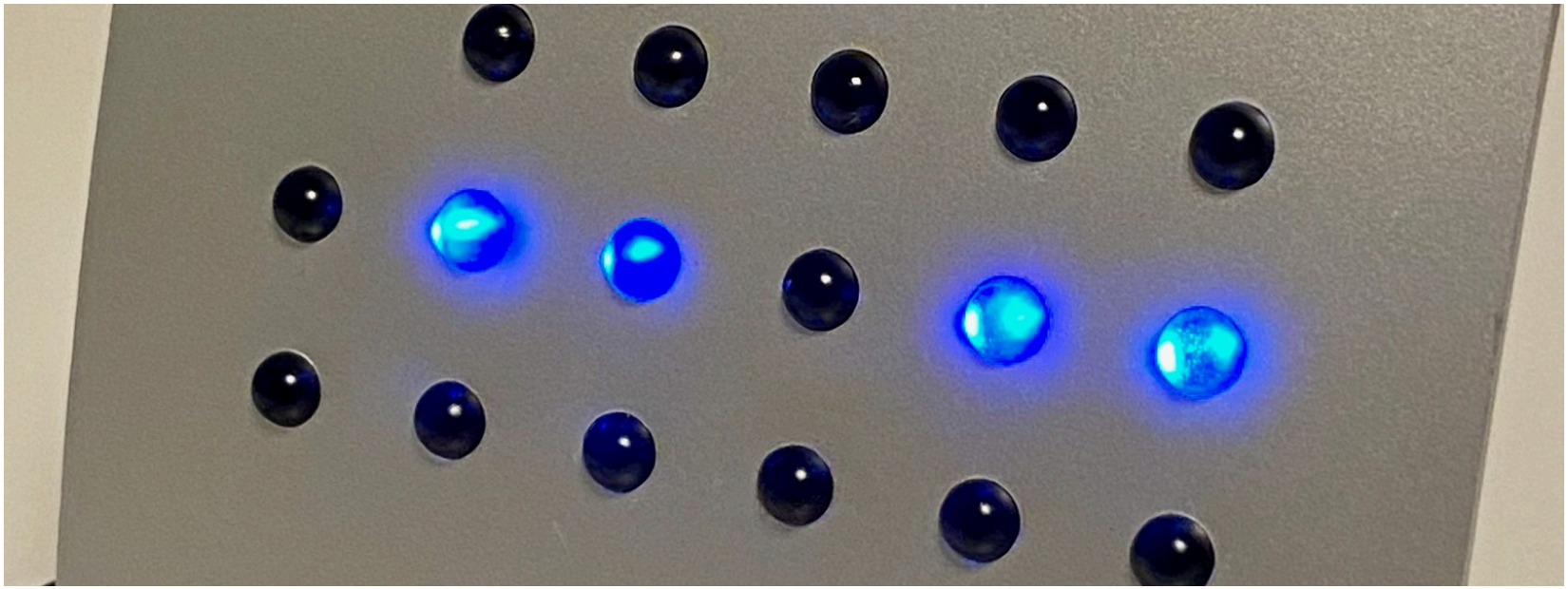
HSBI

Hochschule Bielefeld

University of Applied Sciences and Arts



Codes, code converters and standard components

DET-05 · Digital Electronics

GUIDING QUESTIONS

- Barcode, QR code, Morse code: why are there so many codes, and how do you choose the right one?
- Single light emitting diodes (LEDs) on a binary-coded decimal (BCD) clock, glowing segments on a laboratory power supply: how does a number turn into light?
- Two heat pumps, one shower: which circuit decides which of them heats the water?



Image by Qwen-Image 2.1

INTENDED LEARNING OUTCOMES

- You explain what the codes covered in this lesson are used for.
- You design encoders and code converters for the simpler codes and find faults that have been built into them.
- You implement logic functions with standard components.



Image by Qwen-Image 2.1

AGENDA

1. Codes and coding
2. Error-detecting and error-correcting codes
3. Decoders
4. Encoders and code converters
5. Multiplexers and demultiplexers
6. Logic functions with decoders and multiplexers
7. What have you learnt?



Image by Qwen-Image 2.1

REVIEW OF DET-04

- Prime implicants: the essential ones belong to every minimal disjunctive normal form (DNF)
- Quine–McCluskey algorithm: all prime implicants table by table, selected with the prime implicant chart
- Hazards: a redundant prime implicant removes a logic hazard, a function hazard remains
- A combinational logic circuit depends only on its present inputs; a sequential circuit also stores information

Today: how bits carry information as codes – and standard components that turn one code into another, select a signal or distribute it

HSBI

Hochschule Bielefeld

University of Applied Sciences and Arts

1 Codes and coding

Which code for which job: numbers, digits, positions and texts

- Code and code word
- The binary code: power values and ‘not delivered’
- BCD code and excess-3 code
- One bit at a time: the Gray code
- ASCII and Code 39

CODE AND CODE WORD

- Code: maps the characters of one set onto the characters of a second set, defined by a table and reversible

Examples, each made for its medium: Morse code for light or tone, a barcode for the scanner at a self-checkout, a QR code for the camera of your smartphone

- Code word: the bit pattern that a code assigns to a character – DET-03 and DET-04 called it ‘code’ for short
- With n bits, there are 2^n code words
  - Minimal code: uses all of them
  - Redundant code: uses fewer – additional bits can make errors visible (chapter 2)

THE BINARY CODE: POWER VALUES AND ‘NOT DELIVERED’

- Binary code: a number as a binary number, weighted with the powers of two – digital systems calculate with it

Example: the electrical power of our heat pump – P: the power, N: the whole number that the controller transmits, in units of 0.1 kW

$$P = N \cdot 0.1\,\mathrm{kW}$$

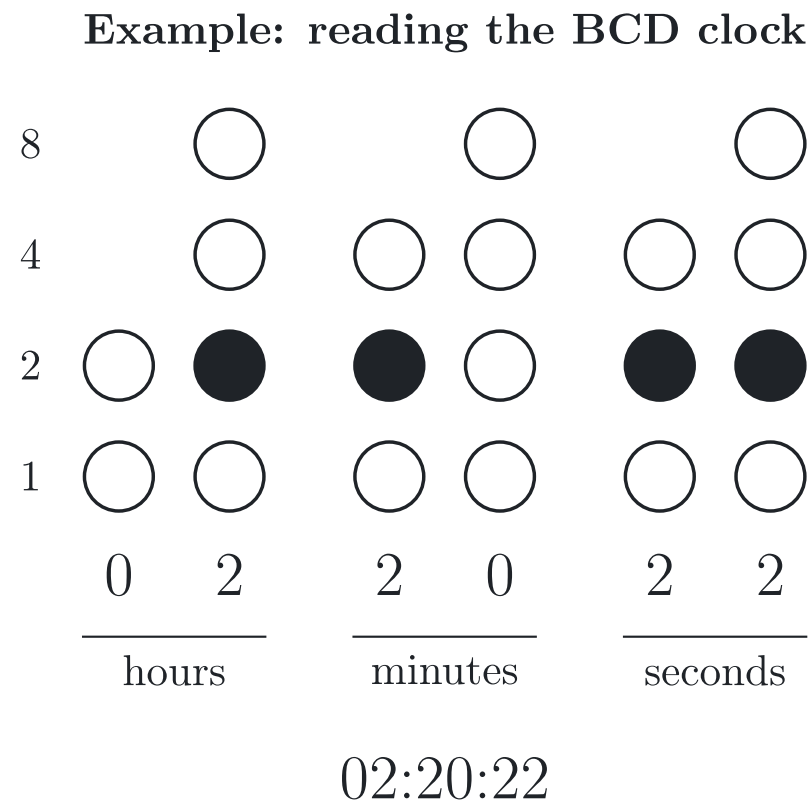
- 47 means 4.7 kW: the unit is part of the agreement between the controller and the reader
- The code word 8000_{16}, written 0x8000, means ‘value not delivered’, for example for the heating circuits 2 and 3
  - Read as a 16-bit number in two's complement: −32 768, far from any temperature or power

BCD CODE AND EXCESS-3 CODE

- BCD code: each decimal digit as a four-bit binary number with the weights 8, 4, 2 and 1

Example: the BCD clock – add the weights of the lit LEDs in each column

- Only ten of the 16 code words are used, 0000 to 1001: a redundant code
- More bits than the binary code: the message number 50 004 of the heat pump needs 16 bits in the binary code and 20 bits in BCD
- Excess-3 code: each digit plus 3; inverting all bits gives the nine's complement

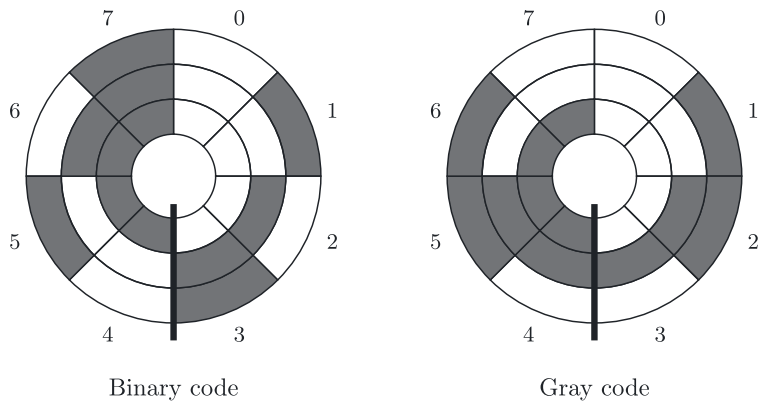


THE GRAY CODE

- Gray code: from one code word to the next, only one bit changes – a unit-distance code, here also cyclic from 7 back to 0
- In the binary code, three bits change from 3 to 4, and a reading in between can be wrong (DET-04, 4.4)

Decimal	Binary code	Gray code
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101
7	111	100

Example: code discs of a rotary encoder



Tracks from the inside out: weights 4, 2 and 1

- At the border between two positions, only one track of the Gray disc changes: a misread is off by one position at most
- Known from DET-03: the labels 00, 01, 11 and 10 of the Karnaugh map

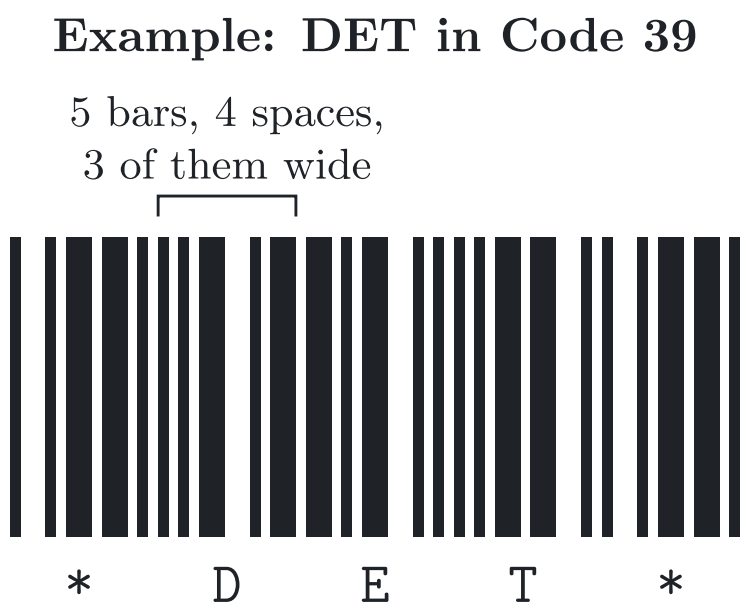
ASCII AND CODE 39

- American Standard Code for Information Interchange (ASCII): an alphanumeric code for letters, digits, punctuation marks and control characters – 128 characters with seven bits

Character	ASCII, hexadecimal	ASCII, binary
space	20	010 0000
0	30	011 0000
9	39	011 1001
A	41	100 0001
B	42	100 0010
a	61	110 0001

Example: a text on your smartphone, in UTF-8 with the code words of ASCII

- Code 39: a barcode for 43 characters



- Exactly three of the nine elements are wide: a single misread bar cannot produce another valid character

HSBI

Hochschule Bielefeld

University of Applied Sciences and Arts

2 Error-detecting and error-correcting codes

When a bit flips on the way: how additional bits reveal the error and even correct it

- Parity bit and Hamming distance
- The Hamming code

PARITY BIT AND HAMMING DISTANCE

- Redundancy, from Latin *redundans*, ‘overflowing’: more information than strictly needed – only then can an error be detected

Example: the digit 7 in BCD, 0111 – one flipped bit gives 0101, the valid digit 5, and nothing reveals the error

- Parity bit with even parity: completes the number of ones to an even number – 0111 becomes 0111 1
  - Detects one flipped bit, but not which one; two flipped bits go unnoticed
  - A chain of XOR gates computes it (DET-03, 4.5)
- Hamming distance: the number of positions in which two code words differ – 0111 and 0101 have the distance 1

Smallest distance between two code words	Detected with certainty	Corrected with certainty	Example
1	none	none	BCD code
2	1 flipped bit	none	BCD code with parity bit
3	2 flipped bits	1 flipped bit	the code of section 2.2 of this lesson

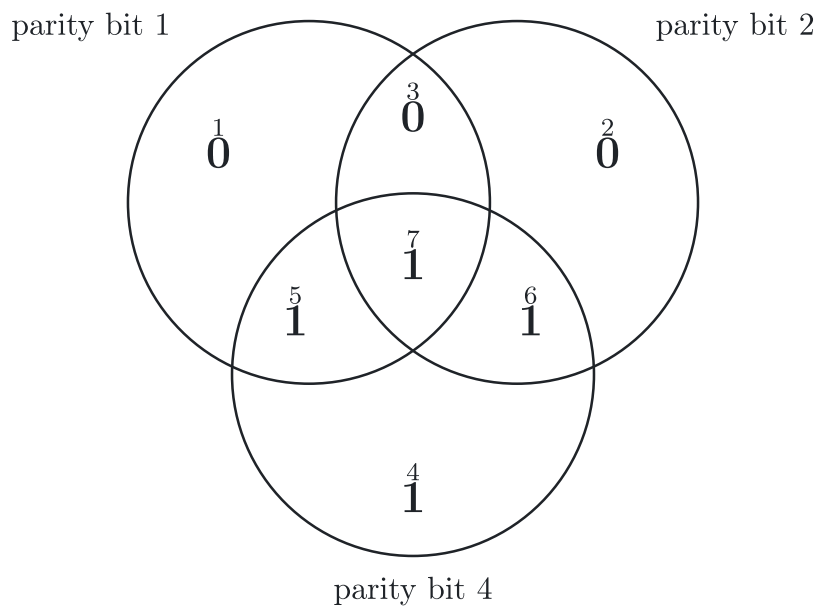
THE HAMMING CODE

- Hamming code with seven bits: parity bits at the positions 1, 2 and 4, data bits at 3, 5, 6 and 7
- Each parity bit completes its set of positions to an even number of ones

Example: the digit 7, data bits 0, 1, 1 and 1 – code word 0001111

- Bit 6 flips, 0001101: the sets of the parity bits 2 and 4 are odd, 2 + 4 = 6 – invert bit 6
- Any two code words differ in at least three positions: corrects one flipped bit or detects two
- Bluetooth protects certain packets with a shortened Hamming code, memory with an error-correcting code (ECC memory) with an extended one

Example: digit 7 in the Hamming code



small: position, large: its bit in the code word 0001111

3 Decoders

Standard components, designed with the six steps of the synthesis (DET-03, 1.1): one output for each code word

- The decoder and the one-hot code
- SG Ready with a lockout
- The 7-segment display and its code
- Minimizing the segments with don't cares
- The 7-segment display decoder as circuit and VHDL code

THE DECODER AND THE ONE-HOT CODE

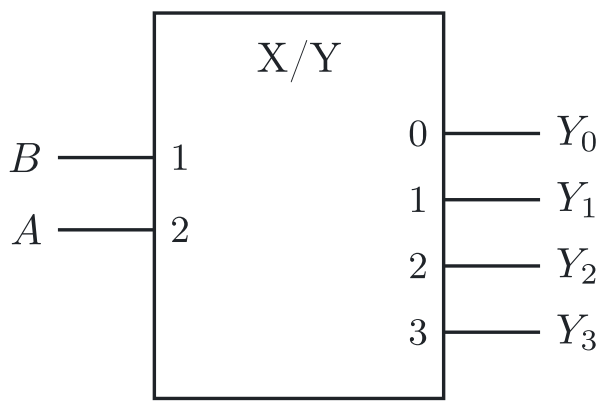
- Decoder: with n inputs and 2^n outputs, it sets exactly one output to 1 for each code word – the one-hot code

Example: a 2-to-4 decoder with the inputs A and B, A with the weight 2, and the outputs Y_0, Y_1, Y_2 and Y_3

A	B	Y_0	Y_1	Y_2	Y_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

- Each output is a single minterm, from $Y_0 = \overline{A} \cdot \overline{B}$ to $Y_3 = A \cdot B$

Schematic symbol: 2-to-4 decoder



- Symbol according to the standard of the International Electrotechnical Commission (IEC), IEC 60617: X/Y, the inputs with their weights, the outputs with the numbers of their code words

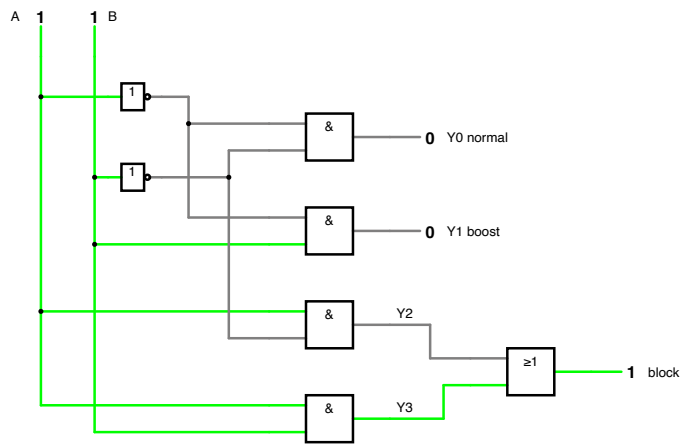
SG READY WITH A LOCKOUT

Example: Smart Grid Ready (SG Ready) – the two switching signals A and B of our heat pump

- The 2-to-4 decoder of DET-02, 6.6: Y_0 normal operation, Y_1 boost, Y_2 block, Y_3 start command
- In our system, the start command must never act: an OR gate leads Y_3 into the block

$$Y_{\text{block}} = Y_2 + Y_3 = A \cdot \overline{B} + A \cdot B = A$$

- Blocked whenever $A = 1$: the block has priority, and the combination 1:1 gets a defined, safe reaction instead of a don't care



Open in the simulator

- Smartphone: in the Bluetooth receiver, a decoder could point to the flipped bit of a code word in the Hamming code

THE 7-SEGMENT DISPLAY AND ITS CODE

Example: the display of a laboratory power supply, with the voltage and the current that you set

- 7-segment display: each digit from seven LEDs, the segments a to g
- 7-segment code: which segments form which digit

Example: the 7-segment code



	BCD clock	7-segment display
Light sources per digit	up to four LEDs with the weights 8, 4, 2 and 1	seven segments a to g
What lights up	the BCD code word	the shape of the digit
Reading	add the weights of the lit LEDs	read the digit at a glance
Code converter	none	7-segment display decoder
Example	the clock in section 1.3	the laboratory power supply

- The 7-segment display shows the shape of the digit and needs a code converter: the 7-segment display decoder

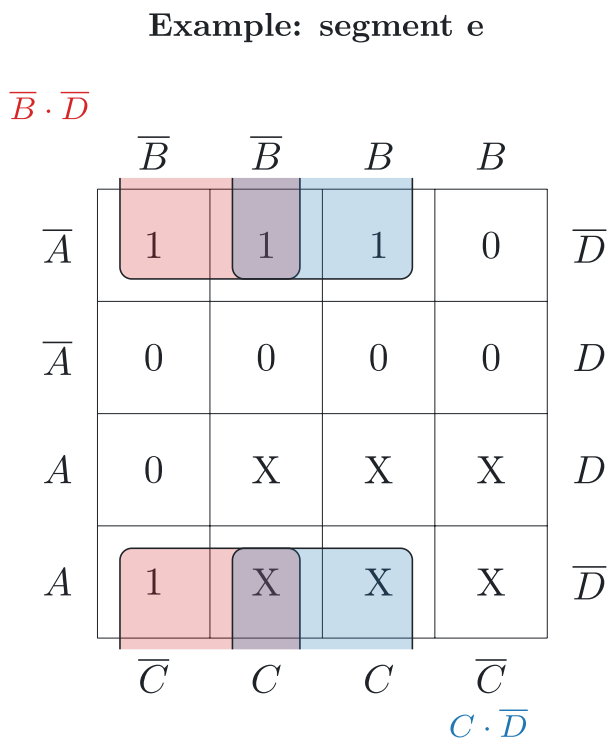
MINIMIZING THE SEGMENTS WITH DON'T CARES

- Steps 1 and 2: the bits A, B, C and D of the BCD code word as inputs, A with the weight 8, and the outputs Y_a to Y_g, one for each segment
  - The code words 1010 to 1111 never occur: don't cares, X
- Steps 3 and 4: one Karnaugh map for each segment; an X counts as a one where it enlarges a loop

Example: segment e, lit for the digits 0, 2, 6 and 8

$$Y_e = \overline{B} \cdot \overline{D} + C \cdot \overline{D}$$

- Without the don't cares: two product terms with three literals each



THE 7-SEGMENT DISPLAY DECODER: CIRCUIT AND VHDL CODE

Step 5: the seven minimal DNFs, with shared AND gates and one OR gate for each segment

Segment	Minimal DNF
a	$Y_a = A + C + B \cdot D + \overline{B} \cdot \overline{D}$
b	$Y_b = \overline{B} + \overline{C} \cdot \overline{D} + C \cdot D$
c	$Y_c = B + \overline{C} + D$
d	$Y_d = \overline{B} \cdot \overline{D} + \overline{B} \cdot C + C \cdot \overline{D} + B \cdot \overline{C} \cdot D$
e	$Y_e = \overline{B} \cdot \overline{D} + C \cdot \overline{D}$
f	$Y_f = A + \overline{C} \cdot \overline{D} + B \cdot \overline{C} + B \cdot \overline{D}$
g	$Y_g = A + \overline{B} \cdot C + B \cdot \overline{C} + C \cdot \overline{D}$

Open in the simulator

- Step 6: switch through 0000 to 1001 and compare each digit with the truth table
- In VHDL code, a selected signal assignment gives each code word its segment pattern, and the synthesis tool minimizes

4 Encoders and code converters

The six steps of the synthesis (DET-03, 1.1) for encoders and code converters – and step 6 for finding a built-in fault

- The encoder
- From binary code to Gray code and back
- The parity bits of the Hamming code as a circuit
- Finding a built-in fault

THE ENCODER

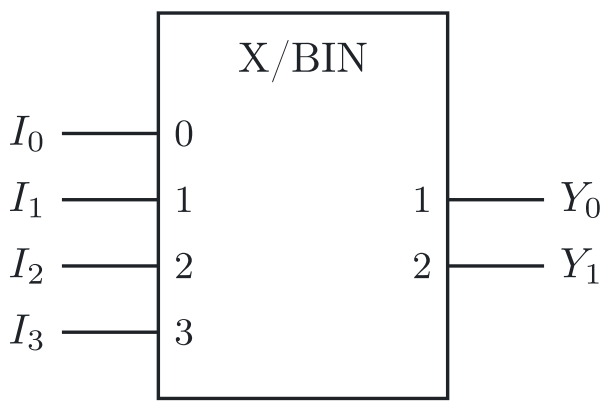
- Encoder: a 1 at one input produces its code word at the outputs – from the one-hot code back into the binary code

Example: a 4-to-2 encoder with the inputs I_0, I_1, I_2 and I_3 and the outputs Y_1 and Y_0

I_3	I_2	I_1	I_0	Y_1	Y_0
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

$$Y_1 = I_2 + I_3 \qquad Y_0 = I_1 + I_3$$

Schematic symbol: 4-to-2 encoder



- Two inputs at once give the OR of both code words; a priority encoder encodes only the active input with the highest priority
- Keypad with ten keys: from the decimal code into BCD with four OR gates

Open in the simulator

FROM BINARY TO GRAY AND BACK

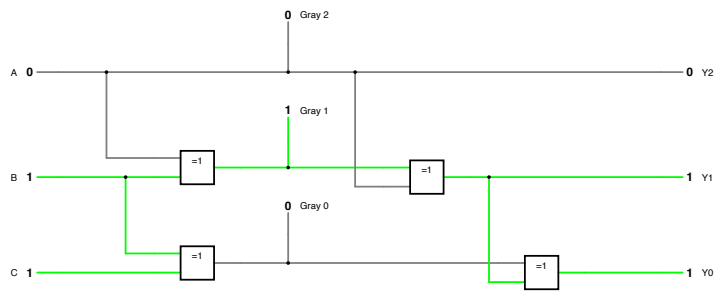
Example: the rotary encoder of a robot arm delivers the Gray code, the controller calculates in the binary code

- Binary to Gray, from the inputs A, B and C to the outputs Y_2, Y_1 and Y_0: each Gray bit is the binary bit, inverted if the next higher binary bit is 1

$$Y_2 = A \qquad Y_1 = A \oplus B \qquad Y_0 = B \oplus C$$

- Back, now with the Gray bits as inputs: a chain from the highest bit down

$$Y_2 = A \qquad Y_1 = A \oplus B \qquad Y_0 = Y_1 \oplus C$$



Open in the simulator

- Step 6: both code converters one behind the other – the outputs equal the inputs

THE PARITY BITS OF THE HAMMING CODE AS A CIRCUIT

Example: the parity bits Y_1, Y_2 and Y_4 for the digit 7, from the data bits A, B, C and D at the positions 3, 5, 6 and 7

$$Y_1 = A \oplus B \oplus D \qquad Y_2 = A \oplus C \oplus D \qquad Y_4 = B \oplus C \oplus D$$

- The map of each parity bit is a chequerboard: for each parity bit two XOR gates with two inputs, six for the whole parity generator
- Receiver: the XOR over each set of four positions; read with the weights 1, 2 and 4, the three results give the number of the flipped bit
- A 3-to-8 decoder could point to this bit, and an XOR gate at each position could invert it
- Simulation: the digit 7 with bit 6 flipped – the results 1, 1 and 0 for the weights 4, 2 and 1 give 6

Open in the simulator

FINDING A BUILT-IN FAULT

- Step 6: compare each output with the code table, row by row – the deviations point to the fault

Example: a code converter from binary to Gray with a built-in fault, with the inputs A, B and C and the output Y_0

Row	A	B	C	Y_0 from the code table, $B \oplus C$	Y_0 in the simulator	Deviation
0	0	0	0	0	0	–
1	0	0	1	1	1	–
2	0	1	0	1	0	yes
3	0	1	1	0	1	yes
4	1	0	0	0	1	yes
5	1	0	1	1	0	yes
6	1	1	0	1	1	–
7	1	1	1	0	0	–

- Only Y_0 deviates, in the rows in which A and B differ: its XOR gate receives A instead of B

Open in the simulator

- Some faults hide: if segment g fails, the 8 turns into a 0 – a lamp test lights all segments

5 Multiplexers and demultiplexers

The six steps of the synthesis (DET-03, 1.1) for selecting and distributing – and one chip that does both

- The multiplexer
- The demultiplexer
- One chip, both directions: the HEF4051 in the BCD clock

THE MULTIPLEXER

- Multiplexer: passes one of its 2^n data inputs to its single output, chosen by its n select inputs – a digital switch

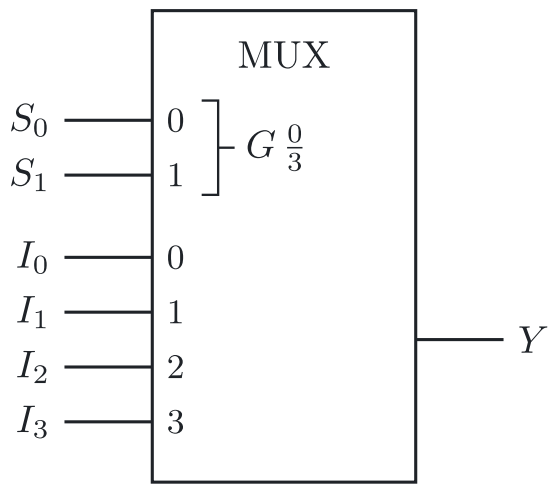
Example: a 2-to-1 multiplexer with the select input S, the data inputs I_0 and I_1 and the output Y

S	Y
0	I_0
1	I_1

$$Y = \overline{S} \cdot I_0 + S \cdot I_1$$

- Heat pump: in boost, a 2-to-1 multiplexer for each bit could select the upper hot-water set point

Schematic symbol: 4-to-1 multiplexer



- Symbol according to IEC 60617: MUX; G 0/3 – the number formed by the select inputs selects the data input

Open in the simulator

THE DEMULTIPLEXER

- Demultiplexer: routes its single input to one of its outputs, chosen by its select inputs; the others are 0

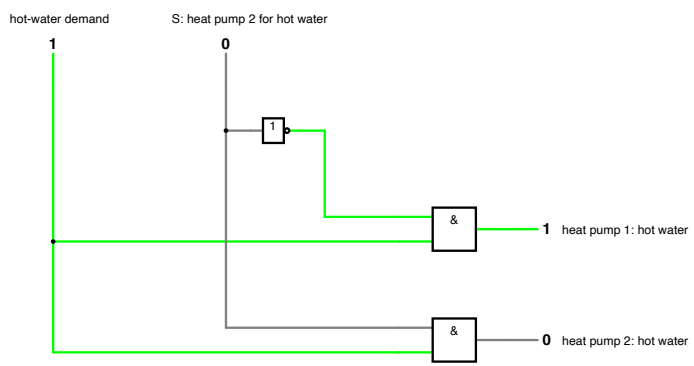
Example: a 1-to-2 demultiplexer with the input I, the select input S and the outputs Y_0 and Y_1

S	Y_0	Y_1
0	I	0
1	0	I

$$Y_0 = \overline{S} \cdot I \qquad Y_1 = S \cdot I$$

- With its input at 1, a demultiplexer with two select inputs is a 2-to-4 decoder; symbol according to IEC 60617: DX

Example: two heat pumps, one shower – the demultiplexer routes the hot-water demand to the heat pump that is defined for hot water



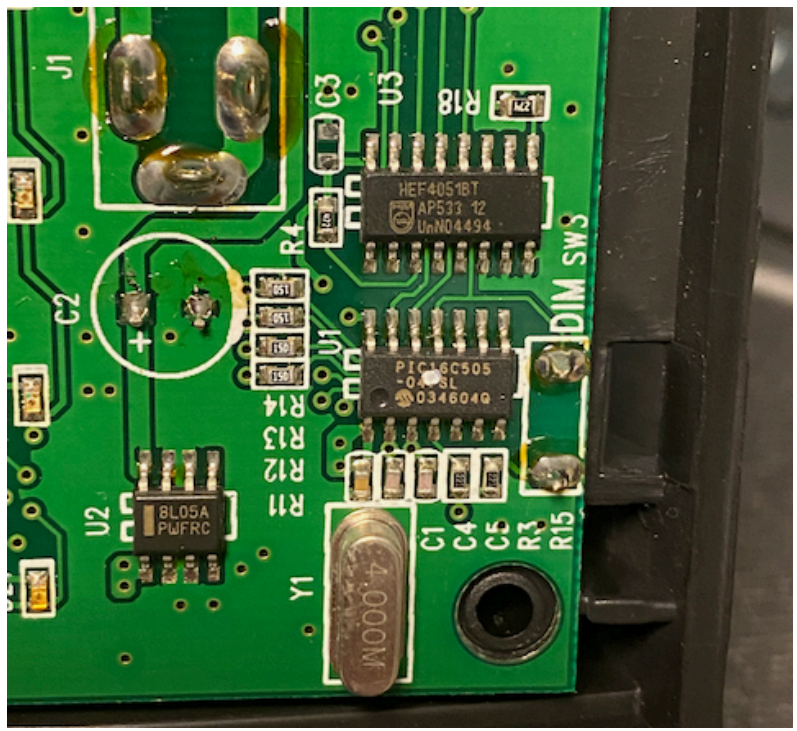
Open in the simulator

ONE CHIP, BOTH DIRECTIONS: THE HEF4051

Example: how does the BCD clock light its 20 LEDs?

- HEF4051: an 8-channel analogue multiplexer/demultiplexer with eight bidirectional analogue switches
- A 3-to-8 decoder closes one switch, chosen by the three select inputs; the enable input at 1 opens all switches
- The switches conduct in both directions: one chip for both jobs
- In the clock, a demultiplexer: it switches the six columns on one after another, faster than the eye can follow
- Time-division multiplexing: each electric line at the inputs gets its own time slot, and their signals share a single electric line one after another

Open in the simulator



HSBI

Hochschule Bielefeld

University of Applied Sciences and Arts

6 Logic functions with decoders and multiplexers

Step 5 of the synthesis (DET-03, 1.1) with standard components instead of single gates

- A decoder and an OR gate
- A multiplexer

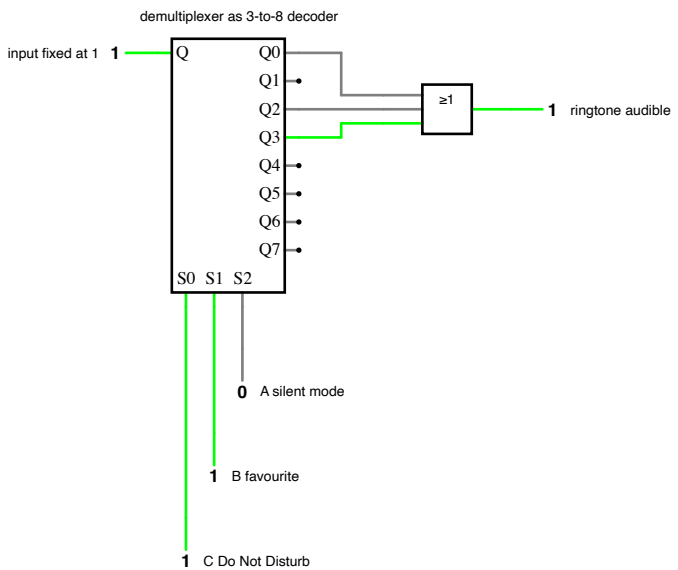
A DECODER AND AN OR GATE

- Each output of a decoder with n inputs and 2^n outputs is a minterm: the OR of the outputs that belong to the ones of the function is its canonical DNF – nothing to minimize

Example: the ringtone of your smartphone (DET-03, 1.2) – Y audible, A silent mode, B favourite caller, C Do Not Disturb

$$Y(A,B,C) = \sum m(0,2,3)$$

- A 3-to-8 decoder with A, B and C, and an OR gate over its outputs 0, 2 and 3
- Further functions of the same inputs: one more OR gate each



Open in the simulator

A MULTIPLEXER

- A 2^n-to-1 multiplexer implements any function of n variables: the variables at the select inputs, the truth table at the data inputs

Example: the ringtone again

- 8-to-1 multiplexer: A, B and C select, the data inputs I_0 to I_7 carry 1, 0, 1, 1, 0, 0, 0 and 0

Open in the simulator

- 4-to-1 multiplexer: A and B select, each data input carries 0, 1, C or $\overline{C}$

A	B	Rows	Y for $C = 0$ and $C = 1$	Data input
0	0	0 and 1	1 and 0	$I_0 = \overline{C}$
0	1	2 and 3	1 and 1	$I_1 = 1$
1	0	4 and 5	0 and 0	$I_2 = 0$
1	1	6 and 7	0 and 0	$I_3 = 0$

- No minimization: a different function only needs different connections

7 What have you learnt?

- The building blocks of knowledge
- Answers to the guiding questions

WHAT YOU HAVE LEARNT

- Codes and coding
- Code converters
- Standard components: decoder, 7-segment display decoder, encoder, multiplexer and demultiplexer

ANSWERS TO THE GUIDING QUESTIONS

- Many codes, because each is made for a different job and medium: you choose by what it represents, how it is transmitted or read, and which errors it has to detect or correct
- How a number turns into light:
  - BCD clock: lights the BCD code word, and you add the weights
  - 7-segment display: a decoder turns the code word into the shape of the digit
- Two heat pumps, one shower: as a circuit, a 1-to-2 demultiplexer



Image by Qwen-Image 2.1

Your turn: first on your own, then together.