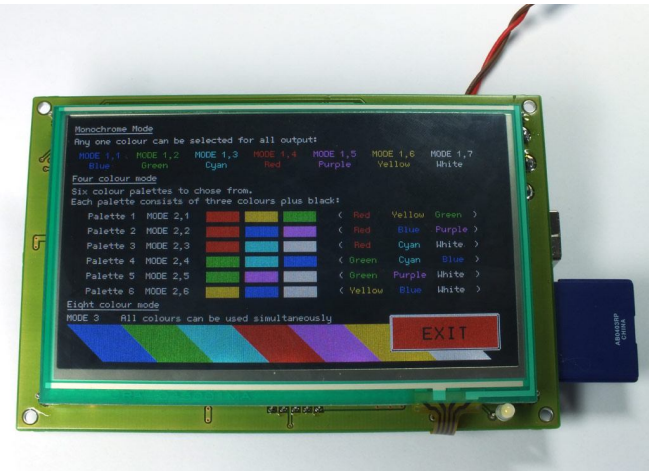
TFT Maximite Manual

Updated to V4.5 of MMBasic and Rev 1.3 and 1.4 of the PCB

Geoff Graham

For updates to this manual and more details on the Maximite
go to http://geoffg.net/tft-maximite.html

TFT Maximite



The TFT Maximite is a version of the popular Colour Maximite created by Carsten Meyer (cm@ct.de) for c't Hardware Hacks magazine.

It includes a touch sensitive 4.3 inch LCD panel that provides a sharp 480 x 272 pixel display which is fully supported by MMBasic. This includes the ability to display and respond to touch sensitive controls (buttons, switches, etc) under control of your MMBasic program.

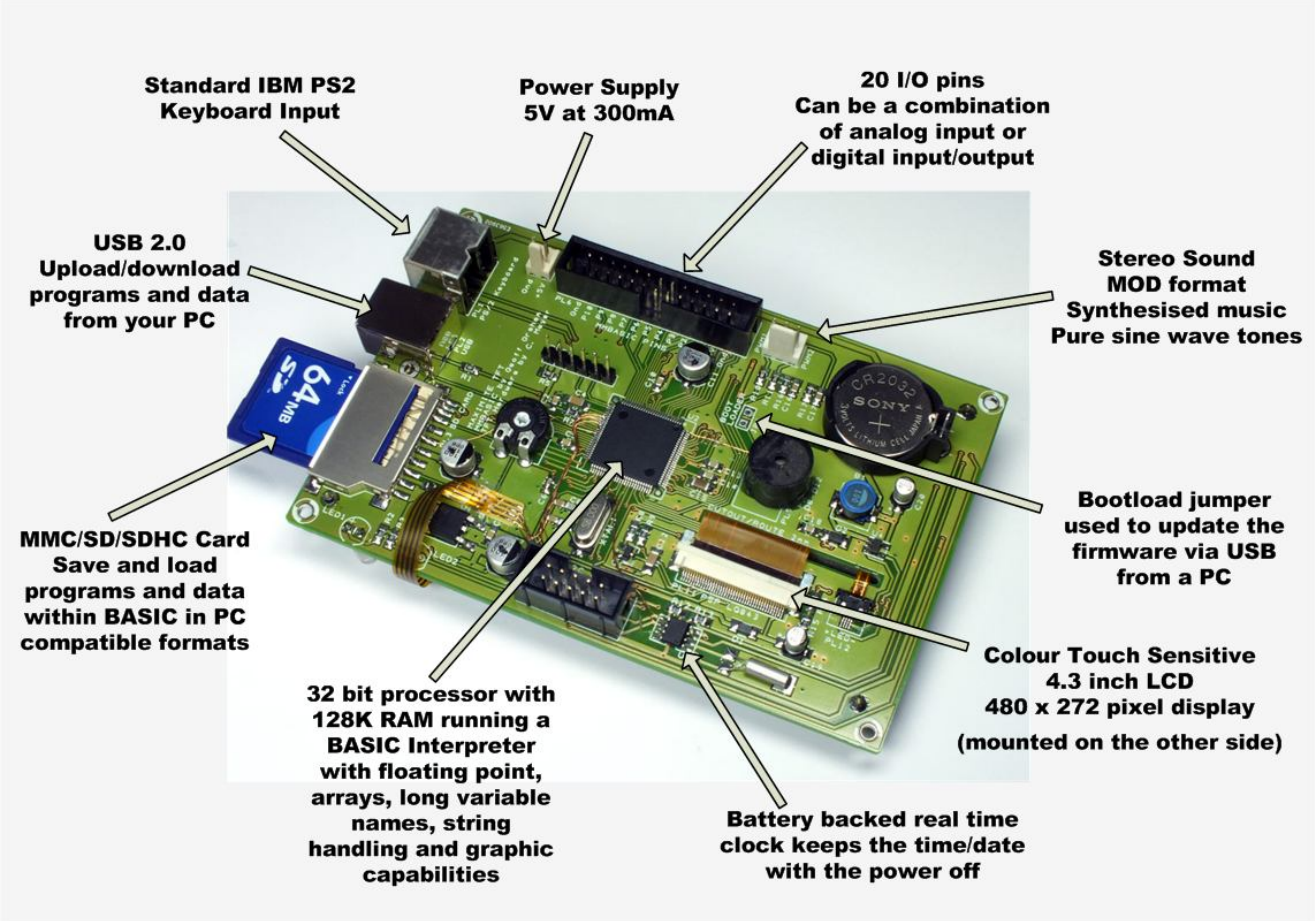
With the exception of the Arduino connector it has all the features of the Colour Maximite including colour, stereo music synthesiser, input from a standard PC compatible keyboard or USB, built in SD memory card, BASIC language and 34 input/output pins.

Refer to the MMBasic Language Manual for details of how to use MMBasic and the general features of the Maximite. This manual can be downloaded from http://geoffg.net/maximite.html.

For new firmware and other updates go to http://geoffg.net/tft-maximite.html

For schematics and hardware details of the TFT Maximite go to https://github.com/heise/MAXIMITE

Features and Connectors



TFT Maximite Technical Details

LCD Display

4.3 inch, 480 x 272 pixel, touch sensitive TFT LCD panel. MMBasic supports eight colours and 80 characters per line by 22 lines per screen using the standard font

USB

Implements the CDC (Communication Device Class) protocol over USB 2.0. This is a serial interface to the BASIC interpreter so, by using a terminal emulator on the host, programs can be entered, edited and run. Using this interface you can upload programs by streaming the text with a suitable terminal editor.

The Windows driver is available from http://geoffg.net/maximite.html. There is native support for the CDC protocol in Linux (the cdc-acm driver) and Apple OS/X.

Keyboard

Standard IBM compatible PS2 keyboard with mini-DIN connector or a USB/mini-DIN adapter.

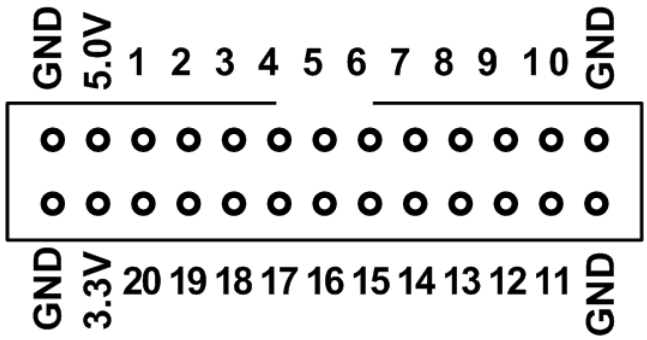
Non ASCII keys (such as the function keys) are mapped to special characters. See Appendix E of the MMBasic Language Manual for the details.

SD/MMC Card Interface

Will accept MMC, SD or SDHC memory cards formatted as FAT16 or FAT32 up to 32GB in capacity. Note that there is no advantage in using a fast SD card as the card is clocked at a fixed 20MHz, regardless of its speed rating.

I/O Connector

I/O pins are addressed in MMBasic as a number. This diagram lists the connections to this connector along with the I/O pin numbers used in MMBasic.



Note that on the TFT Maximite both I/O pins 12 and 13 have a 4.7KΩ pullup resistor installed on the PCB and this pulls the pin up to 5V. This is convenient for when these pins are used for I^2C communications but the load that they present must be also considered when using the I/O pins for general purpose input/output.

On version 1.3 of the printed circuit board (PCB) pin 20 is disabled due to a design fault. However additional I/O pins D0 to D4 are available on connector PL7 and I/O pins D8 to D13 are available on PL9 (refer to the schematic for specific pin details).

On version 1.4 of the printed circuit board (PCB) the pin 20 issue has been corrected. In addition D4 to D7 are available on PL9. This means that with the combination of PL7 and PL9 there are 14 additional I/O pins available ranging from D0 to D13 (refer to the schematic for specific pin details).

Note that I/O pin D13 is also connected to the on board buzzer so this output can be used to control this device.

Pins D0 to D13 are described in the MMBasic User Manual under the Colour Maximite heading. These designations can be used in MMBasic commands related to I/O or alternatively they can be referred to as pins 21 to 34.

Serial I/O Ports

Serial port COM1: uses pin 15 for receive data (data in) and pin 16 for transmit data (data out). If flow control is specified pin 17 will be used for RTS (receive flow control – it is an output) and pin 18 will be CTS (transmit flow control – it is an input). Serial port COM2: uses pin 19 for receive data (data in) and pin 20 for transmit data (data out).

I²C, SPI and 1-Wire Ports

I²C uses pin 12 as the data line (SDA) and pin 13 the clock (SCL) and both have pullup resistors to 5V (see the notes under I/O Connector). SPI and 1-Wire can use any I/O pin.

Battery Backed Clock

To set the battery backed clock you use the standard commands in MMBasic for setting the time (TIME$ and DATE$). From then on MMBasic will automatically retrieve the current time and date on power up and display it under the Maximite logo– just to let you know that your battery backed clock is working correctly.

Electrical Characteristics

Power Supply

5V nominal (4V to 5.5V) at 270mA typical (plus current draw from the I/O pins)

Digital Inputs

Logic Low:	0 to 0.65V
Logic High:	2.5V to 3.3V (I/O pins 1 to 10) 2.5V to 5.5V (I/O pins 11 to 20)
Input Impedance:	>1MΩ. All digital inputs are Schmitt Trigger buffered.
Frequency Response:	Up to 200KHz (pulse width 10nS or more) on the counting inputs (pins 11 to 14).

Analog Inputs (I/O pins 1 to 10)

Voltage Range:	0 to 3.3V
Accuracy:	Typically ±1%. This accuracy is dependent of the accuracy of the 3.3V supply voltage.
Input Impedance:	>1MΩ (for accurate readings the source impedance should be <10K)

Digital Outputs

Typical current draw or sink ability on any I/O pin:	10mA
Maximum current draw or sink on any I/O pin:	25mA
Maximum current draw or sink for all I/O pins combined:	150mA
Maximum open collector voltage (I/O pins 11 to 20):	5.5V

Audio Output

Audio Frequency Response:	<20Hz to 4KHz
Output Level:	0.5V pp (with components as specified) 0 to 3.3V when operated as a PWM output.

Battery Backed Clock

Time keeping accuracy:	±50 ppm (typical at room temperatures)
Battery Life:	10 to 15 years (limited by the battery shelf life)

Special MMBasic Commands

MMBasic for the TFT Maximite has a number of special commands and functions to support the LCD display and touch interface. Most of these commands are used to create and manage touch sensitive objects on the display. MMBasic will maintain these objects and automatically change their visible status in response to touch inputs on the screen when the program is running (and not waiting for input at the INPUT or LINE INPUT commands).

When an object is created it must be assigned a reference number in the range of zero to 31. This can later be used to refer to the object so that its setting can be retrieved or set.

CONFIG FONT 1 or CONFIG FONT 2	Set the default font to Font #1 which is the standard font of 10 x 5 pixel or Font #2 which is a larger font of 16 x 11 pixels. Whichever font is selected it will become the default on power up and will be reinstated whenever control returns to the input prompt. The power must be cycled for the new setting to take effect.
TOUCH SIZE sx, sy	Will set the size of any objects subsequently created using the CREATE option. 'sx' and 'sy' are the width and height of the object in pixels.
TOUCH CREATE r, x, y	Will create and enable a touch sensitive area on the screen with no corresponding visible image. 'r' is the reference number. 'x' and 'y' are its position in pixels.
TOUCH CREATE r, x, y, $caption, colour , B | P | S | C | R [, D]	Will create and enable an object. 'r' is the reference number. 'x' and 'y' are the position of the object. '$caption' is the text to be applied to the object and 'colour' is its colour. ' B | P | S | R | C' is a single character that will select the type of object to display: • B = a push button. Note that when pressed the button remains down and it is the responsibility of the program to set it back up (TouchVal(r) = 0) after it has detected the button press. • P = a latching push button that toggles between OFF and ON • S = a switch that toggles between OFF and ON. Note that the caption is ignored. • C = a check box that toggles between true (display a tick mark) or false (no mark). • R = a radio button. There can only be one set of radio buttons and when any one is selected all other radio buttons will be turned off. The last parameter (,D) is optional and indicates that the object will not respond to touch.
TOUCH CREATE r, x, y, $caption, colour , H | V [,L | R | T | B | N | D]	Will create and enable a horizontal or vertical slider. 'r' is the reference number. 'x' and 'y' are its position in pixels. '$caption' is the text to be applied to the slider and 'colour' is its colour. The single character 'H' will create a horizontal slider while 'V' will create a vertical slider. The last parameter is optional and can be can be one or more of the following characters (for example, LD): • L will cause the left side of the knob to be filled. • R will cause the right side of the knob to be filled. • T will cause the top side of the knob to be filled. • B will cause the bottom side of the knob to be filled. • N will create the slider with no knob • D means that the slider will be created but it will not respond to touch. A slider such as this can be used as a bar graph to display

	analogue values.
	The position of the slider can be found using :
	`value = TOUCHVAL (r)`
	and it can be set using:
	`TOUCHVAL(r) = value`

TOUCH CREATE r, x, y, $caption, colour ,L [, D]	Will create and enable a graphical object that looks like a LED. 'r' is its reference number. 'x' and 'y' are its position. '$caption' is the text to be applied next to the LED and 'colour' is its colour when illuminated. The last parameter (,D) is optional and indicates that the LED will not respond to touch. The diameter of the LED is set by the Y parameter of the last TOUCH SIZE command. Minimum size is 10 pixels. The LED can be turned on and off using: `TOUCHVAL(r) = value`
TOUCH DISABLE r	Will disable a touch item without removing it from memory. Disabled items are automatically shaded grey (dimmed) and will not respond to touch.
TOUCH ENABLE r	Will enable a touch item that has been previously disabled. This redraws the item at its old coordinates. Newly created items are enabled by default.
TOUCH REMOVE r1 [, r2 [, …]] or TOUCH REMOVE ALL	Will remove one or more objects (reference number 'r1', 'r2', etc) from the screen. The shortcut REMOVE ALL will remove all active objects from the screen. This command clears the item from screen and sets the space occupied by the item to the current background colour
TOUCH WAIT or TOUCH RELEASE	TOUCH WAIT will cause MMBasic to wait for a touch to occur. TOUCH RELEASE will cause MMBasic to wait for a current touch to be removed from the screen. Both will prevent MMBasic from responding to interrupts while the command is waiting for the event.
TOUCH INTERRUPT target	Will setup an interrupt which will call 'target' line number, label or user defined subroutine whenever a new touch occurs. Return from an interrupt is via the IRETURN statement except where a user defined subroutine is used (in that case END SUB or EXIT SUB is used). Note that subroutine parameters cannot be used. To disable this interrupt, use numeric zero for the target, ie: TOUCH INTERRUPT 0
TOUCH CALIBRATE	Will enter the calibration routine for the touch sensitive input. You will be asked to touch two positions on the screen and the resulting calibration data will be stored in the non volatile memory section of the real time clock. Control-C can be used to abort the calibration. Calibration should only be required when the TFT Maximite is first used or if the clock battery was removed.
TOUCH BEEP duration	Make a sound on the beeper that lasts for 'duration' milliseconds.
TOUCHVAL(r) = value	This uses TOUCHVAL as a command to manually set the value of an object with the reference number 'r'. For binary objects value can be zero (for OFF) or non zero (for ON). For a slider it can be a number up to the range of the slider in pixels and it will set the slider at that position.

value = TOUCHVAL (r)	This uses TOUCHVAL as a function to retrieve the status of a touched object. A returned value of zero means off and 1 means selected or on. The value may also represent the slider position (range 0 to the TOUCH SIZE used for the slider). If a touch object was not initialized with TOUCH CREATE, this function will return 0.
value = TOUCHED (r)	This will return 1 if item 'r' was touched since the last TOUCHED() call, otherwise 0. The returned value from this function is reset to zero after it has been used and will only be set to 1 again if the item is touched again.
xcoord = TOUCHED (#X) or ycoord = TOUCHED (#Y)	This will return the current x or y coordinate of the current touch point on the screen. If the screen is not being touched -1 will be returned.
refnum = TOUCHED (#I)	This will return the reference number of the last control (i.e. object reference) that was touched.