DAC Programming

Last updated 6/4/21

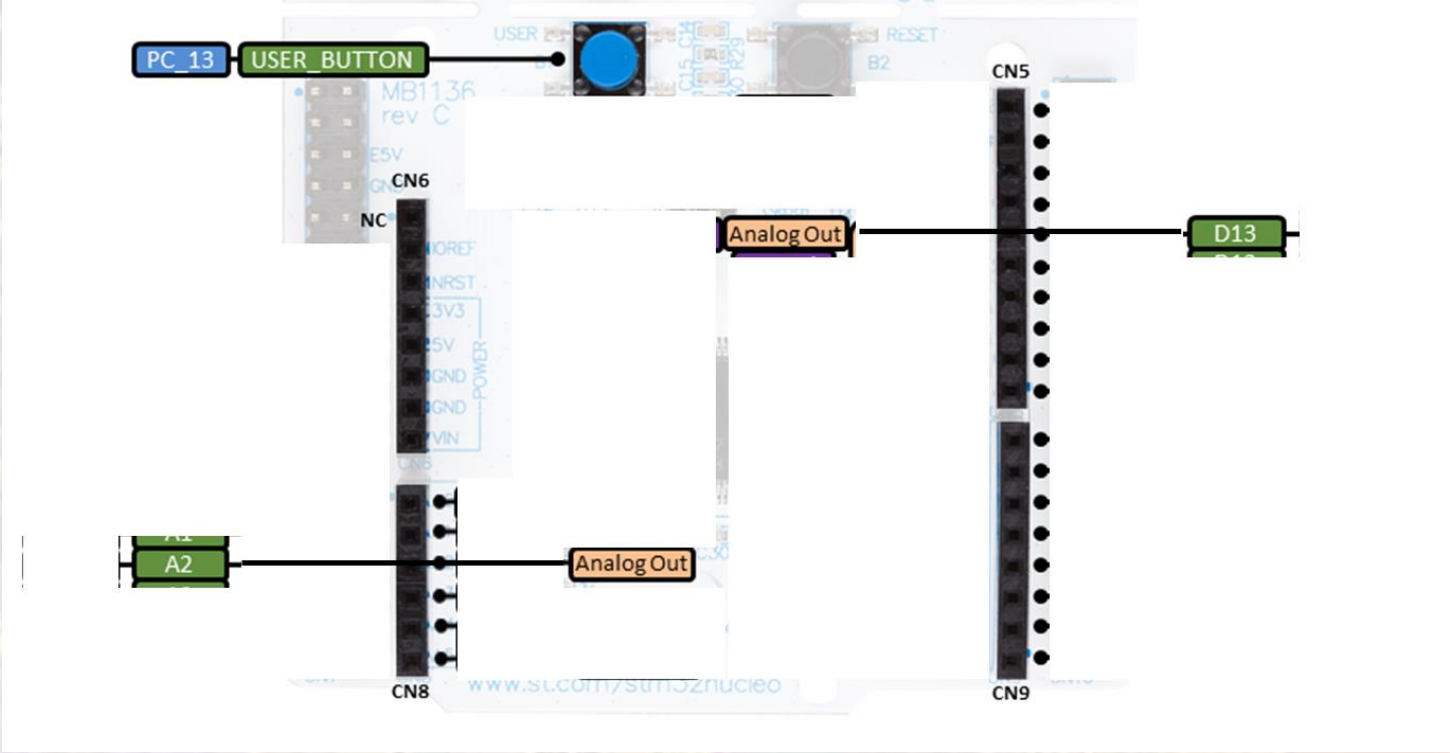
DAC Programming

- DAC Resolution
  - Nucleo-L476RG has two 12Bit DACs
  - Referenced to 3.3V

  - 4096 steps on 3.3v → 805uV/step
  - May be limited by noise

DAC Programming

- DAC Connections
  - Nucleo-L476RG has 2 analog (DAC) outputs assigned to the Arduino header
  - Nucleo-L476RG has 0 additional analog (DAC) outputs assigned to the Morpho header (not shown)



DAC Programming

- AnalogOut Class

AnalogOut Class Reference

Public Member Functions	
	AnalogOut (PinName pin)
	Create an AnalogOut connected to the specified pin. More...
	AnalogOut (const PinMap &&)=delete
	Create an AnalogOut connected to the specified pin. More...
void	write (float value)
	Set the output voltage, specified as a percentage (float) More...
void	write_u16 (unsigned short value)
	Set the output voltage, represented as an unsigned short in the range [0x0, 0xFFFF]. More...
float	read ()
	Return the current output voltage setting, measured as a percentage (float) More...
AnalogOut &	operator= (float percent)
	An operator shorthand for write() More...
AnalogOut &	operator= (AnalogOut &rhs)
	An operator shorthand for write() More...
	operator float ()
	An operator shorthand for read() More...
virtual	~AnalogOut ()

DAC Programming

- Constructors

	AnalogOut (PinName pin)
	Create an AnalogOut connected to the specified pin. More...
	AnalogOut (const PinMap &&)=delete
	Create an AnalogOut connected to the specified pin. More...

```
// Create a DAC object, attached to A2
AnalogOut Defib(A2);
```

DAC Programming

- Member Functions (Methods)

void	write (float value)
	Set the output voltage, specified as a percentage (float) More... 0 – 1.0
void	write_u16 (unsigned short value) 0 – 65535
	Set the output voltage, represented as an unsigned short in the range [0x0, 0xFFFF]. More...
float	read ()
	Return the current output voltage setting, measured as a percentage (float) More...

```
// generate some DAC outputs using member functions
Defib.write(0);
```

DAC Programming

- Operator Overloads

AnalogOut &	operator= (float percent)
	An operator shorthand for write() More...
AnalogOut &	operator= (AnalogOut &rhs)
	An operator shorthand for write() More...
	operator float ()

```
// use the overloaded operators
Defib = 0.33;
```

DAC Programming

- Simple example 1
  - Generate a series of analog output voltages
  - 5ms for each level

```
////////////////////////////////////
//
// dac_class_ex_1 project
//
// created 5/12/21 by tj
// rev 0
//
////////////////////////////////////
//
// DAC example file for class
//
// shows basic DAC commands
//
////////////////////////////////////

#include "mbed.h"
#include <stdio.h>

#define T_WAIT 5000    // in us = 5ms

// Global HARDWARE Objects
// Create DAC objects, attached to A2 and D13
AnalogOut Defib(A2);
AnalogOut Bifed(D13);

int main(void){
    // splash
    printf("dac_class_ex_1 - example for EE2905\n");
    printf("Using Mbed OS version %d.%d.%d\n\n",
            MBED_MAJOR_VERSION, MBED_MINOR_VERSION, MBED_PATCH_VERSION);

    // working variables
    float foo;
```

```
    // run through an endless series of conversions
    while(1){
        // generate some DAC outputs using member functions
        Defib.write(0);
        wait_us(T_WAIT);
        Defib.write(.5);
        wait_us(T_WAIT);
        Defib.write(1.0);
        wait_us(T_WAIT);

        Defib.write_u16(0);
        wait_us(T_WAIT);
        Defib.write_u16(0x8000);
        wait_us(T_WAIT);
        Defib.write_u16(0xFFFF);
        wait_us(T_WAIT);

        // use the read function
        printf("Defib is currently set to %f\n", Defib.read());

        // use the overloaded operators
        Defib = 0.33;
        foo = Defib;
        printf("Defib is currently set to %f\n", foo);
        wait_us(T_WAIT);

        Defib = Bifed = 0.66;
        foo = Defib;
        printf("Defib is currently set to %f\n", foo);
        wait_us(T_WAIT);
    }// end while

    return 0;
}// end main
```

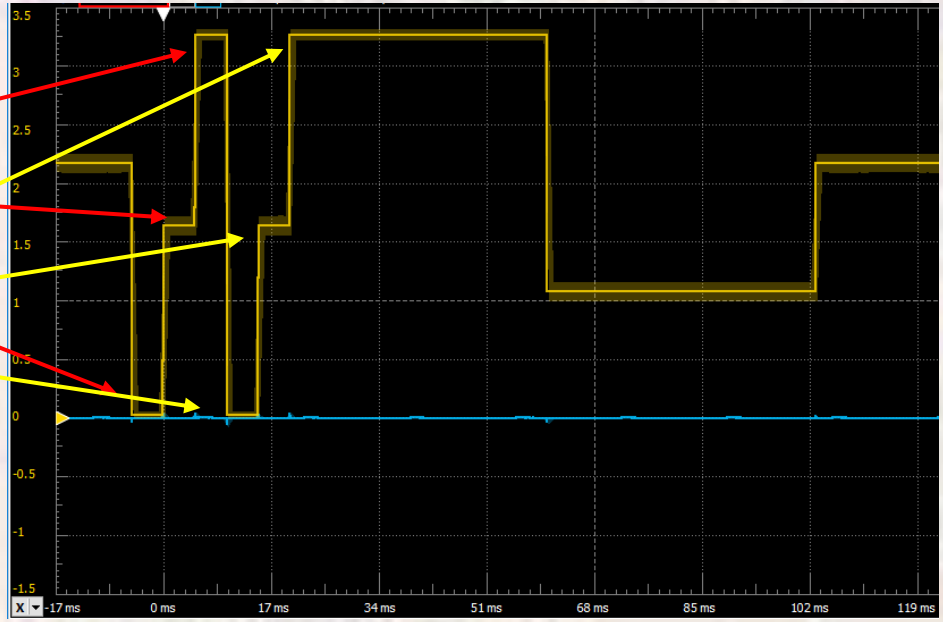
DAC Programming

- Simple example 1 – output levels
  - Generate a series of analog output voltages
  - 5ms for each level

code

```
// generate some DAC outputs using member functions
Defib.write(0);
wait_us(T_WAIT);
Defib.write(.5);
wait_us(T_WAIT);
Defib.write(1.0);
wait_us(T_WAIT);

Defib.write_u16(0);
wait_us(T_WAIT);
Defib.write_u16(0x8000);
wait_us(T_WAIT);
Defib.write_u16(0xFFFF);
wait_us(T_WAIT);
```

Waveform



DAC Programming

- Simple example 1 – output levels
  - Generate a series of analog output voltages
  - 5ms for each level

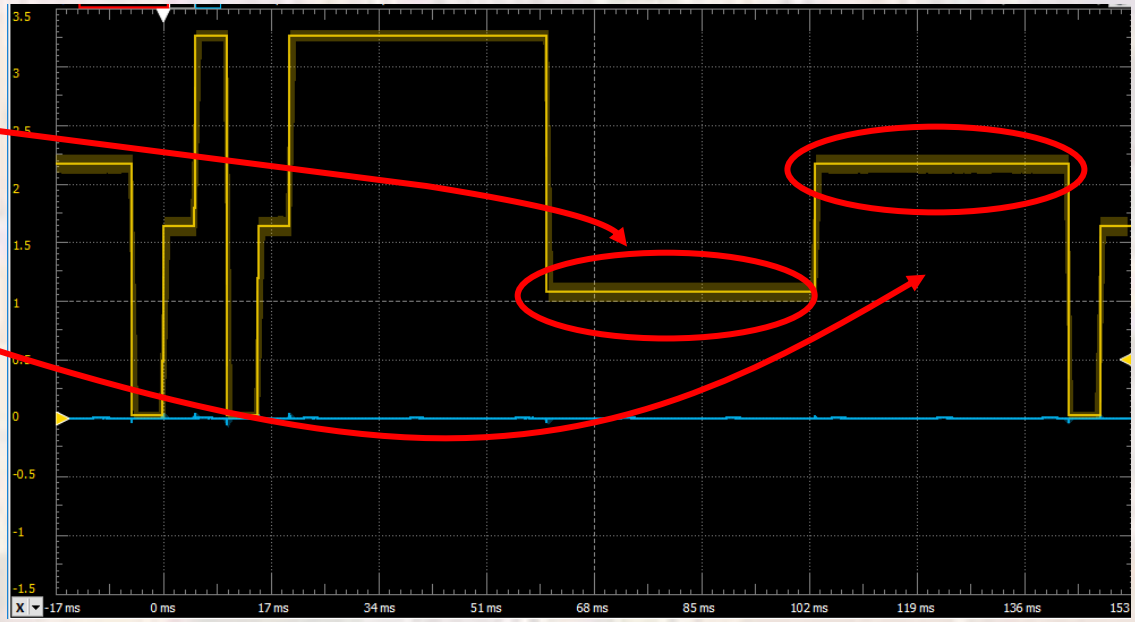
code

```
// use the read function
printf("Defib is currently set to %f\n", Defib.read());

// use the overloaded operators
Defib = 0.33;
foo = Defib;
printf("Defib is currently set to %f\n", foo);
wait_us(T_WAIT);

Defib = Bifed = 0.66;
foo = Defib;
printf("Defib is currently set to %f\n", foo);
wait_us(T_WAIT);
```

output

```
dac_class_ex_1 - example for EE2905
Using Mbed OS version 6.10.0

Defib is currently set to 1.000000
Defib is currently set to 0.329915
Defib is currently set to 0.659829
Defib is currently set to 1.000000
Defib is currently set to 0.329915
Defib is currently set to 0.659829
Defib is currently set to 1.000000
Defib is currently set to 0.329915
Defib is currently set to 0.659829
Defib is currently set to 1.000000
Defib is currently set to 0.329915
Defib is currently set to 0.659829
Defib is curre
```

Waveform



DAC Programming

- Simple example 1 – output levels
  - Generate a series of analog output voltages
  - 5ms for each level

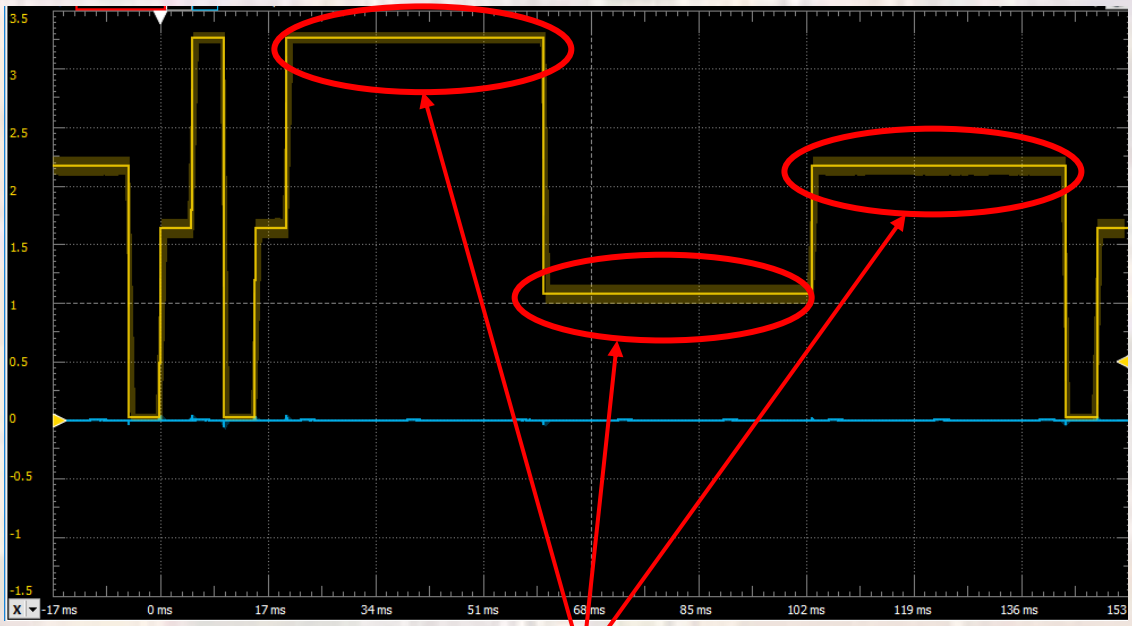
code

```
// use the read function
printf("Defib is currently set to %f\n", Defib.read());

// use the overloaded operators
Defib = 0.33;
foo = Defib;
printf("Defib is currently set to %f\n", foo);
wait_us(T_WAIT);

Defib = Bifed = 0.66;
foo = Defib;
printf("Defib is currently set to %f\n", foo);
wait_us(T_WAIT);
```

output

```
dac_class_ex_1 - example for EE2905
Using Mbed OS version 6.10.0

Defib is currently set to 1.000000
Defib is currently set to 0.329915
Defib is currently set to 0.659829
Defib is currently set to 1.000000
Defib is currently set to 0.329915
Defib is currently set to 0.659829
Defib is currently set to 1.000000
Defib is currently set to 0.329915
Defib is currently set to 0.659829
Defib is currently set to 1.000000
Defib is currently set to 0.329915
Defib is currently set to 0.659829
Defib is curre
```

Waveform



Too long – Why ????

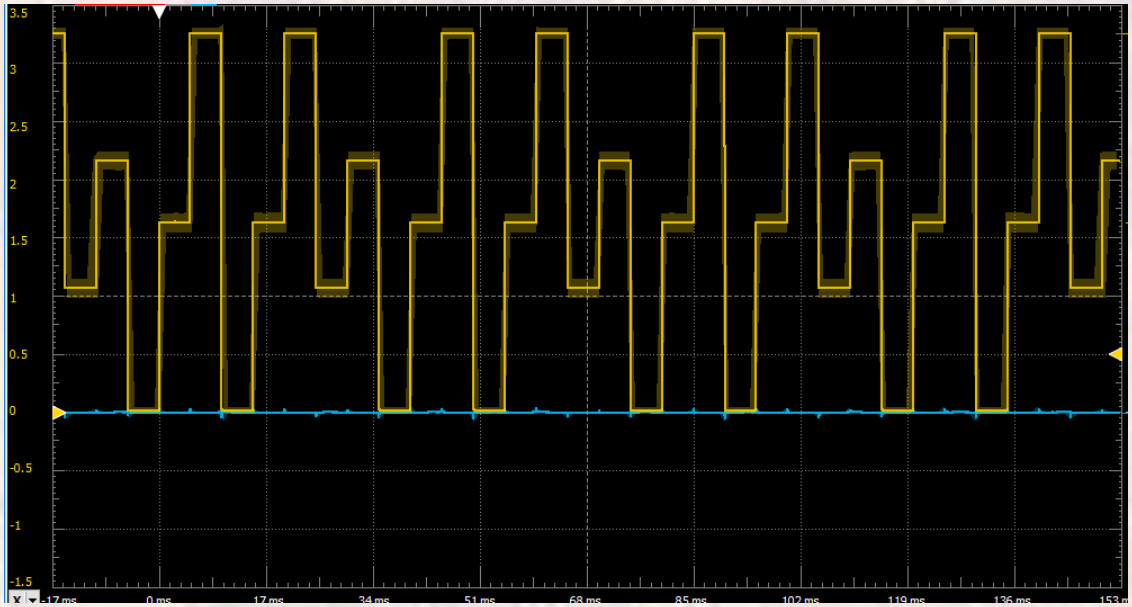
DAC Programming

- Simple example 1 - timing
  - Generate a series of analog output voltages
  - 5ms for each level

Waveform

printf is VERY costly

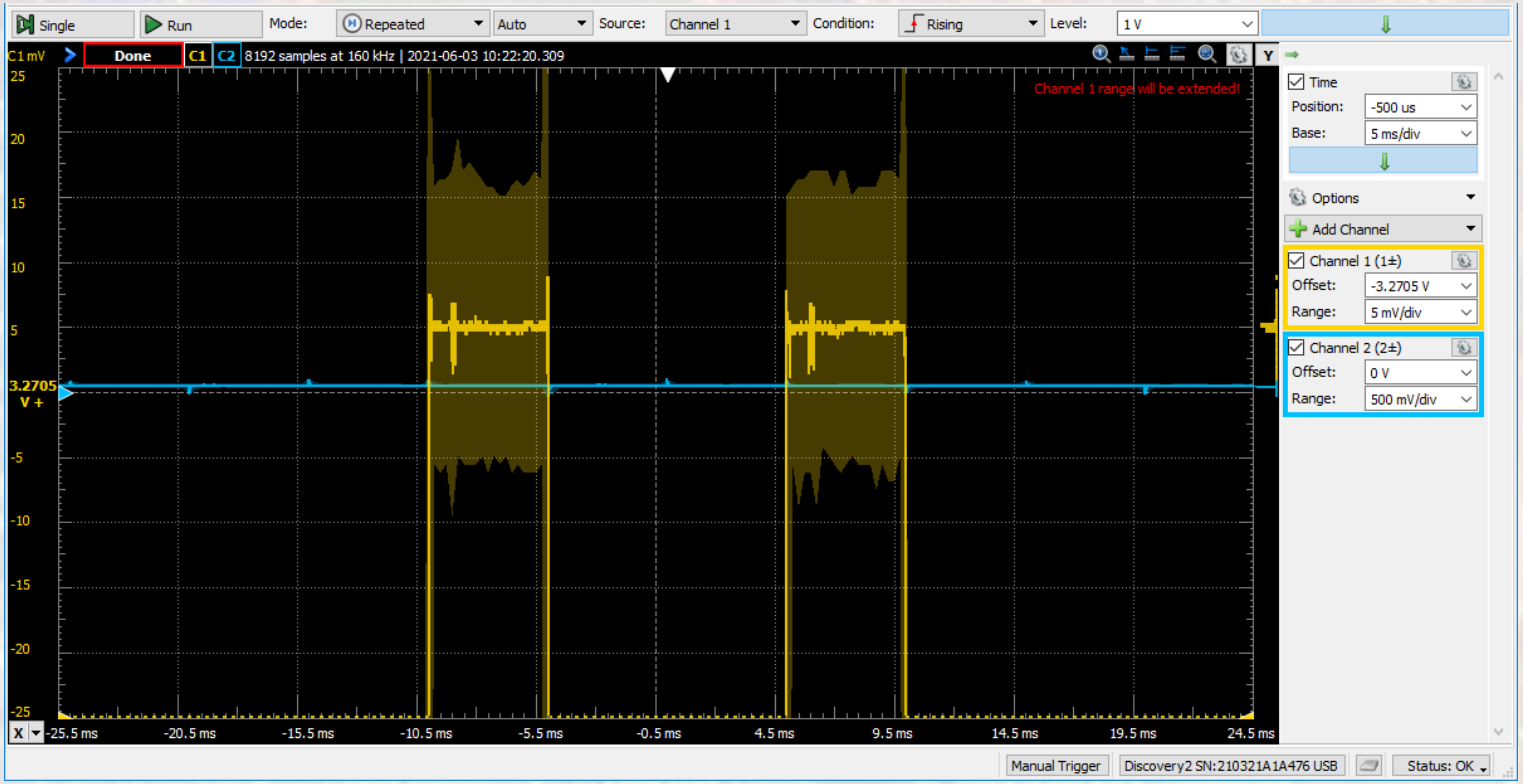
printf statements removed



DAC Programming

- Simple example 1 - noise
  - Generate a series of analog output voltages
  - Output noise ≈ +/- 3mv = 0.1% at full scale

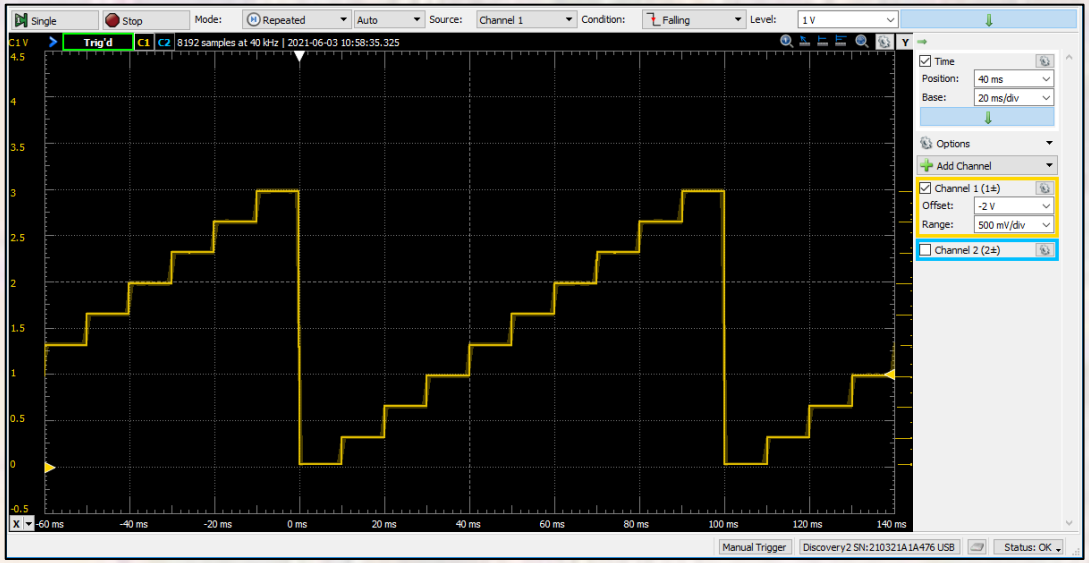
Waveform



DAC Programming

- Simple example 2 part 1
  - Explore ways to write to the DAC
    - Use a for loop

```
////////////////////////////////////
//
// dac_class_ex_2 project
//
// created 5/12/21 by tj
// rev 0
//
////////////////////////////////////
//
// DAC example file for class
//
// explore ways to write to the DAC
//
////////////////////////////////////

#include "mbed.h"
#include <stdio.h>

// Global HARDWARE Objects
// Create a DAC object, attached to A2
AnalogOut Pacer(A2);

int main(void){
    // splash
    printf("dac_class_ex_2 - example for EE2905\n");
    printf("Using Mbed OS version %d.%d.%d\n\n",
           MBED_MAJOR_VERSION, MBED_MINOR_VERSION, MBED_PATCH_VERSION);

    // run through an endless series of conversions
    while(1){
        // change the voltage on the digital output pin by 0.1 * VCC
        for (int i = 0; i < 10; i++){
            Pacer.write(0.1 * i);
            wait_us(10000);
        }// end for
    }// end while

    return 0;
}// end main
```



DAC Programming

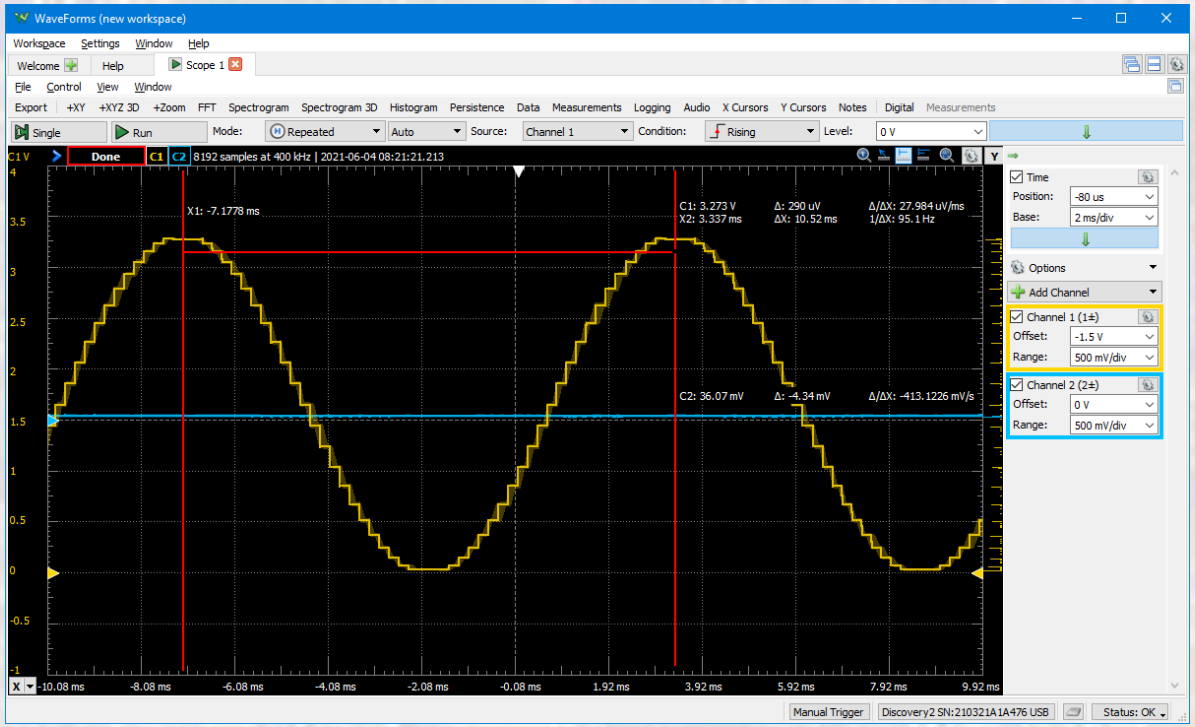
- Simple example 2 part 2
  - Explore ways to write to the DAC
    - Read from an array – specify the frequency

```
////////////////////////////////////
//
// dac_class_ex_2 project
//
// created 5/12/21 by tj
// rev 0
//
////////////////////////////////////
//
// DAC example file for class
//
// explore ways to write to the DAC
//
////////////////////////////////////

#include "mbed.h"
#include <stdio.h>

#define PI 3.14159
#define F_SINE 100 // Hz
#define NUM_PTS 50

// Global HARDWARE Objects
// Create a DAC object, attached to A2
AnalogOut Pacer(A2);

int main(void){
    // splash
    printf("dac_class_ex_2 - example for EE2905\n");
    printf("Using Mbed OS version %d.%d.%d\n\n",
           MBED_MAJOR_VERSION, MBED_MINOR_VERSION, MBED_PATCH_VERSION);

    // working variables
    float sine_ary[NUM_PTS];

    // array setup
    for (int i = 0; i < NUM_PTS; i++){
        sine_ary[i] = sin(i*(2*PI/NUM_PTS))*.5 + 0.5;
    }// end for
```

```
    // run through an endless series of conversions
    while(1){
        // write out a sine wave of variable frequency
        // the sin fn outputs -1 to +1 so we need to divide by 2 to get a
        // total swing of 1. Then need to offset by 0.5 to make everything
        // positive
        // Hardcoding 50 points for the sine wave
         for (int i = 0; i < NUM_PTS; i++){
            Pacer.write(sine_ary[i]);
            // 1000000 to convert to sec, 1/F_sine for period,
            // with NUM_PTS outputs / period
            wait_us(1000000*1.0/F_SINE/NUM_PTS);
        }// end for
    }// end while

    return 0;
}// end main
```

DAC Programming

- Simple example 2 part 2 – frequency error
  - Explore ways to write to the DAC
    - Read from an array – specify the frequency

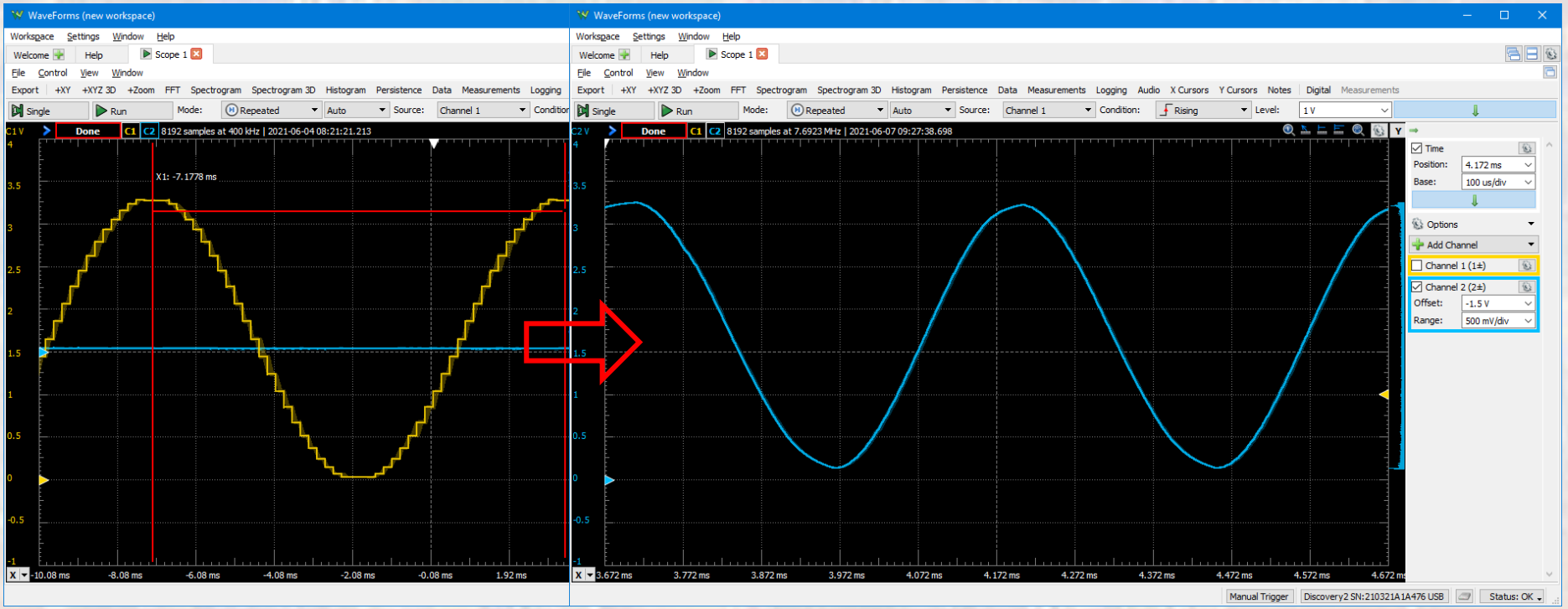


Note: The frequency of the sine wave is ≈ 95Hz

Not the programmed 100Hz

DAC Programming

- Simple example 2 part 2 – filtered output
  - Explore ways to write to the DAC
    - Read from an array – specify the frequency

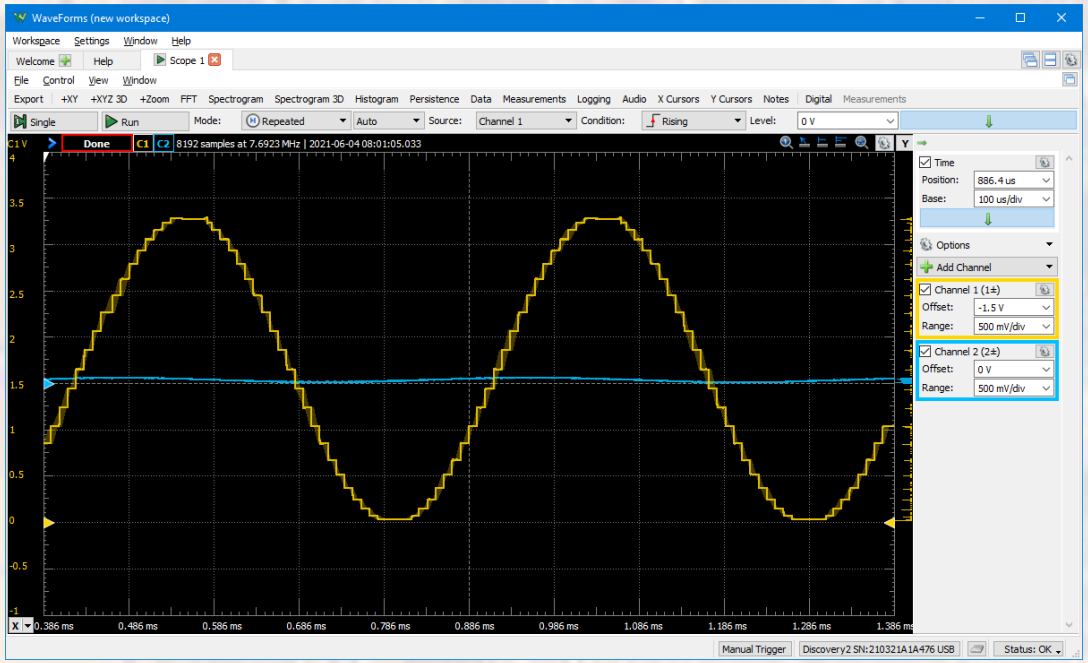


Note: Simple 1pole RC low pass filter

DAC Programming

- Simple example 2 part 3 – maximum frequency
  - Explore ways to write to the DAC
    - Read from an array – don't specify a frequency

```
////////////////////////////////////
//
// dac_class_ex_2 project
//
// created 5/12/21 by tj
// rev 0
//
////////////////////////////////////
//
// DAC example file for class
//
// explore ways to write to the DAC
//
////////////////////////////////////

#include "mbed.h"
#include <stdio.h>

#define PI 3.14159
#define NUM_PTS 50

// Global HARDWARE Objects
// Create a DAC object, attached to A2
AnalogOut Pacer(A2);

int main(void){
    // splash
    printf("dac_class_ex_2 - example for EE2905\n");
    printf("Using Mbed OS version %d.%d.%d\n\n",
            MBED_MAJOR_VERSION, MBED_MINOR_VERSION, MBED_PATCH_VERSION);

    // working variables
    float sine_ary[NUM_PTS];

    // array setup
    for (int i = 0; i < NUM_PTS; i++){
        sine_ary[i] = sin(i*(2*PI/NUM_PTS))*.5 + 0.5;
    }// end for

    // run through an endless series of conversions
    while(1){
        // write out a sine wave of specific frequency
        // the sin fn outputs -1 to +1 so we need to divide by 2 to get a
        // total swing of 1. Then need to offset by 0.5 to make everything
        // positive
        // Hardcoding 50 points for the sine wave
        for (int i = 0; i < NUM_PTS; i++){
            Pacer.write(sine_ary[i]);
        }// end for
    }// end while

    return 0;
}// end main
```



Note: The frequency of the sine wave is ≈ 2KHz

DAC Programming

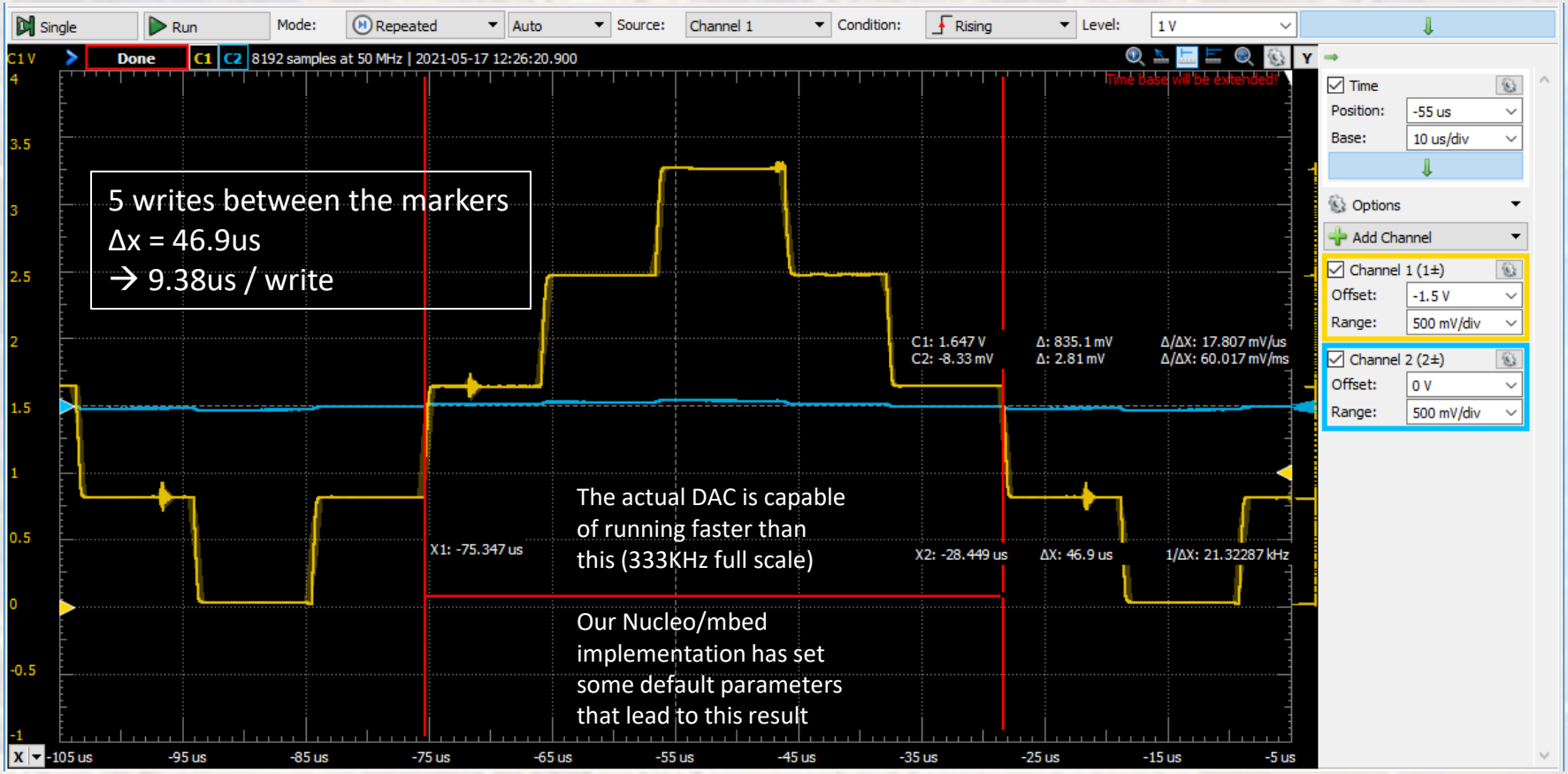
- Simple example 3
  - Estimate how long it takes to do a write to the DAC

```
////////////////////////////////////
//
// dac_class_ex_3 project
//
// created 5/12/21 by tj
// rev 0
//
////////////////////////////////////
//
// DAC example file for class
//
// Minimum write tim to the DAC
//
////////////////////////////////////

#include "mbed.h"
#include <stdio.h>
#define T_WAIT 100    // in us

// Global HARDWARE Objects
// Create a DAC object, attached to A2
AnalogOut Pacer(A2);

int main(void){
    // splash
    printf("dac_class_ex_3 - example for EE2905\n");
    printf("Using Mbed OS version %d.%d.%d\n\n",
           MBED_MAJOR_VERSION, MBED_MINOR_VERSION, MBED_PATCH_VERSION);
```

```
    // run through an endless series of conversions
    while(1){
        // Write a series of values to the DAC with no
        // intervening instructions
        Pacer.write(0.0);
        Pacer.write(0.25);
        Pacer.write(0.5);
        Pacer.write(0.75);
        Pacer.write(1.0);
        Pacer.write(0.75);
        Pacer.write(0.5);
        Pacer.write(0.25);
    }// end while

    return 0;
}// end main
```

DAC Programming

- Simple example 3
  - Estimate how long it takes to do a write to the DAC output



DAC Programming

- Limitations summary
  - ~10us to write a new value to the DAC
  - Fastest 50 point sine wave → 50 x 10us / period = 500us/per → 2KHz
  - Error in 100Hz, 50 point sine wave
    - Period extension : (10us/10ms)x50 = 5.0%
    - 100Hz → 95Hz