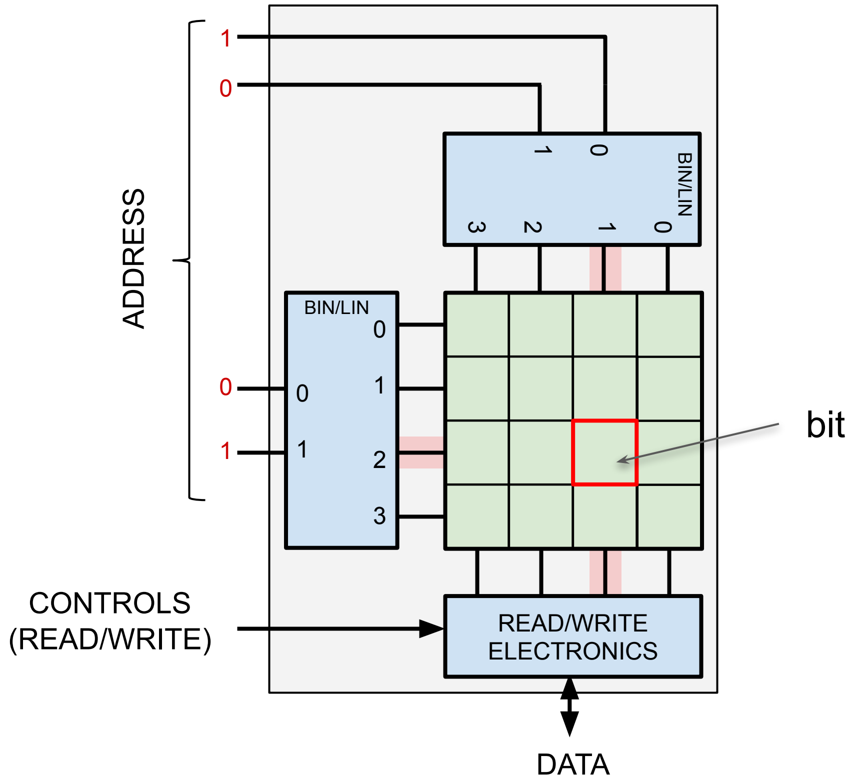
Digital Techniques 2

L12: Semiconductor Memories

UNIVERSITY OF OULU
CIRCUITS and SYSTEMS

Semiconductor Memory Basics

- Memory chips, or memories embedded in ASICs or FPGAs, store data in addressable storage elements organized in array format
- Compared to flip-flop-based registers, memory storage is cheap: one-bit memory cell can be built with 1-6 transistors, compared to > 30 for a flip-flop
- Because of address and R/W logic overhead, small embedded memories are not area-efficient
- Memory is slow: Compared to flip-flops, their access time is long, and only one or few memory locations can be accessed at a time



Memory Chip Functions Overview

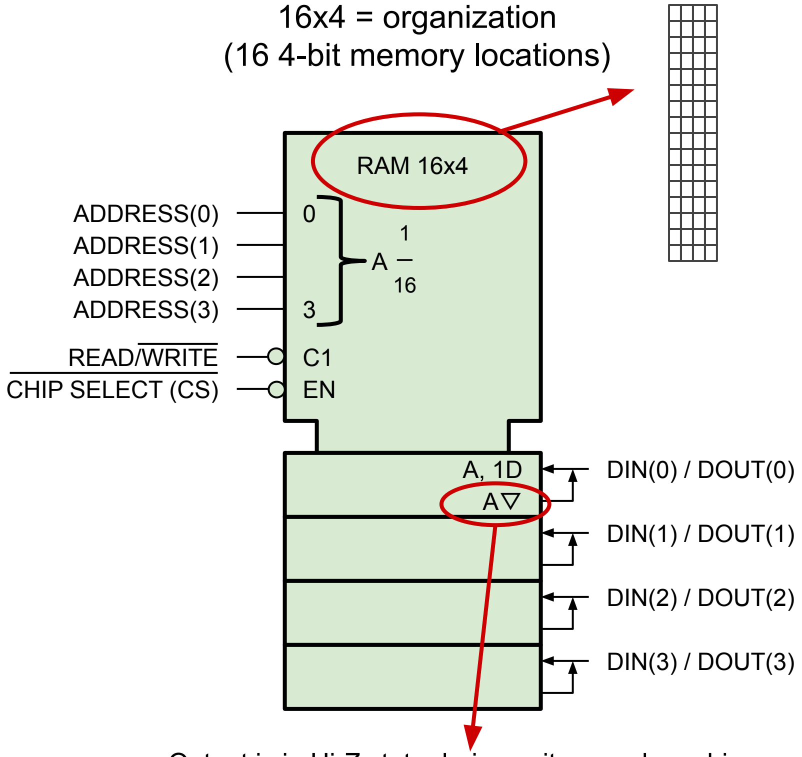
Types of memory:

- Read-Only (ROM)
- Read/Write (RAM, random access memory)

Input and output pins:

- Address inputs
- Data inputs/outputs, (bidirectional)
- Read-write select: level sensitive write clock in RAMs (WE)
- Chip select (CS) input(s)
- Output enable (OE) input

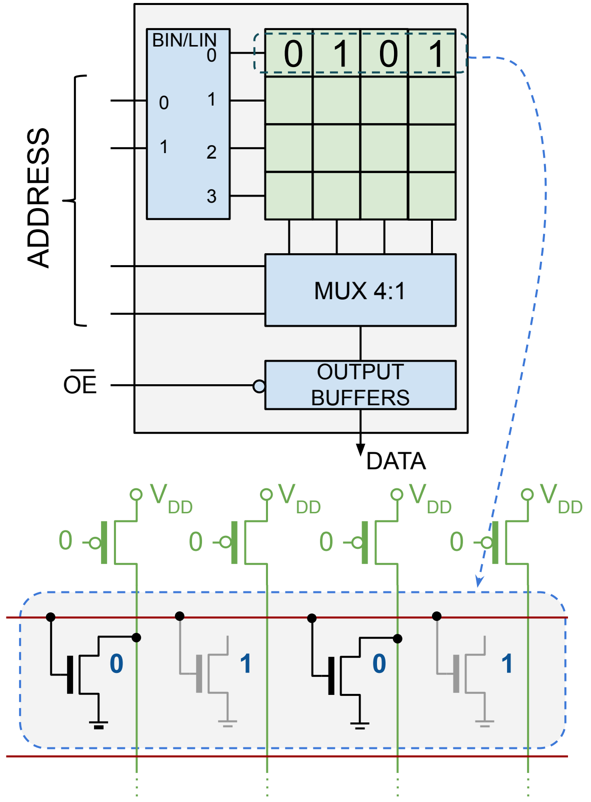
16x4 = organization
(16 4-bit memory locations)



Output is in Hi-Z state during writes or when chip has not been selected.

Mask-Programmable ROMs

- Discrete components that contain a fixed ROM array, or embedded ROMs on ASICs
- One bit is represented by a pull-down transistor that is connected to a bit-line (0) or left unconnected (1)
- "Programming" is done by creating the IC metal layer wires that form bit-line connections (masks are used to define wires in a photolithographic manufacturing process)
- Discrete ROM chips are used as pattern/function generators, earlier as embedded computer program memory (replaced by **Electrically Erasable Programmable ROM** or **FLASH**)
- Embedded ROMs on ASICs have many uses (e.g. boot ROM for embedded processor cores, look-up tables)

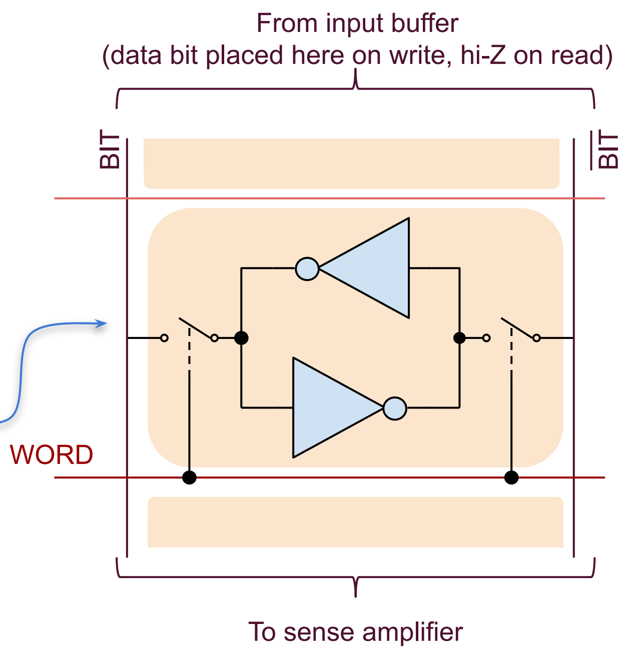


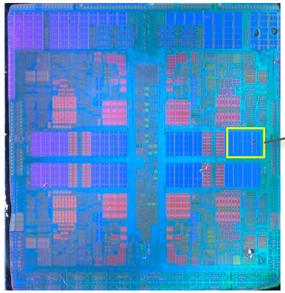
UNIVERSITY OF OULU
CIRCUITS and SYSTEMS

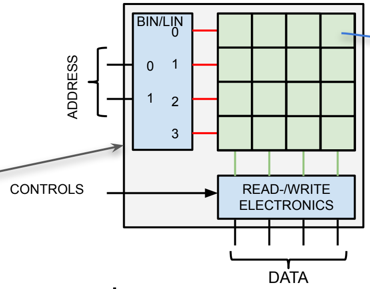
Static RAM

- Data bits are stored in 6-transistor "RS latch" cells (two inverters + two switches)
- SRAM holds data as long as power is on (=static)
- Used as special high-speed memory in electronic equipment
- FPGAs have large amounts of SRAM as *block RAMs* and logic look-up tables that can be configured as *distributed RAM*
- SRAM is also a very common embedded memory on system-on-a-chip ICs

SRAM Memory Cell (One Bit)

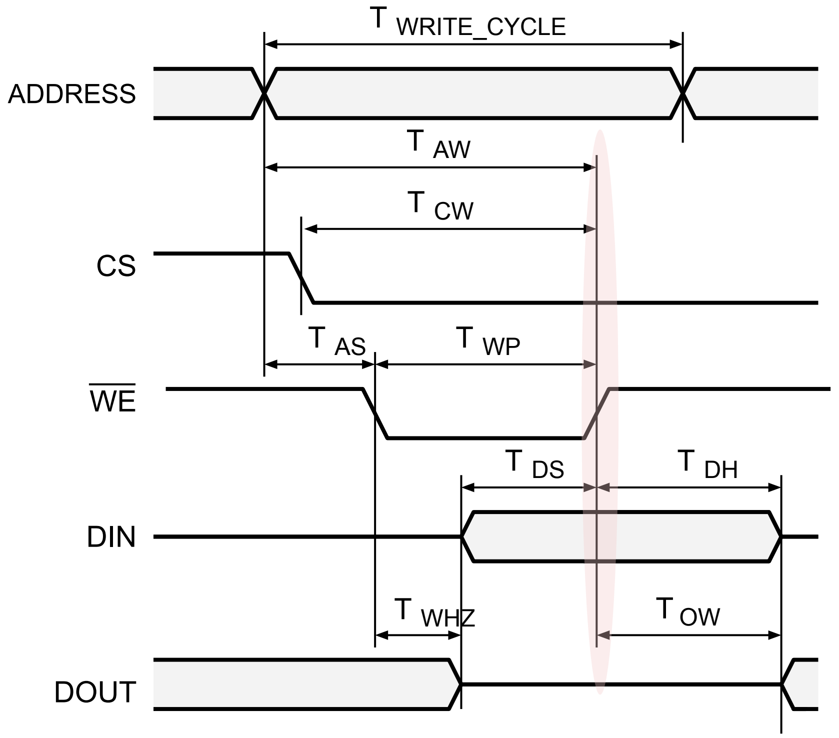



AMD Barcelona Chip (Source)



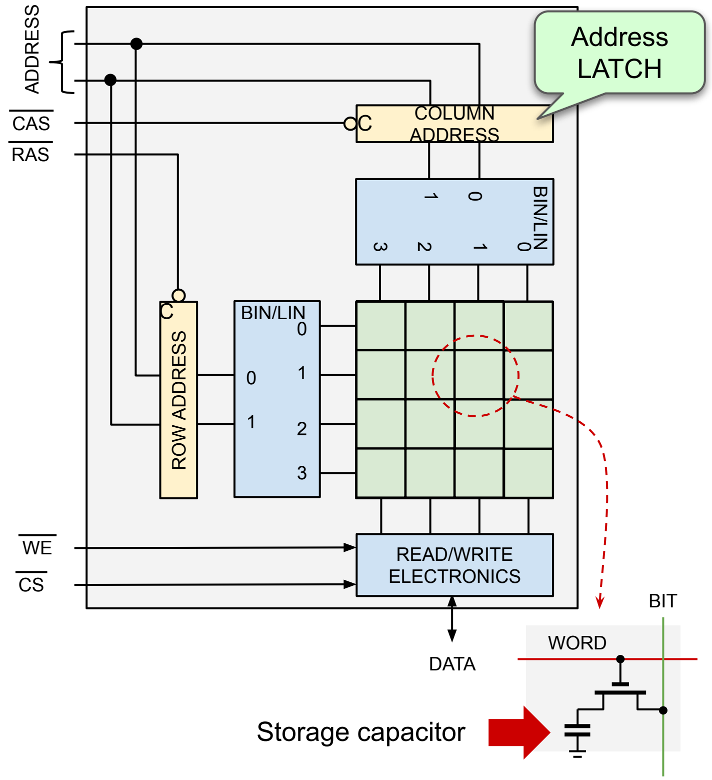
SRAM Timing

- Basic SRAMs are asynchronous by nature
- Write cycle (shown on the right):
  - Data is stored in memory on "back edge" of level-sensitive negative write pulse
  - Many timing constraints relative to "write edge"
- Read cycle
  - Access time form ADDRESS or CS change to DOUT is the most critical parameter
- To adapt SRAM timing requirements to system clock frequency, a **memory controller** must be designed to generate "wait states" etc
- In FPGA SRAMs, synchronous configurations with clocked input/output registers are usually used



Dynamic RAM

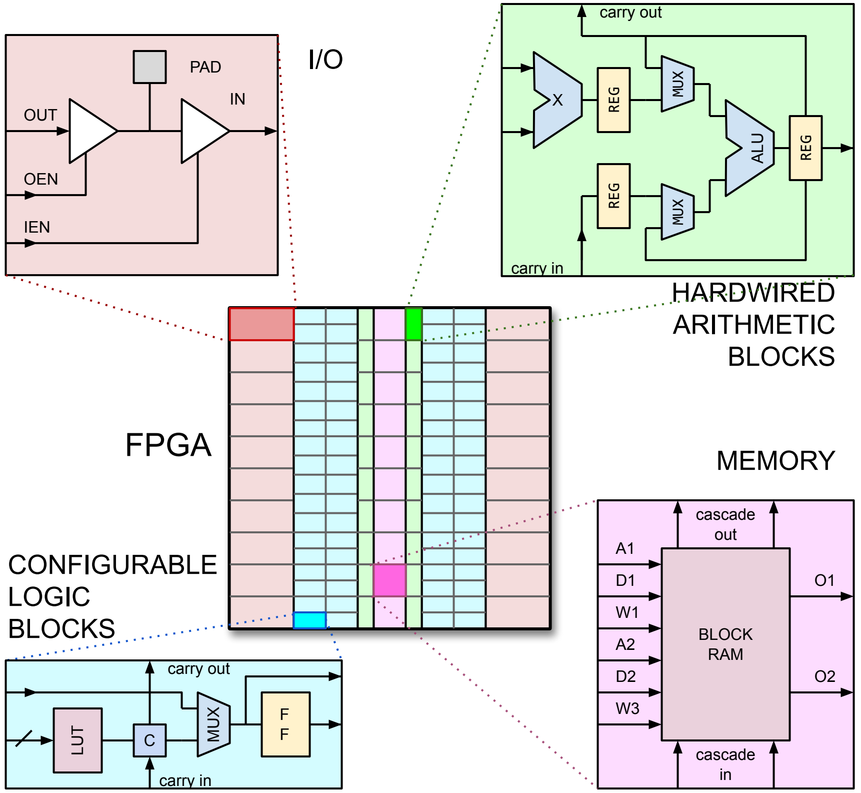
- Data stored as charge in small capacitors (1 transistor per bit => very dense compared to SRAM)
- Because of large number of memory locations, DRAM chips' column and row **address bits are multiplexed** to save pins
- Memory cells must be constantly **refreshed** by reading and writing to prevent data loss because of capacitor discharge (= dynamic)
- **Slow but cheap** ⇒ used as main memory of computers
- Seldom used as embedded RAM on ASICs because of special manufacturing process requirements
- The drawing shows a classical basic DRAM
- Moderns DRAMs are more complex (e.g. **synchronous DRAM** with integrated refresh and burst-mode logic)



UNIVERSITY OF OULU
CIRCUITS and SYSTEMS

FPGA Memory Resources

- FPGA chips typically have a large amount of configurable SRAM resources
- **Block RAM** resources are actual large memory blocks (kilobits/block) in the chip's layout
- Configurable logic blocks contain small look-up table (LUT) SRAMs that are used to implement combinational logic, but can also be configured to function as **distributed RAM**
- RAMs can be directly instantiated in RTL code, but typically they are *inferred* from code that follows certain coding style rules



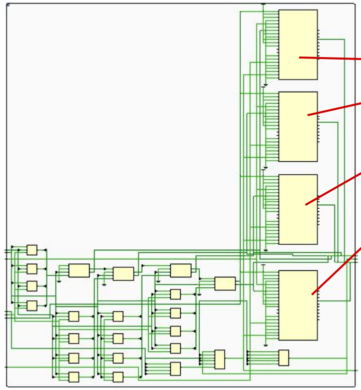
FPGA Memory Inference* Examples

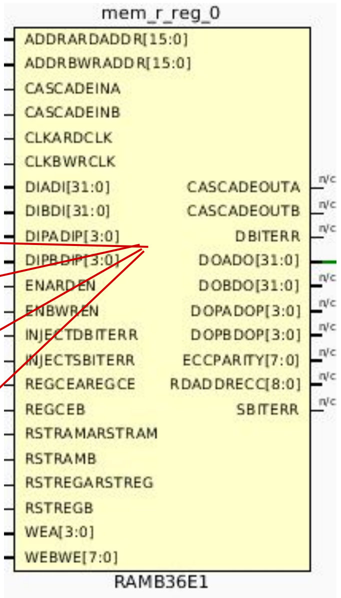
```
module mem (
   input logic         clk,
   input logic         we_in,
   logic [12:0]        addr_in,
   input logic [15:0]  data_in,
   output logic [15:0] data_out
   );

 (* ram_style = "block" *) logic [15:0] mem_r [8192];
 logic [15:0]                           rdata_r;

 always_ff @(posedge clk)
   if (we_in)
      mem_r[addr_in] <= data_in;

 always_ff @(posedge clk)
   rdata_r <= mem_r[addr_in];

   assign data_out = rdata_r;
endmodule
```

SystemVerilog attribute is used to control memory inferencing.

Conventional 8192 element array with 16-bit bit-packed elements

mem_r implemented with 4 block RAM instances



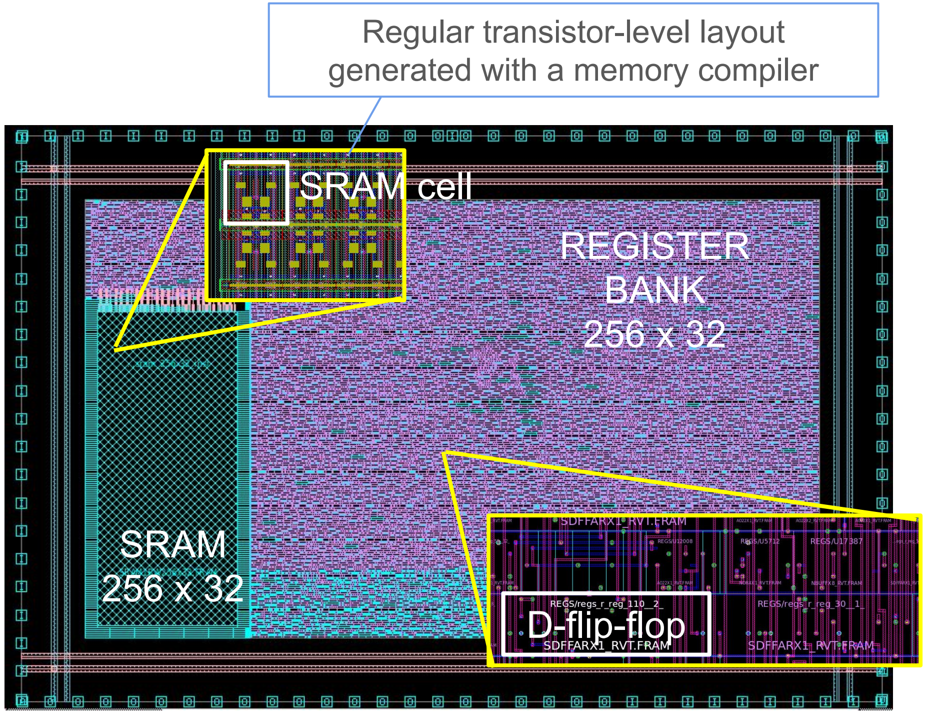


```
+-------------------+------+-------+-----------+-------+
|     Site Type     | Used | Fixed | Available | Util% |
+-------------------+------+-------+-----------+-------+
| Block RAM Tile    |    4 |     0 |       365 |  1.10 |
|   RAMB36/FIFO*    |    4 |     0 |       365 |  1.10 |
|     RAMB36E1 only |    4 |       |           |       |
|   RAMB18          |    0 |     0 |       730 |  0.00 |
+-------------------+------+-------+-----------+-------+
```

*Inference = SRAM is synthesized from code just like registers.

ASIC Memory Resources

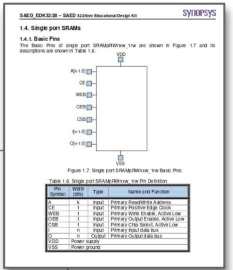
- Because of their area-efficiency, SRAMs are commonly used in SoC designs instead of flip-flop-based register banks
- A custom, transistor-level layout must be designed for every (different) SRAM block in the design using a **memory compiler** program obtained from the silicon vendor
- In RTL code, the SRAM's HDL model is instantiated for simulation
- In synthesis, SRAM instance is mapped to a library part just like gates and flip-flops
- In layout design, SRAM block is usually placed manually outside automatically-placed standard cell areas



Silicon area of equal size SRAM block and register-bank, and comparison of SRAM cell and flip-flop areas.

SRAM Instantiation in ASIC Design

- SRAM model must exist in a **logic library** (sram.db)
- Direct instantiation of an SRAM component is made in the RTL code
- In the synthesis program, the library containing the SRAM is defined as a **link library**
- In elaboration (read_file), the synthesis programs searches for modules that have not been defined in the RTL code from the link libraries
- In optimization (compile_ultra), the SRAMs timing model is used for timing analysis



SAED_EDK32/28 – SAED 32/28nm Educational Design Kit

synopsys

1.4. Single port SRAMs

1.4.1. Basic Pins

The Basic Pins of single port SRAMpRWnxw_1rw are shown in Figure 1.7 and its descriptions are shown in Table 1.6.

VDD
A[k-1:0]
CE
WEB
OEB
CSB
I[n-1:0]
O[n-1:0]
VSS

Figure 1.7. Single port SRAMpRWnxw_1rw Basic Pins

Table 1.6. Single port SRAMpRWnxw_1rw Pin Definition

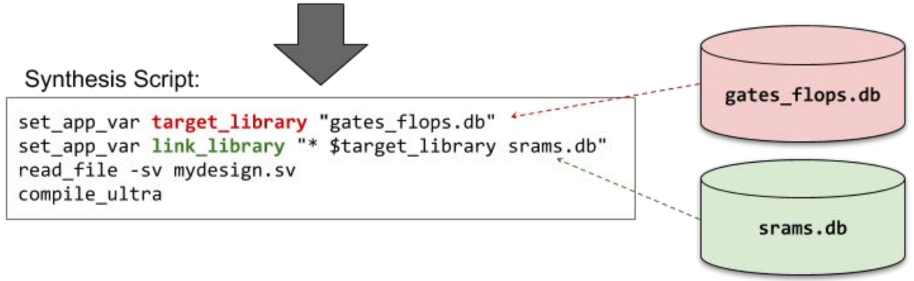
Pin Symbol	Width (bits)	Type	Name and Function
A	k	Input	Primary Read/Write Address
CE	1	Input	Primary Positive-Edge Clock
WEB	1	Input	Primary Write Enable, Active Low
OEB	1	Input	Primary Output Enable, Active Low
CSB	1	Input	Primary Chip Select, Active Low
I	n	Input	Primary Input data bus
O	n	Output	Primary Output data bus
VDD	Power supply		
VSS	Power ground		

mydesign.sv (RTL code)

```
SRAM1RW256x32 sram_256x32_inst
      (.A(addr),
      .I(data_in),
      .WEB(!we_in),
      .CSB(1'b0),
      .OEB(we_in),
      .CE(clk),
      .O(data_out));
```

Synthesis Script:

```
set_app_var target_library "gates_flops.db"
set_app_var link_library "* $target_library srams.db"
read_file -sv mydesign.sv
compile_ultra
```



References

1. Ashenden (2008) Digital Design: An Embedded Systems Approach using Verilog
   Ch. 5 Memories
2. Xilinx 7 Series FPGAs Memory Resources