УДК 629.113.073

О РЕАЛИЗАЦИИ РЕЖИМА РЕАЛЬНОГО ВРЕМЕНИ И СИНХРОНИЗАЦИИ ПРОЦЕССОВ ПРИ МОДЕЛИРОВАНИИ АВТОМОБИЛЯ В SIMULINK

В.Г. Михайлов, к.т.н.,

г. Минск, Республика Беларусь, e-mail sapr7@mail.ru

Рассмотрены существующие методы реализации режима “реального” времени для моделирования автомобиля, выполняемые на специализированных компьютерах, имеющих богатый набор устройств для этого, средств ввода-вывода в виде интерфейсных плат и драйверов к ним включая специальный модуль Simulink Real-Time и их недостатки.

Указаны факторы, определяющие быстродействие системы моделирования.

На основе проведенного исследования предложены новые методы реализации режима реального времени при имитационного моделирования движения, колебаний, автомобиля на основе подбора шага интегрирования, применение утилит Overdrive для разгона/замедления процессора и оптимизации блок-схемы за счет комбинированного использовании мощного компьютера и мини-компьютера Raspberry для считывания параметров органов управления, визуализации дорожной обстановки и модуля S-Function Builder, созданного ПО на C/C++ и объединенного массива макро и микропрофиля реальной дороги, разработана блок-схема в Matlab/Simulink и ПО.

Ключевые слова: Автомобиль, математическая модель, имитационное моделирование, блок-схема, движение, нагруженность, управляемость, продольный и микропрофиль дороги, Matlab/Simulink, модули S-Function Builder, Simulink Real-Time.

Введение

Имитационное моделирование представляют собой сочетание методов компьютерного моделирования с участием водителя с имитированием дорожной обстановки и воздействием на него на стенде [1], [2]. Реализация их требует использования режима реального времени и синхронизации процессов в подсистемах. Обычно оно выполняется на специализированных компьютеров (рисунок 1), имеющих богатый набор устройств для этого, средств ввода-вывода в виде интерфейсных плат и драйверов к ним [3]. А также применяется распараллеливание процессов вычислений. В них используются те же процессоры Intel и AMD (2-х, 4-х ядерные, CPU 3,0–3,5 ГГц), что и в персональных компьютерах. Основное отличие в периферийном оборудовании, наличие специальных слотов для АЦП, ЦАП, PWD и в обеспечении возможности работы в тяжелых промышленных условиях.

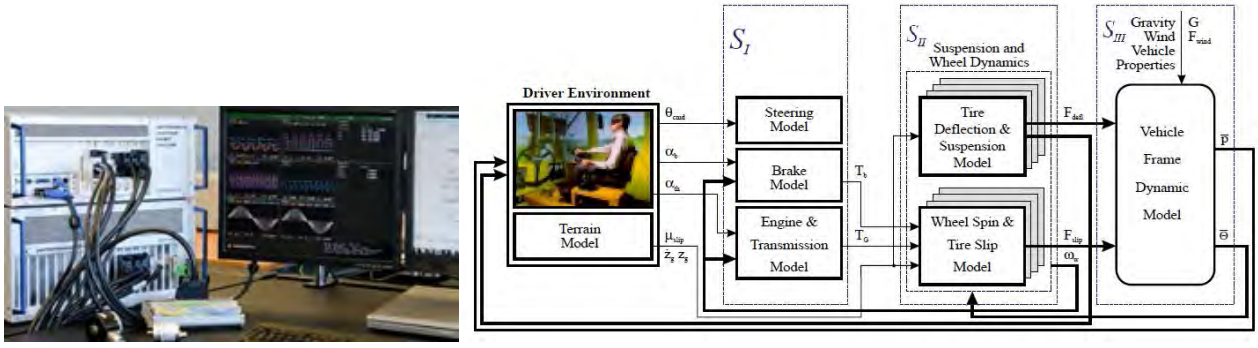

Рисунок 1 – Специализированные компьютеры для имитационных испытаний автомобилей и распараллеливание вычислений между компьютерами

В пакете Simulink имеется специальный модуль Simulink Real-Time, включающий и другие компоненты, базирующийся на ядре Real-Time Kernel, который взаимодействует через интерфейс с основной операционной системой [3]. Этот модуль представляет собой специализированный драйвер (операционную систему), который управляет частотой

центрального процессора (ЦП), прерываниями по таймеру от тактовой частоты ЦП, поддерживает синхросигналы для операционной системы и взаимодействует с устройствами ввода/вывода (I/O blocks).

За счет настройки его окружения, подбора параметров и драйверов можно обеспечить совпадения времени моделирования и реального процесса. Однако Simulink Real-Time не может увеличить быстродействие системы, он может только ее замедлить. Процесс настройки окружения довольно сложный, об чем свидетельствует большой объем руководства Simulink® Desktop Real-Time™ User's Guide 136 стр. [3] и необходимость привлечения специализированных фирм для внедрения таких компьютеров и его программного обеспечения (ПО).

Следует также отметить, что необходимые блоки и ПО приобретаются отдельно. Для моделирования в реальном масштабе времени используется фиксированный шаг.

Имеются ограничения в нем по использованию функций низкоуровневого программирования в языке C/C++. Так, например, нельзя использовать функции fread, fwrite при записи бинарных файлов при обмене информации через сеть, которые могли бы обеспечить более высокое быстродействие по сравнению с предлагаемым использованием баз данных или MS-Excel. Кроме того, в ряде случаев из-за блокированного множителя процессора (CPU) невозможно варьировать частотой процессора, что делает невозможным использование Simulink Real-Time.

Целью данной работы является рассмотрение более простых путей реализации режима реального времени для моделирования автомобиля на персональных компьютерах.

2. Факторы, влияющие на реализацию режима реального времени

Автомобиль представляет собой сложную динамическую систему, содержащую большое количество подсистем и требует распараллеливания вычислений между компьютерами, как это делается за рубежом (рисунок 1).

Основными являются подсистемы движения, колебаний, управляемости, которые должны решатся синхронно, что представляет определенную сложность при моделировании.

Быстродействие системы моделирования зависит от следующих факторов:

– производительности компьютера (его CPU, памяти, шины),

– реализуемых подсистем автомобиля, их связей, взаимодействия;

– использованного метода формирования возмущения дороги, логики управления КПП, какими методами это достигается;

– количества задействованных в системе интеграторов, нелинейных элементов;

– выбранной схемы реализации моделирования и обмена информацией;

– рациональной топологии блок-схемы моделирования;

– подсистем управления моделированием и визуализации обстановки.

Схемы использованных моделей подсистем движения, колебаний, управляемости приведены на рисунке 2.

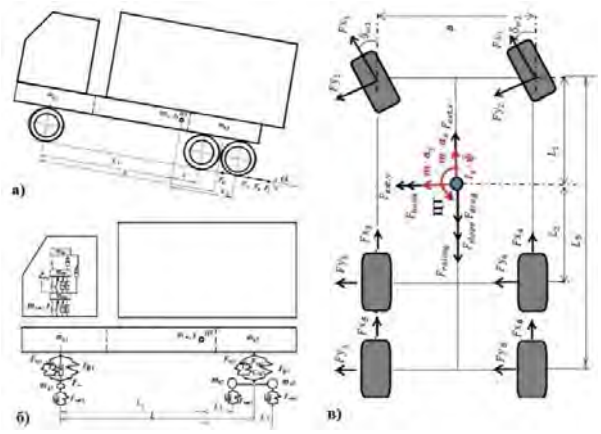

Рисунок 2 – Модели движения (а), колебаний (б) и управляемости (в) автомобиля

Их математические модели и реализация в Simulink рассмотрены в работах [4, 5, 6]. В общей сложности в них задействовано 16 интеграторов, включая моделирование трения и характеристики шин. Реализация возмущения дороги (макро и микропрофиль) выполнена с помощью программы на C/C++ в S-Function Builder [7].

3. Способы реализации режима реального времени

Для обеспечения моделирования автомобиля в реальном масштабе времени могут использоваться следующие способы:

– варьирование шагом интегрирования;

– использование утилит Overdrive для разгона/замедление процессора;

– оптимизация блок-схемы и топологии моделирования.

3.1 Варьирование шагом интегрирования базируется на подборе шага интегрирования для совпадения времени моделирования и реального процесса.

Выбор шага численного интегрирования обуславливается высшей частотой моделирования, для которой необходимо иметь порядка 10 точек на высшей частоте. Определяющими для автомобиля являются его колебания. В режиме реального времени при моделировании можно обеспечить полосу 0–22,4 Гц при шаге интегрирования t = 0,005 с и менее. При шаге t = 0,001 с теоретически возможно достижение 100 Гц. Однако смоделировать процессы более 22,4 Гц не реально из-за значительного снижения амплитуды вибраций, уменьшающейся в квадратичной зависимости с увеличением частоты.

На рисунке 3 показано влияние шага интегрирования на спектр вибраций в третьоктавных полосах частот над передней осью автомобиля. Из него видно, что при шаге 0,001– 0,005 с уровень вибраций до частоты 20 Гц практически не меняется. Лишь только при шаге t=0.008 с уровень резко снижается после 12,5 Гц. Данный график свидетельствует, что для колебаний автомобиля может использоваться шаг 0,001–0,005 с и с помощью его (путем подбора) можно обеспечить соответствие времени моделирования и реального процесса.

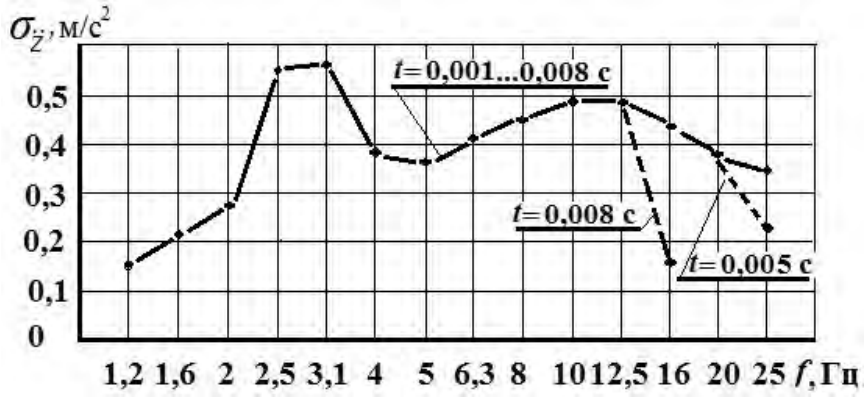

Рисунок 3 – Влияние шага интегрирования на спектр вибраций в третьоктавных полосах частот над передней осью автомобиля

Требуемое значение шага, обеспечивающее моделирование в реальном масштабе времени, может быть определено из графика зависимости времени моделирования от шага интегрирования. Для чего требуется провести исследование и построить график зависимости времени моделирования от шага интегрирования при моделировании совместно движения, колебаний и управляемости автомобиля. Для использованного мной компьютера (2-х ядерного CPU 2,5 ГГц) получена следующая зависимость (рисунок 4).

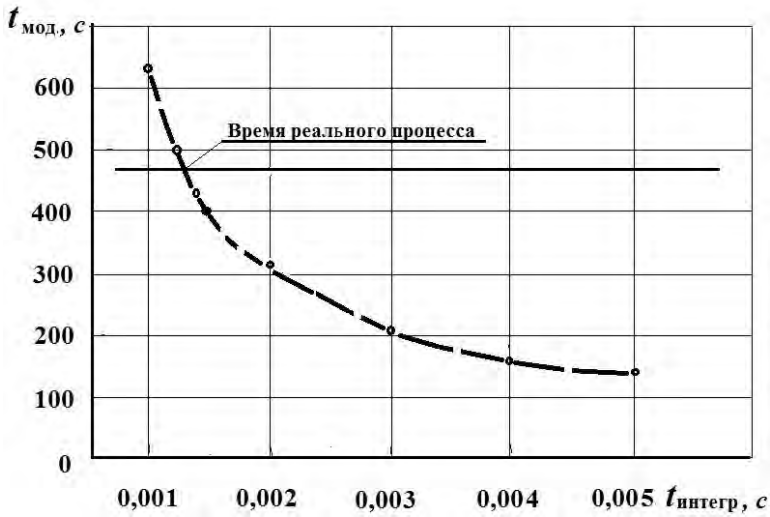

Рисунок 4 – Зависимость шага интегрирования от времени процесса

В данном случае реальному процессу длительностью 470 с соответствует шаг t=0,0013 с. Недостатком этого метода является возможность остановки/зависания процесса моделирования на определенном шаге интегрирования из-за нелинейных элементов блок-схемы и требуется уменьшить шаг интегрирования. И такое же возможно в Simulink Real-Time, где для исключения этого предусматривается проверка на разной производительности.

3.2 Использование утилит Overdrive для разгона/замедления процессора. В этом случае целесообразно вначале проверить наличие незаблокированного множителя используемого процессора. И только потом подобрать утилиту для разгона/замедления процессора. Для процессоров Intel может использоваться утилита PowerTweak, EasyTuneEasyTune, для AMD – OverDrive 4.3.2.

В моем случае для устаревшего 2-х ядерного процессора AMD Athlon 4800+, 2,5 ГГц использовалась утилита CPU-Z, позволившая разогнать процессор только до 2,7 ГГц (580 с), что оказалось недостаточным для обеспечения реального времени (470 с). Пришлось вернутся на частоту 2,5 ГГц и увеличивать шаг интегрирования до 0,013 с.

Для реализации режима реального времени при шаге 0,001 с необходим процессор с тактовой частотой 3,35 ГГц и выше (частота процессоров обычно составляет 3,0; 3,2; 3,3; 3,4; 3,5 ГГц). Некоторую избыточность при 3,5 ГГц и выше можно компенсировать настройкой утилиты на замедление. Использование процессоров с числом ядер более 4-х ничего не дает, т.к. основная загрузка приходится на первые три ядра: соответственно 60–65 %, 20–25 % и 10–15%.

По информации в интернете реально можно разогнать процессор не более чем на 10–15 %. И на эту величину можно ожидать повышение скорости моделирования. Процессор с частотой 4 ГГц может быть разогнан до 4,7 ГГц и то при использовании водяного охлаждения и резкого снижения его ресурса. Из-за чего этот вариант не желателен.

В целом данный способ больше целесообразен для замедления моделирования.

3.3 Оптимизация блок-схемы. На скорость моделирования влияет оптимизация блок-схемы. На основе экспериментов моделирования движения, колебаний и управляемости автомобиля для реализации на симуляторе были оптимизированы следующие схемы, представленные на рисунке 5.

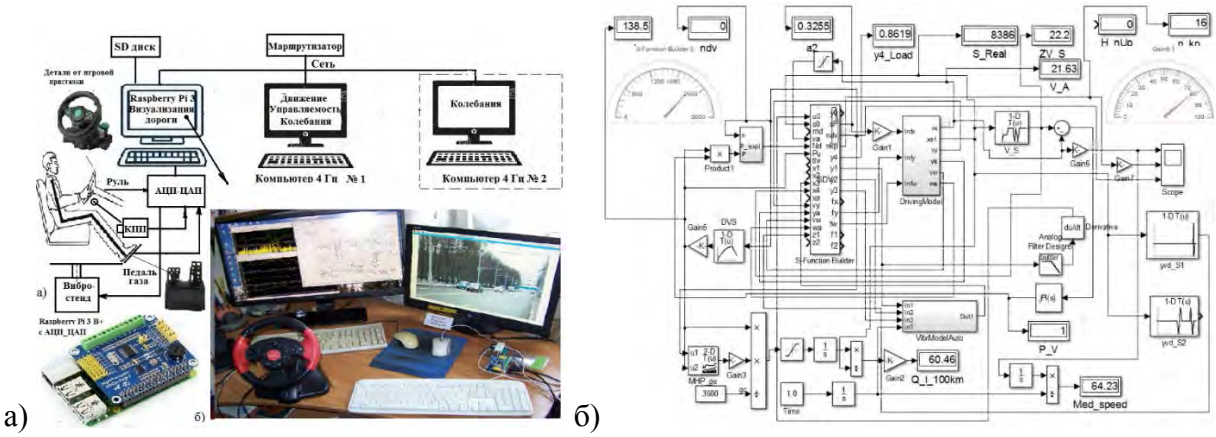

Рисунок 5 – Общая схема (а) и блок-схема (б) имитационного моделирования автомобиля

3.3.1 Для обеспечения реального времени применены следующие решения:

– разделение вычислительных ресурсов между основным PC и мини-компьютером Raspberry Pi 3.

– организация обмена информацией по сети с помощью файлового сервера Samba через файлы обмена (Рисунок 6) на общем SD диске Raspberry с определенной периодичностью на основе сравнения счетчиков циклов.

– использование S-Function Builder для формирования возмущения дороги (макро и микропрофиля) и логики переключения передач КПП с помощью разработанных программ на C/C++. Последняя также использовалась для обмена информации;

– применение откомпилированных программ на C/C++ в S-Function Builder в PC и Raspberry Pi 3;

– осуществление программной синхронной визуализации дорожной обстановки с помощью Raspberry Pi 3, управляемой из Simulink

В работе использовался Simulink версии R2015b, чье быстродействие оказалось в 2 раза выше чем у R2018b из-за перенасыщения последнего новыми функциями.

3.3.2 Для повышения быстродействия в S-Function Builder применен технический прием использования счетчика циклов задержки ***m*** при переключении передач КПП, не останавливающий процесс моделирования. Ранее применялся цикл for () для задержки переключения передач, который оказался неэффективным (замедлял процесс в 2 раза). Переход на новую передачу осуществляется с задержкой t=2 с через нейтральную передачу, при которой $F_к$= с помощью счетчика режима *chm*=0. При отсутствии переключения *chm*=1.

При переключении передач учитывалось ускорение автомобиля ***a2*** и обороты двигателя ***ndvs***, определяемые исходя из скорости автомобиля. Ниже приведен фрагмент программы в S-Function Builder.

```
static void mdlOutputs(SimStruct *S, int_T tid)
{
-----------------------------------
//Переключение КПП вниз
if (a2<-0.2&&ndvs<1100&&chm!=1)
{
n_kp0=n_kp; // номер передачи КПП
n_kp--;
if (n_kp<1)
n_kp=1;
else
ndvs=103.26*Va*u_kp[n_kp0];
//ndvs=83.51*Va*u_kp[n_kp0];
```

```
  if (ndvs>2200)
     ndvs=2200;

*nkp=n_kp;
*ndv=ndvs;
//n_kp0=n_kp;
chm=1;
m=0;

}
*nkp=n_kp;
*ndv=ndvs;
n_kp0=n_kp;
//----------------Переход на нейтральную передачу
if (chm==1) // При переключении передачи
    {
 m++;
 Fv=3.997*Va*Va;   //Fv=Kv*Aa*Va*Va; Сопротивление воздуха
 Fy=811.715+3639*yn;  // 9.81*ma*(0.0039+0.0175 *yn); Fw=9.81*ma*(f*cosa+sina);
// Сопротивление качения и наклона
 jm=1.02599; //1+Jk/(ma*rk)^2+(Jen*kpd (u_kp*u_rk*u_gp)^2)/(ma*rk)^2 учет инерцион
масс
 a2=(-Fv-Fy)/(ma*jm); // ускорение автомобиля
 *sF=a2;
// *nkp=n_kp;
  ndvs=103.26*Va*u_kp[n_kp0];
  if (ndvs>2200)
     ndvs=2200;

   *ndv=ndvs;
   }
else
a2=(Fk-Fv-Fy)/(ma*jm); // ускорение автомобиля
// Конец перехода на нейтральную передачу

if (m==1000)
  {
chm=0;
m=0;
}
}
```

3.3.3 Дополнительным способом повышения скорости моделирования может использоваться отключение/удаление менее значимых несвязанных подсистем и их компонентов и решение их отдельно. Например, отказ от одновременного расчета ускорений в треть октавных полосах частот и запись колебаний в файл, использовав функцию To File. А потом в отдельной программе произвести расчет спектра по этому файлу.

3.4 Разработанная методика заключалось в одновременном проведении имитационного моделирования на двух компьютерах: с помощью основного PC и мини-компьютера Raspberry Pi 3 B+ с платой АЦП_ЦАП ADS1256/ADS1115 и 2-х ядерного основного компьютера с разделением задач. Миникомпьютер Raspberry Pi 3 использовался

для визуализации дорожной обстановки (воспроизведения видеозаписи дороги) пропорциональной скорости движения и получения управляющих сигналов от педали газа, КПП и рулевого колеса. Визуализация дорожной обстановки в Raspberry Pi 3 с переменной частотой просмотра осуществлялась с помощью разработанной программы на языке C/C++ (компилятор GCC 8.2) и библиотеки OpenCv версии 4.0.1. Использованный подход является новым. Он позволил разгрузить основной компьютер от процесса визуализации и управления. Кроме того Raspberry Pi 3 лучше подходит для работы с АЦП, программа которой довольно сложная и составляет 1250 строк и в случае с PC это сильно бы его нагрузило.

Важную роль в примененной блок-схеме играет использование модуля S-Function Builder, который задействован в цикле всех процессов. В нем программно осуществляется формирование возмущение дороги на основе массива точек, интерполяция их промежуточных значений, реализация задержки возмущения на оси, расчет характеристик и всех параметров и выдача сигналов в подсистемы, обмен информацией с Raspberry Pi 3, расчет динамики автомобиля с учетом передачи КПП, педали газа, тормоза, рулевого колеса. Все это решается на основе откомпилированной программы, благодаря чему обеспечивается высокое быстродействие системы. Применено занесение в оперативную память большого массива данных (96000 точек) по дороге (12 км) и работа с ней, что значительно (в 6–7 раз) повысило быстродействие системы по сравнению с вариантом чтения информации из файла. Из общего массива с помощью фильтра выделяется сигнал продольного профиля дороги в полосе 0–0,6 Гц, который подвергается интегрированию для получения угла наклона дороги. Эта же программа используется для учета многопараметрических зависимостей двигателя, передаточных чисел трансмиссии, нелинейных характеристик двигателя, логики переключения передач и обмена информацией. Использованные решения, включая такую топологию блок-схемы позволили значительно (в 2 раза по сравнению с первоначальным вариантом) повысить быстродействие.

3.5 Обмен информации между компьютерами осуществлялся по сети с помощью файлового сервера Samba. В качестве маршрутизатора использовался телефонный HUAWEI HG8245H (1 Гбит/с). Для обмена информации через сеть на Raspberry создана общая папка **share** (folder share**),** доступная, как компьютеру PC, так и Raspberry. В эту папку в виде двух бинарных файлов 8 и 16 байт записывалась и считывалась информация с Simulink (S-Function Builder) и Raspberry с помощью программ на C/C++ с периодичностью 0,2 с (5 Гц). Использовались функции C/C++ fread, fwrite, обеспечивающие минимальное время обмена 0,2 мс, что не тормозит процесс моделирования в Simulink. Сигналы с потенциометрических датчиков руля и педалей газа, тормоза от игровой приставки подавались через АЦП (ADS1256/ ADS 1115) на Raspberry Pi 3 и далее через сеть в основной PC и Simulink.

3.6 Синхронизация процессов моделирования. Синхронизация процессов моделирования обеспечивается применением модуля S-Function Builder и его программы на C/C++ путем привязка процессов моделирования в этих трех подсистемах к текущему значению пути автомобиля и использованием единого массива макро и микропрофиля дороги. Использовались параметры скорости и пройденного пути автомобиля, определяемые из уравнений тяговой динамики автомобиля [6], которые определяют текущую значение по длине пути и через нее высоту макро и микропрофиля дороги. Текущее значение последней используется при расчете колебаний автомобиля, а ее производная для определения угла наклона дороги. Такой же подход применяется для моделирования управляемости, где дополнительно еще задействуется информация положения рулевого колеса, передаваемая через АЦП в Raspberry Pi 3 и далее через файл обмена в Simulink.

Синхронизация визуализации дорожной обстановки также осуществлялась привязкой к значению пройденного пути на текущий момент. В программе на Raspberry использовалась функции cap.read(frame) из библиотеки OpenCV для прокрутки кадров вперед для ускорения просмотра. Задержки изображения осуществлялась путем установки параметра stp=0, останавливающего процесс. Синхронизация изображения и пройденного пути осуществляется

на основе передаваемой информации в файле pda.bt и сравнения их счетчиков кадров. Число кадров, соответствующих пройденному пути *chr*=K*30, где *K*=30**t*/*l*=1,6346, ***t*** и ***l*** – соответственно пересчетный коэффициент, время прохождения/моделирования участка в с, длиной ***l***, м, 30 – частота записи video. Данный подход синхронизации применен впервые.

Заключение

Выполненное исследование показало, что:

1. Применение утилит Overdrive для разгона/замедление процессора больше подходит, когда необходимо незначительно (на 10–15%) увеличить скорость моделирования. Но в целом утилиты лучше применять для замедления процессов.

2. Варьирование шагом интегрирования позволяет в более широком диапазоне изменять скорость моделирования, но при этом сужается полоса воспроизводимы х частот. Недостатком этого метода является возможность остановки/зависания процесса моделирования на определенном шаге интегрирования из-за нелинейных элементов блок-схемы

3. Оптимизация блок-схемы позволяет в комплексе увеличить скорость моделирования. Но ее лучше применять в комплексе с первыми двумя способами.

4. Обеспечение синхронизации процессов моделирования более целесообразно осуществлять путем использованием модуля S-Function Builder и разработанной в нем программы с привязкой процессов к текущему значению пути автомобиля.

5. Все это в совокупности позволило проводить моделирование в реальном масштабе времени синхронно в подсистемах и показало эффективность данного метода.

Литература

1. Mercedes-Benz Innovation Vehicle Developing https://www.mercedes-benz.com/en/mercedes-benz/next/advanced-engineering / [Электронный ресурс /Electronic resource]./– Режим доступа/Access mode: 22.07.2018.

2. Emanuele Obialero, A Refined Vehicle Dynamics Model for Driving Simulator // Charhalmers University of Technology/Göteborg, Sweden 2013. Master's thesis, P.120.

3 Simulink® Desktop Real-Time™ User's Guide https://fenix.tecnico.ulisboa.pt/downloadFile/845043405443236/rtwin_target_ug_r2015a.pdf . // [Electronic resource]. Access mode: 12.03.2019.

4. Михайлов, В.Г. Получение и использование единого массива продольного профиля и микропрофиля дороги для моделирования ТС//журнал автомобильных инженеров № 2, 2018, с.4–7.

5. Михайлов В. Г. О колебательной модели грузового автомобиля / Д.В. Мишута, //Автомобильная промышленность –2016, №7.

6. Михайлов, В.Г., Совместное моделирование движения и нагруженности автомобиля // Автомобильная промышленность –2019, № 7, с.

7. Михайлов, В.Г., Использование S-Function Builder в Matlab/Simulink // Системный анализ и прикладная информатика – 2018, № 4. С.57–64.

Reference

4. Mikhailov_ V.G. Poluchenie i ispolzovanie edinogo massiva prodolnogo profilya i mikroprofilya dorogi dlya modelirovaniya TS//jurnal avtomobilnih injenerov № 2_ 2018_ s.4–7.

5. Mikhailov V. G. O kolebatelnoi modeli gruzovogo avtomobilya / D.V. Mishuta_ //Avtomobilnaya promishlennost –2016_ №7.

6. Mikhailov_ V.G._ Sovmestnoe modelirovanie dvijeniya i nagrujennosti avtomobilya // //Avtomobilnaya promishlennost –2019_ № 6_ s.

7. Mikhailov_ V.G._ Ispolzovanie S-Function Builder v Matlab/Simulink / Sistemnii analiz i prikladnaya informatika – 2018_ № 4. S.57–64.

Дата поступления
в редакцию 30.10.2019

ABOUT IMPLEMENTATION OF A REAL-TIME MODE AND SYNCHRONIZATION WHEN MODELLING THE VEHICLE IN SIMULINK

Mikhailov V.G., PhD (Eng)

Minsk, Republic Belarus, e-mail sapr7@mail.ru

The summary. The existing implementation methods of the mode of "real" time for modeling of the vehicle which are carried out on the special purpose computers having a rich set of devices for this purpose, means of input-output in the form of the interface payments and drivers to them including the special Simulink Real-Time module and their shortcomings are considered.

The factors defining high-speed performance of system of modeling are specified.

On the basis of the conducted research new implementation methods of a real-time mode at simulation modeling of the movement, fluctuations, the car on the basis of selection of a step of integration are offered, application of utilities of Overdrive for acceleration/deceleration of the processor and optimization of the flowchart at the expense of the powerful computer combined use and the Raspberry minicomputer for reading of parameters of governing bodies, management of visualization and the S-Function Builder module created by software on C/C ++ and the integrated array macro and a microprofile of the real road, the flowchart in Matlab/Simulink and software is developed.

Keywords: Vehicle, mathematical model, simulation modeling, flowchart, movement, loading, controllability, longitudinal and microprofile of the road, Matlab/Simulink, modules S-Function Builder, Simulink Real-Time.