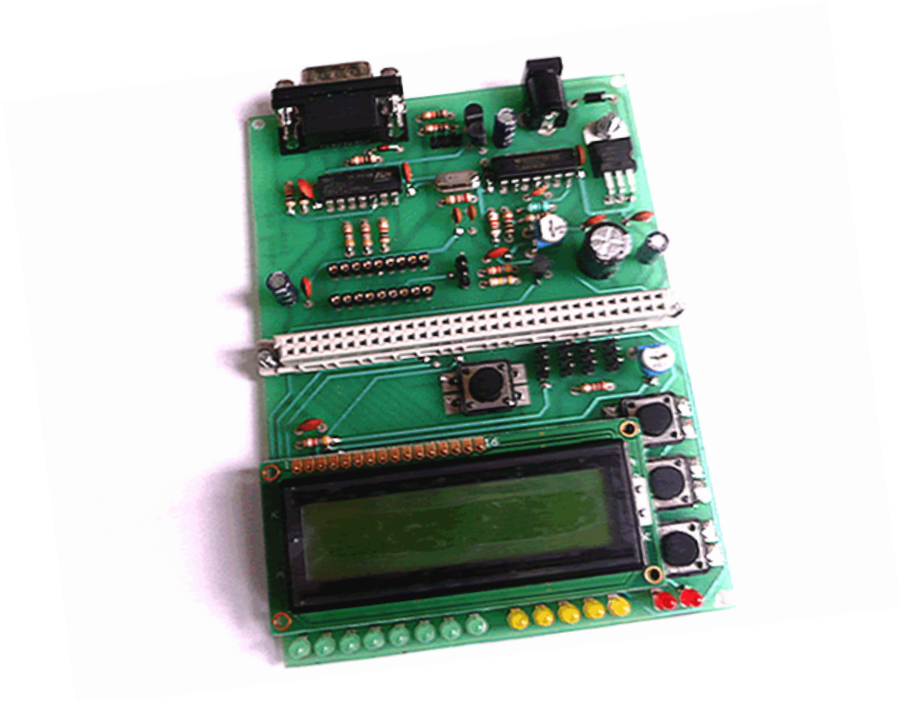
EPSICOM

Ready Prototyping

Colecția IT

EP 0201



Cuprins

PIC16F84
SINGLE BOARD COMPUTER

- Avantaj Pret/Calitate
- Livrare rapida
- Design Industrial
- Proiecte Modificabile
- Adaptabile cu alte module
- Module usor de asamblat
- Idei Interesante

Idei pentru afaceri

Hobby & Proiecte Educationale

www.epsicom.com/kits.php

a division of **EPSICO Manufacturing**

Constructie On Board, numit şi Single-Board computer pentru mult prea cunoscutul microcontroller PIC16F84, este poate cea mai comună placă cu utilizaredirectă, uneori finală, la aplicațiile noastre.

Funcționare

Pe acest board se poate testa orice program pe arhitectură realiyată cu PIC16F84, cu ceasul de 4 - 20MHz.

Clock - cu comutatorul K2 se selectează regimul: ceas cu quartz sau ceas realizat cu un oscilator RC la care frecvența se ajusteaza din P1.

Reset -cu switch-butonul S3 direct pe MCLR

Programarea – in-circuit prin schimbarea tensiunii pe MCLR la 13.5V cu ajutorul unui circuit TL497 tip step-up convertor. Regimul este ales cu S1 şi semnalizat cu D21.

RS232 - conectarea serială cu PC-ul se face printr-un conector DB9 **numai în regim off** (nealimentat). Semnalele DATA şi CLOCK sunt trimise pe RB6 si RB7 prin driverele IC4C şi d iar prin IC4b şi a se transmite către interfată semnalul de pe RB7, respectiv reset şi regim lucru/programare.

Afisaj Led-uri, starile RA si RB sunt afisate cu led-urile D8-D20. Afisarea starilor sunt de un real folos la depanarea programelor la frecvente mici. Regimul de afisare este stabilit cu S8.

Tastatura - cu trei push-butoane S5, S6 şi S7 se modifică starea intrărilor RA1 şi RA0 (separat sau împreună prin SAU cablat) permițând setarea sistemului într-un mod similar cu setarea unui ceas (regim, up-down).

Afişarea alfanumerică cu LCD 2x16 caractere este controlată cu semnalele E (validare) RS (selecție registru) conectate la RA2 şi RA3. Pinul R/W al afişajului este conectat la masă, regim scriere. Cu S9 validăm afişarea pe LCD iar cu S4 şi P2 avem controlul luminii şi contrastului.

Alimentarea se face de la o sursă externă de 9-12V prin K1 şi este semnalizată cu D22. Cu stabilizatorul IC1 obținem tensiunea de 5V.

Programe

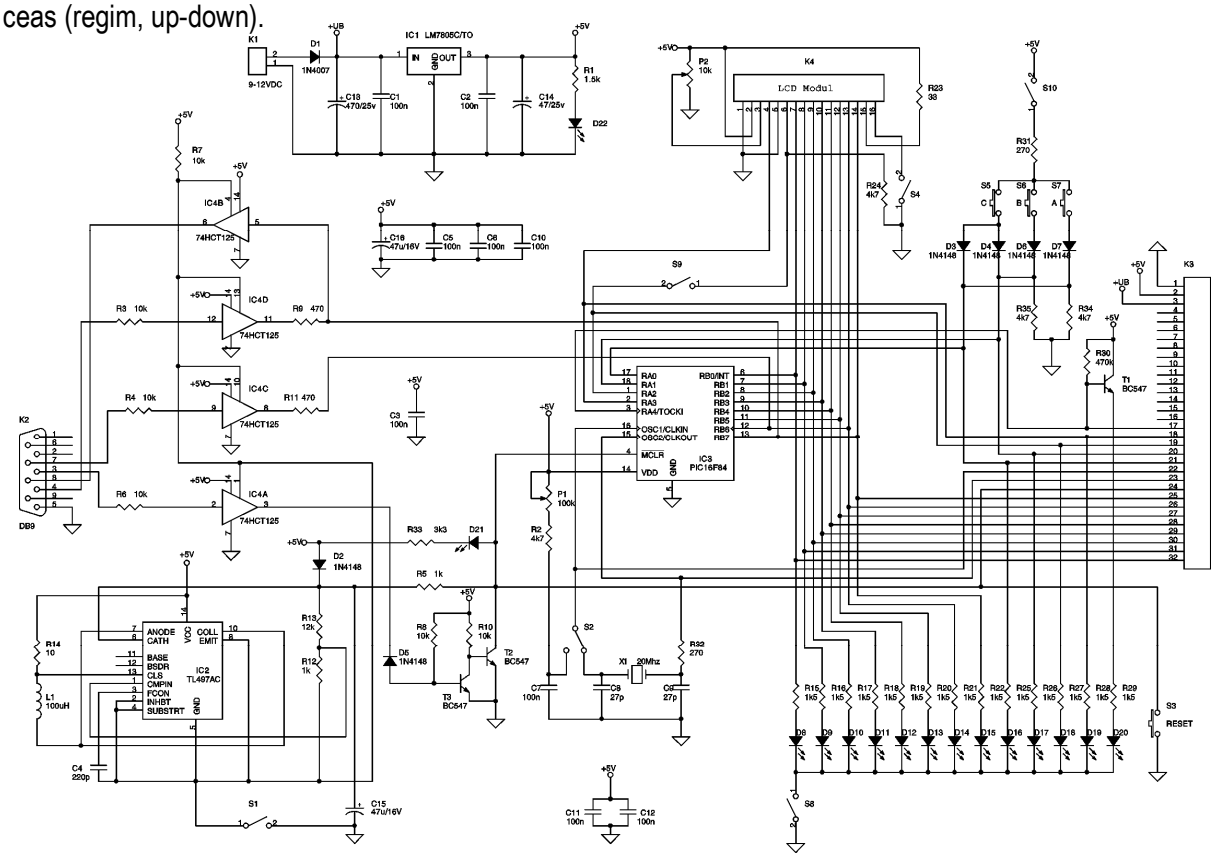
Sunt necesare câteva programe pentru inițiere si utilizare acestui sistem (programe de test pe care le găsim rapid pe internet). Astfel MPLAB.EXE (editor, assembler şi simulator) şi MPASM.EXE (assembler) şi PSIM.EXE (simulator) sunt downloadabile free de pe internet, **NTPicprog**, de la adresa **http://home.swipnet.se/~w24528/NTPicprog** , **ICPROG.EXE** de la **www.ic-prog.com** e super.

Alte resurse de programare ?

http://www.microchip.com

http://www.wolfgang-kynast.de/pic.htm

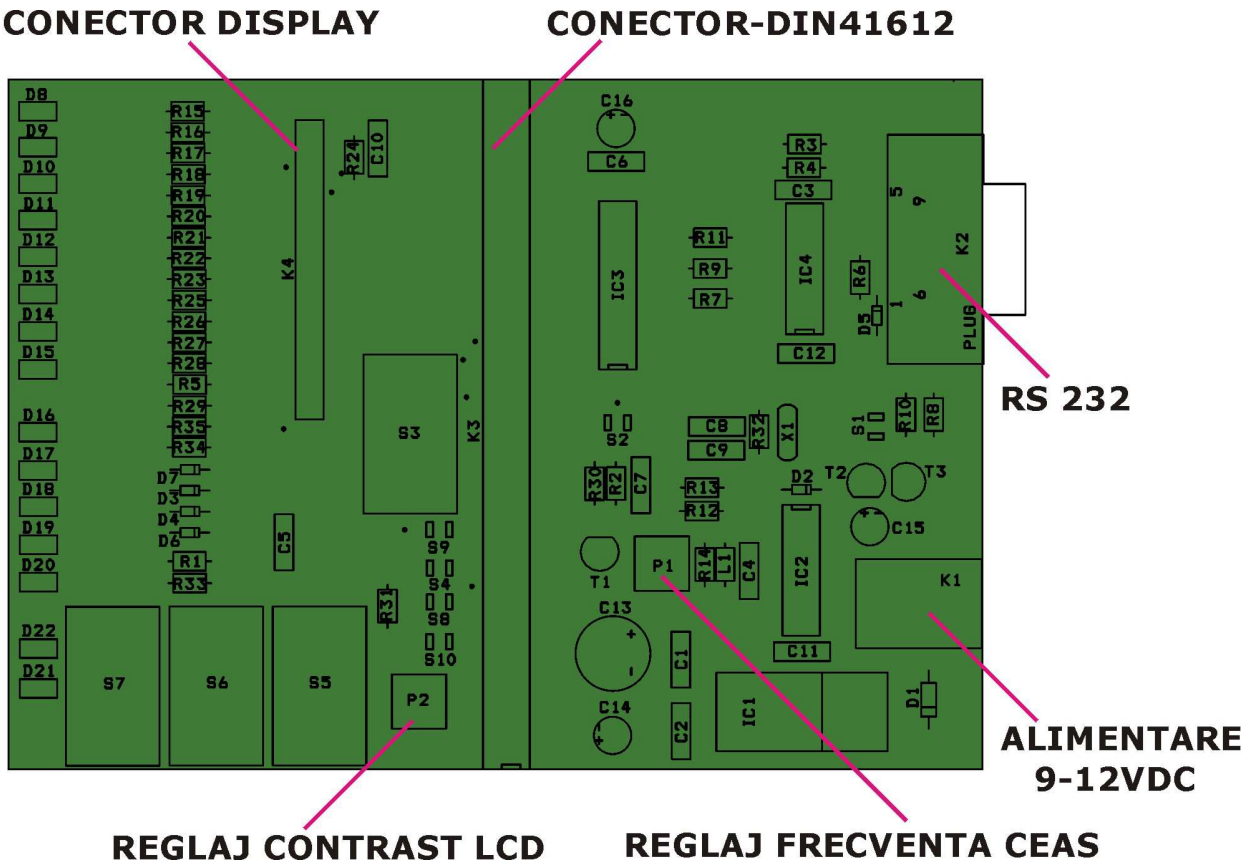
http://www.ludwig-geissler-schule.de/docs/picee/picee.html



Schema electrică

Lista de componente

Nr.Crt.	Componenta	Denumire	Valoare	Cant
1	C1,C2,C3,C5,C6,C7,C10, C11,C12	Condensator NP	100nF	9
2	C4	Condensator NP	220pF	1
3	C9,C8	Condensator NP	27pF	2
4	C13	Condensator POL	470µF/25V	1
5	C14	Condensator POL	47µF/25V	1
6	C16,C15	Condensator POL	47µF/16V	2
7	D1	Diodă	1N4007	1
8	D2,D3,D4,D5,D6,D7	Diodă	1N4148	6
9	D8, D9,D10,D11,D12,D13,D14, D15,D16,D17,D18,D19,D20,D21,D22	Led	LED	15
10	IC1	C.I.	LM7805C	1
11	IC2	C.I.	TL497AC	1
12	IC3	C.I.	PIC16F84	1
13	IC4	C.I.	74HCT125	1
14	K1	Mufa Alimentare	9-12VDC	1
15	K2	Mufa	DB9	1
16	L1	Bobina	100µH	1
17	P1	Semireglabil	100KΩ	1
18	P2	Semireglabil	10KΩ	1
19	R3,R4,R6,R7,R8,R10	Rezistenţă	10KΩ	6
20	R1	Rezistenţă	1.5KΩ	1
21	R2,R24,R34,R35	Rezistenţă	4,7KΩ	4
22	R12,R5	Rezistenţă	1KΩ	2
23	R9,R11	Rezistenţă	470Ω	2
24	R13	Rezistenţă	12KΩ	1
25	R14	Rezistenţă	10Ω	1
26	R15,R16,R17,R18,R19,R20, R21, R22,R25,R26,R27,R28,R29	Rezistenţă	1,5KΩ	13
27	R23	Rezistenţă	33Ω	1
28	R30	Rezistenţă	470KΩ	1
29	R31,R32	Rezistenţă	270Ω	2
30	R33	Rezistenţă	3,3KΩ	1
31	S3, S5, S6, S7	Pushbuton	PUSHBUTON	4
32	T1,T2,T3	Tranzistor	BC547	3
33	X1	Quartz	20Mhz	1
34	K3	Conector	DIN41612	1
35	K4	Conector Display	SIL-16	1
36	S1,S4,S8, S9,S10	Comutator Simplu	JUMPER-3	5
37	S2	Comutator Dublu	JUMPER-3	1



Amplasarea componentelor

Bibliografie

Multi PIC Programmer 5 Ver.2

WinPicProg 1.91

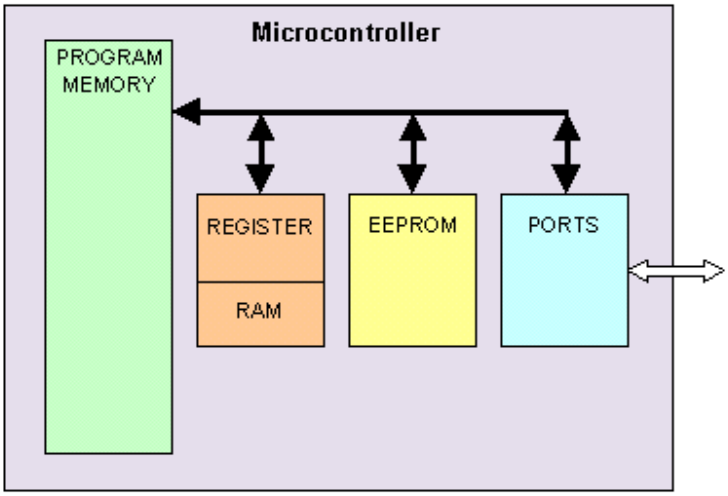
Acest produs se livrează în varianta circuit imprimat, circuit imprimat + componente sau în varianta asamblată în scopuri educaționale și va fi însoțit de documentația completă de asamblare pe CD.

Câteva cuvinte despre structura microcontrollerelor

Cu ajutorul acesor prime informații veți intra în superba lume a mirocontrollerelor.
În primul rând să știm cu ce lucrăm, cum funcționează și cum putem realiza programe pentru ele.
Câteva linii de program reușite și câteva exemple revelatoare vă vor da impulsul la a transforma un chip într-o jucărie teribilă.
Ce ne trebuie pentru început ? Curajul abordării, în fond nu este complicat deloc ! Totul are un început iar apoi vine și performanța.

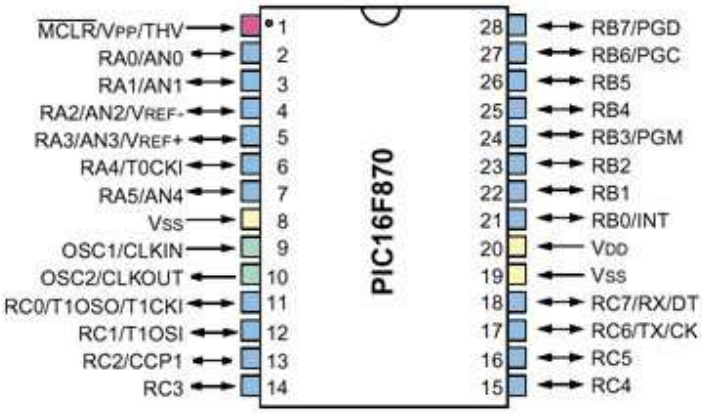
Schema bloc simplificată

La PIC16F84 sau PIC16F870 vom găsi şi vom lucra cu acest grup de patru blocuri.



Memoria program (verde) va conține programul pe care îl înscriem. Programul (Software-ul) conține un set de date și instrucțiuni pe care microcontrolerul le va prelucra. Ele vor fi scrise într-un program cu ajutorul unui PC, instrucțiuni pe care le înscriem apoi în controller, adică le programăm cu un programator în "memoria program".
Această memorie este de tip EEPROM care poate fi rescrisă de mii de ori. Cum o înscriem ? Cu un programator pe care îl veți putea găsi în colectia IT (EP0050 de exemplu pentru o gamă largă de microcontrollere produse de firma MICROCHIP).
Regiştrii şi blocul RAM (portocaliu) conține toate registrele interne precum şi o mică memorie RAM (64-128 octeti) unde puteți stoca date temporare. Există mai mulți regiștrii cu funcții diferite ce vor fi explicați în continuare.
Pentru ce sunt necesari ? Să luăm un exemplu: dacă faci o buclă program, atunci ai nevoie de o variabilă pentru a schimba valoarea de fiecare dată când bucla se execută şi apoi folosim o variabilă definită într-o adresă RAM pentru a pastra valoarea contorului. (Vom reveni mai târziu cu un exemplu edificator, nu este complicat). Conținutul registrului şi datele din RAM vor dispărea la oprirea tensiunii de alimentare, sunt doar date provizorii, temporare.
O altă zonă de memorie, care va funcționa la fel ca și RAM-ul, este memoria **EEPROM (galben)** Aceasta este o memorie mică unde puteți citi precum şi scrie date, însă datele nu vor dispărea atunci când tensiunea dispare. Datele din aceasta memorie pot fi adresate, citite, modificate sau descărcate, însa pot fi și protejate la citire cu un cod (lock).
Ultimul bloc este cel cu **porturi**, adică pinii de intrare sau ieșire pe care îi definim prin program ca intrări sau ieșiri cu funcțiile pe care le dorim (să citim anumite date prin pinii-port de intrare și să oferim datele rezultate din program către anumite circuite cum ar fi de comandă unor elemente de forță, …). În exemplele ce vor urma afla cum și la ce se folosesc.
PIC16F870 este o versiune apărută ulterioar procesorului PIC16F84, ambele circuite conțin mai multe blocuri decât cele descrise. Trecând de la prezentarea generală trecem la citirea fişelor tehnice ale celor două circuite pentru a obține detaliile fiecărui bloc, pentru a le utiliza numai dacă avem strict nevoie de ele. Ambele controlere au timere, watchdogs, sistem de întreruperi şi multe alte blocuri.

Să luăm un exemplu: circuitul PIC16F870



Pentru o mai bună selecție pe categorii a acestora, s-au colorat.
Astfel:
- Pinii desenați cu albastru sunt porturi (pini de intrare/ieşire). Deoarece nu există multe extra funcții în acest microcontroller, unii pini pot fi folosiți în mai multe scopuri: ca intrări la un convertor AD (analog/numeric) intern iar alți pini pot fi conectați la un contor intern, etc. Prima oară vom afla cum utilizăm pinii albaștrii ca simple intrări şi ieşiri.
- Cei doi pini verzi ar trebui să fie conectați la un cuarț pentru a obține un semnal de ceas intern, generatorul de impulsuri de tact.
- Pinii galbeni sunt dedicați alimentării circuitului;
- Pinul roşu este intrarea Reset care va inițializa funcționarea circuitului.

Până aici a fost relativ simplu de înțeles.
Poate părea uneori greu şi plictisitor, însă observațiile din exemple ne scapă mai tot timpul de explicarea detaliilor.

Vrem să învățăm să programăm și nu știm cu ce să începem? De ce anume mai avem nevoie pentru a ne pune la treabă ?

Etapa I

- În primul rând avem nevoie de o platformă de evaluare/dezvoltare a unor circuite realizate cu microcontrollere.
- Luăm un exemplu de circuit din manual sau de pe net, configurăm circuitul și testăm programele deja realizate. Exemplele sunt un mod rapid de învățare. Deschidem fișierele ASM, citim comentariile făcute pentru fiecare linie și vom înțelege foarte repede modul în care funcționează microcontrolerul, vom învăța de asemenea un set minim de instrucțiuni. Foile de catalog ale fiecărui procesor conțin detalii referitoare la instrucțiuni.

Etapa II

Iată o carte EBOOK de bază despre PIC16F84.
Aici gasim o documentație excelentă pentru înțelegerea funcționării și elemente de programare pentrPIC16F84.
Merită citită, este gratuită.

Ce putem face, de exemplu, pentru a aprinde un LED în primă fază și să îl facem apoi să clipească ?

- vom scrie câteva linii de program și le vom salva într-un fisier cu extensia .asm (le găsim în cartea de mai sus). Pentru o întelegere rapidă a procedurii și o corectare rapidă, liniile de program din .asm sunt și vor fi însoțite de comentarii.
- îl asamblăm cu un program numit MPASM (vezi mai jos),
- îl compilăm rezultând un fisier .hex în cod mașină și un fișier .lst de verificare (erorile apărute în limbajul de asamblare pe linia de program vor fi regăsite ca mesaje în fișierul .lst). Fisierul .hex va fi înscris în format recunoscut de procesor.
- Verificați liniile de program și apoi funcționarea aplicației programând controllerul (vezi mai jos).

Etapa III

Ce mai rămâne de făcut ?
Dacă ați ajuns la acest rând înseamnă că ați facut deja foarte multe. De aici înainte folosiți imaginația pentru a realiza aplicații cât mai interesante. Mult succes cu proiectele voastre și nu uitați că cei mai buni profesioniști au fost autodidacții !

Asamblorul

Există multe asambloare pentru PIC16F84 și 16F870 însă ce poate fi mai la îndemână decât programul oferit gratuit de producătorul MICROCHIP , ultima versiune de **MPLAB** ? Microchip a realizat un mediu software complet unde puteți simula software-ul vostru înainte de a realiza circuitul. Acest mediu este numit MPLAB. **Development software (MPLAB® IDE)**

Programatorul

Daca ați asamblat codul veți obține un fisier hex. pe care va trebui să îl înscrieți în microcontroller, adică să programați microcontrollerul.
Pentru a face acest lucru avem nevoie de o interfață (Programator PIC) de la computer la microcontroler şi de un program software cu care să transferăm fișierul din PC în microcontroller.
Iată un programator util pentru o gamă largă de microcontrollere din seria MICROCHIP, cod **EP0050** realizat și de firma noastră.
Dupa instalarea programului și cuplarea programatorului pe portul serial, se lansează programul, se selectează controllerul PIC16F870 și se setează opțiunile de programare (configurare):

Fuserele:
- Watchdogtimer (pagina 101 datasheet)
- Poweruptimer (pagina 89 datasheet)
- Brow out resetare permite (pagina 94 datasheet)
- Low Voltage Programming Enable (Activare tensiune mica de programare)
- Code Protection Data Enable (Activare Codul de protecție a datelor)
- Flash Program Memory Write Enable
- Brow out reset enable (pagina 103 datasheets)

Fișierele necesare de la Microcip: **PIC16F870 Datasheet , Programarea specifică pentru PIC16F87X**