AAT3172 Flash Programming Techniques
Using the AS²Cwire™ Serial Interface

Introduction

The AAT3172 is a software programmable flash LED driver with two independently programmable constant current LED channels. Using only a single wire, complete flash LED control is provided by AnalogicTech's Advanced Simple Serial Control™ (AS²Cwire) serial interface. Two independently programmable constant current LED channels enhance flash LED control flexibility and allow multiple LEDs to be driven with regulated current. The AAT3172 can also be programmed into a constant output voltage mode such that, as an example, keypad lighting and flash LED control can be combined.

The focus of this application note is on programming techniques for camera phone flash applications and corresponding performance considerations. Based on example application scenarios, programming steps are given such that the user can gain a clear understanding of AS²Cwire usage with the AAT3172.

Figure 1 illustrates the typical single LED application where both LED channels are tied together to provide a single, high current drive capability. Because two independently regulated channels are available, two LEDs can be driven in parallel enabling an improvement in overall light efficiency. Further, two LEDs can simultaneously be driven at independent current levels to provide higher resolution brightness control.

For a more complete understanding of the AS²Cwire serial interface, please consult AnalogicTech's AS²Cwire application note (AN-111). Other applications, such as combined flash and keypad lighting, are discussed in additional application notes found in the Resources section of the AnalogicTech website.

Figure 1: Typical Single LED Application.

Programming a Flash Pulse

The complete programmability of the AAT3172 is presented in the AS²Cwire tables found in the product datasheet. The tables contain all of the necessary programming information in a simplified format.

Determining the programming steps involved for a particular application is straightforward. According to the application’s requirement, find the settings in the AS²Cwire tables that match closest to the desired LED current level. Determine the address and data settings corresponding to the chosen current levels and apply the indicated number of EN/SET rising edges listed in the datasheet.
Suppose an application requires the maximum current setting of 300mA/channel (i.e., 600mA total LED current) for a flash pulse time of 500ms. When consulting the datasheet, 300mA/channel is found in the Constant Current Sink Nominal Programming Levels table (Data 1 in the HI Scale column). Because there are HI and LO Scale columns, the HI/LO Scale Register must also be considered. From the HI/LO Scale Register table, the default is the LO Scale (at startup, all registers default to Data 1). Therefore, the HI/LO Scale Register must be programmed to Data 2, such that both channels F1 and F2 operate on the HI Scale.

The programming steps necessary for turn on to 300mA/channel involve a single write of Address 3, Data 2. The single write of Address 3, Data 2 will address the HI/LO Scale Register and set it to HI for channels F1 and F2. The device will then turn on to 300mA/channel. According to the number of EN/SET rising edges listed in the datasheet, apply a group of 20 edges corresponding to Address 3, pause for $T_{LAT}$, and then apply a group of 2 edges corresponding to Data 2.

![Figure 2: Programming Address 3, Data 2 Turns on to the Maximum Flash Current of 600mA.](image)

Figure 2 depicts the resulting behavior. Immediately after writing the address group of edges, the device will temporarily turn on to 50mA/channel. Then, once the data group of edges is written, the device increases to the full 300mA/channel current level. This is further illustrated in the scope snapshots shown in Figures 3 and 4.

![Figure 3: 600mA, 500ms Pulse, Zoomed-in to the Startup Event.](image)

![Figure 4: 600mA, 500ms Pulse Zoomed Out.](image)
Consider a new application that requires the 214mA/channel setting (i.e., 428mA total LED current). The Constant Current Sink Nominal Programming Levels table of the datasheet indicates that Data 4 corresponds to 214mA/channel. This case differs slightly from the previous one because now the data setting is something other than the default Data 1. In this case, the programming steps involve one write of data followed by one write of address and data.

The programming steps necessary for turn on to 214mA/channel involve a write of Data 4, followed by a write of Address 3, Data 2. The first write of Data 4 will set the current level to 36mA/channel (the HI/LO Scale Register defaults to LO), and the subsequent write of Address 3, Data 2 will address the HI/LO Scale Register and set it to HI for channels F1 and F2. The device will then change from 36mA/channel to 214mA/channel. These programming steps are illustrated in Figures 5, 6, and 7.

![Figure 5: Programming Steps for Turn On to 428mA Flash.](image)

![Figure 6: 428mA, 500ms Pulse, Zoomed-In](image)  
Figure 6: 428mA, 500ms Pulse, Zoomed-In to the Startup Event.

![Figure 7: 428mA, 500ms Pulse Zoomed Out.](image)
Achieving Multi-Level, Torch/Flash Control

Some applications require a more sophisticated flash control than programming a single 500ms pulse, as presented earlier. A user may need multi-level control where lower intensity illumination, such as torch mode, can be provided in addition to high intensity flash illumination. This multi-level flash programming requirement is easily accomplished with AS²Wire and the AAT3172.

Suppose an application requires a 100mA / 600mA torch / flash control function. This situation requires multiple programming steps that turn on the flash LED to 100mA and, some time later, change the current level to 600mA. 50mA/channel corresponds to Data 1 of the LO Scale column in the Constant Current Sink Nominal Programming Levels table of the datasheet. Since the LO Scale is the default and 50mA/channel is the default, the programming step only involves taking EN/SET from low to high with a single rising edge.

To subsequently change the current level from 50mA/channel to 300mA/channel, an additional write of Address 3, Data 2 is required. This is illustrated in Figure 8.

![Figure 8: 100mA / 600mA Torch / Flash Control.](image)

Consider a different application that requires a 152mA / 600mA torch / flash control function. 76mA/channel corresponds to Data 13 of the HI Scale column in the Constant Current Sink Nominal Programming Levels table of the datasheet. Therefore, turn on to 76mA/channel requires a write of Data 13, followed by Address 3, Data 2.

To subsequently change the current level from 76mA/channel to 300mA/channel, an additional write of Address 0, Data 1 is required. This is illustrated in Figure 9.

![Figure 9: 152mA / 600mA Torch / Flash Control Function.](image)
Triggering Flash from an External Strobe Signal

Some camera modules are equipped with a separate strobe signal such that they can independently request the high intensity flash. Since the AAT3172 is software programmable, a strobe signal from the camera module can be easily handled. A simplified representation of this application scenario is illustrated in Figure 10.

In this application scenario, the system processor (µP) communicates with the camera module through a serial interface, such as an I²C bus. To signal for flash, the camera module has a separate, open-drain STRB output that the µP monitors. When the STRB output transitions from high to low, the µP detects the change in state and programs the AAT3172 to the high intensity setting. When the STRB signal subsequently transitions from low to high, the µP then puts the AAT3172 into shutdown, turning off the flash.

With adequate µP configuration, the AAT3172 flash LED driver is synchronized to the STRB signal from the camera module with a fast response time. If the µP detects the STRB signal transition by interrupt-on-change or a similar real-time technique, the time delay can be less than 1ms. Given the fast response and simplicity, the AAT3172 is well suited for applications involving an external strobe signal.

Figures 11 and 12 are examples of flash synchronized to an external strobe signal. Figure 11 illustrates the STRB signal triggering a 600mA flash pulse; Figure 12 illustrates 100mA torch mode and a subsequent 600mA flash pulse triggered by the STRB signal.
The results from an actual test fixture are shown in Figure 13. When the strobe signal transitions from high to low (trace 3), it is detected by the µP and the AAT3172 is programmed from the 152mA setting to its maximum current setting of 600mA.
Timing Analysis

The overall timing delays associated with AS^2Cwire programming are analyzed in the following discussion and the overall impact is quantified.

For typical applications, the time spent programming the serial interface is on a significantly smaller scale than the flash pulse time. For a 500ms flash pulse, the total time dedicated to AS^2Cwire programming need not exceed 0.2% (less than 1ms) of the total flash pulse time.

![Figure 14: Serial Interface Activity is Less than 1ms.](image)

From the illustration in Figure 14, the timing events and overall time dedicated to AS^2Cwire programming can be more clearly understood. At time t₁, a group of edges is started on the EN/SET pin signifying address. At time t₂, a group of data edges is applied. t₂ is 500µs later than the last address edge to accommodate the TLAT maximum specification found in the datasheet. At time t₃, the device sets the output to the programmed condition.

To quantify the overall timing impact, the THI and TLO times of the serial interface edge timing must be defined. A reasonable number to assign for THI and TLO is 300ns. From this, the overall timing delay quantities can be calculated. Address 3 requires 20 edges and a 500µs wait is required between the address group of edges and the data group, so:

\[
 t_2 - t_1 = 20 \cdot (2 \cdot 300\text{ns}) + 500\mu\text{s} = 512\mu\text{s}
\]

Since two edges are required for the data group and the output changes after TLAT, which is typically 150µs:

\[
 t_3 - t_2 = 2 \cdot (2 \cdot 300\text{ns}) + 150\mu\text{s} = 151\mu\text{s}
\]

From these results, the total time dedicated to AS^2Cwire programming is:

\[
 t_3 - t_1 = 512\mu\text{s} + 151\mu\text{s} = 663\mu\text{s}
\]
Now it can be seen that, in a typical application, less than 0.7ms is dedicated to AS²Cwire programming. This is a small fraction of a typical flash pulse time that runs hundreds of milliseconds.

The timing analysis is completed by also analyzing the turn-off event. At time t4 in Figure 14, EN/SET goes low and there is a small delay of $T_{OFF}$ until the output turns off at time t5. Because the typical curves in the datasheet show $T_{OFF}$ and $T_{LAT}$ to be about 100µs to 150µs:

$$T_{OFF} = t_5 - t_4 \approx 150\mu s$$

This analysis can further be understood by referring back to the oscilloscope snapshots found in Figures 3 and 4. In the case of Figure 4, the activity on the EN/SET pin is not noticeable and it appears that the output turns on as soon as EN/SET goes high. However, the zoomed-in snapshot of Figure 3 shows the groups of EN/SET pulses and exactly when the output turns on.

**Concluding Remarks**

The AAT3172 and AS²Cwire interface, combined, provide a highly flexible solution for camera phone flash applications. Many examples of software programming have been provided in this application note so the user can quickly gain a clear understanding of programming techniques for flash applications. With this information, the user can most effectively use the device.