Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Высшая школа программной инженерии

Отчёт по лабораторной работе «Программа DP32»

по дисциплине «Пакеты проектирования программно-автоматных комплесов»

Выполнил
студент гр. в3530904/00030 В.С. Баганов

Старший преподователь А.В.Петров

«____»_____ 202__ г.

Санкт-Петербург
2023

Содержание

1. Введение

Лабораторная работа посвящена решению индивидуального задания на упрощенном 32-х битном процессоре dp_32 (digital processor 32-bit), с помощью симуляции в программе Active VHDL.

2. Задание

Найти сумму квадратов наименьшего и наибольшего элементов одномерного массива.

3. Описание программы

В памяти процессора выделяется место для элементов массива. Размер определяется начальным содержимым регистра r1. Работа программы состоит из двух циклов. Сначала из массива выбирается элемент и помещается в переменную min. Затем последовательно из массива читаются элементы и сравниваются с min. Если min окажется меньше текущего выбранного, то чтение продолжается и только что выбранный элемент снова записывается в массив (в конец). Если же выбранный элемент окажется меньше min, то min получает его значение, а прежнее значение min записывается в массив. Выборка элементов продолжается k-1 раз, где k – количество элементов. В результате после процесса перебора в min будет содержаться минимальный элемент. Он заносится в ячейку памяти под номером 110. Далее из массива выбирается элемент и помещается в переменную max. Затем последовательно из массива читаются элементы и сравниваются с max. Если max окажется больше текущего выбранного, то чтение продолжается и только что выбранный элемент снова записывается в массив (в конец). Если же выбранный элемент окажется больше max, то max получает его значение, а прежнее значение max записывается в массив. Выборка элементов продолжается k-1 раз, где k – количество элементов. В результате после процесса перебора в max будет содержаться максимальный элемент. Он заносится в ячейку памяти под номером 111, значение также сохраняется в регистре r4. Далее возводим в квадрат регистры r4 и r8, и суммируем. В результате получаем сумму квадратов минимального и максимального элементов массива.

4. Блок-схема программы

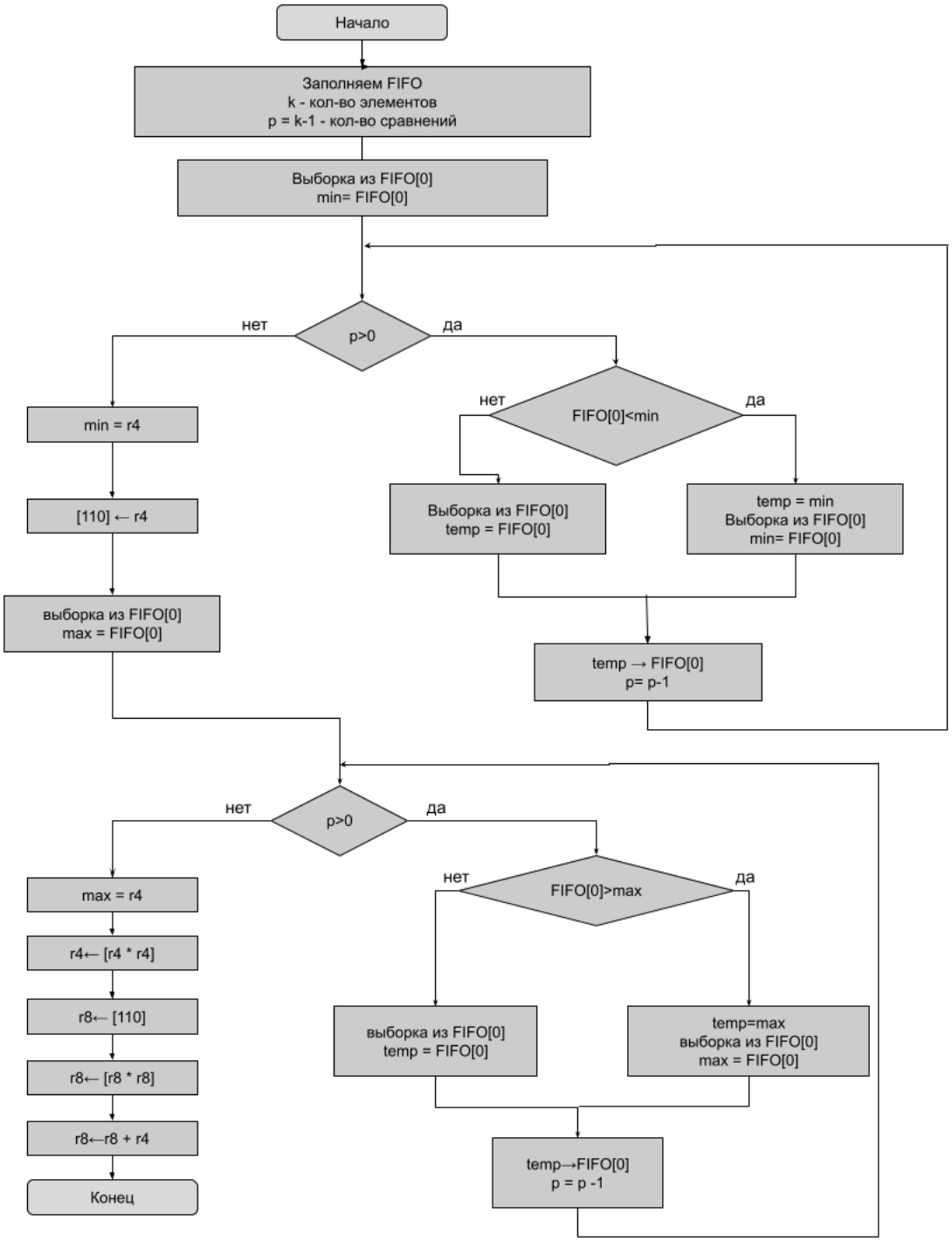


Рисунок 4.1. Блок-схема программы сортировки массива

5. Листинг трансляции программы

```
1. ; 0  -     r0  - регистр для операций
2. ; k = 4 - r1  - кол-во элементов в очереди
3. ; p = 4 - r2  - кол-во сравнений элементов
4. ; n = 4 - r3  - кол-во элементов в FIFO
5. ; min   - r4  - число, минимальный элемент (для сортировки)
6. ; temp  - r5  - промежуточное значение
7. ; temp1 - r6  - промежуточное значение
8. ; temp2 - r7  - промежуточное значение
9. ; max   - r8  - число,максимальный элемент
10.;     134 ячейка r8 -  Результат (Сумма квадратов наименьшего и
 ↪  наибольшего значения)
11.
12.
13.
14.
15.         _____________________ begin
16.         (0000) [1000_0000]     addq r0, r0, 0     ;  r0 ← r0 + 0
17.         (0001) [1001_0005]     addq r1, r0, 5     ;  r1 ← r0 + 5
18.         (0002) [1002_0100]     addq r2, r1, 0     ;  r2 ← r1 + 0
19.         (0003) [1003_0000]     addq r3, r0, 0     ;  r3 ← r0 + 0
20.
21.         _____________________ k_greate_0:
22.
23.         (0004) [3004_0082]     ldq  r4, [r0+100]   ; r4 ← FIFo[0]
24.         (0005) [1101_0101]     subq r1, r1, 1      ; r1 ← r1 - 1
25.         (0006) [1002_0100]     addq r2, r1, 0      ; r2 ← r1 +0
26.
27.         (0007) [1001_0100]     addq r1, r1, 0      ; if r1 = 0
28.         (0008) [4109_0000
29.       (0009) [0000_0010]     bi i, z, r0 if_k_0;  then goto if_k_0
30.         (0010) [1005_0000]     addq r5, r0, 0      ;  r5 ←r0 + 0
31.
32.         _____________________ vyborka:
33.
34.         (0011) [1005_0501]     addq r5, r5, 1      ;  r5 ← r5 +1
35.         (0012) [3006_0582]     ldq  r6, [r5+130]   ;  r6 ← FIFO[r5]
36.         (0013) [3106_0581]     stq  r6, [r5+130]   ;  FIFO[r5-1] ←r6
37.         (0014) [0108_0501]     sub  r8, r5, r1     ;  if r5 < r1
38.         (0015) [500A_00FB]    brq  i, n, z vyborka; then goto vyborka
39.
40.         _____________________if_k_0:
41.
42.         (0016) [1001_0100]     addq r1, r1, 0      ;  if r1 < 0
43.         (0017) [410B_0000
44.       (0018) [0000_002C]   bi i, n, z, r0 else_k_0; then goto else_k_0
45.
46.         _____________________ if_p_0:
47.
48.         (0019) [1002_0200]     addq r2, r2, 0      ;  if r2 < 0
49.         (0020) [410B_0000
50.         (0021) [0000_0029]   bi i, n, z, r0 else_p_0;then goto
 ↪  else_p_0
51.         (0022) [3006_0082]     ldq  r6, [r0+130]   ;   r6 ← FIFO [0]
52.         (0023) [0108_0406]     sub  r8, r4, r6     ;   if r6 > r4
53.         (0024) [410B_0000
54.         (0025) [0000_001E]     bi   i, n, z, r0 min;  then goto min
```

```
46.         (0026) [1005_0400]     addq r5, r4, 0      ;  r5 ← r4 + 0
47.         (0027) [1004_0600]     addq r4, r6, 0      ;  r4 ← r6 + 0
48.         (0028) [4100_0000
49.         (0029) [0000_001F]     bi   r0 sdvig       ;  goto sdvig

50. ______________________min:

51.         (0030) [1005_0600]     addq r5, r6, 0      ;  r5 ← r6 + 0
52.
53. ______________________ sdvig:

54.         (0031) [1007_0000]     addq r7, r0, 0      ;  r7 ← r0 + 0
55.
56.         ______________________ vybor_1:

57.         (0032) [1007_0701]     addq r7, r7, 1      ;  r7 ← r7 + 1
58.         (0033) [3006_0782]     stq  r6, [r7+130]   ;  r6 ← FIFO[r7]
59.         (0034) [3106_0781]     ldq  r6, [r7+129]   ;  FIFO [r7-1] ←r6
60.         (0035) [1108_0101]     subq r8, r1, 1      ;  if r7 <
61.         (0036) [0108_0708]     sub  r8, r7, r8     ;  (r1-1)
62.         (0037) [500A_00FA]   brq i, n, r0, vybor_1; then goto vybor_1
63.
64.         (0038) [3105_0181]     stq  r5, [r1+129]   ; FIFO [r1 -1] ←r5
65.         (0039) [1102_0201]     subq r2, r2, 1      ;  r2 ← r2 - 1
66.
67.         (0040) [5000_00EA]     brq  r0, if_p_0     ;  goto if_p_0
68.
69.         ______________________ else_p_0:
73.
74.         [3104_006F]     stq  r4, [r0+110]   ;  [110]← r4
75.         [0204_0404]     mull  r4, r4, r4    ;  r4 ← r4 * r4
76.         [3008_006E]     stq  r4, [r0+111]   ;  [111]← r4(max)
77.         [0208_0808]     mull  r8, r8, r8    ;  r8 ← r8 * r8
78.         [3108_0070]     stq  r8, [r0+112]   ;  [113]← r8
79.         [0008_0804]     add  r8, r8, r4     ;  r8 ← r8 + r4
80.         [3108_0071]     stq  r8, [r0+111]   ;  [113]← r8
91.
92.
93.
94.
95.         (0099) [0000_0000]
96.
97.                                        FIFO:
98.         (0100) [0000_0002]                      ; FIFO[0]
99.         (0101) [0000_0003]                      ; FIFO[1]
100.        (0102) [0000_0004]                      ; FIFO[2]
101.        (0103) [0000_0005]                      ; FIFO[3]
102.        (0104) [0000_0006]                      ; FIFO[4]
103.        (0105) [0000_0000]
104.        (0106) [0000_0000]
105.        (0107) [0000_0000]
106.        (0108) [0000_0000]
107.        (0109) [0000_0000]
108.        (0110) [0000_0000]                      ; min
109.        (0111) [0000_0000]                      ; min x min
110.        (0112) [0000_0000]                      ; max x max
111.        (0113) [0000_0000]                      ; sum
```

Листинг 1: Результат программы

6. Код программы

```
variable mem :memory_array:=
                          (0⇒X"1000_0000",
                           1⇒X"1001_0005",
                           2⇒X"1002_0100",
                           3⇒X"1003_0000",
                           4⇒X"3004_0064",
                           5⇒X"1101_0101",
                           6⇒X"1002_0100",
                           7⇒X"1005_0000",
                           8⇒X"1005_0501",
                           9⇒X"3006_0564",
                           10⇒X"3106_0563",
                           11⇒X"0108_0501",
                           12⇒X"500A_00FB",
                           13⇒X"1002_0200",
                           14⇒X"410B_0000_0000_0023",
                           16⇒X"3006_0064",
                           17⇒X"0108_0406",
                           18⇒X"410B_0000_0000_0018",
                           20⇒X"1005_0400",
                           21⇒X"1004_0600",
                           22⇒X"4100_0000_0000_0019",
                           24⇒X"1005_0600",
                           25⇒X"1007_0000",
                           26⇒X"1007_0701",
                           27⇒X"3006_0764",
                           28⇒X"3106_0763",
                           29⇒X"1108_0101",
                           30⇒X"0108_0708",
                           31⇒X"500A_00FA",
                           32⇒X"3105_0163",
                           33⇒X"1102_0201",
                           34⇒X"5000_00EA",
                           35⇒X"3104_006E",

                           36⇒X"1000_0000",
                           37⇒X"1001_0005",
                           38⇒X"1002_0100",
                           39⇒X"1003_0000",
                           40⇒X"1004_0000",
                           41⇒X"3004_0064",
                           42⇒X"1101_0101",
                           43⇒X"1002_0100",
                           44⇒X"1005_0000",
                           45⇒X"1005_0501",
                           46⇒X"3006_0564",
                           47⇒X"3106_0563",
                           48⇒X"0108_0501",
                           49⇒X"500A_00FB",
                           50⇒X"1002_0200",
                           51⇒X"410B_0000_0000_0049",
                           53⇒X"3006_0064",
                           54⇒X"0108_0604",
                           55⇒X"410B_0000_0000_003D",
                           57⇒X"1005_0400",
                           58⇒X"1004_0600",
                           59⇒X"4100_0000_0000_003E",
```

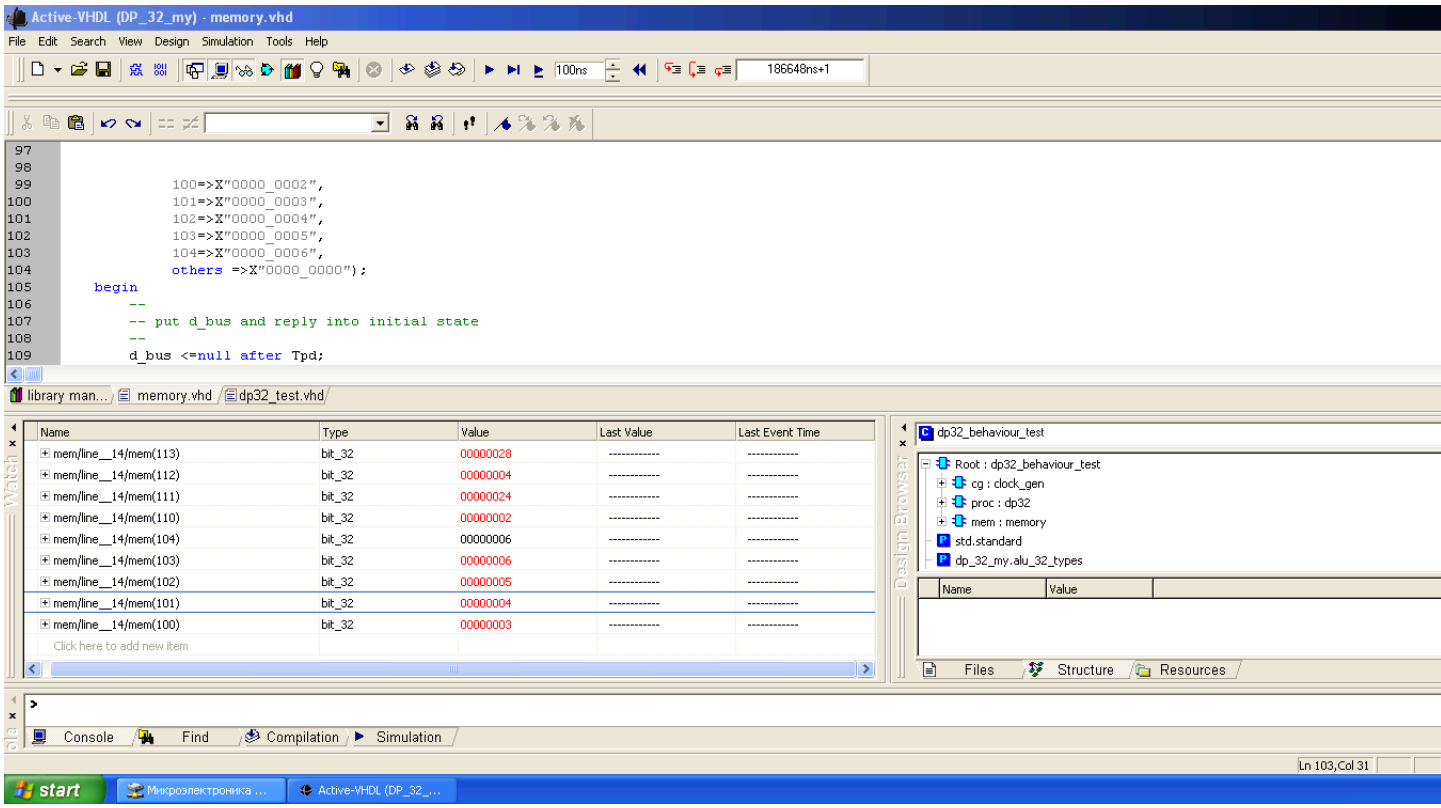
```
61=>X"1005_0600",
62=>X"1007_0000",
63=>X"1007_0701",
64=>X"3006_0764",
65=>X"3106_0763",
66=>X"1108_0101",
67=>X"0108_0708",
68=>X"500A_00FA",
70=>X"3105_0163",
71=>X"1102_0201",
72=>X"5000_00EA",
73=>X"0204_0404",
74=>X"3104_006F",
--73=>X"0204_0404",
75=>X"3008_006E",
76=>X"0208_0808",
78=>X"3108_0070",
79=>X"0008_0804",
80=>X"3108_0071",
--80=>X"0008_0804",
100=>X"0000_0002",
101=>X"0000_0003",
102=>X"0000_0004",
103=>X"0000_0005",
104=>X"0000_0006",
others =>X"0000_0000");
```

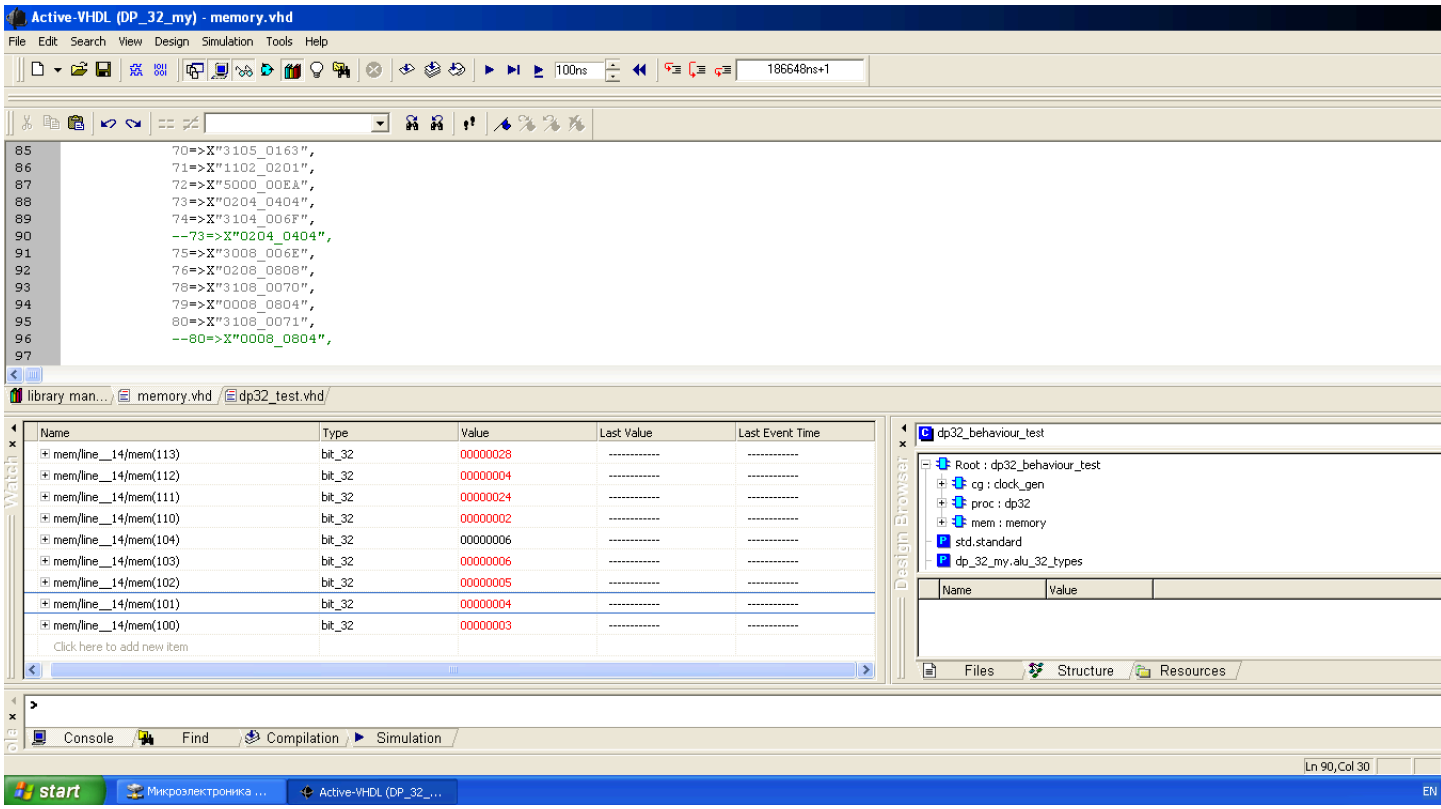
Листинг 2: Код программы

7. Результат работы программы

Name	Type	Value	Last Value	Last Event Time
mem/line__14/mem(113)	bit_32	00000028	----------	----------
mem/line__14/mem(112)	bit_32	00000004	----------	----------
mem/line__14/mem(111)	bit_32	00000024	----------	----------
mem/line__14/mem(110)	bit_32	00000002	----------	----------
mem/line__14/mem(104)	bit_32	00000006	----------	----------
mem/line__14/mem(103)	bit_32	00000006	----------	----------
mem/line__14/mem(102)	bit_32	00000005	----------	----------
mem/line__14/mem(101)	bit_32	00000004	----------	----------
mem/line__14/mem(100)	bit_32	00000003	----------	----------
Click here to add new item				

В 100 по 103 ячейках получили отсортированные значения. Минимальное значение получили в 110 ячейку, а максимальное значение в 104 ячейку. Возвели в квадрат максимальное значение в 111 ячейке (24 из 16-ричной в 10-чную будет 36). В 112 ячейке получили квадрат минимального значения. Результат, сумма квадратов минимального максимального значений массива, записан в ячейке 113 (28 из 16-ричной в 10-чную будет 40).




Active-VHDL (DP_32_my) - memory.vhd
File Edit Search View Design Simulation Tools Help
100ns
186648ns+1
70=>X"3105_0163",
71=>X"1102_0201",
72=>X"5000_00EA",
73=>X"0204_0404",
74=>X"3104_006F",
--73=>X"0204_0404",
75=>X"3008_006E",
76=>X"0208_0808",
78=>X"3108_0070",
79=>X"0008_0804",
80=>X"3108_0071",
--80=>X"0008_0804",
library man...
memory.vhd
dp32_test.vhd
Name
Type
Value
Last Value
Last Event Time
mem/line__14/mem(113)
mem/line__14/mem(112)
mem/line__14/mem(111)
mem/line__14/mem(110)
mem/line__14/mem(104)
mem/line__14/mem(103)
mem/line__14/mem(102)
mem/line__14/mem(101)
mem/line__14/mem(100)
bit_32
Click here to add new item
dp32_behaviour_test
Root : dp32_behaviour_test
cg : clock_gen
proc : dp32
mem : memory
std.standard
dp_32_my.alu_32_types
Name
Value
Files
Structure
Resources
Console
Find
Compilation
Simulation
Ln 90,Col 30
start
Микроэлектроника ...
Active-VHDL (DP_32_...
EN