DESIGN AND IMPLEMENTATION OF ASYNCHRONOUS SRAM

CHENG XIANG

NATIONAL UNIVERSITY OF SINGAPORE

2009

DESIGN AND IMPLEMENTATION OF ASYNCHRONOUS SRAM

CHENG XIANG

(*B.ENG., Beijing Institute of Technology*)

A THESIS SUBMITTED

FOR THE DEGREE OF MATSER OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2009

ACKNOWLEDGEMENTS

First, I would like to acknowledge my supervisor, Professor Lian Yong, for his kind support and guidance during my study at NUS. I appreciate the invaluable assistance and advice that he has given to me. It was an honor to work with him.

I would like to express my sincere thanks to all of my colleagues in Signal Processing and VLSI design Laboratory for their support during the project, who have made the project easier to get through in one way or another.

I am grateful to Xu Xiaoyuan, Zou Xiaodan and Tan Jun for their kind help during the tapeout.

I would also like to thank my parents for their love and encouragement.

The financial support of my project provided by NUS and ASTAR is gratefully acknowledged.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS ……..…... i

TABLE OF CONTENTS …….…….. ii

SUMMARY ……..….. v

LIST OF TABLES …… vii

LIST OF FIGURES …… viii

LIST OF ABBREVIATIONS AND SYMBOLS …… xiii

Chapter 1: INTRODUCTION ……..…… 1

1.1 Introduction to conventional synchronous SRAM ……. 1

1.2 Motivations for asynchronous logic ……..…... 7

1.3 Introduction to asynchronous circuits ……..……... 9

1.4 Objectives and thesis contributions ……..……..…… 13

1.5 Thesis organization ……..……...……..…….. 14

Chapter 2: LITERATURE REVIEW ……..…….. 16

2.1 Review of low power techniques ……..…… 16

2.1.1 Sources of power dissipation ……..…… 16

2.1.2 Minimizing power consumption ……..……...…….. 21

2.2 Review of asynchronous circuits design ……..… 22

2.2.1 Fundamentals of asynchronous circuits…….……..…….. 22

2.2.2 Review of recent asynchronous circuits designs ……...……. 28

Chapter 3: ASYNCHRONOUS SRAM DESIGN ……..……...…….. 38

3.1 Introduction 38

3.2 Self-timed SRAM cell 38

3.3 Specification of SRAM module 40

3.4 Self-timed SRAM design 43

3.4.1 SRAM cell 44

3.4.2 Control part 45

3.4.3 Acknowledge part 46

3.4.4 Data-path 48

Chapter 4: CIRCUIT LEVEL DESIGN AND SIMULATION RESULTS 50

4.1 Introduction 50

4.2 Muller C-element deign 50

4.3 Precharge and select circuit design 55

4.3.1 Precharge control circuit 57

4.3.2 Row decoder 57

4.3.3 Select acknowledge circuit 60

4.4 Acknowledge module design 65

4.5 Layout consideration 69

4.6 Schematic and post-layout simulation 71

Chapter 5: EXPERIMENTAL RESULTS 82

5.1 Introduction 82

5.2 Testing setup 82

5.3 Testing results 85

5.4 Summary of the performance ……..….... 92

Chapter 6: CONCLUSIONS …….....…..... 95

BIBLIOGRAPHY …….....…..... 97

SUMMARY

In recent decades, low power consumption is getting more and more necessary due to the market booming of portable electronic devices. In general, asynchronous circuits have the properties of low power consumption thanks to the dynamic power scaling and no global clock distribution. Therefore, asynchronous circuits design becomes more and more popular, and the design of asynchronous SRAM used in the microcontroller also requires more attention. Some implementations have been emulating asynchronous SRAM using synchronous SRAM and synchronous to asynchronous logic transform interface. However, it requires clock signal and other auxiliary circuits which still consume some unnecessary power. An intrinsic asynchronous SRAM using four-phase, dual-rail protocol has been designed and implemented in this project.

In this thesis, some fundamentals of the asynchronous circuits are introduced first, followed by the summary of asynchronous circuits design in recent years. The specifications of the asynchronous SRAM are presented, including the self-timed memory cell which can tell when the read and write operations end, the control part which controls the precharge and select signal, the acknowledge part which generates the overall read and write acknowledge signal, and the data-path which consists of the input and output modules. Some circuit level designs are also presented followed by the relevant simulation results. The experimental testing is conducted and some parameters including delays, current values and power consumption are measured.

Two versions of asynchronous SRAM have been fabricated with AMS 0.35um double-poly four-metal CMOS process. To verify the function and the integrity of the design, the first version of the 16*8 bits asynchronous SRAM has been fabricated separately from the asynchronous 8051 microcontroller. The second version of the 128*8 bits asynchronous SRAM has been fabricated with the asynchronous 8051 microcontroller in one chip. The experimental testing of the 128*8 bits version relies on the working status of the asynchronous 8051 microcontroller due to the pad limitations. Both of the two versions of the SRAM are proved working well under the supply voltage between 0.81V and 3.5V. The experimental results show that the delays t_{WA}, t_{RA} and t_{RO} of 16*8 bits asynchronous SRAM are 19.0ns, 17.0ns and 12.6ns at 3.3V, and 176.0ns, 168.0ns and 140.0 ns at 1.0V.

LIST OF TABLES

Table 2.1: TSMC process leakage and V_T 20

Table 2.2: Transition states of 4-phase dual-rail protocol 23

Table 2.3: Truth table of the Muller C-element 27

Table 2.4: Truth table of a dual-rail AND gate 33

Table 4.1: Truth table of the generic 2-4 decoder 58

Table 4.2: Simulation results of the 16*8 bits asynchronous SRAM 72

Table 4.3: Simulation results of the 128*8 bits asynchronous SRAM 73

Table 5.1: Experimental results of the 16*8 bits asynchronous SRAM 85

Table 5.2: Experimental SRAM core current / power consumption 87

Table 5.3: Experimental current consumption (uA) of the SRAM core and the total SRAM circuit 89

LIST OF FIGURES

Figure 1.1: Six-transistor CMOS SRAM cell …… 2

Figure 1.2: Simplified model of CMOS SRAM cell during read (Q = 1) ……... 4

Figure 1.3: Simplified model of CMOS SRAM cell during write (Q = 1) …..... 6

Figure 1.4: (a) A synchronous circuit; (b) A synchronous circuit with clock drivers and clock gating; (c) An equivalent asynchronous circuit; (d) An abstract data-flow view of the asynchronous circuit ……..….. 12

Figure 2.1: Illustration of dynamic power dissipation …….. 17

Figure 2.2: Illustration of short circuit power consumption ……. 18

Figure 2.3: Illustration of leakage power consumption ……..... 20

Figure 2.4: Transition diagram of 4-phase dual-rail protocol ……..….. 23

Figure 2.5: A delay-insensitive channel using the 4-phase dual-rail protocol .. 24

Figure 2.6: Illustration of the handshaking on a 2-phase dual-rail channel ….. 25

Figure 2.7: (a) A bundled-data channel. (b) A 4-phase bundled-data protocol. (c) A 2-phase bundled-data protocol …….. 26

Figure 2.8: A normal OR gate and its truth table …….. 26

Figure 2.9: The symbol and possible implementation of the Muller C-element ……. 27

Figure 2.10: A simple 4-phase bundled-data pipeline …….…….…….……... 29

Figure 2.11: A simple 2-phase bundled-data pipeline ……..….……..... 30

Figure 2.12: A simple 3-stage 1-bit wide 4-phase dual-rail pipeline ….....….... 31

Figure 2.13: An N-bit latch with completion detection …. 32

Figure 2.14: The symbol and implementation of a 4-phase dual-rail AND gate ……... 34

Figure 2.15: A circuit fragment with gate and wire delays ……..... 34

Figure 3.1: A standard six-transistor SRAM cell with precharge transistors 39

Figure 3.2: Timing diagram of SRAM write and read operations ……..... 40

Figure 3.3: A standard dual-port SRAM cell and a self-timed SRAM cell ….. 41

Figure 3.4: Specification of SRAM module ……. 42

Figure 3.5: Block diagram of the 4*8 bits asynchronous SRAM ……. 43

Figure 3.6: Transistor diagram of the self-timed SRAM cell ……... 44

Figure 3.7: Timing diagram of the control part signals …… 46

Figure 3.8: Timing diagram of acknowledge part signals …… 48

Figure 3.9: Timing diagram of the data-path signals …… 49

Figure 4.1: Schematic and layout of the 2-input Muller C-element ……. 51

Figure 4.2: Layouts of a) MAJ31 and b) customized Muller C-element …….. 52

Figure 4.3: Transient response of the gate MAJ31 (Post-layout at 1.5V) ……. 52

Figure 4.4: Transient response of the customized Muller C-element (Post-layout at 1.5V) …….. 53

Figure 4.5: Implementation and symbol of the 8-input Muller C-element …... 53

Figure 4.6: Schematic and layout of the 8-input Muller C-element …….. 54

Figure 4.7: Transient response of the 8-input Muller C-element (Schematic and post-layout at 1.5V) …… 54

Figure 4.8: Schematic of the precharge and select circuit ……. 55

Figure 4.9: Schematic of the precharge control circuit ……. 57

Figure 4.10: Schematic of the generic 2-4 decoder ……..……. 58

Figure 4.11: Schematic of the proposed 2-4 decoder …….……... 59

Figure 4.12: Layout of the proposed 2-4 decoder …… 59

Figure 4.13: Schematic of the proposed 4-16 decoder …… 60

Figure 4.14: Schematic of a 2-input OR gate …….. 61

Figure 4.15: Schematic of the 16-input OR gate ……. 62

Figure 4.16: Layout of the 16-input OR gate …….. 62

Figure 4.17: Transient response of the 16-input OR gate (Schematic and post-layout at 1.5V) …… 63

Figure 4.18: Schematic of the precharge and select circuit of the 4*8 bits asynchronous SRAM ……..... 64

Figure 4.19: Transient response of the precharge and select circuit (Post-layout at 1.5V) ……... 65

Figure 4.20: Schematic of the acknowledge module ……... 66

Figure 4.21: Layout of one bit acknowledge circuit …… 67

Figure 4.22: Layout of the acknowledge module …… 67

Figure 4.23: Simulated timing diagram of the acknowledge module (Post-layout at 1.5V) ……... 68

Figure 4.24: Layout of the 16*8 bits asynchronous SRAM ……..… 69

Figure 4.25: Layout of the asynchronous 8051 microcontroller ……...… 70

Figure 4.26: Timing diagram of the signals of the write and read operations ... 71

Figure 4.27: Simulated delay of the 16*8 bits and 128*8 bits asynchronous SRAM ……..…… 74

Figure 4.28: Simulated delay comparison between 16*8 bits and 128*8 bits asynchronous SRAM ……..……..…..…….. 75

Figure 4.29: Simulated timing diagram of 16*8 bits asynchronous SRAM (Schematic and post-layout at 1.5V) ……. 77

Figure 4.30: Simulated timing diagram of 128*8 bits asynchronous SRAM (Schematic and post-layout at 3.3V) ……. 78

Figure 4.31: Simulated timing diagram of 128*8 bits asynchronous SRAM (Schematic and post-layout at 1.5V) ……. 79

Figure 4.32: Simulated timing diagram of 128*8 bits asynchronous SRAM (Schematic and post-layout at 1V) ……..…. 80

Figure 4.33: Simulated timing diagram of 128*8 bits asynchronous SRAM (Schematic and post-layout at 0.87V) ……... 81

Figure 5.1: Experimental test setup for the SRAM module and the asynchronous 8051 microcontroller …… 83

Figure 5.2: Photograph of the PCB used for testing ……. 84

Figure 5.3: Diagram of comparison between the experimental results and post-layout results of the 16*8 bits asynchronous SRAM ……. 86

Figure 5.4: Diagram of experimental SRAM core power consumption ……... 88

Figure 5.5: Experimental timing diagram of write and read operation (3V) … 89

Figure 5.6: Experimental timing diagram of write and read operation (3.3V) . 90

Figure 5.7: Experimental timing diagram of write and read operation (1V) … 90

Figure 5.8: Experimental timing diagram of write and read operation (0.9V) . 91

Figure 5.9: Experimental timing diagram of write and read operation (0.81V) …

…… 91

Figure 5.10: Experimental timing diagram of write and read operation using digital function of the oscilloscope …… 92

Figure 5.11: Die photograph of the 16*8 bits asynchronous SRAM ……..... 93

Figure 5.12: Die photograph of the asynchronous 8051 microcontroller with the 128*8 bits asynchronous SRAM in the red box ……..…… 94

LIST OF ABBREVIATIONS AND SYMBOLS

ARM	Advanced RISC Machine
AMS	Austria Micro Systems
CAD	Computer-Aided Design
CMOS	Complementary Metal-Oxide-Semiconductor
CR	Cell Ratio
DI	Delay-insensitive
DRAM	Dynamic Random Access Memory
DVSCD	Dual-rail Voltage-sensing Completion Detection
GaAs	Gallium Arsenide
MESFET	Metal-Semiconductor Field-Effect Transistor
MDCG	Multiple Delays Completion Generation
MOSFET	Metal-Oxide-Semiconductor Field-Effect Transistor
PR	Pull-up Ratio
QDI	Quasi-delay-insensitive
RAM	Random Access Memory
RTL	Register Transfer Level
SI	Speed-independent
SOI	Silicon on Insulator
SRAM	Static Random Access Memory
VLSI	Very Large Scale Integration

k	Gain Factor
V_{TH}	Threshold Voltage
V_{DSAT}	Saturation Voltage
$f_{0\to1}$	Switching activity
$P_{0\to1}$	Switching probability

CHAPTER 1

INTRODUCTION

1.1 Introduction to conventional synchronous SRAM

Nowadays, computer data storage memory includes volatile type and non-volatile type. Random Access Memory (RAM) belongs to volatile type while Read Only Memory (ROM) and flash memory belong to non-volatile type. In the former, the phrase “random access” comes from the fact that locations in the memory can be written into or read from in any order, regardless of last accessed memory location. RAM can be categorized into Static RAM (SRAM) and Dynamic RAM (DRAM) based on the way in which data are stored in the memory cells. Compared with Dynamic RAM, Static RAM saves a lot of power especially in idle state for it does not need to be periodically refreshed.

Static RAM uses bistable latching circuitry to store each bit. The generic SRAM architecture has six transistors in one memory bit cell which is shown in Figure 1.1. Each bit of data in an SRAM cell is stored in two cross-coupled inverters formed by four transistors (M1, M2, M3, and M4) and this storage cell has two stable states denoted as 0 and 1. Two pass transistors (M5 and M6) connect the bit lines with the memory cell and they are used to control the access to the storage cell during read and write operations. To achieve improved read ability or multi-port functions, some kinds of SRAM using 7-transistor (7T), 8T, 10T, or more transistors per bit [1] have been proposed in addition to such 6T SRAM. For

example, 8T and 10T bit cells provide extra sensing circuit for reading the cell contents and some SRAM implementations for video application need more than one port to perform the read and/or write operation.

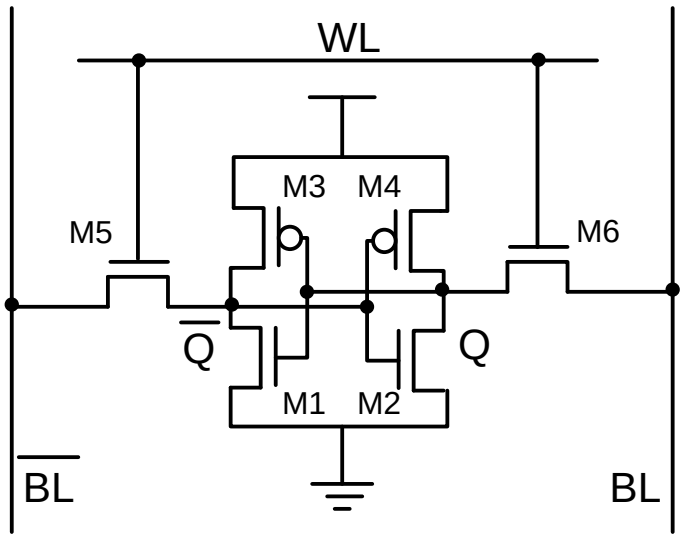


Figure 1.1 Six-transistor CMOS SRAM cell

The two pass transistors M5 and M6 controlled by the word line (*WL* in Figure 1.1) control whether the cell should be connected to the bit lines: *BL* and $\overline{BL}$. They are used to pass data for both read and write operations. In order to improve noise margins, both the signal and its inverse are provided. However, it is not strictly necessary to have two bit lines. For some single ended reading scheme memory, only one bit line is necessary. The bit lines are driven high and low by active components (inverters) in the SRAM cell during read access. This improves SRAM bandwidth compared to DRAM where data is stored in passive components (storage capacitors). In a SRAM, the symmetric structure also allows for differential signaling, making small voltage swings more easily detectable.

In general, the size of an SRAM with *m* address lines and *n* data lines is 2^m words, or $2^m \times n$ bits. To achieve high memory densities, the size of the memory cell should be as small as possible. To ensure the reliable operation of the cell, there are some sizing constraints which will be discussed later.

The SRAM cell has three different states which are standby, reading and writing respectively. The three different states work as follows:

Firstly, when the word line (*WL*) is not asserted, the circuit is idle or in standby state, and the memory cell is disconnected by the pass transistors M5 and M6 from the bit lines. As long as the circuit is connected to the power supply, the two cross coupled inverters formed by M1 to M4 will continue to reinforce each other to keep the data stored in the memory cell.

Secondly, when the data has been requested by CPU or peripheral circuit, the memory goes into the reading state. The simplified model of SRAM cell during reading operation is shown in Figure 1.2. Assume there is a logical 1 stored at *Q* in the memory cell. The read cycle starts when both the bit lines have been pre-charged to a logical 1, then the word line WL is asserted, enabling both the pass transistors M5 and M6. Then the data values stored in Q and $\overline{Q}$ are transferred to the bit lines, leaving BL at its precharged value and discharging $\overline{BL}$ through the two transistors M1 and M5 to a logical 0. On the BL side, the two transistors M4 and M6 pull the bit line toward V_{DD}, or a logical 1. If the content of the memory which is stored at Q is a logical 0, the opposite situation will happen and

BL will be pulled toward 0 and $\overline{BL}$ toward 1.

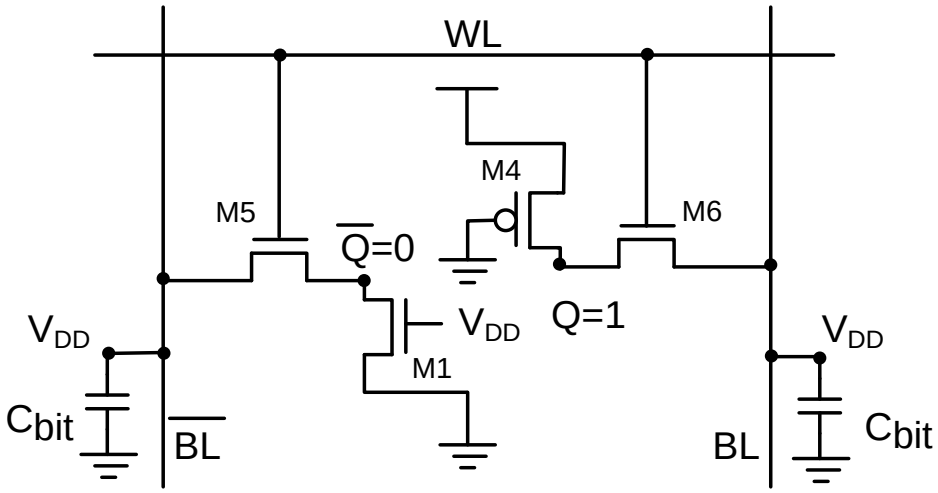


Figure 1.2 Simplified model of CMOS SRAM cell during read (Q = 1)

Careful sizing of the transistors is necessary to avoid other value accidently written into the memory cell. Initially, when the *WL* is asserted, the intermediate node $\overline{Q}$ between the two NMOS transistors M1 and M5 is pulled up towards the precharged value of $\overline{BL}$. The voltage rise of $\overline{Q}$ should stay low enough to avoid causing a substantial current through the transistors M2 and M4, which could flip the memory cell in the worst case. Therefore, it is necessary to keep the resistance of transistor M5 larger that the resistance of M1 to prevent this from happening. By solving the current equation at the maximum allowed value of the voltage ripple ΔV, ignoring the body effect on transistor M5 for simplicity, the boundary constraints on the device size can be derived as follow equation [2]:

$$k_{n,M5}\left((V_{DD}-\Delta V-V_{Tn})V_{DSATn}-\frac{V^2{}_{DSATn}}{2}\right)=k_{n,M1}\left((V_{DD}-V_{Tn})\Delta V-\frac{\Delta V^2}{2}\right) \quad (1.1)$$

which simplifies to

$$\Delta V = \frac{V_{DSATn} + CR(V_{DD} - V_{Tn}) - \sqrt{V^2{}_{DSATn}(1+CR) + CR^2(V_{DD} - V_{Tn})^2}}{CR} \quad (1.2)$$

where CR is called the cell ratio and is defined as

$$CR = \frac{W_1 / L_1}{W_5 / L_5} \quad (1.3)$$

To prevent the node voltage from rising above the transistor threshold (about 0.6V in standard 0.35um CMOS processes), the cell ratio *CR* must be greater than 1.1. It is desirable to keep the cell size minimal while maintaining read stability for large memory arrays. If the size of transistor M1 is minimal, the pass transistor M5 has to be made weaker by increasing its length which is undesirable, because it adds to the load of the bit lines. One preferred solution is to minimize the size of the pass transistor and increase the width of the NMOS transistor M1, though it slightly increases the minimum size of the cell.

Lastly, when the contents of the memory cells need to be updated, the memory goes into the writing state. During the initiation of the write operation, the schematic of the SRAM cell can be simplified to the model of Figure 1.3. As long as the switching has not commenced, it is reasonable to assume that the gates of transistors M1 and M4 stay at V_{DD} and GND respectively. The write cycle begins with applying the data value to be written to the bit lines. For example, if a logical 1 is to be written into the memory cell, logical 1 and logical 0 are applied to the bit lines BL and $\overline{BL}$ respectively. This is similar to applying a reset pulse to a SR-latch causing the flip-flop to change state. Then *WL* is asserted and the value

that is to be stored is latched in.

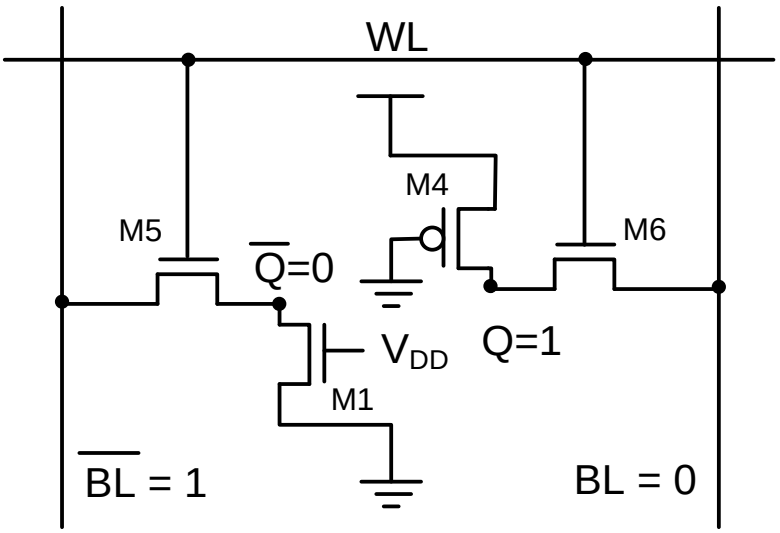


Figure 1.3 Simplified model of CMOS SRAM cell during write (Q = 1)

Note that the sizing constraints imposed by read stability ensures that the voltage of node $\overline{Q}$ is kept below the transistor threshold. Therefore, the new data value of the memory cell has to be written through transistor M6. If node Q can be pulled low enough, i.e. below the threshold of the transistor M1, the reliable writing of the cell is ensured. The conditions for this to happen can be derived by writing out the dc current equations at the desired threshold point, which is as follow equation [2]:

$$k_{n,M6}\left((V_{DD}-V_{Tn})V_Q-\frac{V_Q{}^2}{2}\right)=k_{p,M4}\left((V_{DD}-|V_{Tp}|)V_{DSATp}-\frac{V^2{}_{DSATp}}{2}\right) \qquad (1.4)$$

Solving for V_Q leads to

$$V_Q=V_{DD}-V_{Tn}-\sqrt{(V_{DD}-V_{Tn})^2-2\frac{\mu_p}{\mu_n}PR\left((V_{DD}-|V_{Tp}|)V_{DSATp}-\frac{V^2{}_{DSATp}}{2}\right)} \qquad (1.5)$$

where the pull-up ratio of the cell, PR, is defined as the ratio between the PMOS pull-up and the NMOS pass transistor:

$$PR = \frac{W_4/L_4}{W_6/L_6} \tag{1.6}$$

If the node need to be pulled below the threshold of NMOS transistor M6, the pull-up ratio has to be smaller than 1.9 for standard 0.35um CMOS processes. When both NMOS pass transistor M6 and PMOS pull up transistor M4 are minimum sized, this constraint is met by a large margin. However, the writeability constraint should be met under all the process corners. The worst case happens with weak NMOS devices and strong PMOS devices, and the memory operated at a higher supply voltage. The initial assumption that the transistors M1 and M3 do not participate in the writing process is not completely true in practice. One side of the memory cell eventually follows and engages the positive feedback as soon as the other side of the cell starts switching. In general, the bit line input-drivers are designed to be much stronger than the relatively weak transistors in the memory cell, so they can easily override the previous state of the cross-coupled inverters.

1.2 Motivations for asynchronous logic

Most of the digital circuits designed and fabricated today are synchronous, as all components of the circuit share a clock signal. The clock signal forces a strict timing restraint onto a digital circuit in order to solve race and hazard problems. Asynchronous circuits are fundamentally different from synchronous circuits: they

do not have the clock signal and they use handshaking instead of clock between components to make the necessary communication and synchronization. Therefore, asynchronous circuits have the following advantages:

- No clock distribution and clock skew problems

  No global signal is needed to be distributed across the circuit.

- Robustness towards various variations

  As timing is based on match delays, asynchronous circuits are robust towards supply voltage, process and temperature.

- Better composability and modularity

  Due to simple handshake interface and local timing, it is easy to design the circuit and make the circuit modularity.

- Low power consumption

  Supply voltage can be scaled dynamically according to real time requirements to save power; less strict timing constraint is posed as in synchronous circuits. Clock power overhead is eliminated, as signal transitions only occur when needed.

- Achieve high operating speed

  Operation speed is determined by actual local circuit delays rather than global worst-case latency.

- Less emission of electromagnetic noise

  Local clocks ensure that the clock pulses are generated where and when needed and they tend to tick at random points in time.

Many asynchronous circuits have been designed and fabricated in recent decades due to the advantages of the asynchronous circuit. However, there are also some drawbacks on asynchronous circuits design. First, asynchronous circuits design is different from synchronous circuits design. Researchers and engineers who are already used to synchronous circuits design need time to get used to the new thinking method. Second, the asynchronous circuits design is still a young discipline. Different circuit structures and design methods are proposed by different researchers. Although the essential principles and resulting circuits are similar, they may seem different at a first glance, which adds to the difficulties for the learners. Last but not least, the lack of computer-aided design (CAD) tools and testing tools is an obstacle for the designers. Compared with the advantages of asynchronous design, these drawbacks are getting more and more insignificant. For example, lectures on asynchronous circuit design have been introduced to some universities, more and more students get familiar with this new design method and the functions of the CAD tools are getting increasingly comprehensive.

1.3 Introduction to asynchronous circuits

In this section, comparison between synchronous circuits and asynchronous circuits is presented to give an overview of asynchronous circuits. Clocking versus

handshaking is discussed here.

A synchronous circuit is shown in Figure 1.4 (a). Although the figure shows a pipeline for simplicity, it is intended to represent any synchronous circuit [3]. Designers mostly focus on the data processing and assume that a global clock exists when designing ASICs using hardware description languages and synthesis tools. For instance, as shown in Figure 1.4 (a), a high-level view with a universal clock is presented. The fact that the data clocked into the register R3 is a function CL3 of the data clocked into the register R2 at the previous clock would be expressed as the assignment of variables which is as follows: R3 := CL3(R2). [3]

The reality is different when it comes to physical design. As shown in Figure 1.4 (b), a great number of clock signals resulted by the structure of clock buffers is applied by ASICs today. It takes great effort to design the clock gating circuit and to minimize and control the skew between so many different clock signals. It is not easy to guarantee the two-sided timing constraints which is the setup time to the hold time window dominated by wire delays. What's more, in current commercial CAD tools, wire delay models which buffer-insertion-and-resynthesis process relies on are not completely accurate.

Asynchronous design presents an alternative way to this. As mentioned in Section 1.2, the clock signal is replaced by some kind of handshaking signals between neighboring registers in an asynchronous circuit. For example, an asynchronous circuit which is using the simple request-acknowledge based handshak-

ing protocol is shown in Figure 1.4 (c). An asynchronous circuit is simply a static data-flow structure which is shown as Figure 1.4 (d) if we consider the circuit as follows: first, the data and handshaking signals connect one register to the next as handshake “channel” or “link”; second, data is stored in the registers as tokens tagged with data values; third, combination circuits are transparent to the handshaking between registers which implies that a combinational circuit just simply absorbs a token on each of its input links, performs its computation and then emits a token on each of its output links. If a register’s successor has input and stored the data token that the register was previously holding, this register may input and store a new data token from its predecessor. In other words, the states of the predecessor and successor registers are signaled by the incoming request and acknowledge signals respectively. Complied with this, the data is copied from one register to the next along the path through the circuit. Subsequent registers will be holding the same data value copies in this process. But the old duplicate values will be overwritten by new data values in a carefully ordered mode, the transfer of exactly one data-token will always be enclosed on a handshake cycle.

The “handshake-channel and data-token view” represents a very useful abstraction that is equivalent to the register transfer level (RTL) which is used in synchronous circuits design. This data-flow abstraction separates the structure and function of the circuit from the implementation details of its components.

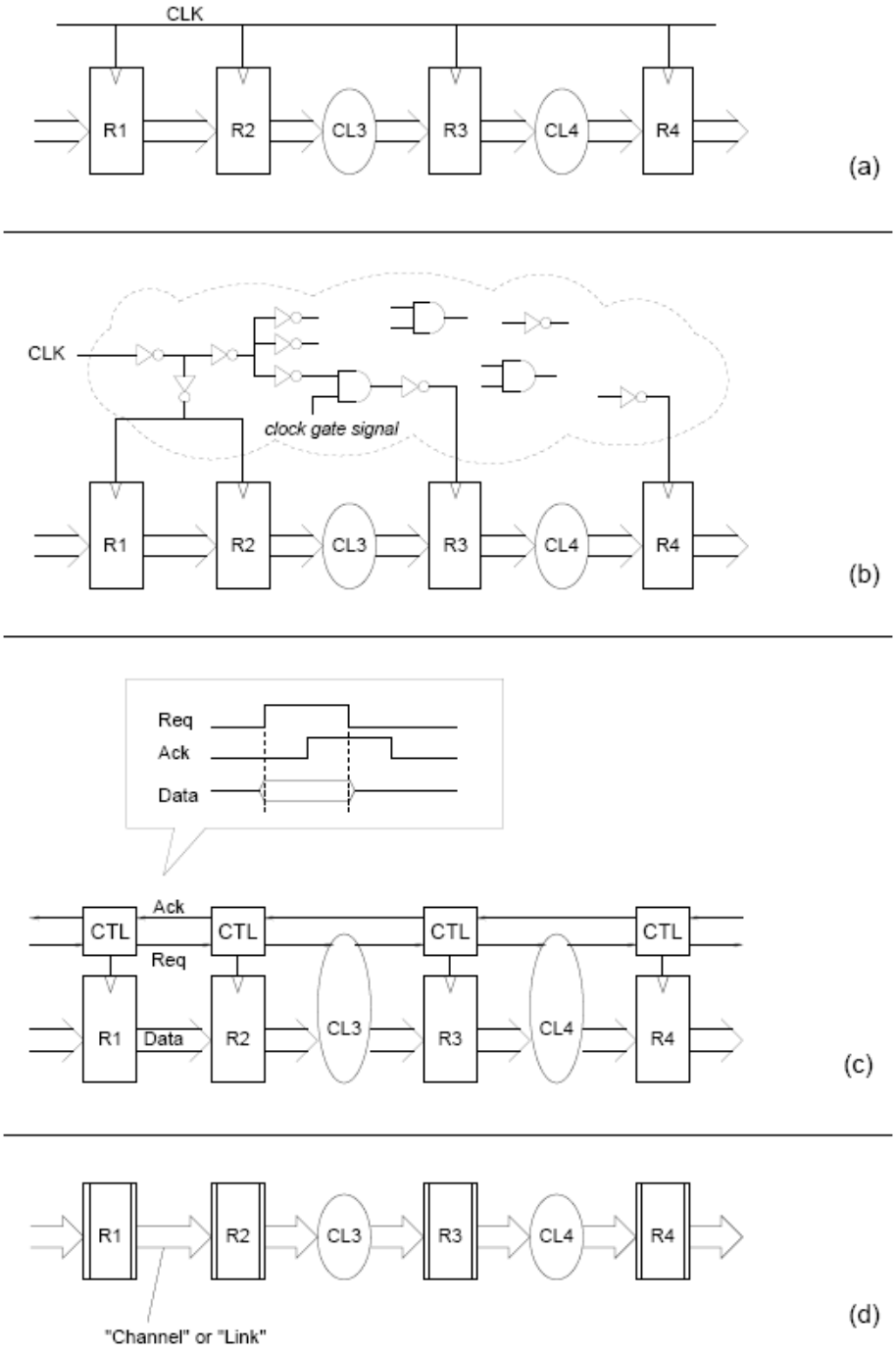


Figure 1.4 (a) A synchronous circuit; (b) A synchronous circuit with clock drivers and clock gating; (c) An equivalent asynchronous circuit; (d) An abstract data-flow view of the asynchronous circuit. (The figure shows a pipeline, but it is intended to represent any circuit topology.)

Compared with the synchronous circuit shown in Figure 1.4 (b) controlled by a periodic clock pulses, the asynchronous circuit shown in Figure 1.4 (c) is con-

trolled by locally derived clock. This local handshaking ensures that clock pulses are generated when and where needed and it is likely to result in less electromagnetic emission.

1.4 Objectives and thesis contributions

The main objective of this project is to design and implement an asynchronous SRAM that can be used directly for the asynchronous 8051 microcontroller also designed in this project group. The emphasis of this research is to design an asynchronous SRAM which can work well at the supply voltage range between 1.0V and 3.3V. Nowadays, some of the implementations of the asynchronous SRAM are using the off-the-shelf synchronous SRAM with asynchronous logic to synchronous logic interface and extra control circuit to emulate as asynchronous SRAM. The problem is that it still needs a clock signal generated by peripheral circuit to synchronize the synchronous SRAM with the peripheral asynchronous circuit which sometimes costs unnecessary power. In this asynchronous design, event-trigger mechanism is used to activate the circuit and most of the time the asynchronous SRAM is in idle state. Therefore, an intrinsic asynchronous SRAM has been chosen to be designed and implemented. The main functional blocks of this design consist of the memory cell, control part, acknowledge part and data-path which will be introduced in Chapter 3. The four-phase dual-rail handshaking protocol is chosen to be the asynchronous communication protocol.

A paper entitled as “The Design of a Sub-Nanojoule Asynchronous 8051

with Interface to External Commercial Memory" was published in the 8th IEEE International Conference on ASIC (IEEE ASICON 2009). This paper presents the design of an asynchronous 8051 microcontroller with interface to external commercial memory. The design consists of an asynchronous core implemented using dual-rail four-phase protocol, a 128-byte internal intrinsic asynchronous SRAM and other synchronous peripherals including interrupts, timers and serial port. Some contents of this paper will be elaborated in Chapter 4.

1.5 Thesis organization

In this thesis, the asynchronous SRAM design is discussed. The simulation results and test results of the asynchronous SRAM are also presented. The thesis is organized into six chapters as follows:

Chapter 2: gives a literature review of the low-power techniques in digital circuits design and the asynchronous circuits design. A detailed introduction of the asynchronous circuit fundamentals will be presented. Previous works on asynchronous circuits design and asynchronous memory design will be summarized.

Chapter 3: presents the overview of the asynchronous SRAM design. The self-timed SRAM cell will be introduced first, followed by the specification of the SRAM module. Then the main parts of the SRAM circuit will be talked about and the SRAM operation sequence will be presented.

Chapter 4: focuses on the circuit level design of the asynchronous SRAM which includes the key parts of the circuit: Muller C-element, precharge and select

circuit and acknowledge circuit. This is followed by some layout considerations and the simulation results.

Chapter 5: The testing setup and the testing results of the fabricated SRAM chip will be presented in this chapter. Performance summary will also be given in this chapter.

Chapter 6: gives conclusions of this work.

CHAPTER 2

LITERATURE REVIEW

2.1 Review of low power techniques

The power dissipation problem is getting worse as technologies scale down and complexity of modern integrated circuit increases. Low power consumption is necessary to digital circuits, especially for portable electrical devices. Therefore, it is very important to understand the source of power consumption, and to have an accurate model to estimate it and decrease it.

2.1.1 Sources of power dissipation

There are three sources of power dissipation in digital circuits: dynamic power consumption, short circuit current and static leakage, which can be displayed in the equation as follows:

$$P_{total} = P_{dyn} + P_{short} + P_{leak} \tag{2.1}$$

Dynamic power consumption, P_{dyn}

Dynamic power consumption is due to charging and discharging capacitances. Each time the load capacitor C_L gets charged through pull-up transistors and gets its voltage raised from 0 to V_{DD}, a current $i_{V_{DD}}$ flows from the power supply through the pull-up transistor networks during the transition, and a certain mount of energy is delivered from the power supply. Some of this energy is dissipated in the PMOS transistors, while the other part of the energy is stored in the load capacitor $C_{L.}$ During the opposite operation which is high to low transition, the load

capacitor is discharged and the stored energy is dissipated in the NMOS transistors.

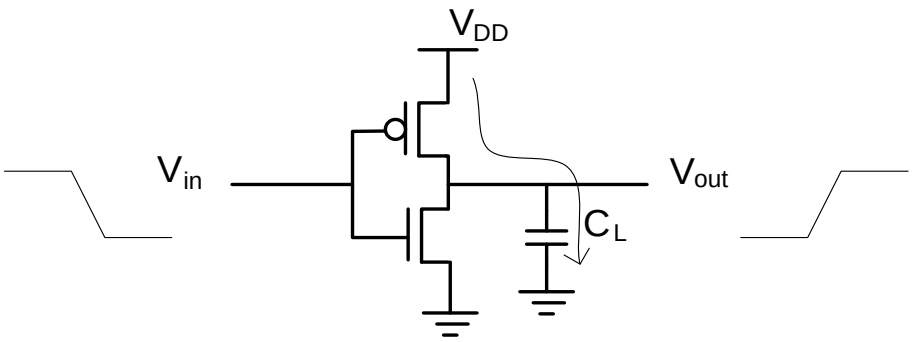


Figure 2.1 Illustration of dynamic power dissipation

During the low to high transition, the energy delivered from the power supply is given by

$$E = C_L \cdot V_{DD}{}^2 \tag{2.2}$$

and the energy for every system cycle is given by

$$Energy/transition = C_L \cdot V_{DD}{}^2 \cdot P_{0\to 1} \tag{2.3}$$

where $P_{0\to 1}$ is switching probability per cycle.

When the capacitance C_L is charged through the power supply, the current I is given by

$$I = C_L \frac{dv}{dt} \tag{2.4}$$

Then the energy stored in capacitor is given by

$$E_{C_L} = \int I \cdot V(t)dt = \int C_L \cdot V(t)dv = \frac{1}{2} C_L \cdot V_{DD}{}^2 \tag{2.5}$$

It can be seen that half of the energy delivered by the power supply is stored in the load capacitor from the equations (2.2) and (2.5). The other half has been dissipated by the PMOS transistors. Assuming the frequency of the system is f, then the dynamic power consumption is given by

$$P_{dyn} = C_L \cdot V_{DD}^{\ 2} \cdot P_{0\to1} \cdot f \tag{2.6}$$

It is convenient to introduce a coefficient $f_{0\to1}$ which is referred to as the circuit activity:

$$f_{0\to1} = P_{0\to1} \cdot f \tag{2.7}$$

then the equation (2.6) can be written as follows:

$$P_{dyn} = C_L \cdot V_{DD}^{\ 2} \cdot f_{0\to1} \tag{2.8}$$

Short Circuit Power Consumption, P_{short}

The short circuit power consumption is caused by a direct current path I_{SC} between V_{DD} and GND for a short period of time during switching which is shown in Figure 2.2.

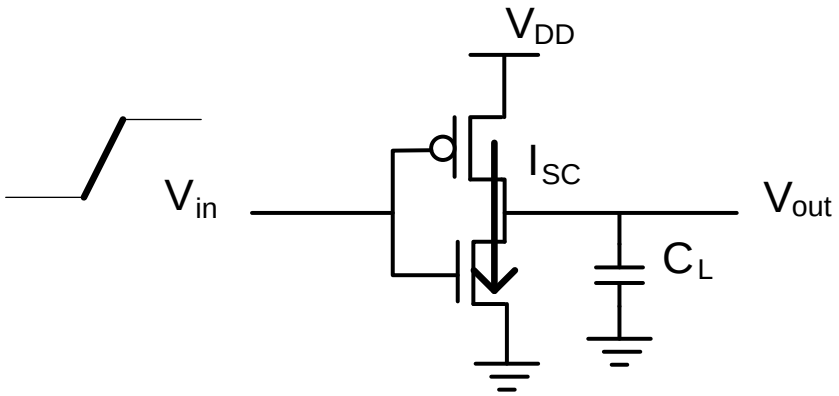


Figure 2.2 Illustration of short circuit power consumption

The NMOS and the PMOS transistors are conducting simultaneously because of the finite slope of the input signal. The short circuit time t_{SC} is the function of the slope duration t_S of the input signal which is given by [2]:

$$t_{SC} = \frac{V_{DD} - 2V_T}{V_{DD}} \cdot \frac{t_S}{0.8} \tag{2.9}$$

The short circuit energy E_{SC} is given by

$$E_{SC} = t_{SC} \cdot V_{DD} \cdot I_{peak} \cdot P_{0\to1} \tag{2.10}$$

where I_{peak} is determined by the saturation current of the PMOS and NMOS transistors which depend on their sizes, process technology, temperature, etc. And it is a strong function of $C_{L.}$ The short circuit power consumption P_{SC} is also proportional to the switching activity and it is given by the following equation:

$$P_{SC} = t_{SC} \cdot V_{DD} \cdot I_{peak} \cdot f_{0\to1} \tag{2.11}$$

Leakage power consumption, P_{leak}

The static leakage power consumption of a circuit P_{leak} is given by

$$P_{leak} = V_{DD} \cdot I_{leak} \tag{2.12}$$

The static current of the CMOS inverter is equal to zero ideally. Unfortunately, there is a leakage current flowing through the reverse-biased diode junctions of the transistors which is shown in Figure 2.3. The leakage current also consists of gate leakage and sub-threshold current. The contribution of leakage power consumption is very small compared with dynamic and short circuit power

consumption, sometimes it can be ignored. However, as the process feature size

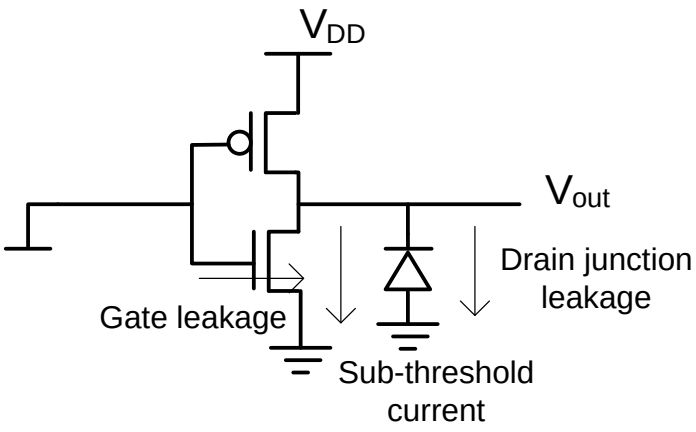


Figure 2.3 Illustration of leakage power consumption

scales down, leakage current is increasing substantially, causing leakage power no longer to be negligible, as shown in Table 2.1. What's more, the significant increase of the transistor count of the nowadays' design causes the rising of the working temperature, which in turn exponentially increases the leakage current.

Table 2.1 TSMC process leakage and V_T

	CL018 G	CL018 LP	CL018 ULP	CL018 HS	CL015 HS	CL013 HS
V_{dd}	1.8 V	1.8 V	1.8 V	2 V	1.5 V	1.2 V
T_{ox} (effective)	42 Å	42 Å	42 Å	42 Å	29 Å	24 Å
L_{gate}	0.16 μm	0.16 μm	0.18 μm	0.13 μm	0.11 μm	0.08 μm
I_{DSat} (n/p) (μA/μm)	600/260	500/180	320/130	780/360	860/370	920/400
I_{off} (leakage) (ρA/μm)	20	1.60	0.15	300	1,800	13,000
V_{Tn}	0.42 V	0.63 V	0.73 V	0.40 V	0.29 V	0.25 V
FET Perf. (GHz)	30	22	14	43	52	80

2.1.2 Minimizing power consumption

The power consumption P_{total} has been given by equation 2.1 which is repeated as follows:

$$P_{total} = P_{dyn} + P_{short} + P_{leak} \tag{2.13}$$

where P_{dyn} usually dominates in most switching intensive circuits. From equations 2.8, 2.11, 2.12, the above equation can be written as

$$P_{total} = C_L \cdot V_{DD}{}^2 \cdot f_{0\to1} + t_{SC} \cdot V_{DD} \cdot I_{peak} \cdot f_{0\to1} + V_{DD} \cdot I_{leak} \tag{2.14}$$

As we can see, reducing V_{DD} has a quadratic effect on P_{dyn} which has been mentioned as the major part of the power consumption. For this reason, minimizing the supply voltage has the highest priority in the power optimization process. However, reducing the supply voltage increases circuit delays, especially as V_{DD} approaches the threshold voltage. This slows down the working speed significantly and increases the short circuit power. However, if the supply voltage is substantially higher than the threshold voltage, there is no need to worry about it. Another way to minimize the power consumption is to reduce the effective capacitance. This can be achieved by reducing both of its components: the switching activity and the physical capacitance. As most of the capacitance of the circuit is owing to the transistor capacitance, it makes sense to keep the transistors to a minimum size whenever possible when designing for low power. A reduction of the switching activity $f_{0\to1}$ is also useful to lower the power consumption.

Short circuit power dissipation can be also reduced by decreasing the supply

voltage. By matching the rise/fall times of input and output signals, short circuit dissipation can be minimized. Similar to the situation of reducing dynamic power consumption, decreasing the switching activity $f_{0\to1}$ can reduce the short circuit power consumption as well.

As aforementioned, leakage power is small compared with dynamic power and short circuit power consumption. However, as process feature scales down, leakage power increases significantly. The proposed asynchronous SRAM is designed for the application which is in idle state most of the time. When implemented in advanced technology nodes, e.g. 0.13 um and beyond, the leakage current can account for substantially large portion of power consumption. Therefore, the CMOS 0.35um process which has a small leakage current is chosen to implement the design.

2.2 Review of asynchronous circuits design

In this section, some basic concepts of asynchronous circuits followed by the review of related asynchronous circuits will be presented.

2.2.1 Fundamentals of asynchronous circuits

In asynchronous circuits, handshake protocols are used to take the place of the clock signal to perform the communication and synchronization. The four most common handshake protocols which are 4-phase dual-rail protocol, 2-phase dual-rail protocol, 4-phase bundled-data protocol and 2-phase bundled-data protocol [3] are introduced in this section.

Dual-rail code requires two signals per bit of information d: one signal $d.t$ indicates a true value and one signal $d.f$ indicates a false value. The $\{d.t, d.f\}$ wire pair is a codeword, as shown in Table 2.2, $\{d.t, d.f\}$ = { 1, 0 } and $\{d.t, d.f\}$ = { 0, 1 } represents "valid data" which are logic 1 and logic 0 respectively; $\{d.t, d.f\}$ = { 0, 0 } represents "no data" or the empty state. $\{d.t, d.f\}$ = { 1, 1 } is not used and transitions between valid codewords are not allowed, which is as illustrated in Figure 2.4.

Table 2.2 Transition states of 4-phase dual-rail protocol

	$d.t$ $d.f$
Empty ("E")	0 0
Valid "0"	0 1
Valid "1"	1 0
Not used	1 1

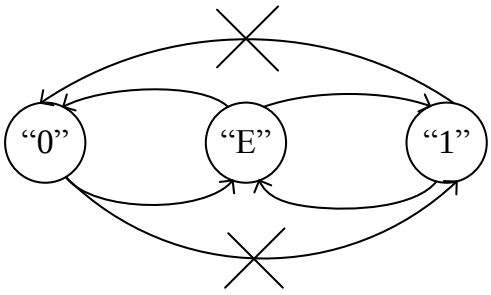


Figure 2.4 Transition diagram of 4-phase dual-rail protocol

The term 4-phase refers to the number of communication actions. As shown in Figure 2.5, the communication cycle works as follows: First, the sender issues a valid codeword, then the receiver absorbs the codeword and set acknowledge high, afterward the sender responds by issuing the empty word, and the receiver acknowledges this by taking acknowledge low. The sender now can issue the next communication cycle.

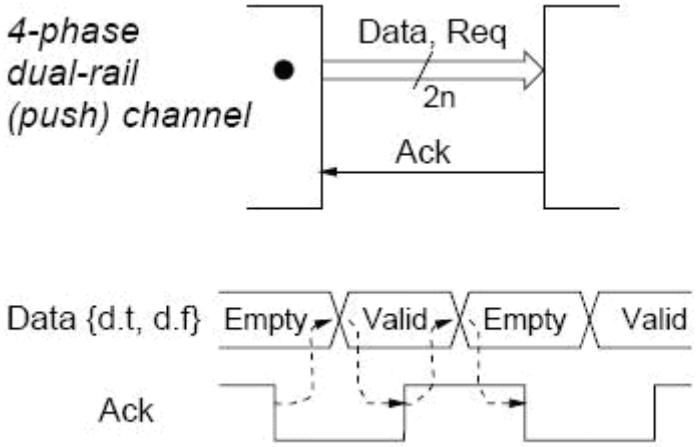


Figure 2.5 A delay-insensitive channel using the 4-phase dual-rail protocol

Compared with 4-phase dual-rail handshaking protocol, the 2-phase dual-rail handshaking protocol also uses 2 wires $\{d.t, d.f\}$ per bit but uses signal transitions to indicate the information. There is no difference between a $0 \rightarrow 1$ and a $1 \rightarrow 0$ transition, they both represent a "signal event". The 2-phase dual-rail handshaking protocol is shown in Figure 2.6.

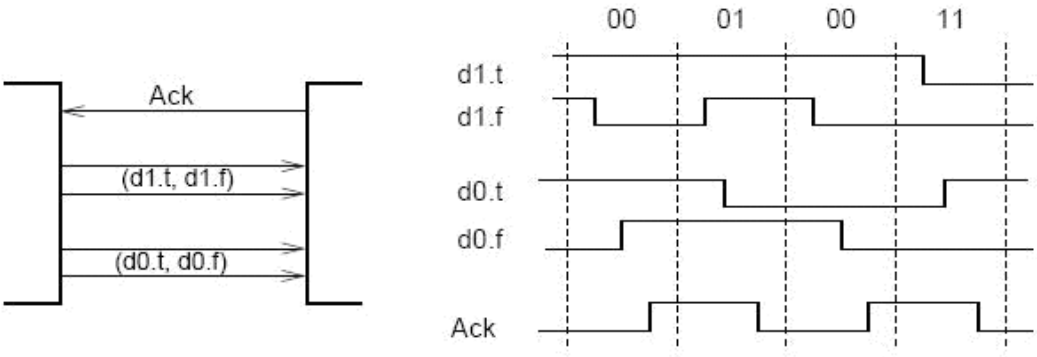


Figure 2.6 Illustration of the handshaking on a 2-phase dual-rail channel

Bundled-data refers to that separate request acknowledge signal wires are bundled with the data signals, which is shown in Figure 2.7 (a). The 4-phase bundled-data protocol is familiar to most digital designers: the communication cycle starts with the sender issuing data and setting the request high, then the receiver absorbs the data and sets acknowledge high, after that the sender responds by taking request low, and the receiver acknowledges this by taking acknowledge low. Now the communication cycle ends and the sender can initiate the next communication cycle. However, this protocol has a disadvantage that superfluous return to zero transitions costs unnecessary time and power. Illustrated in Figure 2.7 (c), the 2-phase bundled-data protocol using the information on the request and acknowledge wires encoded as transitions avoids this. Ideally the 2-phase bundled-data protocol should lead to faster circuits than 4-phase bundled-data protocol. However, there is no general answer as to which protocol is better due to the complex implementation of circuits.

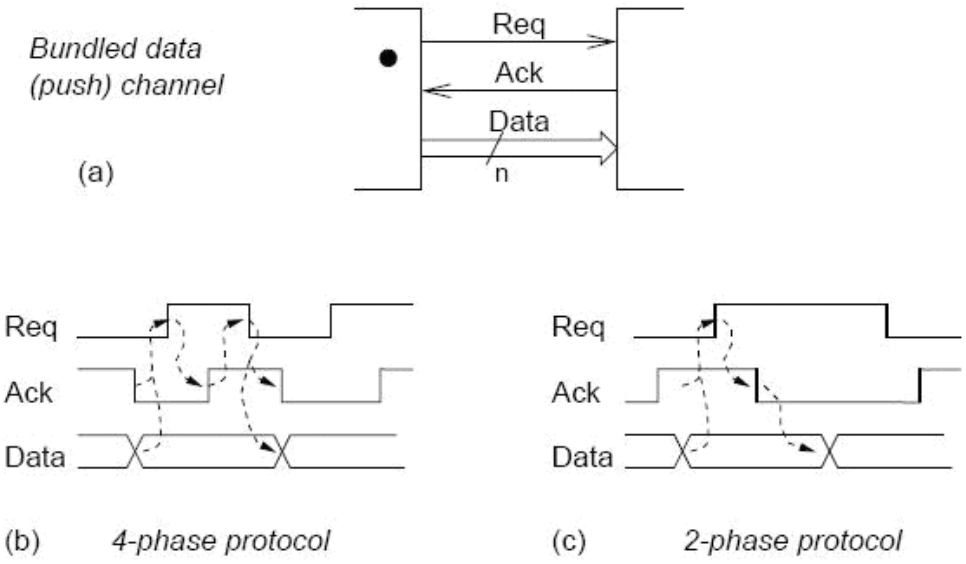
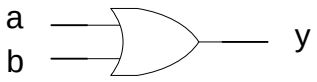


Figure 2.7 (a) A bundled-data channel. (b) A 4-phase bundled-data protocol. (c) A 2-phase bundled-data protocol

The Muller C-element is a fundamental component that is widely used in asynchronous circuits. Consider the simple 2-input OR gate which is shown in Figure 2.8. If the output changes from 1 to 0, it is known that both inputs are now at 0. If the output changes from 0 to 1, it is known that at least one input is 1 but it is not clear that which input is 1. Therefore it can be seen that the OR gate only indicates or acknowledges when both inputs are 0.



a	b	y
0	0	0
0	1	1
1	0	1
1	1	1

Figure 2.8 A normal OR gate and its truth table

Similarly, the AND gate only indicates when both inputs are 1. Similar with an asynchronous set-reset latch, the Muller C-element which is shown in Figure 2.9 is a state-holding element. As illustrated in Table 2.3, when the inputs are both 0's or both 1's, the output is set to 0 or 1. For other input combinations the output does not change. There are also some alternative specifications to describe the Muller C-element, such as if $a = b$, then $y := a$, or $a = b \mapsto y := a$; in Boolean logic, it can be expressed as $y = ab + y(a + b)$ [3].

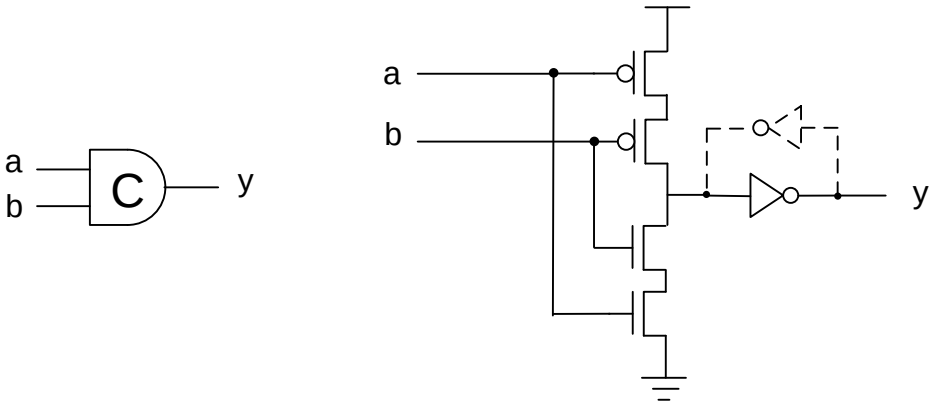


Figure 2.9 The symbol and possible implementation of the Muller C-element

Table 2.3 Truth table of the Muller C-element

a	**b**	**y**
0	0	0
0	1	no change
1	0	no change
1	1	1

2.2.2 Review of recent asynchronous circuits designs

Over the past few decades, many researchers have been working on the asynchronous circuits design [4][5][6][7]. Although there are many methods to design an asynchronous circuit, the controlling scheme only differ in dual-rail encoding or bundled-data encoding, with either transition sensitive or level sensitive signals. In general, there are three most practical circuit implementation styles that are 4-phase bundled-data, 2-phase bundled-data and 4-phase dual-rail which will be introduced as follows:

Firstly, a simple 4-phase bundled-data pipeline without data processing is shown in Figure 2.10 (a), and how combinational circuits or functional blocks can be inserted between the latches is shown in Figure 2.10 (b). Matching delays have to be inserted in the request signal to maintain correct operations. The circuit can be viewed as a traditional “synchronous” data-path consisting of latches and combinational circuits clocked by a distributed gated clock driver. Or it can be seen as an asynchronous data-flow structure composed of two types of handshake components which are latches and function blocks. Although the pipeline implementation is simple, it has some drawbacks: one is that only every other latch is storing data when the C-elements’ state is (0, 1, 0, 1, etc.); another one is speed: the throughput of a pipeline depends on the handshake cycle completion time and this involves communication with both neighbors.

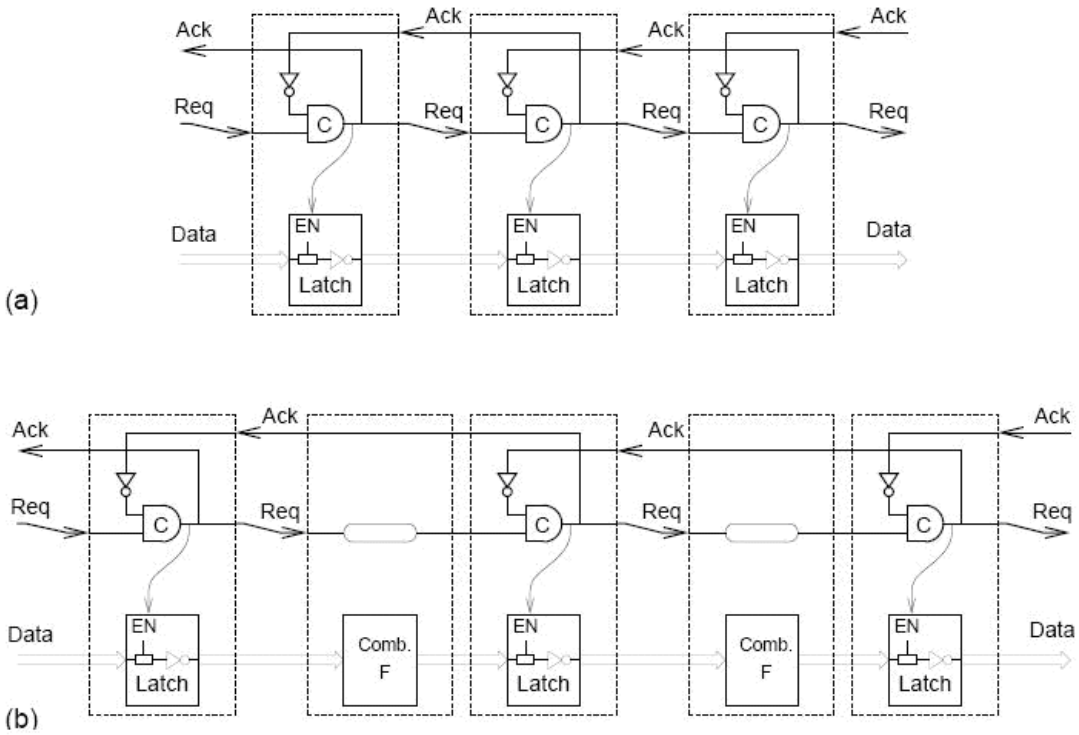


Figure 2.10 A simple 4-phase bundled-data pipeline

Secondly, a 2-phase bundled-data pipeline is shown in Figure 2.11. It also uses a Muller pipeline as the backbone control circuit but the control signals are interpreted as events or transitions. Therefore special capture-pass latches need to be designed: events on the C and P inputs alternate cause the latch to alternate between capture mode and hold mode. Compared to the 4-phase bundled-data approach, the 2-phase bundled-data approach avoids the power and performance loss caused by the return-to-zero part of the handshaking and it is elegant and efficient at the conceptual level. However, the implementation of components that respond to signal transitions is often more complex than the implementation of components that respond to signal levels. For example, the special latch design and the storage elements design are both complicated. To the system with unconditional

data-flows and high speed requirements, the 2-phase bundled-data approach may be the preferred solution. But the price is larger silicon area and higher power consumption; there is no difference between asynchronous design and synchronous design at this point.

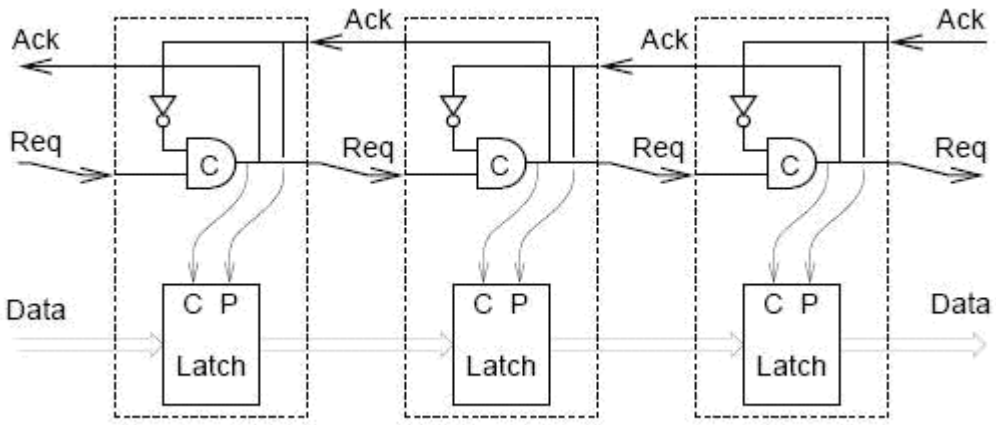


Figure 2.11 A simple 2-phase bundled-data pipeline

Lastly, a 4-phase dual-rail pipeline is introduced. It is also based on Muller pipeline but it combines encoding of data and request in a more elaborate way. The implementation of a 3-stage 1-bit wide pipeline without data processing is shown in Figure 2.12. It can be seen as two Muller pipelines connected in parallel using a common acknowledge signal per stage to synchronize operation. The pair of Muller C-elements in a pipeline stage can store one of the two valid codewords {0, 1} and {1, 0} and causes the acknowledge signal out of that stage to be logic 1, or store the empty codeword $\{d.t, d.f\}$ = {0, 0} and causes the acknowledge signal out of that stage to be logic 0.

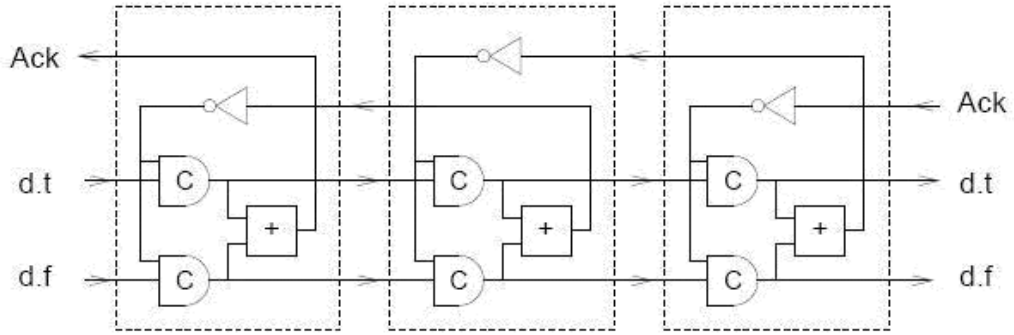


Figure 2.12 A simple 3-stage 1-bit wide 4-phase dual-rail pipeline

An N-bit wide pipeline can be implemented by using a number of parallel 1-bit pipelines. Although to a receiver this does not guarantee all bits in a word arrive at the same time, often the necessary synchronization is done in the functional blocks. The individual acknowledge signals can be combined into one global acknowledge signal using a Muller C-element if bit-parallel synchronization is needed. An N-bit latch is shown in Figure 2.13. A completion detector that indicates whether the N-bit dual-rail codeword stored in the latch is empty or valid is formed by the OR gates and the Muller C-element. An implementation of a completion detector using only one 2-input Muller C-element is also shown in Figure 2.13.

As mentioned in Section 1.3, combinational circuits must be transparent to the handshaking between latches. The sending and receiving latches rely on the acknowledge signal to identify whether the codeword has been received or ready to absorb. All outputs of a combinational circuit must not become valid/ empty until all inputs have already become valid/ empty. Otherwise, the receiving latch

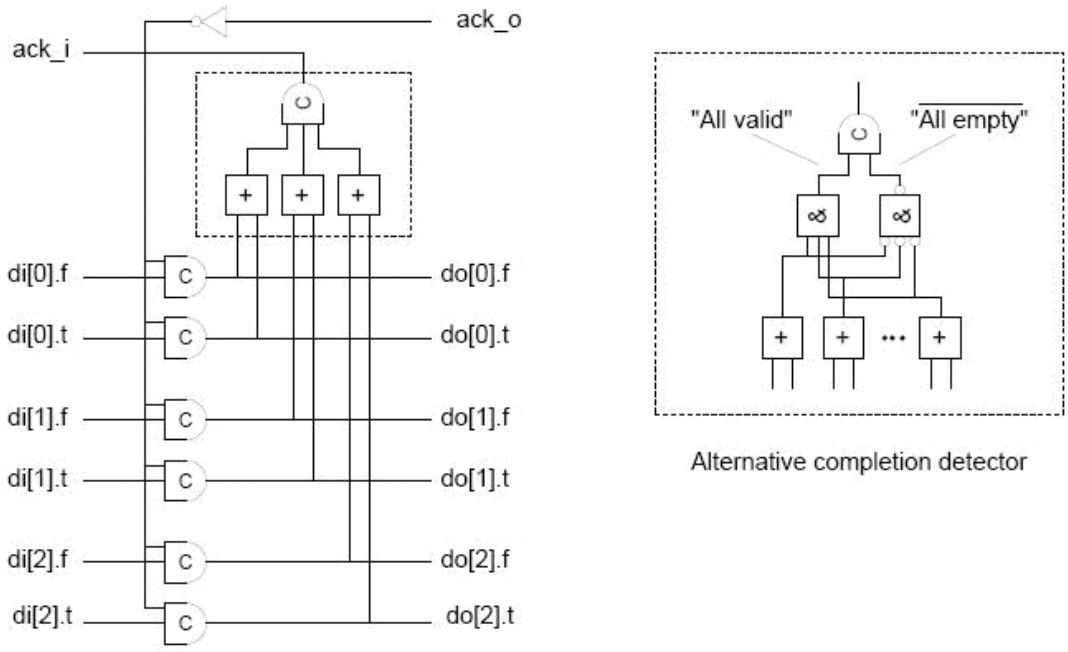


Figure 2.13 An N-bit latch with completion detection

may prematurely set acknowledge signal high/ low before all signals from the sending latch have become valid/ empty. As a result, a 4-phase dual-rail hand-shaking combinational circuit consists of state holding elements and it represents a hysteresis-like behavior in the empty-to-valid and valid-to-empty transitions.

The implementation of a dual-rail AND gate which only uses Muller C-elements and OR gates is shown in Figure 2.14. When the inputs a and b are both logic 1 which means $\{a.t, a.f\}$ and $\{b.t, b.f\}$ are both {1, 0}, the output $\{y.t, y.f\}$ then becomes {1, 0} which also stands for logic 1. When all inputs become empty, the Muller C-elements are all set low, the output then become empty again. It should be mentioned that the Muller C-elements provide both the necessary 'and' operator and the hysteresis in the empty-to-valid and valid-to-empty transitions which are required for transparent handshaking. Other

dual-rail gates such as OR and XOR gates can be implemented in a similar way. When composing gates into larger combinational circuits, the transparency to handshaking is a property that basic gates preserve. Given these basic dual-rail gates, the dual-rail combinational circuits can be built using normal combinational circuit synthesis techniques.

Table 2.4 Truth table of a dual-rail AND gate

a	**b**	**y.f**	**y.t**
E	E	0	0
		No Change	
F	F	1	0
F	T	1	0
T	F	1	0
T	T	0	1

E = empty state, T = true state, F = false state

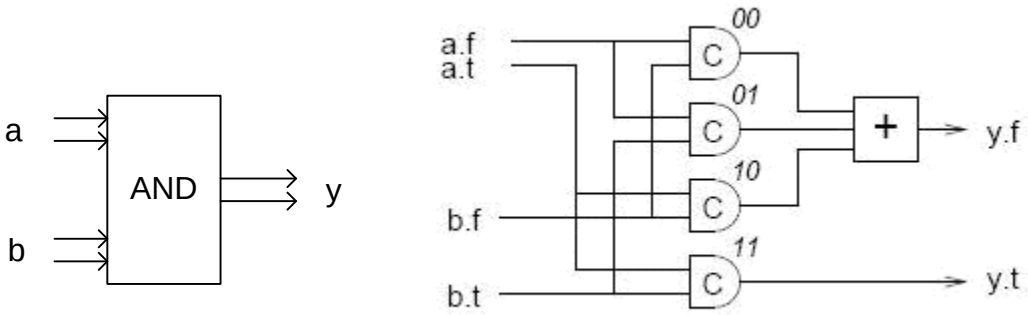


Figure 2.14 The symbol and implementation of a 4-phase dual-rail AND gate

The asynchronous circuits can be classified as being self-timed, speed-independent or delay-insensitive at the gate level depending on the delay assumptions [3]. Gates A, B, and C which are shown in Figure 2.15 are used for the following discussion. The output of gate A forks to inputs of gates B and C. A speed-independent (SI) circuit is a circuit that operates correctly assuming positive, bounded but unknown delay in gates and ideal zero-delay wires. Referring to Figure 2.15 it means d_A, d_B, d_C are arbitrary delays but d1 = d2 = d3 = 0.

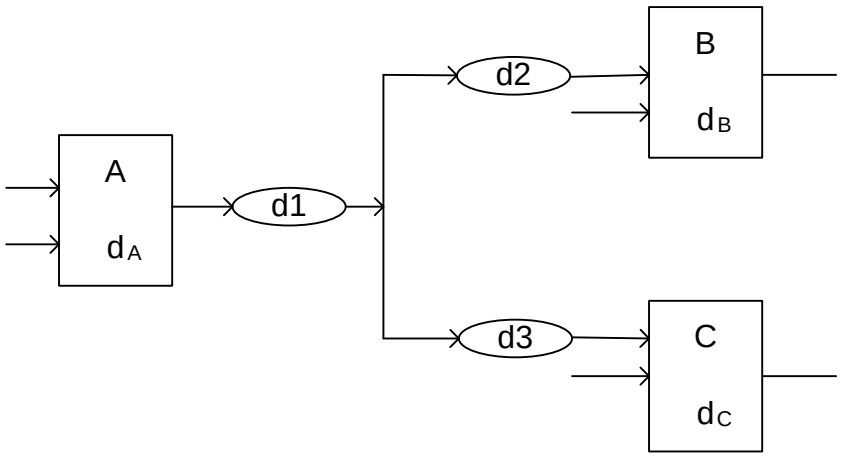


Figure 2.15 A circuit fragment with gate and wire delays

It is not realistic to assume that the wire delay is zero in today's semiconductor processes. However, by allowing arbitrary d1 and d2 and requiring d2 = d3, the wire delays can be lumped into the gates. From a theoretical view, the circuit is still speed-independent.

A delay-insensitive (DI) circuit is an extremely robust circuit that operates correctly assuming positive, bounded but unknown delays in wires and gates. Unfortunately only the circuits composed of Muller C-elements and inverters can be delay-insensitive. And this class of DI circuits is very small, most circuits referred to in the literature as delay-insensitive are only quasi-delay-insensitive (QDI). This is another kind of circuit that are delay-insensitive with the exception of some carefully indentified wire forks where d2 = d3.

A self-timed circuit is a circuit whose correct operation relies on more elaborate and engineering timing assumptions. The different circuit classes which are DI, QDI, SI and self-timed are not mutually exclusive to build complete systems. They can be used at different levels of design in most practical design. For instance, in Amulet processors [8] SI is used for local asynchronous controllers, bundled-data for local data processing and DI is used for high-level composition. Another example is the hearing-aid filter bank design [9]. The DI dual-rail 4-phase protocol inside RAM modules and arithmetic circuits is used to provide robust completion indication; 4-phase bundled data with SI control is used at the top level of design. This underlines that which handshake protocol and circuit implementation style to choose is just one factor to consider when optimizing an

asynchronous digital system.

Various types of asynchronous processors have been developed during the past decades. For example, a CMOS self-timed arithmetic logic unit as part of the ARM microprocessor had been developed by Garside [10]. Also, a CMOS locally clocked sequential microprocessor had been developed by Muscato and Albicki [11]. Moreover, a 100-MIPS GaAs asynchronous microprocessor had been implemented by Tierno and Martin [12]. In addition, Nielsen and Sparso had implemented an IFIR Filter Bank for a Digital Hearing Aid. These designs cannot function without the asynchronous SRAM. As a result, there is an increasing demand for asynchronous memory designs.

In recent decades, many asynchronous SRAMs have been designed. A 64 by 64 bit self-timed static RAM had been fabricated and tested by Edward H. Frank [13]. Although a conventional six-transistor static SRAM cell was used, extra circuit was associated with each row and column of the memory array to make the memory self-timed and the four-phase request/acknowledge interface was applied. To measure the performance of an asynchronous circuit, Jose Tierno and Alain Martin introduced the concept of energy per operation in [14]. They analyzed the sources of power dissipation of the memory. Based on the high-level language specification, they proposed an energy consumption model which is independent of voltage and timing considerations. They proposed a new way to design low-energy asynchronous memory. Meanwhile some other researchers were designing the asynchronous SRAM in a different way, using other kinds of process.

For example, an experimental 1kb GaAs MESFET Static RAM had been designed and fabricated by Ajay Chandna and Richard Brown [15]. This work used a new current mirror memory cell and was not subject to the destructive read problems that constrained the design of the conventional six-transistor memory cell. Using the new memory cell maximized the number of bits allowed per bit line as this led to the biasing arrangement which minimized the leakage currents associated with the bit lines' unselected bits. The design and formal verification of a self-timed static RAM was proposed by Lars Nielsen and Jorgen Staunstrup [16]. This memory was designed for robust operation at a wide range of supply voltage but intended for relatively small specialized SRAM array. The work emphasized on the formal verification of speed-independent. A four-phase handshaking asynchronous static RAM was proposed by Vincent Wing-Yun Sit, Chiu-Sing Choy and Cheong-Fat Chan [17]. This design which used four-phase bundled data handshake protocol was speed-insensitive and used in self-timed systems. The work used the dual-rail voltage-sensing completion detection (DVSCD) scheme to generate the read completion signal and multiple delays completion generation (MDCG) to generate the write completion signal. An asynchronous dual-port 1 MB CMOS SRAM was designed by Tan Soon-Hwei, Loh Poh-Yee and Mohd-Shahiman Sulaiman [18]. The memory array was organized in 64 k words by 16 bits. A fast 128k-bit asynchronous SRAM using a radiation hardened CMOS/SOI process was presented by Zhou Kai, *et al.* [19].

CHAPTER 3

ASYNCHRONOUS SRAM DESIGN

3.1 Introduction

In this chapter, the design of the asynchronous SRAM will be presented. The basic SRAM structure has been shown in Figure 1.1 and the introduction of how the SRAM works has been presented in Section 1.1. In this Chapter, the timing sequence and the design of the self-timed SRAM will be presented. This chapter is organized into four main sections. The comparison of conventional six-transistor SRAM cell and self-timed SRAM cell is introduced in section 3.2. In section 3.3, the specification of the SRAM module is presented. Section 3.4 presents the design of the self-timed SRAM including SRAM cell, control part, acknowledge part and data-path. The timing sequence of each part is also presented.

3.2 Self-timed SRAM cell

Prior to presenting the self-timed SRAM cell, a standard six-transistor SRAM cell with precharge transistors is introduced. As shown in Figure 3.1, two cross coupled inverters that hold the present value of the cell are formed by the four transistors which are at the center of the SRAM cell. The two inverters are connected by the two pass transistors to the bit lines, Bit and $\overline{Bit}$. During either a write or a read operation, the two pass transistors are activated. The precharge transistors are located at the top of the bit lines and they charge the bit lines to logic 1 before either write or read operation takes place. The timing diagrams of

how the write and read operations perform are shown in Figure 3.2. Write and read operations from an array of SRAM cells follow the same protocol.

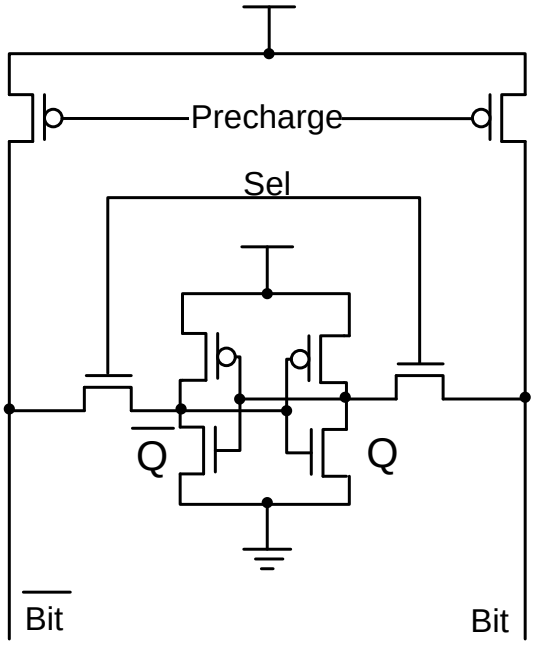


Figure 3.1 A standard six-transistor SRAM cell with precharge transistors

It can be seen when referring to Figure 3.2 that a read operation consists of two self-timed cycles: an evaluation cycle when the data is read and a precharge cycle when the SRAM returns to the initial state [20]. The bit lines serve as outputs and the completion of operation is indicated on these lines. For example, an AND gate can be directly applied to the bit line pair to produce the completion signal. However, during the write operation, the bit lines serve as inputs controlled by the input line driver. Therefore, it can only be known when the line driver has been changed when observing the bit lines state. When the state of the SRAM cell has been changed is unknown. And it is impossible to tell when the write operation has ended. An additional port is needed to observe the state of the SRAM cell

to make the SRAM completely self-timed.

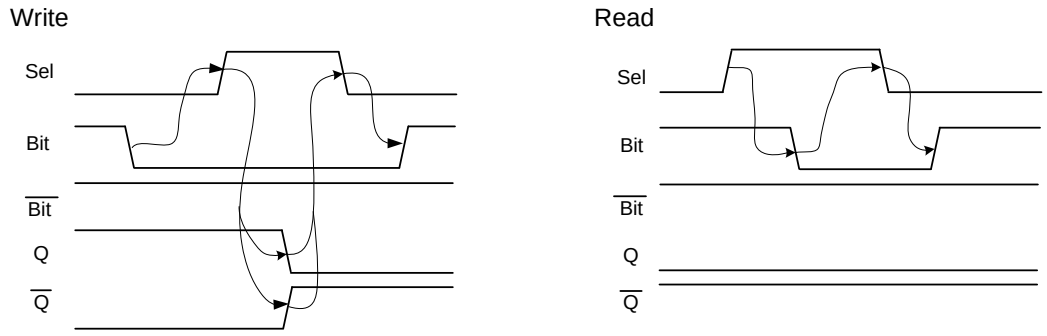


Figure 3.2 Timing diagram of SRAM write and read operations

The proposed asynchronous SRAM uses four-phase handshake protocol and dual-rail data encoding. As presented in Section 2.2, this code requires two signals per data bit $\{d.t, d.f\}$, which indicate a true value and a false value respectively. Dual-port SRAM cell is introduced here. Figure 3.3 shows two dual-port SRAM cells. The one on the left side is a traditional dual-port SRAM using pass transistors to connect the memory cell to the bit lines. The one on the right side does not use pass transistors. The bit lines in the right cell are dedicated for either a write or read operation which is unlike the standard dual-port cell. Since all transitions of both a write and read operation are indicated on the output bit lines, the proposed SRAM cell is referred to as a self-timed SRAM cell.

3.3 Specification of SRAM module

The interface of the SRAM module is shown in Figure 3.4. The SRAM interface consists of three control signals: *Read*, *Write* and *ARW_ack*; one address bus: *Address*; two dual-rail data buses: *Din* and *Dout*. The four-phase dual-rail

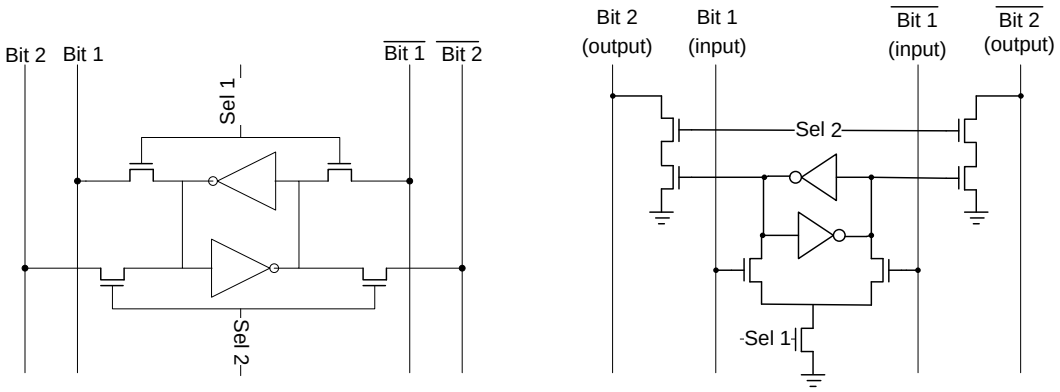


Figure 3.3 A standard dual-port SRAM cell and a self-timed SRAM cell

protocol is used to carry out the communication between the SRAM and the peripheral circuits. The input control signals are *Read* and *Write* which control whether to read from or write into the memory. The pairs of the *Read* and *Write* signal also form a read/write dual-rail signal since read and write operations are not allowed to perform simultaneously in this design. The only output control signal is the global acknowledge signal *ARW_ack* which indicates the end of each write or read operation. All the signals change to the valid state then return to the empty state in every read and write operation. This suffices the requirement of four-phase dual-rail protocol which every valid state should be isolated by the empty state.

Write and read operations

The write or read operation of the proposed SRAM is similar to the typical synchronous SRAM on the timing sequence. In general, during write operation, to prevent the wrong data from being written into the memory cell or the data from being written into the wrong address, the input data and address should be valid

before the write signal arrives and last a little longer than the write signal exists.

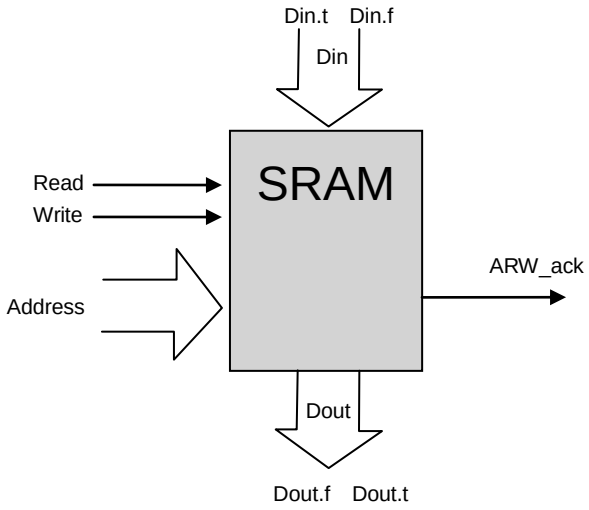


Figure 3.4 Specification of SRAM module

During read operation, the address signal should also be valid on the address bus before the read signal arrives. The typical write and read operations are described as follows. When the input data bus *Din* and the address bus *Address* have the valid value, the write operation begins with the write signal set high, the data on *Din* is written into the memory cell and then the acknowledge is set high. When the write signal becomes low, the address is removed and the input data bus returns to empty state, the write operation ends with all the signals returning to empty state, and the handshake is completed. Similarly, when the address bus has a valid address, the read operation starts with the read signal set high. The content of the addressed memory cell is written into the output data bus *Dout*, and the acknowledge signal is set high. When the read signal becomes low and the address is removed, the read operation ends with all signals returning to empty state.

3.4 Self-timed SRAM design

The structure of the SRAM design is divided into 4 parts which are SRAM cell, control part, acknowledge part and data path, which is shown in Figure 3.5. Each part will be elaborated in the following parts.

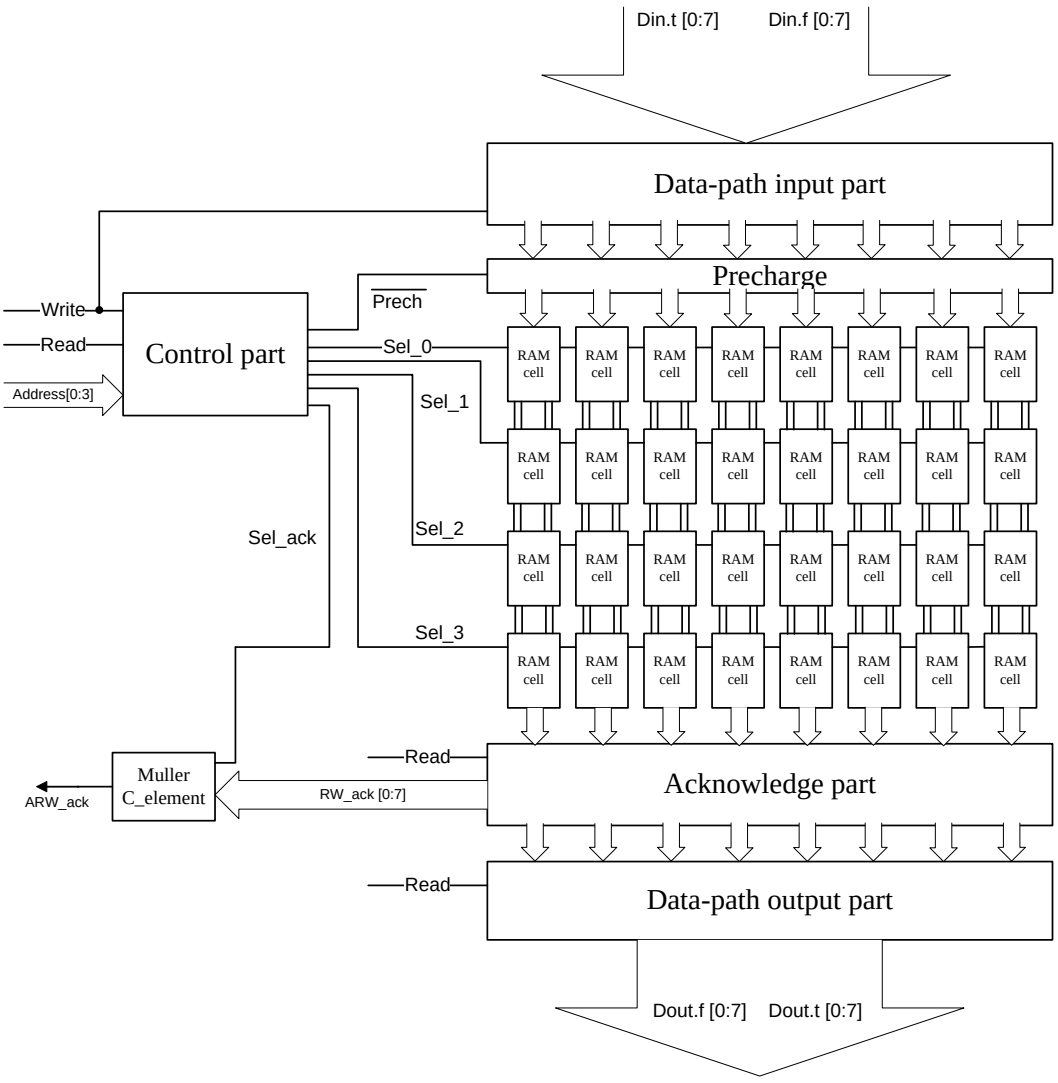


Figure 3.5 Block diagram of the 4*8 bits asynchronous SRAM

3.4.1 SRAM cell

The self-timed cell has been introduced in Section 3.2. The proposed SRAM cell is shown in Figure 3.6. The memory core is connected with two pairs of bit lines which are internal input bus $Di.t$ and $Di.f$, internal output bus $\overline{Do.t}$ and $\overline{Do.f}$ respectively. The internal input bus $Di.t$ and $Di.f$ connect the data-path input part which is at the top of the bit lines with the acknowledge part which is at the bottom. The internal output bus $\overline{Do.t}$ and $\overline{Do.f}$ are connected to the pull-up transistors at the top of the bit lines and the feedback inverter at the bottom.

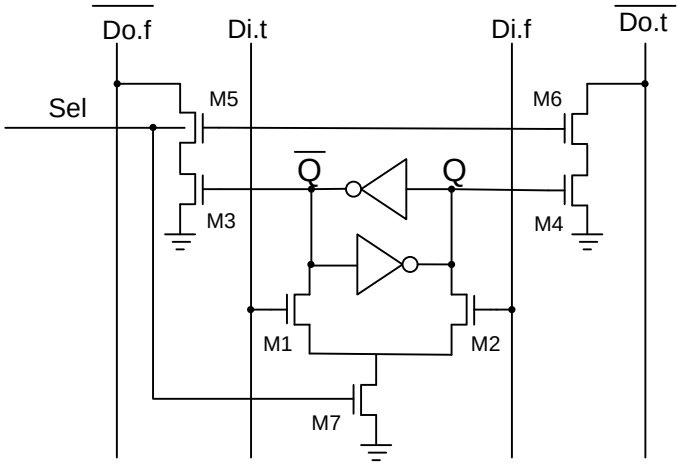


Figure 3.6 Transistor diagram of the self-timed SRAM cell

During the write operation, the data is written into the memory cell through the internal input bus $Di.t$ and $Di.f$. When the data value has been written into the memory cell, one of the internal output bus $\overline{Do.t}$ or $\overline{Do.f}$ will be pulled down to GND and the acknowledge signal will be generated by the acknowledge

part located at the bottom of the internal output bit lines. For example, the data is logic 1 and encoded as { $Di.t$, $Di.f$ } = { 1, 0 }. Then the transistor M1 is conducted and node $\overline{Q}$ is pull down to GND. Meanwhile, node Q is pull up to V_{DD} which turns on transistor M4. Transistor M6 is conducting now as the Select signal is set high, therefore $\overline{Do.t}$ is pulled down to GND. Similarly, during the read operation, one of the internal output bus $\overline{Do.t}$ or $\overline{Do.f}$ will be pulled down to GND and the acknowledge signal will be generated. After the write or read operation, the internal input bus $Di.t$ and $Di.f$ both return to GND while the internal output bus $\overline{Do.t}$ and $\overline{Do.f}$ are both pulled up to V_{DD} causing internal output bus $Do.t$ and $Do.f$ returned to GND.

3.4.2 Control Part

Control part consists of the precharge and row select circuits. For precharge circuit, whenever there is either write or read operation, the precharge signal will be set high. This turns off the precharging and guarantees the operation works well. Before the row select circuit is the N to 2^N address row decoder. When the address bus has a valid value, one row of the SRAM will be selected through the row decoder, then the select signal will be set high, enabling the SRAM cell to perform the write or read operation. To prepare for the next write or read operation, all the input and output signals must return to empty state after the first two phases of either write or read operation. Precharging the internal output data bus in each operation is also included. It is worth noting that precharging and evaluation of the internal output data bus $\overline{Do.t}$ and $\overline{Do.f}$ operating simultaneously causes a

conflict between the precharging pull up transistors and the memory cell pull down transistors, which leads to excessive power dissipation. To avoid such circumstance, the select signal must be deactived before the precharge signal is activated, and vice versa. The typical write and read operation timing diagram of the control part signals is shown in Figure 3.7.

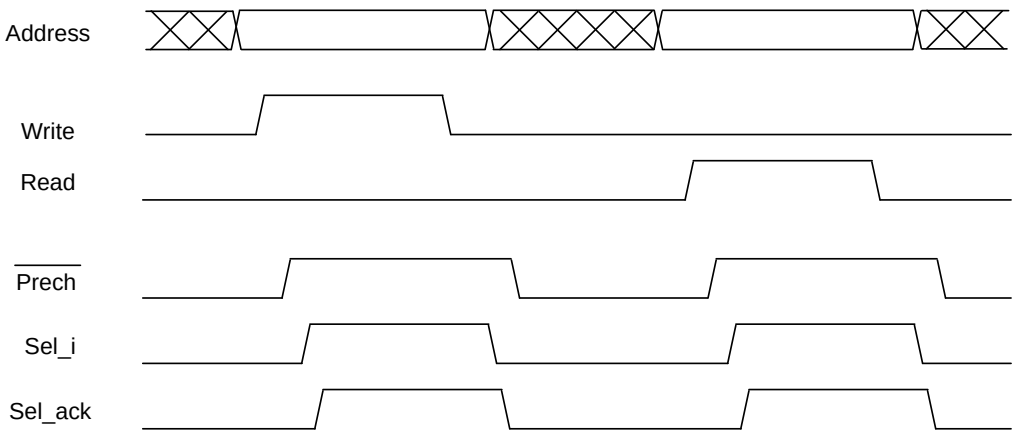


Figure 3.7 Timing diagram of the control part signals

3.4.3 Acknowledge Part

The acknowledge part is composed of select acknowledge circuit and read/write operation acknowledge circuit, which correspond to the control part and data-path respectively. In general, the function of the acknowledge part is that the overall acknowledge signal *ARW_ack* will be set high when the data value has been written into the selected memory cell or the data value of the selected memory cell has been read out; and the signal *ARW_ack* will return to GND when the write or read operation ends. During the write operation, one of the internal input

data bus $Di.t$ or $Di.f$ is pulled up to V_{DD}, the corresponding internal output bus $Do.t$ or $Do.f$ is then pulled up after the data has been written into the memory cell, therefore the signal *RW_ack* is pulled up to V_{DD}. As there are 8 bit memory cells in one row, there are 8 *RW_ack* signals labeled from *RW_ack_0* to *RW_ack_7*. To write different data values into the memory cell leads to different rising time of the *RW_ack* signals. The 8-input Muller C-element is used to collect all the 8 acknowledge signals to form one *RW_ack*. The *RW_ack* signal with the select acknowledge signal *Sel_ack* which is set high during the write operation pass through the Muller C-element, then the overall acknowledge signal *ARW_ack* is set high which notifies the peripheral circuits that the write operation has been finished. The read operation is similar with write operation and the only difference is that the internal data input bus are not changed during the read operation. To insure that the correct *RW_ack* signal is generated, the Read signal is applied to the acknowledge part. After the write/read operation, all the signals return to the empty state: both the *Sel_ack* signal and the four internal data buses $Di.t$ and $Di.f$, $Do.t$ and $Do.f$ are set to zero; RW_ack is then pulled down to GND; *ARW_ack* is set to zero and notify the peripheral circuits that the memory is ready to perform the next write/read operation. The typical write and read operation timing diagram of the acknowledge part signals is shown in Figure 3.8.

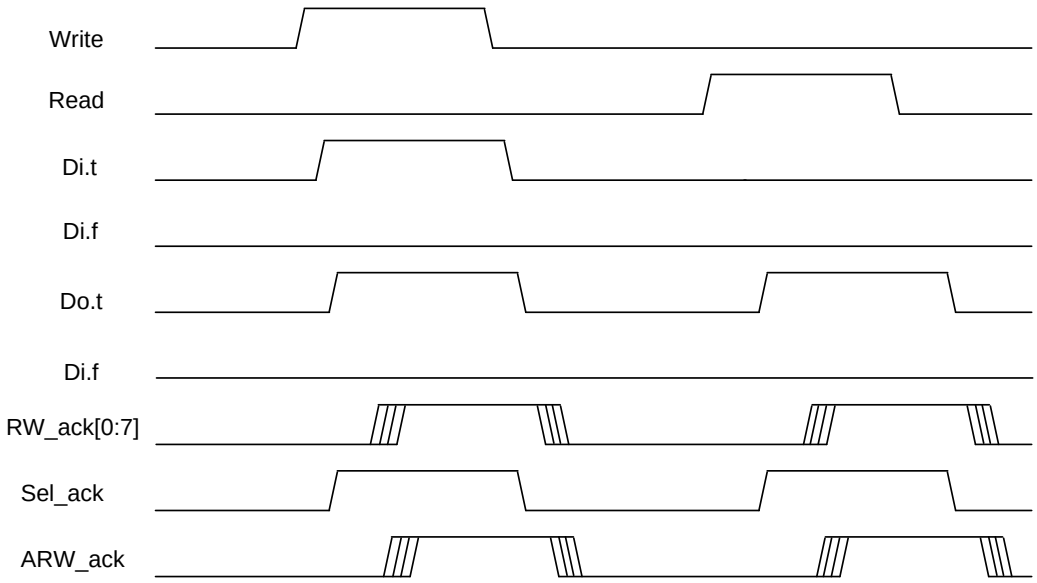


Figure 3.8 Timing diagram of acknowledge part signals

3.4.4 Data Path

Data path consists of the input and output modules, memory cell, internal input and output data buses and the acknowledge module. The input and output modules are used to isolate the SRAM from the peripheral environment and the modules simply consist of Muller C-elements. During the write operation, when the data is written into the memory cell, the internal output data bus are also affected to acknowledge that the write operation has been finished. Therefore the internal output data bus must be isolated with the environment. Combined with the Read signal, Muller C-elements are used here to do the function. Similarly, during a read operation, there can be new data values on the external input data bus, Muller C-elements are used to isolate the internal input data bus with the external input data bus, avoiding the memory data corruption. The typical write and

read operation timing diagram of the data-path signals is shown in Figure 3.9. It has been introduced in Section 3.4.1 that the dual-port SRAM cell is connected with the row select signal and 4 internal input and output data buses $Di.t$ and $Di.f$, $Do.t$ and $Do.f$. One of the internal output data bus is left floating during write/read operation and this causes leakage current due to changing the bus value. To avoid such circumstance, the feedback inverters are brought in to the internal output data bus to ensure the reliable operation of the SRAM. The acknowledge module has been mentioned in Section 3.4.3, while the multiple-input Muller C-element is not included in data-path.

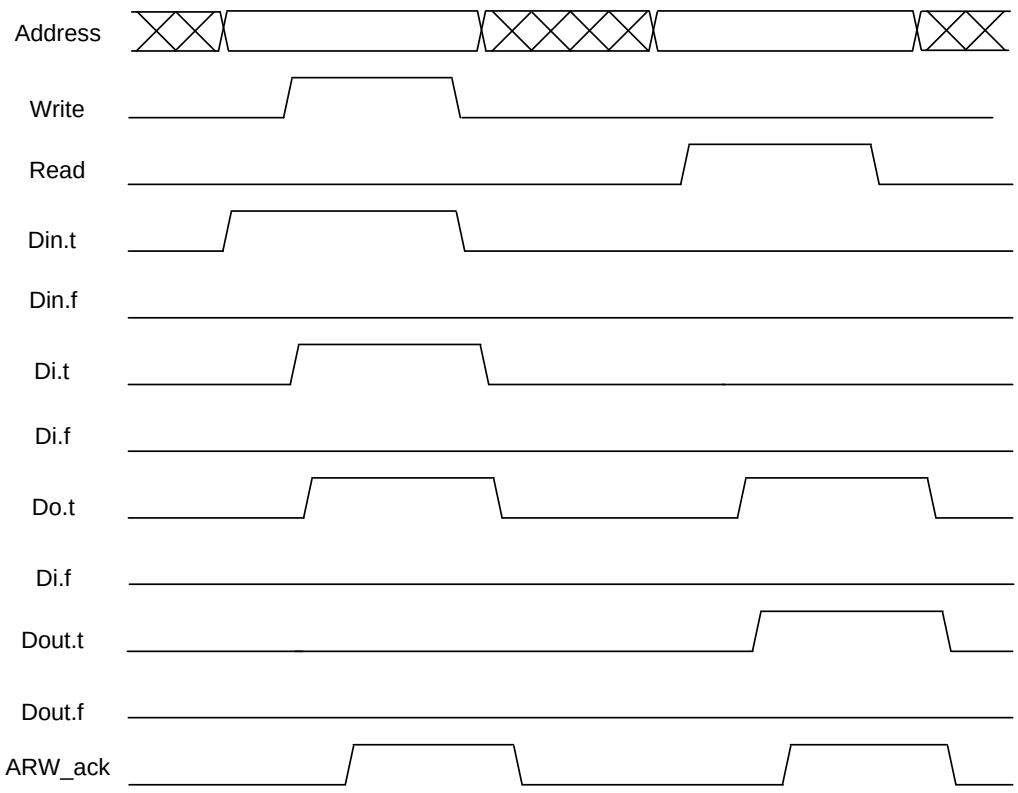


Figure 3.9 Timing diagram of the data-path signals

CHAPTER 4

CIRCUIT LEVEL DESIGN AND SIMULATION RESULTS

4.1 Introduction

This chapter describes the circuit level design and the implementation of the 16*8 bits and the 128*8 bits asynchronous SRAM. The design is using AMS 0.35um double-poly, four-metal CMOS process and operating under 1V-3.3V supply. The design of each part of the asynchronous SRAM has been introduced in Chapter 3. This chapter presents the details of the key circuit blocks design including Muller C-element, precharge and select circuit and acknowledge module. The layouts of some blocks and the simulation results for each individual part are also presented. Some consideration in the layout design is described in Section 4.5. The results of the schematic and post-layout simulation of the design are presented in Section 4.6.

4.2 Muller C-element design

As described in Chapter 2, Muller C-element is a fundamental component widely used in asynchronous circuits design, especially in the design of speed-independent circuits. The schematic and layout of a two-input Muller C-element is shown as Figure 4.1. Balsa is the asynchronous circuit design and synthesis tool used in our project. In Balsa, the majority gate MAJ31 is used to perform the Muller C-element function. However, the gate MAJ31 consumes a large area and power. To pursuit the best tradeoff between power and area, custom

design of the Muller C-element is also needed. Only the width of the gate can be changed due to the fixed gate length in customize design.

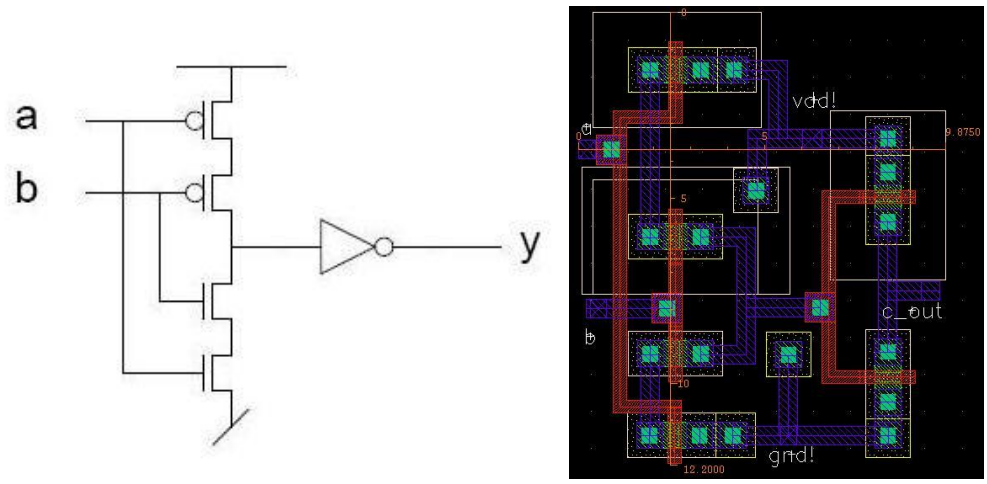
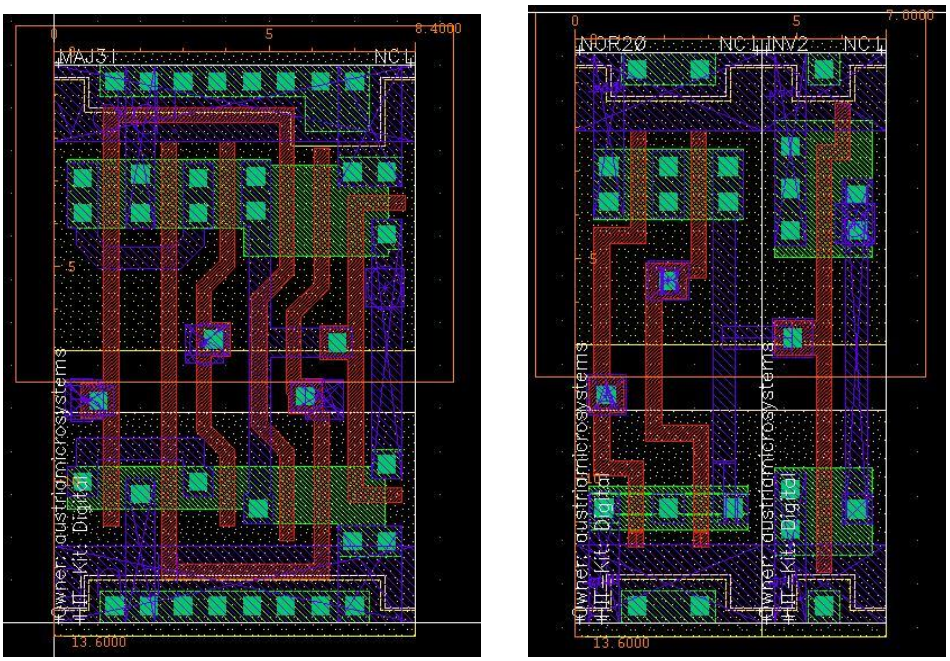


Figure 4.1 Schematic and layout of the 2-input Muller C-element

A NOR gate and an inverter both from the AMS CMOS 0.35um technology are used here to form the Muller C-element. The layouts of gate MAJ31 and the customized design Muller C-element is shown in Figure 4.2 to give a perception of the area of these two gates. The widths of the gate MAJ31 (Figure 4.2 a) and customized Muller C-element (Figure 4.2 b) are 8.4um and 7.0um respectively. Nearly 17% of area is saved when the customized Muller C-element is used to replace the gate MAJ31 which is extensively used in the synthesis of Balsa. The simulation results also show that the power consumption and speed of customized Muller C-element are better than the gate MAJ31. For example, the speed is around 40% higher and the power consumption is 30% lower. The post-layout simulation results of the gate MAJ31 and the customized Muller C-element at 1.5V

are shown in Figure 4.3 and Figure 4.4, respectively.



a **b**

Figure 4.2 Layouts of a) MAJ31 and b) custom Muller C-element

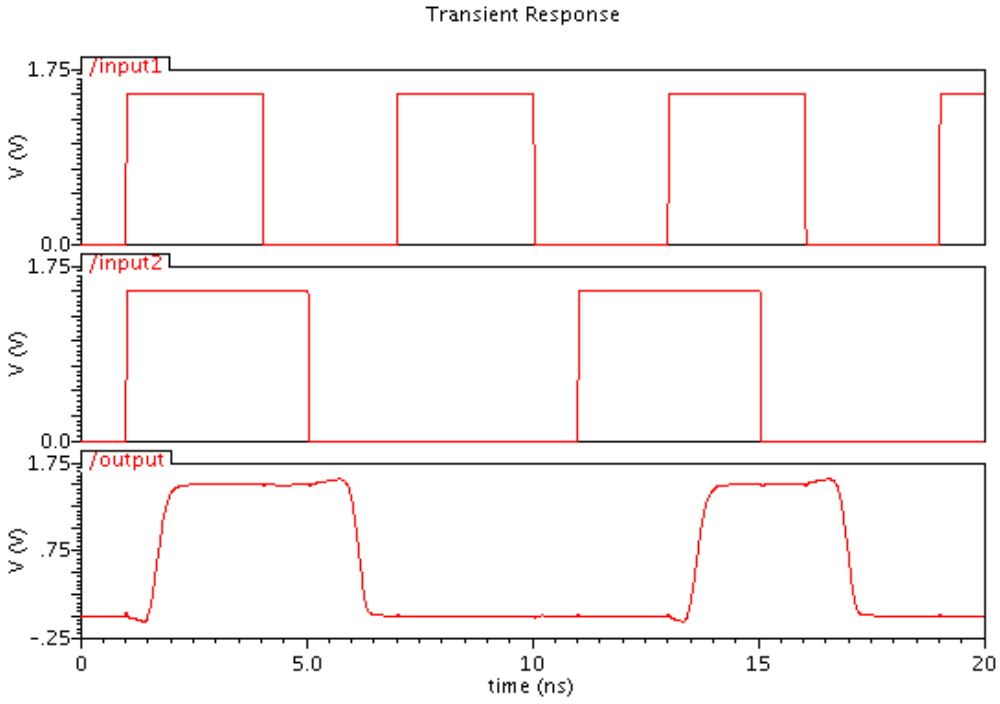


Figure 4.3 Transient response of the gate MAJ31 (Post-layout at 1.5V)

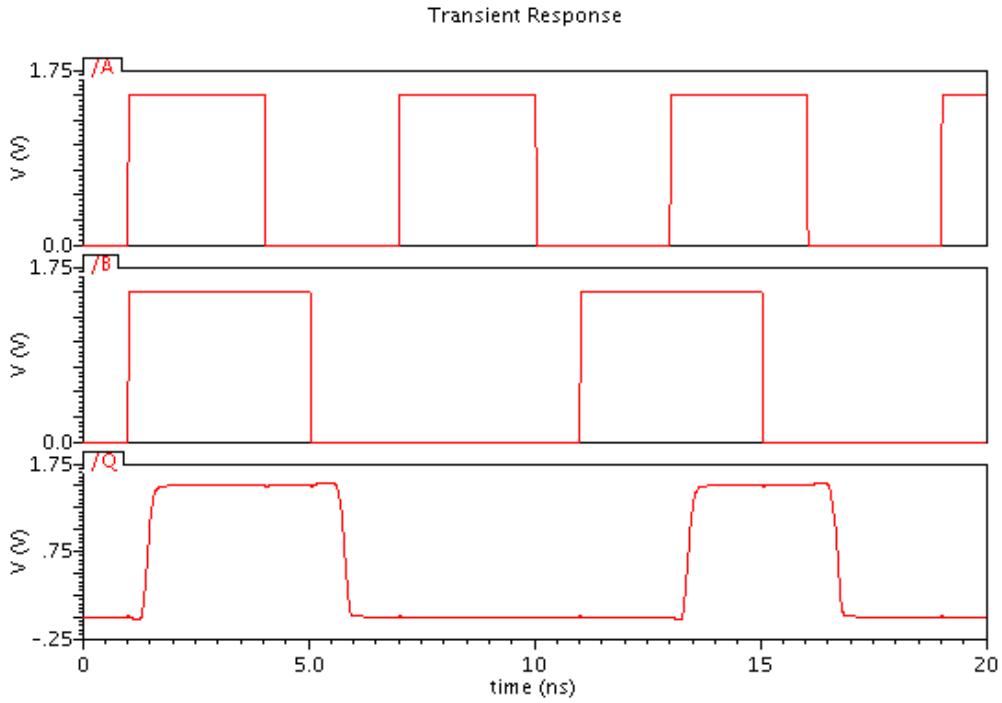


Figure 4.4 Transient response of the customized Muller C-element (Post-layout at 1.5V)

A multiple input Muller C-element has been frequently used for joining signal transitions or completion time detection in self-timed circuits. The asynchronous SRAM is 8 bits in data width, so 8-input Muller C-element implementation is needed. Typical tree structure implementation is applied here. Figure 4.5 shows the implementation and symbol of an 8-input Muller C-element.

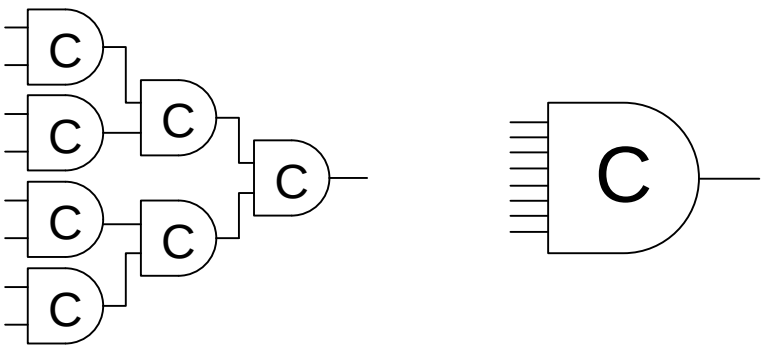


Figure 4.5 Implementation and symbol of the 8-input Muller C-element

The schematic and layout of the 8-input Muller C-element is shown in Figure 4.6. The simulation results of the schematic and post-layout at 1.5V are shown in Figure 4.7.

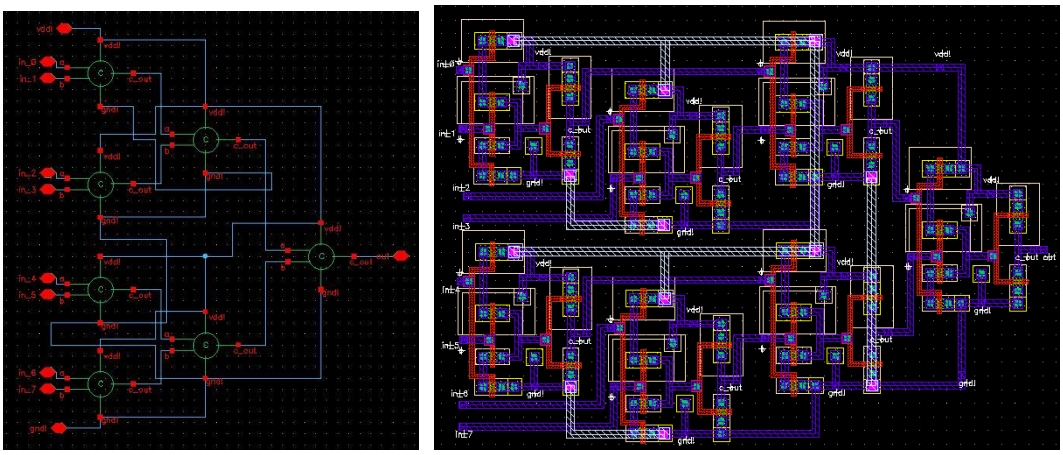


Figure 4.6 Schematic and layout of the 8-input Muller C-element

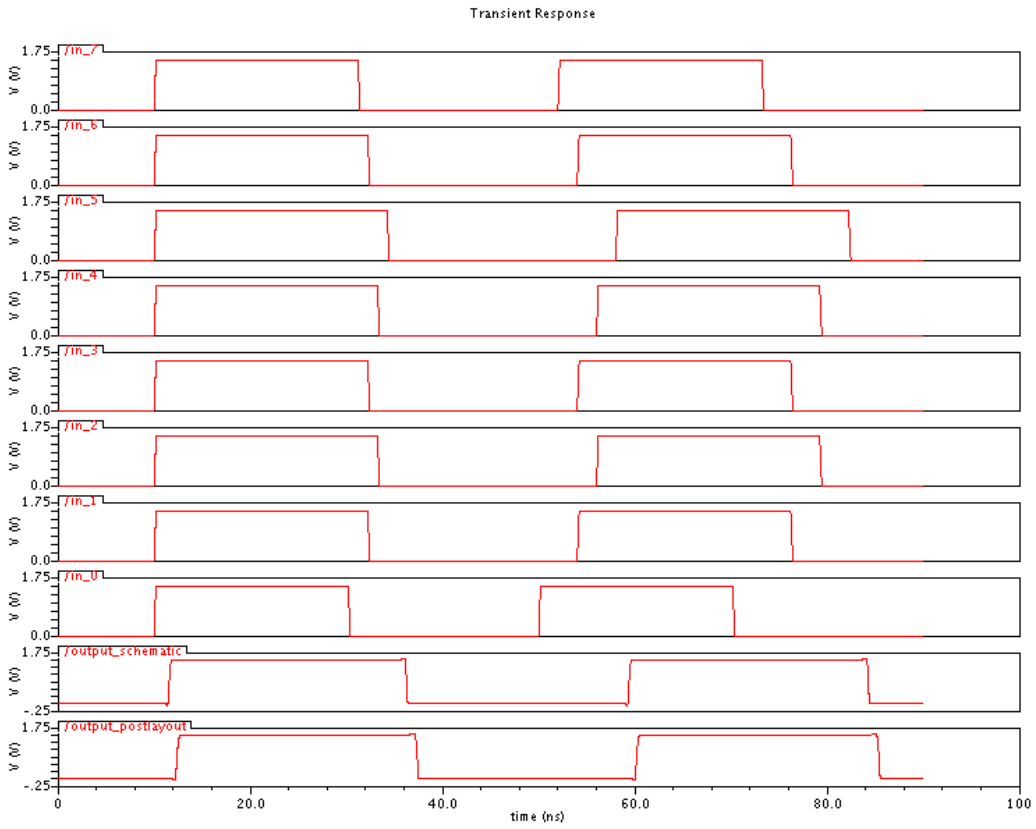


Figure 4.7 Transient response of the 8-input Muller C-element (Schematic and post-layout at 1.5V)

4.3 Precharge and select circuit design

Precharge and select circuit includes prechage control circuit, row decoder, row select and select acknowledge circuit. It belongs to the control part of the asynchronous SRAM. The function of this circuit includes controlling the precharge circuit operation for write and read operation, decoding the address bus and selecting the row, generating the select acknowledge signal for the overall acknowledge signal. The schematic of the precharge and select circuit is shown in Figure 4.8.

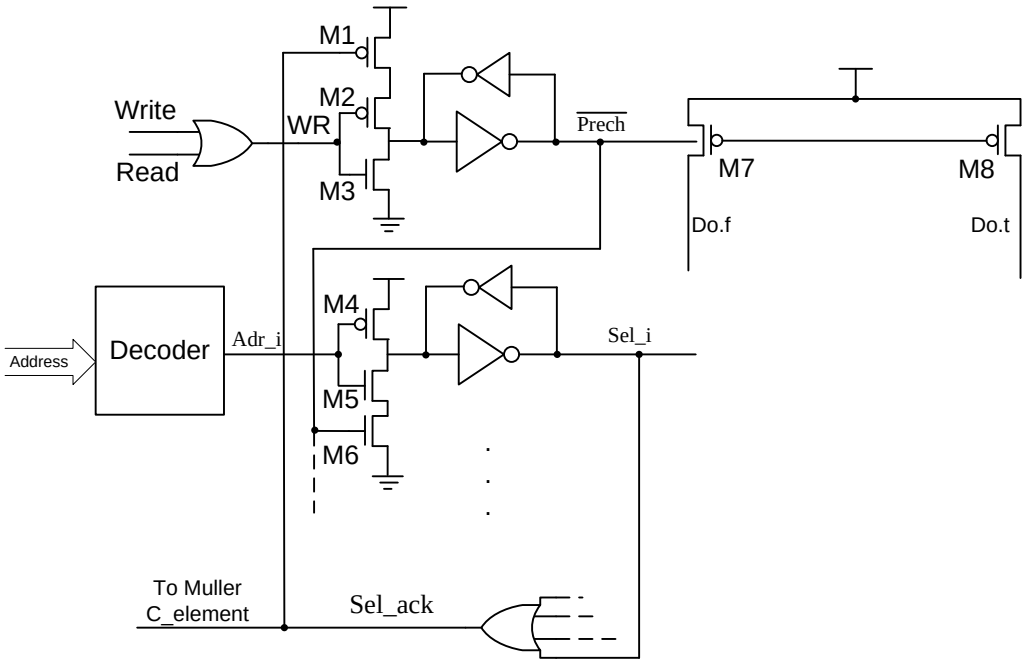


Figure 4.8 Schematic of the precharge and select circuit

When there is no write and read operation, all the signals are low. During either a write or read operation, write or read signal is set high and signal *WR* is set

high, and then the precharge signal $\overline{Prech}$ is pull up and turns off the pull up PMOS transistors. The bit lines are floating now and ready for the write or read operation. In general, address signal *Adr_i* should arrive earlier than the signal *Write* or *Read* to avoid the wrong address from being selected and false operation from being conducted. For this design, address signal also need to be present at the *Adr_i* before the write or read operation. However, before the precharge signal $\overline{Prech}$ is set high, the address signal *Adr_i* can not pass the logic 1 to *Sel_i* as the NMOS transistor M6 does not conduct. Until the $\overline{Prech}$ set high, address signal will pass the logic 1 to the select signal *Sel_i* and the write or read operation can start. Whenever the operation starts, *Sel_ack* is set to logic 1 as it is derived from the OR gate which congregates the signals *Sel_i*. And the signal *Sel_ack* is sent to control PMOS transistor M1 and also sent to the Muller C-element which is used to generate the overall read or write operation acknowledge signal *ARW_ack*. When the write or read operation ends, *Write* or *Read* signal returns to low and signal *WR* also returns to low. However, signal *WR* can not pass the logic 0 to the next stage as PMOS transistor M1 is controlled by *Sel_ack*. When the address signal returns to logic 0, *Sel_i* returns to logic 0, then M1 conduct and pass the logic 0 from signal *WR* to $\overline{Prech}$ and the bit lines are pulled up again. The whole write or read operation ends now. This is to prevent the circumstance that the pre-chagre operation starts before the *Sel_i* has retruned to logic 0 which will lead to a competition between the precharge pull up transistors and the memory cell pull down transistors and causing much power dissipation.

4.3.1 Precharge control circuit

Figure 4.9 shows the schematic of the precharge control circuit. The function of this part has been introduced in section 3.4. It should be noted that the size of the pull up transistors should be adjusted to a reasonable value. As the length of bit lines increases, the capacitance of the bit lines also increases. To maintain a reasonable rising time, the size of the precharge transistors which are M7 and M8 in Figure 4.8 should be increased.

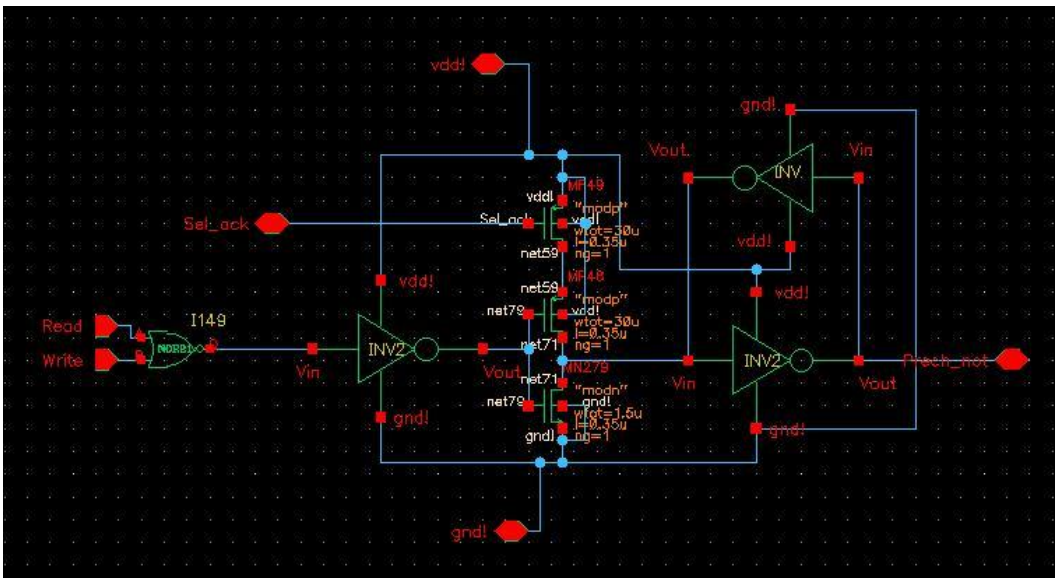


Figure 4.9 Schematic of the precharge control circuit

4.3.2 Row decoder

Generic 2 to 4 decoder diagram is shown in Figure 4.10. As can be seen from Table 4.1, it has a problem that when A_0, A_1 are all 0, D_0 will be selected. While it may be needed sometimes, most of the time it is not desired in asynchronous circuits. For example, when the SRAM is in empty state, all the signals are at logic 0,

no address line should be selected. However, the first row D_0 is selected which should be prevented when this typical decoder is used. In this design, an enable signal En is inserted. It fulfills the function requirement that if the enable signal is low, no address signals will be emitted. The schematic of the proposed 2-4 decoder is shown in Figure 4.11.

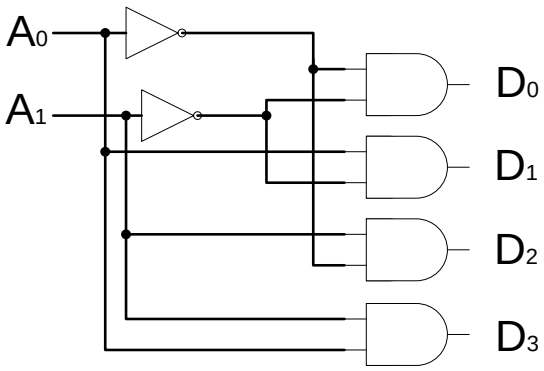


Figure 4.10 Schematic of the generic 2-4 decoder

Table 4.1 Truth table of the generic 2-4 decoder

A_1	A_0	D_3	D_2	D_1	D_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

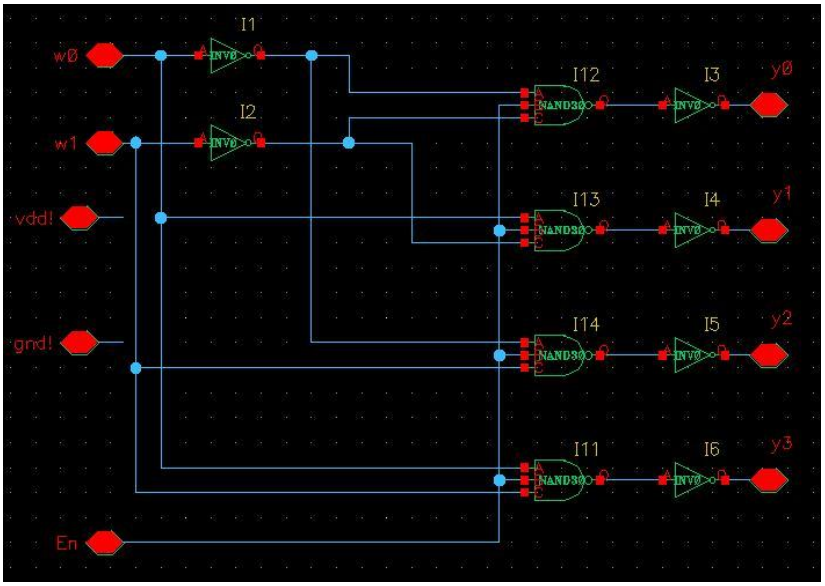


Figure 4.11 Schematic of the proposed 2-4 decoder

The location of the SRAM cell arrays should be considered when drawing the layout. The layout of the proposed 2 to 4 decoder is adjusted to the arrangement of the memory cell arrays and is shown in Figure 4.12.

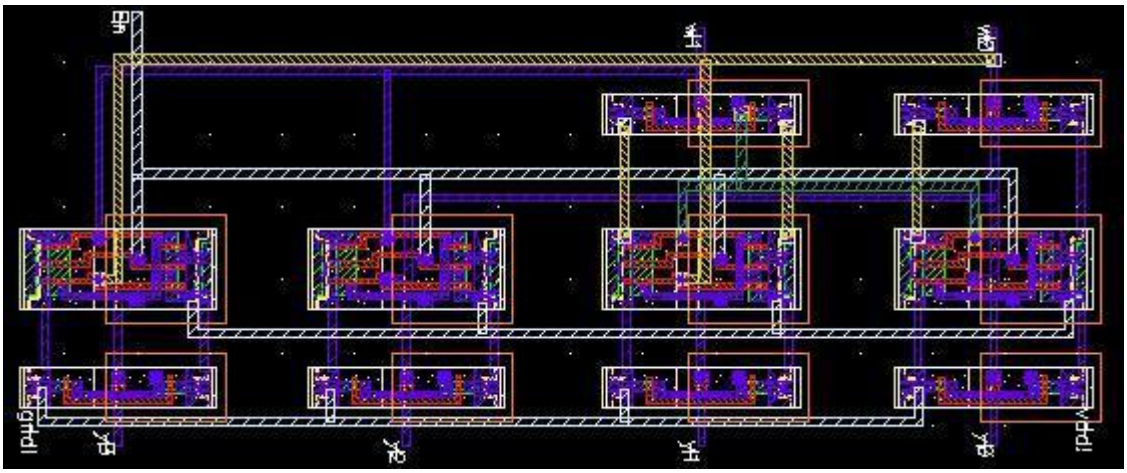


Figure 4.12 Layout of the proposed 2-4 decoder

The 4-16 decoder is formed by five 2-4 decoders and the schematic of the 4-16 decoder is shown as Figure 4.13.

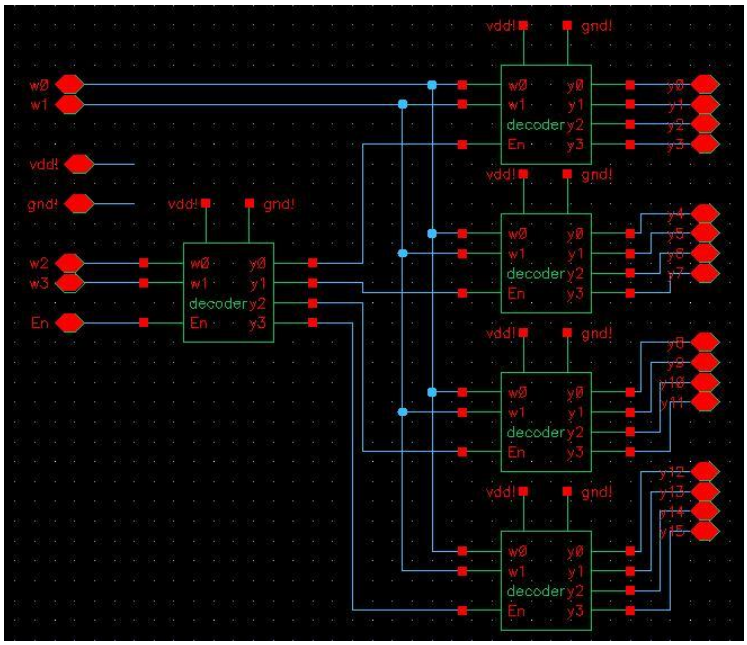


Figure 4.13 Schematic of the proposed 4-16 decoder

4.3.3 Select acknowledge circuit

The select acknowledge circuit provides the signal *Sel_ack* by collecting all the select signals *Sel_i* and sending them to an OR gate. Then the signal *Sel_ack* is sent to both the precharge control part and the acknowledge part. The design of this circuit mainly focuses on the multiple-input OR gate design. The schematic of a 2-input OR gate is shown in Figure 4.14. As can be seen, there are two PMOS transistors connected in serial. The 8-input OR gate is firstly considered to be designed and it leads to 8 PMOS transistors connected in serial. However, this asynchronous design is used for the power supply from 1V to 3.3V. The 8 PMOS transistors connected in serial can not work properly at the power supply of 1V. Not to mention the situation of the 16-input OR gate. Therefore, multiple-input

OR gate is to be designed in a different way.

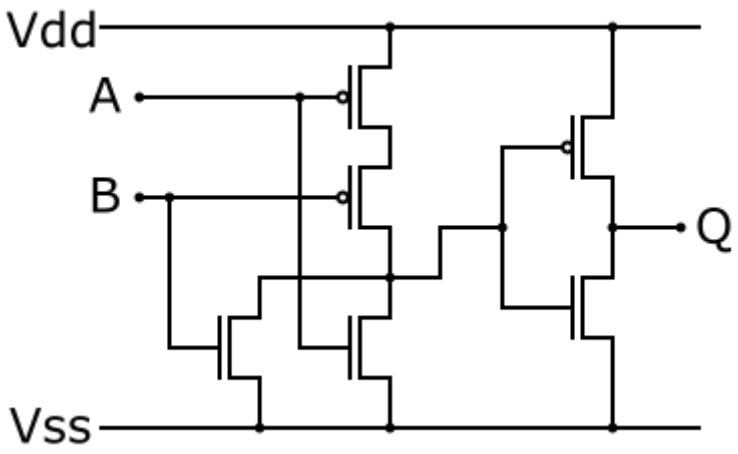


Figure 4.14 Schematic of a 2-input OR gate

Multiple-input OR gate

The Boolean logic of the16-input OR gate can be expressed as follow:

$$
\begin{aligned}
&Y \\
&= \\
&(A_1 + A_2 + A_3 + A_4) + (A_5 + A_6 + A_7 + A_8) + (A_9 + A_{10} + A_{11} + A_{12}) + (A_{13} + A_{14} + A_{15} + A_{16}) \\
&= \\
&\overline{\overline{(A_1 + A_2 + A_3 + A_4) + (A_5 + A_6 + A_7 + A_8) + (A_9 + A_{10} + A_{11} + A_{12}) + (A_{13} + A_{14} + A_{15} + A_{16})} \\
&= \\
&\overline{\overline{A_1 + A_2 + A_3 + A_4} \bullet \overline{A_5 + A_6 + A_7 + A_8} \bullet \overline{A_9 + A_{10} + A_{11} + A_{12}} \bullet \overline{A_{13} + A_{14} + A_{15} + A_{16}}}
\end{aligned}
$$

(4.1)

Based on the equation, the 16-input OR gate can be composed of one 4-input NAND gate and four 4-input NOR gates. The layouts of the NAND gate and NOR gates are from CMOS 0.35um library. The schematic and layout of the 16-input OR gate are shown in Figure 4.15 and Figure 4.16 respectively.

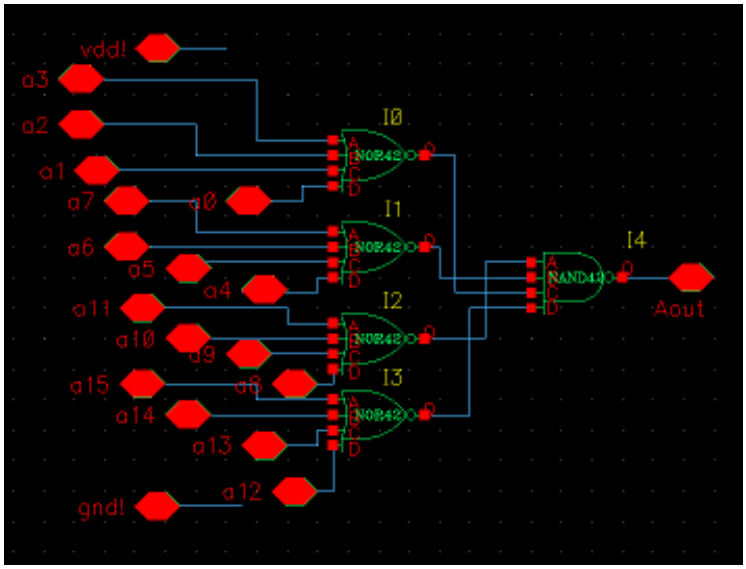


Figure 4.15 Schematic of the 16-input OR gate

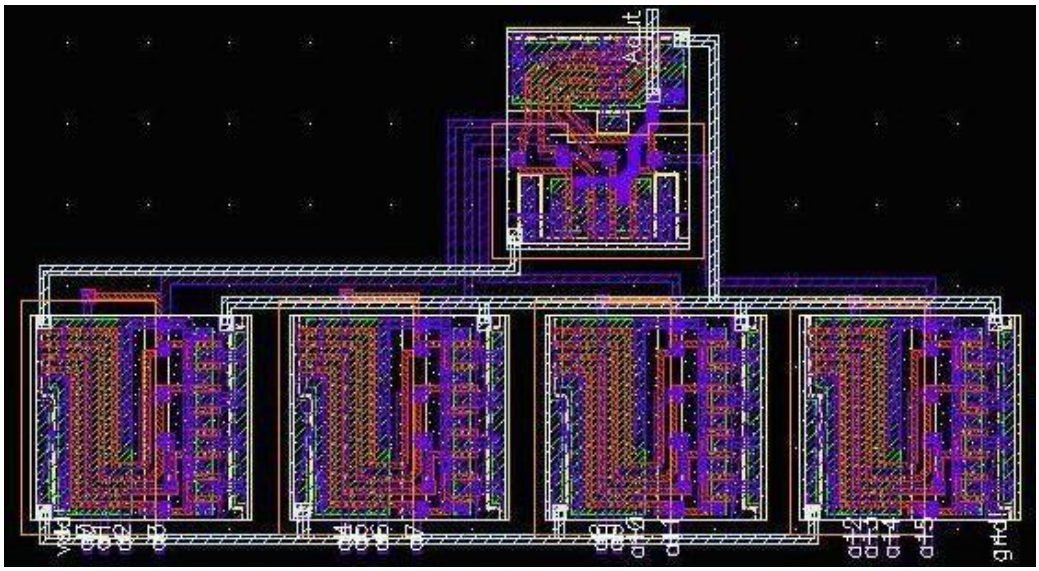


Figure 4.16 Layout of the 16-input OR gate

It has been tested through the schematic and post-layout simulation that the 16-input OR gate can work well at the power supply from 1V to 3.3V. The transient response of the gate including schematic and post-layout simulation results are presented in Figure 4.17.

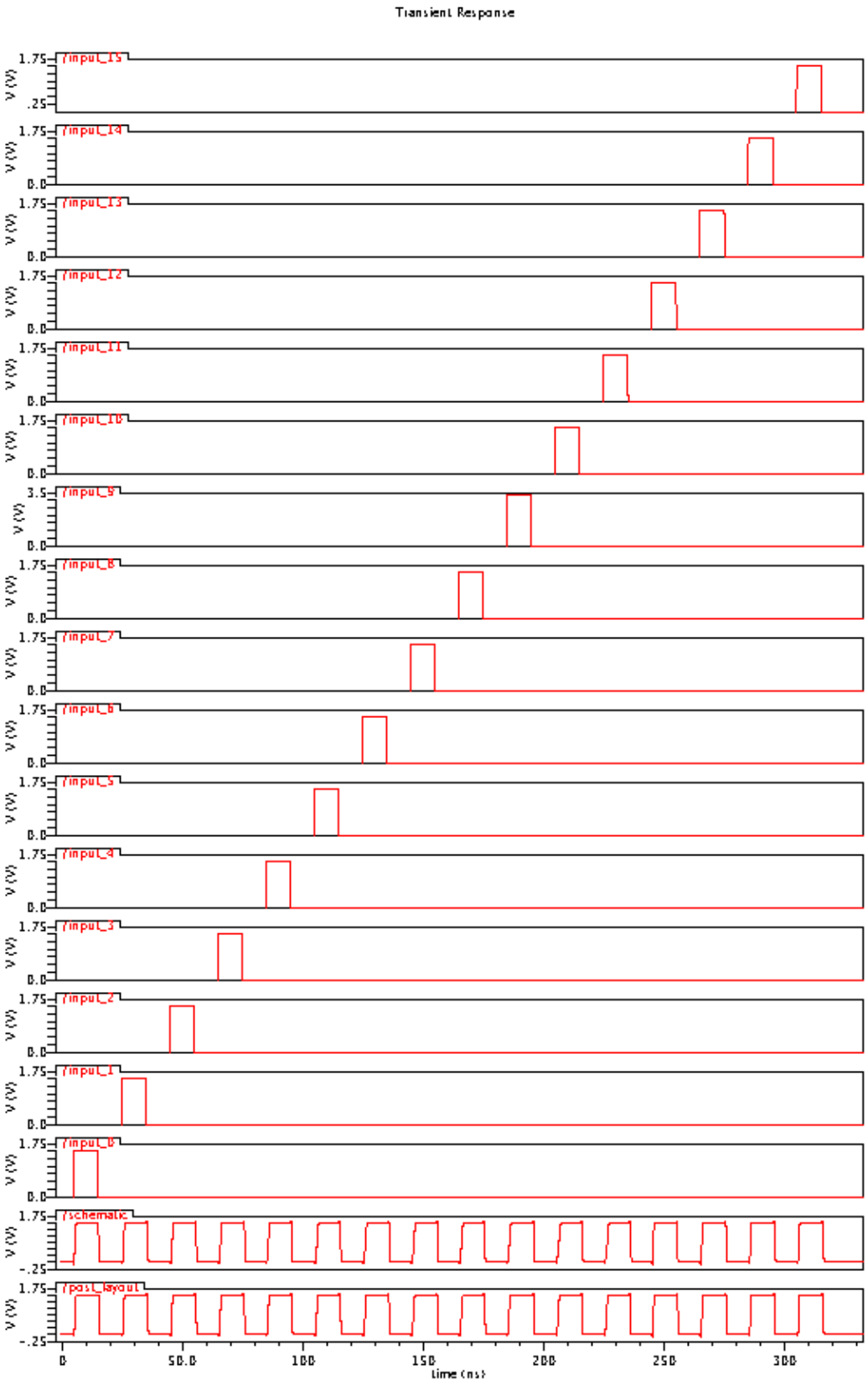


Figure 4.17 Transient response of the 16-input OR gate (Schematic and post-layout at 1.5V)

The schematic of the precharge and select circuit of the 4*8 bits asynchronous SRAM is shown in Figure 4.18.

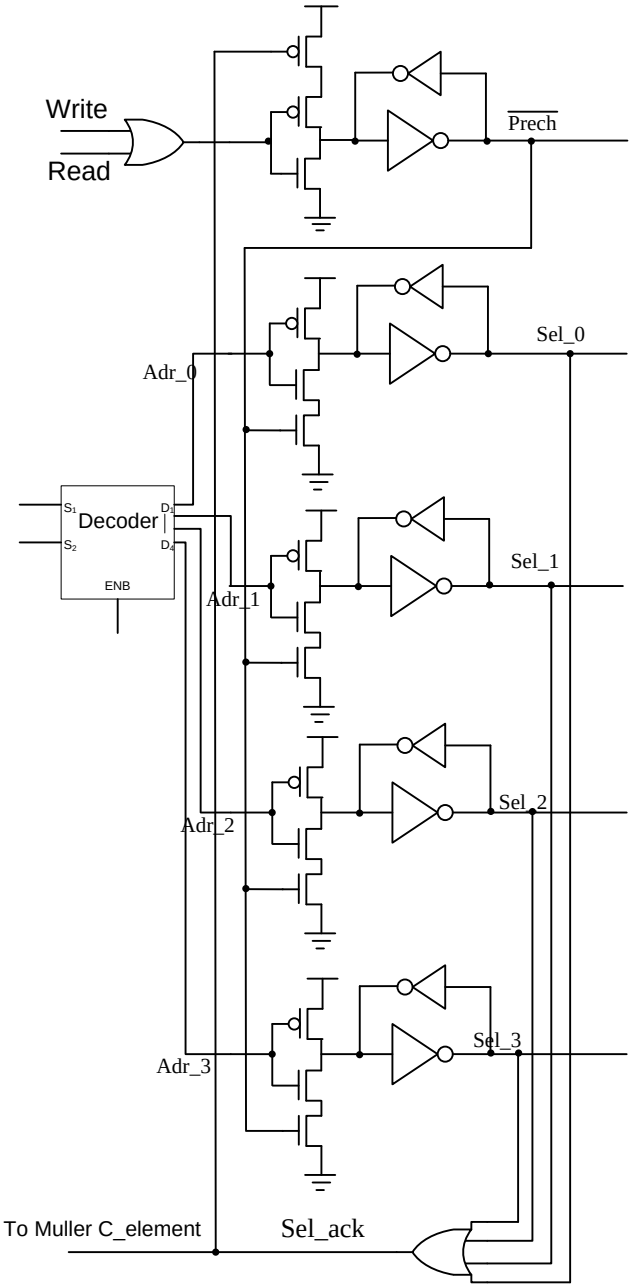


Figure 4.18 Schematic of the precharge and select circuit of the 4*8 bits asynchronous SRAM

The post-layout simulation result of the precharge and select circuit is shown in Figure 4.19.

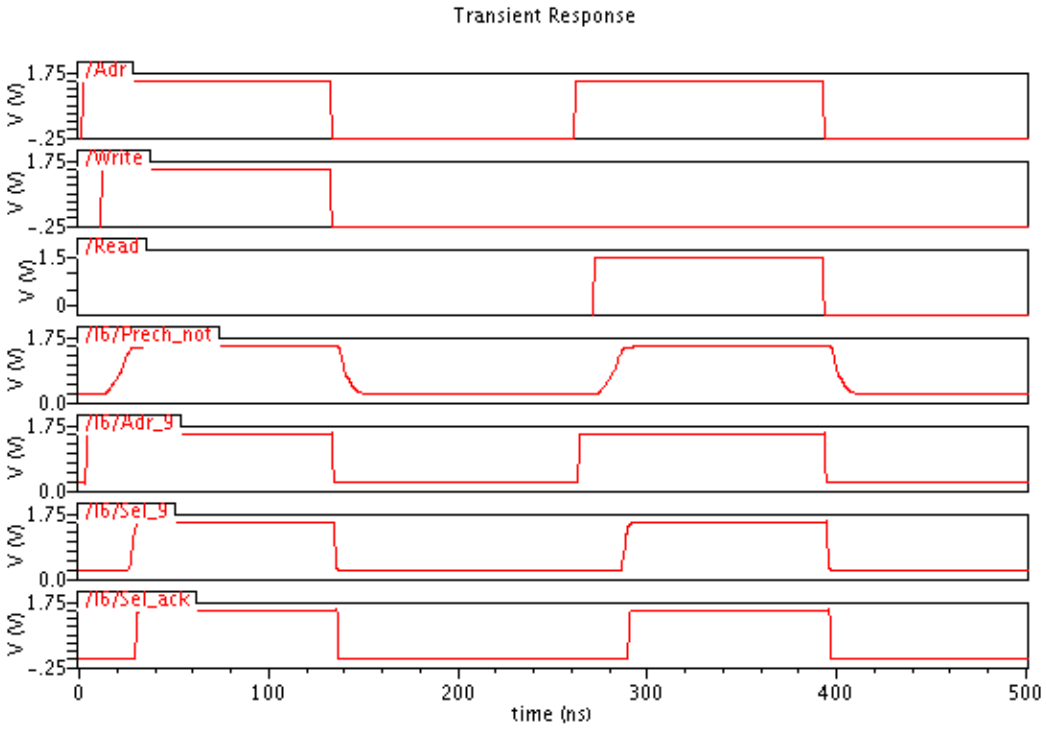


Figure 4.19 Transient response of the precharge and select circuit (Post-layout at 1.5V)

4.4 Acknowledge module design

The acknowledge module which is shown in Figure 4.20 mainly consists of four PMOS transistors connected in series, six NOMS transistors and multiple-input Muller C-element. It is connected to the two pairs of bit lines $Di.t$ and $Di.f$, $Do.t$ and $Do.f$. The schematic of the acknowledge module is shown in Figure 4.20. As can be seen, the acknowledge module is composed of eight acknowledge circuits connected in parallel. During the write operation, the data is written into the internal input bus $Di.t$ and $Di.f$ and one of them is pulled up to V_{DD}. When the data has been written into the memory cell, the corresponding internal output bus $Do.t$ or $Do.f$ is pulled up to V_{DD}. Therefore,

the signal $\overline{RW\_ack}$ is pulled down through the two NMOS transistors M5 and M7 (or M6 and M10) and set the acknowledge signal RW_ack high. When all the data values are written into the 8 bit memory cells, eight acknowledge signals are combined to one read and write acknowledge signal through the 8-input Muller C-element. This signal is combined with the select signal Sel_ack by a 2-input Muller C-element to form the overall acknowledge signal ARW_ack. During the read operation, as the value of internal input bus $Di.t$ and $Di.f$ is not changed, the read signal is directly connected to the two NMOS transistors M8 and M9 to help performing the necessary acknowledge signal.

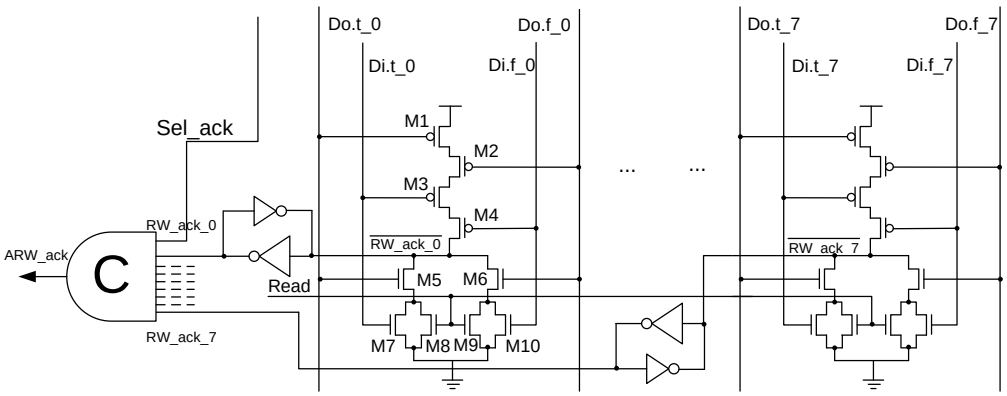


Figure 4.20 Schematic of the acknowledge module

When designing the acknowledge module, the choice of transistors size is crucial. The four PMOS transistors performing the pull up function are connected in series, the equivalent width is the width of each PMOS transistor divided by 4. Therefore, the PMOS transistor size should be relatively big compared to the pull down NMOS transistor size to keep an acceptable rising time. However, the size of the PMOS transistors cannot be too big, otherwise the falling time will be

longer and power consumption will be larger. The layouts of one bit acknowledge circuit and the acknowledge module are shown in Figure 4.21 and Figure 4.22 respectively.

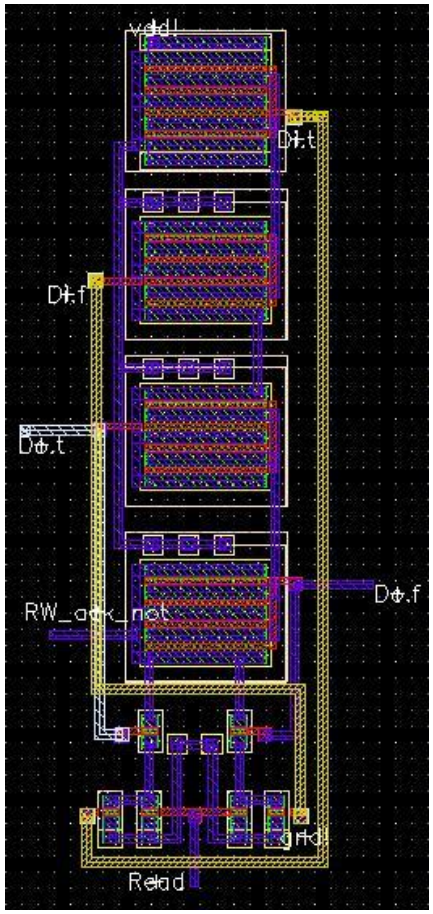


Figure 4.21 Layout of one bit acknowledge circuit

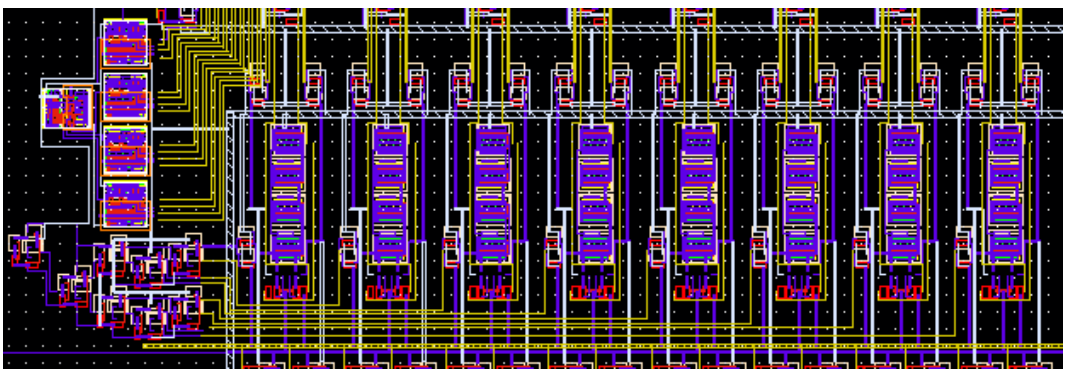


Figure 4.22 Layout of the acknowledge module

The post-layout simulation result is given by Figure 4.23.

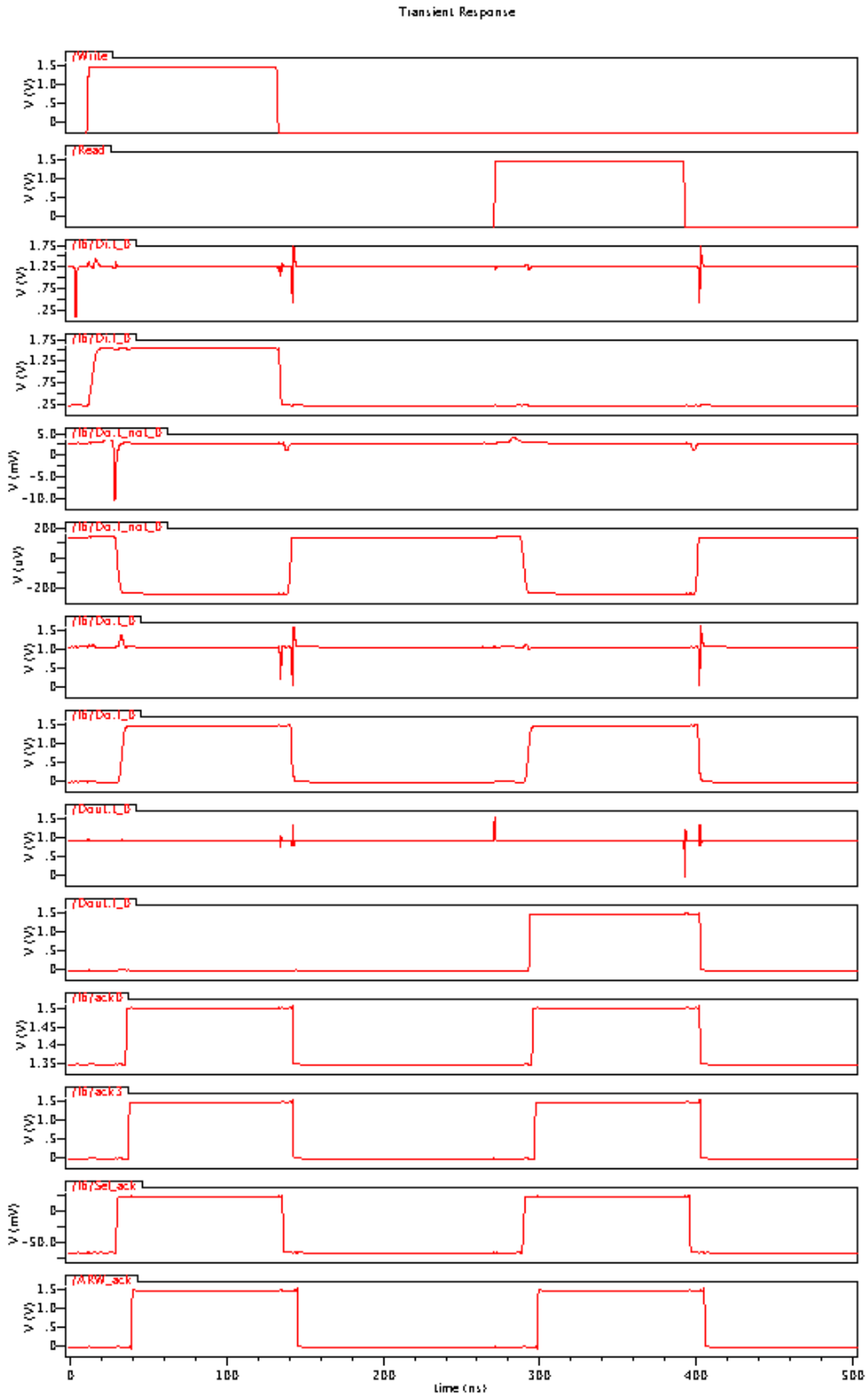


Figure 4.23 Simulated timing diagram of the acknowledge module (Post-layout at 1.5V)

4.5 Layout consideration

There are some strategies that need to be considered when we draw the layout. For example, the arrangement of every function block should make wire routing easy. And digital circuit design requires a compact layout whenever conditions permit. The layout of the 16*8 bits asynchronous SRAM is shown in Figure 4.24.

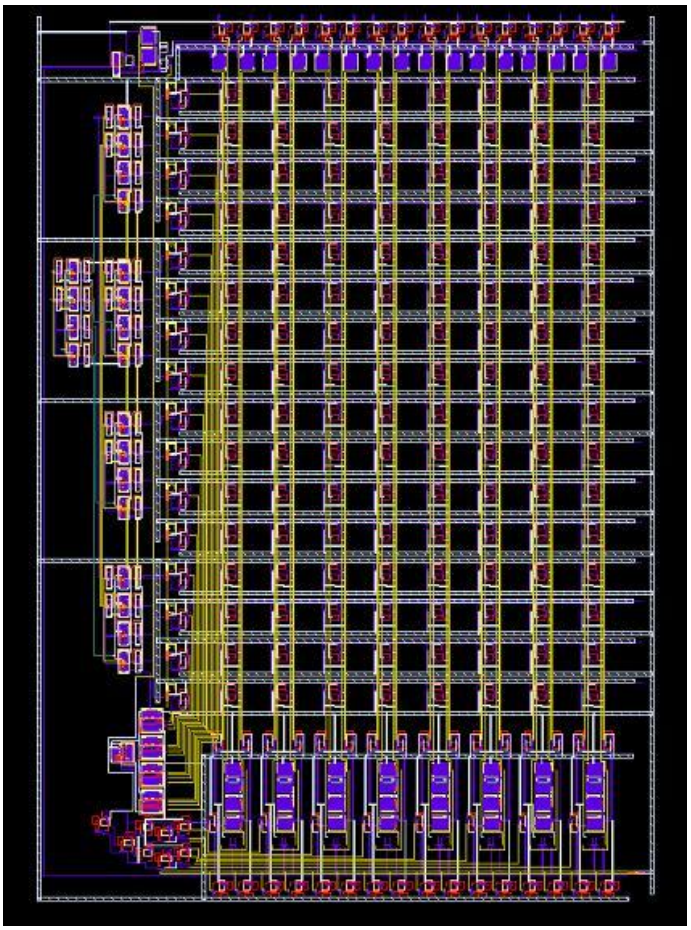


Figure 4.24 Layout of the 16*8 bits asynchronous SRAM

As can be seen, the sizes of both the precharge part and acknowledgement part are relatively big compared to the memory cell. Because the 16*8 bits asynchronous SRAM is only a testing design which is used to verify the function, main

focus has not been put on memory cells compactness. The control part is located at the left side of the layout including 4-16 decoder, precharge and select circuit. The center part shows the data-path, which begins with the data input at the top, connects through the memory cell arrays and the acknowledge circuits in the middle, and ends at data output at the bottom. The layout of the asynchronous 8051 microcontroller [21] including the 128*8 bits SRAM is shown in Figure 4.25. The SRAM module is located at the right side of the microcontroller. Other shape layouts can also be implemented to adapt to the microcontroller core shape.

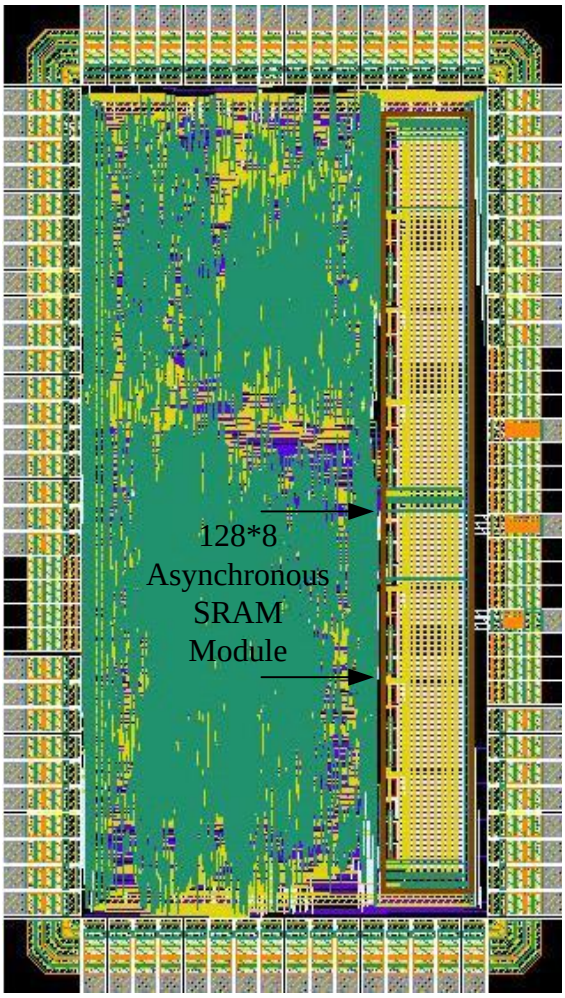


Figure 4.25 Layout of the asynchronous 8051 microcontroller

4.6 Schematic and post-layout simulation

The simulation is carried out by writing a different value into the memory cell then reading it out using Cadence environment. Three parameters are measured to reflect the performance of the working condition. As shown in Figure 4.26, the parameters are t_{WA} which is the delay from the write signal enabled to ARW_ack set high; t_{RO} which is the delay from the read signal enabled to the data read out; t_{RA} which is the delay from read signal enabled to ARW_ack set high.

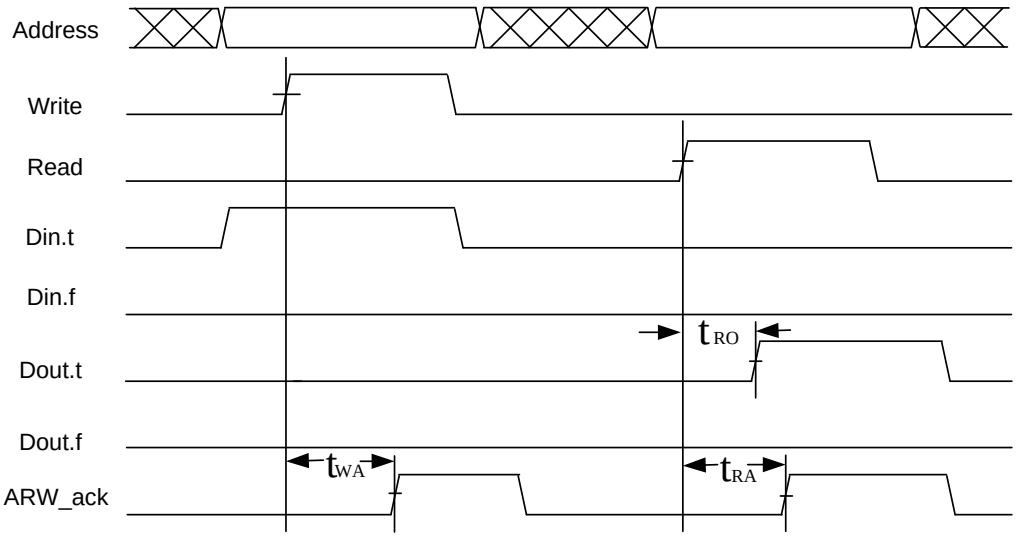


Figure 4.26 Timing diagram of the signals of the write and read operations

The simulation results which are shown in Table 4.2 and Table 4.3 show that the 16*8 bits and the 128*8 bits asynchronous SRAM can work well at the supply range between 0.9V and 3.5V. The relevant charts are shown in Figure 4.27 and Figure 4.28 respectively.

Table 4.2 Simulation results of the 16*8 bits asynchronous SRAM

Supply Voltage (V)	Schematic			Post-layout		
	t_{WA} (ns)	t_{RA} (ns)	t_{RO} (ns)	t_{WA} (ns)	t_{RA} (ns)	t_{RO} (ns)
3.5	10.39	10.27	7.74	11.19	10.83	8.18
3.3	10.83	10.68	8.33	11.61	11.52	8.59
3.0	11.82	11.69	8.83	12.71	12.58	9.43
2.8	12.68	12.52	9.62	13.61	13.51	10.01
2.5	14.21	13.99	10.83	15.24	15.05	11.28
2.2	16.63	16.41	12.79	17.74	17.62	13.36
2.0	18.42	18.35	14.48	19.78	19.63	14.84
1.9	19.69	19.51	15.52	20.97	20.82	15.87
1.8	21.38	20.86	16.44	22.95	22.57	17.46
1.7	23.51	22.65	17.85	24.84	24.44	18.73
1.6	25.72	25.09	19.68	27.52	27.03	20.74
1.5	28.35	27.68	22.02	30.46	29.74	23.21
1.4	32.42	31.38	25.22	34.49	33.91	26.37
1.3	37.61	36.79	29.17	40.19	39.02	30.93
1.2	45.39	44.05	35.54	48.31	47.23	37.16
1.1	57.84	56.29	45.54	61.34	59.92	47.47
1.0	79.59	78.99	64.11	83.93	83.24	66.71
0.9	133.99	132.81	108.32	139.72	138.35	112.11

Table 4.3 Simulation results of the 128*8 bits asynchronous SRAM

Supply Voltage (V)	Schematic			Post-layout		
	t_{WA} (ns)	t_{RA} (ns)	t_{RO} (ns)	t_{WA} (ns)	t_{RA} (ns)	t_{RO} (ns)
3.5	13.59	13.36	10.21	28.03	26.69	23.16
3.3	14.53	14.01	10.85	28.41	28.19	24.4
3.0	15.41	15.35	11.92	31.77	30.61	26.73
2.8	16.48	16.31	12.77	33.13	32.39	28.61
2.5	18.65	18.49	14.48	36.93	36.61	32.25
2.2	21.93	21.42	16.85	44.62	42.31	37.52
2.0	24.92	24.71	19.21	49.98	48.21	41.69
1.9	25.79	25.61	20.57	53.93	51.45	46.05
1.8	28.11	27.89	22.39	58.95	56.02	49.86
1.7	30.61	30.20	24.53	61.21	60.95	54.81
1.6	34.05	33.71	27.19	68.41	68.29	61.01
1.5	38.26	37.79	30.55	94.43	87.11	68.92
1.4	43.01	42.91	34.9	88.51	88.32	79.11
1.3	49.86	49.79	40.97	129.56	104.02	93.56
1.2	61.42	61.21	49.75	168.11	127.51	114.79
1.1	77.25	76.85	63.51	247.12	166.12	149.82
1.0	106.93	104.62	87.95	384.58	243.73	219.27
0.9	173.21	170.53	143.72	650.73	437.51	394.72

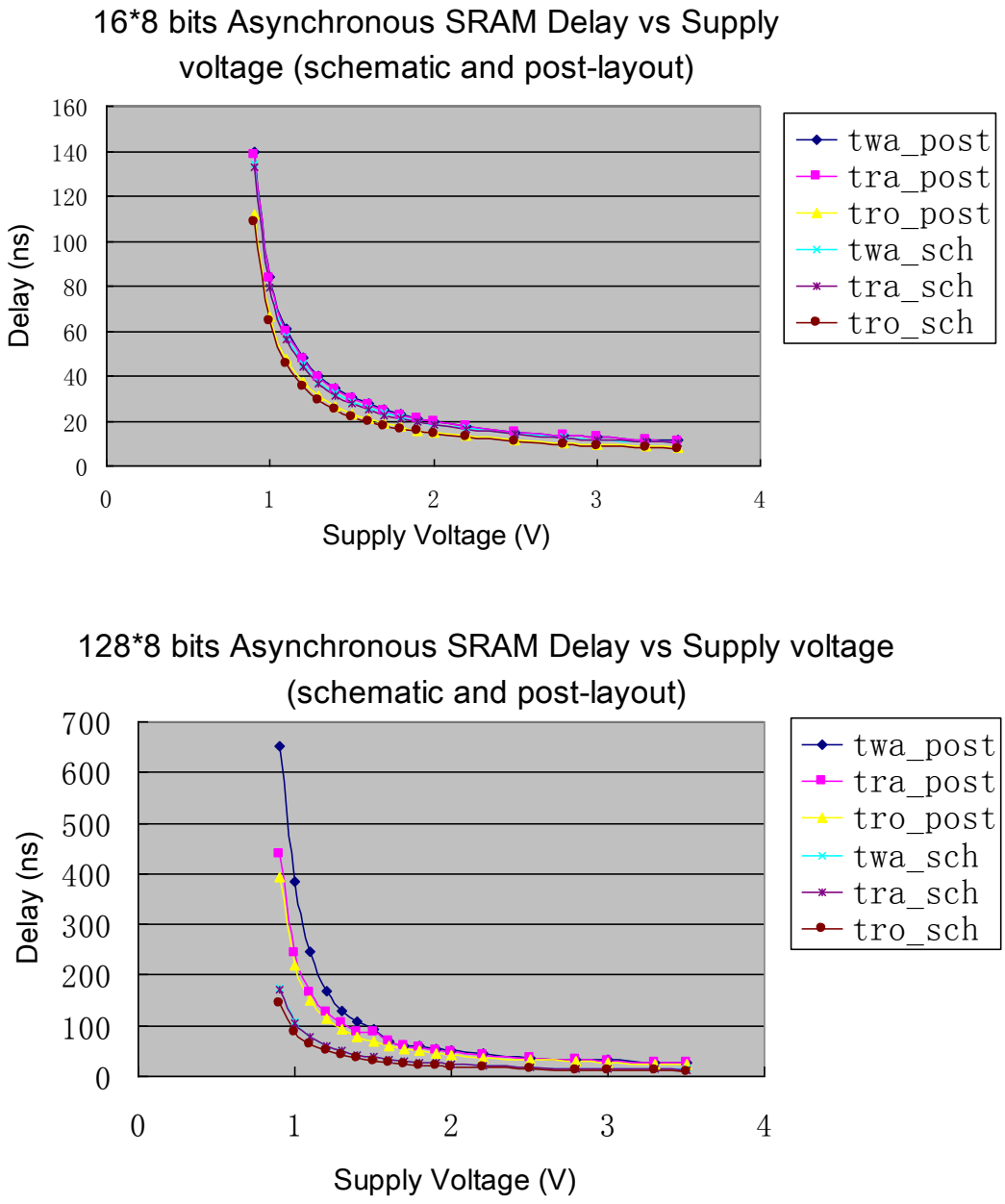


Figure 4.27 Simulated delay of the 16*8 bits and 128*8 bits asynchronous SRAM

As can be seen from Figure 4.27, the difference between the schematic and post-layout simulation results of the 16*8 bits asynchronous SRAM is not very large. However, such difference is getting significant for the 128*8 bits asynchronous SRAM. This is mainly because the Cadence simulation tool Diva does not include the parasitic capacitance of the bit lines in schematic simulations,

which manifests in the 128*8 bits asynchronous SRAM whose capacitance of the bit lines is much larger than that of 16*8 bits asynchronous SRAM.

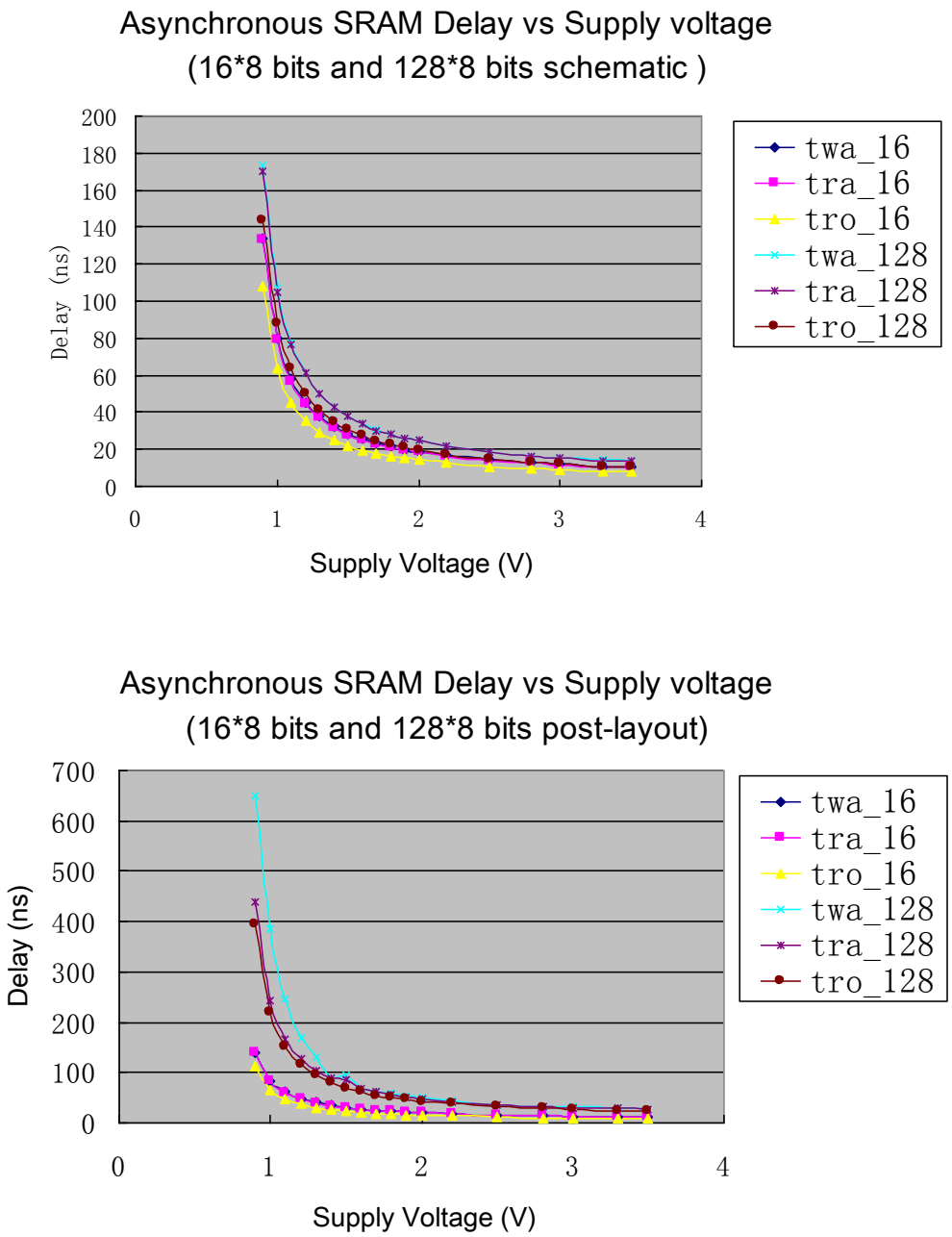
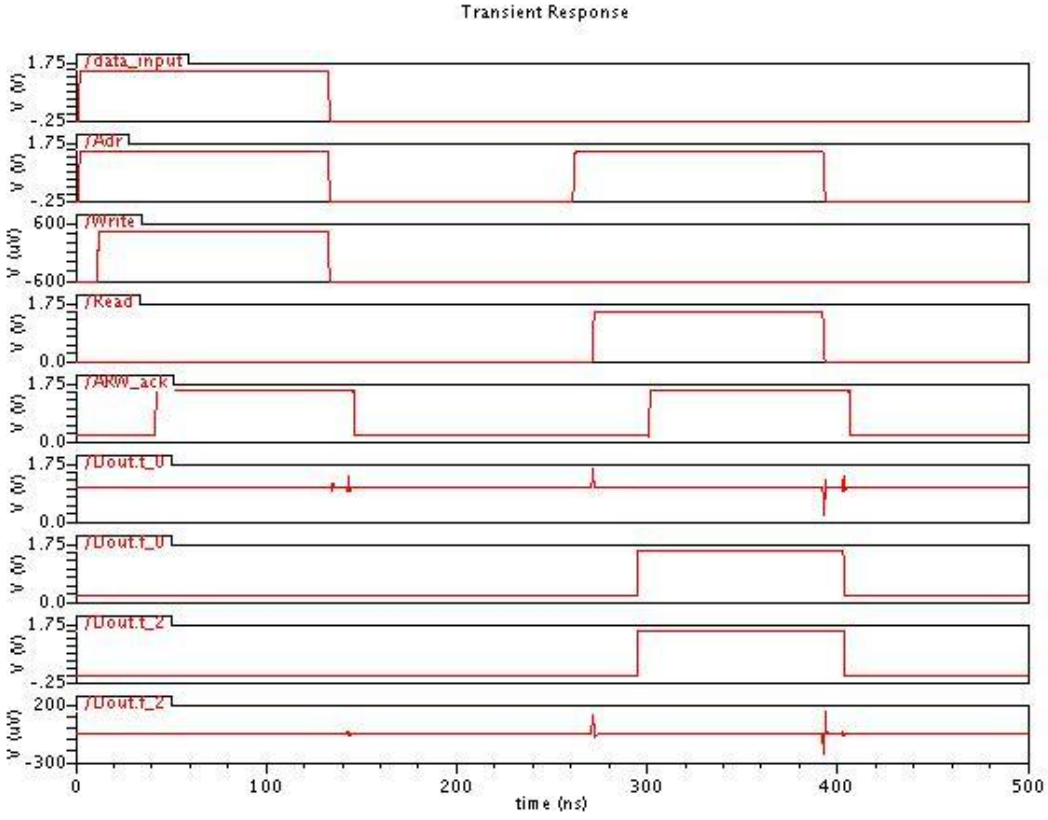


Figure 4.28 Simulated delay comparison between 16*8 bits and 128*8 bits asynchronous SRAM

The schematic and post-layout simulation results comparison between the 16*8 bits and the 128*8 bits asynchronous SRAM is shown in Figure 4.28. As mentioned, the Cadence simulation tool Diva does not include the parasitic capacitance in schematic simulations. And this causes not much difference between the schematic simulation results of the 16*8 bits and the 128*8 bits asynchronous SRAM. As the size of the memory array increases, however, the capacitance of the bit lines and the memory cells also increases which makes the acknowledge module take more time to generate the acknowledge signal for both write and read operations. It can be seen in the post-layout comparison of Figure 4.28 that the simulated delays of the 128*8 bits asynchronous SRAM are relatively larger than that of 16*8 bits. To avoid such large delay, when designing large memory arrays, one possible solution is using memory bank to reduce the length of bit lines. However, this may need extra control circuit and increase the complexity of the overall circuit.

Some simulated timing diagrams of the two fabricated asynchronous SRAM are shown from Figure 4.29 to Figure 4.33.



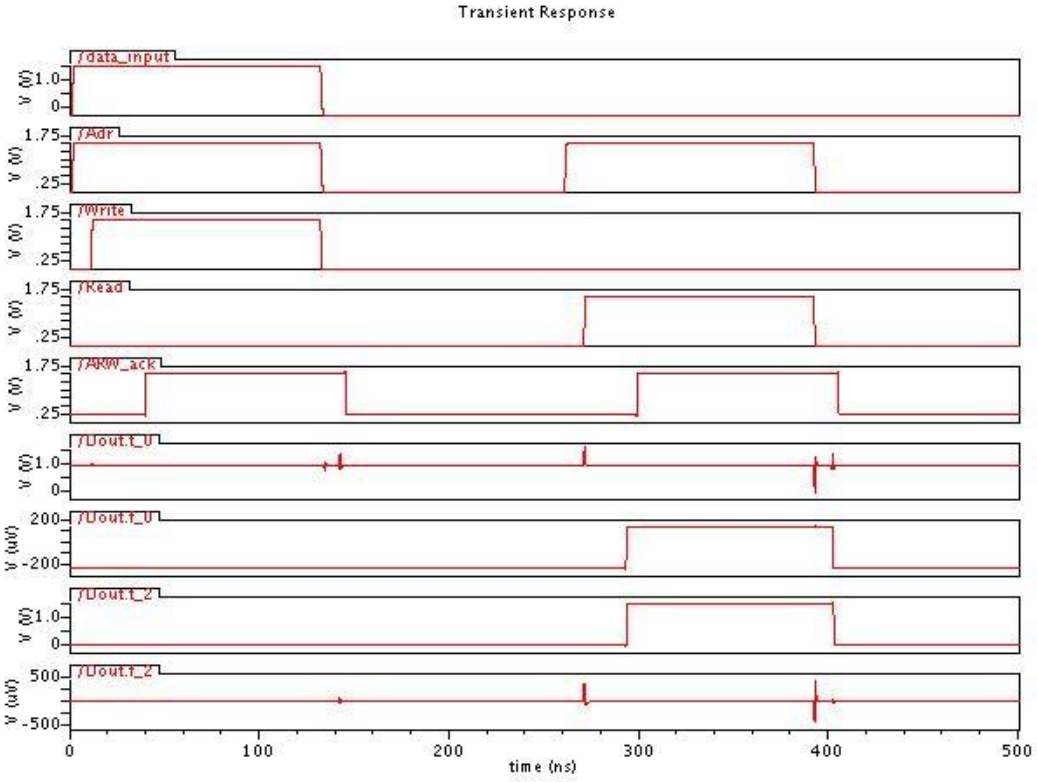


Figure 4.29 Simulated timing diagram of 16*8 bits asynchronous SRAM (Schematic and post-layout at 1.5V)

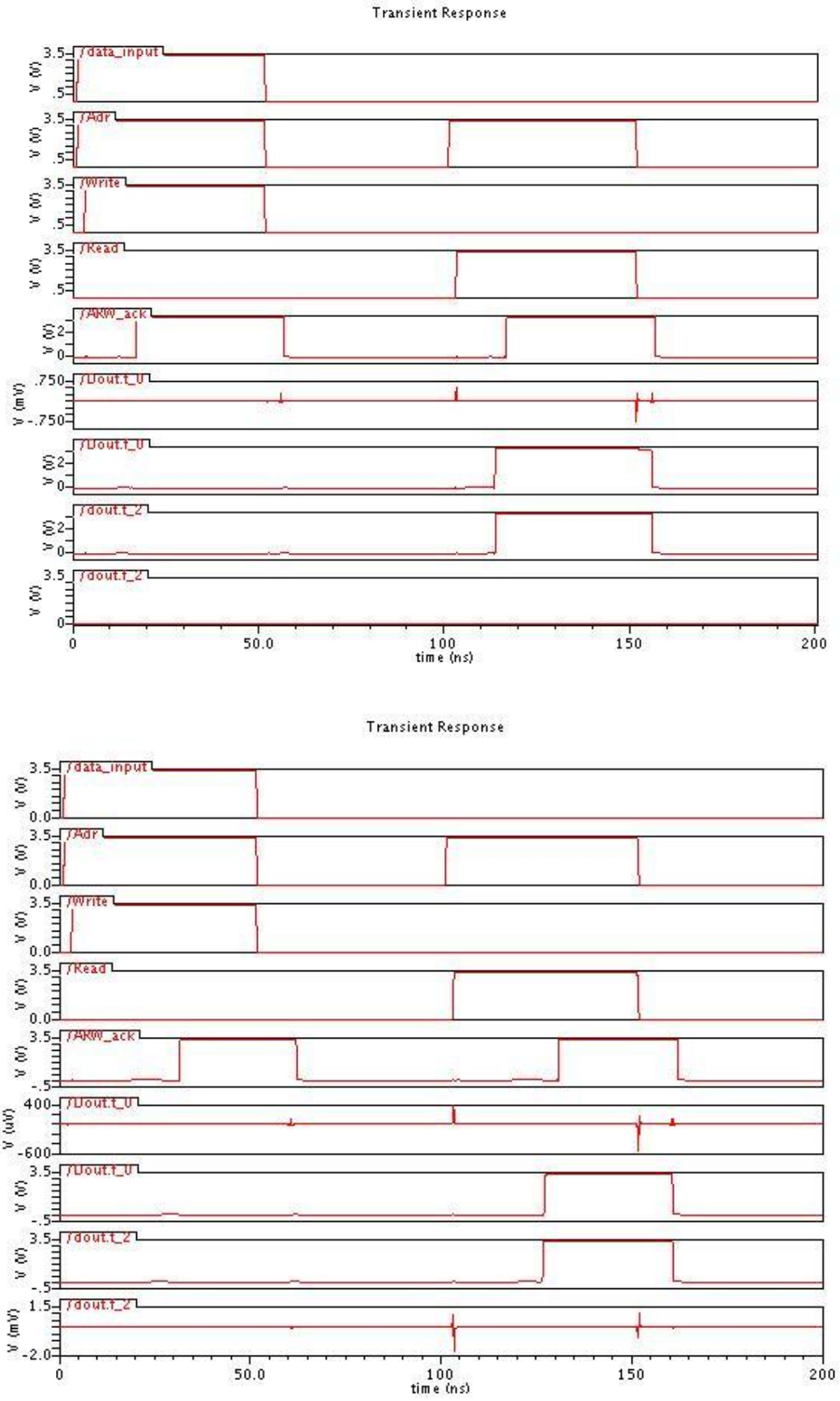
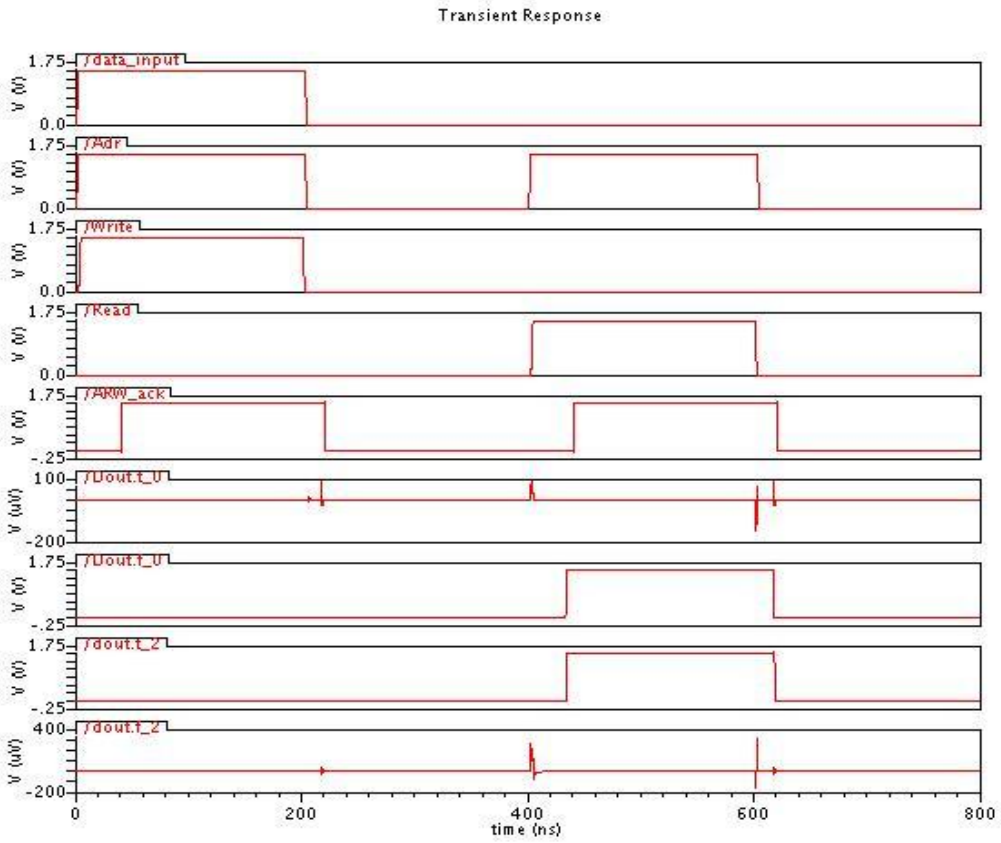


Figure 4.30 Simulated timing diagram of 128*8 bits asynchronous SRAM (Schematic and post-layout at 3.3V)



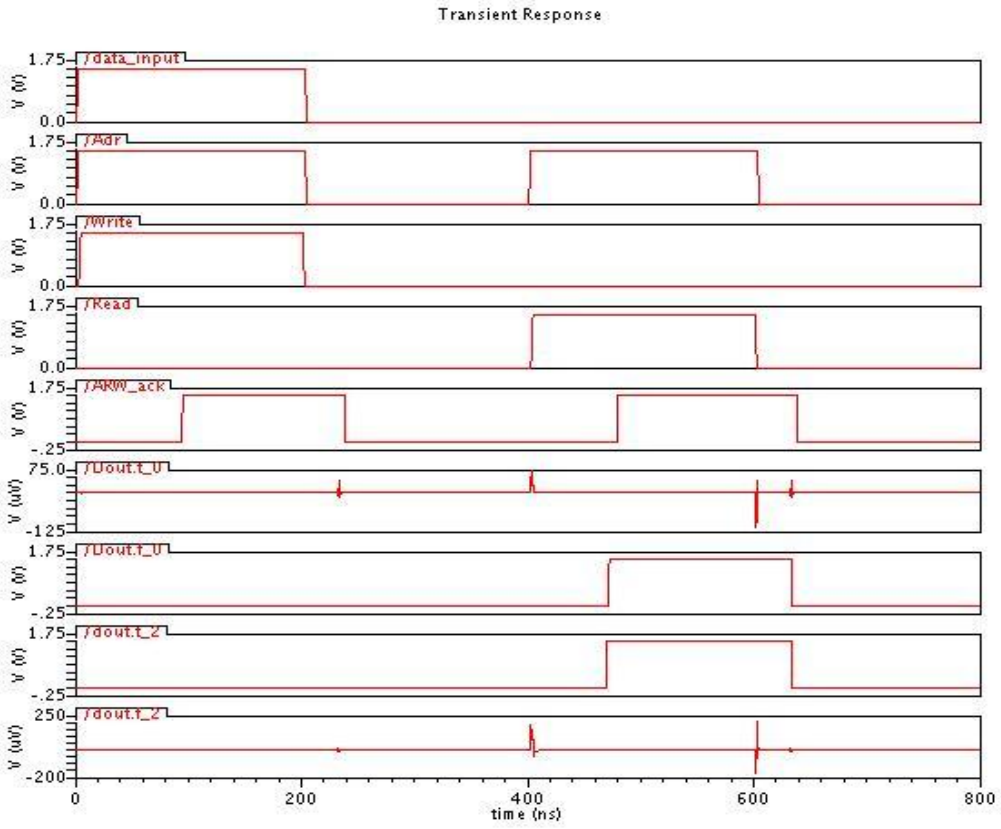


Figure 4.31 Simulated timing diagram of 128*8 bits asynchronous SRAM (Schematic and post-layout at 1.5V)

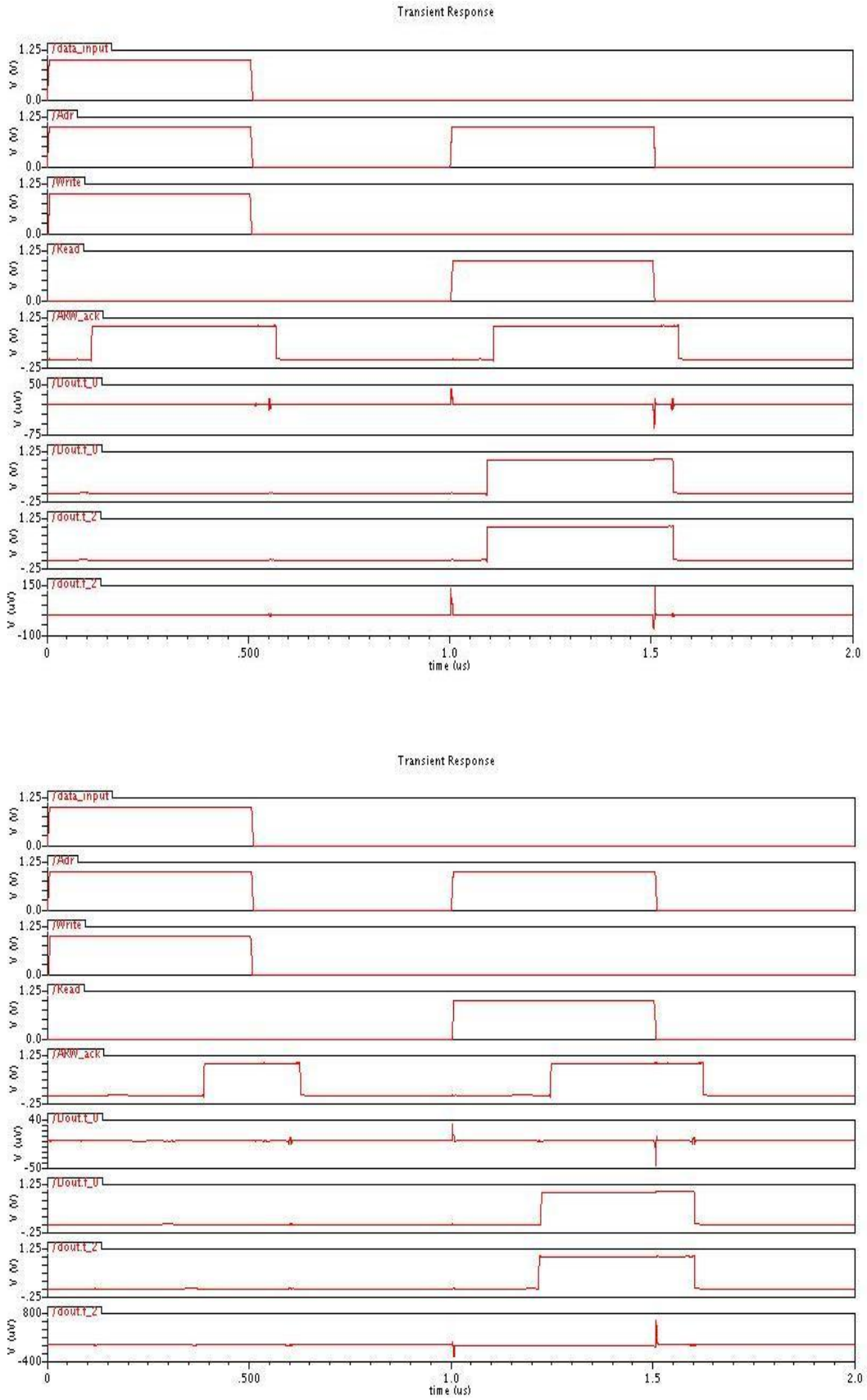


Figure 4.32 Simulated timing diagram of 128*8 bits asynchronous SRAM (Schematic and post-layout at 1V)

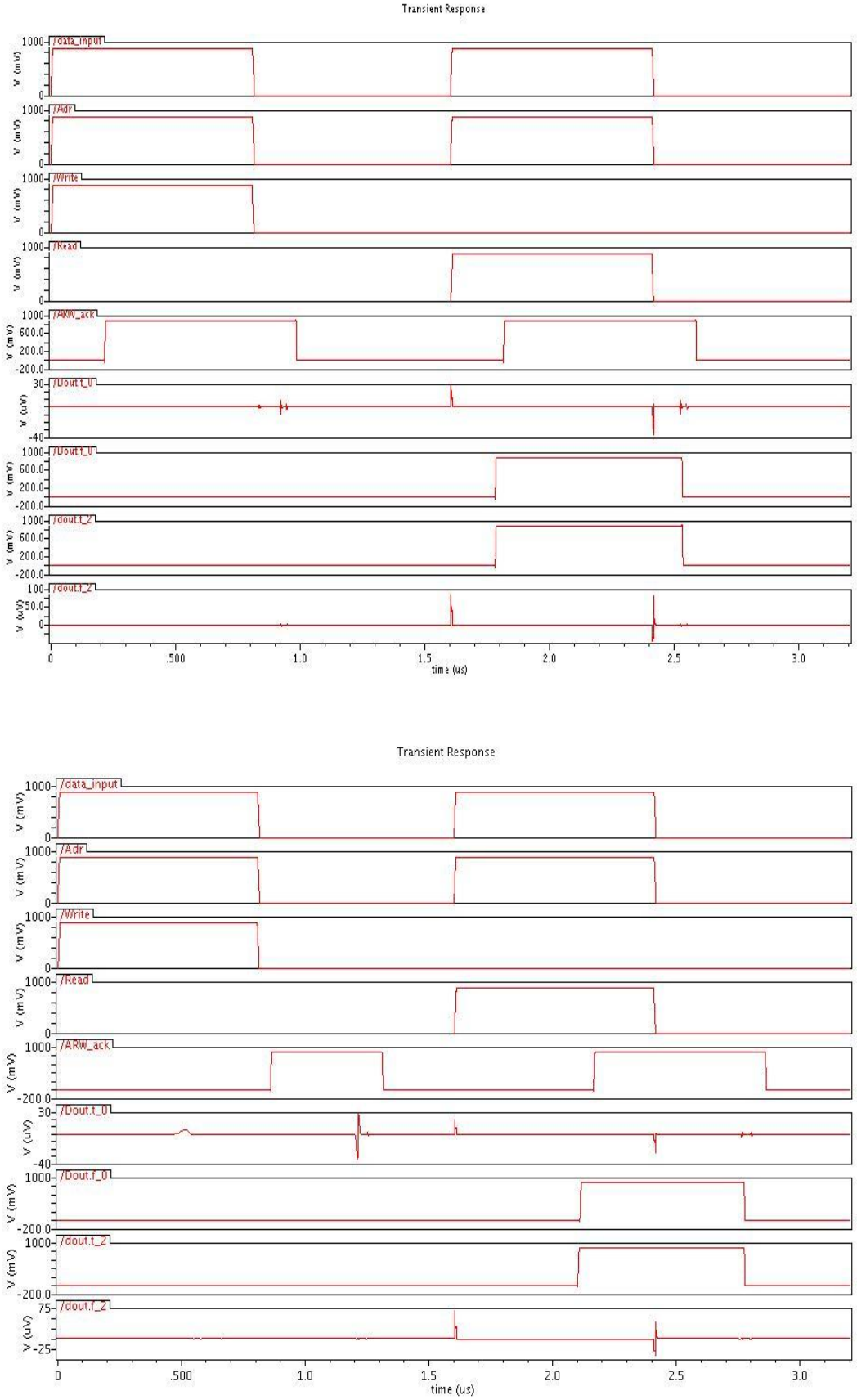


Figure 4.33 Simulated timing diagram of 128*8 bits asynchronous SRAM (Schematic and post-layout at 0.87V)

CHAPTER 5

EXPERIMENTAL RESULTS

5.1 Introduction

There are two versions of asynchronous SRAM that have been fabricated. The first version of the 16*8 bits asynchronous SRAM has been fabricated separately to verify the function and the integrity of the design. The second version of the 128*8 bits asynchronous SRAM has been fabricated with the asynchronous 8051 microcontroller which is also designed in the project group. Two versions of the asynchronous SRAM are all fabricated in AMS 0.35um, double-poly, four-metal process. Due to the pad limitation of the asynchronous 8051 microcontroller, the signal wires of 128*8 bits asynchronous SRAM are not connected with the pads. Therefore the test of the 128*8 bits asynchronous SRAM is based on the working condition of the asynchronous 8051 microcontroller. This chapter mainly focuses on the experimental results of the 16*8 bits asynchronous SRAM, the experimental results of 128*8 bits asynchronous SRAM is also included. The testing setup and the measured results are described in this chapter followed by some discussions about the measured results.

5.2 Testing setup

The complete test setup is shown in Figure 5.1. Two sides PCB is fabricated for testing both the 16*8 bits asynchronous SRAM and the asynchronous microcontroller with separate power supply for the core and the whole circuit including

the pads. Two DC Power Supplies are used to provide the power supply. The input signals for SRAM including write, read, data input bus and address is generated by the Logic Analyzer. The data on output bus and the acknowledge signal are observed using the Oscilloscope. The working current is captured by the Multimeter inserted between the PCB board and the power supply. The photograph of the PCB is shown in Figure 5.2.

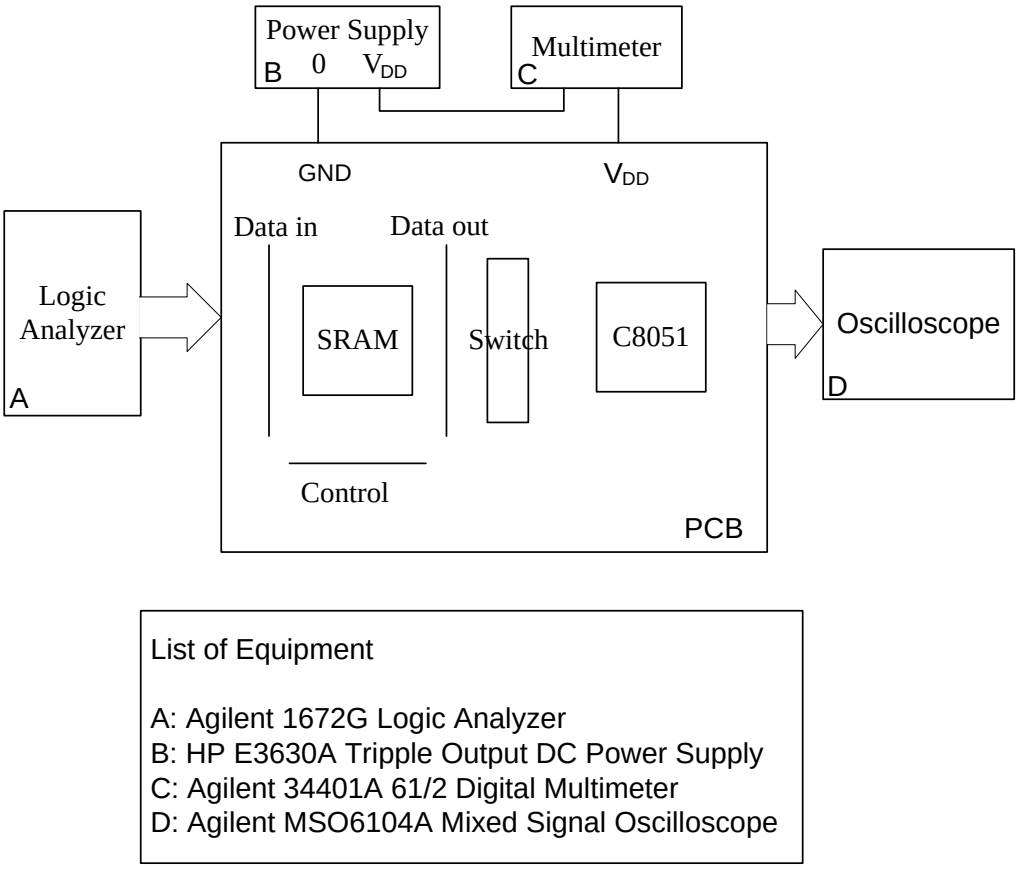


Figure 5.1 Experimental test setup for the SRAM module and the asynchronous 8051 microcontroller

Testing Procedure:

1. First, the SRAM is tested without connecting to the asynchronous 8051 microcontroller core. It is programmed by logic analyzer to write with 5 byte of data in the different SRAM addresses and then read out the data separately.
2. Second, the SRAM is tested with the asynchronous microcontroller core.

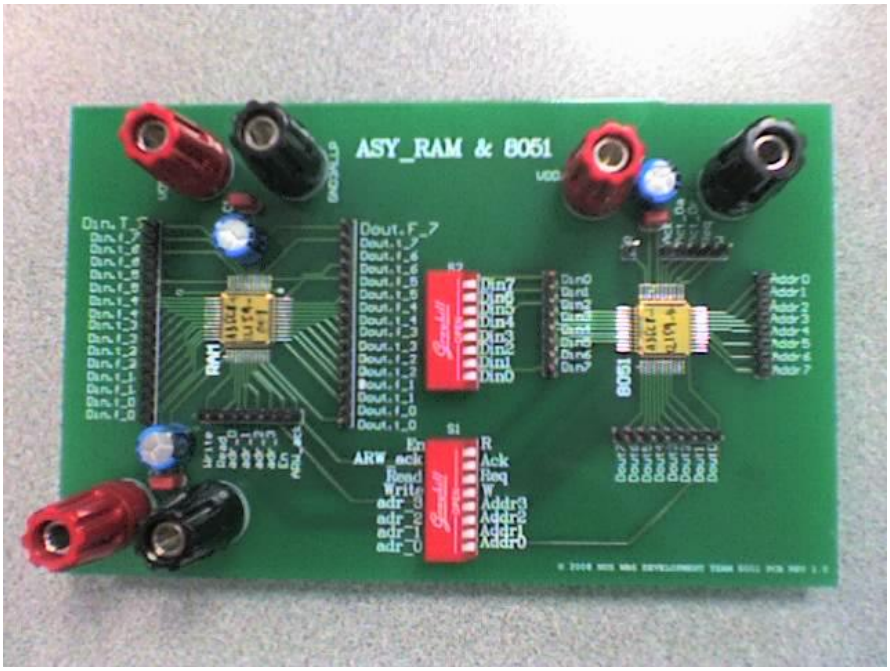


Figure 5.2 Photograph of the PCB used for testing

The working speed of the SRAM is tested under different supply voltage. When the working current and the power consumption is tested, the output of the power supply is fixed, and then the frequency of the input signals controlled by logic analyzer is changed in the range across which the SRAM can work in a good condition. When this series of testing is done, the voltage is changed to different

values and another series of testing are carried out.

5.3 Testing results

The experimental results of the 16*8 bits asynchronous SRAM are shown in Table 5.1. Figure 5.3 shows the comparison between the experimental results and the post-layout results.

Table 5.1 Experimental results of the 16*8 bits asynchronous SRAM

Supply Voltage (V)	**Experimental Results**		
	t_{WA} (ns)	t_{RA} (ns)	t_{RO} (ns)
3.5	17.2	15.6	10.4
3.3	19.0	17.0	12.6
3.0	20.0	18.0	13.0
2.8	21.4	20.5	16.9
2.5	23.6	22.9	19.8
2.2	25.0	24.2	22.4
2.0	27.6	26.7	25.5
1.9	31.7	30.5	29.3
1.8	34.1	33.5	30.8
1.7	37.9	34.2	31.3
1.6	40.2	39.5	35.2
1.5	56.0	54.0	46.0
1.4	61.9	59.8	56.2
1.3	72.3	70.1	66.9
1.2	84.3	77.9	72.0
1.1	126.1	120.3	103.8
1.0	176.0	168.0	140.0
0.9	296.0	280.0	236.0

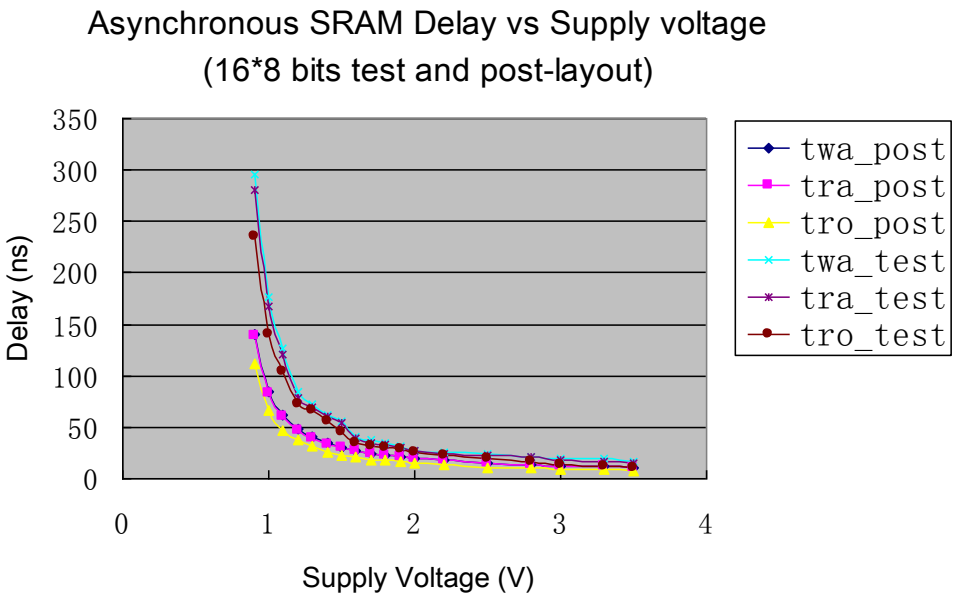


Figure 5.3 Diagram of comparison between the experimental results and post-layout results of the 16*8 bits asynchronous SRAM

As can be seen in Figure 5.3, the difference between the experimental and post-layout results is getting more pronounced when the supply voltage is below around 1.6V. This is because of the excessive pad delay in that range of the supply voltage. The typical usage of the SRAM module is direct connecting with the microcontroller core, and the load capacitance is very small. Therefore, the load capacitance is ignored when the post-layout simulation is conducted. However, the outputs of the fabricated 16*8 bits asynchronous SRAM are connected with the digital pads which see large load capacitance. Therefore, the experimental results are larger than post-layout results. As the supply voltage decreases, the delay of the pads increases significantly and becomes more dominant.

The current of the SRAM core is measured by the multimeter inserted between the power supply and the testing PCB. The power consumption can be derived from the equation which is $P = I \cdot U$. The SRAM core current / power consumption and relevant chart are shown in Table 5.2 and Figure 5.4 respectively. The write and read signals are generated by the logic analyzer with 50% duty cycle and the speed in Hz in Table 5.2 refers to the reciprocals of their periods.

Table 5.2 Experimental SRAM core current / power consumption (uA/ uW)

V / Speed	0.85V	0.9V	1V	1.2V	1.5V	3V	3.3V
4K	0.3 /0.255	0.3/0.27	0.5/0.5	0.6/0.72	0.9 /1.35	3.2/9.6	3.8 /12.54
20K	0.4 /0.34	0.4/0.36	0.9/0.9	0.7/0.84	1.0 /1.50	3.6/10.8	4.1 /13.53
100K	0.7 /0.595	0.7/0.63	0.7/0.7	1.0/1.20	1.5 /2.25	4.7/14.1	5.5 /18.15
200K	1.1 /0.935	1.1/0.99	1.2/1.2	1.5/1.80	2.1 /3.15	6.4/19.2	7.3 /24.09
1M	4.3 /3.655	3.7/3.33	4.1/4.1	5.2/6.24	6.9 /10.35	18.8 /56.4	21.5 /70.95
2M	7.8 /6.63	7.1 /6.39	8.0/8.0	9.8 /11.76	12.9 /19.35	34.8 /104.4	39.3 /129.69
5M	--	17.2 /15.48	18.5 /18.5	23.4 /28.08	30.8 /46.20	82.2 /246.6	92.7 /305.91
10M	--	--	36.3/36.3	46.3 /55.56	60.2 /90.3	155.6 /466.8	177.7 /586.41
20M	--	--	--	--	--	303.6 /910.8	310.1 /1023.33

"--"means the SRAM does not work properly under this circumstance.

Note that due to the precision limitation of the multimeter, the measured current results have a 0.1uA tolerance.

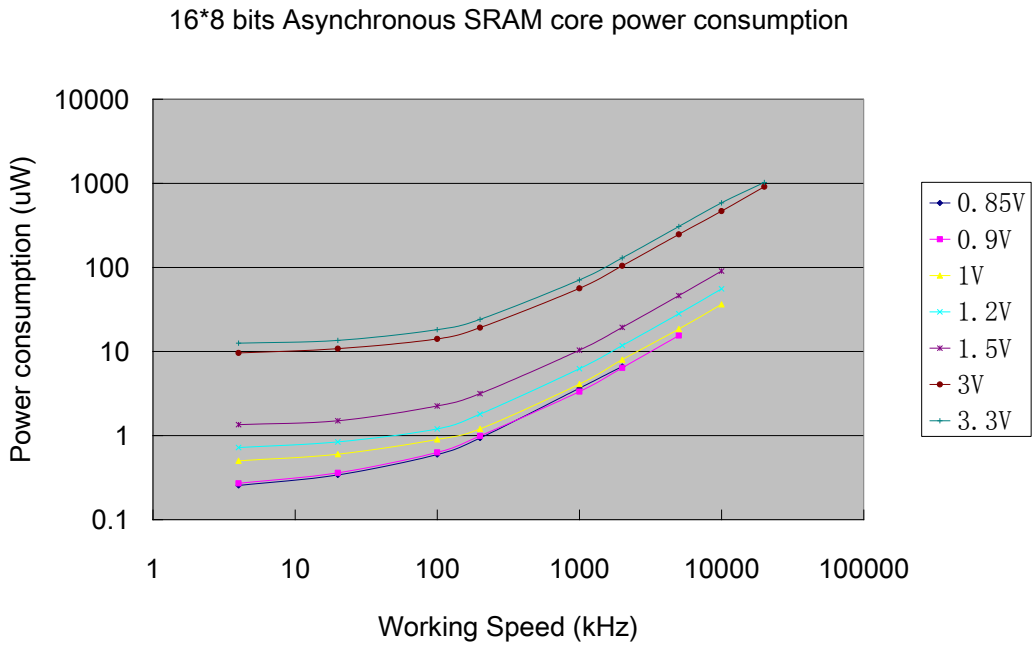


Figure 5.4 Diagram of experimental SRAM core power consumption

The relevant chart is shown in Figure 5.4. As can be seen, the power consumption decreases by around one order as the supply voltage decreased from 3.3V to 1V. It is evident from the chart that the curves bend upwards slightly when the working speed is below about 100k Hz. This is because the dynamic power consumption reduces linearly with the decreasing of working speed. Meanwhile, the static leakage power consumption does not change much and this makes it dominant in the overall power consumption.

The total SRAM's current consumption is significantly larger than SRAM core current consumption, as the pads cost much more power than the SRAM core. When the supply voltage is 1.5V, the SRAM core and total SRAM's current consumptions are shown in Table 5.3 which is as follows:

Table 5.3 Experimental results of the current consumption (uA) of the SRAM core and the total SRAM circuit

Speed	4K	20K	100K	200K	1M	2M	5M	10M
Core	0.9	1.0	1.5	2.1	6.9	12.9	30.8	60.2
Total SRAM	14.7	15.5	19.8	25.1	67.7	120.9	280.1	543.6

Some experimental timing diagrams of write and read operation are shown from Figure 5.5 to Figure 5.9. As shown in Figure 5.5, the four signals are *Write*, *Read*, *ARW_ack*, and Data out respectively.

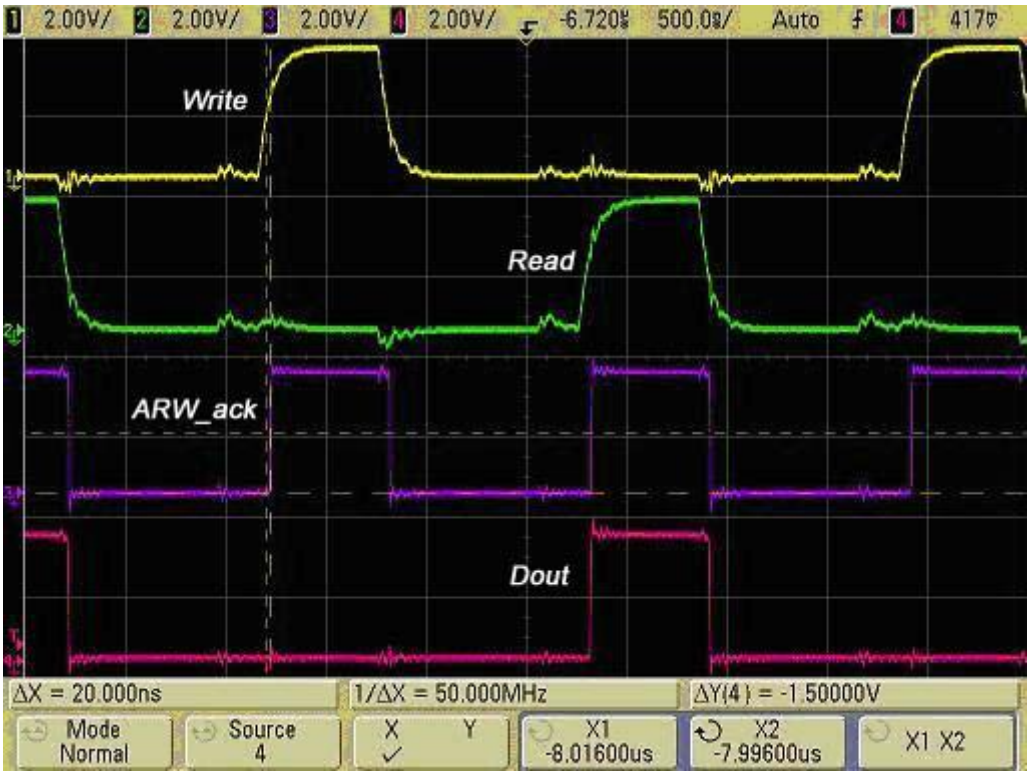


Figure 5.5 Experimental timing diagram of write and read operation (3V)

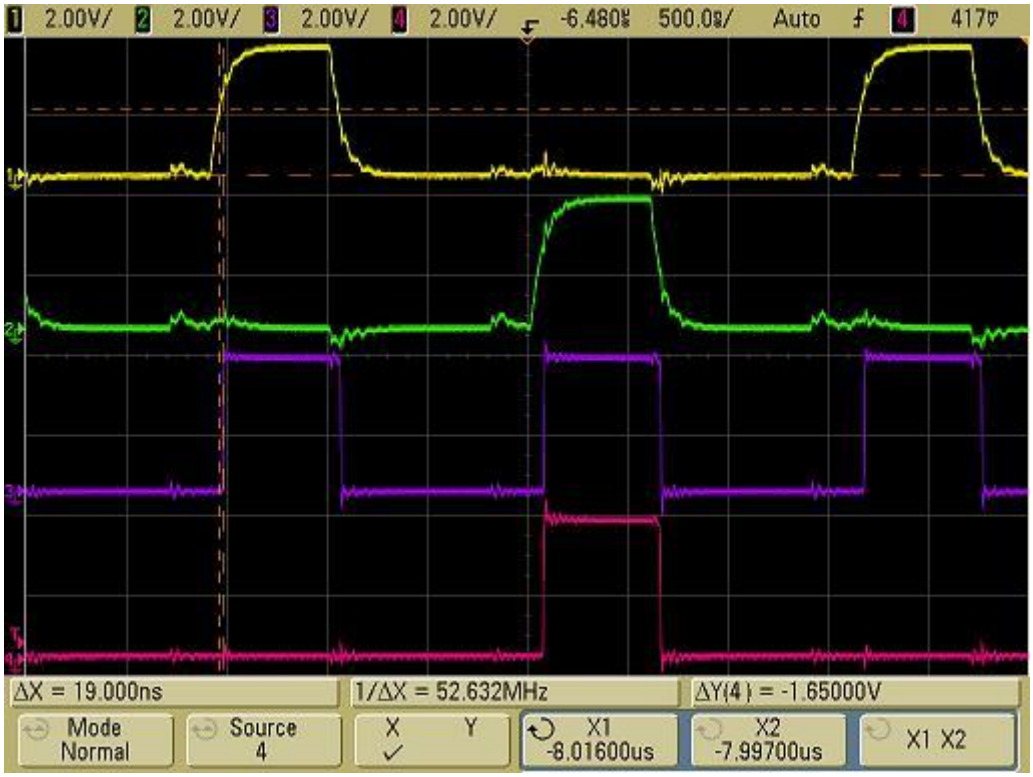


Figure 5.6 Experimental timing diagram of write and read operation (3.3V)

Figure 5.7 Experimental timing diagram of write and read operation (1V)

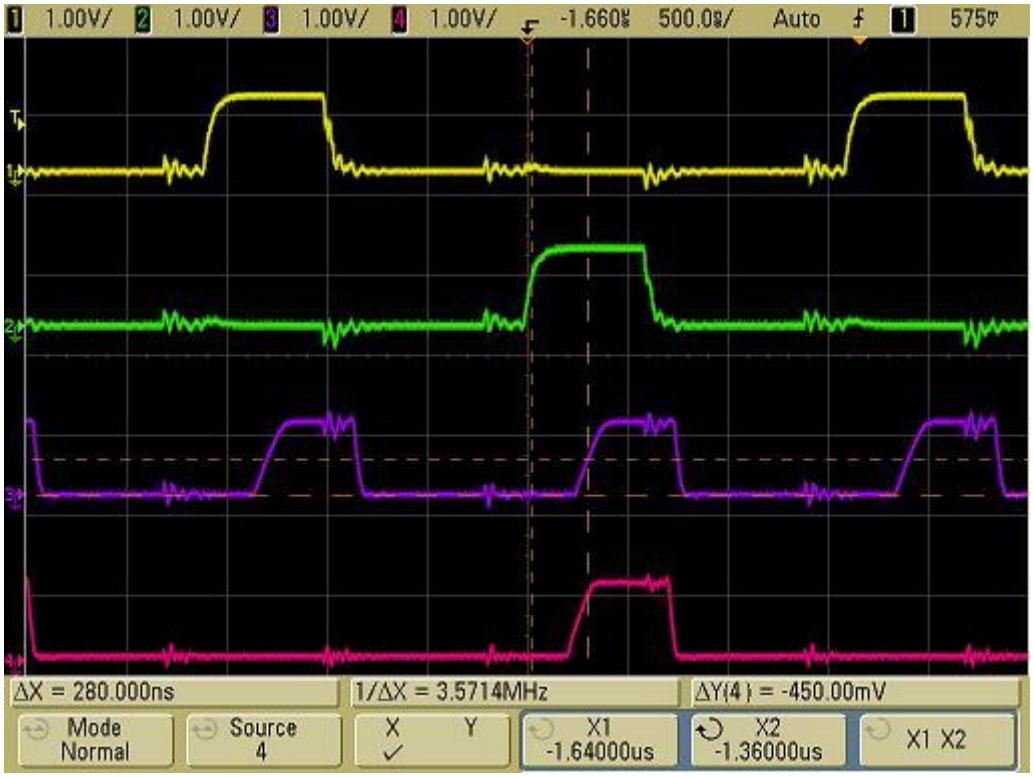


Figure 5.8 Experimental timing diagram of write and read operation (0.9V)

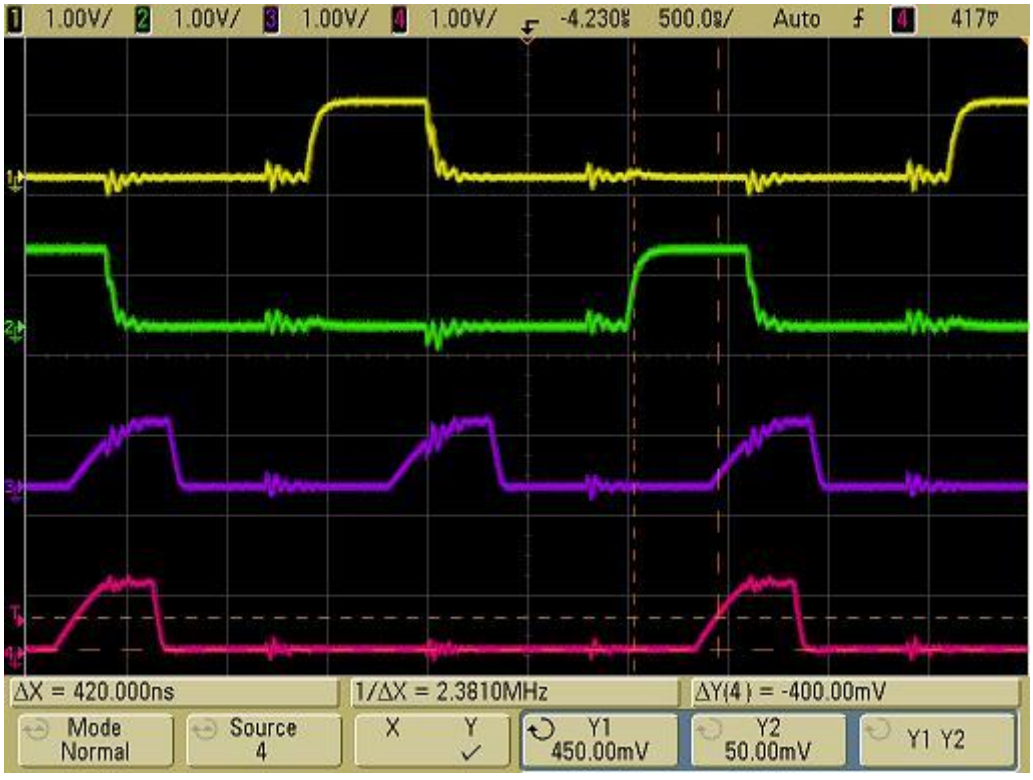


Figure 5.9 Experimental timing diagram of write and read operation (0.81V)

When the digital function of the oscilloscope is used, the timing diagram of the asynchronous SRAM which is shown in Figure 5.10 can be obtained. It is similar with the simulation results. D_0 is the signal ARW_ack, D_1 to D_4 are data output signals, D_5 and D_6 are address and data input respectively, D_7 and D_8 are read and write signal correspondingly.

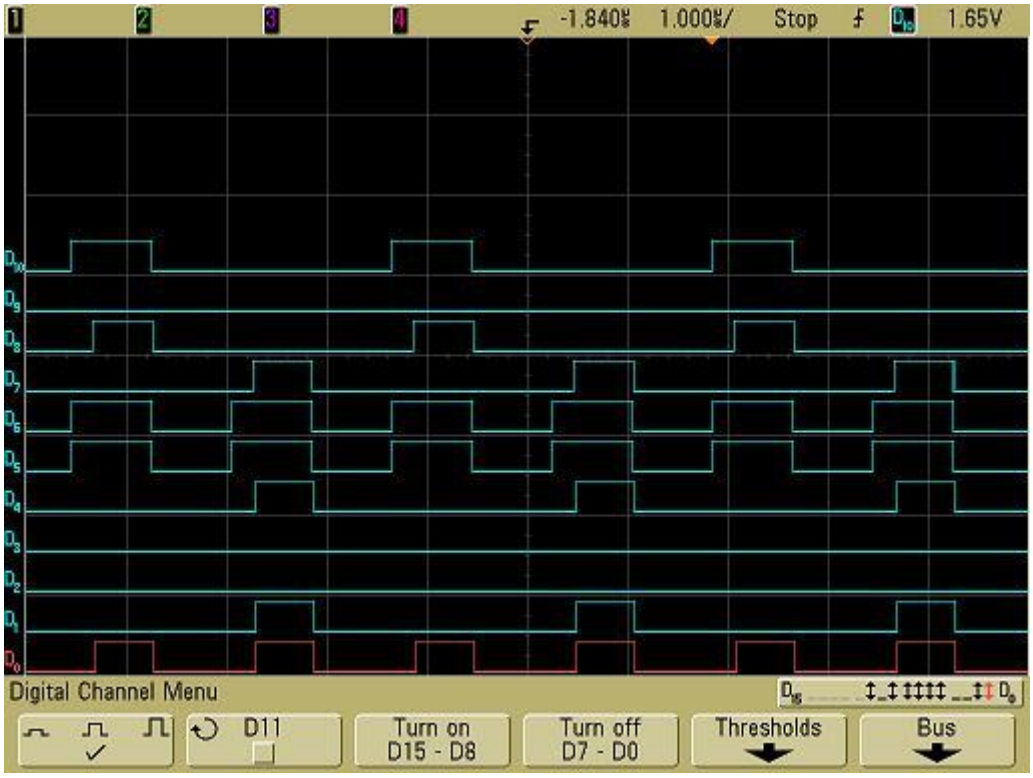


Figure 5.10 Experimental timing diagram of write and read operation using digital function of the oscilloscope

5.4 Summary of the performance

When the 16*8 bits asynchronous SRAM is tested without connected with the asynchronous microcontroller core, it works fine with the supply voltage between 0.81V and 3.5V. When the SRAM is connected with the asynchronous mi-

crocontroller core, it also works fine and can provide the right data when the supply voltage is as low as 0.81V. The experimental results show that the delays t_{WA}, t_{RA} and t_{RO} of 16*8 bits asynchronous SRAM are 19.0ns, 17.0ns and 12.6ns at 3.3V and 176.0ns, 168.0ns and 140.0 ns at 1.0V.

The 128*8 bits asynchronous SRAM embedded in asynchronous 8051 microcontroller is verified through the correct function of the microcontroller. During the test, the microcontroller works well with the supply voltage between 0.81V to 3.5V. At nominal supply of 1.2V, the microcontroller can achieve 0.33MIPS of operating speed. However, due to the pad limitation of asynchronous microcontroller, the delays (t_{WA}, t_{RA} and t_{RO}) cannot be directly probed.

The photographs of the 16*8 bits and 128*8 bits asynchronous SRAM are shown in Figure 5.11 and Figure 5.12, respectively.

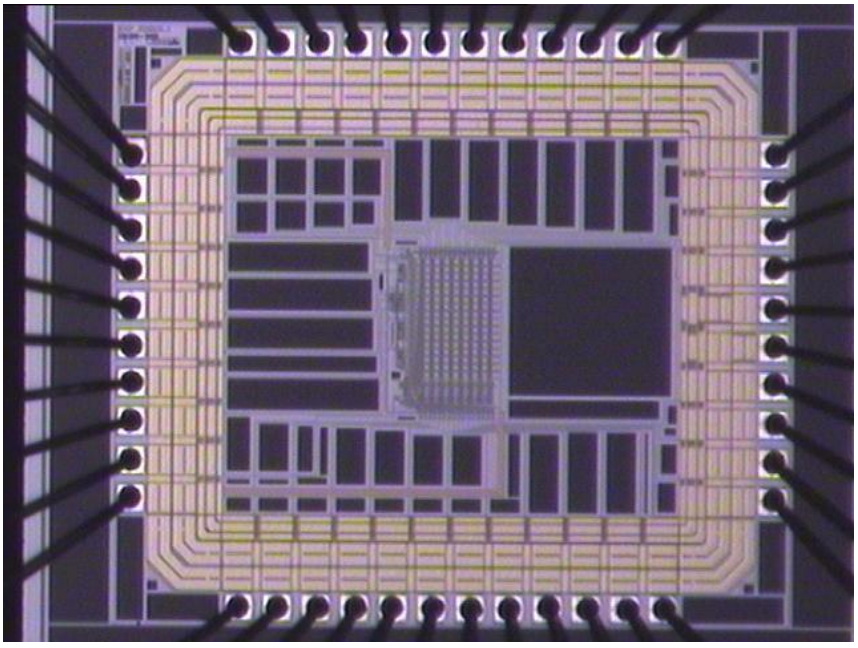


Figure 5.11 Die photograph of the 16*8 bits asynchronous SRAM

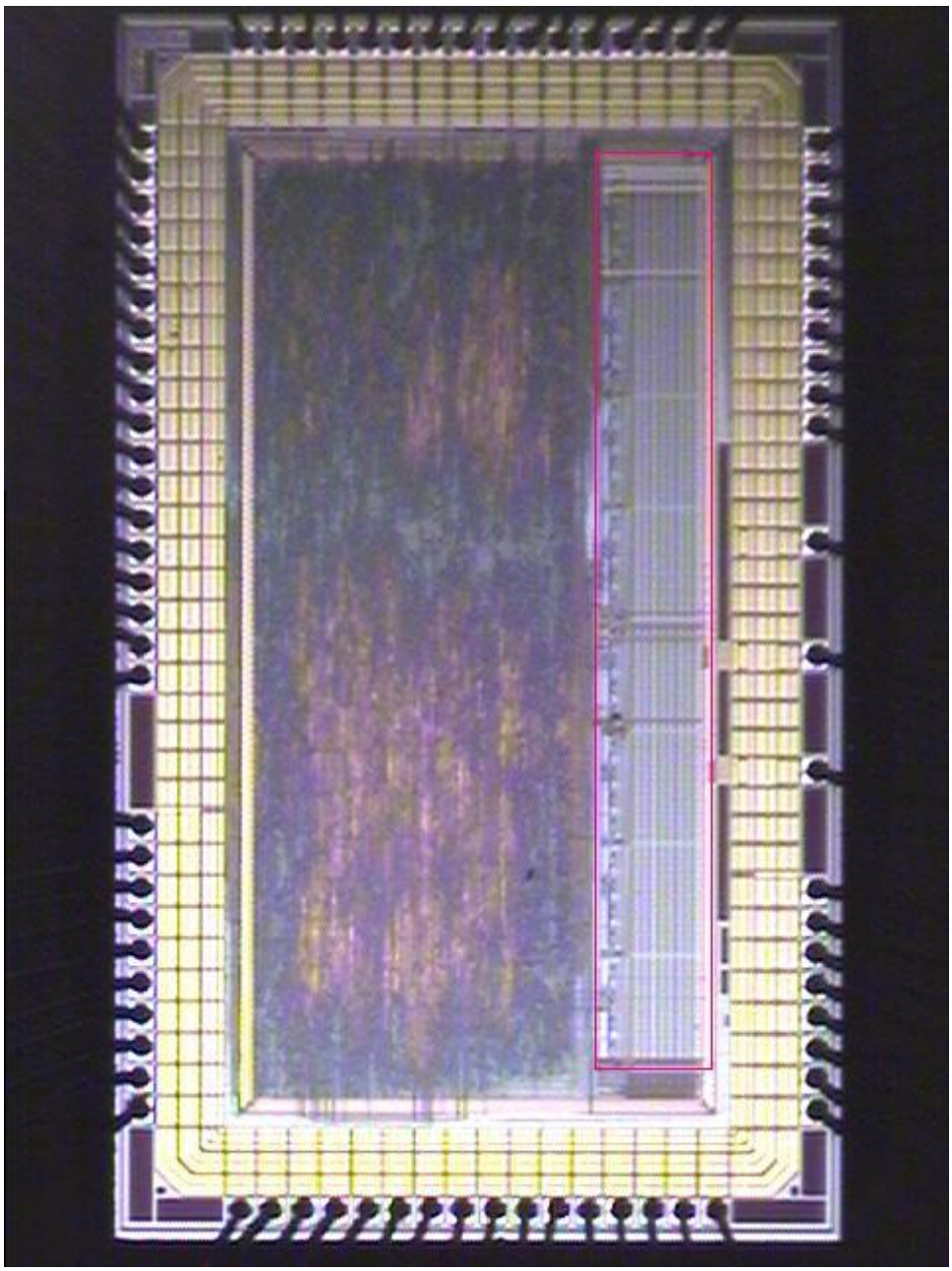


Figure 5.12 Die photograph of the asynchronous 8051 microcontroller with the 128*8 bits asynchronous SRAM in the red box

CHAPTER 6

CONCLUSIONS

In this thesis, an asynchronous SRAM using 4-phase dual-rail protocol has been designed and evaluated. Compared with some applications which use synchronous SRAM and synchronous to asynchronous logic transform interface, the proposed asynchronous SRAM can be used directly for the asynchronous 8051 microcontroller designed in this project group.

As asynchronous logic is not as familiar to the people as the well known synchronous logic, some asynchronous circuit fundamentals and the summary of circuit designs on asynchronous logic have been presented. The self-timed memory cell which is the key part of the asynchronous SRAM is introduced to compare with the traditional memory cell. Some other parts of the asynchronous SRAM have also been presented followed by their simulation results.

Two versions of asynchronous SRAM which are 16*8 bits and 128*8 bits SRAM have been fabricated with AMS 0.35um double-poly four-metal CMOS technology. The former one is used to verify the functionality and the integrity of the design and is fabricated separately with the asynchronous microcontroller core. The latter one is fabricated with the microcontroller as one chip. Through the experimental testing, the two versions SRAM are proved working well at the power supply between 1V and 3.3V which is our design objective.

Further improvements include optimizing of the bit line size, increasing the density of bit cells and modularizing the SRAM. Reducing the bit line length and

using memory bank can decrease the parasitic capacitance of each bit line and increase the operation speed. Increasing the bit cells density can reduce the silicon area and modularizing the SRAM can make it adaptive to different requirements in microcontroller design.

BIBLIOGRAPHY

[1] J. P. Kulkarni, K. Kim, K. Roy, "A 160 mV robust Schmitt trigger based subthreshold SRAM", *IEEE Journal of Solid-State Circuits*, vol. 42, no. 10, pp. 2303-2313, Oct. 2007.

[2] Jan M. Rabaey, Anantha Chandrakasan, Borivoje Nikolic, *Digital Integrated Circuits - A Design Perspective*, Prentice Hall, 2003.

[3] Jens Sparsø, Steve Furber, *Principles of asynchronous circuit design - A systems Perspective*, Kluwer Academic Publishers, 2001.

[4] C. L. Seitz, *Introduction to VLSI System.* Reading, MA: Addison-Wesley, Chap. 7, pp. 218-262, 1980.

[5] Y. K. Tan and Y. C. Lim, "Self-timed system design technique," *Electronics Letters*, vol. 25, no. 5, pp. 284-286, Mar. 1990.

[6] J. Sparsø, C. D. Nielsen, L. S. Nielsen, and J. Staunstrup, "Design of self-timed multipliers: A comparison," *IFIP Transactions A (Computer Science and Technology),* v A-28, pp. 165-179, 1993.

[7] S. Hauck, "Asynchronous design methodologies: An overview," in *Proceedings of the IEEE*, vol. 83, pp. 69-93, Jan. 1995.

[8] S. B. Furber, D. A. Edwards, and J. D. Garside. "AMULET3: a 100 MIPS asynchronous embedded processor". in *Proceedings of International Conference on Computer Design,* pp. 329-334, 2000.

[9] L. S. Nielsen and J. Sparsø. "Designing asynchronous circuits for low power:

An IFIR filter bank for a digital hearing aid". in *Proceedings of the IEEE,* vol. 87, no. 2, pp. 268-281, February 1999.

[10] J. D. Garside, "A CMOS VLSI implementation of an asynchronous ALU," *IFIP Transactions A (Computer Science and Technology)*, vol. A-28, pp. 181-192, 1993.

[11] S. J. Muscato and A. Albicki, "Locally clocked microprocessor," in *Proceedings of Great Lakes Symposium on VLSI Design Automation of High Performance VLSI Systems*, pp. 47-51, 1993.

[12] J. A. Tierno, A. J. Martin, D. Borkovic, and T. K. Lee, "A 100-MIPS GaAs asynchronous microprocessor," in *Proceedings of the IEEE*, vol. 82, pp. 43-49, 1994.

[13] Edward H. Frank, Robert F. Sproull, "A self-timed static RAM", in *Proceedings of Caltech Conference on Very Large Scale Integration*, pp. 275-285, 1983.

[14] J. A. Tierno, A. J. Martin, "Low-energy asynchronous memory design", in *Proceedings of International Symposium on Advanced Research in Asynchronous Circuits and Systems*, pp. 176-185, 1994.

[15] A. Chandna, R. B. Brown, "An asynchronous GaAs MESFET static RAM using a new current mirror memory cell", *IEEE Journal of Solid-State Circuits*, vol. 29, no. 10, pp. 1270-1276, Oct. 1994.

[16] L. S. Nielsen, J. Staunstrup, "Design and verification of a self-timed RAM", in *Proceedings of International Conference on Very Large Scale Integration,*

VLSI 95, pp. 751-758, 1995.

[17] V. W. Y. Sit, C. S. Choy, C. F. Chan, "A four-phase handshaking asynchronous static RAM design for self-timed systems", *IEEE Journal of Solid-State Circuits*, vol. 34, no. 1, pp. 90-96, Jan. 1999.

[18] S. H. Tan, P. Y. Loh, M. S. Sulaiman, "A low-power high-speed 1-Mb CMOS SRAM", in *Proceedings of IEEE International Workshop on Electronic Design, Test and Applications*, pp. 281-286, 2006.

[19] Kai Zhou, Zhongli Liu, Zhiqiang Xiao, Genshen Hong, "Radiation hardened 128K PDSOI CMOS static RAM", in *Proceedings of International Conference on Solid-State and Integrated Circuit Technology*, pp. 1922-1924, 2006.

[20] N. Weste, K. Esraghian, *Principles of CMOS VLSI design – A systems Perspective, 2nd edition*. Addison-Wesley, 1993.

[21] Chao Xue, Xiang Cheng, Yang Guo, and Yong Lian, "The Design of a Sub-Nanojoule Asynchronous 8051 with Interface to External Commercial Memory", *8th IEEE International Conference on ASIC*, Oct. 20-23, 2009.