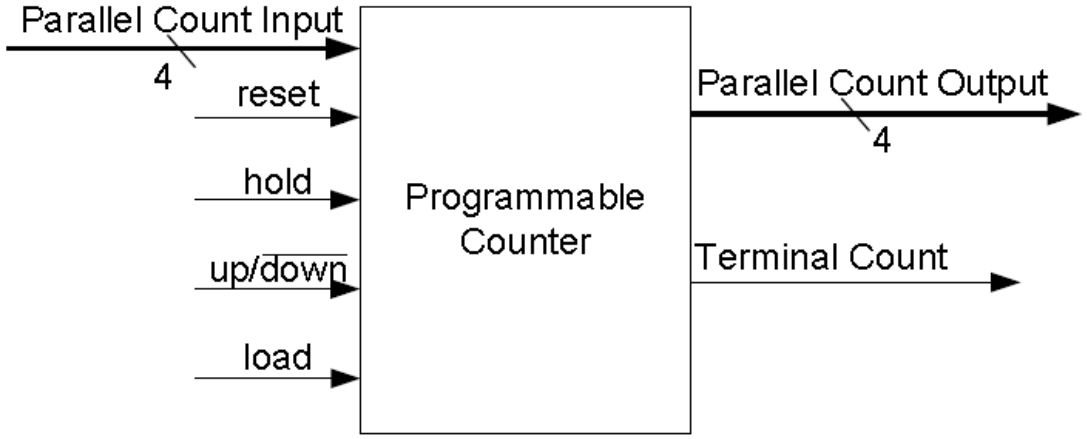
Разработка программируемых счетчиков в LabVIEW для оценочного модуля DE FPGA Board

Цель работы:

Целью лабораторной работы является разработка программируемых счетчиков с использованием структур case и while loop. Результаты проектирования вы проконтролируете на оценочном модуле DE FPGA и системе NI ELVIS.

Задание на проектирование:

Разработать программируемый 4-разрядный счетчик с функциями прямого и обратного счета, сброса, формирования сигнала окончания счета, загрузки параллельного кода через 4-разрядный вход и хранения состояния; требуемая функция выбирается сигналами управления. Входам управления назначен разный приоритет – сброс обладает высшим приоритетом, более низким приоритетом обладает хранение, затем загрузка и, наконец, направление счета (up/down). Счетчик работает на сложение, если на вход up/down подана "Лог.1", и на вычитание – в противном случае. Выход окончания счета (Terminal Count) принимает значение "Лог.1", когда при счете на сложение на параллельных выходах появляется число 15. Этот же выход принимает значение "Лог.0", когда при счете на вычитание на параллельных выходах появляется число 0. Схема счетчика показана ниже.



Выполнение задания:

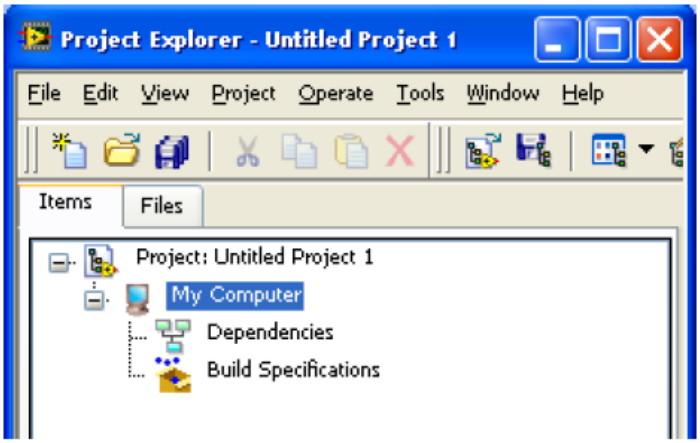
Функциональность схемы требует реализация приоритета управляющих входов. Приоритетность входов может быть реализована с использованием каскадного включения функций *select*. Сигнал окончания счета вычисляется по-разному, в зависимости от направления счета – сложения или вычитания. Это может быть реализовано с помощью структуры case. Функции инкремента и декремента выполняются над целыми беззнаковыми числами, в то время, как параллельные входы булевского типа, поэтому сигналы для этих входов нужно конвертировать в беззнаковые числа, используя функции build array и Boolean number conversion. Выходные сигналы, наоборот, должны быть преобразованы в данные булевского типа, чтобы их можно было подключить к линиям светодиодов (LED) и ввода-вывода (GPIO). Вся схема должна быть заключена в цикл while loop, управляемый сигналом останова.

Для быстрого освоения LabVIEW выполните курс “Learn LabVIEW in 3 hours” (http://www.ni.com/academic/learn_LabVIEW/)

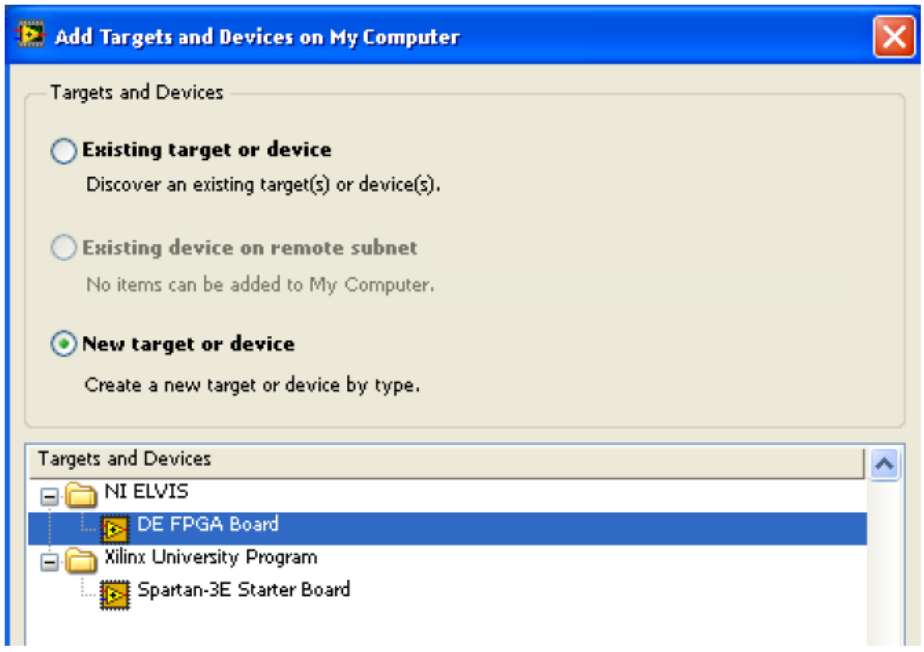
Раздел 1: разработка устройства

Порядок выполнения:

- Разархивируйте файл **counter_Lab.zip** в папку **c:\NI\LabVIEW_Labs**
- Запустите на исполнение LabVIEW **(Start>>All Programs>>National Instruments>>LabVIEW)**
- Выберите вариант **File>>New Project**
- В окне *Project Explorer* щелкните правой кнопкой мыши по пункту **My Computer** и выберите **New>>Targets and Devices**

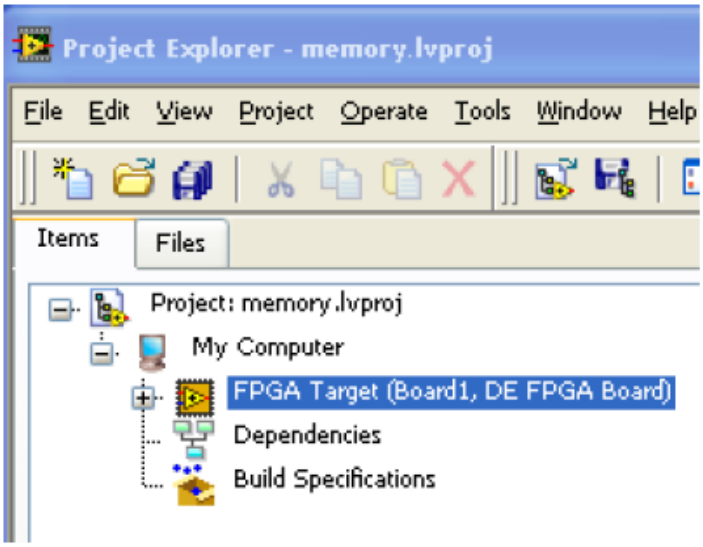


- Установите флажок в позиции *New target or device* и выберите **DE FPGA Board** в секции **NI ELVIS**

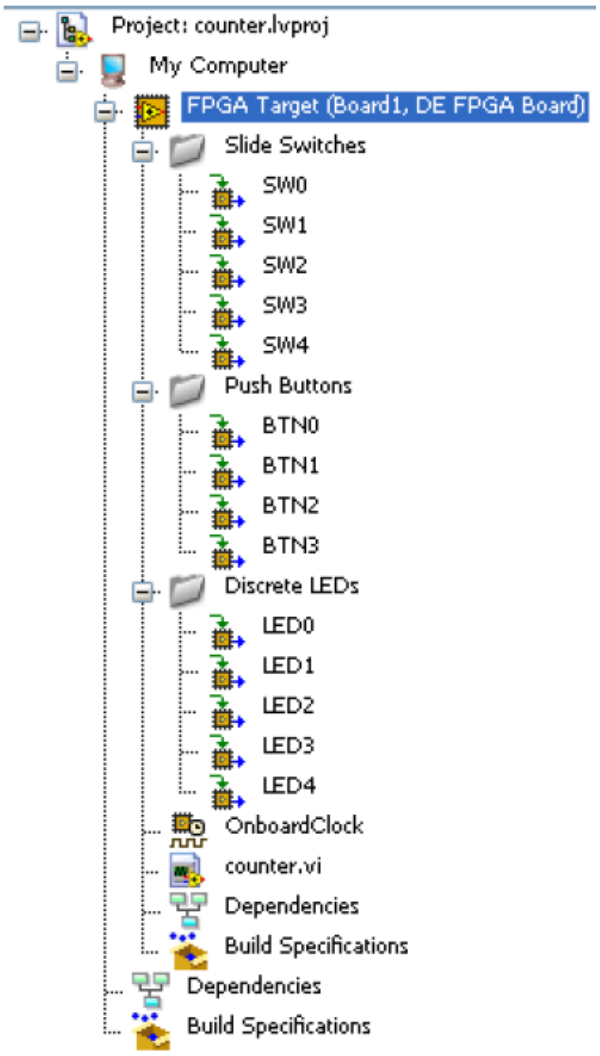


- Щелкните по кнопке OK
- Выберите в меню **File>>Save** и сохраните проект под именем **counter** в папке **c:\NI\LabVIEW_Labs\counter_Lab**

- В окне *Project Explorer* щелкните правой кнопкой мыши по пункту **FPGA Target (Board1, DE FPGA Board**) и выберите **New VI**



- Выберите в меню **File>>Save** и сохраните VI *под именем* **counter** в папке **c:\NI\LabVIEW_Labs\counter_Lab**
- Теперь добавляем каналы ввода-вывода FPGA
- В окне *Project Explorer* щелкните правой кнопкой мыши по пункту **FPGA Target (Board1, DE FPGA Board**) и выберите **New>>FPGA I/O**
- Доступные в оценочном модуле (DE FPGA Board) каналы ввода-вывода отображаются в левой части окна выбора
- В секции **Available Resources** раскройте папку **Slide Switches**, выберите **SW0** для входа направления счета – состояние ON – сложение, OFF – вычитание. Выберите **SW4÷ SW1** для загрузки числа, затем щелкните по кнопке **Add**, чтобы добавить каналы ввода-вывода в проект
- В секции **Available Resources** раскройте папку **Push Buttons**, выберите **BTN0**, **BTN1**, **BTN2** и **BTN3** (BTN0 – для сброса счетчика, BTN1 – для параллельной загрузки, BTN2 – для сохранения и BTN3 – для окончания счета). Щелкните по кнопке **Add**, чтобы добавить каналы ввода-вывода в проект. Аналогично, раскройте папку **Discrete LEDs** и добавьте **LED4** в качестве индикатора окончания счета, а **LED3÷LED0** – для индикации состояния счетчика
- Щелкните по кнопке **ОК** для подтверждения выбора и закройте окно выбора
- Ниже на рисунке показано, как должно выглядеть окно **Project Explorer** с внесенными дополнениями



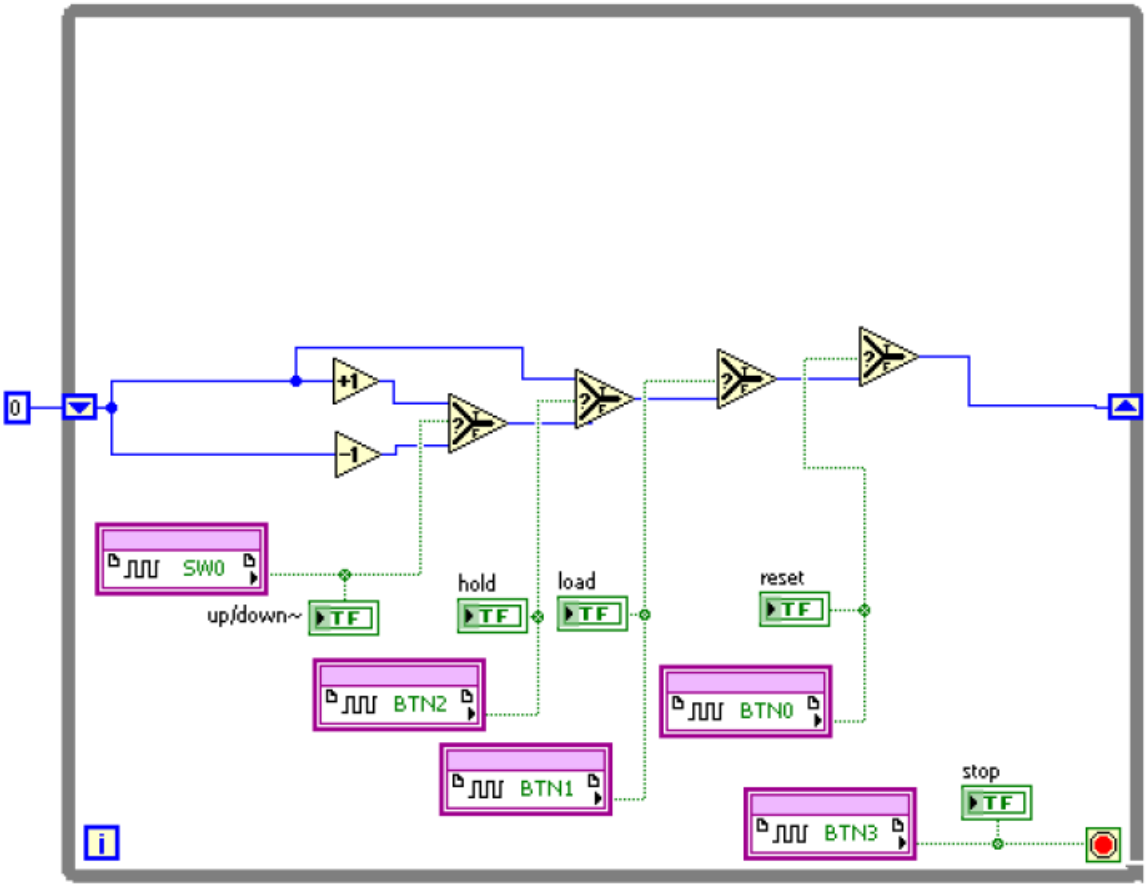
- В окне блок-диаграммы поместите цикл **while loop**
- Щелкните правой кнопкой мыши по границе цикла while loop и выберите вариант **Add Shift Register**
- Создайте вне цикла числовую константу и подключите ее ко входу Shift Register
- Щелкнув правой кнопкой мыши по константе и, выбрав **Representation**, измените представление чисел для константы на **U8**

Поскольку наш программируемый счетчик должен управляться входами с заданным приоритетом – сброс обладает высшим приоритетом, более низким приоритетом обладает хранение, затем загрузка – нам нужно три функции *select*. Нам нужна еще одна функции *select* для управления инкрементом или декрементом

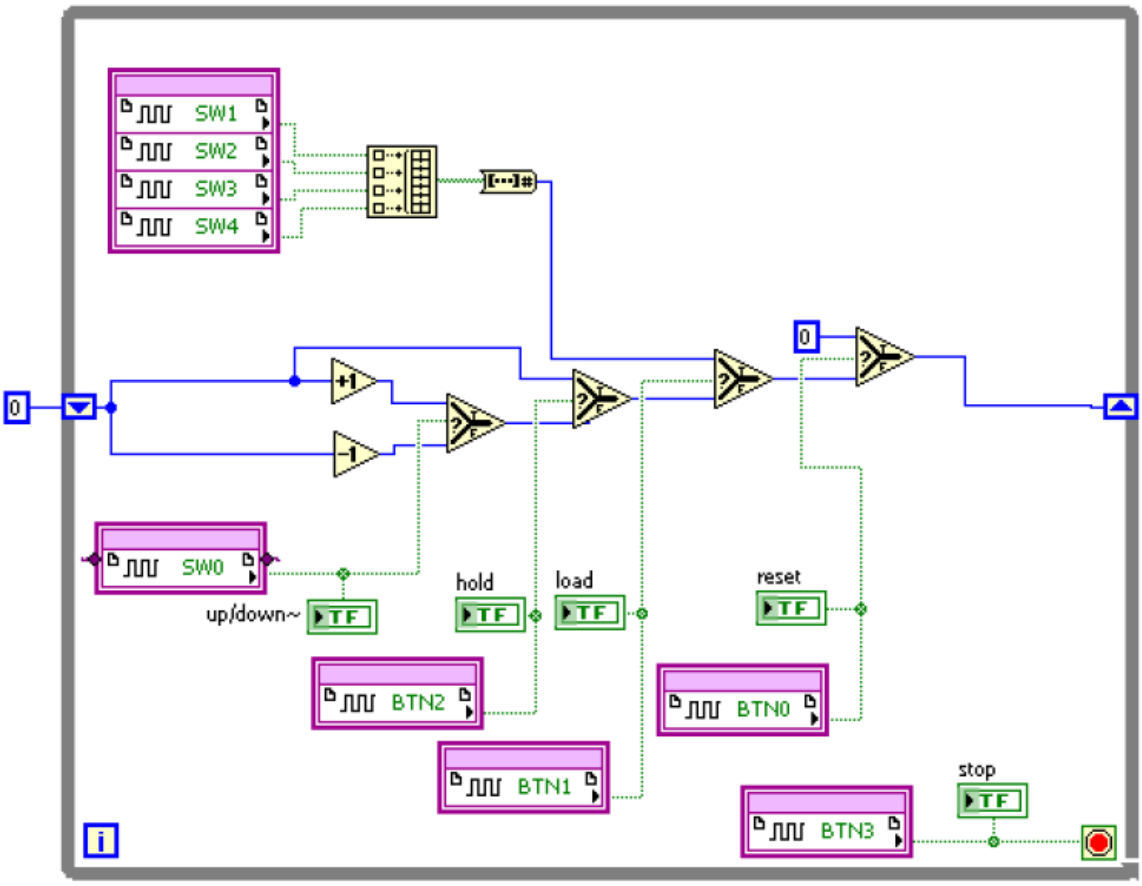
- Поместите на блок-диаграмму 4 функции **select** из палитры **comparison**
- В окне Project Explorer выберите все 4 кнопки и перетащите их в цикл while
- Аналогично выберите в окне Project Explorer переключатель (SW0) и перетащите его в цикл while
- Добавьте функции инкремента (**+1**) и декремента (**-1**) из палитры **Numeric**
- Для узлов SW0 и BTN0÷BTN3 создайте индикаторы
- Обозначьте индикаторы в соответствии с таблицей

BTN0	reset
BTN1	load
BTN2	hold
BTN3	stop
SW0	up/down~

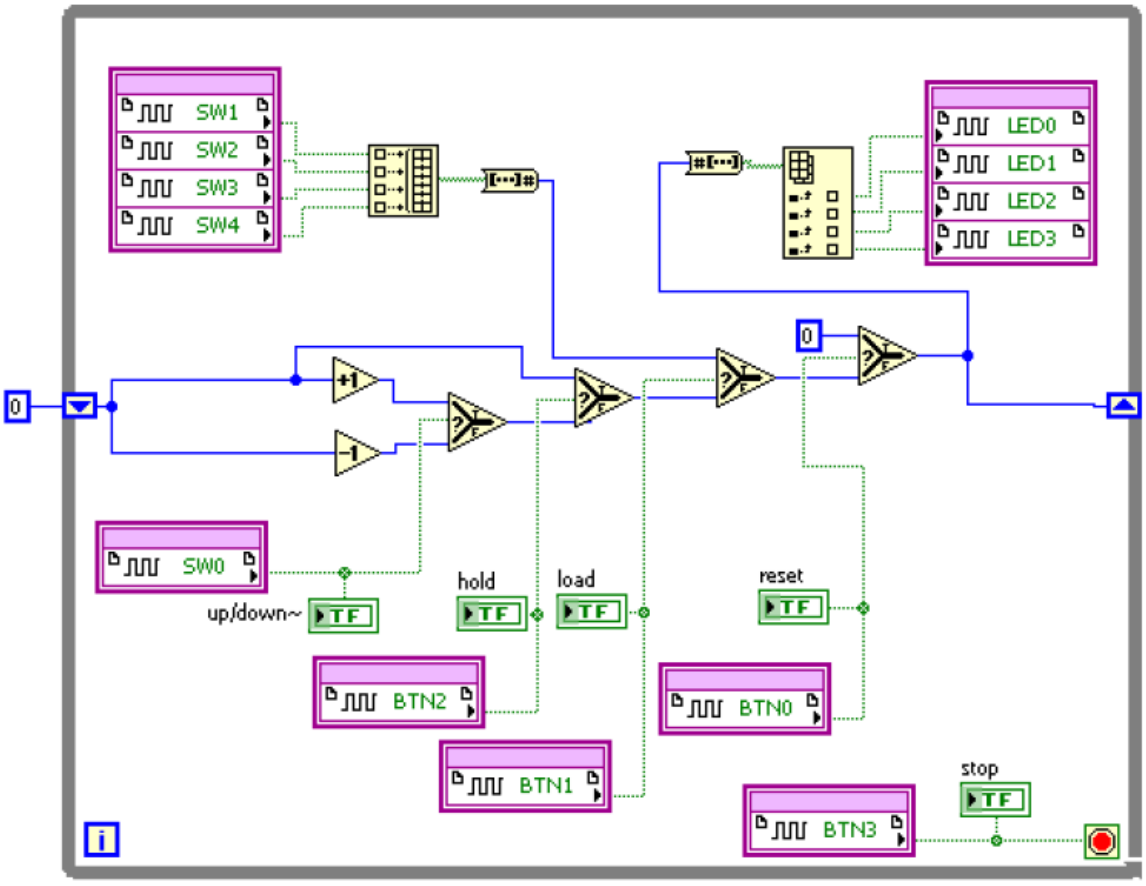
- Соедините индикаторы, как показано на следующей блок-диаграмме



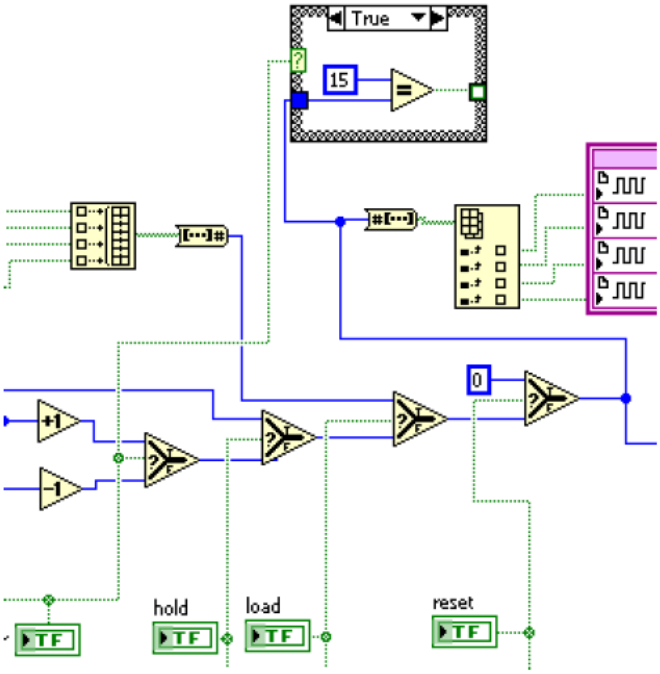
- Создайте константу на входе True крайней правой функции select
- Где-либо внутри цикла while loop щелкните правой кнопкой мыши и поместите из субпалитры **FPGA I/O** узел **FPGA I/O Node** ( ). Вы можете также найти этот узел, щелкнув по кнопке “Search”
- Щелкнув по узлу и выбрав **Slide Switches>>SW1**, присвойте его линии **SW1**
- Растяните узел так, чтобы в него вошли элементы от **SW1** до **SW4**
- Из палитры **Array** поместите функцию **Build Array** и растяните ее на 4 элемента
- Подключите выходы **SW1÷SW4** ко входам функции **Build Array**
- Поместите функцию **Boolean Array to Number** из палитры **Boolean** и соедините выход функции **Boolean Array** со входом функции **Boolean Array to Number**
- Соедините выход функции **Boolean Array** со входом True 2-й справа функции select, которой управляется BTN1
- На этом этапе проектирования блок-диаграмма должна выглядеть, как показано на рисунке ниже



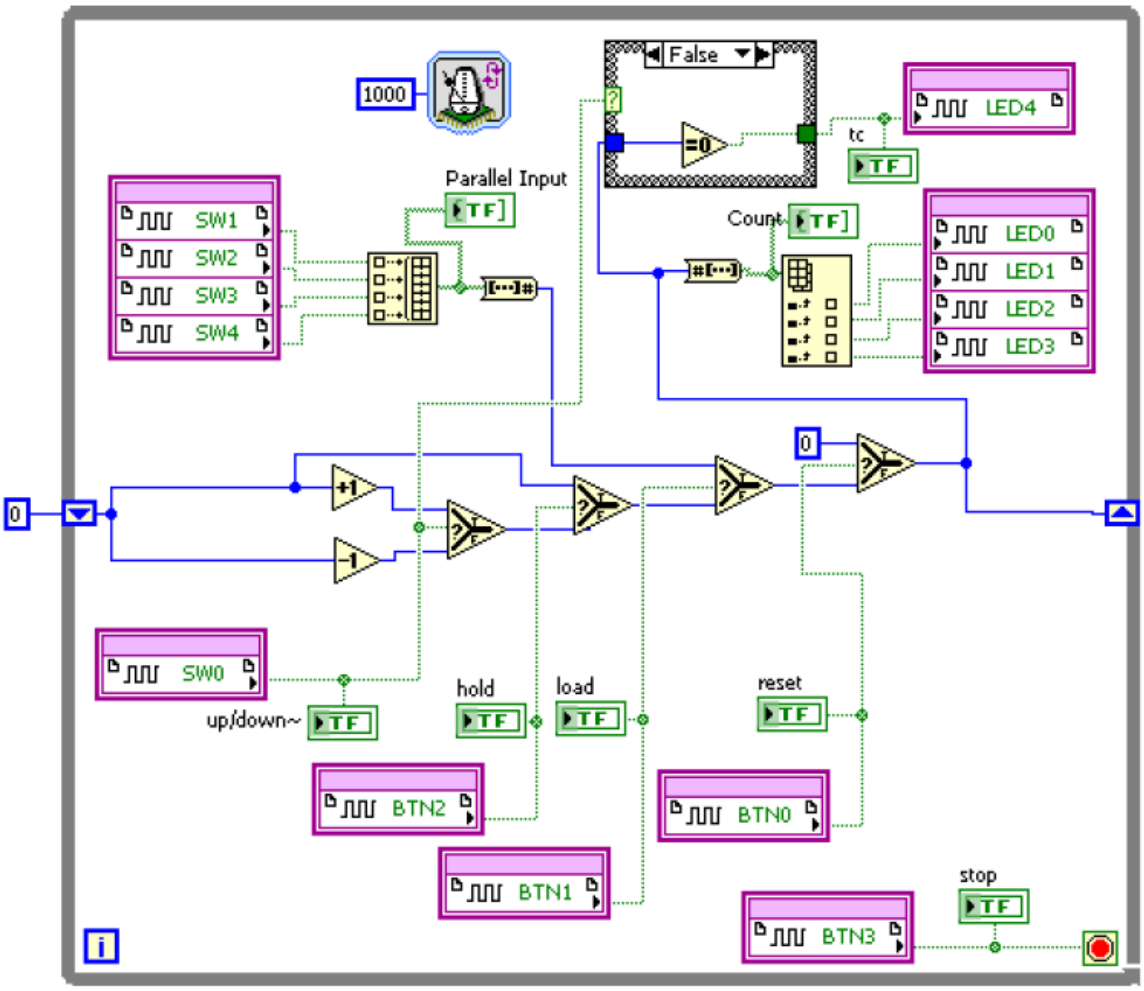
- Добавьте узел **FPGA I/O Node** из субпалитры **FPGA I/O**
- Присвойте его **LED0**
- Щелкнув правой кнопкой мыши по узлу LED и выбрав **Change to Write**, измените его назначение для записи
- Растяните узел так, чтобы в него вошли 4 элемента **LED0 LED3**
- Из палитры **Array** поместите функцию **Index Array**
- Растяните ее на 4 элемента
- Добавьте функцию **Number to Boolean Array** из палитры **Boolean**
- Соедините выход правой крайней функции select со входом функции **Number to Boolean Array**, выход которой соедините со входом функции **Index Array**, а ее выходы к отдельным элементам **LED**
- На этом этапе проектирования блок-диаграмма должна выглядеть, как показано на рисунке ниже



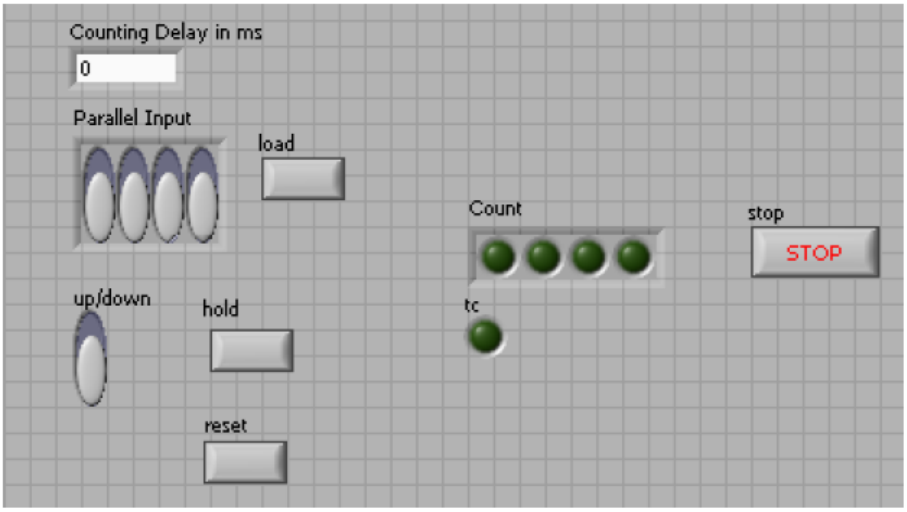
- Добавьте структуру **Case** для выхода окончания счета и подключите элемент up/down к селектору выбора (**?**) структуры Case
- Из субпалитры **comparison** функций сравнения поместите блок **Equal?** (компаратор) во фрейм True структуры **Case**. Соедините выход правой крайней функции select с одним из входов компаратора в структуре **Case**. Создайте константу, равную 15-ти, на втором входе блока **Equal?**
- Соедините выход компаратора с границей структуры **Case**, чтобы вывести результат сравнения. Часть измененной блок-диаграммы показана ниже



- Выберите фрейм **False** структуры Case и поместите в него блок **Equal to 0** (компаратор нуля), соедините входной туннель структуры Case со входом этой функции
- Соедините выход компаратора нуля с выходным туннелем структуры Case
- В окне Project Explorer выделите элемент **LED4**, перетащите его на блок-диаграмму и соедините с выходным туннелем структуры Case
- Добавьте индикатор и обозначьте его **tc**
- Переключитесь на лицевую панель (Ctrl-E)
- Поместите на лицевую панель массив из палитры **Array and Cluster**
- Поместите в массив круглый светодиод и растяните массив (в горизонтальном направлении), чтобы были видны 4 светодиода
- Измените имя массива на **Count**
- Щелкнув правой кнопки мыши по массиву и выбрав пункт **Visible Items>>Index Display**, скройте индикатор индекса массива
- Добавьте еще один массив и поместите в него переключатель. Растяните массив так, чтобы были видны 4 переключателя, и обозначьте массив, как **Parallel Input**. Индикатор индекса этого массива также сделайте невидимым
- Переключитесь на блок-диаграмму и соедините **Count** с выходом функции **Number to Boolean Array**
- Щелкнув правой кнопки мыши по массиву **Parallel Input**, измените его режим на индикатор
- Соедините Parallel Input с выходом функции Build Array
- Из палитры **Timing** поместите узел **Loop Timer** и установите единицу счета **ms**
- Создайте константу на входе узла Loop Timer и присвойте ей значение **1000** (1000 ms = 1s)
- Ниже показан результат проектирования



- Переключитесь на лицевую панель (Ctrl-E)
- Измените внешний вид и оформление различных входов, измените размеры и положение других объектов, чтобы лицевая панель стала более понятной и выразительной, как показано ниже

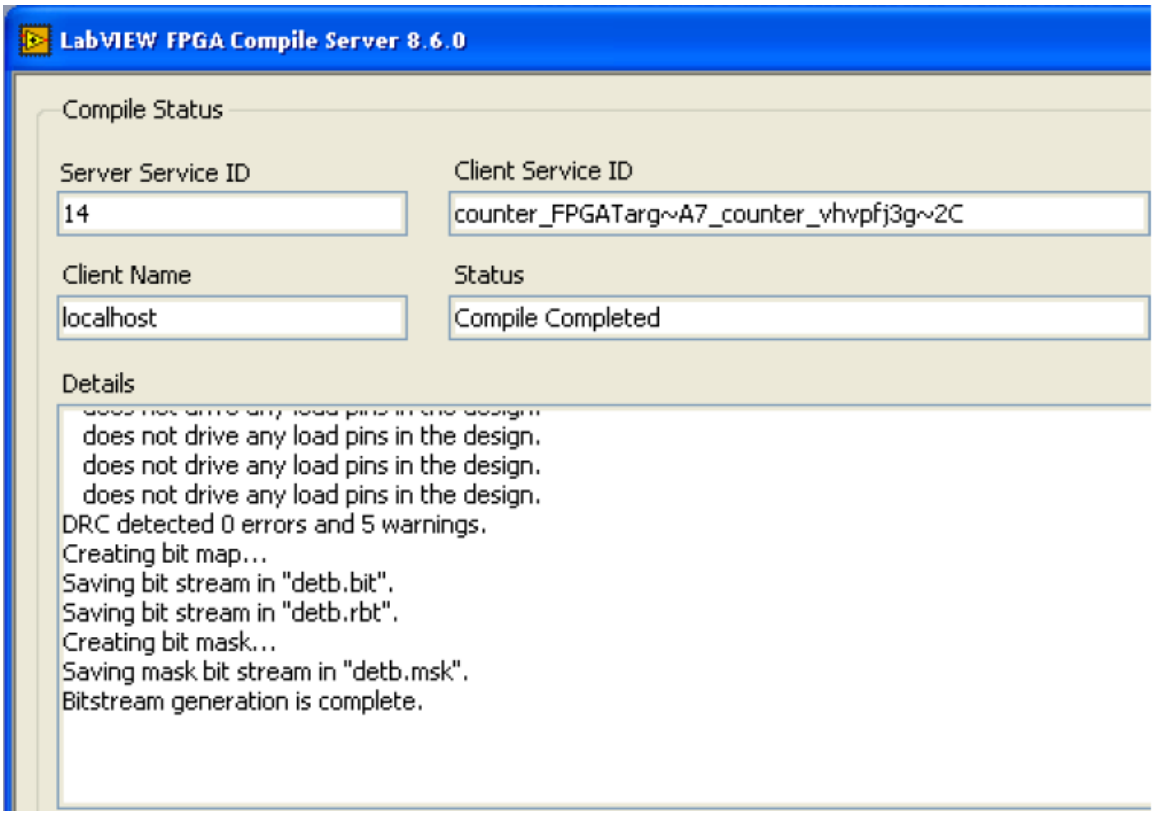


- Сохраните VI

Раздел 2: Проверка результатов проектирования с помощью оценочного модуля

Порядок выполнения:

- Подключите оценочный модуль, USB кабель и включите питание
- Щелкните по кнопке **Run** () или выполните команду меню **Operate>>Run**
- Как только завершится генерация двоичного кода **Bitstream**, о чем появится сообщение в окне хода компиляции **Compile Server**, закройте окно щелчком по кнопке X в правом верхнем углу окна
- Щелкните по кнопке **ОК**, чтобы закрыть сводное окно состояния компиляции



FPGA сконфигурируется, об этом свидетельствует подсветка кнопки RUN и включится светодиод DONE на модуле, и вы можете нажать кнопку, чтобы протестировать ваш проект

Установите на оценочном модуле переключатели SW4÷SW1 в положение 0101 (число 5) и SW0 в положение 1 (сложение). Нажмите кнопку BTN1 (Load) на модуле для загрузки установленного значения, а затем отпустите кнопку [постарайтесь удерживать нажатой кнопку около 1 секунды, т.к. задержка на обнаружение нажатия кнопки в устройстве составляет около 1 секунды]. Наблюдайте счет значений по светодиодам модуля, а также светодиодам лицевой панели.
Нажмите кнопку BTN0, а затем отпустите кнопку, вы увидите, что число в счетчике равно "0000".
Нажмите кнопку BTN2 и оставьте ее нажатой, наблюдайте приостановку счета.
Отожмите кнопку BTN2 и наблюдайте возобновление счета.

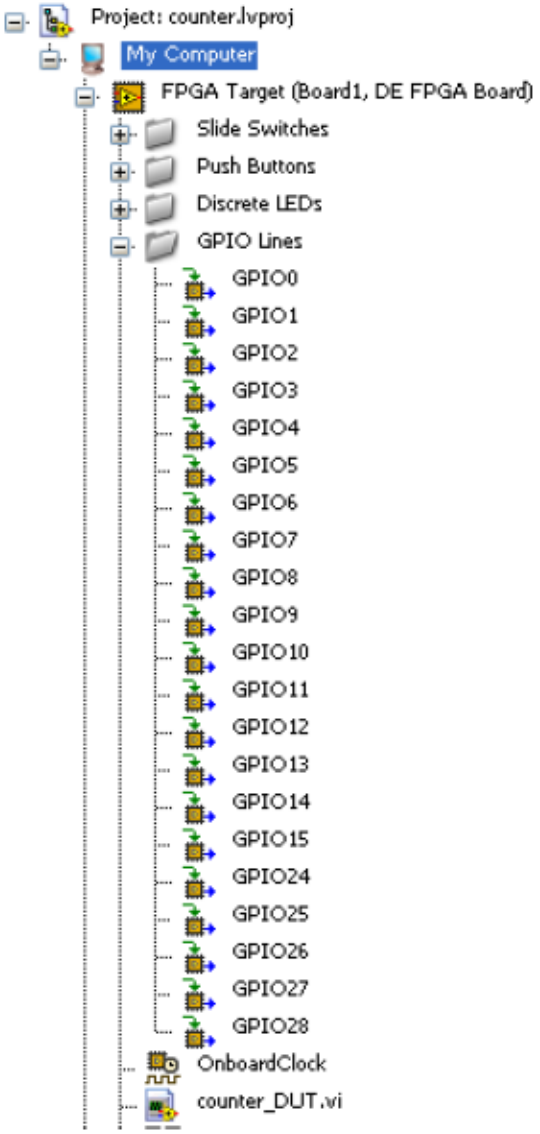
Нажмите кнопку BTN0 для сброса счетчика и, когда он досчитает до "1111" (или 15), светодиод LED4 (**tc**) включится на 1 секунду. Обратите внимание, что если число в счетчике становится больше 15, счетчик продолжает считать, но светодиод **tc** не включается. Счетчик считает до 256, т.к. мы выбрали тип данных U8.
Теперь измените положение переключателя SW0 на "0" (вычитание) и наблюдайте обратный счет значений. Также вы увидите, что **tc** включается, когда счетчик приходит в состояние "0000 (ноль).

После завершения тестирования щелкните по кнопке Stop () или кнопке BTN3 для прерывания работы схемы.

Раздел 3: Проверка результатов проектирования с помощью VI тестера

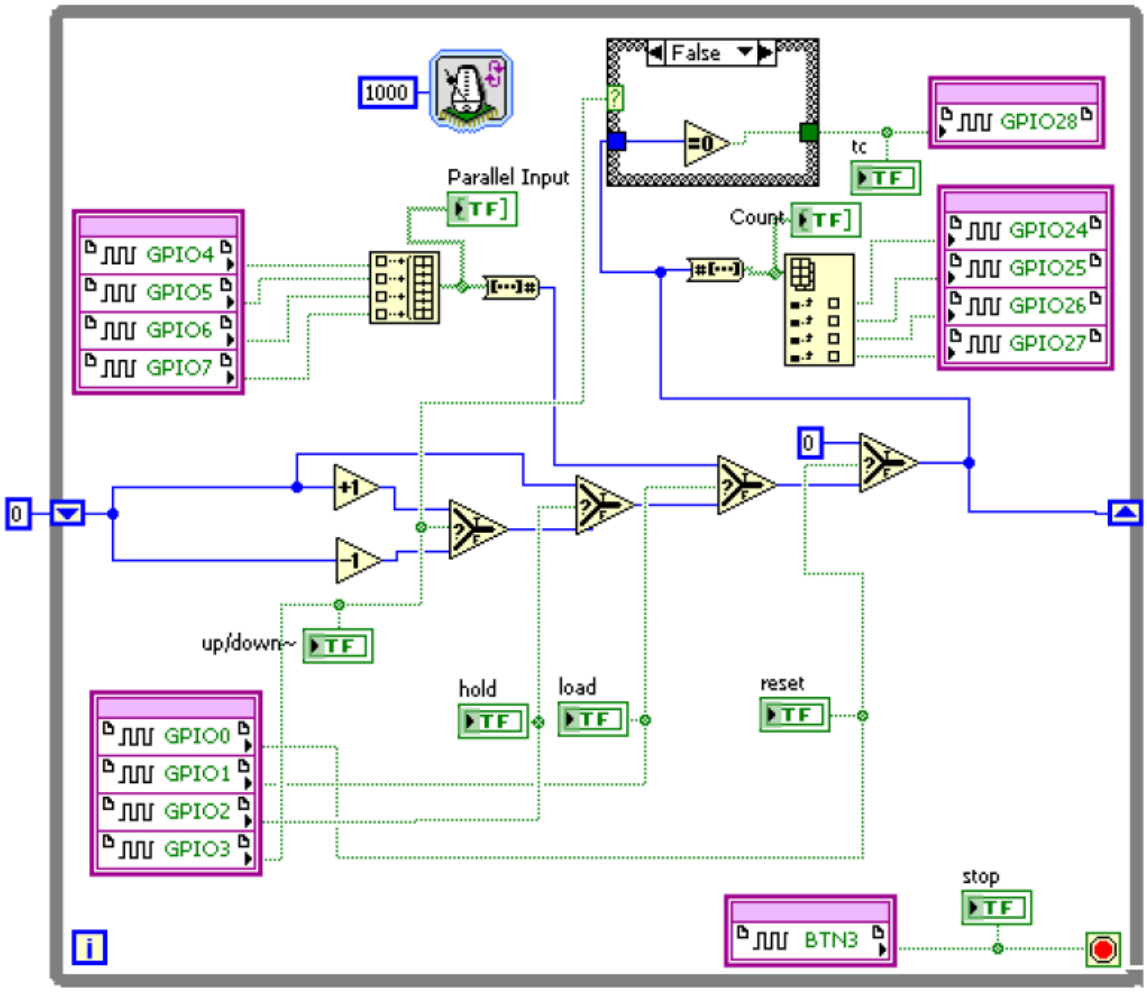
Порядок выполнения:

- Сохраните VI под именем **counter_DUT.vi**
- На блок-диаграмме, оставив соответствующие индикаторы, удалите переключатель, светодиоды и все кнопки, за исключением BTN3
- В окне Project Explorer щелкните правой кнопкой мыши по пункту **FPGA Target** и выберите **New>>FPGA I/O**
- В открывшемся окне отройте папку линий **GPIO**, выделите линии **GPIO0÷ GPIO15** и добавьте их в проект, а затем щелкните по кнопке **ОК**
- Окно Project Explorer должно выглядеть так



- Поместите на блок-диаграмму из субпалитры **FPGA I/O** узел **FPGA I/O Node**
- Щелкните правой кнопкой мыши по добавленному узлу и выберите **GPIO0**. Растяните узел так, чтобы в него вошли элементы от **GPIO0** до **GPIO3**
- Соедините элементы **reset**, **load** и **up/down** с входами **GPIO0**, **GPIO1**, **GPIO2** и **GPIO3**

- Поместите на блок-диаграмму из субпалитры **FPGA I/O** узел **FPGA I/O Node**
- Щелкните правой кнопкой мыши по добавленному узлу и выберите **GPIO4**.Растяните узел так, чтобы в него вошли элементы от **GPIO4** до **GPIO7**
- Соедините вход функции Build Array элементы **reset**, **load** и **up/down** с входами **GPIO4**, **GPIO5**, **GPIO6** и **GPIO7** (см. блок-диаграмму ниже)
- Поместите на блок-диаграмму из субпалитры **FPGA I/O** узел **FPGA I/O Node**
- Щелкните правой кнопкой мыши по добавленному узлу и выберите **GPIO24**.Измените его назначение для записи. Растяните узел так, чтобы в него вошли элементы от **GPIO24** до **GPIO27**
- Соедините выходы функции Index Array с входами **GPIO24**, **GPIO25**, **GPIO26** и **GPIO27** (см. блок-диаграмму ниже)
- Поместите на блок-диаграмму из субпалитры **FPGA I/O** узел **FPGA I/O Node**
- Щелкните правой кнопкой мыши по добавленному узлу и выберите **GPIO28**.Измените его назначение для записи
- Соедините tc с входом **GPIO28** (см. блок-диаграмму ниже)
- Ниже показана полученная блок-диаграмма

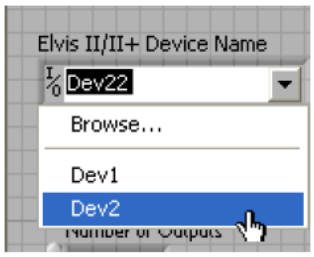


- Сохраните VI
- Щелкните по кнопке **Run**. Начнется компиляция проекта. Когда завершится генерация битового массива, щелкните по кнопке OK, чтобы закрыть диалоговое окно. FPGA сконфигурирован

- В окне Project Explorer щелкните правой кнопкой мыши по ветви **My Computer**, выберите **Add>>File** и из каталога проекта добавьте **Simple FPGA Tester.vi**
- В окне Project Explorer щелкните дважды по **Simple FPGA Tester.vi**, откроется окно VI
- Ознакомьтесь с блок-диаграммой и разберитесь с ее содержанием
- Обратите внимание, что Digital Writer использует каналы DIO с 8 по 15 [канал 15 - младший] для записи в FPGA, а Digital Reader использует каналы DIO с 0 по 7 [канал 7 - младший] для чтения откликов из FPGA
- Проводниками для макетной платы физически соедините контакты разъемов BB1/BB2 и BB5 согласно следующей таблице

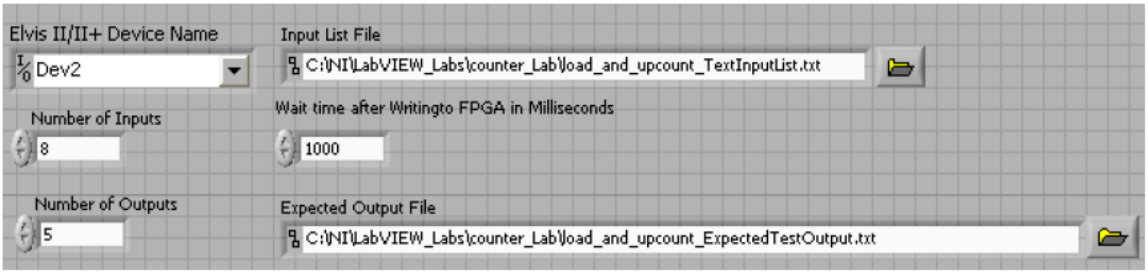
BB1/BB2 Connectors	BB5 Connector
GPIO0	DIO15
GPIO1	DIO14
GPIO2	DIO13
GPIO3	DIO12
GPIO4	DIO11
GPIO5	DIO10
GPIO6	DIO9
GPIO7	DIO8
GPIO24	DIO7
GPIO25	DIO6
GPIO26	DIO5
GPIO27	DIO4
GPIO28	DIO3

- Щелкните по стрелке в поле **Elvis II/II+ Device Name** и выберите устройство, соответствующее Elvis II

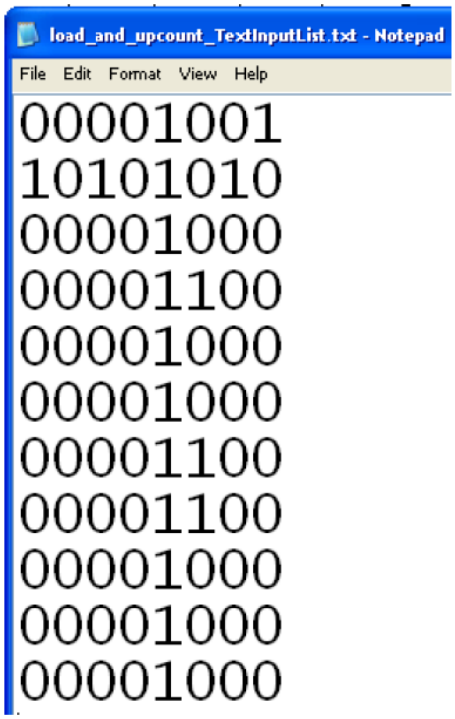


- Измените **Number of Inputs** на **8** и **Number of Outputs** на **5**
- Измените время Wait на **1000** ms
- Щелкните по кнопке Browse для файла Input и добавьте **load_and_upcount_TextInputList.txt**
- Таким же образом добавьте **load_and_upcount_ExpectedOutputData.txt** с ожидаемыми состояниями выходов.

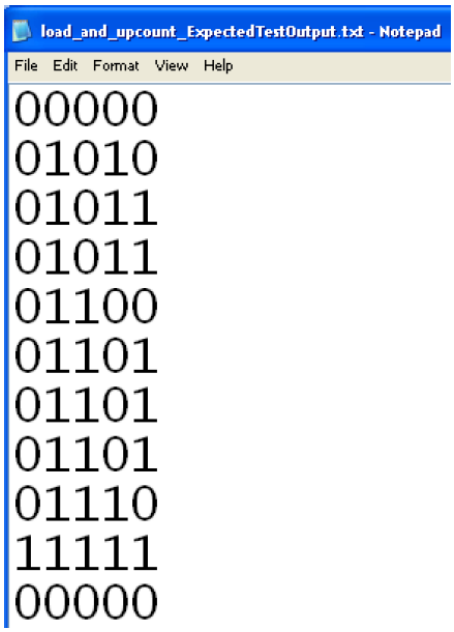
На этом этапе проектирования лицевая панель должна выглядеть похожей на изображенную ниже



Файл входных данных – текстовый, его можно открыть/создать любым текстовым редактором. На следующем рисунке показано содержимое этого файла



В файле входных данных описаны входы в следующем порядке: входы данных для параллельной записи [3], [2], [1], [0] ([3] – левый крайний), up/down, hold, load и reset (крайний правый бит). Файл выходных данных – тоже текстовый, и его тоже можно открыть/создать любым текстовым редактором. На следующем рисунке показано содержимое этого файла



В соответствии с этим выходным файлом каналам с DIO0 по DIO2 автоматически присвоено значение ноль, а канал 3 назначен для tc (terminal count) (левый крайний бит), каналы с 4 по 7 – для выходов счетчика [3], [2], [1], [0] ([3] – правый крайний)

- Удостоверьтесь, что **counter_DUT.vi** продолжает исполняться
- Щелкните по кнопке Run на лицевой панели тестера. Тест запустится на исполнение, а его результаты отображаются на лицевой панели
- Теперь измените содержимое файла ожидаемых выходных реакций. Измените четвертый набор выходных сигналов с 01011 на 11111, при этом тест должен показать ошибку. Сохраните изменения
- Выключите и включите питание ELVIS и оценочного модуля FPGA
- Запустите тест еще раз и обратите внимание, что тестер сообщает об ошибке (Failed). Он также сообщает, где ошибка, какие были входные сигналы, ожидаемые выходные сигналы, какие были реальные выходные сигналы, включается индикатор ошибки
- Остановите исполнение **counter_DUT.vi**, закройте проект, закройте LabVIEW, выключите питание NI ELVIS II и оценочного модуля

Выводы:

В этой работе вы научились проектировать программируемые счетчики с использованием структур while loop и case. Вы также узнали, как реализовать логику приоритетов с помощью функции select для обеспечения требуемой приоритетности функций. Вы протестировали проект с помощью оценочного модуля DE FPGA Board и системы NI ELVIS.