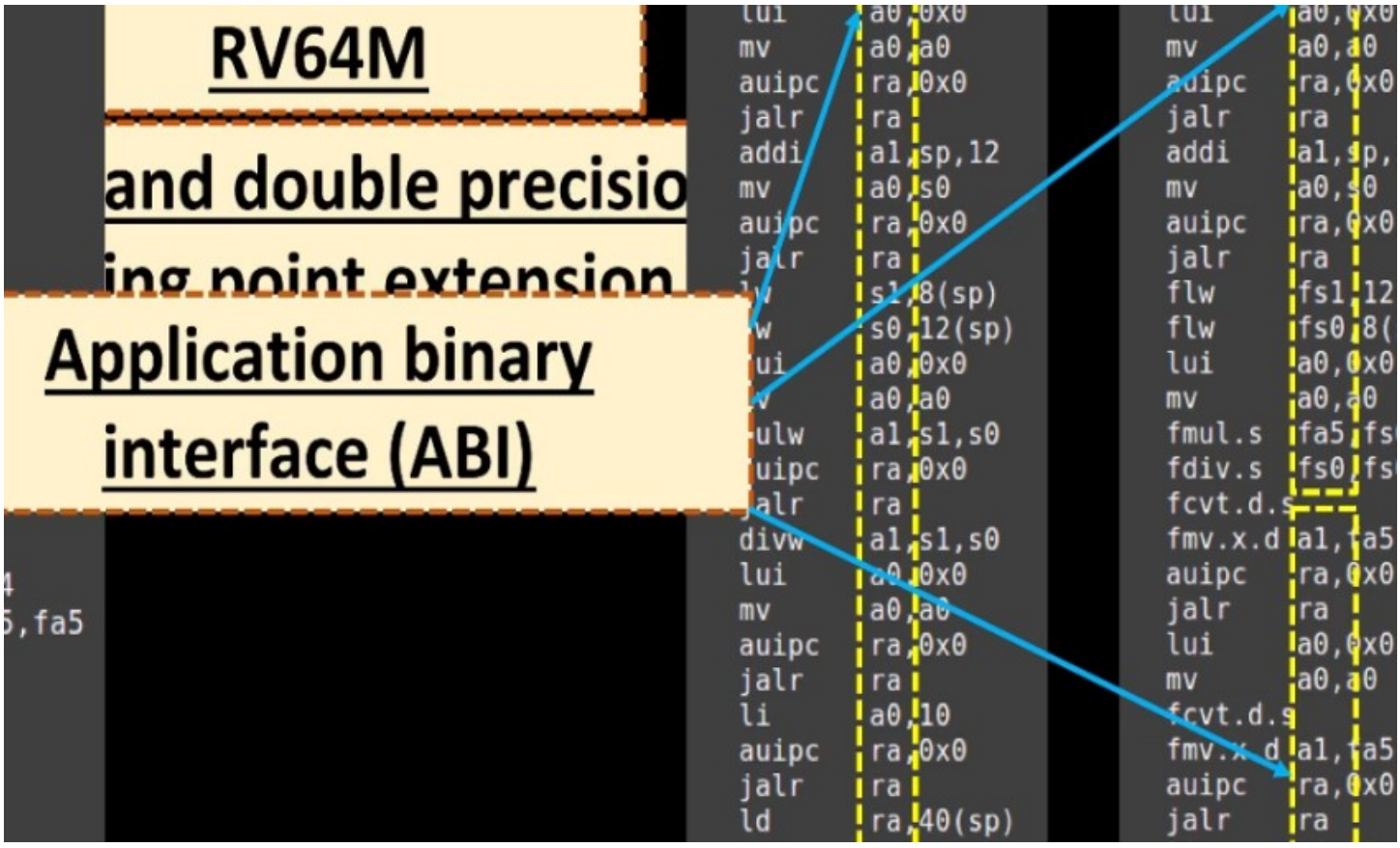
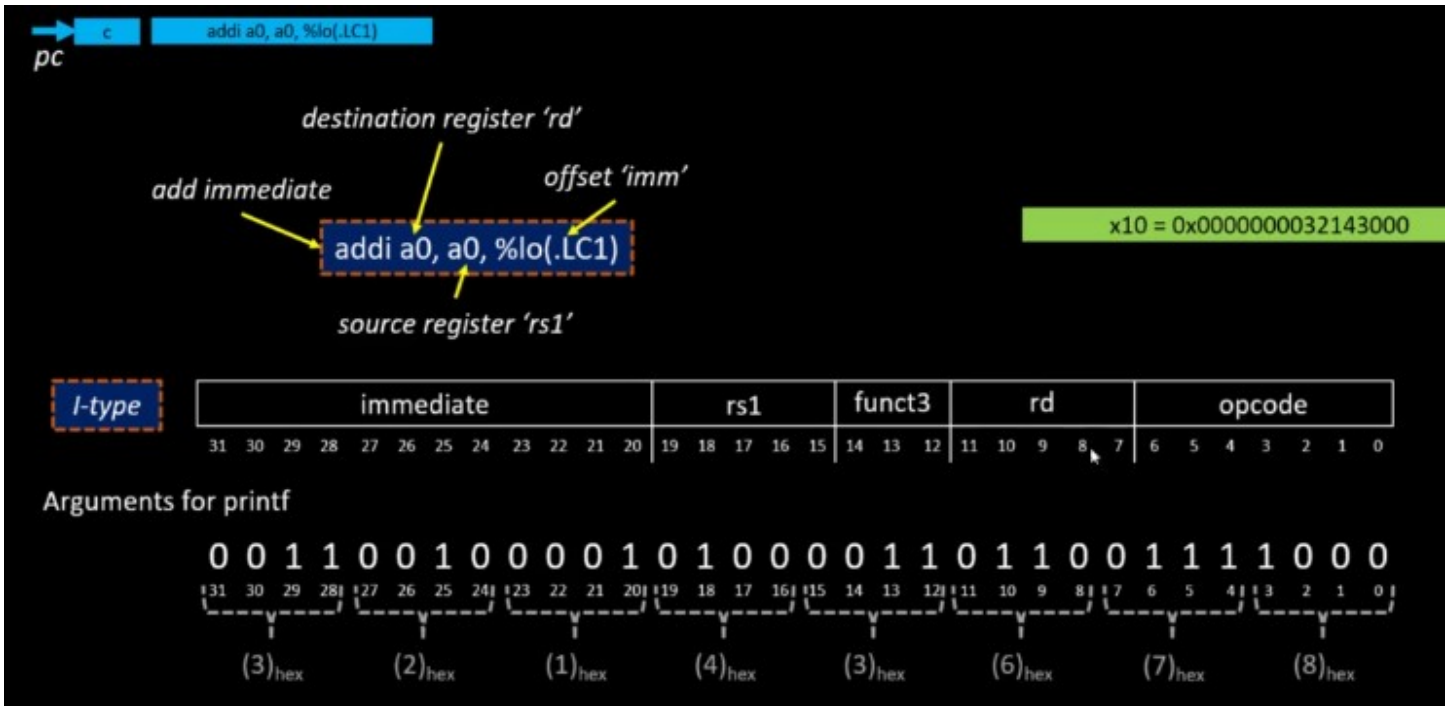
VLSI DESIGN LAB DAY 1 – START WITH BASICS




Single and double precision
floating point extension
RV64F & RV64D


RV64M
and double precisio
Application binary
interface (ABI)

```
kunalg@kunalg-VirtualBox:~$ riscv64-unknown-elf-gcc -Ofast -mabi=lp64 -march=rv64i -o sum1ton.o sum1ton.c
kunalg@kunalg-VirtualBox:~$ gcc sum1ton.c
kunalg@kunalg-VirtualBox:~$ ./a.out
Sum of numbers from 1 to 100 is 5050
kunalg@kunalg-VirtualBox:~$ riscv64-unknown-elf-gcc -Ofast -mabi=lp64 -march=rv64i -o sum1ton.o sum1ton.c
kunalg@kunalg-VirtualBox:~$ spike pk sum1ton.o
bbl loader
Sum of numbers from 1 to 100 is 5050
kunalg@kunalg-VirtualBox:~$ spike -d pk sum1ton.o
: until pc 0 100b0
bbl loader
: reg 0 a2
0x0000000000000000
:
core   0: 0x00000000000100b0 (0x00001637) lui     a2, 0x1
: reg 0 a2
0x0000000000001000
```



2 bit	3 bit	4 bit	
$(00)_{bin}$	$(000)_{bin}$	$(0000)_{bin}$	$(1000)_{bin}$
$(01)_{bin}$	$(001)_{bin}$	$(0001)_{bin}$	$(1001)_{bin}$
$(10)_{bin}$	$(010)_{bin}$	$(0010)_{bin}$	$(1010)_{bin}$
$(11)_{bin}$	$(011)_{bin}$	$(0011)_{bin}$	$(1011)_{bin}$
	$(100)_{bin}$	$(0100)_{bin}$	$(1100)_{bin}$
	$(101)_{bin}$	$(0101)_{bin}$	$(1101)_{bin}$
	$(110)_{bin}$	$(0110)_{bin}$	$(1110)_{bin}$
	$(111)_{bin}$	$(0111)_{bin}$	$(1111)_{bin}$
Total number of Patterns: 2^2: 4	Total number of Patterns: 2^3: 8	Total number of Patterns: 2^4: 16	
'0' to '(2^2-1)' i.e. 0 to 3	'0' to '(2^3-1)' i.e. 0 to 7	'0' to '(2^4-1)' i.e. 0 to 15	

$(-1,152,991,877,645,991,936)_{dec}$

$(0001000000000000010000000000001000000000000000000000000000000000)_{bin}$ ➡ 64 bit

Invert and add '1'–

$(1110111111111111011111111111110111111111111111111111111111111111)_{bin}$

$+ 1$

$(1110111111111111011111111111111000000000000000000000000000000000)_{bin}$

Use *two's complement* representation

Positive numbers – MSB '0', Negative numbers – MSB '1'

$(-2)_{dec}$

$(1111111111111111111111111111111111111111111111111111111111111110)_{bin}$ ➡ 64 bit

$(1 \times -2^{63}) + \ldots\ldots\ldots + (1 \times 2^{31}) + \ldots\ldots\ldots + (1 \times 2^{1}) + (0 \times 2^{0})$

OR

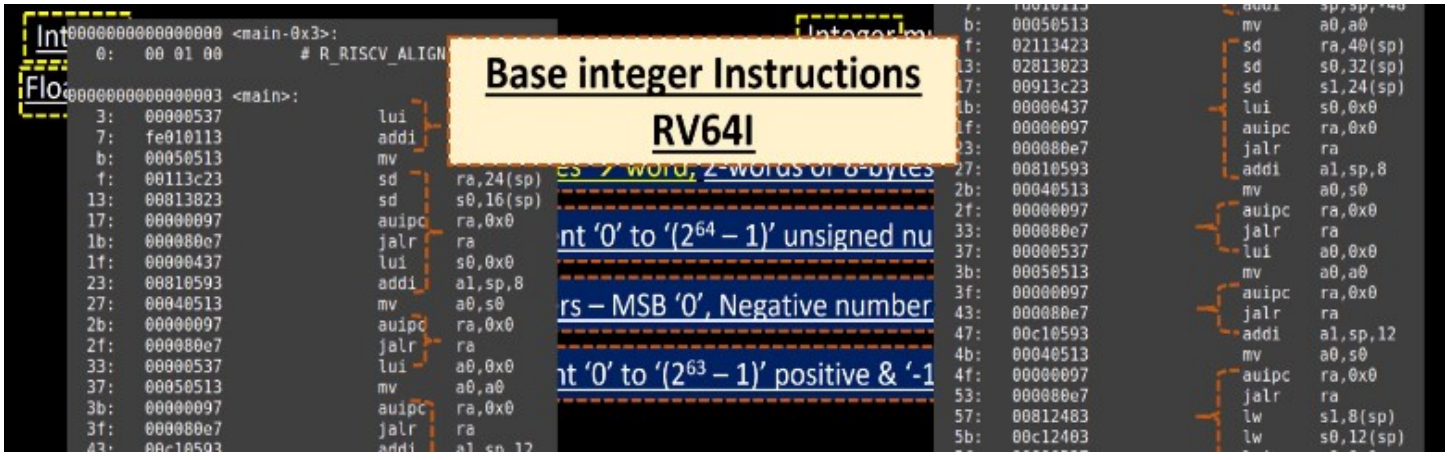
'Invert + 1'

$(-2)_{dec}$

$(0000000000000000000000000000000000000000000000000000000000000010)_{bin}$

WHAT ABT NEGATIVE NUMBERS ➔

These WERE DISCUSSIONS ON:



Lab to find highest and lowest number

Data Type	Memory (bytes)	Format specifier
unsigned int	4	%u
int	4	%d
unsigned long long int	8	%llu
long long int	8	%lld

Application binary interface (ABI)