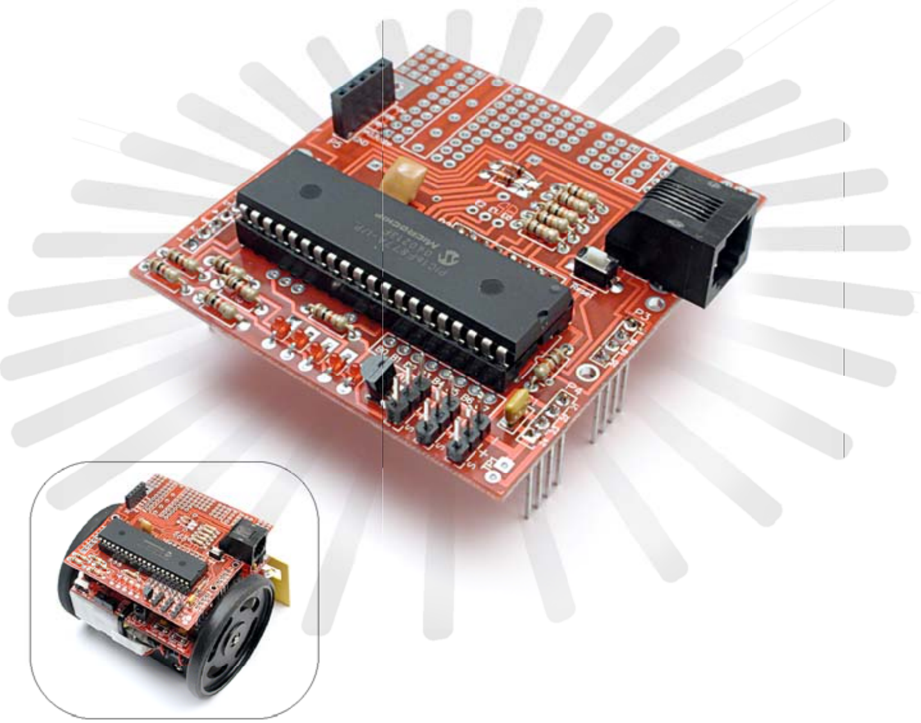
The Solarbotics SUMOVORE

Microchip PIC16F877A Brainboard Add-on



This brainboard uses the popular and powerful Microchip PIC16F877A microcontroller (included!) to take over the functions of the Discrete Brain that comes with your Sumovore.

The PIC16F877A is part of a *very* popular family of microcontrollers, and is programmable in an ever-growing list of languages (some free!).

It's fast, inexpensive, and very powerful. An ideal mate to the Sumovore!

(Sumovore Sumo robot kit and DB25 printer cable/connector req'd)

Produced by
SOLARBOTICS® Ltd
Document Release: June 26, 2004

We strongly suggest you inventory the parts in your kit to make sure you have all the parts listed. Use a pen, pencil, pricked finger, chocolate bar - anything to mark off the items. If anything is missing, contact us for replacement parts information.

Disclaimer of Liability

Solarbotics Ltd. is not responsible for any special, incidental, or consequential damages resulting from any breach of warranty, or under any legal theory, including lost profits, downtime, good-will, damage to or replacement of equipment or property, and any costs or recovering of any material or goods associated with the assembly or use of this product. Solarbotics Ltd. reserves the right to make substitutions and changes to this product without prior notice. (Sorry, gotta make the lawyer happy)

The PIC16F877A Brainboard

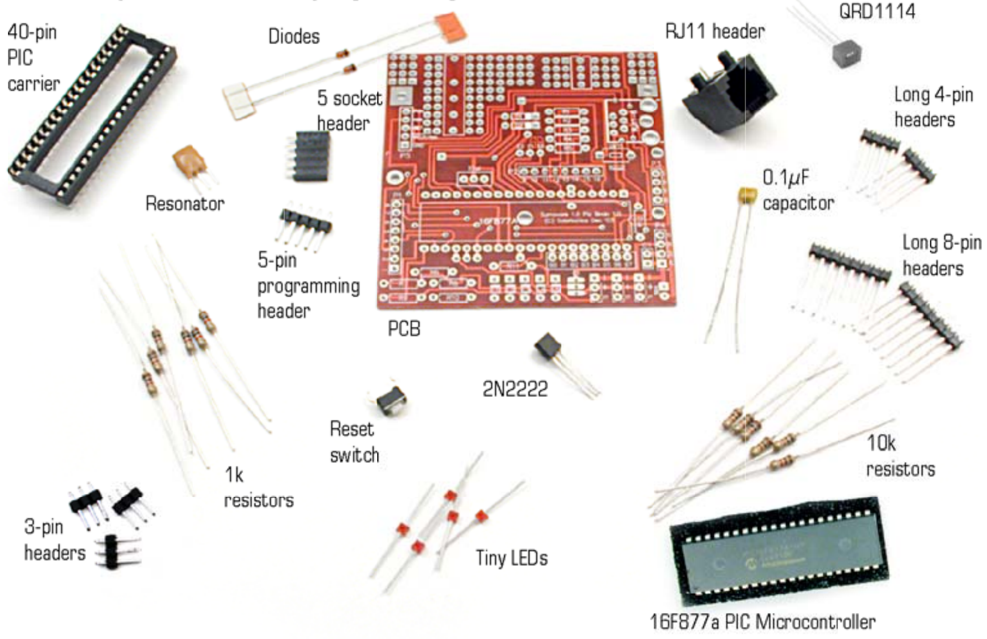
Parts List

16F877A Brainboard Components

1 - Printed Circuit Board (PCB)
5 - Tiny LEDs
1 - 2N2222 NPN Transistor
1 - 0.1μF capacitor (labeled '104')
2 - Diodes
1 - 20MHz Resonator
6 - 1k Resistors (Black/Brown/Red)
6 - 10k Resistors (Black/Brown/Orange)
1 - PIC16F877a 40 pin wide carrier
1 - RJ11-6 Telephone-style programming connector
1 - Microchip PIC16F877A Microcontroller
1 - SPST Push Button Reset Switch
1 - 5-Socket programming header
1 - 5-Pin Header (for building optional programming cable)
2 - 4-Pin Sumovore interface long headers
2 - 8-Pin Sumovore interface long headers
3 - 3-Pin Headers (for optional servo headers)
1 - QRD1114 edge sensor (for Sumovore's middle sensor)

Tools Required

Soldering equipment
Side-cutters or fine snips
Computer for further programming



Looking for a more flexibility out of your Sumovore? Well, welcome to the PIC brainboard - the 16F877A. The '877 offers an impressive list of features, including (but not limited to):

- 20MHz /4 MIPS processing speed
- 33 programmable Input/Output (I/O) lines (that's lots!)
- Eight 10-bit analog-to-digital converters (ADCs)
- Two Pulse-width modulation (PWM) channels
- Only 35 instructions (good for you assembler masochists)
- Two analog comparators
- In-circuit programmable (which we make use of)
- 8kB flash memory, 368 bytes RAM, and 256 bytes EEPROM

We've barely made a dent in what the '877 is capable of, and there's lots more you can do with it. Being a PIC microcontroller which have been around for quite a while, there is a considerable amount of information and software for it on the Internet.

This is **not** a kit for a microcontroller beginner. Anybody using this brainboard should have the appropriate skills, or be ready to learn the techniques that make a microcontroller... microcontrol!

This kit lets you swap out the default discrete brainboard for a programmable version. If you run into any problems, it's a simple process to swap a different brain back in. Didn't you ever have days where that'd be a handy feature for you to have (umm...for the robot, we mean).

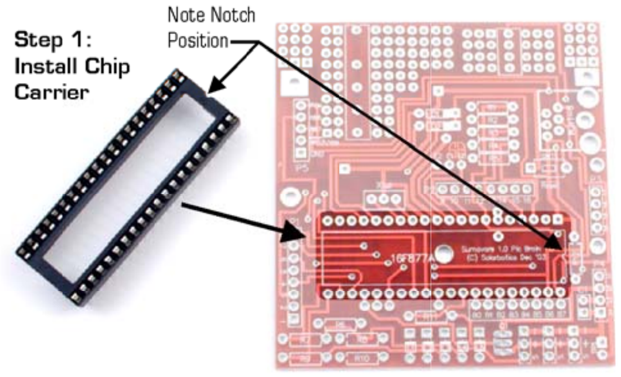
This kit features:

- Microchip PIC 16F877A microcontroller
- 5 indicator LEDs
- RJ11-6 (telephone jack) and 5-pin programming headers
- Three servo (or similar peripheral) headers
- Extra breadboarding space and hard-point mounts
- Microprocessor Reset Switch

We designed the breadboarding space to accommodate extra ICs and support electronics, or simply as a place to mount a servo with double-sided sticky tape! It's a flexible area - use it for whatever strikes your fancy.

Step 1 - '877 Carrier: Everybody wants to start with the most important-looking parts. Well, tough - you can't. You keep your paws off that pretty PIC until the end, as the whole point of the carrier is to protect the chip from seeing any potential ESD (static electricity) which could damage it. So, the next best thing to attack is installing the carrier itself.

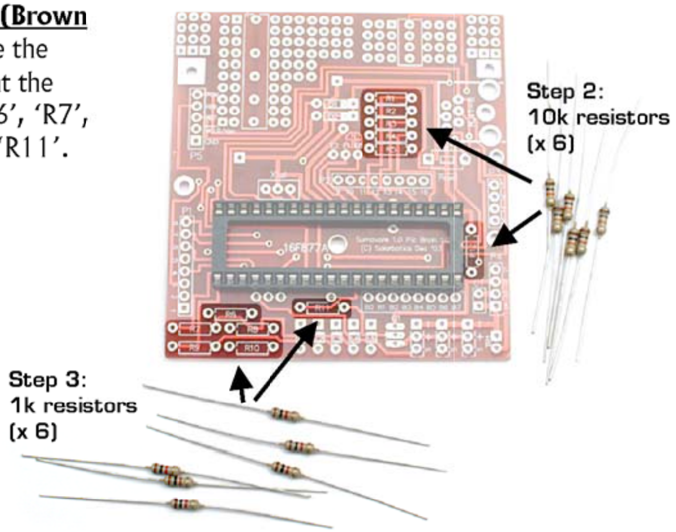
Note that there is a notch on one end. Although not critical, it's nice to keep the notch aligned with the notch drawn on the printed circuit board (PCB) so you're sure to install the microcontroller properly when the time comes.


Step 1: Install Chip Carrier

Note Notch Position

Step 2 - 10k Resistors (Brown / Black / Orange): Pretty easy stuff here. Use your considerable soldering experience (you built the Sumovore yourself, right?), and install these resistors into positions 'R1', 'R2', 'R3', 'R4', 'R5', and... 'R12'?!? (Gotta have a talk with the guy who designed this PCB. R12 - sheesh! Yes, install one into 'R12' as well.

Step 3 - 1k Resistors (Brown / Black / Red): Just like the 10k, but in the group at the bottom in positions 'R6', 'R7', 'R8', 'R9', 'R10', and 'R11'.

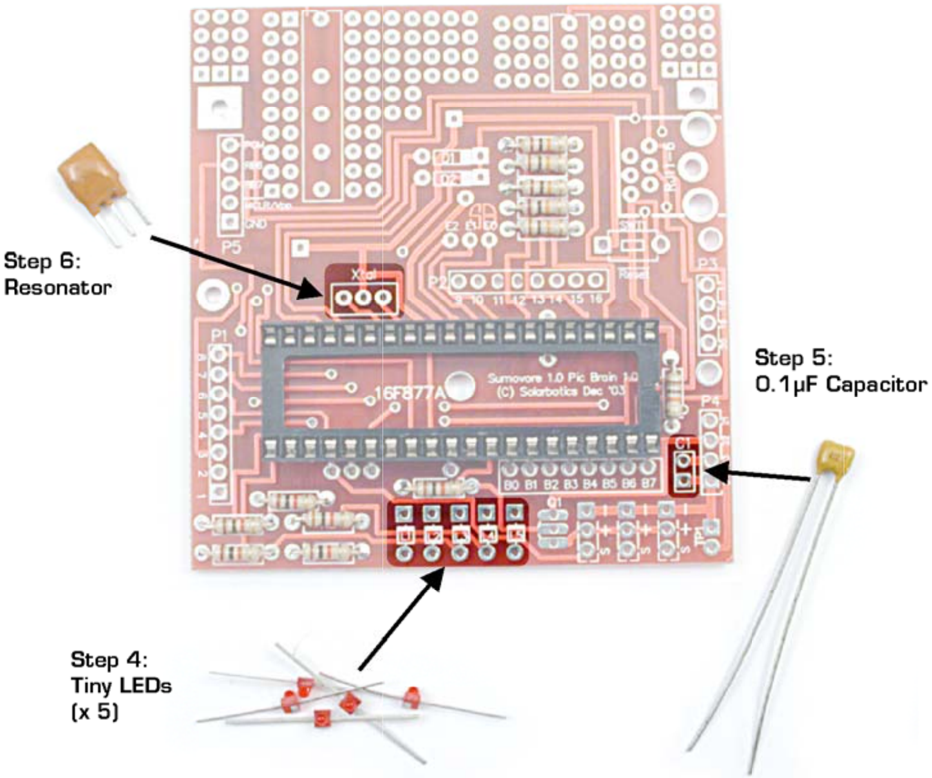

Step 2: 10k resistors (x 6)

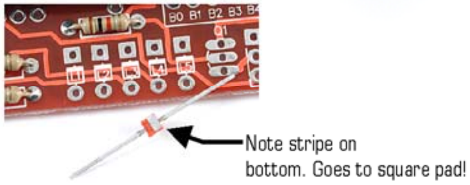
Step 3: 1k resistors (x 6)

Step 4, 5 & 6 - Tiny LEDs, Capacitor, and Resonator: Although not critical to the performance of your Sumovore PICBrain, blinky lights are always helpful. Trust us. Install them into locations 'L1', 'L2', 'L3', 'L4', and 'L5'. Note that the LED has a white stripe that indicates the lead to be inserted into the square pad, ok!?

The 0.1μF "decoupling" capacitor is installed into position 'C1', in any orientation you prefer. Good for keeping the microcontroller power clean!

The 20MHz resonator is installed into position 'Xtal' (which is short-hand for crystal, which a resonator contains). This device isn't polarity-sensitive either.



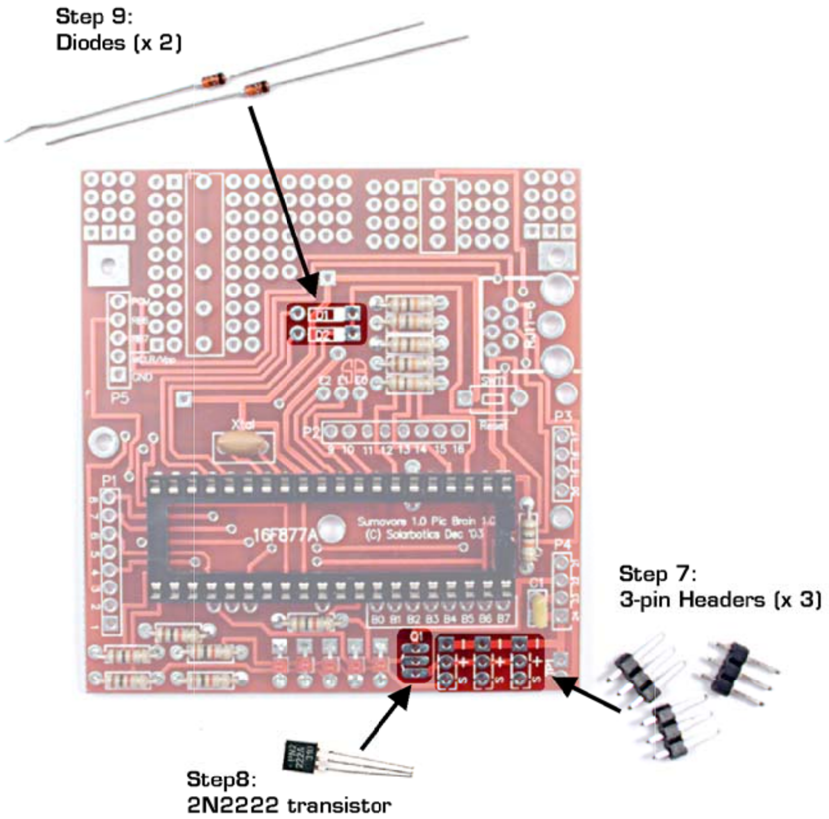


Step 7, 8 & 9 - Transistor, 3-Pin Headers, and Diodes: The transistor installs into position 'Q1', with the curved surface facing to the **right**. You can just see a small part of the curve on the PCB surface to help you identify the correct orientation. This transistor powers the three auxiliary ports (usually used for servos), and isolates them from the batteries when the main power switch is turned off.

The three 3-pin headers are installed at the positions indicated (which should be labeled left-to-right 'S1', 'S2', and 'S3'. Sorry!)

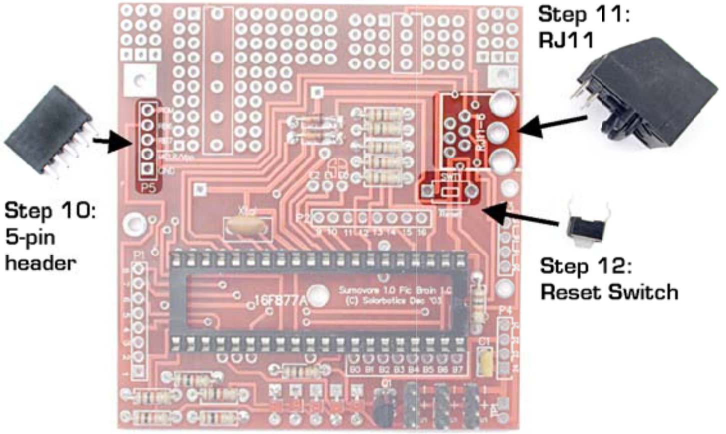
The diodes are installed at positions 'D1' and 'D2'. Watch that the band orientation matches what's printed on the PCB!

Note: The three 3-pin headers are driven through the 'Q1' transistor from the unregulated 6V provided by the quad-AA pack on the Sumovore body. We designed it this way to offer more power to any larger accessory loads like servos that can run at higher voltages than the regulated 5V.



Step 10, 11, & 12 - 5-Pin Header, RJ-11, and Reset Switch: The 5-pin header is most likely going to be your primary method of programming your '877. Install it at location 'P5'. The RJ11 socket is another popular way of connecting the '877 to a programmer, and it's installed at location 'RJ11-6'.

Note: You don't have to install both if you know how you'll be interfacing the microcontroller. We supply you two methods for maximum flexibility.


Step 11: RJ11

Step 10: 5-pin header

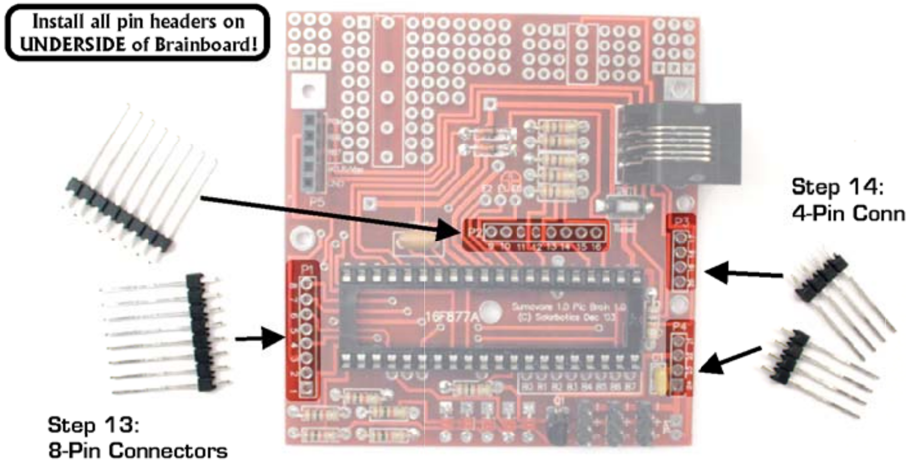
Step 12: Reset Switch

The single-pole, single-throw (SPST) microcontroller reset switch is installed at location 'SWT1'. Nothing to mysterious here!

Step 13 & 14 - 4-Pin and 8-Pin Connector Strips: Let's start by making this very clear: **These parts are installed on the underside of the pcb.** Not this side. Ok? Alright? Kapeesh? Ok - carry on.

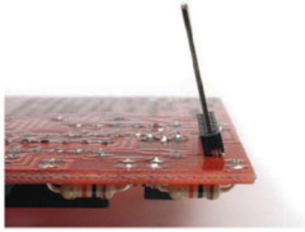
See the next page for details on how to get these lined up juuuust right!

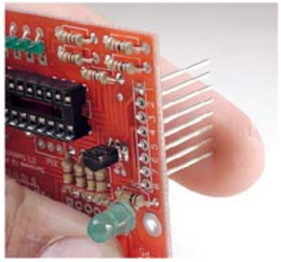
Install all pin headers on UNDERSIDE of Brainboard!

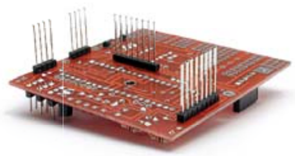

Step 14: 4-Pin Connectors

Step 13: 8-Pin Connectors

Step 13 & 14 - Pin Installation, continued: Remember, install these pins on the underside of the PCB, soldering only one pin per strip initially. This lets you eyeball and adjust them so they're straight up-and-down, which is important so they can mate with the sockets on the Sumovore.


Pin strip installed crooked - good thing you soldered only one pin, so you can remelt the connection and make it stand straight up!

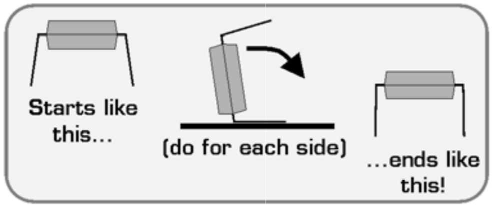

Remelt solder on top pin, and re-adjust pins so they sit properly, then solder the rest of the pins


Finished pin installation, all nice and neat!

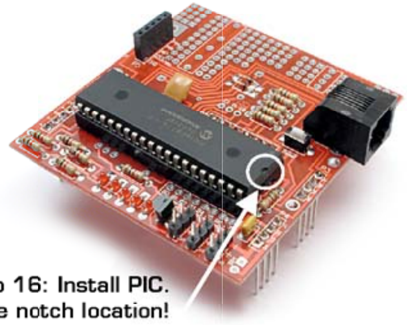
Step 15 & 16 - 16F877A Chip Preparation and Installation: Ever notice how most IC chips have their legs splayed out a bit? This is for machine-installation, where the grab the chip by the sides of the legs, squeeze them in a bit, put them into the holes, and release. When the legs relax back out, the hold the chip in place. Unfortunately for us humans, this is not as easy so you'll have to manually bend the legs in enough so you can easily insert the '877 into the socket.

We really, really recommend you do this on a metal surface, while some part of your skin is touching a ground connection. Like near the kitchen sink, while you're resting a bare elbow on the sink edge. This grounds you, draining off any potential static electricity. Even better if you perform the bend on some aluminum foil also touching the metal sink edge.

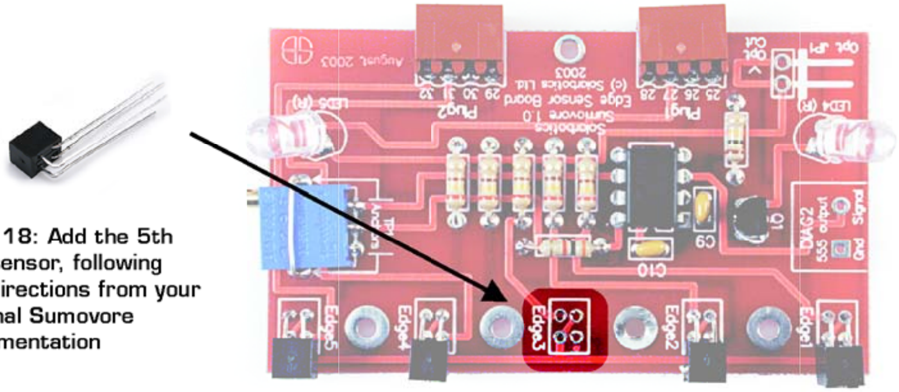
With a firm grip on the chip, lay one side on the surface, and bend the chip over until the legs are sitting at a 90 degree angle to the chip. Repeat for the other side.



Step 15: Bending the Leads inward


Step 16: Install PIC. Note notch location!

Step 18 - Installing the 5th line sensor: Yank the edge-sensor board out of your Sumovore, and install the included line sensor in position 'Edge3', just like you did when you originally built your Sumovore. You don't have to do this, but if you want to make the best use out of your Brainboard, it'd be a good idea!


Step 18: Add the 5th line sensor, following the directions from your original Sumovore documentation

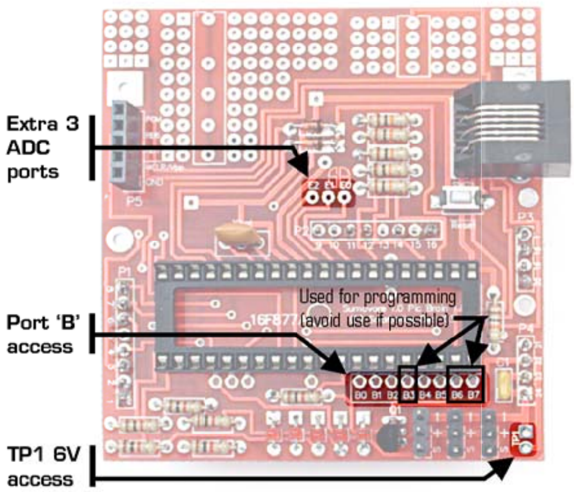
Other Brainboard Points-of-Interest: "Leftovers, get yer leftovers!"

The '877 comes with 8 ADC (analog/digital converters) in the chip's port 'E', of which we're using five for the edge sensors. The leftover three are grouped near the middle of the chip, labled as 'E0', 'E1', and 'E2'.

Practically all of port 'B' has been left untouched, with the exception of the necessary links to 'B3', 'B6' and 'B7' which are used by the programming cable. If you use these three pins, make sure they're for high-impedance usage only. Avoid using them if you can.

If you need raw 6V from the quad AA pack, 'TP1' is your pal. Useful for driving servos, or other high-load circuitry. The three servo headers are driven via this raw 6V through the 'Q1' transistor.

There is also a spare I/O pin available from the PCB underside below the '877 - port 'RA4'. Look, you'll see it...

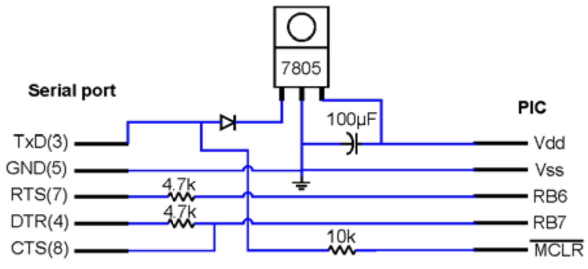


There are a wide variety of ways to program a PIC microcontroller, from expensive powerful commercial units, to bare-bones next-to-no-parts-needed circuits. What we've got for you here is a bit better than the bare-bones approach, and will work on pretty much all computers. It's derived from a programmer designed by David Tait, and will accommodate serial ports that don't quite have enough voltage (like most laptops) to properly program the PIC.

We're showing you both versions of the circuit, so you can try the original or our modified version of the programmer. Of course, if you already own a commercial microcontroller programmer (like a PICStart Plus, ISP-Pro or Warp13a), you're good to go!

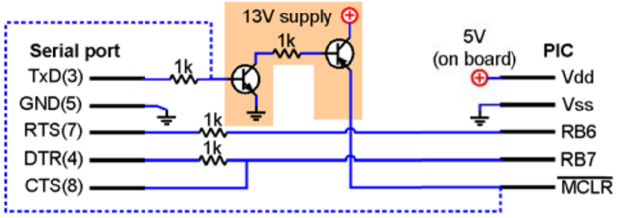
The RJ11-6 jack on this brainboard is configured to work with the ISP-Pro programmer, which is one of the PIC programmers we use in-house at Solarbotics.

Original COM84 By David Tait



Modified COM84

*If your serial port can supply 13V then the you can bypass the transistor voltage buffer, and connect the 1k resistor directly to MCLR



The PIC16F877A Brainboard

Default Program Listing ("sumoline.jal")

This is the default code that your PIC 16F877A ships with. If you mess something up, you can either re-download it from our website, or type it in from what you see here (ug!).

There are a great many languages to pick from to program in, some of which we've linked to on our website. Some are free, some not - it depends on what suits you best.

We've decided to code our default program in JAL ("Just Another Language"), which is a free GPL programming environment developed by Wouter van Ooijen. It's more like PASCAL than C, but it's still pretty easy to work with. Check out the resources page for this brainboard on the Solarbotics website for full documentation.

There are three major sections to the code, being the startup routine, the Sumo routine, and the Line-follower routine. If you start your Sumovore on a black surface (like a sumo ring), the startup routine reads the low inputs from the edge sensors and determines that it should start the sumo routine. When on a white surface (like line-follower usually is, with a black electrical tape line), then the startup routine kicks the Sumovore into running the line-follower routine.

Both of these programs are pretty decent, but there is much more room for creative optimizing. Feel free to modify and hack this code - we' presenting it to you as a good starting point.

```
-- name    : SumoLine.jal
-- author  : GrantM.
-- date    : 12-Jan-2004
-- purpose : Minisumo Linefollower code for the PIC 16F877 Solarbotics
-- Sumovore.

-- target configuration: 16f877 with 20 Mhz Xtal
include 16f877_20

-- standard library
include jlib

-- configure pins
pin_c1_direction = output -- Enable Right motor
pin_c2_direction = output -- Enable Left
pin_c3_direction = output -- Right Motor Direction
pin_c4_direction = output -- Left Motor Direction

pin_d0_direction = output -- LED 1
pin_d1_direction = output -- LED 2
pin_d2_direction = output -- LED 3
pin_d3_direction = output -- LED 4
pin_d4_direction = output -- LED 5

pin_d5_direction = input -- Left IR
pin_d6_direction = input -- Right IR

pin_c3 = low -- Right motor forward
pin_c4 = low -- Left motor forward
pin_c1 = low -- Enable Right motor
pin_c2 = low -- Enable Left motor

-- pin_a0_direction = input  -- Edge Sensor
-- pin_a1_direction = input  -- Edge Sensor
-- pin_a2_direction = input  -- Edge Sensor
-- pin_a3_direction = input  -- Edge Sensor
-- pin_a4_direction = input  -- Edge Sensor

-- Variables
var byte analog, channel, left, mleft, middle, mright, right, sumo

-- "Sub-routine" to read an analog input
function acquire (byte in channel) return byte is
      Bank_1 -- Bank switching
      f877_adcon1 = 0x02 -- LEFT justify, 5 analog inputs
      Bank_0 -- Bank switching
      f877_adcon0 = channel -- Channel select
      delay_1uS( 20 ) -- Delay to allow channel switch
      adcon0_go = high -- Start Conversion
      while adcon0_go loop end loop -- wait for A/D completion
      analog = f877_adresh -- Store the resulting number
      return analog
      adcon0_go = low -- Stop the ADC
end function

 -- Setup PWM

 -- f877_ccpr1l = 75 -- Set PWM max 255 Right motor
  -- f877_ccpr2l = 100 -- Set PWM max 255 Left motor
  -- set PWM mode on CCP1
 -- f877_ccp1con = f877_ccp1con | 0b_0000_1100
 -- set PWM mode on CCP2
 -- f877_ccp2con = f877_ccp2con | 0b_0000_1100
 -- set PWM frequency to ~1kHz
 -- f877_pr2 = 0xFF
 -- enable tmr1
 -- f877_t1con = f877_t1con | 0b_000_0001
 -- enable tmr2
 -- f877_t2con = f877_t2con | 0b_0000_0100
 -- load tmr2 prescaler if a lower frequency is needed
 -- T2CON   = ( T2CON & 0b_1111_1000 ) | 0

-- --------------------------- Look at edge sensors ------------------------
acquire (0x81) -- Read channel 0 (1000 0001)
left = analog  -- Store result in "left"
acquire (0xA1) -- Read channel 4 (1010 0001)
right = analog -- Store result in "right"

if right > 128 then -- Switch between Minisumo/Linefollower
sumo = 1
else
sumo = 0
end if

if sumo == 1 then
-- Sumo 5 Second Startup --
pn_d0 = low
pn_d1 = low
pn_d2 = low
```

The PIC16F877A Brainboard

Default Program Listing ("sumoline.c") cont'd

```
pin_d3 = low
pin_d4 = low

pin_d0 = high
delay_1s
pin_d1 = high
delay_1s
pin_d2 = high
delay_1s
pin_d3 = high
delay_1s
pin_d4 = high
delay_1s

pin_d0 = low
pin_d1 = low
pin_d2 = low
pin_d3 = low
pin_d4 = low
-- End of Startup --

-- Beginning of Minisumo loop
forever loop

acquire (0x81) -- Read channel 0 (1000 0001)
left = analog
acquire (0xA1) -- Read channel 4 (1010 0001)
right = analog

pin_c3 = high -- Right motor forward
pin_c4 = high -- Left motor forward
pin_c1 = high -- Enable Right motor
pin_c2 = high -- Enable Left motor

if left < 128 then
pin_c4 = low  -- Right motor reverse
pin_c3 = high -- Left motor forward
pin_c2 = high -- Enable Right motor
pin_c1 = low  -- Disable Left motor
pin_d0 = high -- Left LED (1)
delay_500ms -- Reverse Time left
delay_200ms
end if

if right < 128 then
pin_c4 = high -- Right motor forward
pin_c3 = low -- Left motor reverse
pin_c2 = low -- Disable Right motor
pin_c1 = high -- Enable Left motor
pin_d4 = high -- Right LED (5)
delay_500ms -- Reverse Time right
delay_200ms
end if

if ! pin_d5 then
pin_c3 = high -- Right motor forward
pin_c4 = high -- Left motor forward
pin_c1 = low -- Enable Right motor
pin_c2 = high -- Enable Left motor
end if

if ! pin_d6 then
pin_c3 = high -- Right motor forward
pin_c4 = high -- Left motor forward
pin_c1 = high -- Enable Right motor
pin_c2 = low -- Enable Left motor
end if

pin_d0 = low
pin_d4 = low

end loop   -- The End.
end if
-- -------------------------- End of Sumo code ---------------------------

-- ---------------------------- Linefollower ------------------------------
if sumo == 0 then

forever loop

acquire (0x81) -- Read channel 0 (1000 0001)
left = analog
acquire (0x89) -- Read channel 1 (1000 1001)
mleft = analog
acquire (0x91) -- Read channel 2 (1001 0001)
middle = analog
acquire (0x99) -- Read channel 3 (1001 1001)
mright = analog
acquire (0xA1) -- Read channel 4 (1010 0001)
right = analog

if left < 128 then -- See white
pin_c3 = high -- Right motor forward
pin_c4 = high -- Left motor forward
pin_c1 = high -- Enable Right motor
pin_c2 = high -- Enable Left motor
pin_d0 = high -- Left LED (1)
end if

if left > 128 then -- See black
pin_c3 = high -- Right motor forward
pin_c4 = low -- Left motor backwards
pin_c1 = high -- Enable Right motor
pin_c2 = high -- Enable Left motor
pin_d0 = low -- Left LED (1)
delay_1ms (75)
end if

if mleft < 128 then -- See white
pin_c3 = high -- Right motor forward
pin_c4 = high -- Left motor forward
pin_c1 = high -- Enable Right motor
pin_c2 = high -- Enable Left motor
pin_d1 = high -- MLeft LED (2)
end if

if mleft > 128 then -- See black
pin_c3 = high -- Right motor forward
pin_c4 = low -- Left motor forward
pin_c1 = high -- Disable Right motor
pin_c2 = high -- Enable Left motor
pin_d1 = low -- MLeft LED (2)
delay_10ms
end if

if middle < 128 then -- See White
pin_c3 = high -- Right motor forward
pin_c4 = high -- Left motor forward
pin_c1 = high -- Enable Right motor
pin_c2 = high -- Enable Left motor
pin_d2 = high -- Center LED (3)
end if

if middle > 128 then -- See black
pin_c3 = high -- Right motor forward
pin_c4 = high -- Left motor forward
pin_c1 = high -- Enable Right motor
pin_c2 = high -- Enable Left motor
pin_d2 = low -- Center LED (3)
end if

if mright < 128 then -- See white
pin_c3 = high -- Right motor forward
pin_c4 = high -- Left motor forward
pin_c1 = high -- Enable Right motor
pin_c2 = high -- Enable Left motor
pin_d3 = high -- MRight LED (4)
end if

if mright > 128 then -- See black
pin_c3 = high -- Right motor forward
pin_c4 = high -- Left motor forward
pin_c1 = low -- Enable Right motor
pin_c2 = high -- Enable Left motor
pin_d3 = low -- MRight LED (4)
delay_10ms
end if

if right < 128 then -- See white
pin_c3 = high -- Right motor forward
pin_c4 = high -- Left motor forward
pin_c1 = high -- Enable Right motor
pin_c2 = high -- Enable Left motor
pin_d4 = high -- Right LED (5)
end if

if right > 128 then -- See black
pin_c3 = low -- Right motor backward
pin_c4 = high -- Left motor forward
pin_c1 = high -- Enable Right motor
pin_c2 = high -- Enable Left motor
pin_d4 = low -- Right LED (5)
delay_1ms (75)
end if

end loop -- End of Linefollower loop
end if
```

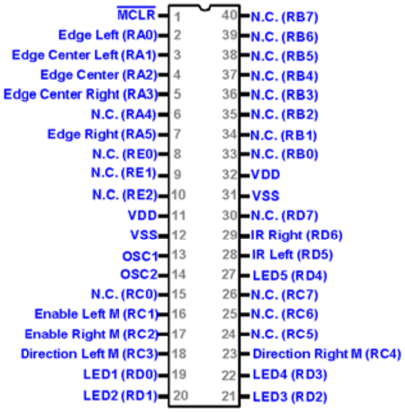
The PIC16F877A Brainboard

Brainboard Schematics

For those of you wanting to do more customizing to your PIC 16F877a Brainboard, here are the microcontroller pin assignments and the PCB schematic.

With all the extra I/O available to you with this microcontroller, you should have no problem in adding in a multitude of modifications to make your PIC-powered Sumovore even more competitive!

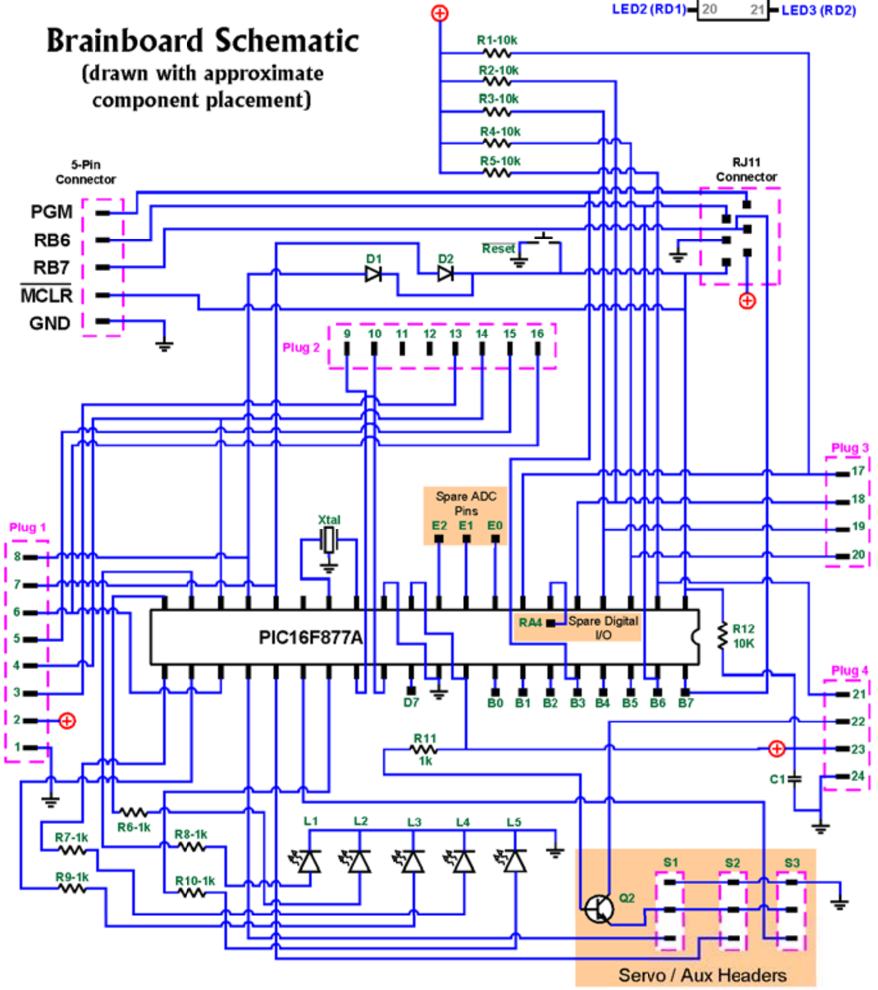
PIC 16F877a Microcontroller Pin Usage



$\overline{\text{MCLR}}$	1	40	N.C. (RB7)
Edge Left (RA0)	2	39	N.C. (RB6)
Edge Center Left (RA1)	3	38	N.C. (RB5)
Edge Center (RA2)	4	37	N.C. (RB4)
Edge Center Right (RA3)	5	36	N.C. (RB3)
N.C. (RA4)	6	35	N.C. (RB2)
Edge Right (RA5)	7	34	N.C. (RB1)
N.C. (RE0)	8	33	N.C. (RB0)
N.C. (RE1)	9	32	VDD
N.C. (RE2)	10	31	VSS
VDD	11	30	N.C. (RD7)
VSS	12	29	IR Right (RD6)
OSC1	13	28	IR Left (RD5)
OSC2	14	27	LED5 (RD4)
N.C. (RC0)	15	26	N.C. (RC7)
Enable Left M (RC1)	16	25	N.C. (RC6)
Enable Right M (RC2)	17	24	N.C. (RC5)
Direction Left M (RC3)	18	23	Direction Right M (RC4)
LED1 (RD0)	19	22	LED4 (RD3)
LED2 (RD1)	20	21	LED3 (RD2)

Brainboard Schematic

(drawn with approximate component placement)



If you have any questions or comments regarding this kit, please contact us!

Solarbotics Ltd.

Toll Free: 866-276-2687 / 403-232-6268 Fax: (403) 226-3741
Website: http://www.solarbotics.com
Email: info@solarbotics.com