БГУИР

Кафедра ЭВМ

Отчет по лабораторной работе № 3

Тактирование. Питание. Энергопотребление

Вариант №3

Выполнил:

студент группы 950502 Гуринович А.В.

Проверил:

Богдан Е.В.

Минск 2022

**1 ЦЕЛЬ РАБОТЫ**

Изучить принципы построения систем с низким энергопотреблением на базе микроконтроллера MSP430F5529.

**2 ИСХОДНЫЕ ДАННЫЕ**

В соответствии с вариантом написать программу, которая по нажатию одной кнопки последовательно переключает заданные тактовые частоты и напряжение питания, второй — входит и выходит в заданный режим пониженного энергопотребления. Для индикации активного режима использовать LED1, режима пониженного энергопотребления — LED3. Для индикации высокого напряжения питания использовать LED4, низкого — LED5. Дополнительно тактовую частоту подать на таймер, прерывание от таймера управляет LED8, и подобрать режим таймера так, чтобы наглядно было видно изменение частоты. При написании кода максимально использовать принципы разработки приложения для пониженного энергопотребления. Обязательно соблюдать условие соответствия частоты и напряжения, а также пошагового переключения режимов питания. Не допускается использовать иные заголовочные файлы, кроме msp430, а для переключения частот не допускается также использовать высокоуровневые библиотеки. Комментарии в тексте программы обязательны, они должны пояснять что именно делает данный фрагмент.

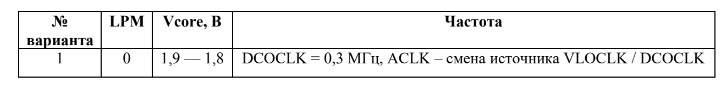
2. Измерить ток потребления микроконтроллера MSP430F5529 в каждом из режимов (измерительную цепь собирать при отключенном питании).

3. Измерить ток потребления экспериментальной платы MSP- EXP430F5529 в каждом из режимов (измерительную цепь собирать при отключенном питании).

4. Измерить напряжение питания Vcc в каждом из режимов.

5. Измерить с помощью осциллографа частоту MCLK в каждом из режимов.

6. В отчет по выполнению работы включить исходный текст программы и результаты измерений. Привести объяснение полученным результатам. Описать, как были рассчитаны установочные параметры для заданного режима. Описать, как именно и какие принципы разработки приложения для пониженного потребления энергии использовались.



**3 ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ**

# **Режимы пониженного энергопотребления**

Серия MSP430 разработана для использования в устройствах с низким энергопотреблением. Микроконтроллер поддерживает 7 режимов пониженного энергопотребления: LPM0 – LPM4, LPM3.5, LPM4.5. Режимы LPM0 – LPM4 конфигурируются битами CPUOFF, OSCOFF, SCG0, SCG1 регистра состояния микроконтроллера SR:

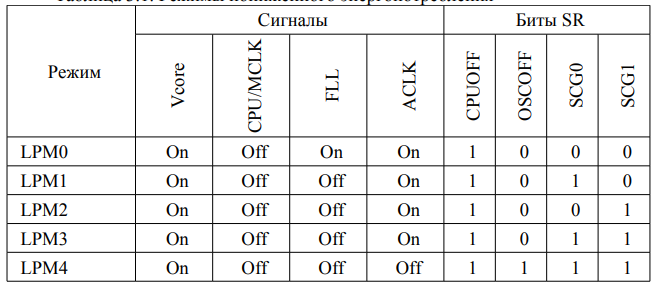
бит 4 – CPUOFF. Установка бита отключает процессор микроконтроллера;

бит 5 – OSCOFF. Установка бита отключает генератор частот, если только он не используется для формирования тактовой частоты MCLK или SMCLK;

биты 6, 7 – SCG0, SCG1 — отключение системного тактового генератора 0 (1), какого именно, зависит от серии устройства, например FLL или DCO.

Комбинации бит регистра состояния, соответствующие режимам, а также какие сигналы микроконтроллера при этом отключаются, приведены в таблице:

Таблица 3.1. Режимы пониженного энергопотребления



Как видим, во всех режимах отключается ЦПУ. При этом в LPM0 и в LPM1 тактирование периферии разрешено, DCO разрешен, если является источником для частот ACLK или SMCLK; в LPM2 и LPM3 разрешена только одна тактовая частота периферии ACLK (SMCLK отключена), DCO разрешен, если является источником для частоты ACLK; в LPM4 – все тактовые частоты запрещены, отключены SMCLK, ACLK, DCO. Самоактивация возможна для режимов LPM0 – LPM3, в этих режимах доступны следующие источники прерываний: таймеры, включая WDT и RTC, АЦП, DMA, USART, цифровые входы-выходы, компаратор, внешние прерывания, USCI, а также другая периферия микроконтроллера и внешние прерывания. В режиме LPM4 самоактивация невозможна, доступны только внешние прерывания. Установка любого из вышеназванных битов регистра состояния сразу же переводит микроконтроллер в требуемый режим. Предыдущий режим сохраняется в стеке (так как это регистр состояния) при выполнении прерывания. Биты CPUOFF, OSCOFF, SCG1 автоматически сбрасываются при обработке прерывания. Управление вернется к предыдущему режиму, если в обработчике прерываний сохраненное значение SR не изменяется, иначе при возврате установится новый режим. Периферия может быть отключена не только общим запретом тактирования, но и индивидуально установками в своих регистрах управления. В режимах LPM цифровые порты ввода-вывода и ОЗУ и регистры не изменяются. В режимах LPMx.5 (LPM3.5, LPM4.5) отключается регулятор напряжения блока управления питанием PMM, содержимое регистров и ОЗУ теряется, но состояние цифровых входов-выходов сохраняется. Выход из этих режимов возможен включением питания, сигналом сброса, а также отдельными периферийными модулями. LPM3.5, в отличие от LPM4.5, позволяет выйти из режима по сигналу от RTC. Выход из LPMx.5 приводит к сбросу, поэтому требуется повторное конфигурирование устройств. Режим LPMx.5 устанавливается битом PMMREGOFF и битами регистра статуса LPM4.

Общие принципы построения систем с низким энергопотреблением:

* максимально длительное время нахождения в режимах пониженного энергопотребления (особенно LPM3 и LPM4);
* использование прерываний для управления ходом выполнения программ;
* включение периферии только по мере необходимости;
* использование интегрированной периферии с низким потреблением энергии вместо программного выполнения функций;
* вычисление ветвей и использование таблиц значений вместо опроса флагов и длительных программных вычислений;
* избегать частого вызова функций и процедур из-за дополнительных затрат;
* использовать однотактные регистры ЦПУ в длинны процедурах;
* отключать недоступные и неиспользуемые сегменты памяти при помощи регистра управления ОЗУ RCCTL0.

# **Питание**

Модуль управления питанием PMM обладает следующими характеристиками:

* + входное напряжение 1,8 - 3,6 В;
  + 4 уровня напряжения для питания ядра (Vcore), задаваемых программно;
  + супервизор уровня напряжения (SVS - Supple Voltage Supervisor) с программируемым порогом как для выходного, так и для входного напряжения;
  + монитор уровня напряжения (SVM - Supple Voltage Monitor ) с программируемым порогом как для выходного, так и для входного напряжения;
  + сброс при низком питании (BOR);
  + программно доступные индикаторы сбоя питания;
  + защита выводов от сбоя по питанию.

Монитор изменения напряжения SVM только отслеживает выход за пороговое значение, супервизор SVS еще и генерирует сигнал сброса. Структура модуля PMM изображена на рисунке:

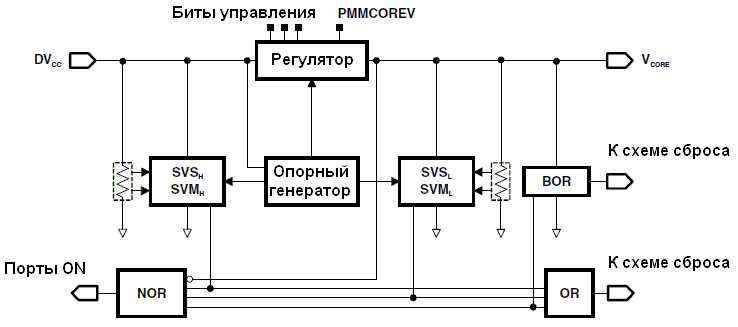
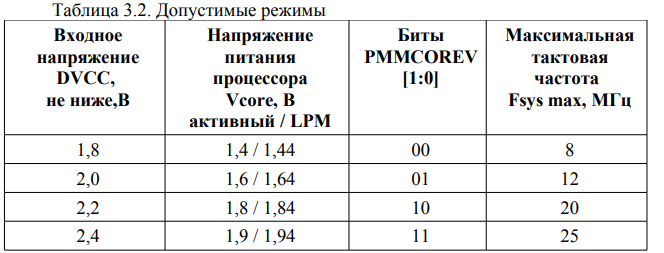


Рис. 3.1 Структура модуля управления напряжением

Повышение частоты тактирования MCLK требует повышения напряжения питания Vcore. Уровень питания определяется битами PMMCOREV. При переключении на более высокую частоту необходимо убедиться, что уровень напряжения питания достаточен для нее. Несоблюдение этого правила может привести к непредсказуемым результатам. Допустимые параметры для разных значений уровня питания приведены на рис. 3.2 и в таблице 3.2. На рисунке цифрами 1, 2, 3, 4 обозначены допустимые значения бит PMMCOREV

Таблица 3.2. Допустимые режимы



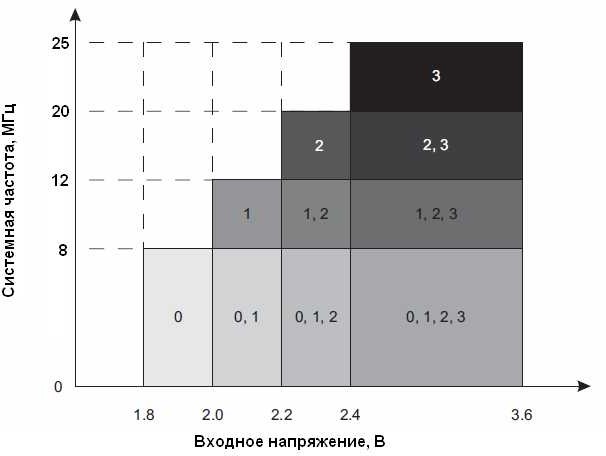


Рис. 3.2 Допустимые режимы

В регистрах монитора и супервизора устанавливаются соответствующие пороговые значения для контроля за уровнем напряжения. Принцип работы с пороговыми уровнями показан на рис. 3.3. В обозначениях SVS относится к супервизору, SVM – к монитору, H – контроль сигнала на высокой стороне (на входе), L – контроль сигнала на низкой стороне (на выходе). Супервизор имеет петлю гистерезиса: при пороге SVSIT- устанавливается соответствующий флаг прерывания и генерируется сигнал сброса, а при пороге SVSIT+ этот режим заканчивается, причем второй порог выше первого. Порог для монитора один, при напряжении питания меньше порогового устанавливается соответствующий флаг. Рекомендуемые значения порогов сведены в таблицу 3.3. Типовые уровни напряжения порога SVSH\_IT- равны 1,68; 1,88; 2,08; 2,18 В (для значений от 00 до 11 поля SVSHRVL) и порога SVSH\_IT+ 1,74; 1,94; 2,14; 2,30; 2,40; 2,70; 3,10; 3,10 В (для значений от 000 до 111 поля SVSMHRRL). Допустимыми значениями для SVSHRVL являются все 4 комбинации, а для SVSMHRRL — зависит от выбора уровня питания PMMCOREV (рис. 3.4).

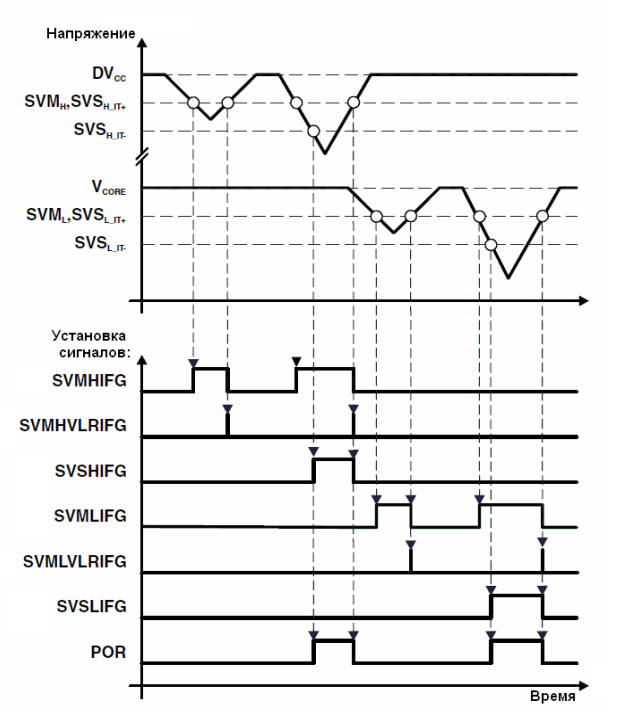
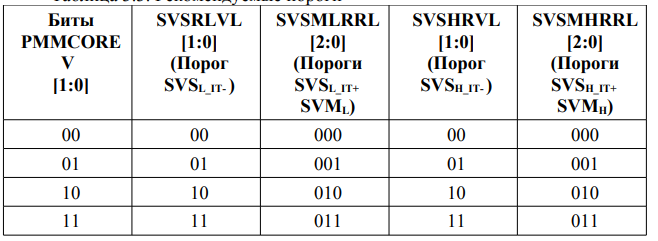


Рис. 3.3 Срабатывание порогов при изменении напряжения

Таблица 3.3. Рекомендуемые пороги



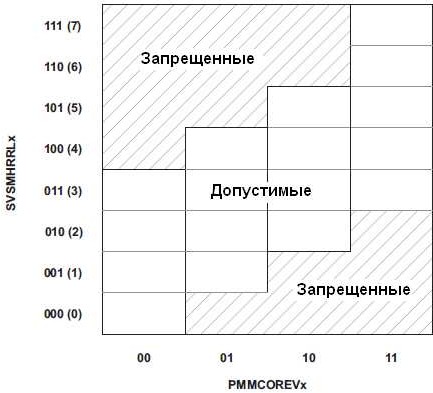


Рис. 3.4 Допустимые режимы порога SVSMHRRL

При изменении уровня Vcore нельзя увеличивать частоту MCLK, пока не установится новый уровень напряжения. Для проверки уровня Vcore используется SVML. При снижении уровня питания необходимо убедиться, что его будет достаточно для установленной частоты. Vcore можно изменять только на один уровень за 1 раз.

При увеличении уровня напряжения необходимо выполнить следующую последовательность действий (см. рис. 3.5):

1. Устанавливаются новые уровни SVMH, SVSH, чтобы убедиться, что входное напряжение DVCC выше планируемого Vcore. Установить новый уровень SVML, дождаться установки флага SVSMLDLYIFG.
2. Установить PMMCOREV для определения нового уровня Vcore.
3. Дождаться установки флага SVMLVLRIFG.
4. Установить новый уровень SVSL.

При снижении уровня напряжения необходимо выполнить следующее:

1. Устанавливаются SVML, SVSL для нового уровня и ожидается установка флага SVSMLDLYIFG.
2. Программируем новый уровень Vcore, задав PMMCOREV.

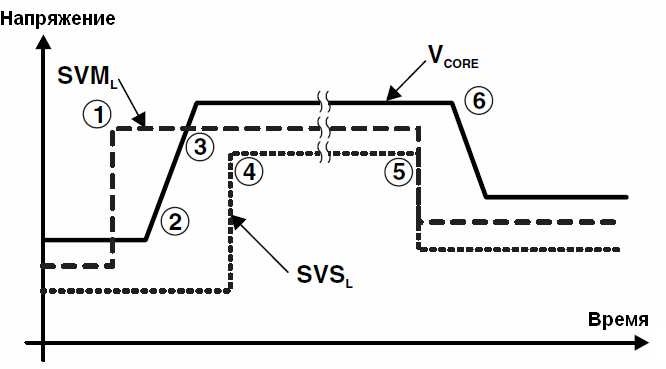


Рис. 3.5 Порядок изменения уровня напряжения питания

Для управления подсистемой питания используются регистры:

PMMCTL0 (0120h) - управление PMM;

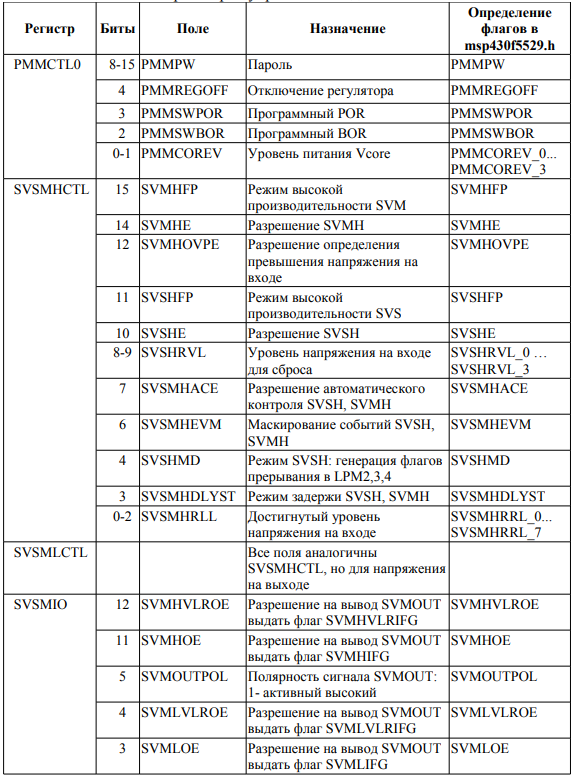
PMMCTL1 (0122h) - управление PMM;

SVSMHCTL (0124h) - управление SVS, SVM на входе; SVSMLCTL (0126h) - управление SVS, SVM на выходе; PMMIFG (012Ch) - флаги прерываний;

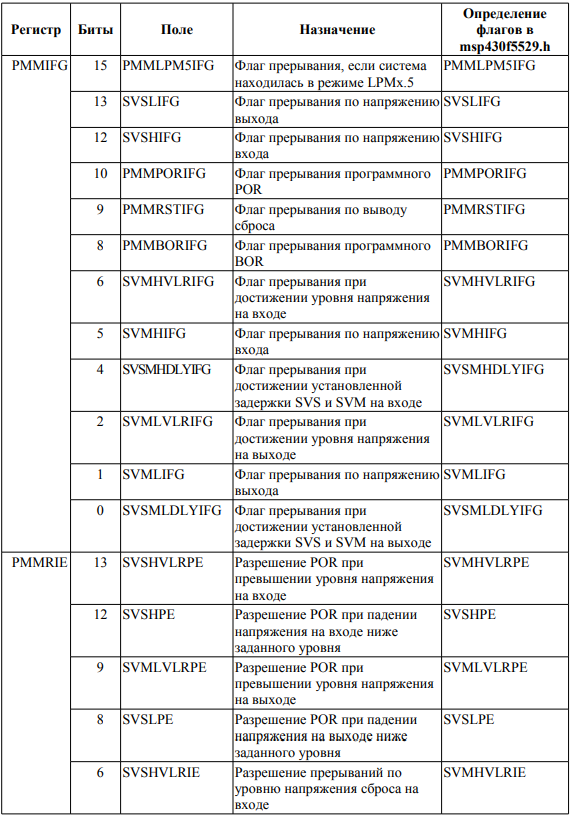
PMMIE (012Eh) - разрешение прерываний; PM5CTL0 (0130h) - управление режимом LPMx.5.

Основные поля регистров подсистемы питания приведены в таблице:

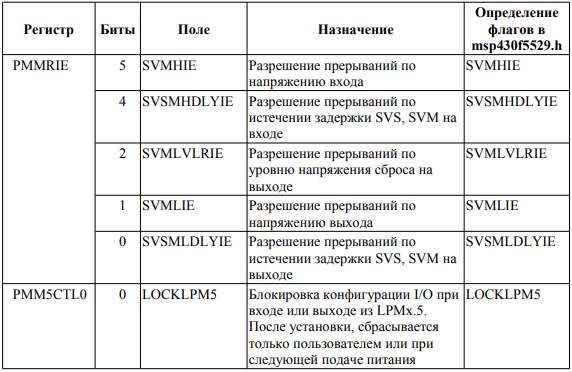
Таблица 3.4 Поля регистров управления подсистемой питания



Продолжение табл. 3.4



Окончание табл. 3.4



* 1. **Тактирование**

В микроконтроллере MSP430F5529 имеются следующие источники синхросигналов:

* внутренний генератор низкой частоты со сверхмалым потреблением (VLO), около 9,4 КГц;
* внутренний низкочастотный генератор (REFO), 32 Кгц;
* 32 КГц кварцевый генератор (XT1 LF);
* интегрированный внутренний цифровой управляемый генератор (DCO), стабилизируется с помощью цифровой автоподстройки частоты (FLL – frequency locked loop);
* высокочастотный кварцевый генератор (XT2) 4 – 32 МГц. На плате используется резонатор 4 МГц.

Три синхросигнала выбираются из этих источников:

* вспомогательная тактовая частота (ACLK);
* главная частота (MCLK), используется для тактирования процессора;
* SMCLK (Sub-Main), используется для тактирования периферии;
* буферный выход частоты ACLK/n (n = 1, 2, 4, 8, 16, 32).

Структура системы тактирования представлена на рисунке:

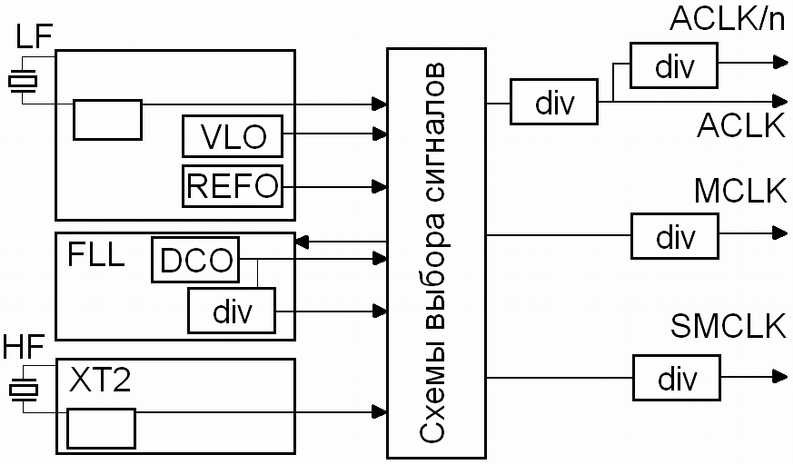


Рис. 3.6 Структура схемы тактирования

Генератор VLO активен, если он используется в качестве источника для любой из частот MCLK, SMCLK, ACLK. REFO — если он установлен как источник: для частоты ACLK в активном режиме, либо LPM0 – LPM3; частоты MCLK в активном режиме; частоты SMCLK в активном режиме и LPM0, LPM1; частоты FLLREFCLK и при этом DCO выбран источником для трех вышеперечисленных вариантов. ХТ1 и XT2 разрешаются для тех же 6 вариантов, что и REFO, а также в случаях, когда поля XT1OFF = 0 (запрещено отключение XT1) либо XT2OFF = 0 соответственно.

После сброса устанавливается следующий режим:

* + источником для ACLK выбирается XT1CLK;
  + источником для MCLK, SMCLK выбирается DCOCLKDIV;
  + разрешается автоподстройка, источником для FLL выбирается XT1CLK;
  + для F5529 выводы кварцевых генераторов разделяются с цифровыми I/O, поэтому после сброса эти выводы конфигурируются как цифровые, и сигналы от кварца недоступны.

При переключении MCLK или CLK с одного источника на другой, момент переключения синхронизируется для избегания возникновения гонок (см. рис. 3.7).

Генератор XT1 в MSP430F5529 поддерживает только низкочастотный (LF) режим. Ток потребления генератора в этом режиме регулируется полем XT1DRIVE. Частотный сигнал можно подавать непосредственно с резонатора (поле XT1BYPASS), в этом случае собственно генератор отключен. Кроме того, в этом случае необходимо установить диапазон выходного напряжения битом XT1BYPASSLV. Кроме того, поскольку резонатор внешний, а выводы разделяются с цифровыми, то необходимо установить нужный режим портов ввода-вывода (P5.4 – XIN, P5.5 – XOUT).

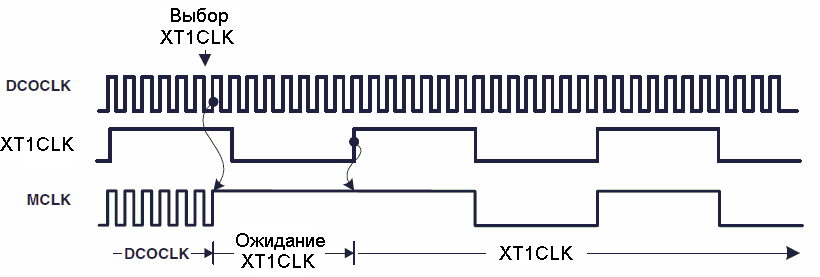


Рис. 3.7 Безопасное переключение частоты MCLK, SMCLK

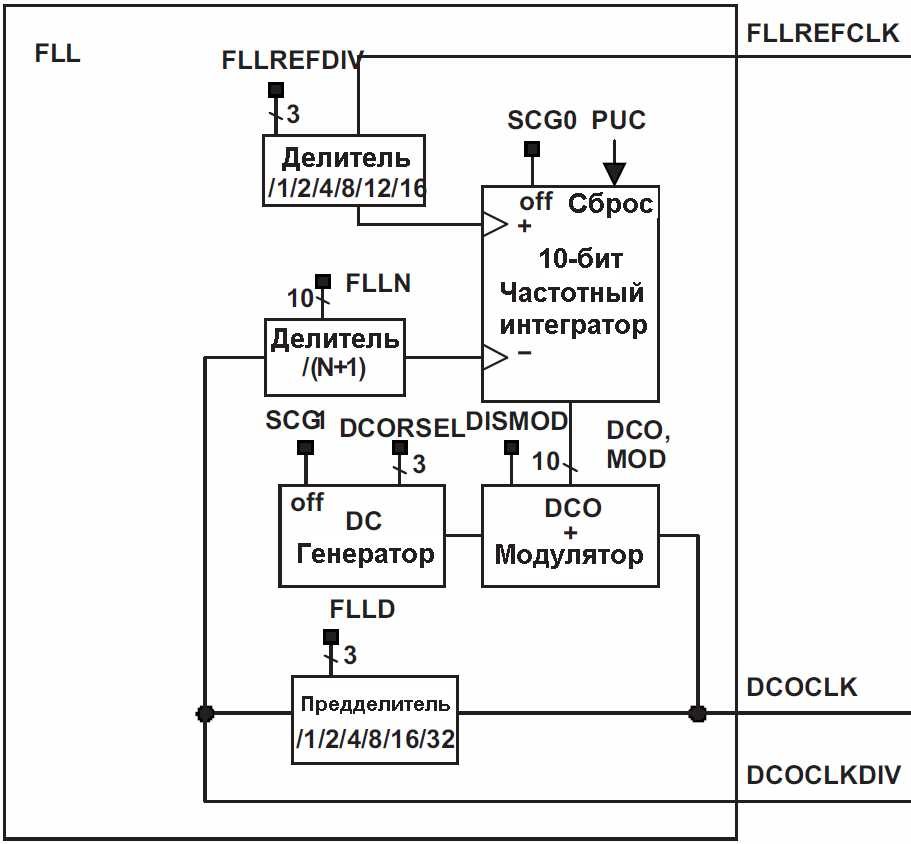


Рис. 3.8 Структура управляемого цифрового генератора

Управление высокочастотным генератором XT2 аналогично, за исключением того, что XT2DRIVE управляет частотным диапазоном. Резонатору соответствуют выводы P5.2 (XT2IN) и P5.3 (XT2OUT).

Генератор DCO имеет широкий диапазон перестраиваемых частот, его структура показана на рис. 3.8. Входная частота FLLREFCLK для FLL выбирается битами SELREF регистра UCSCTL3. Поле FLLREFDIV задает делитель (/n) этой частоты: 1,2,4,…,16. Поле DCORSEL (регистр UCSCTL1) задает один из 8 диапазонов частот. Поле FLLN (регистр UCSCTL2) задает множитель, определяющий 1 из 32 уровней N внутри диапазона. Поле FLLD (регистр UCSCTL2) задает делитель D = 1, 2, 4, …, 32 для частоты DCOCLKDIV на выходе.

На выходе блока генерируются следующие частоты:

DCOCLK = FLLREFCLK / n · ( N + 1 ) · D;

DCOCLKDIV = FLLREFCLK / n · ( N + 1 ).

Следует обратить внимание, что обязательно должен быть установлен соответствующий диапазон DCORSEL, иначе блок автоподстройки частоты установить нижнюю (либо верхнюю) границу диапазона, не достигнув заданной частоты. Для ускорения подстройки частоты используются поля MOD и DCO, устанавливающие начальную частоту блока автоподстройки. Время подстройки на один уровень DCO равно 32 · n тактов FLLREFCLK, и в худшем случае может составить 32 · 32 · n тактов FLLREFCLK. Дождаться стабилизации частоты можно, используя проверку флагов ошибок. Рабочие диапазоны частот DCO в зависимости от режима представлены на рисунке:

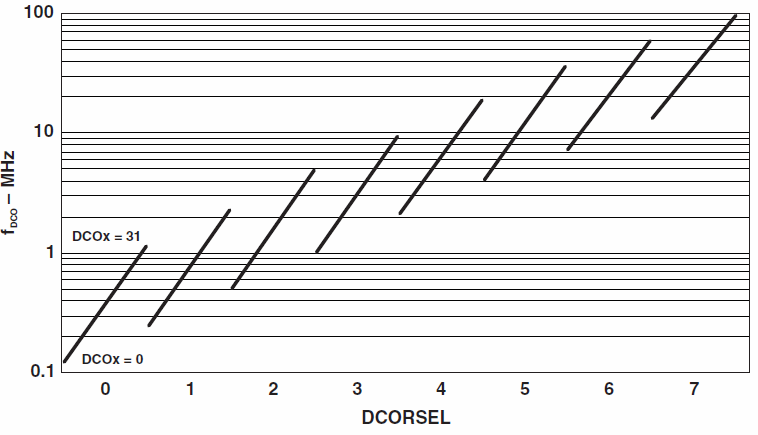
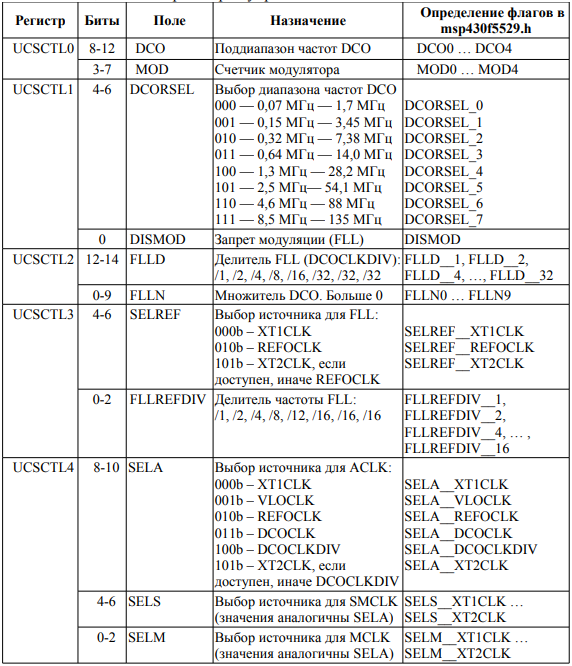


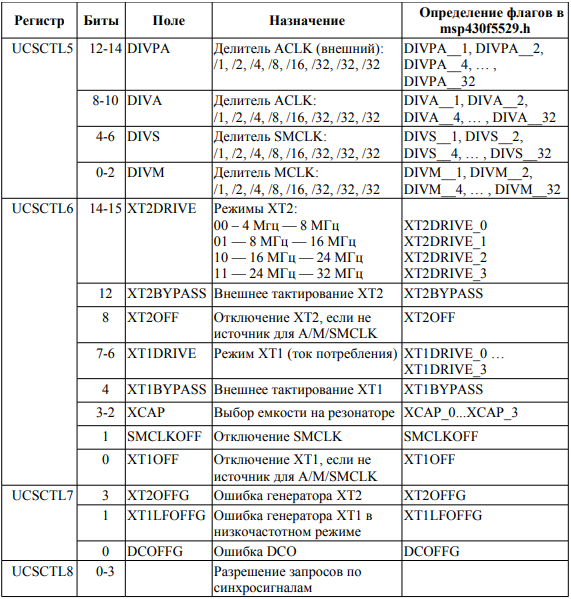
Рис. 3.9 Рабочие диапазоны частот DCO в зависимости от режима

Для управления блоком частот используются регистры UCSCTL0 – UCSCTL8 (адреса соответственно 0160h, 0162h, … , 0170h). Основные поля этих регистров представлены в таблице 3.5.

Таблица 3.5 Поля регистров управления блоком частот



Окончание табл. 3.5



**4 ВЫПОЛНЕНИЕ РАБОТЫ**

**4.1 Листинг кода**

#include <msp430.h>

/\*

\* main.c

\*/

int fl=0;

int fint=0, fint2=0;

int count=0, count2=0,count3=0, level=1;

int delay=0;

int frequency=0;

int sost;

volatile int i=0;

unsigned int level=3;

int main(void) {

WDTCTL = WDTPW | WDTHOLD; // Stop watchdog timer

P1DIR |= BIT0;

P1DIR |= BIT1;

P1DIR |= BIT2;

P1DIR |= BIT3;

P1DIR |= BIT5;

P8DIR |= BIT2;

P7DIR |= BIT7;

P7SEL |= BIT7;

//P8OUT &= ~BIT2;

//P1OUT &= ~BIT1;

P1REN |= BIT7;

P1OUT |= BIT7;

P2REN |= BIT2;

P2OUT |= BIT2;

P1IE|= BIT7;

P2IE|= BIT2;

P1IFG &= ~BIT7;

P2IFG &= ~BIT2;

P1IES |= BIT7;

P2IES |= BIT2;

//P8OUT &= ~BIT2;

//P1OUT |= BIT2;

P1OUT &= ~BIT2;

P1OUT |= BIT0;

P1OUT |= BIT1;

P1IFG=0;

\_\_bis\_SR\_register(GIE);

TA1CTL = TASSEL\_1 + MC\_1 + ID\_1 + TACLR;

TA1CCR0 = 5000;

TA1CCTL0 = CCIE;

//UCSCTL3 |= SELREF\_\_REFOCLK; //Выбор источника для FLL: 010b – REFOCLK

//UCSCTL4 |= SELM\_\_DCOCLK; //Выбор источника для MCLK DCOCLK

UCSCTL1 |= DCORSEL\_1; //источник частоты

UCSCTL2 |= FLLD\_\_4; //делитель частоты D

UCSCTL2 |= FLLN6; // N множитель DCO

// UCSCTL3 |=FLLREFDIV\_\_2; //делитель частоты n

// UCSCTL3 |= DIVM\_\_4; // DIVM\_\_32

\_\_no\_operation();

return 0;

}

#pragma vector=TIMER1\_A0\_VECTOR

\_\_interrupt void TIMER1\_A0\_ISR(void)

{

{

P1OUT ^= BIT5; //если выждали задержку зажигаем светодиод

count++;

if(count>=count2+5){

fint=0;

//TA1CTL = TACLR;

//TA1CCTL0 &=~ CCIE;

}

if(count>=count3+5){

fint2=0;

//TA1CTL = TACLR;

//TA1CCTL0 &=~ CCIE;

}

}

}

#pragma vector=PORT1\_VECTOR

\_\_interrupt void P1INT()

{

if(fint==0){

count2=count;

fint=1;

if(fl==0)

{P1OUT &= ~BIT0;

P8OUT |= BIT2;

fl=1;

\_BIS\_SR(LPM2\_bits+GIE); //перейти в режим энергосбережения LPM2

}

else if(fl==1){

P1OUT |= BIT0;

P8OUT &= ~BIT2;

fl=0;

\_BIC\_SR\_IRQ(LPM2\_bits); //выйти из режима энергосбережения LPM2

}

}

P1IFG=0;

}

#pragma vector=PORT2\_VECTOR

\_\_interrupt void P2INT() // Обработчик прерывания уменьшает делитель частоты

{

if(fint2==0){ //проверка задержки для предотвращения дребезга

count3=count;

fint2=1;

if(frequency==0)

{P1OUT &= ~BIT1;

P1OUT |= BIT2; //включить светодтод сигнализирующий о высоком уровне напряжения

//UCSCTL4=0;

level=2;

PMMCTL0\_H = PMMPW\_H; // открыть PMM регистры для доступа

SVSMHCTL = SVSHE | SVSHRVL0 \* level | SVMHE | SVSMHRRL0 \* level; // управление SVM/SVS на входе

SVSMLCTL = SVSLE | SVMLE | SVSMLRRL0 \* level; // установка SVM в новый уровень

while (!(PMMIFG & SVSMLDLYIFG)); // ожидание пока установится SVM

PMMIFG &= ~(SVMLVLRIFG + SVMLIFG); // очистка ранее установленных флагов

PMMCTL0\_L = PMMCOREV0 \* level; // установка Vcore в новый уровень

if ((PMMIFG & SVMLIFG)) // ожидание пока будет достигнут новый уровень

while (!(PMMIFG & SVMLVLRIFG));

SVSMLCTL = SVSLE | SVSLRVL0 \* level | SVMLE | SVSMLRRL0 \* level; // управление SVM/SVS на выходе

PMMCTL0\_H = 0; // закрыть PMM регистры для доступа

level=3;

PMMCTL0\_H = PMMPW\_H; // открыть PMM регистры для доступа

SVSMHCTL = SVSHE | SVSHRVL0 \* level | SVMHE | SVSMHRRL0 \* level; // управление SVM/SVS на входе

SVSMLCTL = SVSLE | SVMLE | SVSMLRRL0 \* level; // установка SVM в новый уровень

while (!(PMMIFG & SVSMLDLYIFG)); // ожидание пока установится SVM

PMMIFG &= ~(SVMLVLRIFG + SVMLIFG); // очистка ранее установленных флагов

PMMCTL0\_L = PMMCOREV0 \* level; // установка Vcore в новый уровень

if ((PMMIFG & SVMLIFG)) // ожидание пока будет достигнут новый уровень

while (!(PMMIFG & SVMLVLRIFG));

SVSMLCTL = SVSLE | SVSLRVL0 \* level | SVMLE | SVSMLRRL0 \* level; // управление SVM/SVS на выходе

PMMCTL0\_H = 0; // закрыть PMM регистры для доступа

UCSCTL4 &= ~SELS\_\_DCOCLK; //убрать источник DCO

UCSCTL4 |= SELS\_\_REFOCLK; //установить источник REFO

frequency=1;

}

else if(frequency==1){

//PMMCOREV1

P1OUT |= BIT1;

P1OUT &= ~BIT2;

//UCSCTL4=0;

level=2;

PMMCTL0\_H = PMMPW\_H; // открыть PMM регистры для доступа

SVSMLCTL = SVSLE | SVSLRVL0 \* level | SVMLE | SVSMLRRL0 \* level; // управление SVM/SVS на входе

while (!(PMMIFG & SVSMLDLYIFG)); // ожидание пока установится SVM

PMMIFG &= ~(SVMLVLRIFG + SVMLIFG); // очистка ранее установленных флагов

PMMCTL0\_L = PMMCOREV0 \* level; // установка Vcore в новый уровень

PMMCTL0\_H = 0; // открыть PMM регистры для доступа

level=1;

PMMCTL0\_H = PMMPW\_H; // открыть PMM регистры для доступа

SVSMLCTL = SVSLE | SVSLRVL0 \* level | SVMLE | SVSMLRRL0 \* level; // управление SVM/SVS на входе

while (!(PMMIFG & SVSMLDLYIFG)); // ожидание пока установится SVM

PMMIFG &= ~(SVMLVLRIFG + SVMLIFG); // очистка ранее установленных флагов

PMMCTL0\_L = PMMCOREV0 \* level; // установка Vcore в новый уровень

PMMCTL0\_H = 0; // открыть PMM регистры для доступа

UCSCTL4 &= ~SELS\_\_REFOCLK;

UCSCTL4 |= SELS\_\_DCOCLK;

/\*UCSCTL1 |= DCORSEL\_1; //источник частоты

UCSCTL2 |= FLLD\_\_16; //делитель частоты D

UCSCTL2 |= FLLN6; // N множитель DCO

UCSCTL5 = DIVS\_\_8;\*/

frequency=0;

}

}

P2IFG=0;

}

**5 ВЫВОДЫ**

В ходе выполнения лабораторной работы были изучены принципы построения систем с низким энергопотреблением на базе микроконтроллера MSP430F5529.