Домашнее задание N2 по курсу "Цифровые устройства и микропроцессоры".
Разработать цифровое устройство на основе любого микропроцессора, реализующее счетчик, с заданной последовательностью состояний: 7 2 4 3 6 1 0 за счет функций, минимизированных в домашнем задании N1 (Включая принципиальную схему и управляющую программу в кодах выбранного микропроцессора).

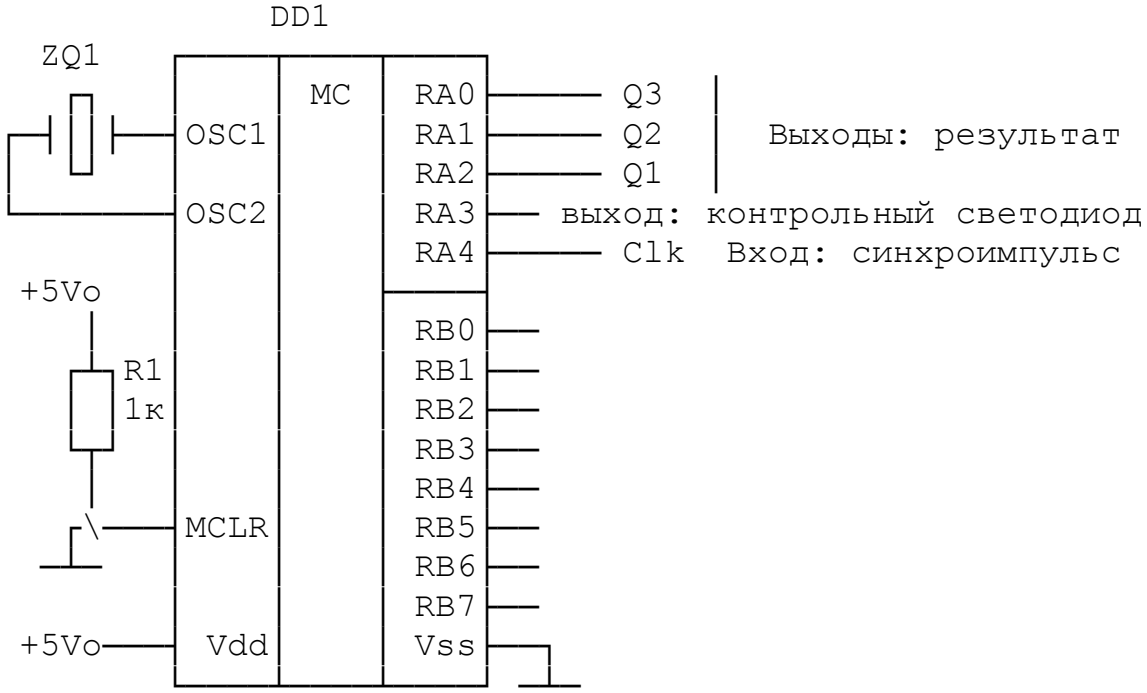
ПРИМЕР №1: Заданы функции D1, D2 и D3 трех переменных Q1, Q2, Q3:

$D1=\overline{Q1}\cdot\overline{Q3}+\overline{Q1}\cdot Q2;$

$D1=Q2\cdot Q3+\overline{Q2}\cdot\overline{Q3};$

$D1=Q1\cdot\overline{Q3}+\overline{Q2}\cdot\overline{Q3};$

Для выполнения поставленной задачи выбран микроконтроллер (МК) PIC16F84A.
Принципиальная схема содержит сам МК, цепь тактирования на основе кварцевого резонатора и цепь сброса.



Для обмена данными используется порт ввода/вывода PORTA.
Слово конфигурации МК выглядит следующим образом:
__CONFIG _CP_OFF & _WDT_OFF & _PWRTE_ON & _XT_OSC
_CP_OFF – отключается защита памяти от чтения;
_WDT_OFF – отключается сторожевой таймер;
_PWRTE_ON – включается таймер задержки сброса МК после подачи питания;
_XT_OSC - тактирование от внутреннего генератора с внешним кварцевым резонатором.

После подачи напряжения питания выполняется сброс в исходное состояние, МК начинает работать и выполняется программа, записанная в память программ, начиная с адреса 0x0000. С адреса 0x0004 может начинаться подпрограмма обработки прерывания. Так как описываемое устройство не использует технологию прерываний, эта особенность не учитывается.

Вначале происходит назначение режимов работы портов ввода/вывода – определение направления передачи информации и устанавливается цифра, соответствующая началу заданной последовательности (в нашем примере это «7»). Эта процедура выполняется единожды за весь сеанс работы. Обработка данных происходит в бесконечном цикле. Весь цикл можно условно разделить на четыре части: ожидание фронта синхроимпульса; вычисление логических функций; возврат на начало цикла; вывод результата. Программа вводит синхроимпульс *Clk* через *PORTA* (*RA4*), вычисляет функции *D1*, *D2*, *D3* трех логических переменных, заменяет старые значения переменных новыми значениями функций и выводит результат через *PORTA (Y-RA1)*.

Следовательно, формат порта *A* следующий:

RA7 *RA6* *RA5*	*RA4*	*RA3*	*RA2*	*RA1*	*RA0*
Отсутствуют, читаются как 0	*Clk*	*св*	*Q1*	*Q2*	Q3

Ожидание фронта синхроимпульса начинается с проверки уровня сигнала *Clk*. Если на линии *Clk* присутствует уровень логической единицы (предыдущий импуль не завершен), выполняется циклический опрос этого сигнала до установки нуля. Затем аналогичным способом ожидается установка единицы.

Для того, чтобы вычислить следующую цифру последовательности, необходимо превратить разряды текущего числа в отдельные переменные *Q1*, *Q2*, *Q3*. Вычисление функций *D1*, *D2*, *D3* обеспечивается последовательностью логических операций над переменными.

Для хранения текущей комбинации определяется ячейка (регистр) с именем *now* и адресом 0x10. Выполнение логических операций над

разными разрядами одного слова невозможно, поэтому предлагается разместить переменные в одноименные разряды отдельных двоичных слов и выделить для каждого из них персональную ячейку оперативной памяти (регистр). Для выполнения этой задачи необходимо обнулять все разряды исходного слова, кроме того, где находится искомая переменная. Затем необходимо выполнить сдвиг слова, чтобы выделенная переменная оказалась в том разряде, который выбран в качестве рабочего. Допустим, в качестве рабочего выбран разряд 0. Чтобы подготовить переменную *D1* к вычислению функции, предлагается выполнить логическое умножение (конъюнкцию) исходного слова с двоичным числом b'00000100' (0x04). В результате во всех разрядах, кроме второго, будет ноль, а во втором останется переменная *D1*.

x	x	x	x	x	*D1*	*D2*	*D3*
0	0	0	0	0	1	0	0
0	0	0	0	0	*D1*	0	0

Далее выполняем двухкратный логический сдвиг вправо:

0	0	0	0	0	0	*D1*	0
0	0	0	0	0	0	0	*D1*

и получаем переменную *D1* во нулевом разряде.

Подобным образом необходимо подготовить остальные переменные. Для их хранения надо выделить три ячейки оперативной памяти (регистра), которым можно присвоить имена, например, *var1*, *var2*, *var3* с адресами 0x0C, 0x0D, 0x0E, соответственно.

После этого становится возможным выполнение логических операций, входящих в функцию. Для хранения промежуточных результатов определяется ячейка с именем *tmp* и адресом 0x0F. Значение функции является значением соответствующего разряда общего результата, т.е. следующей цифры последовательности.

После вычисления каждой функции, ее значение окажется в нулевом разряде. Так как результатом работы счетчика является трехразрядное число, его надо собрать из полученных значений функций. Следовательно, *Q2* надо сдвинуть на один, а *Q1* на два разряда влево. Затем надо очистить остальные разряды, наложив соответствующую маску, собрать все разряды в переменную *now* и вывести в порт.

Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ *Радиоэлектроники и лазерной техники*

КАФЕДРА _____

Домашнее задание №2

По курсу «Цифровые устройства и микропроцессоры»

Вариант № 0

Студент _____ *Глотов А.Н.*
подпись, дата *фамилия, и.о.*

Группа РЛ1-62

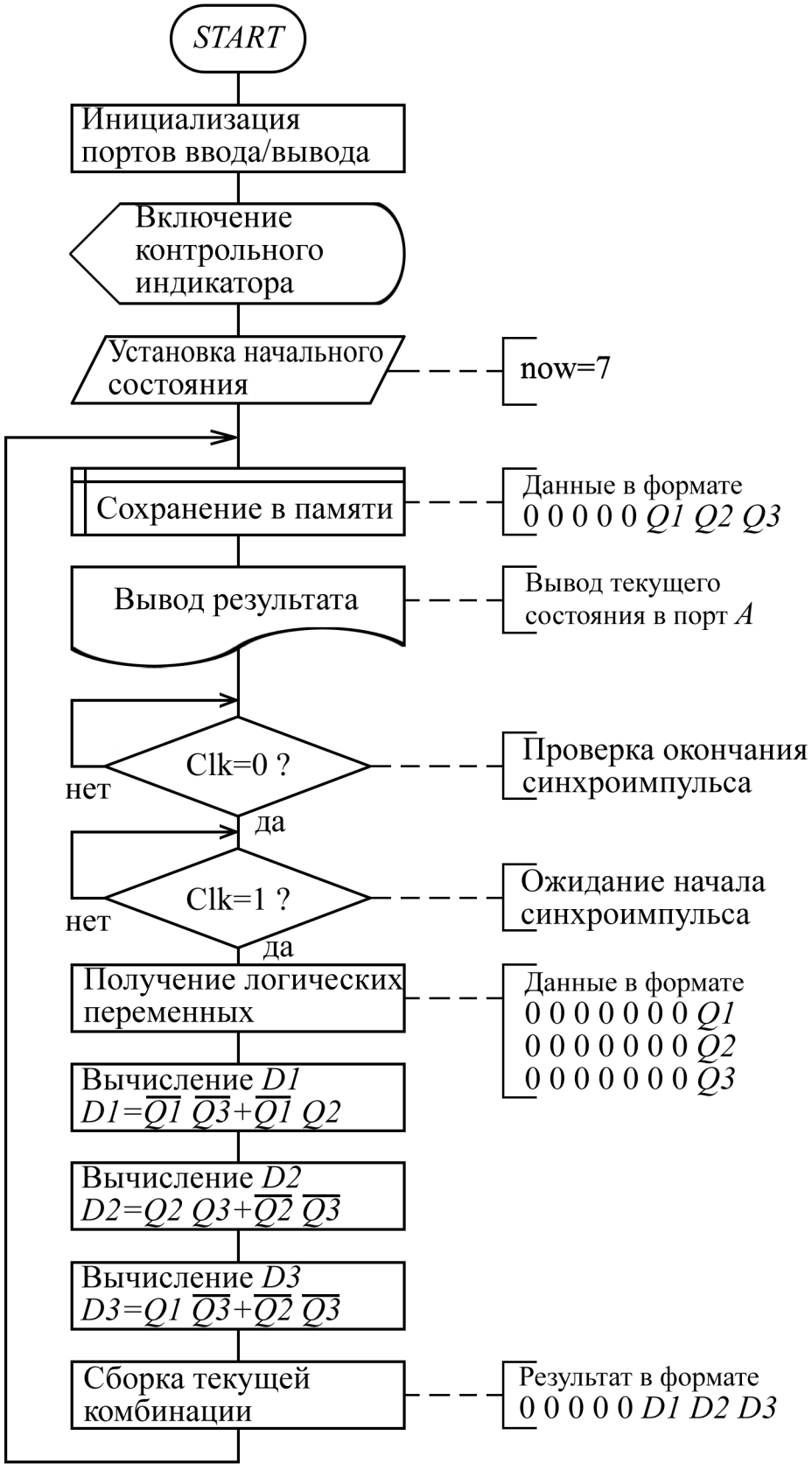
Преподаватель _____ *Глотов А.Н.*
подпись, дата *фамилия, и.о.*

Москва 2020 г.

Вариант 0

Разработать цифровое устройство на основе любого микропроцессора, выполняющее функцию, минимизированную в домашнем задании N1 (Включая принципиальную схему и управляющую программу в кодах выбранного микропроцессора).

Оформить задание с соблюдением ГОСТ 2.702-75, ГОСТ 2.743-91, ГОСТ 2.104-68.



Листинг программы

```
;*****************************************************************
; Filename: example_dz-pic-count.asm
; Программа вводит синхроимпульс через порт RA4, вычисляет очередной код
; заданной последовательности (7 2 4 3 6 1 0) с помощью логических функций
; и выводит результат через линии порта A (Q1-RA2, Q2-RA1, Q3-RA0).
;*****************************************************************
;
list p=16F84A ; list directive to define processor
#include <p16F84a.inc> ; processor specific variable definitions
__CONFIG _CP_OFF & _WDT_OFF & _PWRTE_ON & _XT_OSC
;
;----- назначаем имена переменных и соответствующие им адреса
var1  EQU 0x0C ;
var2  EQU 0x0D ; - переменные
var3  EQU 0x0E ;
tmp   EQU 0x0F ; переменная для промежуточных результатов
now   EQU 0x10 ; переменная, содержащая текущее состояние
d1    EQU 0x11 ; переменная для подпрограммы - задержки
d2    EQU 0x12 ; переменная для подпрограммы - задержки
;-----------------------------------------------------------------
ORG 0x000 ; processor reset vector
bcf STATUS,RP1 ; устанавливаем
bsf STATUS,RP0 ; банк памяти 1
movlw 0x10 ; управляющее слово для порта A
movwf TRISA ; RA4 - на ввод, RA3, RA2, RA1, RA0 - на вывод
movlw 0x00 ; управляющее слово для порта B
movwf TRISB ; все на вывод
bcf STATUS,RP0 ; возвращаемся в банк 0
;-----------------------------------------------------------------
; Приветливо мигнем индикатором на PORTA[3] и оставим его включенным -
; - не обязательный, сервисный фрагмент программы
bcf PORTA, 3     ; зажгли
call  del_200    ; подождали 200 мс
bsf PORTA, 3     ; погасили
call  del_200    ; подождали 200 мс
bcf PORTA, 3     ; зажгли
call  del_200    ; подождали 200 мс
bsf PORTA, 3     ; погасили
call  del_200    ; подождали 200 мс
bcf PORTA, 3     ; зажгли
;-----------------------------------------------------------------
; Установка начального состояния
movlw 0x07       ; первое состояние счетчика
movwf now
; --------------- Основной исполнительный цикл ----------------------
; ------- выводим результат в порт -----
loop             ; метка начала цикла
movf now, w      ; текущее состояние счетчика
movwf PORTA      ; выводим в порт
; ----- Ждем окончания синхроимпульса на PORTA[4] ------
key1
btfsc PORTA,4    ; анализ состояния бита 4 порта A
goto  key1       ; если 1 - продолжаем ждать
; если 0 - дождались окончания импульса
call  del_200    ; подождали 200 мс, пропустили дребезг
; ----- Ждем синхроимпульс на PORTA[4] ------
key2
btfss PORTA,4    ; анализ состояния бита 4 порта A
goto  key2       ; если 0 - продолжаем анализировать сигнал
; если 1 - дождались импульса, продолжаем выполнять программу
call  del_200    ; подождали 200 мс, пропустили дребезг
```

```
;-----------------------------------------------------------------------
;выделяем разряды текущего кода по отдельности и помещаем их в ячейки var1, var2 и var3
movf now, w ; копируем текущий код в аккумулятор
andlw 0x01 ; накладываем маску 000000001 - выделяем младший разряд Q3
movwf var3 ; копируем Q3 в var3
movf now, w ; копируем текущий код в аккумулятор
andlw 0x02 ; накладываем маску 000000010 - выделяем младший разряд Q2
movwf var2 ; копируем Q2 в var2
rrf var2, f ; сдвиг Q2 вправо - выравниваем с Q3 в младшем разряде
movf now, w ; копируем текущий код в аккумулятор w
andlw 0x04 ; накладываем маску 000000100 - выделяем старший разряд Q1
movwf var1 ; копируем Q1 в var1
rrf var1, f ; сдвиг Q1 вправо
rrf var1, f ; сдвиг Q1 вправо - выравниваем с Q3 и Q2 в младшем разряде
clrf now ; очищаем now для нового кода
; --- Вычисляем D1=!Q1*!Q3+!Q1*Q2 ---
comf var1, w ; инвертируем Q1 с сохранением в аккумуляторе w
movwf tmp ; сохраняем в tmp
comf var3, w ; инвертируем Q3 с сохранением в аккумуляторе w
andwf tmp, f ; !Q1*!Q3 с сохранением в tmp
comf var1, w ; инвертируем Q1 с сохранением в аккумуляторе w
andwf var2, w ; !Q1*Q2 с сохранением в w
iorwf tmp, f ; D1 в 0-м разряде tmp
rlf tmp, f ; сдвигаем влево с сохранением в tmp
rlf tmp, w ; сдвигаем влево с сохранением в w - выравниваем с 2-м разрядом
andlw 0x04 ; накладываем маску 000000100 - чистим результат
iorwf now, f ; устанавливаем D1 в 2-й разряд now
; --- Вычисляем D2=Q2*Q3+!Q2*!Q3 ---
movf var2, w ; копируем Q2 в w
andwf var3, w ; Q2*Q3 с сохранением в w
movwf tmp ; Q2*Q3 копируем в tmp
comf var2, f ; инвертируем Q2 с сохранением в var2
comf var3, w ; инвертируем Q3 с сохранением в аккумуляторе w
andwf var2, f ; !Q2*!Q3 с сохранением в var2 (пригодится для D3)
movf var2, w ; копируем !Q2*!Q3 в аккумулятор w
iorwf tmp, f ; D2 в 0-м разряде tmp
rlf tmp, w ; сдвигаем влево с сохранением в w
andlw 0x02 ; накладываем маску 000000010 - чистим результат
iorwf now, f ; устанавливаем D2 в 1-й разряд now
; --- Вычисляем D3=Q1*!Q3+!Q2*!Q3 ---
comf var3, w ; инвертируем Q3 с сохранением в аккумуляторе w
andwf var1, w ; Q1*^Q3 с сохранением в w
iorwf var2, w ; D3 в 0-м разряде w
andlw 0x01 ; накладываем маску 000000001 - чистим результат
iorwf now, f ; устанавливаем D3 в 0-й разряд now - РЕЗУЛЬТАТ
goto loop ; Зацикливаем программу
; ------ Итого 44 машинных цикла, т.е. при тактовой частоте 4 МГц время выполнения
циклической части 44 мкс. Без учета ожидания импульса и устранения дребезга.
; ---- Подпрограмма - задержка 200 мс -------
; Переменные: d1 и d2
del_200
movlw 0xCC
movwf d2
d200_2 movlw 0xF9
movwf d1
nop
nop
d200_1 nop
decfsz d1,f
goto d200_1
nop
decfsz d2,f
goto d200_2
return
; --------- end of DEL_200 -------------
end ; directive 'end of program'
```

Коды команд программы, их адреса в памяти программ и время выполенния в машинных циклах приведены в таблице:

Адрес	Код	Команда	Время, МТ
000	1303	BCF 0x3, 0x6	1
001	1683	BSF 0x3, 0x5	1
002	3010	MOVLW 0x10	1
003	0085	MOVWF 0x5	1
004	30FF	MOVLW 0xff	1
005	0086	MOVWF 0x6	1
006	1283	BCF 0x3, 0x5	1
007	1585	BSF 0x5, 0x3	1
008	2041	CALL 0x41	1
009	1185	BCF 0x5, 0x3	1
00A	2041	CALL 0x41	1
00B	1585	BSF 0x5, 0x3	1
00C	2041	CALL 0x41	1
00D	1185	BCF 0x5, 0x3	1
00E	3007	MOVLW 0x7	1
00F	0090	MOVWF 0x10	1
010	0810	MOVF 0x10, W	1
011	0085	MOVWF 0x5	1
012	1A05	BTFSC 0x5, 0x4	2
013	2812	GOTO 0x12	2
014	2041	CALL 0x41	2
015	1E05	BTFSS 0x5, 0x4	2
016	2815	GOTO 0x15	2
017	2041	CALL 0x41	2
018	0810	MOVF 0x10, W	1
019	3901	ANDLW 0x1	1
01A	008E	MOVWF 0xe	1
01B	0810	MOVF 0x10, W	1
01C	3902	ANDLW 0x2	1
01D	008D	MOVWF 0xd	1
01E	0C8D	RRF 0xd, F	1
01F	0810	MOVF 0x10, W	1
020	3904	ANDLW 0x4	1
021	008C	MOVWF 0xc	1
022	0C8C	RRF 0xc, F	1
023	0C8C	RRF 0xc, F	1
024	0190	CLRF 0x10	1
025	090C	COMF 0xc, W	1
026	008F	MOVWF 0xf	1

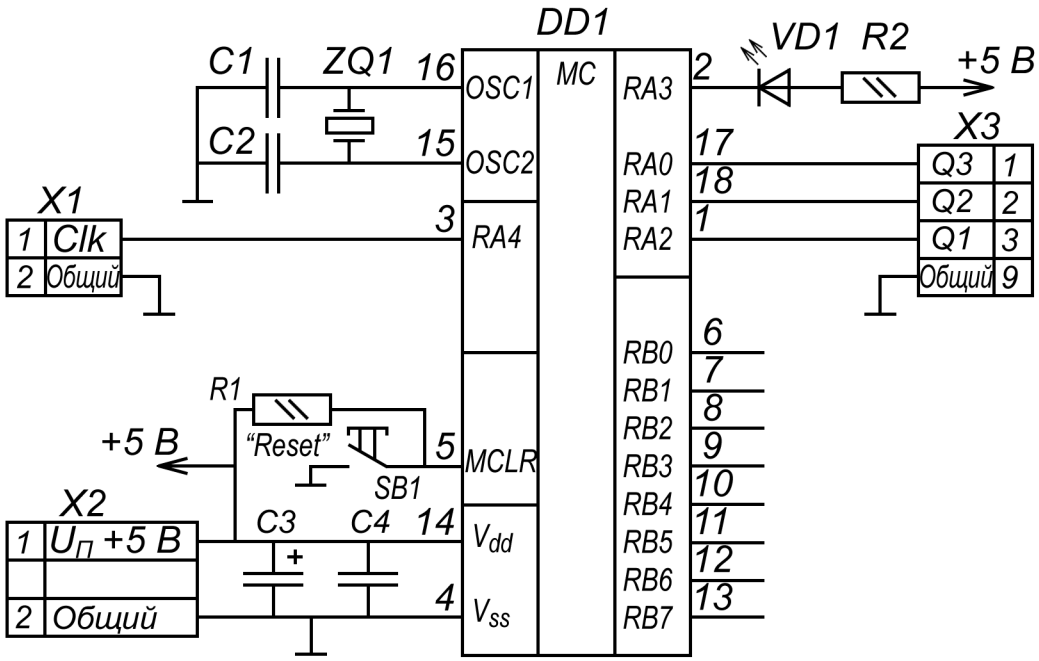
Адрес	Код	Команда	Время, МТ
027	090E	COMF 0xe, W	1
028	058F	ANDWF 0xf, F	1
029	090C	COMF 0xc, W	1
02A	050D	ANDWF 0xd, W	1
02B	048F	IORWF 0xf, F	1
02C	0D8F	RLF 0xf, F	1
02D	0D0F	RLF 0xf, W	1
02E	3904	ANDLW 0x4	1
02F	0490	IORWF 0x10, F	1
030	080D	MOVF 0xd, W	1
031	050E	ANDWF 0xe, W	1
032	008F	MOVWF 0xf	1
033	098D	COMF 0xd, F	1
034	090E	COMF 0xe, W	1
035	058D	ANDWF 0xd, F	1
036	080D	MOVF 0xd, W	1
037	048F	IORWF 0xf, F	1
038	0D0F	RLF 0xf, W	1
039	3902	ANDLW 0x2	1
03A	0490	IORWF 0x10, F	1
03B	090E	COMF 0xe, W	1
03C	050C	ANDWF 0xc, W	1
03D	040D	IORWF 0xd, W	1
03E	3901	ANDLW 0x1	1
03F	0490	IORWF 0x10, F	1
040	2810	GOTO 0x10	2
041	30CC	MOVLW 0xcc	1
042	0092	MOVWF 0x12	1
043	30F9	MOVLW 0xf9	1
044	0091	MOVWF 0x11	1
045	0000	NOP	1
046	0000	NOP	1
047	0000	NOP	1
048	0B91	DECFSZ 0x11, F	2
049	2847	GOTO 0x47	2
04A	0000	NOP	1
04B	0B92	DECFSZ 0x12, F	2
04C	2843	GOTO 0x43	2
04D	0008	RETURN	2

Итого 44 машинных цикла, т.е. при тактовой частоте 4 МГц время выполнения основного исполнительного цикла составит 44 мкс, код программы занял 78 ячеек памяти программ (FLASH ROM) и использованы 7 ячеек памяти данных (ОЗУ).

«Прошивка» FLASH ROM в формате Intel HEX.

```
:020000040000FA
:10000000031383161030850FF3086008312851598
:10001000412085114120851541208511073090003O
:1000200010088500051A12284120051E15284120B8
:10003000100801398E00100802398D008D0C10084F
:1000400004398C008C0C8C0C90010C098F000E096B
:100050008F050C090D058F048F0D0F0D04399004C9
:100060000D080E058F008D090E098D050D088F04F2
:100070000F0D023990040E090C050D04013990048E
:100080001028CC309200F9309100000000000000F0
:0C009000910B47280000920B4328080049
:02400E00F13F80
:00000001FF
```

РП1.ХХХХХХ.002 Э3



Поз. Обозначение	Наименование	Кол.	Примечание
	Микросхемы		
DD1	PIC16F84A-04I/P	1	
	Конденсаторы		
C1, C2	К10-17-22-П30	2	
C3	К53-1-220,0х6,3 В	1	
C4	К10-17-0,22-Н90	1	
	Соединители		
X1	СР50-73ФВ	1	
X2	DS-213B	1	
X3	DB9F	1	
	Резисторы		
R1	МЛТ 0,125-3,3к-5%	1	
R2	МЛТ 0,125-330-5%	1	
	Кварцевый резонатор		
ZQ1	4.000MHzHC-49U	1	4 МГц
	Светодиод		
VD1	ARL-3514UYD-150MCD	1	желтый
	Выключатель		
SB1	TS-A3PS-130 SWT6	1	кнопка тактовая

Изм.	Лист	№ докум.	Подп.	Дата
Разраб.		Глотов А.Н.		28.01.2020г.
Пров.		Жаркова Н.А.		28.01.2020г.
Т. контр.				
Н. контр.				
Утв.				

РП1.ХХХХХХ.002 Э3

Счетчик 7-2-4-3-6-1-0
Схема электрическая принципиальная

Лит.	Масса	Масштаб

Лист 1 | Листов 1

МГТУ им.Н.Э.Баумана
Кафедра РЛ-1
Группа РЛ1-61

Копировал

Формат А4

Перв. примен.

Справ. №

Подп. и дата

Инв. № дубл.

Взам. инв. №

Подп. и дата

Инв. № подл.

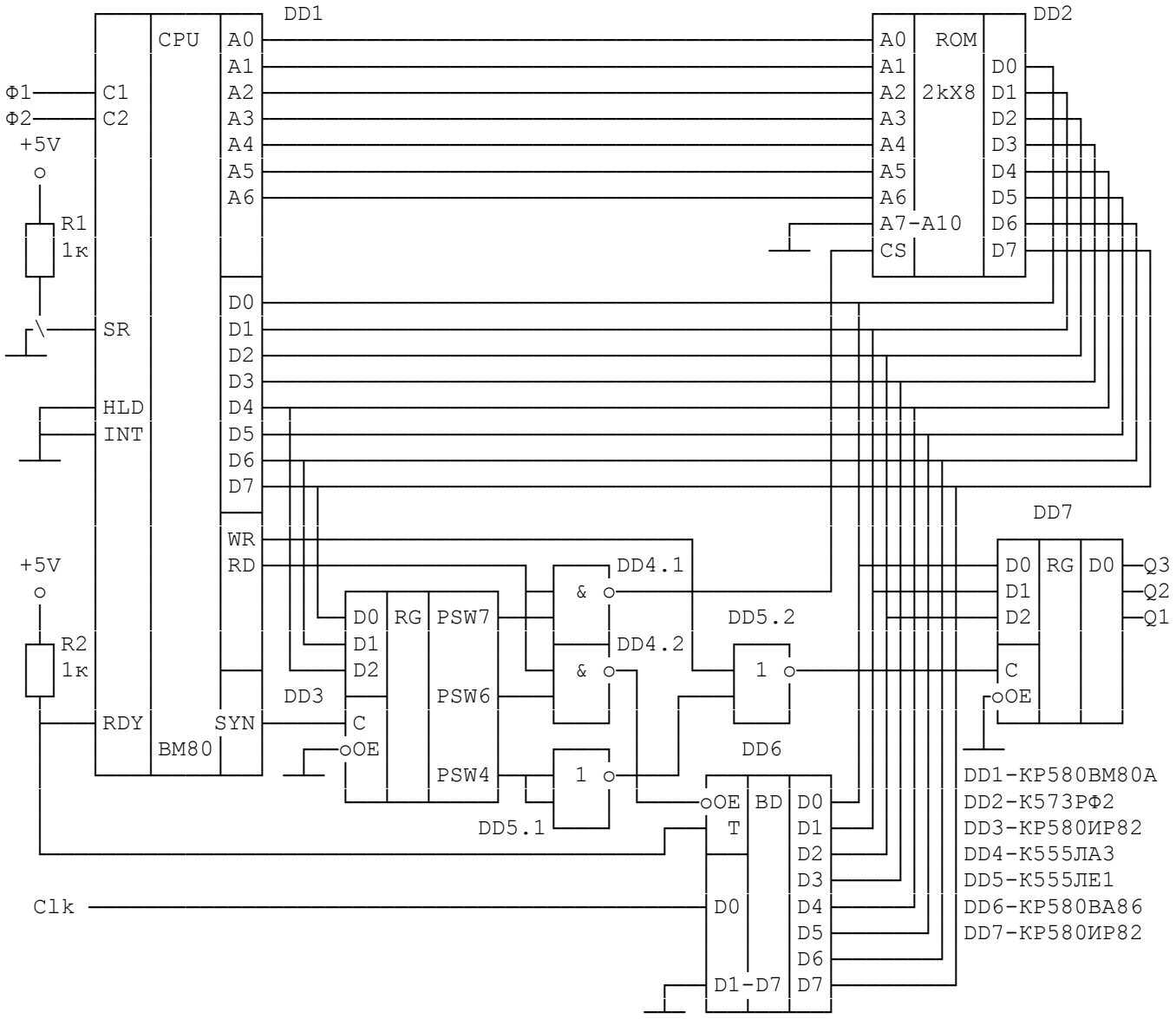
ПРИМЕР №2: Заданы функции D1, D2 и D3 трех переменных Q1, Q2, Q3:

$D1=\overline{Q1}\cdot\overline{Q3}+\overline{Q1}\cdot Q2;$

$D1=Q2\cdot Q3+\overline{Q2}\cdot\overline{Q3};$

$D1=Q1\cdot\overline{Q3}+\overline{Q2}\cdot\overline{Q3};$

Для выполнения поставленной задачи выбран микропроцессор (МП) КР580ВМ80А.
Принципиальная схема содержит сам МП, цепь тактирования, цепь сброса, регистр слова состаяния, память, порт ввода и порт вывода.



Пример построения МП – устройства на основе МП КР580ВМ80А

Рассмотрим принципиальную схему МП - системы, реализующей заданную функцию под управлением программы.

Система построена на базе центрального процессора DD1 КР580ВМ80А, синхронизируемого сигналами Ф1 и Ф2 от внешнего генератора. Управляющая программа содержится в ПЗУ DD2. Ячейки ПЗУ адресуются по семи младшим

линиям шины адреса микропроцессора, что обеспечивает доступ к 128 байтам программы. Выбранный в ПЗУ байт поступает в шину данных системы, образованную внешней шиной данных МП, выводами данных ПЗУ DD2, регистров DD3 и DD7 и шинного формирователя DD6.

Тактовый сигнал будем считывать через порт ввода

с адресом 00000000 = 00H

0	0	0	0	0	0	0	Clk

Выходные сигналы будем выводить через порт вывода

с адресом 00000001 = 01H

0	0	0	0	0	Q1	Q2	Q3

Входной сигнал поступает по линии Clk на вход шинного формирователя DD6, образующего с элементом DD4.2 порт ввода. Выходные сигналы поступают на линии Q1, Q2, Q3 с выхода регистра DD7, образующего с элементами DD5 порт вывода.

В начале каждого цикла на шину данных МП выводится слово состояния процессора (PSW). В рассматриваемой системе используется только 3 разряда PSW - разряд, информирующий о начале цикла вывода данных в порт вывода (PSW4), разряд, информирующий о вводе из порта ввода (PSW6) и разряд, информирующий о чтении памяти (PSW7). Значения этих разрядов записываются в регистр слова состояния на основе регистра-защелки DD3. Запись синхронизируется сигналом SYN (СИНХРО). Эти сигналы позволяют отличать циклы обращения к памяти от циклов обращения к портам ввода - вывода. При выполнении процессором цикла чтения памяти в разряд PSW7 записывается логическая 1, в остальные разряды - нулевые значения. Сигнал PSW7 поступает на вход логического элемента DD4.1. На второй вход подается сигнал RD, информирующем о том, что процессор считывает сигналы с шины данных. Сигнал с выхода этого элемента подается на вход CS микросхемы ПЗУ. Единичный сигнал на CS переводит линии данных ПЗУ в z - состояние. При активном единичном сигнале RD на выходе DD4.1 появляется уровень логического 0, Z - состояние снимается и байт данных или кода команды поступает из ПЗУ в МП.

В процессе выполнения цикла чтения из порта в ходе выполнения команды IN (port) (чтение порта) в разряде PSW6 появляется 1. При активном сигнале RC на выходе элемента DD4.2 появляется лог. 0,

который переводит шинный формирователь DD6 в режим передачи информации с входных линий в шину данных.

В процессе выполнения цикла записи в порт OUT (port) в разряде PSW4 появляется 1. При активном нулевом сигнале WR на выходе элемента DD5.2 появляется лог. 1, которая переводит регистр DD7 в режим записи информации с шины данных. Информация с шины данных поступает на выходные линии Q1 – Q3 и сохраняется на ней до следующего цикла вывода.

Так как в рассматриваемом устройстве используется только один порт ввода и один порт вывода, выборка их по адресам не требуется и, поэтому в программе адреса портов могут быть любыми (00h – FFh).

Единичный сигнал SR устанавливает микропроцессор в исходное состояние. В этом случае в счетчик команд записывается 0, т.е. начинается выполнение программы с адреса 0000H.

```
Адрес  Действие                              оператор   код     мт  Ф
; - Установить в качестве текущей первую цифру последовательности
; - и вывести ее в порт
0000   Загрузить в рег.E первое состояние    MVI E,07H  1E 07   7   2
0002   Скопировать в A                       MOV A,E    7B       5   1
0003   Вывести текущее состояние в порт      OUT 01H    D3 01   10   2
; ---- Организовать проверку окончания синхроимпульса,
; ---- т.е. переход от уровня 1 к 0
0005 Загрузить данные из порта ввода в A    IN 00H     D8 00   10   2
0007 Выделить Clk:(A) AND 00000001          ANI 01H    E6 01    7   2
(Обнуляются все разряды, кроме 0)
0009 Перейти, если результат ненулевой      JNZ 0005H  C20500  10   3
; - Организовать ожидание фронта синхроимпульса,
; - т.е. переход от уровня 0 к 1
000C   Загрузить данные из порта ввода в A  IN 00H     D8 00   10   2
000E Выделить Clk:(A) AND 00000001          ANI 01H    E6 01    7   2
     (Обнуляются все разряды, кроме 0)
0010 Перейти, если результат нулевой        JZ 000CH   CA0C00  10   3
0013 Восстановить текущее состояние в A     MOV A,E    7B       5   1
0014 Выделить Q3:(A) AND 00000001           ANI 01H    E6 01    7   2
(Обнуляются все разряды, кроме 0)
0016 Сохранить Q3 в рег. D                  MOV D,A    57       5   1
0017 Восстановить текущее состояние в A     MOV A,E    7B       5   1
0018 Выделить Q2:(A) AND 00000010           ANI 02H    E6 02    7   2
(Обнуляются все разряды, кроме 1)
001A Сдвиг (A) вправо                       RRC        0F       4   1
001B Сохранить Q2 в рег. C                  MOV C,A    4F       5   1
001C Восстановить текущее состояние в A     MOV A,E    7B       5   1
001D Выделить Q1(A) AND 00000100            ANI 04H    E6 04    7   2
(Обнуляются все разряды, кроме 2)
001F Сдвиг (A) вправо                       RRC        0F       4   1
0020 Сдвиг (A) вправо                       RRC        0F       4   1
0021 Сохранить Q1 в рег. B                  MOV B,A    47       5   1
0022 Очистить E для нового кода             MVI E,00H  1E 00    7   2
```

```
; --- Вычисляем D1=!Q1*!Q3+!Q1* Q2 ---
0024 Инвертировать Q1                          CMA        2F      4  1
0025 Сохранить в H                             MOV H,A    67      5  1
0026 Копировать Q3 в A                         MOV A,D    7A      5  1
0027 Инвертировать Q3                          CMA        2F      4  1
0028 !Q1*!Q3                                   ANA H      A4      4  1
0029 Сохранить в L                             MOV A,L    7D      5  1
002A Копировать Q2 в A                         MOV A,C    79      5  1
002B !Q1*Q2                                    ANA H      A4      4  1
002C D1=!Q1*!Q3+!Q1* Q2                        ORA L      B5      4  1
002D Сдвиг результата влево                    RLC        07      4  1
002E Сдвиг результата влево                    RLC        07      4  1
002F Очистить результат                        ANI 04H    E6 04   7  2
0031 Установить D1 в E                         ORA E      B3      4  1
0032 Сохранить в E                             MOV E,A    5F      5  1
; --- Вычисляем D2= Q2* Q3+!Q2*!Q3 ---
0033 Копировать Q2 в A                         MOV A,C    79      5  1
0034 Q2*Q3                                     ANA D      A2      4  1
0035 Сохранить в H                             MOV H,A    67      5  1
0036 Копировать Q2 в A                         MOV A,C    79      5  1
0037 Инвертировать Q2 в A                      CMA        2F      4  1
0038 Сохранить в L                             MOV A,L    7D      5  1
0039 Копировать Q3 в A                         MOV A,D    7A      5  1
003A Инвертировать Q3 в A                      CMA        2F      4  1
003B !Q2*!Q3                                   ANA L      A5      4  1
003C Сохранить в D(Q3 больше не понадобится)   MOV D,A    57      5  1
003D D2= Q2* Q3+!Q2*!Q3                        ORA H      B4      4  1
003E Сдвиг результата влево                    RLC        07      4  1
003F Очистить результат                        ANI 02H    E6 02   7  2
0041 Установить D2 в E                         ORA E      B3      4  1
0042 Сохранить в E                             MOV E,A    5F      5  1
; --- Вычисляем D3= Q1*!Q3+!Q2*!Q3 ---
0043 Копировать !Q3 в A                        MOV A,L    7D      5  1
0044 Q1*!Q3                                    ANA B      A0      4  1
0045 D3= Q1*!Q3+!Q2*!Q3                        ORA H      B4      4  1
0046 Сдвиг результата влево                    RLC        07      4  1
0047 Сдвиг результата влево                    RLC        07      4  1
0048 Очистить результат                        ANI 04H    E6 04   7  2
004A Установить D3 в E                         ORA E      B3      4  1
004B Сохранить в E                             MOV E,A    5F      5  1
004C Перейти на начало программы               JMP 0002H  C30200 10  3
004D
004E
```

итого 264 машинных такта. При тактовой частоте 2 МГц длительность одного машинного такта 0,5 мкс, значит программа будет выполняться 132 мкс. Прошивка ПЗУ занимает 79 байт.

Адрес	Коды команд															
00	1E	07	7B	D3	01	D8	00	E6	01	C2	05	00	D8	00	E6	01
10	CA	0C	00	7B	E6	01	57	7B	E6	02	0F	4F	7B	E6	04	0F
20	0F	47	1E	00	2F	67	7A	2F	A4	7D	79	A4	B5	07	07	E6
30	04	B3	5F	79	A2	67	79	2F	7D	7A	2F	A5	57	B4	07	E6
40	02	B3	5F	7D	A0	B4	07	07	E6	04	B3	5F	C3	02	00	FF