Basic Block:

1.   Mark Leaders – Scan Quads Forward:

a.    First Quad is a leader

b.   Target of any JUMP is a leader

c.    Quad Following any Conditional Jump is a Leader

2.   BB includes its leader and all subsequent quads up to but not including next Leader.

Live/Dead Assignments:

For Each BB:

1.   Set all nontemporaries alive in Symbol Table (ST)

2.   Set all temporaries dead in ST

3.   Scan BB backwards from last Quad to Leader performing the following for each quad [op,B,C,A]

a.    Copy Live Next Use from ST to current Quad for A,B, and C

b.   Set Dead Next Use in ST for A

c.    Set Live Next Use in ST = to current Quad number for B&C

4.   Scan Block Forward generating Code

5.   Generate stores for variables in registers that are live on exit and not in memory

s – Scan Quads Forward:

a.    First Quad is a leader

b.   Target of any JUMP is a leader

c.    Quad Following any Conditional Jump is a Leader

2.   BB includes its leader and all subsequent quads up to but not including next Leader.

GETREG – Function

              If B is in register that holds no other names and B is dead then return B’s register and

                         update B’s location field in S.T. and in Register Table

              else

              if there is an empty register then return it

              else

              Choose a register; generate stores for all variables in the register; update descriptors in RT and ST

end;

Case of

Binary Arithmetic/logic Quad:

              Reg:=GETREG

              if B not in Reg then

                         if B is in another register=R then

                                    Generate LR Reg,R

                         else

                                    Generate L Reg,B;

              if B=C then

                         Generate opR Reg,Reg

              else

              if C in register=R then

                         Generate opR Reg,R

              else

                         Generate op Reg,C;

              Update ST for A to indicate it is in Reg and not in memory;

              Update RT to indicate A is in Reg;

              if B(C) is dead and is in a register or ST and RT do not agree update RT to remove B(C);

              Move quads LNU to appropriate entries in ST;

Assignment Quad:

              if B is in reg(R) then

Reg:=R

              else

Reg:=GETREG;

Generate L Reg,B

              Change RT to indicate A is in Reg;

              if B is dead, change RT to remove B from Reg

              Change ST to Indicate A is in Reg and nowhere else

              Move quads LNU to appropriate entries in ST;