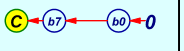
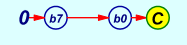
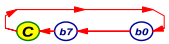
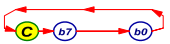
Сводная таблица команд PIC12/16 XLP улучшенной группы микроконтроллеров с аналогиями MCS51 & AVR Tiny&Mega.

ADDWF f,d
ADD Rn,A
ADD A,Rn
JMP PCLATH:(PCL+A)
	SUBWF f,d
	SUB Rn,A
	MOV B,Rn
	SUB B,A
	XCH A,B
	JMP PCLATH:(PCL-A)
		ANDWF f,d
		AND Rn,A
		AND A,Rn
		JMP PCLATH:(PCL AND A)
			IORWF f,d
			OR Rn,a
			OR a,Rn
			JMP PCLATH:(PCL OR A)
				XORWF f,d
				XOR Rn,A
				XOR A,Rn
				JMP PCLATH:(PCL XOR A)
					MOVF f,d R-R
					MOV Rn,Rn (MOV PORTn,PINn) (TEST f)
					MOV A,Rn (MOVFW f)
					(MOV A,@FSR)
						MOVWF f ✱
						MOV Rn,A
						MOVWF INDF = MOV @FSR,A
						JMP PCLATH:A
							COMF f,d R-R
							CPL Rn
							MOV A,Rn
							CPL A
							JMP PCLATH:(NOT PCL)
								CLRF f R-R
								CLR Rn
								CLR INDF = CLR @FSR
								JMP PCLATH:0x00
									CLRW
									CLR A

ADDLW k8 ADD A,#k8	**SUBLW k8** SUB #k8,A !!! результат в регистре A !!!	**ANDLW k8** AND A,#k8	**IORLW k8** OR A,#k8	**XORLW k8** XOR A,#k8	**MOVLW k8** ✱ MOV A,#k8	**MOVLB k4** ✱ MOV BSR,#k4 (0<k4<15)	**MOVLP k7** ✱ MOV PCLATH,#k7 (0<k7<127)	**ADDFSR m,k6** ✱ ADD FSRmH:FSRmL,#k6 (0<m<1) (-31<k6<31)	**ASRF f,d** R-R  ASRF Rn MOV W,Rn ASRF W

ADDWFC f,d
ADDC Rn,A
ADDC A,Rn
	SUBWFB f,d
	SUBB Rn,A
	MOV B,Rn
	SUBB B,A
	XCH A,B
		DECF f,d R-R
		DEC Rn
		MOV A,Rn
		DEC A
			INCF f,d R-R
			INC Rn
			MOV A,Rn
			INC A
				DECFSZ f,d R-R ✱
				DJZ Rn,rstep
				MOV A,Rn
				DJZ A,rstep
					INCFSZ f,d R-R ✱
					INC Rn
					JZ rstep
					MOV A,Rn
					INC A
					JZ rstep
						BCF f,b R-R ✱
						CLR Rn.bit
						BCF INDF = CLR @FSR.bit
							BSF f,b R-R ✱
							SET Rn.bit
							BSF INDF = SET @FSR.bit
								LSLF f,d R-R
								
								LSLF Rn
								MOV W,Rn
								LSLF W
									LSRF f,d R-R
									
									LSRF Rn
									MOV W,Rn
									LSRF W

GOTO addr ✱ JMP PCLATH<6:5:4:3>:addr	**CALL addr** ✱ CALL PCLATH<6:5:4:3>:addr	**RETLW k8** ✱ MOV A,#k8 RET	**RETURN** ✱ RET	**RETFIE** ✱ RETI	**SWAPF f,d** R-R ✱ SWAP Rn MOV A,Rn SWAP A JMP PCLATH:(SWAP PCL)	**BTFSC f,d** R-R ✱ JNB Rn.bit,rstep BTFSC INDF = JNB @FSR.bit,rstep	**BTFSS f,d** R-R ✱ JB Rn.bit,rstep BTFSS INDF = JB @FSR.bit,rstep	**RLF f,d** R-R  RLC Rn MOV A,Rn RLC A	**RRF f,d** R-R  RRC Rn MOV A,Rn RRC A
BRA adr ✱ SJMP adr (-256<adr<255)	**CALLW** ✱ CALL PCLATH:A	**BRW** ✱ JMP PC+A	**RESET** software device RESET	**NOP** ✱ NOP	**CLRWDT** WDR	**SLEEP** SLEEP	**TRIS** ✱ MOV TRIS(r),A	**OPTION** ✱ MOV OPTION,A	

MOVIW ++FSRn
INC FSRn
LD A,@FSRn
(0<n<1)
	MOVIW --FSRn
	DEC FSRn
	LD A,@FSRn
	(0<n<1)
		MOVIW FSRn++
		LD A,@FSRn
		INC FSRn
		(0<n<1)
			MOVIW FSRn--
			LD A,@FSRn
			DEC FSRn
			(0<n<1)
				MOVIW k6[FSRn]
				ADD FSR,#k6
				LD A,@FSRn
				(0<n<1)
				(-31<k<31)					

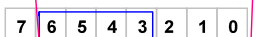
MOVWI ++FSRn ✱ INC FSRn ST @FSRn,A (0<n<1)	**MOVWI --FSRn** ✱ DEC FSRn ST @FSRn,A (0<n<1)	**MOVWI FSRn++** ✱ ST @FSRn,A INC FSRn (0<n<1)	**MOVWI FSRn--** ✱ ST @FSRn,A DEC FSRn (0<n<1)	**MOVWI k6[FSRn]** ✱ ADD FSR,#k6 ST @FSRn,A (0<n<1) (-31<k<31)					

k8 = 0 - 255
d = 0 (w) или 1 (f)
(по умолчанию d=1)

14bit
0<f<0x7F (127)
-256<adr<255
0<addr<2047
0<bit<7
5<r<7
0<n<1
rstep=пропуск команды

обращение к INDF
в качестве f(Rn)
выполняет обращение
к регистру,
адрес которого находится
в соответствующем регистре FSR
... INDF,d = ... @FSR,d

запись в PCL вызывает автоматическую загрузку PCLATH ->> PCL

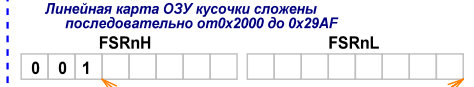


при GOTO или CALL загружается только адрес страницы

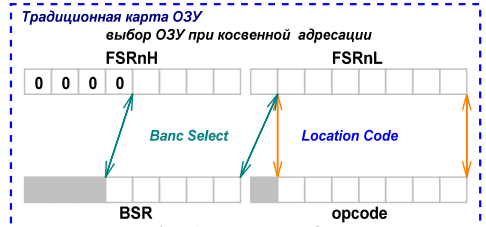
Линейная карта ПЗУ (для младших 8ми бит)
диапазон отображения от 0x8000 (0x0000 ПЗУ)
до 0xFFFF (0x7FFF ПЗУ)



Линейная карта ОЗУ кусочки сложены
последовательно от0x2000 до 0x29AF



Традиционная карта ОЗУ
выбор ОЗУ при косвенной адресации



выбор ОЗУ при прямой адресации

Регистр W представлен в РСФ как РОН области core registers !

Rn=F=0-127 - адрес регистра в текущем банке (стандартная прямая адресация)

Обращение к INDF равноценно обращению к ячейке ОЗУ, адрес которой указан в FSR
(@FSR для mcs51)

Варианты ассемблерной мнемоники:
КОП регисстр,1 = КОП регистр,F = КОП регистр,f = КОП регистр (по умолчанию)
КОП регистр,0 = КОП регистр,W = КОП регистр,w
где "регистр" = объявленное имя регистра или его адрес в текущем банке ОЗУ

шестнадцатеричное :	h'9f'=0x9F
десятичное :	d'100'=.100
восьмеричное :	o'77'
двоичное :	b'01010011'
ASKII :	a'c'='c'

ПРИ ВЫЧИТАНИИ:	
Z=0, C=1	результат положительный
C=Z=1	результат равен 0
C=Z=0	результат отрицательный

STATUS					Z	AC	C
-	-	-	/TO	/TD	Z	DC	C

Когда регистр ввода/вывода используется для модификации самого себя (MOVF PORTn,1) то для записи будут использованы значения с выводов, а не из защелок порта. /только если порт сконфигурирован как выходной/

MOVF и MOVIW модифицируют Z согласно содержимого регистра назначения !!!

rstep = переход на +1 команду (AVR)

✱ - на флаги не воздействует
- расширенная система команд
- аналог при d=1 (->f)
- при Rn=INDF (d=1)
- при d=1 адресат = PCL
- аналог при d=0 (->w)
- без альтернативы (A или Rn)

R-R — данная операция может выполняться непосредственно с регистром без участия акумулятора

Для операций табличного чтения данных и относительных вычисляемых переходов за нулевой адрес (смещение = 0) принимается адрес ячейки памяти, следующей за командой, модифицирующей PCL !!! addwf PCL,f при W=0xFF = "вечный цикл" !!!
Применение команд, модифицирующих PCL требует особого внимания к размещению команд в адресном пространстве и величине результирующего значения в PCL, так как требуется корректное значение результата данных в паре PCLATH4-0:PCL
!!! чтение PCL содержимого PCLATH не изменяет !!!