**Отчет по ЛР3**

**«Применение микроконтроллеров для создания сложных систем»**

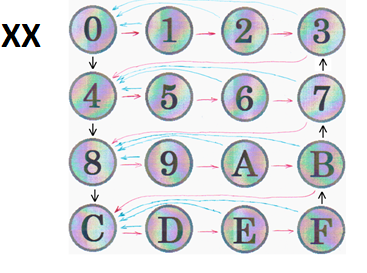
Вариант **ХХ**

Исполнитель: Преподаватель:

ИУ5-7Хб

Москва, МГТУ – 2020

1.Граф переходов системы:



1. Схема алгоритма работы системы



1. Программа для микроконтроллера ATmega 16, реализующая разработанный алгоритм

;

; LR3-AS.asm

;

; Created: 02.10.2020 12:12:33

; Author : Knight

;

.include "m16def.inc" ; подключение библиотеки для работы с ATmega16

.list ; включение листинга

.def temp=r16 ; определение главного рабочего регистра

.def kold=r17

.def k\_\_\_=r18

.def s\_\_\_=r19

;--------------------------------------------

.cseg ; выбор сегмента программного кода

.org 0 ; установка текущего адреса на ноль

;--------------------------------------------

ldi temp,0x80 ; выключение компаратора

out acsr,temp

;--------------------------------------------

ldi temp,0x00 ; 0 --> temp

out ddrd,temp ; Назначаем порт rd на ввод (00000000 --> ddrd)

ldi temp,0xFF ; 0xff --> temp

out ddrb,temp ; Назначаем порт rb на вывод (11111111 --> ddrb)

out portd,temp ; Подключаем подтягивающие резисторы (11111111 --> portd)

;---------------------------------------------

ldi kold, 0x00 ; 0--->kold

ldi s\_\_\_, 0x00 ; 0--->s\_\_\_

out portb, s\_\_\_ ; Вывели s\_\_\_(=0)

;---------------------------------------------

read: ;

in k\_\_\_, pind ; Считали содержимое порта pd (--->k\_\_\_)

cp k\_\_\_, kold ; Сравнили k\_\_\_ и kold

breq read ; Если k\_\_\_=kold, read

tst kold ; Проверили kold

brne remem ; Если kold!=0, remen

tst k\_\_\_ ; Проверили k\_\_\_

breq remem ; Если k\_\_\_=0, remem

jmp lbl1 ;

;---------------------------------------------

remem: ;

mov kold, k\_\_\_ ; Записали k\_\_\_ в kold

jmp read ; Вернулись в read

;---------------------------------------------

lbl1: ;

cpi k\_\_\_, 0x01 ; Сравнили k\_\_\_ и 1

brne lbl2 ; Если k\_\_\_!=1, lbl2

cpi s\_\_\_, 0x0F ; Сравнили s\_\_\_ и 15

brne lbl\_1 ; Если s\_\_\_!=15, lbl\_1

jmp print ; Перешли на Вывод

lbl\_1: ;

inc s\_\_\_ ; s\_\_\_+1--->s\_\_\_

jmp print ; Перешли на Вывод

;---------------------------------------------

lbl2: ;

cpi k\_\_\_, 0x02 ; Сравнили k\_\_\_ и 2

brne lbl4 ; Если k\_\_\_!=2, lbl4

MOV temp, s\_\_\_ ; Перепишем состояние в буферный регистр

ANDI temp, 0x03 ; Получим остаток от деления на 4

SUB s\_\_\_, temp ; Вычтем остаток из текущего состояния

jmp print ; Перешли на Вывод

;---------------------------------------------

lbl4: ;

cpi k\_\_\_, 0x04 ; Сравнили k\_\_\_ и 4

brne remem ; Если k\_\_\_!=4, remem

MOV temp, s\_\_\_ ; Перепишем состояние в буферный регистр

ANDI temp, 0x03 ; Получим остаток от деления на 4

cpi temp, 0x00 ; Сравнили остаток и 0

brne lbl\_41 ; Если остаток!=0, lbl\_41

cpi s\_\_\_, 0x0C ; Сравнили состояние и 12

BREQ lbl\_42 ; Если состояние=12, lbl\_42

inc s\_\_\_ ; ...

inc s\_\_\_ ; ...

inc s\_\_\_ ; ...

inc s\_\_\_ ; s\_\_\_+4--->s\_\_\_

;SUBI s\_\_\_, 0$FB ; Возможно, это будет работать вместо 4 инкрементов

jmp print ; Перешли на Вывод

lbl\_41: ;

cpi temp, 0x03 ; Сравнили остаток и 3

brne lbl\_42 ; Если остаток!=3, lbl\_42

cpi s\_\_\_, 0x03 ; Сравнили состояние и 3

BREQ lbl\_42 ; Если состояние=3, lbl\_42

SUBI s\_\_\_, 0x04 ; Уменьшение состояния на 4

jmp print ; Перешли на Вывод

lbl\_42: ;

jmp print ; Перешли на Вывод

;---------------------------------------------

print: ;

out portb, s\_\_\_ ; Вывели s\_\_\_ в порт pb

jmp remem

start:

inc r16

rjmp start

1. Модель Системы в Proteus

