Индекс УДК: 681.3.066, 681.3.08.

ОСОБЕННОСТИ ПРОГРАММНОЙ ЭМУЛЯЦИИ СЕМЕЙСТВА БОРТОВЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН С ОТКРЫТОЙ СИСТЕМОЙ КОМАНД

О.М.Брехов, А.В.Корнеенкова.

Статья посвящена исследованию и разработке метода и методики программной эмуляции семейства БЦВМ с открытой системой команд. Особенностью рассматриваемого семейства БЦВМ является возможность расширения базового набора команд.

Актуальность.

Современные летательные аппараты по своим летно-техническим и аэродинамическим характеристикам превзошли системы управления и в первую очередь бортовые вычислительные системы. С разработкой новейших устройств управления летательных аппаратов, работающих на новой элементной базе, появилась потребность в модернизации вычислительных систем, обеспечивающих управление подобными устройствами. Вычислительные системы летательных аппаратов состоят из одного и более вычислительного модуля. Переход на новую элементную базу обеспечивает улучшение технических и эксплуатационных характеристик вычислительного модуля, в частности, увеличения быстродействия, точности и надежности вычислений, уменьшения весовых и габаритных характеристик. Модернизация предполагает замену вычислительного модуля, которая может быть реализована либо разработкой нового функционального программного обеспечения, либо разработкой образа программного обеспечения заменяемой БЦВМ для использования на целевой БЦВМ с улучшенными характеристиками.

Разработка нового функционального программного обеспечения требует его отладки и проведения испытаний на борту летательного аппарата, что является дорогостоящим процессом как по времени, так и по цене.

Разработка образа программного обеспечения заменяемой БЦВМ для использования на целевой БЦВМ с улучшенными характеристиками может быть решена с помощью системы программной эмуляции. Образ программного обеспечения должен полностью повторить алгоритмы функционирования задач заменяемой БЦВМ, в том числе по временным и точностным характеристикам.

При разработке системы эмуляции БЦВМ существует необходимость в проведении отладки только комплекса программного эмулирования, исключая при этом процесс отладки полученного функционального обеспечения на борту летательного аппарата. Приведенный способ дает очевидную экономию во времени и как следствие экономию в стоимости испытаний.

Решение проблемы модернизации вычислительных модулей с помощью разработки системы программной эмуляции актуально и в случае, когда вычислительные системы на борту летательного аппарата состоят из нескольких вычислительных модулей, что не редкость на сегодняшний день. В связи с универсальностью системы эмуляции данный подход ведет к упрощению организации бортовых вычислительных систем. С другой стороны появляется возможность расширения набора функциональных задач за счет использования целевой БЦВМ с увеличенными вычислительными мощностями.

В качестве заменяемого вычислительного модуля в работе рассматривается семейство БЦВМ ОСК. Системы команд указанного семейства БЦВМ определяется базовым набором операторов, который может быть изменен введением на микропрограммном уровне нового оператора, что является особенностью открытой системы команд. Поэтому система эмуляции должна быть ориентирована на эмуляцию не только базовой системы команд, но и на эмуляцию с учетом появления нового оператора.

Вычислительные операции семейства БЦВМ ОСК организованы на основе дробной арифметики (или чисел с фиксированной запятой перед старшим разрядом). В случае использования целочисленных вычислений и вычислений на основе плавающей точки в целевой БЦВМ появляется проблема эмуляции дробной арифметики.

При организации программной эмуляции возникают проблемы эмуляции системы ввода-вывода, диспетчеризации и синхронизации.

Таким образом, актуальность задачи определяется необходимостью разработки метода и методики программной эмуляции семейства БЦВМ ОСК с учетом программной эмуляции системы ввода-вывода, системы диспетчеризации, аппарата синхронизации и дробной арифметики. Под критериями эффективности работы системы программной эмуляции следует понимать критерии оценки времени и памяти, занимаемой результирующим кодом.

Разработка метода и методики программной эмуляции семейства БЦВМ с открытой системой команд.

Определение системы программной эмуляции.

Наряду с существующими системами программной эмуляции на основе интерпретации и компиляции [1, 2] для программной эмуляции семейства БЦВМ ОСК предлагается использовать систему программной эмуляции транслирующе-компилирующего типа. Для обоснования сделанного выбора рассмотрим особенности организации всех систем программной эмуляции.

Системы, базирующиеся на интерпретации, предназначены в основном для пошагового исполнения имитационных программ. Интерпретатор последовательно исполняет команды интерпретируемого ПО. Интерпретаторы обладают низким быстродействием и не могут эффективно использоваться в системах реального времени. Использовать интерпретаторы можно при пошаговой эмуляции вычислительной системы в процессе отладки.

Организация системы эмуляции на основе компилятора основана на предварительной компиляции программ моделирования в машинный код целевой БЦВМ и последующем прогоне модели. Функциональные программы, реализованные в виде машинных инструкций, получаются компактными и быстродействующими. Системы, основанные на компиляции, предназначены для массового эффективного моделирования, при котором за один запуск системы эмуляции выполняется большое число тактов (от десятков тысяч до миллионов) исследуемой вычислительной системы. Имитационные программы предварительно компилируются в объектные

модули в формате операционной системы целевой БЦВМ. Основным достоинством такой организации эмуляторов является высокая скорость процесса моделирования. На таких эмуляторах можно выполнить сразу все исследуемое ПО.

Недостатками таких систем является их сильная привязка к платформе целевой БЦВМ и более сложная организация отладочных функций.

Предлагаемый способ построения систем программной эмуляции транслирующе-компилирующего типа основан на использовании в своей реализации транслятора на промежуточный язык с последующим применением компилятора.

Рассмотрим структурную схему системы программной эмуляции транслирующее-компилирующего типа, представленную на рис. 1.

Использование промежуточного кода совместно с трансляторами решает следующие задачи:

- независимость транслятора от архитектуры целевой БЦВМ, т.е. возможность переносимости функционального ПО;
- возможность модернизации функционального ПО, т.е. расширение и замена отдельных функциональных задач, а также оптимизация алгоритмов функционирования задач;
- возможность отладки функционального ПО на инструментальной ЭВМ.

При использовании в качестве промежуточного языка традиционных языков, например, ЯВУ С++, в качестве компилятора под целевую БЦВМ может быть использован стандартный компилятор, например, компилятор ЯВУ С++, с добавлением дополнительных библиотек для обеспечения совместимости с целевой БЦВМ.

Предлагается следующая структура системы трансляции на промежуточный язык, представленная на рис.2.

Транслируемое программное обеспечение БЦВМ ОСК представляет собой дамп памяти последовательно выполняемых операторов транслируемой системы команд, хранящихся в машинных кодах.

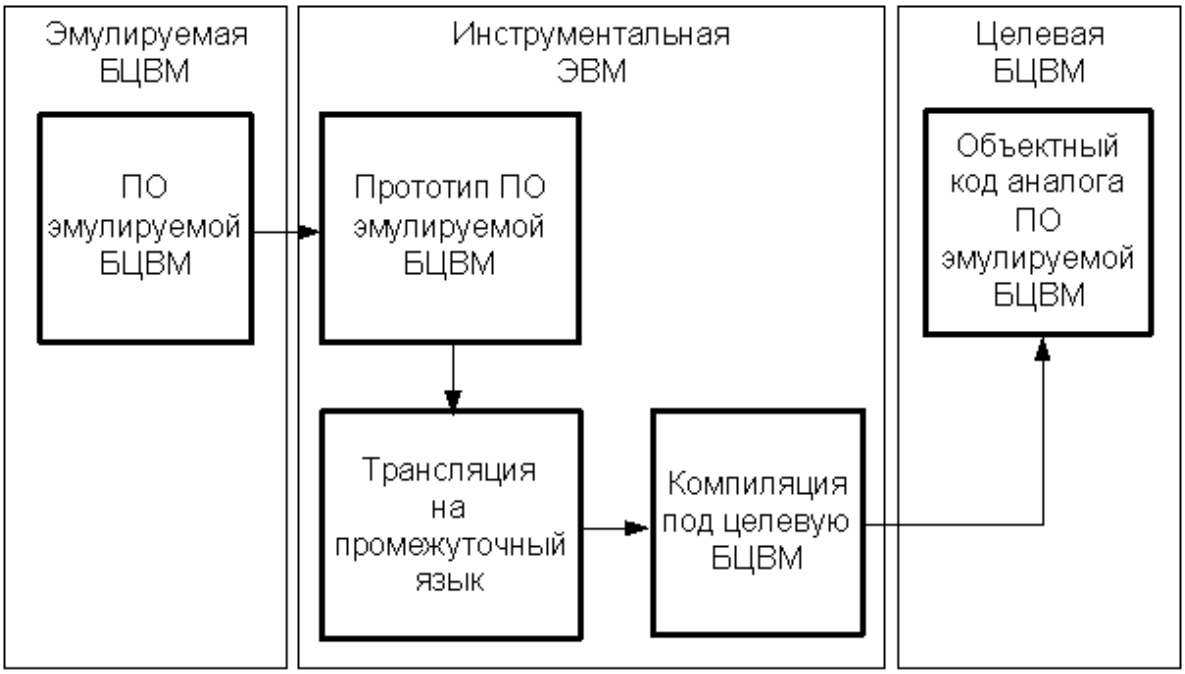

Рис. 1. Структура системы эмуляции путем трансляции с последующей компиляцией

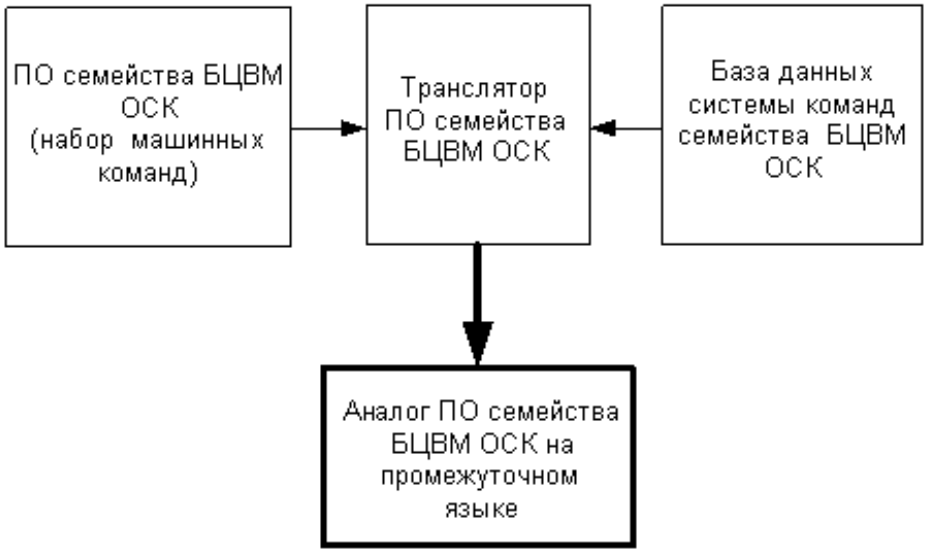

Рис. 2. Организация системы трансляции ПО на промежуточный язык

Процесс трансляции на промежуточный язык с учетом эмуляции систем ввода-вывода, диспетчеризации и аппарата синхронизации заключается в выполнении следующих этапов:

1. формирование адресной трассы;
2. определение схемы эмуляции системы ввода-вывода, диспетчеризации и аппарата синхронизации;

3. распознавание оператора ОСК;

4. представление оператора на промежуточном языке.

Формирование адресной трассы.

Формирование адресной трассы может быть выполнено с помощью следующих способов:

1) способ формирования адресной трассы при последовательном анализе памяти, который основан на анализе памяти функционального ПО без выполнения операторов переходов (рис. 3).

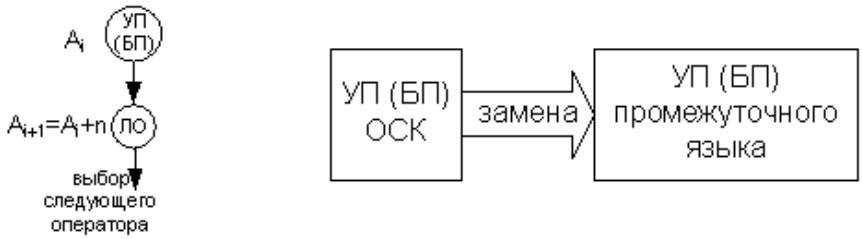

Рис. 3. Формирования адресной трассы при последовательном анализе памяти

При использовании данного способа операторы перехода заменяются соответствующими командами промежуточного языка, при этом адрес следующего транслируемого оператора будет определяться следующим образом: $A_{i+1}= A_i+n$, где: A_i - адрес любого транслируемого текущего оператора; A_{i+1} - адрес следующего транслируемого оператора; n - длина текущего оператора в байтах.

Недостатком данного метода является возможность попадания в область хранения данных, что приводит к сбою работы системы.

2) способ формирования адресной трассы по функциональным задачам, основанный на анализе памяти функционального ПО с частичной интерпретацией операторов переходов (рис. 4).

При использовании данного способа может произойти ситуация игнорирования оператора безусловного перехода в случае, если оператор не обрабатывался (рис. 4.а), и ситуация дополнительного использования команды безусловного перехода промежуточного языка, когда следующий оператор уже был обработан (рис 4.б). При трансляции операторов условного перехода используется стековый механизм для хранения базовых адресов и адресов переходов. Адрес следующего транслируемого оператора в случае трансляции линейных операторов будет

определяться так же, как и в первом способе, а в случае нелинейных операторов как $A_{i+1}= M$, где: A_{i+1} - адрес следующего транслируемого оператора; M - метка или адрес перехода, задаваемый в текущем операторе с помощью непосредственной или базовой адресации.

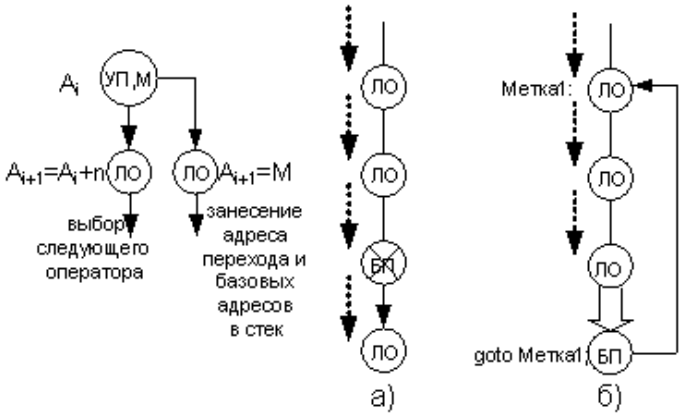

Рис. 4. Формирования адресной трассы по функциональным задачам

Недостаток способа – более сложная реализация транслятора за счет усложнения схем трансляции операторов переходов и установки базовых и индексных регистров, при этом в процессе трансляции исключается возможность попадания в область хранения данных, что является достоинством данного способа и несомненным преимуществом перед первым способом. Поэтому для формирования адресной трассы при организации программной эмуляции семейства БЦВМ ОСК предлагается использовать способ с анализом ПО по функциональным задачам.

Распознавание оператора ОСК.

Предложены следующие способы распознавания оператора ОСК:

1) способ распознавания оператора по коду оператора, основанный на подстановке вместо оператора его интерпретирующей функции на промежуточном языке. При использовании данного способа в случае появления нового оператора необходимо производить анализ базовой системы команд семейства БЦВМ ОСК и изменять исполняемый модуль транслятора. Таким образом, допускается большая степень внедрения пользователя в процесс трансляции, что может привести к ошибкам при трансляции нового оператора. Данный факт является существенным недостатком способа.

2) способ распознавания оператора по его функциональному представлению, основанный на замене оператора его функциональным представлением на языке описания

операторов и дальнейшем переводе на промежуточный язык. Появляется возможность конструировать новый оператор на специализированном языке описания. Использование данного способа дает возможность автоматической трансляции ПО при наличии нового оператора. При этом исполнительный модуль транслятора остается постоянным и не зависит от изменения ОСК, что является достоинством системы. Недостатком метода является большее время работы транслятора. Пренебрегая временем работы транслятора для организации трансляции на промежуточный язык предлагается использовать способ распознавания оператора по его функциональному представлению, как эффективного способа, позволяющего осуществлять автоматическую трансляцию при появлении нового оператора.

Организация функций интерпретации операторов ОСК и оценка по критерию памяти под результирующий код.

Операторы ОСК [3] можно разделить на простые и сложные. Простой оператор – это оператор, которому можно поставить в соответствие аналогичную команду промежуточного языка. Сложный оператор может быть представлен двумя и более командами промежуточного языка. Интерес представляет исследование способов представления сложных операторов на промежуточном языке по критерию занимаемой памяти.

Имеется функциональная задача, состоящая из последовательности n операторов, каждый из которых может быть разбит на некоторые операции. В процессе трансляции на промежуточный язык i-ая операция заменяется на эквивалентный ей код на промежуточном языке. Функции интерпретации операций транслируемой системы команд могут быть организованы двумя способами:

1) каждой операции ставится в соответствие своя интерпретирующая функция, организованная в виде макроса (М) (рис. 5).

Рис. 5. Организация функций интерпретации сложного оператора в виде макроса

Объем памяти V_M, отводимый под функциональную задачу после ее трансляции при использовании макросов, можно оценить следующим образом:

$$V_M = \sum_{j=1}^{L} (k_{Mj} \cdot \sum_{i=1}^{N} n_j \cdot m_{ij}),$$

где:

k_{Mj} – количество простых инструкций, необходимых для реализации операции j-го типа в случае использования макроса;

m_{ij} – количество обращений к вычислительной операции j-го типа в операторе i-го типа;

n_i – количество операторов i-го типа в функциональной задаче;

L – количество типов операций;

N – количество разновидностей типов операторов.

2) с помощью универсальных интерпретирующих функций, через комбинацию которых можно реализовать все операторы ОСК с учетом нового оператора (рис. 6).

Объем памяти $V_{УФ}$, отводимый под функциональную задачу после ее трансляции при использовании универсальных функций, будет оцениваться следующим образом:

$$V_{y\phi}' = \sum_{i=1}^{L} k_{y\phi i} + \sum_{i=1}^{L} \sum_{j=1}^{N} n_i \cdot m_{ij} + L,$$

где:

$k_{y\Phi i}$ – количество простых инструкций, необходимого для реализации операции *i*-го типа при использовании универсальной функции.

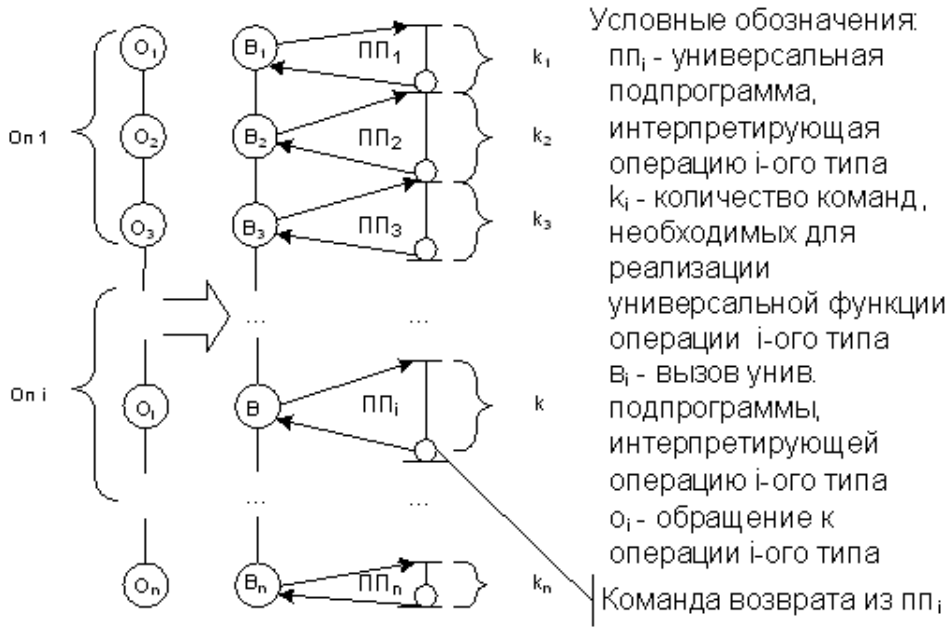

Рис. 6. Организация функций интерпретации сложного оператора в набора универсальных функций

Для определения k_{Mi} и $k_{y\phi i}$ были определены следующие виды простых инструкций, необходимые и достаточные для представления любого вычислительного оператора: бинарная операция, унарная операция, сравнение, возвращение значения и обращение к функции. Исследование способов представления операторов на промежуточном языке было проведено на примере операций типа “сложение”, “вычитание”, “деление”, “умножение” и “блокировка переполнения” с помощью полученных формул в объеме полного функционального обеспечения (рис. 7).

Исследование показало, что если функциональная задача содержит до 5 обращений к указанным операциям, то их организация в виде макроса предпочтительней универсальных функций; в обратном случае использование универсальных функций оказывается

предпочтительней.

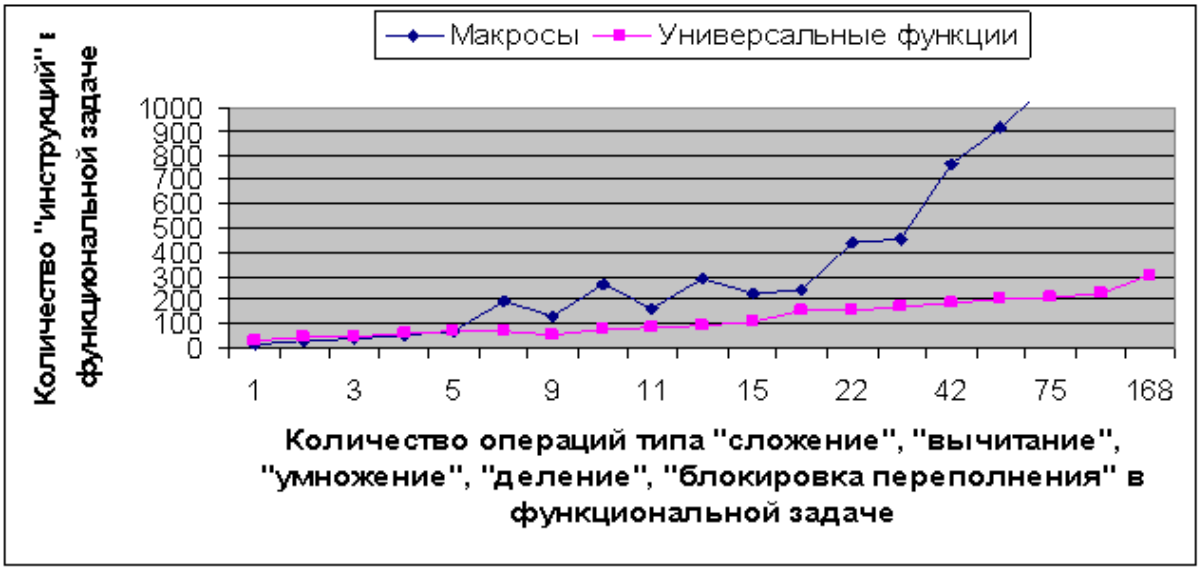

Рис. 7. График зависимости количества “инструкций” от количества вычислительных операций в функциональной задаче

Программная эмуляция дробной арифметики.

Для непосредственной реализации функций интерпретации операций рассмотрим эмуляцию дробной арифметики (или чисел с фиксированной запятой перед старшим разрядом), характерной семейству БЦВМ ОСК. По критериям быстродействия и памяти предлагается использовать следующий подход эмуляции, а именно:

1) при интерпретации логических, тригонометрических операций, операций вычисления квадратного корня и вычислительных операций типа «сложение/вычитание» использовать целые числа, при этом в некоторых случаях необходима корректировка результата операции;
2) при интерпретации вычислительных операций типа «умножение/деление» использовать числа с плавающей точкой, при этом необходимо выполнение преобразование форматов данных.

Организация функций интерпретации операторов ввода-вывода.

Для организации функций интерпретации операторов ввода-вывода необходимо определить

соответствие входных и выходных линий канала ввода-вывода целевой и эмулируемой БЦВМ, и частоту работы канала ввода-вывода.

В случае, когда количество входных и выходных линий канала ввода-вывода целевой БЦВМ превышает количество входных и выходных линий канала ввода-вывода эмулируемой БЦВМ, то производится прямое соответствие линий ввода-вывода, которое используется при интерпретации операторов настройки ввода-вывода и операторов обмена информацией.

В случае ограничения входных и выходных линий канала ввода-вывода целевой БЦВМ для осуществления программной эмуляции системы ввода-вывода БЦВМ ОСК рассмотрены следующие схемы:

1) при модульной структуре организации системы ввода-вывода использовать два или более модулей, соединенных посредством коммутатора (см. рис. 8);
2) передавать две и более порции информации по одной и той же линии за один цикл передачи данных, если это позволяет частота работы канала целевой БЦВМ (см. рис. 9).

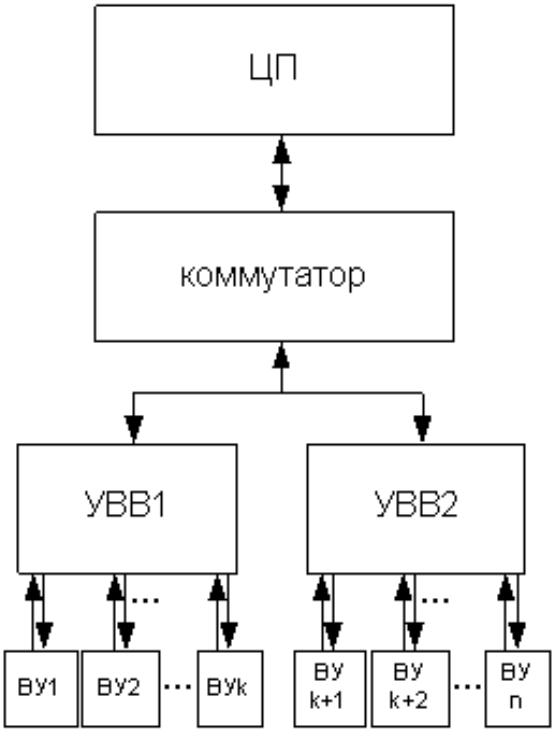

Рис.8. Программная эмуляция системы ввода-вывода БЦВМ ОСК при модульной структуре системы ввода-вывода целевой БЦВМ

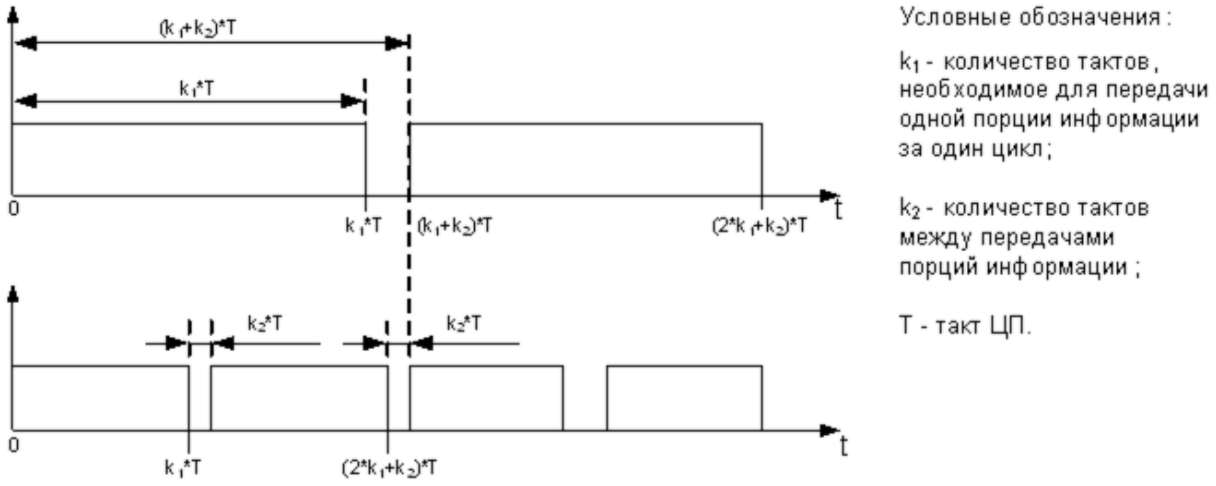

Рис. 9. Программная эмуляция системы ввода-вывода на целевой БЦВМ

При использовании первой схемы необходимо однозначно определить соответствие входных и выходных линий целевой и эмулируемой БЦВМ с учетом номера модуля ввода-вывода и использовать данное соответствие при интерпретации операторов настройки ввода-вывода и операторов обмена информацией.

Использование второй схемы при интерпретации операторов настройки ввода-вывода и операторов обмена информацией предполагает определение соответствия двух линий ввода-вывода эмулируемой БЦВМ и одной линии ввода-вывода целевой БЦВМ. При этом частота передачи информации эмулируемой БЦВМ должна вдвое превышать частоту передачи данных целевой БЦВМ. Установка частоты работы линии осуществляется при интерпретации операторов настройки ввода-вывода.

Организация функций интерпретации операторов системы диспетчеризации.

Для организации функций интерпретации операторов формирования и запуска частотных пакетов, а также для организации выполнения диспетчера задач необходимо определить схему программной эмуляции системы диспетчеризации.

Планирование реального времени БЦВМ ОСК относится к классу статического вытесняющего планирования на основе приоритетов [4], которые назначаются заданиям на основе их периодов.

Для диспетчеризации в семействе БЦВМ ОСК используется общий диспетчер (Д), который запускает частотные пакеты через каждый период Т времени, и диспетчер по готовности (ДГ), который запускается по запросу внешнего устройства. Предлагается организация Д, ДГ и частотных пакетов на основе свойств операционной системы (ОС) целевой БЦВМ, таких как: многозадачность и многопрограммность. Данный подход дает возможность безболезненного совмещения эмулируемого и разрабатываемого ПО на целевой БЦВМ.

Многозадачность дает возможность использования нескольких способов организации системы диспетчеризации. Возможна организация Д, ДГ и частотных пакетов в виде задач ОС, а также Д, ДГ - в виде задачи ОС, а частотных пакетов - в виде подпрограмм. Второй способ организации предполагает *n*-кратное создание задачи диспетчера, в зависимости от разновидности частотных пакетов.

При организации Д, ДГ и частотных пакетов в виде задач ОС целевой БЦВМ время переключения частотных пакетов зависит от времени сохранения и восстановления контекста задач, что является функцией операционной системы. При организации Д, ДГ в виде задачи ОС целевой БЦВМ, а частотных пакетов - в виде подпрограмм, переключение частотных пакетов возложено на диспетчер, поэтому время, отводимое на запуск частотных пакетов в одном цикле, а именно на вызов соответствующих подпрограмм, должно быть несоизмеримо малым по сравнению с временем выполнения частотного пакета.

При реализации системы диспетчеризации с использованием многопрограммности Д, ДГ и частотные пакеты должны быть реализованы в виде самостоятельных приложений, работающих с одной и той же областью памяти. Но при таком подходе время переключения контекста задач больше, чем при реализации системы диспетчеризации с использованием многозадачности. Поэтому целесообразнее использовать первый подход.

Организация программной эмуляции аппарата синхронизации.

Целевая БЦВМ должна быть адаптирована под систему реального времени промышленных образцов, которая требует наличия аппарата синхронизации на входе и на выходе при выполнении

оператора, группы операторов или пакетов функциональных задач. Аппарат синхронизации может быть организован следующими способами: либо синхронизировать каждый оператор, либо синхронизировать группу операторов, т.е. разбить всю программу на функционально законченные части (подпрограммы, состоящие из нескольких операторов), либо синхронизировать процессы с помощью настраиваемого экрана. Под настройкой экрана понимается следующая процедура: сначала необходимо сделать прогон функционального обеспечения БЦВМ ОСК с целью получения образа временной диаграммы выполнения процессов. Зафиксировав моменты готовности и передачи данных между процессами, можно использовать полученный образ при синхронизации процессов на целевой БЦВМ. Режим ожидания может быть реализован задержкой с помощью программируемого таймера, встроенного в целевую БЦВМ. Наиболее эффективной по критерию точности схемой организации аппарата синхронизации является синхронизация по полученному образу временной диаграммы выполнения процессов.

Метод трансляции на промежуточный язык с учетом программной эмуляции системы ввода-вывода, диспетчеризации и аппарата синхронизации.

На основе выше изложенного предлагается метод трансляции на промежуточный язык с учетом программной эмуляции системы ввода-вывода, диспетчеризации и аппарата синхронизации, обеспечивающий трансляцию программного обеспечения, реализованного с использованием открытой системы команд. Данный метод заключается в формировании адресной трассы по функциональным задача, определении схемы программной эмуляции системы ввода-вывода, системы диспетчеризации и аппарата синхронизации, распознавании оператора по его функциональному представлению и представлении оператора на промежуточном языке с использованием набора универсальных интерпретирующих функций.

Методика программной эмуляции семейства БЦВМ ОСК.

Методика программной эмуляции семейства БЦВМ ОСК заключается в следующем:

1) Определение промежуточного индустриального языка.
2) Определение адресной трассы по функциональным задачам.

3) Представление базовой ОСК на языке описания операторов.

4) Определение схемы программной эмуляции системы ввода-вывода и диспетчеризации, а также аппарата синхронизации.

5) Определение набора универсальных функций с учетом особенностей эмуляции дробной арифметики, систем ввода-вывода, диспетчеризации и аппарата синхронизации.

6) Распознавание и перевод оператора по его функциональному представлению на промежуточный язык с использованием набора универсальных функций.

7) Компиляция полученного ПО на промежуточном языке под целевую БЦВМ.

Оценка эффективности программной эмуляции семейства БЦВМ ОСК по критерию времени трансляции ПО.

Для оценки времени трансляции программного обеспечения семейства БЦВМ с учетом изменяемости системы команд была разработана модель определения среднего времени трансляции операторов.

По анализу системы команд бортовой машины можно выделить набор операций, с помощью которых может быть представлен новый оператор. Среднее время обработки транслятором всех операторов с учетом нового оператора $T_{ср}^{нов}$:

$$T_{ср}^{нов} = \frac{m}{N^2}\sum_{i=1}^{N} T_i \cdot m_i + \frac{1}{N+1} \cdot T_{нов},$$

где: $T_{нов}$ – время обработки транслятором нового оператора; m – количество транслируемых операторов; N – количество операторов транслируемой системы команд; T_i – время обработки транслятором i-ого оператора; m_i – количество появлений i-го оператора в функциональной задаче.

Введем следующие характеристики:

$p_{нов}^{i}$ - вероятность того, что новый оператор представлен любыми i операциями эмулируемой системы команд;

$T_{нов}^{i}$ - время обработки нового оператора, представленного любыми i операциями эмулируемой

системы команд; $i=(1 \div n)$.

Тогда:

$$T_{нов} = \sum_{i=1}^{n} p^{i}{}_{нов} \cdot T^{i}{}_{нов}$$

$$T^{k}_{нов} = \frac{(n-1)!}{(k-1)! \cdot (n-k)!} \cdot \frac{1}{n^k} \left(1 - \frac{1}{n}\right)^{n-k} \cdot \left(1 - \frac{1}{n^k}\left(1 - \frac{1}{n}\right)^{n-k}\right)^{\frac{n!}{k! \cdot (n-k)!}-1} \cdot \sum_{i=1}^{n} T_i$$ где

$2 \le k \le n$.

Зная время обработки транслятором *i*-ого операции, и количество операций базовой системы команд и количество операций, через которые можно определить новый оператор, можно оценить среднее время обработки транслятором всех операторов с учетом появления нового оператора.

Выводы.

В статье решена проблема программной эмуляции семейства БЦВМ ОСК, особенностью которой являются:

1) адаптация к появлению нового оператора;
2) решение проблемы программной эмуляции системы ввода-вывода;
3) решение проблемы программной эмуляции системы диспетчеризации;
4) решение проблемы программной эмуляции аппарата синхронизации;
5) решение проблемы программной эмуляции дробной арифметики.

Оценка эффективности работы программного эмулятора выполнялась по критерию занимаемой результирующим кодом памяти и по критерию быстродействия. Оценка эффективности работы программного эмулятора выполнялась по критерию занимаемой результирующим кодом памяти показала, что для представления операторов ОСК на промежуточном языке целесообразней использовать набор универсальных интерпретирующих функций.

Оценка работы программного эмулятора по критерию быстродействия на примере программной эмуляции функциональных задач программного обеспечения БЦВМ ОСК типа БЦВМ Ц100-03 под новую платформу БЦВМ Багет-54 показала, что абсолютная погрешность

вычислений времени обработки программным эмулятором всех операторов составляет 0,0014 мс. Относительная погрешность оценки составляет 0,9%.

Список литературы.

1. Алан Джок. Компиляторы, интерпретаторы и байт-код, "Открытые системы", #06, 2001, //Издательство "Открытые системы", http://www.osp.ru/cw/2001/06/029_0.htm - 15 с.
2. Автоматизация проектирования вычислительных систем. Языки, моделирование и базы данных. / Под ред. М. Брейера. - М.: Мир, 1979, - 464 с.
3. Описание языка операторов БЦВМ Ц-100 ОЯ.08000-01 35 01, М.: 1981 – 43 с.
4. Столингс Вильям. Операционные системы, 4-е издание.: Пер. с англ. – М.: Издательский дом «Вильямс», 2002. – 848 с.

Олег Михайлович Брехов – зав. каф. «Вычислительные машины, системы и сети» Московского авиационного института (государственного технического университета), профессор, д.т.н., телефон: 158-43-82, e-mail: obrekhov@mail.ru.

Анна Викторовна Корнеенкова – м.н.с. каф. «Вычислительные машины, системы и сети» Московского авиационного института (государственного технического университета), телефон: 158-48-99,e-mail: ankorn@yandex.ru.