Lab 7

Michael
Lab Section: 7F34

August 29, 2017

b. Answers to all pre-lab questions

1) If the NAD/DAD board were not available, nor any conventional desktop measuring tools (like a voltmeter), how could you verify that your DAC were functioning properly

ANS: One way to check is to use ADC(analog to digital converter) to verify. The DAC output would be sent to a ACD input, then using conversions and usart (LAB 5), the values can be outputted to a terminal.

2) You probably noticed that your sine wave does not look very good. While at least meeting specifications, how could you increase the quality of your sine wave?

ANS: To increase the quality of my sine wave, 2 things must be done. The first is that more data points would have to be used. The second thing is the timer would have to be decreased (the PER) in order to keep the desired Hz

3) How many DMA channels are available on the XMEGA?

ANS: CH0, CH1, CH2, and CH3 are available for a total of 4 DMA channels on the XMEGA.

4) How many different options are there to trigger the DMA?

ANS: There are 26 different options to trigger the DMA.

5) While you were not asked to implement this in your lab, explain how one might vary the amplitude of an output wave, much like the frequency was varied.

ANS: One might change the amplitude of the output wave by using a reference higher than the one currently used (not using AREFB). Then one would have to increase/add values in the sine table higher than 0xFFF.

c. Problems Encountered

Problems encounter was that in part D, the wave type would not switch if the output was set to 0. This was fixed by resetting the DMA every time it was switched

d. Future Work/Applications

Future application would be using the speaker. Instead of using data for a sine wave, the data can be for musical notes and the notes can be played from the speaker.

e. Schematics

N/A

g. Pseudocode/Flowcharts

Pseudocode for Lab7A.c:

```
MAIN:
    * Call Change_CLK_32HZ Function
    * Call DAC_INIT();
    * Output 1.3V

    WHILE(TRUE);
END

FUNCTION DAC_INIT
    * Set up port A
    * Set up DAC controls

FUNCTION UPDATE_DACA_CH0
    * Update DACA.CH0 data

FUNCTION Change_CLK_32HZ
    * Enable the new oscillator

    WHILE(OSC FLAG not set){}

    * Write the IOREG signature to the CPU_CCP reg
    * Select the new clock source in the CLK_CTRL reg
```

Pseudocode for Lab7B.c:

```
MAIN:
    * Call Change_CLK_32HZ Function
    * Call DAC_INIT
    * Call COUNTER_INIT
    * Call COUNTER_START

    * Enable interrupts

    WHILE(TRUE);
END

FUNCTION COUNTER_INIT
    * Set top of counter

FUNCTION COUNTER_START
    * Set interupt for counter
    * Start COUNTER

ISR TCC0_OVF_vect
    * Preserve Status Reg
    * Change output
    IF(COUNTER equals 100){
        * Reset Counter
    }
    * Clear interrupt flags
    * Restore Status Reg

FUNCTION DAC_INIT
    * Set up port A
    * Set up DAC controls

FUNCTION UPDATE_DACA_CH0
    * Update DACA.CH0 data

FUNCTION Change_CLK_32HZ
    * Enable the new oscillator

    WHILE(OSC FLAG not set){}

    * Write the IOREG signature to the CPU_CCP reg
    * Select the new clock source in the CLK_CTRL reg
```

Pseudocode for Lab7C.c:

```
    MAIN:
    * Call Change_CLK_32HZ Function
    * Call DAC_INIT
    * Call DMA_INIT
    * Call COUNTER_INIT
    * Call COUNTER_START

    * Enable interrupts

    WHILE(TRUE);
END


FUNCTION DMA_INIT
    * Enable DMA
    * Set up ADDRESS CONTROL
    * Set up TRIGGER SOURCE
    * Set up CH BLOCK TRANSFER COUNT
    * Set up DMA as CONTINOUS
    * Set up Source Address
    * Set up Destination Address
    * Set up CH0 control


FUNCTION COUNTER_INIT
    * Set top of counter

FUNCTION COUNTER_START
    * Set interupt for counter
    * Start COUNTER

ISR TCC0_OVF_vect
    * Preserve Status Reg
    * Clear interrupt flags
    * Restore Status Reg

FUNCTION DAC_INIT
    * Set up port A
    * Set up DAC controls

FUNCTION UPDATE_DACA_CH0
    * Update DACA.CH0 data
```

```
FUNCTION Change_CLK_32HZ
    * Enable the new oscillator

    WHILE(OSC FLAG not set){}

    * Write the IOREG signature to the CPU_CCP reg
    * Select the new clock source in the CLK_CTRL reg
```

Pseudocode for Lab7D.c:

```
    MAIN:
    * Call Change_CLK_32HZ Function
    * Call DAC_INIT
    * Call DMA_INIT
    * Call COUNTER_INIT
    * Call COUNTER_START
    * Call USART_INIT
    * Enable interrupts
    WHILE(TRUE){
        * Display Menu
        * Get User input
     SWITCH(user input){
         CASE 's' CASE 'S'
             * Change DMA with Sine data
         CASE 't' CASE 'T'
             * Change DMA with Triangle data
         CASE '0'
             * Turn off TCC0
             * Output 0v
         CASE '1'
             * Change Hz to 50*1
         CASE '2'
             * Change Hz to 50*2
         CASE '3'
             * Change Hz to 50*3
         CASE '4'
             * Change Hz to 50*4
         CASE '5'
             * Change Hz to 50*5
         CASE '6'
             * Change Hz to 50*6
         CASE '7'
             * Change Hz to 50*7
         CASE '8'
             * Change Hz to 50*8
         CASE '9'
             * Change Hz to 50*9
     }
     IF(choice is 1 - 9){
         * Call COUNTER_START()
     }
    }
END
```

```
FUNCTION DISPLAY_MENU
    * Make Menu Strings
    * Set up array of menu strings
    FOR(int i = 0; i < size; i++){
        * Display string
    }

FUNCTION USART_INIT
    * Set port D for USART com
    * Set up Ctrl B and C
    * Set up baud rate

FUNCTION OUT_STRING
    While(char[] does not equal NULL){
        * Call OUT_CHAR
    }

FUNCTION OUT_CHAR
    While(Transmitting){}
    * Return USART Data in a CHAR

FUNCTION IN_CHAR
    * WHILE(Recieving){}
    * Return data

FUNCTION DMA_INIT
    * Enable DMA
    * Set up ADDRESS CONTROL
    * Set up TRIGGER SOURCE
    * Set up CH BLOCK TRANSFER COUNT
    * Set up DMA as CONTINOUS
    IF(MODE is SINE){
        * Call SINE_SOURCE_DES
    }
    ELSE IF(MODEis TRIANGLE){
       * Call TRIANGLE_SOURCE_DES
    }
    * Set up CH0 control

FUNCTION SINE_SOURCE_DES
    * Wait until CH is not busy
    * Set up address with Sine
    * Set up Source Address
    * Set up Destination Address
```

```
FUNCTION TRIANGLE_SOURCE_DES
    * Wait until CH is not busy
    * Set up address with Sine
    * Set up Source Address
    * Set up Destination Address

FUNCTION COUNTER_INIT
    * Set top of counter

FUNCTION COUNTER_START
    * Set interupt for counter
    * Start COUNTER

ISR TCC0_OVF_vect
    * Preserve Status Reg
    * Clear interrupt flags
    * Restore Status Reg

FUNCTION DAC_INIT
    * Set up port A
    * Set up DAC controls

FUNCTION UPDATE_DACA_CH0
    * Update DACA.CH0 data

FUNCTION Change_CLK_32HZ
    * Enable the new oscillator

    WHILE(OSC FLAG not set){}

    * Write the IOREG signature to the CPU_CCP reg
    * Select the new clock source in the CLK_CTRL reg
```

h. Program Code

Code for Lab7A.c:

```
/*Lab 7 Part A
Name: Michael Arboleda
Section #: 7F34
TA Name: Wesley Piard
Description: Output 1.3V using DAC

Lab7A.c
Created: 7/25/2017 12:48:48 AM
*/

#include <avr/io.h>
#include "constants.h"
#include "Clk_32MHz.h"
#include "DAC.h"

int main(void){
   //Set up program
   Change_CLK_32HZ();
   DAC_INIT();
   UPDATE_DACA_CH0(2130);

   //Run forever
    while(TRUE);

   //Return is 0 status
   return 0;
}
```

Code for Lab7B.c:

```
/*Lab 7 Part B
Name: Michael Arboleda
Section #: 7F34
TA Name: Wesley Piard
Description: Output SINE using DAC

Lab7B.c
Created: 7/25/2017 2:08:59 AM
*/

#include <avr/io.h>
#include <avr/interrupt.h>
#include "constants.h"
#include "Clk_32MHz.h"
#include "DAC.h"
#include "TIMER_COUNTER.h"

int main(void){
   //Set up program
   Change_CLK_32HZ();
   DAC_INIT();
   COUNTER_INIT();
   COUNTER_START();

   //enable interrupts.
   PMIC_CTRL = 0x07;
    sei();

   //Run forever
    while(TRUE);

   //Return is 0 status
   return 0;
}
```

Code for Lab7C.c:

```
/*Lab 7 Part C
Name: Michael Arboleda
Section #: 7F34
TA Name: Wesley Piard
Description: Output SINE using DAC and DMA

Lab7C.c
Created: 7/25/2017 3:51:09 AM
*/

#include <avr/io.h>
#include <avr/interrupt.h>
#include "constants.h"
#include "Clk_32MHz.h"
#include "DAC.h"
#include "TIMER_COUNTER.h"
#include "DMA.h"

int main(void){
   //Set up program
   Change_CLK_32HZ();
   DAC_INIT();
   DMA_INIT();
   COUNTER_INIT();
   COUNTER_START();

   //enable interrupts.
   PMIC_CTRL = 0x07;
   sei();

   //Run forever
   while(TRUE);

   //Return is 0 status
   return 0;
}
```

Code for Lab7D.c

```
/*Lab 7 Part D
Name: Michael Arboleda
Section #: 7F34
TA Name: Wesley Piard
Description: Advance Waveform Generation

Lab7D.c
Created: 7/26/2017 2:57:37 AM
*/

#include <avr/io.h>
#include <avr/interrupt.h>
#include "constants.h"
#include "Clk_32MHz.h"
#include "DAC.h"
#include "TIMER_COUNTER.h"
#include "DMA.h"
#include "USART.h"


//Function Prototypes
void DISPLAY_MENU(void);

int main(void){
   //Set up program
   Change_CLK_32HZ();
   DAC_INIT();
   DMA_INIT('s');
   COUNTER_INIT();
   COUNTER_START();
   USART_INIT();

   //enable interrupts.
   PMIC_CTRL = 0x07;
   sei();

   //Set up vars
   char choice;

   //Run forever
   while(TRUE){
      //Display Menu, User input
      DISPLAY_MENU();
      choice = IN_CHAR();
      OUT_STRING("USER␣CHOICE:␣");
      OUT_CHAR(choice);
      OUT_STRING("\r\n");

      //Output based on choice
      switch(choice){
```

```
        case 'S':
        case 's':
          DMA_CHANGE('s');
          break;
        case 'T':
        case 't':
          DMA_CHANGE('t');
          break;
        case '0':
          //Turn off TCC0, Output 0
          COUNTER_STOP();
          TCC0.PER = 0;
          DACA.CH0DATA = 0;
          break;
        case '1':
          UPDATE_PER(1);
          break;
        case '2':
          UPDATE_PER(2);
          break;
        case '3':
          UPDATE_PER(3);
          break;
        case '4':
          UPDATE_PER(4);
          break;
        case '5':
          UPDATE_PER(5);
          break;
        case '6':
          UPDATE_PER(6);
          break;
        case '7':
          UPDATE_PER(7);
          break;
        case '8':
          UPDATE_PER(8);
          break;
        case '9':
          UPDATE_PER(9);
          break;
      }
      //Restart counter from when 0
      if(choice >= '1' && choice <= '9'){
          COUNTER_START();
      }
    }

    //Return is 0 status
    return 0;
}
```

```
void DISPLAY_MENU(void){
   uint8_t menu_size = 12;

   //Menu String
   char *menuS = "S/s: Output a sinusoid\r\n";
   char *menuT = "T/t: Output a triangle wave\r\n";
   char *menu0 = "0: Make the output waveform:0 Hz and 0 V\r\n";
   char *menu1 = "1: Make the output waveform:50Hz (1 x 50 Hz)\r\n";
   char *menu2 = "2: Make the output waveform:100Hz (2 x 50 Hz)\r\n";
   char *menu3 = "3: Make the output waveform:150Hz (3 x 50 Hz)\r\n";
   char *menu4 = "4: Make the output waveform:100Hz (4 x 50 Hz)\r\n";
   char *menu5 = "5: Make the output waveform:250Hz (5 x 50 Hz)\r\n";
   char *menu6 = "6: Make the output waveform:300Hz (6 x 50 Hz)\r\n";
   char *menu7 = "7: Make the output waveform:350Hz (7 x 50 Hz)\r\n";
   char *menu8 = "8: Make the output waveform:400Hz (8 x 50 Hz)\r\n";
   char *menu9 = "9: Make the output waveform:450Hz (9 x 50 Hz)\r\n";

   //Set up array of menu strings
   char *menu[] = {menuS, menuT, menu0, menu1, menu2, menu3, menu4,
      menu5, menu6, menu7, menu8, menu9};

   //Loop thru menu array to display
   for(uint8_t i = 0; i < menu_size; i++){
      OUT_STRING(menu[i]);
   }

   //return
   return;
}
```

i. Appendix

Figure 1: DAD reading of constant 1.3V signal

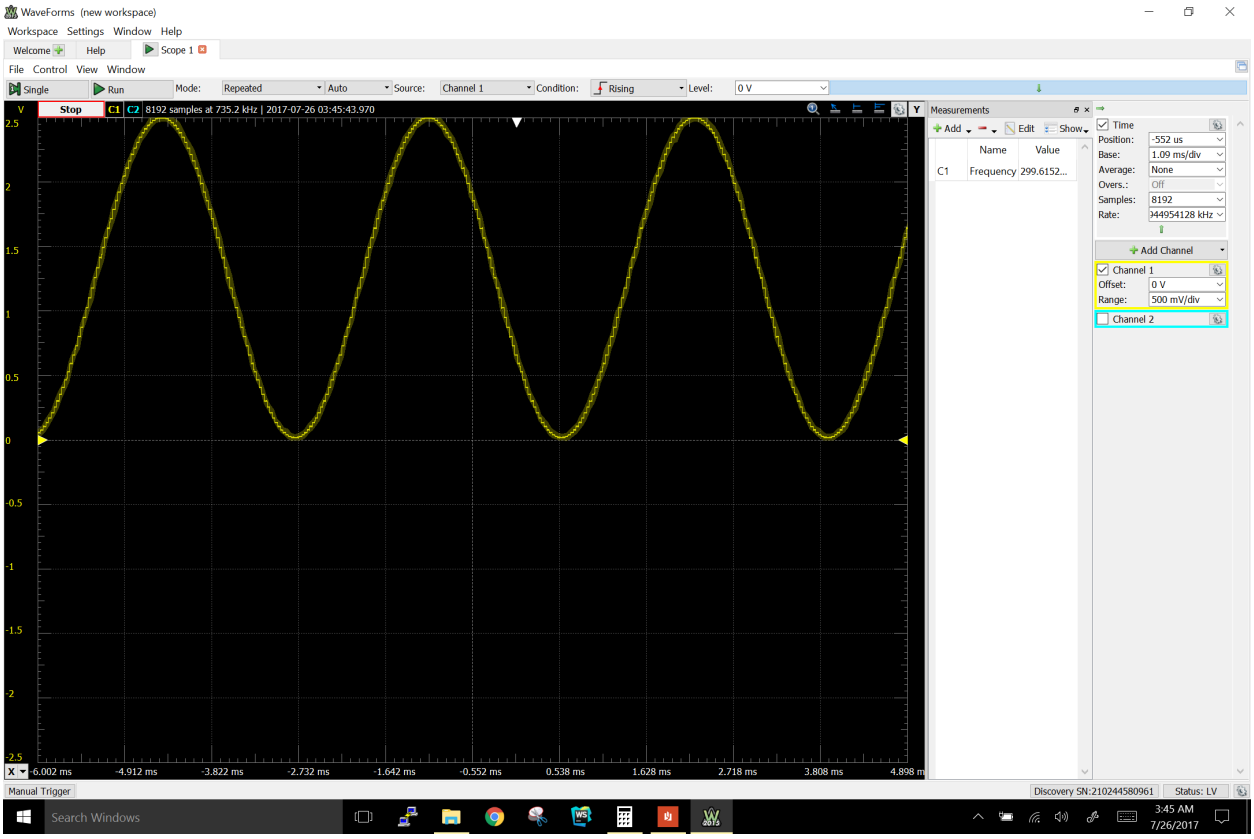

Figure 2: 300Hz sine wave

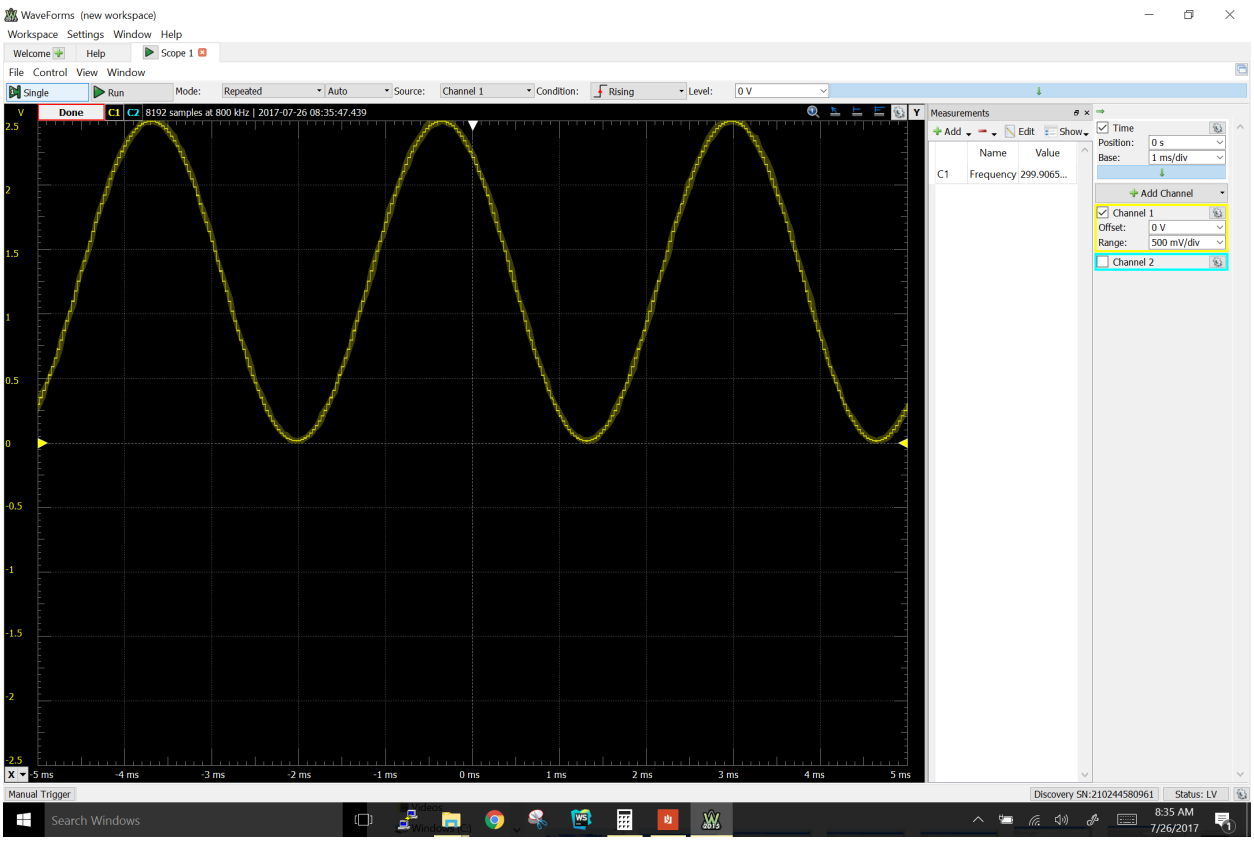

Figure 3: 300Hz sine wave (with DMA)

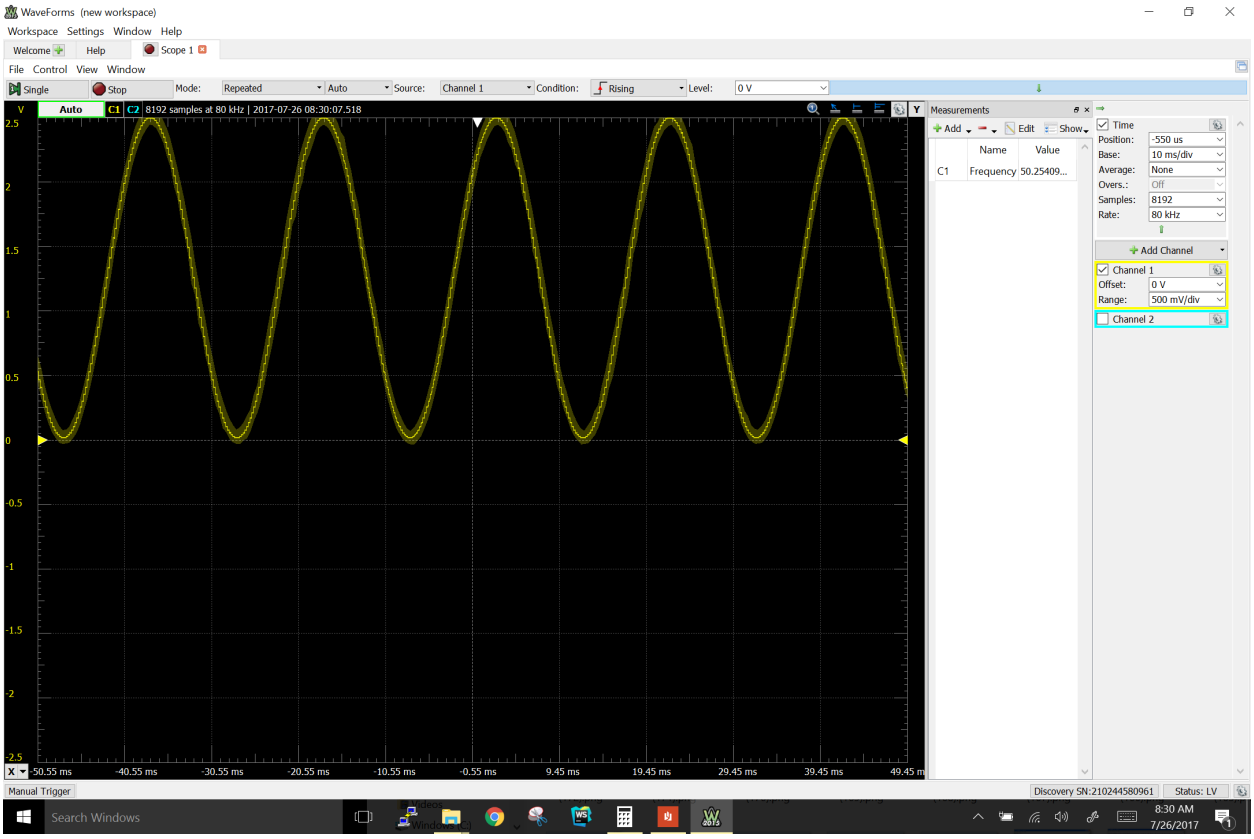

Figure 4: 50Hz sine wave

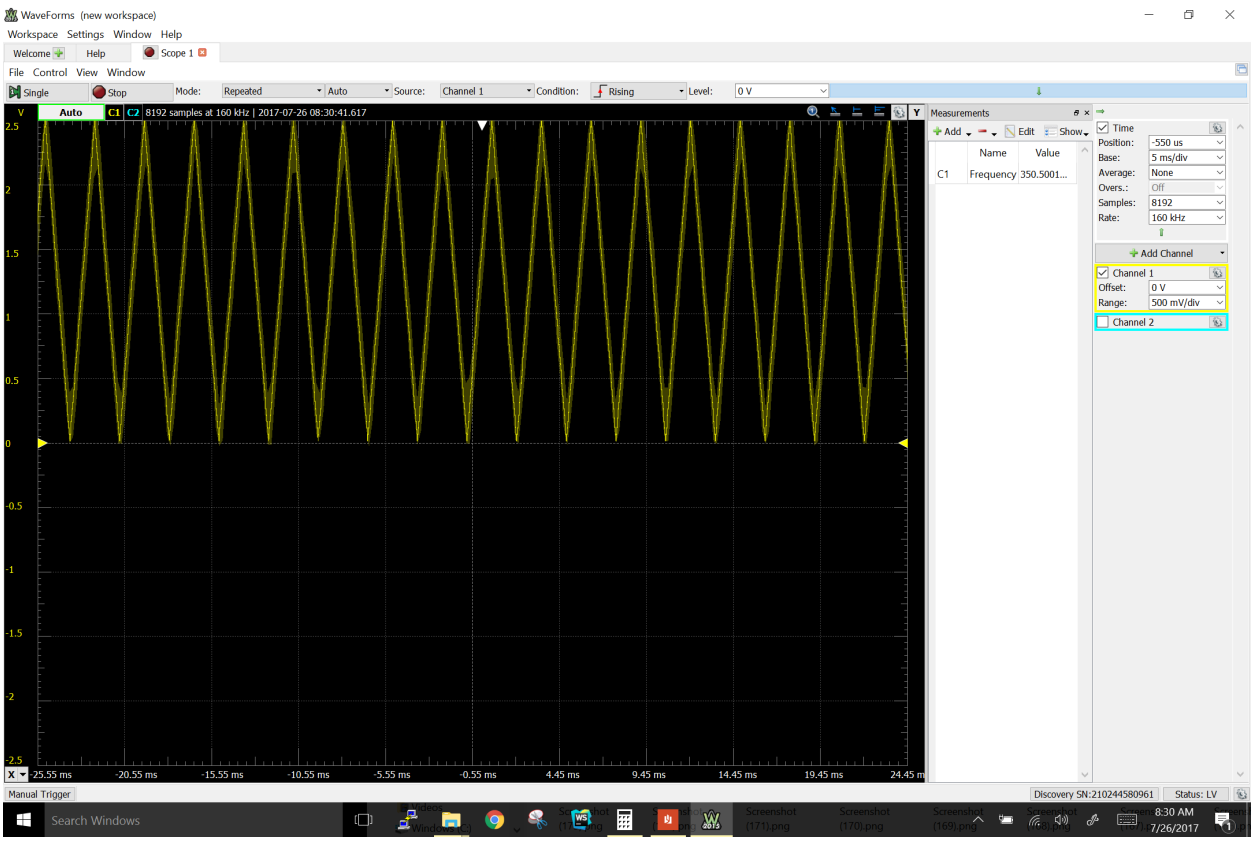

Figure 5: 350Hz triangle wave

Code for Clk_32MHz.h:

```
#ifndef CLK_32MHZ_H_
#define CLK_32MHZ_H_

/*
Name: Michael Arboleda
Section #: 7F34
TA Name: Wesley Piard
Description: Header for CLK_32MHz.c

Clk_32MHz.h
Created: 7/7/2017 11:20:13 PM
*/


//Function Prototype
void Change_CLK_32HZ(void);


#endif /* CLK_32MHZ_H_ */
```

Code for Clk_32MHz.c:

```
/*
Name: Michael Arboleda
Section #: 7F34
TA Name: Wesley Piard
Description: Changes uP freq to 32Mhz

* Clk_32MHz.c
* Created: 7/17/2017 6:21:29 AM
*/


#include <avr/io.h>

//include extern constants
extern const uint8_t NEW_CLOCK_FREQ;


/*********************Function***************************************
; Function Name: Change_CLK_32HZ
; Inputs: No direct input (from stack)
; Outputs: No direct outputs
*/
void Change_CLK_32HZ(void){

   //Set the clk config
   OSC_CTRL = NEW_CLOCK_FREQ;

   //Wait for the right flag to be set in the OSC_STATUS reg
   while((OSC_STATUS & PIN1_bm) != PIN1_bm);

   //Write the IOREG signature to the CPU_CCP reg
   CPU_CCP = CCP_IOREG_gc;

   //Select the new clock source in the CLK_CTRL reg
   CLK_CTRL = CLK_SCLKSEL_RC32M_gc;

   return;
}
```

Code for DAC.h:

```
#ifndef DAC_H_
#define DAC_H_

/*
Name: Michael Arboleda
Section #: 7F34
TA Name: Wesley Piard
Description: Header for DAC

DAC.h
Created: 7/25/2017 1:08:59 AM
*/

void DAC_INIT(void);
void UPDATE_DACA_CH0(uint16_t data);



#endif /* DAC_H_ */
```

Code for DAC.c:

```
/*
Name: Michael Arboleda
Section #: 7F34
TA Name: Wesley Piard
Description: Functions for DAC

DAC.c
Created: 7/25/2017 1:08:39 AM
*/


#include <avr/io.h>

//include extern constants
extern const uint8_t DAC_CTRLA_CONFIG;
extern const uint8_t DAC_CTRLC_CONFIG;

/*********************Function*****************************************
; Function Name: DAC_INIT
; Inputs: No direct input
; Outputs: No direct outputs
*/
void DAC_INIT(void){
   //Set port A for output
   PORTA.DIRSET = PIN2_bm;//0x04

   //Set up DAC controls
   DACA.CTRLA = DAC_CTRLA_CONFIG;
   DACA.CTRLC = DAC_CTRLC_CONFIG;

}

/*********************Function*****************************************
; Function Name: UPDATE_DACA_CH0
; Inputs: uint8_t:data
; Outputs: No direct outputs
*/
void UPDATE_DACA_CH0(uint16_t data){
   //Store data in CH0
   DACA.CH0DATA = data;

   //Return from function
   return;
}
```

Code for DMA.h:

```
#ifndef DMA_H_
#define DMA_H_
/*
Name: Michael Arboleda
Section #: 7F34
TA Name: Wesley Piard
Description: Header for DMA

DMA.h
Created: 7/25/2017 3:54:42 AM
*/


//Function Prototypes
void DMA_INIT(uint8_t mode);
void SINE_SOURCE_DES(void);
void TRIANGLE_SOURCE_DES(void);
void DMA_CHANGE(uint8_t mode);




#endif /* DMA_H_ */
```

Code for DMA.c:

```
/*
Name: Michael Arboleda
Section #: 7F34
TA Name: Wesley Piard
Description: Functions for DMA

DMA.c
Created: 7/25/2017 3:53:36 AM
*/

#include <avr/io.h>

//Include constants
extern uint16_t sine[];
extern uint16_t triangle[];
extern const uint8_t DMA_CTRL_CONFIG;
extern const uint8_t DMA_CH0_ADDRCTRL_CONFIG;
extern const uint8_t DMA_CH0_TRIGSRC_CONFIG;
extern const uint8_t DMA_CH0_TRFCNT_CONFIG;
extern const uint8_t DMA_CH0_REPCNT_CONFIG;
extern const uint8_t DMA_CH0_CTRLA_CONFIG;

//Function Prototype
void SINE_SOURCE_DES(void);

/*********************Function*****************************************
; Function Name: DMA_INIT
; Inputs: No inputs
; Outputs: No outputs
*/
void DMA_INIT(uint8_t mode){
   //Enable DMA
   DMA.CTRL = DMA_CTRL_CONFIG;

   //ADDRESS CONTROL REG: BLOCK;INCREMENT;BURST;INCREMENT
   DMA.CH0.ADDRCTRL = DMA_CH0_ADDRCTRL_CONFIG;

   //TRIGGER SOURCE REG: TCC0
   DMA.CH0.TRIGSRC = DMA_CH0_TRIGSRC_CONFIG;

   //CH BLOCK TRANSFER COUNT: 200=100(8bit)*2(16bit)
   DMA.CH0.TRFCNT = DMA_CH0_TRFCNT_CONFIG;

   //CONTINOUS DMA
   DMA.CH0.REPCNT = DMA_CH0_REPCNT_CONFIG;

   if(mode == 's'){
      //Set up DMA with sine
      SINE_SOURCE_DES();
   }
```

```
   else if(mode == 't'){
      //Set up DMA with sine
      TRIANGLE_SOURCE_DES();
   }

   //Set CH0 CTRL: ENABLE;REPEAT;SINGLE;2BYTE
   DMA.CH0.CTRLA = DMA_CH0_CTRLA_CONFIG;

   //Return from function
   return;
}



/*********************Function*****************************************
; Function Name: SINE_SOURCE_DES
; Inputs: No inputs
; Outputs: No outputs
*/
void SINE_SOURCE_DES(void){
   //Wait until CH is not busy
   while((DMA.CH0.CTRLB & PIN7_bm) == PIN7_bm);

   //Set up address with Sine
   uint32_t SINE_ADDRESS = (uint32_t)sine;

   //Source Address
   DMA.CH0.SRCADDR0 = (uint8_t)(SINE_ADDRESS >> 0);
   DMA.CH0.SRCADDR1 = (uint8_t)(SINE_ADDRESS >> 8);
   DMA.CH0.SRCADDR2 = (uint8_t)(SINE_ADDRESS >> 16);

   uint8_t* dac_ptr = &DACA_CH0DATA;
   uint32_t dac_address = (uint32_t)dac_ptr;

   //Destination Address
   DMA.CH0.DESTADDR0 = (uint8_t)(dac_address >> 0);
   DMA.CH0.DESTADDR1 = (uint8_t)(dac_address >> 8);
   DMA.CH0.DESTADDR2 = (uint8_t)(dac_address >> 16);

   //Return from function
   return;
}



/*********************Function*****************************************
; Function Name: TRIANGLE_SOURCE_DES
; Inputs: No inputs
; Outputs: No outputs
*/
void TRIANGLE_SOURCE_DES(void){
   //Wait until CH is not busy
```

```
   while((DMA.CH0.CTRLB & PIN7_bm) == PIN7_bm);

   //Set up address with Triangle
   uint32_t TRIANGLE_ADDRESS = (uint32_t)triangle;

   //Source Address
   DMA.CH0.SRCADDR0 = (uint8_t)(TRIANGLE_ADDRESS >> 0);
   DMA.CH0.SRCADDR1 = (uint8_t)(TRIANGLE_ADDRESS >> 8);
   DMA.CH0.SRCADDR2 = (uint8_t)(TRIANGLE_ADDRESS >> 16);

   uint8_t* dac_ptr = &DACA_CH0DATA;
   uint32_t dac_address = (uint32_t)dac_ptr;

   //Destination Address
   DMA.CH0.DESTADDR0 = (uint8_t)(dac_address >> 0);
   DMA.CH0.DESTADDR1 = (uint8_t)(dac_address >> 8);
   DMA.CH0.DESTADDR2 = (uint8_t)(dac_address >> 16);

   //Return from function
   return;
}

void DMA_CHANGE(uint8_t mode){
   //Disable DMA
   DMA.CTRL = 0;

   //Software Reset on
   DMA.CTRL = 0b01000000;

   //Re initlize DMA with new wave mode
   DMA_INIT(mode);
}
```

Code for TIMER_COUNTER.h:

```
#ifndef TIMER_COUNTER_H_
#define TIMER_COUNTER_H_

/*
Name: Michael Arboleda
Section #: 7F34
TA Name: Wesley Piard
Description: Header for TIMER_COUNTER

TIMER_COUNTER.h
Created: 7/12/2017 6:09:50 AM
*/


void COUNTER_INIT(void);
void COUNTER_START(void);
void COUNTER_STOP(void);
void UPDATE_PER(uint8_t input);


#endif /* TIMER_COUNTER_H_ */
```

Code for TIMER_COUNTER.c:

```
/*
Name: Michael Arboleda
Section #: 7F34
TA Name: Wesley Piard
Description: Functions for TIMER_COUNTER

TIMER_COUNTER.c
Created: 7/25/2017 2:21:42 AM
*/

#include <avr/io.h>
#include <avr/interrupt.h>
#include "DAC.h"

//Include extern constants
extern const uint16_t TC_PER_CONFIG;
extern const uint8_t INTCTRLA_CONFIG;
extern const uint8_t CLK_DIV_CONFIG;

/*********************Function*****************************************
; Function Name: COUNTER_INIT
; Inputs: Char c
; Outputs: No outputs
*/
void COUNTER_INIT(void){
   // SET top of counter
   TCC0.PER = TC_PER_CONFIG;

   //Return from function
   return;
}



/*********************Function*****************************************
; Function Name: COUNTER_START
; Inputs: No inputs
; Outputs: No outputs
*/
void COUNTER_START(void){
   //SET interupt for counter
   TCC0.INTCTRLA = INTCTRLA_CONFIG;

   //Start COUNTER
   TCC0_CTRLA = CLK_DIV_CONFIG;

   //Return from function
   return;
}
```

```
/*********************Function*****************************************
; Function Name: COUNTER_STOP
; Inputs: No inputs
; Outputs: No outputs
*/
void COUNTER_STOP(void){

   //STOP TIMER INTERRUPT
   TCC0.INTCTRLA = 0x00;

   //STOP TIMER
   TCC0.CTRLA = 0x00;

   //RESET TIMER
   TCC0_CTRLFCLR = TC_CMD_RESTART_gc;

   //Return from function
   return;
}



/*********************Function*****************************************
; Function Name: COUNTER_STOP
; Inputs: uint8_t:input
; Outputs: No outputs
*/
void UPDATE_PER(uint8_t input){
   //Stop Timer
   COUNTER_STOP();

   //Update PER (Hz)
   TCC0.PER = (TC_PER_CONFIG/input);

   //Start Timer
   COUNTER_START();

   //Return from function
   return;
}


/*********************ISR*****************************************
; ISR TYPE: TCC0_OVF_vect
; Inputs: NO inputs
; Outputs: No outputs
*/
ISR(TCC0_OVF_vect){
   //Preserve Status Reg
   uint8_t temp = CPU_SREG;
```

```
    //Clear interrupt flags
    TCC0.INTFLAGS = 0x01;

    //Restore Status Reg
    CPU_SREG = temp;

    //Return from ISR
    return;

}
```

Code for TIMER_COUNTER.c (Part B):

```
/*
Name: Michael Arboleda
Section #: 7F34
TA Name: Wesley Piard
Description: Functions for TIMER_COUNTER

TIMER_COUNTER.c
Created: 7/25/2017 2:21:42 AM
*/

#include <avr/io.h>
#include <avr/interrupt.h>
#include "DAC.h"

//Include extern constants
extern const uint16_t TC_PER_CONFIG;
extern const uint8_t INTCTRLA_CONFIG;
extern const uint8_t CLK_DIV_CONFIG;
extern volatile uint16_t sine[];

//Global Varibles
volatile uint8_t counter = 0;

/*********************Function*****************************************
; Function Name: COUNTER_INIT
; Inputs: No direct input
; Outputs: No direct outputs
*/
void COUNTER_INIT(void){
   // SET top of counter
   TCC0.PER = TC_PER_CONFIG;

   return;
}



/*********************Function*****************************************
; Function Name: COUNTER_START
; Inputs: No direct input
; Outputs: No direct outputs
*/
void COUNTER_START(void){
   //SET interupt for counter
   TCC0.INTCTRLA = INTCTRLA_CONFIG;

   //Start COUNTER
   TCC0_CTRLA = CLK_DIV_CONFIG;

   //Return from function
```

```
   return;
}

/*********************ISR*************************************
; ISR TYPE: TCC0_OVF_vect
; Inputs: No direct input
; Outputs: No direct outputs
*/
ISR(TCC0_OVF_vect){
   //Preserve Status Reg
   uint8_t temp = CPU_SREG;

   //Change output
   UPDATE_DACA_CH0(sine[counter++]);

   //IF END OF SINE TABLE
   if(counter == 100){
      //Reset Counter
      counter = 0;
   }

   //Clear interrupt flags
   TCC0.INTFLAGS = 0x01;

   //Restore Status Reg
   CPU_SREG = temp;

   //Return from ISR
   return;

}
```

Code for USART.h:

```
#ifndef USART_H_
#define USART_H_

/*


USART.h
Created: 7/26/2017 4:37:25 AM
*/

//Function Prototypes
void USART_INIT(void);
void OUT_CHAR(char c);
void OUT_STRING(char* str);
char IN_CHAR(void);




#endif /* USART_H_ */
```

Code for USART.c:

```
/*
Name: Michael Arboleda
Section #: 7F34
TA Name: Wesley Piard
Description: Set up USART

USART.c
Created: 7/26/2017 4:34:28 AM
*/


#include <avr/io.h>

extern const uint8_t pin_Tx;
extern const uint8_t pin_Rx;
extern const uint8_t TxRx_On;
extern const uint8_t usart_ctrl_C;
extern const uint8_t BSCALE;
extern const uint8_t upper_BSEL;
extern const uint8_t BSEL;


/*********************Function*****************************************
; Function Name: USART_INIT
; Inputs: No inputs
; Outputs: No outputs
*/
void USART_INIT(void){
   // Set port D for USART com
   PORTD.DIRSET = pin_Tx;
   PORTD.OUTSET = pin_Tx;
   PORTD.DIRCLR = pin_Rx;

   //Set up Ctrl B and C
   USARTD0_CTRLB = TxRx_On;
   USARTD0_CTRLC = usart_ctrl_C;

   //Set up baud rate
   USARTD0.BAUDCTRLA = BSEL;
   USARTD0.BAUDCTRLB = (BSCALE | upper_BSEL);

   return;
}



/*********************Function*****************************************
; Function Name: OUT_CHAR
; Inputs: Char c
; Outputs: No outputs
```

```
*/
void OUT_CHAR(char c){

   // Wait until prev receive done
   while((USARTD0.STATUS & PIN5_bm) == 0);

   //Output char thru USART
   USARTD0.DATA = c;

   return;
}



/*********************Function*****************************************
; Function Name: OUT_STRING
; Inputs: Char pointer c
; Outputs: No outputs
*/
void OUT_STRING(char* str){
   //While char is not null
   while(*str){
      //Output char
      OUT_CHAR(*str++);
   }
   return;
}



/*********************Function*****************************************
; Function Name: IN_CHAR
; Inputs: Char c
; Outputs: No outputs
*/
char IN_CHAR(void){
   while((USARTD0.STATUS & PIN7_bm) == 0);
   return USARTD0_DATA;
}
```

Code for constants.h:

```
#ifndef CONSTANTS_H_
#define CONSTANTS_H_

/*
Name: Michael Arboleda
Section #: 7F34
TA Name: Wesley Piard
Description: Changes uP freq to 32Mhz

constants.h
Created: 7/7/2017 11:53:20 PM
*/
#include <avr/io.h>

//OVERALL DEFS
#define TRUE 1
#define FALSE 0
#define NULL 0

//****************** CLK_32MHz.c ***********************************
const uint8_t NEW_CLOCK_FREQ = 0b00000010;

//****************** DAC.c *****************************************
const uint8_t DAC_CTRLA_CONFIG = 0b00000101;//0x05
const uint8_t DAC_CTRLC_CONFIG = 0b00011000;//0x18


//****************** TIMER_COUNTER.c *******************************
const uint16_t TC_PER_CONFIG = 822;//50hz
const uint8_t INTCTRLA_CONFIG = 0b00000011;//0x03
const uint8_t CLK_DIV_CONFIG = 0b00000100;//0x04


//****************** DMA.c *****************************************
const uint8_t DMA_CTRL_CONFIG = 0b10000000;//0x80
const uint8_t DMA_CH0_ADDRCTRL_CONFIG = 0b01011001 ;//0x59
const uint8_t DMA_CH0_TRIGSRC_CONFIG = 0b01000000;//0x40
const uint8_t DMA_CH0_TRFCNT_CONFIG = 200;
const uint8_t DMA_CH0_REPCNT_CONFIG = 0x00;
const uint8_t DMA_CH0_CTRLA_CONFIG = 0b10100101;//0xA5


//****************** USART.h ***************************************
const uint8_t pin_Tx = PIN3_bm;
const uint8_t pin_Rx = PIN2_bm;
const uint8_t TxRx_On = 0b00011000; // 0x18
// asynch, 8 databits, no parity, 1 start, and 1 stop
const uint8_t usart_ctrl_C = 0b00000011;
// BSEL 28,800
const uint8_t upper_BSEL = 0b00001000;
```

```
const uint8_t BSEL = 0b10001110;
const uint8_t BSCALE = 0b10110000; // -5 BSCALE


// SINEWAVE FROM http:/ /www.daycounter.com/Calculators/Sine-GeneratorCalculator.phtml
// PARAMETERS: 100 POINTS, AMPLITUDE FFF(4096)
uint16_t sine[] = {0x800,0x881,0x901,0x980,0x9fd,0xa79,0xaf2,0xb68,0xbdb,0xc49,
   0xcb4,0xd19,0xd7a,0xdd5,0xe2a,0xe79,0xec1,0xf03,0xf3d,0xf70,
   0xf9c,0xfc0,0xfdc,0xff0,0xffc,0xfff,0xffc,0xff0,0xfdc,0xfc0,
   0xf9c,0xf70,0xf3d,0xf03,0xec1,0xe79,0xe2a,0xdd5,0xd7a,0xd19,
   0xcb4,0xc49,0xbdb,0xb68,0xaf2,0xa79,0x9fd,0x980,0x901,0x881,
   0x800,0x77f,0x6ff,0x680,0x603,0x587,0x50e,0x498,0x425,0x3b7,
   0x34c,0x2e7,0x286,0x22b,0x1d6,0x187,0x13f,0xfd,0xc3,0x90,
   0x64,0x40,0x24,0x10,0x4,0x0,0x4,0x10,0x24,0x40,
   0x64,0x90,0xc3,0xfd,0x13f,0x187,0x1d6,0x22b,0x286,0x2e7,
0x34c,0x3b7,0x425,0x498,0x50e,0x587,0x603,0x680,0x6ff,0x77F};

//TRIANGLE WAVE FROM http:/ /www.daycounter.com/Calculators/Triangle-WaveGenerator
//-Calculator.phtml
//PARAMETERS: 100 POINTS, AMPLITUDE FFF(4096)
uint16_t triangle[] = {0x52,0xa4,0xf6,0x148,0x19a,0x1ec,0x23d,0x28f,0x2e1,0x333,
   0x385,0x3d7,0x429,0x47b,0x4cd,0x51f,0x571,0x5c3,0x614,0x666,
   0x6b8,0x70a,0x75c,0x7ae,0x800,0x852,0x8a4,0x8f6,0x948,0x99a,
   0x9ec,0xa3d,0xa8f,0xae1,0xb33,0xb85,0xbd7,0xc29,0xc7b,0xccd,
   0xd1f,0xd71,0xdc3,0xe14,0xe66,0xeb8,0xf0a,0xf5c,0xfae,0xfff,
   0xfae,0xf5c,0xf0a,0xeb8,0xe66,0xe14,0xdc3,0xd71,0xd1f,0xccd,
   0xc7b,0xc29,0xbd7,0xb85,0xb33,0xae1,0xa8f,0xa3d,0x9ec,0x99a,
   0x948,0x8f6,0x8a4,0x852,0x800,0x7ae,0x75c,0x70a,0x6b8,0x666,
   0x614,0x5c3,0x571,0x51f,0x4cd,0x47b,0x429,0x3d7,0x385,0x333,
0x2e1,0x28f,0x23d,0x1ec,0x19a,0x148,0xf6,0xa4,0x52,0x0};

#endif /* CONSTANTS_H_ */
```