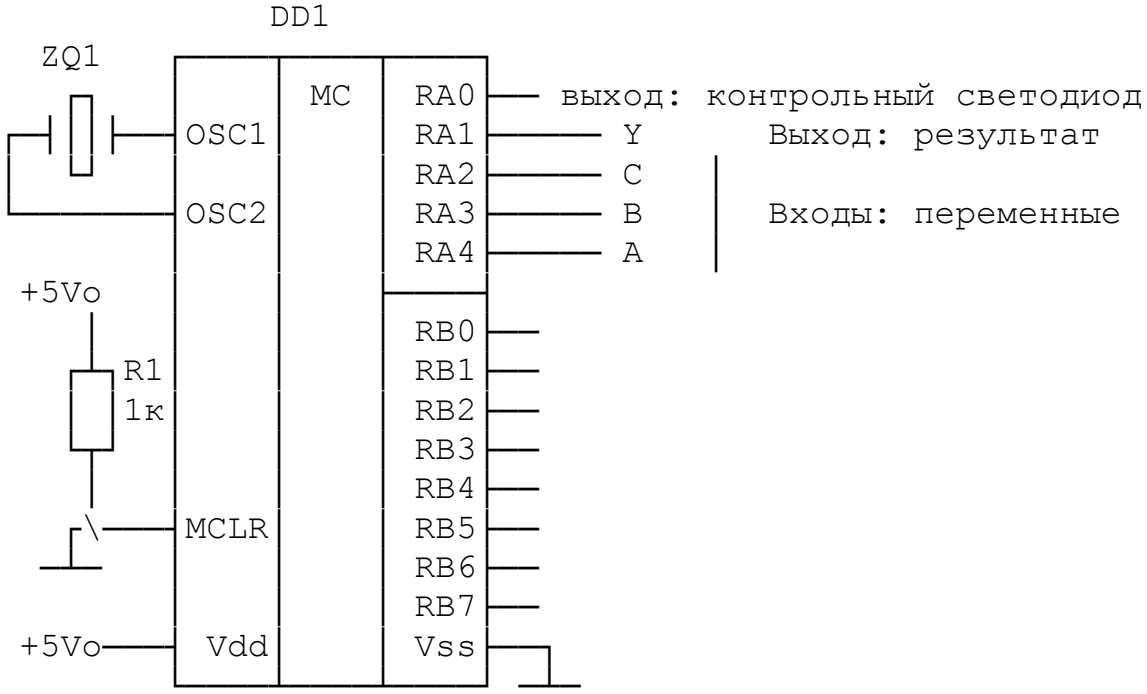
Домашнее задание N2 по курсу "Электроника и МПТ".
Разработать цифровое устройство на основе любого микропроцессора, выполняющее функцию, минимизированную в домашнем задании N1 (Включая принципиальную схему и управляющую программу в кодах выбранного микропроцессора).

ПРИМЕР №1: Задана в виде таблицы истинности логическая функция Y трех переменных A, B, C.

A	B	C	Y
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

$$Y = \overline{A}\cdot\overline{B} + \overline{B}\cdot\overline{C} + A\cdot B\cdot C = \overline{A+B} + \overline{B+C} + A\cdot B\cdot C$$

Для выполнения поставленной задачи выбран микроконтроллер 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 устройство 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. Эта процедура выполняется единожды за весь сеанс работы. Обработка данных происходит в бесконечном цикле. Весь цикл можно условно разделить на три части: ввод информации и ее обработка; вычисление логической функции; вывод результата и возврат на начало цикла. Программа вводит три сигнала через *PORTA* (*A–RA4, B–RA3, C–RA2*), вычисляет функцию трех логических переменных $Y = \overline{A} \cdot \overline{B} + \overline{B} \cdot \overline{C} + A \cdot B \cdot C$ и выводит результат в *PORTA (Y–RA1)*.

Следовательно, формат порта *A* следующий:

RA7 *RA6* *RA5*	*RA4*	*RA3*	*RA2*	*RA1*	*RA0*
Отсутствуют, читаются как 0	*A*	*B*	*C*	*Y*	0

Таким образом, в результате чтения сигналов, поданных на линии порта, логические переменные *A, B, C* оказываются в регистре-аккумуляторе *W* в виде четвертого, третьего и второго разрядов двоичного слова. Остальные пять разрядов не несут полезной информации и должны быть обнулены. Для хранения принятой комбинации сигналов определяется ячейка (регистр) с именем *datainp* 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 2. Чтобы подготовить переменную *A* к вычислению функции, предлагается

выполнить логическое умножение (конъюнкцию) исходного слова с двоичным числом b'00010000' (0x10). В результате во всех разрядах, кроме четвертого, будет ноль, а в четвертом останется переменная *A*.

```
x x x A B C x x
0 0 0 1 0 0 0 0
0 0 0 A 0 0 0 0
```

Далее выполняем двухкратный логический сдвиг вправо:

```
0 0 0 0 A 0 0 0
0 0 0 0 0 A 0 0
```

и получаем переменную *A* во втором разряде.

Подобным образом необходимо подготовить остальные переменные. Для их хранения надо выделить три ячейки оперативной памяти (регистра), которым можно присвоить имена, например, *vara*, *varb*, *varc* 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.

После вычисления функции, ее значение окажется во втором разряде. Так как для ввода переменных и вывода результата выбран один порт *A*, выводить результат сразу нельзя, потому что второй разряд занят для ввода переменной *C*. Следовательно, результат надо сдвинуть вправо, он окажется в первом разряде и тогда выводить 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 «Электроника и микропроцессорная техника»

Вариант № 0

Студент _____ _____
подпись, дата *фамилия, и.о.*

Группа РЛ2-51

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

Москва 2019 г.

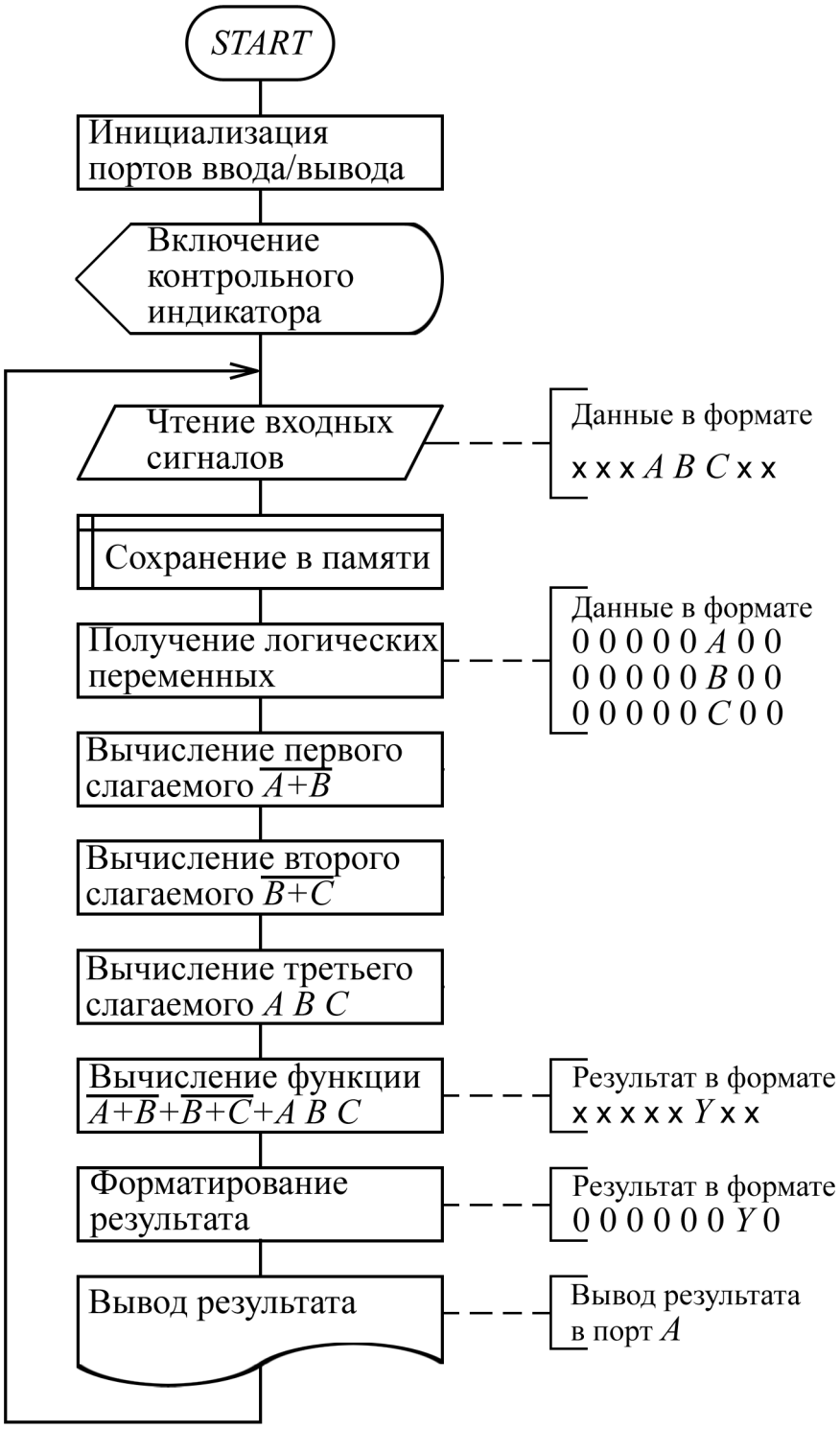
Задание

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.

Вариант 0

Функция получена при выполнении ДЗ №1: $Y = \overline{A} \cdot \overline{B} + \overline{B} \cdot \overline{C} + A \cdot B \cdot C$



Листинг программы

```
;*****************************************************************
;    Filename:       example_dz-pic.asm
;     Программа вводит три сигнала через порт A (A-RA4, B-RA3, C-RA2),
;   вычисляет функцию трех логических переменных Y и выводит результат
;   в порт A (Y-RA1) в прямом виде
;*****************************************************************

      list      p=16F84A            ; list directive to define processor
      #include <p16F84a.inc>        ; processor specific variable definitions

      __CONFIG   _CP_OFF & _WDT_OFF & _PWRTE_ON & _XT_OSC

;***** VARIABLE DEFINITIONS
vara        EQU         0x0C  ;     A
varb        EQU         0x0D  ;     B     - входные переменные
varc        EQU         0x0E  ;     C
tmp         EQU         0x0F
datainp     EQU         0x10
d1          EQU         0x11
d2          EQU         0x12
;*****************************************************************
      ORG   0x000       ; processor reset vector
      bcf   STATUS,RP1  ; устанавливаем
      bsf   STATUS,RP0  ; банк памяти 1
      movlw 0x1C        ; управляющее слово для порта A
      movwf TRISA       ; RA4, RA3, RA2 - на ввод, RA1, RA0 - на вывод
      movlw 0xFF        ; управляющее слово для порта B
      movwf TRISB       ; все на ввод (пока)
      bcf   STATUS,RP0  ; возвращаемся в банк 0

; Приветливо мигнем индикатором на PORTA[0] и оставим его включенным -
; - не обязательный, сервисный фрагмент программы
      bsf   PORTA, 0    ; погасили
      call  del_100     ; подождали 100 мс
      call  del_100     ; подождали 100 мс
      bcf   PORTA, 0    ; зажгли
      call  del_100     ; подождали 100 мс
      call  del_100     ; подождали 100 мс
      bsf   PORTA, 0    ; погасили
      call  del_100     ; подождали 100 мс
      call  del_100     ; подождали 100 мс
      bcf   PORTA, 0    ; зажгли
      call  del_100     ; подождали 100 мс
      call  del_100     ; подождали 100 мс
; ---------------------------------------------------------------------------
; ---------------  Основной исполнительный цикл  ----------------------------
; ----------  получаем комбинацию переменных и раскладываем их по разным регистрам
; x  x  x  A  B  C  Y  0 --  формат порта A
start movf  PORTA, w    ; читаем данные из порта A
      movwf datainp     ; сохраняем
      andlw 0x04        ; накладываем  маску 00000100 - выделяем C
      movwf varc        ; сохраняем в varc
      movf  datainp, w  ; восстанавливаем исходные данные в аккумулятор
      andlw 0x08        ; накладываем  маску 00001000 - выделяем B
      movwf varb        ; сохраняем в varb
      rrf   varb, f     ; сдвиг B вправо - выравниваем с C
      movf  datainp, w  ; восстанавливаем исходные данные в аккумулятор
      andlw 0x10        ; накладываем  маску 00010000 - выделяем A
      movwf vara        ; сохраняем в vara
      rrf   vara, f     ; сдвиг A вправо
      rrf   vara, f     ; сдвиг A вправо - выравниваем с C
```

```
; --------------- вычисляем функцию -------------------------------------
;  y=!(A+B)+!(B+C)+ABC
; ---------------------------------------------------------------------
      movf  vara, w     ; A - в аккумулятор
      iorwf varb, w     ; в аккумуляторе A+B
      movwf tmp         ; сохраняем в tmp
      comf  tmp, f      ; инвертируем A+B с сохранением в tmp
      movf  varb, w     ; B - в аккумулятор
      iorwf varc, w     ; дизъюнкция B+C в аккумуляторе
      xorlw 0x04        ; инвертируем B+C с сохранением в w
      iorwf tmp, f      ; !(A+B)+!(B+C) с сохранением в tmp
      movf  vara, w     ; A - в аккумулятор
      andwf varb, w     ; конъюнкция A•B в аккумуляторе
      andwf varc, w     ; конъюнкция A•B•C в аккумуляторе
      iorwf tmp, f      ; !(A+B)+!(B+C)+A•B•C с сохранением в tmp - РЕЗУЛЬТАТ
      rrf   tmp, w      ; сдвиг вправо - выравниваем с y с сохранением в w
      andlw 0x02  ; накладываем маску 00000010 - чистим результат перед выводом в порт
      movwf PORTA       ; выводим результат в порт

      goto  start       ; Зацикливаем программу
; ------ Итого 30 машинных циклов, т.е. при тактовой частоте 4 МГц время выполнения
цикла 30 мкс.

; Подпрограмма - задержка 100 мс DEL_100
; Переменные: d1 и d2
del_100     movlw   0x66
            movwf   d2
d100_2      movlw   0xF9
            movwf   d1
            nop
            nop
d100_1      nop
            decfsz      d1,f
            goto  d100_1
            nop
            decfsz      d2,f
            goto  d100_2
            return
; --------- end of DEL_100 -------------

            end             ; directive 'end of program'
```

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

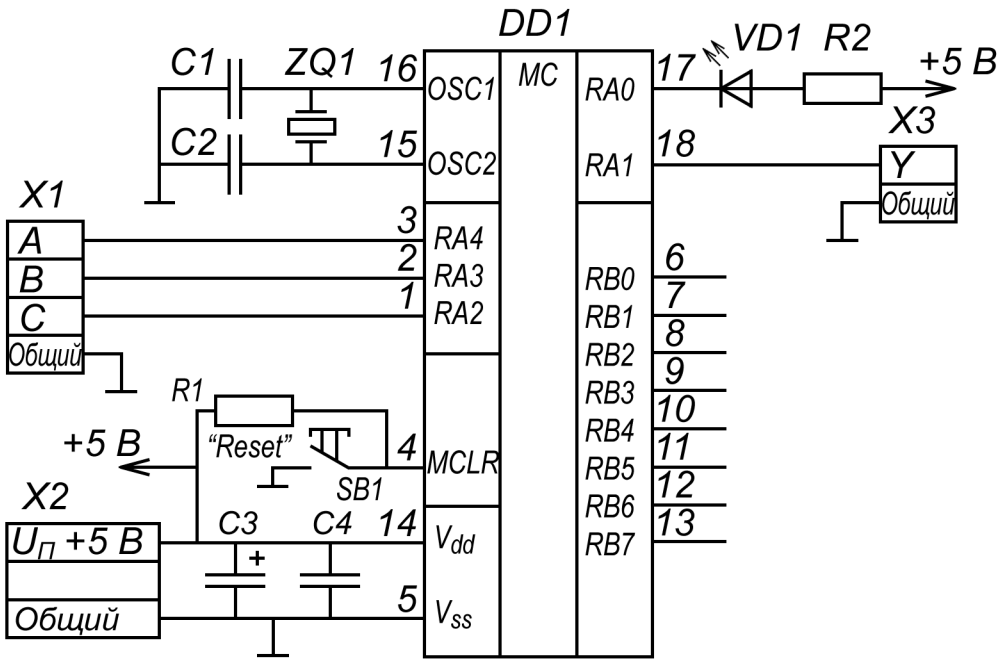
Адрес	Код	Команда	Время, МЦ
0000	1303	BCF 0x3, 0x6	1
0001	1683	BSF 0x3, 0x5	1
0002	301C	MOVLW 0x1c	1
0003	0085	MOVWF 0x5	1
0004	30FF	MOVLW 0xff	1
0005	0086	MOVWF 0x6	1
0006	1283	BCF 0x3, 0x5	1
0007	1405	BSF 0x5, 0	1
0008	2030	CALL 0x30	1
0009	2030	CALL 0x30	1
000A	1005	BCF 0x5, 0	1
000B	2030	CALL 0x30	1
000C	2030	CALL 0x30	1
000D	1405	BSF 0x5, 0	1
000E	2030	CALL 0x30	1
000F	2030	CALL 0x30	1
0010	1005	BCF 0x5, 0	1
0011	2030	CALL 0x30	1
0012	2030	CALL 0x30	1
0013	0805	MOVF 0x5, W	1
0014	0090	MOVWF 0x10	1
0015	3904	ANDLW 0x4	1
0016	008E	MOVWF 0xe	1
0017	0810	MOVF 0x10, W	1
0018	3908	ANDLW 0x8	1
0019	008D	MOVWF 0xd	1
001A	0C8D	RRF 0xd, F	1
001B	0810	MOVF 0x10, W	1
001C	3910	ANDLW 0x10	1
001D	008C	MOVWF 0xc	1
001E	0C8C	RRF 0xc, F	1
001F	0C8C	RRF 0xc, F	1
0020	080C	MOVF 0xc, W	1
0021	040D	IORWF 0xd, W	1
0022	008F	MOVWF 0xf	1
0023	098F	COMF 0xf, F	1
0024	080D	MOVF 0xd, W	1
0025	040E	IORWF 0xe, W	1
0026	3A04	XORLW 0x4	1
0027	048F	IORWF 0xf, F	1
0028	080C	MOVF 0xc, W	1
0029	050D	ANDWF 0xd, W	1
002A	050E	ANDWF 0xe, W	1
002B	048F	IORWF 0xf, F	1
002C	0C0F	RRF 0xf, W	1
002D	3902	ANDLW 0x2	1
002E	0085	MOVWF 0x5	1
002F	2813	GOTO 0x13	2
0030	3066	MOVLW 0x66	1
0031	0092	MOVWF 0x12	1
0032	30F9	MOVLW 0xf9	1
0033	0091	MOVWF 0x11	1
0034	0000	NOP	1
0035	0000	NOP	1
0036	0000	NOP	1
0037	0B91	DECFSZ 0x11, F	2
0038	2836	GOTO 0x36	2
0039	0000	NOP	1
003A	0B92	DECFSZ 0x12, F	2
003B	2832	GOTO 0x32	2
003C	0008	RETURN	2

Итого 30 машинных циклов, т.е. при тактовой частоте 4 МГц время выполнения основного исполнительного цикла составит 30 мкс, код программы занял 61 ячейку памяти программ (FLASH ROM) и использованы 7 ячеек памяти данных (ОЗУ).

«Прошивка» FLASH ROM в формате Intel HEX.

```
:020000040000FA
:10000000031383161C308500FF308600831205140D
:1000100030203020051030203020051430203020D2
:1000200005103020302005089000043980010089B
:100030000839800800C100810398C008C0C8C0C3C
:100040000C080D048F008F090D080E04043A8F046C
:100050000C080D050E058F040F0C023985001328BE
:1000600066309200F9309100000000000000910B12
:0A0070003628000092083228080029
: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	
	Коннекторы		
X1	DB9F	1	
X2	DS-213B	1	
X3	СР50-73ФВ	1	
	Резисторы		
R1	МЛТ 0,125-3,3к-5%	1	
R2	МЛТ 0,125-330-5%	1	
	Кварцевый резонатор		
ZQ1	4.000MHzHC-49U	1	
	Светодиод		
VD1	ARL-3514UYD-150MCD	1	желтый
	Выключатель		
SB1	TS-A3PS-130 SWT6	1	кнопка тактовая

Перв. примен.

Справ. №

Подп. и дата

Инв. № дубл.

Взам. инв. №

Подп. и дата

Инв. № подл.

Изм.	Лист	№ докум.	Подп.	Дата
Разраб.				
Пров.				
Т. контр.				
Н. контр.				
Утв.				

РП1.ХХХХХХ.002 Э3

Модуль цифровой
Схема электрическая принципиальная

Лит.	Масса	Масштаб

Лист 1 Листов 1

МГТУ им.Н.Э.Баумана
Кафедра РЛ-1
Группа РЛ2-51

Копировал Формат А4

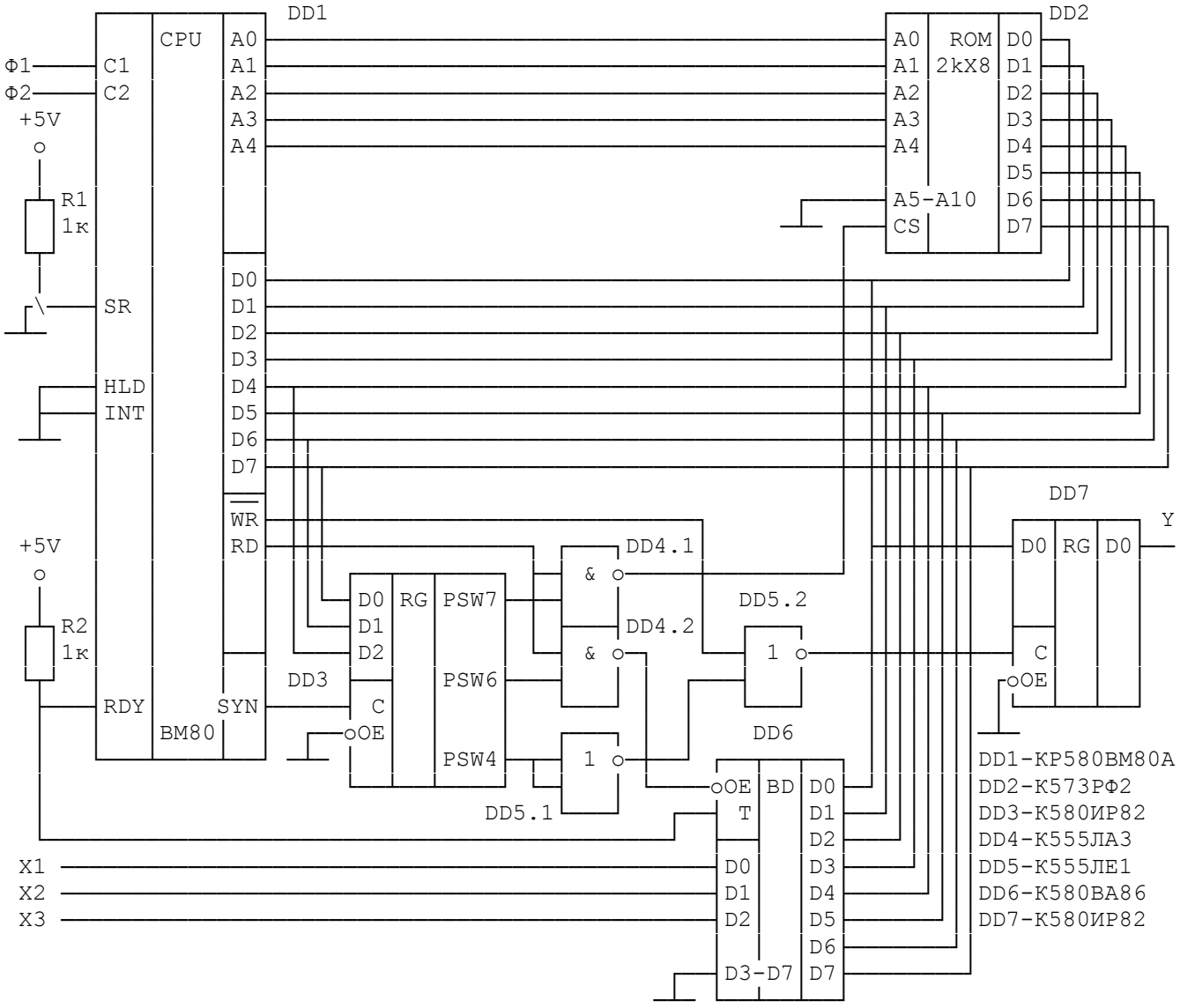
ПРИМЕР №2: Задана в виде таблицы истинности логическая функция Y трех переменных X1, X2, X3

X3	X2	X1	Y
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

$$Y = \overline{X3}\cdot\overline{X2} + \overline{X2}\cdot\overline{X1} + X3\cdot X2\cdot X1 = \overline{X2}\cdot(\overline{X3}+\overline{X1}) + X3\cdot X2\cdot X1 =$$

$$= \overline{X2}\cdot\overline{X3\cdot X1} + X3\cdot X2\cdot X1 = \overline{X2+X3\cdot X1} + X3\cdot X2\cdot X1$$

Для выполнения поставленной задачи выбран микропроцессор КР580ВМ80А.



Пример построения МП – устройства на основе МП КР580ВМ80А

Рассмотрим принципиальную схему МП - системы, реализующей заданную функцию под управлением составленной 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 шести младшим линиям шины адреса микропроцессора, что вполне достаточно для выборки 40 байт 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.

Входные сигналы поступают по линиям X1, X2, и X3 на вход шинного формирователя DD6, образующего с элементом DD4.2 порт ввода. Выходной сигнал поступает на линию Y с выхода регистра DD7, образующего с элементами DD5 порт вывода.

В начале каждого цикла на шину данных МП выводится слово состояния процессора. 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 RC, 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 RC на выходе DD4.1 появляется уровень логического 0, Z - состояние снимается и байт данных поступает из ПЗУ в МП.

В процессе выполнения цикла чтения из порта IN (port) 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 X1, X2 и X3 на четыре младших разряда шины данных.

В процессе выполнения цикла записи в порт OUT (port) в разряде PSW4 появляется 1. При активном нулевом сигнале TR на выходе элемента DD5.2 появляется лог. 1, которая переводит регистр DD7 в режим записи информации с шины данных. Бит D0 с шины данных поступает на выходную линию Y 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.

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

Входные сигналы будем считывать через порт ввода
с адресом 00H

0	0	0	0	0	X3	X2	X1

Выходные сигналы будем выводить через порт вывода
с адресом 01H

0	0	0	0	0	0	0	Y

	Действие	оператор	код	мт
1.	Загрузить данные из порта ввода в A	IN 00H	D8 00	10
2.	Сохранить слово данных в рег. E	MOV E,A	5F	5
3.	Выделить X3: (A) AND 00000100 (Обнуляются все разряды, кроме 2)	ANI 04H	E6 04	7
4.	Сдвиг (A) вправо	RRC	0F	4
5.	Сдвиг (A) вправо	RRC	0F	4
6.	Сохранить X3 в рег. B: B <- (A)	MOV B,A	47	5
7.	Восстановить слово данных в A	MOV A,E	7B	5
8.	Выделить X2: (A) AND 00000010 (Обнуляются все разряды, кроме 1)	ANI 02H	E6 02	7
9.	Сдвиг (A) вправо	RRC	0F	4
10.	Сохранить X2 в рег. C: C <- (A)	MOV C,A	4F	5
11.	Восстановить слово данных в A	MOV A,E	7B	5
12.	Выделить X1: (A) AND 00000001 (Обнуляются все разряды, кроме 0)	ANI 01H	E6 01	7
13.	Сохранить X1 в рег. D: D <- (A)	MOV D,A	57	5
14.	Конъюнкция X1·X2:	ANA C	A1	4
15.	Конъюнкция X1·X2·X3:	ANA B	A2	4
16.	Сохранить X1·X2·X3 в рег. L:	MOV L,A	6F	5

17. Поместить X1 в A:	MOV A,D	7A	5
18. Конъюнкция X1·X3:	ANA B	A2	4
19. Дизъюнкция X2+X1·X3:	ORA C	B1	4
20. Инверсия $\overline{X2+X1\cdot X3}$: ($\bar{A}$)	CMA	2F	4
21. Дизъюнкция $\overline{X2+X1\cdot X3}$+X1·X2·X3:	ORA L	B5	4
22. Вывод Y в порт вывода:PORT <- (A)	OUT 01H	D3 01	10
23. Перейти на начало программы	JMP 0000H	C3 00 00	10
		итого	127

При тактовой частоте 2 МГц длительность одного машинного такта 0,5 мкс, следовательно программа будет выполняться 63,5 мкс и займет в памяти 30 байт.

Прошивка ПЗУ

Адрес	Коды команд															
00	D8	00	5F	E6	04	0F	0F	47	7B	E6	02	0F	4F	7B	E6	01
10	57	A1	A2	6F	7A	A2	B1	2F	B5	D3	01	C3	00	00	FF	FF

.