1) ინთერმედიეთ რეპრეზენტაციას ვპარსავთ ანტლერის გამოყენებით. შესაბამის გრამატიკა მოცემულია - IR.g4 ფაილში.

2) გაპარსვის შედეგად დაგენერირებულ პარსირების ხეზე გვაქ აგებული ვოლქერ ფეთერნის გამოყენებით კლასი, რომელიც შემდგომ აგებს ბეისქ ბლოკების გრაფს თითოეული ფუნქციისთვის. ასევე გვაქვს სტრუქტურები სხვადსხვა ინფორმაციის შესანახად რომლებსაც ამ ვოლქერშივე ვაგებთ:

    ა. კლასი - რომელიც ინახავს სტატიკურ მეხსიერებას და ყველა მასში შემავალ ფუნქციას.
    ბ. ფუნქცია - ინახავს არგუმენტების და ლოკალური ცვლადებისთვის არსებულ მეხსიერებას, ბეისიქ ბლოკების გრაფს ამ ხისთვის.
    გ. ბეისიქ ბლოკების გრაფი - შედგება ბეისიქ ბლოკებისგან, რომლებიც ინახავენ ამ ბლოკში არსებულ ყველა ინსტრუქციას და დაკავშირებული არიან სხვა ბეისიქ ბლოკებთან.
    დ. მეხსიერება - გვაქ სხვადასხვა ტიპის მეხსიერების კლასები, რომლებიც სხვადსხვა ტიპის მეხსიერებებს აკონტროლებე, რეგისტრები, სტეკი, გლობალური მეხსიერება და ა. შ.

3) შემდგომ გვაქ ორი კლასი ტრანზლაციისთვის და რეგისტრების ალოკაციისთვის, რომელიც ამ იერარქიულ სტრუქტურებს გადაუყვებიან და არჩევენ ინსტრუქციებს და გამოყოფენ რეგისტრებს.

ტრანზლატორი უყვება კლას, მასში შემავალ ფუნქციას და სათითაო ბრძანებას. იგი იყენებს, რეგისტრების ალოკატორ კლას, რომელსაც ატყობინებს ფუნქციაში შესვლა/გამოსვლის და სხვა ივენთების შესახებ. რეგისტრების ალოკატორი კი 3 ტიპის გვაქ და შემოსული ინფორმაციის მიხედვით წყვეტს რა დროს უდნა გამოიყიოს ამა-თუ იმ ფუნქციისთვის რეგისტრი. ამის მიხედვით ტრანსლატორი აგენერირებს საჭირო კომანდების სიას. ტრანსლატორი უზრუნველჰყოფს როგორც სტატიკური, ისე ლოკალური, კონსტანტა რიცხვების ამოღებას და მას ასევე შეუძლია ინტებისა და ფლოუტების კონვერტაცია საჭიროების მიხედვით.

4) ესემბლიში ჩვენი ფრეიმის ლეიაუთი ფუნქციების გამოძახების დროს იდენტურია ქვემოთ მოცემული ლეიაუთის:

Offset from $sp	Usage	
<framesize - 4 >($sp)	Reserved for $ra	Saving Return address
<framesize - 8 >($sp) <framesize - 12>($sp) <framesize - 16>($sp)	Reserved if needed for $s0 Reserved if needed for $s1 Reserved if needed for $s...	Protecting caller's saved data
	... ...	Spill Locations for local $s and $t variables
24($sp) 20($sp) 16($sp)	Reserved if needed for arg ... Reserved if needed for arg 6 Reserved if needed for arg 5	Used to pass args > 4
12($sp) 8 ($sp) 4 ($sp) 0 ($sp)	Reserved for $a3 Reserved for $a2 Reserved for $a1 Reserved for $a0	Used by called function for spilling
Top of call frame : Lower address		

თუმცა ამასთან ერთად სხვა რეგისტრებსაც ვაკონტროლებთ:

1.მაგალითად fp - რომელსაც ვიყენებთ უფრო კომფორტულად, რათა მივწვდეთ ფუნქციის არგუმენტებს, ყოველი ფუნქციაში შემოსვლისას ვინახავთ გამომძახებლის fp-ს და ამის შემდგომ ვუტოლებთ გაუზრდელი სთექის ფოინთერის მისამართს.

2. ასევე ვინახავთ მემორიში არგუმენტების რეგისტრებში არსებულ არგუმენტებს ყოველი ფუნქციის გამოძახების წინ, ხოლო ფუნქციიდან გამოსვლის შემდგომ ვაბრუნბთ არგუმენტებს უკან რეგისტრებში.

5) რეგისტრების ალოკაცია

ა) ნეივ მეთოდი მარტივად უყურებს ყველა გამოყენებულ ცვლადს კონკრეტული ბრძანეებისთვის და როდესაც ტრანზლატორი ფუნქციაში შესვლას ატყობინებს საჭირო ცვლადები მეხსიერებიდან რეგისტრებში გადააქვს, ხოლო როდესაც ფუნქციიდან გამოსვლას შეიტყობს უკან აბრუნებს ამ ცვლადებს მეხსიერებაში.

ბ) ბეისიქ ბლოკებზე დაფუძნებული ალგორითმი, როდესაც ახალ ბეისიქ ბლოკში შესვლაზე შეიტყობს ამოარჩევს ყველაზე ხშირად გამოყენებულ ცვლადებს და მათ რეგისტრებში გადმოიტანს. ამასთან ერთად იტოვებს რამოდენიმე რეგისტრს იმისთვის, რომ მეხსიერებაში არსებული ცვლადების გადმოტანა შეძლოს (საჭიროების შემთხვევაში) რათა არ შეფერხდეს ინსტრუქციების გაშვება

გ) სანამ ალგორითმს განვიხილავთ, ასევე მნიშვნელოვანია ვახსენო, რომ ლაივნეს ანალიზისთვის გამოვიყენეთ fixed point იტერაციის მეთოდი, როგორც ეს ჩვენ კომპილატორების წიგნშია აღწერილი. ანალიზი ხორციელდება პირდაპირ კლასსის ობიექტის გამოყენებით. ანალიზის შედეგი გამოიყენება ინტერფიერენს გრაფის აგებაში, ყველა ცვლადს შორის, რომელსაც აქვს გადაკეთა, იდგმება გრაფის გვერდი. შემდგომ ლექციაში გარჩეული ბრიგსის ოპტიმისტური ალგორითმით ხდება გრაფის თითოეულ წვეროზე რეგისტრის შეწყვილება. ზოიერთ ნოუდს შეიძლება რეგისტრი არ შეუწყვილდეს.იტოვებს რამოდენიმე რეგისტრს იმისთვის, რომ მეხსიერებაში არსებული ცვლადების გადმოტანა შეძლოს (საჭიროების შემთხვევაში) რათა არ შეფერხდეს ინსტრუქციების გაშვება

მცირედი ბაგები გვაქვს, როგორც ტესტებიდან შევიტყვეთ, თუმცა ყველაფრის კოდი ადგილზეა