ЎЗБЕКИСТОН РЕСПУБЛИКАСИ ОЛИЙ ВА ЎРТА МАХСУС
ТАЪЛИМ ВАЗИРЛИГИ

ФАРҒОНА ПОЛИТЕХНИКА ИНСТИТУТИ

«Электроника ва асбобсозлик»
кафедраси

«Микропроцессор техникаси»

фанидан
амалий машғулотлар учун

УСЛУБИЙ ҚЎЛЛАНМА

ФАРҒОНА – 2012 й

ЎЗБЕКИСТОН РЕСПУБЛИКАСИ
ОЛИЙ ВА ЎРТА МАХСУС ТАЪЛИМ ВАЗИРЛИГИ

ФАРҒОНА ПОЛИТЕХНИКА ИНСТИТУТИ

«ЭЛЕКТРОНИКА ВА АСБОБСОЗЛИК»
кафедраси

«Микропроцессор техникаси» фанидан

амалий машғулотлар учун

УСЛУБИЙ ҚЎЛЛАНМА

ФарПИ услубий кенгашида тасдиқланган
Баён № _____
«___» _____ 2012 й.

ФАРҒОНА – 2012 й

Микропроцессор техникаси фанидан мавжуд ўқув дастурига киритилган янгиликларни, замонавий рақобатбардош техника ва технологияларни талабалар томонидан мукаммал ўзлаштирилишини таъминлаш учун ушбу услубий қўлланма 2011-2012 ўқув йили киритилган ишчи дастур асосида тайёрланди.

Бу кўрсатма «Микропроцессор техникаси» фанини ўрганувчи талабалар учун мўлжалланган бўлиб, замонавий рақобатбардош электрон қурилмалар яратиш, уларни дастурлаш технологияларини мукаммал билган мутахасислар тайёрлаш мақсадида рақамли қурилмалар ва микроконтроллерлар ҳамда уларни дастурлаш технологиялари билан таништирувчи материаллар келтирилган. Услубий қўлланмада дастурлаш ва дастурни созлаш ишларини бажариш учун Протеус тизимида эмулятор ва симуляторлардан фойдаланиш тартиби, PIC micro микроконтроллерларининг командалар тизимида тузилган дастур ва дастур фрагментларини ишлатиб кўриш тартиби, микроконтроллерли қурилмаларни лойиҳалаш ишларини бажариш тартиблари ва талабалар билимини текшириш учун назорат саволлари келтирилган.

Қўлланма ўқув жараёнига электрон қурилмаларни профессонал лойихаловчисининг ПРОТЕУС номли автоматлаштирилган иш ўрни тизимини киритилганлиги муносабати билан ана шу тизим асосида талабалар замонавий микроконтроллерлар ва уларни дастурлаш воситалари билан таништириш имкониятига ҳамда лойиҳалаш жараёнида компютер ва замонавий оргтехника воситаларидан фойдаланишнинг амалий кўникмаларига эга бўладилар.

Ушбу услубий қўлланма «ЭМЭ» кафедраси йиғилишида кўриб чиқилган
Баён №_____ «_____»_____2012 й.

Ушбу услубий қўлланма Энергетика факультети кенгашининг 2012 йил “ ”_____ даги ____ - сонли қарори билан тасдиқланди.

Кенгаш раиси: Э.Юнусалиев

Тузувчи: т.ф.н., доц. Н.К.Умаралиев

PIC серияли микроконтроллерларнинг асосий қурилмалари

Microchip компаниясининг PIC (Peripheral Interface Controllers – ташқи интерфейс контроллерлари) серияли микроконтроллерлари барча илғор технологяларни мужассамлантирганлиги, фойдаланувчи томонидан электр усулида программалаштирилувчи ППЗУ, минималь энергоистеъмоли, юқори тезликли, ривожланган RISC-архитектурали, функциональ тугалланганлиги ва минималь размерлари билан ажралиб туради.

Microchip Technology Inc. компанияси назорат ва бошқариш тизимлари учун электрон компонентлар ишлаб ишлаб чиқаришга ихтисослашган. Асосий ишлаб чиқарилаётган махсулотлар:

- 8 – разрядли универсаль микроконтроллерлар (PICmicro™ MCU);
- Ихтисослаштирилган хотира микросхемалари;
- Фойдаланишни чекловчи қурилмалар (KeeLoq);
- Лойихалашнинг техник ва дастурий воситалари.

Бугунги кунда PICmicro микроконтроллерларининг архитектураси электрон компонентларнинг дунё бозорида энг кўп тарқалган уч хил архитектурадан бири – Гарвад архитектурасидир.

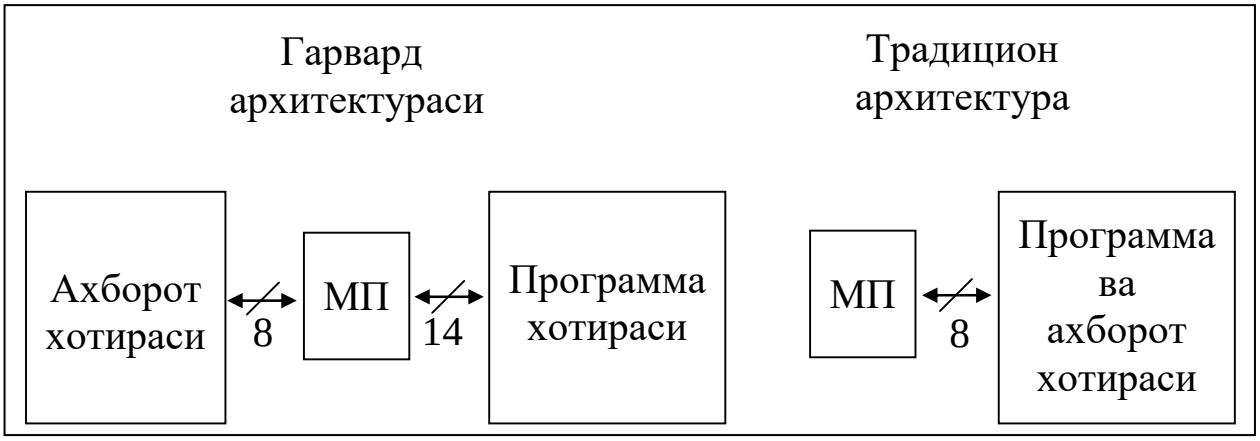
Гарвад архитектурасида программа хотираси ва ахборот хотираси алохида. Бу хотираларга алохида шиналар орқали мурожат қилинганлиги туфайли унумдорлик традицион архитектурали процессорларга нисбатан кескин юқоридир.

Традицион архитектурали микроконтроллерларда аборот ва командалар битта ахборот шинаси орқали ўқилади, бу ўз навбатида ахборот шинасини катта нагрузка билан ишлашига олиб келади.

Гарвард архитектурали микроконтроллерларда команда битта циклда ўқилади (ҳамма командалар узунлиги бир ҳил – 14 разрядли). Бу вақтда ахборот хотирасидан ўқиш ёки ёзиш мумкин, чунки бу шина алоҳида.

Гарвард ва традицион архитектурали микроконтроллерларни 1 – расмда солиштирилган:

1- расм. Гарвард ва традицион архитектуранинг фарқи



PICmicro микроконтроллерларининг қуйидаги бир неча афзалликларини таъкидлаш мумкин:

- Қурилма лойихалаш вақтининг озлиги;
- Махсулот чиқариш жараёнида программа кодини ўзгартириш мумкинлиги;
- Программа кодини ўзгартиришнинг орзонлиги (қолипни ўзгартириш талаб этилмайди);
- Махсулотни номерлашнинг қулайлиги;
- Созлаш жараёнига ишлатиладиган ахборот(калибровочная информация)ни сақлаш;
- Лойихалашга ишлатилган микросхемани тайёр қурилмада ҳам ишлатиш мумкинлиги.

Барча PICmicro микроконтроллерни командалар разрядига кўра уч группага бўлиш мумкин:

1. Базавий оила: командалари 12-разрядли.
2. Ўрта оила: командалари 14-разрядли.
3. Катта оила: командалари 16-разрядли.

Ушбу қўлланма PIC16CXXX – ўрта оила микроконтролларига тегишли.

Микроконтроллерлар структураси

Микроконтроллернинг ҳар бир қисми қуйидаги уч группага тегишли бўлиши мумкин:

1. Микроконтроллер ядроси;
2. Периферия модули;
3. Микроконтроллернинг махсус қисми.

Микроконтроллер ядроси

Ядро асосий қисм бўлиб, у микроконтроллерни ишлашга мажбур қилади.

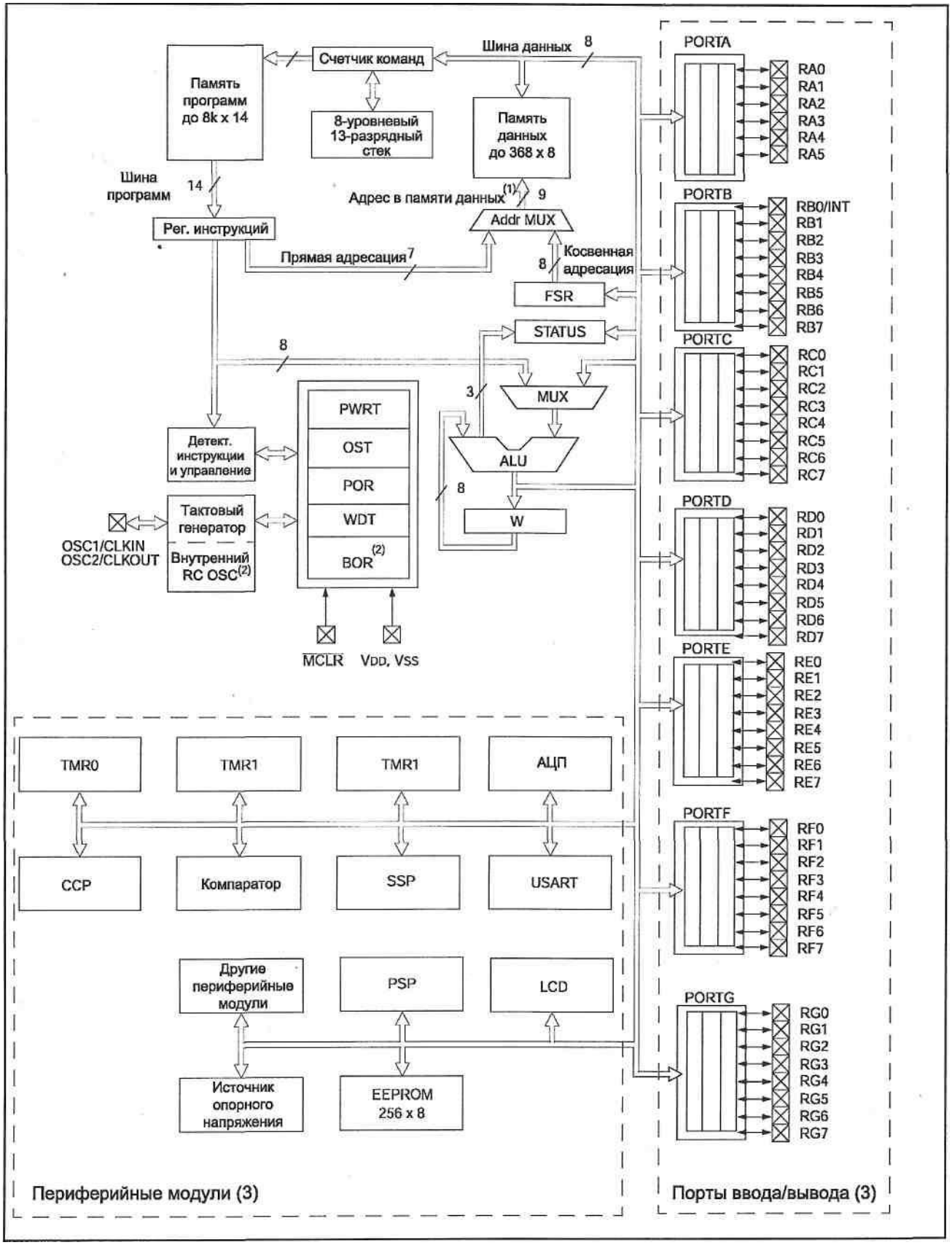
Бу гурух таркибига қуйидагилар киради:

1. Такт генератори
2. Дастлабки ҳолатга қайтиш схемаси (Логика*сброса)
3. Марказий процессор (CPU)
4. Арифметик-логик қурилма (АЛУ)
5. Командалар хотира қурилмаси
6. Ахборот хотира қурилмаси
7. Узилишлар тизими (Прерывания)
8. Командалар тизими

Периферийя модуллари

Периферийя модуллари ташқи схемалар билан алоқа интерфейсини ташкил қилиш имконини беради. Масалан, киритиш – чиқариш универсаль портлари, суюқ кристалли индикаторлар (ЖКИ) драйверлари, АЦП киришлари, ШИМ чиқишлари ва вақт интерваллларини ҳисоблаш қурилмалари (таймерлар).

2 – расм. *PICmicro ўрта оила микроконтроллерларнинг умумий структура схемаси*



Қуйида ишлатилиши мумкин бўлган периферийя модуллари келтирилган:

1. Универсаль киритиш/чиқариш портлари PORTA – PORTG
2. Таймерлар TMR0, TMR1, TMR2
3. Захват/Сравнение/ШИМ (CCP)
4. Синхрон кетма-кет порт (SSP)
5. Асосий синхрон кетма – кет порт (SSP)
6. Етакловчи синхрон кетма – кет порт (MSSP)
7. USART
8. Таянч кучланиш манбаси
9. Компараторлар
10. 8-разрядли АЦП
11. Асосий 8-разрядли АЦП
12. 10-разрядли АЦП
13. Интегралловчи АЦП
14. ЖКИ драйвери
15. Етакланувчи параллель порт (PSP)

Микроконтроллерларнинг махсус қурилмалари:

1. Конфигурация битлари
2. Манба уланганда ишловчи интегралланган “Сброс” схемаси (POR)
3. Манба кучланиши пасайишидан ишловчи “Сброс” схемаси (BOR)
4. Сторожевой таймер
5. Энергия иқтисод режими (SLEEP)
6. Интегралланган (ички) RC такт генератори
7. Ўз – ўзини (ички схемали) (внутрисхемное) программалаштириш

Лойихачиларга ёрдам

Microchip компанияси лойихаловчиларга программа кодини эффектив яратиш ва созлаш имконини берувчи инструменталь воситаларининг кенг номенклатурасини таклиф этади. Бу инструменталь воситаларини шартли равишда тўртта категорияга бўлиш мумкин:

1. Объект кодни генерациялаш;

2. Программа созловчилар;
3. Программаторлар;
4. Демонстрацион платалар.

Microchip компаниясида яратилган барча инструменталь воситалар, MPLAB IDE номли лойихалашнинг интегралланган мухити бошқарувида ишлайдилар.

Объект кодни генерацияловчи инструменталь воситалар таркиби:

- MPASM;
- MPLAB-C;
- MP-DriveWay™.

Бу программалар таркибига ҳар бир микроконтроллер учун қўшимча файллар киради. Бу файлларда регистрлар номлари адреси ёки бит номери кўрсатилиб аниқланади

Программаларни созлаш учун инструменталь воситалар таркиби:

- PICMASTER ички схема эмулятори;
- ICEPIC ички схема эмулятори;
- MPLAB-SIM дастурий симулятори.

Программа кодини созлагач, бу программани микроконтроллер хотирасига киритиш керак. Бунинг учун Microchip турли даражадаги икки хил программатор таклиф этади:

- PICSTART;
- PROMATEII.

Демонстрацион платалар лойихаловчиларга микроконтроллерларни конкрет масала учун қўллаш мумкинлигини бахолаш имкони беради. Таклиф этилаётган демонстрацион платалар:

- PICDEM-1;
- PICDEM-2 .....

Лойихалашнинг ҳар бир инструменталь воситаси ҳақидаги тўлиқ маълумотни Microchip компаниясининг Web сахифалари **www.microchip.com** ва **www.microchip.ru** дан олиш мумкин**.**

Командалар тизими

Ҳар бир команда битта 14 – разрядли сўздан иборат бўлиб, команда типини аниқловчи код операция (OPCODE) дан ва команда операциясини аниқловчи бир ёки бир неча операндлардан ташкил топади. Командаларнинг тўлиқ рўйхати 1 таблицада келтирилган.

Аккумулятор типидаги командалар ортогональ (симметрик) бўлиб, учта асосий группага бўлинади:

- Байт устида амал бажарувчи командалар;
- Бит устида амал бажарувчи командалар;
- Бошқариш командалари ва константалар билан амал бажарувчи командалар.

Операция коди форматини 2 таблицадан кўринг.

Байт устида амал бажарувчи командалар учун ‘f ‘ регистр кўрсаткичидир, 'd' натижа адресини кўрсатгичидир. Регистр кўрсаткичи командада қайси регистр ишлатилишини аниқлайди. Натижа адресини кўрсатгичи натижани қаерда сақланишини аниқлайди. Агар 'd'=0 бўлса, натижа W регистрида сақланади. Агар 'd'=1 бўлса, натижа командада ишлатилган регистрда сақланади.

Бит устида амал бажарувчи командаларда 'b' операцияда иштирок этувчи бит номерини аниқлайди, ' f ' - бўлса шу бит қайси регистрдаги ахборотга тегишли эканлигини кўрсатади.

Бошқариш командалари ёки константа билан амал бажарувчи командаларда 'k' саккиз ёки ўнбир битли константани ёки литераллар қийматини кўрсатади.

Шартли командалардан ташқари барча командалар бир машина циклида бажарилади. Шартли командаларда шарт бажарилган холда ва PC команда кўрсатгичини ўзгартирувчи инструкция хосил бўлган холда командалар икки машина циклида бажарилади. Икки машина циклида бажарилувчи команда бажарилаётганда иккинчи машина циклида NOP (операция йўқ) инструкцияси бажарилади. Бир машина цикли тўрт тактдан иборат бўлади. Такт генератори частототаси 4 МГц бўлса, барча командалар 1мкс да бажарилади, Шартли командаларда шарт бажарилса ёки PC команда кўрсатгичи ўзгарса, команда 2мкс вақтда бажарилади.

1 – *Таблица. Ўрта оиладаги* микроконтроллерлар командалари рўйхати

Командалар мнемокоди	Тарифи	Цикл	14-разрядли код	Флаг ўзгариши	Изох
			Бит 13 | Бит 0		
Байт устида амал бажарувчи командалар					
ADDWF f,d	W ва f ни қўшиш	1	00 0111 dfff ffff	C,DC,Z	1,2
ANDWF f,d	Битлар бўйича W 'ВА' f	1	00 0101 dfff ffff	Z	1,2
CLRF f	f ни тозалаш	1	00 0001 1fff ffff	Z	2
CLRW	W ни тозалаш	1	00 0001 0xxx xxxx	Z	
COMF f,d	f ни инвертирлаш (инкорлаш)	1	00 1001 dfff ffff	Z	1,2
DECF f,d	f дан 1ни айириш (декремент)	1	00 0011 dfff ffff	Z	1,2
DECFSZ f,d	f дан 1ни айириш ва 0 бўлса ўтказиб юбориш	1(2)	00 1011 dfff ffff		1,2,3
INCF f,d	f га 1 ни қўшиш (инкремент)	1	00 1010 dfff ffff	Z	1,2
INCFSZ f,d	f га 1 ни қўшиш ва пропустить агар 0	1(2)	00 1111 dfff ffff		1,2,3
IORWF f,d	Битлар бўйича W 'ВА' f	1	00 0100 dfff ffff	Z	1,2
MOVF f,d	f ни узатиш	1	00 1000 dfff ffff	Z	1,2
MOVWF f	W ни f га узатиш	1	00 0000 lfff ffff		
NOP	Операция йўқ	1	00 0000 0xx0 0000		
RLF f,d	f ни чапга перенос орқали циклик силжитиш	1	00 1101 dfff ffff	C	1,2
RRF f,d	f ни ўнгга перенос орқали циклик силжитиш	1	00 1100 dfff ffff	C	1,2
SUBWF f,d	W дан f ни айириш	1	00 0010 dfff ffff	C, DC, Z	1,2
SWAPF f,d	f регистрдаги ярим байтларни ўзаро ўрнини алмаштириш	1	00 1110 dfff ffff		1,2
XORWF f,d	Бит бўйича: W 'Тенгсизлик' f	1	00 0110 dfff ffff	Z	1,2
Бит устида амал бажарувчи командалар					
BCF f,b	f регистридаги b битни тозалаш	1	01 00bb bfff ffff		1,2
BSF f,b	f регистридаги b битни ўрнатиш	1	01 01bb bfff ffff		1,2
BTFSC f,b	f регистридаги b битни текшириш, агар 0 бўлса кейинги команда бажарилмасдан ўтказиб юборилсин	1(2)	01 10bb bfff ffff		3
BTFSS f,b	f регистридаги b битни текшириш, агар 1 бўлса кейинги команда бажарилмасдан ўтказиб юборилсин	1(2)	01 11bb bfff ffff		3
Бошқариш командалари ва константалар билан операциялар					
ADDLW k	Константани W билан қўшиш	1	11 111x kkkk kkkk	C,DC,Z	
ANDLW к	Бит бўйича: Константа 'ВА' W	1	11 1001 kkkk kkkk	Z	
CALL к	Подпрограммани чақириш	2	10 0kkk kkkk kkkk		
CLRWDT -	WDT ни тозалаш	1	00 0000 0110 0100	-TO, -PD	
GOTO к	Шартсиз ўтиш	2	10 1kkk kkkk kkkk		
IORLW к	Бит бўйича: Константа 'ЁКИ' W	1	11 1000 kkkk kkkk	Z	
MOVLW к	Константани W га жўнатиш	1	11 00xx kkkk kkkk		
RETFIE	Подпрограммадан қайтиш (узилишга рухсат билан)	2	00 0000 0000 1001		
RETLW к	Константани W га юклаб подпрограммадан қайтиш	2	11 01xx kkkk kkkk		
RETURN	Подпрограммадан қайтиш	2	00 0000 0000 1000		
SLEEP	SLEEP режимига ўтиш	1	00 0000 0110 0011	-TO, -PD	
SUBLW к	Константадан W ни айириш	1	11 110x kkkk kkkk	C,DC,Z	
XORLW к	Бит бўйича: Константа 'Тенгсизлик' W	1	11 1010 kkkk kkkk	Z	

Изох:

1. Киритиш/Чиқариш порти билан "Ўқиш - модификациялаш - ёзиш" операцияси бажарилганда (масалан MOVF PORTB,1) ахборот чиқиш тириггеридан эмас балки портнинг чиқишларидан ўқилади. Масалан, чиқиш тириггерига '1' ёзилган бўлсаю, портнинг чиқишида қуйи даражали сигнал бўлса, портга '0' ни қиймати қайта ёзилади.
2. TMR0 га ёзиш бажарилганда (ва d=1 бўлса) TMR0 предделители TMR0 модулига уланган бўлса, у дастлабки холатга қайтарилади.
3. Шартли командаларда шарт бажарилган холда ва PC команда кўрсатгичини ўзгартирувчи инструкция хосил бўлган холда командалар икки машина циклида бажарилади. Иккинчи циклда NOP командаси бажарилади.

Изох 1. Барча реализация қилинмаган операция кодлари кейинги ишланмаларга мўлжалланган. Хужжатлаштирилмаган операция кодини ишлатиш олдиндан айтиб бўлмайдиган оқибатларга олиб келиши мумкин.

Изох 2. Тузаётган дастурингиз PICmicro микроконтроллерларининг кейинги версияларида ҳам муаммосиз ишлаши учун TRIS ва OPTION инструкцияларидан фойдаланманг !

2. Командалар формати

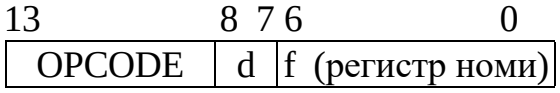
3 расмда уч группа командaнинг формати кўрсатилган. Команда коди 3 дан то 6 битгача бўлиб, 35 та командани реализация қилиш имконини беради.

Ушбу қўлланмада келтирилган барча мисолларда қуйидаги ўнолтилик сон формати ишлатилади: 0xhh, бу ерда h – ўнолтилик рақам.

Иккилик сонни ифодаси қуйидагича: 00000100b, бу ерда b – иккилик сон кўрсаткичи.

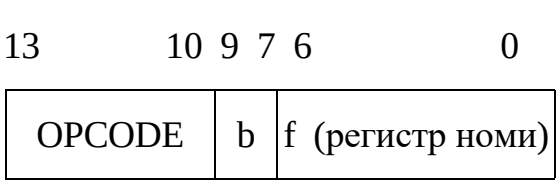
3 – расм. Ўрта авлод микроконтроллерлари командалари формати.

Байт устида амаллар



d = 0 - натижа **w** да сақланади
d = 1 - натижа **f** да сақланади
f – 7 – разрядли адрес (регистрнинг)

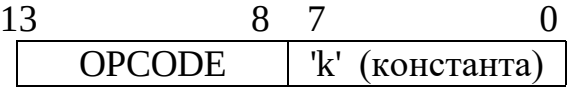
Бит устида амаллар



b - 3-разрядли номер (регистрдаги бит номери)
f – 7 – разрядли адрес (регистрнинг адреси)

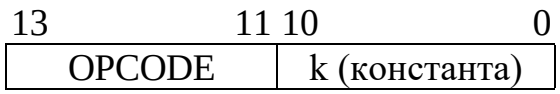
Бошқариш командалари ва константалар билан амаллар

Умумий хол:



k - 8-разрядли сон (қиймат)

Фақат CALL ва GOTO инструкциялари учун



k – 11-разрядли сон (қиймат)

Таблица -2 Операция коди майдонларининг тарифи

Майдон	Таърифи
f	Регистр адреси (0x00 дан 0x7F гача)
w	Ишчи региср (аккумулятор)
b	8-разрядли регистрдаги бит номери
k	Константа (сон ёки белги)
X	Фарқи йўқ (0 / 1). Ассемблер x=0 генерациялайди.
d	Операция натижасининг адрес кўрсаткичи: d = 0 бўлса натижа w регистрига ёзилади; d = 1 бўлса натижа f регистрига ёзилади; кўрсатилмаса d = 1
label	Белги (метка) номи
TOS	Стек боши
PC	Команда кўрсатгичи (хисобчиси – саноқчиси)
PCLATH	Команда кўрсатгичининг катта байти буфери
GIE	Узилишларга глобаль рухсат бити
WDT	Назоратчи таймер
-TO	Флаг переполнения WDT
-PD	Флаг сброса по включению питания
dest	Қабул қилувчи, w регистри ёки хотира регистри
[ ]	Қўшимча параметрлар
( )	Таркиби (ичидаги)
→	Қиймат бериш
< >	Бит майдони
€	Тўпламдан
Курсив	Фойдаланувчи аниқлайдиган термин

3. Махсус регистрлардан фойдаланиш

Ўрта авлод PICmicro микроконтроллерлари командалар системаси барча регистрларга тўғридан – тўғри мурожат қилиш имконини беради. Бази бир регистрлардан фойдаланишда нозик томонлари мавжуд бўлиб, буларни дастур яратувчи алоҳида ҳисобга олиши лозим.

4.1 Командаларни бажаришда STATUS регистри мақсад регистри

Агар STATUS регистридан фойдаланаётган команда натижасида Z, DC ва C флаглар ўзгартирилиши кутилса, бу учта битнинг ўзгариши команда томонидан блокировка қилинган. Бу битлар микроконтроллер ядроси иш мантиқига асосан 1 ёки 0 холатига ўтади. STATUS регистрини ўзгартирувчи командалар -TO ва -PD битларига таъсир этмайди.Шунинг учун STATUS регистри иштирок этган команда натижаси кутилганидан бошқачароқ чиқиши мумкин. Масалан, CLRF STATUS

командаси учта катта битни 0 холатига ва Z битни 1 холатига ўтказиб қўяди. Яъни, STATUS регистрининг холати ушбу команда бажарилганидан сўнг **000uu1uu** кўринишга ўтади, бу ерда **u** ўзгармас бит.

4.2 PCL ахборот манбайи ва мақсад регистри сифатида

PCL регистри билан боғлиқ ***Ўқиш***, ***Ёзиш*** ва "***Ўқиш*** – ***Модификация*** – ***Ёзиш*** " командалари қуйидаги натижаларни ҳосил қилиши мумкин:

PCL дан ***Ўқиш***:	PCL → dest; PCLATH ўзгармайди.
PCL га ***Ёзиш*** –:	PCLATH → PCH; 8 – разрядли қиймат →PCL.
"***Ўқиш*** – ***Модификация*** – ***Ёзиш***":	PCL → АЛУ операнди; PCLATH → PCH; 8 - разрядли қиймат → PCL.

Бу ерда PCH – команда кўрсатгичининг катта байти (адресланмайдиган регистр), PCLATH - команда кўрсатгичининг катта байтининг буфери, dest – мақсад регистри.

4.3 Бит операциялари

Регистрнинг исталган бир бити ўзгартирилганда аввал ахборотлар хотирасидан регистр тўлиқ ўқилади, керакли бит ўзгартирилади ва ахборот қайтдан регистрга ёзилади ("ўқиш - модификация - ёзиш"). Бу фактни баъзи бир махсус регистрлар билан ишлаганда ҳисобга олиб қўйиш керак (масалан, порт регистрлари).

Изох: Бошқарувчи битлар холатини ўзгартириш Q1 тактида бажарилади (узилиш флаглари хам), шунинг учун ушбу битли регистрларга "ўқиш - модификация - ёзиш" командаси бажарилганда ҳеч қандай муаммо юзага келмайди.

Мантиқий амаллар

Айтайлик W регистрида 0000 1111 коди мавжуд
F регистрида 1111 1001 коди мавжуд

Битлар устида “ВА” амали натижаси	0000 1001 бўлади
Битлар устида “ЁКИ” амали натижаси	1111 1001 бўлади

Битлар устида “Тенгсизлик” амали 1111 0110 бўлади

5. Команда бажарилиши тактлари

Командaнинг хар бир цикли (Тсу) тўртта тактдан иборат (Q1-Q4). Q такт давомийлиги бўйича такт генераторининг даврига тенг. (Tosc). Q тактлари бир команда цикли учун декодлаш, ахборотни ўқиш, ахборотни қайта ишлаш, натижани ёзиш ишларини ўта синхрон бажарилишини таъминлайди. Диаграммада Q тактларнинг команда циклига боғланиши кўрсатилган.

Команда цикли (T_{CY}), 4- тактдан иборат бўлиб, умумий холда қуйидагича кўринади:

Q1: Команда тахлили ёки мажбурий бўш операция (NOP)

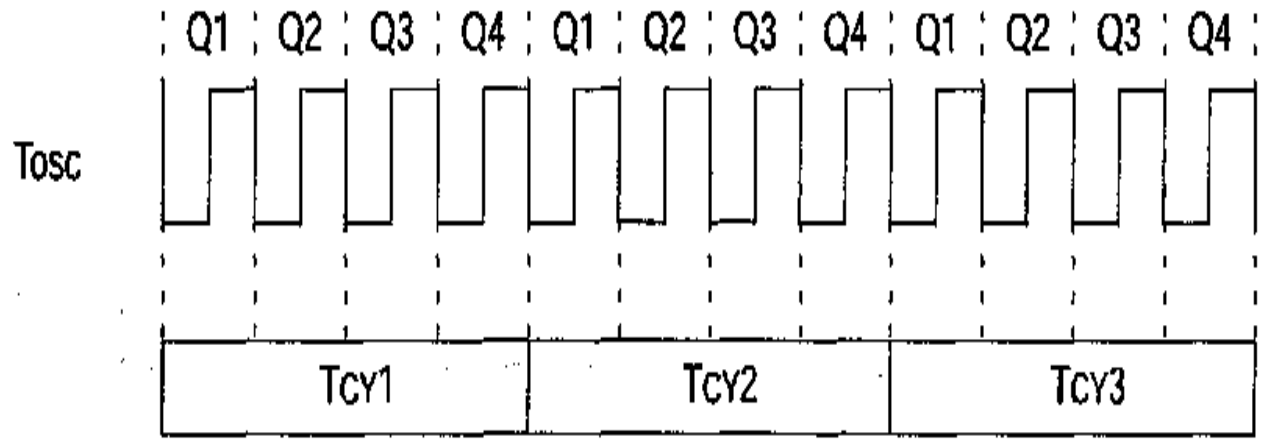
Q2: Ахборотни ўқиш операцияси ёки операция йўқ

Q3: Ахборотни қайта ишлаш

Q4: Ахборотни ёзиш операцияси ёки операция йўқ

Командaнинг Q тактлар бўйича операциялар таркибини командалар таърифидан кўринг.

Расм. 4 Q тактларнинг циклик такрорланиши вақт диаграммаси



6. Командаларнинг таърифи

ADDLW Константани W билан қўшиш

Синтаксис: *[label]* ADDLW

Операндлар: 0 < k < 255

Операция:, (W) + k -> (W)

Флаг ўзгариши: C, DC, Z

Код:

Таърифи:

11	lllx	kkkk	kkkk

Команда узунлиги: 1

Цикллар: 1

Тактлар бўйича командани бажарилиши: W регистрдаги код 8 - разрядли константа 'k' билан қўшилади. Натижа W регистрида сақланади.

Q1	Q2	Q3	Q4
Командни декодлаш	'к' константа-ни ўқиш	Бажариш	W регистрга ёзиш

Мисол 1:

```
ADDLW  0x15
Команда бажарилгунча
        W = 0x10
Команда бажарилгандан сўнг
        W = 0x25
```

Мисол 2:

```
ADDLW  MYREG
Команда бажарилгунча
        W = 0x10
        MYREG = 0x37 (адрес регистра)
Команда бажарилгандан сўнг
        W = 0x47
```

Мисол 3:

```
ADDLW  HIGH (LU_TABLE)
Команда бажарилгунча
        W = 0x10
        LU_TABLE = 0x9375 (программа
        хотирасидаги адрес)
Команда бажарилгандан сўнг
        W = 0xA3
```

ADDWF W ва f ни қўшиш

Синтаксис: *[label]* ADDWF f,d

Операндлар: 0 <f< 127
d € [0,1]

Операция: (W) + (f) -> (dest)

Флаг ўзгариши: C, DC, Z

Код:

00	0111	dfff	ffff

Таърифи: W ва f регистрлардаги сонларни қўшиш.
Агар d=0 бўлса, натижа W регистрида сақланади.
Агар d=1 бўлса, натижа f регистрида сақланади.

Сўзлар: 1
Цикллар: 1

Тактлар бўйича командани бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	' f ' регистрни ўқиш	Бажариш	Натижани ёзиш

Мисол 1:

```
ADDWF  FSR , 0
Команда бажарилгунча      Команда бажарилгандан сўнг
      W = 0x17                  W = 0xD9
      FSR = 0xC2                FSR = 0xC2
```

Мисол 2:

```
ADDWF  INDF, 1
Команда бажарилгунча
      W = 0x17
      FSR = 0xC2
Адреси  FSRда берилган регистр қиймати = 0x20
Команда бажарилгандан сўнг
      W = 0x17
      FSR = 0xC2
Адреси  FSRда берилган регистр қиймати = 0x37
```

Мисол 3:

```
ADDWF   PCL , 0
Хол 1
Команда бажарилгунча      Команда бажарилгандан
      W = 0x10            сўнг
      PCL = 0x37          PCL = 0x47
      C = x               C = 0
Хол 2
Команда бажарилгунча      Команда бажарилгандан
      W = 0x10            сўнг
      PCL = 0xF7          PCL = 0x07
      PCH = 0x08          PCH = 0x08
      C = x               C = 1
```

ANDLW константа ва W ўртасида “Битлар бўйича ВА”

Синтаксис: *[label]* ANDLW k

Операндлар: 0 < k < 255

Операция: (W) .AND. k->(W)

Флаг ўзгаришлари: Z

Код:

11	1001	kkkk	kkkk

Таъриф: W регистрдаги код ҳамда 8 – разрядли 'k' константа ўртасида “Битлар бўйича ВА” опрацияси бажарилади. Натижа W регистрига ёзилади

Сўз: 1

Цикл: 1

Командани тактлар бўйича бажарилиши:

Q1	Q2	Q3	Q4
Командани декодлаш	k' константани ўқиш	Бажариш	W га ёзиш

Мисол 1:

```
ANDLW       0x5F(0101 1111)
Команда бажарилгунча
            W = 0xA3 (1010 0011)
Команда бажарилгандан сўнг
            W = 0x03 (0000 0011)
```

Мисол 2:

```
ANDLW   MYREG
Команда бажарилгунча
            W = 0xA3
            MYREG = 0x37 (регистр адреси)
Команда бажарилгандан сўнг
W = 0x23
```

Мисол 3:

```
ANDLW   HIGH (LU_TABLE)
Команда бажарилгунча
            W = 0xA3
            LU_TABLE  = 0x9375
            (программалар хотираси  адреси)
Команда бажарилгандан сўнг
            W = 0x83
            Яъни
            1010 0011(A3)
            1001 0011(93)
            1000 0011(83)
```

ANDWF W “Битлар бўйича ВА” f

Синтаксис: *[label]* ANDWF f,d

Операндлар: 0<f < 127
D € [0,1]

Операция: (W) .AND. (f) -> (dest)

Флаг ўзгаришлари: Z

Код:

00	0101	dfff	ffff

Таъриф: W ва f регистрлардаги кодлар устида битлар бўйича “ВА” амали бажарилади. Бунда
агар d=0 бўлса, натижа W регистрида сақланади.
агар d=1 бўлса, натижа f регистрида сақланади.

Сўзлар: 1

Цикллар: 1

Тактлар бўйича команда бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	f регистрни ўқиш	Бажариш	Натижани ёзиш

Мисол1: ANDWF FSR , 1

Команда бажарилгунча
W = 0x17 (0001 0111)
FSR = 0xC2(1100 0010)
Команда бажарилгандан сўнг
W = 0x17
FSR = 0x02 (0000 0010)

Мисол2: ANDWF FSR., 0

Команда бажарилгунча
W = 0x17 (0001 0111)
FSR = 0xC2(1100 0010)
Команда бажарилгандан сўнг
W = 0x02 (0000 0010)
FSR = 0xC2

Мисол3: ANDWF INDF,1

Команда бажарилгунча
W = 0x17 (0001 0111)
FSR = 0xC2
Адреси FSR да кўрсатилган регистр қиймати = 0x5A (0101 1010)
Команда бажарилгандан сўнг
W = 0x17
FSR = 0xC2
Адреси FSR да кўрсатилган регистр қиймати = 0x12 (0001 0010)

BCF f регистрдаги b битни тозалаш

Синтаксис: *[label]* BCF f,b

Операндлар: 0 <f< 127

0 < b < 7

Операция: 0 → (f<b>)

Флаг ўзгаришлари: Йўқ

Код:	01	00bb	bfff	ffff

Таъриф: f регистрдаги b битга “0” ёзиш

Сўзлар: 1

Цикллар: 1

Тактлар бўйича команда бажарилиши	Q1	Q2	Q3	Q4
	Командани декодлаш	' f ' регистрни ўқиш	Бажариш	‘ f ‘регистрига ёзиш '

Мисол1:

```
BCF     FLAG_REG,7
Команда бажарилгунча
        FLAG_REG = 0xC7 (1100 0111)
Команда бажарилгандан сўнг
        FLAG_REG = 0x47 (0100 0111)
```

Мисол2:

```
BCF     INDF, 3
Команда бажарилгунча
        W = 0x17
        FSR = 0xC2
        Адреси FSR да кўрсатилган
        регистр қиймати = 0x2F (0010 1111)
Команда бажарилгандан сўнг
        W = 0x17
        FSR = 0xC2
        Адреси FSR да кўрсатилган
        регистр қиймати = 0x27 (0010 0111)
```

BSF f регистридаги b битга “1” ёзиш (ўрнатиш)

Синтаксис: *[label]* BSF f,b

Операндлар: 0<f < 127
0 < b < 7

Операция: 1 →(f<b>)

Флаг ўзгаришлари: Йўқ

Код:

01	01bb	bfff	ffff

Таъриф: f регистридаги b битга “1” ёзиш

Сўзлар: 1

Цикллар: 1

Тактлар бўйича командани бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	' f ' регистрни ўқиш	Бажариш	f регистрига ёзиш '

Мисол1:

BSF FLAG_REG,7

Команда бажарилгунча
FLAG_REG = 0x0A (0000 1010)
Команда бажарилгандан сўнг
FLAG_REG = 0x8A (1000 1010)

Мисол2:

BSF INDF, 3

Команда бажарилгунча
W = 0x17
FSR = 0xC2
Адреси FSR да кўрсатилган
регистр қиймати = 0x20 (0010 0000)
Команда бажарилгандан сўнг
W = 0x17
FSR = 0xC2
Адреси FSR да кўрсатилган регистр
қиймати = 0x28 (0010 1000)

BTFSC **f регистрдаги b битни текшириш,агар 0 бўлса ўтказиб юбориш**

Синтаксис: *[label]* **BTFSC** f,b

Операндлар: 0<f < 127
0 < b < 7

Операция: Агар (f<b>) = 0 бўлса ўтказиб юбориш

Флаг ўзгаришлари: Йўқ

Код:

01	01bb	bfff	ffff

Таъриф: f регистрида b бит = 1 бўлса, кейинги команда бажарилади
f регистрида b бит = 0 бўлса, кейинги команда бажарилмайди ва команда иккита циклда бажарилиб, иккинчи циклда NOP бажарилади

Сўзлар: 1

Цикллар: 1(2)

Тактлар бўйича команда бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	' f ' регистрни ўқиш	Бажариш	Операция йўқ

Агар бажарилмаса (2-цикл)

Q1	Q2	Q3	Q4
Операция йўқ	Операция йўқ	Операция йўқ	Операция йўқ

Мисол1:

```
HERE    BTFSC   FLAG, 4
FALSE   GOTO    PROCESS_CODE
TRUE    •
```

Хол 1 Команда бажарилгунча
PC = адрес HERE
FLAG = xxx0 xxxx
Команда бажарилгандан сўнг
FLAG<4> = 0, бўлгани учун
PC = адрес TRUE

Хол 2 Команда бажарилгунча
PC = адрес HERE
FLAG = xxx1 xxxx
Команда бажарилгандан сўнг
FLAG<4> = 1, бўлгани учун
PC = адрес FALSE

BTFSS f регистрдаги b битни текшириш, агар 1 бўлса ўтказиб юбориш

Синтаксис: *[label]* **BTFSS** f,b

Операндлар: 0<f < 127
0 < b < 7

Операция: Агар (f<b>) = 1 бўлса ўтказиб юбориш

Флаг ўзгаришлари: Йўқ

Код:

01	01bb	bfff	ffff

Таъриф: f регистридаги b бит = 0 бўлса, кейинги команда бажарилади
f регистридаги b бит = 1 бўлса, кейинги команда бажарилмайди ва команда иккита циклда бажарилиб, иккинчи циклда NOP бажарилади

Сўзлар: 1

Цикллар: 1(2)

Тактлар бўйича команда бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	' f ' регистрни ўқиш	Бажариш	Операция йўқ

Агар бажарилмаса (2-цикл)

Q1	Q2	Q3	Q4
Операция йўқ	Операция йўқ	Операция йўқ	Операция йўқ

Мисол1:

```
HERE    BTFSS    FLAG,4
FALSE   GOTO     PROCESS_CODE
TRUE    •
```

Хол 1

Команда бажарилгунча
PC = адрес HERE
FLAG = xxx0 xxxx
Команда бажарилгандан сўнг
FLAG<4> = 0, бўлгани учун
PC = адрес FALSE

Хол 2

Команда бажарилгунча
PC = адрес HERE
FLAG = xxx1 xxxx
Команда бажарилгандан сўнг
FLAG<4>= 1, бўлгани учун
PC = адрес TRUE

CALL Подпрограммани чақириш

Синтаксис: *[label}* CALL k
Операндлар: 0 < k < 2047
Операция: (PC) + 1 -» TOS,
k->PC<10:0>,
(PCLATH<4:3>) -> PC<12:11 >
Флаг ўзгаришлари: Йўқ

Код:

10	0kkk	kkkk	Kkkk

Таъриф: **Подпрограммани чақириш**. Кейинги инструкция адреси (PC+1) стек бошига ёзилади. Ўнбир бит адрес команда кодидан команда кўрсатгичи PC га юкланади PC<10:0>. Команда кўрсатгичи PC нинг 11 ва 12 разрядлари PC<12:11> га PCLATH регистридан 2 бит юкланади. CALL командаси иккита циклда бажарилади.

Сўзлар:
Цикллар: 1
Тактлар бўйича 2
командининг
бажарилиши

1-цикл

Q1	Q2	Q3	Q4
Командани декодлаш	‘к’константа ни ўқиш	Бажариш	Операция йўқ

2-цикл

Q1	Q2	Q3	Q4
Операция йўқ	Операция йўқ	Операция йўқ	Операция йўқ

Мисол1:

```
HERE    CALL    THERE
Команда бажарилгунча
        PC = адрес HERE
Команда бажарилгандан сўнг
        PC  =  адрес THERE
        TOS = адрес HERE + 1
```

CLRF f ни тозалаш

Синтаксис: *[label]* CLRF f

Операндлар: 0 <f< 127

Операция: 00h → (f) 1 →Z

Флаг ўзгаришлари: Z

Код:	00	0001	lfff	ffff

Таъриф: 'f ' регистрни тозалаш (нол ёзиш) ва Z флагни шрнатиш

Сўзлар: 1

Цикллар: 1

Командаларни тактлар бўйича бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	Операция йўқ	Бажариш	'f ' регистрга ёзиш

Мисол1:

```
CLRF    FLAG_REG
Команда бажарилгунча
        FLAG_REG = 0x5A Команда
бажарилгандан сўнг
        FLAG_REG = 0x00
        Z=1
```

Мисол2:

```
CLRF    INDF
Команда бажарилгунча
        FSR = 0xC2
        Адреси FSR да кўрсатилган
        регистр қиймати = 0xAA
Команда бажарилгандан сўнг
        FSR = 0xC2
        Адреси FSR да кўрсатилган
        регистр қиймати = 0x00
        Z=1
```

CLRW W ни тозалаш

Синтаксис: *[label]* CLRW

Операндлар: Йўқ

Операция: 00h → (W) 1 → Z

Флаг ўзгаришлари: Z

Код:

00	0001	0xxx	xxxx

Таъриф: W регистрни тозалаш W ва Z флагни ўрнатиш

Сўзлар: 1

Цикллар: 1

Тактлар бўйича командани бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	Операция йўқ	Бажариш	W регистрига ёзиш

Мисол1: CLRW

Команда бажарилгунча

W = 0x5A

Команда бажарилгандан сўнг

W = 0x00

Z = 1

CLRWDT WDT ни тозалаш

Синтаксис: *[label]* CLRWDT

Операндлар: Йўқ

Операция: 00h → WDT,
00h → предделитель WDT,
1 →-TO
1 →-PD

Флаг ўзгаришлари: -TO, -PD

Код:

00	0000	0110	0100

Таъриф: CLRWDT инструкцияси WDT ни ва предделителни ноллайди (агар у WDT га уланган бўлса) STATUS регистрида –TO ва –PD битларига 1 ёзилади.

Сўзлар: 1

Цикллар: 1

Тактлар бўйича командани бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	Операция йўқ	Бажариш	WDT га 0 ёзиш

Мисол1:

```
CLRWDT
Команда бажарилгунча
        Счетчик WDT = ?
        Делитель WDT = 1:128
Команда бажарилгандан сўнг
        Счетчик WDT = 0
        Счетчик делителя WDT = 0
        -TO=1
        -PD= 1
        Делитель WDT = 1:128
```

Изох: CLRWDT командаси WDT нинг бўлиш коэффициентига таъсир этмайди.

COMF f ни инвертирлаш

Синтаксис: *[label]* COMF f,d

Операндлар: 0 <f < 127
d € [0,1]

Операция: (-f) → (dest)

Флаг ўзгаришлари: Z

Код:

00	1101	dfff	ffff

Таъриф: ' f ' регистрдаги барча битларни инвертирлаш.
Агар d-0 бўлса, результат W регистрига ёзилади.
Агар d-1 бўлса, результат ' f ' регистрига ёзилади.

Сўзлар: 1

Цикллар: 1

Тактлар бўйича командалар бажарилиши:

Q1	Q2	Q3	Q4
Командани декодлаш	' f ' регистрни ўқиш	Бажариш	Натижани ёзиш

Мисол1:
```
COMF    REG1, 0
Команда бажарилгунча
        REG1 =0x13
Команда бажарилгандан сўнг
        REG1 =0x13
        W = 0xEC
```

Мисол2:
```
COMF    INDF, 1
Команда бажарилгунча
        FSR = 0xC2
        Адреси FSR да кўрсатилган
        регистр қиймати = 0xAA
Команда бажарилгандан сўнг
        FSR = 0xC2
        Адреси FSR да кўрсатилган
        регистр қиймати = 0x55
```

Мисол3:
```
COMF    R E G 1,1
Команда бажарилгунча
        REG1 = 0xFF
Команда бажарилгандан сўнг
        REG1 = 0x00
        Z=1
```

DECF f даги коддан 1 ни айириш (декремент f)

Синтаксис: *[label]* DECF f,d

Операндлар: 0 < f < 127
d € [0,1]

Операция: (f) - 1 → (dest)

Флаг ўзгаришлари: Z

Код:

00	00H	dfff	ffff

Таъриф: ' f ' регистрдаги кодни декрементлаш.
Агар d=0 бўлса, результат W регистрга ёзилади.
Агар d=1 бўлса, результат f регистрга ёзилади.

Сўзлар: 1

Цикллар: 1

Тактлар бўйича командани бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	f регистрни ўқиш	Бажариш	Натижани ёзиш

Мисол1: **DECF C N T,1**
Команда бажарилгунча
CNT = 0x01
Z = 0
Команда бажарилгандан сўнг
CNT = 0x00
Z=1

Мисол2: **DECF INDF,1**
Команда бажарилгунча
FSR = 0xC2
Адреси FSR да кўрсатилган
регистр қиймати = 0x01
Z = 0
Команда бажарилгандан сўнг
FSR = 0xC2
Адреси FSR да кўрсатилган
регистр қиймати = 0x00
Z=1

Мисол3: **DECF CNT ,0**
Команда бажарилгунча
CNT = 0x10
W = x
Z = 0
Команда бажарилгандан сўнг
CNT = 0x10
W = 0x0F
Z = 0

DECFSZ f дан 1 айириш, агар 0 бўлса кейинги команда бажарилмайди

Синтаксис: *[label]* DECFSZ f,d

Операндлар: 0 < f < 127
d € [0,1]

Операция: (f) - 1 → (dest) 0 бўлса кейинги команда бажарилмасин

Флаг ўзгаришлари: Йўқ

Код:

00	1011	dfff	ffff

' f ' регистрдаги кодни декрементлаш.

Таъриф: Агар d=0 бўлса, результат W регистрга ёзилади.
Агар d=1 бўлса, результат f регистрга ёзилади.
Натижа 0 бўлмаса кейинги команда бажарилади
Натижа 0 бўлса кейинги команда бажарилмайди
Команда икки циклда бажарилади. Иккинчи циклда NOP

Сўзлар: 1

Цикллар: 1(2)

Тактлар бўйича командани бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	f регистрни ўқиш	Бажариш	Натижани ёзиш

Кейинги команда бажарилмаса (2 цикда)

Операция йўқ	Операция йўқ	Операция йўқ	Операция йўқ

Мисол 1

```
HERE    DECFSZ  CNT.1
        GOTO    LOOP
CONTINUE
```

Хол 1 Команда бажарилгунча
PC = адрес HERE
CNT = 0x01
Команда бажарилгандан сўнг
CNT = 0x00
PC = адрес CONTINUE

Хол 2 Команда бажарилгунча
PC = адрес HERE
CNT = 0x02
Команда бажарилгандан сўнг
CNT = 0x01
PC = адрес HERE + 1

GOTO Шартсиз ўтиш

Синтаксис: *[label]* GOTO k

Операндлар: 0 < k < 2047

Операция: k→PC<10:0>,
(PCLATH<4:3>) → PC<12:11>

Флаг ўзгаришлари: Йўқ

Код:

10	1kkk	kkkk	kkkk

Таъриф: Шартсиз ўтиш бажарилади.
Ўн бир битли адрес команда кодидан команда кўрсаткичи PC га юкланади: PC<10:0>.
Кўрсатгичнинг икки катта бити PC<12:11> PCLATH регистридан юкланади.
GOTO командаси иккита циклда бажарилади.

Сўзлар: **1**

Цикллар: **2**

Тактлар бўйича командалар бажарилиши

1- цикл

Q1	Q2	Q3	Q4
Командани декодлаш	‘к’константан и ўқиш	Бажариш	Операция йўқ

2- цикл

Q1	Q2	Q3	Q4
Операция йўқ	Операция йўқ	Операция йўқ	Операция йўқ

Мисол 1: GOTO THERE

Команда бажарилгандан сўнг
PC = адрес THERE

INCF f га 1 қўшиш (инкремент)

Синтаксис: *[label]* INCF f,d

Операндлар: 0 <f< 127
d € [0,1]

Операция: (f) + 1 → (dest)

Флаг ўзгаришлари: Z

Код:

00	1010	dfff	ffff

Таъриф: ' f ' регистрдаги кодга 1 қўшиш.(инкрементлаш)
Агар d=0 бўлса, результат W регистрга ёзилади.
Агар d=1 бўлса, результат f регистрга ёзилади.

Сўзлар: 1

Цикллар: 1

Тактлар бўйича командани бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	f регистрни ўқиш	Бажариш	Натижани ёзиш

Мисол 1: **INCF CNT , 1**

Команда бажарилгунча
CNT = 0xFF
Z = 0
Команда бажарилгандан сўнг
CNT = 0x00
Z=1

Мисол 2: **INCF INDF, 1**

Команда бажарилгунча
FSR = 0xC2
Адреси FSR да кўрсатилган регистр қиймати = 0xFF
Z = 0
Команда бажарилгандан сўнг
FSR = 0xC2
Адреси FSR да кўрсатилган регистр қиймати = 0x00
Z=1

Мисол 3: **INCF CNT, 0**

Команда бажарилгунча
CNT = 0x10
W = x
Z = 0
Команда бажарилгандан сўнг
CNT = 0x10
W = 0x11
Z = 0

INCFSZ f га 1ни қўшиш, агар 0 бўлса кейиги команда бажарилмасин

Синтаксис: *[label]* INCFSZ f,d

Операндлар: 0 < f < 127
d € [0,1]

Операция: (f) - 1 → (dest) 0 бўлса кейинги команда бажарилмасин

Флаг ўзгаришлари: Йўқ

Код:

00	1111	dfff	ffff

' f ' регистрдаги кодни декрементлаш.

Таъриф: Агар d=0 бўлса, результат W регистрга ёзилади.
Агар d=1 бўлса, результат f регистрга ёзилади.
Натижа 0 бўлмаса кейинги команда бажарилади
Натижа 0 бўлса кейинги команда бажарилмайди
Команда икки циклда бажарилади. Иккинчи циклда NOP

Сўзлар: 1

Цикллар: 1(2)

Тактлар бўйича командани бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	f регистрни ўқиш	Бажариш	Натижани ёзиш

Кейинги команда бажарилмаса (2 цикда)

Операция йўқ	Операция йўқ	Операция йўқ	Операция йўқ

Мисол 1

```
HERE    INCFSZ  CNT, 1
        GOTO    LOOP
CONTINUE
```

Хол 1 Команда бажарилгунча
PC = адрес HERE
CNT = 0xFF
Команда бажарилгандан сўнг
CNT = 0x00
PC = адрес CONTINUE

Хол 2 Команда бажарилгунча
PC = адрес HERE
CNT = 0x00
Команда бажарилгандан сўнг
CNT = 0x01
PC = адрес HERE + 1

IORLW константа ва W устида битлар бўйича “ЁКИ” амали

Синтаксис: *[label]* IORLW k
Операндлар: 0 < k < 255
Операция: (W) .OR. k -> (W)
Флаг ўзгаришлари: Z

Код:

11	1000	kkkk	kkkk

Таъриф: 8-разрядли 'k' константа ва W регистридаги код устида битлар бўйича “ЁКИ” амали бажарилади
Натижа W регистрга ёзилади.

Сўзлар: 1
Цикллар: 1

Тактлар бўйича командалар бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	‘к’ константани ўқиш	Бажариш	W регистрига ёзиш

Мисол 1:

IORLW 0x35
Команда бажарилгунча
W = 0x9A
Команда бажарилгандан сўнг
W = 0xBF
Z = 0

Мисол 2:

IORLW MYREG
Команда бажарилгунча
W = 0x9A
MYREG = 0x37 (регистр адреси)
Команда бажарилгандан сўнг
W = 0x9F
Z = 0

Мисол 3:

IORLW HIGH (LU_TABLE)
Команда бажарилгунча
W = 0x9A
LU_TABLE = 0x9375
(программалар хотирасидаги адрес)
Команда бажарилгандан сўнг
W = 0x9B

Мисол 4:

IORLW 0x00
Команда бажарилгунча
W = 0x00
Команда бажарилгандан сўнг
W = 0x00
Z=1

IORWF W ва f регистрларидаги кодлар устида битлар бўйича “ЁКИ” амали

Синтаксис: *[label]* IORWF f,d

Операндлар: 0 <f< 127
d € [0,1]

Операция: (W) .OR. (f) → (dest)

Флаг ўзгаришлари: Z

Код:

00	0100	dfff	f f f f

Таъриф: W ва f даги кодлар устида битлар бўйича “ЁКИ” амали бажарилади
Агар d=0 бўлса, натижа W регистрида сақланади.
Агар d=1 бўлса, натижа ' f ' регистрида сақланади.

Сўзлар: 1

Цикллар: 1

Тактлар бўйича командалар бажарилиши:

Q1	Q2	Q3	Q4
Командани декодлаш	f регистрни ўқиш	Бажариш	Натижани ёзиш

Мисол 1: **IORWF RESULT, 0**
Команда бажарилгунча
RESULT = 0x13 W = 0x91
Команда бажарилгандан сўнг
RESULT = 0x13 W = 0x93 Z = 0

Мисол 2: **IORWF INDF, 1**
Команда бажарилгунча
W = 0x17
FSR = 0xC2
Адреси FSR да кўрсатилган регистр қиймати = 0x30
Команда бажарилгандан сўнг
W = 0x17 FSR = 0xC2
Адреси FSR да кўрсатилган регистр қиймати = 0x37
Z = 0

Мисол 3: **IORWF RESULT, 1**

Хол 1 Команда бажарилгунча
RESULT = 0x13 W = 0x91
Команда бажарилгандан сўнг
RESULT = 0x93 W = 0x91 Z = 0

Хол 2 Команда бажарилгунча
RESULT = 0x00 W = 0x00
Команда бажарилгандан сўнг
RESULT = 0x00 W = 0x00 Z=1

MOVF f га жўнатиш

Синтаксис: *[label]* MOVF f,d

Операндлар: 0<f < 127
d € [0,1]

Операция: (f) → (dest)

Флаг ўзгаришлари: Z

Код:

00	1000	dfff	ffff

Таъриф: ' f ' регистдаги код қабул қилувчи регистрга жўнатилади
Агар d=0 бўлса, W регистрига жўнатилади.
Агар d=1 бўлса, ' f ' регистрига жўнатилади. d=1 ' f ' даги кодни нолга тенглигини текширишда ишлатилади

Сўзлар: 1

Цикллар: 1

Тактлар бўйича команда бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	' f ' регистрни ўқиш	Бажариш	Натижани ёзиш

Мисол 1: **MOVF FSR, 0**
Команда бажарилгунча
W = 0x00 FSR = 0xC2
Команда бажарилгандан сўнг
W = 0xC2 FSR = 0xC2 Z = 0

Мисол 2: **MOVF INDF,1**
Команда бажарилгунча
W = 0x17 FSR = 0xC2
Адреси FSR да кўрсатилган регистр қиймати = 0x00
Команда бажарилгандан сўнг
W = 0x17 FSR = 0xC2
Адреси FSR да кўрсатилган регистр қиймати = 0x00
Z=1

Мисол 3: **MOVF FSR, 1**

Хол 1 Команда бажарилгунча
FSR = 0x43
Команда бажарилгандан сўнг
FSR = 0x43 Z = 0

Хол 2 Команда бажарилгунча
FSR = 0x00
Команда бажарилгандан сўнг
FSR = 0x00 Z=1

MOVLW константани W га жўнатиш

Синтаксис: *[label]* MOVLW k
Операндлар: 0 < k < 255
Операция: k→(W)
Флаг ўзгаришлари: Йўқ
Код:

11	00xx	Kkkk	kkkk

Таъриф: Константани W регистрга жўнатиш.
Ишлатилмайдиган битларга ассемблер '0' ёзиб қўяди.
Сўзлар: 1
Цикллар: 1
Тактлар бўйича команда бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	‘k’ константани ўқиш	Бажариш	W регистрига ёзиш

Мисол 1: MOVLW 0x5 A
Команда бажарилгандан сўнг
W = 0x5A

Мисол 2: MOVLW MYREG
Команда бажарилгунча
W = 0x10
MYREG = 0x37 (регистр адреси)
Команда бажарилгандан сўнг
W = 0x37
Z = 0

Мисол 3: MOVLW HIGH (LU_TABLE)
Команда бажарилгунча
W = 0x10
LU_TABLE = 0x9375
(программалар хотирасидаги адрес)
Команда бажарилгандан сўнг
W = 0x93

MOVWF **W ни f га жўнатиш**

Синтаксис: [label] MOVWF f
Операндлар: 0<f< 127
Операция: (W) → (f)
Флаг ўзгаришлари: Йўқ

Код:	00	0000	1fff	ffff

Таъриф: W регистрдаги кодни ' f ' регистрига жўнатилади.
Сўзлар: 1
Цикллар: 1

Тактлар бўйича командани бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	Регистрни ўқиш ' f '	Бажариш	F регистрига ёзиш '

Мисол1: **MOVWF OPTION_REG**
Команда бажарилгунча
OPTION = 0xFF
W = 0x4F
Команда бажарилгандан сўнг
OPTION = 0x4F
W = 0x4F

Мисол2: **MOVWF INDF**
Команда бажарилгунча
W = 0x17
FSR = 0xC2
Адреси FSR да кўрсатилган
регистр қиймати = 0x00
Команда бажарилгандан сўнг
W = 0x17
FSR = 0xC2
Адреси FSR да кўрсатилган
регистр қиймати = 0x17

NOP Операция йўқ

Синтаксис: *[labe!* NOP

Операндлар: Йўқ

Операция: Операция йўқ

Флаг ўзгаришлари: Йўқ

Код:

00	0000	0xx0	0000

Таъриф: Операция йўқ

Сўзлар: 1

Цикллар: 1

Тактлар бўйича командани бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	Операция йўқ	Операция йўқ	Операция йўқ

Мисол 1:

```
HERE    NOP
```

Команда бажарилгунча
PC = адрес HERE
Команда бажарилгандан сўнг
PC = адрес HERE + 1

OPTION OPTION регистрини юклаш

Синтаксис *[label]* OPTION

Операндлар: Йўқ

Операция (W)→ OPTION

Флаг ўзгаришлари: Йўқ

Код:

00	0000	0110	0010

Таъриф: W регистрдаги кодни OPTION регистрига жўнатиш. Инструкция P1C16C5X. микроконтроллер командалари билан мосликни таъминлаш учун ишлатилади OPTION регистрига Ўқиш/Ёзиш командаларини бевосита ёки билвосита усулда бажариш мумкин

Сўзлар: 1

Циклар: 1

Янги ишлаб чиқарилаётган PIC 16CXX микроконтроллерлари билан программа таъминоти мослигини таъминлаш учун бу командадан фойдаланманг.

RETFIE — Подпрограммадан қайтиш (узилишга рухсат билан)

Синтаксис: *[label]* **RETFIE**

Операндлар: Йўқ

Операция: TOS → PC
1 →GIE

Флаг ўзгаришлари: Йўқ

Код:

00	0000	0000	1001

Таъриф: Узилишларни қайта ишловчи подпрограммадан қайтиш. Стека боши TOS команда счетчиги PC га юкланади. Узилишларга глобаль рухсат флаги GIE(INTCON<7>)га '1' ёзилади.
Инструкция 2 циклда бажарилади

Сўзлар: 1

Цикллар: 2

Тактлар бўйича командани бажарилиши

	Q1	Q2	Q3	Q4
1-й цикл	Командани декодлаш	Операция йўқ	Бажариш	Операция йўқ
	Q1	Q2	Q3	Q4
2-й цикл	Операция йўқ	Операция йўқ	Операция йўқ	Операция йўқ

Мисол1: **RETFIE**
Команда бажарилгандан сўнг
PC = TOS
GIE = 1

RETLW — Подпрограммадан константани W регистрига юклаб, қайтиш

Синтаксис: *[label]* RETLW k

Операндлар: 0 < k < 255

Операция: k→ (W)
TOS → PC

Флаг ўзгаришлари: Йўқ

Код:

11	01xx	кккк	кккк

Таъриф: В регистр W загружается 8-разрядная константа. Вершина стека TOS загружается в счетчик команд PC. Инструкция выполняется за 2 цикла. 1

Сўзлар:

Цикллар:

Бажарилиши тактлар бўйича командалар 1-й цикл

Q1	Q2	Q3	Q4
Командан	‘к’	Бажарилиши	W

Q1	Q2	Q3
Q4		

2-й цикл

Операция	Йўқ	Операция	Йўқ

Мисол1:

```
        CALL    TABLE ; W – TABLE таблица
                        ;элементининг номерига тенг
        * W –  энди TABLE таблица элементига тенг
        *
TABLE   ADDWF PC ;      W=элемент номери
        RETLW  k1 ; TABLE таблицасининг боши
        RETLW  k2
        RETLW  k3
        RETLW  k4
        RETLW  k5
        RETLW  k6
        RETLW  k7
        RETLW  k8
        *
        *
        *
        RETLW  kn ; TABLE таблицасининг охири
```

Команда бажарилгунча
W = 0x07 ;таблица элементи номерига тенг бўлса,
Команда бажарилгандан сўнг
W = к8; таблица элементига тенг бўлади.
PC = TOS = адрес HERE +1

RETURN Подпрограммадан қайтиш

Синтаксис: *[label]* RETURN
Операндлар: Йўқ
Операция: TOS -> PC
Флаг ўзгаришлари: Йўқ
Код: 00 0000 0000 1000
Таъриф: Подпрограммадан қайтиш. Стек боши TOS команда кўрсатгичи PC га юкланади.
Инструкция 2 циклда бажарилади.
Сўзлар: 1
Цикллар: 2
Тактлар бўйича командани бажарилиши

1-й цикл

Q1	Q2	Q3	Q4
Командани декодлаш	Операция йўқ	Бажариш	Операция йўқ

2-й цикл

Q1	Q2	Q3	Q4
Операция йўқ	Операция йўқ	Операция йўқ	Операция йўқ

Мисол1: **RETURN**
Команда бажарилгандан сўнг
PC = TOS

f регистрдаги кодни чапга

RLF **Carri орқали циклик силжитиш**

Синтаксис: *[label]* RLF f,d

Операндлар: 0 <f< 127
d € [0,1]

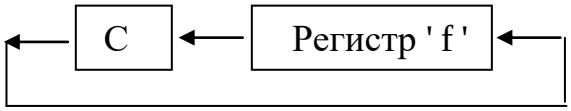
Операция: Таърифни қаранг

Флаг ўзгаришлари: C

Код:

00	1101	dfff	ffff

Таъриф: ' f ' регистрдаги кодни STATUS регистрининг C бити орқали чапга циклик силжитиш бажарилади. Агар d=0 бўлса, натижа W регистрида сақланади. Агар d=1 бўлса, натижа ' f ' регистрида сақланади.



Сўзлар: 1

Цикллар: 1

Тактлар бўйича командани бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	' f ' регистрни ўқиш	Бажариш	Натижани ёзиш

Мисол1: **RLF REG1, 0**

Команда бажарилгунча
REG1 =11100110
C = 0

Команда бажарилгандан сўнг
REG1 =11100110
W = 1100 1100
C = 1

Мисол2: **RLF INDF,1**

Хол 1
Команда бажарилгунча
W = xxxx xxxx
FSR = 0xC2
Адреси FSR да кўрсатилган регистр қиймати = 0x3A (0011 1010)
C = 1

Команда бажарилгандан сўнг
W = 0x17
FSR = 0xC2
Адреси FSR да кўрсатилган регистр қиймати = 0x75 (0111 0101)
C = 0

Хол 2
Команда бажарилгунча
W = xxxx xxxx
FSR = 0xC2
Адреси FSR да кўрсатилган регистр қиймати = 0xB9 (1011 1001)
C = 0

Команда бажарилгандан сўнг
W = 0x17
FSR = 0xC2
Адреси FSR да кўрсатилган регистр қиймати = 0x72 (0111 0010)
C = 1

f регистрдаги кодни ўнгга

RRF **Carri орқали циклик силжитиш**

Синтаксис: *[label]* RRF f,d

Операндлар: 0 <f< 127
d € [0,1]

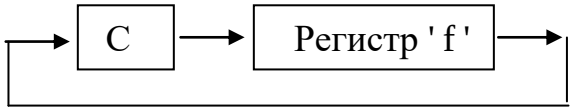
Операция: Таърифни қаранг

Флаг ўзгаришлари: C

Код:

00	1100	dfff	ffff

Таъриф: ' f ' регистрдаги кодни STATUS регистрининг C бити орқали ўнгга циклик силжитиш бажарилади. Агар d=0 бўлса, натижа W регистрида сақланади. Агар d=1 бўлса, натижа ' f ' регистрида сақланади.



Сўзлар: 1

Цикллар: 1

Тактлар бўйича командани бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	' f ' регистрни ўқиш	Бажариш	Натижани ёзиш

Мисол1: **RRF REG1, 0**

Команда бажарилгунча
REG1 =11100110
W = xxxx xxxx
C = 0

Команда бажарилгандан сўнг
REG1 =11100110
W = 0111 0011
C = 0

Мисол2: **RRF INDF,1**

Хол 1 Команда бажарилгунча
W = xxxx xxxx
FSR = 0xC2
Адреси FSR да кўрсатилган регистр қиймати = 0x3A (0011 1010)
C = 1

Команда бажарилгандан сўнг
W = 0x17
FSR = 0xC2
Адреси FSR да кўрсатилган регистр қиймати = 0x9D (1001 1101)
C = 0

Хол 2 Команда бажарилгунча
W = xxxx xxxx
FSR = 0xC2
Адреси FSR да кўрсатилган регистр қиймати = 0x39 (0011 1001)
C = 0

Команда бажарилгандан сўнг
W = 0x17
FSR = 0xC2
Адреси FSR да кўрсатилган регистр қиймати = 0x72 (0001 1100)
C = 1

SLEEP — SLEEP (кутиш) режимига ўтиш

Синтаксис: *[label]* SLEEP

Операндлар: Йўқ

Операция: 00h → W D T
00h → предделитель W D T

Флаг ўзгаришлари: 1 →-TO
0→-PD
-TO, -PD

Код:

00	0000	0110	0011

Таъриф: Манба уланганлиги белгиси -PD га '0' ёзиш.
WDT тўлганлиги белгиси -TO га '1' ёзиш.
WDT таймер ва унинг предделителини тозалаш (дастабки холатга қайтариш)
Микроконтроллерни SLEEP режимига ўтказиш ва такт генераторини тўхтатиб қўйиш (ўчириш).

Сўзлар: **1**

Цикллар: **1**

Тактлар бўйича командани бажарилиши

Командани декодлаш	Операция йўқ	Операция йўқ	SLEEP режимига ўтиш

Мисол1: SLEEP

Изох. SLEEP командаси WDT нинг бўлувчи коэффициентига таъсир кўрсатмайди.

SUBLW Константадан W даги кодни айириш

Синтаксис: *[labe]* SUBLW k

Операндлар: 0 < k < 255

Операция: k - (W) → (W)

Флаг ўзгаришлари: C, DC, Z

Код:

11	110x	Kkkk	kkkk

Таъриф: 8-разрядли 'k' константадан W регистридаги кодни айириш
Натижа W регистрида сақланади.

Сўзлар: 1

Цикллар: 1

Тактлар бўйича командани бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	‘k’ константани ўқиш	Бажарилиши	W регистрига ёзиш

Мисол1: **SUBLW 0x02**

Хол 1
Команда бажарилгунча
W = 0x01
C = ?
Z = ?
Команда бажарилгандан сўнг
W = 0x01
C = 1;
натижа мусбат
Z = 0

Хол 2
Команда бажарилгунча
W = 0x02
C = ?
Z = ?
Команда бажарилгандан сўнг
W = 0x00
C = 1 ;
натижа нулга тенг
Z = 1

Хол 3
Команда бажарилгунча
W = 0x03
C = ?
Z = ?
Команда бажарилгандан сўнг
W = OxFF
C = 0 ;
натижа манфий
Z = 0

Мисол 2: **SUBLW MYREG**

Команда бажарилгунча
W = 0x10
MYREG = 0x37 (регистр адреси)

Команда бажарилгандан сўнг
W = 0x27
C = 1

SUBWF f дан W ни айириш

Синтаксис: [label] SUBWF f,d

0<f< 127

Операндлар: d € [0,1]

Операция: (f) - (W) → (dest)

Флаг ўзгаришлари: C, DC, Z

Код:

00	0010	dfff	ffff

Таъриф: " f " регистрдаги коддан. W регистрдаги кодни айириш
Агар d=0 бўлса, натижа W регистрга ёзилади.
Агар d=1 бўлса, натижа ' f ' регистрга ёзилади.

Сўзлар: 1

Цикллар: 1

Тактлар бўйича командалар бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	" f " регистрни ўқиш	Бажариш	Натижани ёзиш

Мисол1: **SUBWF REG1 , 1**

Хол 1

Команда бажарилгунча
REG1 = 0x03
W = 0x02
C = x
Z = x

Команда бажарилгандан сўнг
REG1 = 0x01
W = 0x02
C =1 ;
натижа мусбат
Z = 0

Хол 2

Команда бажарилгунча
REG1 = 0x02
W = 0x02
C = x
Z = x

Команда бажарилгандан сўнг
REG1 = 0x00
W = 0x02
C =1;
натижа нул
Z=1

Хол 3

Команда бажарилгунча
REG1 = 0x01
W = 0x02
C = x
Z = x

Команда бажарилгандан сўнг
REG1 = 0xFF
W = 0x02
C =0
натижа манфий
Z=0

SWAPF f регистридаги ярим байтлар ўрнини алмаштириш

Синтаксис: [label] SWAPF f,d

Операндлар: 0 < f <127

Операция: d € [0,1]
(f<3:0>) → (dest<7:4>)
(f<7:4>) → (dest<3:0>)

Флаг ўзгаришлари: Йўқ

Код:

00	1110	dfff	ffff

Таъриф: f регистридаги ярим байтлар ўрнини алмаштирилади.
Агар d=0 бўлса, натижа W регистрида сақланади.
Агар d=1 бўлса, натижа ' f ' регистрида сақланади.

Сўзлар: 1

Цикллар: 1

Тактлар бўйича командани бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	' f ' регистрни ўқиш	Бажариш	Натижани ёзиш

Мисол1: **SWAPF R EG, 0**

Команда бажарилгунча
REG = 0xA5

Команда бажарилгандан сўнг
REG = 0xA5
W = 0x5A

Мисол2: **SWAPF INDF, 1**

Команда бажарилгунча
W = 0x17
FSR = 0xC2
Адреси FSR да кўрсатилган регистр қиймати = 0x20

Команда бажарилгандан сўнг
W = 0x17
FSR = 0xC2
Адреси FSR да кўрсатилган регистр қиймати = 0x02

Мисол3: **SWAPF REG, 1**

Команда бажарилгунча
REG = 0xA5

Команда бажарилгандан сўнг
REG = 0x5A

TRIS TRIS регистрига юклаш

Синтаксис: *[label]* TRIS f

Операндлар**:**

Операции: (W) → TRIS регистр f

Флаг ўзгариши: Йўқ

Код:	00	0000	0110	0fff

Таъриф: W даги кодни регистр TRIS ига кўчириб ёзиш. Инструкция PIC16C5X микроконтроллерлар оиласида программалар мослигини таминлаш учун сақлаб қолинган . OPTION регистрига Ёзиш/Ўқиш командасини бевосита ёки билвосита адресация орқали бажариш мумкин

Сўзлар:

Цикллар

PIC16CXX микроконтроллерлар оиласида программалар мослигини таминлаш учун бу командадан фойдаланмаслик тавсия этилади

XORLW Константа ва W даги код устида бит бўйича **'Тенгсизлик'**

Синтаксис:	*[label]* XORLW k			
Операндлар:	0 < k < 255			
Операция:	(W) .XOR. k → (W)			
Флаг ўзгаришлари:	Z			
Код:	11	1010	Kkkk	kkkk
Таъриф:	8-разрядли 'k' константа ва W даги код устида бит бўйича **'Тенгсизлик'** амали бажарилади. Натижа W регистрига ёзилади			
Сўзлар:	1			
Цикллар	1			

Тактлар бўйича командани бажарилиши	Q1	Q2	Q3	Q4
	Командани декодлаш	‘k’ константани ўқиш	Бажариш	W регистрига ёзиш

Мисол1: **XORLW** **0xAF** (1010 1111)

Команда бажарилгунча
W = 0xB5(1011 0101)

Команда бажарилгандан сўнг
W = 0x1A(0001 1010)
Z=0

Мисол 2: XORLW MYREG

Команда бажарилгунча
W = 0xAF
MYREG = 0x37
(адрес регистра)

Команда бажарилгандан сўнг
W = 0x18
Z = 0

Мисол 3: XORLW HIGH (LU_TABLE)

Команда бажарилгунча
W = 0xAF
LU_TABLE = 0x9375
(программа хотираси адреси)

Команда бажарилгандан сўнг
W = 0xЗС
Z = 0

XORWF W ва f устида 'Тенгсизлик' амали

Синтаксис: *[label]* XORWF f,d
Операндлар: O <f< 127
Операция: d € [0,1]
(W) .XOR. (f) → (dest)
Флаг ўзгаришлари: Z
Код:

00	0100	dfff	ffff

Таъриф: W ва f регистрлардаги код устида бит бўйича **'Тенгсизлик'** амали бажарилади.
Агар d=0 бўлса, натижа W регистрида сақланади.
Агар d=1 бўлса, натижа f регистрида сақланади.
Сўзлар: 1
Цикллар: 1

Тактлар бўйича команда бажарилиши

Q1	Q2	Q3	Q4
Командани декодлаш	f регистрни ўқиш	Бажариш	Натижани ёзиш

Мисол1: **XORWF REG, 1**

Команда бажарилгунча
REG = 0xAF
W = 0xB5

Команда бажарилгандан сўнг
REG = 0x1A
W = 0xB5

Мисол2: **XORWF REG, 0**

Команда бажарилгунча
REG = 0xAF
W = 0xB5

Команда бажарилгандан сўнг
REG = 0xAF
W = 0x1A

Мисол3: **XORWF INDF, 1**

Команда бажарилгунча
W = 0xB5
FSR = 0xC2
Адреси FSR да кўрсатилган регистр қиймати = 0xAF

Команда бажарилгандан сўнг
W = 0xB5
FSR = 0xC2
Адреси FSR да кўрсатилган регистр қиймати = 0x1A

7. Тез-тез бериладиган саволларга жавоблар

Агар ўз саволингизга бу бобдан жавоб топа олмасангиз, бу саволни қуйидаги электрон адрес: support@rnicrochip.ru га ёзинг.

Савол 1: Қандай қилиб бевосита W регистрининг қийматини ўзгартириш мумкин? W регистрининг қийматини декрементлаш талаб этилади

Жавоб 1:

Бир неча усуллар мавжуд:

1. PICmicro микроконтроллерларида W регистри билан ишловчи командалар мавжуд. Масалан, агар W регистридаги кодни декрементлаш керак бўлса, буни ADDLW 0xFF командаси билан амалга оширса бўлади (бу ерда 0x - MPASM ассемблер транслятори учун ўн олтилик сон белгиси).
2. Маълумки, кўпгина командалар ахборот хотирасининг регистри қийматини ўзгартириши мумкин, бунда натижани W регистрида сақлаш мумкин. Бу W регистрга юклаш пайтида декрементлаш мумкинлигини англатади. Командада натижа регистри сифатида W кўрсатилиши керак: (DECF регистр адреси, 1).

Савол 2: TRIS командасини ишлатганда PIC16CXXX микроконтроллерлари учун бирор бир хавф мавжудми? Микроконтроллер хужжатларида бу командани ишлатиш тавсия этилмаган.

Жавоб 2:

Микроконтроллерларнинг янги авлоди билан дастурий бирликни таъминлаш учун TRIS командасини ишлатмаслик керак. Яна шуни ҳисобга олиш керакки, TRIS командаси билан фақат A, B ва C портларни созлаш мумкин. Янги микроконтроллерларда бу команда бўлмаслиги мумкин.

Савол 3: TRIS командасини ишлатганда ахборот хотираси банкини (банк 1ни) танлаш керакми ? PORTA. нинг киритиш/чиқариш каналлари йўналишини созлаш талаб этилади.

Жавоб 3:

TRIS командасини ишлатганда ахборот хотираси банкини улаш талаб этилмайди. Микроконтроллерларнинг янги авлоди билан дастурий бирликни таъминлаш учун **TRIS** командасини ишлатмаслик керак

Савол 4: Хужжатларда "ўқиш – модификация – ёзиш" командасини ишлатганда эҳтиёт бўлиш лозимлиги кўрсатилган. Негалигини тушунтириб беринг?

Жавоб 4:

"Ўқиш – модификация – ёзиш" структурали командага – бит устида амал бажарувчи BCF командаси оддий мисол бўлади. Айтайлик, портга чиқаришни бошқарувчи битга нуль ёзиш бажарилмоқда. Бу холда амалда киритиш/чиқариш портининг тўлиқ холатини ифодаловчи байт ўқилади, сўнгра, керакли битга нуль ёзилади ва янги байт киритиш/чиқариш портига ёзилади (ёки регистрга). Регистрнинг жорий қийматига боғлиқ бўлган ҳар қандай команда, "ўқиш – модификация – ёзиш" структурали командадир. (ADDWF, SUBWF, BCF, BSF, INCF, XORWF ва б.). Регистрнинг жорий қийматига боғлиқ бўлмаган ҳар қандай команда, "ўқиш – модификация – ёзиш" структурали команда эмасдир (MOVWF, CLRF ва б.).

"Ўқиш – модификация – ёзиш" структурали командасини киритиш/чиқариш порти учун бажарилишини кўриб чиқайлик. Масалан, TRISB

регистрининг барча битларидаги “1” лар PORTB ни чиқишга йўналтиради ва PORTB нинг барча чиқишларида мантиқий юқори даража ҳосил қилади. Энди, сиз мантиқий қуйи даража мавжуд бўлган RB3 ни киришга йўналтирмоқчисиз. RB6 да мантиқий қуйи даража ўрнатиш учун BCF PORTB, 6 командасини бажарасиз. Агар сиз яна RB3 ни чиқишга йўналтирсангиз, аввал унга юқори мантиқий даража ўрнатган бўлсангиз ҳам энди унда мантиқий қуйи даража шаклланади. BCF командаси портнинг бошқа чиқишига (RB6) бажарилаётганда портнинг барча чиқишларидаги холати ўқилади (RB3 дан 0). 6 бит керакли қийматга ўзгартирилади, аммао, RB3 дан '0' ўқилганлиги учун у портнинг буферига ёзилиб қолади. Бу канал чиқишга йўналтирилганда, янги қиймат каналга берилади.

Савол 5: BCF командасини ишлатилганда портнинг бошқа каналлари қуйи мантиқий даражага ўтиб қолади. Нега?

Жавоб 5:

1. Қачон "Ўқиш – модификация – ёзиш" командаси портнинг бошқа каналлари холатини ўзгартиришини қуйидаги тарзда кўрсатиб бериш мумкин. Айтайлик, PORTC нинг барча каналлари чиқишга йўналтирилган ва барчасида қуйи мантиқий даража мавжуд. Хар бир каналга светодиод уланган у шу каналда юқори мантиқий даража пайдо бўлганда ёнади. Хар бир светодиодга 10мкФ сиғимли конденсатор параллел уланган. Микроконтроллер программаси 20МГц такт частота билан бажарилади. Энди светодиодларни кетма кет ёқувчи қуйидаги командаларни шакллантирайлик: BSF PORTC,0; BSF P0RTC,1; BSF P0RTC,2 ва х.к.. Бу командалар ишлагандан сўнг фақат охирги каналда юқори мантиқий даража бўлишини ва охирги светодиод ёнишини кўришингиз мумкин. Чунки, конденсаторлар зарядланиши учун маълум вақт талаб этилади ва кейинги каналга шу каналдаги конденсатор зарядланишини кутиб ўтирмасдан “1” ёзилди, вахоланки бу амални бажаришдан аввал миропроцессор шу портнинг барча каналлари холатини ўқигани туфайли олдинги каналда “0” холати ўқилди ва бу “0” каналнинг чиқиш буферига ёзилиб, шу каналда қуйи мантиқий даража хосил қилди. ("Ўқиш – модификация – ёзиш" командаси). Бу эффектни фақат микроконтроллернинг юқори такт частотларда ва киритиш/чиқариш портларидан операцияларни кетма-кет бажарганда ҳисобга олиш керак.
2. Бундай хол агар Сиз киритиш/чиқариш портлари каналларини ADC0N1 регистрида керакли тарзда кўрсатмаган бўлсангиз PIC16C7XX микроконтроллерларида юз бериши мумкин. Агар канал аналог сигналли чиқиш деб кўрсатилган бўлса каналдаги кучланишдан қатий назар “Ўқиш” '0' натижа беради. Бу канал холати доимо ўқилади деган қоидага зиддир. Сиз TRISA регистрида аналог сигналли чиқишни рақамли чиқиш каби кўрсатишингиз мумкин ва чиқишдаги мантиқий даражани бошқаришингиз мумкин, аммо “Ўқиш” доимо '0' натижа беради. Шунинг учун, агар Сиз портга "Ўқиш – модификация – ёзиш" командаси билан мурожат қилсангиз барча аналог чиқишлар '0' ўқилади ва команда томонидан ўқилган қиймат ўзгартирилади ва чиқиш буферига '0' ёзилади. Аналог киришларда мантиқий бўлмаган кучланиш мавжуд бўлиши мумкин, шунинг учун

рақамли кириш буферлари узиб қўйилган.

Илова: 6 позицияли соат дастури

```
          LIST    p=16F84 ; PIC16F844 процессорга мўлжаллранган

          #include "P16F84.INC" ; Ушбу файл қўшиб
ишлатилади

              CBLOCK 0x10       ; Ўзгарувчилар
                 state     ;
                 secs      ; секундлар
                 mins      ; минутлар
                 hours     ; соатлар
                 ticks     ; 1 секунд(тиккиллаш)
                 idc
                 bcd
              ENDC

; Битлар тақсимоти учун константалар.
; Бу ерда BIT_x константалар актуал бит номерлари ва
;         MASK_x эса шу бит шаблонидир.
BIT_HSEL      EQU H'0000'
BIT_TSET      EQU H'0001'
BIT_HSET      EQU H'0002'
BIT_MSET      EQU H'0003'

BIT_H24       EQU H'0000'
BIT_PM        EQU H'0001'
BIT_SET       EQU H'0002'
BIT_HSB       EQU H'0003'

MASK_H24      EQU H'0001'
MASK_PM       EQU H'0002'
MASK_SET      EQU H'0004'
MASK_HSB      EQU H'0008'


; Ушбу Macro MOVLW инструкцияларни генерациялайди ва булар
ўз набатида эталон узилишларни хосил қилади:
break         MACRO arg
              DW    0x3100 | (arg & H'FF')
              ENDM

              ORG   0
entrypoint    goto  initialize  ; Дастурга кириш нуқтаси
                                ; (дастур боши)

              ORG   4
intvector     goto    clock     ; Узилиш вектори боши

initialise    ; Регистрларга бошланғич қиймат бериш:
              clrw                    ; W га нуль жўнатиш.
              movwf   PORTA           ; PORTA га нуль
жўнатиш.
```

```
                movwf   PORTB    ; PORTB га нуль жўнатиш.
                bsf     STATUS,RP0    ; Bank 1 ни танлаш
                movlw   H'1F'         ; PORTA учун
киритиш/чиқариш ниқоби (шаблони).
                movwf   TRISA       ; TRISA регистрни
ўрнатиш.
                movlw   H'01'       ; PORTB учун
киритиш/чиқариш ниқоби (шаблони).
                movwf   TRISB       ; TRISB регистрини
ўрнатиш.
                bcf     STATUS,RP0  ; Reselect Bank 0.

; Initialise clock: Соатни юргазиб юбориш:
                clrf    state
                bsf     state,BIT_HSB
                movlw   D'0'
                movwf   hours
                movlw   D'0'
                movwf   mins
                movlw   D'0'
                movwf   secs

; 50Hz узилишлар қийматини тозалаш: Clear 50Hz tick count:
                clrf    ticks

; Clear interrupt disable count (idc) semaphore:
; Узилишларни тақиқловчи индикаторни тозалаш:
                clrf    idc

; Initialise display:
; Дисплейни дастлабки холатга келтириш:
                call    wr_hours
                call    wr_mins
                call    wr_secs
                call    wr_state

; Finally initialise interrupts for clock on RB0/INT pin:
; RB0/INT даги соат узилишларини дастлабки холатга
келтириш:
                movlw   H'90'
                movwf   INTCON

start
;When not processing an interrupt we sit and check input
pins:
;Узилиш бўлмаганда киришларни назорат қиламиз
     call    chk_tset   ; Time set select active?
                        ; Вақтни киритиш кнопкаси босиқми?
     call    chk_hsel ; H12/H24 display format select
active?
; 12/24 соат формати кнопкаси босиқми?
                goto    start
```

```
;-------------------------------------------------------------
--; Interrupt handler.
; We come here for every tick of the time base.
; Узилишларни тахлил қилишни боши.
; Бу ерга хар бир ярим сония ўтгач келамиз
clock          ; Toggle half-second flag and set state
               ; Ярим секунд белгиси ва қайсидир кнопка
босилди
outputs:
               incf    ticks,F   ; Increment clock ticks.
                                 ; Тиккилашлар сонига 1
қўшамиз
               movf    ticks,W   ; Get ticks value.
                                 ; Тиккилашлар сонини W га
оламиз
               sublw   D'25'              ; У 25 ми? (W=25-
W)?
               btfss   STATUS,Z          ; Test zero flag.
                                         ; Натижа нулми?
               goto    endclock          ; Return. ХА

toggle_hs  ;Half second-toggle HS flag, write it and
return:
           ;Ярим секунд–икки холатли HS flag,ёзиш ва
қайтиш:

               clrf    ticks     ; Reset timebase,
                                 ; вақт асосини тозалаш
               movf    state,W   ; Get state.Холатни W га
олиш
               xorlw   MASK_HSB  ; Toggle half-second bit.
                                 ; икки холатли HS бит
               movwf   state     ; Save it back to регистр.
                                 ; Холатни регистрга сақлаш
               call    wr_state  ;Display it.Дисплейга
чиқариш
               movf    state,W          ; Get state.
               btfss   state,BIT_HSB    ; Is bit now clear?
                                        ; 3 - бит тозами?
               goto    endclock          ;Return.Йўқ,қайтиш.
inc_secs       ; Incrmement seconds... ХА, секунд қўшамиз
               movf    secs,W    ; Get it into W.
               sublw   D'60'     ; 60 га тенгми, (W=60-W)?
               incf    secs,F    ; Increment seconds count.
               btfsc   STATUS,Z  ; Test zero flag, skip
clear
; if no set. Z=0 бўлса кейиги команда бажарилмайди.
               goto    reset_secs       ; Clear seconds,
; increment minutes. Секунд = 0 ва минут + 1
               call    wr_secs  ; Write seconds it to
display.
```

```
; Секундларни дисплейга         ёздик
              goto    endclock ; Done.Секундларни санашда
; давом этамиз

reset_secs    clrf    secs       ; секунд=0
              call    wr_secs    ; Write seconds to
display.
                                 ; Секундни дисплейга ёзамиз
              incf    mins,F     ; Increment minute count.
                                 ; минут + 1
              movf    mins,W     ; Get it into W.
              sublw   D'60'      ; минут = 60? (W=60-W)?
              btfsc   STATUS,Z   ; Test zero flag, skip
clear
; if no set. Z=0 бўлса кейинги команда бажарилмайди
              goto    reset_mins      ; Clear minutes,
; increment hours. Минут=0 ва соат + 1 қилишга ўтамиз
              call    wr_mins   ;Write minutes it to
display.
                                ; Минутни дисплейга ёзамиз
              goto    endclock  ;Done.Секундни санашга
ўтамиз

reset_mins    clrf    mins       ; Reset minute count to
zero.
                                 ; Минут=0
              call    wr_mins    ; Write minutes to display.
                                 ; Минутни дисплейга ёзамиз
              call    inc_hours ; Increment hours,соат + 1
                                  ; display it with PM flag.
                                  ; дисплейга PM ёзувини
                             ёзамиз
endclock      movlw   H'90'
              movwf   INTCON     ; соатни тиккилашига рухсат
              retfie             ; Return
;-----------------------------------------------------------
--
; Subroutine. Check the state of the HSEL input and
; set h12/h24 format as required.
; Подпрогамма. HSEL холатини киритиш ва
; h12/h24 да сўралган форматни ўрнатиш

chk_hsel  btfsc   PORTA,BIT_HSEL  ; Test 12/24 select.
          goto    set_h12          ;H12 12 соатли форматда.
         ; Ўтказиб юборамиз H12 24 соатли форматда бўлмаса.
set_h24  btfsc   state,BIT_H24   ; Are we on 12 hour
format?
                                  ; 12 соатли форматдамизми?
         retlw   0                ; No, so no need to do
                                   ; anything...
                                   ; Йўқ, хеч нарса керак
                              эмас
```

```
         bcf     INTCON,GIE    ; Disable interrupts.
                                ; Узилишларга рухсат йўқ
         incf    idc,F          ; Increment count of
number
                                ; of disables.
         bsf     state,BIT_H24  ; Clear h12 flag.
                               ; h12 флагни тозалаш
         movf    hours,W        ; Get hours value.
         sublw   D'12'          ; Is it 12:xx? Соат 12?
         btfsc   STATUS,Z       ; Test zero flag.
         clrf    hours         ; Reset to zero.
                               ; Z=0 бўлса соат = 0
         movf    hours,W       ; Get hours value.W=соат
         btfsc   state,BIT_PM  ; Is the PM indicator set?
                               ; PM индикатори борми?
         addlw   D'12'         ; Add 12 to get 24 hour
value.
                              ;бор бўлса 12 қўшилади.
         movwf   hours   ; Save result (does nothing for
AM)
         bcf     state,BIT_PM ; Clear PM flag.
         call    wr_hours     ; Write hours. Соатни ёзиш
         call    wr_state     ; Write H12 state and PM
state.
         goto    chk_hsel_iec ; Done.

set_h12  btfss   state,BIT_H24  ; Are we on 12 hour
format?
                                ; Хозир 12 соатли форматми?
         retlw   0      ; Yes, so no need to do anything...
                  ; Ха, state да 0-разряд 0,қайтиш керак

         bcf     INTCON,GIE     ; Disable interrupts.
                               ; Узилишларни ўчириб қўйиш
         incf    idc,F ;Increment count of number of
disables
                      ; Ўчириб қўйишлар сонига + 1
         bcf     state,BIT_H24  ; Set h12 flag.

         movf    hours,W       ; Get hour value.
         sublw   D'11'         ; W=11-W.
; C is clear for a borrow (W>=12) бўлса C=0 аксинча C=1.
         btfss   STATUS,C       ; Test carry flag.
         goto    set_h12_pm     ; Set PM.
;соат 12дан кичик бўлса кейинги код бажарилади

set_h12_am    bcf     state,BIT_PM    ; Clear PM bit.
                           ; state нинг 1 битига 0 ёзилади
              movf    hours,W       ; Get hours.
              btfsc   STATUS,Z      ; Is it zero? соат
нулми?
              addlw   D'12'         ; Yes, add 12 to get
```

```
; 00:xx to 12:xx. Ха,  00:xx   ни 12:xx га айлантирамиз.
             movwf   hours      ; Save any result.
                                ;Натижани сақлаб қўямиз
            call    wr_hours   ; Display hours. Соатни
ёзамиз
            call    wr_state   ; Display H12 and PM states.
            goto    chk_hsel_iec    ; Done.

set_h12_pm    bsf     state,BIT_PM    ; Set PM bit.
              movlw   D'12'           ; Constant. 12->W
              subwf   hours,F         ; hours=hours-12
; соатни (23..12 -> 11..0) кўринишга ўтказдик.
              btfsc   STATUS,Z        ; Zero set?
         movwf   hours           ; Yes, so reset to '12'.
                                 ; Ха, 12 ни 00 га келтирдик
         call    wr_hours        ; No, so leave hours alone
; and display it.
; Йўқ, соатни ўзгартирмаймиз ва шундайлигича дисплейга
ёзамиз
         call    wr_state        ; Display H12 and PM
states.
         goto    chk_hsel_iec    ; Done.

chk_hsel_iec  decfsz  idc,F           ; Decrement idc.
; If zero we can reenable interrupts.
; Агар 0 бўлса кейиги команда ишламайди ва узилишларга
рухсат
; тикланади
              retlw   1               ; Return without
; enabling interrupts. Узилишларга рухсатсиз қайтиш

chk_hsel_done movlw   H'90'      ; Constant for GIE and
T0IE.
             movwf   INTCON      ; Set interrupt регистр.
; Узилишларга рухсат берилди
             retlw   1           ; Return.

;-------------------------------------------------------
--
chk_tset      btfsc   PORTA,BIT_TSET  ; Set mode? Режим?
              retlw   0

              bcf     INTCON,GIE      ; Disable interrupts.
; Узилишларни тақиқлаб қўйилди
              incf    idc,F           ; Increment count
;of number of disables. Тақиқлар сони + 1
              bsf     state,BIT_SET   ; Set the 'set mode'
; bit.ига 1 ёзилди.
              clrf    secs            ; Setting time resets
; seconds count. Секундлар сони = 0
              call    wr_secs         ; Display it. ёзилди
```

```
              bcf             state,BIT_HSB   ; Clear
seconds toggle.
; Секундлар триггерини нуллаш
              call    wr_state        ; Update the state
;output latch.Холат регистри қийматини чиқиш зашелкасига
ёзиш

set_loop      call    chk_hsel        ; Check for H12/H24
; display change. H12/H24 ни танлашни текшириш
              btfss   PORTA,BIT_MSET  ; If pin is high then
;switch is open.PORTA нинг BIT_MSET оёғи=1 бўлса калит очиқ
              goto    set_mins        ; Bit not set, switch
;closed,set minutes.Бит=0,калит ёпиқ,минутларни киритиш
керак
              btfss   PORTA,BIT_HSET  ; If pin is high then
;switch is open.Соатни киритиш калити очиқ
              goto    set_hours       ; Bit not set, switch
;closed, set hours. Бит=0,калит ёпиқ,соатларни киритиш
керак
              btfss   PORTA,BIT_TSET  ; If pin is high then
; switch is open. Калит очиқ, кейиги команда бажарилмайди
              goto    set_loop        ; Bit not set, switch
;closed, loop. Калит ёпиқ, давом этамиз

              bcf     state,BIT_SET   ; Clear the 'set
mode'
;bit.Ўзгартириш режими белгисини тозалаш(ўчириш/нуллаш)
              call    wr_state        ; Update the state
; output latch. Холатни чиқиш зашёлкасига(буферига) ёзиш

chk_tset_idc  decfsz  idc,F           ; Decrement idc. If
;zero we can reenable interrupts.Узилишларни тўхтатиш
сонини ; 1 га камайтирамиз. Натижа 0 бўлса узилишга
рухсатни тиклаш ; мумкин
              retlw   1               ; Return without
;enabling interrupts. Узилишга рухсатсиз қайтиш
chk_tset_done movlw   H'90'      ; Constant for GIE and
T0IE.
;Глобал ва таймердан узилишларга рухсат беришга константа
              movwf   INTCON          ; Set interrupt
;Узилишлар регистрига «рухсат» константаси ёзилди.
              retlw   1               ; Return.

set_mins      incf    mins,F    ; Increment minute=minute+1
              movf    mins,W    ; Get it into W.
              sublw   D'60'     ; Is it 60 (W=60-W)?
              btfsc   STATUS,Z  ; Test zero flag, skip clear
if no set.
              clrf    mins      ; Clear minutes.
              call    wr_mins   ; Write minutes.
```

```
debounce_mset btfss            PORTA,BIT_MSET  ; Wait for
MSET button
; to be released. MSET кнопкасини бўшатилишини кутамиз

              goto    debounce_mset   ; Loop.
              goto    set_loop        ; Released so recheck
;buttons. Кнопкалар бўшатилишини қайта текширамиз

set_hours     call    inc_hours       ; Increment hours and
; display.Соат+1 ва уни дисплейга ёзиш
debounce_hset btfss   PORTA,BIT_HSET  ; Wait for HSET
button ;to be released. HSET кнопкасини бўшатилишини
кутамиз
              goto    debounce_hset   ; Loop.
              goto    set_loop        ; Released so recheck
;buttons. Кнопкалар бўшатилишини қайта текширамиз
;-----------------------------------------------------
--; Increment hours, set PM indicator bit as necessary.
; Соат + 1, PM indicatorни ёқиш (ўрнатиш)

inc_hours     btfsc   state,BIT_H24   ; 12h display? 12
;соатли дисплеймми?
              goto    reset_on24      ; No,  so we reset
when
;we get to 24.Йўқ, 24 соатли формат сўралган
reset_on12    movf    hours,W         ; Get hours
              sublw   D'12'           ; Is it 12 (W=12-W)?
              btfsc   STATUS,Z        ; Test zero flag,
skip
;clear if no set. Соат 12 бўлса кейинги команда ишламайди.
              clrf    hours           ; Clear hours.соат=0
              incf    hours,F         ; Increment
hours.соат+1
              call    wr_hours        ; Write it. Ёзиш
              movf    hours,W         ; Get hours
              sublw   D'12'           ; Is it 12 (W=12-W)?
              btfss   STATUS,Z        ; Test zero flag.
              return                  ; Zero not set, so not
;12, so return. Соат 12 бўлмаган бўлса қайтиш.
              movlw   MASK_PM         ; Get PM state bit
mask.
              xorwf   state,F  ;Toggle PM state bit in
state.
                              ;Холат регистрида PM триггерига
1
              call    wr_state        ; Update state
outputs.
    ;Янгиланган холатни портга чиқариш
              return                  ; Return.

reset_on24    incf    hours,F     ; Incrment hours. Соат+1
              movf    hours,W     ; Get it in W.
```

```
              sublw   D'24'    ; Is it 24 (W=24-W)?
              btfsc   STATUS,Z    ; Test zero flag, skip
; clear if no set.
; Соат 24 дан кичик бўлса кейинги команда бажарилмайди
              clrf    hours       ; Clear hours 24->0.
              call    wr_hours    ; Write hours to display.
              return

;-----------------------------------------------------------
--; Get seconds value, split to get BCD pair and write to
port.
; Секуднинг иккилик коддини иккилк-ўнлик жуфтликка ўтказиш
; (конвертациялаш)ва портга чиқариш(ёзиш)
wr_secs       movf    secs,W          ; Get seconds.
              call    bin2bcd         ; Convert to BCD.
              movwf   bcd             ; Save result.
              andlw   0xF0            ; Mask BCD pair to
;leave upper digit in W upper nibble.
;Жуфтликнинг катта тўртлигини W га ажратиб олдик
              iorlw   0x0A  ; Select strobe.
;строб сигналини қўшдик
              movwf   PORTB ; Write to PORTB. Ёздик
              andlw   0xF0  ; Clear strobe (by selecting
Q0).
; Стробдан тозаладик.
              movwf   PORTB ; Write to PORTB.Ёздик
              swapf   bcd,W
; Get lower BCD digit in to W upper nibble.
; Қуйи рақамни W регистрининг юқори тўртлигига олдик
              andlw   0xF0   ; Mask of strobe selection
bits.
                             ; Юқори тўртликни ажратиб олдик
              iorlw   0x0C   ; Set WR strobe. Строб қўшдик
              movwf   PORTB  ; Write to PORTB. Ёздик
              andlw   0xF0   ; Clear strobe.Стробни
тозаладик
              movwf   PORTB  ; Write to PORTB. Ёздик
              return         ; Return.

;-----------------------------------------------------------
--; Get minutes value, split to get BCD pair and write to
port.
; минутлар иккилик коддини иккилик-ўнлик жуфтликка ўтказиш
; (конвертациялаш)ва портга чиқариш(ёзиш)(Секундга ўхшаш)

wr_mins       movf    mins,W          ; Get minutes value.
              call    bin2bcd         ; Convert to BCD.
              movwf   bcd             ; Save result.
              andlw   0xF0
; Mask BCD pair to leave upper digit in W upper nibble.
              iorlw   0x06  ; Select strobe.
              movwf   PORTB ; Write to PORTB.
```

```
              andlw   0xF0      ; Clear strobe (by selecting
Q0).
              movwf   PORTB ; Write to PORTB.
              swapf   bcd,W
; Get lower BCD digit in to W upper nibble.
              andlw   0xF0
; Mask of strobe selection bits.
              iorlw   0x08              ; Set WR strobe.
              movwf   PORTB             ; Write to PORTB.
              andlw   0xF0              ; Clear strobe.
              movwf   PORTB             ; Write to PORTB.
              return                    ; Return.

;-----------------------------------------------------
; Get hours value, split to get BCD pair and write to port.
; соатлар иккилик коддини иккилик-ўнлик жуфтликка ўтказиш
; (конвертациялаш)ва портга чиқариш(ёзиш)(Секундга ўхшаш)

wr_hours      movf    hours,W           ; Get hours value/
              call    bin2bcd           ; Convert to BCD.
              movwf   bcd               ; Save result.
              andlw   0xF0              ; Mask BCD pair to
leave upper digit in W upper nibble.
              iorlw   0x02              ; Select strobe.
              movwf   PORTB             ; Write to PORTB.
              andlw   0xF0              ; Clear strobe (by
selecting Q0).
              movwf   PORTB             ; Write to PORTB.
              swapf   bcd,W             ; Get lower BCD digit
in to W upper nibble.
              andlw   0xF0              ; Mask of strobe
selection bits.
              iorlw   0x04              ; Set WR strobe.
              movwf   PORTB             ; Write to PORTB.
              andlw   0xF0              ; Clear strobe.
              movwf   PORTB             ; Write to PORTB.
              return                    ; Return.
;-----------------------------------------------------
--
; Write state bits to state latch.
; Холат битларини портга чиқариш (ёзиш)
wr_state  swapf   state,W       ; Get state bits in to
W<4:1>.
                                ; Холат битларини W ёзиш
          iorlw   0x0E          ; Set WR strobe.
                                ; WR strobe ни қўшиш
          movwf   PORTB         ; Write to PORTB.га ёзиш
          andlw   0xF0          ;Clear strobe by selecting
Q0.
                                ; strobe ни тозалаш
          movwf   PORTB         ; Write to PORTB.га ёзиш
          return
```

```
;--------------------------   ----------------------------
-----
; Routine to convert a binary value  (0..63) in to a BCD
pair
; The result is returned in W. The lookup is done as a
; 'calculated goto' by modifying PCL
; (the lower eight bits of the program counter).
; PCL is only eight bits wide with the top five bits coming
; from the PCLATH регистр. We use an ORG
; statement to ensure that addition of the offset (0..63)
to
; the table address doesn't overflow the PCL регистр
; (any overflow would be lost and would result in a jump to
; some other part of the code).
;-----------------------------------------------------------
--
; Бу подпрограмма катталиги (0..63)гача бўлган иккилик
; коддаги сонларни ўнлик-иккилик жуфтлигига ўтказади.
; Натижа W регистрига ёзилади. Керакли сон кодини қидириш
; командаси PCL ни ўзгартириш билан “хисобланадиган goto”
; каби шакллантирилади (команда кўрсатгичининг кичик 8 та
; разрядида).
; PCL PCLATH регистрининг 5та юқори разряди ва шу 8 та
; разрядли коддан ташкил топади.
;
        ORG     0x0100

bin2bcd clrf    PCLATH   ; Clear PCLATH.
        bsf     PCLATH,0 ; Set bit zero so that goto yields
                         ; an address 0x01xx.
        andlw   H'3F'    ; Ensure we limit lookup.
        addwf   PCL,F    ; Add offset in W to PCL to
     calc.goto

              retlw   0x00
              retlw   0x01
              retlw   0x02
              retlw   0x03
              retlw   0x04
              retlw   0x05
              retlw   0x06
              retlw   0x07
              retlw   0x08
              retlw   0x09
              retlw   0x10
              retlw   0x11
              retlw   0x12
             retlw   0x13
             retlw   0x14
             retlw   0x15
             retlw   0x16
             retlw   0x17
             retlw   0x18
```

```
    retlw   0x19
    retlw   0x20
    retlw   0x21
    retlw   0x22
    retlw   0x23
    retlw   0x24
    retlw   0x25
    retlw   0x26
    retlw   0x27
    retlw   0x28
    retlw   0x29
    retlw   0x30
    retlw   0x31
    retlw   0x32
    retlw   0x33
    retlw   0x34
    retlw   0x35
    retlw   0x36
    retlw   0x37
    retlw   0x38
    retlw   0x39
    retlw   0x40
    retlw   0x41
    retlw   0x42
    retlw   0x43
    retlw   0x44
    retlw   0x45
    retlw   0x46
    retlw   0x47
    retlw   0x48
    retlw   0x49
    retlw   0x50
    retlw   0x51
    retlw   0x52
    retlw   0x53
    retlw   0x54
    retlw   0x55
    retlw   0x56
    retlw   0x57
    retlw   0x58
    retlw   0x59
    retlw   0x60
    retlw   0x61
    retlw   0x62
    retlw   0x63
END
```