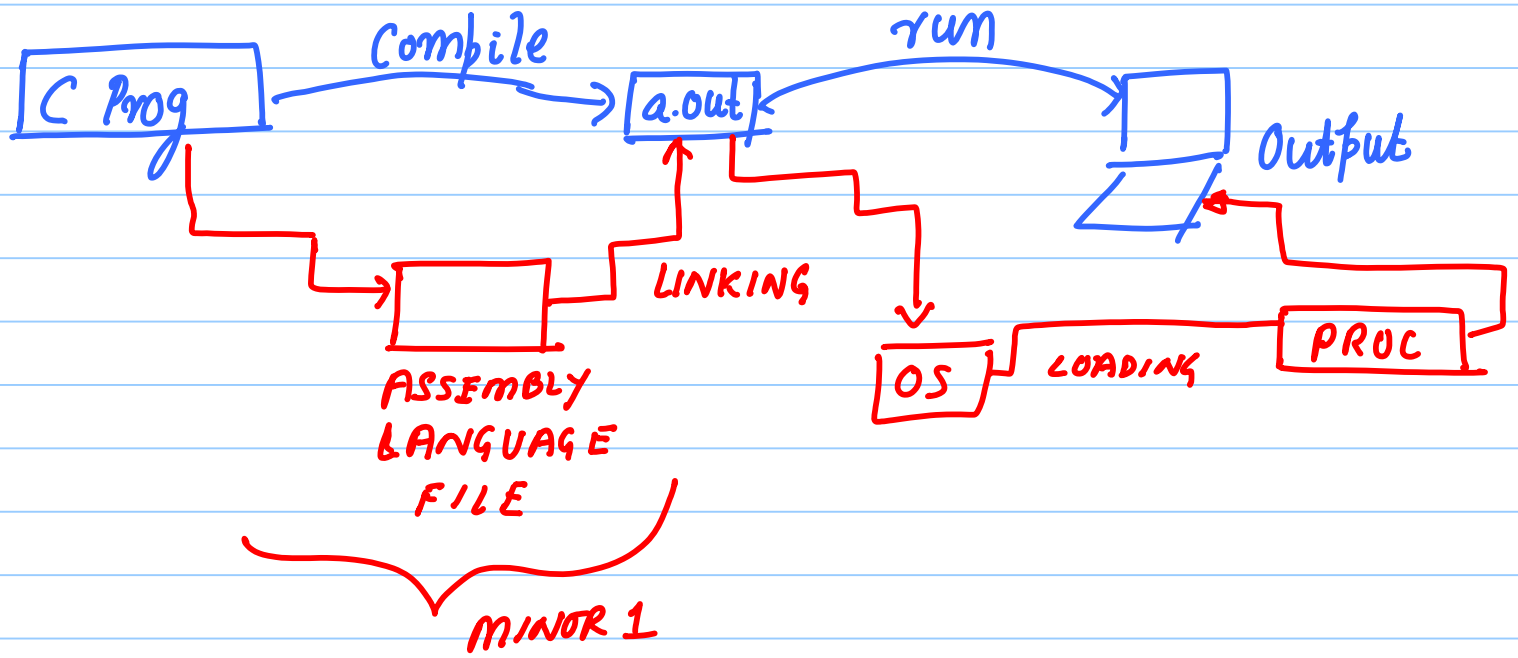
July 25

Note Title 25-07-2012



C Program

```
int a;

int b; c;
a=3; b=5;
c = a+b;
```

R0 → a
R1 → b
R2 → c

Assembly

opcode

```
MOV  R0, #3   (a=3)
MOV  R1, #5   (b=5)
ADD  R2, R0, R1
```

1 2

R2 : dest, R0 : src1, R1 : src2

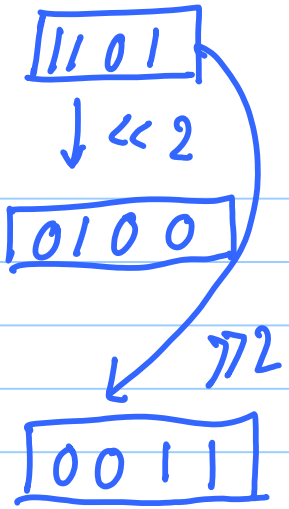
1) map (C variables → Assembly vars)
[register allocation]
[R0 ..... R12]

1) Arithmetic opcodes.

ADD	R0, R1, R2	$R_0 = R_1 + R_2$
SUB	R0, R1, R2	$R_0 = R_1 - R_2$

2) Logical opcodes.

AND	R_0, R_1, R_2	$R_0 = R_1 \& R_2$
ORR	R_0, R_1, R_2	$R_0 = R_1 \mid R_2$
MVN	R_0, R_1	$R_0 = \sim R_1$



LSL R0, R1, R2 — R0 = R1 << R2

LSR R0, R1, R2 — R0 = R1 >> R2

ADD R0, R1, #5 (Immediate)

ADD R0, R1, R2 (Register)

Loops - (IF - ELSE) [Control Flow op-codes]

C program

R0 → a = 3 ;

if (a > 5)

R1 → b = 6 ;

else

Assembly

```
mov  R0, #3
cmp  R0, #5
BGT  .L1
```

b = 8;

```
MOV   R1, #8
B     .L2
.L1
      MOV   R1, #6
.L2
```

B ← Branch.

- B <TARGET>
- B offset (later)

B [suffix]

B + { ← branch
EQ ← equal
NE ← ≠
< ← LT
<= ← LE
> ↔ GT
>= ← GE