APPLICATION NOTES

4D SYSTEMS
TURNING TECHNOLOGY INTO ART

ViSi-Genie Magic Button Counters Arduino

DOCUMENT DATE: **24th MAY 2019**
DOCUMENT REVISION: **1.1**

Description

This application note shows how to write a sketch for an Arduino program that handles **REPORT_EVENT** messages coming from the display module.
Note: Workshop Pro is needed for this application.

Before getting started, the following are required:

- Any of the following 4D Picaso and gen4 Picaso display modules:

gen4-uLCD-24PT	gen4-uLCD-28PT	gen4-uLCD-32PT
uLCD-24PTU	uLCD-32PTU	uVGA-III

  and other superseded modules which support the ViSi Genie environment

- The target module can also be a Diablo16 display

gen4-uLCD-24D series	gen4-uLCD-28D series	gen4-uLCD-32D series
gen4-uLCD-35D series	gen4-uLCD-43D series	gen4-uLCD-50D series
gen4-uLCD-70D series		
uLCD-35DT	uLCD-43D series	uLCD-70DT

  Visit www.4dsystems.com.au/products to see the latest display module products that use the Diablo16 processor. The display module used in this application note is the uLCD-32PTU, which is a Picaso display. This application note is applicable to Diablo16 display modules as well.
- 4D Programming Cable / uUSB-PA5/uUSB-PA5-II for non-gen4 displays(uLCD-xxx)
- 4D Programming Cable & gen4-PA, / gen4-IB / 4D-UPA for gen4 displays (gen4-uLCD-xxx)
- micro-SD (µSD) memory card
- Workshop 4 IDE (installed according to the installation document)
- Any Arduino board with a UART serial port
- 4D Arduino Adaptor Shield (optional) or connecting wires
- Arduino IDE
- When downloading an application note, a list of recommended application notes is shown. It is assumed that the user has read or has a working knowledge of the topics presented in these recommended application notes.

Content

Application Overview

To understand this application note more quickly, the reader is advised to read and understand first the application note below.

ViSi-Genie Magic Button Counters

ViSi-Genie Magic Button Counters shows how up and down button counters are implemented in ViSi-Genie with the use of Magic Event objects. The up and down button counters, besides updating the LED digits object on the display, also cause REPORT_EVENT messages to be sent to the serial port. The application note then shows and discusses the format of these messages with the help of the GTX tool.

In this application note, a sketch for an Arduino program that receives and handles REPORT_EVENT messages coming from the display will be discussed. Also, it will be shown how the current value of the LED digits on the display is extracted from the messages so that it can be printed on the serial monitor of the Arduino IDE.

Setup Procedure

At this point, it is assumed that the reader has a working setup of the project described in the application note **ViSi-Genie Magic Button Counters.** The remaining task now is to write the sketch for the Arduino program and upload the program to the Arduino host.

Note: The attached sketch was tested on an Arduino Uno. A software serial port was used for communicating with the display. The hardware serial port, Serial0, was used for communicating with the Serial Monitor of the Arduino IDE.

Program the Arduino Host

A thorough understanding of the application note **ViSi-Genie Connecting a 4D Display to an Arduino Host** is required before attempting to proceed further beyond this point. **ViSi-Genie Connecting a 4D Display to an Arduino Host** provides all the basic information that a user needs to be able to get started with ViSi-Genie and Arduino. The following is a list of the topics discussed in **ViSi-Genie Connecting a 4D Display to an Arduino Host**.

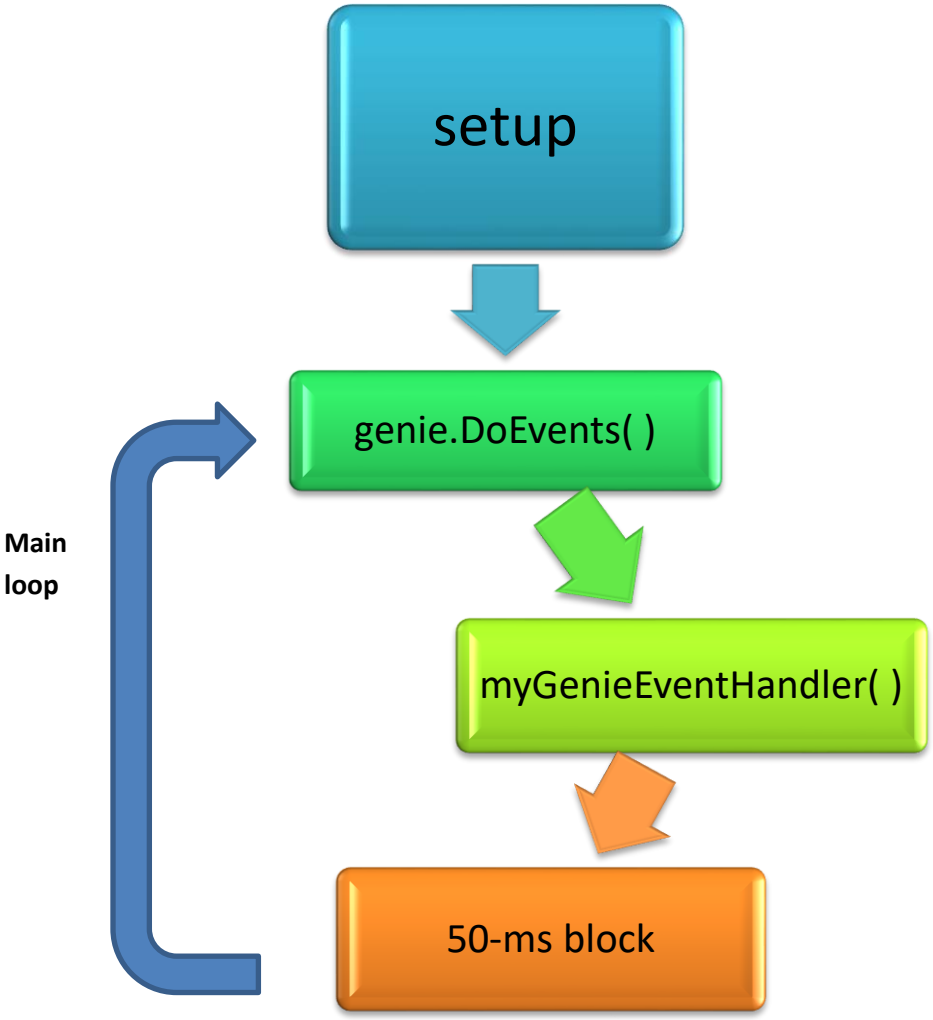
- How to download and install the ViSi-Genie-Arduino library
- How to open a serial port for communicating with the display and how to set the baud rate
- The genieAttachEventHandler() function
- How to reset the host and the display
- How to set the screen contrast
- How to send a text string
- The main loop
- Receiving data from the display
- The use of a non-blocking delay in the main loop
- How to change the status of an object
- How to know the status of an object
- The user's event handler

Discussion of any of these topics is avoided in other ViSi-Genie-Arduino application notes unless necessary. Users are encouraged to read **ViSi-Genie Connecting a 4D Display to an Arduino Host** first.

Understanding the Demo Sketch

Open the sketch "**MagicButtonCountersSS**" attached to this document.

Program Flow



Setup

- Assignment of the software serial pins. This is actually prior to the setup routine. Visit the Arduino website for more information about the SoftwareSerial library.
- Initialization of serial communication
- Reset routine (this is important)

Main loop: The 50-ms Block

This is the conditional block shown below.

```
if (millis() >= waitPeriod)
{
  //do something here
  waitPeriod = millis() + 50;
}
```

This block runs every 50 ms (approximate). The user can change the frequency of execution of this block by changing the literal constant "50" to another value. The block above does nothing except to update value of the variable "**waitPeriod**". As indicated above, this is where the user can insert other lines for the sketch. The above is an implementation of the "**non-blocking**" delay method. When writing an Arduino sketch that communicates with a 4D display using the ViSi-Genie-Arduino library, it is best to use a non-blocking delay instead of a "**blocking** delay". A blocking delay usually makes use of the function "**delay(**period in ms**)**". The use of a blocking delay is not ideal since the processor does nothing for the duration of the delay period. The method "**genie.DoEvents()"** of the ViSi-Genie-Arduino library has timing-sensitive processes that run in the background. Blocking delays, therefore, tend to suspend the execution of the internal processes of the ViSi-Genie-Arduino library. Consequences include programs that seem to run very

slowly or messages coming from the display being lost. This explains why the use of a non-blocking delay is preferred. For more information on blocking and non-blocking delays, visit the Arduino community.

Main Loop: genie.DoEvents()

This method executes the internal processes of the library, one of which is the queueing of events or messages coming from the display. This method also executes the user-defined event handler "**myGenieEventHandler()**".

```
void loop()
{
  static long waitPeriod = millis();

  genie.DoEvents(); // This calls th

  if (millis() >= waitPeriod)
  {
    //do something here
    waitPeriod = millis() + 50;
  }
}
```

Main Loop: genie.DoEvents() - myGenieEventHandler( )

This is the user-defined event handler. It takes a message from the queue and evaluates it in a manner which is to be defined by the user.

For this application, the event handler performs the following steps (as indicated below).

1. Take a message or an event from the queue of events.
2. Evaluate if the event is a REPORT_EVENT message.
3. Evaluate if the event is from a LED digits object.
4. Evaluate if the event is from Leddigits0.
5. If the conditions in 2, 3, and 4 above are met, the value inside the event is extracted.
6. The value is printed.

```
void myGenieEventHandler(void)
{
  genieFrame Event;
  genie.DequeueEvent(&Event);

  int ledDigits_val = 0;
  //If the cmd received is from a Reported Event
  if (Event.reportObject.cmd == GENIE_REPORT_EVENT)
  {
    if (Event.reportObject.object == GENIE_OBJ_LED_DIGITS)
    {
      if (Event.reportObject.index == 0)
      {
        ledDigits_val = genie.GetEventData(&Event);
        Serial.print("Counter value: ");
        Serial.println(ledDigits_val);
      }
    }
```

Set Up the Project

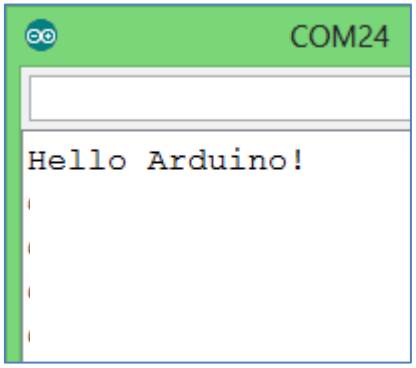
Refer to the section "**Connect the Display Module to the Arduino Host"** of the application note "**ViSi-Genie Connecting a 4D Display to an Arduino Host**" for the following topics:

- Using the New 4D Arduino Adaptor Shield (Rev 2.00)
  - Definition of Jumpers and Headers
  - Default Jumper Settings
  - Change the Arduino Host Serial Port
  - Power the Arduino Host and the Display Separately
- Using the Old 4D Arduino Adaptor Shield (Rev 1)
- Connection Using Jumper Wires
- Changing the Serial port of the Genie Program
- Changing the Maximum String Length

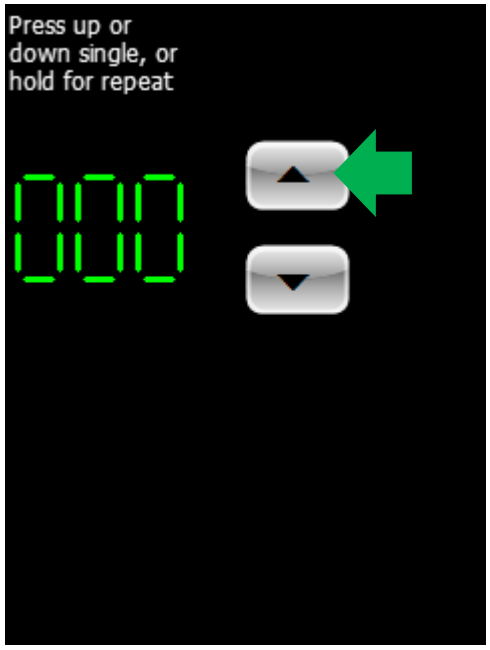
Note: The attached Arduino sketch uses a software serial port to communicate with the display module and a hardware serial port to communicate with Serial Monitor running on the PC.

Expected Serial Terminal Output

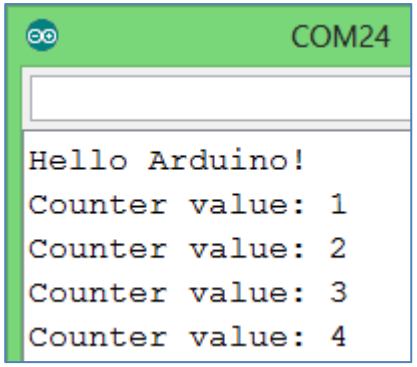
Start Up



Count Up

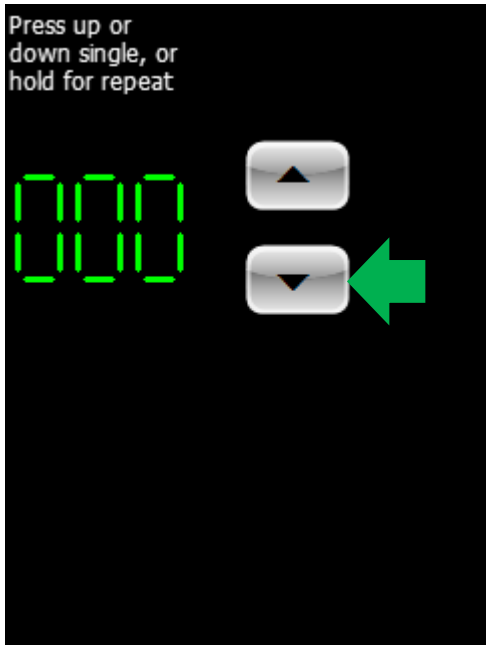


1. Press and hold the count up button.

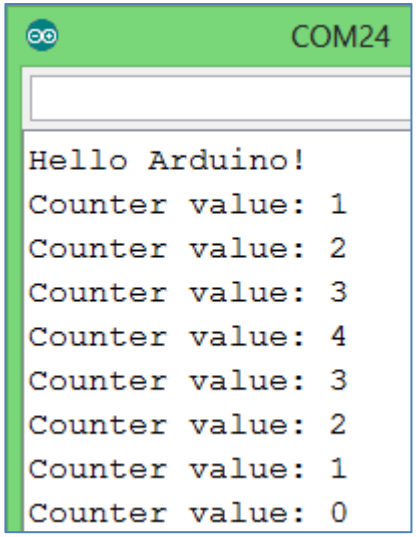


2. The value of Leddigits0 should now increment. The values are printed on the serial monitor.

Count Down



1. Press and hold the count down button.



2. The value of Leddigits0 should now decrement. The values are printed on the serial monitor.

Proprietary Information

Disclaimer of Warranties & Limitation of Liability