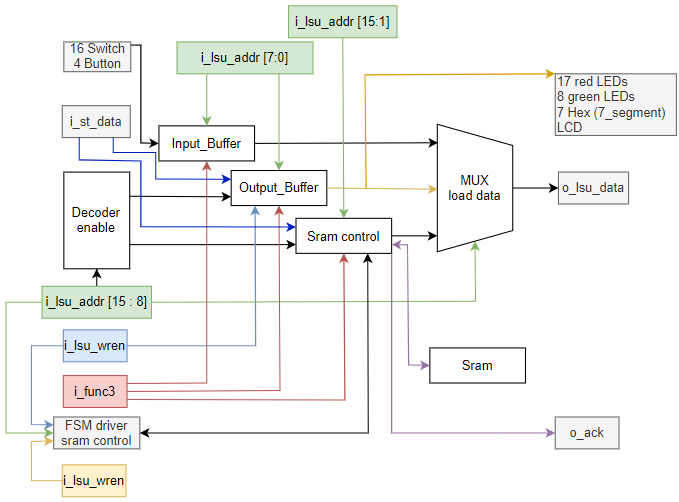
2.5. I/O System and Memory:

In this part, we implement a Load-Store Unit (LSU) which functions as an I/O System and Memory. The LSU includes 5 main components: Decoder, Input\_Buffer, Output\_Buffer and Data\_Memory that present in Figure 1.

Figure 1: LSU Diagram.



LSU’s address is range 0x2000 to 0x781F, which represented in 16 bits, but i\_lsu\_data represented in 32 bits (4 byte). We only use 2 byte low of LSU address and don’t care the other.

DECODER: base on i\_lsu\_adder [15 : 8] (second byte) to select the store region (Output\_buffer and Data\_Memory).

MUX: base on i\_lsu\_adder [15 : 8] (second byte) to select the load region.

\*I/O System

The proccessor interacts with environment through Input\_Buffer and Output\_Buffer.

Input\_Buffer and Output\_Buffer is implemented as 8-bit register array (similar to Instruction Memory).

Input\_Buffer: contains value of Switch and Button Signal. When we interact with Switch and Button that reflected in the region of Input\_Buffer (the address of register). It’s address is in range 0x7800 to 0x781F. The Input\_Buffer is implemented as 32 x 8 bit-registers (32 byte). Because we use the second byte to select region of LSU, we just use the first byte as address for interact with Input\_Buffer. We can’t store data in Input\_Buffer, only load the data from Input\_Buffer.

Output\_Buffer: is connected to the red LEDs, green LEDs, 7-segment displays and LCD1602. It’s address is in range 0x7000 to 0x703F. The Output\_Buffer is implemented as 64 x 8 bit-registers (64 byte). Similar to Input\_Buffer, we just use the first byte as address for interact with Out\_Buffer. We can load or store data in Output\_Buffer.

\*Memory

The Data\_Memory is used to store data and load it out when needed. It’s address is in range 0x2000 to 0x3FFF. In this implementation, we use sram IS61WV25616 replace for Data\_Memory.

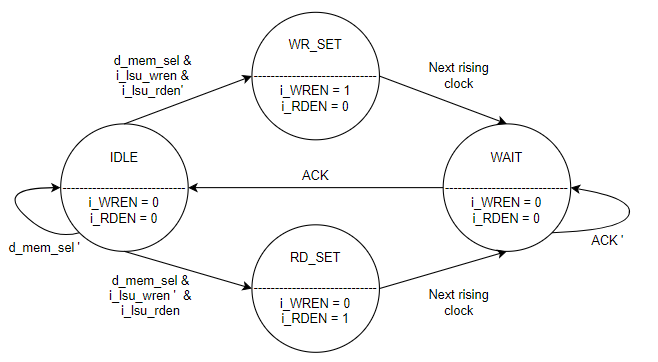
We use the sram\_control module which is provided.

According to the datasheet, this sram is 2 bytes data width corresponding to an address. We need to adjust the LSU’s address that fit to sram’s address. Solution is shift right LSU’s address one bit, we have sram’s address. Because one LSU’s address can be stored address, but one sram’s address can be stored two byte data. So the LSU’s address is “double” than the sram’s address.

Sram has 18 bits address width, we just use 15 bit low to control sram. So 3 bit high set to zero.

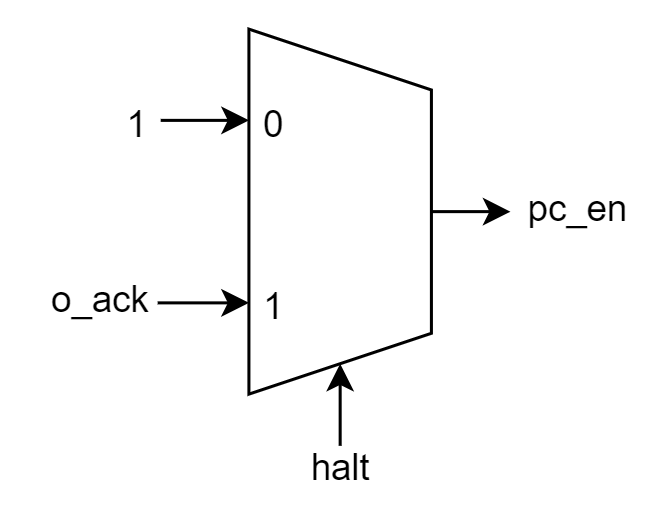
To driver this module, we design a combination logic for driver BMASK and store data by func\_3 (sw, sb, sh, lw, lb, lbu, lh, lhu) and a ASM for driver WREN and RDEN that is present in Figure 2.

Figure 2: FSM Driver Sram Control.



Because the operation delay 2 cycle for write and 5 cycle for read, we need to hold the PC value, wait for the operation completely. Solution is design a combinational logic that can stop PC update next value after each cycle which present in Figure 3.

Figure 3: Combinational logic of control program counter.



The halt signal is active when meet these condition: Opcode is of R-format layout (Store) or I-format load and i\_lsu\_adder in Data\_Memory’s address range.

The ack signal is feelback signal of sram control module. When it’s active means the operation completely and allow PC update next value.

2.5.1 Specification

|  |  |  |  |
| --- | --- | --- | --- |
| **Signal** | **Width** | **Direction** | **Description** |
| i\_clk  i\_rst\_n  i\_lsu\_addr  i\_st\_data  i\_lsu\_wren  i\_func\_3  o\_ld\_data  o\_io\_ledr  o\_io\_ledg  o\_io\_hex0…7  o\_io\_lcd  i\_io\_sw  i\_io\_btn  o\_ack  SRAM\_ADDR  SRAM\_DQ  SRAM\_CE\_N  SRAM\_WE\_N  SRAM\_LB\_N  SRAM\_UB\_N  SRAM\_OE\_N | 1  1  32  32  1  3  32  32  32  7  32  32  4  1  18  16  1  1  1  1  1 | input  input  input  input  input  input  output  output  output  output  output  input  input  output  output  inout  output  output  output  output  output | Globa clock, active on the rising edge.  Globa low active reset.  Address for data read or write.  Data to be store.  Write enable signal (1 if writting).  Select function sw, sb, sh, lw, lb, lbu, lh, lhu.  Data read from memory.  Output for red LEDs.  Output for green LEDs.  Output for 7-segment displays.  Output for the LCD register.  Input for switch.  Input for button.  Signal to pc\_halt\_by\_sram  SRAM address  SRAM data bus  SRAM chip enable (active-low)  SRAM write enable (active-low)  SRAM output enable (active-low)  SRAM lower byte enable (active-low)  SRAM upper byte enable (active-low) |

3.4. I/O System and Memory: