HDL Memory - SRAM

Last updated 1/26/21

HDL Memory - SRAM

- SRAM – synchronous write, asynchronous read

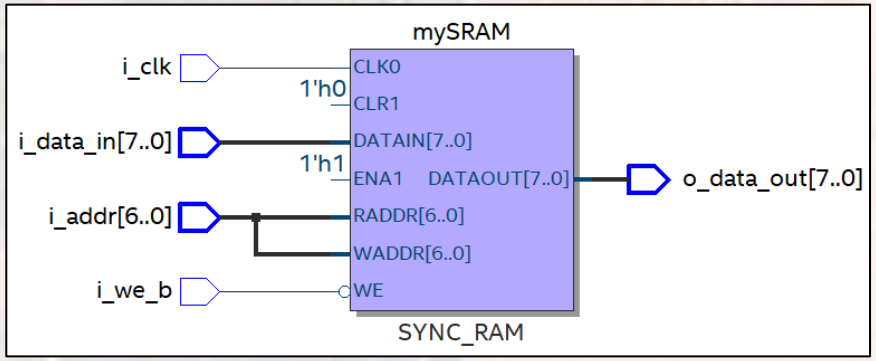
```
----------------------------------------
--
-- sram_128B.vhdl
--
-- created 4/25/17
-- tj
--
-- rev 0
----------------------------------------
--
-- 128B synchronous RAM
--
----------------------------------------
--
-- Inputs:  clk, addr, we_b, data_in
-- Outputs: data_out
--
----------------------------------------
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity sram_128B is
   port(
      i_clk:      in    std_logic;
      i_we_b:     in    std_logic;
      i_addr:     in    std_logic_vector(6 downto 0);
      i_data_in:  in    std_logic_vector(7 downto 0);
      o_data_out: out   std_logic_vector(7 downto 0)
   );
end;
```

```
architecture behavioral of sram_128B is

   --
   -- create type
   --
   type sram_type is array (0 to 127) of std_logic_vector (7 downto 0);
   --
   -- create memory
   --
   signal mySRAM: sram_type;

   begin
      --
      -- Synchronous SRAM write process
      --
      process (i_clk)
      begin
         if (rising_edge(i_clk)) then
            if(i_we_b ='0') then
               mySRAM(to_integer(unsigned(i_addr))) <= i_data_in;
            end if;
         end if;
      end process;

      --
      -- Asyncronous SRAM read
      --
      o_data_out <= mySRAM(to_integer(unsigned(i_addr)));

end behavioral;
```


© tj

HDL Memory - SRAM

- SRAM – synchronous write, asynchronous read



Flow Status	Successful - Tue Jan 28 11:25:58
Quartus Prime Version	18.0.0 Build 614 04/24/2018 SJ
Revision Name	memories
Top-level Entity Name	sram_128B
Family	MAX 10
Device	10M50DAF484C7G
Timing Models	Final
Total logic elements	1,322 / 49,760 (3 %)
Total registers	1024
Total pins	25 / 360 (7 %)
Total virtual pins	0
Total memory bits	0 / 1,677,312 (0 %)
Embedded Multiplier 9-bit elements	0 / 288 (0 %)
Total PLLs	0 / 4 (0 %)
UFM blocks	0 / 1 (0 %)
ADC blocks	0 / 2 (0 %)

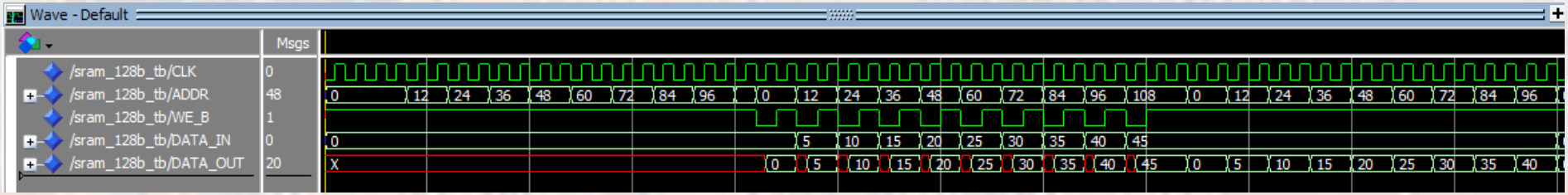
© tj

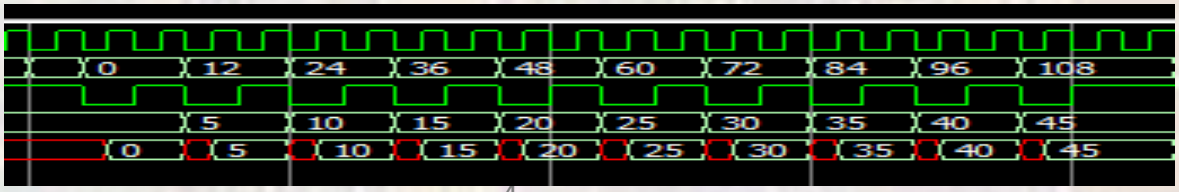
HDL Memory - SRAM

- SRAM – synchronous write, asynchronous read

```
-- Run Process
run: Process      -- note - no sensitivity list allowed
begin
   -- Initalize values
   ADDR <= (others => '0');
   DATA_IN <= (others => '0');
   WE_B <= '1';

   -- Read from a few addresses
   for i in 0 to 9 loop
      wait for 2*PER;
      ADDR <= std_logic_vector(to_unsigned(i*12,7));
   end loop;
```

```
   -- Write to a few addresses
   for i in 0 to 9 loop
      wait for 1*PER;
      ADDR <= std_logic_vector(to_unsigned(i*12,7));
      DATA_IN <= std_logic_vector(to_unsigned(i*5, 8));
      WE_B <= '0';
      wait for 1*PER;
      WE_B <= '1';
   end loop;

      -- Read from a few addresses
   for i in 0 to 9 loop
      wait for 2*PER;
      ADDR <= std_logic_vector(to_unsigned(i*12,7));
   end loop;
end process run;

-------------------------------------------
-- End test processes
-------------------------------------------
```





© tj

HDL Memory - SRAM

- SRAM – synchronous write – generic

© tj

HDL Memory - SRAM

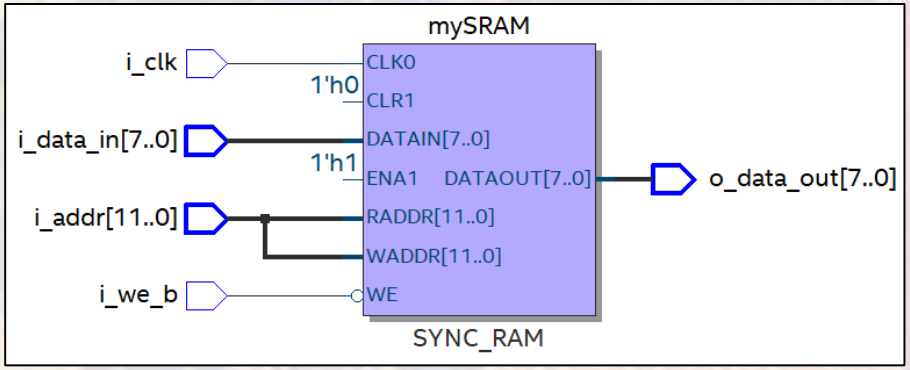
- SRAM – synchronous write - generic

```
----------------------------------------
--
-- sram_MxN.vhdl
--
-- created 4/25/17
-- tj
--
-- rev 0
----------------------------------------
--
-- synchronous RAM with registered address and generics
--
----------------------------------------
--
-- Inputs:  clk, addr, we_b, data_in
-- Outputs: data_out
--
----------------------------------------
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
use ieee.math_real.all;

entity sram_MxN is
   generic(
      mem_width:  positive := 8;
      mem_depth:  positive := 4096
   );
   port(
      i_clk:      in    std_logic;
      i_we_b:     in    std_logic;
      i_addr:     in    std_logic_vector((integer(ceil(log2(real(mem_depth))))) - 1) downto 0);
      i_data_in:  in    std_logic_vector((mem_width - 1) downto 0);
      o_data_out: out   std_logic_vector((mem_width - 1) downto 0)
   );
end;
```

```
architecture behavioral of sram_MxN is

   --
   -- create type
   --
   type sram_type is array (0 to (mem_depth - 1)) of std_logic_vector ((mem_width - 1) downto 0);
   --
   -- create memory
   --
   signal mySRAM: sram_type;

   begin

      --
      -- SRAM write process
      --
      process (i_clk)
      begin
         if (rising_edge(i_clk)) then
            -- read logic
            if(i_we_b ='0') then
               mySRAM(to_integer(unsigned(i_addr))) <= i_data_in;
            end if;
         end if;
      end process;

      --
      -- SRAM asynchronous read
      --
      o_data_out <= mySRAM(to_integer(unsigned(i_addr)));

end behavioral;
```

HDL Memory - SRAM

- SRAM – synchronous write - generic



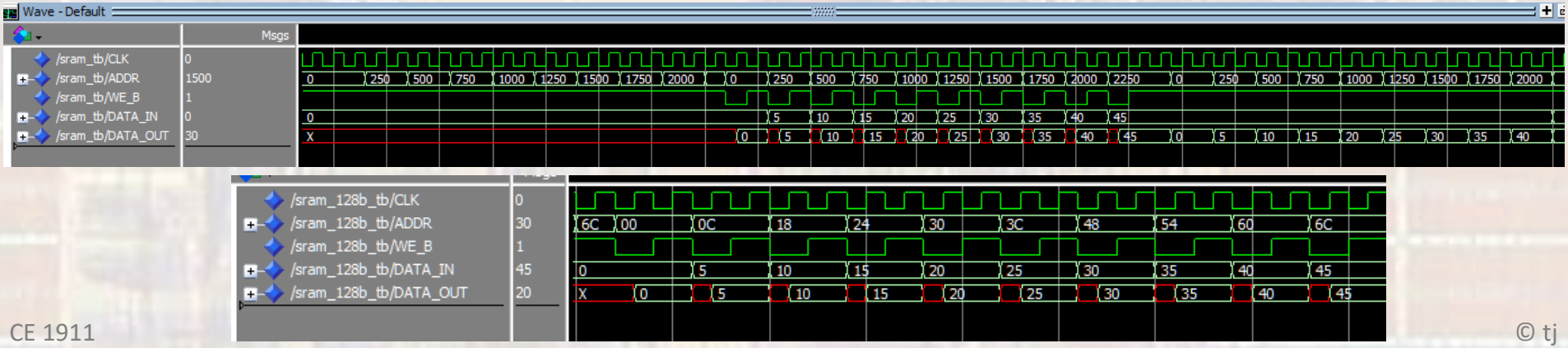
Flow Status	Successful - Tue Jan 28 12:26:3
Quartus Prime Version	18.0.0 Build 614 04/24/2018 S.
Revision Name	memories
Top-level Entity Name	sram_MxN
Family	MAX 10
Device	10M50DAF484C7G
Timing Models	Final
Total logic elements	58,752
Total registers	32768
Total pins	30
Total virtual pins	0
Total memory bits	0
Embedded Multiplier 9-bit elements	0
Total PLLs	0
UFM blocks	0
ADC blocks	0

© tj

HDL Memory - SRAM

- SRAM – synchronous write - generic

```
-- Run Process
run: Process      -- note - no sensitivity list allowed
begin
   -- Initalize values
   ADDR <= (others => '0');
   DATA_IN <= (others => '0');
   WE_B <= '1';

   -- Read from a few addresses
   for i in 0 to 9 loop
      wait for 2*PER;
      ADDR <= std_logic_vector(to_unsigned(i*250,(integer(ceil(log2(real(mem_depth)))))));
   end loop;

   -- Write to a few addresses
   for i in 0 to 9 loop
      wait for 1*PER;
      ADDR <= std_logic_vector(to_unsigned(i*250,(integer(ceil(log2(real(mem_depth)))))));
      DATA_IN <= std_logic_vector(to_unsigned(i*5, mem_width));
      WE_B <= '0';
      wait for 1*PER;
      WE_B <= '1';
   end loop;

      -- Read from a few addresses
   for i in 0 to 9 loop
      wait for 2*PER;
      ADDR <= std_logic_vector(to_unsigned(i*250,(integer(ceil(log2(real(mem_depth)))))));
   end loop;
end process run;
```



© tj

HDL Memory - SRAM

- SRAM – synchronous read/write - generic

Allows the compiler to use on-chip memory blocks

© tj

HDL Memory - SRAM

- SRAM – synchronous read/write - generic

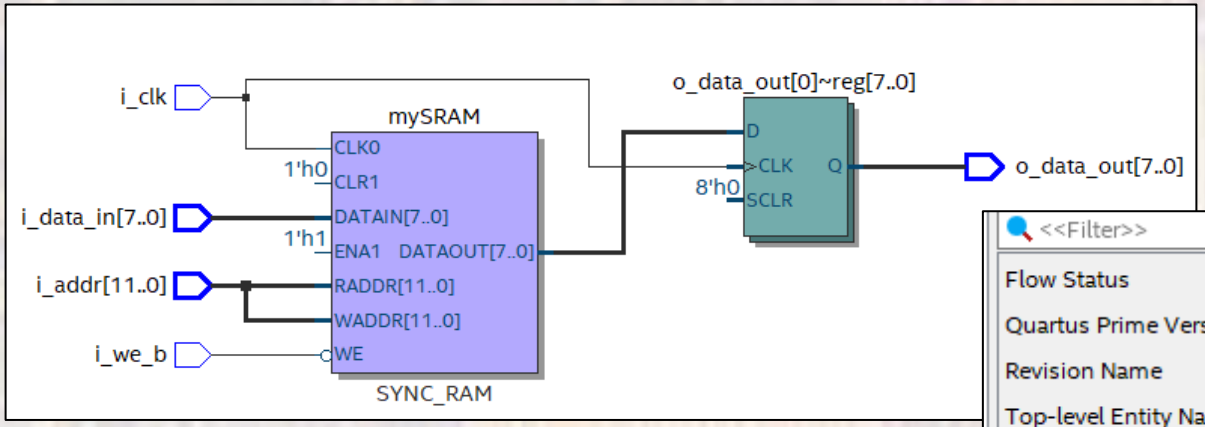
```
----------------------------------------
--
-- sram_sync_rw.vhdl
--
-- created 4/25/17
-- tj
--
-- rev 0
----------------------------------------
--
-- synchronous RAM with synchronous RW
--
----------------------------------------
--
-- Inputs:  clk, addr, we_b, data_in
-- Outputs: data_out
--
----------------------------------------
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
use ieee.math_real.all;

entity sram_sync_rw is
   generic(
      mem_width:  positive := 8;
      mem_depth:  positive := 4096
   );
   port(
      i_clk:      in    std_logic;
      i_we_b:     in    std_logic;
      i_addr:     in    std_logic_vector(((integer(ceil(log2(real(mem_depth))))) - 1) downto 0);
      i_data_in:  in    std_logic_vector((mem_width - 1) downto 0);
      o_data_out: out   std_logic_vector((mem_width - 1) downto 0)
   );
end;
```

```
architecture behavioral of sram_sync_rw is

   --
   -- create type
   --
   type sram_type is array (0 to (mem_depth - 1)) of std_logic_vector ((mem_width - 1) downto 0);
   --
   -- create memory
   --
   signal mySRAM: sram_type;

   begin

      --
      -- SRAM write process
      --
      process (i_clk)
      begin
         if (rising_edge(i_clk)) then
            -- read logic
            if(i_we_b ='0') then
               mySRAM(to_integer(unsigned(i_addr))) <= i_data_in;
            end if;
            --registerd output
            o_data_out <= mySRAM(to_integer(unsigned(i_addr)));
         end if;
      end process;

end behavioral;
```


© tj

HDL Memory - SRAM

- SRAM – synchronous read/write - generic



<<Filter>>

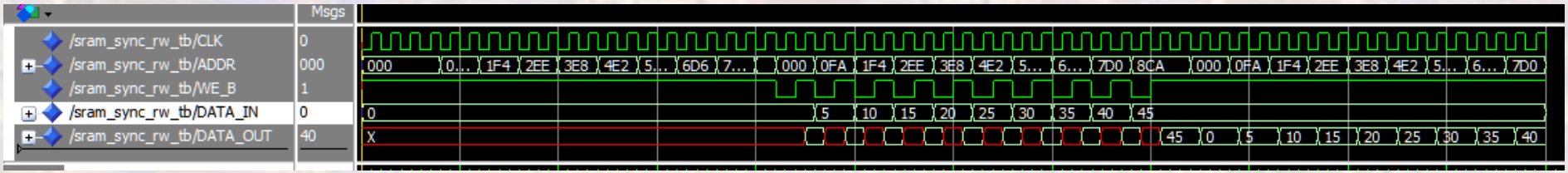
Flow Status	Successful - Tue Jan 28 12:40:50
Quartus Prime Version	18.0.0 Build 614 04/24/2018 SJ
Revision Name	memories
Top-level Entity Name	sram_sync_rw
Family	MAX 10
Device	10M50DAF484C7G
Timing Models	Final
Total logic elements	0
Total registers	0
Total pins	30
Total virtual pins	0
Total memory bits	32,768
Embedded Multiplier 9-bit elements	0
Total PLLs	0
UFM blocks	0
ADC blocks	0

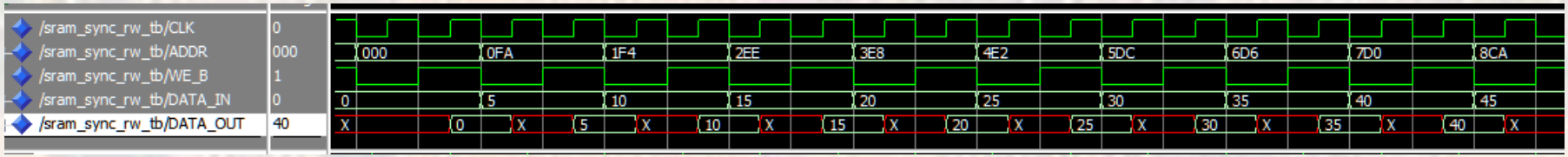
30

© tj

HDL Memory - SRAM

- SRAM – synchronous read/write - generic





 © tj

HDL Memory - SRAM

- SRAM – synchronous read/write – registered address - generic

HDL Memory - SRAM

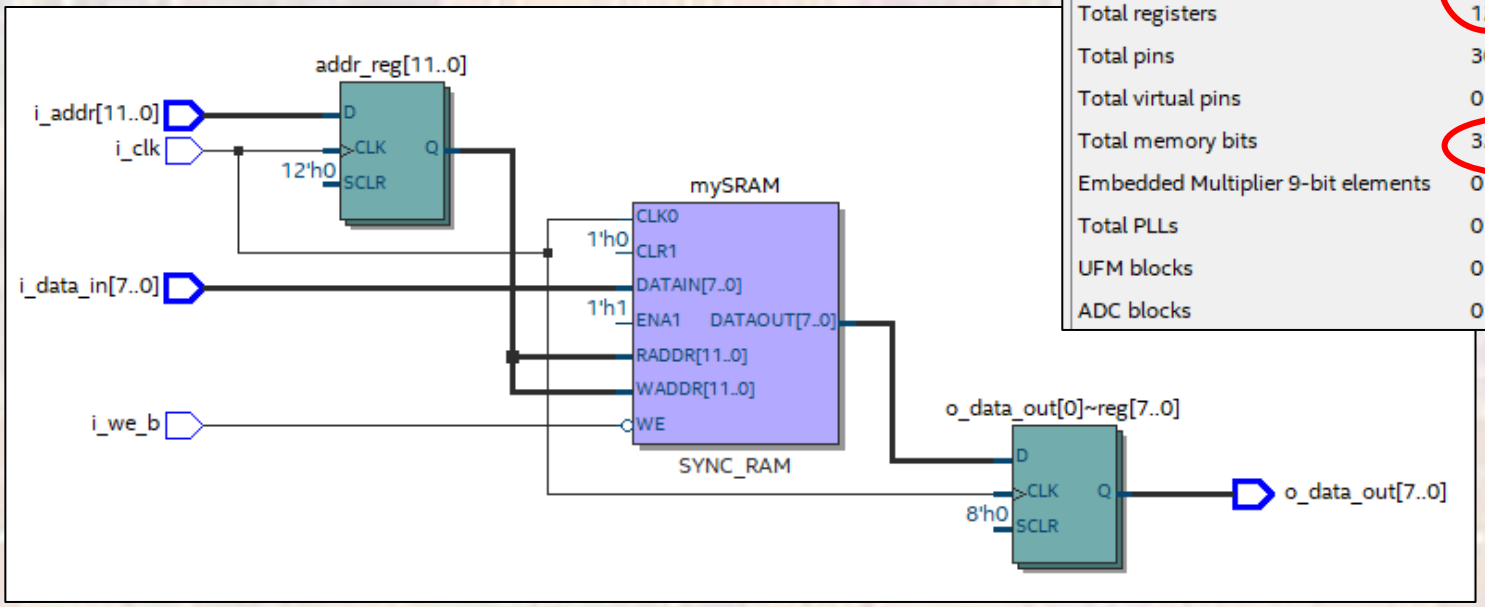
- SRAM – synchronous read/write – registered address - generic

```
architecture behavioral of sram_reg_addr is

   --
   -- create type
   --
   type sram_type is array (0 to (mem_depth - 1)) of std_logic_vector ((mem_width - 1) downto 0);
   --
   -- create memory
   --
   signal mySRAM: sram_type;
   --
   -- create registered address signal
   --
   signal addr_reg: std_logic_vector(((integer(ceil(log2(real(mem_depth))))) - 1) downto 0);


   begin

      --
      -- SRAM write process
      --
      process (i_clk)
      begin
         if (rising_edge(i_clk)) then
            -- read logic
            if(i_we_b ='0') then
               mySRAM(to_integer(unsigned(addr_reg))) <= i_data_in;
            end if;
            --address register
            o_data_out <= mySRAM(to_integer(unsigned(addr_reg)));
            addr_reg <= i_addr;
         end if;
      end process;

end behavioral;
```

```
-----------------------------------------
--
-- sram_reg_addr.vhdl
--
-- created 4/25/17
-- tj
--
-- rev 0
-----------------------------------------
--
-- synchronous RAM with registered address
--
-----------------------------------------
--
-- Inputs:  clk, addr, we_b, data_in
-- Outputs: data_out
--
-----------------------------------------
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
use ieee.math_real.all;

entity sram_reg_addr is
   generic(
      mem_width:  positive := 8;
      mem_depth:  positive := 4096
   );
   port(
      i_clk:      in    std_logic;
      i_we_b:     in    std_logic;
      i_addr:     in    std_logic_vector(((integer(ceil(log2(real(mem_depth))))) - 1) downto 0);
      i_data_in:  in    std_logic_vector((mem_width - 1) downto 0);
      o_data_out: out   std_logic_vector((mem_width - 1) downto 0)
   );

end;
```

HDL Memory - SRAM

- SRAM – synchronous read/write – registered address - generic



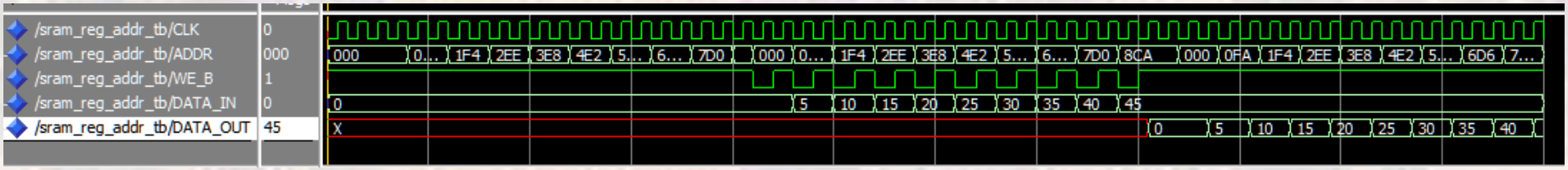
<<Filter>>

Flow Status	Successful - Tue Jan 28 13:07:24 2
Quartus Prime Version	18.0.0 Build 614 04/24/2018 SJ Lit
Revision Name	memories
Top-level Entity Name	sram_reg_addr
Family	MAX 10
Device	10M50DAF484C7G
Timing Models	Final
Total logic elements	12
Total registers	12
Total pins	30
Total virtual pins	0
Total memory bits	32,768
Embedded Multiplier 9-bit elements	0
Total PLLs	0
UFM blocks	0
ADC blocks	0

© tj

HDL Memory - SRAM

- SRAM – synchronous read/write – registered address - generic



 © tj