Proceedings of the 5th International Conference on Computing and Data Science
DOI: 10.54254/2755-2721/13/20230748

Design of asynchronous FIFO with adjustable depth based on Verilog

Xiangjie Li [1, 4,†], Shimeng Fu[2, †],Xuanbo Wang[2, †] and Yinqiu Wang[3, †],

[1] Shenzhen University, Shenzhen, China
[2] Xi'an University of Technology, Xi'an, China
[3] Jiangsu Normal University, Xuzhou, China

[4]lixiangjie2020@email.szu.edu.cn
[†]All authors' contributions are consistent

Abstract. The integrated circuit industry is the basic industry of the digital economy and one of the basic sources of strength to build the new competitive advantage of China's economy in the future. With the development of microelectronic technology, IC technology has becoming more and more integrated, at the same time, the data flow in the different chips are getting complex. To overcome the restriction of the data transmission, FIFO become a way to solve the problem. However, different chips usually work in various clock domains, so the data transmission across clock domains is the key to make the communication between different chips efficient. Meanwhile, the rate of space usage is also a part of chip performance, so the depth of the FIFO is important. Because of the above, an asynchronous FIFO with adjustable depth is designed on Verilog, used Quartus II and Modelsim to draw an RTL diagram of the whole module and simulation.

Keyword: FIFO, Verilog, depth adjustment.

1. Introduction

FIFO, which means first in and first out, is a dual port data register, the difference between FIFO and ordinary buffer is that there is no external address line, the advantage is to reduce the input signal control line, the disadvantage is that the data can only be written in sequence, sequence read [1]. To solve the problem of different working clock domains, synchronization of the output is necessary, the module of synchronization allow data input from other modules to be valid. The whole asynchronous FIFO can be divided into six modules: Module 1—write_operate: Write enable controls whether to write, the input read pointer is processed to determine whether the FIFO is full then generate the full signal. When FIFO is full, stop writing. Module 2—read_operate: Read enable controls whether to read, the input write pointer is processed to determine whether the FIFO is empty then generate the empty signal. When FIFO is empty, stop reading. Module 3—write_syn: Make the read pointer in read clock domains into write clock domains. The use of two stage D triggers to reduce the occurrence of metastability. Module 4—read_syn: Make the write pointer in write clock domains into read clock domains. The use of two stage D triggers to reduce the occurrence of metastability. Module 5—FIFO_data: Control the data location based on the write or read pointer. Top module: When using the whole module, the parameter can be changed, which can achieve the purpose of adjusting the depth [2].

DOI: 10.54254/2755-2721/13/20230748

2. Major module principle

2.1. Conversion principle of binary code and gray code

Because of the transmission of data in different module, if the data change many bits, metastability may happen. However, gray code is a binary code that changes one bit at a time, which means that even there is a two stage D triggers, gray can make the address transmission without problem. Binary to Gray: The binary code is moved one bit to the right and then XOR with the original binary code [3]. Gray to Binary: The highest bit of the gray code is reserved as the highest bit of the binary code. The binary code secondary high level generation process is obtained by using the binary high level and the secondary high level gray code to be different, and the value of other bits is like the binary code secondary high level generation process. As shown in Table 1.

Table 1. Code value conversion table.

Code value conversion table		
Decimal(N)	Binary(B3B2B1B0)	Gray(G3G2G1G0)
0	0 0 0 0	0 0 0 0
1	0 0 0 1	0 0 0 1
2	0 0 1 0	0 0 1 1
3	0 0 1 1	0 0 1 0
4	0 1 0 0	0 1 1 0
5	0 1 0 1	0 1 1 1
6	0 1 1 0	0 1 0 1
7	0 1 1 1	0 1 0 0
8	1 0 0 0	1 1 0 0
9	1 0 0 1	1 1 0 1
10	1 0 1 0	1 1 1 1
11	1 0 1 1	1 1 1 0
12	1 1 0 0	1 0 1 0
13	1 1 0 1	1 0 1 1
14	1 1 1 0	1 0 0 1
15	1 1 1 1	1 0 0 0
N	$B_n = G_n / B_{n-1} = B_n \oplus G_{n-1}$	$G = B \oplus (B \gg 1)$

2.2. Empty and full condition judgment mechanism

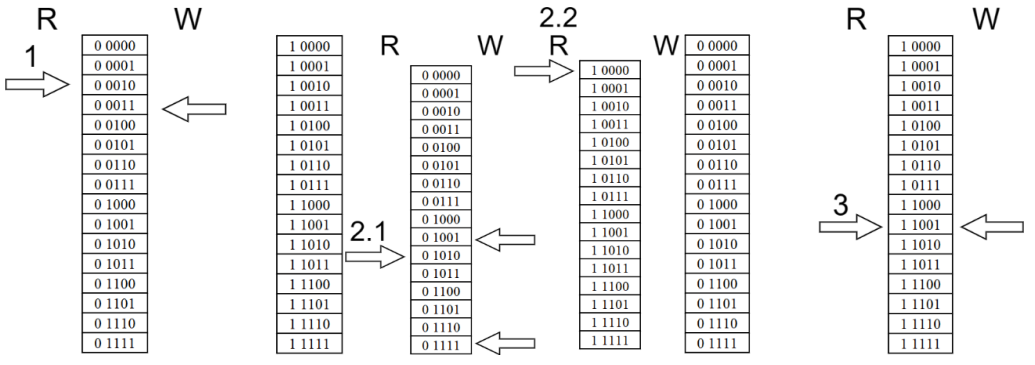

Figure 1. Chart for full and empty.

Compare the position of the write pointer and the read pointer to determine whether it is full or empty. The $depth + 1$ bits pointer is used for the compare, As shown in Figure 1. which means that there is

DOI: 10.54254/2755-2721/13/20230748

an extra bit pointer to compare the order of write pointer and read pointer. However, one thing to note is that only depth to 0 bits is an address, the $depth + 1$ bit of the pointer is the flag of the round, it can't represent any address. No full and no empty: In this situation the position of write pointer and read pointer don't satisfy any circumstance, so the FIFO isn't full or empty. Full: This situation is divided into two circumstances. The first one, the two pointer is in the same round, and write pointer is locating the bits before read pointer. The second one, the two pointer isn't in the same round, but the write pointer is at the end of that round, and read pointer is the first one in its round. Empty: In this situation the two pointer is in the same round, and they have the same position [4]. So, the judgment condition of full and empty is clear: when the write address is 1 less than read address, FIFO is full, when the write address is same as read address. When FIFO is full, write operate stop; when FIFO is empty, read operate stop.

2.3. Signal synchronization in different clock domains

Because of the development of IC, more flip-flop is used as a normal device in the design of the logic circuit. It must need data setup and hold time during the use of flip-flop (setup time is the minimum time required for the data on the flip-flop data side to be stable, hold time is the minimum time for data-on-data side stay stable after the rising edge from clock coming. This design adopts double latch method. Two D-triggers are connected, when signal is transmitted between two different clock domains, the signal will pass the two-stage latch, and make the signal stable. This kind of structure can reduce the probability of metastable state and reduce the harm of metastable state to a certain extent. However, this kind of structure will cause the delay of a clock in the input signal, which needs to be dealt with in the design of the system and must be paid attention.

3. Relevant principles of adjustable depth module

To realize the continuous reduction of chip size and the increase of space utilization, the adjustable depth is necessary, the module could select the best depth according to the input read/write clock to achieve high space utilization.

3.1. Minimum depth estimation

In the write operate, data isn't entered smoothly, there may be a data burst, which means that there are suddenly appear a large number of data will be written. To avoid data lost, FIFO is best to choose a depth that can accommodate the data [5,6]. First, the frequency of write clock (replace with wclk) and read clock (replace with rclk), and need to know the maximum burst each time. When frequency of write less than frequency of read, there will be no accumulation of data, every time the data write into FIFO, it can be read, so depth can be set at will. When frequency of write more than frequency of read, after each burst, data accumulates in the FIFO, then the known condition to calculate the minimum depth. The basic calculation process is explained below: Known: wclk(Hz), rclk(Hz), burst. Write burst need time: $t = burst/wclk$ The number of read int: $cont = rclk \times t$ The minimum depth: $depth_{min} = (\text{burst} — \text{cont}) + 1$ To sum up:. $depth_{min} = burst \times (1 — \frac{r_{clk}}{w_{clk}}) + 1$ The above is the basic case for calculating the minimum depth of the FIFO. Considering the delay in the gray code conversion, the minimum depth must be greater than $depth_{min}$.

3.2. More key factors

First, when there is an idle cycle, we can understand that the writing frequency and reading frequency decrease, so the equivalent data writing time and the equivalent reading data time are as follows: Known: wclk(Hz), rclk(Hz),idle cycle. Write data equivalent time: $(idle\ cycle + 1) \times (1/wclk)$. Read data equivalent time: $(\text{idle cycle} + 1) \times (1/\text{wclk})$. The experimental data show a significant increase in the presence of idle cycles and this calculation is also applicable to the case where the read and write enable signal (wr_en/re_en) has a duty cycle. For example, when idle cycle = 1, the corresponding duty cycle is 50%. Second, there is a random read and write situation, and we start with

DOI: 10.54254/2755-2721/13/20230748

the situation that has the greatest impact on the depth of the FIFO, when the write frequency is the largest, and the read frequency is the slowest [7]. we could use the known condition to calculate the minimum depth. The basic calculation process is explained below: Known: wclk (Hz), rclk (Hz), random read data frequency (data/clock), burst. Write burst need time: $t = burst/wclk$,Read data frequency:rclk $rclk' = 1/[1/(rclk \times random\ read\ \ data\ frequency)]$, The number of read in t: $cont = rclk' \times$ t, The minimum depth: $depth_{min} =$ (burst— cont) + 1.

4. Code implementation on Verilog

4.1. Write operate and read operate

The main steps of read and write operations are similar: In write operations, write addresses and write Pointers are generated to generate full signals; Read operation, generate read address, read pointer, generate empty signal [8].

4.2. Write_syn and read_syn

In write clock synchronization, the address output under the read clock is input to the write clock domain by beating two beats to prevent the metastable data caused by the address change from affecting the write and read of the data. The same is true for read clock synchronization.

4.3. FIFO data

Set the data depth as 4 bits of binary number (that is, 16 bits of data), and the data width as 8 bits of binary number and fill the data into the FIFO successively under the write clock [9]; Data can be written only when it is not full and the write switch is on. Read data through the read pointer address control, pointer to the address, then output data in this address.

4.4. Top module

By using these five modules to realize the FIFO function, and because the parameters are common, the depth of FIFO can be adjusted when using the top module. As shown in Figure 2.Take Altera Cyclone V development board as an example, when the development board is working under configuration mode, it can be pre-configured with five built-in modules which maybe different depth [10].
According to the code module line, can make RTL diagram.

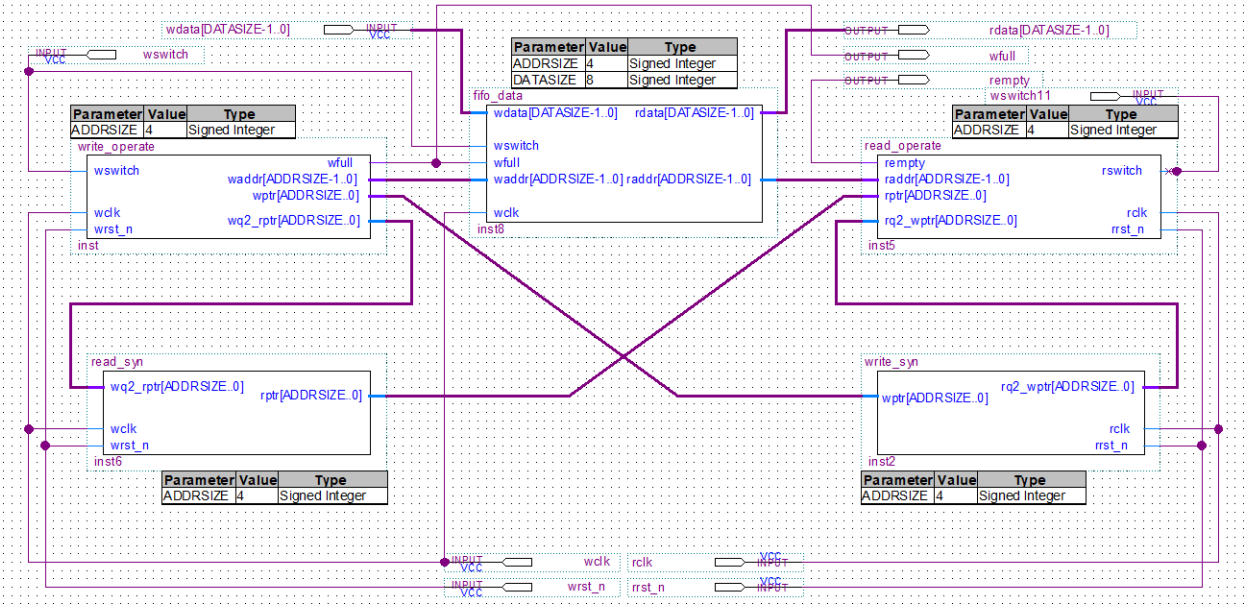

Figure 2. RTL diagram.

DOI: 10.54254/2755-2721/13/20230748

5. Simulation experiment on Modelsim

The figure 3 below shows the fundamental function of the FIFO. In this figure, the depth (ADDRSIZE in code) is set as 4. The first line is wclk and the second line is rclk, the write operates and read operate work under their respective clock domains: wswitch, rswitch are respectively write and read the control switch, in the switch set high power usually, will carry out the data write and read. wrst_n and rrst_n respectively represent the reset of the write operation and read operation. When the signal is set to low power level, the read pointer and the write pointer is reset. At the same time, the write reset is also entered into the data module to clear the internal data. wfull and rempty are the full and empty judgments of data respectively. They are judged based on the write address and read address and output full signal in the write operation module and empty signal in the read operation module. waddr and raddr are addresses for writing and reading data respectively. These addresses are entered into the read and write operation module respectively for judgment. wq2_rptr and rq2_wptr indicate that the read pointer is sent to the write operation after being synchronized in the write clock domain, and the write pointer is sent to the read operation after being synchronized in the read clock domain, respectively [11]. The write pointer points to the next digit of the address. wdata and rdata are respectively write data and read data. As shown in Figure 3

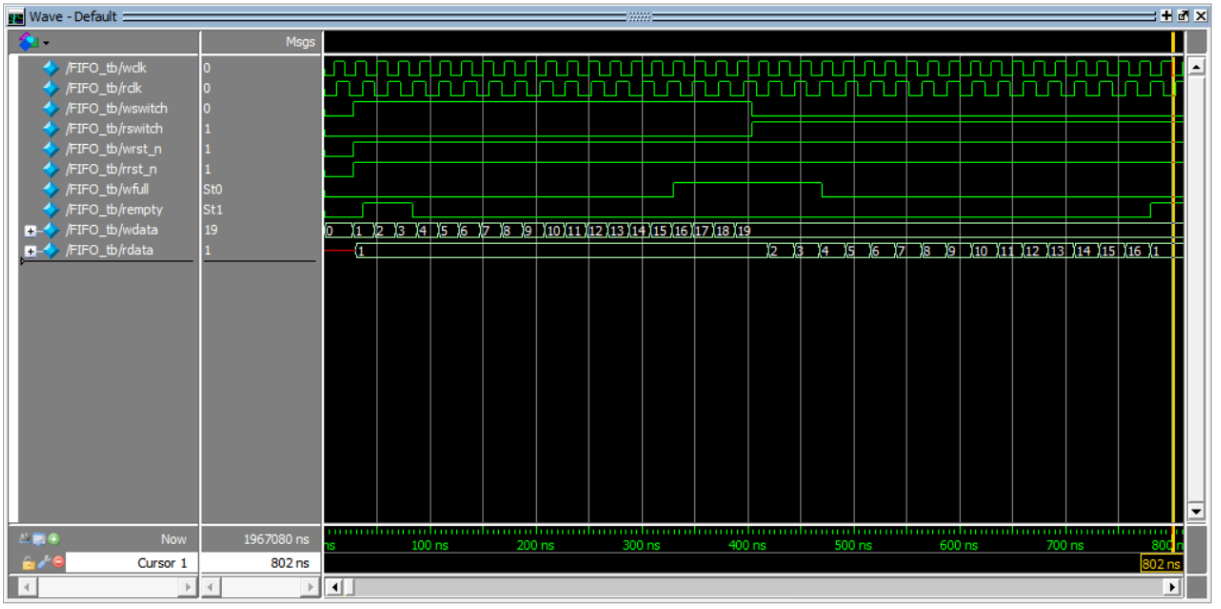

Figure 3. Simulation diagram.

Read the image according to the simulation image time: when rst is set to 0, read and write operations are not carried out and data is set to 0; After rst set 1, wswitch set 1, rswitch set 0, when the data is not written, the empty signal has been reported, but does not affect the write operation, at this time, write data and write pointer address with write clock a push forward, write 9 data after wswitch set 0 to stop writing, because the default data depth of the whole FIFO is 16, then not full signal, In addition, the reset position of the read pointer is [0], that is, the first data 1 is read directly; After rswitch is set to 1, the data is read. As the read clock reads the data one by one, it can be seen that the contents and sequences of the written and read data are the same. However, because of the FIFO is full, the data after 16 isn't recorded in the FIFO, and the full signal set 1 before the read operate is working.

In fact, because of the particularity of the parameter in Verilog, so in order to achieve the purpose of depth adjustable, the functionality of the development board, however, this FIFO design is imperfect, there is still room for advancement. A couple of ways: First, the design can use some IP cores which can raise the priority of the declared parameter; thus, a judgement before the module work to change

DOI: 10.54254/2755-2721/13/20230748

the depth. What's more, some development boards which can control the incoming parameters when the module is working.

6. Conclusion

For the current common buffer may make the whole chip work slowly and increase the workload of the chip. The appearance of asynchronous FIFO not only makes the circuit work faster, but also can work in different clock domains. The improvement of depth adjustment will further expand the application scope of FIFO. It has a good application prospect.

References

[1] Davis W R, Oh E C, Sule A M, et al. Application exploration for 3-D integrated circuits: TCAM, FIFO, and FFT case studies[J]. IEEE transactions on very large scale integration (VLSI) systems, 2009, 17(4): 496-506.

[2] Chandra V, Xu A, Schmit H, et al. An interconnect channel design methodology for high performance integrated circuits[C]//Proceedings Design, Automation and Test in Europe Conference and Exhibition. IEEE, 2004, 2: 1138-1143.

[3] Han H S, Stevens K S. Clocked and asynchronous FIFO characterization and comparison[C]//2009 17th IFIP International Conference on Very Large Scale Integration (VLSI-SoC). IEEE, 2009: 101-108.

[4] Laberge S, Negulescu R. An asynchronous FIFO with fights: case study in speed optimization[C]//ICECS 2000. 7th IEEE International Conference on Electronics, Circuits and Systems (Cat. No. 00EX445). IEEE, 2000, 2: 755-758.

[5] Fenstermaker L R, O'Conner K J. A low-power generator-based FIFO using ring pointers and current-mode sensing[C]//1993 IEEE International Solid-State Circuits Conference Digest of Technical Papers. IEEE, 1993: 242-243.

[6] Nagalaxmi T, Rao E S, Chandrasekar P. Design and Development of Low Power Clock and Data Recovery Circuit for Asynchronous Network on Chips[J]. Journal of Integrated Circuits and Systems, 2022, 17(3): 1-9.

[7] Basha S M, Arun A, Subha T D, et al. FPGA implementation of integrated circuit interactions along with modem connectivity and adapter transponder[C]//AIP Conference Proceedings. AIP Publishing LLC, 2022, 2519(1): 050037.

[8] Das S, Basu U, Das R, et al. FPGA Implementation of Asynchronous FIFO[C]//Proceedings of International Conference on Industrial Instrumentation and Control: ICI2C 2021. Singapore: Springer Nature Singapore, 2022: 399-407.

[9] George M L, Innocent-George N. Development of a General Purpose First-In-First-Out (FIFO) Core[J]. i-Manager's Journal on Circuits & Systems, 2022, 10(1): 1.

[10] Sumbul H E, Wu T F, Li Y, et al. System-Level design and integration of a prototype AR/VR hardware featuring a custom low-power DNN accelerator chip in 7nm technology for codec avatars[C]//2022 IEEE Custom Integrated Circuits Conference (CICC). IEEE, 2022: 01-08.

[11] Short S E, Burnett D C. An Analog Frontend Controller for Continuous Sensor Sampling Without CPU Interaction for Fully-Integrated Single-Chip Sensor Systems[C]//2022 IEEE 13th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON). IEEE, 2022: 0426-0430.