BUILDING A RAM WITH INITIALIZED VALUES WITH XILINX WEBPACK

- Open your project.
- Project -> New Source.
- Select VHDL module
  - File Name: “DataRAM”
  - Select Next
  - Port Name “DataAddress”, in, MSB=7, LSB=0 (for a 256-entry RAM)
  - Port Name “clk”, in
  - Port Name “ReadMem”, in
  - Port Name “WriteMem”, in
  - Port Name “DataIn” , in, MSB=15, LSB=0
  - Port Name “DataOut”, out, MSB=15, LSB=0
  - Select Next
  - Select Finish
- Edit the VHDL code DataRAM.vhd, so it looks like this:

```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity DataRAM is
  Port ( DataAddress : in std_logic_vector(7 downto 0);
         clk : in std_logic;
         ReadMem : in std_logic;
         WriteMem : in std_logic;
         DataIn : in std_logic_vector (15 downto 0);
         DataOut : out std_logic_vector(15 downto 0));
end DataRAM;

architecture Behavioral of DataRAM is

  type ram_type is array (0 to 255) of std_logic_vector(15 downto 0);
  signal tmp_ram: ram_type := (
    1 => X"0008",
    9 => X"abcd",
    32 => X"3232",
    33 => X"ffff",
    34 => X"fffe",
    35 => X"0001",
    36 => X"0011",
    37 => X"a6a7",
    38 => X"aa38",
    39 => X"aa39",
    40 => X"bb40",
    41 => X"bb41",
    42 => X"4242",
    43 => X"ffff",
    44 => X"fffe",
    45 => X"0001",
    46 => X"0011",
  );
```

47 => X"a6a7",
48 => X"aa48",
49 => X"aa49",
50 => X"bb50",
51 => X"bb51",
52 => X"5252",
53 => X"ffff",
54 => X"fffe",
55 => X"0001",
56 => X"0011",
57 => X"a6a7",
58 => X"aa58",
59 => X"aa59",
60 => X"bb60",
61 => X"bb61",
62 => X"6262",
63 => X"ffff",
64 => X"fffe",
65 => X"0001",
66 => X"0011",
67 => X"a6a7",
68 => X"aa68",
69 => X"aa69",
70 => X"bb70",
71 => X"bb71",
72 => X"7272",
73 => X"ffff",
74 => X"fffe",
75 => X"0001",
76 => X"0011",
77 => X"a6a7",
78 => X"aa78",
79 => X"aa79",
80 => X"bb80",
81 => X"bb81",
82 => X"8282",
83 => X"ffff",
84 => X"fffe",
85 => X"0001",
86 => X"0011",
87 => X"a6a7",
88 => X"aa88",
89 => X"aa89",
90 => X"bb90",
91 => X"bb91",
92 => X"9292",
93 => X"fffe",
94 => X"fffe",
95 => X"0001",
96 => X"0001",
97 => X"0103",
98 => X"abab",
99 => X"3456",
100 => X"789a",
101 => X"cdcd",
102 => X"cdab",
103 => X"8765"
begin
process (ReadMem, DataAddress)
begin
  if ReadMem='1' then
    DataOut <= tmp_ram(conv_integer(DataAddress));
  else
    DataOut <= (DataOut'range => 'Z');
  end if;
end process;

process (clk, WriteMem)
begin
  if (clk'event and clk='1') then
    if WriteMem='1' then

This is for a 256-entry RAM (8-bit address). Your initial RAM contents will be exactly the same as specified here. Your code will change slightly if you have a larger RAM (see values in red), but these initial values will be the same.

- In “Sources” window, click DataRAM.vhd
- In “Processes” window, click “Check Syntax” (in “Synthesize XST”) – correct errors.
- In “Processes” window, click “Create Schematic Symbol” (in “Design Entry Utilities”)
- Open Schematic in which you want to include your RAM.
- “dataram” should now be added to your library of symbols. You may want to edit its shape to make it look nicer.
- Test it alone to make sure it is reading the right values, with the bits in the order you expect, etc.