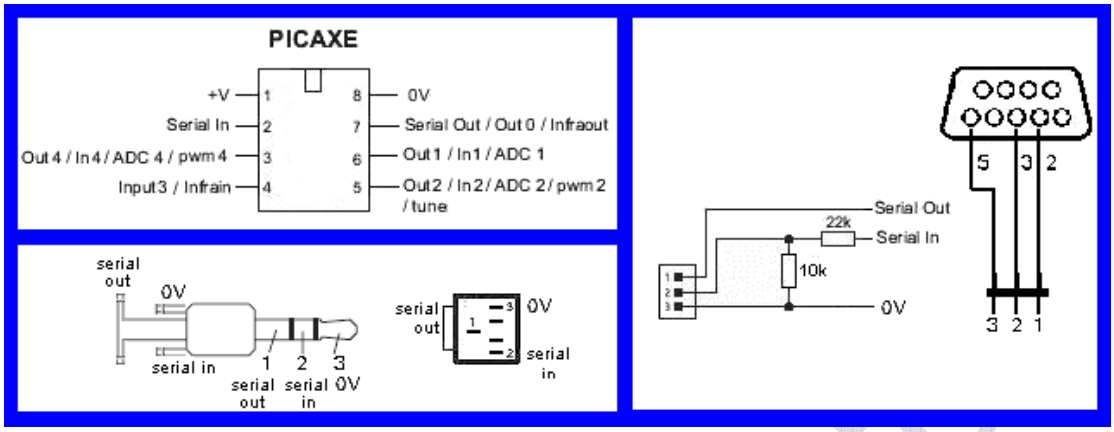
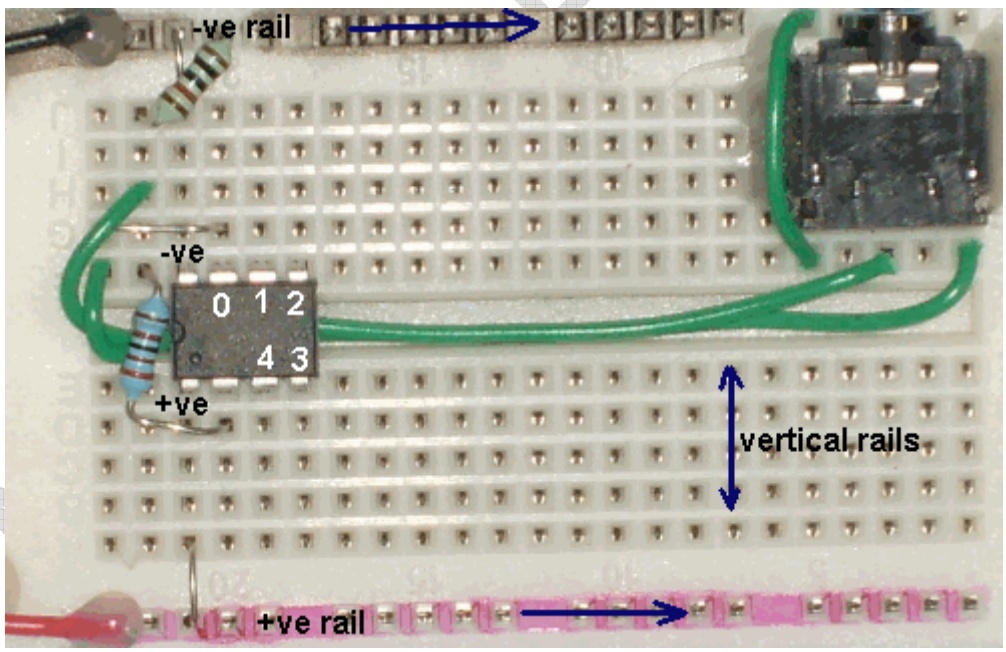
PICAXE I/O ROUTINES AND FUNCTIONS

Pin configuration for PICAXE-08M chip:



Two programming lead connectors are shown:

Left: Standard programming lead with 3.5mm stereo plug
Right: Cable made from old mouse cable connected to DB9 female connector and a 3 pin header strip.



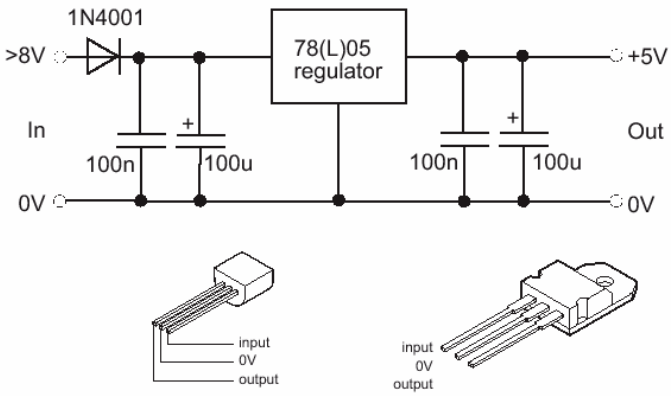
Note the dimple on the left of the chip indicates its orientation.
-ve pin (aka 0v) in the photo is leg 8 in the schematic diagram above.
Pin 0, 1, 2 in the photo are leg 7, 6 and 5 in the schematic diagram.

DC Power Supply: Never use a supply higher than 5V!

Your kit comes supplied with a 3 x AA switched battery box.
Use this until you become more familiar with electronics and PICAXE chips.

Regulated Power Supply:

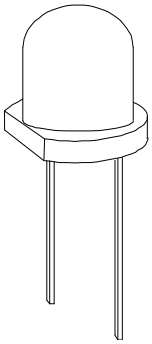
Some users may wish to use a 'wall adapter' style power supply (e.g. part PWR009). It is essential that a good quality regulated 9V DC device is used with a 5V regulator. Unregulated devices may give excessive voltages (under low load conditions) that will damage the microcontroller.



The 9V DC supply must be regulated to 5V using a voltage regulator (e.g. 7805 (1A capability) or 78L05 (100mA capability)). The full regulation circuit is shown below. The 1N4001 diode provides reverse connection protection, and the capacitors help stabilise the 5V supply. **Note** that voltage regulators do not generally function correctly unless the input supply in this circuit is approximately **8V or higher**.

Care with LED's

LED is short for **L**ight **E**mitting **D**iode. They glow when connected to a circuit. LEDs require a minimum voltage, about 1.6-2.0 Volts before they begin to glow.

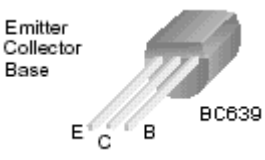


LEDs must be connected the right way around. They have a **positive** and **negative** leg. The short leg with the flat side on the LED always connects to the NEGATIVE side of the circuit closest to the NEGATIVE power supply rail.
The longer leg connects to the side towards the POSITIVE power supply rail (usually to the PICAXE pin).

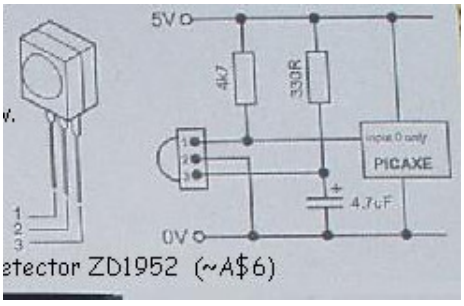
CAUTION: Do not touch the legs of the LED to a 9V battery as this will blow the LED and you will have to buy another!

Common pin configurations

Common Transistor layout:



IR Decoder layout:



Example programs

1. Single LED flasher - connect a LED to pin 2 and 0V (-ve)

```
Main:           ' main program starts here
    high 2      ' connect pin 2 to high voltage state to turn on the LED
    wait 1      ' wait for 1 second
    low 2       ' connect pin 2 to low voltage state to turn off LED
    pause 1000  ' pause for 1000 milliseconds=1s
    goto main   ' go back to start point "main"
```

Extras for experts

Can you …..

- Make the LED flash from any of the output pins 0, 1, 4
- Make the LED stay on for longer
- Make the LED stay off for longer
- Make more than one LED flash

2. Flash together - connect a LED to pin 2 and 0V (-ve), and another to pin 1

```
Main:           ' main program starts here
    high 2      ' connect pin 2 to high voltage state to turn on the LED
    low 1       ' connect pin 2 to low voltage state to turn off LED
    pause 500   ' pause for 500 milliseconds= 0.5s
    low 2       ' connect pin 2 to low voltage state to turn off LED
    high 1      ' connect pin 1  to high voltage state to turn off LED
    pause 500   ' pause for 500 milliseconds= 0.5s
    goto main   ' go back to start point "main"
```

Extras for experts

Can you …..

- Make the LEDs flash together flash
- Make the LED flash at a faster rate
- Make the LED stay off for longer
- Make more than one LED flash
- Use the TOGGLE command instead or HIGH AND LOW?

3. Counting flashes - connect a LED to pin 2 and 0V (-ve)

```
Main:               ' main program starts here
    For b1 = 1 to 10' Saves the value 1 into memory space b1
    high 2          ' connect pin 2 to high voltage state to turn on the LED
    pause 500       ' pause for 500/1000 second = 0.5s
    low 2           ' connect pin 2 to low voltage state to turn off LED
    pause 500       ' pause for 500 /1000 seconds=0.5s
    next b1         ' increases the value in b1 by 1 each loop until value 10
```

Extras for experts

Can you …..

- Increase the number of LED flashers
- Attach a speaker and get it to play a tune (see below) after 8 flashes of the LED

4. Light alert

```
Main:                       'main program starts here
     readadc 1,b1           'read voltage on pin 1 and save in memory space b1
     If b1>150 then Music   ' jump to 'Noise' if voltage number higher than 150
     wait 2                 ' wait for 2 seconds
     goto main              ' jump back to 'Main'

Music:                      'play James,Bond theme -  cut and paste  this code!
tune 0,  2, ($20, $62, $62, $22, $E2, $20, $20, $20, $20, $63, $63, $E3, $22, $22, $22, $20, $62,
$62,$22,$E2,$20,$20,$20,$20,$63,$63,$23,$E3,$22,$21,$20,$00,$AB,$27,$25,$A7)
goto main                   'jump back to 'Main'
```

Extras for experts
Can you …..

- Insert line debug after the readadc line and check the input voltage number on pin 1. Shade the LDR to alter its resistance and see the voltage numbers change.
- Alter the threshold for the speaker to sound to make it trigger more easily.

SENSOR CIRCUITS

1. Temperature sensor

10k thermistor terminated between +ve and pin 1
10k pull down resistor between pin1 and -ve

```
Main:                                ' main program starts here
     readadc 1,b0                    ' read temperature sensor on pin 1
     serout 0, n2400, ("temp:", #b0) 'send txt message along programming lead
     pause 1000                      ' pause for 1000 seconds=1s
     goto Main                       ' go back to start  and repeat this loop forever
```

Extras for experts
Can you …..

- Keep the programming editor running and press F8 to see the data from the sensor
- Set the baud rate to match the serout code above, eg select the n2400 radio button

2. Light intensity sensor

LDR terminated between +ve and pin 1
10k pull down resistor between pin1 and -ve

```
Main:                                ' main program starts here
     readadc 1,b0                    ' read temperature sensor on pin 1
     serout 0, n2400, ("light:", #b0) 'send txt message along programming lead
     pause 1000                      ' pause for 1000 seconds=1s
     goto Main                       ' go back to start  and repeat this loop forever
```

Extras for experts
Can you …..

- Keep the programming editor running and press F8 to see the data from the sensor
- Set the baud rate to match the serout code above, eg select the n2400 radio button

3. Infra-red light sensor

Infra red terminated between +ve and pin 1
10k pull down resistor between pin1 and -ve

```
Main:                                   ' main program starts here
      readadc 1,b0                      ' read temperature sensor on pin 1
      serout 0, n2400, ("IR light:", #b0)       'send txt message along programming lead
      pause 1000                        ' pause for 1000 seconds=1s
      goto Main                         ' go back to start  and repeat this loop forever
```

Extras for experts. Can you …..

- Keep the programming editor running and press F8 to see the data from the sensor
- Set the baud rate to match the serout code above, eg select the n2400 radio button
- Make a **Terminator** target practice game like the one on the Nexus Research Group website!

4. Touch sensor

A piece of hook up wire in +ve rail. Another piece of hook up wire in pin 1 rail.
4.7M pull down resistor between pin1 and -ve

```
Main:                                   ' main program starts here
      readadc 1,b0                      ' read temperature sensor on pin 1
      serout 0, n2400, ("touch:", #b0)  'send txt message along programming lead
      pause 1000                        ' pause for 1000 seconds=1s
      goto Main                         ' go back to start  and repeat this loop forever
```

Extras for experts. Can you …..

- Keep the programming editor running and press F8 to see the data from the sensor
- Set the baud rate to match the serout code above, eg select the n2400 radio button
- Can you have this as an alarm instead? Use as a standalone unit that will flash a LED or play a tune if the wires are touched?
- How about a moisture sensor? Will a dry sponge work as opposed to a damp sponge?
- Or a conductivity sensor to see what materials can conduct electricity?

Night Light / Novelty Jelly

Connect a blue LED to pin 2 and 0V (-ve), and a yellow LED to pin 4. For a birthday treat, place the entire circuit in a jar with a tight sealed lid and place in the bottom of a jelly mold so when the jelly is upright, the LEDS can be seen thru the glass and illuminate the jelly! Or for a spooky Halloween effect, carve a pumpkin and place two red LED's in the circuit for a pumpkin that glows in the dark!

```
Main:
      for b1= 0 to 255 step 2           ' counter loop so LED has multiple PWM cycles
        pwm 2,b1,1                      ' PWM pin 2 LED one cycle increasing pulse width
        pwm 4,b1,1                      ' PWM pin 4 LED one cycle increasing pulse width
      next b1                           ' effect is a pleasing surging brightness increase

      for b1= 255 to 0 step -2
        pwm 2,b1,1                      ' PWM pin 2 LED one cycle decreasing pulse width
        pwm 4,b1,1                      ' PWM pin 4 LED one cycle  decreasing pulse width
      next b1

      goto main
```

The PICAXE-08 Commands

DIGITAL OUTPUT

HIGH	Switch an output pin high (on). ***High 2*** turns pin 2 on
LOW	Switch an output pin low (off). ***Low 2*** turns pin 2 off
TOGGLE	Toggle the state of an output pin.
OUTPUT	Set a pin as output. ***Output 1*** makes pin 1 an output pin
INPUT	Set a pin as input. ***Input 1*** makes pin 1 an input pin
REVERSE	Reverse the input/output state of a pin.
PULSOUT	Output a pulse on a pin for given time.

DIGITAL INPUT

IF… THEN	Jump to new program line depending on input condition. ***If b1 < b2 then motoroff***
PULSIN	Measure the length of a pulse on an input pin.

ANALOGUE OUTPUT

SOUND Output a sound. 0 = no sound, 255 = hiss. ***Sound 2, (100, 20)***
sound pin 2, 5kHz tone, 0.2 s

PLAY Play an internal tune on the PICAXE output pin 2. ***PLAY tune,LED***
- Tune is a variable/constant (0 - 3) which specifies which tune to play
  - 0 - Happy Birthday (All parts)
  - 1 - Jingle Bells (08M/14M only)
  - 2 - Silent Night (08M only)
  - 3 - Rudolph the Red Nosed Reindeer (08M only)
- LED is a variable/constant (0 -3) which specifies if other PICAXE-08M outputs flash at the same time as the tune is being played.
  - 0 - No outputs
  - 1 - Output 0 flashes on and off
  - 2 - Output 4 flashes on and off
  - 3 - Output 0 and 4 flash alternately

TUNE Play a tune on the PICAXE output pin 2.. ***TUNE LED, speed, (note, note, note...)***
- LED is a variable/constant (0 -3) as above for
- speed is a variable/constant (1-15) which specifies the tempo of the tune.
- notes are the actual tune data generated by the Tune Wizard.

PWM Provide a **p**ulse **w**idth **m**odulation output pulse. ***PWM 1, 50, 10***
pulse pin 1, 50/255 duty, 10 cycles

ANALOGUE INPUT

READADC Read analogue channel into a variable. ***Read 1, b0*** ' read analogue pin 1 into b0

PROGRAM FLOW

FOR.. NEXT	Establish a FOR-NEXT loop ***For b1=0 to 100 step 10*** 'Counts in tens
BRANCH	Jump to address specified by offset
GOTO	Jump to address ***If b1=5 goto motoroff***
GOSUB	Jump to subroutine at address.
RETURN	Return from subroutine
IF.. THEN	Compare and conditionally jump .

VARIABLE MANIPULATION

{LET}	Perform variable mathematics.
LOOKUP	Lookup data specified by offset and store in variable.
LOOKDOWN	Find target's match number (0-N) and store in variable
RANDOM	Generate a pseudo-random number

SERIAL I/O

SEROUT Output serial data from output pin. ***Serout 0, n2400, (65)***
pin 0, 2400 bps, sends ASCII 65 (=A)

SERIN Serial input data on input pin. ***Serin 0, n2400, ("A")*** pin 0, 2400 bps, waits for ASCII 65

INTERNAL EEPROM ACCESS

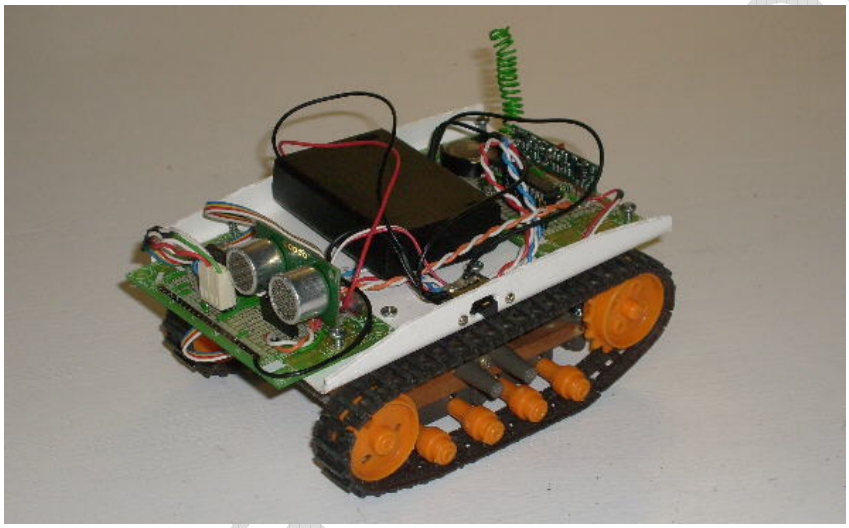
EEPROM	Store data in data EEPROM before downloading BASIC program
READ	Read data EEPROM location into variable
WRITE	Write variable into data EEPROM location ***Write 220, b1*** store byte b1 into address 220

POWER DOWN

NAP	Enter low power mode for short period (up to 2.3 sec)
SLEEP	Enter low power mode for period (up to 65535 sec)
END	Power down until reset, an indefinate sleep

MISCELLANEOUS

PAUSE	Wait for up to 65535 milliseconds
WAIT	Wait for up to 65 seconds
DEBUG	Outputs variable value to PC screen via programming lead ***Debug b0*** shows b0 value on screen



CASI, the PICAXE autonomous Mars rover prototype by Michael Fenton that can also be remotely controlled by a Casio Graphics calculator

Parts List:

Breadboard
PICAXE 08M
3 x AA switched battery box
UV Led (silver tape)
Infra-red photo transistor (black tape)
10k Thermistor temperature sensor
2 each of yellow, red, blue, white ultrabright LED's
Speaker
10k resistors
4.7M resistor
Hook up wire
CD-ROM with Mac OSX and Windows software
Serial cable
3.5mm stereo socket
Instruction manual, circuit diagrams, example codes