УСТРОЙСТВО ЧПУ

"МАЯК-600"

ИНСТРУКЦИЯ К ЯЗЫКУ РАЗМЕТКИ ИНТЕРФЕЙСА

1 Оглавление

1 Зоны экрана

Экран разделен на зоны, в которых индицируется название режима, меню и другая информация, определяемая режимом работы устройства. Например, экран режима «Автомат» выглядит следующим образом:

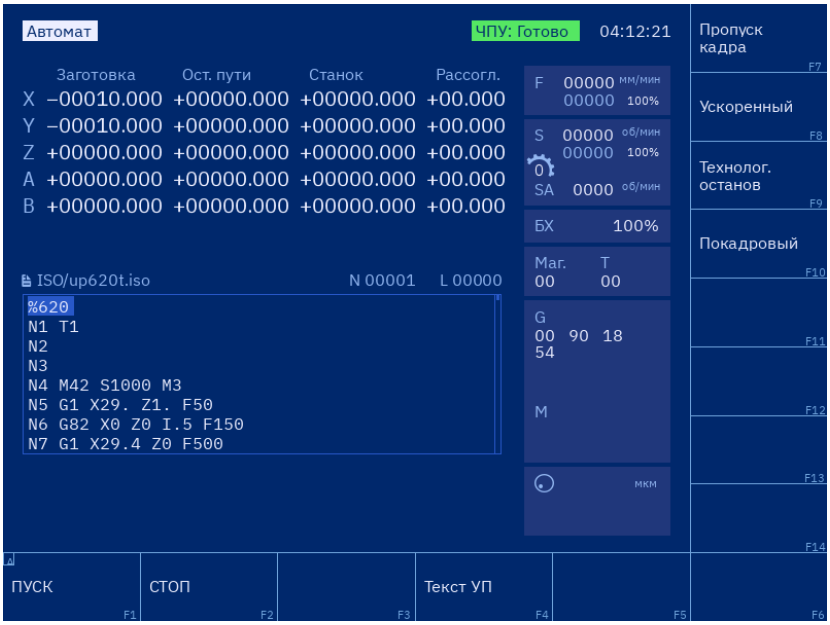

Рисунок 1

Все экраны имеют ряд обязательных элементов, которые являются общими для всех и расположены в строго определенных местах. К ним относятся:

- горизонтальный ряд меню;
- вертикальный ряд меню;
- область наборной строки и сообщений из БПрО;
- область, доступная блокам и пользовательскому интерфейсу, далее по тексту доступная область.

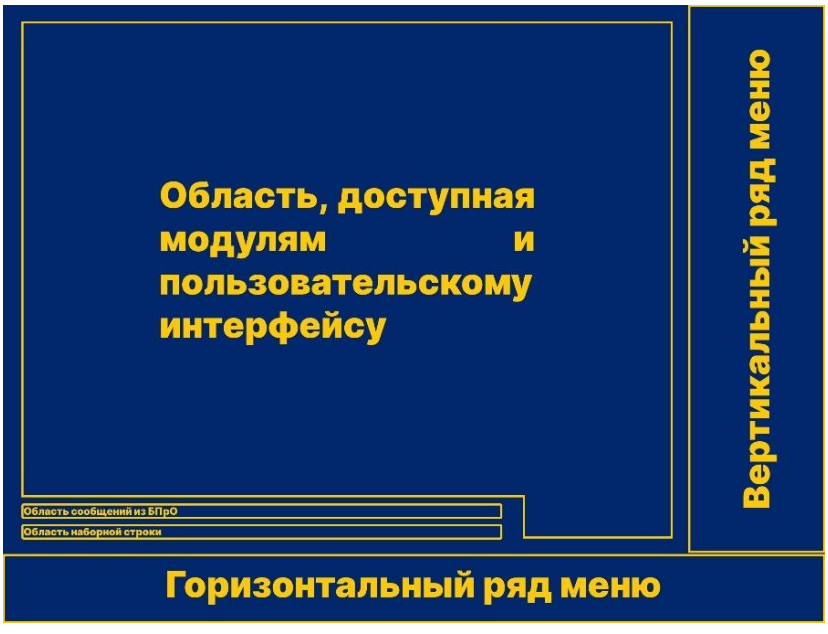

Рисунок 2

Начало координат находится в левом верхнем углу, соответственно координаты точек указываются относительно этого угла и проходят через центр пикселя. Например, точка с координатами (1, 2) будет находиться на один пиксель вправо и на два пикселя вниз от начала координат.

Экран УЧПУ "Маяк" с диагональю 10,4 дюйма имеет разрешение 800x600 пикселей, с диагональю 15 дюймов – 1024x768 пикселей.

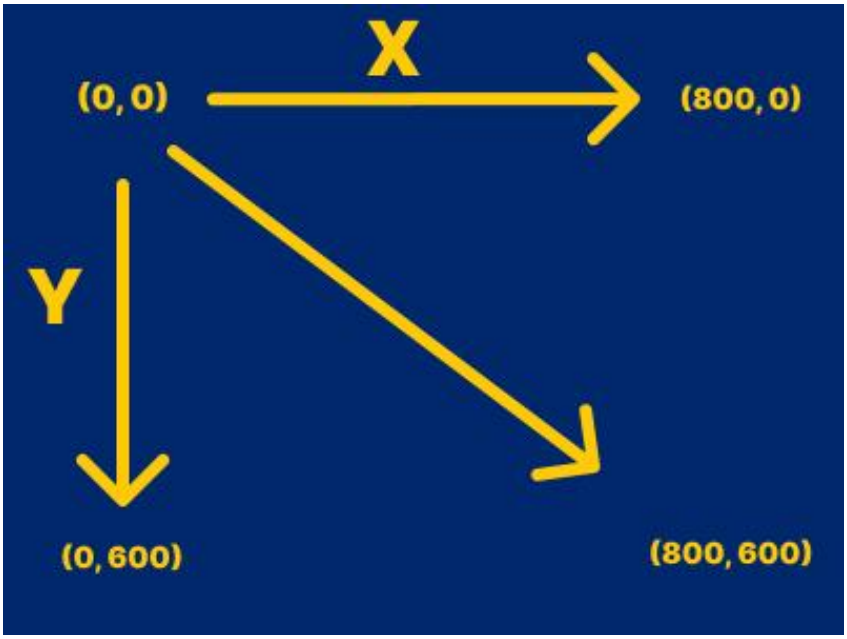

Рисунок 3

2 Иерархия

В файлах описания экранных интерфейсов используется специальный язык разметки. С помощью этого языка осуществляются определение структуры и оформление страниц. Так же он позволяет добавить функционал (например изменение значения переменной с помощью специальной команды, переход на другую страницу при нажатии определённой клавиши). Всё это позволяет точно определить местоположение и значение каждого блока в документе, что важно для правильного его отображения и обработки.

Рассмотрим три независимые иерархии, которые используются в данном языке разметки. Первые из них иерархии, связанные с выводом информации в доступную область, они включают в себя следующие блоки: форму **(form)**, сетку **(grid)**, вид **(view)**, элемент **(item)**, модуль.

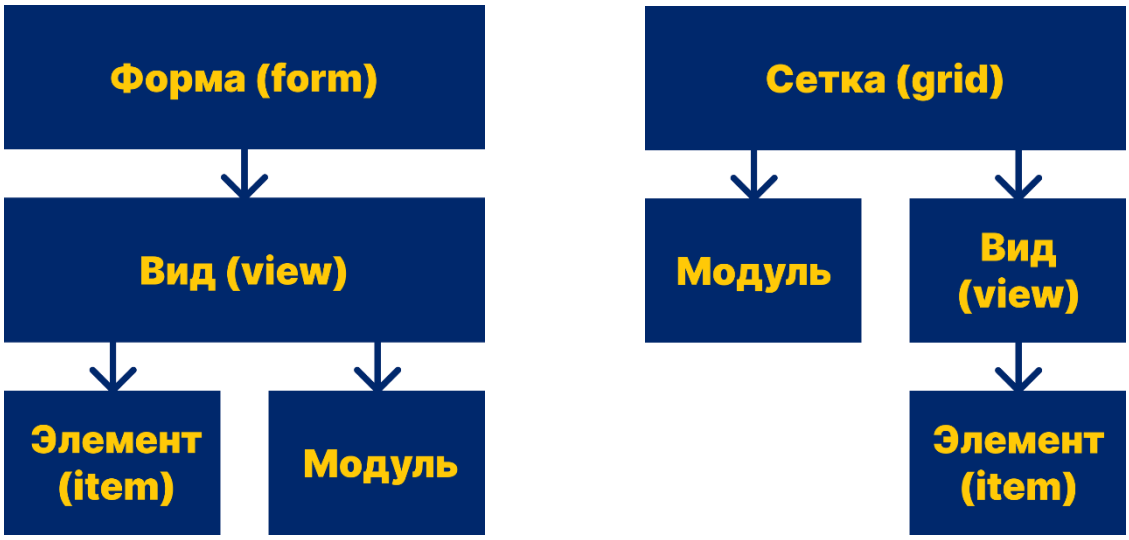

Рисунок 4

Следующая иерархия связана с выводом текста на клавиши меню горизонтального и вертикального рядов и их функциональности:

Рисунок 5

Таким образом, иерархия является ключевым элементом и обеспечивает правильное структурирование и организацию информации в документах, что делает их более эффективными и легкими для понимания.

3 Доступная область

3.1 Форма (form)

Корневым экраном является форма (**form**). Каждому режиму соответствует одна форма, которая меняется одновременно со сменой режима, и где описаны страницы индикации для соответствующего режима. В форму можно вставлять один или несколько видов (**view**).

Синтаксис формы выглядит следующим образом:

```
{ form номер_формы (параметры_формы)…
      { hotkey ... }
      { view ... }
      ...
      { view ... }
} form
```

Соответствия номеров форм и режимов (в порядке расположения в Меню переключения режимов):

- «Ручной» форма 1 (**form 1**),
- «Автомат» форма 0 (**form 0**),
- «Выход в ноль» форма 2 (**form 2**),
- «Выход в точку» форма 6 (**form 6**),
- «Фиксированные перемещения» форма 8 (**form 8**),
- «Преднабор» форма 7 (**form 7**)

В параметрах формы можно указывать следующие значения:

- **menu=*номер_меню***, при переключении на данную форму будет вызвано указанное меню;
- размер доступной области (см. пункт 3.8).

Так же в форме могут располагаться "горячие клавиши" **(hotkey)**. Синтаксис горячей клавиши:

```
{ hotkey имя_горячей_клавиши { pressed команда } }
```

Имена "горячих клавиш" задаются в файле hotkeys.dat.

Пример формы из файла formavt.dat – экран режима «Автомат», с дополнительными страницами (видами), результат показан на рисунках 6 и 7:

```
{ form 0 (menu=10, xmax=665, ymax=600)
   { hotkey KEY_ENTER { pressed d_set_ust } }
   { view ()
      @clock.dat@
      { item # ()
         GAP_H    GAP_V    yes   yes            ' Автомат  ' }
      { item m30 (invbgr=Error_C invfgr=MAIN_BGR width=106)
         111  GAP_V   yes   yes ' ЧПУ: Авария  '
         { if m30=255 invbgr=GREEN format=' ЧПУ: Готово   ' }   }
      { item m20 (fgr=B3 width=160)
         111+123     GAP_V   no          yes    '<24>'
```

```
            { if m20=0 fgr=MAIN_BGR } } ; Статус отработки УП

    @box_name_s.dat@     ; Названия осей плюс признаки выхода в ноль и в точку
    @box_zgt_avt_s.dat@  ; Значения "Заготовка"
    @box_ost_s.dat@      ; Значения "Ост. пути"
    @box_st_s.dat@       ; Значения "Станок"
    @box_rsg_s.dat@      ; Значения "Рассогласование"
    @box_f_avt.dat@      ; Зона F
    @box_s_avt.dat@      ; Зона S
    @box_r.dat@          ; Зона БХ
    @box_t.dat@          ; Зона T
    @box_gm.dat@         ; Зона с G и M
    @box_mhv.dat@        ; Зона маховика
    @box_up.dat@         ; Блок УП
    @m28form.dat@        ; m28 и m29
  } view


  ;Заготовка + Остаток пути ====================================
  { view (fgr=AN_C bgr=REG_F invfgr=REG_F invbgr=AN_C)
    @clock.dat@
    { item # ()
       GAP_H    GAP_V    yes   yes            ' Автомат  ' }
    { item m30 (invbgr=Error_C invfgr=MAIN_BGR width=106)
       111  GAP_V   yes   yes ' ЧПУ: Авария  '
       { if m30=255 invbgr=GREEN format=' ЧПУ: Готово   ' }   }
    { item m20 (fgr=B3 width=160)
       111+123     GAP_V   no   yes    '<24>'
           { if m20=0 fgr=MAIN_BGR } } ; Статус отработки УП

    @box_name_s_2.dat@   ; Названия осей плюс признаки выхода в ноль и в точку
    @box_zgt_and_ost_avt_s.dat@  ; Значения "Заготовка" и "Ост. пути"
    @box_f_avt.dat@      ; Зона F
    @box_s_avt.dat@      ; Зона S
    @box_r.dat@          ; Зона БХ
    @box_t.dat@          ; Зона T
    @box_gm.dat@         ; Зона с G и M
    @box_mhv.dat@        ; Зона маховика
    @box_up_2.dat@       ; Блок УП
    @m28form.dat@        ; m28 и m29

  } view
} form
```

Рисунок 6. Основной экран режима «Автомат»

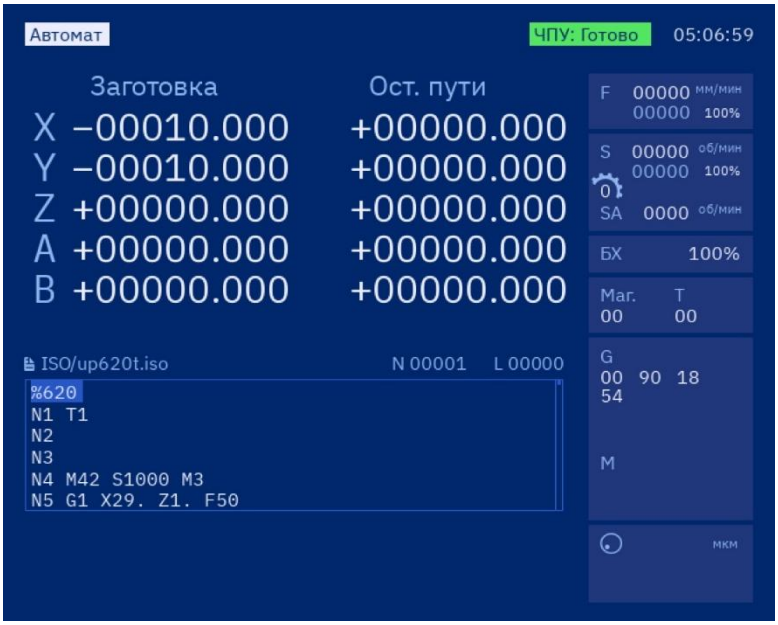

Рисунок 7. Дополнительный экран режима «Автомат»

3.2 Сетка (grid)

Сетка (**grid**) позволяет разбить основной экран на заданное число столбцов и строк в границах доступной области. В рамках сетки можно задавать независимые области, границы которых будут проходить по ее линиям. В любую из областей можно размещать или вид или модуль, которые будут занимать ее целиком. С области на область можно переходить командой **`grid_next`**, если у них стоит параметр **`grid_selectable`**, не равный нулю. Координаты и размеры данных областей задаются не в пикселях, а в прямоугольниках сетки (нумерация с нуля), и начало координат находится в левом верхнем углу. Линии сетки не показываются, но считается, что они имеют толщину один пиксель, что следует иметь в виду при определении

размеров области. Сама область отдельно не объявляется, а ее параметры указываются в параметрах размещенного в ней вида или модуля.

Синтаксис сетки выглядит следующим образом:

```
{grid название_сетки (параметры_сетки)
      ...
} grid
```

В параметрах сетки можно указывать следующие значения:

- размер доступной области (см. пункт 3.8);
- **menu=*номер_меню*** при переключении на данную сетку будет вызвано указанное меню;
- **grid_rows** количество строк, на которое будет разбита доступная область;
- **grid_cols** количество столбцов, на которое будет разбита доступная область;
- **grid_border** цвет рамки (границы) модуля или вида, который находится в данной сетке;
- **grid_bgr** цвет всей доступной области, которая находится в координатах сетки.

Пример работы сетки, где каждый прямоугольник — это отдельный вид (разбиение на 18 столбцов и 31 строку):

```
{ grid g_edit_default (grid_rows = 31 grid_cols = 18)
      { view (xmax=800 ymax=600 windowed=1 grid_x=0 grid_y=0 grid_w=1 grid_h=1
            grid_selectable=0 bgr = ggg )}view
      { view (xmax=800 ymax=600 windowed=1 grid_x=0 grid_y=1 grid_w=1 grid_h=1
            grid_selectable=0 bgr = ggg )}view
      { view (xmax=800 ymax=600 windowed=1 grid_x=0 grid_y=2 grid_w=1 grid_h=1
            grid_selectable=0 bgr = ggg )}view
      { view (xmax=800 ymax=600 windowed=1 grid_x=1 grid_y=0 grid_w=1 grid_h=1
            grid_selectable=0 bgr = ggg )}view
      { view (xmax=800 ymax=600 windowed=1 grid_x=1 grid_y=1 grid_w=1 grid_h=1
            grid_selectable=0 bgr = ggg )}view
      { view (xmax=800 ymax=600 windowed=1 grid_x=2 grid_y=0 grid_w=1 grid_h=1
            grid_selectable=0 bgr = ggg )}view

      { view (xmax=800 ymax=600 windowed=1 grid_x=15 grid_y=30 grid_w=1 grid_h=1
            grid_selectable=0 bgr = ggg )}view
      { view (xmax=800 ymax=600 windowed=1 grid_x=16 grid_y=29 grid_w=1 grid_h=1
            grid_selectable=0 bgr = ggg )}view
      { view (xmax=800 ymax=600 windowed=1 grid_x=16 grid_y=30 grid_w=1 grid_h=1
            grid_selectable=0 bgr = ggg )}view
      { view (xmax=800 ymax=600 windowed=1 grid_x=17 grid_y=28 grid_w=1 grid_h=1
            grid_selectable=0 bgr = ggg )}view
      { view (xmax=800 ymax=600 windowed=1 grid_x=17 grid_y=29 grid_w=1 grid_h=1
            grid_selectable=0 bgr = ggg )}view
      { view (xmax=800 ymax=600 windowed=1 grid_x=17 grid_y=30 grid_w=1 grid_h=1
            grid_selectable=0 bgr = ggg )}view
} grid
```

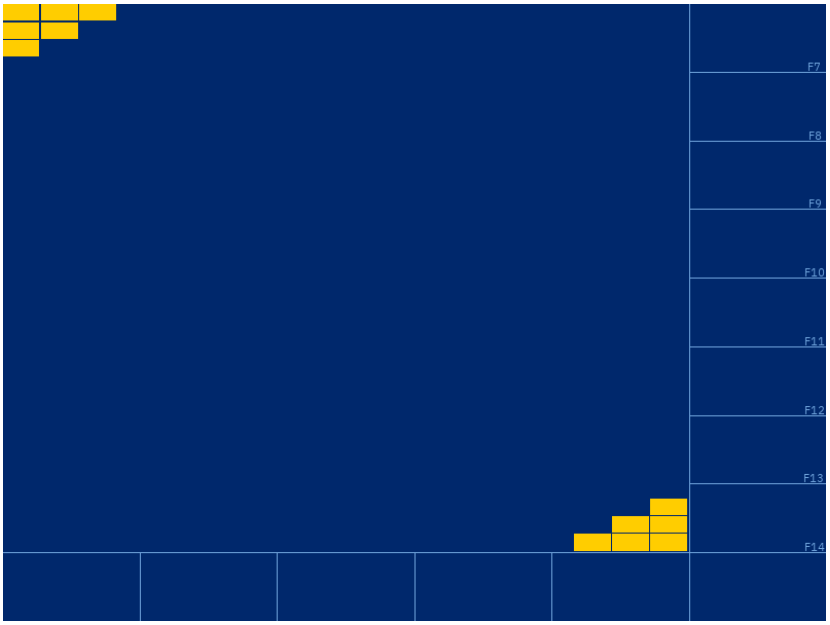

Рисунок 8. Пример работы сетки **(grid)**

3.3 Вид (view)

Вид (**view**) используется для выделения необходимой области и отображения в ней элементов, графических примитивов и т.п. в заданных местах в заданной последовательности.

Вид всегда является неотъемлемой частью или формы (**form**), или сетки (**grid**), и в зависимости от этого меняются указываемые для него параметры.

Синтаксис вида выглядит следующим образом:

```
{view (параметры_вида)
      ...
} view
```

3.3.1 Вид (**view**) в форме (**form**)

В параметрах вида, который находится в форме, можно указывать следующие значения:

- размер доступной области (см. пункт 3.8);
- цвета (см. пункт 3.8);
- **menu=*номер_меню*,** при переключении на данный вид будет вызвано указанное меню;
- **page=0** при таком параметре нельзя попасть на вид с помощью клавиш <Страница вверх> и <Страница вниз>.

Пример вида в форме (formavt.dat – экран режима «Автомат» без дополнительных страниц):

```
{ form 0 (menu=10, xmax=665, ymax=600)
   { hotkey KEY_ENTER { pressed d_set_ust } }
   { view (fgr=AN_C bgr=REG_F invfgr=REG_F invbgr=AN_C)
      @clock.dat@
```

```
      { item # ()
            GAP_H    GAP_V    yes    yes            ‘ Автомат  ‘ }
      { item # (invbgr=Error_C invfgr=MAIN_BGR width=105)
            456 16   yes   no ‘’
            { if m30=0 invfgr=MAIN_BGR format=’ ЧПУ: Авария  ‘ }}
      { item # (invbgr=Error_C invfgr=MAIN_BGR width=106)
            456 16   yes   no ‘’
            { if m30=255 invbgr=GREEN format=’ ЧПУ: Готово  ‘ }}
      { item # (width=1 bgr=MAIN_BGR)
            560 16 no yes ‘ ‘
            { if m30=0   bgr=MAIN_BGR format=’ ‘}
            { if m30=255 bgr=MAIN_BGR format=’ ‘}}

      { item m20 (fgr=B3  bgr=B7 width=160)
            111  GAP_V   no   yes    ‘            ‘
            { if m20#0 format='<24>'} } ; Статус отработки УП

      @box_name_s_2.dat@   ; Названия осей плюс признаки выхода в ноль и в точку
      @box_zgt_and_ost_avt.dat@  ; Значения «Заготовка» и «Ост. пути»
      @box_f_avt.dat@      ; Зона F
      @box_s_avt.dat@      ; Зона S
      @box_r.dat@          ; Зона БХ
      @box_t.dat@          ; Зона T
      @box_gm.dat@         ; Зона с G и M
      @box_mhv.dat@        ; Зона маховика
      @box_up_2.dat@       ; Блок УП
      @m28form.dat@        ; m28 и m29

  } view
} form
```

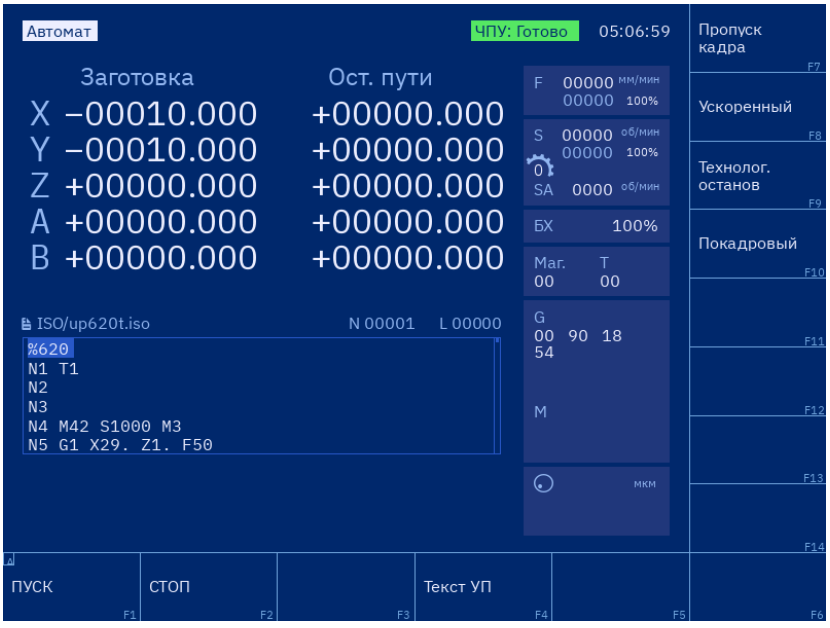

Рисунок 9. Пример работы вида (**view**) в форме (form)

3.3.2 Вид (**view**) в сетке (**grid**)

В параметрах вида, который находится в сетке, можно указывать следующие значения:

- размер доступной области (см. пункт 3.8);

- **windowed:**
  - **windowed = *1*** все расчеты будут проводиться относительно целого экрана (размер экрана 800 пикселей на 600 пикселей);
  - **windowed = *no*** все расчеты будут проводиться относительно доступной области (задаётся параметрами **xmax** и **ymax**);
- цвета (см. пункт 3.8);
- **grid_x** расположение вида (в прямоугольниках сетки) по x;
- **grid_y** расположение вида (в прямоугольниках сетки) по y;
- **grid_w** ширина вида (в прямоугольниках сетки);
- **grid_h** высота вида (в прямоугольниках сетки);
- **grid_selectable** – если указан этот параметр не равный нулю, то на данный вид можно переходить (с помощью команды **grid_next**);
- **menu=*номер_меню*,** при переключении на данный вид будет вызвано указанное меню.

Принцип работы вида: представленная ниже часть кода выделяет доступную область (выделена желтым цветом), координаты которой такие:

- в ячейках сетки (grid)
  - координата x = 5;
  - координата y = 10;
  - ширина – 28;
  - высота – 43;
- в пикселях
  - координата x = 88 пикселей;
  - координата y = 85 пикселей;
  - ширина – 489 пикселей;
  - высота – 362 пикселя.

```
{ grid g_edit_error ()
   { view (xmax=800 ymax=600 windowed=1 grid_x= 5 grid_y=10 grid_w= 28 grid_h = 43
      bgr=ggg)}
} grid
```

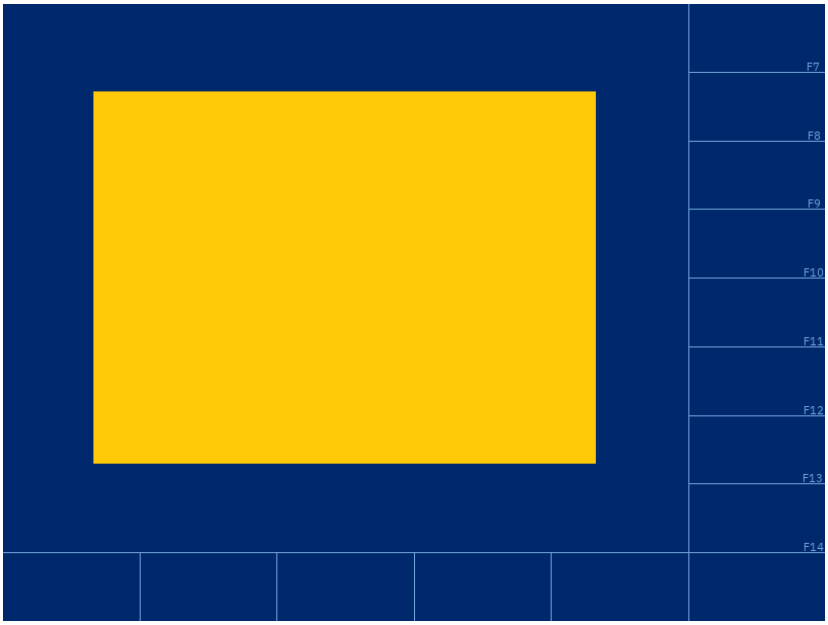

Рисунок 10. Принцип работы вида (**view**)

Пример вида из файла tables.dat:

```
{ grid g_version ()
   { version_tab }
   { view (xmax=800 ymax=600 windowed=1 grid_y=0 grid_h=DEF_GRID_Y
         grid_selectable=0 bgr = MAIN_BGR )
         { item # (height=FONTSIZE_MID invfgr=MAIN_BGR invbgr=CHANNEL0)
               0  GAP_V   yes   yes    ' Версия ПрО  '          }
   } view
   @m28grid.dat@     ; Вывод m28 и m29
} grid
```

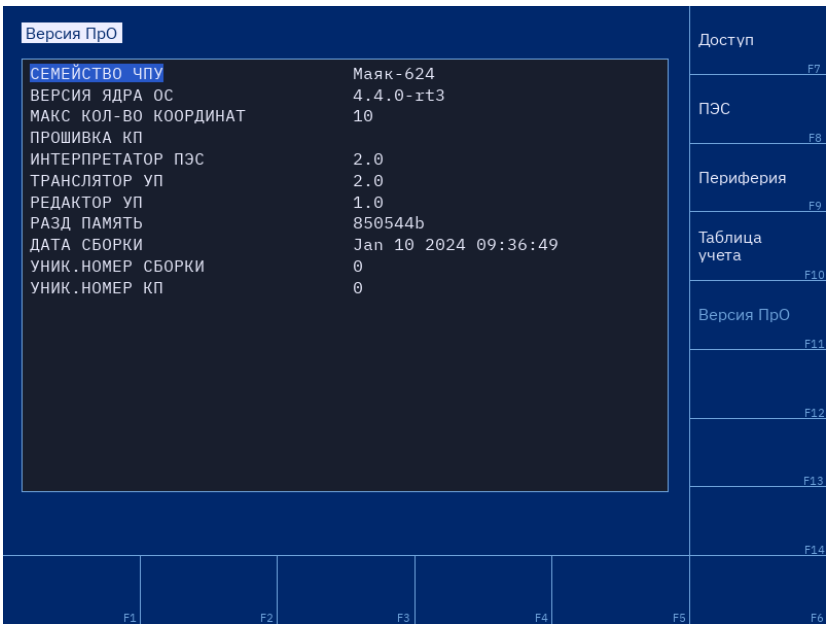

Рисунок 11. Пример работы вида (**view**) в сетке (**grid**)

3.4 Модули

Модули используются для реализации вспомогательных функций УЧПУ, например, осциллограф, редактор, файловые операции и т.п. У каждого модуля есть своё уникальное имя, которое используется в качестве ключевого слова, и индивидуальные параметры. Модуль может располагаться как в сетке **(grid)**, так и в виде **(view)**, и, соответственно, в зависимости от этого меняется синтаксис и количество указываемых параметров.

3.4.1 Модуль в сетке (**grid**)

Рассмотрим сначала модуль в сетке. Обычно они используются в файлах modulesm.dat или tables.dat. Объявление модуля, который будет вызван в сетке, должно быть в одном с ней файле. В сетке может быть сразу несколько модулей. Синтаксис такого модуля вместе с последующей сеткой выглядит следующим образом:

```
{ имя_модуля
      параметры_модуля
}

{ grid имя_сетки ( параметры_сетки )
      { имя_модуля параметры_модуля_связанные_с_сеткой }
      ...
} grid
```

Имена модулей жестко заданы в БПрО.

Так же в модуле в сетке могут располагаться "горячие клавиши":

{ hotkey ***имя_горячей_клавиши*** **{ pressed** ***команда*** **} }.**

Имена "горячих клавиш" задаются в файле hotkeys.dat.

Параметры модуля, связанные с сеткой:

- модули в файле modulesm.dat:
  - **grid_x** расположение модуля (в прямоугольниках сетки) по x;
  - **grid_y** расположение модуля (в прямоугольниках сетки) по y;
  - **grid_w** ширина модуля (в прямоугольниках сетки);
  - **grid_h** высота модуля (в прямоугольниках сетки);
  - **grid_selectable** если указан этот параметр не равный нулю, то это позволяет переключаться между модулями, указанными в сетке (grid);
  - **grid_menu=номер_меню** при переключении на данный модуль будет вызвано указанное меню;
- модули в файле tables.dat:
  - **compact** при параметре равным 0 отображается развернутое описание, при параметре равным 1 отображается сокращенное описание;
  - **format** при параметре равным 0 отображается порядковый номер со значением, при параметре равным 1 порядковый номер не отображается, только значение;
  - **columns** количество отображаемых колонок;
  - **horizontal** при параметре равным 1 отображаются свернутые разряды, а если параметр равен 0 - разряды показываются полностью.

Примеры модуля в сетке из файла modulesm.dat:

```
{ ipperrors (menu=5001 height=4 ratio=0.5) }
```

```
{  ipped  (menu=5000  height=3  ratio=0.5  tab=3  title_bgr=MAIN_BGR  title_h=3.917
     title_start_pos=12) }

{ grid ippeditor ()
     { ipped grid_h=DEF_GRID_H-DEF_IPPERR_H grid_selectable=1 }
     { ipperrors grid_menu=5005 grid_y=DEF_GRID_Y+DEF_GRID_H-DEF_IPPERR_H
          grid_h=DEF_IPPERR_H grid_selectable=1}
     { view (xmax=800 ymax=600 windowed=1 grid_y=0 grid_h=DEF_GRID_Y
          grid_selectable=0 bgr = MAIN_BGR )
          { item # (height=FONTSIZE_MID ratio=0.5 fgr=CHANNEL0)
               0  GAP_V   yes   yes    ' Редактор ПЭС  '          }
          { item txted_name (height=FONTSIZE_MID ratio=0.5 fgr=CHANNEL0
               width=270)
               128  GAP_V   no   yes    '<!26>'          }
          { item txted_pos (height=FONTSIZE_MID ratio=0.5 fgr=CHANNEL0 width=180
               align=TEXT_RIGHT)
               450  GAP_V   no   yes    '<!26>'  }
     } view
     @m28grid.dat@  ; Вывод m28 и m29
} grid
```

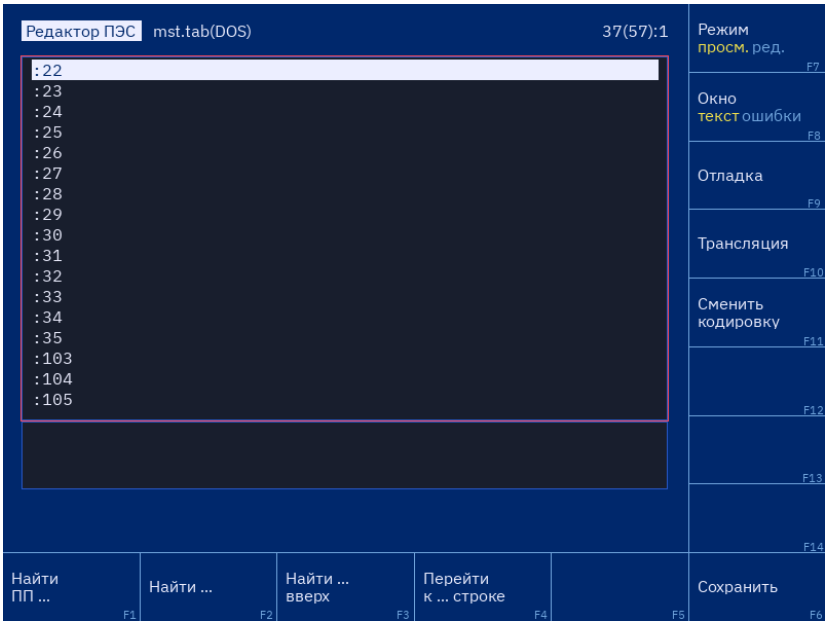

Рисунок 12. Пример модуля в сетке (**grid**)

3.4.2 Модуль в виде (**view**)

Синтаксис модуля в виде выглядит следующим образом:

{ ***имя_модуля параметры_модуля*** **}**

Имена модулей, как и в случае с сеткой, жестко заданы в БПрО. Параметры модуля, связанные с видом, следующие:

- **plugin_x** координата x верхнего левого угла рамки модуля;
- **plugin_y** координата y верхнего левого угла рамки;
- **plugin_w** ширина рамки;

- **plugin_h** высота рамки;
- **max_row** количество строк (данный параметр необходим только для модулей редакторов и индикации УП);
- **max_col** количество столбцов (данный параметр необходим только для модулей редакторов и индикации УП).

Примеры модуля в виде из файла box_up.dat:

```
{ indicator_UP  ;plugin_border=RGB(0,255,0)
      plugin_x=GAP_H  plugin_y=BLOCK_UP_Y+24  plugin_w=466  plugin_h=156  max_row=8
      max_col=40}
```

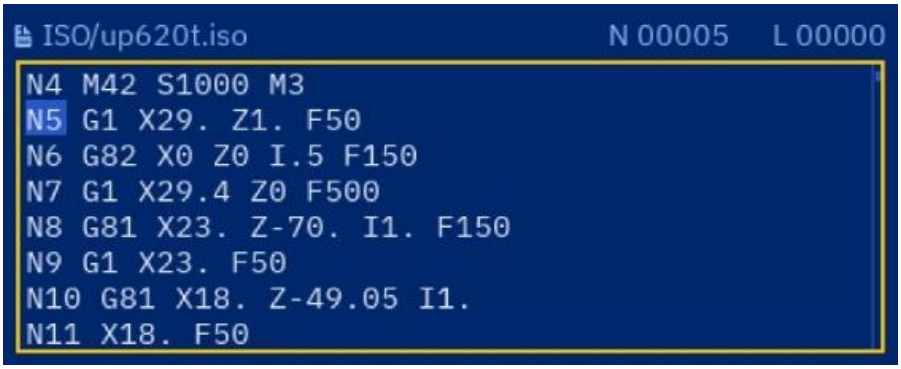

Рисунок 13. Пример модуля (в желтой рамке) в виде (**view**)

3.5 Элемент (item)

Элемент (**item**) — это команда индикации строки, то есть вывод на экран какого-либо текста. Элемент всегда находится в виде (**view**), а его координаты всегда лежат внутри вида. Таким образом, началом координат для элемента является левый верхний угол вида.

Синтаксис элемента (**item**) выглядит следующим образом:

```
{ item имя_элемента ( параметры_элемента )
      координата_x координата_y инверсия_цветов(yes/no) показывать_всегда(yes/no)
      'текст' команды
      {условие}
}
```

Теперь подробнее рассмотрим наполнение элемента и начнем с его имени. В ***имя_элемента*** может быть символ # – "пустое" выражение, для того, чтобы просто или по условию вывести какой-либо текст в указанном месте с заданными характеристиками. В остальных случаях именем элемента является D-ячейка, параметр, элемент массива данных БПрО, и т.д., и выводиться в виде текста на экране будет, соответственно, значение D-ячейки, параметра, элемента массива данных БПрО и т.п.

Параметры элемента, следующие:

- **height** высота строки (кегль) с текстом (задаётся в пикселях);
- цвета (см. пункт 3.8);
- **width** ширина строки с тестом (задаётся в пикселях). Если не задавать ширину, то она установится автоматически по ширине выведенного текста;

- **align** расположение текста в заданной строке:
  - **align=*left*** выровнять текст по левому краю строки;
  - **align=*right*** выровнять текст по правому краю строки;
  - **align=*top*** выровнять текст по верхнему краю строки;
  - **align=*bottom*** выровнять текст по нижнему краю строки;
  - **align=*center*** выровнять текст по центру строки;
  - также в параметре **align** можно совмещать некоторые параметры через знак **+**, например **align=*right+bottom*** – текст строки будет выровнен по правому и нижнему краям строки.
- **fonts** указание, какую гарнитуру использовать из доступных.

Координаты элемента всегда указываются в пикселях. В качестве координат могут выступать как числа, так и константы (см. пункт 3.9), за которыми закреплено число. Началом координат для элемента является левый верхний угол прямоугольника доступной области, которая задаётся в виде. Например, координаты доступной области такие (выделена желтым цветом): координата x = 88 пикселей, координата y = 85 пикселей, ширина 489 пикселей, высота 362 пикселя, а координаты элемента (**item**) (выделено зеленым цветом) будут такими: координата x = 0 пикселей, координата y = 0 пикселей, ширина 81 пикселей, высота 33 пикселя. Код, который задаёт строку с такими координатами выглядит так:

```
{ view (xmax=800 ymax=600 windowed=1 grid_x= 5 grid_y=10 grid_w= 28 grid_h = 43
     bgr=ggg)
     {item # (height=33 width=81 bgr=ttt fgr=EDIT_E_FGR align=center)
           0 0 no yes 'Item'
     }
  } view
```

А выглядеть это будет так:

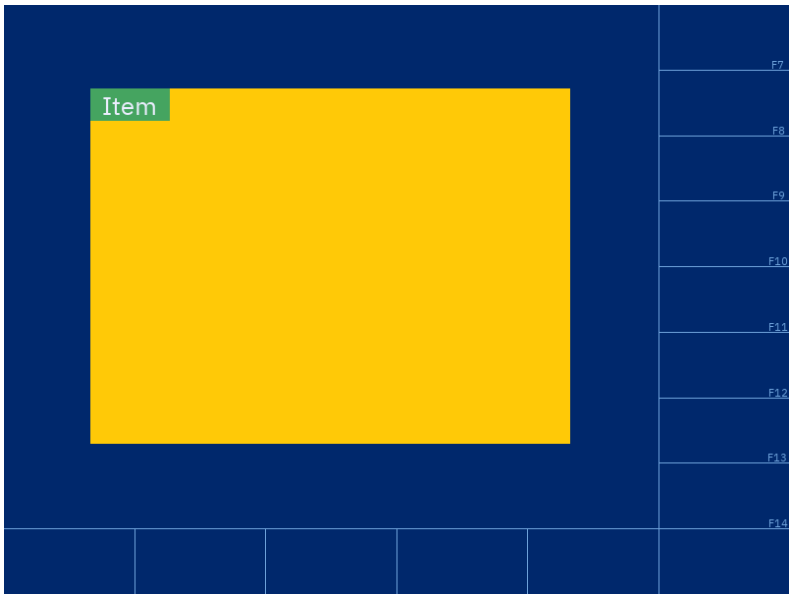

Рисунок 14. Пример элемента (**item**)

Инверсия цветов позволяет поменять местами цвет фона и цвет текста при выводе строки. В приведённом выше коде поменяем параметр инверсии цветов с no на yes, поменяем bgr на invfrg и fgr на invbgr:

```
{ view (xmax=800 ymax=600 windowed=1 grid_x= 5 grid_y=10 grid_w= 28 grid_h = 43
bgr=ggg)
```

```
    {item # (height=33 width=81 invfgr=ttt invbgr=EDIT_E_FGR align=center)
          0 0 yes yes 'Item'
    }
} view
```

И теперь выглядеть это будет так:

Рисунок 15. Пример элемента (**item**) с инверсией цветов

Показывать всегда – это признак индикации строки с текстом, при котором, если значение элемента равно нулю, то:

- `no` – не индицировать;
- `yes` – индицировать.

3.5.1 Текст

Теперь перейдём к тексту элемента. В качестве текста строки может выступать как просто текст (символы латиницы и кириллицы, цифры, символы (~!@#$%^&*()_+{}"<>?[]/\|`№;:?-=.,) кроме символа ', так как он открывает и закрывает блок с текстом), так и специальные форматы для вывода текста. Эти форматы используются для вывода значений D-ячеек, параметров, элементов массива данных БПрО и т.п.:

- **`[]`** будет выводиться просто число, то есть если в элементе задано '[]', а число для вывода равно `12345`, то вывод его на экране будет такой: `12345`, а если число для вывода будет `-12`, то вывод будет такой: `-12`. Данный формат применим как для положительных чисел, так и для отрицательных;
- **`[m]`**, где **`m`** – это количество знаков для вывода. Например, если в элементе задано `'[5]'`, число для вывода равно `12345`, то вывод его на экране будет выглядеть так: `12345`, а если число для вывода будет `12`, то вывод будет такой: `00012`. Данный формат применим только к положительным числам;
- **`[+m]`**, где **`m`** – это количество знаков для вывода, включая символ плюс или минус. Например, если в элементе задано `'[+7]'`, число для вывода `12345`, то вывод числа на экране будет выглядеть так: `+012345`, а если число для вывода будет `-12345`, то вывод будет такой: `-012345`. Данный формат применим как для положительных чисел, так и для отрицательных;
- **`[m.n]`**, где **`m`** – это количество знаков для вывода, включая точку, **`n`** – число знаков после точки. Например, если в элементе задано `'[9.3]'`, число для вывода `12345`, то вывод числа на экране будет выглядеть так: `00012.345`, а если будет `'[9.1]'`, то вывод числа

на экране будет выглядеть так: `0001234.5`. Данный формат применим только к положительным числам;

- **`[+m.n]`** где **`m`** – это количество знаков для вывода, включая знак числа и точку, а **`n`** – это число знаков после точки. Например, если в элементе задано `'[+9.3]'`, число для вывода `12345`, то вывод его на экране будет выглядеть так: `+0012.345`, а если число для вывода будет `-12345`, то вывод его на экране будет выглядеть так: `-0012.345`. Данный формат применим как для положительных чисел, так и для отрицательных;
- **`[m!n]`** где **`m`** – это количество знаков для вывода, **`n`** – число знаков, которые отсекаются с конца выводимого числа. Например, если в элементе задано `'[+6!2]'`, число для вывода `12345`, то вывод на экране будет выглядеть так: `000123`, а если будет `'[6!1]'`, то вывод числа на экране будет выглядеть так: `001234`. Данный формат применим только к положительным числам;
- **`[+m!n]`** где **`m`** – это количество знаков для вывода, **`n`** –число знаков, которые отсекаются с конца выводимого числа. Например, если в элементе задано `'[+6!2]'`, число для вывода равно `12345`, то вывод его на экране будет выглядеть так: `+00123`, а если равно `-12345`, то будет выглядеть так: `-00123`. Данный формат применим как для положительных чисел, так и для отрицательных;
- **`&(a)`** где **`a`** – это либо число, либо математические вычисления. В математических расчётах также можно использовать имена переменных. Например, если в элементе задано `'&(-568)[]'`, то число для вывода будет `-568`, а если задано `'&(m28*1000/m29+55)[10.3]'` и при этом m28 будет иметь значение `172`, а m29 – `10`, то вывод текста будет выглядеть так: `000017.255`. Вместо формата **`[]`** может быть: **`[m]`**, **`[+m]`**, **`[m.n]`**, **`[+m.n]`**, **`[m!n]`**, **`[+m!n]`**. Для всех этих форматов будут действовать характеристики, описанные выше. Так же при использовании **`&(a)`** всегда будет выводиться число **`a`** или результат вычислений, даже если в имени элемента стоит D-ячейка, параметр, элемент массива БПрО и др.;
- **`<m>`**, где **`m`** – это количество символов текста для вывода (если есть заданный текст для вывода, который прописывается в файле txtvals.dat, который находится в корневом каталоге). Например, выводится D-ячейка D38 со значением 172, в элементе будет `'<51>'`, текст для вывода будет: `Не задана зона рассогласования при позиционировании`, то вывод строки на экране будет выглядеть так: `Не задана зона рассогласования при позиционировании`, а если будет `'<25>'`, то вывод на экране будет такой: `Не задана зона рассогласо`.

Все эти форматы можно сочетать. Возьмём, к примеру ту же D38 с тем же значением `172`, а в текстовом поле теперь будет `'![3] <51>'`, то вывод на экране будет выглядеть так: `!172 Не задана зона рассогласования при позиционировании`.

3.5.2 Команды

Команды — это не обязательная часть элемента, но добавляющая функциональность к выводу строки в элементе. Они позволяют изменять значение переменной, которая указана в качестве имени элемента. Синтаксис команды выглядит следующим образом:

`+aa#bb -cc#dd`

Где:

- ***+aa#bb*** – увеличение (**aa** – старт команды, **bb** – стоп команды);
- ***-cc#dd*** – уменьшение (**cc** – старт команды, **dd** – стоп команды).

Для выполнения команды **+aa** (**-cc**) необходимо навести курсор (инверсное изображение) на элемент с помощью клавиш "↑" или "↓" и нажать клавишу "→" ("←"). Команды **bb** (**dd**) выполняются по отжатию клавиши "→" ("←"). Нужные команды необходимо описать в файле.

К примеру, нужно вывести на экран в режиме "Ручной" строку "Позиция" и содержимое ячейки D65 (одна цифра без знака). При нажатии клавиши "→" записать в ячейку D65 значение "1", при нажатии клавиши "←" – значение "2".

В файл, где описываются команды (в стандартном интерфейсе такой файл называется action.dat) необходимо будет записать следующие:

```
{ action dpl d65 = 1 }
{ action dmn d65 = 2 }
```

В файл box_f_nal.dat

```
{ item d65 (height=30 ratio=0.8 fgr=RGB(150,150,150))
      516 66 no yes 'Позиция [1]' +dpl -dmn }
```

При нажатии на клавишу "←" будет индицироваться надпись «Позиция 2», а при нажатии "→" будет надпись «Позиция 1».

Рисунок 16. Пример работы команд в элементе (**item**)

3.5.3 Условие

Условие в элементах не обязательная, но очень полезная часть. Она помогает сделать вывод строки с текстом более вариативным. Так, например, с помощью условий можно менять значения параметров элемента и сам выводимый текст, его индикацию при выделении.

Синтаксис условия выглядит следующим образом:

{ if *выражение_условия новые_значения_параметров* format='*Текст*'}

Условие находится в конце элемента и всегда заключено в фигурные скобки ({}). Само условие всегда начинается со слова **if**, после этого идёт ***выражение_условия***. Выражение условия состоит из: имени переменной, закреплённой за D-ячейкой, параметром, элементом массива БПрО и т.п. (используется в именах многомерных переменных и указывает на одну из ячеек), знака сравнения (> больше, < меньше, = равно, # не равно) и величина, с которым её сравнивают. В качестве величины могут выступать как числа, так и имена других переменных.

Синтаксис для ссылок на ячейки обмена данными БПрО:

Ячейки	Описание	Синтаксис	Пример
D-ячейки	Нумерация начинается с 0	***dномер_D_ячейки***	d70

Параметры	0-299	Одномерный массив параметров	***uномер_парамета***	u12
	300-399	Одномерный массив параметров. В качестве номера параметра используются числа от 0 (0 это параметр 300, 1 это параметр 301 и т.д.)	***cномер_парамета***	c12
	400-499	Многомерный массив параметров. В качестве номера параметра используются числа от 0 (0 это параметр 400, 1 это параметр 401 и т.д.), а в качестве номера бита используется номер координаты (0 это 1-ая координата, 1 это 2-ая координата и т.д.)	***uномер_парамета.номер_бита***	u0.0
	500-599	Многомерный массив параметров. В качестве номера параметра используются числа от 0 (0 это параметр 500, 1 это параметр 501 и т.д.), а в качестве номера бита используется номер координаты (0 это 1-ая координата, 1 это 2-ая координата и т.д.)	***cномер_парамета.номер_бита***	c15.1
Массивы	M	Одномерный массив, нумерация начинается с 0	***mномер_ячейки***	m30
		Многомерный массив, нумерация начинается с 0, а в качестве номера бита используется номер координаты (0 это 1-ая координата, 1 это 2-ая координата и т.д.)	***mномер_ячейки.номер_бита***	m1.0
	I	Одномерный массив, нумерация начинается с 0	***iномер_ячейки***	i2
		Многомерный массив, нумерация начинается с 0, а в качестве номера бита используется номер координаты (0 это 1-ая координата, 1 это 2-ая координата и т.д.)	***iномер_ячейки.номер_бита***	i6.1
	P	Одномерный массив, нумерация начинается с 0	***pномер_ячейки***	p51

Пример выражения условия: **m28=0,** где m28 – это имя переменной, = — это знак сравнения, 0 – это величина, с которой сравнивают значение **m28.**

Выражения условий можно комбинировать с помощью логических операций И (**&&**) и ИЛИ **||**, например: **((p7=0)&&(p30=p2))||((p7=1)&&(p30=p8)).**

< меньше	Используется в выражении условия.
> больше	
= равно	
# не равно	
&& логическое И	Используется для комбинации выражений условия
|| логическое ИЛИ	

В **format** указывается формат выводимой по условию строки. Для этого можно использовать все форматы и их сочетания, которые используются в самом элементе. Например: **format=' ЧПУ: Готово'** - просто вывод текста, **format='&(m17*1000)/i34[6.3]'** – вывод числа.

Теперь рассмотрим несколько примеров элемента:

- formavt.dat (элемент находится в виде, который, в свою очередь, находится в составе формы) – экран режима «Автомат»:

```
{ item # ()                                                    ;№1
     GAP_H GAP_V  yes   yes   ' Автомат  ' }
{ item # (invbgr=Error_C invfgr=MAIN_BGR width=105)            ;№2
     456   16     yes   no    ''
     { if m30=0 invfgr=MAIN_BGR format=' ЧПУ: Авария  ' }}
{ item # (invbgr=Error_C invfgr=MAIN_BGR width=106)            ;№3
     456   16     yes   no    ''
     { if m30=255 invbgr=GREEN format=' ЧПУ: Готово  ' }}
{ item m20 (fgr=B3  bgr=B7 width=160)
     111   GAP_V  no    yes   '           '
     { if m20#0 format='<24>'} } ; Статус отработки УП
```

Рисунок 17. item # под номером 1

Рисунок 18. `item #` под номером 2 при выполнении условия, что `m30=0`

Рисунок 19. `item #` под номером 3 при выполнении условия, что `m30=255`

Рисунок 20. `item m20` при выполнении условия

Рисунок 21. `item m20` при невыполнении условия

3.5.4 Специальные элементы

Так же есть специальные элементы:

- time – часы;
- up – имя файла с УП;
- coment – комментарий к УП;
- edit_name;
- up_gr;
- txted_pos – позиция курсора в тексте;
- txted_name – название редактируемого файла;
- diskfree – размер свободного дискового пространства;
- txted.

Элементы (item), используемые в диалоге (фрезерный вариант):

- Svar
- Mvar
- WWvar
- YYvar
- XXvar
- Wvar
- Yvar
- Xvar
- Qvar
- MMMvar
- MMEFvar
- RRvar
- MEFvar
- Rvar
- MMvar
- AAvar
- Dvar
- AAAvar
- Fvar
- VFIZMvar
- VDIZMvar
- VKIZMvar
- VHIZMvar
- VWIZMvar
- VUIZMvar
- VCIZMvar
- VBIZMvar
- VAIZMvar
- ZZIZMvar
- YYIZMvar
- XXIZMvar
- GIZMvar
- Nvar
- VQIZMvar
- VVIZMvar
- PIZMvar
- HIZMvar
- ZIZMvar
- YIZMvar
- XIZMvar
- Pvar
- VSvar
- GVvar
- GSvar
- Gvar
- VVHvar
- IVvar
- HVvar
- ICCvar
- VACvar
- VUCvar
- VBvar
- VWvar
- JCCvar
- VKvar
- VXvar
- Cvar
- VWWvar
- VUvar
- JCvar
- ICvar
- VAvar
- PPPvar
- KKvar
- HHvar
- VLvar
- VFvar
- Bvar
- VVCvar
- PPvar
- VDvar
- VHvar

- PMvar
- Avar
- Kvar
- Jvar
- Ivar
- GIvar
- GPvar
- Hvar
- PAGEvar
- PRNvar
- Zvar

Элементы, используемые в диалоге (токарный вариант):

- PRNvar
- Nvar
- Gvar
- GSvar
- GVvar
- GIvar
- Xvar
- Zvar
- Rvar
- Hvar
- Fvar
- Pvar
- Ivar
- Jvar
- Kvar
- VVXvar
- VVZvar
- Bvar
- VHIZMvar
- VDIZMvar
- VWIZMvar
- Dvar
- FFvar
- Cvar
- Avar
- PMvar
- VCvar
- VHvar
- VPvar
- VZvar
- VXvar
- VSvar
- VKvar
- IIvar
- JJvar
- KKvar
- KFvar
- CCvar
- IIIvar
- VUIZMvar
- VKIZMvar
- VVIZMvar
- KKKvar
- ZKvar
- XKvar
- CCCvar
- AAAvar
- PPvar
- PPPvar
- VLvar
- VVCvar
- VFvar
- HHvar
- VDvar
- PCvar
- IJvar
- JIvar
- VUvar
- VVPvar
- Qvar
- IIZMvar
- JIZMvar
- HVVvar
- Wvar
- XXvar
- ZZvar
- AAvar
- WWvar
- RRvar
- Mvar
- MMvar
- MMMvar
- Svar
- MEFvar
- MMEFvar
- PAGEvar
- GIZMvar
- XIZMvar
- ZIZMvar

3.6 Графические примитивы

Графические примитивы представляют собой базовые геометрические формы, используемые для создания более сложных объектов и изображений.

3.6.1 Залитый прямоугольник (box)

Залитый прямоугольник (**box**) – это команда для индикации залитого прямоугольника с заданными параметрами и координатами вывода. Он всегда находится в виде, его координаты задаются в пикселях и всегда лежат внутри вида. Таким образом началом координат для залитого прямоугольника является левый верхний угол вида.

Синтаксис залитого прямоугольника выглядит следующим образом:

{ box color=*цвет_прямоугольника* ifcolors='*цвет_прямоугольника* if *условие*' x=*координата_x* y=*координата_y* x2=*ширина* y2=*высота*}

Где:

- ***цвет_прямоугольника*** – цвет, которым будет залит прямоугольник, может задаваться:
  - константой, которой задан цвет в файле (в стандартном интерфейсе такой файл называется colors.dat);
  - через цветовую систему RGB в формате: **color = RGB(*RR,GG,BB*)**, каждое из значений RR, GG и BB принимает числовое значение от 0 до 255;
- **ifcolors** – параметр (необязательный), который позволяет менять цвет залитого прямоугольника в зависимости от условия, где:
  - ***цвет_прямоугольника*** (см. предыдущий параметр);
  - ***условие*** составляется аналогично выражению условия в элементе (**item**);
- ***координата_x*** – в качестве координат могут выступать как числа, так и константы (см. пункт 3.9), за которыми закреплено число, если меняется значение константы, то, соответственно меняется и координата;
- ***координата_y –*** в качестве координат могут выступать как числа, так и константы (см. пункт 3.9), за которыми закреплено число, если меняется значение константы, то, соответственно меняется и координата;
- ***ширина*** – ширина залитого прямоугольника (задаётся в пикселях);
- ***высота*** – высота залитого прямоугольника (задаётся в пикселях).

Пример залитого прямоугольника:

```
{ box  color=RGB(255, 205, 0) x=506 y=112  x2=142 y2=84   }
```

Рисунок 22. Пример работы залитого прямоугольника (черный цвет)

3.6.2 Прямоугольник (rect)

Прямоугольник (**rect**) – это команда для индикации рамки прямоугольника с заданными параметрами и координатами вывода. Он всегда находится в виде, его координаты задаются в пикселях и всегда лежат внутри вида. Таким образом началом координат для прямоугольника является левый верхний угол вида.

Синтаксис прямоугольника (**rect**) выглядит следующим образом:

```
{rect color=цвет_прямоугольника ifcolors='цвет_прямоугольника if условие'
     x=координата_x y=координата_y x2=ширина y2=высота}
```

Где:

- ***цвет_прямоугольника*** – цвет рамки прямоугольника, может задаваться:
  - константой, которой задан цвет в файле (в стандартном интерфейсе такой файл называется colors.dat);

- через цветовую систему RGB в формате: **`color = RGB(`*`RR,GG,BB`*`)`**, каждое из значений RR, GG и BB принимает числовое значение от 0 до 255;

- **`ifcolors`** – параметр (необязательный), который позволяет менять цвет прямоугольника в зависимости от условия, где:
  - ***`цвет_прямоугольника`*** (см. предыдущий параметр);
  - ***`условие`*** составляется аналогично выражению условия в элементе (**item**);
- ***`координата_x`*** – в качестве координат могут выступать как числа, так и константы (см. пункт 3.9), за которыми закреплено число, если меняется значение константы, то, соответственно меняется и координата;
- ***`координата_y`*** – в качестве координат могут выступать как числа, так и константы (см. пункт 3.9), за которыми закреплено число, если меняется значение константы, то, соответственно меняется и координата;
- ***`ширина`*** – ширина прямоугольника (задаётся в пикселях);
- ***`высота`*** – высота прямоугольника (задаётся в пикселях).

Пример прямоугольника:

```
{ rect   color= RGB(255, 205, 0)   x=300   y=10   x2=50   y2=25 }
```

Рисунок 23. Пример работы прямоугольника (**rect**) (желтая рамка)

3.6.3 Линия (line)

Линия (**line**) – это команда для индикации линии с заданными параметрами и координатами вывода. Она всегда находится в виде, ее координаты задаются в пикселях и всегда лежат внутри вида. Таким образом началом координат для линии является левый верхний угол вида.

Синтаксис линии (**line**) выглядит следующим образом:

`{line color=`*`цвет_линии`* **`ifcolors='`*`цвет_линии`*** **`if`** ***`условие`*****`'`**
`x=`*`координата_x_начала_линии`* **`y=`*`координата_y_начала_линии`***
`x2=`*`координата_x_конца_линии`* **`y2=`*`координата_y_конца_линии`*****`}`**

Где:

- ***`цвет_линии`*** – цвет линии, может задаваться:
  - константой, которой задан цвет в файле (в стандартном интерфейсе такой файл называется colors.dat);
  - через цветовую систему RGB в формате: **`color = RGB(`*`RR,GG,BB`*`)`**, каждое из значений RR, GG и BB принимает числовое значение от 0 до 255;
- **`ifcolors`** – параметр (необязательный), который позволяет менять цвет линии в зависимости от условия, где:

- ***цвет_линии*** (см. предыдущий параметр);
- ***условие*** составляется аналогично выражению условия в элементе (**item**);

- ***координата_x_начала_линии*** – в качестве координат могут выступать как числа, так и константы (см. пункт 3.9), за которыми закреплено число, если меняется значение константы, то, соответственно меняется и координата;
- ***координата_y_начала_линии*** – в качестве координат могут выступать как числа, так и константы (см. пункт 3.9), за которыми закреплено число, если меняется значение константы, то, соответственно меняется и координата;
- ***координата_x_конца_линии*** – в качестве координат могут выступать как числа, так и константы (см. пункт 3.9), за которыми закреплено число, если меняется значение константы, то, соответственно меняется и координата;
- ***координата_y_конца_линии*** – в качестве координат могут выступать как числа, так и константы (см. пункт 3.9), за которыми закреплено число, если меняется значение константы, то, соответственно меняется и координата.

Пример линии (под цифрой **1** – начало линии, **2** – конец линии):

```
{ line color= RGB(255, 205, 0) x=300 y=10 x2=350 y2=50 }
```

Рисунок 24. Пример работы линии (**line**)

3.6.4 Изображения

Инструкция по работе с изображениями подробно описана в инструкции «Графика в индикации и меню».

3.7 Вложенные файлы

Вложенные файлы – это ещё один очень удобный и полезный инструмент данного языка разметки. С помощью него можно разбить части кода на отдельные небольшие файлы, которые потом можно использовать сразу в нескольких других файлах. С помощью вложенных файлов код становится более компактным и читабельным.

Во вложенных файлах используются те же иерархии, как и в остальных файлах. Для того, чтобы вложить один файл в другой, необходимо между двумя знаками `@` без пробелов указать название файла, который необходимо вложить, с его расширением: ***@имя_файла.расширение@***. Например: `@m28grid.dat@` (файл с выводом ячеек D38 и D39).

Рассмотрим пример из файла formavt.dat:

```
@box_name_s.dat@     ; Названия осей плюс признаки выхода в ноль и в точку
@box_zgt_avt_s.dat@  ; Значения "Заготовка"
@box_ost_s.dat@      ; Значения "Ост. пути"
@box_st_s.dat@       ; Значения "Станок"
@box_rsg_s.dat@      ; Значения "Рассогласование"
@box_f_avt.dat@      ; Зона F
@box_s_avt.dat@      ; Зона S
@box_r.dat@          ; Зона БХ
@box_t.dat@          ; Зона T
@box_gm.dat@         ; Зона с G и M
@box_mhv.dat@        ; Зона маховика
@box_up.dat@         ; Блок УП
@m28form.dat@        ; m28 и m29
```

3.8 Параметры

Почти во всех разделах есть параметры, которые задаются идентичным образом. Вот список таких параметров и методы их задачи:

- размер доступной области в пикселях, для того чтобы все остальные параметры, заданные в пикселях, нормировались к указанному размеру:
  - **xmax** размер доступной области по **x** в пикселях;
  - **ymax** размер доступной области по **y** в пикселях;
- **bgr** цвет доступной области, может задаваться:
  - константой, которой задан цвет в файле (в стандартном интерфейсе такой файл называется colors.dat);
  - через цветовую систему RGB в формате: **brg = RGB(*RR,GG,BB*)**, каждое из значений RR, GG и BB принимает числовое значение от 0 до 255;
- **fgr** цвет символов в доступной области, может задаваться:
  - константой, которой задан цвет в файле (в стандартном интерфейсе такой файл называется colors.dat);
  - через цветовую систему RGB в формате: **frg = RGB(*RR,GG,BB*)**;
- **invbgr** цвет курсора, может задаваться:
  - константой, которой задан цвет в файле (в стандартном интерфейсе такой файл называется colors.dat);
  - через цветовую систему RGB в формате: **invbrg = RGB(*RR,GG,BB*);**
- **invfgr** цвет текста в курсоре, может задаваться:
  - константой, которой задан цвет в файле (в стандартном интерфейсе такой файл называется colors.dat);
  - через цветовую систему RGB в формате: **invfrg = RGB(*RR,GG,BB*).**

3.9 Константы

Константы позволяют облегчить написание кода, а именно они дают возможность задать какое-либо значение и использовать его в дальнейшем в других файлах. Изменяя значение константы в месте, где она задается, изменяется ее значение и в тех местах, где она используется. С помощью констант возможно задавать как числовые значения, так и цветовые.

Синтаксис для задания численных констант:

#имя_константы=численное_значение

Пример:

```
#Height_1=34
```

Синтаксис для задания цветовых координат:

#имя_константы=код_RGB

Пример:

```
#Red=RGB(255,0,0)
```

Также возможно задание констант с помощью других констант, это возможно как для числовых констант, так и для цветовых.

Пример:

```
#Const_2=Const_1
```

В стандартном интерфейсе для задания констант используются следующие файлы:

- defaults.dat – используется для задания численных констант, таких как замер гарнитуры, координаты вывода, ширину строки;
- colors.dat – используется для задания цветовых констант;
- файлы для расчета знакоместа координат:
    - krd-2.dat – для двух координат (фрезерный вариант);
    - krd-3.dat – для трех координат (фрезерный вариант);
    - krd-4.dat – для четырех координат (фрезерный вариант);
    - krd-5.dat – для пяти координат (фрезерный вариант);
    - krd-t.dat – для двух координат (токарный вариант).

4 Меню (menu)

Клавиши меню могут выполнять различные функции в зависимости от контекста. Конкретные функции и внешний вид клавиш меню могут варьироваться в зависимости от пользовательского интерфейса, программного обеспечения и настроек системы.

На экране всегда присутствуют два ряда меню: горизонтальный и вертикальный. Горизонтальный ряд задействует клавиши с F1 по F6, а вертикальный – с F7 по F14. Так же есть специальные клавиши для расширения меню: клавиша Ctrl и клавиша Alt. Смены меню происходят независимо от остальных зон согласно информации, указанной в файлах пользовательского интерфейса.

Синтаксис меню выглядит следующим образом:

```
{ menu номер_меню
   { f1 (access) { text "Текст" ( параметры ) { условие } } { pressed команда }}
   …
} menu
```

Где:

- ***номер_меню*** у каждого меню уникальный;
- **f1 … f6** (`altf1 … altf6` при нажатии клавиши `Alt`, `ctrlf1 … ctrlf6` при нажатии клавиши `Ctrl`) описания клавиш горизонтального меню. Если необходимо сделать какую-либо клавишу не активной, то можно её пропустить и не указывать;
- **f7 … f14** (`altf7 … altf14` при нажатии клавиши `Alt`, `ctrlf7 … ctrlf14` при нажатии клавиши `Ctrl`) описания клавиш вертикального меню. Если необходимо сделать какую-либо клавишу не активной, то можно её пропустить и не указывать;
- **access** – это уровень доступа, при котором будет доступна клавиша (необязательный параметр);
- **text "*Текст*"** – указывается выводимый текст на клавишах меню. Для разбиения текста на строки необходимо использовать знак $, например так: "Запись$в файл". В описании клавиши может быть несколько **text**;
- параметры (необязательный параметр) используются для задания цвета символов (**fgr**) и цвета фона клавиши (**bgr**):
  - **bgr** (см. пункт 3.8);
  - **fgr** (см. пункт 3.8);
- условие (необязательный параметр) работает похожим образом, как и в элементе, но только без указания `format`. Пример: `{ if (d317=4)||(d318=4) fgr=CPARAM };` Часто условия в меню используются для индикации (изменения цвета текста) нажатия клавиши;
- **pressed команда** – действие, которое будет выполнено после нажатия клавиши;
- **released команда** – действие, которое будет выполнено после отжатия клавиши (необязательный параметр);

В **pressed** и **released** используются специальные команды, с помощью которых можно переключаться на определенное меню, сетку или модуль, также возможно задание

новых значений для D-ячеек, параметров и ячеек массивов БПрО. Ниже приведен список таких команд:

- **mm** – переключение на предыдущее меню;
- **mmномер_меню –** переключение на указанное меню, например переключение на 10 меню: mm10;
- вызов сетки, синтаксис будет следующим: **pressed *название_сетки*** или **released *название_сетки***;
- вызов модуля, синтаксис будет следующим: **pressed *имя_модуля*** или **released *имя_модуля***;
- задание нового значения для D-ячейки, параметра или ячейки массивов БПрО;
- **inputkill –** очистить наборную строку;
- команды с условием могут быть такими:
    - **ifok** используется для команд, если условие выполнено. Синтаксис **ifok** следующий:

      %=условие* ifok *число_команд команда_1* 0 *команда_2

      Где:
        - ***условие*** – это выражение сравнения, неравенства;
        - ***число_команд*** – это количество ***команда_1*** плюс ***команда_2***;
        - ***команда_1*** – это команда, которая выполняется, если условие ложно; таких команд может быть несколько или не быть вовсе, тогда синтаксис будет следующим: ***%=условие* ifok 0 *команда_2***;
        - ***команда_2*** – это команда (она может быть только одна), которая выполняется, если условие истинно;

    - **ifnotok** используется для команд, если условие не выполнено. Синтаксис следующий:

      %=условие* ifnotok *число_команд команда_1* 0 *команда_2

      Где:
        - ***условие*** – это выражение сравнения, неравенства;
        - ***число_команд*** – это количество ***команда_1*** плюс ***команда_2***;
        - ***команда_1*** – это команда, которая выполняется, если условие истинно; таких команд может быть несколько или не быть вовсе, тогда синтаксис будет следующим: ***%=условие* ifnotok 0 *команда_2***;
        - ***команда_2*** – это команда (она может быть только одна), которая выполняется, если условие ложно;

Рассмотрим пример меню из файла menu-me.dat:

```
{ menu 1070 ; oscilloscope - просмотр сигнала
   { f1 { text "Просмотр"              }
        { text "$сигнал"  ( fgr =CPARAM)  }
        { text "$память"  ( fgr =CPARAM3) }
        { text " файл"    ( fgr =CPARAM3) }
        { pressed view_oscilloscope_buffer mm1071 }
   }
   { Ctrlf6 { text "Возврат" }    { pressed next }  }
} menu

{ menu 1070 ; oscilloscope - просмотр сигнала

    { f7 { text "Копировать$память$в файл 1" }
         { pressed "1" save_buffers }
    }
    { f8 { text "Пуск записи$" }
        { text "1" (fgr=CPARAM3)
           { if d404=0 fgr=MENU_FGR }
        }
        { text " 2" (fgr=CPARAM3)
           { if d404=1 fgr=MENU_FGR }
        }
        { text " 3" (fgr=CPARAM3)
           { if d404=2 fgr=MENU_FGR }
        }
        { text " 4" (fgr=CPARAM3)
           { if d404=3 fgr=MENU_FGR }
        }
        { text " 5" (fgr=CPARAM3)
           { if d404=4 fgr=MENU_FGR }
        }
         { ; записываем в наборную строку содержимое ячейки d404
           pressed %=d404#0 ifok 3
                     "1" start_log 0
                   %=d404#1 ifok 3
                     "2" start_log 0
                   %=d404#2 ifok 3
                     "3" start_log 0
                   %=d404#3 ifok 3
                     "4" start_log 0
                   %=d404#4 ifok 3
                     "5" start_log 0
         }
    }

   { f9 { text "Тип входа$" }
        { text "=" (fgr=CPARAM3)
            { if d405=0 fgr=CPARAM }
        }
        { text " ~" (fgr=CPARAM3)
            { if d405=1 fgr=CPARAM }
        }
        { pressed %=d405=0 ifok 2 d405=0 0 d405=1, switch_mode }
   }
} menu
```

Рисунок 25. Пример работы меню

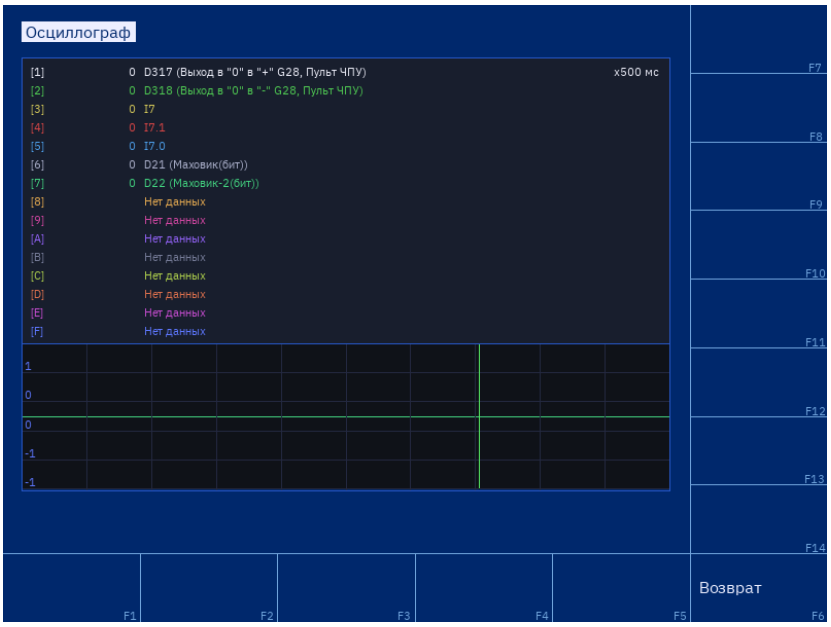

Рисунок 26. Пример работы меню при нажатии клавиши `Ctrl`

Среди всех остальных видов меню можно выделить меню переключения режимов. Его можно вызывать из любого экрана вне зависимости от отображаемого меню нажатием клавиши `РЕЖИМ` (Tab на компьютерной клавиатуре), деактивируется после повторного нажатия на клавишу `РЕЖИМ`. Меню переключения режимов использует только горизонтальный ряд меню.

Пример Меню переключения режимов из файла menu-5e.dat:

```
{ menu 0 ; tab-меню
   { f1 { text "Ручной" }           { pressed !s1-0 form mm } }
   { f2 { text "Автомат" }          { pressed !s0-0 form mm } }
   { f3 { text "Выход$в ноль" }     { pressed !s2-0 form mm } }
   { f4 { text "Выход$в точку" }    { pressed !s6-0 form mm } }
   { f5 { text "Фиксир.$перемещ."}  { pressed !s8-0 form mm d406=1 } }
   { f6 { text "Преднабор" }        { pressed !s7-0 form mm } }
   { altf3 (access=-1) { text "Выполнить" } { pressed execute } }
} menu
```

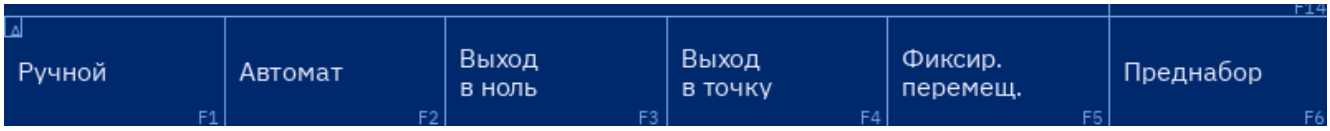

Рисунок 27. Пример работы меню переключения режимов

Рисунок 28. Пример работы меню переключения режимов при нажатии клавиши `Alt`

5 Параметры по умолчанию (default)

Блок параметров по умолчанию (**default**) может быть в каждом файле и располагаться в его начале. В этом блоке можно указать все значения параметров, которые будут применены ко всем нижеследующим частям файла (**form**, **view**, **item**, **menu** и т.д.), что в значительной мере облегчит написание файла. Параметры, которые были указаны в параметрах по умолчанию, можно не указывать в параметрах самого блока (**form**, **view**, **item**, **menu** и т.д.). Синтаксис параметров по умолчанию выглядит следующим образом:

```
{ default
      …
} default
```

Пример параметров по умолчанию из файла modulesm.dat:

```
{ default
  xmax = 100
  ymax = 100
  height = 3
  fgr =EDIT_E_FGR
  bgr =EDIT_E_BGR
  border =EDIT_E_BORD
  describe =1
  grid_rows = DEF_GRID_ROWS_2
  grid_cols = DEF_GRID_COLS
  grid_border = MAIN_BGR
  grid_bgr = MAIN_BGR
  grid_x = DEF_GRID_X
  grid_y = DEF_GRID_Y
  grid_w = DEF_GRID_W
  grid_h = DEF_GRID_H
  grid_selectable = 0
  invfgr = MAIN_BGR
  invbgr = CHANNEL0
} default
```

6 Комментарии

Комментарии используются для написания аннотаций и для временного отключения части кода. Язык разметки допускает использование только однострочных комментариев. Комментарий начинается с символа **;** и продолжается до конца строки.

Пример комментариев из файла formavt.dat (сами комментарии выделены зелёным цветом):

- Аннотация
  - ```
    ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
    ; ABTOMAT ++++++++++++++++++++++++++++++++++++++++++++++++++++
    ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
    ```
  - ```
    @box_zgt_avt_s.dat@  ; Значения "Заготовка"
    ```
  - ```
    @box_ost_s.dat@      ; Значения "Ост. пути"
    ```
- Временное отключение части кода
  - ```
    ; { item m30 (fgr=B1)
          ; 102  GAP_V   no   yes 'ЧПУ' }
    ; { item m30 (fgr=Error_C)
          ; 102+39  GAP_V   no   yes 'Авария'
          ; { if m30=255 fgr=GREEN format='Готово' }   }
    ```
  - ```
    { view ();bgr=RED)
    ```
    ```

7 Ошибки

Ошибки могут возникнуть в том случае, если были допущены ошибки в словах, опечатки, были указаны переменные, которые ранее нигде не объявлялись, и, если какой-либо элемент или примитив выходит за границы доступной области.

Теперь рассмотрим примеры:

- ошибка в файле formavt.dat – пропущена буква в слове **view** (ошибка выделена красным):

```
{ form 0 (menu=10, xmax=665, ymax=600)
   { hotkey KEY_ENTER { pressed d_set_ust } }
   { viw ();bgr=RED)
      @clock.dat@
      { item # ()
```

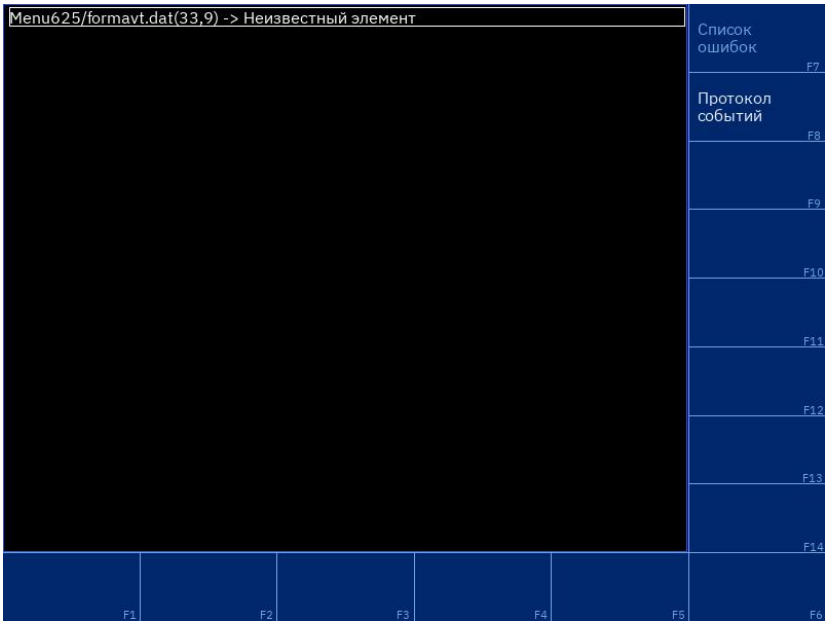

Рисунок 29. Пример вывода ошибки

При запуске графического интерфейса выходит такое сообщение об ошибке, в котором указан путь к файлу с ошибкой, его имя, расположение ошибки (в данном случае это 33 строка 9 символ), и тип ошибки.

- ошибка в файле m28grid.dat – координата x у элемента **(item)** выходит за пределы доступной области (ошибка выделена красным):

```
{ view (xmax=800 ymax=600 windowed=1 grid_x=28 grid_y=59 grid_w=9 grid_h=3
grid_selectable=0 bgr = MAIN_BGR)
{ item m28 (height=FONTSIZE_MID fgr=Messag_C width=50)
      400 3   no   yes    '![3]'
      { if m28=0 fgr=MAIN_BGR} }
{ item m29 (height=FONTSIZE_MID fgr=Error_C width=50)
      74 3   no   yes    '?[3]'
      { if m29=0 fgr=MAIN_BGR}}}
```

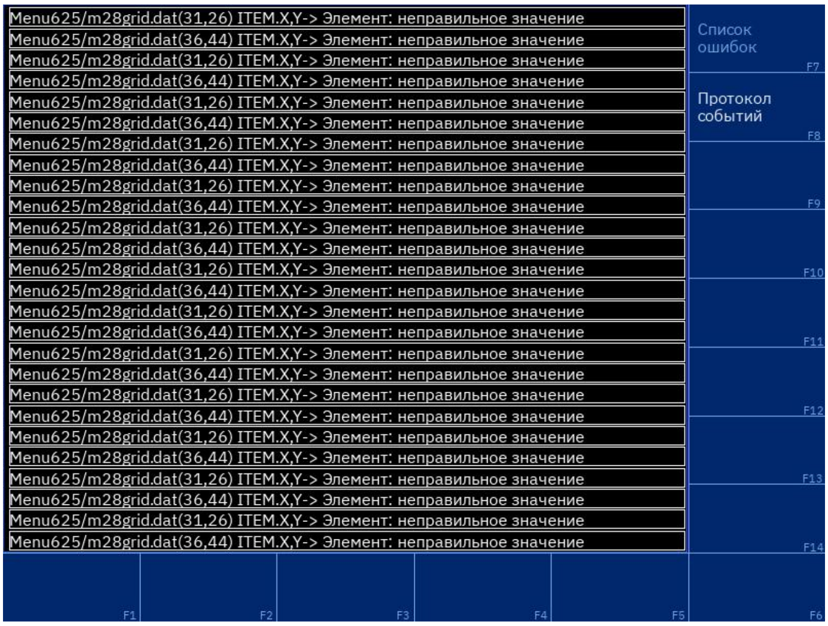

Рисунок 30. Пример вывода ошибок

При запуске графического интерфейса выходит такое сообщение об ошибке, в котором указан путь к файлу с ошибкой, его имя, расположение ошибки и тип ошибки.

8 Файлы и их назначения

В этом разделе приведен список файлов, которые используются в интерфейсе, идущему в базовой поставке УЧПУ "Маяк".

№	Имя файла	Описание	Ссылка из файла
1	action.dat	Команды по нажатию клавиш «←», «→», «↑», «↓», <ВВОД> (<Enter>)	maya-te.dat maya-Xe.dat
2	box_f_avt.dat	Блок F для автоматических режимов	formavt.dat formprn.dat
3	box_f_nal.dat	Блок F для режимов «Ручной», «Выход в ноль», «Выход в точку», «Фиксированные перемещения»	formnal_X.dat* formvi0_X.dat* formvit_X.dat* formfix_X.dat*
4	box_fix.dat	Величина фиксированного перемещения	formfix_X.dat*
5	box_gm.dat	Блок G для автоматических режимов	formavt.dat formprn.dat
6	box_mhv.dat	Блок маховика для режимов «Ручной» и «Автомат»	formnal_X.dat* formavt.dat
7	box_name_s.dat	Названия осей и признаки выхода в ноль и в точку (для основных страниц режимов)	formnal_X.dat* formavt.dat formvi0_X.dat* formvit_X.dat* formfix_X.dat* formprn.dat
8	box_name_s_2.dat	Названия осей и признаки выхода в ноль и в точку (для дополнительных страниц режимов)	formnal_X.dat* formavt.dat formvi0_X.dat* formvit_X.dat* formfix_X.dat* formprn.dat
9	box_ost_fix.dat	Значения «Остаток пути» для режима «Фиксированные перемещения»	formfix_X.dat*
10	box_ost_s.dat	Значения «Остаток пути»	formavt.dat formfix_X.dat* formprn.dat
11	box_prn.dat	Блок УП для режима «Пред набор»	formprn.dat
12	box_r.dat	Блок быстрого хода	formnal_X.dat* formavt.dat formvi0_X.dat* formvit_X.dat* formfix_X.dat* formprn.dat
13	box_rsg.dat	Значения «Рассогласование»	formnal_X.dat* formvi0_X.dat* formvit_X.dat* formfix_X.dat*
14	box_rsg_s.dat	Значения «Рассогласование»	formnal_X.dat* formavt.dat formvi0_X.dat* formvit_X.dat* formfix_X.dat* formprn.dat

15	box_s_avt.dat	Блок S для автоматических режимов	formavt.dat formprn.dat
16	box_s_nal.dat	Блок S для режимов «Ручной», «Выход в ноль», «Выход в точку», «Фиксированные перемещения»	formnal_X.dat* formvi0_X.dat* formvit_X.dat* formfix_X.dat*
17	box_st_and_rsg_avt.dat	Значения «Станок» и «Рассогласование» для автоматических режимов (для дополнительного экрана)	formavt.dat formprn.dat
18	box_st_s.dat	Значения «Станок»	formnal_X.dat* formavt.dat formvi0_X.dat* formvit_X.dat* formfix_X.dat* formprn.dat
19	box_t.dat	Блок T	formnal_X.dat* formavt.dat formvi0_X.dat* formvit_X.dat* formfix_X.dat* formprn.dat
20	box_up.dat	Блок УП для режима «Автомат»	formavt.dat
21	box_up_2.dat	Блок УП для режима «Автомат» (для дополнительных страниц)	formavt.dat
22	box_zgt_and_ost_avt.dat	Значения «Заготовка» и «Остаток пути» для автоматических режимов	formavt.dat formprn.dat
23	box_zgt_and_st_avt.dat	Значения «Заготовка» и «Станок» для режима «Преднабор» (используется в привязке инструмента)	formprn.dat
24	box_zgt_and_st_nal.dat	Значения «Заготовка» и «Станок» для режима «Ручной» (используется в привязке инструмента)	formnal_X.dat*
25	box_zgt_avt_s.dat	Значения «Заготовка» для автоматических режимов	formavt.dat formprn.dat
26	box_zgt_fix_s.dat	Значения «Заготовка» для режимов «Выход в ноль», «Выход в точку», «Фиксированные перемещения»	formvi0_X.dat* formvit_X.dat* formfix_X.dat*
27	box_zgt_nal_s.dat	Значения «Заготовка» для режима «Ручной»	formnal_X.dat*
28	ciklf.dat	Список циклов в диалоге (фрезерный вариант)	dialfrz.dat
29	ciklt.dat	Список циклов в диалоге (токарный вариант)	dialtok.dat
30	clock.dat	Часы	formnal_X.dat* formavt.dat formvi0_X.dat* formvit_X.dat* formfix_X.dat* formprn.dat
31	colors.dat	Назначение цветов	maya-te.dat maya-Xe.dat
32	defaults.dat	Числовые константы (координаты, размеры шрифтов, разбиения сетки (grid))	maya-te.dat maya-Xe.dat

33	dialfrz.dat	Диалог (фрезерный вариант для 2, 3 ,4 и 5 координат)	maya-Xe.dat
34	dialtok.dat	Диалог (токарный вариант)	maya-te.dat
35	errors.cfg	Текстовые сообщения и ошибки из ПЭС	txtvals.dat
36	formavt.dat	Описание страниц индикации в режиме «Автомат»	maya-te.dat maya-Xe.dat
37	formfix_2.dat	Описание страниц индикации в режиме «Фиксированные перемещения» (2 координаты)	maya-te.dat maya-2e.dat
38	formfix_3.dat	Описание страниц индикации в режиме «Фиксированные перемещения» (3 координаты)	maya-3e.dat
39	formfix_4.dat	Описание страниц индикации в режиме «Фиксированные перемещения» (4 координаты)	maya-4e.dat
40	formfix_5.dat	Описание страниц индикации в режиме «Фиксированные перемещения» (5 координаты)	maya-5e.dat
41	formnal_2.dat	Описание страниц индикации в режиме «Ручной» (2 координаты)	maya-te.dat maya-2e.dat
42	formnal_3.dat	Описание страниц индикации в режиме «Ручной» (3 координаты)	maya-3e.dat
43	formnal_4.dat	Описание страниц индикации в режиме «Ручной» (4 координаты)	maya-4e.dat
44	formnal_5.dat	Описание страниц индикации в режиме «Ручной» (5 координаты)	maya-5e.dat
45	formprn.dat	Описание страниц индикации в режиме «Преднабор»	maya-te.dat maya-Xe.dat
46	formvi0_2.dat	Описание страниц индикации в режиме «Выход в ноль» (2 координаты)	maya-te.dat maya-2e.dat
47	formvi0_3.dat	Описание страниц индикации в режиме «Выход в ноль» (3 координаты)	maya-3e.dat
48	formvi0_4.dat	Описание страниц индикации в режиме «Выход в ноль» (4 координаты)	maya-4e.dat
49	formvi0_5.dat	Описание страниц индикации в режиме «Выход в ноль» (5 координаты)	maya-5e.dat
50	formvit_2.dat	Описание страниц индикации в режиме «Выход в точку» (2 координаты)	maya-te.dat maya-2e.dat
51	formvit_3.dat	Описание страниц индикации в режиме «Выход в точку» (3 координаты)	maya-3e.dat
52	formvit_4.dat	Описание страниц индикации в режиме «Выход в точку» (4 координаты)	maya-4e.dat
53	formvit_5.dat	Описание страниц индикации в режиме «Выход в точку» (5 координаты)	maya-5e.dat
54	hash_d.cfg	Описание D-ячеек и таблица соответствия D-ячеек и параметров	
55	hotkeys.dat	Описание «горячих» клавиш	maya-te.dat maya-Xe.dat

56	keys.dat	Коды клавиатуры	maya-te.dat maya-Xe.dat
57	krd-2.dat	Расчет знакоместа координат фрезерный вариант на 2 координаты	maya-2e.dat
58	krd-3.dat	Расчет знакоместа координат фрезерный вариант на 3 координаты	maya-3e.dat
59	krd-4.dat	Расчет знакоместа координат фрезерный вариант на 4 координаты	maya-4e.dat
60	krd-5.dat	Расчет знакоместа координат фрезерный вариант на 5 координаты	maya-5e.dat
61	krd-t.dat	Расчет знакоместа координат токарный вариант	maya-te.dat
62	m_functions.dat	Список М-функций для станка	formprn.dat
63	m28form.dat	Вывод m28 и m29 в формах (form)	formnal_X.dat* formavt.dat formvi0_X.dat* formvit_X.dat* formfix_X.dat* formprn.dat
64	m28grid.dat	Вывод m28 и m29 в сетках под модулями и таблицами	dialtok.dat dialfrz.dat modulesm.dat tables.dat
65	machine.cfg		dialtok.dat dialfrz.dat modulesm.dat tables.dat
66	maya-2e.dat	Ссылки на файлы описания экранных интерфейсов (2 координаты)	machine.cfg
67	maya-3e.dat	Ссылки на файлы описания экранных интерфейсов (3 координаты)	machine.cfg
68	maya-4e.dat	Ссылки на файлы описания экранных интерфейсов (4 координаты)	machine.cfg
69	maya-5e.dat	Ссылки на файлы описания экранных интерфейсов (5 координаты)	machine.cfg
70	maya-te.dat	Ссылки на файлы описания экранных интерфейсов (токарный вариант)	machine.cfg
71	menu-2e.dat	Меню режимов фрезерный вариант на 2 координаты	maya-2e.dat
72	menu-3e.dat	Меню режимов фрезерный вариант на 3 координаты	maya-3e.dat
73	menu-4e.dat	Меню режимов фрезерный вариант на 4 координаты	maya-4e.dat
74	menu-5e.dat	Меню режимов фрезерный вариант на 5 координат	maya-5e.dat
75	menu-me.dat		
76	menu-te.dat	Меню режимов токарный вариант	maya-te.dat
77	modulesm.dat	Описание модулей	maya-te.dat maya-Xe.dat

78	pic123.bm	Рисунки для «Помощь» (QR-коды)	pic123.cfg
79	pic123.cfg	Рисунки для «Помощь» (QR-коды)	tables.dat
80	picf.bm	Рисунки для диалога (фрезерный вариант)	picf.cfg
81	picf.cfg	Рисунки для диалога (фрезерный вариант)	dialfrz.dat
82	pict.bm	Рисунки для диалога (токарный вариант)	pict.cfg
83	pict.cfg	Рисунки для диалога (токарный вариант)	dialtok.dat
84	shpind.dat	Индикация оси шпинделя для основных страниц режимов	formnal_X.dat* formavt.dat formvi0_X.dat* formvit_X.dat* formfix_X.dat* formprn.dat
85	tab.cfg	Комментарии к параметрам, входам/выходам, списку ПП ПЭС	tblvals.dat
86	tables.dat	Описание модулей «таблиц» (параметры, диагностика и т.д.)	maya-te.dat maya-Xe.dat
87	tblvals.dat	Комментарии к параметрам УЧПУ	
89	txtvals.dat	Текстовые сообщения об ошибках БПрО	

*Вместо X может стоять 2, 3, 4 или 5.

Разработчиком ПЭС дополняются следующие файлы (они выделены цветом):

1) errors.cfg – тексты ошибок и сообщений;
2) hash_d.cfg – назначение D-ячеек и комментарии к ним;
3) m_functions.dat - список M-функций для станка;
4) machine.cfg – настройка конфигурации системы;
5) tab.cfg – тексты комментариев к таблицам ввода/вывода и параметрам;
6) tblvals.dat - комментарии к параметрам УЧПУ;
7) txtvals.dat - текстовые сообщения об ошибках БПрО;
8) Файлы системной и пользовательской ПЭС.

Файлы errors.cfg, tab.cfg, hash_d.cfg и файлы с ПЭС могут быть подготовлены на ПК в любом текстовом редакторе (кодировка DOS). Имена файлов должны содержать только строчные (маленькие) буквы (TAB.CFG и tab.cfg – разные имена файлов!).

9 Корневой каталог

Корневой каталог – это начальный каталог (папка) в иерархической системе каталогов, который обычно обозначается символом. Он содержит все остальные каталоги и файлы в системе и является точкой отсчета для всех путей к файлам и каталогам. Вызывается корневой каталог на клавишу ФАЙЛ (при наличии) или сочетанием клавиш Alt+F.

В корневом каталоге находятся следующее:

- папки:
  - bind;
  - ISO папка с файлами УП;
  - Menu625 папка с интерфейсными файлами;
  - ScreenShots папка для снимков экрана;
  - sys.
- файлы:
  - 16m30_p_new.cfg
  - access.dat
  - auto.cfg
  - def.dmp
  - errors.cfg
  - gfun.txt
  - hash_d.cfg
  - machine.cfg
  - mntdirs.cfg
  - mst.tab
  - podpr220.ckl
  - qsel.cfg
  - read.me!
  - sys_16m30_p.cfg
  - sys-t.tab
  - tab.cfg
  - tab_1620m4.cfg
  - tblvals.dat
  - txtvals.dat
  - udp.cfg
  - upp.iso
  - userhelp.t