Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Рязанский государственный радиотехнический

университет имени В.Ф. Уткина»

Кафедра «ЭВМ»

Отчет о практической работе №3

«Модуль управления (контроллер) тактовых частот RST\_CLK. Настройка тактирования процессора. Включение периферийных устройств»

по дисциплине

«Специализированные ЭВМ»

Выполнили:

Студенты группы 045

Вашкулатов Н.А.

Анохин В.А.

Проверил:

доц. каф. ЭВМ Устюков Д.И.

**Цель работы**: познакомиться с общими принципами тактирования узлов в микропроцессорных системах.

**Ход работы**

**Задание 3.1.** Переключение на внешний генератор HSE. Создайте проект согласно описанному в первой работе алгоритму. Добавьте следующий программный код в файл main.c.

**Код программы:**

#include "MDR32F9Qx\_port.h" //подключение библиотеки работы с портами

#include "MDR32F9Qx\_rst\_clk.h" //подключение библиотеки контроллера

static PORT\_InitTypeDef PortInit; //стуктура инициализации порта

void Delay(int del) //процедура задржки del - кол-во циклов

{

for (int i=0; i<del; i++); //1 цикл – n тактов процессора

}

void LedConfig (void) //Процедура инициализации

{

PortInit.PORT\_Pin = PORT\_Pin\_0; //вывод 0

PortInit.PORT\_FUNC = PORT\_FUNC\_PORT; //режим порт

PortInit.PORT\_MODE = PORT\_MODE\_DIGITAL; //цифровой

PortInit.PORT\_OE = PORT\_OE\_OUT; //сигнал на вывод

PortInit.PORT\_SPEED = PORT\_SPEED\_SLOW; //медленный фронт

PORT\_Init(MDR\_PORTC, &PortInit); //инициализация параметров

}

void CPUCLKConfig(void) //настройка тактирования ЦПУ

{

RST\_CLK\_HSEconfig(RST\_CLK\_HSE\_ON); //включение внешнего

//высокочастотного генератора

while (RST\_CLK\_HSEstatus()!=SUCCESS); //запрос состояния,

RST\_CLK\_CPU\_PLLconfig(RST\_CLK\_CPU\_PLLsrcHSEdiv2,RST\_CLK\_CPU\_PLLmul1);

//только если готов, то переход к следующим действиям

RST\_CLK\_CPUclkSelection(RST\_CLK\_CPUclkCPU\_C3);//переключение на CPU\_C3

}

int main (void) //основная процедура

{

RST\_CLK\_PCLKcmd(RST\_CLK\_PCLK\_PORTC, ENABLE);//разрешение тактирования PORTC

LedConfig(); //инициализация вывода светодиода

for(int i = 0; i < 5; i++){

Delay(1000000); //задержка

PORT\_SetBits(MDR\_PORTC, PORT\_Pin\_0); //включение светодиода

Delay(1000000); //задержка

PORT\_ResetBits(MDR\_PORTC, PORT\_Pin\_0);

}

CPUCLKConfig(); //переключение тактового генератора

while(1) { //бесконечный цикл

Delay(1000000); //задержка

PORT\_SetBits(MDR\_PORTC, PORT\_Pin\_0); //включение светодиода

Delay(1000000); //задержка

PORT\_ResetBits(MDR\_PORTC, PORT\_Pin\_0); //выключение светодиода

}

}

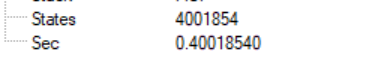


Рисунок 1 – Время выполнения до включения hse



Рисунок 2 – Время выполнения до включения hse после задержки

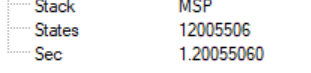


Рисунок 3 – После hse



Рисунок 4 – Уменьшение частоты в 2 раза

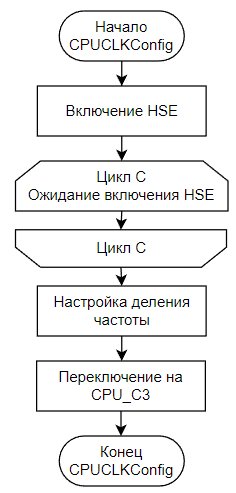
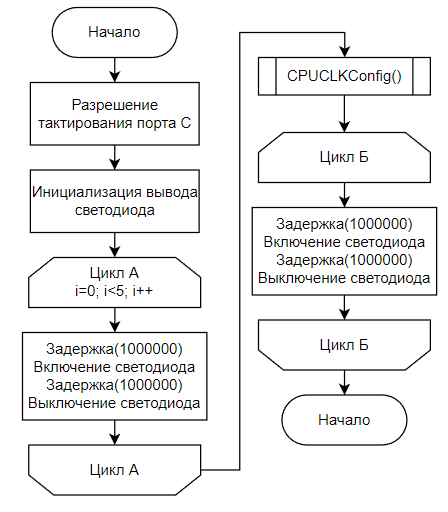


Рисунок 5 – Схема алгоритма

**Задание 3.2.** Анализ изменения частоты работы с использованием анализатора. Измените программу из задания 3.1 таким образом, чтобы на выводы порта A выводилось значение от счётчика (аналогично заданию 1.3). Измените частоту и источник тактового сигнала в соответствии с вариантом задания: источник: HSI, значение делителя: /8.

**Код программы:**

#include "MDR32F9Qx\_port.h" //подключение библиотеки работы с портами

#include "MDR32F9Qx\_rst\_clk.h" //подключение библиотеки контроллера тактовых сигналов

static PORT\_InitTypeDef PortInit; //стуктура инициализации порта

static RST\_CLK\_FreqTypeDef freqStats;

void portSetUp(){

PortInit.PORT\_Pin = PORT\_Pin\_All;

PortInit.PORT\_OE = PORT\_OE\_OUT;

PortInit.PORT\_FUNC = PORT\_FUNC\_PORT;//порт в режиме стандартной функции

PortInit.PORT\_MODE = PORT\_MODE\_DIGITAL;//цифровой режим порта

PortInit.PORT\_SPEED = PORT\_SPEED\_SLOW;//скорость перехода сигнала от

PORT\_Init(MDR\_PORTA, &PortInit);

}

void CPUCLKConfig(void) //настройка тактирования ЦПУ

{

RST\_CLK\_CPUclkPrescaler(RST\_CLK\_CPUclkDIV8);

RST\_CLK\_CPUclkSelection(RST\_CLK\_CPUclkCPU\_C3);

}

void countToPortA(int repeats){

uint32\_t count = 0;

for(int i = 0; i < repeats; i++){

while(count <= 255){

PORT\_Write(MDR\_PORTA, count++);

}

count = 0;

}

}

int main (void) //основная процедура

{

RST\_CLK\_PCLKcmd(RST\_CLK\_PCLK\_PORTA, ENABLE);//разрешение тактирования PORTA

portSetUp(); //инициализация вывода

countToPortA(5);

CPUCLKConfig(); //переключение тактового генератора

countToPortA(5);

uint32\_t count = 0;

while(1){

PORT\_Write(MDR\_PORTA, count++);

count = 0;

}

}

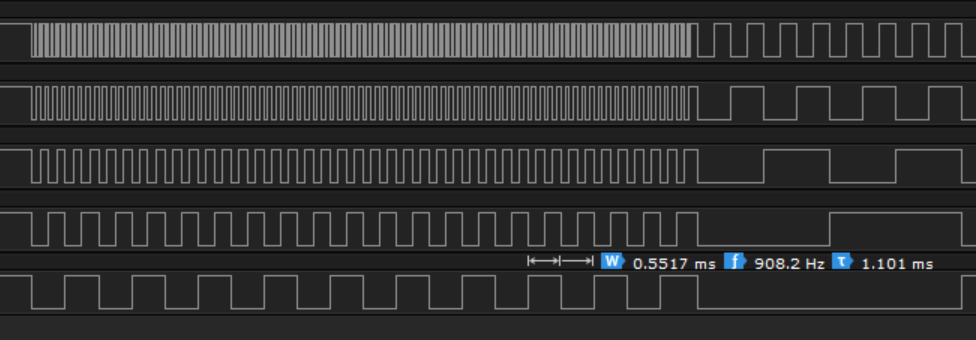


Рисунок 6 – До деления на 8

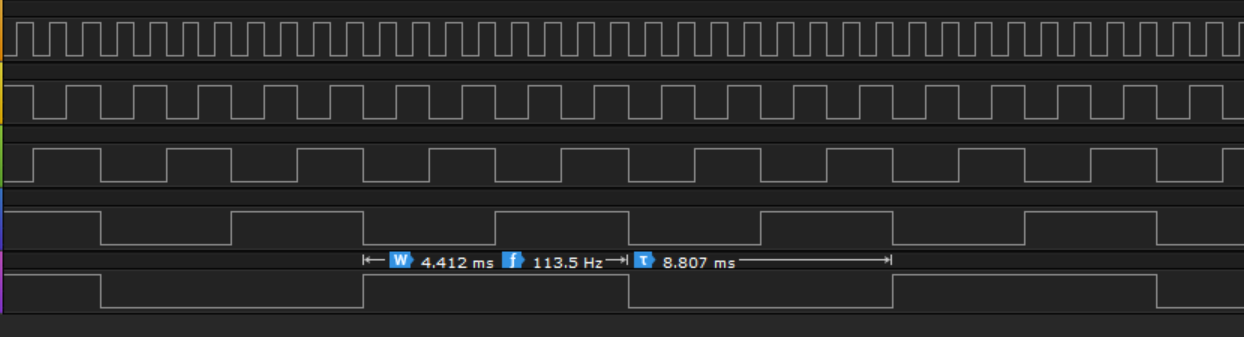


Рисунок 7 – После деления на 8

**Задание 3.3.** Подключение модуля CPU PLL. Измените процедуру CPUCLKConfig() согласно примеру. Изобразите схему алгоритма процедуры main(), включая действия процедуры CPUCLKConfig. Выполните сборку программы. Загрузите программу в контроллер. Проанализируйте изменение работы системы.

**Код программы:**

#include "MDR32F9Qx\_port.h" //подключение библиотеки работы с портами

#include "MDR32F9Qx\_rst\_clk.h" //подключение библиотеки контроллера тактовых сигналов

static PORT\_InitTypeDef PortInit; //стуктура инициализации порта

static RST\_CLK\_FreqTypeDef freqStats;

void Delay(int del) //процедура задржки del - кол-во циклов

{

for (int i=0; i<del; i++); //1 цикл – n тактов процессора

}

void LedConfig (void) //Процедура инициализации

{

PortInit.PORT\_Pin = PORT\_Pin\_0; //вывод 0

PortInit.PORT\_FUNC = PORT\_FUNC\_PORT; //режим порт

PortInit.PORT\_MODE = PORT\_MODE\_DIGITAL; //цифровой

PortInit.PORT\_OE = PORT\_OE\_OUT; //сигнал на вывод

PortInit.PORT\_SPEED = PORT\_SPEED\_SLOW; //медленный фронт

PORT\_Init(MDR\_PORTC, &PortInit); //инициализация параметров

}

void CPUCLKConfig(void) //настройка тактирования ЦПУ

{

RST\_CLK\_HSEconfig(RST\_CLK\_HSE\_ON);

//запрос состояния, только если готов, то переход к следующим действиям

while (RST\_CLK\_HSEstatus()!=SUCCESS);

RST\_CLK\_CPU\_PLLconfig(RST\_CLK\_CPU\_PLLsrcHSEdiv1, RST\_CLK\_CPU\_PLLmul8);

RST\_CLK\_CPU\_PLLcmd(ENABLE);

while (RST\_CLK\_CPU\_PLLstatus()!=SUCCESS);

RST\_CLK\_CPU\_PLLuse(ENABLE);

RST\_CLK\_CPUclkSelection(RST\_CLK\_CPUclkCPU\_C3);

}

int main (void) //основная процедура

{

RST\_CLK\_PCLKcmd(RST\_CLK\_PCLK\_PORTC, ENABLE);//разрешение тактирования PORTC

LedConfig(); //инициализация вывода светодиода

for(int i = 0; i < 5; i++){

Delay(1000000); //задержка

PORT\_SetBits(MDR\_PORTC, PORT\_Pin\_0); //включение светодиода

Delay(1000000); //задержка

PORT\_ResetBits(MDR\_PORTC, PORT\_Pin\_0);

}

CPUCLKConfig(); //переключение тактового генератора

while(1) //бесконечный цикл

{

Delay(1000000); //задержка

PORT\_SetBits(MDR\_PORTC, PORT\_Pin\_0); //включение светодиода

Delay(1000000); //задержка

PORT\_ResetBits(MDR\_PORTC, PORT\_Pin\_0); //выключение светодиода

}

}

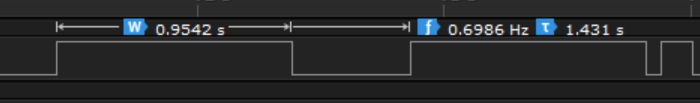


Рисунок 8 – До использования PLL

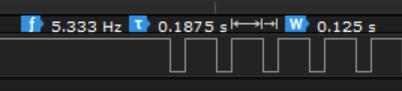


Рисунок 9 – После использования PLL

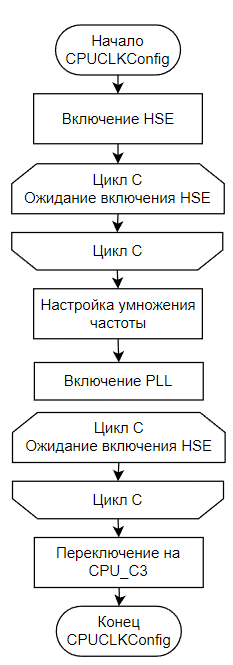
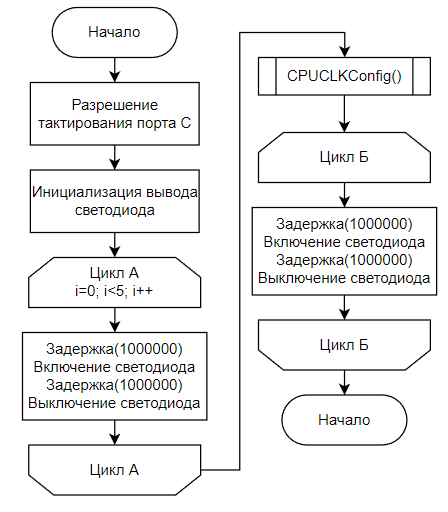
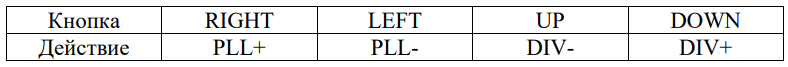


Рисунок 10 – Схема алгоритма

**Задание 3.4.** Управление частотой. Измените программу из задания 3.3 таким образом, чтобы можно было изменять частоту процессора нажатием на кнопки (см. таблицу 1). Выполните сборку программы. Загрузите программу в микроконтроллер. Проанализируйте работу системы.

Таблица 1 – Действия для кнопок



При нажатии на кнопку right скорость мигания светодиода увеличивается, а при нажатии на left – уменьшается. При нажатии на кнопку down скорость мигания светодиода уменьшается в 2 раза, а при на up – возвращается в норму.

**Код программы:**

#include "MDR32F9Qx\_port.h" //подключение библиотеки работы с портами

#include "MDR32F9Qx\_rst\_clk.h" //подключение библиотеки контроллера тактовых сигналов

static PORT\_InitTypeDef PortInit; //стуктура инициализации порта

static uint32\_t PLL = RST\_CLK\_CPU\_PLLmul9;

static uint32\_t DIV = RST\_CLK\_CPU\_PLLsrcHSEdiv1;

struct ButtonStatus {

int UP;

int DOWN;

int LEFT;

int RIGHT;

};

static struct ButtonStatus pressedButtons = {0,0,0,0};

void updatePLLConfig(void){

RST\_CLK\_CPU\_PLLuse(DISABLE);

RST\_CLK\_CPU\_PLLcmd(DISABLE);

RST\_CLK\_CPU\_PLLconfig(DIV, PLL);

RST\_CLK\_CPU\_PLLcmd(ENABLE);

while (RST\_CLK\_CPU\_PLLstatus()!=SUCCESS);

RST\_CLK\_CPU\_PLLuse(ENABLE);

}

void pressRight(void){

if(PLL < RST\_CLK\_CPU\_PLLmul9)

PLL++;

updatePLLConfig();

}

void pressLeft(void){

if(PLL > RST\_CLK\_CPU\_PLLmul1)

PLL--;

updatePLLConfig();

}

void pressUp(void){

if(DIV < RST\_CLK\_CPU\_PLLsrcHSEdiv2)

DIV++;

updatePLLConfig();

}

void pressDown(void){

if(DIV > RST\_CLK\_CPU\_PLLsrcHSIdiv1)

DIV--;

updatePLLConfig();

}

void Delay(int del) //процедура задржки del - кол-во циклов

{

for (int i=0; i<del; i++); //1 цикл – n тактов процессора

}

void LedConfig (void) //Процедура инициализации

{

PortInit.PORT\_Pin = PORT\_Pin\_0; //вывод 0

PortInit.PORT\_FUNC = PORT\_FUNC\_PORT; //режим порт

PortInit.PORT\_MODE = PORT\_MODE\_DIGITAL; //цифровой

PortInit.PORT\_OE = PORT\_OE\_OUT; //сигнал на вывод

PortInit.PORT\_SPEED = PORT\_SPEED\_SLOW; //медленный фронт

PORT\_Init(MDR\_PORTC, &PortInit); //инициализация параметров

}

void CPUCLKConfig(void) //настройка тактирования ЦПУ

{

RST\_CLK\_HSEconfig(RST\_CLK\_HSE\_ON);

//запрос состояния, только если готов, то переход к следующим действиям

while (RST\_CLK\_HSEstatus()!=SUCCESS);

RST\_CLK\_CPU\_PLLconfig(DIV, PLL);

RST\_CLK\_CPU\_PLLcmd(ENABLE);

while (RST\_CLK\_CPU\_PLLstatus()!=SUCCESS);

RST\_CLK\_CPU\_PLLuse(ENABLE);

RST\_CLK\_CPUclkSelection(RST\_CLK\_CPUclkCPU\_C3);

}

void ButtonsPinCfg(void){

PORT\_StructInit(&PortInit);

PortInit.PORT\_Pin = (PORT\_Pin\_5 | PORT\_Pin\_6);

PortInit.PORT\_OE = PORT\_OE\_IN;

PortInit.PORT\_FUNC = PORT\_FUNC\_PORT;//порт в режиме стандартной функции

PortInit.PORT\_MODE = PORT\_MODE\_DIGITAL;//цифровой режим порта

PortInit.PORT\_SPEED = PORT\_SPEED\_SLOW;//скорость перехода сигнала от

PortInit.PORT\_PULL\_DOWN = PORT\_PULL\_DOWN\_OFF;

PortInit.PORT\_PULL\_UP = PORT\_PULL\_UP\_ON;

PortInit.PORT\_GFEN = PORT\_GFEN\_ON;

PORT\_Init(MDR\_PORTB, &PortInit);

PortInit.PORT\_Pin = (PORT\_Pin\_1 | PORT\_Pin\_3);

PORT\_Init(MDR\_PORTE, &PortInit);

}

int isLeftPressed(){

return PORT\_ReadInputDataBit(MDR\_PORTE, PORT\_Pin\_3);

}

int isRightPressed(){

return PORT\_ReadInputDataBit(MDR\_PORTB, PORT\_Pin\_6);

}

int isUpPressed(){

return PORT\_ReadInputDataBit(MDR\_PORTB, PORT\_Pin\_5);

}

int isDownPressed(){

return PORT\_ReadInputDataBit(MDR\_PORTE, PORT\_Pin\_1);

}

int main (void) //основная процедура

{

RST\_CLK\_PCLKcmd(RST\_CLK\_PCLK\_PORTC, ENABLE);//разрешение тактирования PORTC

RST\_CLK\_PCLKcmd(RST\_CLK\_PCLK\_PORTB, ENABLE);

RST\_CLK\_PCLKcmd(RST\_CLK\_PCLK\_PORTE, ENABLE);

LedConfig(); //инициализация вывода

ButtonsPinCfg();

CPUCLKConfig(); //переключение тактового генератора

uint32\_t count = 0;

while(1){

count++;

if(count < 25000){

PORT\_SetBits(MDR\_PORTC, PORT\_Pin\_0);

} else {

if(count < 50000){

PORT\_ResetBits(MDR\_PORTC, PORT\_Pin\_0);

} else {

count = 0;

}

}

if(isLeftPressed()){

if(pressedButtons.LEFT == 0){

pressedButtons.LEFT = 1;

pressLeft();

}

} else {

pressedButtons.LEFT = 0;

}

if(isRightPressed()){

if(pressedButtons.RIGHT == 0){

pressRight();

pressedButtons.RIGHT = 1;

}

} else {

pressedButtons.RIGHT = 0;

}

if(isDownPressed()){

if(pressedButtons.DOWN == 0){

pressDown();

pressedButtons.DOWN = 1;

}

} else {

pressedButtons.DOWN = 0;

}

if(isUpPressed()){

if(pressedButtons.UP == 0){

pressUp();

pressedButtons.UP = 1;

}

} else {

pressedButtons.UP = 0;

}

}

}

**Задание 3.5.** Настройка модуля RST\_CLK прямым обращением к регистрам. Перепишите подпрограмму инициализации из задания 3.3, не используя библиотечные функции. Все операции с контроллером тактовых частот выполнить прямым обращением к регистрам.

**Код программы:**

#include "MDR32F9Qx\_port.h" //подключение библиотеки работы с портами

#include "MDR32F9Qx\_rst\_clk.h" //подключение библиотеки контроллера тактовых сигналов

static PORT\_InitTypeDef PortInit; //стуктура инициализации порта

static RST\_CLK\_FreqTypeDef freqStats;

void Delay(int del) //процедура задржки del - кол-во циклов

{

for (int i=0; i<del; i++); //1 цикл – n тактов процессора

}

void LedConfig (void) //Процедура инициализации

{

PortInit.PORT\_Pin = PORT\_Pin\_0; //вывод 0

PortInit.PORT\_FUNC = PORT\_FUNC\_PORT; //режим порт

PortInit.PORT\_MODE = PORT\_MODE\_DIGITAL; //цифровой

PortInit.PORT\_OE = PORT\_OE\_OUT; //сигнал на вывод

PortInit.PORT\_SPEED = PORT\_SPEED\_SLOW; //медленный фронт

PORT\_Init(MDR\_PORTC, &PortInit); //инициализация параметров

}

void CPUCLKConfig(void) //настройка тактирования ЦПУ

{

//RST\_CLK\_HSEconfig(RST\_CLK\_HSE\_ON);

MDR\_RST\_CLK->HS\_CONTROL |= 1;

//запрос состояния, только если готов, то переход к следующим действиям

//while (RST\_CLK\_HSEstatus()!=SUCCESS);

while(MDR\_RST\_CLK->CLOCK\_STATUS & 0b100);

//RST\_CLK\_CPU\_PLLconfig(DIV, PLL);

MDR\_RST\_CLK->PLL\_CONTROL |= 0x400;

MDR\_RST\_CLK->CPU\_CLOCK |= 0b10

//RST\_CLK\_CPU\_PLLcmd(ENABLE);

MDR\_RST\_CLK->PLL\_CONTROL |= 0b100;

//while (RST\_CLK\_CPU\_PLLstatus()!=SUCCESS);

while(MDR\_RST\_CLK->CLOCK\_STATUS & 0b010);

//RST\_CLK\_CPU\_PLLuse(ENABLE);

MDR\_RST\_CLK->CPU\_CLOCK |= 0b100

//RST\_CLK\_CPUclkSelection(RST\_CLK\_CPUclkCPU\_C3);

MDR\_RST\_CLK->CPU\_CLOCK |=0x100

}

int main (void) //основная процедура

{

RST\_CLK\_PCLKcmd(RST\_CLK\_PCLK\_PORTC, ENABLE);//разрешение тактирования PORTC

LedConfig(); //инициализация вывода светодиода

for(int i = 0; i < 5; i++){

Delay(1000000); //задержка

PORT\_SetBits(MDR\_PORTC, PORT\_Pin\_0); //включение светодиода

Delay(1000000); //задержка

PORT\_ResetBits(MDR\_PORTC, PORT\_Pin\_0);

}

CPUCLKConfig(); //переключение тактового генератора

while(1) //бесконечный цикл

{

Delay(1000000); //задержка

PORT\_SetBits(MDR\_PORTC, PORT\_Pin\_0); //включение светодиода

Delay(1000000); //задержка

PORT\_ResetBits(MDR\_PORTC, PORT\_Pin\_0); //выключение светодиода

}

}

**Вывод**: в ходе выполнения работы были ознакомлены с общими принципами тактирования узлов в микропроцессорных системах.