In the name of God

Micro LAB project report

Designing a digital oscilloscope

Sajjad Akherati

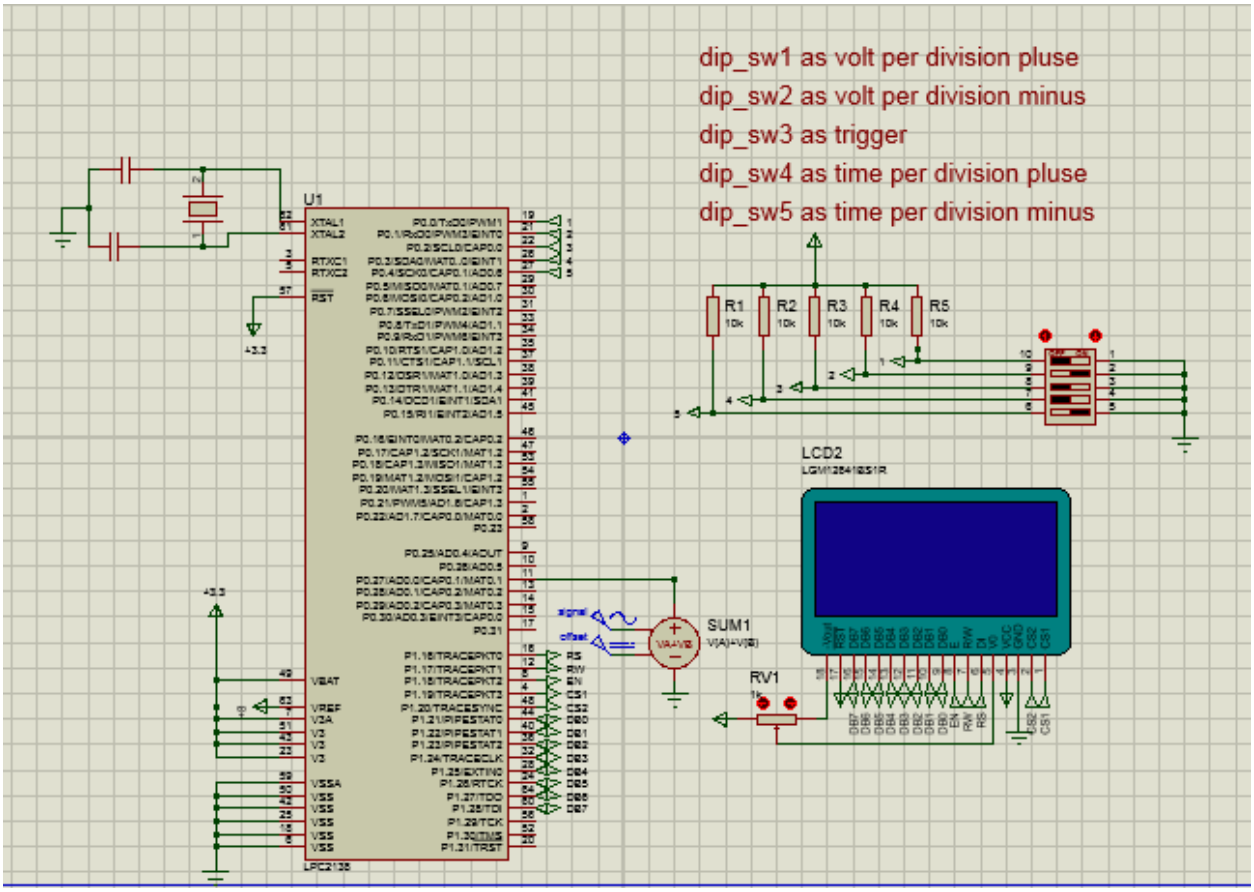
Id = 96101154

In this project, I tried to design an oscilloscope with the following specifications:

1. It displays an analog signal that is enters to its channel.
2. We can adjust its time per division and volt per division.
3. We can trigger and hold on and display a part of received signal.

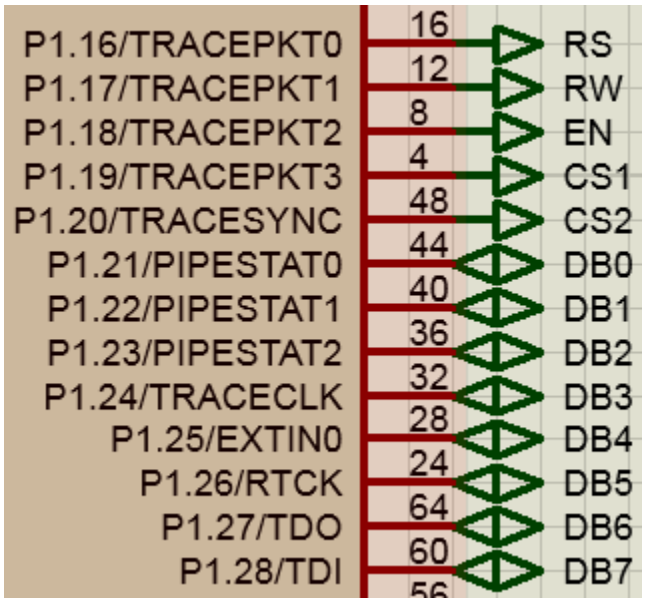
to implement such an oscilloscope, I used a LPC2138 of ARM7 family and a GLCD and some DIP SWITHCH.

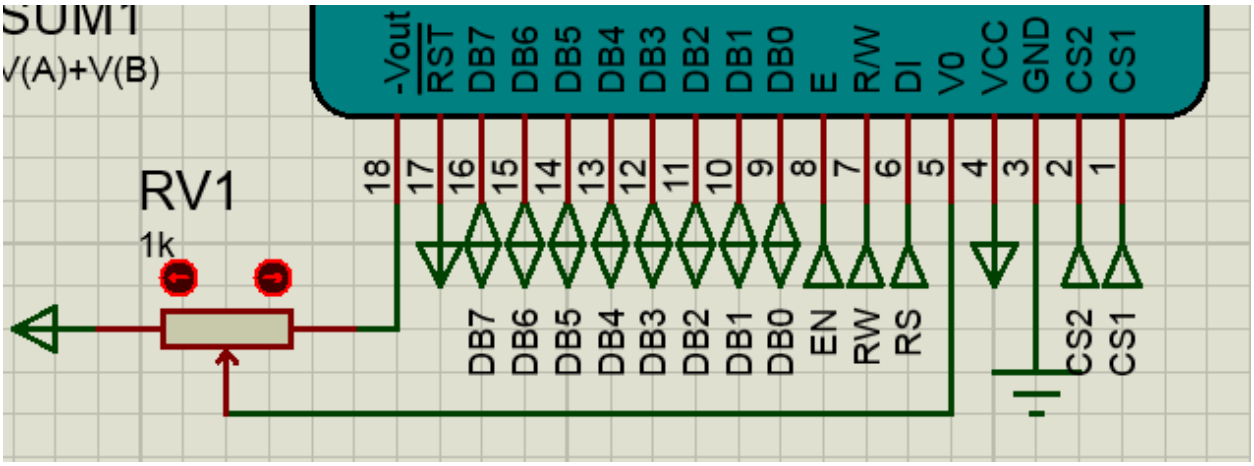
Let's take a look at the circuit:



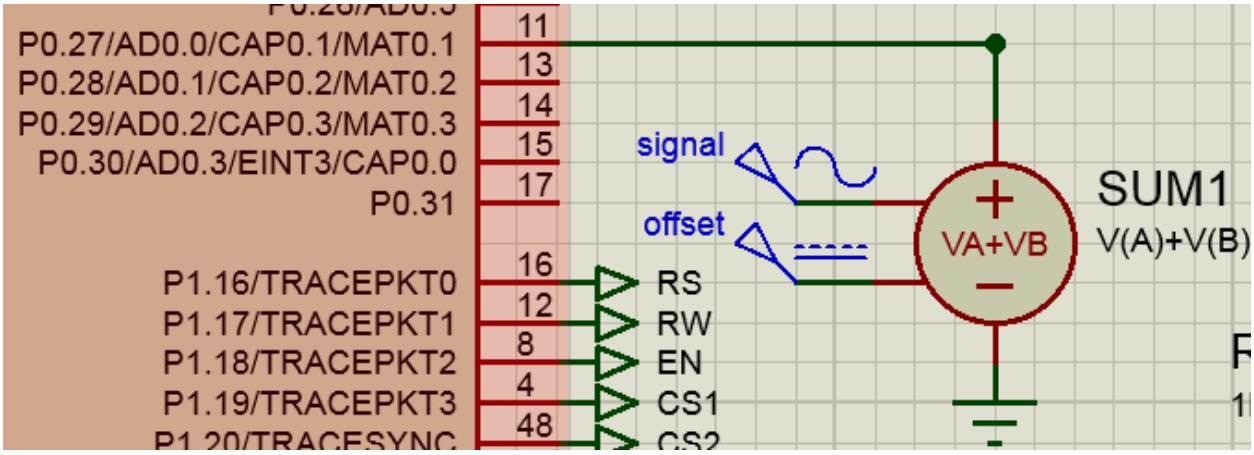
You can see the pin configuration of different parts of the designed oscilloscope in the following figures:

Pin configurations for GLCD:

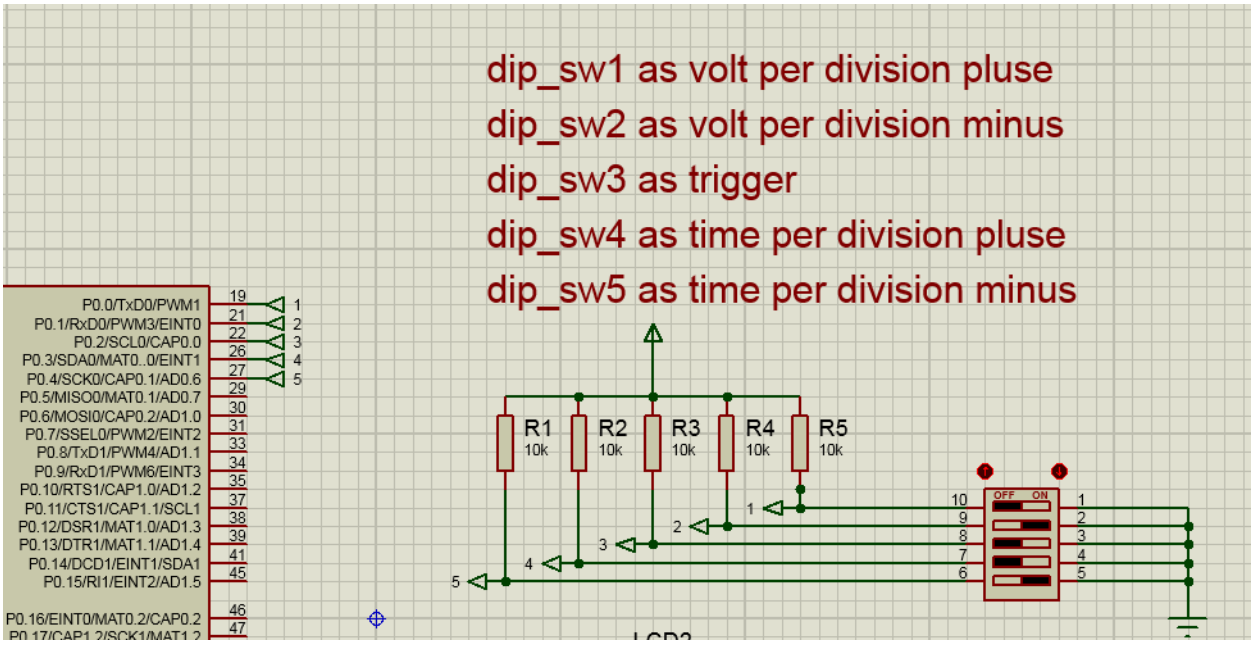




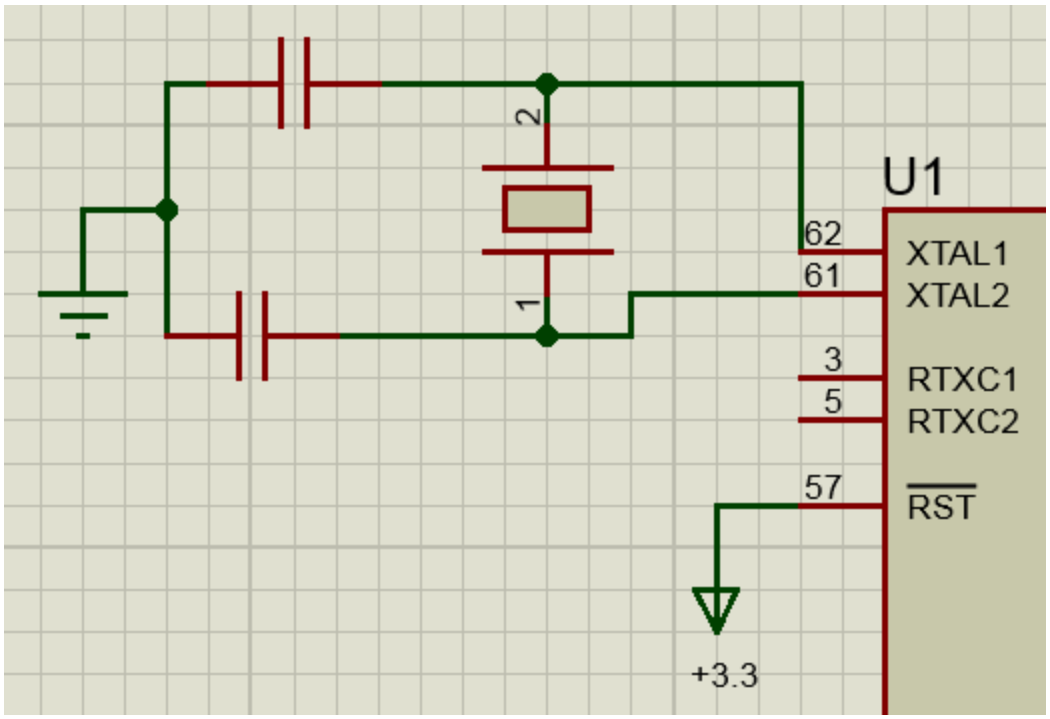
I used the internal ADC of ARM microcontroller: (AD0.0):

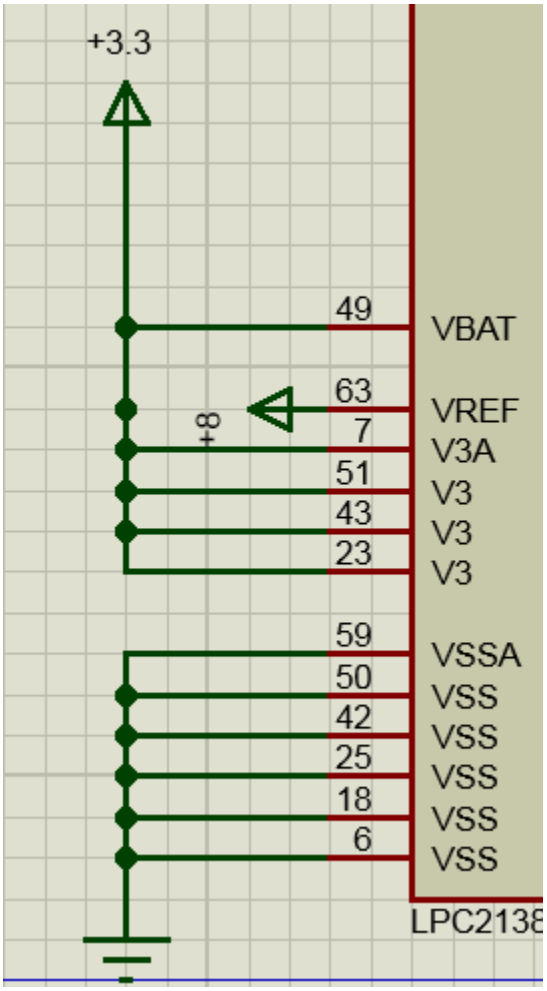


I used some dip switches to adjust VPDIVE, VTDIVE and force triggering; they are shown in the following figure:



And at the end, to use the micro we must set some pins of the micro that showing in the next figures:


U1
62
XTAL1
61
XTAL2
3
RTXC1
5
RTXC2
57
RST
+3.3
2
1


+3.3
49
VBAT
63
VREF
+8
7
V3A
51
V3
43
V3
23
V3
59
VSSA
50
VSS
42
VSS
25
VSS
18
VSS
6
VSS
LPC2138

Notice to the above figure, I used 8v as VREF; we know that it is impossible in the real model but here we just want to simulate the model and use this number has no problem in simulation; in the real one we can use amplifiers and … and instead we can decrease VREF an use lower values.

GLCD library:

In the code to use GLCD I wrote some functions that describes them below:

1. GLCD_SetUp:
   This function is used to configure the controller pins for LCD operation.
2. GLCD_Init:
   This function is used to initialize the graphic LCD. It initializes the LCD for selected mode 8-bit.
3. GLCD_Clear:
   This function clears the graphic LCD and moves the cursor to beginning of the first line on Page0.
4. GLCD_SetDot:
   This function is used to draw a dot. It has the following inputs:
   1) x coordinate in range 0:127
   2) y coordinate in range 0:63
   3) color of pixel to draw. When 0 it is used to clear and when 1 it is used to draw.
5. GLCD_GoToPage:
   This function moves the Cursor to beginning of the specified line. If the requested line number is out of range, it will not move the cursor.
6. GLCD_GoToLine:
   This function moves the Cursor to beginning of the specified line. If the requested line number is out of range, it will not move the cursor.
7. GLCD_GoToNextLine:
   This function moves the Cursor to beginning of the next line. If the cursor is on last line and NextLine command is issued then it will move the cursor to first line.
8. GLCD_DataRead:
   This function is used to read data at current cursor position on GLCD.
9. GLCD_SetCursor:
   This function moves the Cursor to specified position.
10. GLCD_GetXYData:
    This function is used to read data on GLCD line having point (x, y).
11. GLCD_DrawHoriLine:
    This function is used to draw horizontal line. It gets the y coordinates of source and destination and draw it.
12. GLCD_DrawVertLine:
    This function is used to draw vertical line. It gets the X coordinates of source and destination and draw it.

To implement the above function for LPC2138 I used a library was wrote for AVR ATMEGA16 for GLCD;

I used the above library and transferred each function to its corresponding one for LPC2138.

This library is from Osama's LAB and I have attached its user manual in my file; its name is 'GLCD_LIBRARY.pdf'. for more detailed explanation about above functions you can check the attached documentation. It has been attached some link in this documentation that you can refer to see some examples for the AVR ATMEGA16 or download descripted libraries; I also bring these links here:

CodeVisionAVR

GCC

Procedure of my design:

1. In the code, first of all I configured the pins as mentioned in followed pages and then I draw nine vertical lines and five horizontal lines to segment the GLCD to 32 parts like an oscilloscope.
2. After that in an infinitive while loop we have the following steps:
   1- We sampling from the input analog signal.
   2- We draw the signal waveform so far.
   3- We check that the order to triggering or adjusting VPDIVE or TPDIVE has been received and if it is true, we apply that to the oscilloscope system.

Libraries declaration:

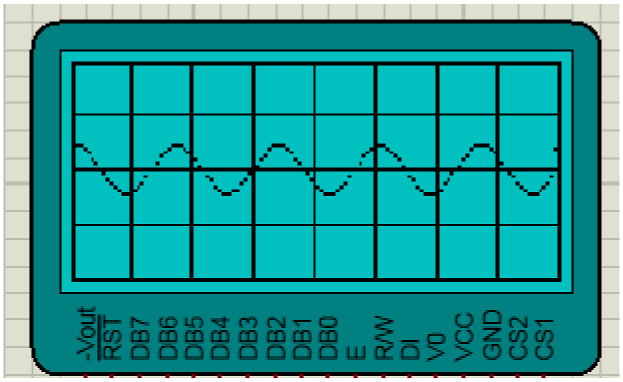
1. Stdutils.h:
   In this h file, there is descripted some useful parameter and variables such as:
   1) Basic data types
   2) Definition of common basic 8bit masks
   3) Port direction configurations
   4) Commonly used constants
   5) Standard enumeration and constant
   6) Some macros for bit manipulations
   7) Macros to find the mod of a number
   8) Macros for Dec2Ascii, Hec2Ascii and Acsii2Hex conversion
   9) Macros to extract the nibbles
   10) Macros to extract the Byte
   11) Some other macros
   I have downloaded the above library from the internet to implement the GLCD library.
2. glcd.h:
   it has been described in the followed pages.
3. gpio.h:
   using this library, we can map ports and pins of the LPC2138 microcontroller to some parameters a value; using the above library to:
   1) define a name for each pin; for example, P0_0 for p0.1 and …
   2) Constants for PIN Function Selection
   3) Determine the function of a GPIO Pin.
   4) Set the direction of a Pin
   5) Write an update the value on the specified Pin
   6) Read the status of the selected pin
   This library was downloaded from the internet.
4. delay.h:

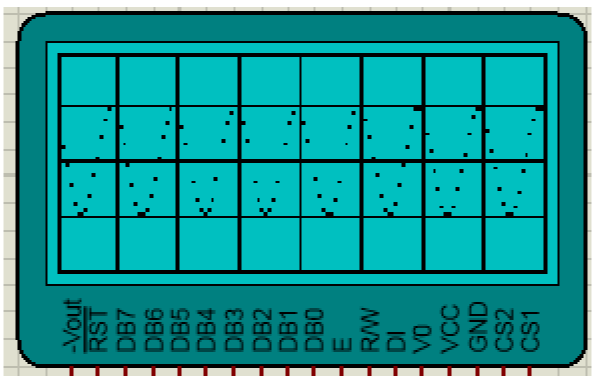
I had written this library to delay the circuit when we need some delay in a part of code.

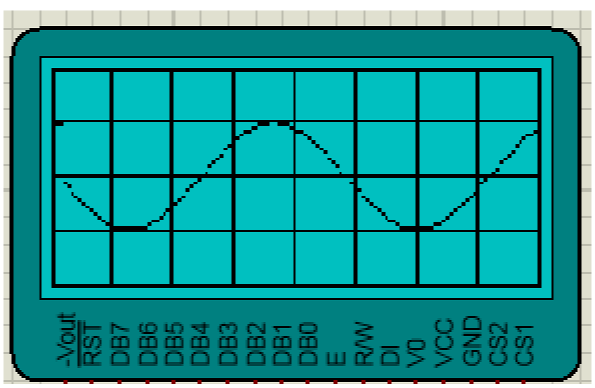
Each function that has declared in the above library has been implemented on an c file.

Results:

In the following figures you can see some results:







The above results are belonging to the same signal and the difference is that we have adjusted the VPDIVE and TPDIVE in different figures.