Вариант G Модель процессора (1.4)

Разработать приложение, позволяющее промоделировать выполнение на некотором *процессоре* программы на низкоуровневом языке (ассемблере) в виде некоторой объектной модели. Программа представляет собой последовательность команд языка. *Описатель команды*[3] состоит из *описателей операндов* и информации о *типе операции* (исполнение операции реализуется отдельным объектом-функтором). В записи операторов языка могут использоваться *регистры* и *идентификаторы* (связанные с адресами в памяти). Регистр – определяется номеров в блоке регистров; идентификатор – последовательность латинских букв длиной не более 8 символов.

Описатель *унарной команды* содержит следующую информацию: поле метки (может отсутствовать) и указателя на операцию; описателя операнда. Результат выполнения операции помещается в операнд.

Описатель *бинарной команды* содержит следующую информацию: поле метки (может отсутствовать) и указателя на операции; два описателя операндов. Результат выполнения операции помещается в первый операнд.

Описатель оператора *перехода* содержит следующую информацию: идентификатор метки перехода в памяти команд.

Описатель оператора *объявления данных* содержит следующую информацию: поле метки – идентификатор данных (обязательное поле), непосредственный операнд (число). При конструировании объекта данного типа под операнд выделяется место в памяти программ.

Описатель оператора *инициализации потока*[2] содержит следующую информацию: поле метки (может отсутствовать) и идентификатор метки перехода в памяти команд. Новый поток начинает исполнение с заданной метки перехода, в то время как текущий поток продолжает своё исполнение со следующей команды.

Описатель оператора окончания потока одержит следующую информацию: поле метки (может отсутствовать). Программа считается завершённой когда все потоки завершили своё исполнение

Процессор состоит из *устройства управления*, *исполнительных устройств*, выполняющих определенные наборы операторов, *блока регистров общего назначения*, *памяти программ* и *памяти данных*.

При выполнении программы, очередная команда извлекается устройством управления из памяти программ, назначается на свободное исполнительное устройство, после чего команда выполняется исполнительным устройством с установкой блокировок на соответствующие операнды на время выполнения команды. Оператор перехода устройство управления выполняет самостоятельно

Блок регистров характеризуется количеством регистров, поддерживает операции доступа к регистрам по номеру и блокировки регистров. Хранение регистров осуществляется при помощи вектора[1].

Память данных характеризуется объемом, поддерживает операции доступа к данным по идентификатору и блокировки на адреса в памяти, выделение памяти под впервые встретившийся идентификатор.

Исполнительные устройства характеризуются набором и временем выполнения команд.

Память программ считается условно бесконечной, каждая команда занимает одну ячейку в памяти; содержит регистр - счетчик команд, хранящий адрес текущей команды.

Обеспечить также выполнение следующих операций.

- Для модели процессора
  - редактирование модели процессора;
  - выполнить программу.
- Для памяти программ:
  - редактирования программы;
  - показать всю программу;
  - выдать текущую команду;
  - изменить значение регистра счетчика команд.
- Для любой команды:
  - вывести информацию о команде;
  - редактирования команды.
- Для операнда
  - Вернуть значение операнда.
- Для исполнительного устройства:
  - исполнить команду;
  - установить / снять блокировку на ресурс памяти.

1. Шаблонный класс — вектор.
2. Оператор реализуется только при выполнении задания «Многопоточность». Необходимо также обеспечить потокобезопасность всех обращений к регистрам и памяти.
3. Расширяемый класс — команда. Необходимо предусмотреть возможность переопределения основных операций команды в новом классе - наследнике (например, команда условного перехода).