ECE 4750: Digital Signal Processing Lab

Project 3: CCS Introduction

Angel Rodriguez

Utah Valley University

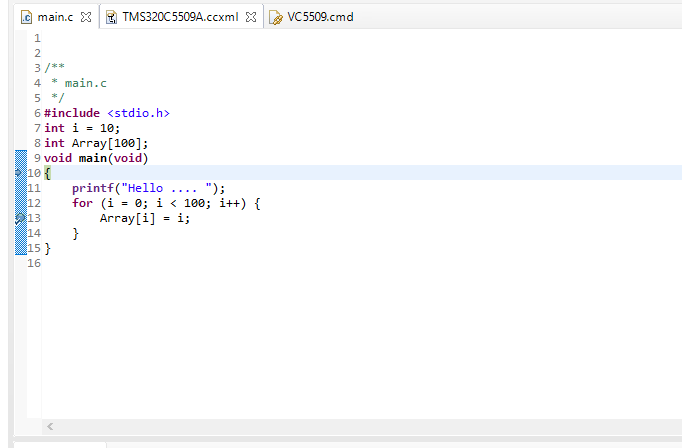
# Introduction

The objective of this lab is to learn how to use Code Composer Studio (CCS) to implement convolution in C. This project is focused on the basics of CCS and how to load the code from the computer into a TI DSP memory. Voice recordings of the input signal x[n] will be convolved with the impulse response y[n] to derive an output y[n] = x[n] \* h[n]. X[n] will include the recordings of our own voice within an area

# Process

The process of completing the lab includes connecting DSPLab to PC, installing the C5500 compiler, creating a new project, creating a simple C code in CSS, running the code, and the debugging features of CSS. After connecting the equipment to the computer via USB, the CCS program was initialized. A new project was created with a default main.c file which was edited as shown in Figure 1. The simple C code in CSS was run using the build button and with the code shown in Figure 1. The debugger was run with a breakpoint on line 13. The expressions for the Array are depicted on Figure 2. It seems that the expressions include the name of the variable, the value stored in the variable, and the memory address of that variable. The memory browser shown in Figure 3 shows the addresses of the array values. Here it can be seen that you can find the memory address of Array by using “&Array”. The address shown is 0x00525C. I decided to go ahead and type 0x00525C, and sure enough, the same information was shown. The results are shown in Figure 4. In both figures 3 and 4, a couple of things are highlighted: “0000” and the memory address “0x00525C”. I figured that based on this as well as Figure 2, the memory address of Array is the same address as Array[0]. Based on this information, I can conclude that Array points to the first element of Array since their memory addresses are the same. My thought process predicted the address of Array[1], the element after Array[0]. If the address of Array[0] is 0x00525C, the address of Array[1] should be the next hexadecimal number, 0x00525D. Checking Figure 2 confirmed this to be true. Next step would be to find out of entering “0x00525D” in the memory browser would highlight the value after 0000 which would be 0001 in Figure 4. Entering “0x00525D” in the memory browser highlights “0001” as shown in Figure 5. Data shown in the figures were from different times of the debugging process, so there may be slight variation in the values shown in the expressions window and the memory browser window. Looking at both in the same time frame should give the same values for all correlated memory addressed. This can be seen in Figure 6. Notice that the figure has been edited with color-coded mark-ups to indicate corresponding values for 3 different memory addresses/array elements.

# Figures

  
Figure 1:  
Code for main.c

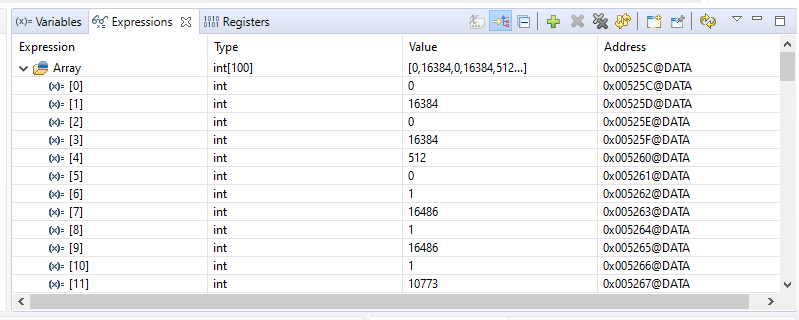


Figure 2:  
Expressions for Array depicting values of Array

Graphical user interface, text, application

Description automatically generated

Figure 3:  
Memory browser depicting Array memory locations based on input &Array

Graphical user interface, text, application

Description automatically generated  
Figure 4:  
Memory browser depicting Array memory locations based on input 0x00525C

Graphical user interface, text, application

Description automatically generated  
Figure 5:  
Memory browser depicting Array memory locations based on input 0x00525D

Graphical user interface, application

Description automatically generated  
Figure 6:  
Memory browser and expressions edited with color-coded mark-ups to indicate corresponding values for 3 different memory addresses/array elements.

# Conclusion

The lab ends with learning the manuals and working with DSPLab unit. Concepts learned include: connecting DSPLab to PC, installing the C5500 compiler, creating a new project, creating a simple C code in CSS, running the code, and the debugging features of CSS.

# Attachments

C Code of file main.c :

/\*\*

\* main.c

\*/

**#include** <stdio.h>

**int** i = 10;

**int** Array[100];

**void** **main**(**void**)

{

**printf**("Hello .... ");

**for** (i = 0; i < 100; i++) {

Array[i] = i;

}

}