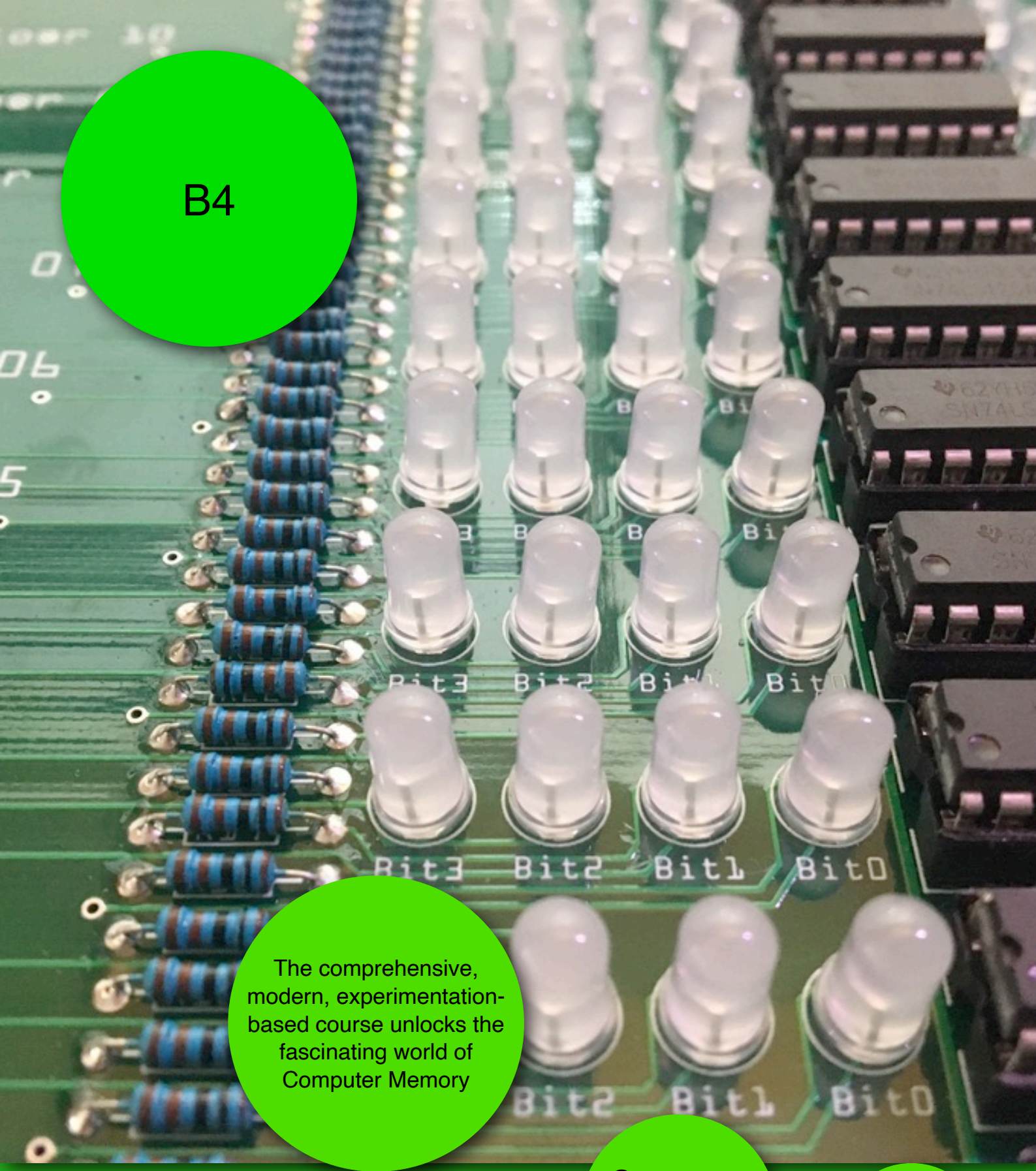


B4

The comprehensive, modern, experimentation-based course unlocks the fascinating world of Computer Memory

Computer Memory Kit

0010
0000
0010
0000

DIGITAL TECHNOLOGIES INSTITUTE

"*After growing wildly for years, the field of computing appears to be reaching its infancy.*"

John Pierce
American engineer and author
1910-2002

Table of Contents

Included parts 5

Hello parents, teachers and students 6

B4's parts.....6

Wires and connectors.....8

A word about power.....9

Please look after me.....9

About binary and decimal numbers 10

Lessons 10

Overview 10

Lesson 1: Getting to know the RAM module 11

Lesson 2: How does computer memory work?.....15

Lesson 3: Putting theory to practice.....19

Chips.....19

Wires 20

Lesson 4: How can a computer remember things?.....22

Logic gates.....22

Building memory with logic gates.....24

Lesson 5: Engineering.....28

Lesson 6: Integrating the RAM Module into the B4 Computer Processor Kit..30

Summary.....35

Appendix A: Further reading 36

Appendix B: Troubleshooting 37

Appendix C: Solutions 38

WARNING:
CHOKING HAZARD - Small Parts
Not for children under 3 years.
PHOTOSENSITIVE EPILEPSY -
Some of the experiments produce light flashes that can potentially trigger seizures in people with photosensitive epilepsy

Safety instructions

The B4 operates on 5 Volts and only draws a few milliamperes. Nevertheless, it is an electrical device and should be handled as such. We recommend to treat it with care, and to keep it on a non-conductive, dry and level surface. Do not scratch the surface of the printed circuit boards with sharp or metallic instruments, as this might damage the wires.

Acknowledgements

We would like to thank Charles Petzold, the author of 'Code: The Hidden Language of Computer Hardware and Software', published in 1999. His book has both inspired and guided the design of the B4. We recommend it as additional reading material for students.

We would further like thank Henrik Maier from proconX for his guidance and feedback on the electrical engineering design, fabrication and component selection, which has been invaluable to transforming the B4 from a breadboard prototype to a robust design that can be used in the classroom.

Special thanks to Dr. Hayden White for his support and input which have been invaluable to get the B4 off the ground. His regular feedback on the development of the B4 has influenced many of the design decisions.

A big thank you is owed to Mrs. Sharon Singh and her year 8 students at St. John's Anglican College in Forest Lake, QLD, have prompted the idea that led to the RAM module and this guide.

Dr. Karsten Schulz, CEO, The Digital Technologies Institute.

Included parts

1x RAM Module
2x Variable

2 x 4 Pin Wires
2 x 2 Pin Wires
1 x 1 Pin Wires
1x USB Cable

1x Printed Student Handbook

Power Consumption:
5V, 350mA, 1.75 W DC.

This product complies with the Restriction of Hazardous Substances Directive and is lead free.

The illustrations in this handbook can slightly differ from the actual modules. However, the functionality is the same.

This handbook has been made with great care. Should you find errors or have ideas to improve it, please email us at enquiries@digital-technologies.institute.

Designed and manufactured in Australia
(c) Digital Technologies Institute PTY LTD, 2016-18 AD. All rights reserved.

Hello parents, teachers and students

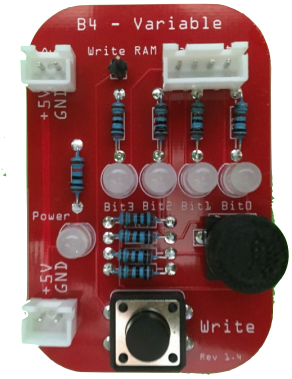
Computers have a fantastic ability to remember things. We take it now for granted that our laptops have working memories of 8 gigabytes or more. Also, they can store terra-bytes of data on their hard-discs. In this book, we look at the computer's working memory, the RAM. Typically, computer RAM is hidden inside a little black box. We have opened it up and enlarged it. The result is a big and interactive memory board that shows us some of the inner workings of a data RAM module. This course consists of practical and theoretical parts.

We have designed the B4 Computer Memory Kit also to work stand-alone. However, If you already own a B4 Computer Processor Kit, then this new RAM module can be used to replace the Data RAM module from the B4 Computer Processor Kit to show in even greater depth the inner workings of a computer.

Now let's see what's in the box:

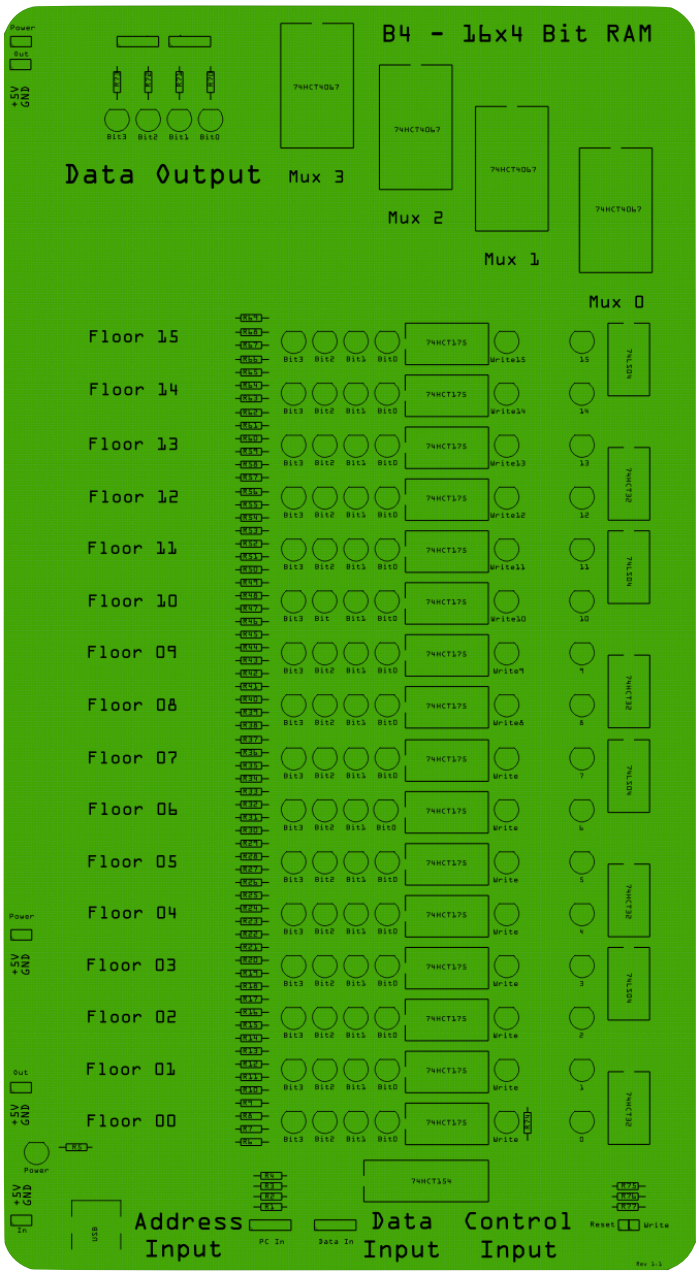
B4's parts

This kit consists of three modules which are needed to explore computer memory.



With the **Variable** we can produce binary data simply by rotating the knob. You can think of it as a variable in a computer program. With its button we store data RAM module. This kit ships with two Variable modules. The knob of the Variable was made on a 3D printer.

The **Random Access Memory** (RAM) module has room for 16 numbers that are 4 bit wide, thus representing the decimal numbers of 0 to 15. The Data RAM is a 16 by 4, or 16x4, memory module. It has a total storage capacity of 8 bytes.



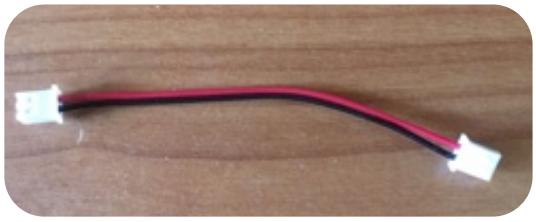
We now have a basic understanding of the modules of our kit. Don't worry if you haven't understood everything yet. We will revisit each module in more depth during the following experiments.

Wires and connectors

In order to connect the B4 modules with the computer and with each other, the B4 comes with 4 types of wires. They are:

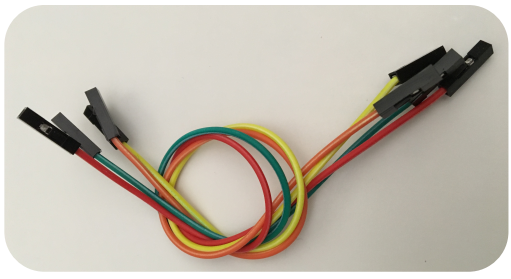
A **USB cable** to provide electricity from a power source to the B4's Program Counter module and from there, to all other modules connected to the Program Counter. You can connect the USB cable to a PC, Laptop, USB Hub, USB battery, or any other suitable 5V power source with a USB port.

2 pin **power wires** with black and red wires. They are part of the B4's power distribution system and transport electricity from module to module. Each module has one power input and 1-2 power outputs.



4 pin **data wires**. These transport 4 bit data and program counter signals between the modules.

1 pin **control wires**. They transport operation codes and instruct some of the modules of the B4 to do special things, such as storing data. The 1 pin wires come in many different colours. However, they all work the same and their colour has no influence on their function

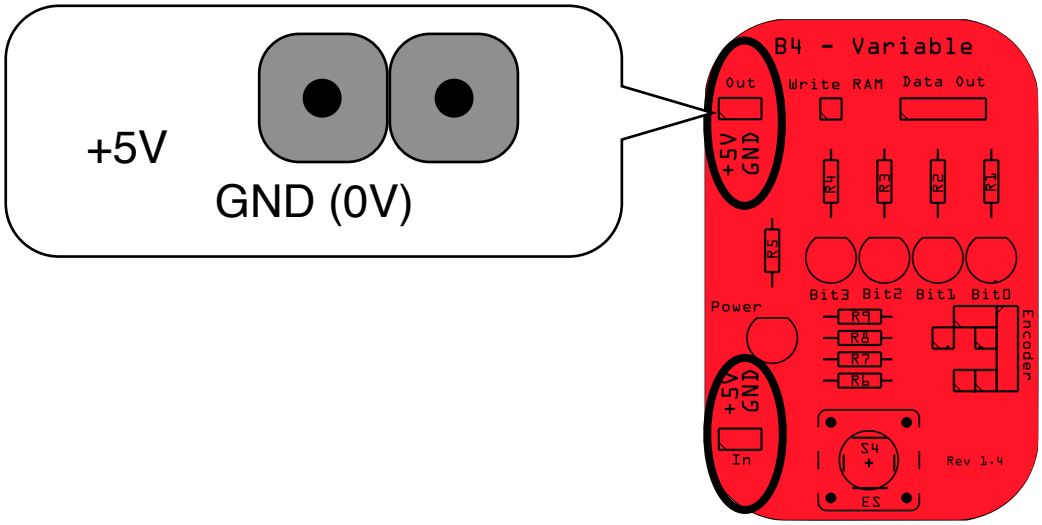


You will find corresponding connectors on the modules. The 2 and 4 pin connectors are directional and the wires will easily click into them. Unless you apply excessive force you should not be able to accidentally plug them in the wrong way.

In the diagrams in this book, we use the following wiring notation. A solid line denotes a power or data wire. A line with two arrows denotes a 1-pin control wire. This is just to make the setup a little bit easier for you.

Symbol	Meaning
	power or data wire
	control wire

A word about power
Each of the B4's modules has an electric power distribution system on the left hand side of the modules.



+5V is on the left and GND (Ground, or 0V) is on the right. The wires will always connect in the right way, **but sometimes we will need to connect a single wire to either +5V or GND during some of the experiments**. When asked to connect to +5V, just plug a single wire into the left pin of the power node. If asked to connect to GND, plug a single wire into the right pin of a power node.

Please look after me
The B4 is fairly robust and will last a long time with proper care. As long as you don't plug wires into connectors they are not designed to go in and as long as you don't drop the modules, step on them or use them as a doorstopper, things should be just fine.
Always only plug the 2 pin wires into 2 pin connectors. the same applies to 4 pin wires and connectors. **Under no circumstances plug a 2 pin wire into a 4 pin connector**.

About binary and decimal numbers

To be clear about the distinction of binary and decimal numbers, we add a capital 'B' to binary numbers. This way we can distinguish for example 11 (decimal eleven) from B11 (decimal 3), or 10 (decimal ten) from B10 (decimal 2). In this handbook, it is assumed that you are already familiar with binary numbers. If not, then quickly download the handbook of the B4 Computer Processor Kit (if you are a secondary school student) or the B4 Primary School starter kit (if you are a primary school student) and check out the first experiment. These books are available as free PDFs from the Digital Technologies Institute website at: https://www.digital-technologies.institute/handbooks

Ok, that is enough preparation for now. We will collect more details as we work through the experiments. Let's get started.

Lessons

Overview
In this handbook, we have prepared several experiments that will help you to get to know the modules of the B4, how they are being used and what functions they perform. Most experiments consist of one or more experiments. You will learn how to combine modules to that they perform functions together, which they could not perform individually. Ultimately, you will develop a computer and learn about coding from the ground up. You will also learn how a computer works internally and what critical role timing plays in the proper function of a computer's internal and external communication.

We recommend that you take the experiments in sequence. But if you are already a computer genius, feel free to jump around. We should mention that the B4 can do much more than what this handbook says. Feel free to explore and try out different things as you like.

Lesson	Title	Learning Objectives
1	Getting to know the RAM module	Function of the RAM module. Reading and writing of data. How to address memory. Memory as a fundamental component of a computer to remember.
2	How does Computer Memory work?	We identify the main components and their function needed to make a RAM module
3	Putting theory to practice	We find the main components and their connections from the previous lesson on the RAM module board.
4	How can a computer remember things?	We explore the inner workings of a RAM modules by visiting binary logic and gates.
5	Engineering	We look at ways to building logic gates with switches.
6	Integrating the RAM Module into the B4 Computer Processor Kit	We integrate the RAM module into the B4 Computer Processor Kit, by replacing the B4's Data RAM module.

Lesson 1: Getting to know the RAM module

Modules Required: RAM, 2x Variable

Analyse
You can imagine our RAM module as a 16-floor high-rise with 4 rooms on each floor. To store data into the RAM Module, we first want to tell it on which floor we want our data to be stored. We call this the **address**. Then, we give it 4 bits of **data** and finally tell it to actually store it with a **control** command.

Floor 15				
Floor 14				
Floor 13				
Floor 12				
Floor 11				
Floor 10				
Floor 9				
Floor 8				
Floor 7				
Floor 6				
Floor 5				
Floor 4				
Floor 3				
Floor 2				
Floor 1				
Floor 0				

RAM: 16x 4 bit.

Build

1) Place the RAM module and the two Variables in front of you as shown in the following figure
2) Connect the left Variable to the power port of the RAM module with a 2-pin wire.
3) Connect the right Variable to the power port of the left variable with a 2-pin wire.
4) Connect the left Variable to the Address Input of the RAM module with a 4-pin wire.
5) Connect the right Variable to the Data Input port of the RAM module with a 4-pin wire.
6) Connect the Write RAM pin of the right Variable to the Write pin of the RAM module with a 1-pin control wire.
7) Connect the RAM module to a USB power source with the USB cable.

Experiment 1.1

1) Set the left Variable to B0000.
2) Set the data on the right Variable to B1010.
3) Press the button on the right Variable to store the data. The RAM's LEDs will now display B1010. Congratulations, you have just stored 4 bit of data in the RAM at Address B0000.

Experiment 1.2

1) Set the left Variable to B0001. That's our new address where we want to write the next 4 bit of data
2) Set the data on the right Variable to B0010.
3) Press the button on the right Variable to store the data. The RAM's LEDs will now display B0010. You have just stored 4 bit of data (the value 0010) in the RAM at Address B0001.

Observe

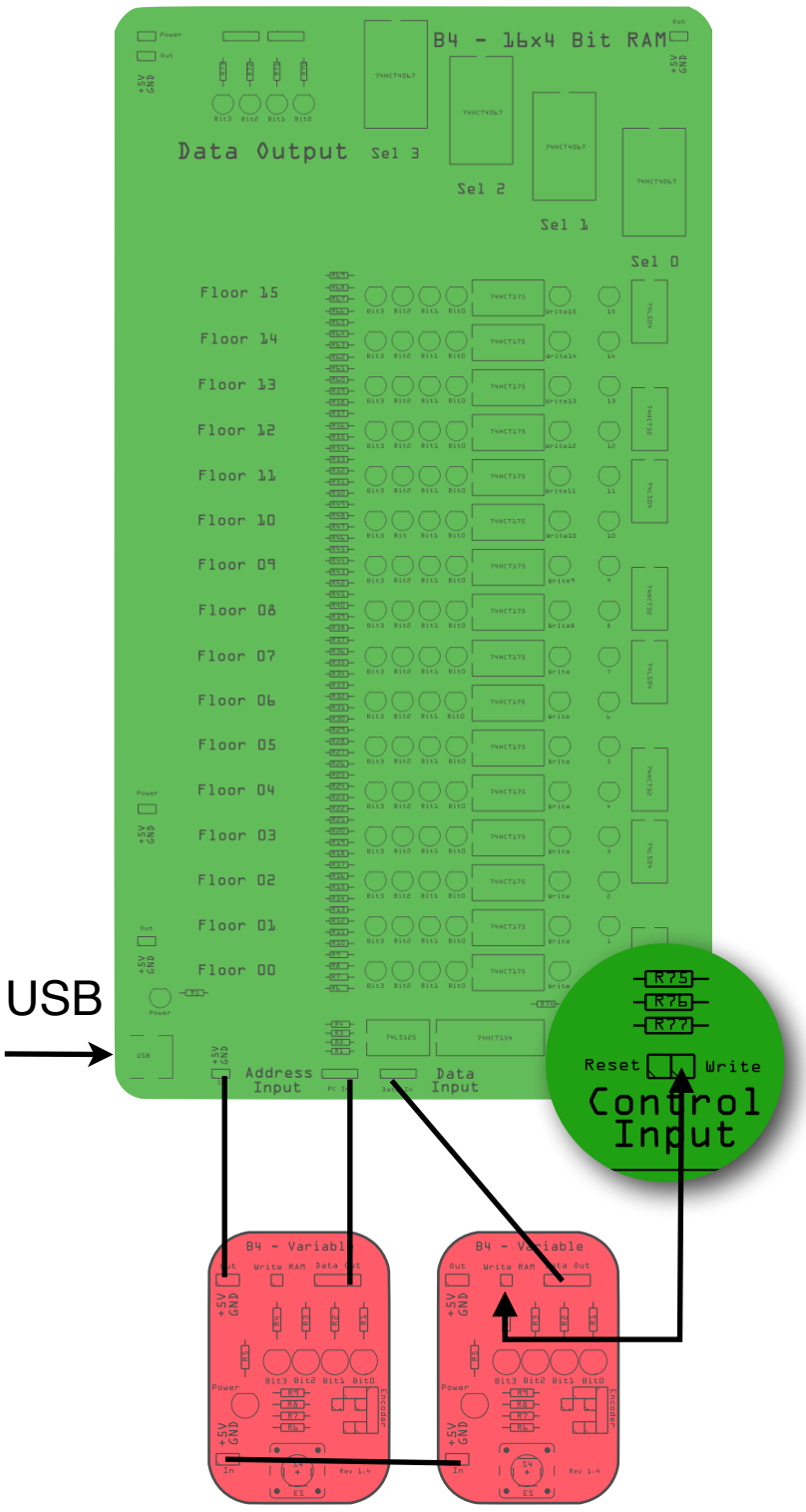
The data from the right Variable always appears on the floor number that the left Variable is set to.

Observe

When you change the value of the left Variable (the address), a red LED on the right hand side of the RAM module moves to the corresponding floor number. For example, when the address is B0111 (decimal 7), then the red LED on the 7th floor is lit.

Observe

When you press the button on the right Variable, a green LED will lite up on the RAM module. The green LED is on the same floor as the red LED. When we press the button, we write data into the Data RAM at the address set by the left Variable.



Setup of experiment 1

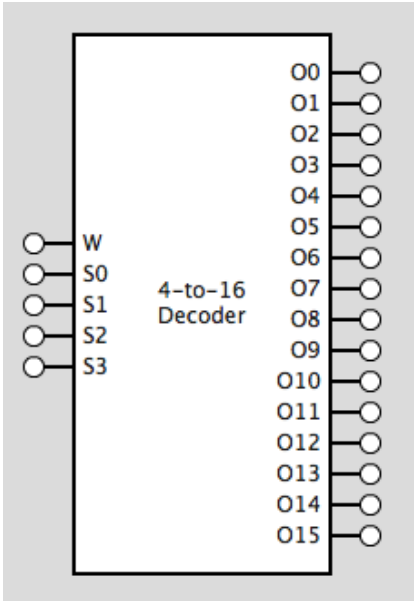
Evaluate
We can individually address each floor of the RAM module. We call this ‘addressing memory’. A computer program can read and write data at each address individually, by telling the RAM module the address. What other types of addresses do you know in the world?

During programming and operation of the B4, ensure that the RAM module remains connected to power. This ensures that the RAM modules doesn’t forget its data.

Lesson 2: How does computer memory work?

Analyse
To make a RAM module, we need three basic parts 1) The memory cells. 2) A Decoder 3) A Selector The memory cells are pretty obvious. We will later look at how they work. But what are the functions of the decoder and of the selector?

Decoder
The decoder helps us to choose the memory cell that we want to write information into. Without the decoder, each memory cell would need its own connector plug on the board. This would make the RAM overly complicated. So, for 16 memory cells, we would need 16 data ports. Each port has 4 bit, so our data bus would be 16x4=64 bit wide. That's not too bad. But for a 1gigabyte RAM module, we would need a data bus that is 8 billion wires wide. Can you imagine a billion wires? Noone could build this. The decoder solves this problem. It is a switches that takes a 4-bit address, and then makes one out of 16 outputs active. So, if the input if B0111, then the 7th output will be active.



4-to 16 decoder

Decoder

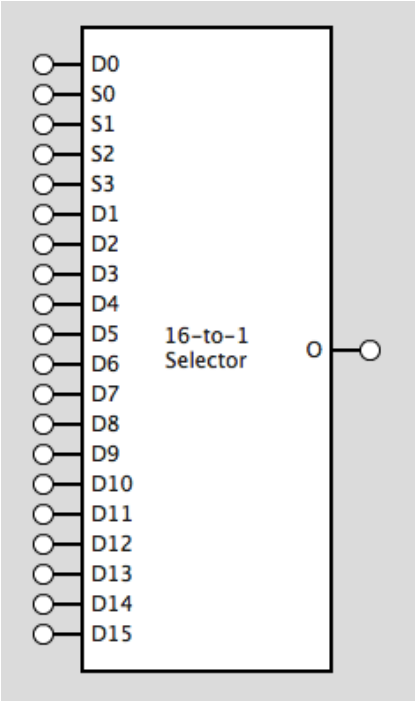
Because our decoder takes 4 bit addresses in order to make one of 16 lines active, we call it a 4-to-16 decoder. You can see a schematic picture of it above.

Selector

The selector is in charge of reading information from the 16 memory cells. When you want the 4 bits from memory cell 7, then you simply give the selector the address (B0111) and the selector will then give you the 4 bits from cell 7.

The selector is a number of cleverly-arranged switches which gives us just 4 out of 64 bits.

Actually, our selector consists of 4 selectors, each responsible for one of 16 bits. It looks like this:



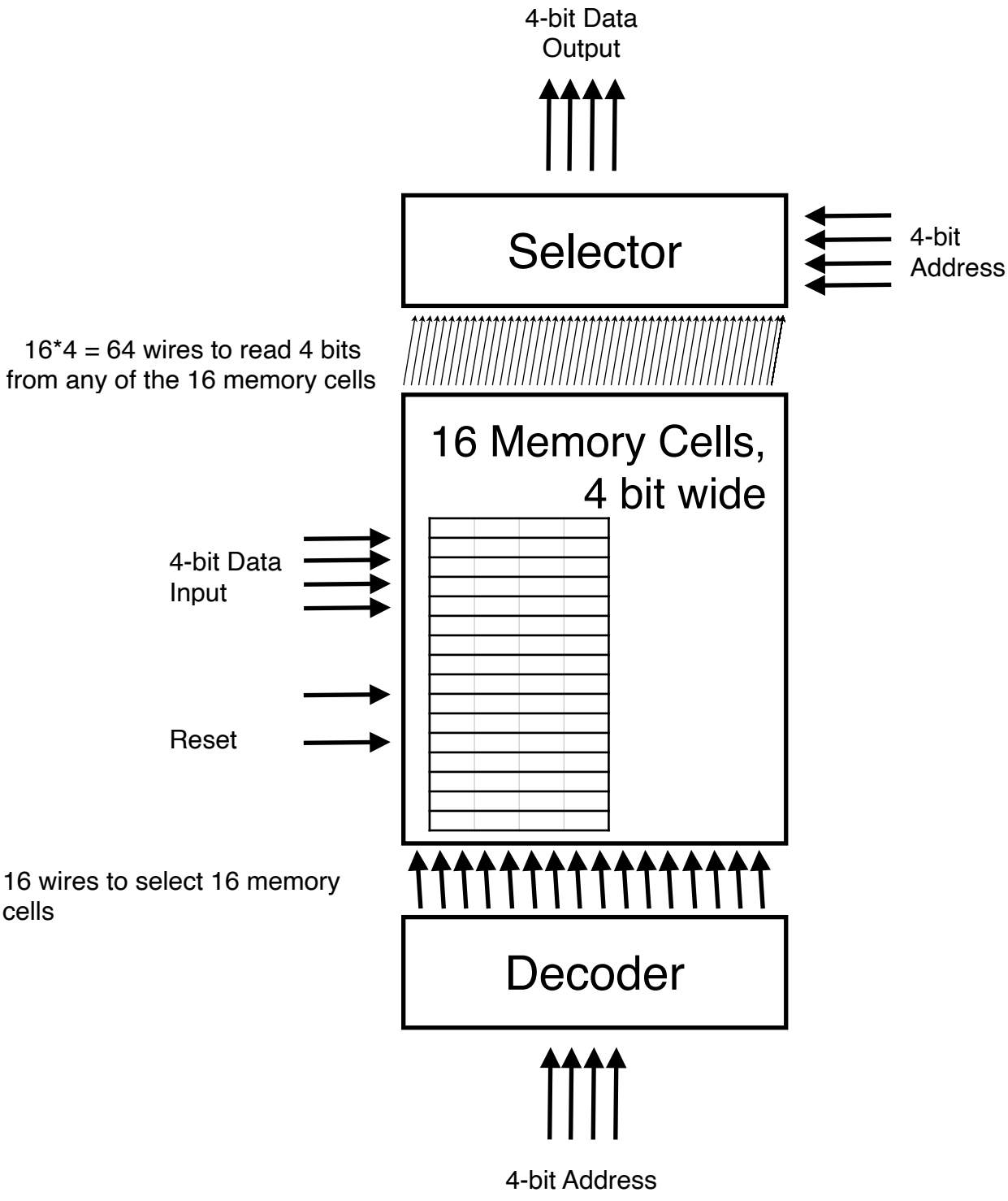
16-to-1 selector

Decoder, Selector, and Memory Cells together

The decoder, selector, and memory cells are sandwiched together as shown in the following image.

The inside of our RAM module (schematic)

Decoder, Selector, and Memory Cells together
Here, you can see how, using the 4-bit address data, the decoder selects one of 16 memory cells. The memory cells store the data and the selector makes sure that we can get the data out of the memory cells.



Evaluate
There are two main conclusions from this architecture. 1) We can only access one memory cell at any given time. 2) We can access memory cells in any (random order). We don't need to read or write sequentially. Therefore, our memory module is called a **R**andom **A**ccess **M**emory, or just **RAM**.

Evaluate deeper
In order to reduce the engineering complexity of making a RAM module, we have introduced the decoder and selector into our design. As a consequence, we can only ever access one memory cell at any given time. Every design choice has consequences. Any benefit we gain on one side comes at the cost of something else.

Question 2.1	
?	Why is our design choice with the decoder and selector a good choice?
	Can you think of a different way of designing the RAM module?
	Can you think of other design choices that had consequences for a product?

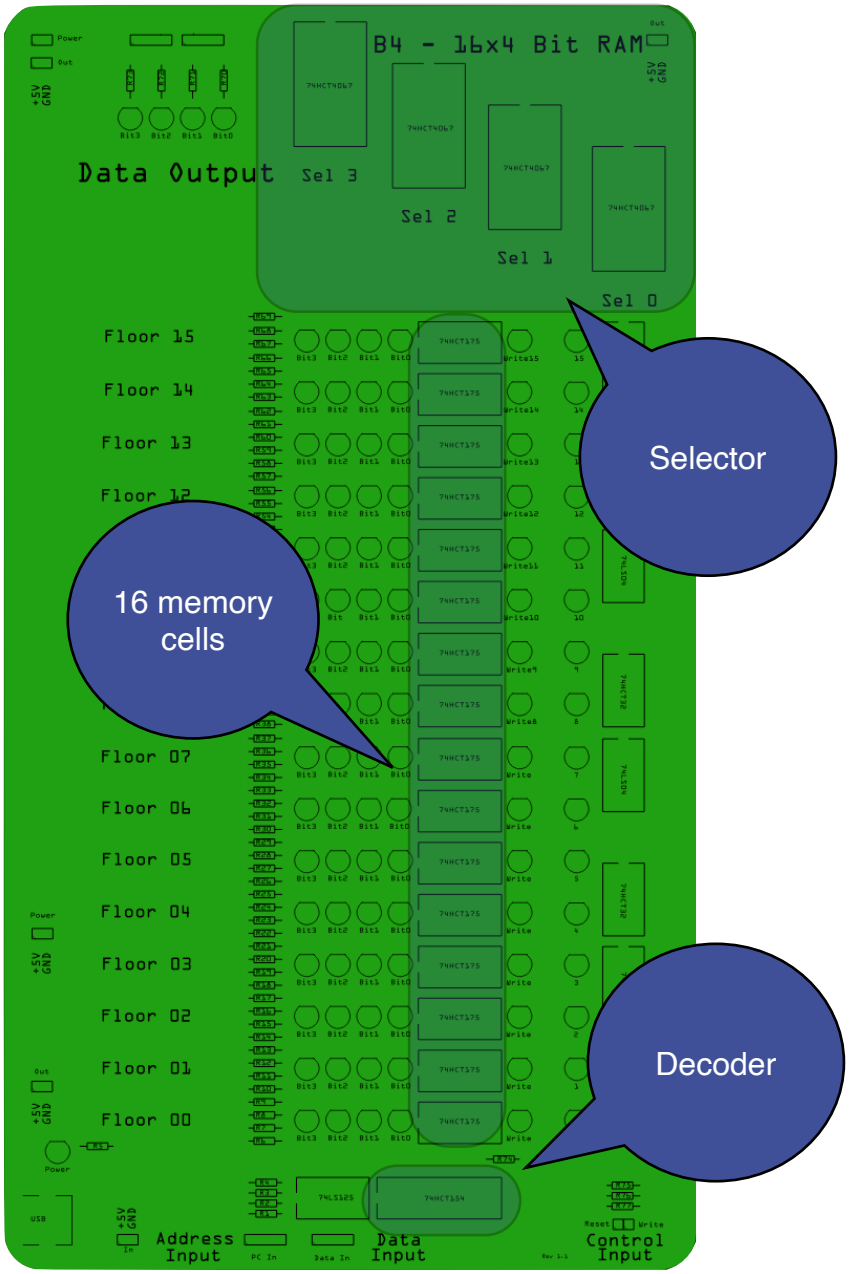
Lesson 3: Putting theory to practice

Modules Required: RAM

By now take a look at the RAM module and see if we can find any of the things that we have discussed in the previous section. Let's take a look:

Chips

First, we find the integrated circuits for the decoder, selector and memory cells. We have highlighted them for you in the following picture.

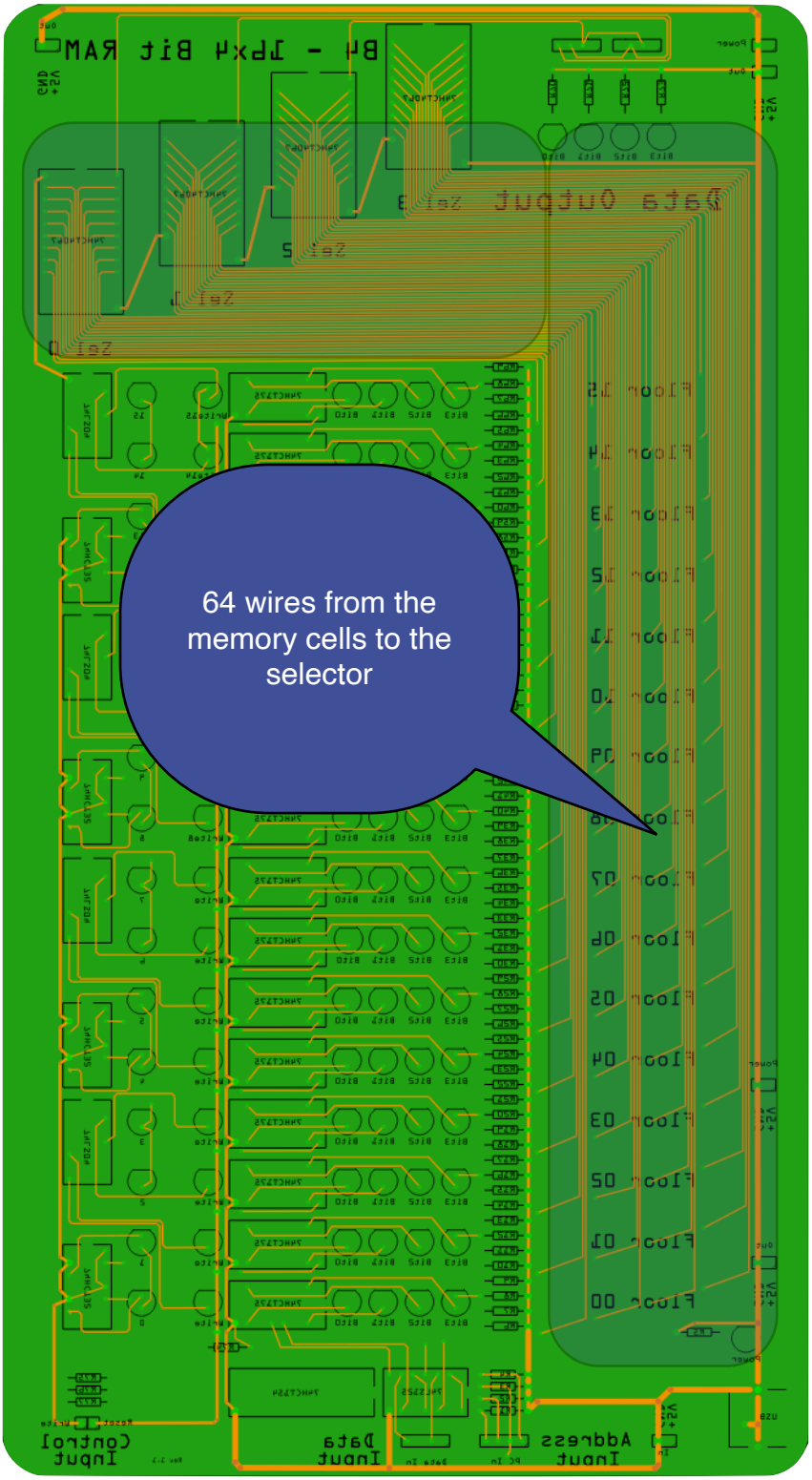


The main chips on our RAM module

The remaining chips on the right are logic chips that provide support for the decoder. They drive the two rows of red and green LEDs and coordinate the writing into the memory cells.

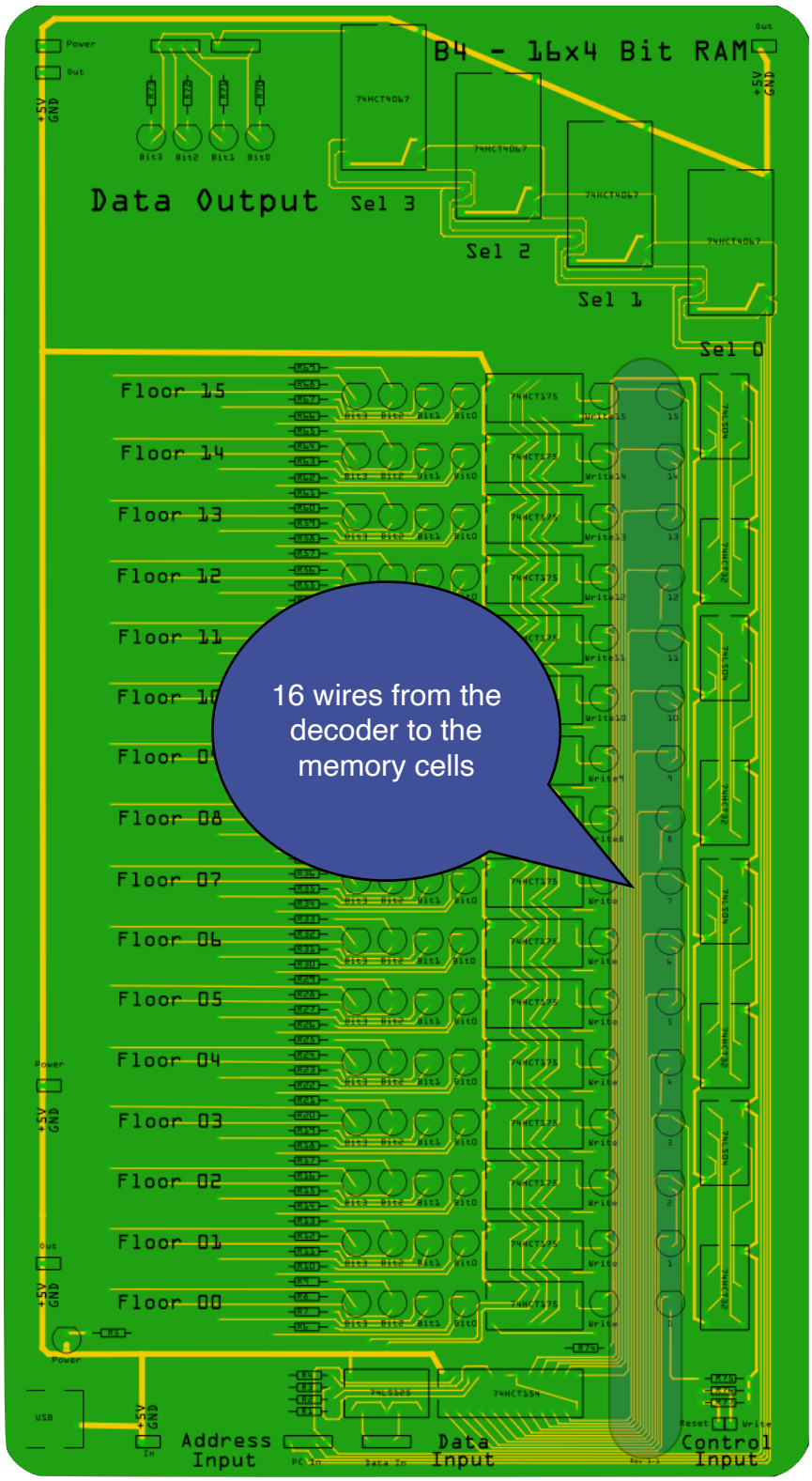
Wires

Let's now look for the wires. First, we look on the rear side of the RAM module.



64 wires from the memory cells to the Selector chips. You find them on the rear side of the RAM module

On the front of the memory module, we find the wires from the decoder to the memory cells.



16 wires from the decoder to the memory cells
You find them on the front side of the RAM module

The remaining wires provide power to the chips, or connect to resistors, LEDs, etc.

Lesson 4: How can a computer remember things?

Our RAM is made of hundreds of little switches. The switch nature is true for all the logic chips that you find in computers. These are the little black boxes with legs. They look like this:

A logic gate Integrated circuit

Logic gates

To build a simple memory cell, we need little machines, called gates, than can do certain things. These tasks are negation, AND, and Not OR (NOR)

Let's take a look: If you want both, apples and bananas you would say: " I would like apples AND bananas". This indicates to anyone hearing you that you want both.

This is called an AND operation.

apples AND bananas

Let's assume that you want to build a little machine that looks at the inputs to tell us whether your request has been met, with a simple TRUE/FALSE output statement.

We can use a truth table to determine if these conditions are met. Below, we have written the truth tables for AND, NOT (NAND), NOT-OR (NOR)

apples	bananas	output
TRUE	TRUE	TRUE
TRUE	FALSE	FALSE
FALSE	TRUE	FALSE
FALSE	FALSE	FALSE

apples AND bananas truth table

apples	output
TRUE	FALSE
FALSE	TRUE

NOT truth table

apples	bananas	output
TRUE	TRUE	TRUE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
FALSE	FALSE	FALSE

apples OR bananas truth table

A NOT-OR (NOR) gate is simply an OR gate, which has a NOT-gate at the end. So it will invert its output compared to an OR gate. This means that when the OR gate produced a TRUE output, the NOR gate produces a FALSE and when OR resulted in FALSE, then NOR will be TRUE. The NOR truth table is shown below.

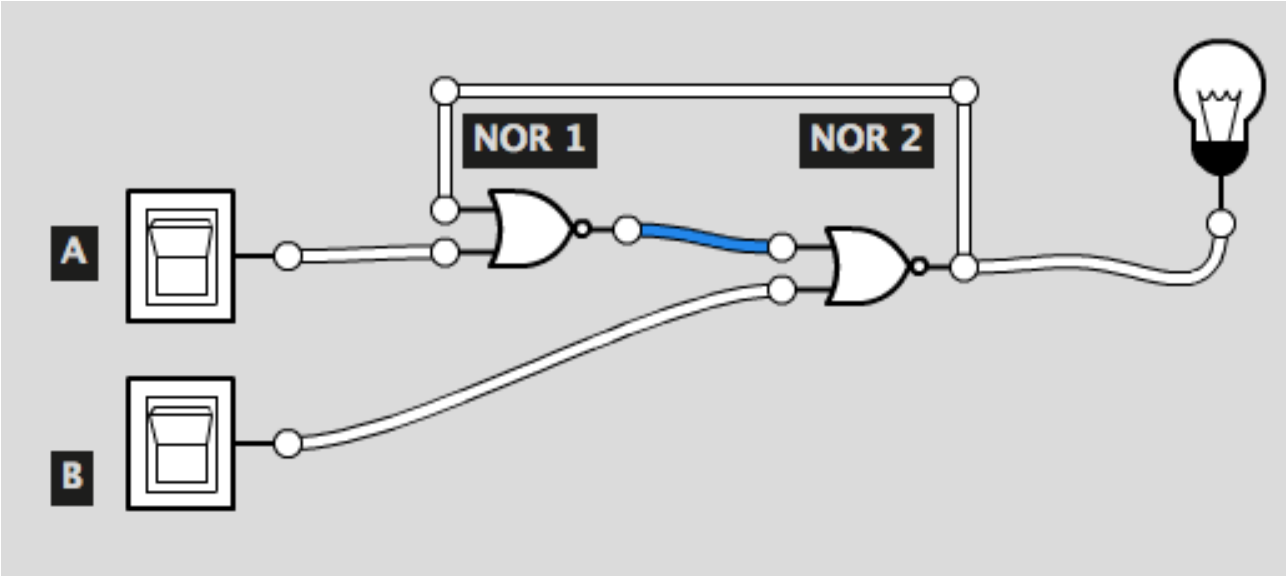
apples	bananas	output
TRUE	TRUE	FALSE
TRUE	FALSE	FALSE
FALSE	TRUE	FALSE
FALSE	FALSE	TRUE

NOR truth table A Logical Memory Machine

Cleverly combined, logic gates can remember. Memory, as you have seen throughout this handbook, is a fundamental function of a computer.

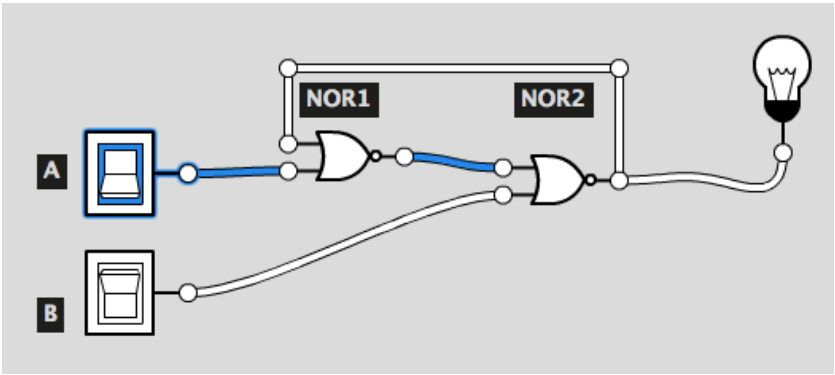
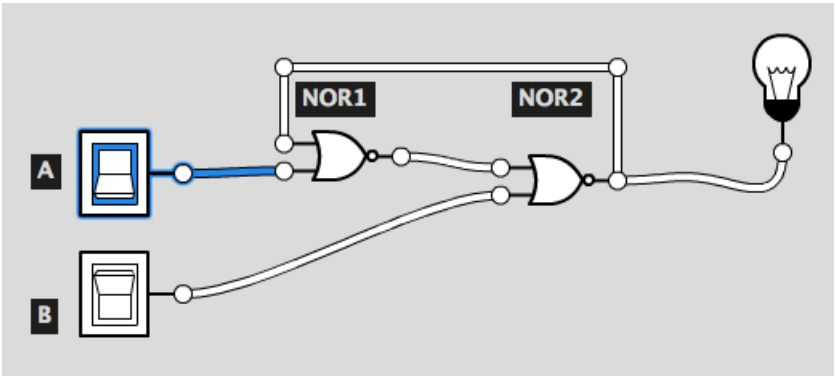
We now take two NOR gates and wire them up in a way that the output of each gate is connected to the input of the other. This, as you will see, is a common characteristic in computers: The output of one part is the input of another, and vice versa. This is called a feedback loop.

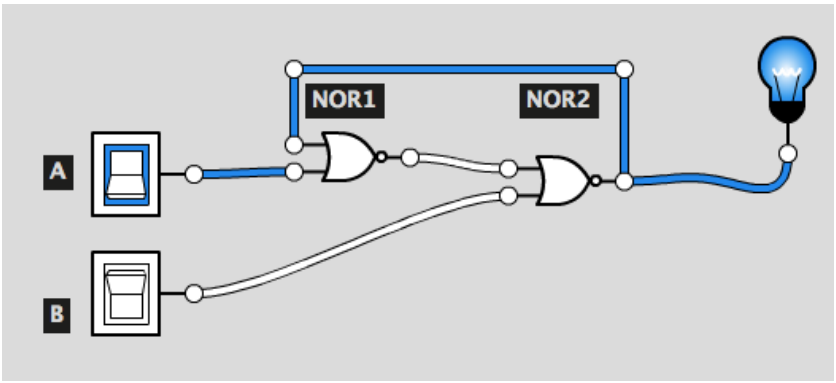
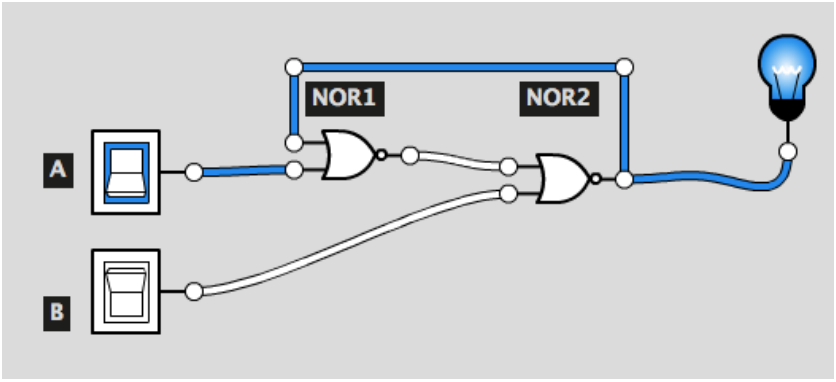
Building memory with logic gates



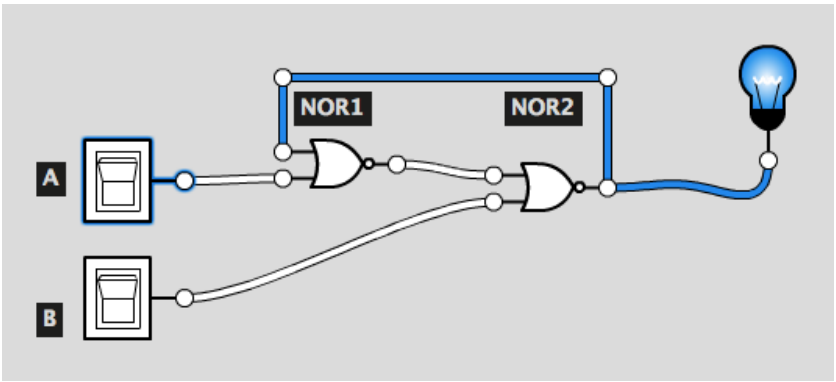
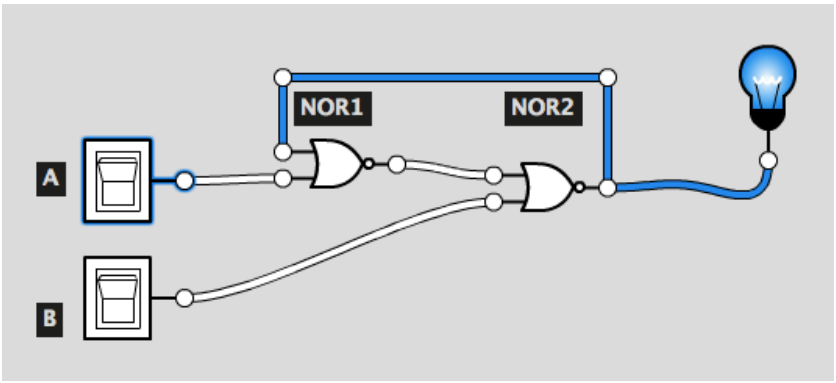
A feedback circuit with two NOR gates

Let's explore this circuit by playing with our input switches A and B. We start by pressing button A.

	Button A has been pressed. NOR1 received a 1 from button A and a 0 from NOR 2 ...
	... 0 NOR 1 is 0, which the NOR1 gate now gives to the NOR2 gate ...

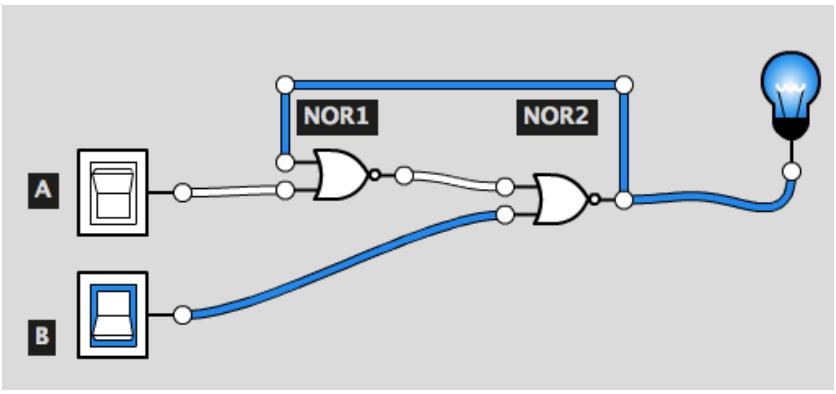
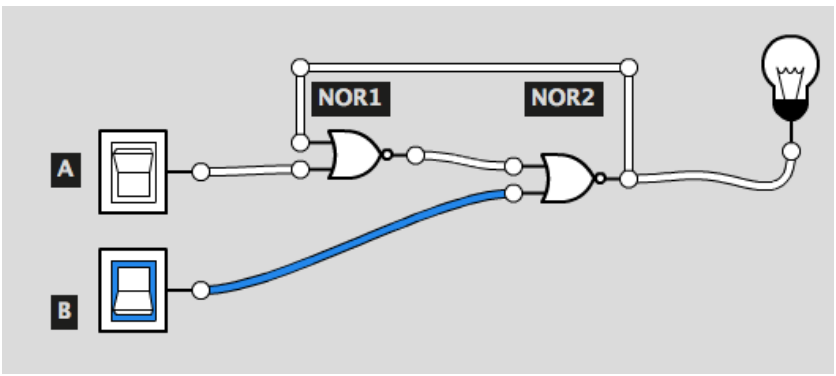
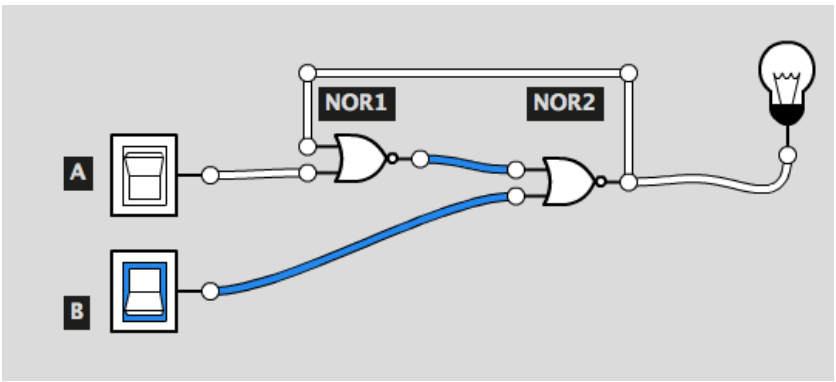
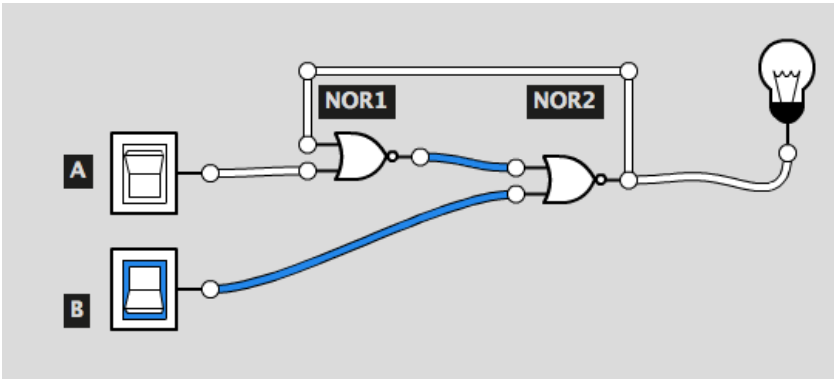
	... 0 NOR 0 is 1, which the NOR2 gate now gives to the light bulb and to the NOR1 gate ...
	... and because 1 NOR 1 is 0, the NOR1 gate does not change its output. The circuit is now stable.

But what happens when we release button A? Let's find out.

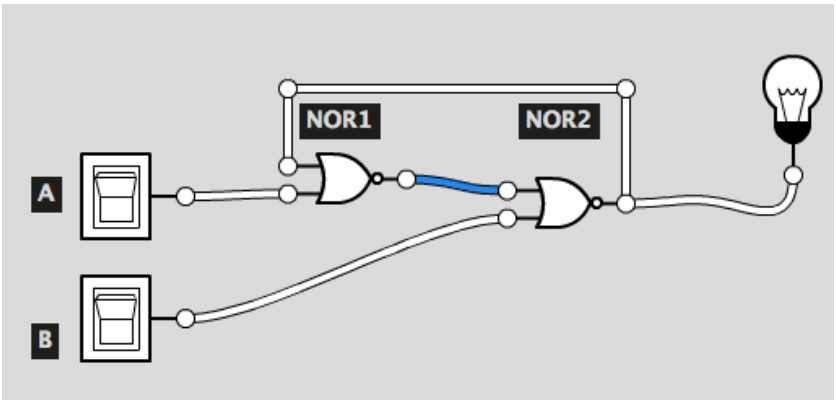
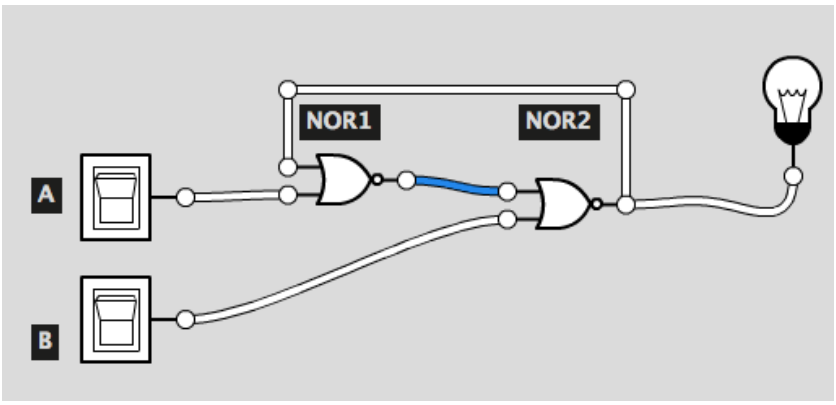
	Button A has been released. NOR1 receives a 0 from button A and a 1 from gate NOR2 ...
	... 0 NOR 1 is 0, which is the same result as 1 NOR 1 when the button A was still pressed. As a result, the output of NOR1 does not change and therefore the output of NOR 2 stays constant as well. The light stays on.

Releasing button A has no impact on the output of our circuit. It has remembered that A has been pressed. We have just constructed a 1 bit memory cell - congratulations !

Our memory cell not only needs to remember when it was activated (1), but also when it should reset to 0. An this is the function of button B. Let's now explore when button B is pressed. We continue from the previous picture.

	Button B has been pressed. NOR2 receives a 1 from button B and a 0 from NOR1 ...
	... 0 NOR 1 is 0, which the NOR2 gate now gives to the output and to the NOR1 gate ...
	... 0 NOR 0 is 1, which the NOR1 gate now gives to the NOR2 gate ...
	... and because 1 NOR 1 is 0, the NOR2 gate does not change its output. The circuit is now stable.

Finally, we release button B. This produces the following steps

	Button B has been released. NOR2 receives a 0 from button B and a 1 from gate NOR1 ...
	... 0 NOR 1 is 0, which is the same result as 1 NOR 1. As a result, the input of NOR2 does not change and therefore the output of NOR 2 stays constant as well. The light stays off.

You have probably seen the similarities between switching the buttons A and B on, and between switching them off. We can say that one gate plays the helper for the other gate to keep it either on and off. In this relationship neither of the two gates plays any greater or lesser role than the other gate. It is interesting to note that neither of the two NOR gates is able to store information by itself. However, two NOR gates, properly connected with each other has the ability to memorise information. This circuit is called a flip-flop. The first electronic flip flop was invented by two British physicists in 1918. Since then, many different types of flip-flops have been invented. Some of them use other gate types than NOR, such as NAND (NOT AND) gates. However, common to all flip-flops is the feedback characteristic between at least two gates and that flip-flops can hold a state. Some flip-flops only require one input switch, as opposed to the two input switches that our flip-flop uses. Our flip-flop is a SR NOR flip-flop. SR means ‘set-reset’ and denotes two inputs: one to *Set* the flip-flop to an output of 1 and another to *Reset* the flip-flop’s output to 0. In our SR NOR flip-flop, button A is the set button and B is the reset button.

T make a 4-bit memory cell, all we need to do is to put multiple of these flip-flops next to each other. So, for a 4-bit memory cell, we need 4 flip-flops. The flip-plop that our RAM module uses is a little bit more sophisticated and is called a level-triggered D-type flip-flop. It will only remember when a voltage rises. This is not important here, but you can do a google search if you want to learn more about this flip-flop.

Lesson 5: Engineering

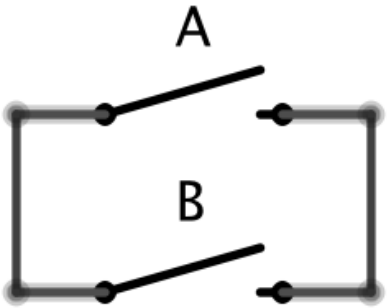
To this point, we have learned that we need gates to build memory. Each of these gates can be constructed of a cleverly-arranged set of little switches, called transistors. They have been around since the 1920's, but developed in earnest since the 1940's. Transistors are electronic switches that can be closed by applying an electric current. They can be fabricated in semiconductor materials and can be made so tiny so that billions of them fit on a chip the size of your fingernail. A typical AND or OR gate would require 2 transistors, a XOR gate 6 and a NOR gate 2.

For example, to build an AND gate, one would arrange two switches in sequence as follows:

Realising an AND gate with two switches

The circuit can only be closed by closing the switches A and B.

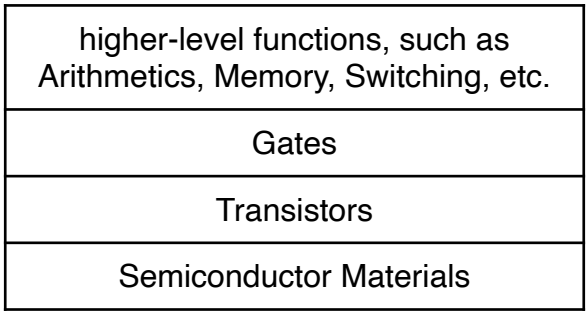
In order to make an OR Gate, we would arrange the switches in parallel, so that when either is pressed, current can flow. This would look like this:



Realising an OR gate with two switches

According to Wikipedia, the largest transistor count in a commercially available single-chip processor in the year 2016 is over 7.2 billion. This is the Intel Broadwell-EP Xeon processor.[1]

You can imagine that a chip consists of transistors that have been arranged in such a way that they form all sorts of different gates which are interconnected in clever ways so that they form arithmetic units that can perform calculations, such as adding. Other gates interact to work as memory and other gates engage in the control flow of data. This is quite extraordinary, as the underlying transistors can only switch on and off. By connecting them intelligently, we can let them perform very complex functions, which you see every day when you use a computer. Brilliant research was required to produce special materials, such as semiconductors, which have defined capabilities to conduct electronic current only when an electric charge is applied to them. In the diagram below, you see how each design step in the design process from semiconductors to transistors and from there to gates and higher-level functions has led to increased functionality and sophistication. Semiconductors were first discovered around the year 1821. It took 150 years of research and development until the first integrated micro-controller, the Intel 4004, was released.



[1] Source: Wikipedia: https://en.wikipedia.org/wiki/Transistor_count

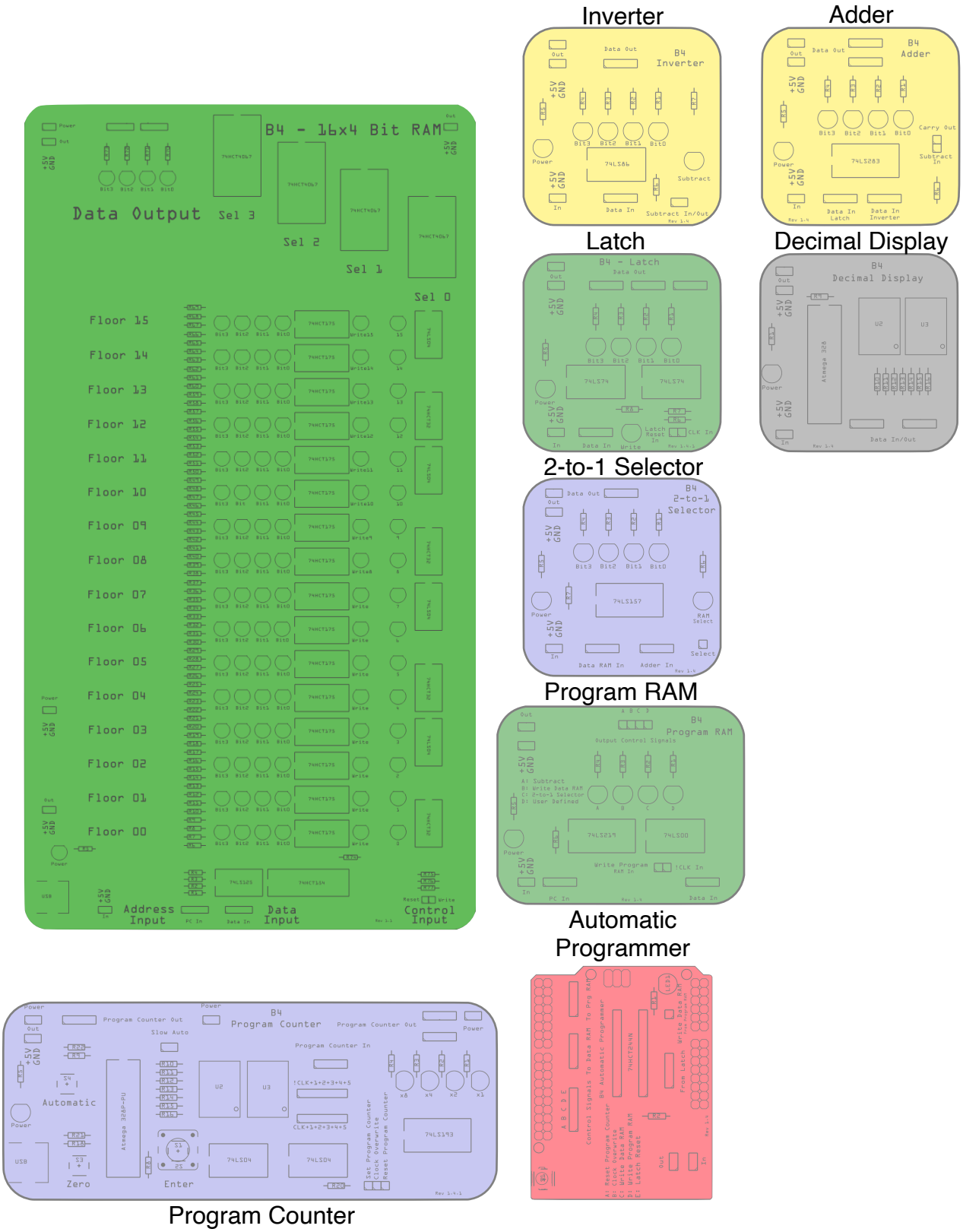
Lesson 6: Integrating the RAM Module into the B4 Computer Processor Kit

Modules Required: RAM, B4 Computer Processor Kit (not included)

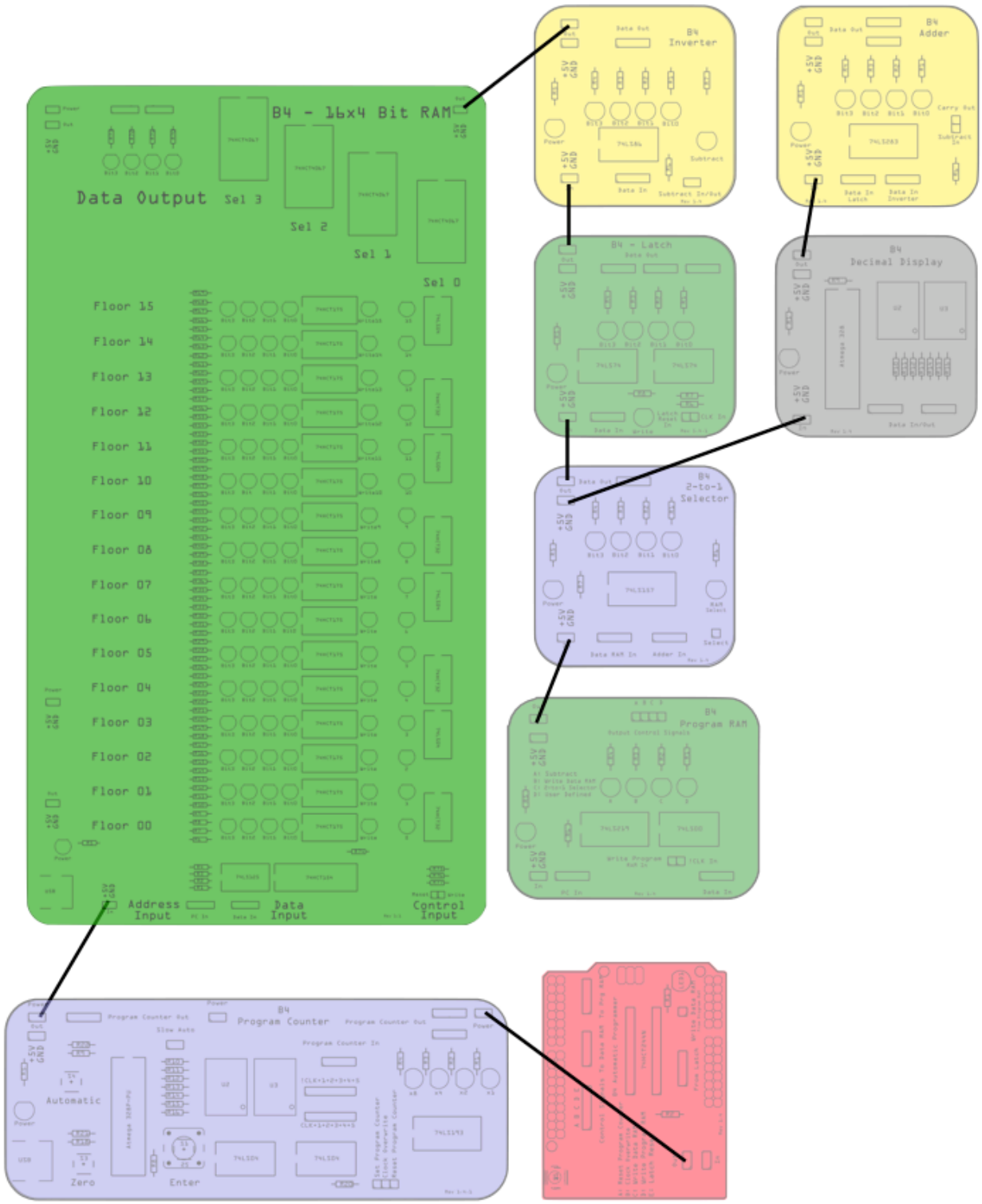
If you have a B4 Computer Processor Kit, then you can do all the experiments from that kit with the RAM module, instead of the Data RAM that ships with the B4 Computer Processor Kit. Since the RAM module is much larger than the Data RAM module in terms of its physical dimension, you might need to experiment a little bit until you have all modules arranged in the optimal way.

For example, for the most complex setup used in experiments 11 and 12 of the B4 Computer Processor Kit handbook, we have found that the following arrangement works quite well. The boards are arranged differently as compared to the step-by-step assembly instructions from experiment 11 in the B4 Computer Processor Kit handbook, but the function is very much the same.

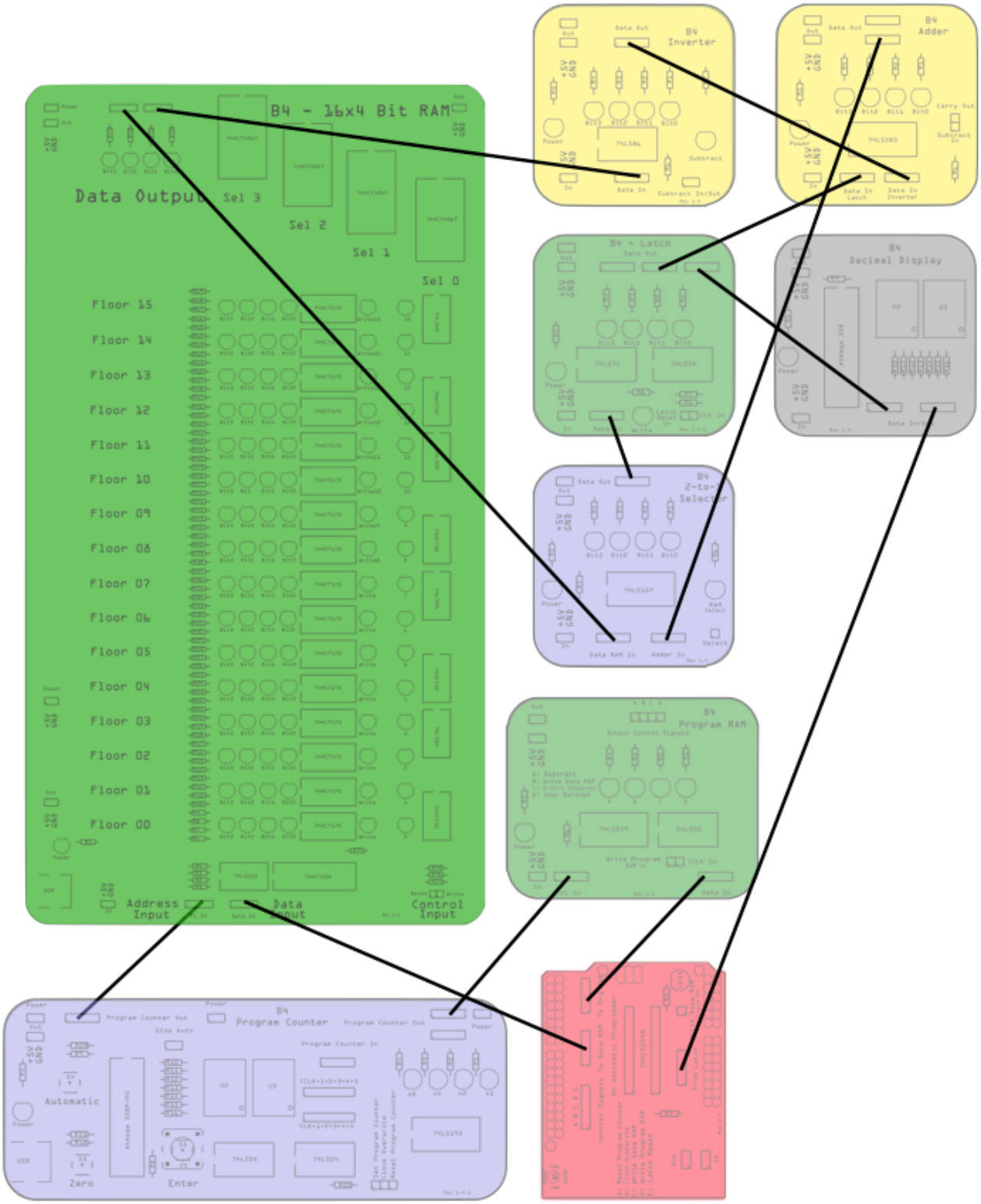
We'll do this step by step on the following pages, starting with the arrangement of the modules



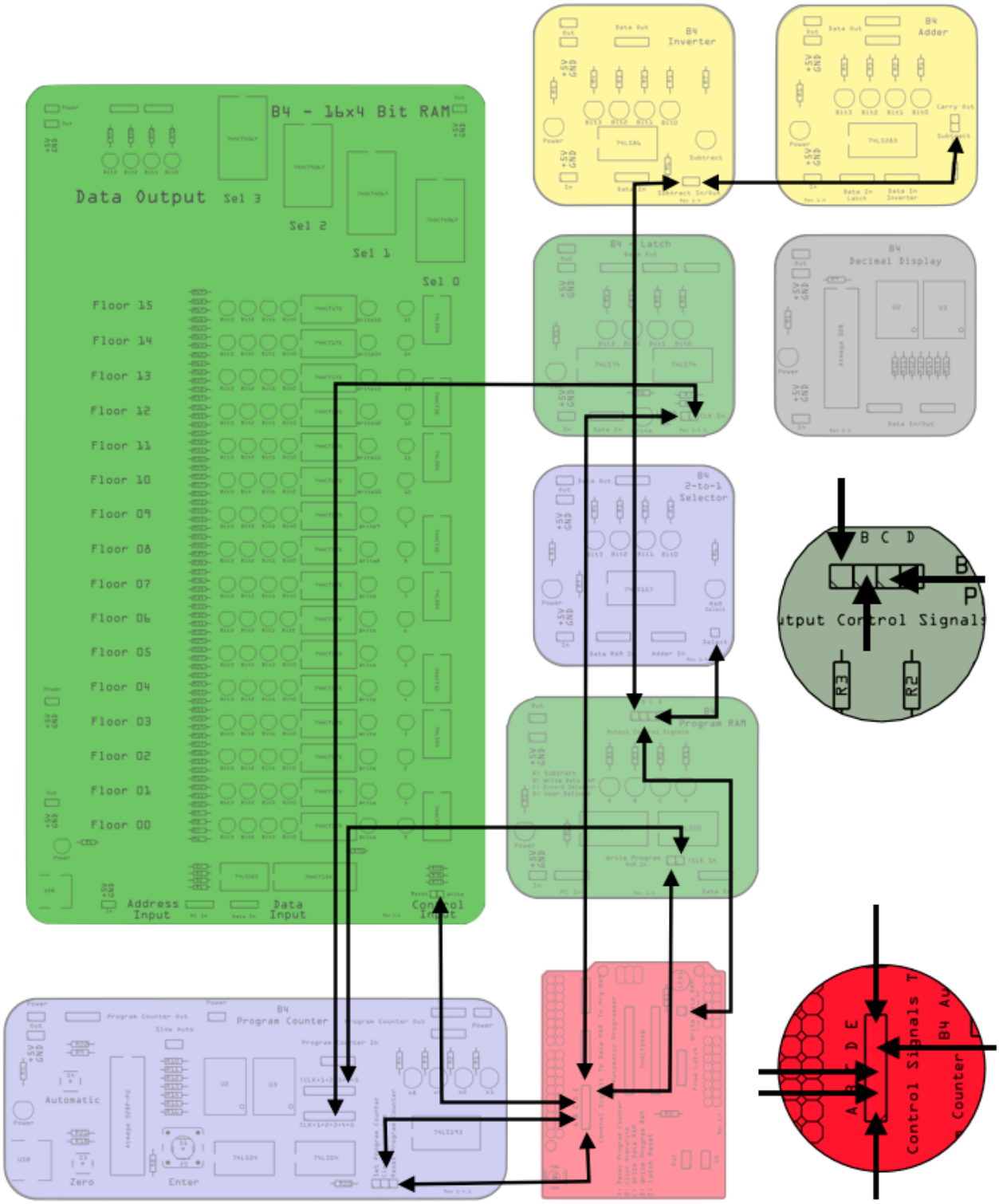
Setup of Experiment 6: Module arrangement



Setup of Experiment 6:Power Wiring only



Setup of Experiment 6: Data Wiring only



Setup of Experiment 6:1-Pin Control Wiring only

Congratulations, we are done: Make sure you connect the USB cable to the Automatic Programmer. To program the B4, just follow the instructions for the experiments 11 and 12 in the B4 Computer Processor Kit handbook.

Summary

In this book, we have learned how the 2,500-year long history of logic has led to a method of Boolean algebra in which we can define logical functions we call gates. Intelligently arranged, these gates can be put to good use to add or store information. Gates themselves are made of transistors, also intelligently arranged to perform the desired function of the gates, such as AND, XOR, NOR, etc. Computer chips can consist of billions of transistors. The design of a computer chip is, therefore, a high-tech task that requires many scientists and engineers.

Question 5.1	
?	Compare your knowledge about transistors that form gates to what you know about biological systems. Can you identify similarities?
	If transistors were made of mechanical parts that moved, rather than semiconductor materials, what disadvantages would this bring?
	How much does it cost to manufacture a microprocessor? What would be the price per transistor for this microprocessor?

Appendix A: Further reading

Below we have listed some really good resources that we used during the design of the B4. We very much recommend reading them.

Charles Petzold, CODE The Hidden Language of Computer Hardware and Software, 1999
http://www.charlespetzold.com/code/
Logic Gate: https://en.wikipedia.org/wiki/Logic_gate
Digital Logic Gates: http://www.electronics-tutorials.ws/logic/logic_1.html
Flip Flops: https://en.wikipedia.org/wiki/Flip-flop_(electronics)
History of Logic: https://en.wikipedia.org/wiki/History_of_logic
Transistor: https://en.wikipedia.org/wiki/Transistor

Appendix B: Troubleshooting

Every good experiment has the potential for failure. This is usually the moment when we learn something new. Below is a list of the typical errors and their solutions.

Symptom	Solution
Green light of a module is off	Check if power cable is connected
	Check if the wires at the power cable plugs are fully inserted. Change cable.
Unexpected behaviour. Odd output of the modules. Looks erratic.	Check if all wires are properly connected. Tick them off one by one on the schematic of the corresponding experiment.
	Check if the wires at the data cable sockets are fully inserted. Change cable.
	Have you inserted the correct module? Check!
All lights are off	Connect USB cable to a computer, USB power outlet or USB battery.
	There may be a short circuit, usually cause by a power cable. Disconnect all power cables from the RAM module and check if the RAM module's green LED comes on. If yes, carefully connect one module after the other.

Still got problems? Email us at: enquiries@digital-technologies.institute.

Appendix C: Solutions

Here are the solutions to the tasks from the different chapters in this book.

Question 2.1		Solution
?	Why is our design choice with the decoder and selector a good choice?	Yes, as it is a scalable solution. If we want to double the storage capacity of the RAM module, we only need to add one additional address wire.
	Can you think of a different way of designing the RAM module?	
	Can you think of other design choices that had consequences for a product?	A sports car is designed to go fast on a smooth racing track. It will do poorly off-road.

Question 5.1		
?	Compare your knowledge about transistors that form gates to what you know about biological systems. Can you identify similarities?	Transistors form gates, which form higher-level functions, such as arithmetics, memory, switching, etc. Similarly, cells form organs, which in turn form organisms.
	If transistors were made of mechanical parts that moved, rather than semiconductor materials, what disadvantages would this bring?	Mechanical parts are larger, consume more electricity and wear more quickly than semiconductors. Modern processors consist of billions of transistors. Let's assume we had a 1 billion transistor chip and we wanted to build it with relays, which are electromechanical switches. If each transistor were to be replaced with one relay, then we would require 1 billion relays. Let's further assume that 1 relay would require 1cm^3 (the size of a sugar cube) of space and that we need another 1cm^3 of space around each relay for wiring, etc.. So 2cm^3 of space per relay. That would be 2 billion cm^3. for all our 1 billion relays. That's 2,000,000,000 cm^3 = 2,000 m^3, or the equivalent of a cube with a side length of 12.6m, equivalent to a 4 storey building. If each relay required 50mA of current at 5V, then we'd need 50mA*1,000,000,000=50,000,000A. 50,000,000A*5V=250,000,000W, which is 250 Mega Watt. A smaller coal fired power plant produces 500 megawatt of electricity and burns 1.4 million tons of coal each year. We'd need half of this. In summary: If we could build such a relays computer, it would be the size of a 3 storey house, require half a coal-fired power plant and consume 700,000 tons of coal each year. This would be a tad too big for our pants. Not to mention the heat that the 700,000 tons of coal generate.

	How much does it cost to manufacture a microprocessor? What would be the price per transistor for this microprocessor?	Let's pick the XBox One processor which has 5 billion transistors. The XBox console'r retail price is about $350. Let's assume that the cost of the processor is maybe $50. So, the price per transistor is $50/5billion=$0.000 000 01 or 0.000001 cents, thats a thousands of a thousands of a cent per transistor. Let's put this into perspective: The print edition of the New York times newspaper has about 140,000 words. The average length of an English word is 5 letters. We conclude that the New York times contains 5*140,000=700,000 letters. If it costs $2 to make one copy of the New York times , then the cost per letter is $2/700,000=$0.000 003 or 0.000 3 cents. 0.0003 divided by 0.000001 is 300. So, making a transistor in a chip is about 300 times cheaper than printing a letter in a newspaper. What if we estimated the price of the processor wrong?If it is less than $50, then the ratio is greater than 300:1. If it is more than $50, let's say $100, then the ratio is 150:1.