CSCI-GA.2130-001 – Compiler Construction, Spring 2019

Project 3

Name: Yusen Su
NetID: ys3547
University ID (UID): N15412725
Collaborator Name: Yijian Xie
NetID: yx1891

Work Log

1. Project plan
    - Read the requirement and remarked the implementation hard point
    - Started generating code transferring for expressions and adding scheme pattern
    - Defined transformation rules for statements and statement
    - Debugged and fixed issues of the compiler
    - Addressed comparison code generation and other implementation limitation
    - Wrote the document
2. Project implementation
    - Using hacs and referenced by example files
    - Discussed issues with partners
        i. How to design technique for register allocation
        ii. How to design branch statement code for ARM 32
        iii. How to design local variable stack allocation
        iv. What is the difference way to design ARM 32
    - Shared the implementation code with my partners
        i. Designed and implemented expression (me) statements (Yijian) code generation
        ii. Fix bugs and issues (both)

Remark

1. Expression code generator
    - Function calls
        - Allow function use at most five parameters
        - For each argument, load value to correspond register ($R_0 - R_4$)
        - Make function call like three address code
    - Pointer and reference
        - Allow a pointer value use star to dereference for assignment
            - E.g. $*a \rightarrow LDR\ R_5, [R_5, \#0]$
        - Allow a variable to reference the address of its correspond type
            - E.g. $\&a \rightarrow LDR\ R_4, [R_{12}, -\#4]$
    - Binary operator specification
        - All the generation rules followed the manual of MinARM 32

2. Statement code generator
    - Assignment
        - Allow each assignment for variable will check vt attribute and generate correct load variable code
    - Declaration variable
        - Allow each declaration to assign and allocate on the stack (update offset and vt)
    - Test expression
        - For if and while statement, the test expression only allow to have a type of int or pointer (There is an issue in .MC file, where *string should be string)
    - Return
        - For each return statement, it should assign value to R_0 if necessary
        - Branch to correspond label

Testing

1. Test performance
    - The following files will generate ARM code succeed (correct code)
        - strings.MC
        - strcpy.MC
        - strlen.MC
2. Test limitation
    - Code generating for assignment with comparison operators will not work
        - For example, “a = 1 < 3” will not assign 1 to a
    - Register allocation
        - The code generator we designed is not assigned each expression with any further unused register
            - Each expression, we use R4, R5 mostly because we store and load variable each time
    - Branch condition
        - The condition I generated only allow logical operator (&&, || and !), identifier, comparison operators and constants (my partner will not have this limitation)
        - Any other operators written inside condition expression will not generate branch code
            - The reason is the branch implemented inside the comparison operator

Code

Please find Pr3Yusen.hx file for more details.