Microprocessor Systems

ELE 271

Laboratory 9:

Reading ADC Inputs Through Polling

Due 10APR2023

By: Nick Bean ,Taylor Brookes, Toussaint Falangani

**Introduction**

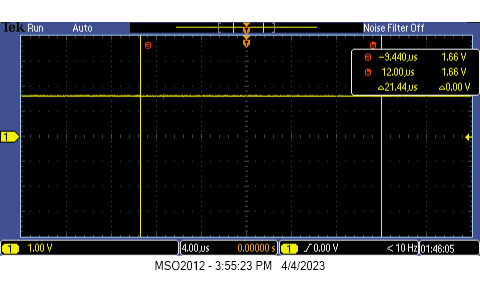
Our goal for this lab was to read the ADC inputs via polling.

* In Part 1 we copied the code directly from the given main.c file and ran it on Keil. We were able to get the program to run without any changes. Then we used the PB505 to output 3.3V across PC0 and showed that having the variable “reading” = 2048 triggered the LED at an input voltage of 1.66V on the oscilloscope on pin PC0. This occurs at 2048 because the ADC is a 12-bit value, where 2^12 = 4096. 2048 is half of 4096, and therefore should toggle the LED at 3.3V / 2 = 1.66V.
* In part 2 we created a buffer/array of ints (ints are 32-bit) to store the high 12-bit value (around AA8 hex or 2728 in decimal) in 8 hex values (ex: 0000 0AA8) (32-bit) 10 times and then store the low 12-bit value (around 80 hex or 128 in decimal) in 8 hex values (ex: 0000 0080) 10 times, repeatedly. We did this for both 100Hz and 10Hz input square waves. With our own method, we were able to accomplish the task mentioned above. 100Hz required waiting 20 counts per sample to get 10 high and 10 low. “sample” = 25 had not enough samples (7 and 8 of each) while “sample” = 10 had too many, and “sample” = 20 had 10 and 11 of each which is close enough to 10 of each. 10Hz required waiting 200 units per sample to get 10 high and 10 low.

**Part 1**

In Part 1, we are tasked with outputting the voltage that is required to turn on the LED for pin PC0. We can calculate this to be 3.3V / 2 = 1.66V, and can experimentally prove it by showing that at 1.66V the LED begins to turn on.

[image of 1.66V output that LED triggers at]



**Part 2**

In Part 2, we are tasked with creating an int array that stores 10 high and 10 low samples per cycle of 100Hz and 10Hz. We achieved this by writing a new dead loop function to replace the while(1) loop. We achieved 10 high and 10 low samples when sample = 20 for 100Hz and sample = 200 for 10Hz. The variable place is the running length of our int array, and stops the loop when we have reached the predetermined end of the array. We initialized the array outside of the main method to store the start of the value near 0x2000 0000 as:

int values[1000] = {0};

1000 was an arbitrary choice of the length of the array. The variable “place” runs the function until the end of the array is reached. The variable “reading” is the value of the ADC at that cycle. Here is our replacement dead loop function:

int sample = 0;

int place = 0;

unsigned int reading;

while (place <= 999){

reading = adc\_read();

if (sample == 200) {

values[place] = reading;

place++;

sample = 0;

}

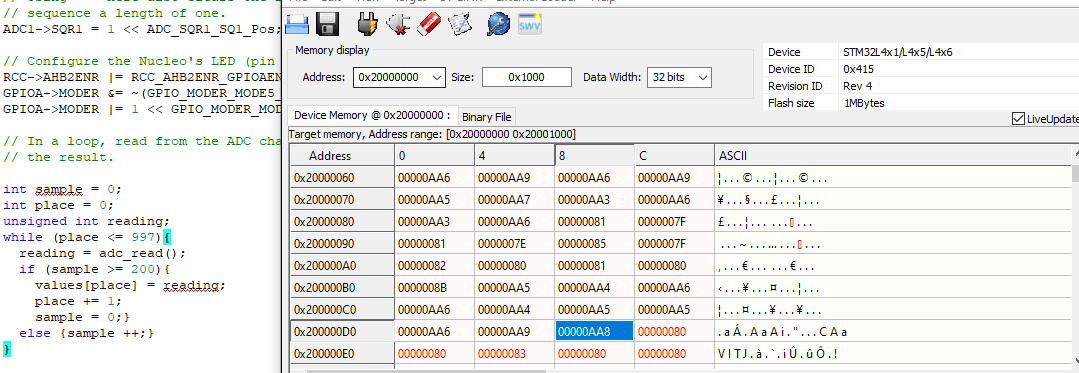
else {

sample++;

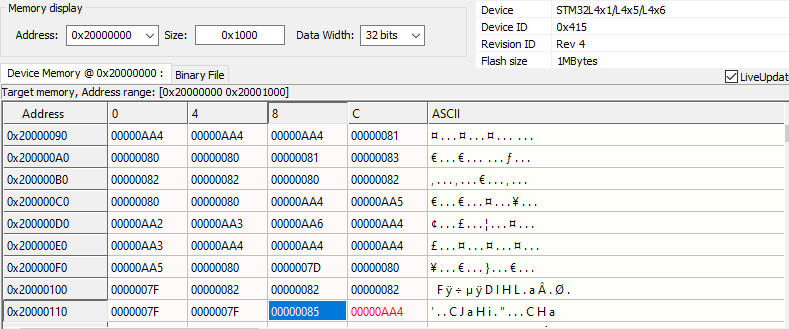
}

}

[image of 10 high and 10 low values at 10Hz, sample rate of once every 200 cycles]



[image of 10 high and 10 low values at 100Hz, sample rate of once every 20 cycles]



**Conclusion**

Our goal for this lab was to read the ADC inputs via polling. Part 1 was to show that the LED toggled when “reading” = 2048 at 1.66V, which is ½ of the ADC’s 12-bit max value and ½ of the 3.3V input. Part 2 stored 10 high and 10 low values in a buffer over 100Hz and 10Hz input square waves at “sample” = 20 and 200 respectively. 10Hz requires 1 sample every 200 cycles, which is less than that of 100Hz because 10Hz has a period 10x longer than 100Hz.