

Introduction

The presence of offsets in operational amplifiers is an unfortunate fact of life. These offsets become particularly troublesome in implementations with high gain where the offset of the OpAmp is, approximately speaking, multiplied by the gain of the circuit. In such cases, the resulting output offset can be enough to limit the available signal swing, or can even saturate the OpAmp in extreme cases. This effect can occur in unexpected places as well as gain stages. Some filter configurations, for example, have high gains internally which can detrimentally interact with OpAmp offsets to give poor overall circuit performance.

The chopping schemes in the CAB OpAmp (and chopping amplifier proper) do not operate well with a large intrinsic OpAmp offset. The combination of large gain, significant OpAmp offset and chopping may lead to a large square wave being produced at the OpAmp output. While this is usually at a frequency high enough not to be problematic, it can lead to settling problems in subsequent circuits.

Circuit topologies exist which do a lot to deal with OpAmp offsets. Half cycle structures are often employed to compensate for OpAmp offset, but these are not always possible to deploy in a given situation. The programmable nature of the AN231E04 means it is practically impossible to compensate for the effects of OpAmp offset in all possible situations, so the AN231E04 deals with offsets by greatly reducing them at source.

The AN231E04 dpASP chip contains an auto-null scheme to reduce the input offset of the CAB, IO and chopper amplifiers. This document explains what it is and how to use it by either configuration or re-configuration. It is divided into 2 main sections – the first is a detailed description of auto-nulling in the AN231E04, the second is a practical guide on how to use it.

1. Detailed Description of Auto-nulling

1.1 Auto-null offset correction

The auto-null scheme is a well known and widely used technique to reduce OpAmp offsets. Essentially it works by injecting a current into the OpAmp which acts to compensate for the intrinsic offset. The resulting OpAmp has an offset very much reduced from its intrinsic value. An auto-null scheme has been implemented in the complex IO and the CAB OpAmps so that a low offset

signal path can be made from input to output pin in the dpASP array. A chopping amplifier is also provided in the IO which can give significant gain and both auto-null and chopping offset correction; this is for situations where the input signal is small and needs the application of a large gain before signal processing elsewhere.

The dpASP auto-null schemes operate on the differential offsets only. The common mode offsets are left uncorrected because they are not multiplied by user circuit topology. Common mode offsets are rejected by default by the differential nature of the dpASP.

1.2 Details of the auto-null implementation

Activation of the auto-null schemes is possible in a number of ways, as detailed below. These methods differ only in terms of the users programming choice; the mechanisms within the chip are the same. Throughout this document “OpAmps” refers generically to the operational amplifiers in the complex IO, the CAB or the chopper amplifier.

1.2.1 Primary configuration

This is the primary usage mechanism. Auto-null is selectively activated for all OpAmps which are powered up for use in a given application. Other OpAmps are left powered down with their intrinsic offsets still present. After configuration and the subsequent auto-null is complete, the OpAmps come alive and signal processing begins.

Once an auto-null has been completed for an OpAmp, the offset correction values are stored in local RAM. This will retain its values until the device is reset, power is cycled, or auto-nulling is disabled. The RAM is not part of the memory map, and is local to each OpAmp.

1.2.2 Reconfiguration

After a primary configuration it may be required to periodically re-run the auto-null if operating conditions have significantly changed. Changes in operating conditions can cause the OpAmp offset to drift and the compensation previously applied may then no longer be correct. It may also be required to power up OpAmps which were not previously auto-nulled, and then activate the auto-null scheme for them. Both these options are done via re-configuration.

1.2.3 Suspended animation

It is possible to perform an auto-null on an OpAmp (either via primary or re-configuration) then power the OpAmp down. When such an OpAmp is subsequently

powered up it will have retained its low offset values, provided there have been no significant changes in operating conditions and provided that there has been no power cycle, reset or disabling of auto-null.

1.2.4 No auto-null

By default auto-null is on but can be disabled easily, and may also be de-activated after being deployed. Either of these states will yield an OpAmp which displays its intrinsic offset voltage. Although this eradicates the delay required to auto-null before an OpAmp can be used, it is probably of limited value.

1.3 OpAmp circuit details

Autozero schemes exist in the IO, CAB and chopper amplifiers. These all have the same basic architecture, where a test current is injected into the differential input stage, as per Figure1 below. The magnitude and polarity of the current necessary to cancel out the OpAmp offset-voltage is determined by a successive-approximation routine.

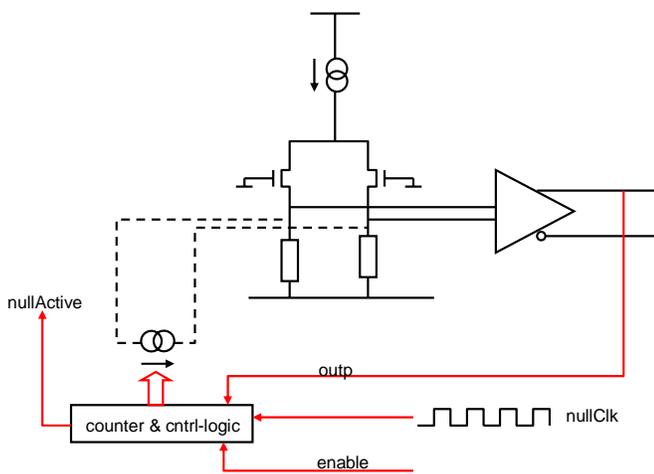


Figure 1. Principle of offset nulling

1.4 Auto-null User Requirements

The auto-null cycle for any OpAmp requires some external conditions to be met requiring a configuration from the user and an external clock. Necessary conditions are:

- Providing an external clock on either ACLK or DCLK.
- Setting a clock divider to give a suitably slow internal clock.
- Setting a bit(s) to enable the auto calibration.
- Waiting the specified duration before expecting valid functionality from the OpAmp undergoing auto-null. OpAmps which are not set to run an auto-null cycle will continue signal processing.

The external clock is required to drive the state machine(s) which operate the auto-null schemes, and it must be suitably divided down to around 100Hz internally. A special clock divider and associated dedicated routing is provided for this purpose, so no special connectivity is required from the user. The auto-null clock has a fixed divider by 1000 as a pre-scaler and a subsequent post-scaler to allow the user finer control of the final auto-null clock frequency.

The AN231E04's auto-null scheme uses the OpAmps in open loop mode, where they may respond at a frequency equal to their dominant pole frequency. 100Hz is regarded as a safe value. Higher frequencies may be used with the increasing risk that the auto-null result will not be as small as it otherwise would be, due to incomplete OpAmp settling.

The required action from the user can all be accomplished in a single primary configuration, including any other required circuit functions if desired. An example is given in appendix A.

1.5 End of auto-null indication

The auto-null scheme has an explicit way of signaling externally that it has finished. Pins LCC_B, MEMSETUP or MEMCLK can be programmed to give this function once the device is configured.

1.6 Chopping and auto-null interactions

The use of chopping is a good way to render the offset of an OpAmp to appear very close to null. While chopping is active, the differential output of an OpAmp switches between $+offset_voltage$ and $-offset_voltage$ at the chopping frequency. If large gains are applied then the output approximately switches between $+Gain*offset_voltage$ and $-Gain*offset_voltage$ which can potentially be several 100mV or more. The chopped amplifier will continue to operate under these conditions, but the large square wave being driven out causes settling time problems for this and other circuits, and is also a reasonably strong noise coupling source.

For these reasons it is highly desirable to auto-null any chopped amplifier. Thus the effective magnitude of the $offset_voltage$ quantity is much smaller and the settling and noise problems are greatly reduced.

2. Practical Guide to Auto-nulling

2.1 Set-up in AnadigmDesigner2

Fortunately, AnadigmDesigner2 makes it very simple to use auto-nulling in the AN231E04. All the user has to do is enable it by doing the following:

- a. Right click over the circuit
- b. Select the Chip tab
- c. Check the box labeled "Use Offset Nulling"

This is shown in figure 2.

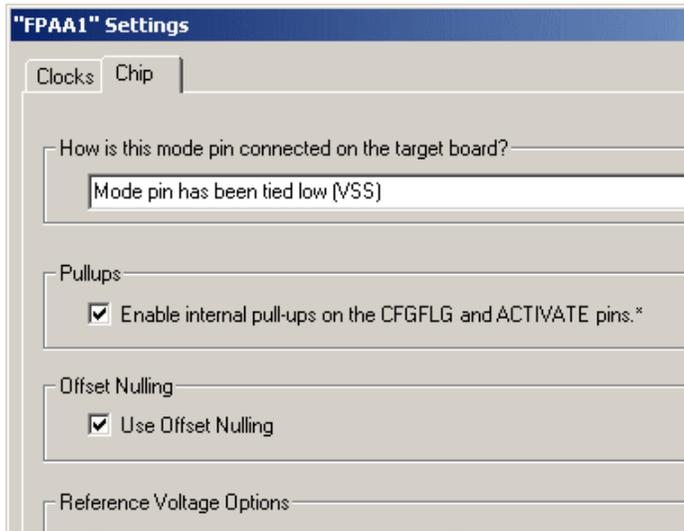


Figure 2. Enabling offset nulling in AD2

This is all the user has to do since AnadigmDesigner2 will now create the appropriate configuration data to enable offset nulling and calculate the required nulling clock divider to give the correct frequency for the nulling clock (~100Hz). In fact the user doesn't even need to do this because auto-nulling is enabled by default in AnadigmDesigner2.

There are some points to note however:

- a. Only OpAmps that are actually used in the circuit will be auto-nulled.
- b. Once auto-nulled, OpAmps will stay auto-nulled until reset, power down or auto-nulling is disabled.
- c. Auto-nulling requires a delay of approximately 60ms immediately after configuration, during which time the OpAmps being auto-nulled cannot be used (see section 2.3).

Regarding point a. above, if a circuit is loaded with a primary configuration in which auto-nulling is enabled, and that circuit does not use every OpAmp, and then the

user wishes to do a re-configuration (i.e. an update, not a primary configuration) to another circuit which enables previously unused OpAmps, then those OpAmps (that were disabled during the primary configuration + auto-nulling) will be non-functional. In this situation the user must restart auto-nulling. He can do this in one of the following ways:

- i. If the reconfigured circuit has auto-nulling disabled, after the reconfiguration has been sent to the dpASP a second short auto-null reconfiguration is sent that turns auto-nulling back on. This forces auto-nulling to be run again for all of the OpAmps in the new circuit. The short auto-null reconfiguration is shown in Appendix B.
- ii. The alternative is to reset the circuit and send the second circuit as a primary configuration, with auto-nulling enabled of course. This will have the additional delay of a reset as well as a new auto-nulling sequence.

NOTE : the short auto-null reconfiguration mentioned in i. above and described in Appendix B can be sent to the dpASP after a primary configuration or reconfiguration in which auto-nulling was disabled. This will cause auto-nulling to be executed. However, if the preceding configuration had auto-nulling enabled, the short auto-null reconfiguration will have no effect.

If the user cannot tolerate any delay when changing to the new circuit, he can do one of the following:

- i. Start with a primary circuit that has every OpAmp enabled that he wishes to use in subsequent circuits, and of course has auto-nulling enabled. All of the OpAmps will be auto-nulled after the primary configuration, and he can then reconfigure between all of his circuits (keeping auto-nulling enabled). For each of the reconfigured circuits, the OpAmps will keep their auto-nulled offsets (ref point b. above) and there will be no additional auto-nulling sequences so the reconfigured circuits will become functional immediately.
- ii. If offsets are not critical in the new circuit then the user could disable auto-nulling in the reconfiguration. This will mean that the circuit becomes functional immediately but the OpAmps will have their (larger) intrinsic offsets.

Regarding point c. it is important to note that auto-nulling incurs a delay, and that OpAmps will be non functional while being auto-nulled. The user should therefore ensure that there is a delay of 60ms after configuration before the circuit is used. It is possible to enable the dpASP to assert a signal when auto-nulling is complete (see figure 3).

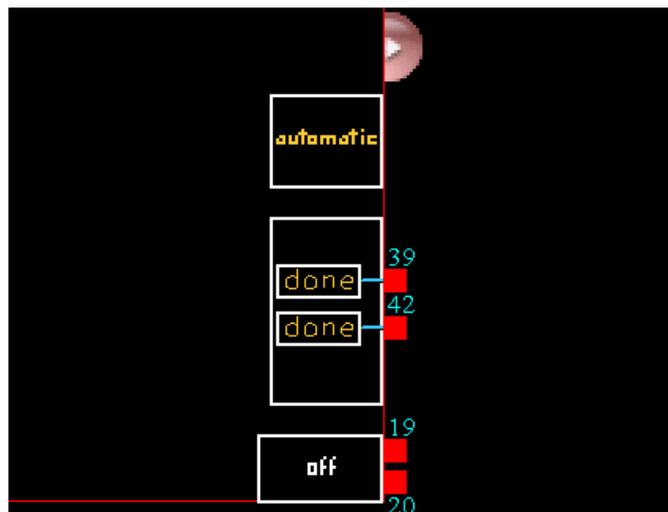


Figure 3. Digital IO Cell with auto-null done

To enable an auto-nulling done signal, double click on the digital IO cell in AnadigmDesigner2 and select Auto-null Done for either Dout1 (pin 39) or Dout2 (pin 42). Figure 3 shows the digital IO cell with both signals enabled. Note that pin 39 is also called LCCb and pin 42 is also called MEMCLK.

2.2 Auto-nulling Delay

Figure 4 shows waveforms for a primary configuration of a circuit containing a single GainInv CAM. The bottom trace shows SCLK. After completion of configuration (SCLK stops pulsing) the auto-null done signal goes low and the analog output goes to VMR but there is no output signal. After about 60ms the analog output starts, and after that the auto-null done signal is asserted.

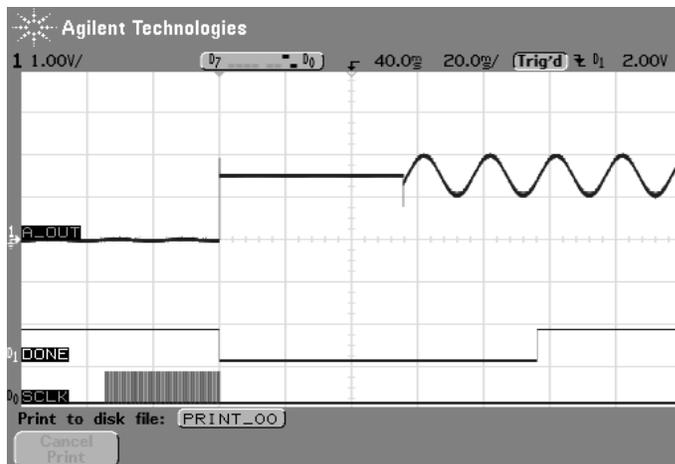


Figure 4. Primary configuration with auto-null

It can be seen that the done signal does not get asserted immediately after the OpAmp starts to work. The purpose of the auto-null done signal is simply to provide the user with a signal to tell the rest of his system that it is safe to use the analog circuit in the dpASP. The user can either wait 60ms or use the done signal.

The exact delay to completion of auto-nulling and to the assertion of the auto-null done signal can be calculated from the master clock frequency and the auto-nulling clock divider. As stated previously, the auto-nulling clock is obtained by dividing the master clock by a fixed pre-scaler of 1000 and a variable secondary divider. The secondary divider is calculated by AnadigmDesigner2 to give an auto-null clock of as near to 100Hz as possible. AnadigmDesigner2 puts the secondary divider value into byte 14, bank 0 of the configuration data (see Appendix A). Note that clock dividers only have even values except for 1, so to get the actual divisor value multiply the configuration byte by 2 (a configuration byte of 0 means a divisor of 1). Finally it is necessary to know that auto-nulling is complete in 6 auto-null clocks and the auto-nulling done signal is asserted in 10 auto-null clocks.

So the formulae for auto-nulling delays are:

$$\text{Auto-null complete delay} = 6 \times 1000 \times \text{Div} / \text{Fmaster}$$

$$\text{Assert done signal delay} = 10 \times 1000 \times \text{Div} / \text{Fmaster}$$

Where Div is the auto-nulling secondary divider and Fmaster is the frequency of the master clock.

NOTE: the clock dividers actually take effect before the end of the configuration so if the delay is taken from the end of the configuration byte stream these delays will be slightly shorter.

2.3 Distribution of the offset voltages

The intrinsic offset voltages follow a Gaussian distribution, centred around a mean value. Ideally the mean is zero, but it is often a few mV away from that due to imperfections in the layout of differential circuitry. After auto-nulling, the offset voltage is within the given bounds, but the distribution is now a quantised and more rectangular. The two examples shown in figures 5 and 6 contain 10,000 OpAmp datapoints.

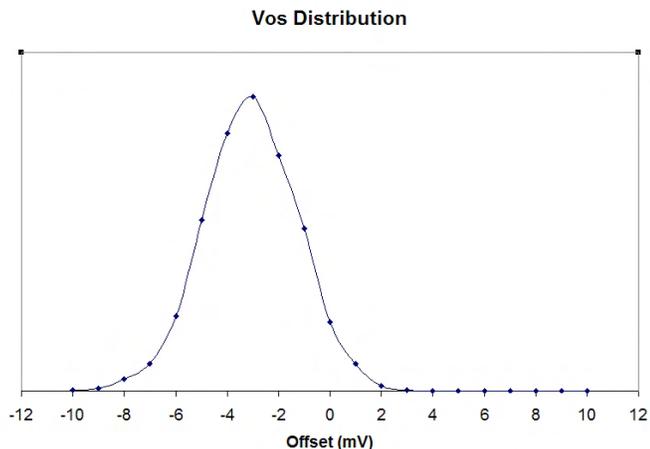


Figure 5. Intrinsic IO OpAmp offset

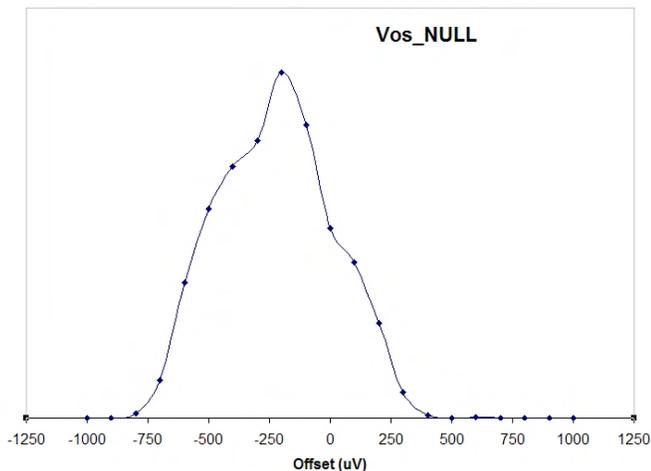


Figure 6. Post-autozero offset histogram

Note: Statistical measurement error biases the shape of this graph, especially at very low offset value.

Appendix A - Primary configuration

Here is an example of a configuration bit-stream used as a primary configuration and where offset correction of the OpAmp in IO1 is required:

```

Line Hex Binary Notes
00 00 00000000 Dummy byte
01 00 00000000 Dummy byte
02 00 00000000 Dummy byte
03 00 00000000 Dummy byte
04 00 00000000 Dummy byte
05 D5 11010101 Sync
06 B7 10110111 DIC byte
07 20 00100000 DIC byte
08 01 00000001 DIC byte
09 00 00000000 DIC byte
10 01 00000001 PRIMARY CONFIGURATION - Set Device ID = 1
11 C1 11000001 Control byte
12 CE 11001110 Start byte address = 14 - update2follow - Basic Error Check
13 00 00000000 RAMbank = 0
14 04 00000100 4 data bytes in Block #1
15 51 01010001 Bank:0 - ByteAddr:14 AZ-clock, secondary divider ratio = 162
16 00 00000000 Bank:0 - ByteAddr:15
17 01 00000001 Bank:0 - ByteAddr:16 Null_En_IO<0> (IO1)
18 F1 11110001 Bank:0 - ByteAddr:17 Power references and GWPRUP.
19 2A 00101010 End of Block #1
20 DF 11011111 Start byte address = 31 - update2follow - Basic Error Check
21 01 00000001 RAMbank = 1
22 01 00000001 1 data bytes in Block #2
23 FF 11111111 Bank:1 - ByteAddr:31 Alt-ID = 255 (default = 255)
24 2A 00101010 End of Block #2
25 91 10010001 Start byte address = 17 - NOupdate2follow - Basic Error Check
26 02 00000010 RAMbank = 2
27 03 00000011 3 data bytes in Block #3
28 F0 11110000 Bank:2 - ByteAddr:17 IO1: LocalP <-> IO_P LocalN <-> IO_N
29 82 10000010 Bank:2 - ByteAddr:18 IO1 Enable filter IO1 LPWR UP
30 01 00000001 Bank:2 - ByteAddr:19 IO1 HP_LP (OpAmp power mode)
31 2A 00101010 End of Block #3
32 00 00000000 Dummy byte

```

Particular point to notice are,

Line 15, the auto-null clock is divided down by a user setting of 162, with the fixed 1000 of the associated pre-scaler this gives a 98Hz auto-null clock for the demo board 16MHz clock. NOTE: AnadigmDesigner2 will calculate this divider ratio automatically.

Line 17, the IO1 OpAmp is selected for auto-null. Simply adding more bits to this byte and byte 15 would select other OpAmps for auto-null. NOTE: AnadigmDesigner2 will set these bits according to the user's circuit.

Line 28, the OpAmp has its inputs connected to the package pins (set by AnadigmDesigner2).

Lines 29 and 30, the OpAmp is powered up in high power mode. If the OpAmp is not powered up (in either high or low power modes) it will simply ignore the auto-null cycle (set by AnadigmDesigner2).

Appendix B – Auto-null Reconfiguration

Here is the short auto-null reconfiguration. This byte stream is to start up auto-nulling following a primary configuration or a reconfiguration in which auto-nulling was disabled. Note that this byte stream will have no effect if the preceding configuration had auto-nulling enabled. This byte stream is designed to turn auto-null back on after it was disabled, and so force a new auto-nulling sequence to be initiated.

Hex	Notes

D5	sync
01	ID
C1	Control byte
CB	start with byte 11 (another block to follow)
00	bank 0
01	1 byte to follow
40	enables auto-nulling for the chopper amplifier
2A	end of block
8F	start with byte 15 (last block)
00	bank 0
02	2 bytes to follow
FF	enables auto-nulling for all the CAB OpAmps
0F	enables auto-nulling for all the IO OpAmps
2A	end of block
00	dummy byte



<http://www.anadigm.com>

For more information Contact, Anadigm Technical Support
support@anadigm.com