MMBasic for a Generic PIC32 Platform

MMBasic is an implementation of the BASIC language with floating point and string variables, long variable names, arrays of floats or strings with multiple dimensions and powerful string handling. It is generally compatible with Microsoft BASIC so it is easy to learn and run.

MMBasic was originally written for the Maximite, a small self contained computer based on the Microchip PIC32 microcontroller. This document describes a minimal version of MMBasic (called PIC32-Generic) which can run on almost any generic PIC32 based board.

The intention is to provide developers with a simple implementation of MMBasic to use as the starting point for porting MMBasic to a new chip or hardware environment. This implementation only requires a UART interface for the user to run and test BASIC programs. The hooks are there to add storage and other functions but they are not required in the first instance.

For more details and downloads related to MMBasic go to the MMBasic Home Page at http://mmbasic.com.

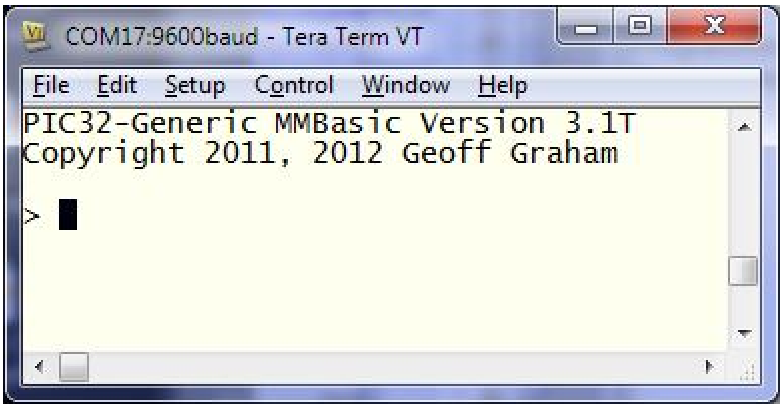
Running the Compiled HEX File

This zip file contains a pre compiled HEX file which can be loaded onto a board with either the 64 pin or 100 pin variants of the PIC32 supporting 512K flash and 128K RAM. Typical boards include the Microchip Explorer-16 board with PIC32MX PIM or the UBW32 development board (from sparkfun.com). It should also run on the PIC32 starter board although this has not been tested.

The basic requirements for running the compiled hex file are:

- A board with the PIC32MX695F512H/L or PIC32MX795F512H/L chip.
- An 8MHz crystal on the primary oscillator
- A computer running a serial emulator connected to UART 3A on the chip. Baud rate should be 9600 with 8 bits one stop bit and no parity or flow control. Receive data on the chip is the pin supporting U3ARX and transmit is U3ATX.

Program the hex file and on resetting the chip the following should appear on the output of UART 3A Tx.


COM17:9600baud - Tera Term VT
File Edit Setup Control Window Help

```
PIC32-Generic MMBasic Version 3.1T
Copyright 2011, 2012 Geoff Graham

> 
```

Note

Do not use this version with the Maximite, DuinoMite, CGMMSTICK, and other Maximite type computers as this will overwrite the boot loader used in these computers. The correct MMBasic for these computers can be found at http://geoffg.net/maximite.html.

Documentation

MMBasic is documented in the MMBasic Language Manual from http://mmbasic.com/downloads.html. This manual covers the more full featured versions of MMBasic for the Maximite or DOS versions but it is applicable to this version with the exception of the following features that are not implemented:

- File I/O and storage related commands.
- Video, graphics, fonts, special keyboard keys and the full screen editor.
- External I/O, sound, interrupts and communications protocols (serial, I2C, SPI and 1-wire).
- Date/time and timing functions/commands and the CONFIG command.

Source Code

The source to this and other versions of MMBasic is available from http://mmbasic.com.

Porting to Other Platforms

Obviously the best way to understand the requirements for porting MMBasic to a new platform is to request the source and look through it and its documentation. These notes are intended to provide a high level view of what is required.

In its minimal version MMBasic compiles to about 94K of flash and requires just over 5K of RAM and a minimum of 4K for the stack (9K total). More memory is required to store programs, variables, etc so the interpreter will require at least 16K of RAM to run a small BASIC program. Generally 32K RAM or more is preferred as that will allow for larger programs and arrays.

Keep in mind that adding extra functionality (SD card support, file I/O, etc) will add to these requirements.

The language itself is hardware independent other than requiring 32 bit integers and an ANSI C compiler. Supporting the language are two main files that will need customising to suit a particular platform:

- Main.c
  This is the main entry point and is responsible for configuring the chip, basic character I/O and the main loop which inputs a line and dispatches it to the interpreter.
- Memory.c
  This allocates RAM for the interpreter's use (program memory, variable memory and the heap). The configuration of this has been kept flexible so that the language can run on chips with unusual memory mapping. MMBasic has its own dynamic heap system which is also implemented in this file.

MMBasic has a simple mechanism for adding functionality that is both elegant and efficient. This allows the easy addition of new commands, functions and operators. For example, let us say that you want to implement a function that returns a special value generated by your hardware.

In one file you would define the C function to perform the operation:

```
void fun_special(void) {
    fret = (float)special_data_to_return;
}
```

And in another file you insert the definition of your function into the function table:

```
{ "Special(",  T_FUN | T_NBR,  0, fun_special },
```

MMBasic will do the rest and in your BASIC program you can use the function like any other:

```
PRINT Special()/100
```