

Задание на курсовой проект по СУЭП

Общие положения

Курсовой проект выполняется на отладочных платах Texas Instruments controlCARD CC2803X на базе микроконтроллера Piccolo F28035 (или аналогичных). Студентам выдается готовый проект для среды Code Composer Studio v5 (и выше), содержащий модель системы «преобразователь — двигатель». Студенты получают от преподавателя индивидуальное задание на разработку системы управления конкретным типом привода с заданной структурой системы управления. Отладка проекта может осуществляться в лаборатории Учебно-научно-консультационного центра «Texas Instruments – МЭИ».

Описание модели

Двигатели подключены к инверторам, каждый из которых имеет три стойки. Управление инвертором выполнено подобно реализации в микроконтроллерах. Задается общий период ШИМ в тиках таймера, производящего счет вверх/вниз с частотой тактирования 60 МГц. При заданной частоте ШИМ значение периода определяется выражением:

$$drive.tpr = \frac{60\,000\,000}{2 f_{\text{ШИМ}}}$$

где $drive.tpr$ — период таймера в тиках, $f_{\text{ШИМ}}$ — частота ШИМ в Гц.

Для всех стоек одновременно задается величина «мертвого времени» — $drive.dt$, в тиках таймера ШИМ. Для каждой стойки задается скважность ($drive.cmpr1$, $drive.cmpr2$, $drive.cmpr3$) управления от 0 до $drive.tpr$. Подключение к инвертору для асинхронных и синхронных двигателей производится по схеме рис. 1. Для двигателя постоянного тока — по схеме рис. 2.

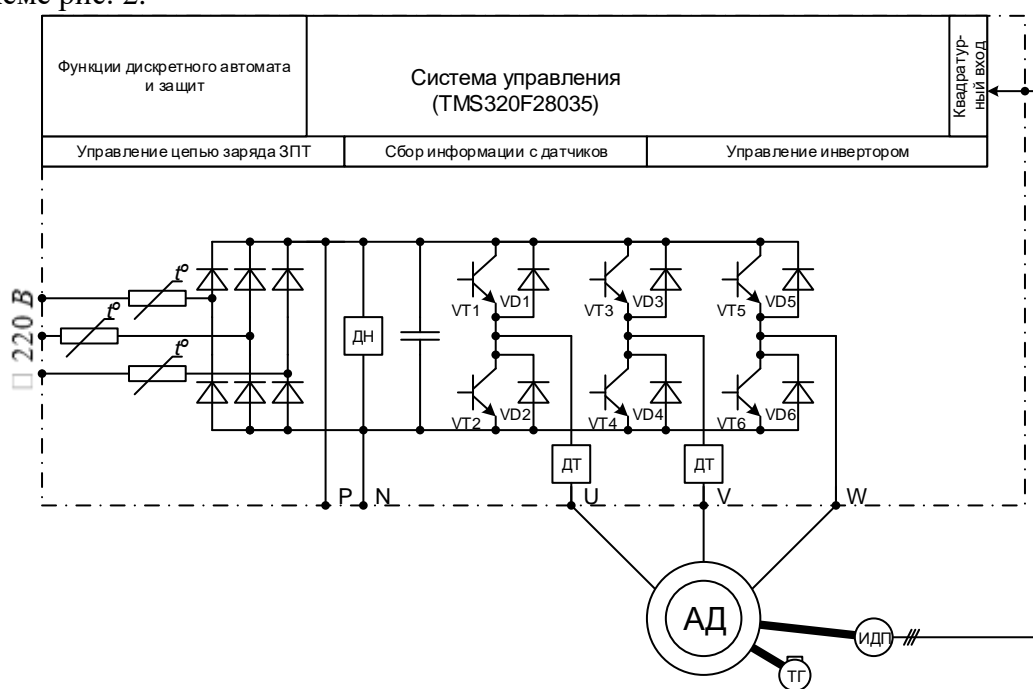


Рис. 1. Подключение асинхронного и синхронного электродвигателя

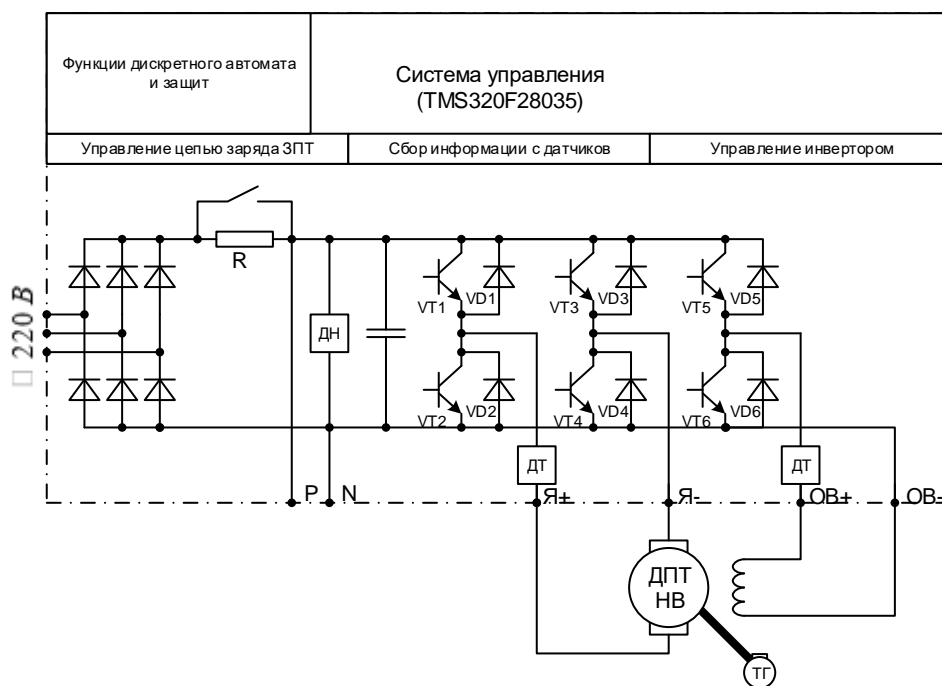


Рис. 2. Подключение двигателя постоянного тока

Напряжение звена постоянного тока составляет 540 В.

Датчики токов фаз заводятся на АЦП микроконтроллера в формате ± 5 В. -5 В соответствует коду 0, а $+5$ В соответствует коду 4095. Если выходной сигнал датчика тока выходит за диапазон входа АЦП, преобразователь отключается, выставляя сигнал аварии.

Датчик тока обмотки возбуждения имеет коэффициент 800 единиц на Ампер. То есть при протекании тока в 1 А с АЦП приходит код 800.

Двигатель имеет тахогенератор, сигнал с которого заводится на АЦП. Текущая скорость (в об/мин) определяется по формуле $n = (x_{\text{АЦП}} - 2047) \cdot k_{\text{ТГ}}$.

На валу двигателя имеется импульсный датчик положения с разрешением на оборот, указанным для каждого варианта в отдельности. Код положения доступен системе управления.

На валу синхронной машины имеется датчик положения на эффекте Холла. Датчик имеет три элемента, сдвинутые на 120 электрических градусов. Первый элемент соответствует нулевому биту *drive.hallSensor* и включен в диапазоне углов от $-\frac{\pi}{2}$ до $+\frac{\pi}{2}$.

Таблицы двигателей с исходными данными

Варианты с 1 по 10 — асинхронные двигатели с напряжением питания 220 В (действующее фазное)

Варианты с 11 по 20 — синхронные неявнополюсные двигатели с постоянными магнитами

Таблица 1 – Варианты асинхронных двигателей и синхронных двигателей с постоянными магнитами

№ варианта	Число пар полюсов	Сопротивление резистора измерительной цепи датчика тока	Число первичных витков при коэффициенте датчика тока 1000:1	Коэффициент преобразования сигнала ТГ, (об/мин)/АЦП	Количество импульсов на механический оборот квадратного датчика положения ротора
1	1	82	5	2	10000
2	1	68	5	2	5000
3	1	51	2	2	3000
4	1	51	1	2	2000
5	1	27	1	2	1000
6	2	82	5	1	5000
7	2	51	1	1	5000
8	2	33	1	1	5000
9	3	68	3	0,667	3000
10	3	22	3	0,667	3000
11	1	100	5	2	1000
12	1	68	5	2	2500
13	1	51	2	2	3000
14	1	51	1	2	5000
15	1	26	1	2	10000
16	2	100	5	1	5000
17	2	51	1	1	2500
18	2	33	1	1	2000
19	3	82	3	0,667	3000
20	3	24	3	0,667	9000

Варианты с 21 по 30 – Двигатели постоянного тока

Таблица 2 – Варианты двигатели постоянного тока

№ варианта	Номинальная скорость	Сопротивление резистора измерительной цепи датчика тока	Число первичных витков при коэффициенте датчика тока 1000:1	Коэффициент преобразования сигнала ТГ, (об/мин)/АЦП	Количество импульсов на механический оборот квадратного датчика положения ротора
21	1000	51	2	0,667	1000
22	2000	51	2	2	2500
23	1500	68	2	2	3000
24	1000	51	2	0,667	5000

№ варианта	Номинальная скорость	Сопротивление ротора измерительной цепи датчика тока	Число первичных витков при коэффициенте датчика тока 1000:1	Коэффициент преобразования сигнала ТГ, (об/мин)/АЦП	Количество импульсов на механический оборот квадратурного датчика положения ротора
25	2000	33	2	2	10000
26	3000	51	1	2	5000
27	2000	68	1	2	2500
28	3000	51	1	2	2000
29	1500	51	1	1	3000
30	1000	33	2	0,667	9000

Задание

1. Совершить пуск двигателя в среде Code Composer Studio. Это может быть частотный, софт-стартерный пуск или пуск напряжением (для двигателя постоянного тока).
2. Определить частично или полностью параметры исследуемого двигателя. Оценить мощность двигателя. Разработать систему базовых величин для исследуемого двигателя. Для двигателя постоянного тока обязательно необходимо определить номинальное напряжение обмотки возбуждения.
3. Реализовать модель двигателя в среде Simulink пакета Matlab.
4. По заданию, выданному преподавателем, разработать структурную схему системы управления. Описать принципы работы представленной структуры. Произвести синтез регуляторов системы управления.
5. Реализовать систему управления в среде Simulink пакета Matlab.
6. Настроить регуляторы и проверить правильность работы системы управления.
7. Перевести структуру системы управления в относительные единицы для программной реализации.
8. Реализовать систему управления программно в среде Code Composer. Проверить работу системы управления, при необходимости подстроить регуляторы.
9. Снять статические и динамические характеристики получившейся системы.
10. Подготовить расчетно-пояснительную записку, содержащую описательную часть, расчеты и экспериментальные графики.
11. Подготовить презентацию с основными итогами работы для защиты.

Пункты 3, 5 и 6 можно не выполнять, если студент уверен, что сможет реализовать систему в полном объеме в среде Code Composer.

Варианты систем управления

Назначаются преподавателем в зависимости от типа привода, который изучает студент в рамках бакалаврской, магистерской или дипломной работы.

Для асинхронных двигателей:

1. Скалярная система управления с S-образным задатчиком интенсивности.
2. Скалярная система управления с компенсацией скольжения по оценке момента.
3. Скалярная система управления с отрицательной обратной связью по скорости.
4. Векторная система управления с ориентацией по потокосцеплению ротора с датчиком скорости.
5. Прямое управление моментом с релейными регуляторами и наблюдателем потокосцеплений статора по статорным уравнениям.

6. Прямое управление моментом с релейными регуляторами и наблюдателем потокосцеплений ротора (с датчиком скорости) с пересчетом в статорные.
7. Векторное управление с релейными регуляторами токов.
8. Векторное управление с непрерывными контурами токов и контуром положения.
9. Векторная система управления с непрерывными регуляторами токов и ориентацией по потокосцеплению статора.
10. Векторная бездатчиковая система с оценкой ЭДС двигателя.

Для синхронных двигателей:

1. Система с автокоммутацией и ограничением тока в фазе.
2. Система с автокоммутацией и ограничением тока в фазе с контуром положения.
3. Система векторного управления с ориентацией по потоку ротора.
4. Система векторного управления с размагничиванием по оси d при достижении ограничения по выходу инвертора.
5. Система прямого управления моментом с релейными регуляторами.
6. Система прямого управления моментом с релейными регуляторами и контуром положения.
7. Система векторного управления с релейными регуляторами.
8. Система векторного управления с контуром положения.
9. Система бездатчикового векторного управления с оценкой ЭДС двигателя.
10. Система прямого управления моментом с непрерывными регуляторами момента и потока.

Для двигателей постоянного тока:

1. Подчиненное регулирование тока и скорости.
2. Модальное управление.
3. Двухзонное подчиненное регулирование.
4. Подчиненное регулирование с релейным регулятором тока.
5. Регулирование с релейным контуром тока якоря и регулированием скорости в канале возбуждения.
6. Регулирование с непрерывным контуром тока якоря и регулированием скорости в канале возбуждения.
7. Подчиненное регулирование с оценкой скорости двигателя.
8. Регулирование момента ДПТНВ с ПИ-регулятором тока якоря и коррекцией возмущающего воздействия ЭДС двигателя.
9. Система подчиненного регулирования с контуром положения.
10. Подчиненное регулирование с релейными контурами тока и скорости.

Описание проекта

Проект для реализации системы управления в среде Code Composer полностью собран и готов для работы на плате controlCARD CC2803X, находящейся в лаборатории аудитории М-215.

В составе проекта находятся файлы библиотек реального времени, IQ-математики, файлы описания прерываний и стандартных процедур инициализации. Они не требуют каких-либо модификаций.

В проект включен файл model.obj, содержащий модель преобразователя и двигателя. Вызов процедур модели осуществляется из файла main.c:

```
// Подключение необходимых заголовочных файлов
#include "DSP28x_Project.h"
#include "model.h"

// Определение структуры модели двигателя
MODELDATA drive = MODEL_DEFAULTS;
```

```
// Заготовка функции расчёта системы управления
void controlSystem (void){
    // drive.cmp1 = ...;
    // drive.cmp2 = ...;
    // drive.cmp3 = ...;
}

// Прототип функции-обработчика прерывания
interrupt void controlIsr(void);

// Основная функция "main()", в которой происходит инициализация системы
int main(void) {
    // Настройка периферии микроконтроллера
    InitSysCtrl();
    memcpy(&RamfuncsRunStart, &RamfuncsLoadStart, &RamfuncsLoadEnd - &RamfuncsLoadStart);
    InitFlash();
    InitPieCtrl();
    InitPieVectTable();

    // Настройка таймера для прерываний на частоте 5 кГц
    EALLOW;
    EPwm1Regs.TBPRD = 6000;
    EPwm1Regs.TBCTL.bit.CLKDIV = TB_DIV1;
    EPwm1Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;
    EPwm1Regs.ETSEL.bit.INTEN = 1;
    EPwm1Regs.ETSEL.bit.INTSEL = TB_CTR_ZERO;
    EPwm1Regs.ETPS.bit.INTPRD = 1;

    PieVectTable.EPWM1_INT = controlIsr;
    PieCtrlRegs.PIEIER3.all = M_INT1;
    IER |= M_INT3;
    EPwm1Regs.TBCTL.bit.CTRMODE = TB_COUNT_UPDOWN;

    CpuTimer0Regs.PRD.all = 0xFFFFFFFF;
    CpuTimer0Regs.TCR.bit.TSS = 0;
    EDIS;

    // В первую очередь необходимо задать вариант в переменной "n"
    // и внутренний период таймера ШИМ 6000
    drive.n = 29;
    drive.tpr = 6000;

    // Теперь нужно вызвать функцию "Init" для инициализации модели
    drive.Init(&drive);

    // Глобально разрешаем прерывания
    EINT;

    // Бесконечный цикл
    while (1) {

    }
}

// Прерывание для расчёта модели двигателя
interrupt void controlIsr(void) {
    // Пользовательская функция с расчётом системы управления,
    // в которой осуществляется расчёт и присвоение уставок сравнения cmp1, cmp2, cmp3
    controlSystem();
}
```

```

// Выполнение модели двигателя
drive.Execute(&drive);

// Очистка флагов прерывания
EPwm1Regs.ETCLR.bit.INT = 1;
PieCtrlRegs.PIEACK.bit.ACK3 = 1;
}

```

Наклонным шрифтом отмечены места, которые могут модифицироваться студентом и куда он должен вписывать функции своей системы управления. Так, в начале функции **main** следует разместить инициализацию данных системы управления, а в **controlIsr** разместить саму систему управления.

Программа построена таким образом, что сначала запускается инициализация, а периодический вызов модели осуществляется в прерывании таймера, настроенного на частоту 5000 Гц. Частота работы виртуальной модели может отличаться, что не повлияет на период отработки одного прерывания. Кроме того, возможна ситуация, когда система управления будет выполняться дольше одного периода таймера прерываний, что приведет к неизбежному рассогласованию между модельным временем и реальным. Для оценки реального времени введена переменная **drive.time**, которая хранит модельное время в секундах с момента начала симуляции. Для частоты ШИМ 5000 Гц и системе управления, уместяющейся в период ШИМ вместе с расчетом модели в функции **drive.Execute**, модельное время будет совпадать с реальным.

Модель имеет заголовочный файл **model.h**:

```

#include "IQmathLib.h"

// faults
#define ILLEGAL_STUDENT_NUMBER 1
#define MAX_CURRENT_FAULT 2
#define MAX_SPEED_FAULT 3

// motor types (for internal use)
#define INDUCTION_MOTOR 1
#define SYNC_MOTOR 2
#define DC_MOTOR 3
#define SRD 4
#define SyncRM 5

#define PI 3.1415926535897932384626433832795
#define INV_2PI 0.15915494309189533576888376337251

typedef struct MODELDATA {
    unsigned int n; // Номер варианта
    unsigned int cmpr1; // Уставка сравнения для модуля ШИМ (стойка А)
    unsigned int cmpr2; // Уставка сравнения для модуля ШИМ (стойка В)
    unsigned int cmpr3; // Уставка сравнения для модуля ШИМ (стойка С)
    unsigned int tpr; // Период счётчика модуля ШИМ (для счёта вверх-вниз)
    unsigned int dt; // Мёртвое время
    unsigned int adcSpeed; // Код АЦП: скорость с тахогенератора
    unsigned int qepCounter; // Счётчик импульсов инкрементального энкодера
    unsigned int hallSensor; // Показания датчика Холла
    unsigned int iA; // Код АЦП: ток фазы А (или ток якоря для ДПТ)
    unsigned int iB; // Код АЦП: ток фазы В (или ток ОВ для ДПТ)
    unsigned int iC; // Код АЦП: ток фазы С (для ДПТ не используется)
}

```

```

    int fault;                // Код аварии
    float load;               // Нагрузка двигателя (Нм)
    float time;               // Время (с)
    void (*Init)(volatile struct MODELDATA*);
    void (*Execute)(volatile struct MODELDATA*);
} MODELDATA ;

typedef volatile struct MODELDATA Tmodel;

#define MODEL_DEFAULTS { 0,0,0,0,0,0,0,0,0,0,0,0,0, Model_Init, Model_Execute}

void Model_Init(Tmodel*);
void Model_Execute(Tmodel*);

```

Структура данных содержит:

unsigned int **n** – номер варианта (1-10 – асинхронные двигатели; 11-20 – синхронные двигатели; 21-30 – двигатели постоянного тока);

unsigned int **cmpr1** – скважность верхнего ключа первой стойки инвертора в тиках таймера;

unsigned int **cmpr2** – скважность верхнего ключа второй стойки инвертора в тиках таймера;

unsigned int **cmpr3** – скважность верхнего ключа третьей стойки инвертора в тиках таймера;

unsigned int **tp** – период ШИМ в тиках таймера с частотой тактирования 150 МГц, считающего вверх/вниз;

unsigned int **dt** – «мертвое время» в тиках таймера (возможно нулевое задание);

unsigned int **adcSpeed** – сигнал с тахогенератора, подключенного к АЦП, пропорциональный скорости вращения. Нулевой скорости соответствует число 2084;

unsigned int **qepCounter** – сигнал по положению с квадратурного импульсного датчика; обнуляется каждый оборот;

unsigned int **hallSensor** – сигнал с датчика на эффекте Холла; первому биту соответствует фаза А, второму – фаза В, третьему – фаза С;

unsigned int **iA** – сигнал с датчика тока фазы А (или якоря двигателя постоянного тока) после преобразования АЦП. Нулю тока соответствует число 2048;

unsigned int **iB** – сигнал с датчика тока фазы В (или возбуждения двигателя постоянного тока) после преобразования АЦП. Нулю тока соответствует число 0 для ДПТНВ и 2048 для асинхронного и синхронного двигателей;

unsigned int **fault** – признак аварии модели (0 – нет аварии, 1 – неверный вариант, 2 – максимально-токовая защита, 3 – защита по максимальной скорости). При срабатывании защиты модель прекращает работу, оставаясь в состоянии на момент аварии;

float **load** – нагрузка на валу двигателя в Н·м;

float **time** – текущее время модели в секундах от начала симуляции.

Таким образом, студент может воздействовать на параметры: **n**, **cmpr1**, **cmpr2**, **cmpr3**, **tp**, **dt**, **load**. Определять состояние системы возможно по параметрам: **adcSpeed**, **iA**, **iB**, **qepCounter**, **hallSensor**, **fault**.