MMBasic pico LA

A logic analyzer for the Raspberry Pi Pico running MMBasic

Introduction

The Raspberry Pi Pico, based on the RP2040 chip, is usually programmed using Python. Geoff Graham and Peter Mather have used this same hardware platform to create a “Boot to Basic” computer running MMBasic. By making optimal use of the RP2040 peripherals, this small computer , called PicoMiteVGA, requires only a few passive components and connectors.

```
PicoMiteVGA MMBasic Version 5.07.04b8
Copyright 2011-2021 Geoff Graham
Copyright 2016-2021 Peter Mather

>
```

The MMBasic software and instructions to build the computer can be downloaded from Geoff Grahams website.
https://geoffg.net/picomitevga.html

The logic analyzer described in this document, is the result of a challenge Peter Mather started when the development of MMBasic, that went on for over a year, started to support the RP2040 PIO and especially DMA to and from the PIO.

To run the Logic Analyzer (from here on called LA), you need MMBasic version 5.07.07 RC4 or newer.

What is a logic analyzer ?

A logic analyzer is a tool that allows you to inspect logic signals. It shows the logic signals as waveforms as a function of time.

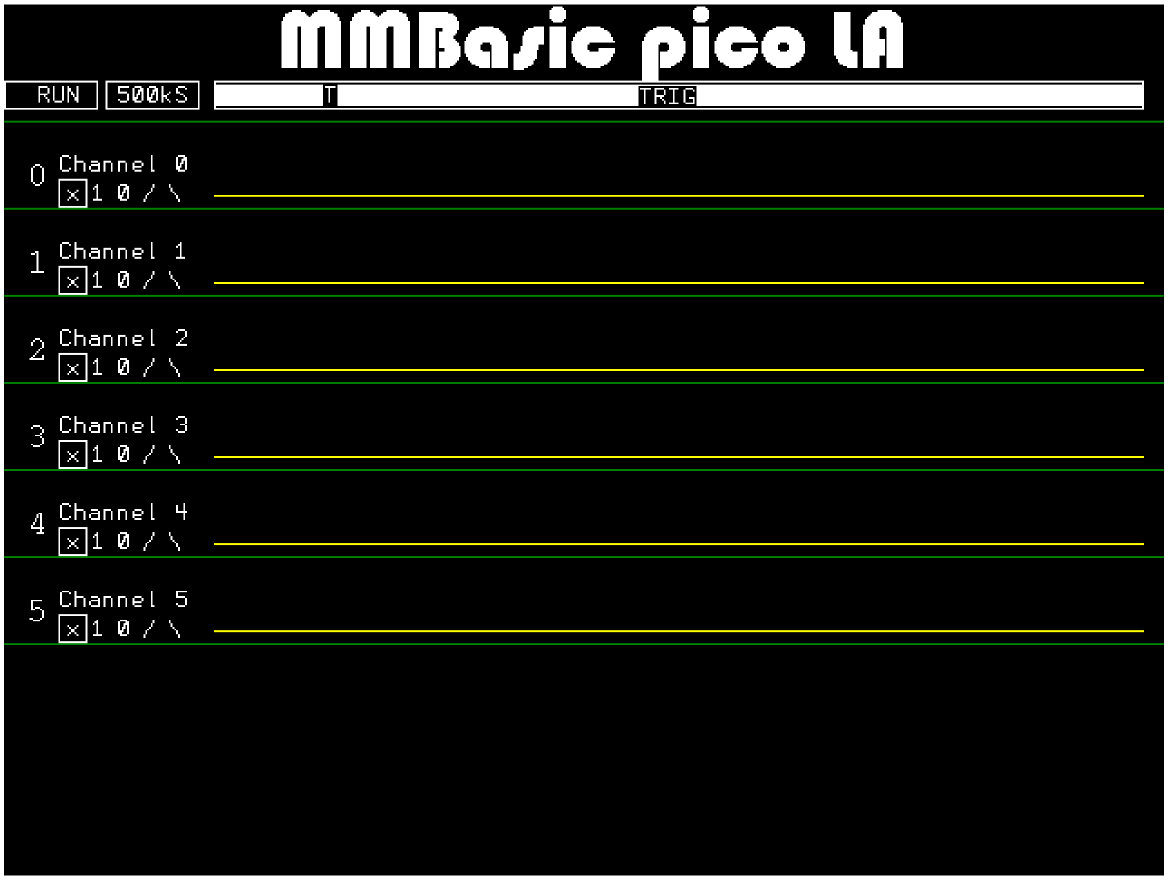
This logic analyzer is not an oscilloscope, in that it shows the analog voltages. It only shows logic high and logic low (for 3.3V signals).

Appart from showing the logic levels, a logic analyzer can provide tools for analysis of the logic signals. It can measure pulse width or can decode serial or parallel busses.

At time of writing the MMBasic Pico LA can decode I2C bus trafic, and it can measure time (pulse width).

MMBasic Pico LA

The PicoMiteVGA stand alone basic computer uses many of the RP Pico board for connecting monitor, keyboard, SD card, and audio output. The remaining (unused) pins GP0…..GP5 are used as inputs for the logic analyzer. Only 6 pins. The Pico LA is a 6 channel logic analyzer.



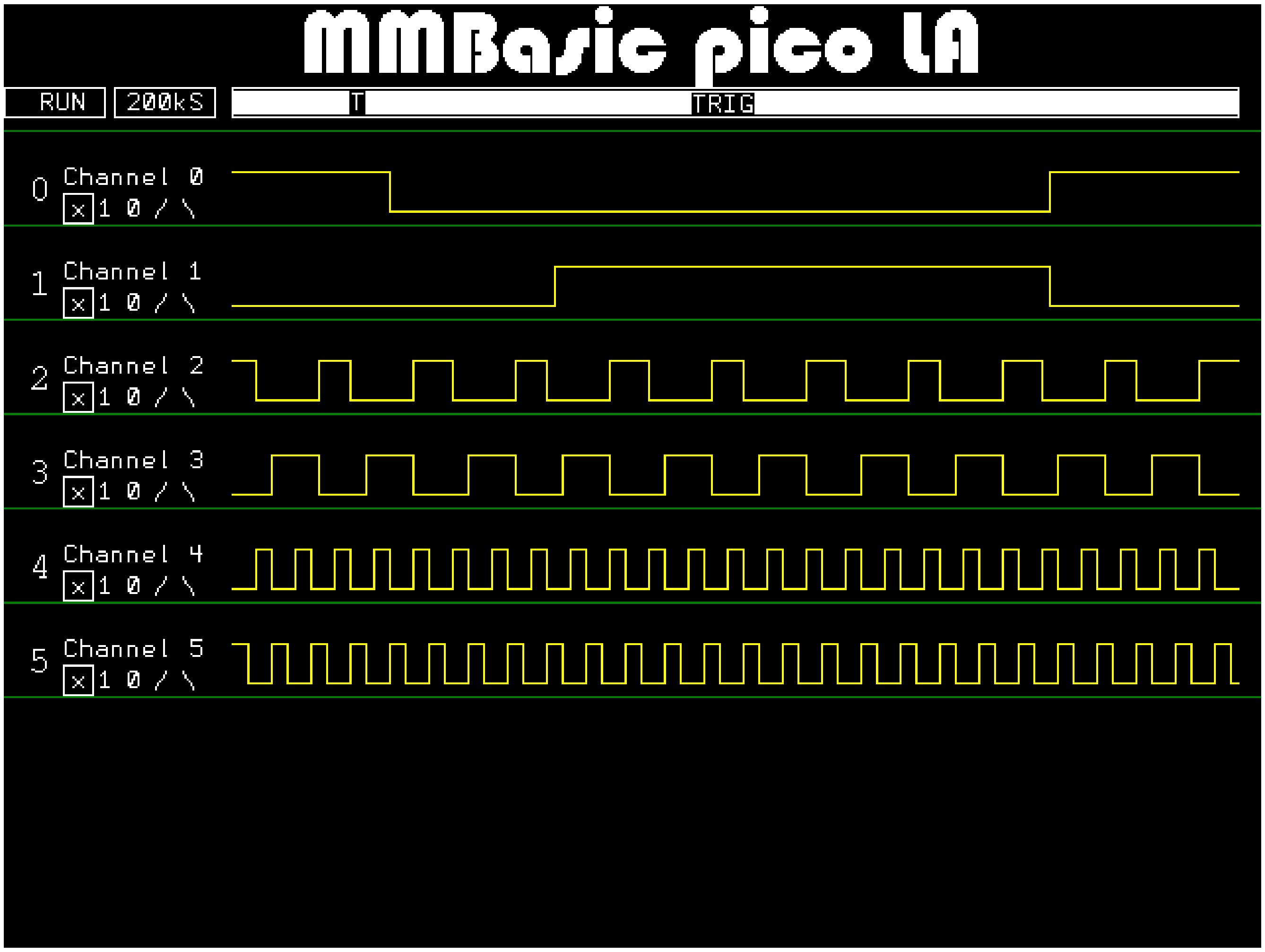
When you install the MMBAsic Pico LA on the PicoMiteVGA, you unzip the files, and place the 2 files (banner.bmp and LA_22.bas) in one folder on either SD card (B: drive), or the internal filesystem (A: drive).

Running the LA_22.bas file shows above screen.

The top half of the screen shows generic analyzer settings. The bottom have shows the signals that are present on the 6 input channels GP0...GP5, and their associated trigger settings.

Default the logic analyzer is free running (showing the actual signals continuously), and in absence of input signals, will shows nothing exciting.

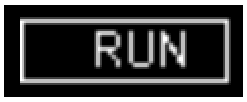
Once you connect signals to the inputs, the waveforms will show on screen. Reading the 6 input signals into the PicoMiteVGA memory is called “acquisition”.



Above is just an example of various input signals to GP0...GP5 of the analyzer. As you will experience, the screen is constantly updating, showing you the live behaviour of the input signals. In case you want to study the signals you can stop the analyzer by using the PicoMiteVGA keybord.

S = Stop the analyzer after a single (new) acquisition (each S starts a new single run)
R = Run the analyzer repeatedly

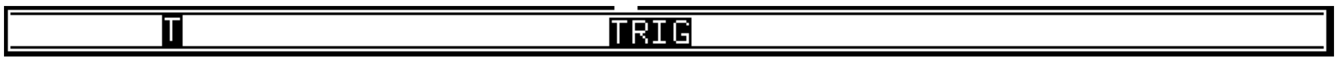
The top left indicator shows the status of the logic analyzer acquisition.



RUN = running continuosly
SNGL = processing a single acquisition (busy ~~or~~ waiting for trigger)
HALT = acquisition stopped

Acquisition and the screen display

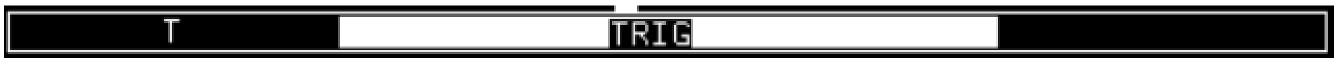
The white bar above the signal waveforms shows the size of the total acquisition memory.



This bar shows that the full acquisition memory is shown on screen (*). To see more detail you have to zoom in on the waveform. Zoom is controlled from the arrow keys on the keyboard.

Arrows:
UP – zoom in into the center of the display waveform
DOWN – zoom out
LEFT – move the zoom window towards the beginning of the acquisition
RIGHT – move the zoom window towards the end of the acquisition

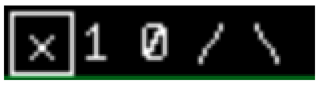
Below is an example of the bar after pressing a single **UP** key. The middle of the acquisition memory is shown. The wave forms will show more detail now.



The T and TRIG text will be explained in the chapter TRIGGERING

Triggering

Default the logic analyzer will show whatever is on the 6 input pins, signals that could change to fast for the human to analyze. Use triggering to stop the logic analyzer at a specific event. Each of the 6 channels can be used as a condition for the event trigger.



Each channel has 5 possible ways to contribute to the trigger condition.

X = ignore this channel for the trigger condition
1 = this channel must be logic “1” for a trigger event
0 = this channel must be logic “0” for a trigger event

/ = a rising edge is a condition for the trigger event
\ = a falling edge is a condition for the trigger event

You change the trigger setting by pressing the channel number key (numeric keys 0...6). Each successive press of the numeric key will advance the trigger choice selector. At the 5'th advance you have the option to change the channel legend (name). Then the trigger selection cycle will repeat.

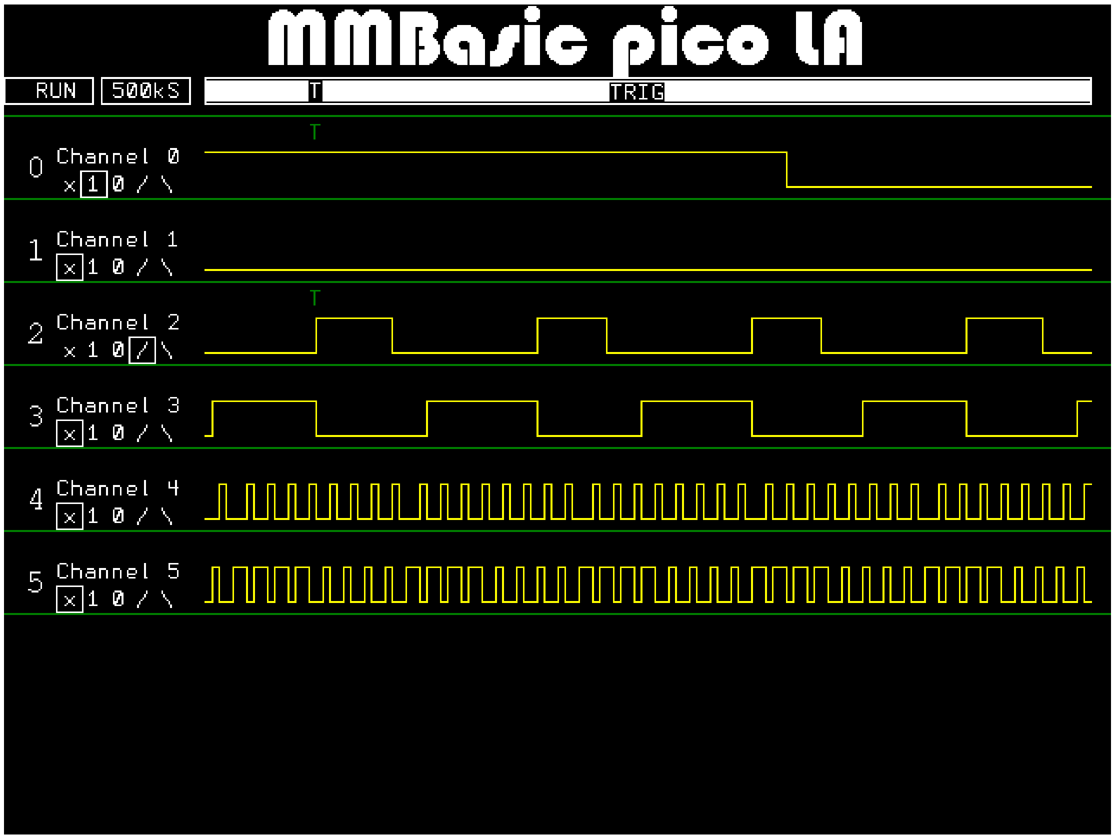
0 = change trigger selection and channel name for GP0
1 = change trigger selection and channel name for GP1
2 = change trigger selection and channel name for GP2
3 = change trigger selection and channel name for GP3
4 = change trigger selection and channel name for GP4
5 = change trigger selection and channel name for GP5

A typical event setup will consist of a single edge condition, and one or more logic level conditions.

Note : only 1 edge trigger condition is allowed (it is highly unlikely that 2 edges occur at exactly the same time).

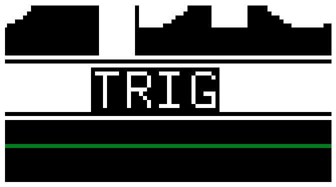
Below is a screen showing rising edge trigger on GP2 under condition that GP0 is "1". The green "T" characters shows the trigger event for each waveform contributing. The white T in the zoom bar shows where the trigger event is in relation to the display screen.

Important: the trigger event can be invisible (outside the display window).



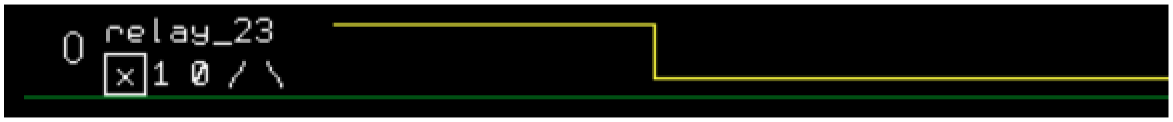
When setting up a trigger condition, you **ARM** the logic analyzer. Once the event has occurred, the logic analyzer is **TRIG**gered.

The trigger status is shown in the centre of the zoom bar.



Changing channel name

The channel name change can make things easier to understand while debugging.



Trigger know-how

Here are some things to consider when configuring a trigger condition:

1/ One should be careful in setting up the trigger conditions. It is quite common, and will happen to many of us, to set up a condition that does not occur. In example : connect the same signal to GP0 and GP1, and set GP0 to trigger on “1”, and GP1 to trigger on “0”. This will simply never happen.

2/ The logic analyzer event trigger will trigger on fast signals. So it is possible that the trigger indicator will indicate the analyzer has triggered, but this is not visible in the displayed waveforms. This is possibly due to ALIASING.

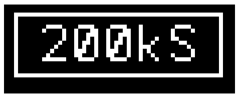
Aliasing

Aliasing is a result of digitizing a signal. The logic analyzer takes samples of the signals at the input pins. The logic analyzer memory will contain the status of the signals at the moment the sample is taken. But the time between the samples, the signals are ignored. All changes between samples are not in the analyzers memory, and therefore cannot be displayed on screen.

Aliasing affects 2 core functions of the logic analyzer

Acquisition

To capture faster changes in the input signals MMBasic Pico LA can change the sampling speed. Sampling speed is indicated in the top half of the screen:



This means that 200.000 x per second a sample is taken, every 5 microseconds. Every change in the input signals faster than 5us may be missed in the acquisition. This setting is nice for UART signals at 9600 baud (104us per bit) but not for I2C (10us per bit) @100kHz, 2.5us per bit at 400kHz).

The sampling speed is controlled with the “**+**” and “**–**“ keys in a 1-2-5 sequence.

+ = sample faster
- = sample slower

Note that the acquisition memory fills faster at a higher sampling speed. You see more detail, but capture less time.

Display

The acquisition memory contains 4096 samples. But the VGA screen (640 pixels wide) cannot show all samples. The software is configured to show maximum 256 samples as a waveform. When zoomed out, 256 of the 4098 samples are shown, and 3840 are not shown. Zooming in may show details that where not visible before.

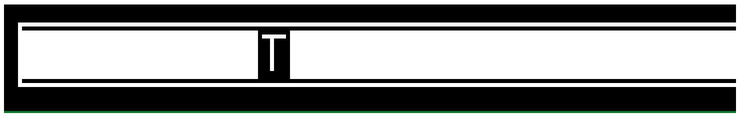
Trigger position

To make best use of the acquisition memory, the MMBasic Pico LA has the option to map the aqcuisition memory different in relation to the trigger event. The position of the trigger event in memory is changed with the “**T**” key.

T = toggle between 1/8, ½, and 7/8 event position in memory.

If you are interested what situation caused the event, at trigger position at 7/8 is handy
If you are interested in the result of an event, then a trigger position at 1/8 is better

Default is trigger position at 1/8, as indicated in the zoom bar.

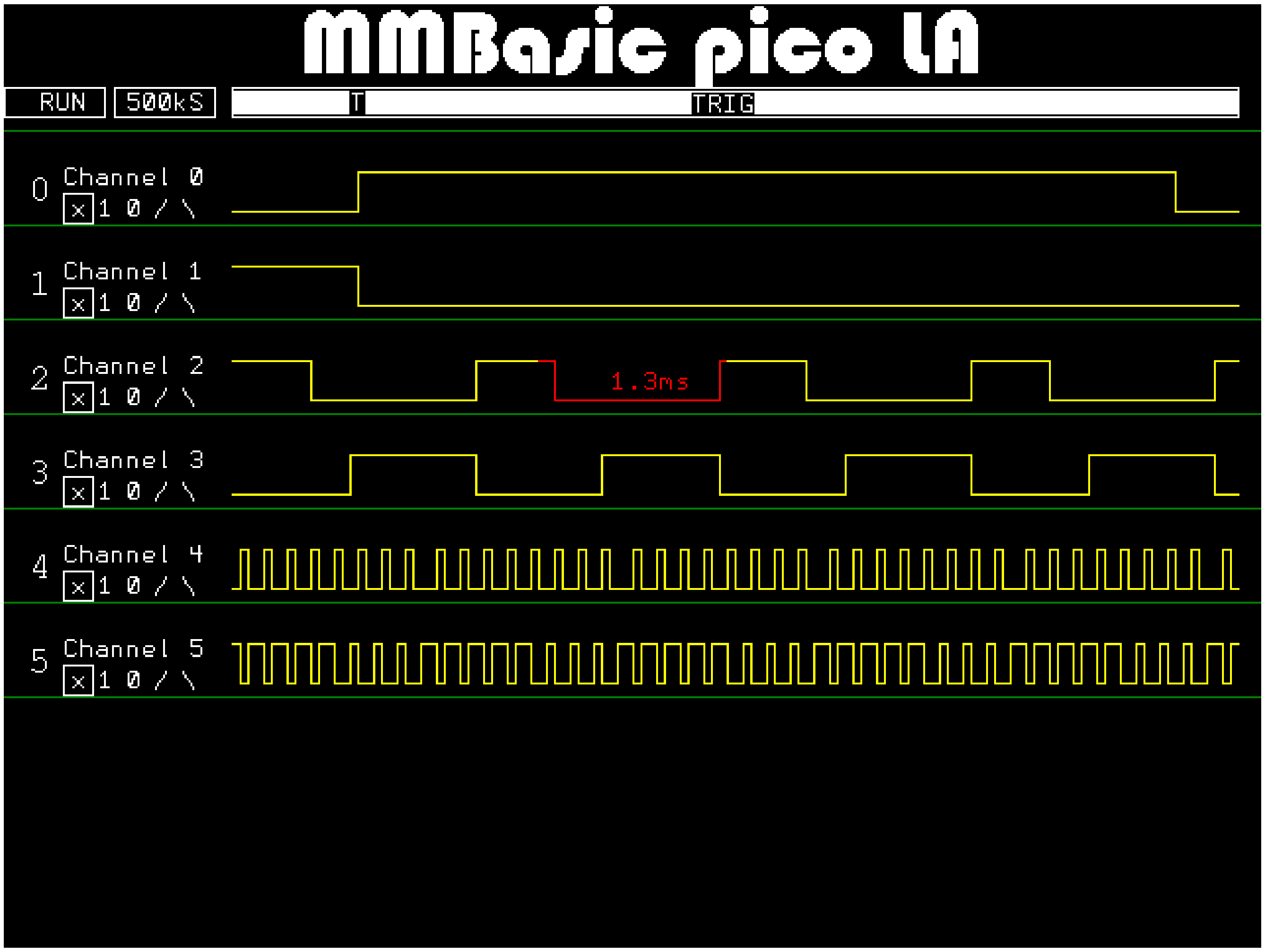


Analysis tools

Software LA_21_b24.bas has following analysis tools implemented that analyze the content of the acquisition memory.

Measuring

Pulse time (high or low) can be measured by pressing the keyboard key “**m**” once. As result part of the screen will color red, indication the area of interest, and the value measured. The area of interest can be moved using the arrow keys to different channels, or different edges.



Pressing the “**m**” a second time will stop measurement mode, and the red section will disappear.

M = start / stop measurement mode
UP = move to waveform above current focus
DOWN = move to waveform below current focus
LEFT = move 1 edge earlier
RIGHT = move 1 edge later

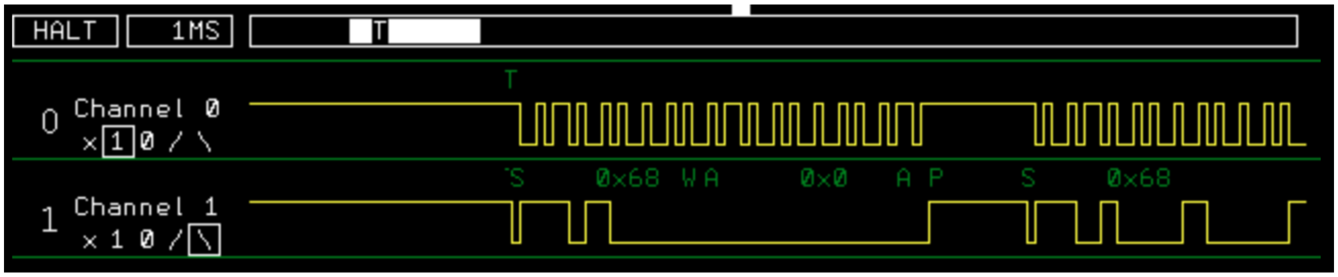
note: the measurement algorithm uses acquisition memory for calculation, but only the memory related to the visual signals. It calculates between 2 edges. When only 1 edge is visible, it will indicate “no edges found”. Zoom out, until 2 edges are visible, then measure again.
The measurement accuracy is the chosen sampling speed.

I2C decoding

I2C decoding as implemented in LA_22.bas requires you to connect SCL to GP0 and SDA to GP1. The decoder will decode the whole acquisition memory, but only show the decoded signals that are inside the visible waveform.

For I2C decoding to work best, it is best to set up a trigger condition. An I2C message starts with an I2C START condition, that is defined as a HIGH to LOW transition of SDA while SCL is HIGH.

When the trigger has happened, press the "I" key on the keyboard to start the decoding. Green text will show the decoded message between the SCL and SDA waveforms. Use the arrow keys to zoom to the desired section.



Pressing "I" again leaves I2C decode mode, and returns to acquisition mode.

i = start / stop I2C decode mode
UP = zoom in
DOWN = zoom out
LEFT = move to earlier captured data
RIGHT = move to later captured data

Status

The MMBasic Pico LA is by no means a finished product. But the current release(LA_22.bas) is usable, and needed some documentation to explain how it can be used.

Specification

Sampling speed: 1kS …. 40MS
Trigger: 16ns...160ns depending complexity of configuration
Channels : 6 (*)
Memory depth: 4096
Display window: 256
Logic: 3.3V

The LA will work on a PicoMiteVGA running 126MHz, but is more responsive at 252MHz.
Software version PicoMiteVGA V50707RC4 or newer.

The acquisition engine runs on PIO1 (state machines 0=sampler and 1=trigger) and uses DMA to transfer samples to the RP2040 memory. The user interface and display are MMBasic code.

Development is documented in TBS forum in thread:

https://www.thebackshed.com/forum/ViewTopic.php?FID=16&TID=15507

(*)= by changing "max_chan_no%=5" to "max_chan_no%=7", and disabling the PicoMiteVGA audio in the OPTION settings, channels GP6 and GP7 can also be used for input to create a 8 channel logic analyzer.