2.5. Разработка генератора промежуточного кода

Для последовательности генерации исполняемого файла вводится дополнительный этап генерации промежуточного представления кода, который преобразует дерево синтаксического разбора в ассемблерные команды. На данном этапе работы необходимо разработать алгоритм генерации, выполнить программную реализацию.

2.5.1. Разработка алгоритма генерации

Схема алгоритма для генерации промежуточного кода представлена на рисунке 20

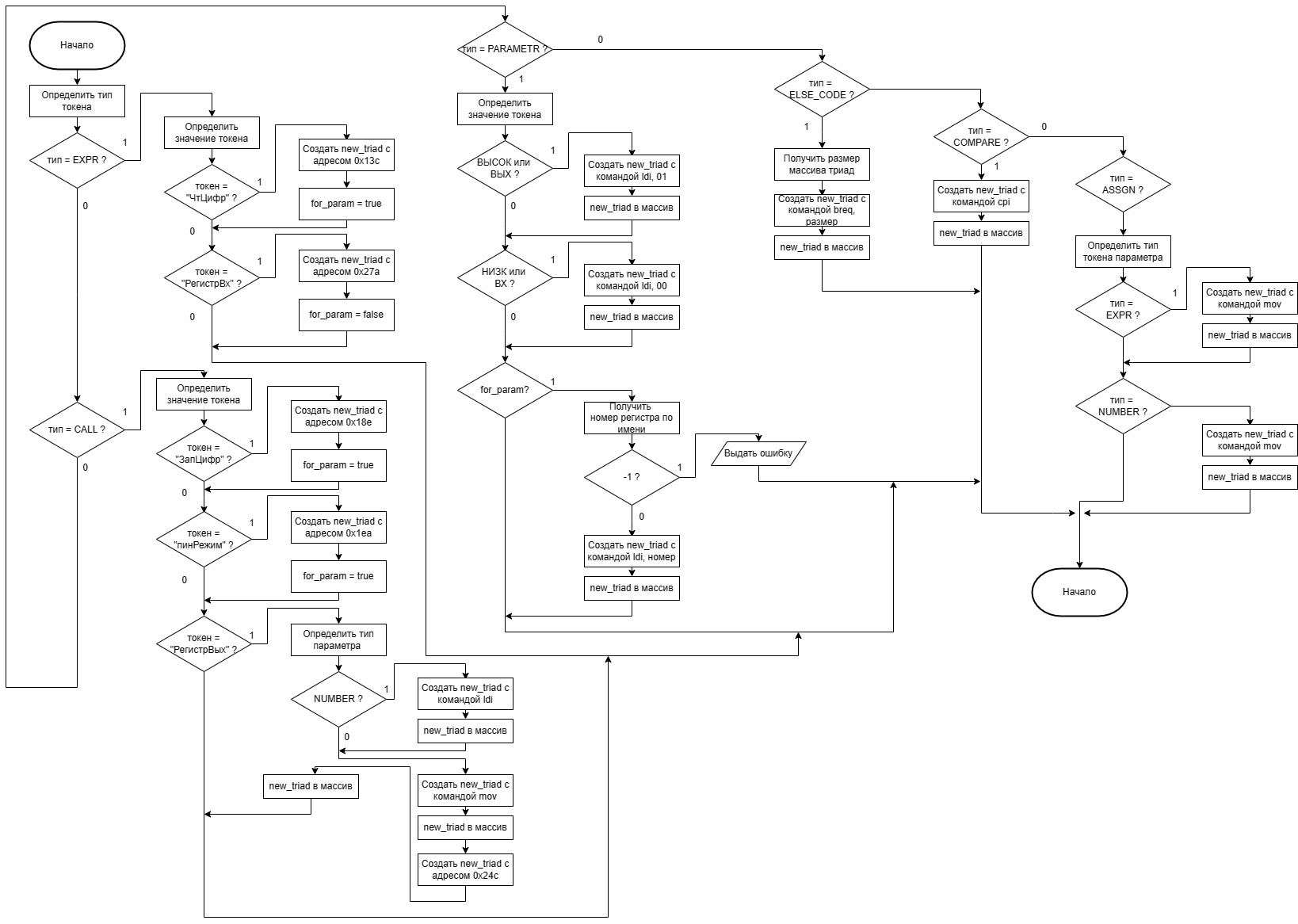


Рисунок 20 – Схема алгоритма генерации промежуточного кода

Первоначально происходит определение типа токена. Если его тип равен EXPR или CALL, то создается new\_triad, в которой в качестве параметров передается «call, <адрес начала функции>». Для повышения читаемости схемы алгоритма описание всех параметров было опущено, но описание основного параметра (адреса – для вызова функции, команды – для операции) оставлено. После определения типа происходит определение имени функции, это необходимо для того, чтобы подставить правильный адрес начала, а также для того, чтобы выполнить проверку на корректность. Так как EXPR возможен только в функциях, которые возвращают значение – чтЦифр и РегистрВх, то во всех остальных случаях ошибка. В случае с CALL все наоборот: функции не должны возвращать значение.

Дополнительно стоит отметить, что в функции РегистрВых происходит определение не всех параметров, а только того, который отвечает за данные. И для корректного отображения в команды ассемблера необходимо определить его тип: если тип – число, то команда ldi, иначе – команда mov.

Также в алгоритме был введен флаг for\_param, необходимый для пропуска разбора параметров, относящихся к функции РегистрВых. Его использование обусловлено тем, что пропуск узла с данной функцией и ее потомками при обходе дерева невозможен, так как это приведет к усложнению разработки и отладки кода.

Продолжая описание разбора параметров, стоит отметить, что при попадании в этот узел первоначально происходит проверка на наличие ключевых слов (НИЗК, ВЫСОК и т.д.) и формирования соответствующей команды. Если же это не ключевое слово, то происходит попытка получить регистр параметра, через таблицу имен. Если такого параметра нет, то выдается ошибка, иначе происходит формирование команды с проверкой флага, о котором было написано выше. Если флаг установлен в состояние, позволяющее выполнить формирование команды, то в качестве параметров в команду передается полученный номер регистра.

Команда для операции ELSE\_CODE – breq, так как в разрабатываемом языке допустимы лишь операции сравнения на равно или не равно. В качестве параметра передается число, означающее кол-во операций, выполняющихся в теле if.

2.5.2. Программная реализация

Для реализации алгоритма был выбран рекурсивный обход дерева разбора. Кроме того, при обходе необходимо выполнять проверку с таблицей имен для получения регистра адреса. В пункте про семантический анализ это было описано и реализовано. И в качестве минимизации времени выполнения программы было принято решение об объединении семантического анализа и генерацию промежуточного кода в одну функцию.

Что касается вычисления параметра для команды breq, то для решения этой задачи был создан массив, временно хранящий набор команд, с момента обнаружения узла условия до момента узла ELSE\_CODE. Далее вычисляется размер списка, и передается в качестве параметра с командой breq.

Листинг программы приведен в приложении В.