UNIT 5 · SESSION 5

Binary, Decimal, Hexadecimal

A wire is either carrying voltage or it is not. Everything you have ever seen on a screen is built out of that one fact.

2ND SEMESTER · SCHOOL YEAR 2026-2027

Introduction to Computing · Prof. Ryan V. Labrador

Saturday, 12 September 2026 · 2-hour lecture

TODAY

Two hours, five parts

Part 1	**Why binary exists**	The physical reason, not the mathematical one
Part 2	**Place value**	The rule you already use in decimal, applied to any base
Part 3	**Converting**	Every direction, worked slowly, with numbers you can check
Part 4	**Hex in the real world**	Colours, memory, MAC addresses, IPv6
Part 5	**Binary addition**	Four rules, one carry, and what overflow means
After	**Laboratory**	You build a converter web page and publish it

LEARNING OBJECTIVES

By the end of today you will be able to...

1. Explain why computers use binary, in terms of the physical hardware.
2. Read place value in any base — decimal, binary, octal or hexadecimal.
3. Convert a binary number to decimal, and a decimal number to binary.
4. Convert between binary and hexadecimal using the four-bit grouping rule.
5. Say where hexadecimal is used in real systems, and why it is used there.
6. Add two binary numbers by hand, and explain what overflow is.

PART 1

Why binary exists

Not because it is elegant. Because it is the only thing that survives a noisy wire.

THE PHYSICAL REASON

A wire is on, or it is off



Two states

Voltage present, or voltage absent. A transistor is open or closed. There is no third condition that survives being knocked about inside a warm machine.



Reliable

To store a decimal digit directly you would need ten voltage levels that never blur into each other. Heat, cable length and a cheap power supply blur them.



Cheap

Two states means a simple switch. Billions of simple switches are affordable. Billions of precise ten-level meters are not, and never have been.

Binary is not a design choice made for beauty. It is what still reads correctly after the signal has been mistreated.

Ten symbols, or two

Decimal has ten symbols and runs out after 9. Binary has two and runs out after 1 — so it runs out constantly, and adds a new column constantly.

DECIMAL	BINARY	DECIMAL	BINARY	DECIMAL	BINARY
0	0	4	100	8	1000
1	1	5	101	9	1001
2	10	6	110	10	1010
3	11	7	111	11	1011

A new binary column opens at 2, 4, 8, 16, 32. Those numbers will follow you for the rest of your career.

PART 2

Place value

You have used this rule since primary school. Today you find out it was never about ten.

What 4072 actually means

PLACE	10^3	10^2	10^1	10^0
Worth	1000	100	10	1
Digit	4	0	7	2
Contributes	4000	0	70	2

4000 + 0 + 70 + 2 = 4072. Each place is worth ten times the place to its right, because the base is ten.

The same rule, doubling instead of tenfold

	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Worth	128	64	32	16	8	4	2	1
Bits	**1**	**1**	**0**	**1**	**0**	**1**	**1**	**0**
Adds	128	64	0	16	0	4	2	0

11010110 = 128 + 64 + 16 + 4 + 2 = 214. Write the worths above the bits and the conversion does itself.

Where the number 11 comes from



Copy this shape into your notebook. Every binary-to-decimal question for the rest of the semester is this picture.

Three steps, and a sanity check

Step 1	**Write the worths**	Start at 1 on the right and double leftwards, one number above each bit.
Step 2	**Ignore every zero**	A 0 contributes nothing at all. Only the columns holding a 1 matter.
Step 3	**Add the survivors**	Add the worths that sit above a 1. That total is your answer.
Check	**Is it even possible?**	Eight bits can never exceed 255. If you got 900, you have added something twice.

Practise now: 1011 → 8 + 0 + 2 + 1 = 11. 100000 → 32. 11111111 → 255.

Method one: divide by two, keep the remainders

Divide by 2 over and over. Write down each remainder. When you reach zero, read the remainders from the BOTTOM upwards.

Convert 45 to binary — divide by 2 until you reach 0

DIVIDE	ANSWER	REMAINDER
45 ÷ 2	22	1
22 ÷ 2	11	0
11 ÷ 2	5	1
5 ÷ 2	2	1
2 ÷ 2	1	0
1 ÷ 2	0	1

READ THE REMAINDERS BOTTOM TO TOP

101101

45 in binary

CHECK IT BACK

32 + 8 + 4 + 1 = 45

Six divisions, six remainders, one answer. The last remainder you write is the first bit you read.

Read the remainders bottom to top: 101101. Check it: 32 + 8 + 4 + 1 = 45. Correct.

Method two: take the biggest place that fits

Converting 202, left to right. Does this worth fit in what is left? Yes → write 1 and subtract. No → write 0.

WORTH	128	64	32	16	8	4	2	1
Fits?	Yes	Yes	No	No	Yes	No	Yes	No
Bit	**1**	**1**	**0**	**0**	**1**	**0**	**1**	**0**
Left over	74	10	10	10	2	2	0	0

202 = 11001010. Two methods, one answer. Use whichever one you trust when you are nervous.

PART 3

Octal and hexadecimal

Binary is correct and unreadable. These two exist so that humans can say it out loud.

BASE 8

Octal — three bits at a time

Digits	**0 to 7 only**	There is no digit for eight, exactly as decimal has no single digit for ten.
Worths	**1, 8, 64, 512**	Each place is worth eight times the place to its right.
The trick	**Three bits per digit**	Split the binary into threes from the right. Each group becomes one octal digit.
In the wild	**File permissions**	chmod 755 on a Linux server is octal — and your class server runs on exactly that.

202 = 11001010 → group from the right → 011 001 010 → 3 1 2. So 202 decimal is 312 in octal.

BASE 16

Sixteen digits — so six of them are letters

DEC	HEX	BINARY	DEC	HEX	BINARY
0	0	0000	8	8	1000
1	1	0001	9	9	1001
2	2	0010	10	A	1010
3	3	0011	11	B	1011
4	4	0100	12	C	1100
5	5	0101	13	D	1101
6	6	0110	14	E	1110
7	7	0111	15	F	1111

Binary and hex are the same thing

Binary → hex

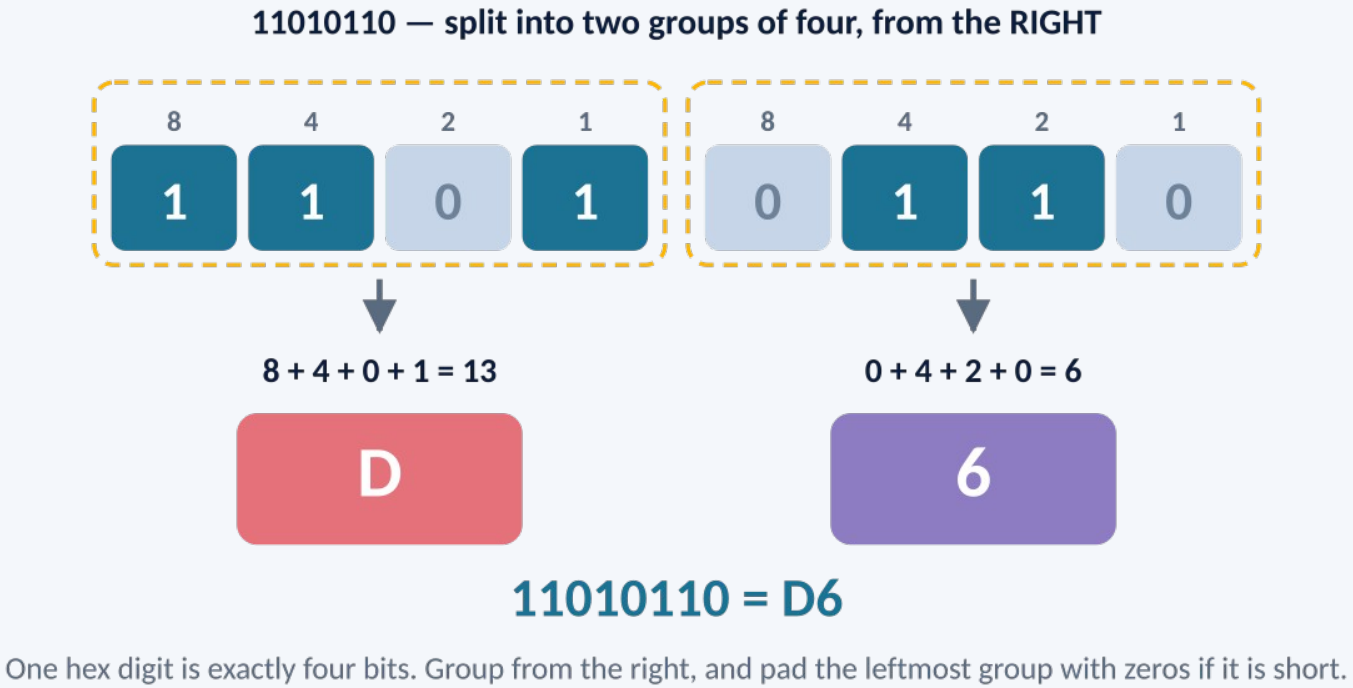
- Split into groups of four, starting from the RIGHT.
- 11010110 → 1101 | 0110
- 1101 is 13, which is D. 0110 is 6.
- Answer: D6
- If the left group is short, pad it with zeros.

Hex → binary

- Expand each hex digit into exactly four bits.
- 2F → 2 is 0010, F is 1111
- Answer: 00101111
- Never drop the leading zeros inside a group.
- Check: 32 + 8 + 4 + 2 + 1 = 47

One hex digit = exactly four bits. That is the entire reason hexadecimal exists.

The same eight bits, read as two hex digits



One hex digit is exactly four bits. Group from the right, and pad the leftmost group with zeros if it is short.

The dashed boxes are the only thing you have to get right. Draw them from the right-hand end, every single time.

Hex → decimal, and decimal → hex

DIRECTION	WORKING	ANSWER
Hex 2F to decimal	(2 × 16) + 15	**47**
Hex A3 to decimal	(10 × 16) + 3	**163**
Hex FF to decimal	(15 × 16) + 15	**255**
Decimal 214 to hex	214 ÷ 16 = 13 remainder 6	**D6**
Decimal 47 to hex	47 ÷ 16 = 2 remainder 15	**2F**

Two hex digits cover 0 to 255 exactly — one byte. That is next Saturday's entire topic.

Where you will actually meet hexadecimal

Colours	Web and design	#1C7293 is three bytes: red, green, blue, two hex digits each.
Memory	Addresses	0x7FFF is shorter than 32767 and lines up cleanly with the underlying bits.
Networking	MAC addresses	Six bytes, twelve hex digits — 00:1A:2B:3C:4D:5E on every device you own.
Networking	IPv6	Eight groups of four hex digits, because the IPv4 addresses ran out.
Files	Hex dumps	Open any file in a hex viewer and you are reading its bytes, two digits at a time.

Hex is not another kind of number. It is binary read four bits at a time, so a human can say it aloud.

A colour is three bytes

Every colour on these slides is three bytes written in hexadecimal



Each channel is one byte — 00 to FF, or 0 to 255. Three bytes give 256 × 256 × 256 = 16,777,216 colours.

Each channel is one byte: 0 to 255, or 00 to FF. Six hex digits, three bytes, 16,777,216 possible colours.

PART 5

Binary addition

Four rules. You already know three of them without being told.

THE RULES

There are only four

0 + 0

= 0

0 + 1

= 1

1 + 0

= 1

1 + 1

= 10

write 0, carry 1

Third case	**1 + 1 + 1 = 11**	Once a carry arrives you are adding three bits. Write 1, carry 1.
Not new	**9 + 1 = 10 in decimal**	You have carried since primary school. Binary just carries far more often.

WORKED EXAMPLE

1011 + 0110, column by column

Work right to left. Add the two bits and any carry that arrived, write the low bit, push the carry one column left.

COLUMN	16	8	4	2	1
Carried in	1	1	1	0	0
1011 (11)		1	0	1	1
0110 (6)		0	1	1	0
Answer	1	0	0	0	1

1011 + 0110 = 10001. In decimal: 11 + 6 = 17, and 10001 = 16 + 1 = 17. Always check in decimal.

WHEN IT RUNS OUT

Overflow

The limit	**A byte holds 0 to 255**	11111111 is the largest value eight bits can carry. There is no 256th pattern.
Add one	**11111111 + 1**	The correct answer is 100000000 — nine bits wide. The byte has only eight.
What happens	**It wraps to zero**	The ninth bit falls off the end. What remains in storage is 00000000.
Why you care	**Real systems break**	Counters that reset, scores that wrap, timers that jump backwards. All of them are this.

Choose the size before you store the number. Getting that decision wrong is a genuine, expensive, career-length problem.

Six conversions, one card

Bin → Dec	**Add the worths**	Write 1 2 4 8 16 above the bits from the right. Add where there is a 1.
Dec → Bin	**Divide by 2**	Keep the remainders. Read them bottom to top.
Bin → Hex	**Group four from the right**	Each group becomes one hex digit. Pad the left group with zeros.
Hex → Bin	**Each digit becomes four bits**	Keep the leading zeros inside every group.
Dec → Hex	**Divide by 16, or go via binary**	Remainders 10 to 15 are written A to F.
Hex → Dec	**Multiply by 1, 16, 256**	2F = (2 × 16) + 15 = 47.

MARKS LOST

Six mistakes, all avoidable

Reading backwards

The worths grow leftwards. The rightmost bit is 1, not the biggest.

Dropping leading zeros

Inside a group of four, 0010 is not 10. Losing zeros loses the value.

Grouping from the left

Always group from the right. 11010110 is 1101 0110, never 110 1011 0.

Writing 13 instead of D

In hex, remainders above nine are letters. 13 and D are different answers.

Not saying the base

10 in binary is two. 10 in hex is sixteen. State the base, every time.

Holding the carry in your head

Binary carries constantly. Write the carry row down. Every time.

Every one of these appears on a prelim paper somewhere. Now you have seen all six in advance.

HOLD ON TO THIS

Four things from today

One	Two states, because physics	Binary exists because a wire can reliably be on or off, and nothing else.
Two	Place value is universal	Same rule in every base. Only the multiplier changes.
Three	One hex digit is four bits	That single fact is why hexadecimal is used everywhere it is used.
Four	Always check in decimal	Convert your answer back. It costs ten seconds and catches almost everything.

Next Saturday is the Prelim. Units 1 to 5 — and this unit is the one you can practise most easily.

BEFORE THE PRELIM

Three numbers you have already met

503	**Nothing is running**	Your program will not start. Not one page works. Read log.txt, fix it, upload — it comes back on its own.
404	**Running, no such address**	Your program IS alive. It just has no route with that name. Check the spelling in app.route.
500	**It crashed while answering**	The route was found and then something in it broke. There is a real traceback waiting in log.txt.

All three are examined next Saturday. Be able to say which one proves your program is alive.

AFTER THE BREAK

In the laboratory today

By hand	**Eight conversions**	Your group works out eight answers on paper first. Those become your truth set.
Build	**/convert**	Sixteen rows of decimal, binary, octal and hex — produced by ONE while loop, not typed.
Test	**Machine against paper**	Your page has to agree with your paper. If it does not, one of them is wrong.
Yours	**Your own number**	The day you were born, or the end of your student number, converted by your own loop.
AI task	**Make it fail**	Ask an AI for a converter. It will use something you have not been taught. Rewrite it in what you know.

Do the paper first. A group that types the code before working out the answers has nothing to test against.

PROGRAMMING LAB ACTIVITY

This afternoon

Sixteen rows that nobody typed

What you are building today

Together	**The group page**	A /convert page: decimal, binary, octal and hexadecimal from 0 to 15, produced by one while loop rather than sixteen typed rows.
On your own	**Your own version — this is the marked one**	Your own number — the day you were born, or the end of your student number — converted by your own loop, with the division steps shown.
To match	**Your page.html twin**	The same thing again, typed out by hand as HTML.

WHERE IT GOES LIVE

`gNN.cic.rmlsystems.net/convert`	`gNN.cic.rmlsystems.net/<yourfolder>/`	`gNN.cic.rmlsystems.net/<yourfolder>/page.html`

Eight identical pages is eight zeros. Your version has to be yours — your data, your words.

NEW THIS WEEK

The Python you have not used before

`13 // 2`

whole-number division — 6, the part that carries on

`13 % 2`

the remainder — 1, the bit you write down

`while number > 0:`

keep going until the number runs out

`digits = str(remainder) + digits`

build the answer up in front, as you go

at the shell

```
number = 13
digits = ""

while number > 0:
    remainder = number % 2
    digits = str(remainder) + digits
    number = number // 2

print(digits)
```

It prints: 1101

TWENTY POINTS

How this is marked

6	**It runs**	Your address loads and there is no traceback on it.
6	**It does this week's thing**	The table comes out of a while loop. Nobody typed sixteen rows.
4	**It is yours**	Your own number, with your own division steps visible on the page.
2	**The twin matches**	page.html shows the same thing, by hand.
2	**Worksheet complete**	Including the AI sentence and the twin count.

DUE Sunday 13 September, 11:59pm · the full guide is at labs.cic.rmlsystems.net/guides/

Two symbols. That is all it has.

Every photograph, every song, every peso balance, every message you have ever sent is a very long row of ons and offs. This afternoon you write the code that reads them.

Questions?