**Задание Ш02.** В пакете прикладных программ *National Instruments LabView* создать виртуальный прибор, предназначенный для перевода значений из одной, заданной системы счисления в другую, указанную систему счисления.

На блок-диаграмме все функциональные, то есть значащие пересечения линий связи (узлы) обозначить точками. Для этого в меню перейти к настройкам «*Tools > Options…*», далее в выпадающем списке перейти к настройкам блок-диаграммы (*Block Diagram*) и выставить галочку напротив пункта «*Show dots at wire junctions*».

По итогам выполнения работы сдаются строго два файла:

- отчёт, выполненный в текстовом редакторе *Microsoft Office Word* (*\*.doc* или *\*.docx*);

- файл виртуального прибора *National Instruments LabView* (*\*.vi*) по индивидуальной части работы.

Отправленные поодиночке файлы проверке не подлежат. При отсутствии одного из упомянутых файлов зачёт по заданию не выставляется.

**Требования к именам файлов:**

**Общий вид формата имени файла:** «*Дата. Задание. Фамилия.mcdx*»

**Формат записи даты:** «*ГГГГММДД*», где *ГГГГ* – четыре цифры текущего года, *ММ* – две цифры текущего месяца, *ДД* – две цифры текущего дня.

**Формат записи задания:** «Задание *NNk*», где *NN* – две цифры номера задания, *k* – обозначение «о», если файл содержит общую часть; обозначение «и», если файл содержит индивидуальную часть; обозначение «ои», если файл содержит как общую, так и индивидуальную части.

**Если устранить замечания по работе удаётся в тот же день:** после фамилии ставится пробел и в круглых скобках записывается номер попытки исправления.

**Примеры правильных имён файлов, которые сдаются на проверку впервые:**

«*20190301. Задание 02ш. Иванов.docx*»

«*20190301. Задание 02ш. Иванов.vi*»

**Внимание!** Не забудьте выполнить автоматическую нумерацию страниц в отчёте.

Отчёт по выполненной работе должен содержать:

0. Титульный лист.

1. Формулировку цели работы.

2. Описание задачи согласно выданному варианту.

3. Составление блок-схемы алгоритма программы.

4. Подбор и расчёт тестовых примеров.

5. Листинг кода составленного программного обеспечения (блок-диаграммы *LabView*).

6. Графический пользовательский интерфейс программного обеспечения (передняя панель виртуального прибора *LabView*) и его описание.

7. Расчёт тестовых примеров с использованием составленного программного обеспечения.

8. Формулировку вывода о проделанной работе (обезличено – исключить из вывода местоимения, такие как «я», «мы» и другие).

Рекомендации к отчёту, доказывающие самостоятельность выполнения работы и упрощающие процедуру проверки отчёта преподавателем:

1. Выполнение дополнительных скриншотов для случаев, когда текстовое описание проделанных действий становится громоздким или трудным к восприятию.

2. Нумерация рисунков (если есть) с подписями, содержащими названия рисунков, например, «Рисунок 1 – Пользовательский интерфейс *Microsoft Office Excel*».

**Цель работы (одна из возможных формулировок)**: освоение навыков работы по составлению строк, блоков-функций целочисленного деления, сдвиговых регистров и расширенной структуры *CASE* с более, чем двумя значениями.

**Рекомендации к работе:**

***Алгоритм перевода из десятичной системы счисления в двоичную систему.***

1. Выполнить деление исходного числа на 2. Если результат деления больше или равен 2, продолжать делить его на 2 до тех пор, пока результат деления не станет равен 1.

2. Выписать результат последнего деления и все остатки от деления в обратном порядке в одну строку.

***Алгоритм перевода из двоичной системы счисления в десятичную систему.***

1. Пронумеровать разряды двоичного числа справа налево, начиная с нуля.

2. Умножить каждый ненулевой разряд на 2 в степени его номера и сложить результаты.

***Алгоритм перевода из шестнадцатеричной системы счисления в двоичную систему.***

1. Задать таблицу соответствия четырёхразрядного формата в двоичной системе по отношению к каждому разряду в шестнадцатеричной системе. Например, F это 1111, A – 1010.

2. Поразрядно отщеплять от шестнадцатеричного значения элементы справа налево и записывать четырёх разрядный компонент так же справа налево. Например: 9AF это 1001 1010 1111.

***Алгоритм перевода из двоичной системы счисления в шестнадцатеричную систему.***

1. Задать таблицу соответствия четырёхразрядного формата в двоичной системе по отношению к каждому разряду в шестнадцатеричной системе. Например, 0001 это 1, 1011 – B.

2. Проверить, кратна ли заданная двоичная последовательность 4-м. Если нет, то дополнить её слева недостающим количеством нулей.

3. Идти по последовательности справа налево и отделять четвёрку разрядов. Каждую четвёрку сравнивать с таблицей соответствия и выдавать шестнадцатеричный разряд.

4. Записывать каждый новый разряд так же справа налево.

**Примеры реализации виртуальных приборов**

**Данные входные / выходные, представленные строковым массивом:**

Из раздела «*Array*» разместим на передней панели подложку для массива.

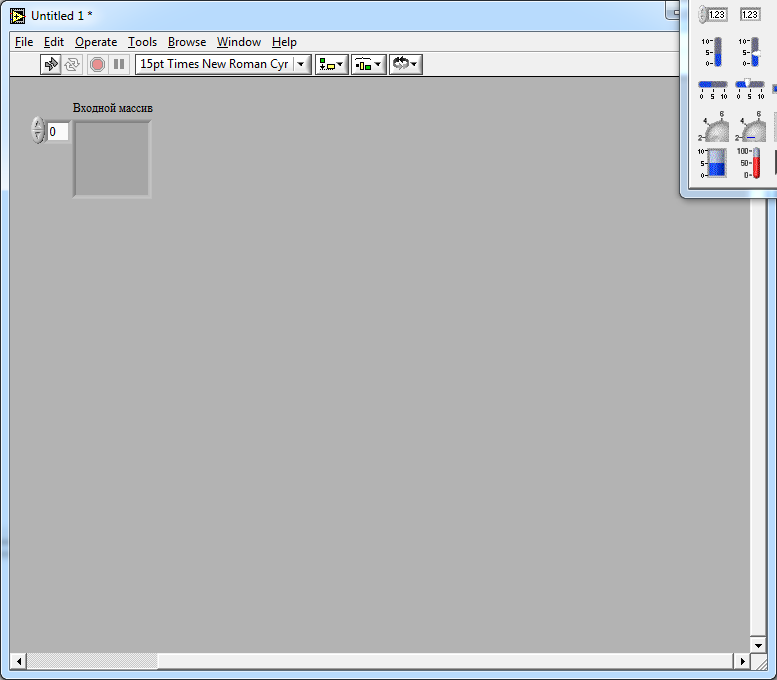


Рисунок 1 – Представление нетипизированного массива на передней панели ВП

На подложке разместим строковый контроллер для определения типа массива строковым.

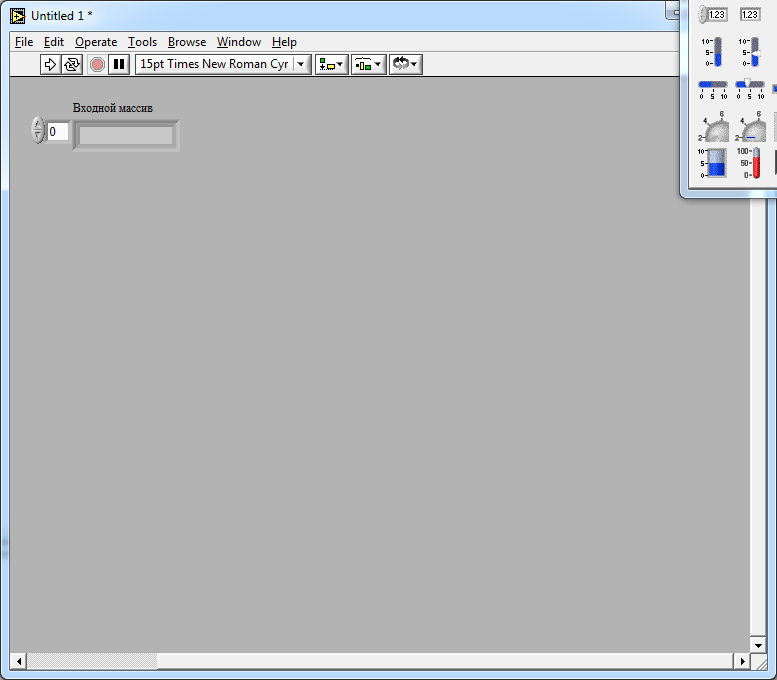


Рисунок 2 – Строковый массив на передней панели ВП

Строковый массив для удобства ввода и контроля данных растягивается на нужное количество ячеек вниз. Пусть задано шестнадцатеричное число «*1А4*» (можно записать его и в обратном порядке, снизу вверх, но для этого важно дать текстовый комментарий для пользователя на графическом пользовательском интерфейсе).

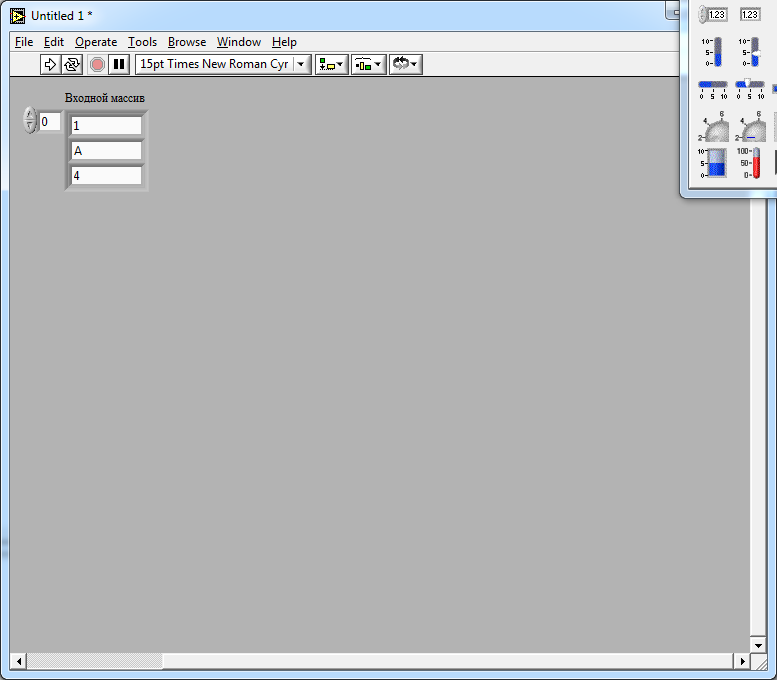


Рисунок 3 – На графический пользовательский интерфейс введён рассматриваемый пример

Проведём декомпозицию поставленной задачи. Решим сначала подзадачу индикации каждого разряда в десятичной системе счисления. Добавим числовой контроллер для указания интересующей ячейки с шестнадцатеричной цифрой.

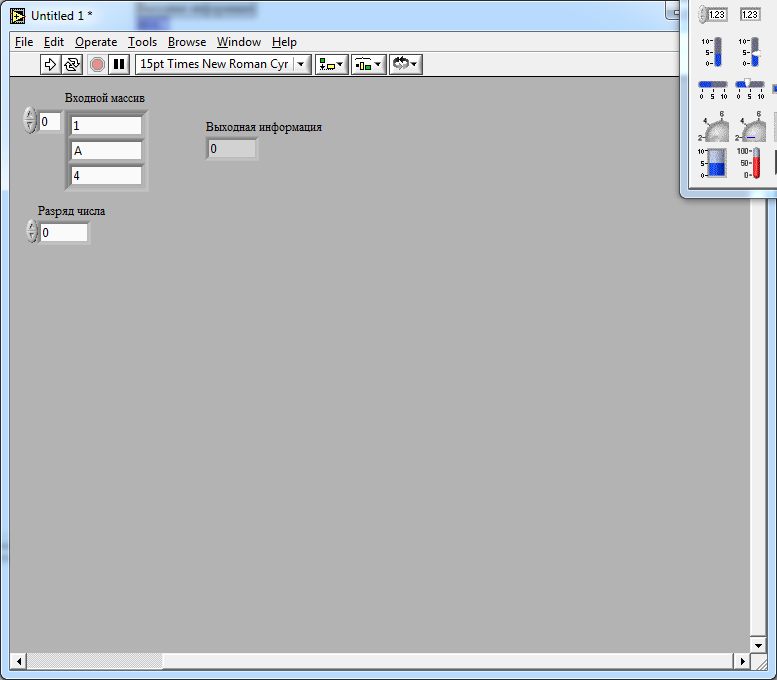


Рисунок 4 – Дополнение графического пользовательского интерфейса элементами управления

**Важно!** С точки зрения разрядности некоторого числа использованные индексы массива задействованы в обратном порядке, потому необходимо реализовать операцию инверсии индексов массива – это ещё одна подзадача.

Для её решения, первым шагом, важно получить информацию о размерности массива и вычесть из неё указанный интересующий разряд. Так в рассматриваемом примере нулевому разряду должна соответствовать третья ячейка массива с индексом «2», первому разряду – вторая ячейка массива с индексом «1», второму разряду – первая ячейка массива с индексом «0». Реализуем это.

Выставляем на блок-диаграмму компонент «*Array Size*», предназначенный для определения размерности массива, далее исключаем из величины размерности массива интересующий нас разряд и сокращаем результат на единицу (поскольку индексация считается с нуля, а количество элементов массива с единицы).

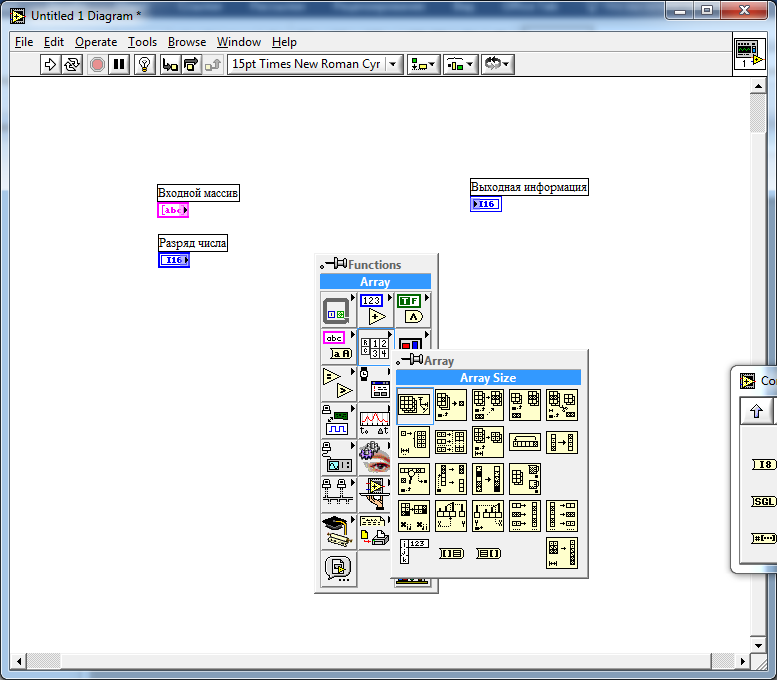


Рисунок 5 – Место расположения функции, отвечающей за определение размерности массива

В данном случае реализован механизм для обращения к конкретному индексу массива по интересующей разрядности. Протестируем второй разряд. Должны получить на выходе нулевой индекс.

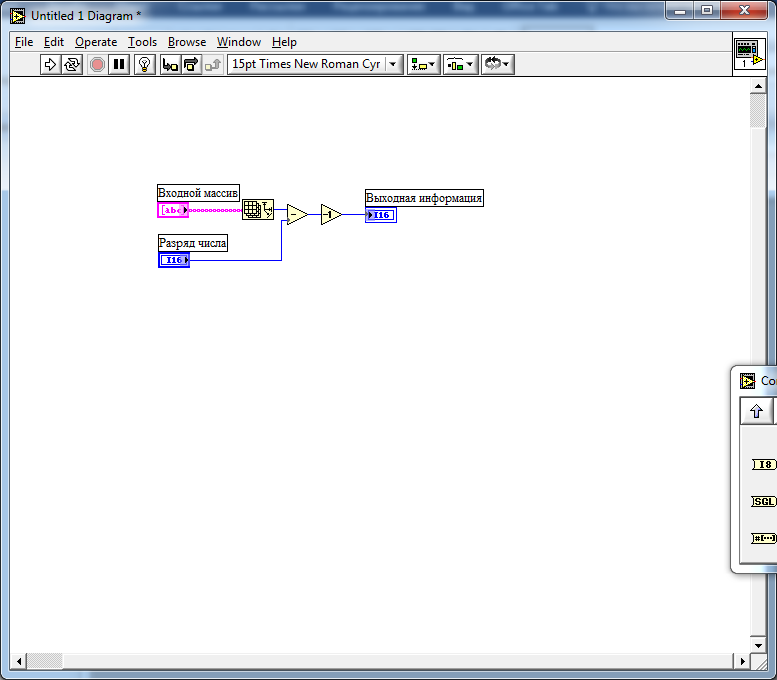


Рисунок 6 – Блок-диаграмма решения подзадачи поиска соответствия разряда числа корректному индексу массива

Результат запуска виртуального прибора соответствует ожиданиям.

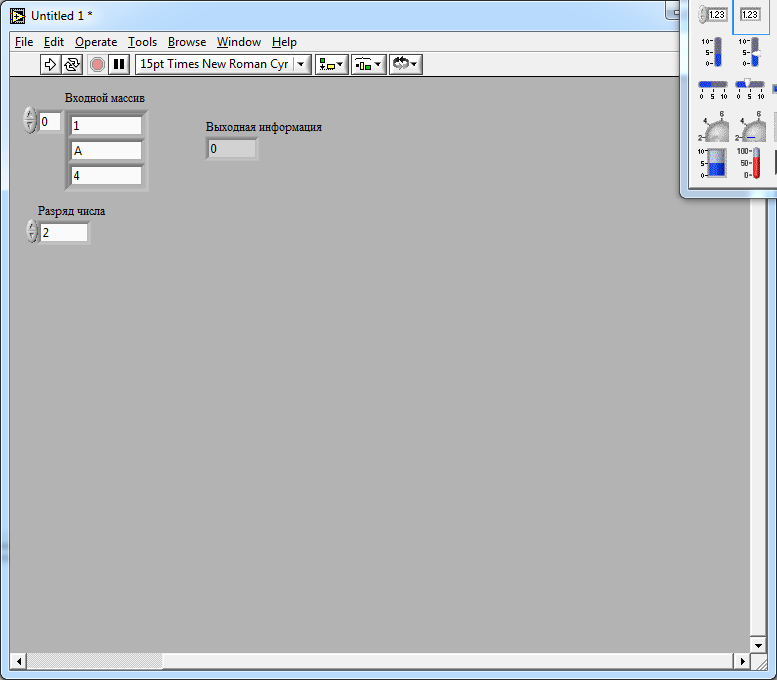


Рисунок 7 – Результат решения подзадачи поиска соответствия разряда числа корректному индексу массива

Продолжаем составление программного кода – создадим ядро конвертации. Пишется оно не сложно. Для всех значений из диапазона «0-9» необходимо выполнить перевод из строки в целое число, а для значений диапазона «А-F» самостоятельно указать нужную целочисленную константу согласно Таблице 1.

Таблица 1 – Таблица перевода

|  |  |
| --- | --- |
| **16-я** | **10-я** |
| A | 10 |
| B | 11 |
| C | 12 |
| D | 13 |
| E | 14 |
| F | 15 |

Возьмём из массива указанный индекс. За эту операцию ответственна стандартная функция пакета прикладных программ *National Instruments LabView* под названием «*Index Array*».

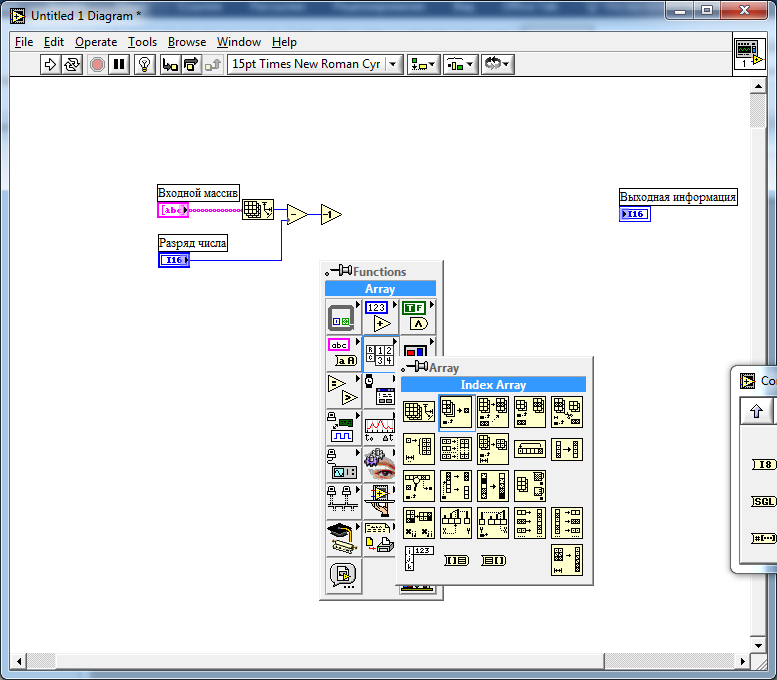


Рисунок 8 – Место расположения функции для получения элемента массива по указанному индексу

Таким образом, выделить элемент с интересующим индексом можно по следующей схеме, размещённой на блок диаграмме.

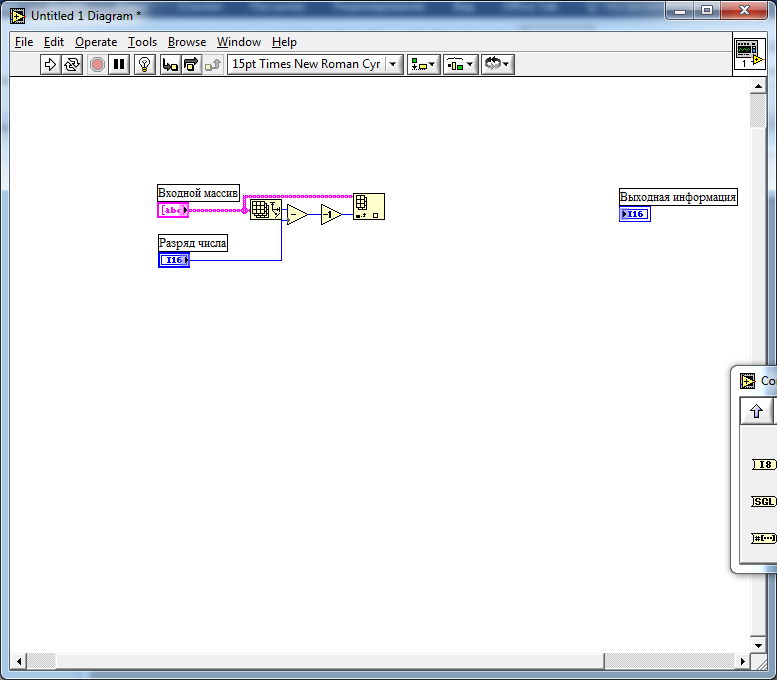


Рисунок 9 – Фрагмент схемы для обращения к указанному разряду шестнадцатеричного числа, записанного в массив

Далее *CASE*-структура должна помочь в определении строкового шаблона, который содержится в интересующей ячейке массива.

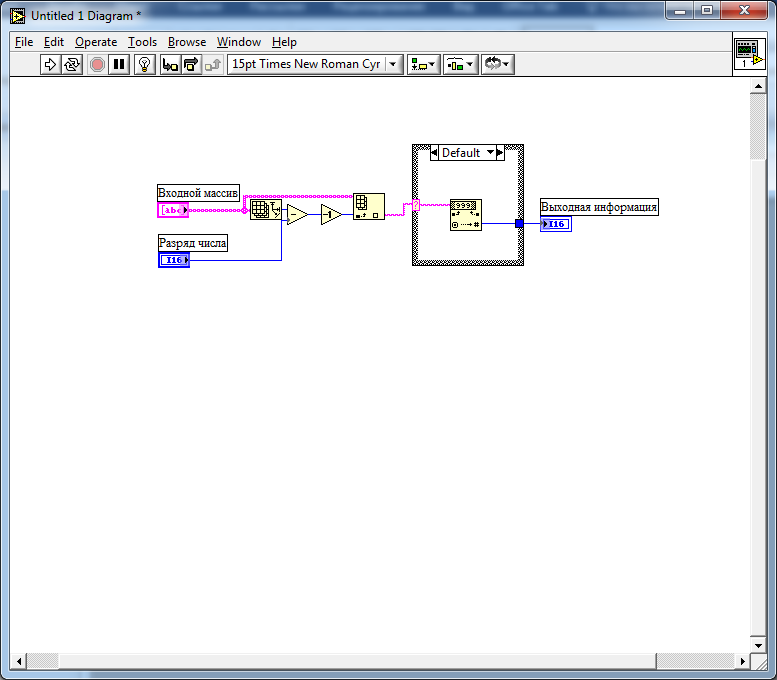


Рисунок 10 – Размещение на блок-диаграмме CASE-структуры, работающей на детектирование строковых шаблонов

Внутри блока действий по умолчанию (*Default*) необходима разместить функцию «*Decimal String To Number*» (для преобразование десятичной строки в десятичное число).

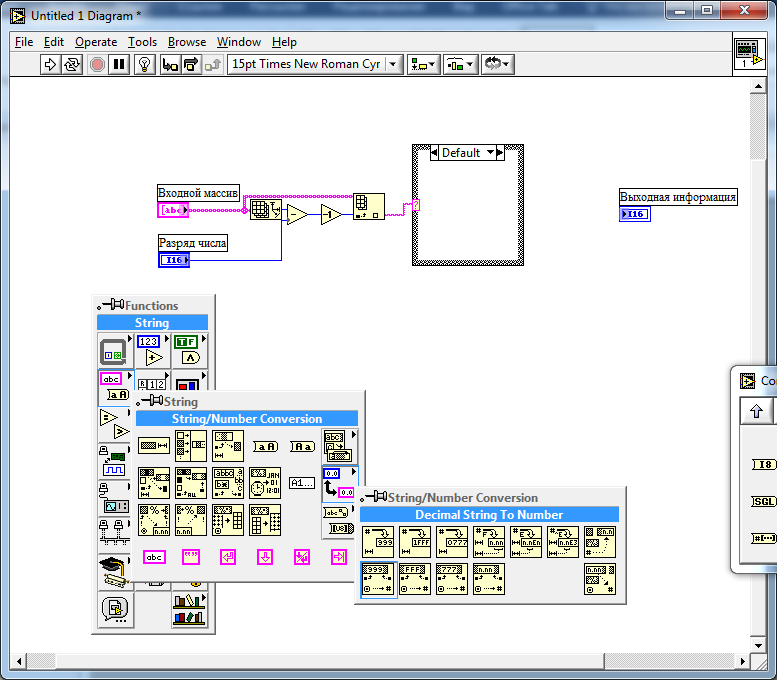


Рисунок 11 – Место расположения функции для преобразования десятичной строки в десятичное число

Таким образом блок действий по умолчанию становится ответственным за работу с диапазоном «0-9».

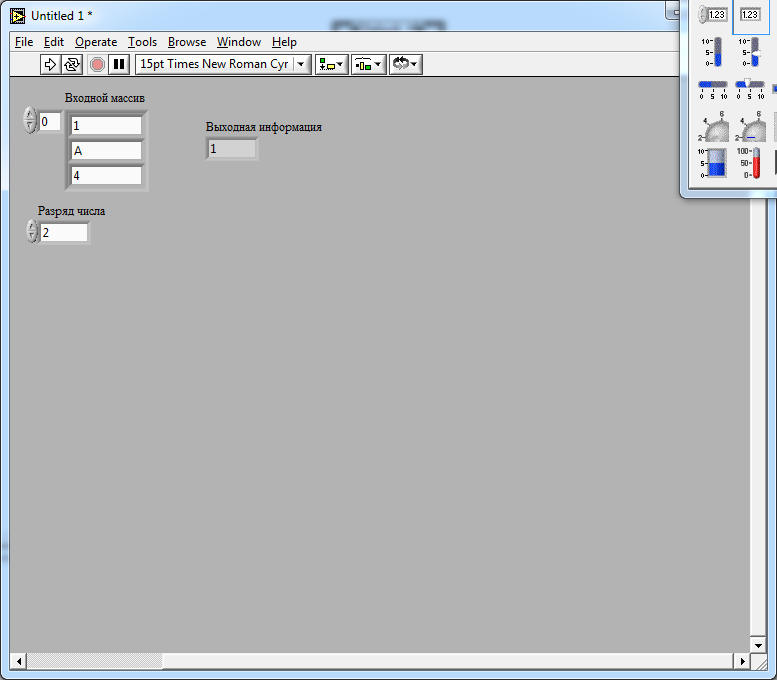


Рисунок 12 – Результат вывода численного значения второго разряда числа «*1А4*»

Далее в *CASE*-структуру последовательно добавляются строковые шаблоны «*A*», «*B*», «*C*», «*D*», «*E*», «*F*» посредством выпора в контекстном меню структуры опции добавления случая вслед за выбранным «*Add Case After*». Внутри случаев, соответствующих строковым шаблонам, размещаются соответствующие численные константы, направляемые на выход.

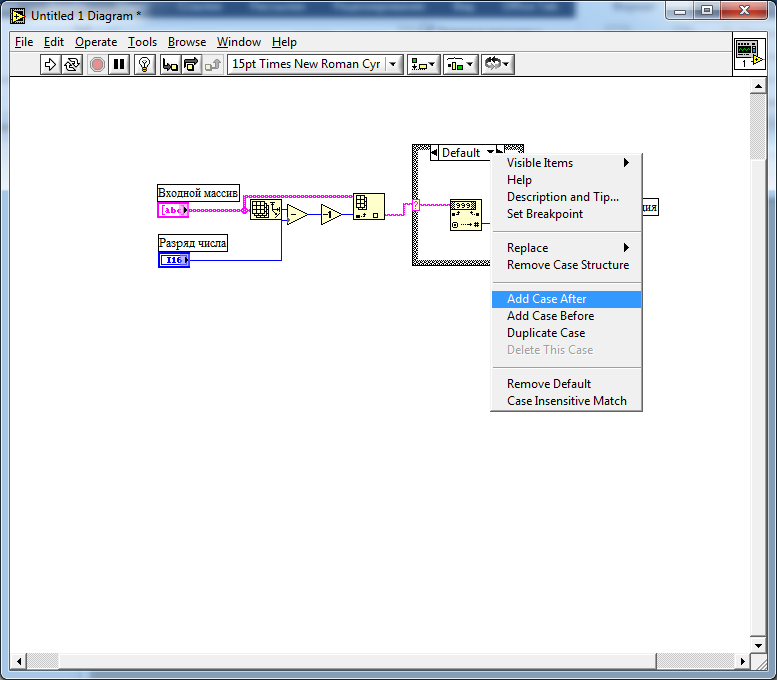


Рисунок 13 – Добавление нового случая в структуру

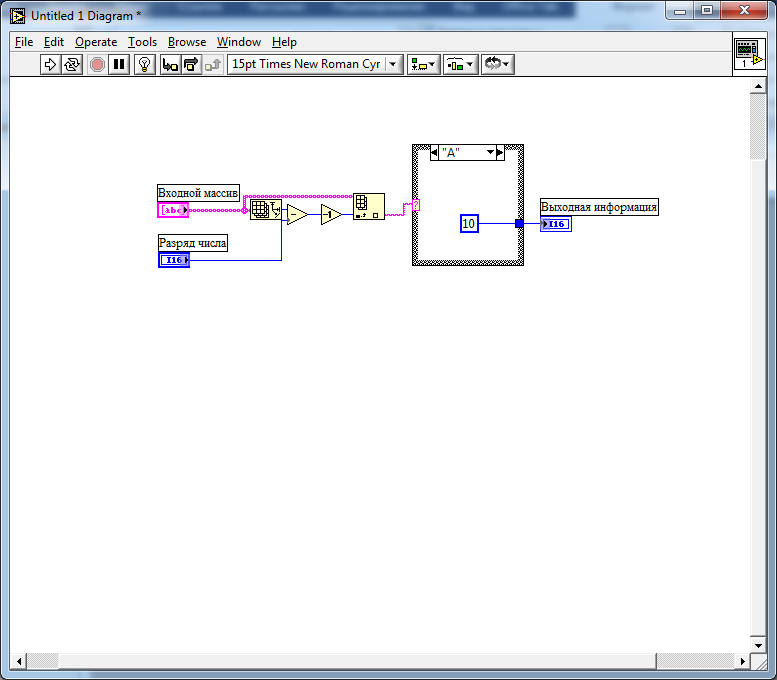


Рисунок 14 – Пример обработки случая с шаблоном «*A*»

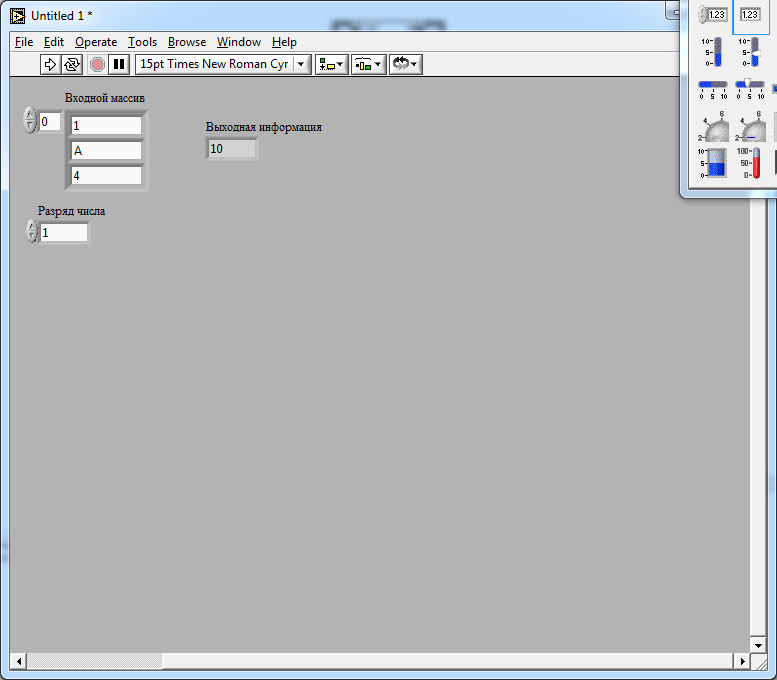


Рисунок 15 – Результат представления в десятичной системе счисления значения первого разряда шестнадцатеричного числа

**Пример работы с меню:**

Для работы с меню необходимо разместить на графическом пользовательском интерфейсе, передней панели ВП соответствующий элемент управления.

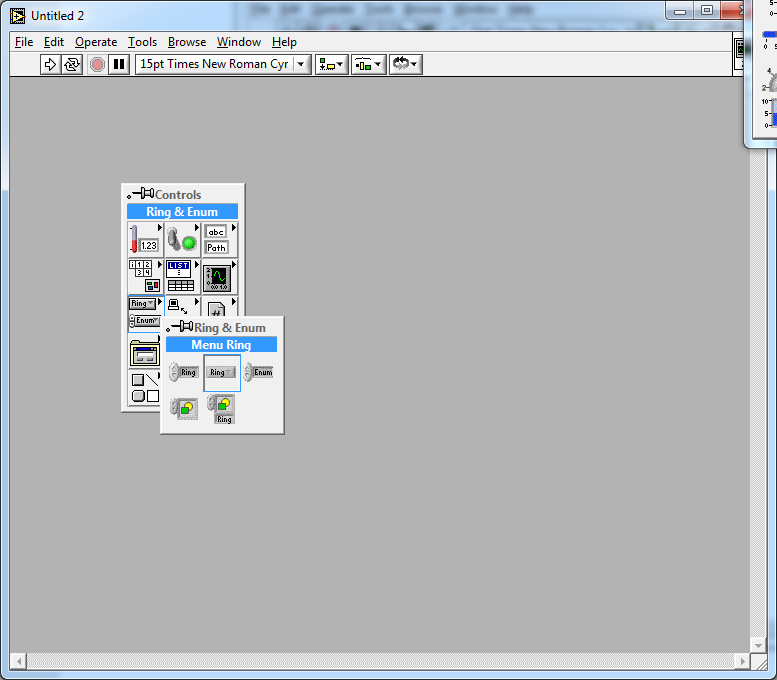


Рисунок 16 – Место расположения интерфейсного элемента управления «Меню»

После размещения меню оно должно быть наполнено управляющими элементами – пунктами меню, за которыми в указанном порядке закрепляются целочисленные индексы.

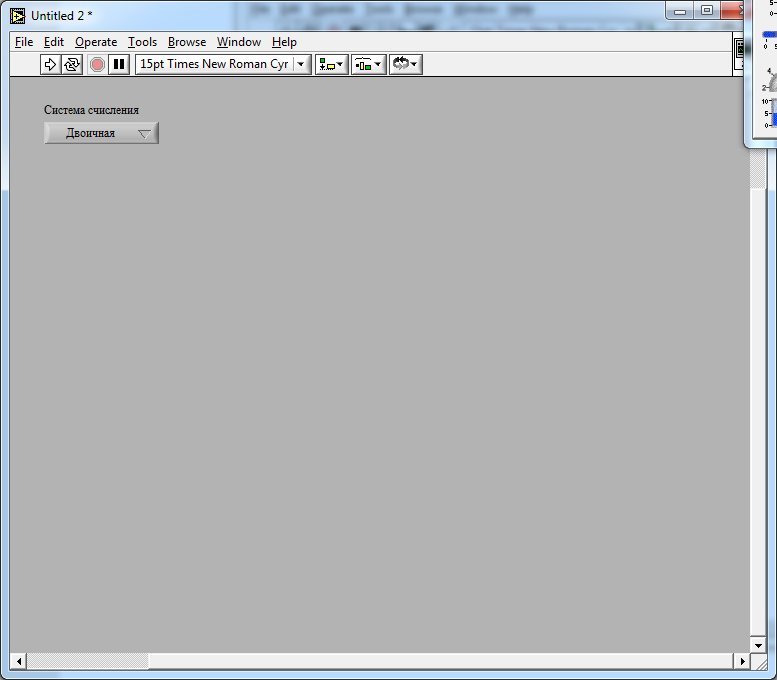


Рисунок 17 – Размещение меню на передней панели ВП

Пример добавления новых пунктов меню представлен ниже.

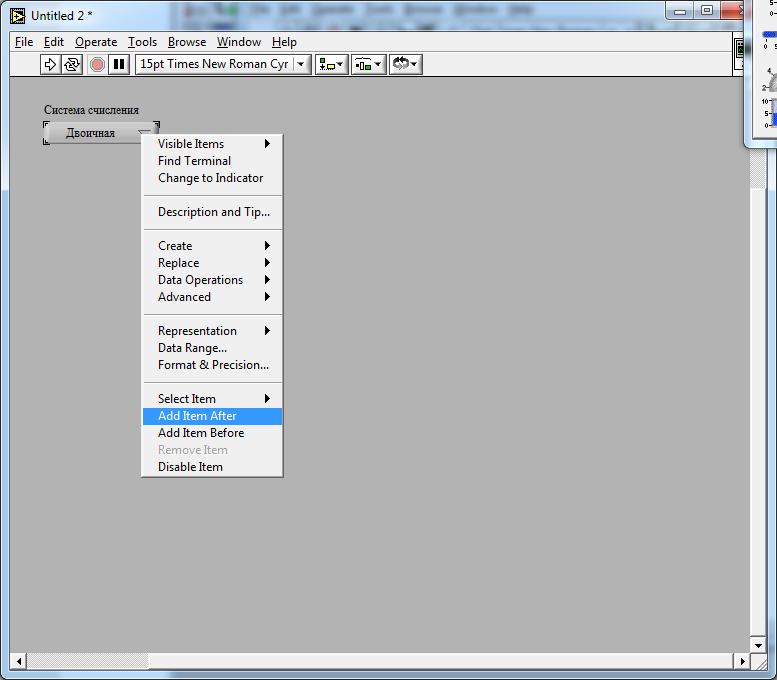


Рисунок 18 – Добавление нового элемента в меню ВП

Результатом заполнения меню является функциональный комбинированный список, в котором можно установить позицию, заданную по умолчанию для последующих запусков виртуального прибора.

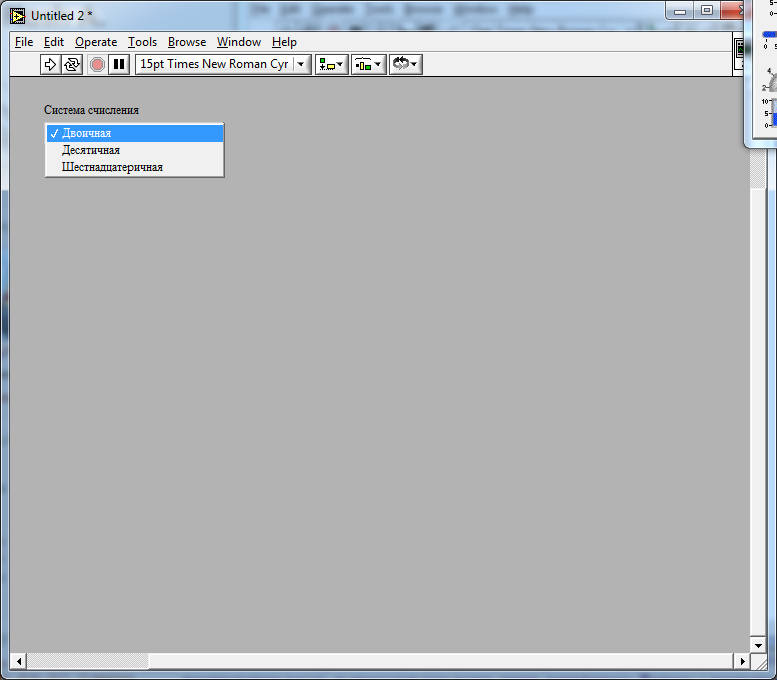


Рисунок 19 – Просмотр созданных элементов меню

На блок-диаграмме работа с меню осуществляется через *CASE*-структуру.

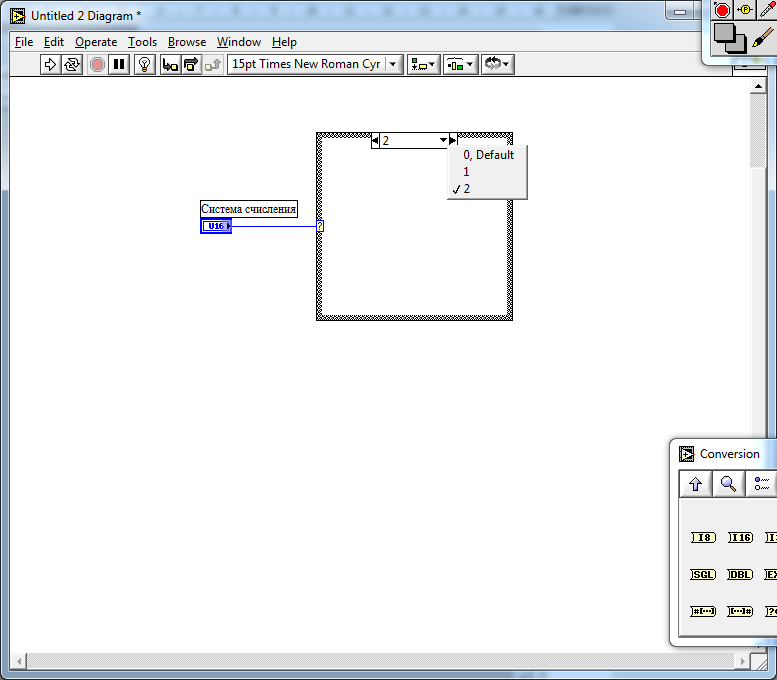


Рисунок 20 – Подключение меню к *CASE*-структуре

**Варианты индивидуального задания**

Таблица 2 – Варианты индивидуального задания по системам счисления

|  |  |
| --- | --- |
| № | Задание |
| 1 | Разработать универсальный конвертор из двоичной системы счисления в 3–10-чные системы счисления (с управляющим числовым элементом). Входной тип данных – массив, выходные типы данных – целые. |
| 2 | Разработать универсальный конвертор из шестнадцатеричной системы счисления в 2–10-чные системы счисления (с выбором основания системы счисления через меню). Входной тип данных – массив, выходные типы данных – строки. |
| 3 | Разработать конвертор из системы счисления, в которой задействован весь кириллический алфавит (А-Я) в десятичную систему счисления. Входной тип данных – строка, выходной тип данных – целое. |
| 4 | Разработать универсальный конвертор из двоичной системы счисления в 11–20-чные системы счисления (с выбором основания системы счисления через меню). Входной тип данных – строка, выходные типы данных – строки. |
| 5 | Разработать конвертор из системы счисления, в которой задействован весь латинский алфавит (A-Z), в двоичную систему счисления. Входной тип данных – массив, выходной тип данных – массив. |
| 6 | Разработать конвертор из двоичной системы счисления в десятичную и шестнадцатеричную системы счисления. Входной тип данных – массив, выходные типы данных – массивы. |
| 7 | Разработать универсальный конвертор из восьмеричной системы счисления в 2–7-чные системы счисления (с выбором основания системы счисления через меню). Входной тип данных – строка, выходные типы данных – массивы. |
| 8 | Разработать универсальный конвертор из двоичной системы счисления в 11–20-чные системы счисления (с выбором основания через управляющий числовой элемент). Входной тип данных – массив, выходные типы данных – массивы. |
| 9 | Разработать универсальный конвертор из двоичной системы счисления в 3–10-чные системы счисления (с выбором основания системы счисления через меню). Входной тип данных – массив, выходные типы данных – целые. |
| 10 | Разработать конвертор из системы счисления, в которой задействован весь кириллический алфавит (А-Я) в двоичную систему счисления. Входной тип данных – массив, выходной тип данных – массив. |
| 11 | Разработать конвертор из шестнадцатеричной системы счисления в двоичную и десятичную системы счисления. Входной тип данных – массив, выходные типы данных – строки. |
| 12 | Разработать конвертор из системы счисления, в которой задействован весь латинский алфавит (A-Z), в десятичную систему счисления. Входной тип данных – строка, выходной тип данных – целое. |
| 13 | Разработать универсальный конвертор из двоичной системы счисления в 3–10-чные системы счисления (с управляющим числовым элементом). Входной тип данных – строка, выходные типы данных – строки. |
| 14 | Разработать конвертор из восьмеричной системы счисления в десятичную и шестнадцатеричную системы счисления. Входной тип данных – массив, выходные типы данных: для шестнадцатеричной – строки, для десятичной – целое. |
| 15 | Разработать конвертор из двоичной системы счисления в десятичную и шестнадцатеричную системы счисления. Входной тип данных – массив, выходные типы данных – строки. |

|  |  |
| --- | --- |
| 16 | Разработать конвертор из десятичной в системы счисления, в такую систему счисления, в которой задействован весь кириллический алфавит (А-Я). Входной тип данных – целое, выходной тип данных – строка. |
| 17 | Разработать конвертор из шестнадцатеричной системы счисления в двоичную и десятичную системы счисления. Входной тип данных – строка, выходные типы данных: для двоичной – строка, для десятичной – целое. |
| 18 | Разработать конвертор из двоичной системы счисления в десятичную и шестнадцатеричную системы счисления. Входной тип данных – строка, выходные типы данных – массивы. |
| 19 | Разработать универсальный конвертор из шестнадцатеричной системы счисления в 2–10-чные системы счисления (с выбором основания системы счисления через меню). Входной тип данных – строка, выходные типы данных – массивы. |
| 20 | Разработать универсальный конвертор из восьмеричной системы счисления в 2–7-чные системы счисления (с выбором основания системы счисления через меню). Входной тип данных – массив, выходные типы данных – строки. |
| 21 | Разработать универсальный конвертор из двоичной системы счисления в 11–20-чные системы счисления (с выбором основания системы счисления через меню). Входной тип данных – массив, выходные типы данных – массивы. |
| 22 | Разработать универсальный конвертор из восьмеричной системы счисления в 2–7-чные системы счисления (с управляющим числовым элементом). Входной тип данных – целое, выходные типы данных – целые. |
| 23 | Разработать универсальный конвертор из шестнадцатеричной системы счисления в 2–10-чные системы счисления (с управляющим числовым элементом). Входной тип данных – массив, выходные типы данных – целые. |
| 24 | Разработать конвертор из восьмеричной системы счисления в десятичную и шестнадцатеричную системы счисления. Входной тип данных – строка, выходные типы данных – строки. |
| 25 | Разработать конвертор из десятичной в системы счисления, в такую систему счисления, в которой задействован весь латинский алфавит (A-Z). Входной тип данных – целое, выходной тип данных – массив. |
| 26 | Разработать универсальный конвертор из двоичной системы счисления в 11–20-чные системы счисления (с выбором основания через управляющий числовой элемент). Входной тип данных – строка, выходные типы данных – строки. |
| 27 | Разработать универсальный конвертор из двоичной системы счисления в 3–10-чные системы счисления (с выбором основания системы счисления через меню). Входной тип данных – строка, выходные типы данных – строки. |
| 28 | Разработать конвертор из восьмеричной системы счисления в двоичную и шестнадцатеричную системы счисления. Входной тип данных – целое, выходные типы данных – строки. |
| 29 | Разработать конвертор из шестнадцатеричной системы счисления в двоичную и десятичную системы счисления. Входной тип данных – массив, выходные типы данных: для десятичной – целое, для двоичной – массив. |
| 30 | Разработать конвертор из двоичной системы счисления в десятичную и шестнадцатеричную системы счисления. Входной тип данных – строка, выходные типы данных – строки. |

**Шифры вариантов:**

Таблица 3 – Шифры вариантов индивидуального задания

|  |  |  |  |
| --- | --- | --- | --- |
| 1) | 13) | 16) | 25) |
| 2) | 14) | 20) |  |
| 3) |  |  |  |
| 4) | 18) | 22) | 30) |
| 5) | 10) | 23) | 27) |
| 6) |  |  |  |
| 7) | 17) |  |  |
| 8) | 12) | 29) |  |
| 9) | 11) | 24) |  |
| 15) |  |  |  |
| 19) |  |  |  |
| 21) |  |  |  |
| 26) |  |  |  |
| 28) |  |  |  |