DIGITAL TECHNICS I

Dr. Bálint Pődör

Óbuda University, Microelectronics and Technology Institute

8. LECTURE: NUMBER SYSTEMS AND CODES II

1st year BSc course 1st (Autumn) term 2018/2019

1

8. LECTURE

1. BCD type and other special codes
2. Arithmetical operations in BCD type codes
3. Encoders, decoders and code converters
4. Simple code converter circuits

2

REVISION: REPRESENTING NEGATIVE NUMBERS IN BINARY SYSTEM

Numbers: magnitude and sign.

Digital systems: storage in registers, for each bit one elementary storage cell (flip-flop, bistable).

Sign storage: one (additional) elementary cell. Usually the MSB.

Positive number: sign +, digital system: sign bit 0.
Negative number: sign -, digital system: sign bit1.

3

REPRESENTING NEGATIVE NUMBERS IN BINARY SYSTEM

Example: representing -5 on 4 bits

Sign and magnitude
+5 → 0 1 0 1   -5 → 1 1 0 1

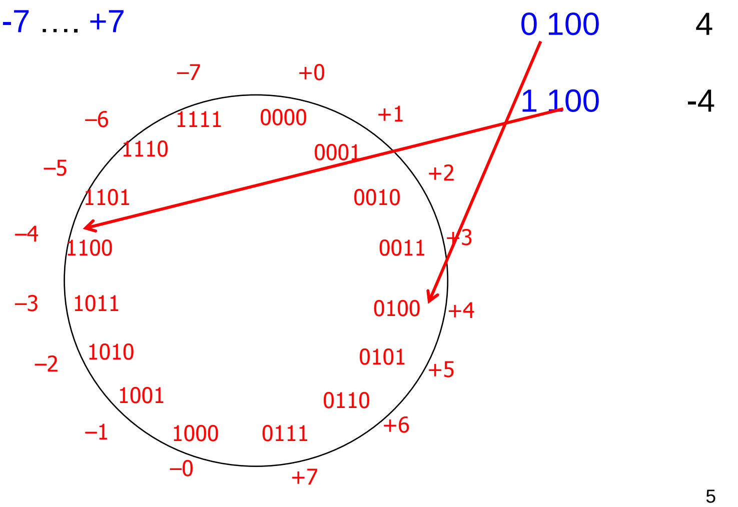
1's complement
+5 → 0 1 0 1   -5 → 1 0 1 0

2's complement
+5 → 0 1 0 1   -5 → 1 0 1 0
+1
1 0 1 1

4

SIGN AND MAGNITUDE (ON 4 BIT)

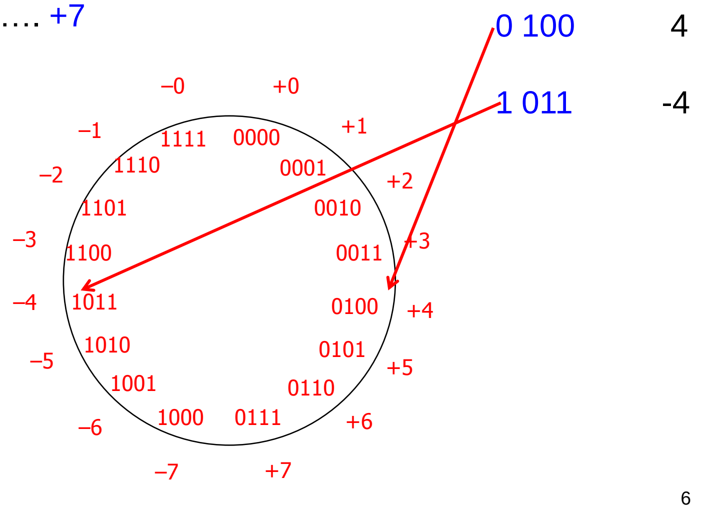
Range: -7 …. +7



5

1'S COMPLEMENT (ON 4 BIT)

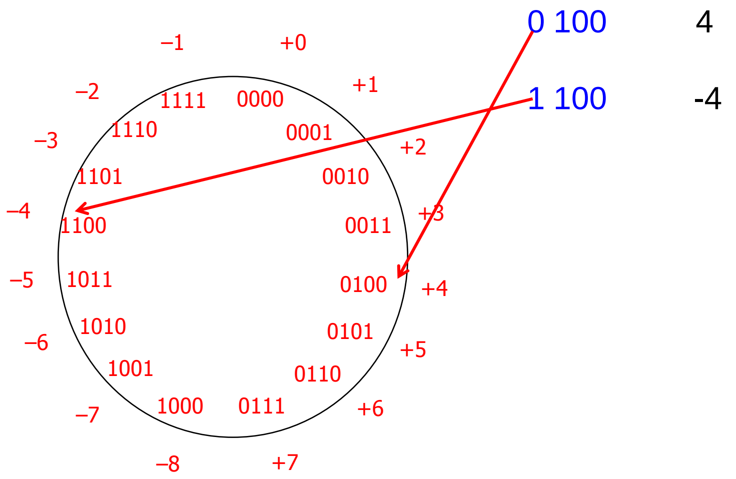
Range: -7 …. +7



6

2'S COMPLEMENT (4 BIT)

Range: -8 …. +7



7

BINARY ENCODING Of DECIMAL DIGITS

Storing and processing information: binary (1s or 2s complement) system.

Data in- and output: decimal system.

Expressing the individual digits of the decimal system (symbols 0,1,… 9) with binary codes:

Binary Coded Decimal (BCD)

8

TETRAD CODES AND THEIR ALGORITHMS

Weighted (positional value) codes
- ”normal” (natural) BCD code (NBCD), Aiken code, etc.

Non-weighted codes
- Stibitz (excess-3) code, Glixon code and other related one-step codes, etc.

Tetrad (nibble) code: a4a3a2a1 ai = 0,1

Weights: s4s3s2s1

Value of decimal digit:

d = a4s4 +a3s3 + a2s3 +a1s1

9

WEIGHTED TETRAD CODES

Important weighted tetrad codes and their weights:

8 4 2 1	normal or natural BCD code
5 4 2 1	
2 4 2 1	Aiken code (self-complementing)
4 2 2 1	Aiken code
5 3 1 1	
7 4 2 1	code minimizing the number of 1s
7 4 -2 -1	

10

NORMAL BCD CODE

Natural code
- Uses the binary equivalent of each digit
- Natural positional values: 8 4 2 1

$$d = 8a_4 + 4a_3 + 2a_2 + 1a_0$$

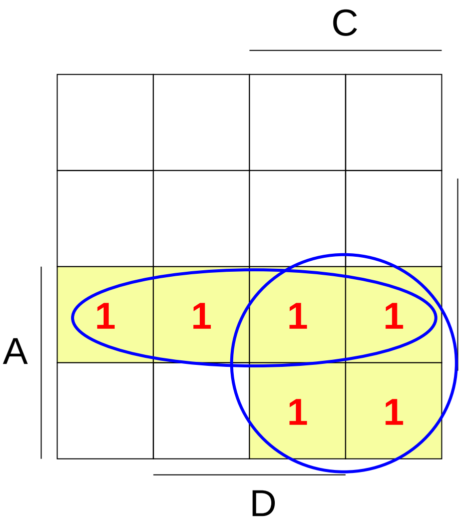
Six non-used (non-allowed) combinations:
(1010, ... 1111) pseudo-tetrad.

Decimal digit	8421 code
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
unused	1010
	1011
	1100
	1101
	1110
	1111

Valid code words

Not used (invalid) code words

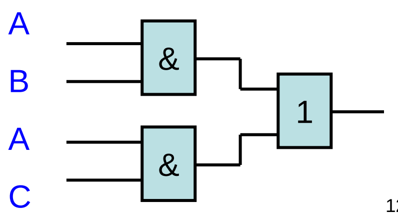
IDENTIFICATION OF INVALID CODEWORDS ON THE K-TABLE



Minimized:

P = A B + A C

Error detection logic:



AIKEN CODE

- 4,2,2,1 or 2,4,2,1 positional values
- Several arrangements are possible
- Self-complemeting BCD: 2 → 0010 and 7 → 1101
 (9's or diminished radix complement),
- Arithmetics: instead of subtraction addition of 9's
 complement +1:

$$d = 4a_4 + 2a_3 + 2a_2 + 1a_0$$

Tetrade indices: 0,1,2,3,6,9,12,13,14,15

Example:	528	528
	-347	+652
	171	1170 ⇒ +1 and discard overflow!

13

In-dex	bináris kombináció	Súlyozott kódok (a hozzárendelt decimális számjegyek)			nem súlyozott kód
		NBCD	4,2,2,1	8,4,(-2),(-1))	3 többletes
0	0000	0	0	0	-
1	0001	1	1	-	-
2	0010	2	2	-	-
3	0011	3	3	-	0
4	0100	4	-	4	1
5	0101	5	-	3	2
6	0110	6	4	2	3
7	0111	7	-	1	4
			Az önkomplementálás tükröző vonala		
8	1000	8	-	8	5
9	1001	9	5	7	6
10	1010	-	-	6	7
11	1011	-	-	5	8
12	1100	-	6	-	9
13	1101	-	7	-	-
14	1110	-	8	-	-
15	1111	-	9	9	-

Self-complementing codes, e.g.

4 2 2 1

8 4 -2 -1

Excess-3

14

VARIOUS BCD TYPE CODES

Decimal digit	8421 code	5421 code	2421 code	Excess 3 code	2 of 5 code
0	0000	0000	0000	0011	11000
1	0001	0001	0001	0100	10100
2	0010	0010	0010	0101	10010
3	0011	0011	0011	0110	10001
4	0100	0100	0100	0111	01100
5	0101	1000	1011	1000	01010
6	0110	1001	1100	1001	01001
7	0111	1010	1101	1010	00110
8	1000	1011	1110	1011	00101
9	1001	1100	1111	1100	00011
unused	1010	0101	0101	0000	any of
	1011	0110	0110	0001	the 22
	1100	0111	0111	0010	patterns
	1101	1101	1000	1101	with 0, 1,
	1110	1110	1001	1110	3, 4, or 5
	1111	1111	1010	1111	1's

WEIGHTED BCD CODES

Examples of weighted BCD codes (1)

Decimal digit	8; 4; 2; 1	2; 4; 2; 1	8; 4; -2; -1
0	0000	0000	0000
1	0001	0001	0111
2	0010	0010	0110
3	0011	0011	0101
4	0100	0100	0100

Examples of weighted BCD codes (2)

Decimal digit	8; 4; 2; 1	2; 4; 2; 1	8; 4; -2; -1
5	0101	1011	1011
6	0110	1100	1010
7	0111	1101	1001
8	1110	0011	1000
9	1001	1111	1111

16

EXCESS-3 OR STIBITZ CODE

Displaced weighted code

$$d = 8a_4 + 4a_3 + 2a_2 + 1a_0 - 3$$

- Uses the binary code of ”decimal digit + 3”
- Self-complementing code
- Aritmethics: carry on the 5th bit, the result should be corrected.

Tetrad indices: 3,4,5,6,7,8,9,10,11,12,

NON-WEIGHTED BCD CODES

Examples of non-weighted BCD codes (1)

Decimal digit	Excess 3	Biquinary	1-out-of-10
0	0011	0100001	1000000000
1	0100	0100010	0100000000
2	0101	0100100	0010000000
3	0110	010100	0001000000
4	0111	011000	0000100000

Examples of non-weighted BCD codes (2)

Decimal digit	Excess 3	Biquinary	1-out-of-10
5	1000	1000001	0000010000
6	1001	1000010	0000001000
7	1010	1000100	0000000100
8	1011	1001000	0000000010
9	1100	1010000	0000000001

ARITHMETICAL OPERATIONS IN TETRAD CODES

Many digital systems (processors, computers) can perform the arithmetical operations or a part of them directly on BCD numbers.

E.g. the microprocessors can perform BCD addition, several of them subtraction too. Certain special processors can perform BCD multiplication and division too.

The BCD addition is reduced to binary addition. The tetrads of the operands are added as binary numbers, and if necessary (illegal codewords or decimal carry is generated during the addition), a systematic correction is performed.

ADDITION IN NORMAL BCD (8421) CODE

If the sum of two tetrads is not larger than 9, the result is valid, no correction is necessary.

If the sum of two tetrads is larger than 9, (decimal carry and illegal codeword or pseudo-tetrad is generated) the result is valid only in binary system and not in BCD. The necessary correction is to add decimal 6 or i.e. binary 0110 to the actual tetrad.

The correction should be performed beginning form the least significant tetrad and going upwards step-by-step.

BCD (8421) ADDITION

Demo example:

decimal	BCD
427	0100 0010 0111
+ 131	+ 0001 0011 0001
558	0101 0101 1000

The sums were not larger than 9 at any decimal position no correction is necessary

BCD ADDITION: +6 CORRECTION

789	0111 1000 1001	
+ 213	+ 0010 0001 0011	
1002	1001 1001 1100	
	+ 0110	+6 corrrection
	1001 1010 0010	
	+ 0110	+6 correction
	1010 0000 0010	
	+ 0110	+6 correction
	1 0000 0000 0010	

BCD (8421) ADDITION ALGORITHM

$A_{BCD} +_{BCD} B_{BCD} = A_{BCD} +_{bin} B_{BCD}$

if $A_{BCD} +_{bin} B_{BCD} \leq 9$

$A_{BCD} +_{BCD} B_{BCD} = A_{BCD} +_{bin} B_{BCD} +_{bin} 6_{BCD}$

if $A_{BCD} +_{bin} B_{BCD} > 9$

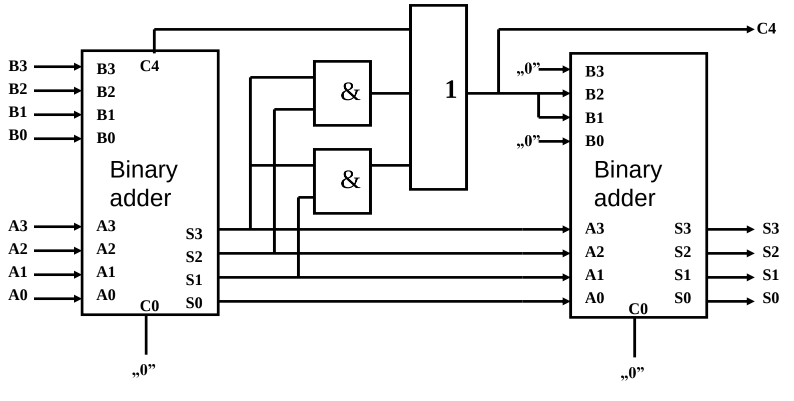
23

BCD ADDITION IMPLEMENTATION

Carry between decades

A + B > 9. illegal code

Decimal 6 (binary 0 1 1 0) correction



24

SSI MODULAR LOGIC: 4-BIT BCD ADDER

4-bit BCD adder **74F583**

FEATURES

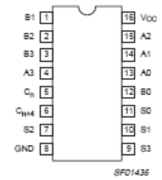
- Adds two decimal numbers
- Full internal look-ahead
- Fast ripple carry for economical expansion
- Sum output delay 19.5 ns max.
- Ripple carry delay 8.5 ns max.
- Input to ripple delay 13.0 ns max.
- Supply current 60 mA max.

DESCRIPTION

The 74F583 4-bit coded (BCD) full adder performs the addition of two decimal numbers (A0–A3, B0–B3). The look-ahead generates BCD carry terms internally, allowing the 74F583 to then do BCD addition correctly. For BCD numbers 0 through 9 at A and B inputs, the BCD sum forms at the output.

In addition of two BCD numbers totalling a number greater than 9, a valid BCD number and carry will result. For input values larger than 9, the number is converted from binary to BCD. Binary to BCD conversion occurs by grounding one set of inputs, An or Bn, and applying a 4-bit binary number to the other set of inputs. If the input is between 0 and 9, a BCD number occurs at the output. If the binary input falls between 10 and 15, a carry term is generated. Both the carry term and the sum are the BCD equivalent of the binary input. Converting binary numbers greater than 16 may be achieved by cascading 74F583s.

PIN CONFIGURATION



TYPE	TYPICAL PROPAGATION DELAY	TYPICAL SUPPLY CURRENT (TOTAL)
74F583	9.0 ns	45 mA

ORDERING INFORMATION

PACKAGE	COMMERCIAL RANGE V_{CC} = 5 V ±10% T_{amb} = 0°C to +70°C	DRAWING NUMBER
16-pin plastic DIP	N74F583N	SOT38-4
16-pin plastic SO	N74F583D	SOT109-1

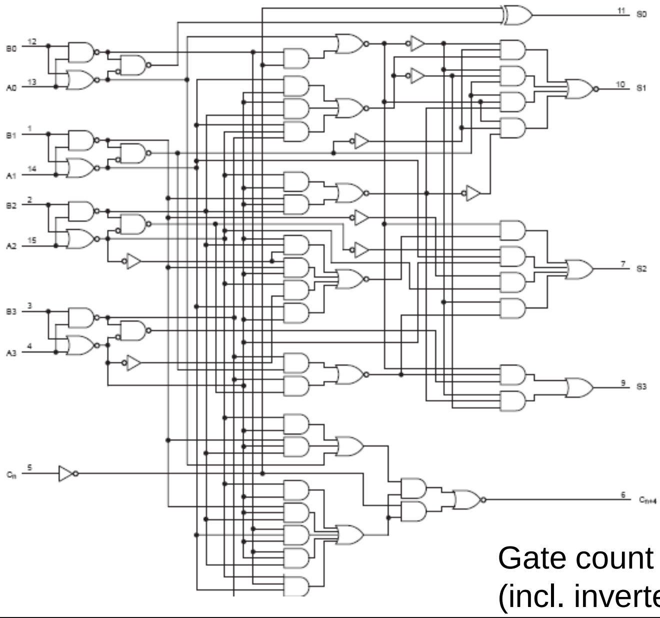
SSI MODULAR LOGIC: 4-BIT BCD ADDER

The 74F583 4-bit coded (BCD) full adder performs the addition of two decimal numbers (A0–A3, B0–B3). The look-ahead generates BCD carry terms internally, allowing the 74F583 to then do BCD addition correctly.

For BCD numbers 0 through 9 at A and B inputs, the BCD sum forms at the output.

In addition of two BCD numbers totalling a number greater than 9, a valid BCD number and carry will result

4-BIT BCD ADDER LOGIC DIAGRAM



Gate count 75
(incl. inverters)

27

ADDITION IS STIBITZ (EXCESS-3) BCD CODE

Addition is performed like in binary, and correction is always necessary.

Rule (algorithm):
If no decimal carry, subtract 3 (0011), if decimal carry add 3 (0011) at each tetrade.

Advantage:
The correction itself does not generate decimal carry.
Thereore the correction procedure can be performed for Ech tetrade (decimal position) independently and simultaneously.

28

EXCESS-3 (STIBITZ) CODE: ADDITION

Example:

decimal	Excess-3
427	0111 0101 1010
+ 131	+ 0100 0110 0100
558	1011 1011 1110
	-0011 -0011 -0011
	1000 1000 1011
	5 5 8

Because there was no decimal carry at any of the tetrades, -3 correction is necessary!

29

ADDITION IN AIKEN CODE

Addition like in binary, then correction for each tetrade:

- If no illegal codeword, no correction.

- If illegal codeword, and also decimal carry the correction is + 6, if no decimal carry the correction is - 6.

30

SUBTRACTION IN TERTRADE CODES

Like in the binary system the subtraction is reduced to addition in complementary code.

Both 9s and 10s complement representation can be used, for the former of course +1 correction should be applied to the result.

31

MULTIPLICATIVE OPERATIONS IN TETRADE CODES

Appropriate algorithms are available tor multiplication and division in BCD system.
A common is the division and multiplication by 2 (used e.g. in binary-to-BCD or BCD-to-binary conversion). Or this, in order to improve efficiency, there exist separate algorithms.

Binary: shift with one position to right or left.

BCD multiplication with 2: shift one position to left, and +6 correction as appropriate.

32

ONE-STEP CODES, GRAY CODE

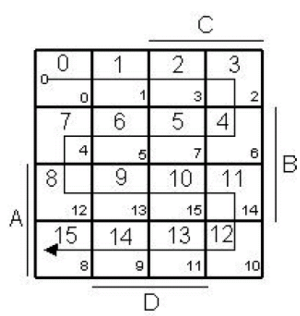
The Gray code is a special case of the one-step codes. It is a set of 2^n n-bit codewords in such a sequence that any two neighbouring codewords differ only in one bit (place). This applies also to the first and last codewords (cyclic property).

Applications: measurements and instrumentation, automatics, position (linear or angle) sensing and encoding, etc.

33

4-BIT GRAY CODE ON THE KARNAUGH MAP

mi	I	A	B	C	D
0	0	0	0	0	0
1	1	0	0	0	1
3	2	0	0	1	1
2	3	0	0	1	0
6	4	0	1	1	0
7	5	0	1	1	1
5	6	0	1	0	1
4	7	0	1	0	0
12	8	1	1	0	0
13	9	1	1	0	1
15	10	1	1	1	1
14	11	1	1	1	0
10	12	1	0	1	0
11	13	1	0	1	1
9	14	1	0	0	1
8	15	1	0	0	0



Construction rules of Gray code on 4-bits

34

BIN/GRAY CONVERSION EXAMPLE: 3-BIT GRAY CODE

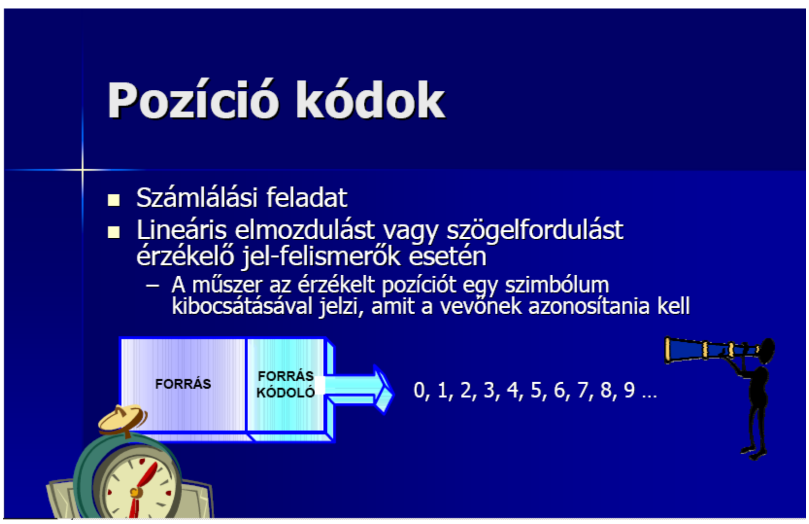
Dec	Bin	Gray
0	0 0 0	0 0 0
1	0 0 1	0 0 1
2	0 1 0	0 1 1
3	0 1 1	0 1 0
4	1 0 0	1 1 0
5	1 0 1	1 1 1
6	1 1 0	1 0 1
7	1 1 1	1 0 0

Bin/Gray conversion:

- first bit is the same as the first bit (MSB) of binary code,
- the second bit is given by XOR of the 1st and 2nd bit of bunary code,
- the third bit is given by XOR of the 2nd and 3rd bit of bunary code,
- and so on.

35

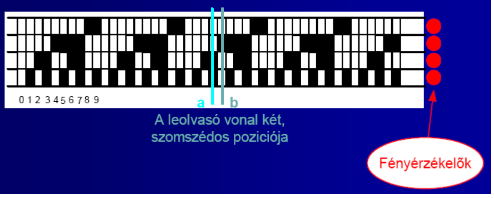
POSITION CODES



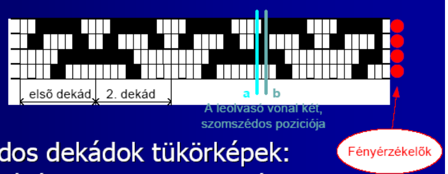
36

POSITION CODES

- Pl. NBCD „kellemetlen”
  - Átmenetkor több érzékelő is vált



- Gray kód
  - Két „szomszédos” kódszó mindig egy-egy bitben tér el:



- Szomszédos dekádok tükörképek:
  - Dekád-váltáskor is csak egy bit változik

BINARY/GRAY AND GRAY/BINARY CONVERSION ALGORITHMS

Binary: b3b2b1b0	Gray: a3a2a1a0
Binary → Gray	Gray → Binary

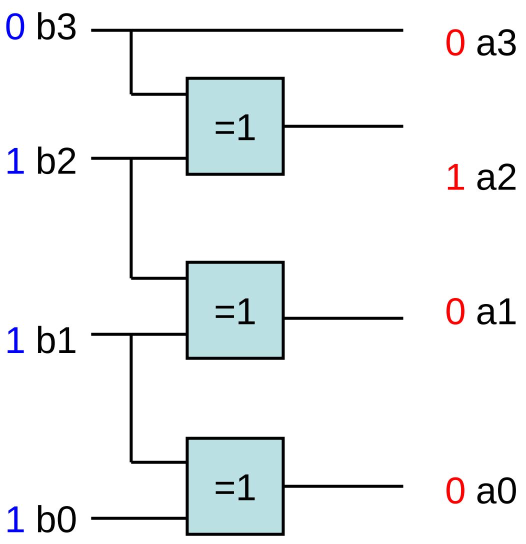
$a3 = b3$	$b3 = a3$
$a2 = b3 \oplus b2$	$b2 = a3 \oplus a2$
$a1 = b2 \oplus b1$	$b1 = a3 \oplus a2 \oplus a1 = b2 \oplus a1$
$a0 = b1 \oplus b0$	$b0 = a3 \oplus a2 \oplus a1 \oplus a0 =$ etc.

Binary → Gray: $ai = bi{+}1 \oplus bi$

Gray → Binary: $bi = bi{+}1 \oplus ai$

38

BINARY/GRAY CODE CONVERTER



39

OTHER ONE-STEP CODES

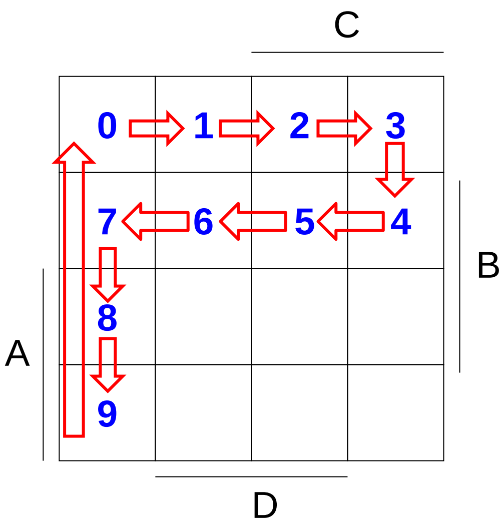
Various similar (one-step) codes are known, also in BCD versions .

E.g. Glixon-code, tetrad codewords

0000 (0) → 0001 (1) → 0011 (2) → 0010 (3) →
0110 (4) → 0111 (5) → 0101 → 0100 (7) →
1100 (8) → 1000 (9)

40

BCD-TO-GLIXON CODE CONVERSION



Synthesize (normal) BCD/Glixon code converter. It can be supposed that the invalid codewords never occur on the inputs..

The Glixon code is a one-step BCD code. Codewords from 0 through 9 are 0-tól 9-ig 0000, 0001, 0011, 0010, 0110, 0111, 0101, 0100, 1100, and 1000.

41

EXAMPLE 5 (EXCLUSIVE-OR LOGIC): BCD-TO-GLIXON CODE CONVERTER

The Glixon code is a one-step BCD code (the Hamming distance is 1). The code words from 0 to 9 are

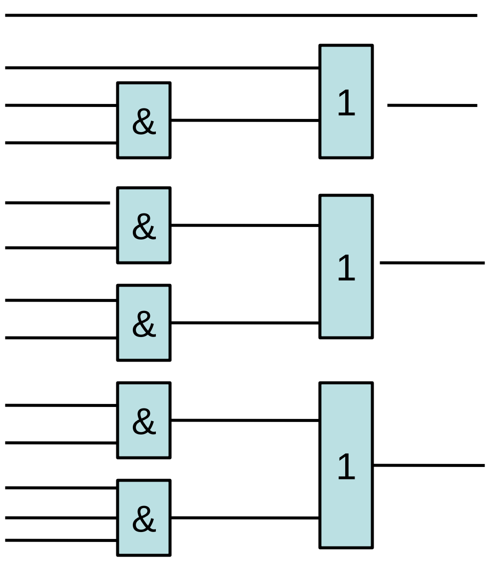
0000(0)	0001(1)	0011(2)	0010(3)	0110(4)
0111(5)	0101(6)	0100(7)	1100(8)	1000(9)

Normal BCD code:	ABCD (A is the MSB)
Glixon code:	E3, E2, E1, E0 (E3 is the MSB).

E3 = Σ4(8,9)X(10-15)
E2 = Σ4(4-8)X(10-15)
E1 = Σ4(2-5)X(10-15)
E0 = Σ4(1,2,5,6)X(10-15)

42

MINIMIZED TWO LEVEL AND-OR CIRCUIT



$E3 = A$

$E2 = B + A\overline{D}$

$E1 = B\overline{C} + \overline{B}C$

$E0 = C\overline{D} + \overline{A}\,\overline{C}D$

Pin count 18
Gate count 8

43

AND-OR-EXCLUSIVE LOGIC IMPLEMENTATION

Using XOR gates in implementing (partially) symmetric logic functions make possible to obtain more economical solutions than using standard optimized two-level AND-OR or OR-AND circuits.

Here this approach can be applied to E1 and E0.

Full or partial symmetry is evident on the Karnaugh map by noting the chessboard patterns.

44

AND-OR-XOR LOGIC: E1

C

		1	1
1	1		
X	X	X	X
		X	X

B

A

D

$E1 = \bar{B}\,C + B\,\bar{C}$

(three gates)

$E1 = B \oplus C$

One gate instead of three!

45

AND-OR-XOR LOGIC: E0

C

	1		1
	1		1
X	X	X	X
		X	X

B

A

D

$E0 = C\,\bar{D} + \bar{A}\,\bar{C}\,D$

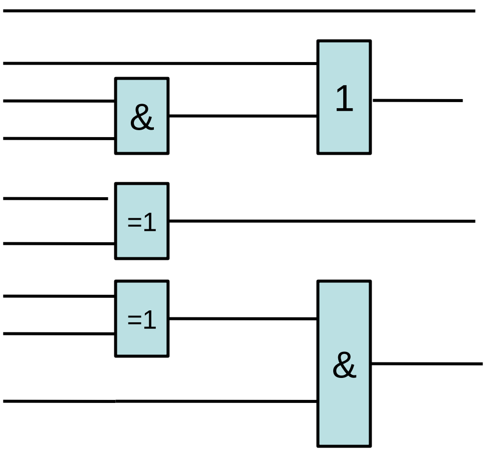
(three gates)

$E0 = \bar{A}\,(C \oplus D)$

Two gates instead of three!

46

AND-OR-XOR LOGIC IMPLEMENTATION



$E3 = A$

$E2 = B + A\overline{D}$

$E1 = B \oplus C$

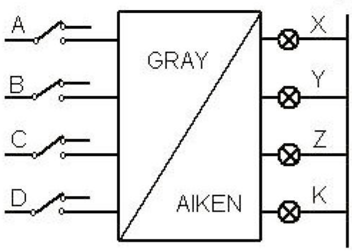
$E0 = \overline{A}(C \oplus D)$

Pin count 11
Gate count 5

Additional benefit: 2 inverters instead of 4 at the input!

47

OTHER EXAMPLE: GRAY/AIKEN CODE CONVERSION



It should operate from 0 to 8!
- Setting-up the truth table
- Minimization of logic
- Realization with NAND/NAND network

		Gray code				Aiken code				
		8	4	2	1					
mi	I	A	B	C	D	X	Y	Z	K	
0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	1	0	0	0	1	1
3	2	0	0	1	1	0	0	1	0	2
2	3	0	0	1	0	0	0	1	1	3
6	4	0	1	1	0	0	1	0	0	4
7	5	0	1	1	1	1	0	1	1	5
5	6	0	1	0	1	1	1	0	0	6
4	7	0	1	0	0	1	1	0	1	7
12	8	1	1	0	0	1	1	1	0	8

48

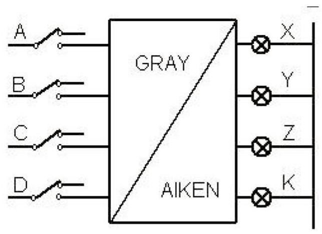
GRAY/AIKEN CODE CONVERTER: MINIMIZED FUNCTIONS

$X = \bar{C}\,B + B\,D$

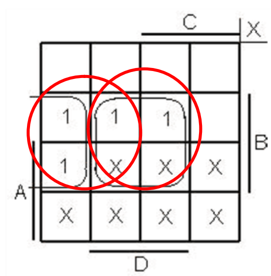
$Y = B\,\bar{D} + C\,\bar{B}$

$Z = A + C\,D + \bar{B}\,C$

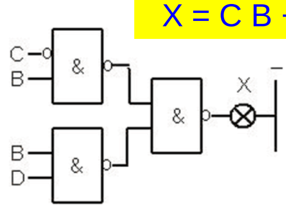
$K = \bar{A}\,B\,\bar{C}\,\bar{D} + \bar{B}\,\bar{C}\,D + B\,C\,D + \bar{B}\,C\,\bar{D}$

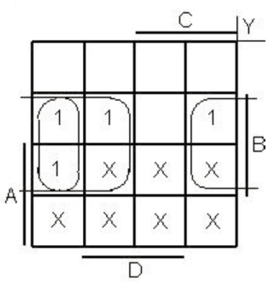


IMPLEMENTATION (X AND Y)

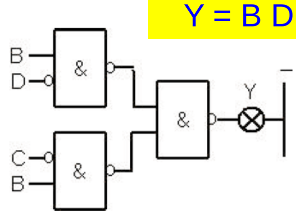


$X = \bar{C}\,B + B\,D$

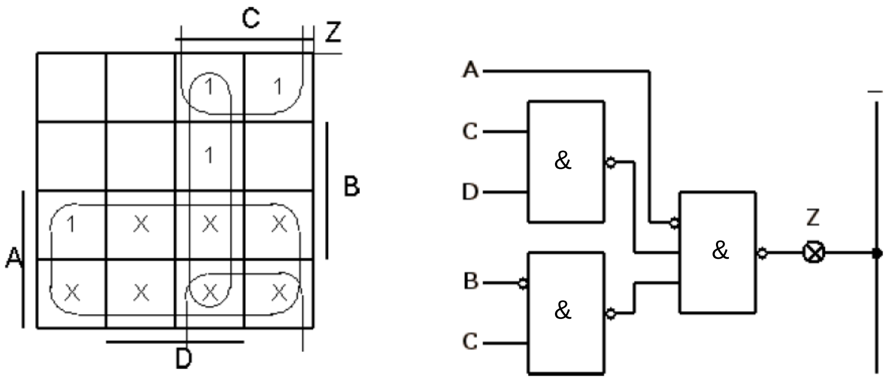




$Y = B\,\bar{D} + C\,\bar{B}$



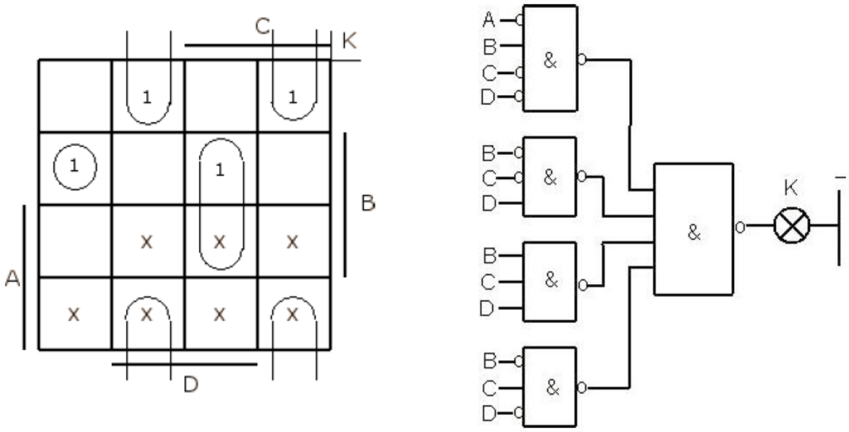
IMPLEMENTATION (Z)



$Z = A + C\,D + \overline{B}\,C$

51

IMPLEMENTATION (K)



$K = \overline{A}\,B\,\overline{C}\,\overline{D} + \overline{B}\,\overline{C}\,D + BCD + \overline{B}\,C\,\overline{D}$

52

BINARY-TO-BCD CONVERSION

Correction (+6):

- generation of invalid code (pseudo-tetrad),
- stepping through decade boundaries.

Conversion process:

- binary number to be converted shifted to left beginning with the MSB,
- after each step the conditions for correction are examined, and the eventual corrections are implemented.

53

BINARY-TO-BCD CONVERSION ALGORITHM

	* 1 1 1 0 0 1(bin) = 57(dec)	
	1 * 1 1 0 0 1	left shift
	1 1 * 1 0 0 1	left shift
	1 1 1 * 0 0 1	left shift
	* 1 1 1 0 * 0 1	left shift
Invalid tetrade	+0 1 1 0 *	correction
	1 * 0 1 0 0 * 0 1	left shift
	1 0 * 1 0 0 0 * 1	left shift
decade	1 0 1 * 0 0 0 1 *	left shift
crosing	+0 1 1 0 *	correction
	1 0 1 * 0 1 1 1 *	
	5 7	

54

BINARY-TO-BCD CONVERSION

After shift correction +6 if any of the two conditions are fulfilled.

The correction can b performed before the shift by adding +3, if the decimal value of the tetrad is 5 or larger.

Advantage: in this case because of the correction numbers larger than 9 never occur, so the correction with respect to crossing the decade boundary is automatic.

Therefore only one type of correction logic circuit is necessary.

55

END OF LECTURE

56