

VLSI DESIGN LAB DAY 1 – START WITH BASICS

Let's start with basics

Integer addition

```
#include<stdio.h>

int main()
{
    int num1,num2;
    register int sum;
    printf("\nEnter the Number 1 : ");
    scanf("%d",&num1);
    printf("\nEnter the Number 2 : ");
    scanf("%d",&num2);
    sum = num1 + num2;
    printf("\nSum of Numbers : %d",sum);
    printf("\n");
    return(0);
}
```

Integer mul/div

```
#include<stdio.h>

int main()
{
    int num1,num2;
    register int mul, div;
    printf("\nEnter the Number 1 : ");
    scanf("%d",&num1);
    printf("\nEnter the Number 2 : ");
    scanf("%d",&num2);
    mul = num1 * num2;
    div = num1 / num2;
    printf("\nInt multiplication of Numbers : %d",mul);
    printf("\nInt division of Numbers : %d",div);
    printf("\n");
    return(0);
}
```

basics

Pseudo Instructions

Integer mul

```
#include<stdio.h>

int main()
{
    int num1,num2;
    register int mul, div;
    printf("\nEnter the Number 1 : ");
    scanf("%d",&num1);
    printf("\nEnter the Number 2 : ");
    scanf("%d",&num2);
    mul = num1 * num2;
    div = num1 / num2;
    printf("\nInt multiplication of Numbers : %d",mul);
    printf("\nInt division of Numbers : %d",div);
    printf("\n");
    return(0);
}
```

Base integer Instructions

RV64I

```
0000000000000003 <main>:
3: 00000537      lui    a0,0x0
7: fd010113      addi   sp,sp,-32
b: 00050513      mv     a0,a0
f: 02113423      sd     ra,24(sp)
13: 02813023      sd     s0,16(sp)
17: 00913c23      auipc  ra,0x0
1b: 00000437      jalr   ra,ra
1f: 00000097      lui    s0,0x0
23: 000080e7      addi   a1,sp,8
27: 00810593      mv     a0,s0
2b: 00040513      auipc  ra,0x0
2f: 00000097      jalr   ra,ra
33: 000080e7      lui    a0,0x0
37: 00000537      mv     a0,a0
3b: 00050513      auipc  ra,0x0
3f: 00000097      jalr   ra,ra
43: 000080e7      addi   a1,sp,8
47: 00c10593      mv     a0,s0
4b: 00040513      auipc  ra,0x0
4f: 00000097      jalr   ra,ra
53: 000080e7      lui    a0,0x0
57: 00812483      lw     s1,8(s0)
5b: 00c12403      lw     a0,12(s0)
5f: 00000537      lui    a0,0x0
63: 00050513      mv     a0,a0
67: 028485bb      mulw   a1,s1,a0
6b: 00000097      auipc  ra,0x0
6f: 000080e7      jalr   ra,ra
73: 0284c5bb      divw   a1,s1,a0
77: 00000537      lui    a0,0x0
7b: 00050513      mv     a0,a0
7f: 00000097      auipc  ra,0x0
83: 000080e7      jalr   ra,ra
87: 00a00513      li     a0,10
8b: 00000097      auipc  ra,0x0
8f: 000080e7      jalr   ra,ra
93: 02813083      ld     ra,24(sp)
97: 02013403      ld     s0,32(sp)
9b: 01813483      ld     s1,24(sp)
9f: 00000513      li     a0,0
a3: 03010113      addi   sp,sp,32
a7: 00008067      ret
```

Single and double precision floating point extension RV64F & RV64D

```
addi a1, sp, 12
mv a0, s0
auipc ra, 0x0
jalr ra, ra, 0
flw fa4, 0(a0)
flw fa5, 4(a0)
lui a0, 0x0
mv a0, a0
fadd.s fa5, fa4, fa5
fcvt.d.s fa5, fa5
fmv.x.d a1, fa5
auipc ra, 0x0
jalr ra, ra, 0
li a0, 10
auipc ra, 0x0
jalr ra, ra, 0

00c12487 flw fs1, 12(a0)
00812407 flw fs0, 8(a0)
00000537 lui a0, 0x0
00050513 mv a0, a0
109477d3 fmul.s fa5, fs1, fs0
18947453 fdiv.s fs0, fs1, fs0
420787d3 fcvt.d.s fa5, fs0
e20785d3 fmv.x.d a1, fa5
00000097 auipc ra, 0x0
000080e7 jalr ra, ra, 0
83: 00000537 lui a0, 0x0
87: 00050513 mv a0, a0
8b: 420407d3 fcvt.d.s fa5, fs0
8f: e20785d3 fmv.x.d a1, fa5
93: 00000097 auipc ra, 0x0
97: 000080e7 jalr ra, ra, 0
0b: 00000513 lui a0, 0x0
```

RV64M

Single and double precision
floating point extension

Application binary interface (ABI)

```
lui a0, 0x0
mv a0, a0
auipc ra, 0x0
jalr ra, ra, 0
addi a1, sp, 12
mv a0, s0
auipc ra, 0x0
jalr ra, ra, 0
flw s1, 8(sp)
flw s0, 12(sp)
lui a0, 0x0
mv a0, a0
ulw a1, s1, s0
auipc ra, 0x0
jalr ra, ra, 0
divw a1, s1, s0
lui a0, 0x0
mv a0, a0
auipc ra, 0x0
jalr ra, ra, 0
li a0, 10
auipc ra, 0x0
jalr ra, ra, 0
ld ra, 40(sp)

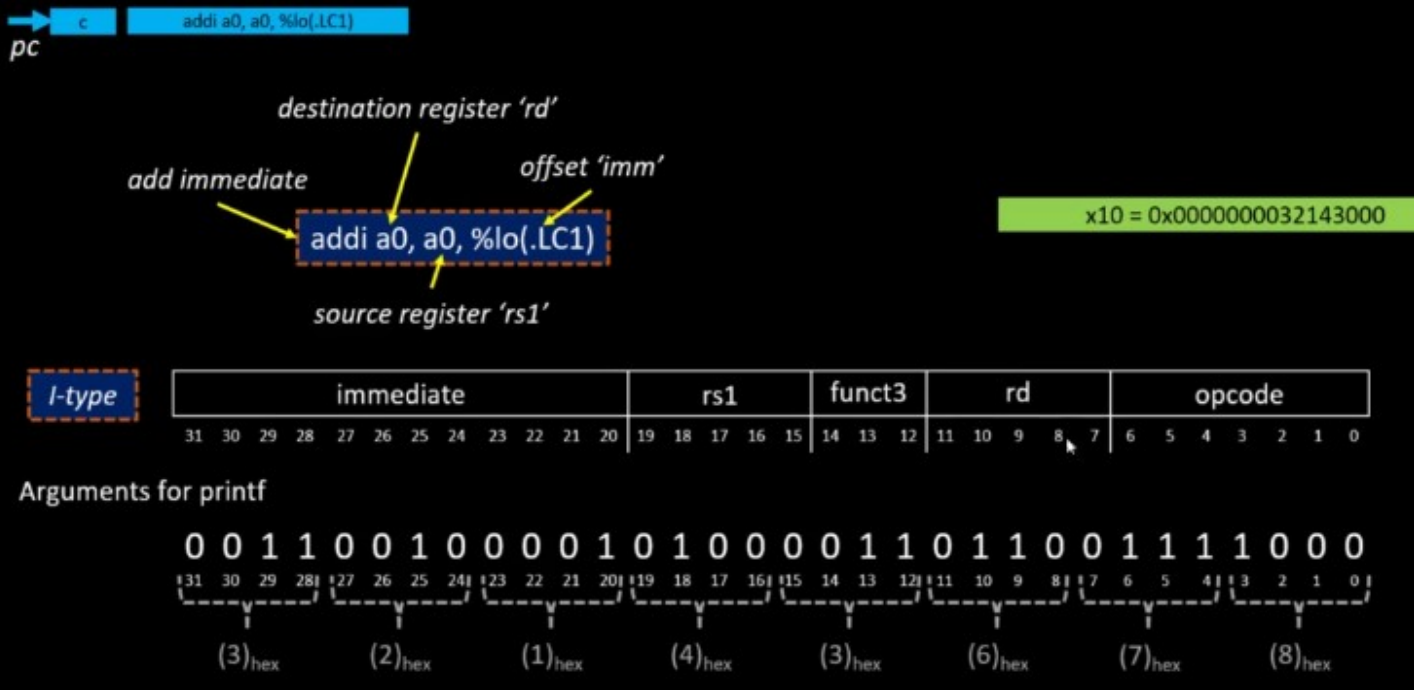
lui a0, 0x0
mv a0, a0
auipc ra, 0x0
jalr ra, ra, 0
addi a1, sp, 12
mv a0, s0
auipc ra, 0x0
jalr ra, ra, 0
flw fs1, 12(a0)
flw fs0, 8(a0)
lui a0, 0x0
mv a0, a0
fmul.s fa5, fs1, fs0
fdiv.s fs0, fs1, fs0
fcvt.d.s fa5, fs0
fmv.x.d a1, fa5
auipc ra, 0x0
jalr ra, ra, 0
lui a0, 0x0
mv a0, a0
fcvt.d.s fa5, fs0
fmv.x.d a1, fa5
auipc ra, 0x0
jalr ra, ra, 0
```



```

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: 0x000000000000100b0 (0x00001637) lui      a2, 0x1
: reg 0 a2
0x00000000000001000

```



2 bit

(00)_{bin}

(01)_{bin}

(10)_{bin}

(11)_{bin}

3 bit

(000)_{bin}

(001)_{bin}

(010)_{bin}

(011)_{bin}

(100)_{bin}

(101)_{bin}

(110)_{bin}

(111)_{bin}

4 bit

(0000)_{bin}

(1000)_{bin}

(0001)_{bin}

(1001)_{bin}

(0010)_{bin}

(1010)_{bin}

(0011)_{bin}

(1011)_{bin}

(0100)_{bin}

(1100)_{bin}

(0101)_{bin}

(1101)_{bin}

(0110)_{bin}

(1110)_{bin}

(0111)_{bin}

(1111)_{bin}

Total number of Patterns: 2^2 : 4
'0' to ' $(2^2 - 1)$ ' i.e. 0 to 3

Total number of Patterns: 2^3 : 8
'0' to ' $(2^3 - 1)$ ' i.e. 0 to 7

Total number of Patterns: 2^4 : 16
'0' to ' $(2^4 - 1)$ ' i.e. 0 to 15

$$(11101111111111101111111111110000000000000000000000000)_{\text{bin}}$$
$$(1 \times 2^{63}) + \dots + (1 \times 2^{31}) + \dots + (1 \times 2^1) + (0 \times 2^0)$$

Integer

```

0: 00000000 <main-0x3>:
0: 00 01 00 # R_RISCV_ALIGN

```

Float

```

0: 00000000 <main>:
3: 00000537 lui ra,24(sp)
7: fe010113 addi s0,16(sp)
b: 00050513 mv ra,0x0
f: 00113c23 sd ra,0x0
13: 00013023 auipc ra,0x0
17: 00000097 jalr ra
1b: 000000e7 lui a1,sp,8
1f: 00000437 addi a0,s0
23: 00010593 mv a0,s0
27: 00040513 auipc ra,0x0
2b: 00000097 jalr ra
2f: 000000e7 lui a0,0x0
33: 00000537 mv a0,a0
37: 00050513 auipc ra,0x0
3b: 00000097 jalr ra
3f: 000000e7 lui a0,0x0
43: 000000e7 mv a0,a0
47: 00c10593 auipc ra,0x0
4b: 00040513 jalr ra
4f: 00000097 mv a0,s0
53: 000000e7 auipc ra,0x0
57: 00012403 jalr ra
5b: 00c12403 lw s0,12(sp)

```

Base integer Instructions

RV64I

ES: 7 word, 2- words or 8-bytes

int '0' to $(2^{64} - 1)$ unsigned number

rs – MSB '0', Negative number

int '0' to $(2^{63} - 1)$ positive & '-1'

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)

