1, Milyen a nyitott kollektoros kimenet?
Működés: tranzisztor nyitva, akkor folyik áram, ha zárva, akkor nem. Lényeg: TTL bemenetről vezérelhetünk nem TTL jellegű kimeneteket. Megj.: nyitott kollektoros kimenetű eszköz önmagában nem működik, táp felé ellenállást kell rá kötni.
Ha nyitott kollektoros kapuk kimeneteit összekötjük, és a közös vezetéket a tápfeszültséghez csatlakoztatjuk, akkor ez a közös vezeték csak akkor lesz H szintű, ha valamennyi kimenet H szinten van.
2, Hogyan készíthetünk stabilizált tápfeszültség forrást?
Cél: 5V-os feszültség előállítása. 1. lehetőség: 4 db 1,5V-os elem 1 tokban, így 6V-t kapunk, amit még elviselnek az alkatrészek. 2. lehetőség: stabilizált tápegység
3, Ismertesse az analóg komparátor működését!
2 analóg jelet (feszültséget) hasonlít össze, a kimenete digitális. Az analóg feszültségről eldönti, hogy egy adott feszültségnél kisebb, vagy nagyobb.
A kimenet és a bemenet összehasonlítása: az analóg bemeneti jelet megfelelő bináris számokká kell alakítani. (MSB=legnagyobb, LSB=legkisebb helyiértékű bit; kvantálás: egy analóg mennyiség értékváltozási tartományának felosztása meghatározott számú egységekre; Digitális jel: $D_1*2^{-1}+D_2*d^{-2}+\ldots+D_n*2^{-n}$, annál nagyobb a felbontóképesség minél több bit áll rendelkezésre a szám ábrázolásához).
Mintavételezés: a bemenet ált. nem egyenfesz. A bemeneti jel gyorsan/lassan változhat. Meghatározott időközönként a jelből mintát kell venni. Az analóg jel mintavételezési gyorsaságának legalább 2szer akkorának kell lennie, mint a jel legnagyobb frekvenciája (1) pillanatérték átalakító=gyors, 2) átlagérték átalakító=lassú átalakításokhoz)
4, Az 555-ös időzítő áramkör felépítése és alkalmazásai.
Cél: meghatározott ideig működtessen egy eszközt.
Lenyomjuk a kapcsolót => zárt kapcsoló : S=1, R=0 => Q=1, Q=0 => tranzisztoron keresztül nem folyik áram, kondenzátor töltődik R-en keresztül; nyitott kapcsoló : S=0, R=1 => Q=0, Q=1 => tranzisztoron keresztül folyik áram, kondenzátor kiürül
5, Ellenállás létra és R-2R felépítése és használata D/A konverterekben.
6, Műveleti erősítők alkalmazási lehetőségei TTL környezetben.
Bemenet lehet analóg feszültség; változások jó méréséhez erősíteni kell: digitalizálni egy mikrofon jelét. 1) kettő ellenállás: bemenetre sorosan, másik párhuzamosan => jó digitalizálhatóság-> egész intervallumot figyeljük.
Kondicionálni kell a jelet, R1-től függ a bemeneti ellenállás; digitalizáláshoz az egész tartományt nézni kell, dinamika tartomány beállítható.
Analóg komparátorként is alkalmazható.
7, Magyarázza meg, mik a multiplexerek, demultiplexerek, dekódolók
Dekódolók: beírás alacsony feszültségnél a memóriába; címdekódoló: olyan többkimenetű áramkör, amelynek kimenetén logikai 1 jelenik meg.
Multiplexer: MUX független bemeneti jeleket időben egymás után a kimenetre juttatja, kombinációs logikai áramkörökkel, nem mechanikus kapcsolókkal valósítják meg.
Demultiplexer: MUX fordítottja – bemeneten megjelenő adatokat utasítással az egyik meghatározott kimenetre kapcsolja (adatelosztó)
8, Hogyan működik a digitális komparátor?
8bites komparátor: kimenet 0, ha A=C, különben 1. Használat: címbuszon lévő címet össze kell hasonlítani az adott periféria címével. Egyenlőség vizsgálata: 2 db 8bites bináris szám => ad egy alacson szintű jelet; lehet A>C-re is ad jelet -> invertálás A<C => címbusz eldönti, hogy melyik perifériának szól.
XOR kapukkal megoldani annyit, ahány bites a címbusz, kimeneteket VAGY kapukkal összerakni ENABLEből 8 kimenet + még egy 0 bnemenet -> 9 bemenetű VAGY kapu összeintegrálva; kimenet akkor alacsony, ha A=C.
9, Hogyan működik a teljes összeadó?
ALU alapvető építőeleme. Figyeli az átvitelt; 2 félösszeadóból és egy VAGY kapuból áll.
10, Milyen feladatokra használhatók a tároló regiszterek és osztók?
11,Hogyan lehet nyomógombos kapcsolót TTL bemenethez illeszteni?
Első eset: kapcsoló nyitva = 0, kapcsoló zárva = 1:
Második eset: fordítva
12, Hogyan lehet láthatóvá tenni egy kimenet logikai állapotát?
TTL-nél a kimenet egy logikai állapota lehet alacson (0V), vagy magas (5V); egy LED-et kapcsolunk hozzá (olyan dióda, ami akkor világít, ha áram folyik rajta).
Első eset: logikai 1 kimenetnél világítson a LED:
Második eset: logikai 0 kimenetnél világítson a LED -> diódát meg kell fordítani.
13,Ismertesse az I/O ciklus idődiagramját!
I/O port: azoka regiszterek, melyek segítségével a periféria és a processzor kommunikálnak.
A címbusz jelölhet memóriát is. Az adatbuszt minden eszköz használja, meg kell oldani, hogy ne beszéljenek egyszerre, ehhez minden az órajel ciklushoz igazodik.
AEN: cím engedélyezési tartomány = mikor jelenik meg az az adat a buszon, amire az adott perifériának szüksége van; aktív, ha alacsony _ -> érvényes cím van ekkor; adat: lehet alacsony/magas, érvényes, amit le kell venni a buszról.
IOWrite: alacsony értéknél írhat, I/O periféria az adatbuszra beolvas; írás: adott pillanatban; beolvas: felfelé nyíl esetén.
IOR: beolvas I/O egységről. Áramkör: meghatározott I/O címről olvas/ír. AEN vonalat elő kell állítani.
14,Az output port felépítése.
Comparator: összehasonlítja a kapott címet az output címével, ki: 0, ha megegyezik, különben 1. egyik bemenet I/O címre kötve, másik bemenet: címbuszra kötve.
IOW: 0, ha írhat, 1, ha nem írhat.
CLK: jelváltásra tárol (lehet fel, vagy lefutó)
DATA LATCH: flip-flopok segítségével átírja az adatot, ha kell.
Azt a regisztersort kell működtetni, ahova a kiírás történik.
Ma: a bemenetet rácsatlakoztatják az adatbuszra -> beolvas; a bemenet minden órajelnél minden tárba egyszerre olvas be, kimeneti oldal a legközelebbi beolvasásig nem változik.
15,Az input port felépítése.
Ha a megfelelő inputról akarunk beírni és be is írhatunk (IOR engedélyezi), akkor az adat a TRI STATE BUFFER segítségével beíródik. TSB: 3 kimenetes; csak akkor kapcsol rá, ha nem aktív az állapot; 3. állapot=semleges; ha nem megfelelő időben ad jelet a gép elfagy. A buszról történő beolvasást a processzor végzi, a vonal állapota lesz az állapot (0/1).
16,A PC printer portjának felépítése.
17,Ismertesse az ALU felépítését és működését!
Mikroprocesszor része, alapegysége: MUX (műveletek kiválogatása, megfelelő utasításkódok bitjei); 2 db 8bites regiszter tartalma között végzünk műveleteket. Utasítás regiszter részei: utasításkód(3 bit -> 8 féle utasítást lehet megvalósítani) és cím. Az utasításkód a MUX címvezetékére kerül. MUX bemenetei: a lehetséges műveleti eredmények. MUX kimenete: A_i-be megy => műveletek eredménye A reg-be képződik.
ALU: kis számú aritmetikai és logikai művelet végrehajtását teszi lehetővé (nagy teljesítményű proc -> több művelet végrehajtása); aritmetikai műv: +,-; logikai műveletek: és, vagy, kizáró vagy, komplementálás.
Műveletek 1 vagy 2 operandus között jönnek létre. 1 operandus az akkumulátorban van. A műv eredménye is az akkumulátorban lesz. (Legalább 1 akkumulátora van minden mikroproc-nak). Az operandus eljut az ALU-hoz egy átmeneti tároló regiszteren át. A bemenetet és a kimenetet választják el egymástól.
Feltétel/státuszregiszter: állapota az éppen végrehajtott művelet er-től függ. Mikroproc összehasonlítja a státuszreg bitjét logikai változó 0-val vagy 1-el. Az összehasonlítás eredményének fv-ében alakul a program további menete. Bitjei: Carrey – átvitel, Overflow – túlcsordulási bit, előjelbit (1=-, 0=+), zéró – 1, ha az akkumulátor tartalma 0-vá válik, paritásbit – akkumulátorban lévő 1-esek számát jelzi (ha ps ->1)
18,Ismertesse a CLEAR/SET utasítás működését.
4 bemenetű multiplexer; az utasításnak dekódolásra alkalmasnak kell lennie;
10 bit – 2 utasítás (clear (0)/ set(1)) közül melyikről van szó a 4. bit mondja meg.
Utasítás formája: 0 1 0 0/1-(S/C) b b b f f f f f f; 1-3 bit: csak akkor legyen engedélyezett ez az áramkör, ha a bejövő utasítás 3 bitje ez; 4. bit: clear/sec utasítást választjuk; 4-7 bit hányadik bitre végezzük a műveletet; maradék: megadjuk a fileregiszter címét, amiben módosítani szeretnénk egy bitet.
19,Milyen alapvető digitális áramkörökből épül fel a számítógép?
Bináris adatok tárolása flip-flop-okból kialakított tárolóregiszterekkel; logikai kapuk; számlálók; dekóderek; MUX; digitális komparátor.
20,Magyarázza el, mi a Harvard-felépítés és hasonlítsa össze a Neumann-felépítésű számítógéppel
21,A blokkvázlat alapján mutassa be a PIC mikrovezérlők működését!
A program memóriát a PC címezi meg; a PC által megcímzett memóriahelyről kiolvassuk az aktuális utasítás szót az utregbe, onnan ez a tartalom az utdekódolóba kerül, ott megtörténik az utasítás dekódolása, majd a dekódolás alapján a végrehajtása; ha az aktuális utasítás közvetlen ugrás, akkor a közvetlen címet (ahonnan folytatni kell a programot) az ugró utasítás végrehajtásakor beírjuk a PC-be, átírva a PC tartalmát. Aritmetikai-logikai utasításoknál az egyik operandus w/omega regiszterben van, a művelethez szükség van a másik operandusra, amelynek címét az utasítás operandus mezője tartalmaz. Az utasítás dekódolásakor az ált célú regiszter tömböt fogja megcímezni ez a mezőtartalom, onnan az adatbuszon keresztül a megcímzett regiszter tartalma az ALU-ba kerül. Megtörténik a művelet végrehajtása -> az eredmény vagy a w/omega regiszterben marad, vagy visszaíródik a megcímzett reg-be.
22, Csoportosítsa a PIC mikrovezérlők lábkivezetéseit!
Tápfeszültség; MCLR – tok belső regisztereit alaphelyzetbe hozó pont; oszcillátor kivezetés; reset-láb; port kivezetés (bitenként programozható, hogy melyik legyen input, ill. output); {a tok működtetéséhez 5 kivezetés szükséges: a tápfeszültség két pontja (2), a tok belső regisztereit alaphelyzetbe hozó MCLR pont (1), két oszcillátor-kivezetés szükséges (2)}
23, Milyen lépésekből áll egy utasítás végrehajtása, és mi a pipe-line?
Utvégrehajtás lépései: utelőkészítés, lehívás; utszámláló reg tartalmának növelése; műveleti kód értelmezése, dekódolás, operandus címének meghatározása; a művelethez szükséges adatok előkészítése; végrehajtás; az eredmény elhelyezése.
Pipe-line: míg az egyik utasítást végrehajtjuk, addig lehetséges a következő utasítás memóriából történő lehívása és dekódolása; ha az éppen végrehajtandó utasítás ugró utasítás, akkor nem jól működik (ilyenkor a következő utasítást az ugrás helyéről kell betölteni, a már lehívott utasítást pedig el kell dobni).
24, Csoportosítsa, és röviden mutassa be a PIC mikrovezérlők utasításkészletét!
Logikai (and, or, xor, nor, not,…): 8 bites adatok bitenkénti logikai kapcsolatát végzik (maszkolás), 2es komponenst tud képezni;
Aritmetikai + shift (ADD,SUB,DEC,INC,RL,RR): regisztertartalmak összeadását, kivonását, 1gyel való növelését/csökkentését, carryn keresztült történő jobbra és balra forgatást végeznek, feltételvizsgálathoz kötött értéknövelés/csökkentés;
Bit (adott bit törése, egybe állítása,…): minden regiszter bitcímezhető, egyetlen utasítással meg lehet változtatni az utasítás vhanyadik bitjének értékét, biteket külön lehet kapcsolni;
Adatmozgató (MOV):fileregisztereket és a w/omega regiszter közötti adatcserét tesznek lehetővé;
Programvezérlő (CALL, GOTO, RETURN, JMP,…);
Rendszervezérlő (SLEEP, CLRWDT): a processzort alacsony fogyasztású állapotba küldi a SLEEP, külső utasításra újraéled, a watchdogot törlő utasítás a CLRWDT, beállított értékről nő/csökken, mindig le kell nullázni.
25,Milyen feladatot lát el a veremtár?

Általában a főmemóriában helyezkedik el, adatokat csak a verem tetejére lehet tenni és onnan elővenni (FIFO). Többször ismétlődő programrészeket nem szoktuk minden előfordulási helyére írni, hanem egy közös, un szubrutinba foglaljuk. Ennek végrehajtásakor egy ugró utasítással elugrunk a szubrutin elejére, majd a végrehajtása után ismét vezérlésátadással térünk vissza a hívást követő helyen lévő utasításra, de ez csak úgy lehetséges, ha visszatérési címet valahogy eltároljuk, erre szolgál a verem. A CALL sub1 utasítás végrehajtása előtt a CALL utasítást követő utasítás memóriacímét a processzor a verembe helyezi, majd a programszámlálóba a sub1 címét tölti. A sub1 szubrutin végén lévő RETURN utasítás a veremből előveszi az oda elmentett visszatérési címet és a programszámlálóba visszatölti (vagyis folytatódhat a CALL utáni utasítás végrehajtása). Ez a verem technika akkor is jól működik, ha sub1 szubrutinban is van egy CALL sub2 utasítás, ilyenkor a verem tetejét jelző mutatót eggyel meg kell növelni, hogy a most elhelyezkedő visszatérési cím ne írja felül az előzőt. PIC-nél a veremmélység 8.

26, Két operandust igénylő utasítások esetén hol képződik az eredmény?

Az eredmény vagy a w/omega regiszterben marad, vagy visszaíródik a megcímzett regiszterbe.

27,Magyarázza el a státuszbitek jelentését!

C: átvitel bit (1, ha 8.bitre volt átvitel, 0, ha nem); DC: digit átvitel, 1, ha 4.bitre volt átvitel, 0, ha nem; Z: zéro bit 1, ha az aritmetikai/logikai utasías eredménye 0, 0 különben; PD: PowerDown bit 1 bekapcsoláskor és CLRWDT utasításnál, 0 SLEEP utasításnál – a proc tényleges állapotát mutatja meg; TO: TimeOut bit – csak olvasható – 1 bekapcsoláskor, CLRWDT vagy SLEEP után, 0, ha WDT túlcsordulás történt; RP1-PR0: regiszter bank választó bitek (00=Bank0, 01=Bank1, 10=Bank2, 11=Bank3), minden Bank mérete 128 bájt; IRP: regiszter bank választás indirekt címzésnél 0==Bank0,1, 1=Bank2,3.

28,Miért van szükség memórialapok kialakítására, és mi az a lapváltás?

A korlátozott uthosszból adódó korlátozott prmemória címzés memórialapok bevezetését igényli. Lap=olyan adatblokk, melynek mérete (512 byte és 8 Kbyte között) azonos és rögzített. Lapváltás: a lapok csak egyenlő méretű üres keretbe kerülhetnek, ha nincs üres hely, akkor vmilyen stratégia (véletlenszerű választás, legrégebben bent lévő lap cseréje, legrégebben nem használt lap cseréje, adott idő alatt nem használt lap cseréje, legkevésbé használt lap cseréje) alapján ki kell üríteni egyet. Mindig csak a szükséges lapok vannak a memóriába, aminek hátránya, hogy nem lehet az azonos jellegű/tartalmú adatterületeket együtt kezelni, összefogni, mint szegmensváltáskor.

29,Mik az RP0, RP1, IRP bitek

Státuszregiszter bitjei.

RP1-PR0: regiszter bank választó bitek (00=Bank0, 01=Bank1, 10=Bank2, 11=Bank3), minden Bank mérete 128 bájt; IRP: regiszter bank választás indirekt címzésnél 0==Bank0,1, 1=Bank2,3.

30,Mi a RESET feladata, és hogyan működik ez a PIC-eknél?

RESET folyamat: a tápfesz bekapcsoláskor a mikrovezértlő belső áramköreit, regisztereit a megfelelő működés miatt jól meghatározott alaphelyzetbe kell állítani. Ehhez a belső órajel stabil működése szükséges. Bekapcsoláskor elindul egy fgtlen belső RC oszcillátorról működő 10 bites számláló (PWRT), ennek szerepe az esetleges lassú tápfeszültség növekedésének kompenzálása. Amikor ez túlcsordul, elindul egy az oszcillátor esetleges lassú berezgése miatti hibás működés kivédését célzó, második késleltetést biztosító számláló (OST). Több esemény idézhet elő RESET-et: tápfesz bekapcsolása, csökkenése; MCLR alacsony szintre húzása magas szintről; WDT által normál működéskor; paritáshiba. Legtöbb esetben elegendő az MCLR pontnak a tápfeszre kötése, a belső késleltetések helyes RESET folyamatot eredményeznek.

31,Órajel generálása PIC-eknél.

A PIC kontrollerek áramköreinek működését ütemező órajel előállítására több lehetőség van. LP: alacsony frekvenciájú kristály; XT: kristály vagy kerámia rezonátor; HS: nagy frekvenciás kristály (érzékeny a zavarokra => le kell árnyékolni, hogy ne zavarja interfer.); RS: küldő ellenállás-kondenzátor oszcillátor áramkör CLKOUT kimenettel (a CLK OUT kimeneten megjelenik egy TTL szintű belső órajelből származtatott órajel, amely a tök áramköri környezetében felhasználható); belső RC oszcillátor használata => nem kell külső elemet használni. Valamilyen órajelre mindig szükség van, mert anélkül el sem indul. Az órajel előállítása lényegében rezgéskeltés, ezért belső analóg elemek használatét igényli, amelyek kapcsolása és értékeik függnek az alkalmazott rezgéskeltő típusától, ezért a tok programozásakor meg kell adni a használt rezgéskeltő típusát.

32,Hogyan valósítható meg indirekt címzés?

Lehetséges úgy műveleteket végezni, hogy nem az utban lévő cím határozza meg az operandust, hanem a 4es címen lévő FSR tartalmát tekintjük címnek, és az ily módon megcímzett regiszter tartalmával végezzük a műveletet. Technikai megoldás: ha az utban szereplő operandus címe nulla, akkor az FSR-ben lévő tartalmat tekintjük címnek.

Az utban azt a tárolóhelyet látjuk, ahonnan az operandus címe kiolvasható. 1) vmely memória tárolóhely felhasználása, memória-rekesz; 2)proc vmely regiszterének segítségével. Operandus címét tartalmazó címek pointerek. Vektortábla: indirekt címek csoportos tárolására alkalmas; gyakran használt rutinok kezdőcímeinek elhelyezésére.

33,Milyen célt szolgál a Watch Dog Timer?

A kontrollereknél használt watchdog lényegében egy saját szabadonfutó RC oszcllátor órajellel léptet 8 bites számláló, amelyhez egy szintén 8 bites utóosztó kapcsolódik. A programban elhelyezett, periodikusan végrehajtott CLRWDT utasítással töröljük a számlálót és az utóosztót. Ha a program „elkószál” (vmilyen elektromos zaj miatt, rossz címre lépve hibásan működik), akkor a törlés elmaradása miatti túlcsordulás nullázza és újraindítja a kontrollert. SLEEP-ben is működik és a túlcsordulás felébreszti a kontrollert. WDT engedélyezése/tiltása az egyik programozható biztosítékkal lehetséges.

34,Mi a polling? Mi a megszakítás (interrupt-IT)?

Ha egy szgépes rendszerben vmilyen esemény létrejöttét kívánjuk érzékelni, ezt szokásos módon kétféleképpen tehetjük meg: 1)polling=programozott átvitel: a külső esemény létrejöttét egy bemeneti kapu bitváltozásának figyelésével érzékelhetjük. Alkalmazása lassítja a rendszer tényleges működési sebességét, hiszen a mikroproc. Idejének nagy részét azzal tölti, hogy ciklikusan megvizsgálja a kijelölt bemeneti bit állapotát. 2) megszakítás: az esemény maga jelzi a processzor számára állapotának megváltozását. A megszakítás az eredetileg futó pr utasításainak végrehajtását leállítja, és a processzor un megszakítási alprogramot hajt végre, ami az esemény kezelését elvégzi, majd ennek befejezésével a processzor visszatér a megszakított pr végrehajtására. A megszakítás olyan speciális szubrutinhívás, amelynél a hívás bekövetkezésének időpontját nem tudjuk. Olyan rendszerekben, ahol több esemény okozhat megszakítást a megszakítások prioritása dönti el a kiszolgálási sorrendet, ha egyszerre két megszakítás is fellép. A pr futása nem minden esetben szakítható meg káros következmények nélkül => a legtöbb rendszer biztosítja, hogy a megszakítások programból tilthatók/engedélyezhetők legyenek.

35,Milyen időbeli viszonyokra kell ügyelnünk a megszakítások használatánál?

A megfelelő működés érdekében figyelnünk kell a megszakított főprogram és a megszakított alprogram viszonyára. CPU két programot hajt végre: periódikusan a megszakítási programot, fennmaradt időben a főprogramot. A főprogram csak a megszakítási pr befejezése után, a köv megszakítás kezdetéig futhat. A programok időbeli viselkedését a tervezés során gondosan elemezni kell.

36, Hogyan lehet programozni a PIC-eket?

A tokok programozásakor a prmemóriát és esetleg a belső EEPROM adatmemóriát töltjük fel az adott tartalommal. Programozási szempontból 3 típusú tok áll rendelkezésre: 1)egyszer programozható OTP tok; 2) ugyanezen tokok kvarcdarabokkal ellátott ultraibolya fénnyel történő EPROM-os változatai; 3)elektromosan törölhető flash EPROM tokok. Programozói állapotba kerülünk, ha a MCLR lábat VPP feszültségre emeljük, ezután a programozó berendezés a CLOCK lábon órajelekkel érvényesítve elküld egy parancsot, majd a tok az órajellel ütemezve RB7 lábon keresztül válaszol. A programozás történhet sorosan, ill. párhuzamosan (több tok lábát felhasználva).

37, Ismertesse röviden a PIC perifériákat.

A perifériákat programozási szempontból 4 bitcsoporttal jellemezhetjük: 1)parancsbitek=ezekkel állítjuk be a periféria működésmódját; 2)statuszbitek=ezeket a periféria állítja, jelezve vmilyen állapotát; 3)bemeneti adatbitek=ezek érkeznek a külvilág felől a perifériába. Periféria állapotának figyelése(polling, megszakítás 34.tétel).

Perifériák: 1) minden tokban megtalálható közös elemek: I/O lábak, 8 bites számláló/időzítő, watchdog; 2)családelvet megvalósító perifériák: TMR1, TMR2, számlálók/időzítő; A/D konverter; belső adat EEPROM; soros I/O; PWM; CAPTURE/COMPARE modulok; párhuzamos SLAVE port; LCD modul

38, PIC I/O portok működése.

A kontroller lábkivezetéseinek állapotát tudjuk a prba beolvasni, ill a prból a 0 és 1 állapotnak megfelelően, a lábak feszültségét befolyásolni. A programhoz rendelt I/O regisztereket pontosan úgy kezeljük, mint az egyéb fileregeket. A portok olvasásakor mindig a láb állapotának beolvasása történik meg. A portok írása ténylegesen egy D tárolóba történő beírás. RESET hatására minden port bemenet lesz. A TRIS regiszter tartalmával tudjuk a port irányát beállítani (1=in, 0=out). Bármelyik utasítás, amely az I/O regiszterek vmelyikét vagy vmelyik bitjét használja operandusként, a művelet végzése előtt a teljes 8 bites értéket olvassa be a lábakról (READ), elvégzi a műveletet (MODIFY), majd azt írja vissza az adatokba (WRITE). Átvitel módja: soros/párhuzamos. I/O portok regiszterei: parancs; státusz; adatkimenet; adatbemenet. I/O portok elérése lehet közvetlen, ill. tárolóhoz rendelt módon

39, Számlálók és időzítők a PIC-ben. TMR0, TMR1, TMR2

A számláló áramköröket perifériaként használva a processzort tudjuk tehermentesíteni: a számláló bemenetére egy külső esemény esetén megjelenő jelváltást kapcsolva, a számláló önállóan képes a külső események számlálására. A processzornak csak le kell kérdezni a számláló tartalmát. Ha a kontrollert működtető leosztott órajelet juttatunk erre a számláló bemenetre, akkor a számláló tartalma és az órajel periódus idejének szorzata az eltelt idővel arányos. A számláló által átfogható számlálási tartomány növelése miatt a számlálót alkotó flip-flopokból álló számlálólánc hosszát növelik meg.

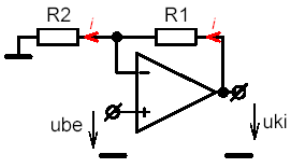
TMR0: időzítő áramkör; 2 bemenetű multiplexe; kontroll reg-gel beállítani: ha 0 – belső jel, ha 1 – külső jel. TMR1: 16 bites + felbontást növelő elosztó + bemenet lehetőség: egy kisfrekvenciás kvarcosszc.; létezik asszinkron számláló üzemmód. TMR2: 8 bites, soros adatátviteli egység ütemadójaként használjuk; periódusreg: nadott érték elérése->saját magát törli; van elő és utóosztója

40, PWM modulátor működési elve és felhasználási lehetőségei.

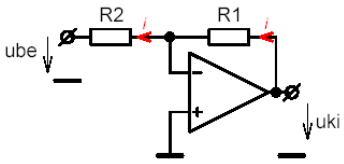
PWM modul segítségével impulzusszélesség-modulációt tudunk létrehozni: állandó T periódusidő mellett a bekapcsolás w idejét változtatva, a kimenő jel közértékét tudjuk változtatni. Periudikus jel, az impulzusszélesség aránya hordozza az info-t, 8 bites pontosságú adatokat szeretnénk átvinni (= 8 bites felbontású PWM jel): nem bitenként, hanem analóg módon kódolunk.

Műveleti erősítők:

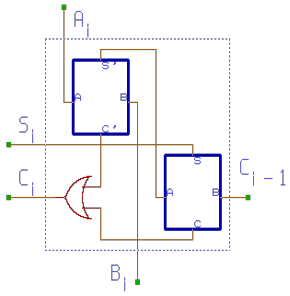
Nem invertáló erősítő



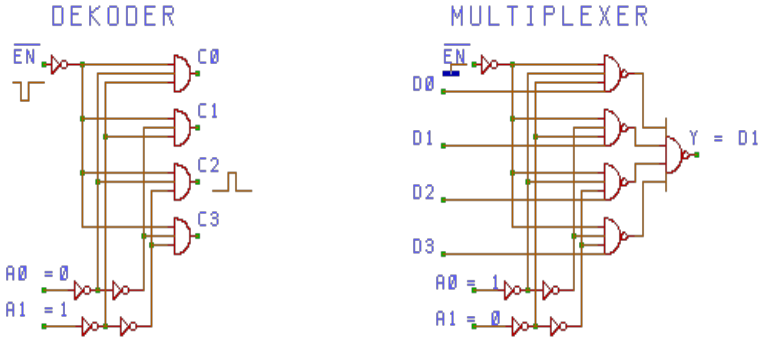
Invertáló erősítő



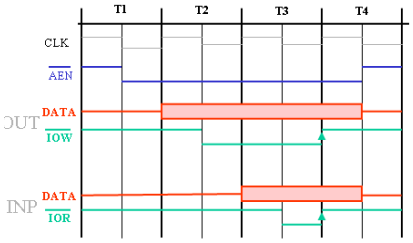
Teljes összeadó:



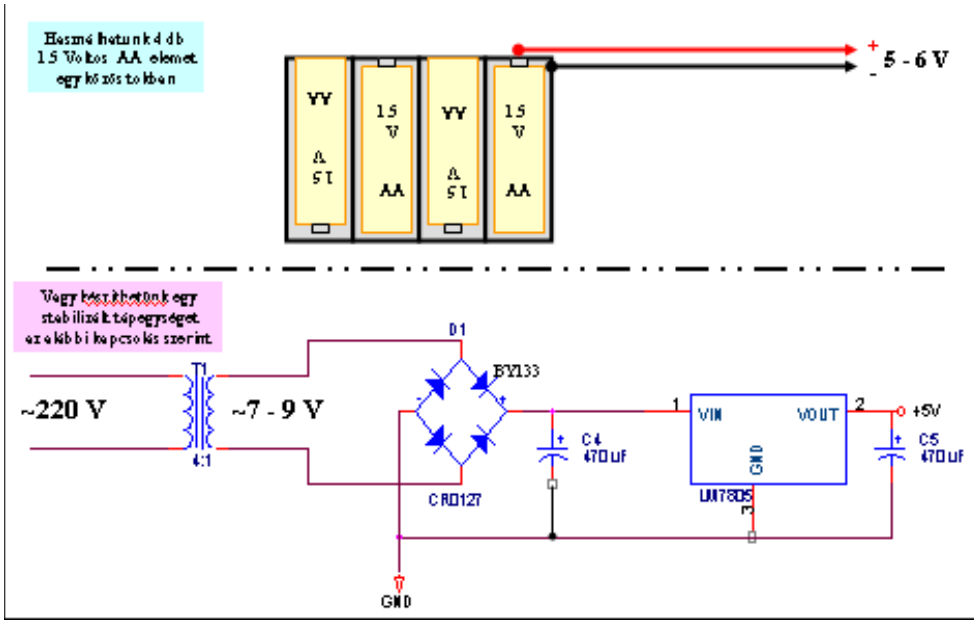
Multiplexer, dekódoló



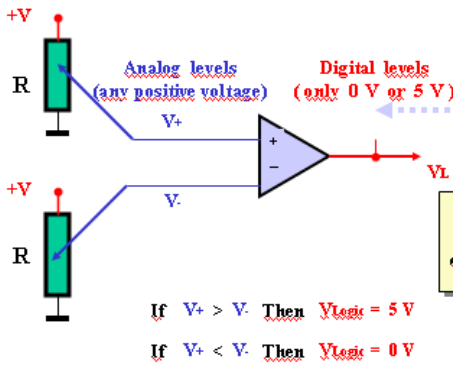
I/O idődiagram



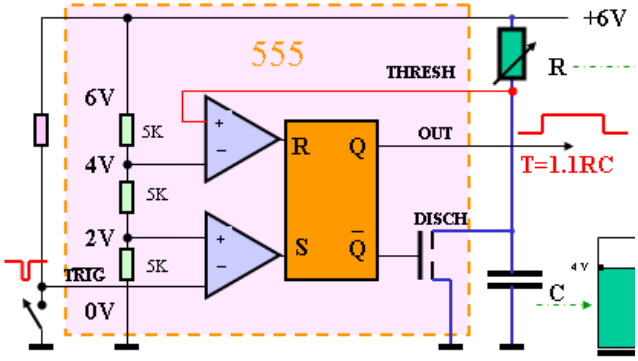
Tápfeszültség forrás/ Stabilizált tápegység:



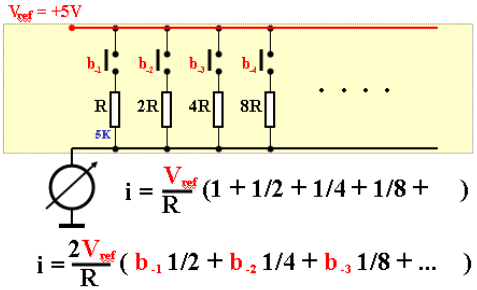
Analóg komparátor:



555ös időzítő áramkör:



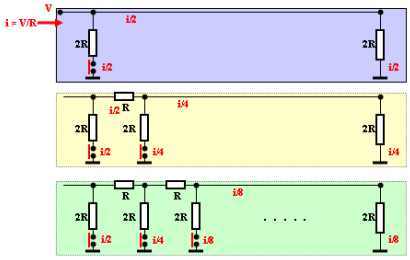
Ellenállás létra



$$i = \frac{V_{ref}}{R}(1 + 1/2 + 1/4 + 1/8 + \quad)$$

$$i = \frac{2V_{ref}}{R}(b_{-1}1/2 + b_{-2}1/4 + b_{-3}1/8 + \ldots \quad)$$

R-2R létra



D/A konverter