JOINT INSTITUTE FOR NUCLEAR RESEARCH
Dzhelepov Laboratory of Nuclear Problems

**FINAL REPORT ON THE START PROGRAMME**

# *Measurement of main characteristics for different types of semiconductor radiation detectors*

**Supervisor:**
Dr. Uladzimir Kruchonak

**Student:**
Luparau Anatoli, Belarus
Gomel State Technical University

**Participation period:**
July 07 – August 17,
Summer Session 2024

Dubna, 2024

## Contents

# 1 Introduction

## 1.1 Overview of Semiconductor Radiation Detectors

Semiconductor radiation detectors are essential tools in various fields, including particle physics, medical imaging, and environmental monitoring. These detectors, such as those made from materials like GaAs, Si, and CdZnTe, are highly sensitive to different types of radiation, including particles and photons. The ability to accurately measure and analyze the signals generated by these detectors is crucial for understanding their performance and optimizing their use in practical applications. Despite their widespread use, challenges remain in accurately characterizing the signal responses and electrical properties of these detectors, particularly in terms of signal amplitude, duration, and the carrier transport properties within the semiconductor material.

## 1.2 Objectives

The primary objective of this study was to measure and analyze the key characteristics of semiconductor radiation detectors, specifically the signal amplitude and duration when exposed to light pulses. These measurements were used to calculate the charge carrier mobility and lifetime within the detectors. Additionally, the study aimed to assess the integrity of detector matrices by measuring the current-voltage (I-V) characteristics of wafers at their edges. This work involved the design and implementation of an experimental setup, data acquisition using an oscilloscope, and data analysis using ROOT CERN software [1].

# 2 Experimental setup and procedure

## 2.1 Scope of work

The scope of this work involved the following tasks:

Assemble the Experimental Setup: Design and build a circuit incorporating a transistor, a current-limiting resistor, and an LED to measure the signal characteristics of semiconductor detectors. The setup was required to generate light pulses with precise frequency and pulse width.

Measure Signal Characteristics: Perform measurements to determine the amplitude and duration of the signals generated by GaAs, Si, and CdZnTe detectors when exposed to light pulses. This required using an oscilloscope to capture and record the signals.

Calculate Charge Carrier Properties: Analyze the measured data to calculate the charge carrier mobility and lifetime within the detectors. This involved fitting the data to appropriate equations and using software tools for analysis.

Conduct Wafer Measurements: Measure the current-voltage (I-V) characteristics of detector wafers in a prepared measurement chamber. Calculate the average resistance of the wafers to assess their quality and uniformity.

## 2.2 Circuit configuration of LED pulses generator

For this setup, an N-channel MOSFET RD01MUS2B [2] was chosen due to its high-frequency characteristics. The assembled circuit is shown in Figure 1.

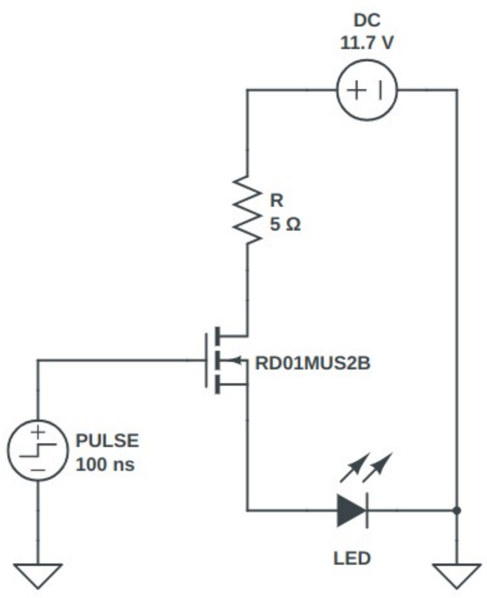
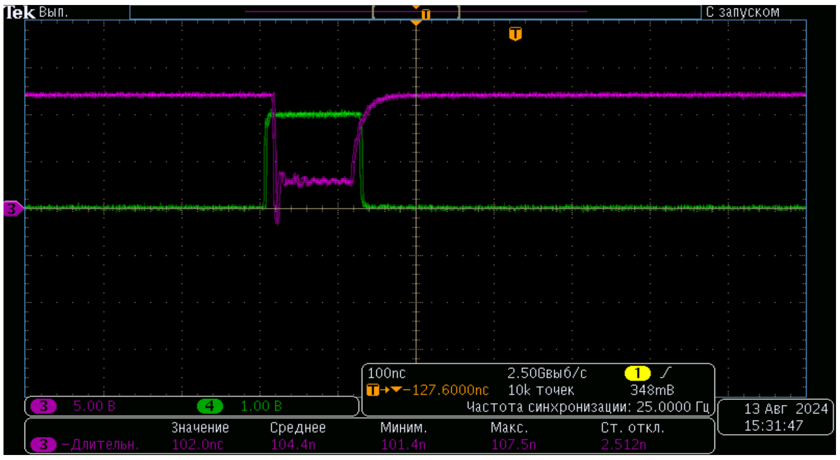


Fig. 1. Scheme for the experiment

The signal generator was set to a frequency of 25 Hz with a pulse duration of 100 ns. The voltage source was set to 11.7 V, and the resistor had a resistance of 5 Ohm.

The oscilloscope trace in Figure 2 indicates that the setup is functioning correctly. The voltage across the LED is shown in purple, while the signal from the generator is displayed in green.



| | Значение | Среднее | Миним. | Макс. | Ст. откл. |
|---|---|---|---|---|---|
| 3 -Длительн. | 102.0нс | 104.4н | 101.4н | 107.5н | 2.512н |

Fig. 2. Oscilloscope trace from the LED, (3) — Signal from LED, (4) — Signal from generator

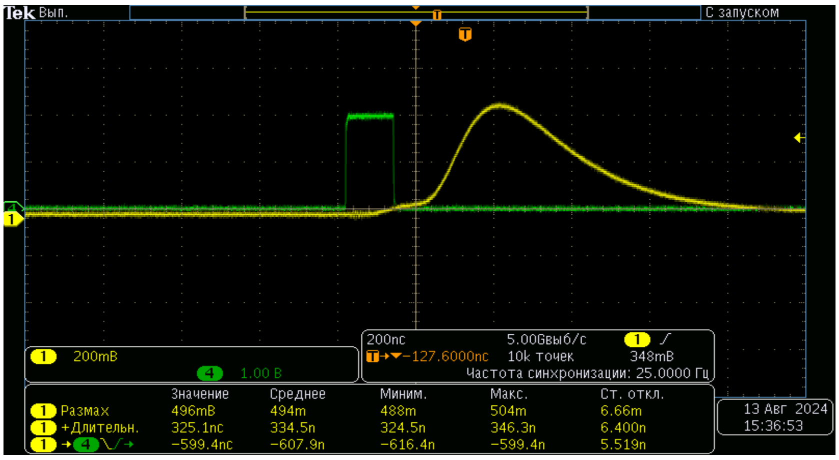
A voltage is applied to the GaAs, Si, and CdZnTe detectors, and the signal amplitude and duration are measured through the amplifier using an oscilloscope.

### 2.3 Measurement of LED pulses response

The measurement process involved applying a bias voltage to the detector and illuminating it with an LED. For each voltage level, I measured the signal amplitude, duration, and delay relative to the generator signal. These measurements were systematically recorded in a data table.

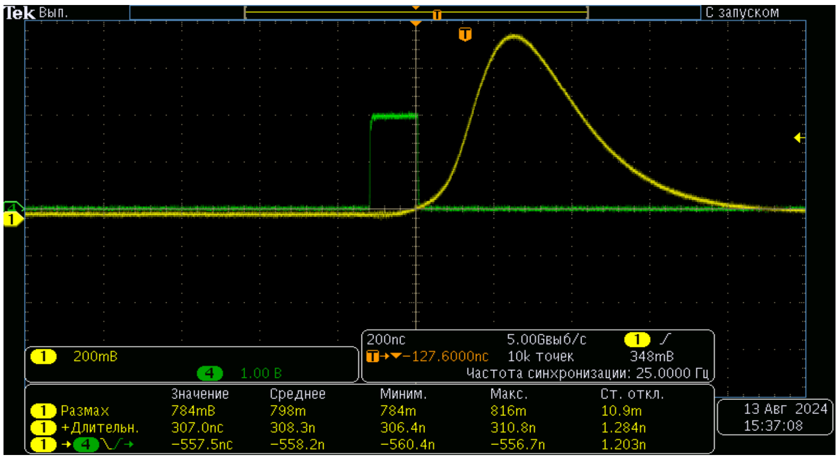
As the experiment progressed, the applied voltage to the detector was gradually increased. After each voltage increment, the same set of measurements was repeated, including the signal amplitude, duration, and delay. This process was continued until the detector reached its saturation point, where the signal characteristics no longer showed significant changes with further increases in voltage. The data collected during this process provided insights into the performance and response of the detectors under varying conditions.

The measurements are shown in Figures 3 and 4.



| | Значение | Среднее | Миним. | Макс. | Ст. откл. |
|---|---|---|---|---|---|
| 1 Размах | 496mB | 494m | 488m | 504m | 6.66m |
| 1 +Длительн. | 325.1nc | 334.5n | 324.5n | 346.3n | 6.400n |
| 1 → 4 | -599.4nc | -607.9n | -616.4n | -599.4n | 5.519n |

Fig. 3. Signal from the detector (yellow) and generator (green) at 100V applied to the detector



| | Значение | Среднее | Миним. | Макс. | Ст. откл. |
|---|---|---|---|---|---|
| 1 Размах | 784mB | 798m | 784m | 816m | 10.9m |
| 1 +Длительн. | 307.0nc | 308.3n | 306.4n | 310.8n | 1.284n |
| 1 → 4 | -557.5nc | -558.2n | -560.4n | -556.7n | 1.203n |

Fig. 4. Signal from the detector (yellow) and generator (green) at 150V applied to the detector

The recorded data and the standard deviation are presented in Table 1. Probably, this behavior indicates the onset of detector saturation. While the signal amplitude continues to increase, the duration of the signal stops decreasing and starts to rise. This suggests that the detector is no longer operating in its linear range, and its response becomes nonlinear due to the accumulation of excess charge.

| U, V | Signal, mV | Sigma | Width, ns | Sigma | Delay, ns | Sigma |
|---|---|---|---|---|---|---|
| 30 | 22 | 0.9 | 875 | 4.6 | 916 | 4.2 |
| 35 | 25.2 | 0.7 | 798 | 3.9 | 975 | 2.9 |
| 40 | 32.9 | 0.2 | 702 | 4.8 | 1032 | 3.2 |
| 45 | 41.7 | 0.4 | 619 | 1.6 | 1098 | 2.7 |
| 50 | 52.4 | 0.3 | 566 | 1.1 | 1159 | 2.9 |
| 60 | 74.4 | 0.4 | 493 | 1.4 | 1241 | 1.4 |
| 70 | 97.1 | 0.4 | 447 | 0.6 | 1295 | 0.4 |
| 80 | 120 | 0.5 | 428 | 0.8 | 1325 | 0.5 |
| 90 | 150 | 0.5 | 395 | 0.4 | 1362 | 0.5 |
| 100 | 174 | 0.7 | 372 | 0.7 | 1394 | 0.7 |
| 110 | 201 | 0.7 | 356 | 0.8 | 1413 | 0.3 |
| 120 | 230 | 0.6 | 343 | 0.2 | 1432 | 0.3 |
| 130 | 258 | 0.8 | 327 | 0.3 | 1452 | 0.5 |
| 140 | 302 | 1.1 | 321 | 0.3 | 1463 | 0.4 |
| 150 | 327 | 1 | 313 | 0.2 | 1475 | 0.2 |
| 160 | 357 | 0.7 | 305 | 0.1 | 1486 | 0.2 |
| 170 | 383 | 0.5 | 298 | 0.5 | 1496 | 0.6 |
| 180 | 424 | 1.1 | 290 | 0.3 | 1506 | 0.2 |
| 190 | 449 | 0.9 | 286 | 0.2 | 1513 | 0.2 |
| 200 | 475 | 0.7 | 283 | 0.2 | 1519 | 0.2 |
| 210 | 506 | 1 | 280 | 0.2 | 1524 | 0.2 |
| 220 | 535 | 0.7 | 276 | 0.2 | 1530 | 0.2 |
| 230 | 556 | 1 | 274 | 0.3 | 1534 | 0.2 |
| 240 | 586 | 0.9 | 274 | 0.2 | 1535 | 0.3 |
| 250 | 610 | 0.7 | 273 | 0.2 | 1540 | 0.2 |
| 260 | 641 | 0.8 | 269 | 0.3 | 1545 | 0.2 |
| 270 | 665 | 0.6 | 269 | 0.2 | 1547 | 0.2 |
| 280 | 703 | 0.8 | 266 | 0.2 | 1551 | 0.2 |
| 290 | 722 | 0.9 | 265 | 0.2 | 1554 | 0.2 |
| 300 | 748 | 1.1 | 263 | 0.1 | 1557 | 0.1 |
| 310 | 772 | 1.2 | 262 | 0.1 | 1559 | 0.1 |
| 320 | 792 | 1.3 | 261 | 0.1 | 1561 | 0.2 |

Tab. 1. Measurement Results for the CdZnTe Detector

## 2.4 Measurement of I-V characteristics of wafers

Regarding the I-V measurements, the setup is already prepared, and it is only necessary to place the wafers in the insulating chamber, move the needle across the contacts, and conduct the measurements using a computer program. The installed wafer is shown in Figure 5.

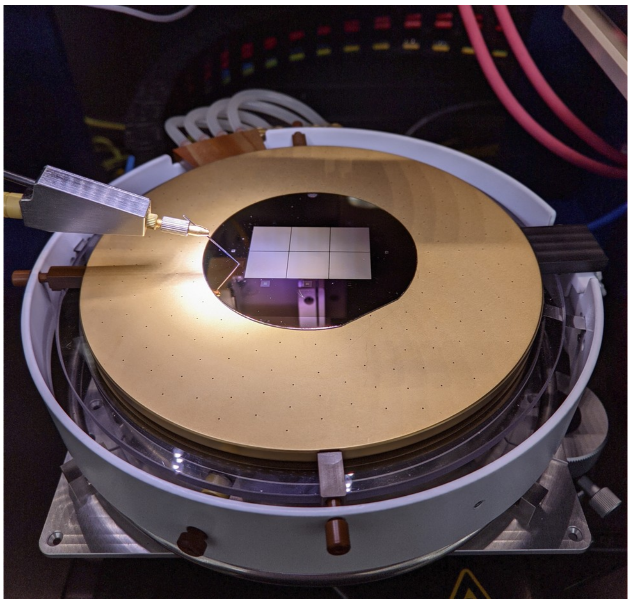
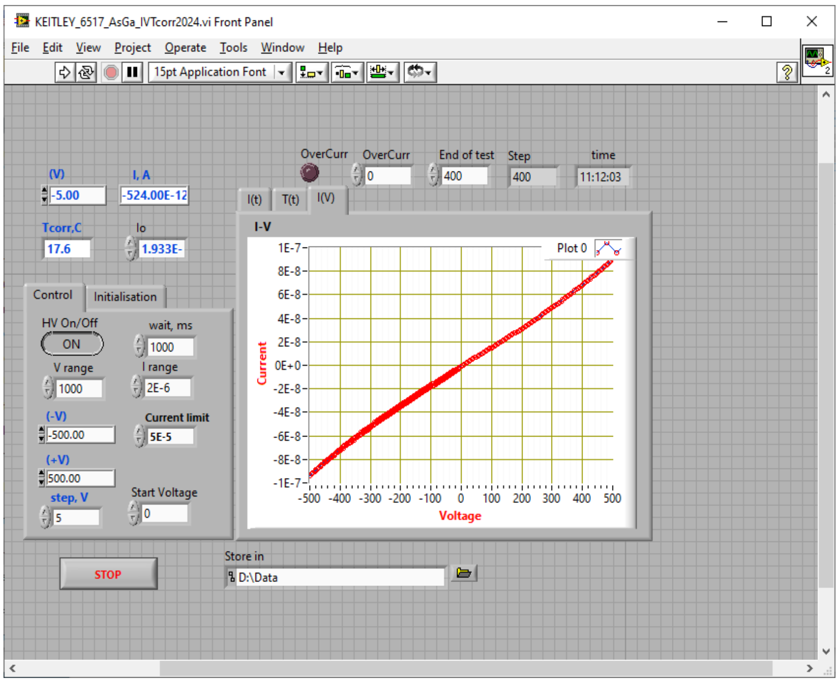


Fig. 5. Wafer AG-450#30 in the setup

Regarding the I-V characteristics measurements of the wafers, the results are shown in Figure 6.



KEITLEY_6517_AsGa_IVTcorr2024.vi Front Panel

(V) -5.00
I, A -524.00E-12
Tcorr,C 17.6
Io 1.933E-
OverCurr
OverCurr 0
End of test 400
Step 400
time 11:12:03

Control
HV On/Off ON
wait, ms 1000
V range 1000
I range 2E-6
(-V) -500.00
Current limit 5E-5
(+V) 500.00
step, V 5
Start Voltage 0

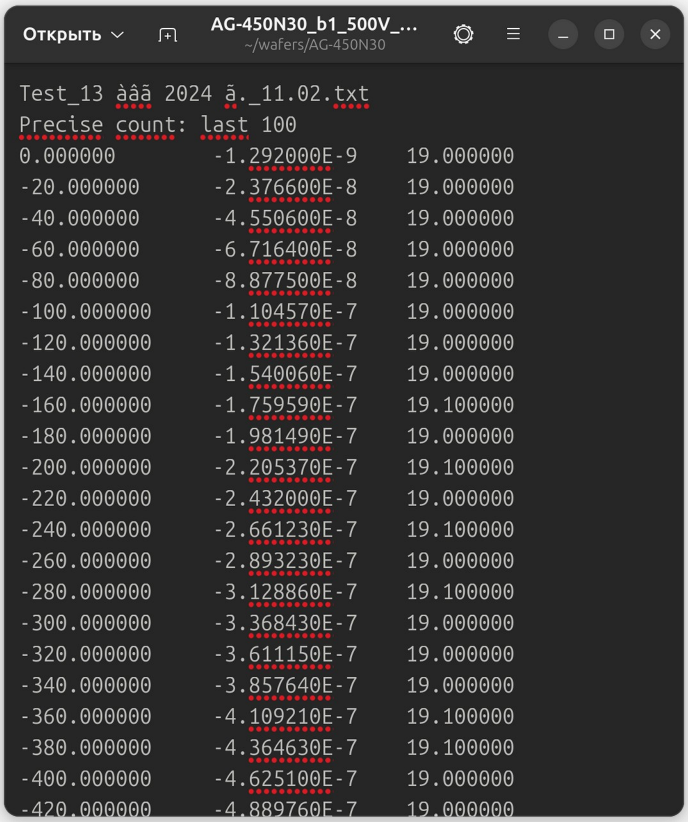
Store in D:\Data

Fig. 6. I-V characteristics of GaAs detector

On the wafers, measurements are taken at areas labeled "s" and "b". The measurement results are saved in a file named after the wafer, including the area label and number, as well as the voltage range.

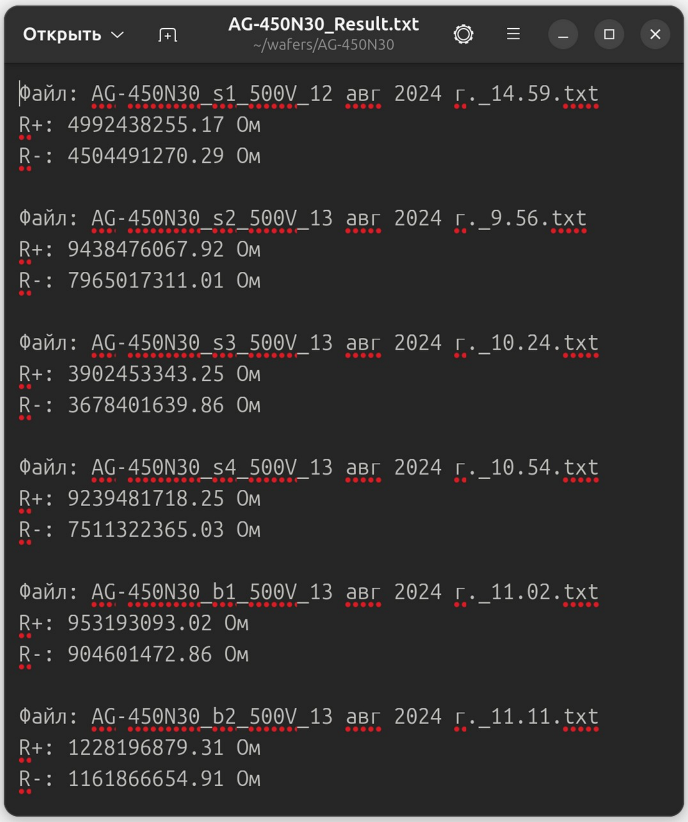
Calculations will be presented for the wafer AG-450#30, with a thickness of 520 µm. It is necessary to calculate the average resistance for U>0 and U<0 on each pad. For this purpose, a program was written that iterates through all the pads, with information stored in separate text files, and saves the result in a file named "Name_Result.txt". The code for the program is provided in Appendix 2.

The contents of the file with the I-V measurements are shown in Figure 7, with the first column representing voltage and the second column representing current. The program's operation is illustrated in Figure 8.



```
Test_13 àâã 2024 ã._11.02.txt
Precise count: last 100
0.000000        -1.292000E-9    19.000000
-20.000000      -2.376600E-8    19.000000
-40.000000      -4.550600E-8    19.000000
-60.000000      -6.716400E-8    19.000000
-80.000000      -8.877500E-8    19.000000
-100.000000     -1.104570E-7    19.000000
-120.000000     -1.321360E-7    19.000000
-140.000000     -1.540060E-7    19.000000
-160.000000     -1.759590E-7    19.100000
-180.000000     -1.981490E-7    19.000000
-200.000000     -2.205370E-7    19.100000
-220.000000     -2.432000E-7    19.000000
-240.000000     -2.661230E-7    19.100000
-260.000000     -2.893230E-7    19.000000
-280.000000     -3.128860E-7    19.100000
-300.000000     -3.368430E-7    19.000000
-320.000000     -3.611150E-7    19.000000
-340.000000     -3.857640E-7    19.000000
-360.000000     -4.109210E-7    19.100000
-380.000000     -4.364630E-7    19.100000
-400.000000     -4.625100E-7    19.000000
-420.000000     -4.889760E-7    19.000000
```

Fig. 7. Voltage and current of wafer AG-450#30



```
Файл: AG-450N30_s1_500V_12 авг 2024 г._14.59.txt
R+: 4992438255.17 Ом
R-: 4504491270.29 Ом

Файл: AG-450N30_s2_500V_13 авг 2024 г._9.56.txt
R+: 9438476067.92 Ом
R-: 7965017311.01 Ом

Файл: AG-450N30_s3_500V_13 авг 2024 г._10.24.txt
R+: 3902453343.25 Ом
R-: 3678401639.86 Ом

Файл: AG-450N30_s4_500V_13 авг 2024 г._10.54.txt
R+: 9239481718.25 Ом
R-: 7511322365.03 Ом

Файл: AG-450N30_b1_500V_13 авг 2024 г._11.02.txt
R+: 953193093.02 Ом
R-: 904601472.86 Ом

Файл: AG-450N30_b2_500V_13 авг 2024 г._11.11.txt
R+: 1228196879.31 Ом
R-: 1161866654.91 Ом
```

Fig. 8. Calculation of resistance for the wafer pads AG-450#30

This process is then repeated for the subsequent wafers. The program is versatile for such measurements, requiring only the format name as "Name_Pad_Voltage_DateTime.txt".

Let's calculate the resistivity using the formula $\rho = R \cdot \frac{S}{l}$ for both positive and negative voltage. For the calculation, we'll use a cross-sectional area (S) of 0,5cm*0,5cm, with the thickness (l) of our wafer being 520μm. For the s1 pad and positive voltage $\rho = 4992438255{,}17 \cdot \frac{0{,}5 \cdot 0{,}5}{0{,}052} = 24\ [G\Omega \cdot cm]$, for negative voltage $\rho = 4\,504491270{,}29 \cdot \frac{0{,}5 \cdot 0{,}5}{0{,}052} = 21{,}65\ [G\Omega \cdot cm]$. This result indicates the integrity of the wafer, as a typical resistivity ranges from $10^9$ to $10^{10}$ Ω*cm.

## 3 Calculation of charge carrier mobility and lifetime

Using ROOT CERN, I generated graphs from the data collected in the tables and fitted them with appropriate functions[3]. For the signal, the fitting function used was (par[1]·x[0]/par[0])·(1−exp(−par[0]/x[0])), and for the duration, the fitting function was par[0]/x[0]+par[1]. The formula used for fitting the signal was derived from the Hecht [4] equation:

$CCE = \frac{\mu \tau U}{d^2} \cdot \left(1 - e^{-\frac{d^2}{\mu \tau U}}\right)$, where:

- CCE is the collected charge signal,
- μ is the charge carrier mobility,
- τ is the charge carrier lifetime,
- U is the applied voltage,
- d is the thickness of the detector.

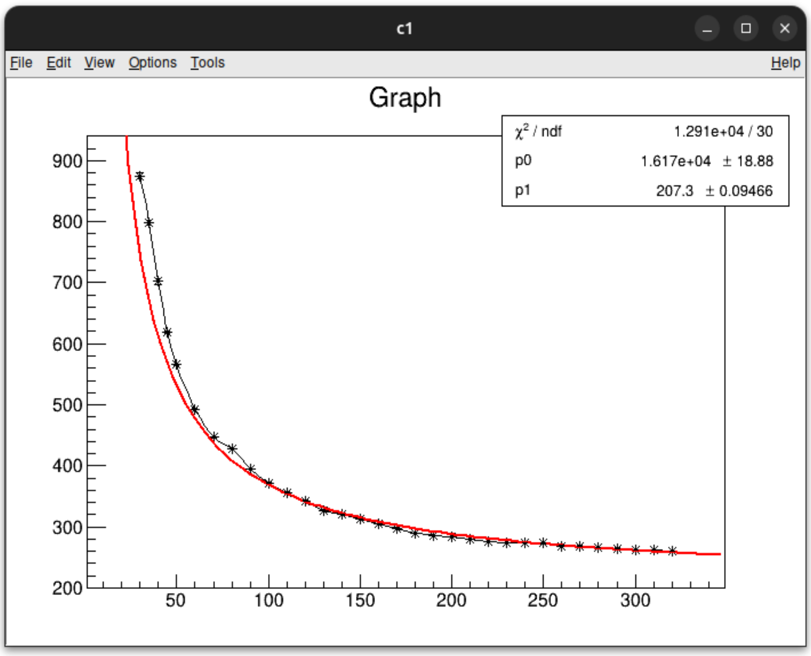
And for fitting the duration:

$t = \frac{d^2}{\mu U}$, where t is the time.

The par[0] values for the duration are divided by $10^9$, as the measurements were in nanoseconds and the calculations require seconds. The thickness d is kept in centimeters, as its unit of measurement does not affect the result.

Let us proceed with the calculation of μ and τ for the CdZnTe detector:

The plot and fitting of the duration for the CdZnTe detector in ROOT CERN are shown in Figure 9. The Root CERN code for the signal and time is presented in Appendix 1.



```
χ² / ndf    1.291e+04 / 30
p0          1.617e+04  ± 18.88
p1          207.3  ± 0.09466
```

Fig. 9. Fitting of duration

To determine μ, we use the formula $\mu = \frac{d^2}{par[0]}$.

The thickness of our CdZnTe is 900 μm, and the time is measured in nanoseconds.

Therefore: $\mu = \frac{0{,}09^2}{16170 \cdot 10^{-9}} = 500{,}927 \left\{\frac{cm^2}{V \cdot s}\right\}$

Now, to find τ, we need par[0] for the signal, with the fitting shown in Figure 10.

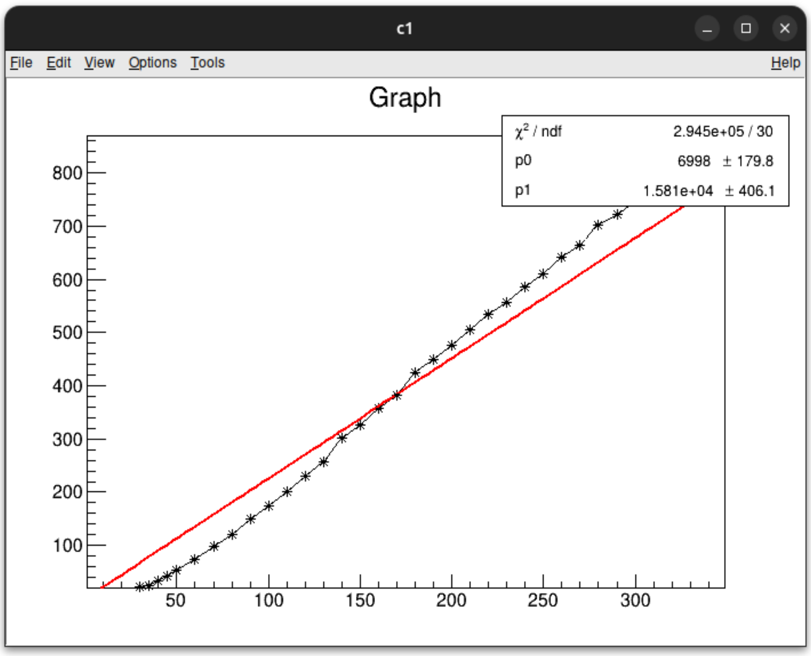


Fig. 10. Fitting of signal

$$\tau = \frac{d^2}{\mu \cdot par[0]} = \frac{0{,}09^2}{500{,}927 \cdot 6998} = 2{,}31 \{ns\}$$

Following this approach, the remaining detectors, Si and GaAs, can also be evaluated, as well as the Si detector for hole carriers (charge carriers). The results of these calculations are presented in Table 2.

| **Detector** | GaAs 500μm for electrons | Si 500μm for electrons | Si 500μm for holes | Si 500μm for holes Am241 alpha | CdZnTe 900μm for electrons |
|---|---|---|---|---|---|
| **Charge Carrier Mobility,** $\left\{\frac{cm^2}{V \cdot s}\right\}$ | 9920,005 | 648,624 | 632,608 | 601,279 | 500,927 |
| **Charge Carrier Lifetime, ns** | 4,425 | 16,996 | 18,611 | 182,547 | 2,31 |

Tab. 2. Calculation results

GaAs exhibits very high charge carrier mobility and short lifetime, indicating fast charge movement and rapid recombination.

The charge carrier mobility in silicon is lower than in GaAs, but the longer lifetime indicates a slower recombination process. The hole mobility is almost the same as that of electrons, but the lifetime is slightly longer, which is typical for silicon. When measured with americium, the hole mobility slightly decreases, but the charge carrier lifetime significantly increases, which is due to the effect of the radiation.

CdZnTe has the lowest mobility and charge carrier lifetime, which is typical for this material used in detectors.

These results highlight the differences in charge carrier behavior across various materials. GaAs exhibits high mobility and a short lifetime, silicon has lower mobility but a longer lifetime, while CdZnTe shows the lowest values for both parameters.

## 4 Conclusion

In this work, we measured and analyzed the characteristics of semiconductor detectors (GaAs, Si, CdZnTe) using an LED setup and an oscilloscope. We focused on signal amplitude, duration, and delay to calculate charge carrier mobility and lifetime.

Our results demonstrate significant differences in charge carrier behavior across different materials. GaAs exhibits high mobility(9920,005 $\left\{\frac{cm^2}{V \cdot s}\right\}$) and short lifetime(4,425 ns), indicating rapid charge movement and quick recombination. Silicon shows lower mobility(648,624 $\left\{\frac{cm^2}{V \cdot s}\right\}$) but a longer lifetime(16,996 ns), suggesting a slower recombination process. CdZnTe has the lowest values for both parameters(500,927 $\left\{\frac{cm^2}{V \cdot s}\right\}$, 2,31 ns), which is typical for this detector material. Alpha irradiation by Am241 notably increases the charge carrier lifetime in silicon(182,547 ns), although it slightly reduces the hole mobility(601,279 $\left\{\frac{cm^2}{V \cdot s}\right\}$).

Additionally, we conducted I-V measurements of GaAs wafers to verify thematrix integrity. The I-V measurements of wafer pads were conducted using a ready-made setup, and the data was processed through a custom program. The program calculated the resistance of each pad and saved the results, confirming the functionality and reliability of the wafer matrices(ranging from $10^9$ to $10^{10}$, whereas we have around $24*10^9$).

## 5 References

[1] e-document: URL: https://root.cern.ch/root/htmldoc/guides/users-guide/ROOTUsersGuideA4.pdf

[2] e-document: URL: https://www.mitsubishielectric.com/semiconductors/hf/products/datasheet/rd01mus2b.pdf

[3] e-document: URL: https://root.cern/doc/master/group__tutorial__fit.html

[4] Semiconductor Detectors for Nuclear Radiation" by Yu.K. Akimov

## Appendix 1 ROOT CERN commands for fitting

**For signal:**

```
float x[32] =
{30,35,40,45,50,60,70,80,90,100,110,120,130,140,150,160,170,180,
190,200,210,220,230,240,250,260,270,280,290,300,310,320};
float y[32] =
{22,25.2,32.9,41.7,52.4,74.4,97.1,120,150,174,201,230,258,302,32
7,357,383,424,449,475,506,535,556,586,610,641,665,703,722,748,77
2,792};
double fitfunc(double *x, double *par) { return
(par[1]*x[0]/par[0])*(1-exp((-par[0])/x[0])); }
TF1* ff = new TF1("ff", fitfunc, 30.0, 320.0, 2);
ff->SetParameter(0,1);
float yerr[32] =
{0.9,0.7,0.2,0.4,0.3,0.4,0.4,0.5,0.5,0.7,0.7,0.6,0.8,1.1,1,0.7,0
.5,1.1,0.9,0.7,1,0.7,1,0.9,0.7,0.8,0.6,0.8,0.9,1.1,1.2,1.3};
TGraphErrors *grapherr = new TGraphErrors(32,x,y,0,yerr);
grapherr->Fit("ff");
grapherr→Draw("AC*");
```

**For time:**

```
float x[32] =
{30,35,40,45,50,60,70,80,90,100,110,120,130,140,150,160,170,180,
190,200,210,220,230,240,250,260,270,280,290,300,310,320};
float y[32] =
{875,798,702,619,566,493,447,428,395,372,356,343,327,321,313,305
,298,290,286,283,280,276,274,274,273,269,269,266,265,263,262,261
};
float y2[32] =
{916,975,1032,1098,1159,1241,1295,1325,1362,1394,1413,1432,1452,
1463,1475,1486,1496,1506,1513,1519,1524,1530,1534,1535,1540,1545
,1547,1551,1554,1557,1559,1561};
float yerr1[32] =
{4.6,3.9,4.8,1.6,1.1,1.4,0.6,0.8,0.4,0.7,0.8,0.2,0.3,0.3,0.2,0.1
,0.5,0.3,0.2,0.2,0.2,0.2,0.3,0.2,0.2,0.3,0.2,0.2,0.2,0.1,0.1,0.1
};
float yerr2[32] =
{4.2,2.9,3.2,2.7,2.9,1.4,0.4,0.5,0.5,0.7,0.3,0.3,0.5,0.4,0.2,0.2
,0.6,0.2,0.2,0.2,0.2,0.2,0.2,0.3,0.2,0.2,0.2,0.2,0.2,0.1,0.1,0.2
};
TGraphErrors *widtherr = new TGraphErrors(32,x,y,0,yerr1);
TGraphErrors *delayerr = new TGraphErrors(32,x,y2,0,yerr2);
double fitfunc(double *x, double *par) { return
(par[0]/x[0]+par[1]); }
TF1* ff = new TF1("ff", fitfunc, 30.0, 320.0, 2);
```

```
widtherr->Fit("ff");
delayerr->Fit("ff");
widtherr->Draw("AC*");
delayerr->Draw("AC*");
```

## Appendix 2 Program for calculating wafer resistances

```
#include <iostream>
#include <fstream>
#include <sstream>
#include <vector>
#include <string>
#include <filesystem>
#include <iomanip>
#include <algorithm>
#include <regex>

namespace fs = std::filesystem;

struct ResistanceData {
    double positiveSum = 0.0;
    double negativeSum = 0.0;
    int positiveCount = 0;
    int negativeCount = 0;
};

ResistanceData processFile(const std::string& filename) {
    std::ifstream file(filename);
    if (!file) {
        std::cerr << "Не удалось открыть файл: " << filename << 
std::endl;
        return {};
    }

    ResistanceData data;

    std::string line;
    while (std::getline(file, line)) {
        std::istringstream iss(line);
        double voltage, current;
        if (iss >> voltage >> current) {
            if (current != 0) {
                double resistance = voltage / current;
                if (voltage > 0) {
                    data.positiveSum += resistance;
                    data.positiveCount++;
                } else if (voltage < 0) {
                    data.negativeSum += resistance;
                    data.negativeCount++;
                }
            }
        }
    }
    file.close();
    return data;
}

bool isValidFile(const std::string& filename, const std::string&
suffix) {
    size_t pos1 = filename.find(suffix);
    size_t pos2 = filename.find("_500V_");
    size_t pos3 = filename.find(".txt");
    return (pos1 != std::string::npos && pos2 !=
std::string::npos && pos3 != std::string::npos && pos2 > pos1 &&
pos3 > pos2);
}

std::vector<std::string> splitString(const std::string& str) {
    std::regex re(R"(\d+|\D+)");
    std::sregex_token_iterator it(str.begin(), str.end(), re);
    std::vector<std::string> parts(it, {});

    return parts;
}

bool naturalSortCompare(const std::string& a, const std::string&
b) {
    std::vector<std::string> partsA = splitString(a);
    std::vector<std::string> partsB = splitString(b);

    auto itA = partsA.begin();
    auto itB = partsB.begin();

    while (itA != partsA.end() && itB != partsB.end()) {
        if (std::isdigit((*itA)[0]) && std::isdigit((*itB)[0]))
{
            int numA = std::stoi(*itA);
            int numB = std::stoi(*itB);
            if (numA != numB) {
                return numA < numB;
            }
        } else {
            if (*itA != *itB) {
                return *itA < *itB;
            }
        }
        ++itA;
        ++itB;
    }

    return partsA.size() < partsB.size();
```

```
}

void processFilesWithSuffix(std::ofstream& outFile, const
std::string& path, const std::string& suffix) {
    std::vector<std::string> validFiles;
    std::string baseName;

    for (const auto& entry : fs::directory_iterator(path)) {
        if (entry.is_regular_file()) {
            std::string filename =
entry.path().filename().string();
            if (isValidFile(filename, suffix)) {
                validFiles.push_back(filename);
                if (baseName.empty()) {
                    baseName = filename.substr(0,
filename.find(suffix));
                }
            }
        }
    }

    std::sort(validFiles.begin(), validFiles.end(),
naturalSortCompare);

    for (const std::string& filename : validFiles) {
        ResistanceData data = processFile(filename);

        double positiveAverage = (data.positiveCount > 0) ?
data.positiveSum / data.positiveCount : 0.0;
        double negativeAverage = (data.negativeCount > 0) ?
data.negativeSum / data.negativeCount : 0.0;

        outFile << "Файл: " << filename << std::endl;
        outFile << "R+: " << positiveAverage << " Ом" <<
std::endl;
        outFile << "R-: " << negativeAverage << " Ом" <<
std::endl;
        outFile << std::endl;
    }
}

int main() {
    std::string path = ".";
    std::string resultFileName;

    for (const auto& entry : fs::directory_iterator(path)) {
        if (entry.is_regular_file()) {
            std::string filename =
entry.path().filename().string();
```

```
            if (filename.find("_s") != std::string::npos ||
filename.find("_b") != std::string::npos) {
                resultFileName = filename.substr(0,
filename.find_first_of("_")) + "_Result.txt";
                break;
            }
        }
    }

    if (resultFileName.empty()) {
        std::cerr << "Не удалось определить имя выходного
файла." << std::endl;
        return 1;
    }

    std::ofstream outFile(resultFileName);
    if (!outFile) {
        std::cerr << "Не удалось создать файл для вывода: " <<
resultFileName << std::endl;
        return 1;
    }

    outFile << std::fixed << std::setprecision(2);

    processFilesWithSuffix(outFile, path, "_s");
    processFilesWithSuffix(outFile, path, "_b");

    outFile.close();

    std::cout << "Результаты сохранены в файл: " <<
resultFileName << std::endl;

    return 0;
}
```