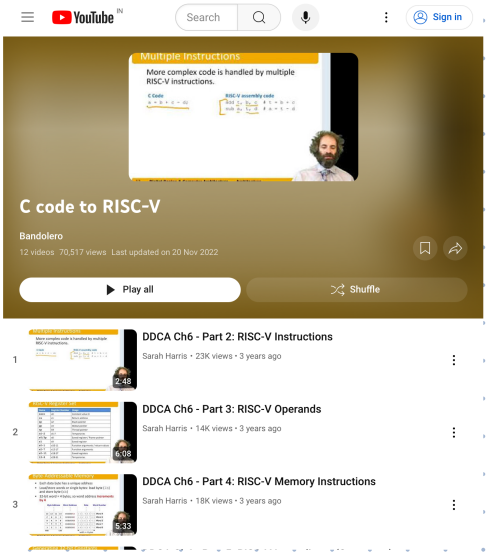
Best resource :



https://youtube.com/playlist?list=PLeFoe3FAv19dfNDjiUeJ6sD91vzPRqbch&si=tua0KsY7DUeApFYK

(only 32-bit)

Memory → Register → operations

constants (Immediates)

Essential instructions :

s0-s11 : Registers

t0-t6 : Temps

zero : constant 0

ra : Read address

Assembly code :

Arithmetic :

add	s0, s1, s2	s0 = s1 + s2
addi	s0, s1, 3	s0 = s1 + 3
sub	s0, s1, s2	s0 = s1 - s2
subi	s0, s1, 3	s0 = s1 - 3
mulh	s0, s1, s2	s0 = s1 x s2 (But only the upper 32-bit)
mul	s0, s1, s2	s0 = s1 x s2 (Only the lower 32-bit)

div	S0, S1, S2	S0 = S1 / S2 (Quotient)
rem	S0, S1, S2	S0 = S1 % S2 (Remainder)

Loading / Storing:

ld : load double
sd : Store double
} 64-bits

lw : load word
sd : Store word
} 32-bits

lb : Load byte
sb : Store byte

Can be both decimal & hex.

ld reg/temp offset(base)
Destination Value

sw reg/temp offset(base)
Value Destination

Example problem:

Q2 [5 Marks]
Assume that the base address of the array A, B, C, and D are already available in registers x20, x21, x22, and x23 respectively. Write the RISC-V assembly code for the following code segment. Please consider the size of each array element as 8 bytes.

A[7]=A[10]+A[22]+A[13]
(A = D, A = B, A = C)

```
ld  s0,  80(x21)
ld  s1,  176(x22)
ld  s2,  104(x23)
add t0,  s0, s1
add t0,  t0, s2
```

ld t0, 56(x20)