Программа расчёта потерь энергии заряженными частицами в элементах многодетекторного телескопа

В спектрометрах заряженных частиц в качестве устройств детектирования применяют телескопы, состоящие из нескольких ППД различной толщины.

Выделение частиц с массами M_i в интервалах энергий $\Delta E_i = E_{вi} - E_{нi}$ осуществляют с помощью устройства А-Л отбора, которое состоит из нескольких групп амплитудных дискриминаторов , соединённых своими выходами с многовходовой комбинационной логической схемой.

Конкретные значения порогов дискриминаторов берут исходя из геометрии телескопа:

1) число детекторов ;
2) материал из которого сделаны детекторы;
3) толщина чувствительного слоя;
4) толщина материал входного окна;
5) толщина и материал задней (выходной) стенки;
6) толщины и материала поглотителей (иногда вводят между ППД).

Расчет значений порогов базируется на вычислении зависимости потерь энергии в чувствительном слое каждого ППД из телескопа. ($\Delta E_{1i} = \Delta E_{1i}(E_i)$, ……. $\Delta E_{mi} = \Delta E_{mi}(E_i)$) от энергии частицы с массой M_i .

Уравнения Бете

$$-\frac{dE}{d\xi} = 0{,}307\, q^2 \frac{Z}{A} \frac{(1 + E/Mc^2)^2}{(1 + E/Mc^2)^2 - 1} \times$$

$$\times \left\{ \ln \left[\frac{6{,}2 \cdot 10^4}{Z} \cdot \frac{E}{Mc^2} \left(1 + \frac{0{,}5\, E}{Mc^2} \right) \right] + \frac{1}{\left(1 + \frac{E}{Mc^2} \right)^2} \right\}, \qquad (1)$$

$$-\frac{dE}{d\xi} = 0{,}307 \frac{Z}{A} \cdot \frac{(1 + E/mc^2)^2}{(1 + E/mc^2)^2 - 1} \times$$

$$\times \left\{ \ln \left[\frac{3{,}3 \cdot 10^4}{Z} \frac{E}{mc^2} \left(1 + \frac{0{,}5\, E}{mc^2} \right)^{1/2} \right] + \frac{1}{2} \frac{1 + 0{,}82\, (E/mc^2)^2}{(1 + E/mc^2)^2} \right\}. \qquad (2)$$

(1) – для ионизационных тормозных потерь тяжелыми заряженными частицами

(2) – для электронов

E – энергия частицы (МэВ))

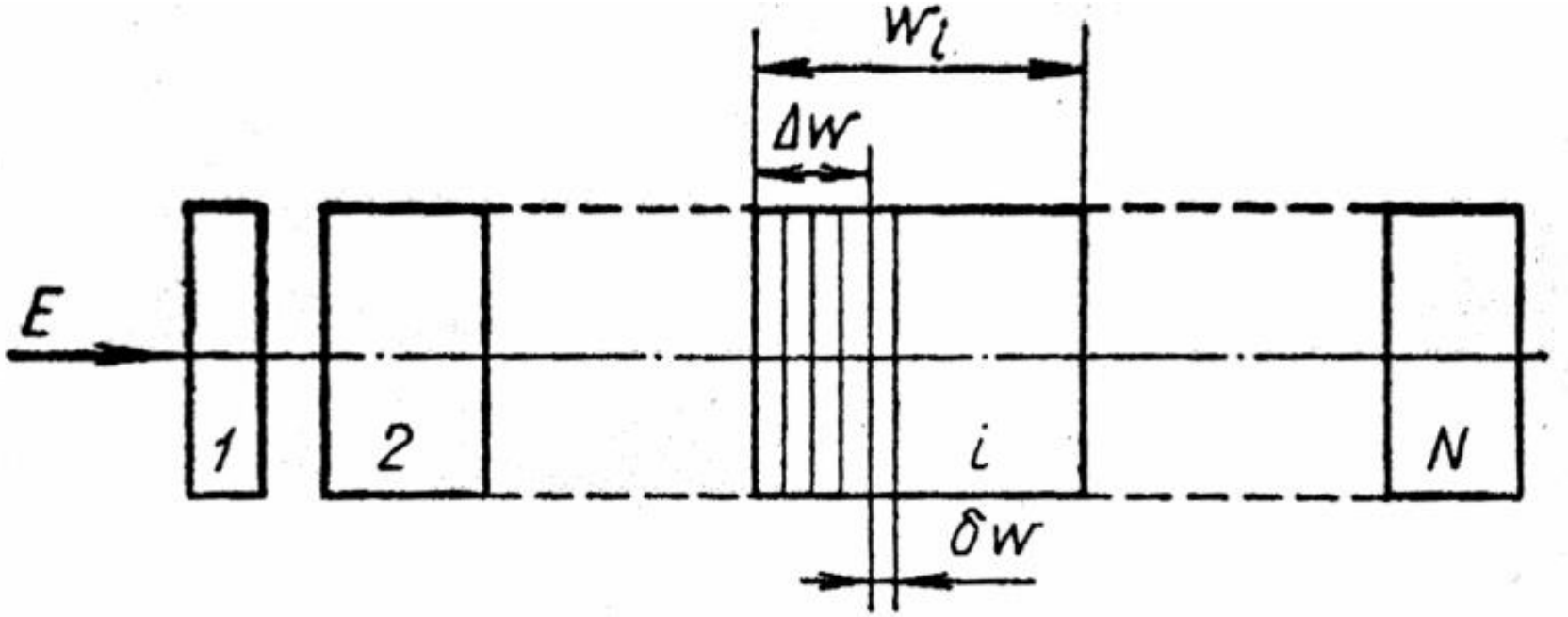
mc^2 – энергия покоя электронов (МэВ)

Mc^2 – энергия покоя тяжелых частиц (МэВ)

q – заряд в единицах элементарного заряда

W – толщина тормозящего слоя (г / $см^2$)

Обобщенная структура телескопа



Пакет из N элементов телескопа. Что понимается под элементом телескопа ? Любой однородный слой, например, входное окно ППД и т.д. . Каждый элемент характеризуется : Z-атомным номером, A- массовым числом, W- толщиной элемента.

Обозначения:

i , i_{max} – номер текущего и последнего элементов телескопа;

A_i , Z_i - физические параметры i -го элемента;

W_i - толщина i -го элемента;

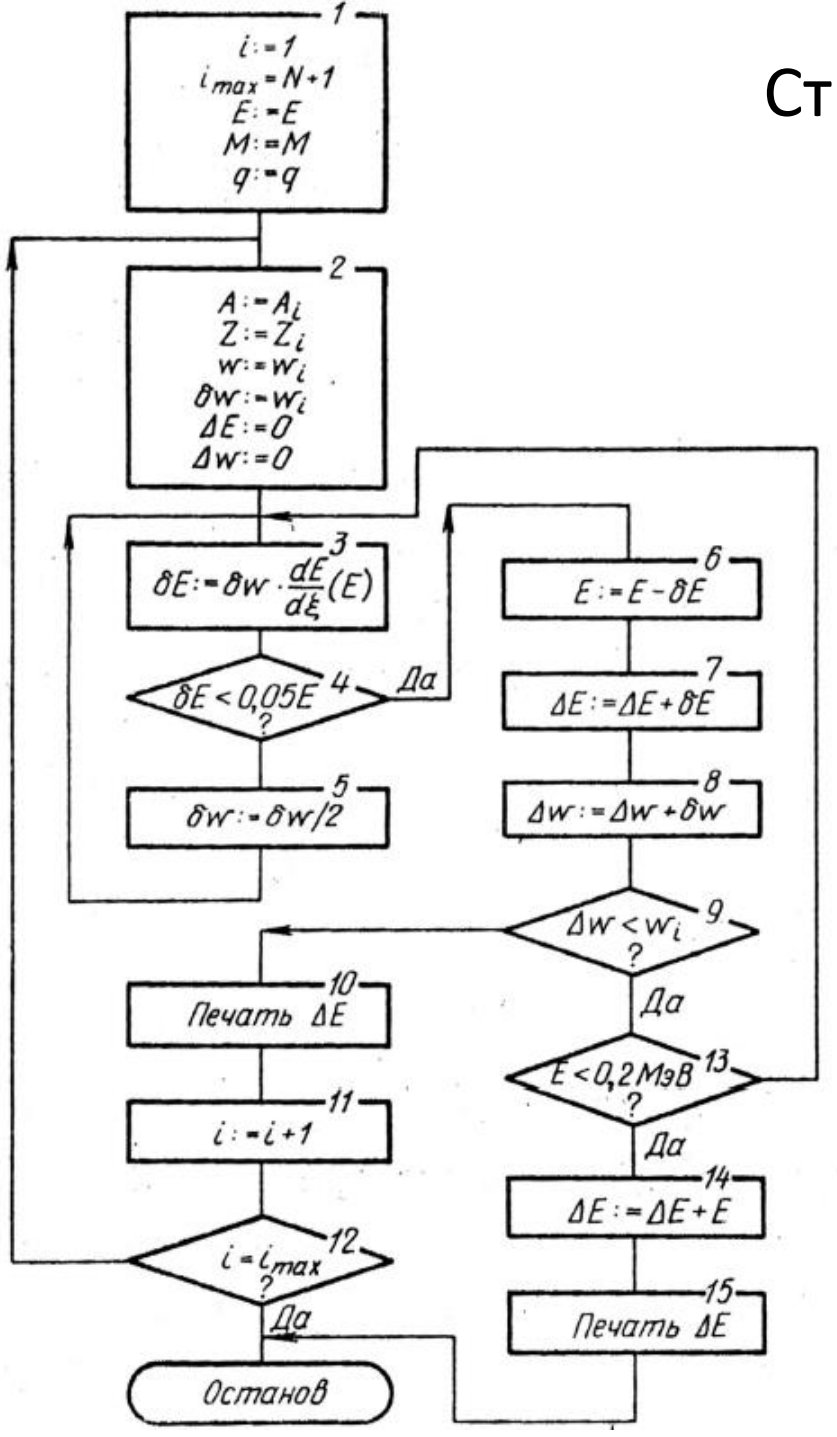
δW - толщина элементарного слоя текущего элемента;

ΔW - часть толщины текущего элемента, для которой уже просуммированы потери

$\delta E = \delta w \left(dE/ d\xi \right)_E$ - потеря энергии в элементарном слое частицей с энергией E

ΔE – потеря энергии в слое ΔW

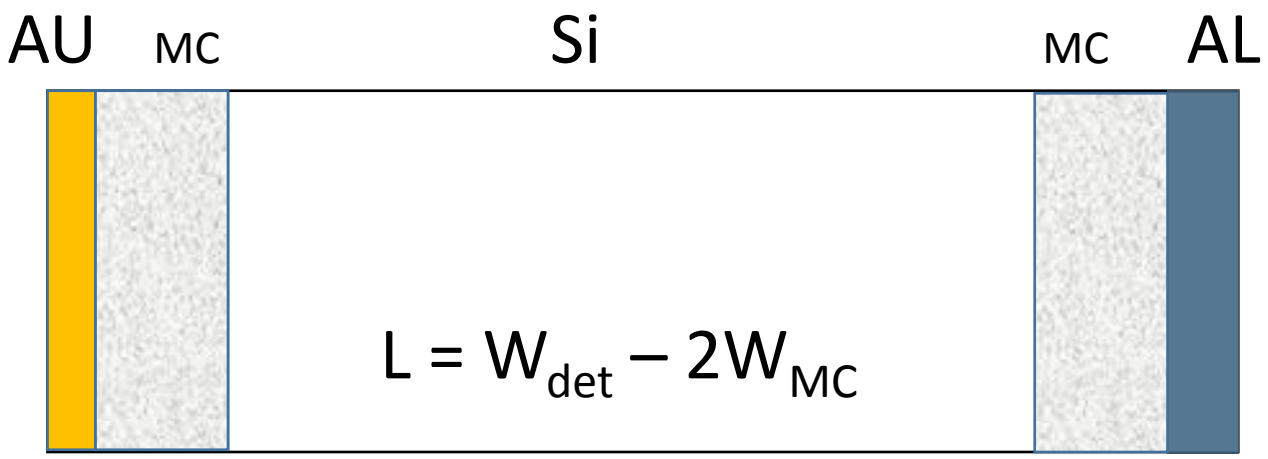
Структурная схема алгоритма



<u>Требования к программе.</u>

1) Для задания параметров материала слоя и толщины слоя использовать оператор DATA
2) Вычисление потерь энергии в «тонком» слое оформить в виде процедуры SUBROUTINE
3) Для проверки работы программы использовать данные Таблиц 1-3

Схематическая конструкция детектора



Геометрия телескопа

1) Четыре кремниевых ППД толщиной $W1_{det}$ $W2_{det}$ $W3_{det}$ $W4_{det}$
   $W1_{det}$ = 10 мкм
   $W2_{det}$ = 100 мкм
   $W3_{det}$ = 1000 мкм
   $W4_{det}$ = 1000 мкм
2) Каждый ППД имеет входное окно W_{Au} , мёртвый слой кремния W_{MC} и алюминиевый контат W_{Al} .
   W_{Au} = 0.01 мкм
   W_{MC} = 0.04 мкм
   W_{Al} = 0.02 мкм

Диапазон энергий частиц на входе телескопа.

Вариант№1	Энергия протонов E, МэВ	0.1−150 МэВ
Вариант№2	Энергия α - частиц E, МэВ	1 – 40 МэВ
Вариант№3	Энергия электронов E, МэВ	0.01 – 5 МэВ

Таблица 1

Потеря энергии протонов

Энергия протонов E, МэВ	Потеря энергии в чувствительном слое ППД, МэВ			
	ΔE_1	ΔE_2	ΔE_3	$\Delta E_3 + \Delta E_4$
.1	0	0	0	0
.15	0	0	0	0
.2	.195118	0	0	0
.3	.292385	0	0	0
.5	.491202	0	0	0
.7	.691137	0	0	0
1	.497132	.485924	0	0
1.5	.352442	1.13274	0	0
2	.281034	1.70543	0	0
3	.205778	2.78214	0	0
5	.139025	1.62458	3.22108	3.22108
7	.107107	1.16144	5.71766	5.71766
10	.0810175	.847135	9.05916	9.05916
15	.0588176	.603039	7.94463	14.3247
20	.0468092	.476464	5.42543	13.6358
30	.0339236	.343416	3.63027	7.71752
50	.0226952	.22905	2.33936	4.7725
70	.0175125	.176583	1.78626	3.60904
100	.0134175	.135223	1.36067	2.73525
150	.0100719	.101476	1.01808	2.04094

Таблица 2

Потеря энергии α-частиц

Энергия α-частиц E, МэВ	Потеря энергии в чувствительном слое ППД, МэВ			
	ΔE_1	ΔE_2	ΔE_3	$\Delta E_3 + \Delta E_4$
1	.97341	0	0	0
2	1.96481	0	0	0
3	2.96462	0	0	0
4	1.98853	1.94369	0	0
5	1.63678	3.30011	0	0
6	1.40977	4.53094	0	0
7	1.24753	5.69604	0	0
8	1.12414	6.8217	0	0
9	1.026635	7.92139	0	0
10	.94651	9.00273	0	0
11	.879821	10.0707	0	0
12	.823112	11.1285	0	0
13	.774182	12.1784	0	0
14	.731464	11.2477	1.93979	1.93979
15	.693795	9.61094	4.62214	4.62214
20	.556101	6.49832	12.8843	12.8843
25	.4679	5.16958	19.3057	19.3057
30	.405993	4.36179	25.1781	25.1781
35	.359886	3.80327	30.7847	30.7847
40	.32407	3.38854	36.2366	36.2366

Таблица 3

Потеря энергии электронов

Энергия электронов E, МэВ	Потеря энергии в чувствительном слое ППД, МэВ			
	ΔE_1	ΔE_2	ΔE_3	$\Delta E_3 + \Delta E_4$
.01	0	0	0	0
.02	.0194868	0	0	0
.03	.0295345	0	0	0
.04	.0169913	.0221654	0	0
.05	.0134468	.0358293	0	0
.06	.0114404	.0478872	0	0
.07	.0101041	.0592548	0	0
.08	9.13714E-03	.0702448	0	0
.09	8.40069E-03	.0810033	0	0
.1	7.81831E-03	.0916017	0	0
.15	6.09901E-03	.0745376	.0685785	.068578
.2	5.25413E-03	.0577987	.136224	.136224
.3	4.43528E-03	.0461881	.248696	.248696
.5	3.84647E-03	.0391083	.456393	.456393
.7	3.65132E-03	.0369036	.400869	.658726
1	3.56500E-03	.0359317	.365363	.790199
1.5	3.57688E-03	.03601	.359104	.720119
2	3.63601E-03	.0365974	.3637	.723625
3	3.76749E-03	.0379219	.376806	.748643
5	3.98273E-03	.0400959	.399165	.794408

Подпрограммы и модули
(обзор для FORTRAN 90–95)

Со структурой главной программы Вы уже познакомились на предыдущем занятии.

```
PROGRAM
операторы_описания
исполняемые_операторы
CONTAINS
        Внутренние_подпрограммы
END
```

Внутренние подпрограммы. Понятие.

а) не считаются самостоятельными программными единицами;

б) могут использоваться только тем программным компонентом, в котором написаны ;

в) *извне* компонента-носителя они *недоступны*;

г) не могут содержать собственных внутренних подпрограмм.

Внешние подпрограммы.

Заголовок имеет вид:[префикс_специф.] специф._подпрог. имя_подпрог.[(список_формальных_параметров)],
где спецификатор_подпрограммы – один из операторов:①SUBROUTINE – процедура или ②FUNCTION – функция
① - не имеет возвращаемого значения, всё предаётся через формал. параметры или глобальные переменные
② - возвращает значение заданного типа,[префикс_спецификатора] – содержит тип возвращаемого значения

Обращение к процедуре или функции:

① - → CALL имя_подпрограммы [(список_фактических_параметров)]

② - → A= MYFUNC (X)

Интерфейсы.

Для организации вызова любой подпрограммы достаточно знать её *интерфейс*: заголовок, список формальных параметров и их описание.

Пример:

```
PROGRAM main
REAL X, Y,Z
INTERFACE
        SUBROUTINE outer_sub(A, B, C)
        REAL A, B, C
        END SUBRUOTINE outer_sub
END INTERFACE
DATA X, Y /0.987, 1.234/
READ(*,*)  S
CALL outer_sub(X, Y, S)
END
```

В INTERFASE нельзя включать DATA FORMAT

Параметры подпрограмм

Обмен информацией с подпрограммами может происходить:

а) через глобальные переменные;

б) через параметры подпрограмм, так называемый механизм формальных-фактических параметров

Пример:

```
SUBROUTINE sub (A, B,C)
C=SQRT (A**2 + B**2)
END SUBROUTINE sub
PROGRAM main
……..
CALL sub (X, Y, Z)
END PROGRAM main
```

Ограничения, накладываемые на фактические параметры

◊ любые действия, влияющие на фактический параметр в обход формального параметра, запрещены;

◊ если хотя бы часть фактич. параметра получает значение через форм. параметр, то ссылаться на этот фактич. параметр можно только через форм. параметр.

Т.е. если в подпрограмме введен форм. параметр А , то он будет доступен только как А, независимо от того, какой фактич. параметр подставляется на его место.

Возможность доступа к фактич. параметру появляется когда он становится глобальной переменой из модуля или переменная, доступная через COMMON - блок.

Вид связи формального параметра

Можно указывать назначение формального параметра: атрибут INTEN (параметр) , где параметр принимает значения IN, OUT, INOUT.

Использование имен внешних подпрограмм в качестве фактического аргумента

Оператор EXTERNAL (a1, a2, a3, …) ,где ai – имя внешней FUNCTION или SUBROUTINE , которое является фактическим аргументом.

Ключевые и необязательные параметры

Обычно назначение параметра определяется его позицией в списке аргументов оператора вызова

Пример. Пусть есть SUBROUTINE KEY (NUM, RECOD, FILE)

а) позиционный вариант вызова (стандарт Фортран 77):

CALL KEY(NUMBER, TALE, WR)

б) в ключевом варианте вызова порядок параметров может быть любым:

CALL KEY(RECORD = TALE, FILE = WR, NUM = NUMBER)

При вызове подпрограммы часть параметров может быть опущена. Такие параметры называются ***необязательными***. Соответствующие им формальные параметры отмечаются атрибутом OPTIONAL.

Пример.

```
SUBROUTINE SUB( STR, LEN, IBEG )
……
INTEGER , OPTIONAL :: IBEG
```

В главной программе CALL SUB (A, B)

Добавив строчку в подпрограмме

```
IF ( . NOT . PRESENT (IBEG ) )    IBEG = 1
```

Можно получить значение параметра по умолчанию.

Модули

Организация доступа к глобальным данным.

Структура модуля:

```
MODULE   имя_ модуля
         [ операторы_ описания ]
[ CONTAINS
модульные_подпрограммы ]
END    [ MODULE    [ имя_модуля ]  ]
```

Помещение в модуль описания данных.

Пример.

```
MODULE   ARRAY
         REAL,  DIMENSION  ( 20 )   : :   A
END   MODULE  ARRAY
```

Затем этот модуль включаем в подпрограммы SUB1, SUB2, SUB3.

```
SUBROUTINE  SUB1
USE  ARRAY
……
END SUBROUTINE SUB1
```

и т.д.

Массив A будет доступен для всех трех подпрограмм.

```
PROGRAM   SUBS
          CALL  SUB1
          CALL  SUB2
          CALL  SUB3
END   PROGRAM    SUBS
```

Рекурсивные процедуры и функции

Предполагается, что подпрограммы не содержат обращений к себе : по умолчанию ***рекурсия подпрограмм запрещена*** (стандарт Фортран 77). Подпрограмму можно объявить рекурсивной, используя декларацию

```
RECURSIVE   SUBROUTINE   FACTORIAL ( N, NF )
DATA NF  / 1 /        →  этот оператор при рекурсивном вызове выполняется только один раз
IF (N . GT . 1 )     THEN
          NF = NF * N
          CALL  FACTORIAL ( N – 1 , NF )
END IF
END
```

Области видимости имён (именованные объекты) и меток

Для них существует пространство , в рамках которого они доступны по своему идентификатору. Это пространство называется областью видимости.

К именованным объектам относятся переменные, именованные константы, подпрограммы (внутренние и внешние), производные типа.

Оператор USE

Открывает программному компоненту доступ к переменным, подпрограммам, интерфейсам и другим объектам, заключенным в указанном модуле. Простая форма USE имя_модуля

Полная форма → USE *имя_модуля , список_переименований*

где *, список_переименований* состоит из пар локальное_имя = > модульное_имя, лок._имя => модул._имя …

или из директивы ONLY : [only_список]

Пример.

Есть модули A и B , содержащие разные функции с одним именем S .

Тогда в главной программе

```
PROGRAM prog
USE A, AS => S
USE B, BS => S
DATA X, Y / 1.8, 1.5 /
WRITE ( *, * ) ‘ sin ( X + Y ) =‘ , AS ( X, Y )
WRITE ( *, * ) ‘ cos ( X+ Y ) =‘ , BS ( X, Y )
END PROGRAM prog
```

Области общей памяти

COMMON - блоки могут быть именованными и неименованными.

Пример.

```
COMMON / BASE / X, Y, Z, INDEX (10)
COMMON / / RESULT ( 10000 )
```

В COMMON - блок ***нельзя*** включать:

- - именованные константы
- - формальные параметры
- - выделяемые массивы.
- - функции

В различных программах имена переменных в COMMON – блоках могут ***не совпадать.***

Пример.

```
COMMON / BASE / A ( 3 ) , I , J , K ( 8 )
                X , Y , Z   INDEX ( 10 )
```