AN5412

Increasing the DAC Resolution and Techniques to Reduce the DAC Error

Author: *Author: Naveen Raj*
Microchip Technology Inc.

Applications, such as instrumentation and waveform generators, demand higher resolution digital-to-analog converters (DACs). The increased resolution often comes with a higher cost due to testing expenses. This application note explores methods for enhancing DAC resolution by employing multiple DACs, voltage references and operational amplifiers. Additionally, users have the flexibility to enhance DAC performance through software adjustments for their specific applications. The implementation detailed in this application note offers a cost-effective solution for applications requiring a high number of DAC channels.

This document approaches three topics related to DAC applications:

1. Increasing the DAC resolution using multiple DACs
2. Increasing the DAC resolution and improving the DAC INL (integral nonlinearity) and DNL (differential nonlinearity) errors using multiple DACs
3. Improving the INL and DNL errors using multiple DACs

There are specific advantages for each solution discussed in these three topics:

- The Use of Existing Devices to Obtain a Higher Resolution DAC
- Flexibility to Adjust the Resolution According to the Preferences of the User for the Application
- Implementation of a Software Control Solution to Minimize Errors Associated While Using the DAC
- No Additional Testing Required as Existing Devices are Used to Generate Higher Resolution DACs
- No ADC Feedback Needed
- Software Solutions Target and Assist in Areas Where Errors are Prevalent
- The Increasing Number of Channels Offers a Significant Cost Advantage

1.0 IMPROVING THE DAC RESOLUTION USING MULTIPLE DACS AND ADDERS

To obtain one DAC with higher resolution in this context, it is necessary to incorporate two DACs, two voltage references and two operational amplifiers (or an alternative adder). The voltage reference used for DAC0 needs to be different from the voltage reference of DAC1. The desired higher resolution is achieved by summing up the outputs of these two DACs using the operational amplifier. V_{REF0} (voltage reference 0) is always lower than V_{REF1} (voltage reference 1) as DAC1 provides the most significant bits, while DAC0 provides the least significant bits for the higher resolution DAC. The correlation between V_{REF0} and V_{REF1} is addressed later in the implementation of the described hardware.

1.1 Example of 14-bit DAC using two 12-bit DACs

With a 5V reference and a 12-bit DAC, each step of the code is 5/4095 = 0.00122V. For a 14-bit DAC output resolution, each code is 5V/16383 = 0.0003V. As shown in Figure 2, this resolution is achieved by adding the sum of the two DAC outputs.

In this implementation, the DAC1 is configured to the 5V external $V_{REF,}$ and DAC0 is configured to the 1.25V external V_{REF}. Therefore, each step code for DAC1 and DAC0 is different.

For DAC0 each step of the code is 1.25V/4095 = 0.0003V, and for DAC1 each step of the code is 5/4095 = 0.00122V. Incrementing the DAC0 code with a pattern of 0,1,2,3,0,1,2,3 and DAC1 with 0,0,0,0,1,1,1,1,2,2,2,2 allows for achieving a linear code up to 16383 (see Table 1).

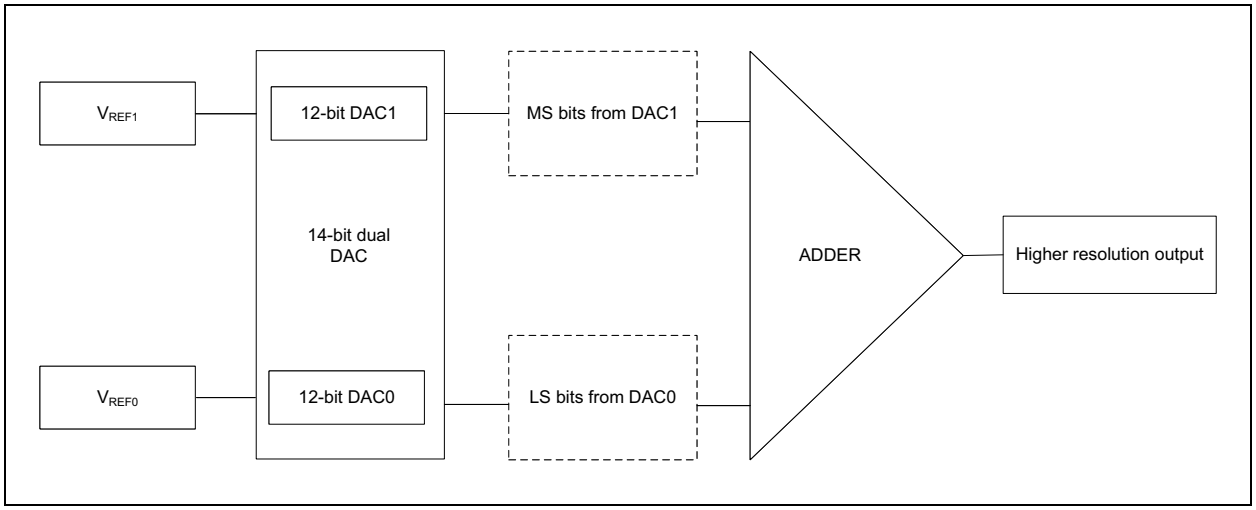

FIGURE 1: *Basic Block Diagram of the Higher Resolution DAC.*

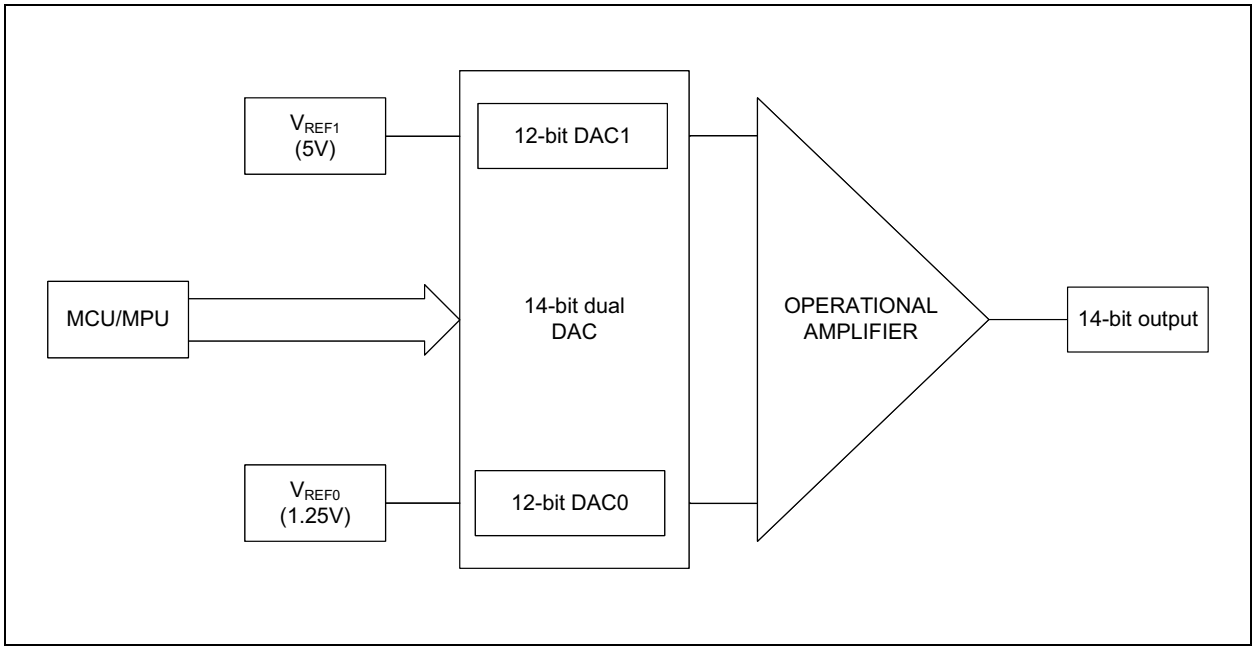

FIGURE 2: *Example of a 14-bit DAC implementation using two 12-bit DACs.*

TABLE 1: BASIC PATTERN OF INCREMENTING DAC0 AND DAC1 TO ACHIEVE A 14-BIT DAC FROM TWO 12-BIT DACS

Code	DAC1 code	DAC1 output	DAC0 code	DAC0 output	DAC1 + DAC0
0	0	0.00000	0	0.0000	0.0000
1	0	0.00000	1	0.0003	0.0003
2	0	0.00000	2	0.0006	0.0006
3	0	0.00000	3	0.0009	0.0009
4	1	0.00122	0	0.0000	0.0012
5	1	0.00122	1	0.0003	0.0015
6	1	0.00122	2	0.0006	0.0018
7	1	0.00122	3	0.0009	0.0021
8	2	0.00244	0	0.0000	0.0024
9	2	0.00244	1	0.0003	0.0028
10	2	0.00244	2	0.0006	0.0031
11	2	0.00244	3	0.0009	0.0034
...	...	...	...	...	...
16376	4094	4.998778999	0	0	4.998778999
16377	4094	4.998778999	1	0.00031	4.999088999
16378	4094	4.998778999	2	0.00061	4.999388999
16379	4094	4.998778999	3	0.00092	4.999698999
16380	4095	5	0	0	5
16381	4095	5	1	0.00031	5.00031
16382	4095	5	2	0.00061	5.00061
16383	4095	5	3	0.00092	5.00092

Table 1 illustrates the 16383 code pattern and Figure 3 shows DAC1 and DAC0 output.

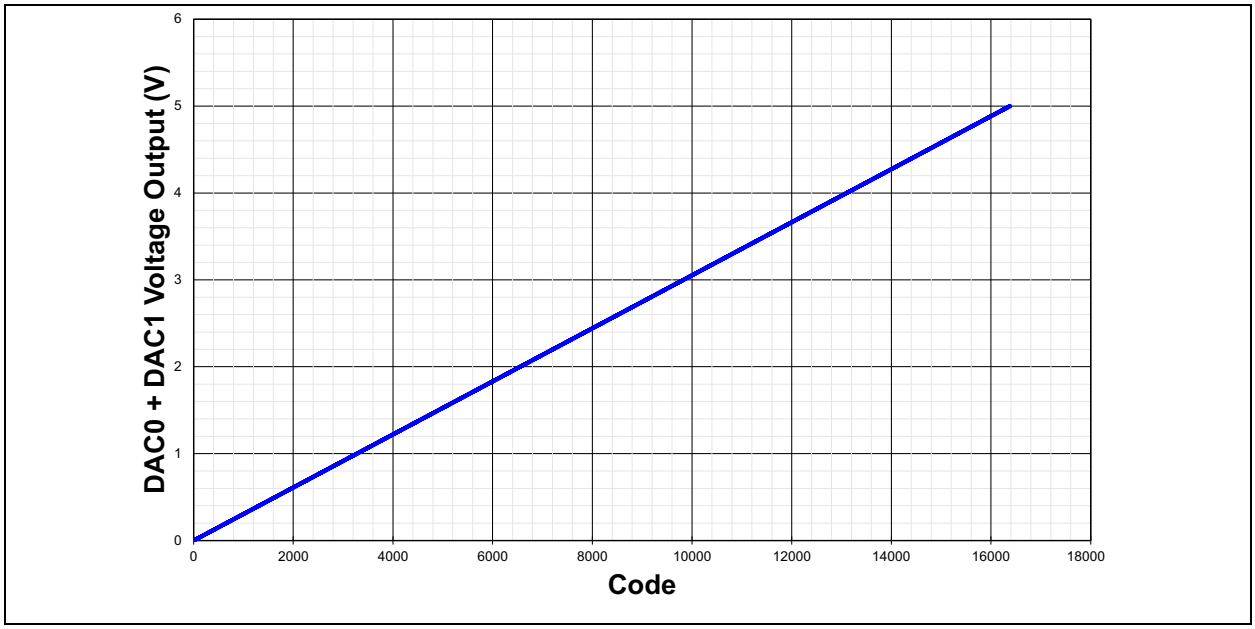

FIGURE 3: *14-bit Output of DAC1 + DAC0.*

1.2 SELECTION OF VOLTAGE REFERENCES

The selection of the voltage references determines the final resolutions that are achieved. The following references are available with the MCP1501/1502 family. Table 2 lists the combinations with the MCP1501/1502 and the resolution achieved using those voltage references.

TABLE 2: POSSIBLE COMBINATIONS OF MCP150X VARIANTS AND THE RESOLUTION ACHIEVED

Possible configuration with MCP150X variants	Possible DAC resolution
MCP150X (5V and 2.5V)	13-bits
MCP150X (5V and 1.25V)	14-bits
MCP150X (4.096V and 2.048V)	13-bits
MCP150X (4.096V and 1.024V)	14-bits
MCP150X (2.5V and 1.25V)	13-bits
MCP150X (4.096V and 1.048V)	13-bits

There is a certain relationship between V_{REF1} and V_{REF0}. Consider a 12-bit DAC operating at 5V, allowing for an output resolution of 5/4095 = 0.00122V. The applicable formula is $V_{REF0} = V_{REF1}/2^n$, where n = 1, 2, 3, 4 and so forth. Each incremental value of n enhances the DAC resolution by 1 bit. However, the achieved resolution is constrained by the availability of V_{REF0}.

Table 3 illustrates the number of bits achievable with two 12-bit DACs using V_{REF1} and V_{REF0}. For n = 1 ($V_{REF0} = V_{REF1}/2$, resulting in a 13-bit resolution) and n = 2 ($V_{REF0} = V_{REF1}/2$^n, resulting in a 14-bit resolution), the first column displays the relationship between V_{REF0} and V_{REF1}.

TABLE 3: THE RESOLUTION POSSIBILITIES OF A 12-BIT DAC USING EXTERNAL V_{REF} COMBINATIONS

V_{REF0}	Codes	Possible number of bits achieved by two 12-bit DACs						
		12-bit	13-bit	14-bit	15-bit	16-bit	17-bit	18-bit
	0	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000
	1	0.00122100	0.00061043	0.00030519	0.00015259	0.00007630	0.00003815	0.00001907
$V_{REF1}/2$	2	0.00244200	0.00122085	0.00061039	0.00030519	0.00015259	0.00007629	0.00003815
	3	0.00366300	0.00183128	0.00091558	0.00045778	0.00022889	0.00011444	0.00005722
$V_{REF1}/4$	4	0.00488400	0.00244170	0.00122078	0.00061037	0.00030518	0.00015259	0.00007629
	5	0.00610501	0.00305213	0.00152597	0.00076296	0.00038148	0.00019074	0.00009537
	6	0.00732601	0.00366256	0.00183117	0.00091556	0.00045777	0.00022888	0.00011444
	7	0.00854701	0.00427298	0.00213636	0.00106815	0.00053407	0.00026703	0.00013351
$V_{REF1}/8$	8	0.00976801	0.00488341	0.00244156	0.00122074	0.00061036	0.00030518	0.00015259
	9	0.01098901	0.00549383	0.00274675	0.00137333	0.00068666	0.00034333	0.00017166
	10	0.01221001	0.00610426	0.00305194	0.00152593	0.00076295	0.00038147	0.00019074
	11	0.01343101	0.00671469	0.00335714	0.00167852	0.00083925	0.00041962	0.00020981
	12	0.01465201	0.00732511	0.00366233	0.00183111	0.00091554	0.00045777	0.00022888
	13	0.01587302	0.00793554	0.00396753	0.00198370	0.00099184	0.00049591	0.00024796
	14	0.01709402	0.00854597	0.00427272	0.00213630	0.00106813	0.00053406	0.00026703
	15	0.01831502	0.00915639	0.00457792	0.00228889	0.00114443	0.00057221	0.00028610
$V_{REF1}/16$	16	0.01953602	0.00976682	0.00488311	0.00244148	0.00122072	0.00061036	0.00030518
	17	0.02075702	0.01037724	0.00518830	0.00259407	0.00129702	0.00064850	0.00032425
	18	0.02197802	0.01098767	0.00549350	0.00274667	0.00137331	0.00068665	0.00034332
	19	0.02319902	0.01159810	0.00579869	0.00289926	0.00144961	0.00072480	0.00036240
	20	0.02442002	0.01220852	0.00610389	0.00305185	0.00152590	0.00076295	0.00038147
	21	0.02564103	0.01281895	0.00640908	0.00320444	0.00160220	0.00080109	0.00040054
	22	0.02686203	0.01342937	0.00671428	0.00335704	0.00167849	0.00083924	0.00041962

V_{REF0}	Codes	Possible number of bits achieved by two 12-bit DACs						
	23	0.02808303	0.01403980	0.00701947	0.00350963	0.00175479	0.00087739	0.00043869
	24	0.02930403	0.01465023	0.00732467	0.00366222	0.00183108	0.00091553	0.00045777
	25	0.03052503	0.01526065	0.00762986	0.00381481	0.00190738	0.00095368	0.00047684
	26	0.03174603	0.01587108	0.00793505	0.00396741	0.00198367	0.00099183	0.00049591
	27	0.03296703	0.01648150	0.00824025	0.00412000	0.00205997	0.00102998	0.00051499
	28	0.03418803	0.01709193	0.00854544	0.00427259	0.00213626	0.00106812	0.00053406
	29	0.03540904	0.01770236	0.00885064	0.00442518	0.00221256	0.00110627	0.00055313
	30	0.03663004	0.01831278	0.00915583	0.00457778	0.00228885	0.00114442	0.00057221
	31	0.03785104	0.01892321	0.00946103	0.00473037	0.00236515	0.00118257	0.00059128
$V_{REF1}/32$	32	0.03907204	0.01953363	0.00976622	0.00488296	0.00244144	0.00122071	0.00061035
	33	...	...	...	...	...	...	...
	...	...	...	...	...	...	...	...
	...	...	...	...	...	...	...	...
	63	0.07692308	0.03845684	0.01922725	0.00961333	0.00480659	0.00240328	0.00120163
$V_{REF1}/64$	64	0.07814408	0.03906727	0.01953244	0.00976592	0.00488289	0.00244142	0.00122071

The same formula applies to increasing the resolution of any DAC, such as 8-bit or 10-bit. Table 4 demonstrates the method for enhancing the resolution of 8-bit DACs using two 8-bit DACs and two voltage references, along with detailing the relationship between V_{REF1} and V_{REF0} (calculations based on a 5V V_{REF1} and a 8-bit resolution).

TABLE 4: RESOLUTION RELATION WITH 8-BIT DAC AND EXTERNAL VREFS

VREF0	Codes	8-bit	9-bit	10-bit	12-bit	13-bit
	0	0	0	0	0	0
	1	0.019531	0.009766	0.004883	0.002441	0.001221
$V_{REF1}/2$	2	0.039063	0.019531	0.009766	0.004883	0.002441
	3	0.058594	0.029297	0.014648	0.007324	0.003662
$V_{REF1}/4$	4	0.078125	0.039063	0.019531	0.009766	0.004883
	5	0.097656	0.048828	0.024414	0.012207	0.006104
	6	0.117188	0.058594	0.029297	0.014648	0.007324
	7	0.136719	0.068359	0.03418	0.01709	0.008545
$V_{REF1}/8$	8	0.15625	0.078125	0.039063	0.019531	0.009766
	9	0.175781	0.087891	0.043945	0.021973	0.010986
	10	0.195313	0.097656	0.048828	0.024414	0.012207
	11	0.214844	0.107422	0.053711	0.026855	0.013428
	12	0.234375	0.117188	0.058594	0.029297	0.014648
	13	0.253906	0.126953	0.063477	0.031738	0.015869
	14	0.273438	0.136719	0.068359	0.03418	0.01709
	15	0.292969	0.146484	0.073242	0.036621	0.018311
$V_{REF1}/16$	16	0.3125	0.15625	0.078125	0.039063	0.019531

1.3 DATA COLLECTED FOR THE 12-BIT DACs

The design implements a 14-bit DAC by using two 12-bit DACs and an operational amplifier as the adder. Figure 4 illustrates the block diagram of the board used to collect data. Data was collected for all 16383 code implementations following the increment pattern outlined in Table 1. DAC0 is incremented with the pattern 0, 1, 2, 3, 0, 1, 2, 3..., while DAC1 follows the pattern 0, 0, 0, 0, 1, 1, 1, 1, 2, 2, 2, 2. This approach allows for achieving a linear code up to 16383.

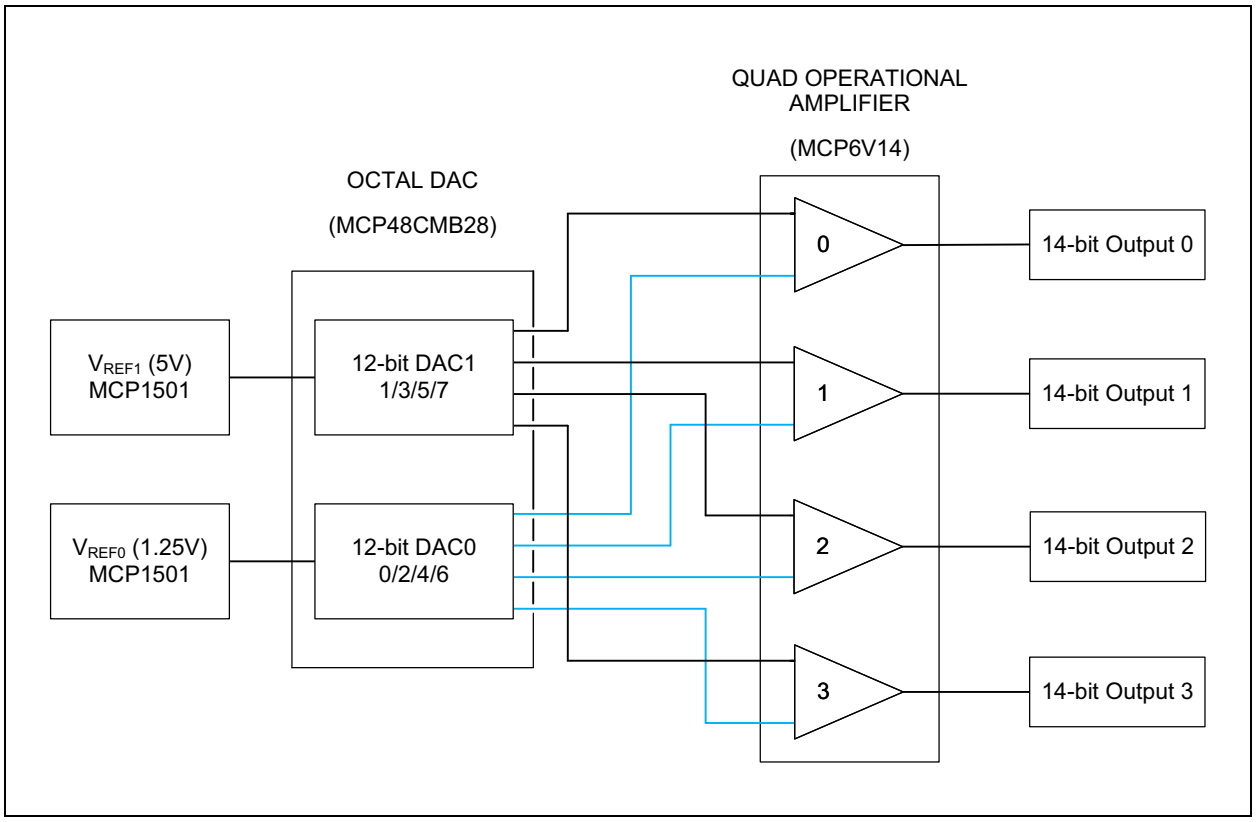

FIGURE 4: *Block Diagram of the Implemented Board for Data Collection.*

The schematic and data for a 14-bit system are depicted in Figure 5. The on-board DAC is an octal DAC (MCP48CMB28), and the voltage references employed are MCP1501. The operational amplifier used is MCP6V14. Note that resistor tolerances should be less than 1% to ensure optimal results.

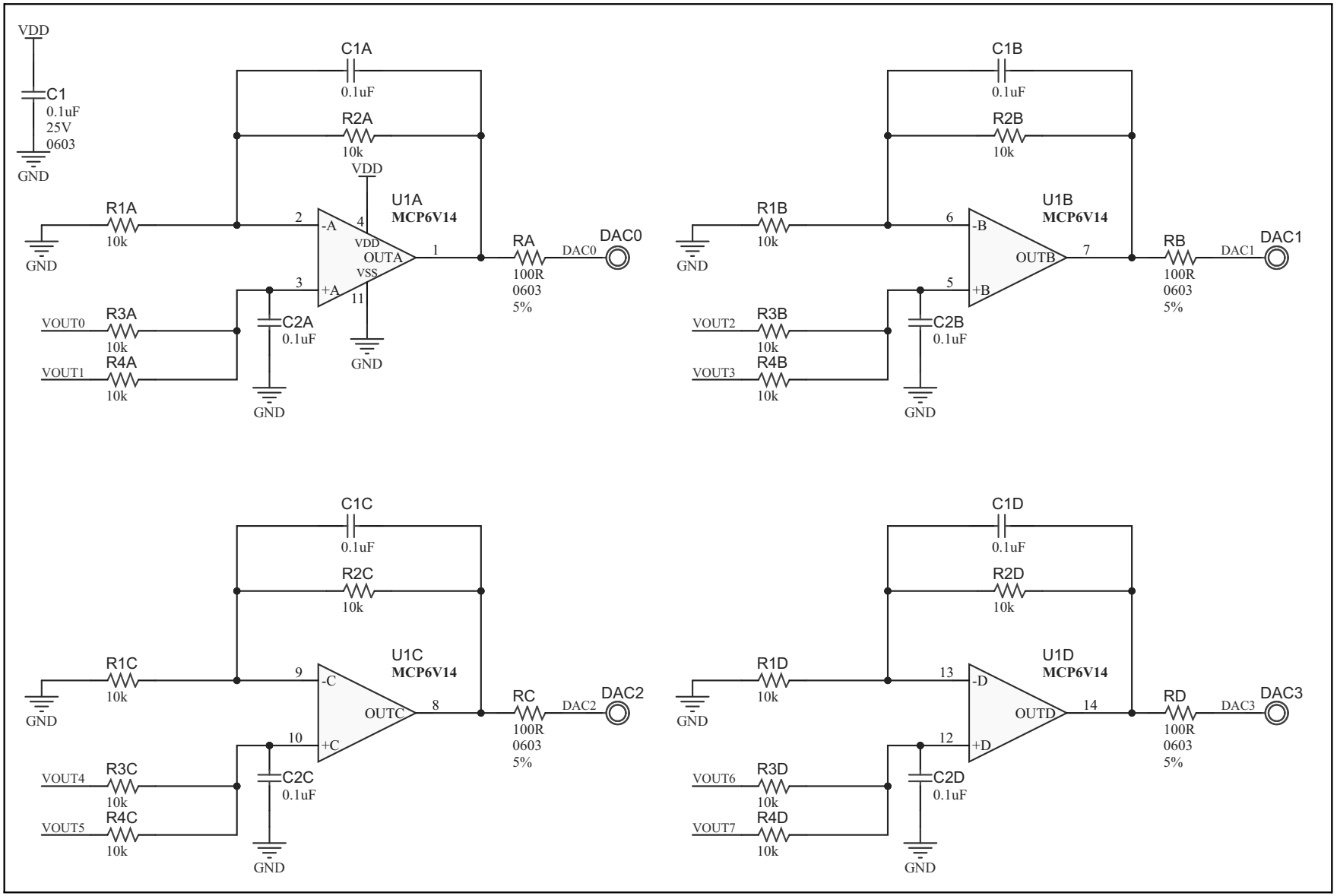

FIGURE 5: *MCP6V14 Operational Amplifier Circuit Schematic.*

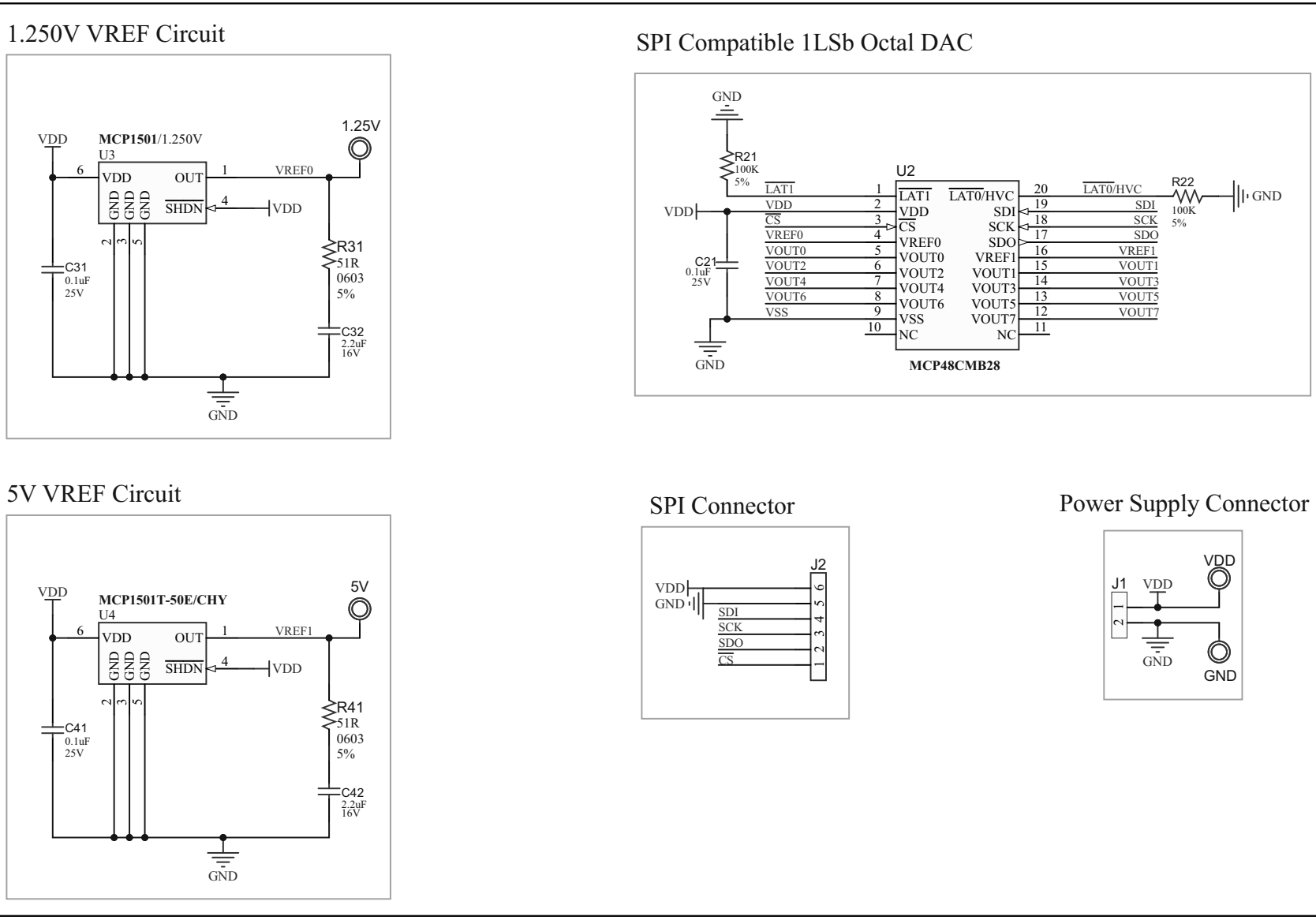

FIGURE 6: *14-bit DAC SCHEMATIC – V_{REF} Circuits, Program Connector, Power Supply Connector, MCP48CMB28 DAC.*

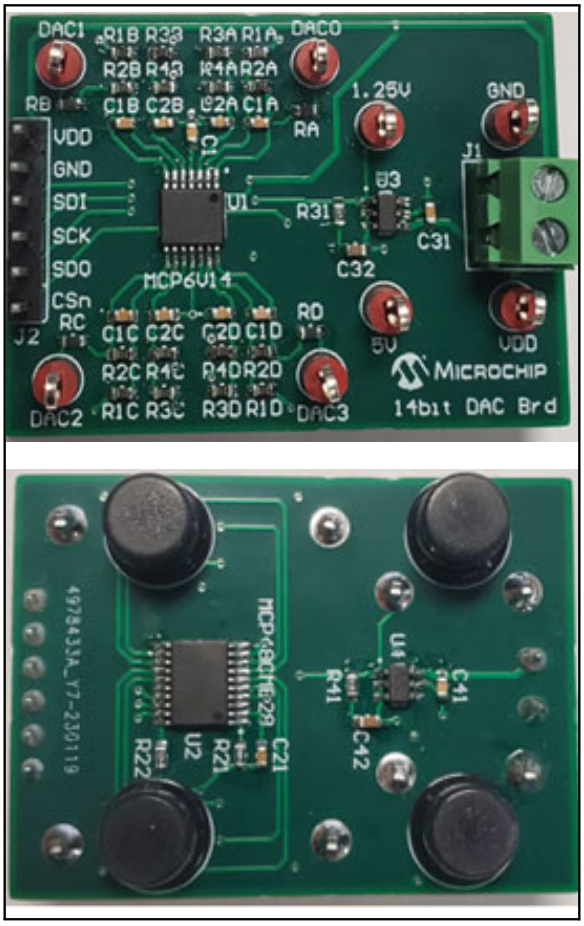

FIGURE 7: *Implemented board from which the data was taken.*

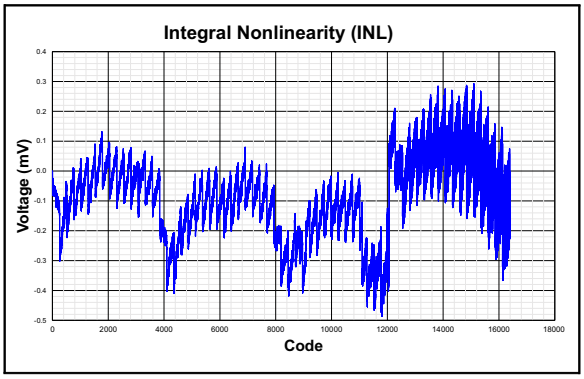

FIGURE 8: *INL 14-bit DAC output.*

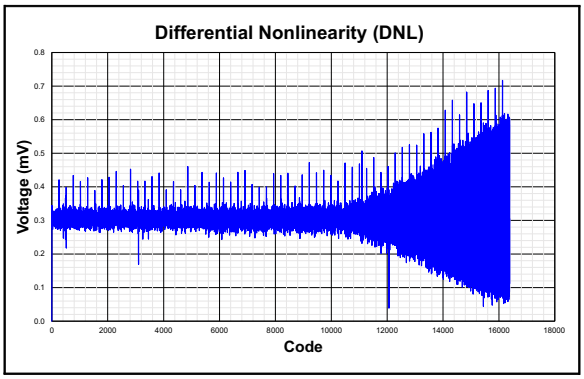

FIGURE 9: *DNL 14-bit DAC output.*

EXAMPLE 1: SAMPLE CODE FOR 14-BIT DAC USING TWO 12-BIT DACS.

```
// Generic code to increment and write 14-bit data using two 12-bit DACs (PIC16F1716)

void SPI_Initialize (void);
void SPI1_WriteDACWord (unsigned char addr,unsigned int data);
void SPI1_WriteDAC14 (unsigned long int data);
#define CS0 PORTCbits.RC6
#define CS0_TRIS TRISCbits.TRISC6
#define CS0_DIGITAL ANSELCbits.ANSC6
#define HIGH    1
#define LOW     0
#define DAC0    0
#define DAC1    1
unsigned int low, high, DAC14data;

main()
{
    while(1)
    {
        SPI_Write (DAC14data);
    }
}

// Function to write 14-bit data

void SPI1_WriteDAC14 (unsigned long int data14)
{
    low = data14 & 0x0003;
    high = data14 >> 2;
    SPI1_WriteDACWord (DAC1,high);
    SPI1_WriteDACWord (DAC0,low);
}

//Function to Write to SPI1 Module

void SPI1_WriteDACWord (unsigned char addr,unsigned int data)
{
    CS0 = LOW;
    SSP1BUF = (addr << 3) & 0xF8;     //Shift and & write command AD4:AD3:AD2:AD1:AD0:0:0:X
    while(!PIR1bits.SSP1IF);          //Wait for the interrupt
    PIR1bits.SSP1IF = 0;              //Clear the interrupt flag

    SSP1BUF = (data >> 8) & 0x00FF;       // Shift the data to transmit the MS byte
    while(!PIR1bits.SSP1IF);
    PIR1bits.SSP1IF = 0;

    SSP1BUF = (data) & 0x00FF;      //transmit the LS byte
    while(!PIR1bits.SSP1IF);
    PIR1bits.SSP1IF = 0;
    CS0 = HIGH;
}
```

```
void SPI_Initialize (void)

{
    CS0_TRIS = 0;
    CS0 = 1;
    CS0_DIGITAL = 0;
    SSPCLKPPS = 19;
    SSPDATPPS = 20;
    RC3PPS = 16;
    RC5PPS = 17;

    //SPI setup

    SSP1STAT = 0x40;
    SSP1CON1 = 0x00;
    SSP1ADD = 0x27;
    TRISCbits.TRISC3 = 0;
    SSP1CON1bits.SSPEN = 1;
}
```

The MCP48CMB28 offers an advantage in the configuration of external references, as illustrated in Figure 10. Specifically, external reference Pin 5 is designated for even channels (DAC0, DAC2, DAC4, DAC6), while external reference Pin 16 is configured for odd channels (DAC1, DAC3, DAC5, DAC7). This design facilitates the use of MCP1501/1502/1.25V (V_{REF0}) for even channels and MCP1501/1502/5V (V_{REF1}) for odd channels. Consequently, this configuration reduces costs by enabling the usage of the same references for multiple channels.

The cost advantage becomes more noticeable as the number of channels increases. This is because the same V_{REF0} and V_{REF1} (MCP1501/1502) can be efficiently employed across multiple DACs, allowing for the creation of additional DAC channels without the need for extra voltage references. The voltage output capacity of MCP1501/1502 is 20 mA. Considering the input V_{REF} requirements for the external V_{REF} pin at 354.1uA, a single MCP1501/1502 with an output capacity of 20 mA can drive a total of 20mA/0.354mA = 56 DACs.

Since each octal DAC provides 4 channels for a 14-bit DAC configuration, theoretically, the application generates approximately 56 * 4 = 224 channels, using the same external voltage references. This demonstrates the scalability and efficiency of the setup in accommodating a higher number of DAC channels within the given specifications.

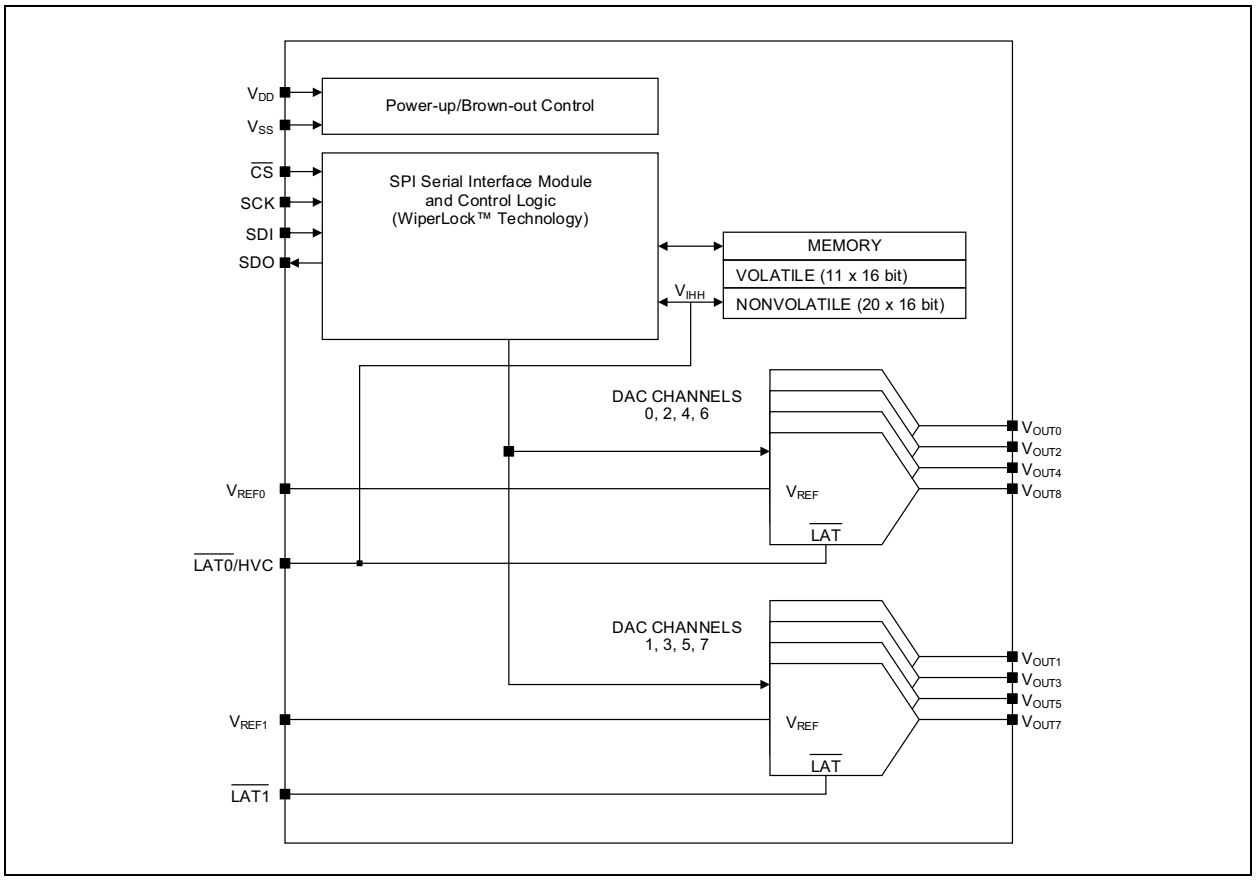

FIGURE 10: *MCP48CMB28 Family Block Diagram.*

Figure 11 illustrates the configuration of multiple MCP48CMB28 DACs connected to a single external voltage reference (MCP1501/1502), achieving around 220 channels of 14-bit DAC.

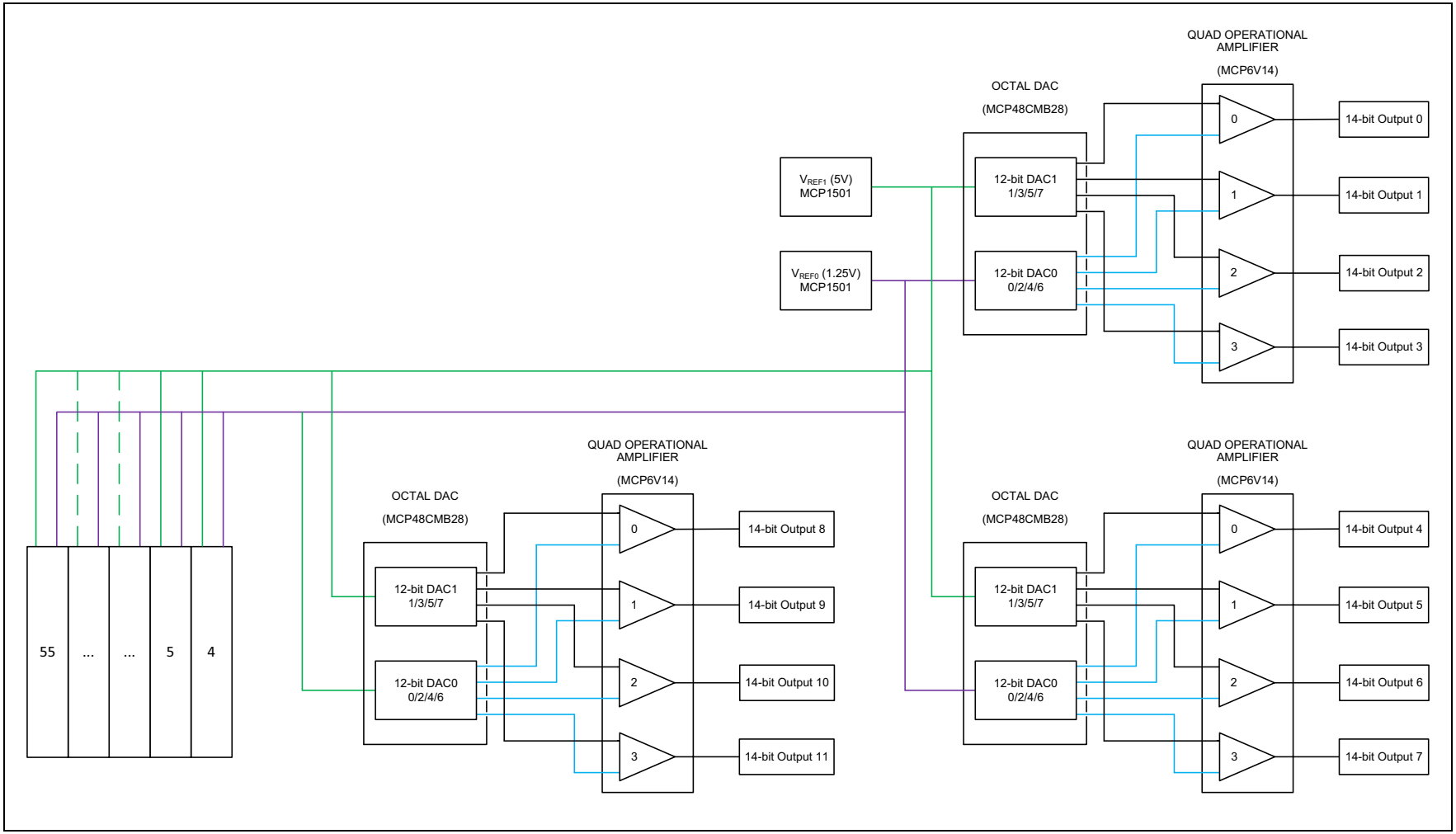

FIGURE 11: *Configuration for Implementing 220 Channel 14-bit DAC using the same MCP1501/1502 Reference.*

2.0 INCREASING THE DAC RESOLUTION AND IMPROVING THE DAC INL AND DNL ERRORS USING MULTIPLE DACS

To increase the accuracy of the DAC with the discussed 14-bit DAC implementation, the first step is to identify the codes where the errors are the highest.

The implemented schematic produces a 14-bit DAC output. However, observed INL and DNL errors differ from the MCP48CMB28, as depicted in Figure 12. This section explores a software modification technique to reduce INL and DNL errors based on the discussed example code. Start by pinpointing codes with the highest errors and, subsequently, adapt the software within the identified error windows.

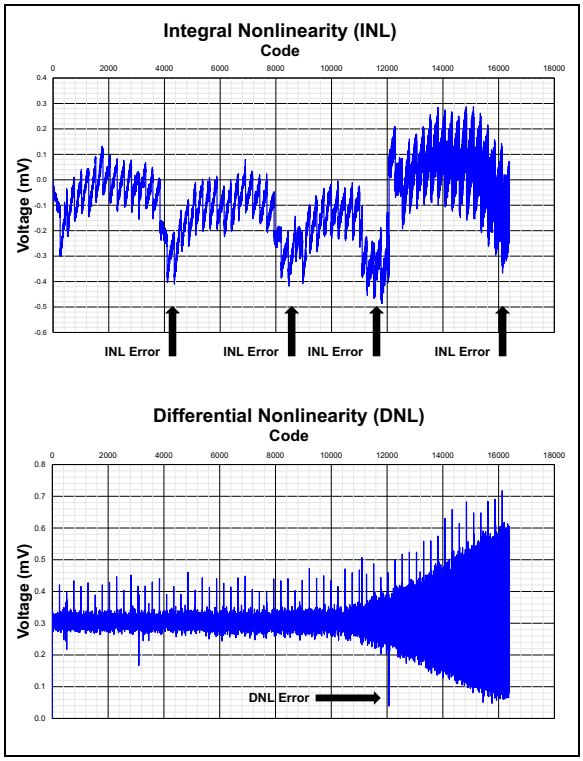

FIGURE 12: *INL and DNL Error Points.*

Figure 12 indicates a rise in DNL beyond code 12000. Notably, INL error peaks at four points: DAC1 codes 1023, 2046, 3069 and 4095 (or for 14-bit code of 4092, 8184, 12276, 16383).

The DAC increment pattern, as detailed in Table 1, aligns with Table 5. The peaks in INL errors correspond to specific DAC1 codes. The strategy is to prevent DAC1 operation in these error-prone zones. To achieve this, halt DAC1 just before reaching codes 1023, 2046, 3069 and 4095. Instead, use DAC0 in those codes and extend DAC0 beyond 0, 1, 2 and 3.

For instance, halt DAC1 for 250 codes in those intervals, limiting its operation from 0 to 1000 (250 * 4). By stopping DAC0 at 1000, it ensures that DAC0 won't operate at 1023.

For example, DAC1 can pause 100 codes before reaching 1023, resume 150 codes after 1023 and follow a similar pattern for codes 2046 and 4095, with the exception of 4095, the pause is 250 codes before reaching it. Table 5 illustrates where codes are paused and restarted.

TABLE 5: 14-BIT CODE OPERATING REGION OF PEAK ERROR AND THE DAC1 AND DAC0 CODES

14-bit code (DAC1+DAC0)	3695–4696	7787–8788	11879–12880	15383
DAC0 Operation in the Region	0–999	0–999	0–999	0–999
DAC1 Code Stops	923	1946	2969	3845
DAC1 Code Restart	1174	2197	3220	

In Table 6, the complete code is displayed, and the areas where DAC1 stops and DAC0 increments continuously are highlighted in gray.

TABLE 6: INCREMENT PATTERN AND ERROR LOCATIONS

Code	DAC1 code	DAC1 output	DAC0 code	DAC0 output	DAC1 +DAC0
0	0	0.0000	0	0.0000	0.0000
1	0	0.0000	1	0.0003	0.0003
2	0	0.0000	2	0.0006	0.0006
3	0	0.0000	3	0.0009	0.0009
4	1	0.0012	0	0.0000	0.0012
5	1	0.0012	1	0.0003	0.0015
6	1	0.0012	2	0.0006	0.0018
7	1	0.0012	3	0.0009	0.0021
	...	...	...	...	...
	...	...	...	...	...
	...	...	...	...	...
3692	923	1.1270	0	0.0000	1.1270
3693	923	1.1270	1	0.0003	1.1273
3694	923	1.1270	2	0.0006	1.1276
3695	923	1.1270	3	0.0009	1.1279
3696	924	1.1282	0	0.0000	1.1282
3697	924	1.1282	1	0.0003	1.1285
3698	924	1.1282	2	0.0006	1.1288
3699	924	1.1282	3	0.0009	1.1291
3700	924	1.1282	4	0.0012	1.1294
	...	...	...	...	...
	...	...	...	...	...
	...	...	...	...	...
4692	924	1.1282	996	0.3040	1.4322
4693	924	1.1282	997	0.3043	1.4325
4694	924	1.1282	998	0.3046	1.4328
4695	924	1.1282	999	0.3049	1.4332
4696	1174	1.4335	0	0.0000	1.4335
4697	1174	1.4335	1	0.0003	1.4338
4698	1174	1.4335	2	0.0006	1.4341
4699	1174	1.4335	3	0.0009	1.4344
	...	...	...	...	...
	...	...	...	...	...
	...	...	...	...	...
7784	1946	2.3761	0	0.0000	2.3761
7785	1946	2.3761	1	0.0003	2.3764
7786	1946	2.3761	2	0.0006	2.3767
7787	1946	2.3761	3	0.0009	2.3770
7788	1947	2.3773	0	0.0000	2.3773
7789	1947	2.3773	1	0.0003	2.3776
7790	1947	2.3773	2	0.0006	2.3779

TABLE 6: INCREMENT PATTERN AND ERROR LOCATIONS

Code	DAC1 code	DAC1 output	DAC0 code	DAC0 output	DAC1 +DAC0
7791	1947	2.3773	3	0.0009	2.3782
7792	1947	2.3773	4	0.0012	2.3785
	...	...	...	...	...
	...	...	...	...	...
	...	...	...	...	...
8784	1947	2.3773	996	0.3040	2.6813
8785	1947	2.3773	997	0.3043	2.6816
8786	1947	2.3773	998	0.3046	2.6819
8787	1947	2.3773	999	0.3049	2.6822
8787	2197	2.6825	0	0.0000	2.6825
8787	2197	2.6825	1	0.0003	2.6828
8787	2197	2.6825	2	0.0006	2.6831
8787	2197	2.6825	3	0.0009	2.6835
	...	...	...	...	...
	...	...	...	...	...
	...	...	...	...	...
7784	2969	3.6252	0	0.0000	3.6252
7785	2969	3.6252	1	0.0003	3.6255
7786	2969	3.6252	2	0.0006	3.6258
7787	2969	3.6252	3	0.0009	3.6261
11880	2970	3.6264	0	0.0000	3.6264
11881	2970	3.6264	1	0.0003	3.6267
11882	2970	3.6264	2	0.0006	3.6270
11883	2970	3.6264	3	0.0009	3.6273
11884	2970	3.6264	4	0.0012	3.6276
	...	...	...	...	...
	...	...	...	...	...
	...	...	...	...	...
12876	2970	3.6264	996	0.3040	3.9304
12877	2970	3.6264	997	0.3043	3.9307
12878	2970	3.6264	998	0.3046	3.9310
12879	2970	3.6264	999	0.3049	3.9313
12880	3220	3.9316	0	0.0000	3.9316
12881	3220	3.9316	1	0.0003	3.9319
12882	3220	3.9316	2	0.0006	3.9322
12883	3220	3.9316	3	0.0009	3.9325
	...	...	...	...	...
	...	...	...	...	...
	...	...	...	...	...
	...	...	...	...	...
7784	3845	4.6947	0	0.0000	4.6947
7785	3845	4.6947	1	0.0003	4.6951
7786	3845	4.6947	2	0.0006	4.6954
7787	3845	4.6947	3	0.0009	4.6957

TABLE 6: INCREMENT PATTERN AND ERROR LOCATIONS

Code	DAC1 code	DAC1 output	DAC0 code	DAC0 output	DAC1 +DAC0
15384	3846	4.6960	0	0.0000	4.6960
15385	3846	4.6960	1	0.0003	4.6963
15386	3846	4.6960	2	0.0006	4.6966
15387	3846	4.6960	3	0.0009	4.6969
15388	3846	4.6960	4	0.0012	4.6972
	...	...	...	...	...
	...	...	...	...	...
	...	...	...	...	...
16380	3846	4.6960	996	0.3040	5.0000
16381	3846	4.6960	997	0.3043	5.0003
16382	3846	4.6960	998	0.3046	5.0006
16383	3846	4.6960	999	0.3049	5.0009

The software for implementing the pattern is nearly identical to Example 1 with the only change occurring in the `SP1_WriteDACWord()` function. Rather than writing the entire code, four key error points are addressed. In these instances, DAC0 undergoes a linear increment from 0 to 1000. The adjustment in the `SP1_WriteDACWord()` function aligns with the structure outlined in Table 6 and is illustrated in Example 2.

EXAMPLE 2: SAMPLE CODE FOR 14-BIT DAC WITH ERROR COMPENSATION.

```
void SPI1_WriteDAC14 (unsigned long int data14)
{
low = data14 & 0x0003;
high = data14 >> 2;
if(data14 > 3695 && data14 < 4696)
    {
low = data14-3696;      // LSb DAC1 increments @3696 from 0-999
SPI1_WriteDACWord (DAC1,924 );
SPI1_WriteDACWord (DAC0,low );
    }
    else if(data14 > 7787 && data14 < 8788)
    {
    low = data14-7788;          // LSb DAC1 increments @ 7788 from 0-999
  SPI1_WriteDACWord (DAC1,1947);
    SPI1_WriteDACWord (DAC0,low);
    }
    else if(data14 > 11879 && data14 < 12880)
    {
    low = data14-11880;          // LSb DAC1 increments @ 11880 from 0-999
  SPI1_WriteDACWord (DAC1,2970);
    SPI1_WriteDACWord (DAC0,low);
    }
    else if(data14 > 15383)
    {
    low = data14-15384;          // LSb DAC1 increments @ 15384 from 0-999
  SPI1_WriteDACWord (DAC1,3846);
    SPI1_WriteDACWord (DAC0,low);
    }
    else
    {
    SPI1_WriteDACWord (DAC1,high);
    SPI1_WriteDACWord (DAC0,low);
   }
}
```

Implementing error compensation in the software, as demonstrated in Example 2, leads to a reduction in INL and DNL. Figure 13 provides a comparison of DNL and INL errors, showcasing the impact of error compensation.

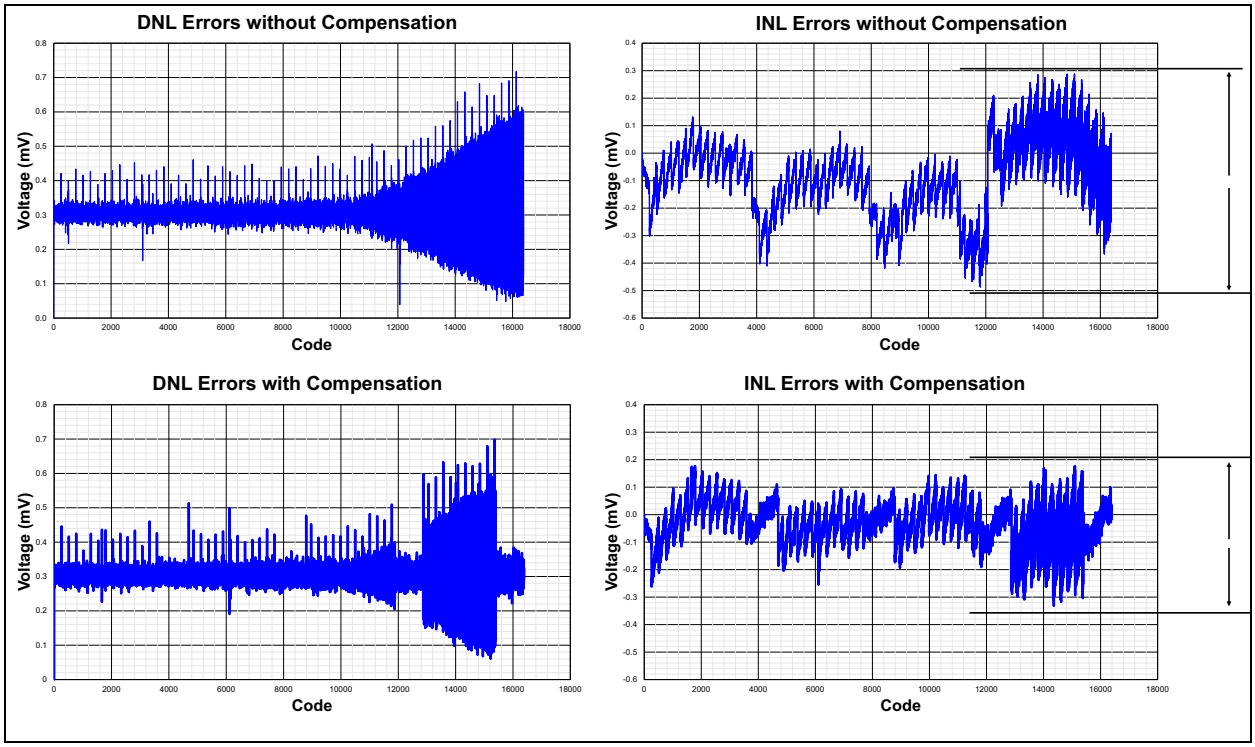

FIGURE 13: *DNL and INL without and with error compensation.*

3.0 IMPROVING THE INL AND DNL ERRORS USING MULTIPLE DACS

In the previous section, the focus was on enhancing DAC resolution and to improve INL and DNL errors. This section, however, explores techniques to enhance INL and DNL errors using multiple DACs without increasing the resolution. The key distinction lies in situations where resolution augmentation is unnecessary. In such cases, there is no requirement to supply separate voltage references for DAC1 and DAC0, as depicted in Figure 14. The voltage references for both DACs can be the same, sourced externally or internally based on DAC features.

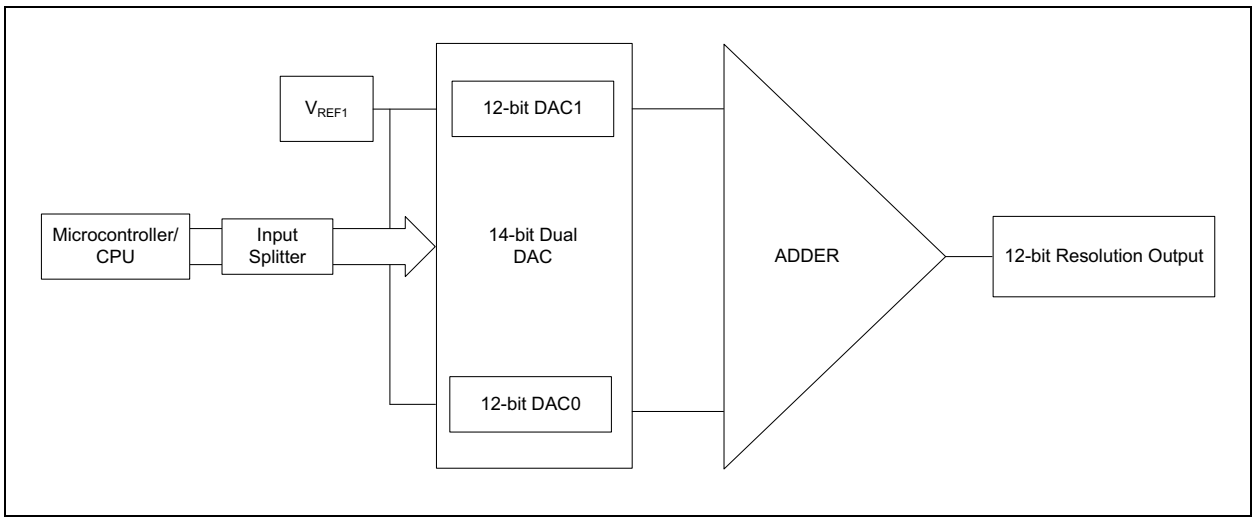

FIGURE 14: *Block Diagram of Error Improvement without Increasing the Resolution.*

Advantages of the discussed implementation include:

- **Cost efficiency**:
  - Lower accuracy devices can be employed to achieve higher accuracy, reducing the overall implementation cost.
- **Versatility in implementation**:
  - The approach is adaptable for implementation in both software and hardware.
- **User flexibility**:
  - With the software option, users have the flexibility to increase accuracy according to their system requirements.

The implementation uses any adder, such as an operational amplifier adder, as illustrated in Figure 15.

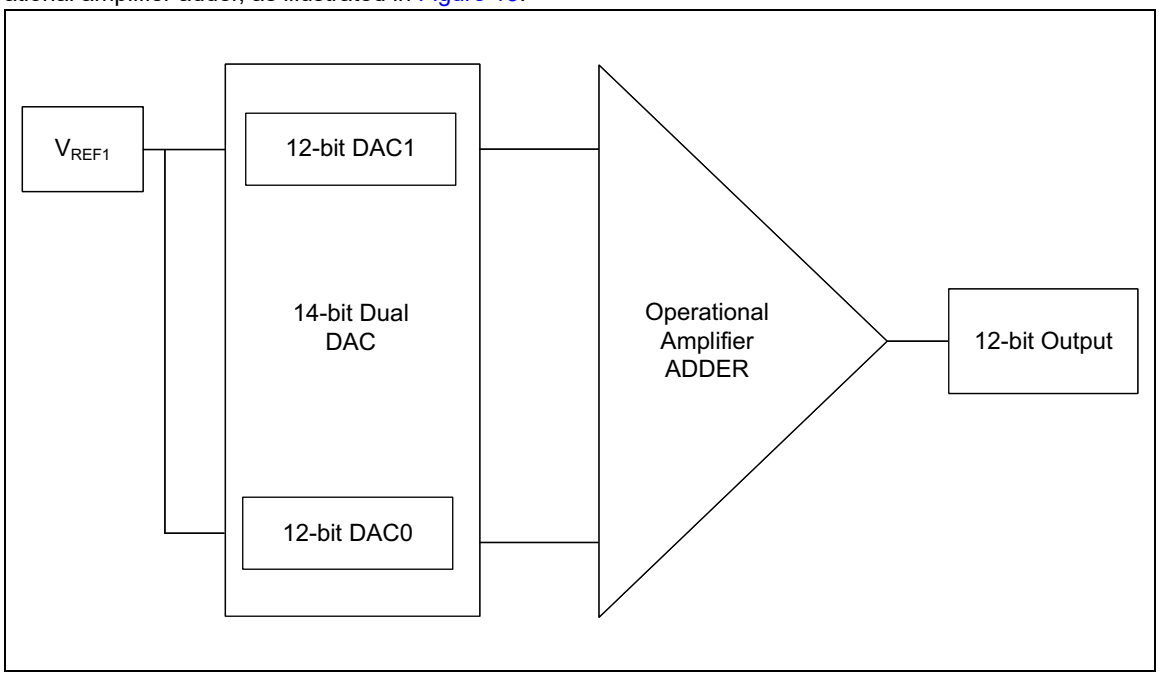

FIGURE 15: *Block Diagram for Adding the two DACs.*

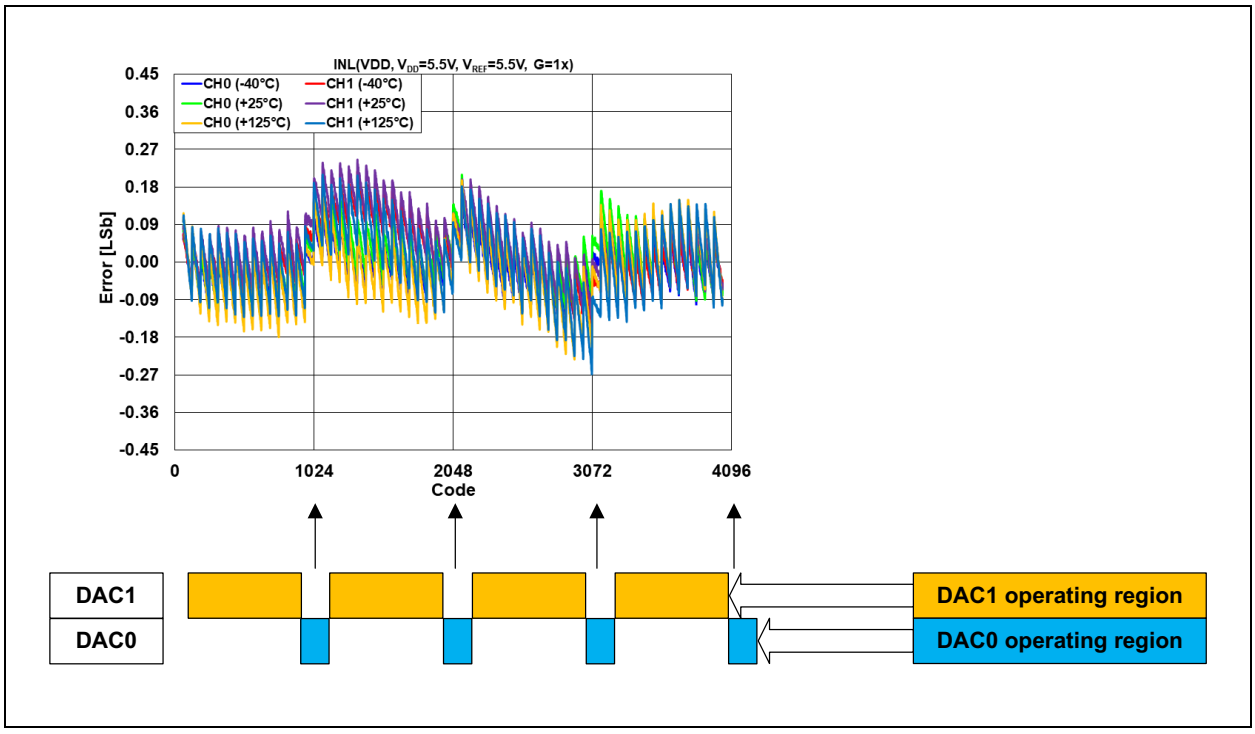

FIGURE 16: *Visual Representation of the DAC Operation to Reduce the INL Error.*

In Figure 17, the highest errors occur at codes 1023, 2046, 4069 and 4095. To address this, let DAC1 stop 100 codes before and after these points.

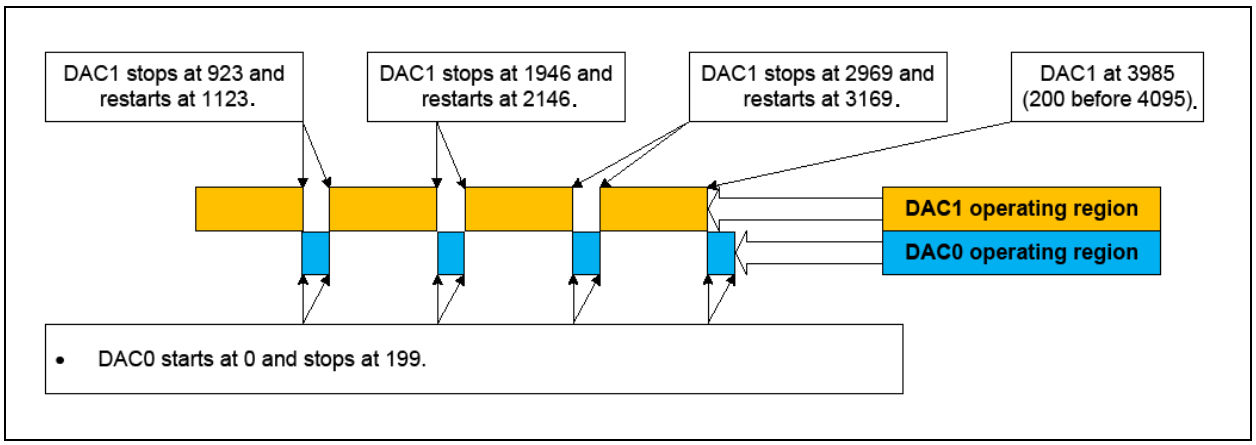

FIGURE 17: *Starting and Stopping Codes for DAC1 and DAC0.*

The code to start and stop DAC0/DAC1 resembles the Example 2 code. Figure 18 shows the implementation of DAC0 and DAC1 codes along with corresponding graphical representations.

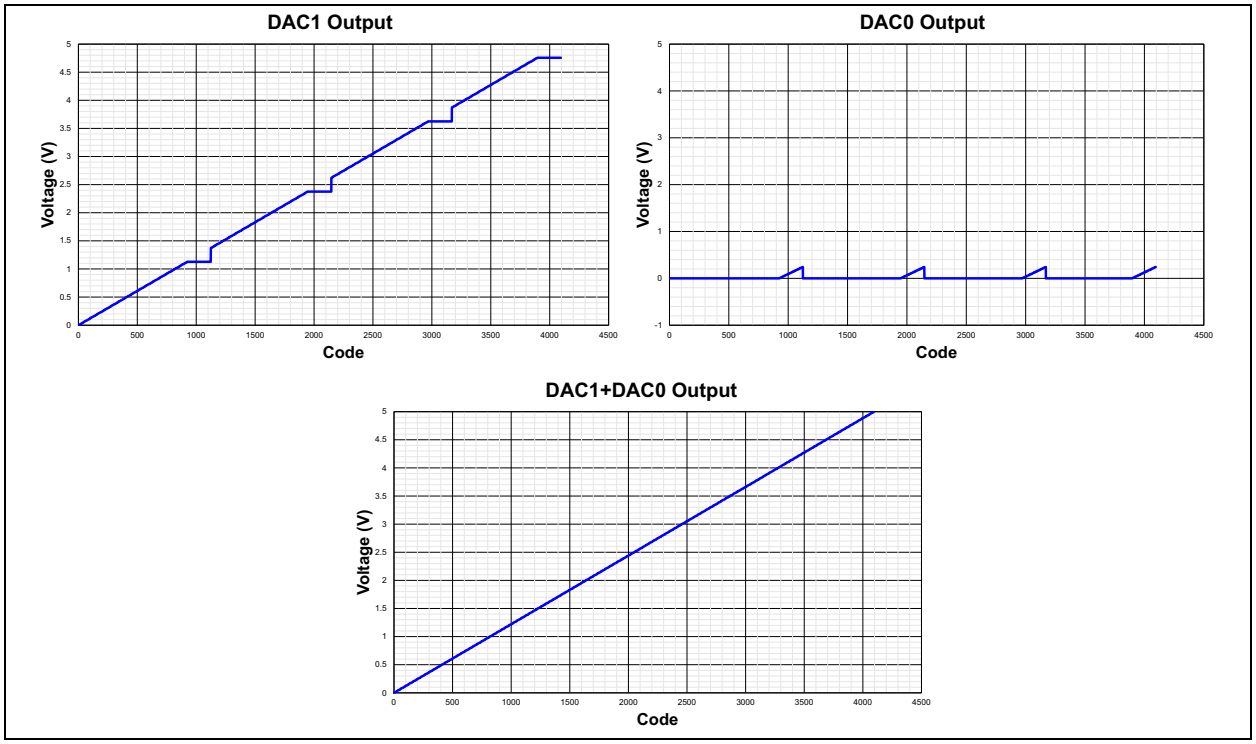

FIGURE 18: *Graphical representation for the codes of DAC1 AND DAC0.*

Note the following details of the code protection feature on Microchip products:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications, and under normal conditions.
- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip product is strictly prohibited and may violate the Digital Millennium Copyright Act.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is "unbreakable" Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.

Trademarks

ISBN: 978-1-6683-4388-3

For information regarding Microchip's Quality Management Systems, please visit www.microchip.com/quality.

Worldwide Sales and Service

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://www.microchip.com/support
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Austin, TX
Tel: 512-257-3370

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Novi, MI
Tel: 248-848-4000

Houston, TX
Tel: 281-894-5983

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453
Tel: 317-536-2380

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608
Tel: 951-273-7800

Raleigh, NC
Tel: 919-844-7510

New York, NY
Tel: 631-435-6000

San Jose, CA
Tel: 408-735-9110
Tel: 408-436-4270

Canada - Toronto
Tel: 905-695-1980
Fax: 905-695-2078

ASIA/PACIFIC

Australia - Sydney
Tel: 61-2-9868-6733

China - Beijing
Tel: 86-10-8569-7000

China - Chengdu
Tel: 86-28-8665-5511

China - Chongqing
Tel: 86-23-8980-9588

China - Dongguan
Tel: 86-769-8702-9880

China - Guangzhou
Tel: 86-20-8755-8029

China - Hangzhou
Tel: 86-571-8792-8115

China - Hong Kong SAR
Tel: 852-2943-5100

China - Nanjing
Tel: 86-25-8473-2460

China - Qingdao
Tel: 86-532-8502-7355

China - Shanghai
Tel: 86-21-3326-8000

China - Shenyang
Tel: 86-24-2334-2829

China - Shenzhen
Tel: 86-755-8864-2200

China - Suzhou
Tel: 86-186-6233-1526

China - Wuhan
Tel: 86-27-5980-5300

China - Xian
Tel: 86-29-8833-7252

China - Xiamen
Tel: 86-592-2388138

China - Zhuhai
Tel: 86-756-3210040

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-3090-4444

India - New Delhi
Tel: 91-11-4160-8631

India - Pune
Tel: 91-20-4121-0141

Japan - Osaka
Tel: 81-6-6152-7160

Japan - Tokyo
Tel: 81-3-6880- 3770

Korea - Daegu
Tel: 82-53-744-4301

Korea - Seoul
Tel: 82-2-554-7200

Malaysia - Kuala Lumpur
Tel: 60-3-7651-7906

Malaysia - Penang
Tel: 60-4-227-8870

Philippines - Manila
Tel: 63-2-634-9065

Singapore
Tel: 65-6334-8870

Taiwan - Hsin Chu
Tel: 886-3-577-8366

Taiwan - Kaohsiung
Tel: 886-7-213-7830

Taiwan - Taipei
Tel: 886-2-2508-8600

Thailand - Bangkok
Tel: 66-2-694-1351

Vietnam - Ho Chi Minh
Tel: 84-28-5448-2100

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4485-5910
Fax: 45-4485-2829

Finland - Espoo
Tel: 358-9-4520-820

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Garching
Tel: 49-8931-9700

Germany - Haan
Tel: 49-2129-3766400

Germany - Heilbronn
Tel: 49-7131-72400

Germany - Karlsruhe
Tel: 49-721-625370

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Germany - Rosenheim
Tel: 49-8031-354-560

Israel - Hod Hasharon
Tel: 972-9-775-5100

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Italy - Padova
Tel: 39-049-7625286

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Norway - Trondheim
Tel: 47-7288-4388

Poland - Warsaw
Tel: 48-22-3325737

Romania - Bucharest
Tel: 40-21-407-87-50

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

Sweden - Gothenberg
Tel: 46-31-704-60-40

Sweden - Stockholm
Tel: 46-8-5090-4654

UK - Wokingham
Tel: 44-118-921-5800
Fax: 44-118-921-5820