УДК 681.3.062

Использование среды MATLAB-Simulink для реализации вычислительных алгоритмов в целочисленных микропроцессорных системах

Фадин Д.А.

Конструкторское бюро приборостроения им. академика А.Г. Шипунова, ул. Щегловская засека, 59, Тула, 300001, Россия

e-mail: fadmiral@rambler.ru

Аннотация

В статье рассматривается методика создания программного кода для бортового сигнального микропроцессора управляемой ракеты средней дальности. Методика предусматривает последовательное преобразование алгоритма управления в «физическую», «дискретную» и «целочисленную» модели MATLAB-Simulink. Создание моделей производится с учетом особенностей вычислительной среды микропроцессора. Модели позволяют оценивать скорость выполнения микропроцессором формул методов численного интегрирования. Приводятся примеры создания целочисленных динамических звеньев с применением формата «фиксированная точка». Рассматривается реализация математических функций с помощью специализированных алгоритмов, оперирующих аргументами, заданными в двоичной системе счисления. Предлагается ряд способов оптимизации «целочисленной» модели алгоритма управления по количеству и последовательности математических операций и

форматам представления данных. Оцениваются конечные ошибки решения «целочисленной» модели Simulink.

Ключевые слова: алгоритм управления, процессор цифровой обработки сигналов, среда MATLAB, моделирование в Simulink, дискретизация по времени, численное решение дифференциальных уравнений с временной дискретизацией, формат «фиксированная точка», нормирование данных, CORDIC-алгоритм.

Введение

В данной работе приведена методика создания программного кода (ПК) для целочисленного сигнального микропроцессора (МП), входящего в бортовую микропроцессорную систему (МПС), которая обеспечивает автономное наведение малогабаритной вращающейся управляемой ракеты (УР) средней дальности. Определение координат УР в пространстве осуществляется блоком спутниковой навигации (БСН) или на основе данных блока инерциальных чувствительных элементов (БИЧЭ) и гироскопа крена (ГК) по заданному алгоритму построения инерциальной навигационной системы (ИНС). Алгоритм построения ИНС (АИНС) реализует алгоритмы преобразования систем координат (АПСК) и интегрирования и предусматривает коррекцию расчетных координат по данным БСН.

Координаты УР в стартовой системе координат преобразуются МПС в команды управления рулевым приводом (РП) согласно заданному алгоритму формирования команд управления (АФКУ). АФКУ содержит АПСК, преобразующий координаты от БСН или ИНС из стартовой системы координат в измерительную на основании рассчитанных алгоритмом целеуказания (АЦУ)

параметров цели. Также АФКУ «замыкает» вычислительный контур посредством сигналов датчиков РП.

БИЧЭ, ГК, БСН и РП УР являются заимствованными блоками с заданными параметрами. Частота выдачи линейных ускорений и угловых скоростей БИЧЭ и ГК составляет 500 Гц, а частота выдачи навигационных определений БСН – 10 Гц. Автоколебательный РП имеет заданную частоту функционирования 10 кГц.

Функциональная схема заданного алгоритма управления (АУ), реализующего АИНС и АФКУ, приведена на рисунке 1.

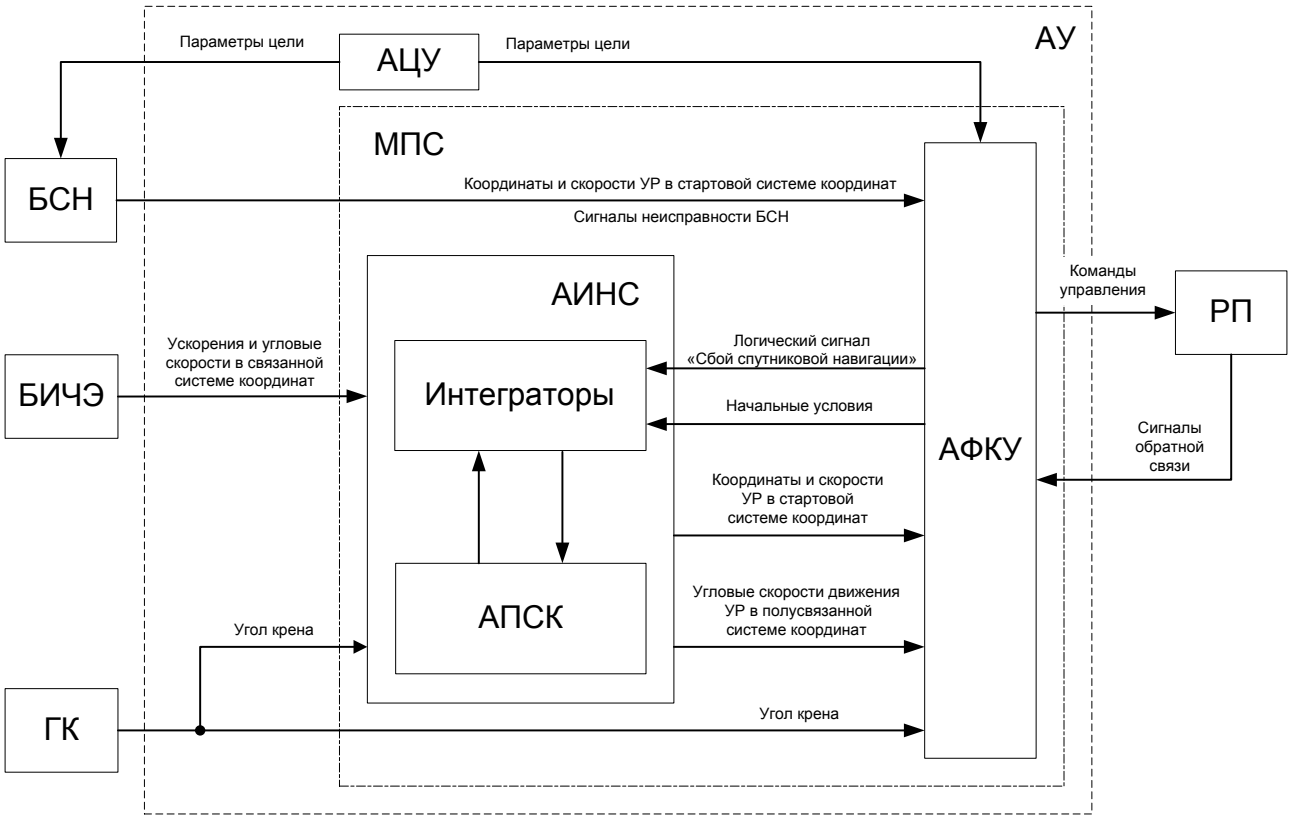

Рисунок 1 – Функциональная схема алгоритма управления

Синтез АУ был осуществлен в физических величинах с помощью численных методов четвертого порядка на основании моделей блоков и аэродинамической модели УР. АУ обладает следующими особенностями:

– описанием в «непрерывной» форме с высокой точностью вычислений в физических величинах в десятичной системе счисления;

– многообразием расчетных траекторий и широким динамическим диапазоном входных данных;

– заданными частотами обмена МПС с датчиками и РП;

– регулярным преобразованием трехмерных прямоугольных систем координат и, следовательно, использованием тригонометрических функций и матриц;

– ограничением времени вычисления управляющих команд (не более 100 мкс).

Для расчета команд управления МПС использует 16-битный цифровой МП обработки сигналов TMS320C55xx, оптимизированный для выполнения операций с целочисленными типами данных с максимальной длиной не более 40 бит. Он заимствован из телеуправляемой ракеты ближней зоны и имеет существенные ограничения вычислительных ресурсов. В ходе тестирования скорости выполнения математических операций и функций с различными форматами данных в МП с тактовой частотой 192 МГц был выявлен ряд особенностей:

– высокое относительное время выполнения операций с плавающей точкой (до 25 раз медленнее операций с целыми числами);

– скорость выполнения операций с данными длиной 40 бит в несколько раз ниже аналогичного показателя 32-битного формата.

На основании проведенных тестов была произведена оценка времени выполнения в МП математических структур и операций АУ, показанная в таблице 1.

Таблица 1 – Оценка времени выполнения математических структур АУ при реализации в МП

Математические структуры и операции	Динамические звенья	Функции *arctg*, *asin* и $\sqrt{x}$	Функции *sin* и *cos*	Умножение / деление	Сложение / вычитание
Количество	12	7	70	70 / 10	≈ 100
Общее время выполнения в МП, мкс					
Тип данных:					
- int32	≈ 3 – 15*	**	**	≈ 11,2 / ≈ 6,7	≈ 3,4
- int40	≈ 12 – 60*	**	**	≈ 36 / ≈ 40	≈ 11,4
- float	≈ 30 – 150*	до 350	до 210	≈ 56,6 / ≈ 20	≈ 79
** – в зависимости от метода численного решения* *** – стандартными функциями в целых числах не реализуется*					

Как видно из таблицы 1, использование вещественного формата оказывается нецелесообразным из-за низкой скорости его выполнения. Следовательно, необходима трансформация исходного АУ в его целочисленный аналог, адаптированный под особенности вычислительной среды МП.

Создание ПК АУ в среде отладки МП затруднено в связи с ограниченными производителем МПС функциями отладочного интерфейса (система «вход-выход»). Отсутствие доступа к внутренней оперативной памяти МП не позволяет

осуществить создание ПК в заданные сроки методом прямого программирования, поэтому отработка АУ осуществляется в инженерной среде MATLAB-Simulink.

Создание целочисленной модели АУ делится на следующие этапы:

1. Создание Simulink-модели АУ в p-области преобразования Лапласа с вещественным форматом представления данных с первичной дискретизацией передаваемых данных по времени («физическая» модель).

2. Определение методов численного решения динамических звеньев (ДЗ) «физической» модели в z-области («дискретная» модель).

3. Переход в «дискретной» модели от вещественного формата представления данных к формату «фиксированная точка» («целочисленная» модель).

4. Тестирование и корректировка «целочисленной» модели алгоритмов управления для обеспечения нужной точности вычислений.

Создание «физической» модели

«Физическая» модель алгоритма управления в системе Simulink включает математические структуры формирования команд управления и структуры преобразования данных систем навигации. При этом модель разбивается на математические модули исходя из принадлежности к составным частям УР, частот дискретизации, динамики звеньев и времени их работы. Подобное разбиение в дальнейшем значительно облегчает тестирование модели и позволяет производить независимую проверку отдельных блоков и функций.

Дискретизация тестовых данных по времени осуществляется с помощью встроенных Simulink-блоков «*Rate Transition*», а задание нужной частоты решения

блоков – с помощью опции «*Treat as atomic unit*». Значительная часть математических модулей модели реализуется в виде классов внешних функций MATLAB (m-функций), что в дальнейшем упрощает их интерпретацию при создании ПК.

ДЗ АУ в операторной форме задаются стандартными динамическими блоками Simulink. Решение «физической» модели выполняется в p-области с помощью встроенного в Simulink численного метода четвертого порядка «*ode4* (*Runge-Kutta*)». Полученные в ходе решения «физической» модели результаты используются для следующих этапов в качестве эталонных.

Создание «дискретной» модели

При создании «дискретной» модели осуществляется выбор метода численного решения ДЗ «физической» модели в z-области. Задачей данного этапа является нахождение баланса между точностью численного решения и количеством элементарных математических операций при их реализации в МП.

Численное решение дифференциального уравнения (ДУ) первого порядка:

$$y' = f(t, y) \tag{1}$$

одношаговыми методами согласно [1] описывается следующей формулой:

$$y_{k+1} = y_k + \sum_{i=1}^{n} \omega_i \cdot f(t_k + \Delta t_i, y_k + \Delta y_i) \cdot \Delta t\,, \tag{2}$$

где y_k и y_{k+1} – значения интеграла функции правой части (1) на текущем и следующем шаге решения;

ω_i – весовые коэффициенты численного метода;

Δt_i, Δy_i – приращения аргументов функции;

Δt и n – шаг решения и порядок численного метода соответственно.

Обычно также вводят обозначения:

$$k_i = f(t_k + \Delta t_i, y_k + \Delta y_i) \cdot \Delta t,\ i = 1 \ldots n. \tag{3}$$

Из формулы (3) видно, что метод порядка n требует выполнения n итераций для вычисления коэффициентов k_i. Методы низкого порядка имеют низкую точность численного решения, но более простую реализацию. Приращения Δt_i и Δy_i зависят не только от шага, но и от метода численного решения. Модификации методов второго и более высоких порядков могут оперировать приращениями Δt_i, кратными половине или одной третьей шага решения. Использование в (1) решетчатых функций с дискретизацией Δt обуславливает необходимость дополнительного определения коэффициентов k_i в правилах (2).

Применение для интегрирования решетчатой функции с шагом Δt метода Эйлера (первого порядка) и метода Эйлера-Коши (второго порядка) [2] с приращениями Δt_i, равными шагу численного решения, эквивалентно записи оператора дифференцирования p в виде:

$$p = \frac{1}{T} \cdot \left(1 - z^{-1}\right), \tag{4}$$

$$p = \frac{2}{T} \cdot \frac{1 - z^{-1}}{1 + z^{-1}}, \tag{5}$$

где $T = \Delta t$ – приращение времени (шаг решения);

z^{-1} – оператор задержки на такт.

Формулу метода Рунге-Кутта третьего порядка с приращениями Δt_i, равными половине шага решения [1], можно формально применить для расчета численного решения решетчатой функции с шагом Δt, если принять $T = 2 \cdot \Delta t$. Тогда правило операции дифференцирования (оператора *p*) описывается формулой:

$$p = \frac{3}{T} \cdot \frac{1 - 4z^{-1} - z^{-2}}{1 + z^{-1}}, \qquad (6)$$

что для операции интегрирования соответствует известной квадратурной формуле правила Симпсона [1]. Формулы методов более высоких порядков будут содержать в знаменателе значения интеграла из предыдущих тактов решения, например, для четвертого порядка – z^{-2} и так далее. С увеличением порядка метода численного решения ошибка интегрирования снижается, но при этом возрастает количество выполняемых математических операций.

Расчет всех ДЗ АУ по формуле (6) затруднен в связи с увеличением времени выполнения алгоритма в МП. При необходимости эта формула может использоваться для увеличения точности ДЗ. Формула (5) имеет более удобную запись в целых числах, поэтому именно она была использована при создании «дискретной» модели.

После решения «дискретной» модели можно оценить ошибки, вносимые выбранным методом численного решения. Для заданных классов входных сигналов и величины шага решения ошибка оказывается допустимой. При необходимости ошибка может быть уменьшена путем коррекции.

Создание «целочисленной» модели

При создании «целочисленной» модели осуществляется переход от вещественного представления чисел к целочисленному. «Целочисленная» модель должна учитывать особенности представления данных в МП:

– выполнять бинарные целочисленные операции арифметико-логического устройства (АЛУ);

– исключать неконтролируемую потерю значащих разрядов данных;

– контролировать потерю значащих разрядов при сдвиге вправо (округление типа «floor»).

Переход к целочисленному формату осуществляется путем записи дробных чисел модели в формате «фиксированная точка» [3]. Для удобства записи условно назовем такой формат *d.m*, где *m* – длина дробной части в битах. Переход к формату *d.m* осуществляется путем выделения *m* двоичных разрядов для хранения дробной части исходного вещественного числа, что эквивалентно математической операции:

$$A_{d.m} = round(A \cdot 2^m), \tag{7}$$

где $A_{d.m}$ – представление числа *A* в формате *d.m*;

A – исходное вещественное число в формате «плавающая точка»;

round – оператор округления до ближайшего целого числа.

Точность представления числа в десятичной системе счисления определяет следующее условие, накладываемое на битовую длину дробной части формата *d.m*:

$$m_f \geq round(\log_2(10^{fd}) + 0{,}5), \;\; fd \geq 0, \tag{8}$$

где m_f – битовая длина дробной части числа в формате *d.m*;

fd – требуемая точность в десятичных позициях после запятой;

round – оператор округления до ближайшего целого числа.

Общая битовая длина числа *A* в формате *d.m* составляет:

$$L_A = ceil\left(\log_2\left(\left|A_{d.m}\right|\right)\right)+1 = d_A + m_A + 1, \quad (9)$$

где *ceil* – оператор округления до наибольшего целого числа;

d_A – битовая длина целой части числа *A*;

m_A – битовая длина дробной части числа *A*.

Представление числа *A* = 10,07 в форматах *d.4*, *d.7* и *d.11* приведено в таблице 2.

Таблица 2 – Представление числа в формате «фиксированная точка»

Формат	Число $A_{d.m}$	Длина, Бит	Фактическое значение (A_{fd})	Ошибка представления / количество десятичных разрядов дробной части
d.4	161	8	10,0625	– 0,0075 / 1
d.7	1289	11	10,0703125	+ 0,0003125 / 2
d.11	20623	15	10,06982421875	– 0,00017578125 / 3

Битовая длина дробной части числа в формате *d.m*, как и общая битовая длина числа, ограничена максимальной длиной разрядной сетки данных МП (*L = 40* бит). При невозможности размещения результата в разрядной сетке возникает «переполнение» («overflow») и происходит перенос верхних значащих разрядов в младшие разряды с изменением знака. Другой специфической ошибкой бинарного представления является «антипереполнение» («underflow»), когда, например, в

результате деления может происходить потеря всех значащих разрядов результата [3].

Правила выполнения математических операций с числами формата «фиксированная точка», представленные в таблице 3, определяются формулой (7). Сдвиги на a разрядов влево (условное обозначение $<< a$) или вправо ($>> a$) осуществляются для защиты от потери значащих разрядов (нормирующий пред-сдвиг). Операция сдвига вправо равнозначна делению с исключением дробной части результата и задает тип округления данных в МП, который следует учитывать при создании «целочисленной» модели. Для корректного выполнения операций в формате «фиксированная точка» и правильной интерпретации результата необходимо знать текущее количество разрядов дробной части, представленное во втором столбце таблицы 3.

Таблица 3 – Правила выполнения базовых математических операций в формате «фиксированная точка» с защитой от потери данных

Операция	Выполнение операции	Длина дробной части	Соотношения форматов и правила сдвиговых операций
$A_{d.m} \pm B_{d.n}$	$A \pm (B << (m-n))$	M	для $m > n$
$A_{d.m_A} \cdot B_{d.m_B}$	$(A >> a) \cdot (B >> b)$	$m_A + m_B - a - b$	$m_A + m_B \le a + b$ $a + b \le L - (L_A + L_B) - 1$
$\frac{A_{d.m_A}}{B_{d.m_B}}$	$\frac{A << a}{B >> b}$	$m_A - m_B + a - b$	$a \le L - L_A - 1,$ $m_B - m_A + 1 \le a + b$

Если точность результата операции превышает заданную требуемую точность,

то применяется нормирующий точность пост-сдвиг вправо a_{fd}, рассчитываемый по формуле:

$$a_{fd} = fix(\lg(2^{m_A})) - fd_A, \tag{10}$$

где fd_A – требуемое количество десятичных позиций дробной части числа A,

m_A – текущая битовая длина дробной части числа A,

fix – оператор округления с исключением дробной части.

Отметим, что операция сдвига является самой быстрой операцией процессора и не оказывает существенного влияния на скорость выполнения АУ.

Если операции нормирующих сдвигов, представленные в таблице 3, нарушают условие (8), то результат имеет пониженную точность представления, что требует введения корректирующих коэффициентов, повышающих точность. Корректирующий коэффициент в формате $d.m$ равен отношению эталонного результата математической операции в вещественном формате к результату, полученному в «целочисленной» модели:

$$K_{кор}\{d.m\} = round\left(\frac{Rs\{f\}}{Rs\{d.n\}}\{f\}\cdot 2^m\right), \tag{11}$$

где $Rs\{f\}$ – результат выполнения операции в вещественном формате;

$Rs\{d.n\}$ – результат выполнения операции в формате «фиксированная точка».

Формула (8) была использована для записи коэффициентов и данных контура управления с «начальной» точностью в формате $d.m$ при создании прототипа «целочисленной» модели. Все блоки модели, реализующие алгоритм управления в МП, работают с целочисленными форматами длиной не более 40 бит (внутренние

форматы Simulink – «*int*» и «*sfix*»). Для контроля правильности решения модели используются встроенные в MATLAB форматы «фиксированная точка» (*fixdt*) и встроенные в Simulink детекторы переполнений и антипереполнений, настраиваемые опцией «*Simulation*» – «*Configuration Parameters*» – «*Data Validity*».

Создание целочисленных динамических звеньев

ДЗ АУ реализуется в целочисленной модели на основе методов (4) – (6). Из двенадцати представленных в АУ ДЗ девять составляют интеграторы, а остальные ДЗ можно реализовать на их основе. Приблизительное время выполнения в МП методов численного решения по формулам (4) – (6) при реализации интегрирования и использовании 40-битных переменных составляет 1,25, 2 и 4,5 мкс соответственно. Применение формулы (4) ограничено низкой точностью решения, а применение формулы (6) – повышенным временем выполнения в МП. В связи с этим на начальном этапе для реализации ДЗ в МП была выбрана формула (5) немодифицированного метода Эйлера-Коши (билинейное преобразование) [2].

При единичном коэффициенте передачи алгоритм интегрирования входного сигнала $x[n]$ по формуле (5) можно записать в виде:

$$y[n] = y[n] \cdot z^{-1} + \frac{T}{2} \cdot \left(x[n] + x[n] \cdot z^{-1}\right), \qquad (12)$$

где $x[n]$ – вход интегратора;

$y[n]$ – выход интегратора;

T – длительность такта решения;

z^{-1} – элемент задержки на такт.

Структурная схема целочисленного интегратора на основе билинейного преобразования со сбросом начальных условий (НУ) представлена на рисунке 2.

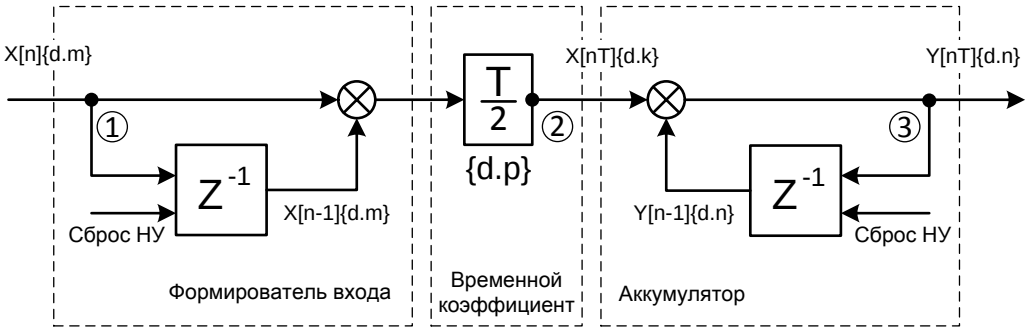

Рисунок 2 – Структурная схема численного интегрирования методом Эйлера-Коши

На рисунке 2 пунктиром выделены основные составные части алгоритма интегрирования. В фигурных скобках указан формат представления данных при выполнении математических операций.

В случае реализации алгоритма интегрирования (12) в вещественном формате обеспечивается высокая точность записи временного коэффициента $T/2$, трансформирующего «такты» МП в «физическое» время. За счет значительно более высокого диапазона представления чисел в вещественном формате, перенос коэффициента $T/2$ из точки «2» в точку «1» или в точку «3» не влияет на результат выхода. При целочисленных операциях перенос множителя $T/2$, который в целочисленном представлении всегда больше единицы, не является коммутативным. Перенос его в точку «1» увеличивает вероятность переполнения разрядной сетки, а перенос в точку «3» снижает точность выхода. В целых числах, как правило, искажается значение коэффициента $T/2$ (формат *d.p*), что требует введения дополнительной коррекции по схеме (11). Таким образом, реализация

алгоритма (12) в целых числах требует введения нормирующих сдвигов согласно правилам, приведенным в таблице 3.

Одна из возможных реализаций алгоритма (12) в системе Simulink представлена на рисунке 3.

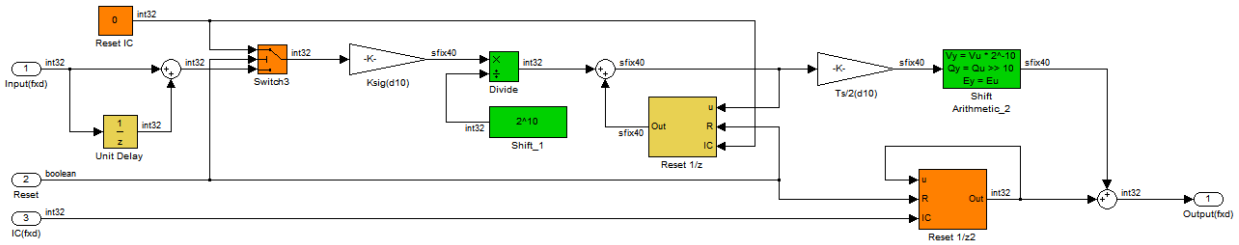

Рисунок 3 – Simulink-модель целочисленного интегратора Эйлера-Коши

Желтым цветом на схеме показаны элементы задержки на такт в составе формирователя входа и аккумулирующего контура. Элементы модели, выделенные оранжевым цветом, реализуют коррекцию начальных условий (*Reset IC* и *IC(fxd)*) по внешнему сигналу (*Reset*), а элементы модели, выделенные зеленым цветом (*Shift_1* и *Shift Arithmetic_2*), реализуют нормирование дробной части данных. Система Simulink допускает два способа нормирования – с помощью стандартного блока *Shift Arithmetic* или делением на сдвиговый коэффициент с последующим округлением типа *Zero*. При нормировании данных в МП рекомендуется использовать двоичный сдвиг как более быструю операцию.

Коэффициент модели «*Ts/2(d10)*» является временным коэффициентом, записанным в формате *d.10*. Коэффициент *Ksig(d10)* учитывает искажение коэффициента $T/2$ при записи в целочисленной форме и рассчитывается согласно (11). Отметим, что с помощью подбора длины дробной части формата

«фиксированная точка» можно добиться взаимного сокращения операций умножения на коэффициенты *Ksig* и *Ts/*2 и нормирующих сдвигов *Shift_1* и *Shift Arithmetic_2*, например, для частоты решения 500 Гц коэффициент *Ts/*2*(d10)* будет равен единице. Представленная на рисунке 3 простейшая базовая модель интегратора работает на частоте 500 Гц, имеет четыре 40-битных операнда и приблизительное время выполнения в МП около 2 мкс.

Увеличение частоты дискретизации данных приводит к увеличению количества операций сложения в аккумуляторе, уменьшая время работы блока без переполнения (точка «3» рисунка 2). В то же время для сигналов с низкой динамикой частота интегрирования существенно не влияет на ошибку, следовательно, возможно интегрирование с пониженной частотой. Понижение номинальной частоты решения (500 Гц) значительно влияет на ошибку решения только трех интеграторов АУ, для решения остальных можно использовать более низкие частоты. Снижение частоты решения интеграторов возможно вплоть до частоты расчета управляющих команд – 10 Гц.

Точность базовой модели может быть улучшена не только путем повышения порядка метода численного решения, но и применением некоторых оптимизированных методов обработки и хранения данных. Примерами таких методов являются: размещение данных в нескольких переменных (далее условно называемый «Метод векторных вычислений»), управляемые сбросовые схемы (далее «Метод искусственных переполнений»), а также динамическое изменение

частоты решения, порядка численного метода, формата «фиксированная точка» и конвейеризация обработки данных.

При использовании «метода векторных вычислений» выделенные группы разрядов входа («координаты вектора», например, биты 0 – 15 и 16 – 31 исходного 32-битного целого числа в формате «фиксированная точка») интегрируются независимо. После проведения интегрирования данные, полученные от интегратора старших бит, сдвигаются на $16-n$ разрядов влево и суммируются с данными интегратора младших бит, сдвинутыми на n разрядов вправо («преобразование в скаляр»). Число n и «размерность вектора» подбирается исходя из необходимой точности выхода.

Модель интегратора с алгоритмом «векторных» вычислений представлена на рисунке 4.

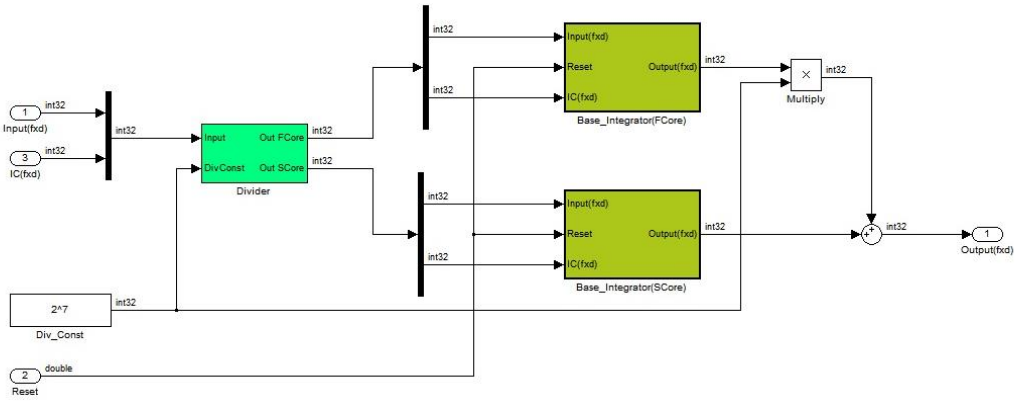

Рисунок 4 – Simulink-модель двухзвенного интегратора

Логический блок «метода искусственных переполнений» обнуляет аккумулирующий контур интегратора при достижении определенного порога. На выходе расположен элемент памяти, запоминающий и учитывающий данные выхода

в моменты контролируемых переполнений. Simulink-модель аккумулятора «метода искусственных переполнений» представлена на рисунке 5.

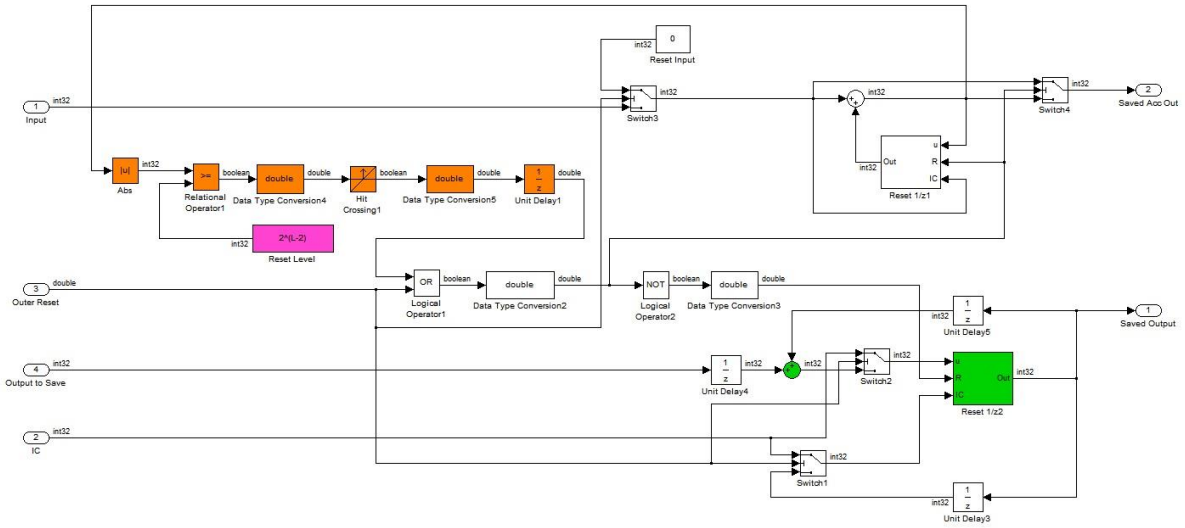

Рисунок 5 – Simulink-модель аккумулятора с логическим сбросом

Ошибки решения целочисленного алгоритма (5) (формат 40 бит) и альтернативных схем интегрирования относительно решения методом Рунге-Кутта четвертого порядка представлены на рисунке 6.

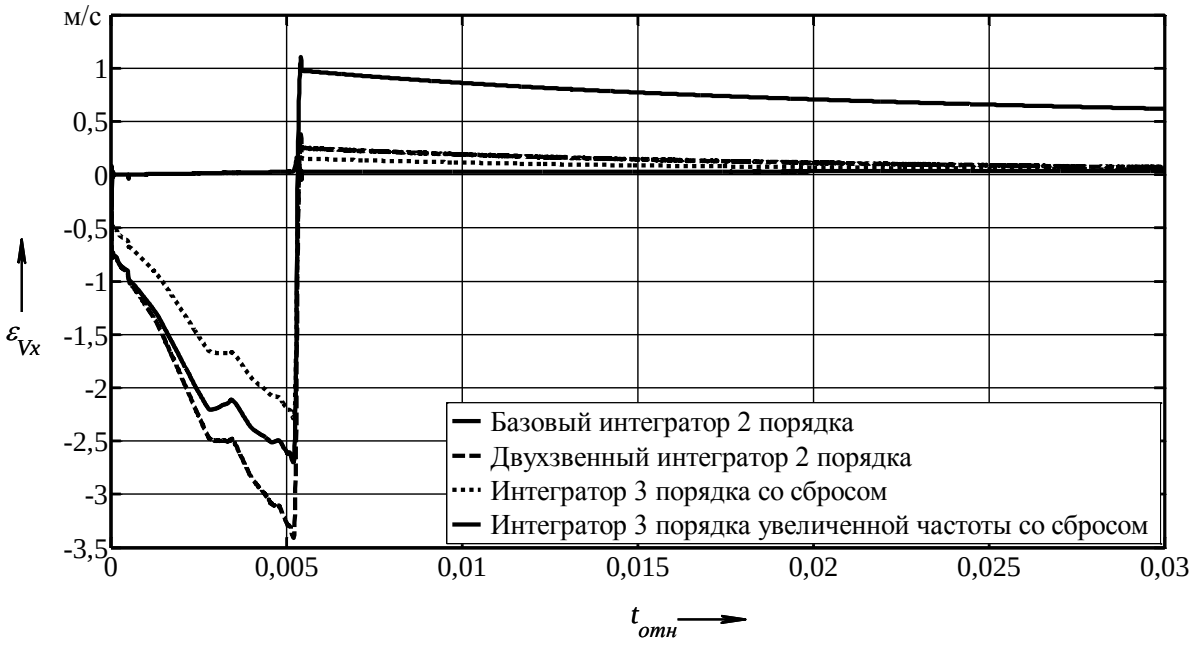

Рисунок 6 – Сравнение ошибок алгоритмов интегрирования

Расчетное время исполнения программного кода улучшенных алгоритмов превышает базовое в полтора – три раза, но их применение позволяет уменьшить ошибки, связанные с ограничениями разрядной сетки, а также увеличить частоту решения блока и точность записи коэффициентов (*Ksig* и *Ts/2)*, что повышает точность решения.

Создание целочисленных функций

Заданный АУ содержит около 77 функций, использующихся для определения длины вектора и углов его направления в декартовой системе координат и вычисления координат точки при повороте осей системы координат. Математически эти задачи решаются с помощью прямых и обратных тригонометрических функций и функции извлечения квадратного корня. Данные функции содержатся в наборе стандартных функций МП, но они имеют значительное время выполнения ввиду того, что работают с переменными формата «плавающая точка».

Таким образом, функции алгоритма управления должны быть представлены специализированными алгоритмами, оперирующими аргументами функций, заданными в двоичной системе счисления. В настоящее время существует большое количество методов вычисления функций в МП [4], но для двоичной системы счисления наиболее эффективным оказывается использование табличных и итерационных методов.

При решении «физической» модели АУ были получены область определения и область значений функций. Функции с малой областью значений функции (например, функции «синус» и «косинус») реализуются в МП в виде целочисленных

таблиц, генерируемых и проверяемых с помощью функций MATLAB. Для сокращения объемов таблицы генерируются в узком диапазоне области определения, а в дальнейшем распространяются на полный требуемый диапазон.

Функции с большими диапазонами данных реализуются в целочисленной модели АУ способом Волдера алгоритма CORDIC (COordinate Rotation in a DIgital Compute) [4]. С помощью рекуррентных соотношений, содержащих только операции сдвига и алгебраического сложения, CORDIC-алгоритм выполняет «поворот» вектора на плоскости на произвольный угол.

Для *i+1*-го шага решения CORDIC-алгоритма справедливы соотношения:

$$\begin{cases} x_{i+1} = K_i \cdot \left(x_i - \sigma_i y_i \cdot 2^{-i} \right), \\ y_{i+1} = K_i \cdot \left(y_i + \sigma_i x_i \cdot 2^{-i} \right), \\ z_{i+1} = z_i + \sigma_i \cdot \alpha_i, \\ i = 0,1,2,\ldots n, \end{cases} \tag{13}$$

где $K_i = \cos(arctg(2^{-i}))$ – коэффициент деформации вектора на *i*-й итерации;

z_{i+1} – текущий угол поворота вектора;

$\alpha_i = arctg(2^{-i})$ – целочисленные коды элементарных углов поворота;

n – количество шагов решения.

Углы α_i задаются таблично в формате «фиксированная точка», а начальные условия x_0, y_0, z_0 и вид оператора σ_i выбираются исходя из вида решаемой функции, например, для вычисления функции *arctg(Y/X)* в первой четверти необходимо установить:

$$\begin{cases} \sigma_i = sign(y_i), \\ x_0 = X, \\ y_0 = Y, \\ z_0 = 0. \end{cases} \quad (14)$$

Для решения функции $\alpha = arctg(Y/X)$ в диапазоне $|\alpha| \leq 180°$ необходимо преобразовать начальные условия z_0, используя свойства тригонометрических функций, а начальные условия x_0, y_0 – привести к первой четверти. После выполнения n итераций на выходе CORDIC-алгоритма получается:

$$x_n = K_n\sqrt{X^2 + Y^2},\ y_n \approx 0,\ z_n \approx arctg\left(\frac{Y}{X}\right), \quad (15)$$

где $K_n = \prod_{i=0}^{n-1}\cos(arctg(2^{-i})) \approx 1{,}6473$ – общий коэффициент деформации вектора.

Из формулы (15) видно, что одновременно можно вычислить как длину вектора, так и его угол относительно оси X. Используя CORDIC-алгоритм в таком режиме для нескольких плоскостей и подбирая число итераций n, можно добиться высокой точности решения пространственной геометрической задачи, предусмотренной алгоритмами управления. Реализация CORDIC-алгоритмов осуществляется в виде отдельных m-функций и проверяется независимо от отработки «целочисленной» Simulink-модели.

Тестирование «целочисленной» модели

После преобразования всех математических операций, предусмотренных АУ, в целочисленный формат получается «целочисленная» модель АУ с требуемой точностью выходных данных. Точность «целочисленной» модели подтверждается

путем сравнения её данных с тестовыми, полученными от «физической» модели алгоритма управления.

Часть математических операций АУ (функциональный блок АЦУ рисунка 1) можно выполнить в вычислительном комплексе носителя УР с целью «разгрузки» бортовой МПС. Такие математические операции и данные можно легко выделить на этапе работы с Simulink. Они обрабатываются в модели в формате «плавающая точка», а в МПС передаются в виде целочисленных констант.

Ошибки расчета команд управления «целочисленной» моделью в заданном поле команд «физической» модели $U_{упр} = \pm 2{,}5\ B$ представлены на рисунке 7.

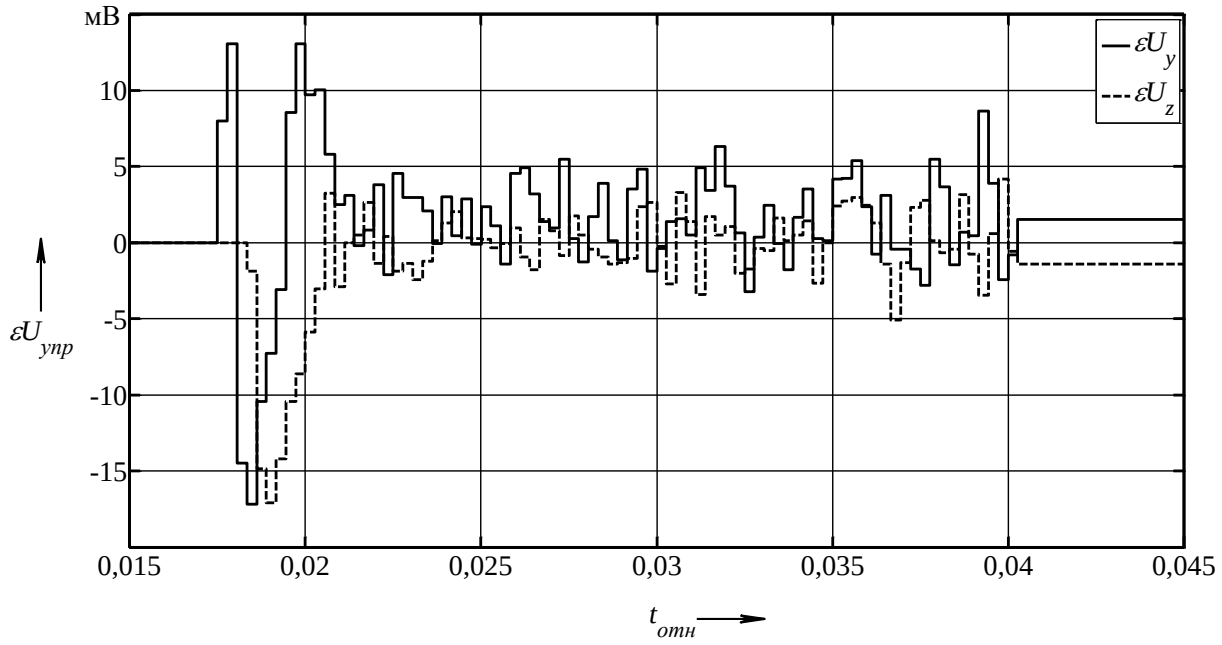

Рисунок 7 – Ошибки расчета команд управления «целочисленной» моделью в вертикальном εU_y и горизонтальном εU_z канале

После подтверждения точности вычислений модели и выделения области АУ для «переноса» на борт УР можно приступить к созданию ПК. Создание ПК МП TMS320C55xx выполняется на языке C++ посредством поэтапной интерпретации

математических операций модели. Отметим, что язык MATLAB поддерживает автоматическую генерацию C и C++кода встроенными инструментами. Перевод моделей и m-функций из системы MATLAB на синтаксис языка C и C++ вручную также не вызывает затруднений. Это связано с хорошей визуализацией математических операций в системе Simulink и правилами построения m-функций, присущими многим языкам высокого уровня. Таким образом, создание прототипа и отладка ПК на основе «целочисленной» модели могут быть выполнены в сжатые сроки.

После передачи «целочисленной» модели программисту, работающему на языке C++, исполнитель АУ в системе MATLAB может приступить к ее оптимизации, снабжая программиста данными, необходимыми для отладки ПК.

Полученная «целочисленная» модель:

– оптимизирована под целочисленные вычислительные системы, так как выполняет все математические операции контура управления путем «быстрых» операций бинарной арифметики;

– оптимизирована по количеству выполняемых математических операций, удобна для отладки и позволяет заменить отладчики встроенных МПС вычислительной средой MATLAB-Simulink;

– может быть легко преобразована в ПК на языках высокого уровня за счет высокоуровневой модульной структуры MATLAB-Simulink;

– сокращает время создания ПК МПС УР, уменьшает количество ошибок при программировании и время корректировки ПК при изменении АУ.

Работоспособность созданного по такой методике ПК проверена в ходе лабораторных испытаний и подтверждена натурными испытаниями УР.

Библиографический список

1. Корн Г.А., Корн Т.М. Справочник по математике. – Москва: Наука, 1974. – 832 с.

2. Бесекерский В.А., Попов Е.П. Теория систем автоматического регулирования. – Москва: Наука, 1972. – 768 с.

3. Солонина А.И., Улахович Д.А., Яковлев Л.А. Алгоритмы и процессоры цифровой обработки сигналов. – СПб: БХВ-Петербург, 2002. – 464 с.

4. Байков В.Д., Смолов В.Б. Аппаратурная реализация элементарных функций в ЦВМ. – Ленинград: Изд-во ЛГУ, 1975. – 96 с.