Appendix A

LUA Program Code Used in SIMION8.0 for Time Varying Electric Field

```
simion.workbench_program()
    adjustable switch_time = 17
    adjustable plate_voltage = 10
    functionsegment.tstep_adjust()
    if ion_time_of_flight < switch_time then
    ion_time_step = min(ion_time_step, switch_time- ion_time_of_flight)
    end
    end
    functionsegment.fast_adjust()
    if ion_time_of_flight < switch_time then
    adj_elect02 = plate_voltage
    else adj_elect02 = 0
    end
    end
    do
    functionsegment.other_actions()
    if switch_time == ion_time_of_flight then
    sim_update_pe_surface = 1
    end
    end
    end
```

Appendix B

Arduino Program for Electron Spectrometer Operation and DAQ Automation

```
// Set 'TOP' for PWM resolution. Assumes 16 MHz clock.
    const unsigned int TOP = 0xFFFF; // 16-bit resolution. 244 Hz PWM
    // const unsigned int TOP = 0x7FFF; // 15-bit resolution. 488 Hz PWM
    // const unsigned int TOP = 0x3FFF; // 14-bit resolution. 976 Hz PWM
    // const unsigned int TOP = 0x1FFF; // 13-bit resolution. 1953 Hz PWM
    // const unsigned int TOP = 0x0FFF; // 12-bit resolution. 3906 Hz PWM
    // const unsigned int TOP = 0x07FF; // 11-bit resolution. 7812 Hz PWM
    // const unsigned int TOP = 0x03FF; // 10-bit resolution. 15624 Hz PWM
    define pwmPin 9
    long pulseCount = 0;
    float inputVoltage = 0;
    float delayTime = 120;//Delay time in seconds
    void PWM16Begin()
    // Stop Timer/Counter1
    TCCR1A = 0; // Timer/Counter1 Control Register A
    TCCR1B = 0; // Timer/Counter1 Control Register B
    TIMSK1 = 0; // Timer/Counter1 Interrupt Mask Register
    TIFR1 = 0; // Timer/Counter1 Interrupt Flag Register
    ICR1 = TOP;
    OCR1A = 0; // Default to 0
    OCR1B = 0; // Default to 0
```

```
// Set clock prescale to 1 for maximum PWM frequency TCCR1B |= (1 « CS10);
// Set to Timer/Counter1 to Waveform Generation Mode 14: Fast PWM with TOP set by ICR1
TCCR1A |= (1 « WGM11);
TCCR1B |= (1 « WGM13) | (1 « WGM12) ;
void PWM16EnableA()
// Enable Fast PWM on Pin 9: Set OC1A at BOTTOM and clear OC1A on OCR1A compare
TCCR1A |= (1 « COM1A1);
pinMode(9, OUTPUT);
inline void PWM16A(unsigned int PWMValue)
OCR1A = constrain(PWMValue, 0, TOP);
void setup()
Serial.begin(9600);
PWM16Begin();
// On the Arduino UNO T1A is Pin 9 and T1B is Pin 10
PWM16A(0); // Set initial PWM value for Pin 9
PWM16EnableA(); // Turn PWM on for Pin 9
attachInterrupt(0, handlePulse, RISING); //Connect the pulse counter to pin 2
Serial.begin(9600);
Serial.println(); //Required to start
void loop()
//testingCounts();
//Serial.print statements are for debugging ( except last one )
if(Serial.available())
inputVoltage = Serial.parseFloat();
//Serial.print("Voltage Input: ");
```

```
//Serial.println(inputVoltage,4);
long pwm = (long)(inputVoltage*65535/10);
//Serial.print("Pwm: ");
//Serial.println(pwm);
PWM16A(pwm);
//analogWrite(pwmPin,pwm);
delay(2000);//Required for the voltage output through RC filter to reach steady state
//Serial.println("Output voltage set. Now counting pulses");
//Now start counting pulses
pulseCount = 0;
delay(delayTime*1000); //Total time delay. Pulses are being counted during this time
Serial.print(""); //Required for some reason we dont know
Serial.println(String(pulseCount));
void testingCounts()
Serial.println(pulseCount);
delay(10);
void handlePulse()
pulseCount = pulseCount +1;
```

MATLAB Code for DAQ:

This sends voltage commands to Arduino for electron spectroscopy experiment, and receives the counts detected

```
close all; clear all;
s = serial('COM8');
set(s,'BaudRate',9600);
fopen(s);
V_array = 3.7:0.01:4.3;
```

```
counts_array = 0 * V_array ;
fgetl(s);
fori = 1 : length(V_array)
fprintf(s, num2str(V_array(i)));
pause(20);
counts_array(i) = str2num(fgetl(s));
end
fclose(s);
delete(s);
dlmwrite('xenon210220171718.txt', [V_array'counts_array'])
```

Appendix C

Function Fit Program for Metastable Decay Life-time Measurement

```
float timetochannel=ROOT::TMath::Power(10,-12)*25;
    float mass1=15.0;
    float mass2=87.0;
    float tau=150*ROOT::TMath::Power(10,-9);
    float tauchannel=tau/timetochannel;
    //(1/(160*1000))*70;
    TF1 f2 ( "f2","exp(-(x/[2])*(([0]+[1])/([0]-[1])))" ,45000,68000); //function
    f2->SetParameter(0,mass1);
    f2->SetParameter(1,mass2);
    f2->SetParameter(2,tauchannel);
    float constant=f2->GetMaximum();
    float counts=68;
    TF1 f ( "f","([4]/[3])*exp(-(x/[2])*(([0]+[1])/([0]-[1])))" ,45000,68000); //function
    f->SetParameter(0,mass1);
    f->SetParameter(1,mass2);
    f->SetParameter(2,tauchannel);
    f->SetParameter(3,constant);
    f->SetParameter(4,counts);
```