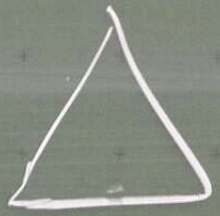
NOR

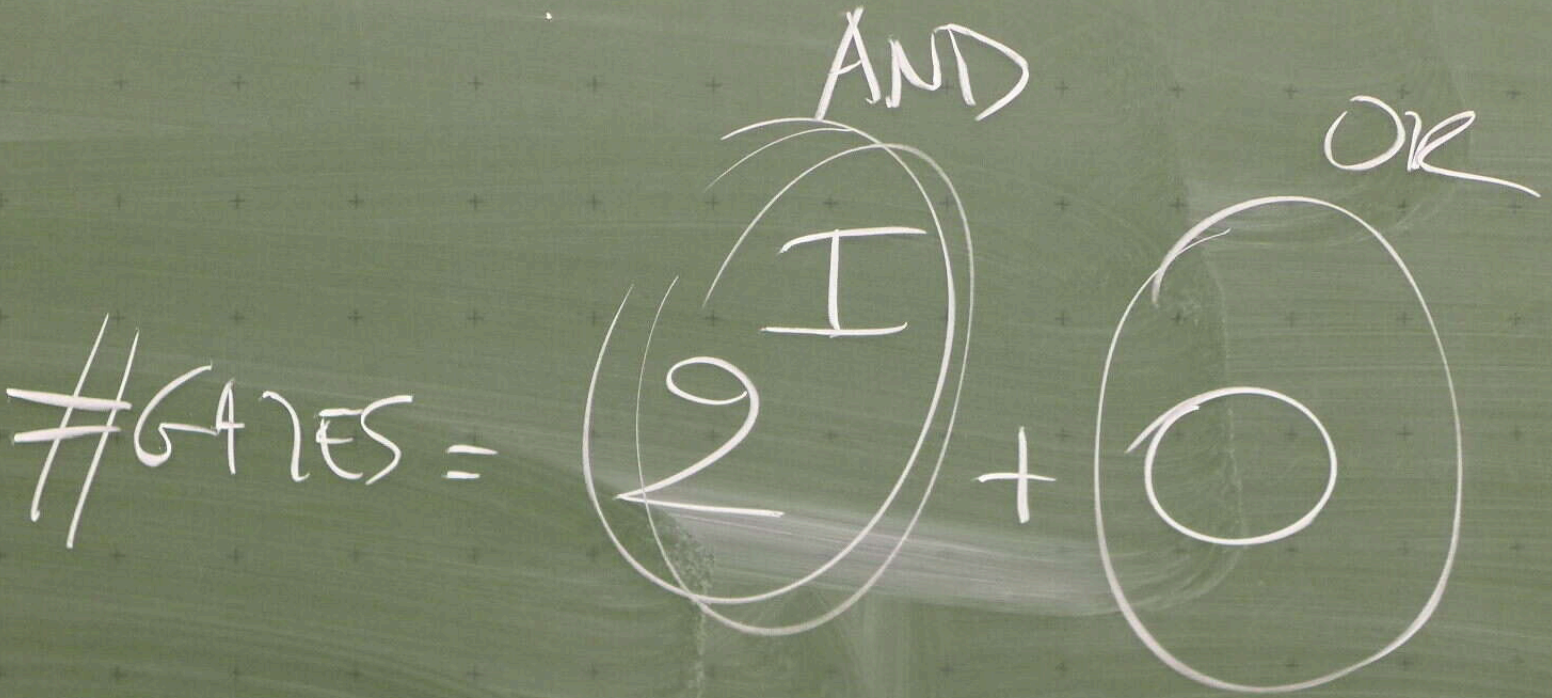
NAND

OR

AND

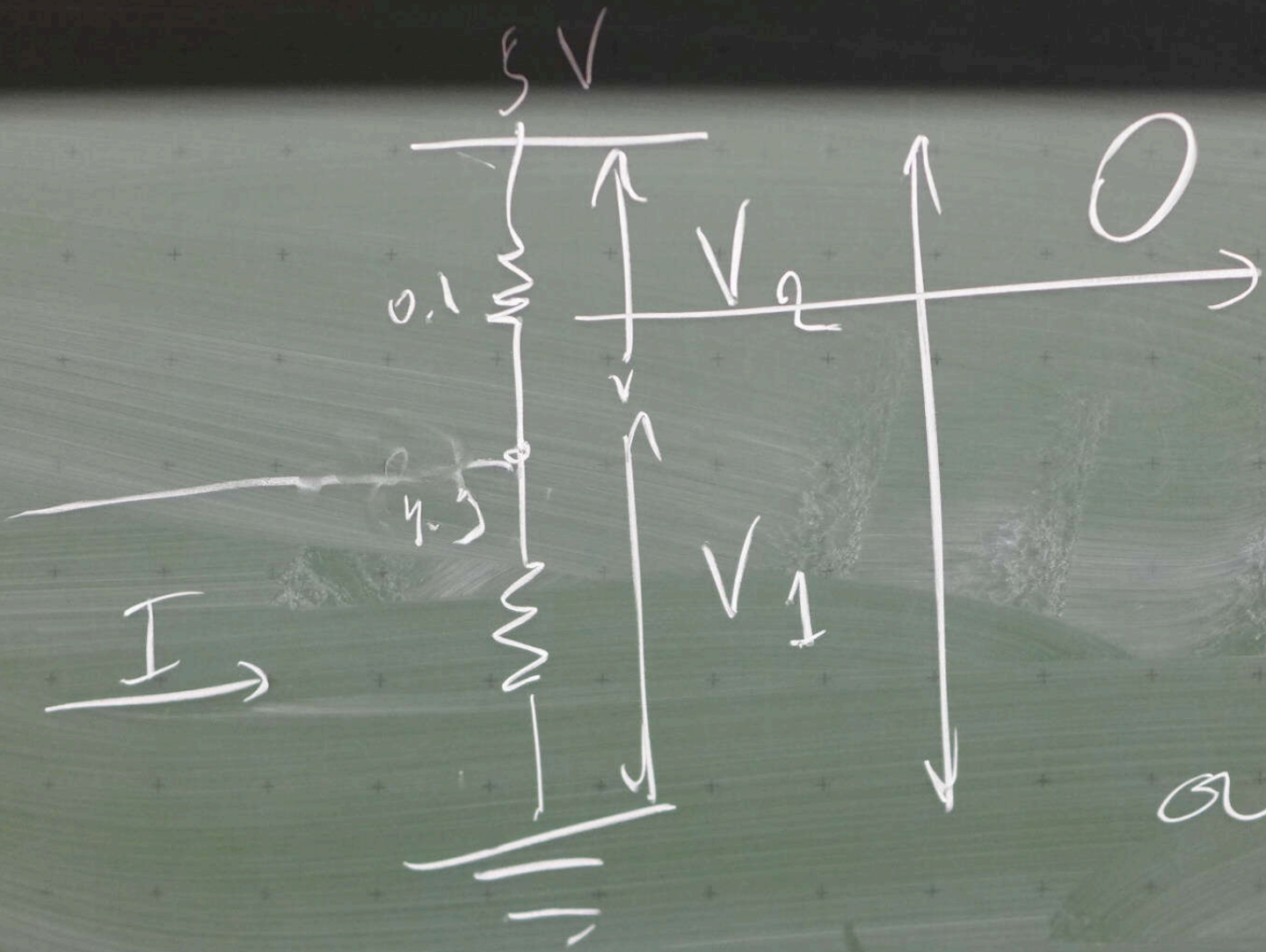


NOT


AND
OR
#GATES = 2^I + O

$$out_1 = (in1 \cdot in2) + (in3 \cdot 1)$$

$$= (in1 + in3) \cdot (in2 + in3)$$


5 V
0.1
4.3
I
V_2
V_1
O

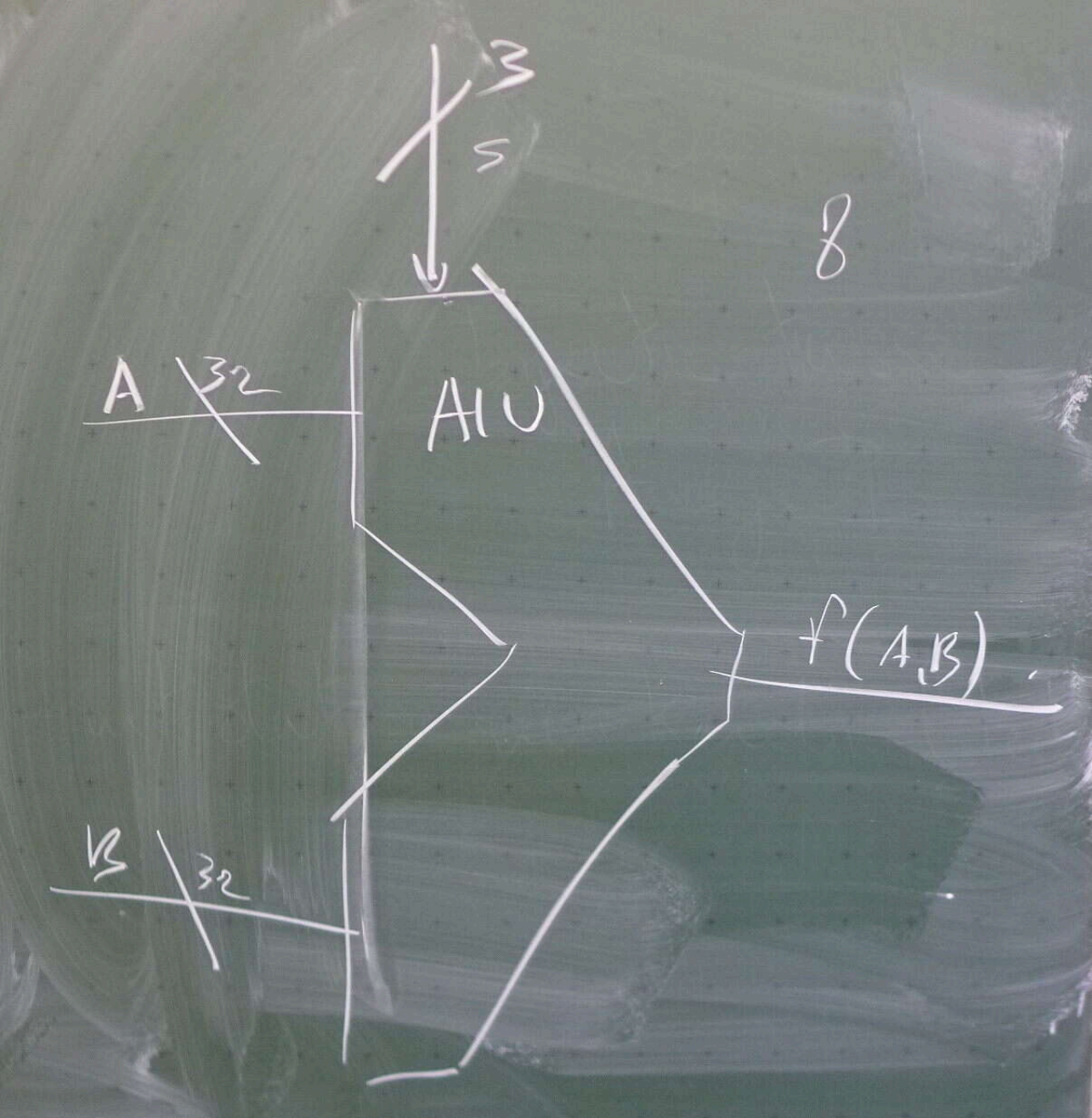
$[\![1 + 1]\!] =$

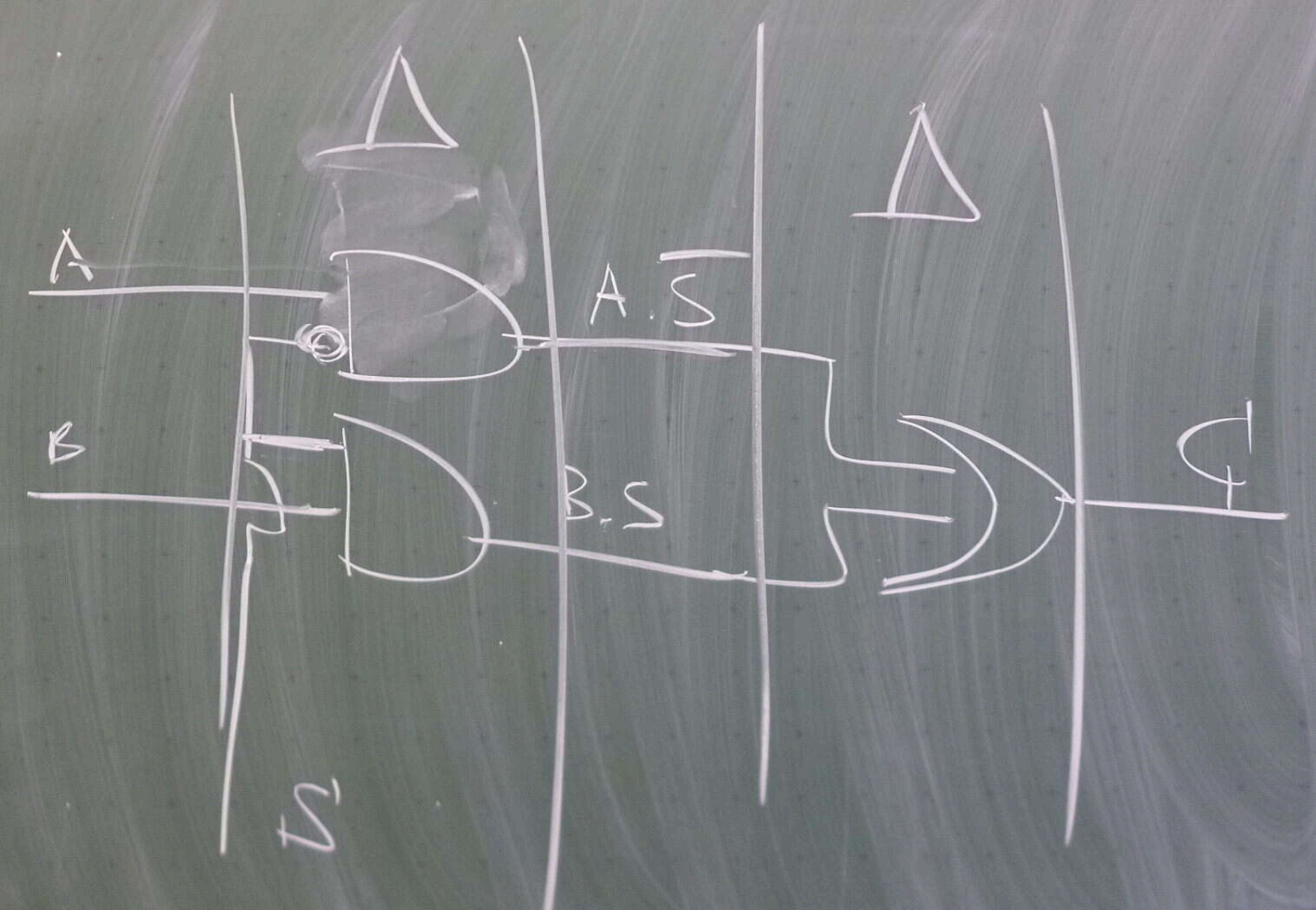
BOOLEAN

1

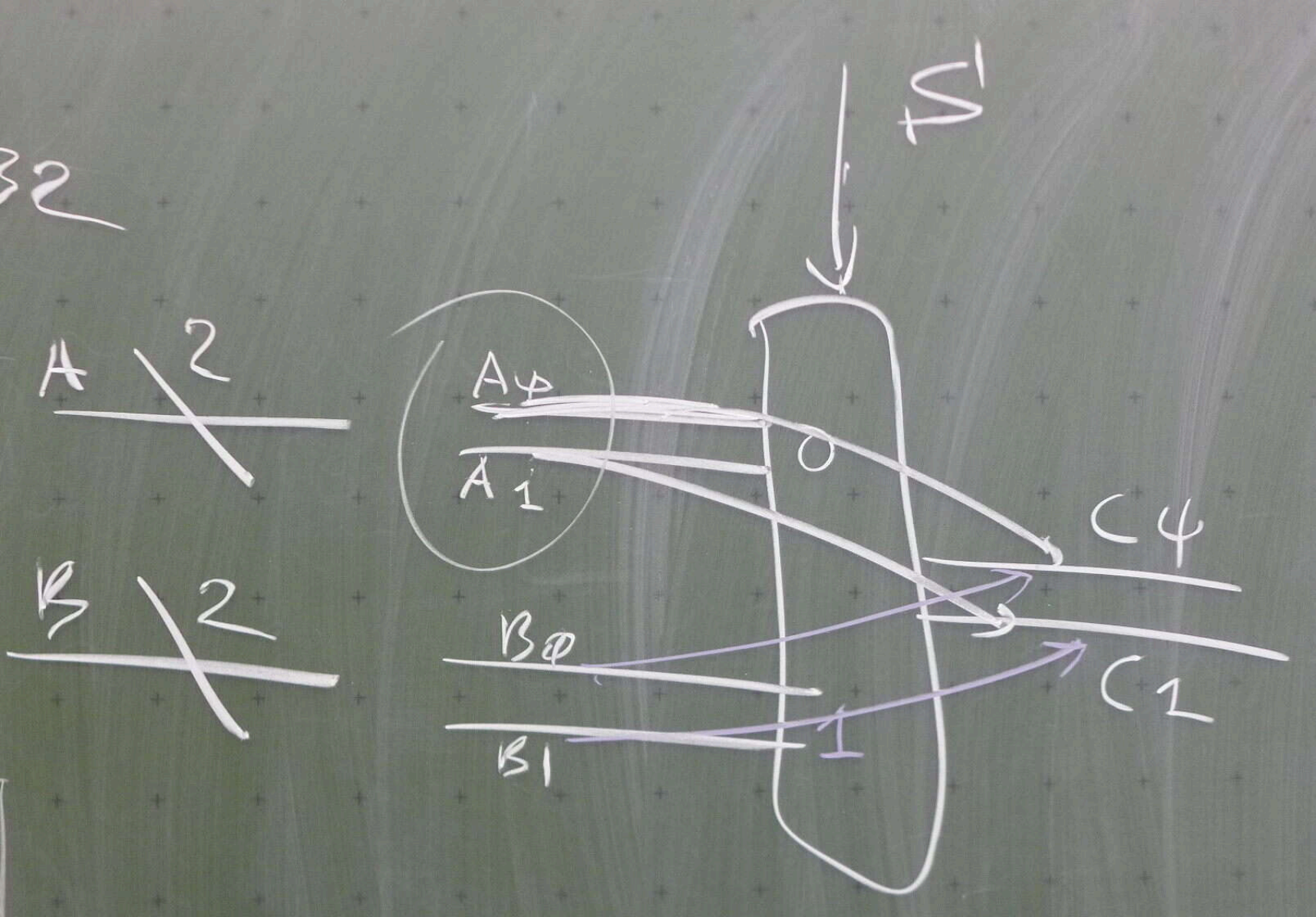
$\mathbb{N}$

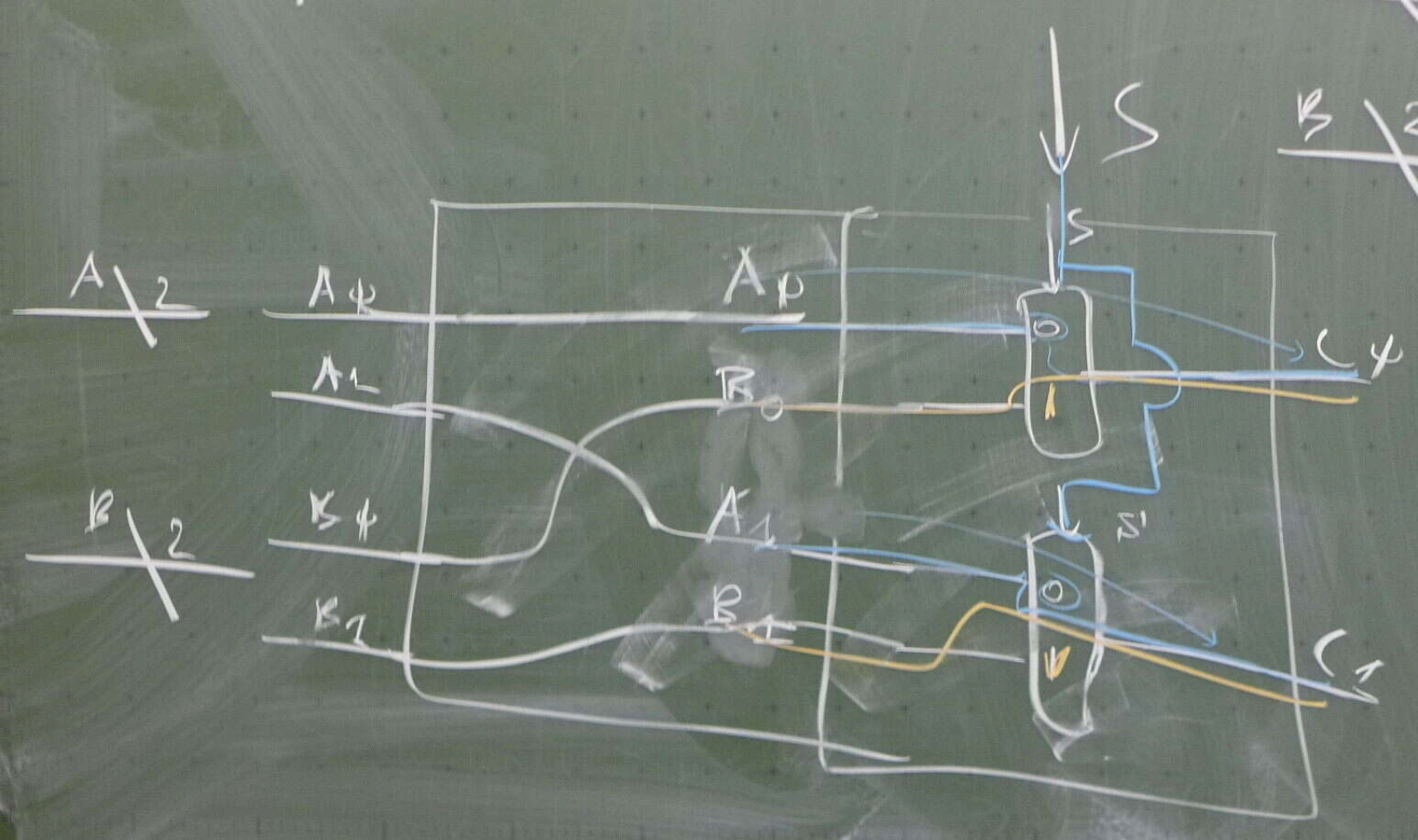
2

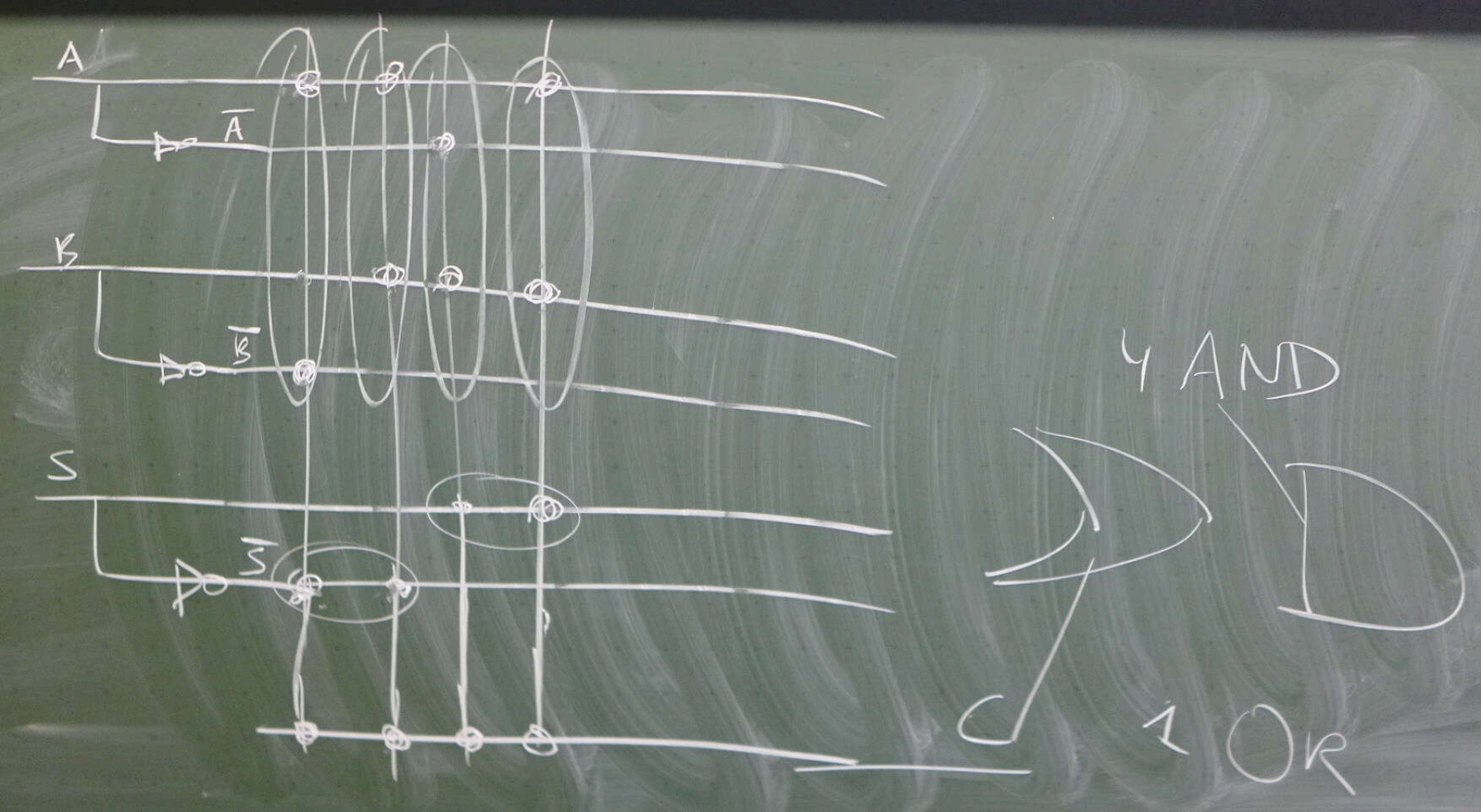

3
5
8
A
32
AUV
f(A,B)
B
32


A
B
A.S
B.S
C

$$\#\text{GATES} = 2^{I=64} + O^{32}$$


32
A
2
B
2
A0
A1
B0
B1
S1
0
1
C0
C1

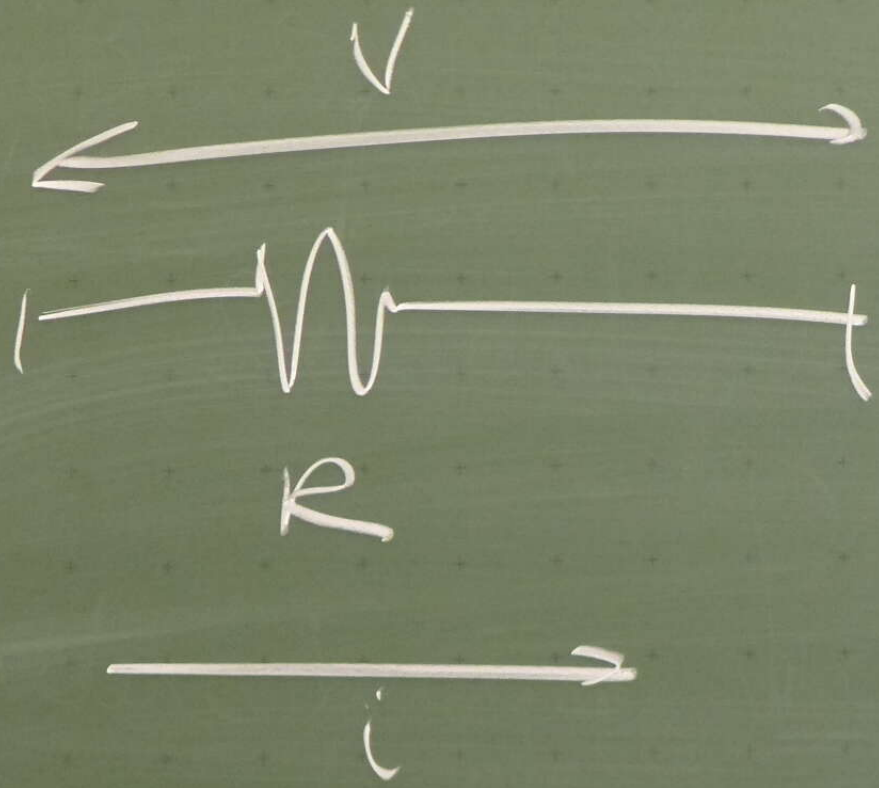

S
B
A/2
A ψ
A p
A 1
B 0
B/2
B ψ
A 1
B 1
B 1
S'
C ψ
C 1


A
$\overline{A}$
B
$\overline{B}$
S
$\overline{S}$
4 AND
1 OR

$$\overline{S} \cdot \left(A \cdot \overline{B} + \overline{S} A \cdot B \right) + S \cdot \left(\overline{A} \cdot B + S \cdot A \cdot B \right)$$

$$S \cdot \left(A \cdot \overline{B} + A \cdot B \right) + S \cdot \left(\overline{A} \cdot B + A \cdot B \right)$$

$$\overline{S} \cdot A \cdot \underbrace{\left(\overline{B} + B \right)}_{1} + S \cdot B \cdot \underbrace{\left(\overline{A} + A \right)}_{1}$$



$$V = Ri$$

$$Ri^2$$ HEAT

IMPLEMENTIERT

PROGRAMM ↑ SORTING ALGORITHMUS

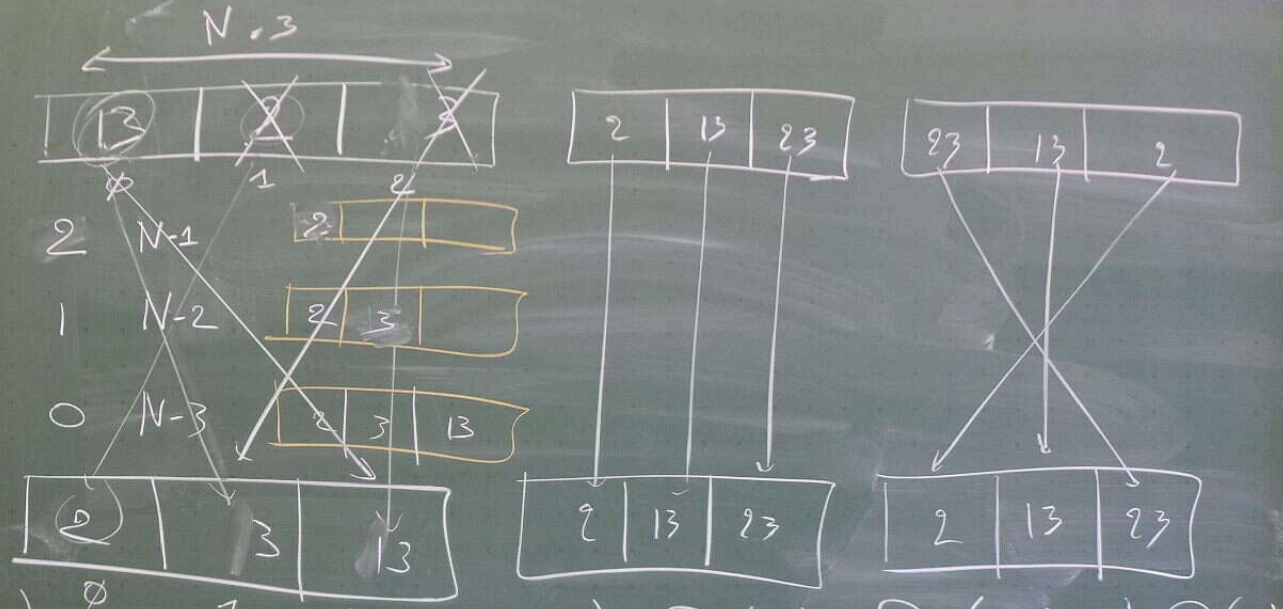
$t(x \leqslant y) = \Delta$

KOMPLEXITÄT $\sim 3\Delta$

KOMPLEXITÄT $= (0 + 1 + 2 + 3 + \cdots + (N-2) + (N-1))\Delta$

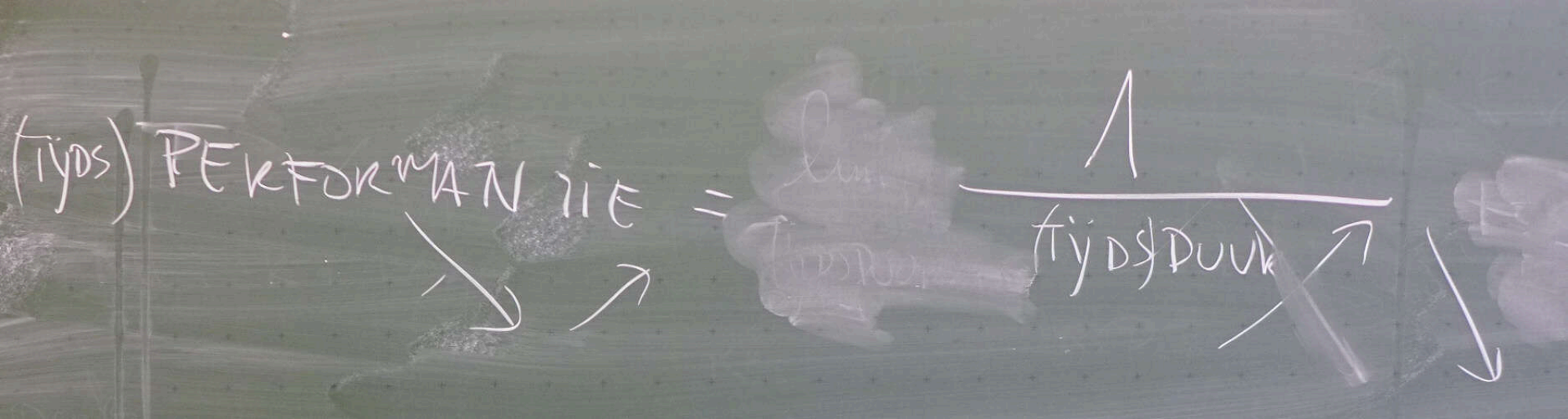
$= \frac{N(N-1)}{2}\Delta = \frac{N^2}{2} - \frac{N}{2}$

$N \gg\gg \quad \Theta(N^2) \quad \Theta(2^N) \quad \Theta(N) \quad \Theta(\ln N) \quad \Theta(1)$



$$1 + 2 + \cdots + (N-1) + N = \frac{N(N+1)}{2}$$

$$1 + 2 + \cdots + (N-1) = \frac{(N-1)N}{2}$$


(tijds) PERFORMANTIE = 1 / tijdsduur

- REQUIREMENTS

ANALYST

ALGORITHM

COMPLEXITY? WORST AVG BEST

INSERTION SORT $\Theta(N^2)$ ✗ ✓ ✓

BUBBLE SORT

PROGRAMMER

QUICK SORT $\Theta(N \log N)$ ✓ ✓ ✓

- PROGRAMMA (HLL)

newvec[i] = vec[j];

COMPILER

- PROGRAMMA (ASM, MACHINE)

CODE

ADD R1 R5 R2

ASSEMBLER

0110 0111 0101 ... 11

32 bit

32

DATAPATH

IDEAL.

#GATES = $2^I + O$

TIME = 2Δ

- HARDWARE (DATAPATH)

CMP

ADD

MULT

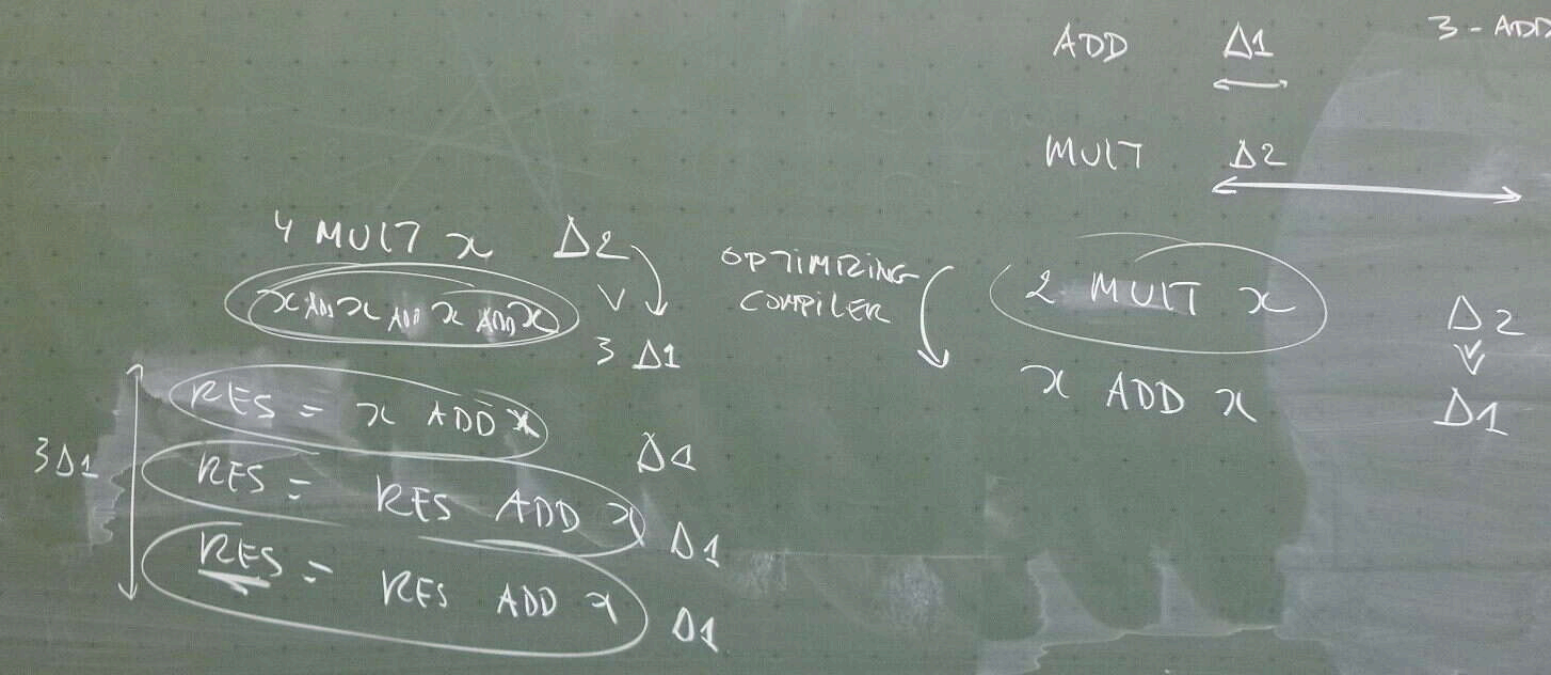
COMPLEXITY? OPERATIONS

BITS / 32 bit

OPERATIONS

Δ Δ

2Δ


ADD Δ1
MULT Δ2
3 - ADD
4 MULT x Δ2
x ADD x ADD x ADD x
3 Δ1
3Δ1
RES = x ADD x
Δ1
RES = RES ADD x
Δ1
RES = RES ADD x
Δ1
OPTIMIZING COMPILER
2 MULT x
x ADD x
Δ2
Δ1

- PROGRAMMA (ASM, MACHINE)

3 - ADDRESS CODE

ADD R3 R5 R1

ASSEMBLER

32 bit

32

DATAPATH

IDEAAL:

#GATES = $2^I + O$

TIME = 2Δ

- HARDWARE (DATAPATH)

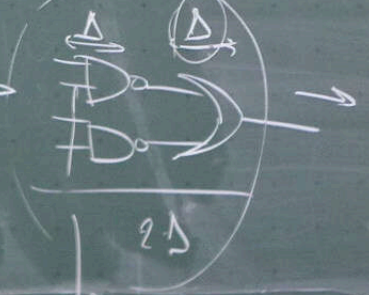
CMP
ADD
MULT

COMPLEXITEIT OPERATIONS

BITS

32 bit

OPERATIONS



$\Delta 2$

$\Delta 1$

REQUIREMENTS

ANALYST

ALGORITME

PROGRAMMEUR

PROGRAMMA

```
newvec[i] = vec[j];
```

(HLL)

COMPILER

	COMPLEXITEIT	WORST	AVG	BEST
INSERTION SORT	$\Theta(N^2)$	×	✓	✓
BUBBLE SORT				
QUICK SORT	$\Theta(N(\ln N)$	✓	✓	✓

PROGRAM (HLL)

SUM = 0

FOR CTR IN 0..5:

SUM = SUM + CTR

(IV-1)

(SUM += CTR)

=

1

LOC = 4

WHILE TRUE:

PASS

LOC = 2

```
SUM = 0
[ CTR = 0
  SUM = SUM + CTR
[ CTR = 1
  SUM = SUM + CTR
[ CTR = 2
```

INSTRUCTIONS

$2 \times 10 + 1 = 21$

$2 \times N + 1$ ---

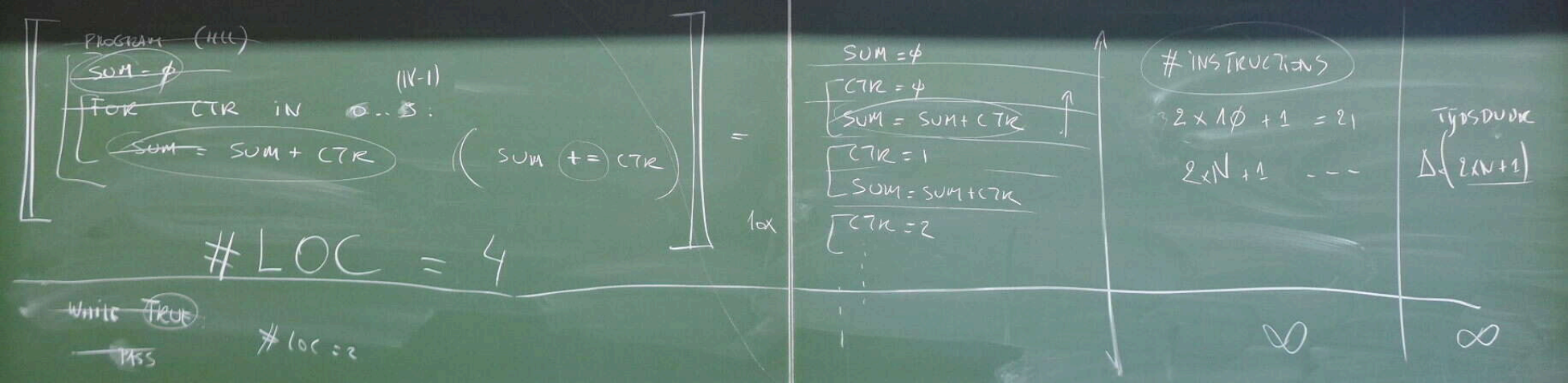
TYDSDUUR

$\Delta t(2 \times N + 1)$

10x

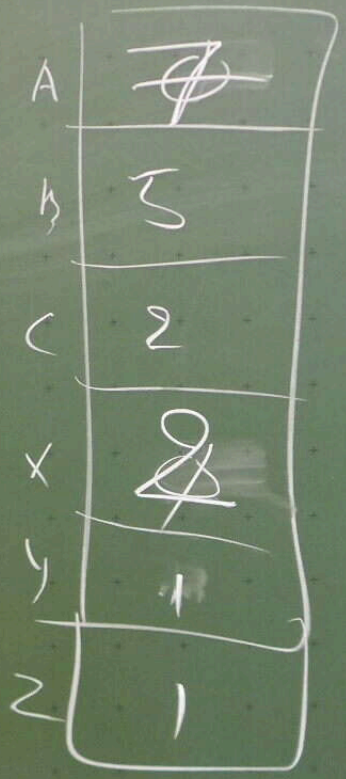
∞

∞

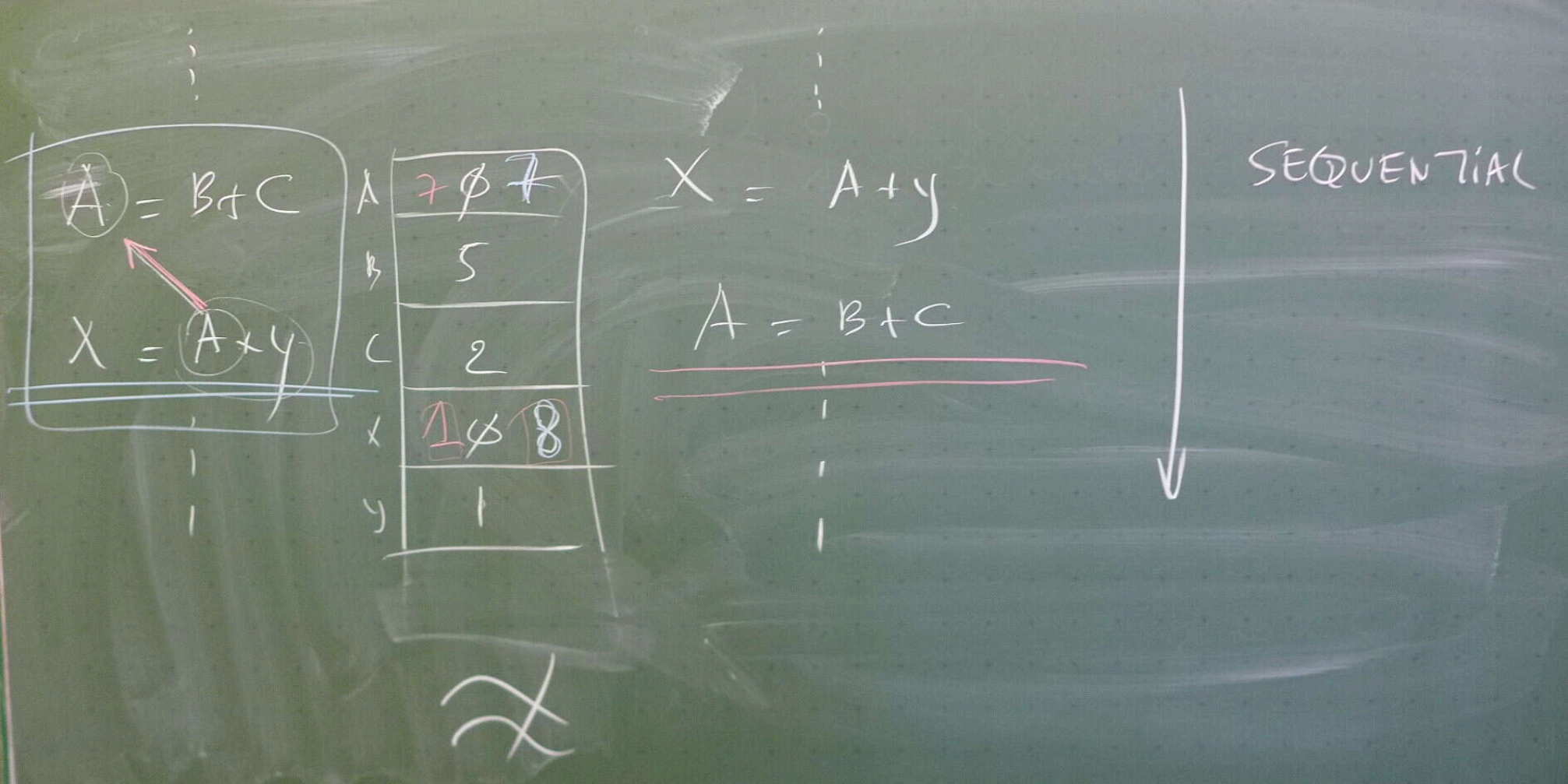

PROGRAM (HLL)
SUM = 0
FOR CTR IN 0..5:
SUM = SUM + CTR
(IV-1)
SUM += CTR
=
10x
#LOC = 4
WHILE TRUE
PASS
#LOC = 2
SUM = 0
CTR = 0
SUM = SUM + CTR
CTR = 1
SUM = SUM + CTR
CTR = 2
# INSTRUCTIONS
2 x 10 + 1 = 21
2xN + 1 ---
∞
TYDSDUUR
Δ(2xN+1)
∞

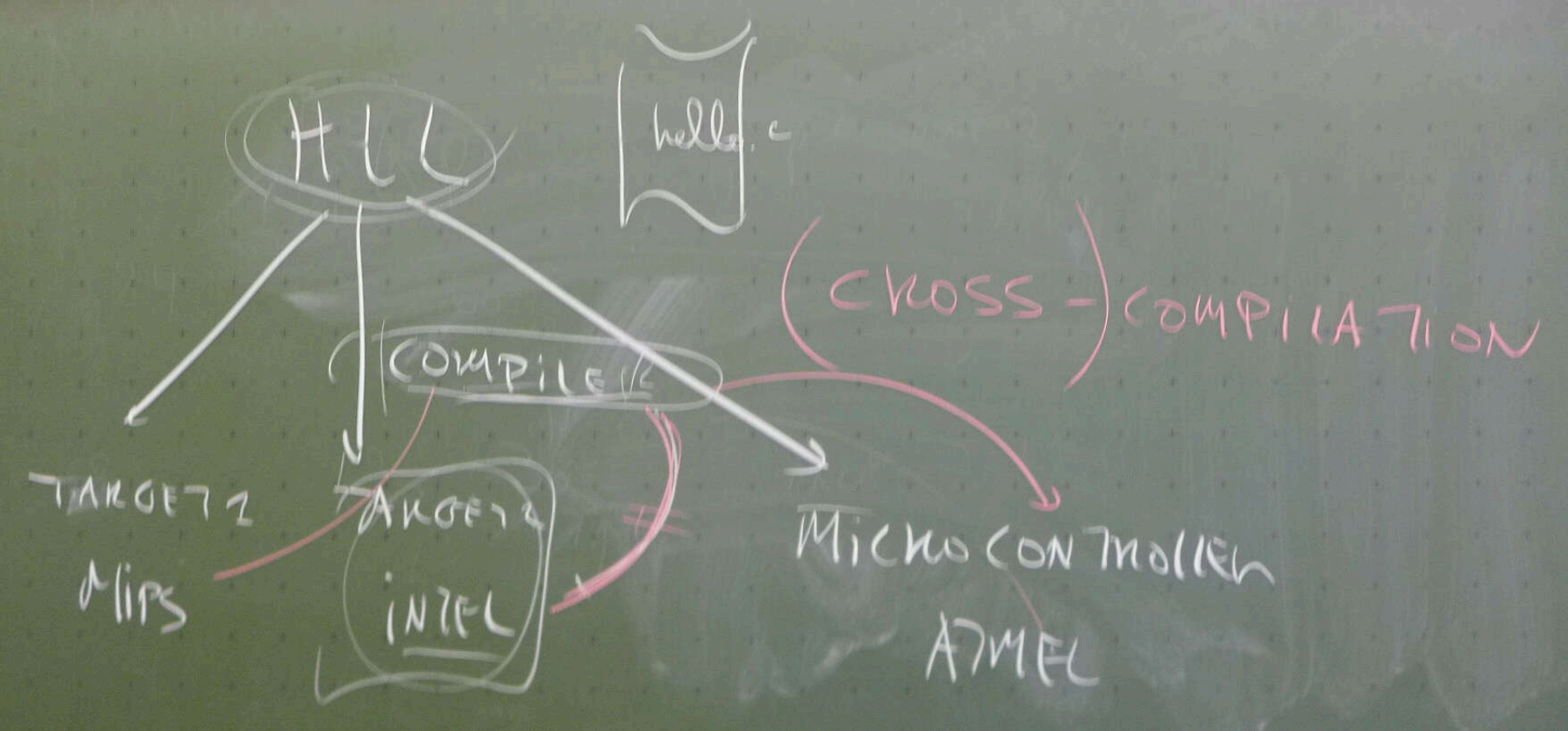
HLL

```
A = B+C;
X = y+z;
```




```
X = y+z;
A = B+C;
```


A = B + C
X = A + y
A
B
C
x
y
7 ϕ 7
5
2
1 ϕ 8
1
X = A + y
A = B + C
SEQUENTIAL


HLL
hello.c
COMPILER
(CROSS-)COMPILATION
TARGET 1
MIPS
TARGET 2
INTEL
MICROCONTROLLER
ATMEL

C++

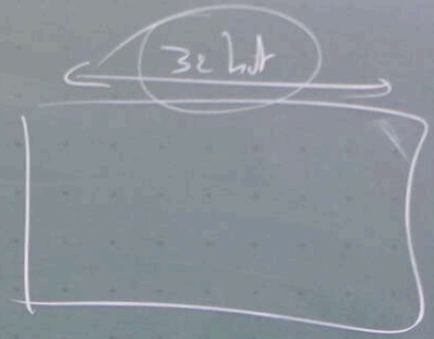
int i;

STRONGLY TYPED

STATIC

i = 10;

i = "ABCDEF";



STRING STR;

STR = "ABC";

STR = STR + "XYZ";

+ : STRING x STRING → STRING

STRING STRING

i = 10 * "ABC"

INT STRING

PYTHON

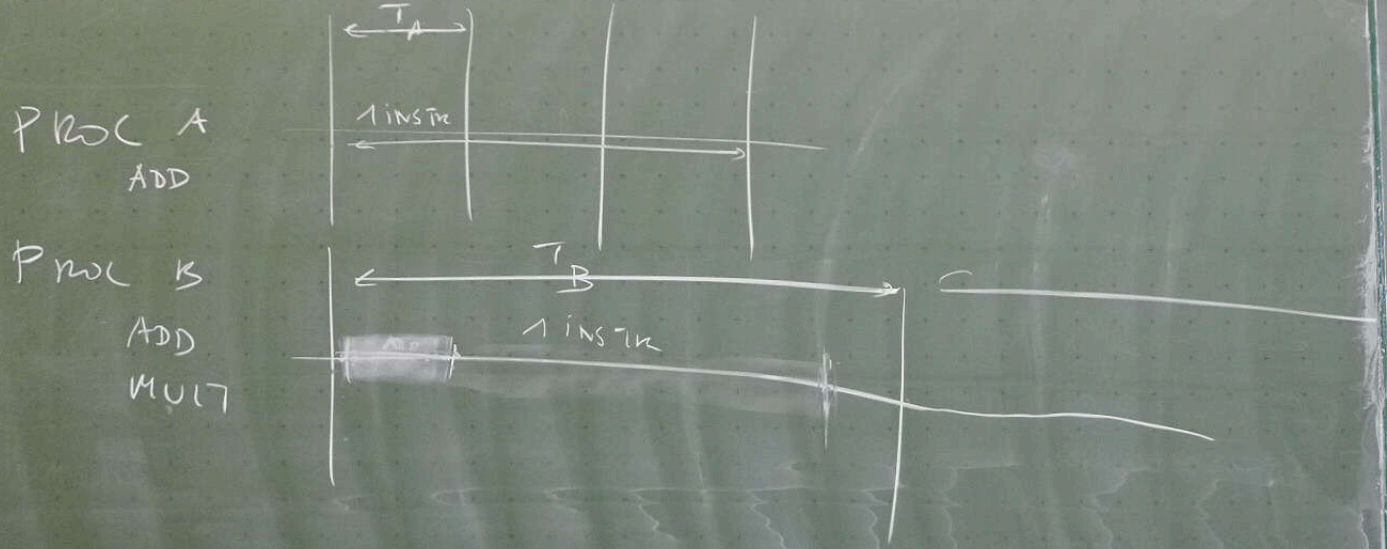
STRONGLY TYPED

DYNAMIC

i = 10

i = "ABC"

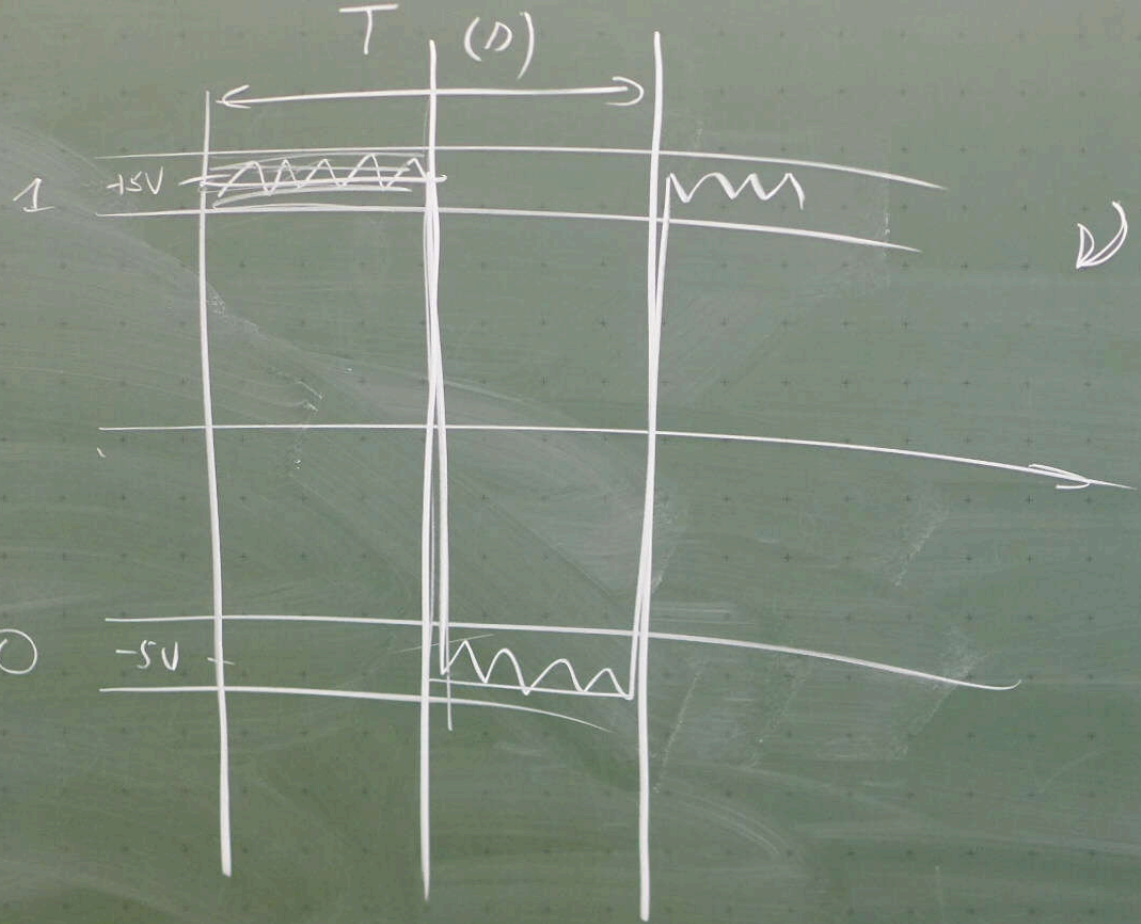
** pyobject



3 ADD INSTR: $3 \times T_A$

3 ADD INSTR: $3 \times T_B$

3 MULT INSTR: $3 \times T_B$



$$D = \frac{1}{T} \left(\frac{1}{s} = Hz\right)$$

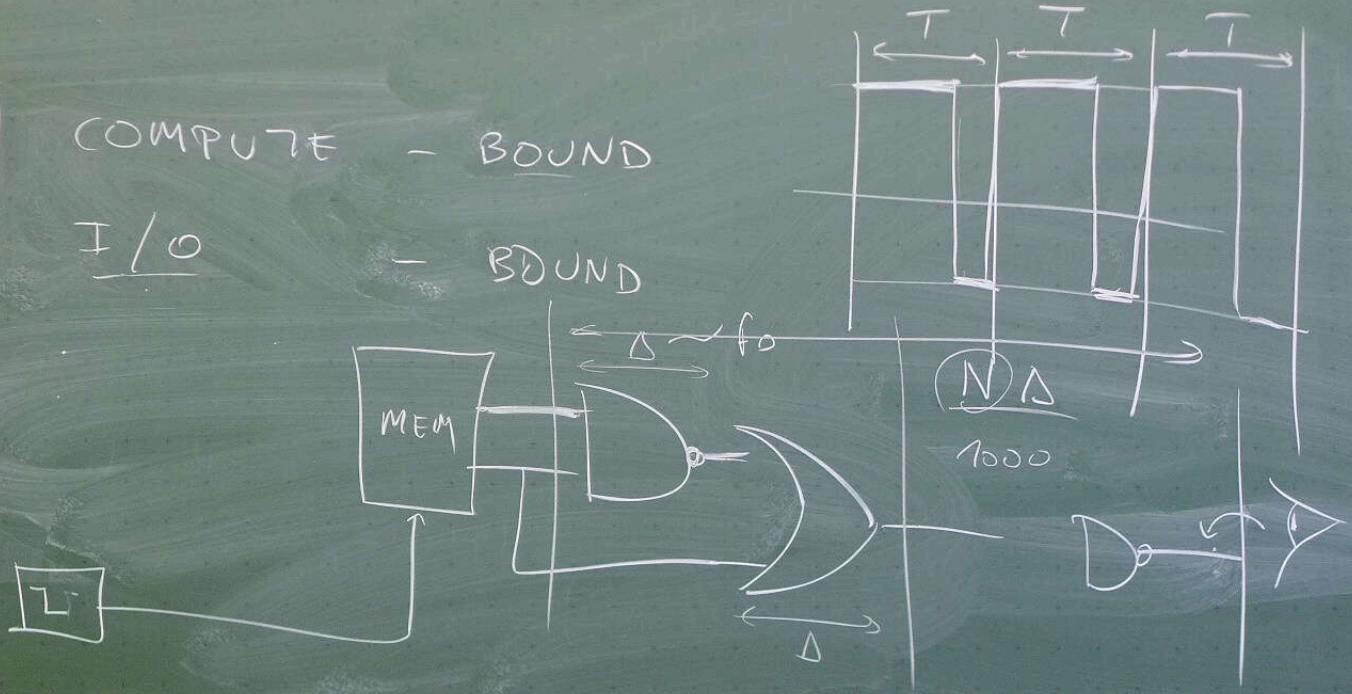
m	-3
μ	-6
n	-9
p	-12
f	-15

$$\nu = \frac{1}{T} \left(\frac{1}{3} = H_2\right)$$

COMPUTE - BOUND

I/O - BOUND

m -3
μ -6
n -9
p -12
f -15



$$\text{CPU TIME}_A = \frac{\text{INSTN COUNT}_A \times \text{CPI}_A}{R_A} \times T_A \qquad \text{CPI}_A = 2, \quad 250\text{ps}$$

$$\text{CPU TIME}_B = \frac{\text{INSTN COUNT}_B \times \text{CPI}_B}{R_B} \times T_B \qquad \text{CPI}_B = 1.2, \quad 500\text{ps}$$

$$\frac{\text{CPU TIME}_A}{\text{CPU TIME}_B} = \frac{2 \times 250\text{ps}}{1.2 \times 500\text{ps}} = \frac{1}{1.2}$$

$$10\,s = \text{CPU Time}_A = \frac{\#\text{CPU}_A\ \text{clock cycles}}{\text{clock rate CPU A}} \qquad 2\times10^9$$

$$6\,s = \text{CPU Time}_B = \frac{\#\text{CPU}_B\ \text{clock cycles}}{\text{clock rate CPU}_B} = \frac{1.2 \times \#\text{CPU}_A\ \text{clock cycles}}{2}$$

$$\frac{10}{6} = \frac{\#\text{cycles}_A \times \text{clock rate CPU}_B}{2\times10^9 \times \#\text{cycles}_A \times 1.2}$$

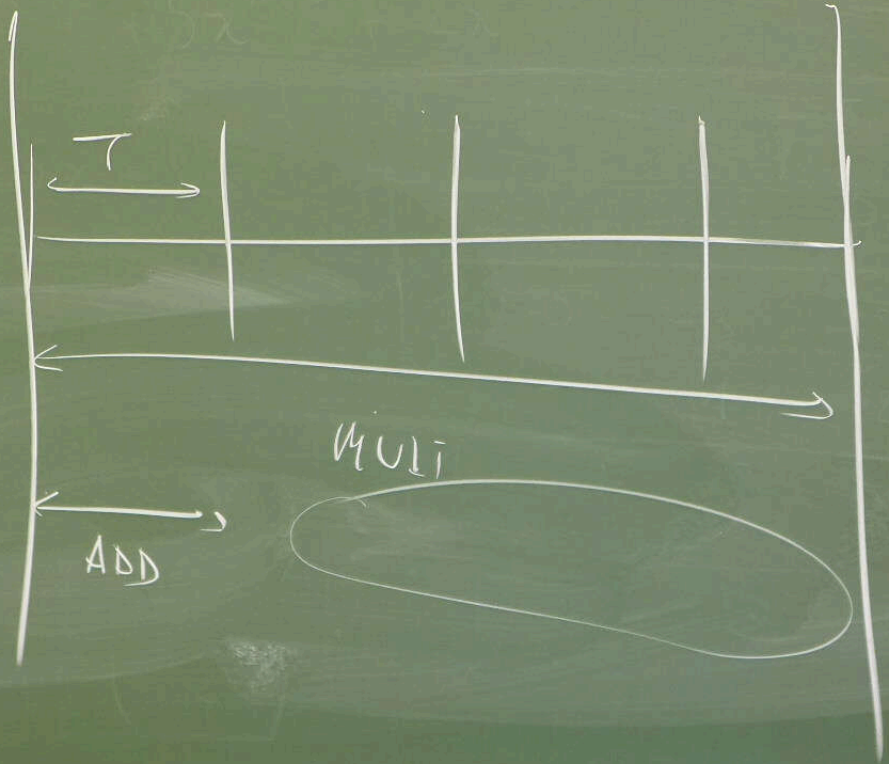
$$\text{clock rate CPU}_B = \frac{10 \times 2\times10^9 \times 1.2}{6} = 4\times10^9\ \text{Hz} = 4\ \text{GHz}$$

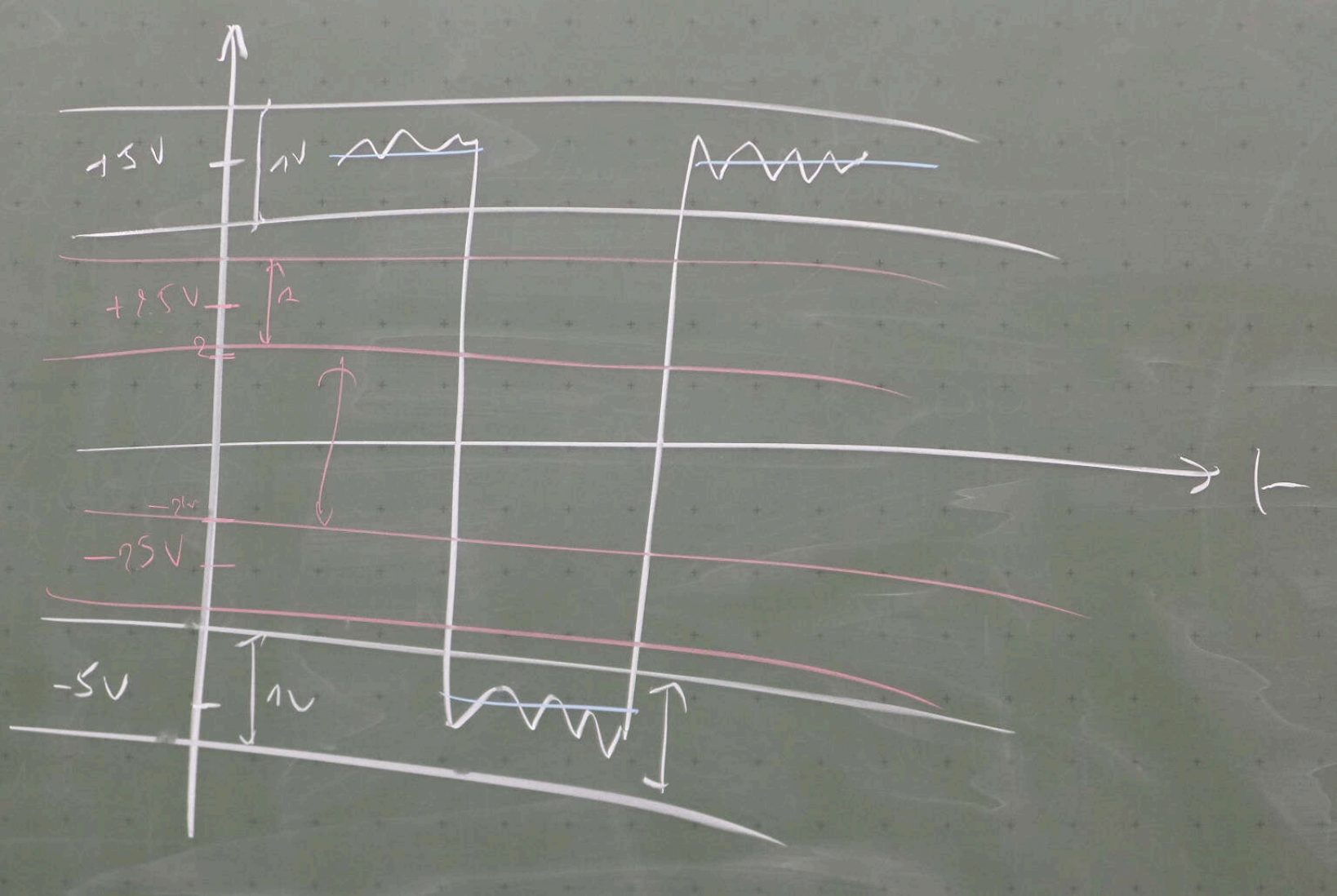
$$F = m\,a$$

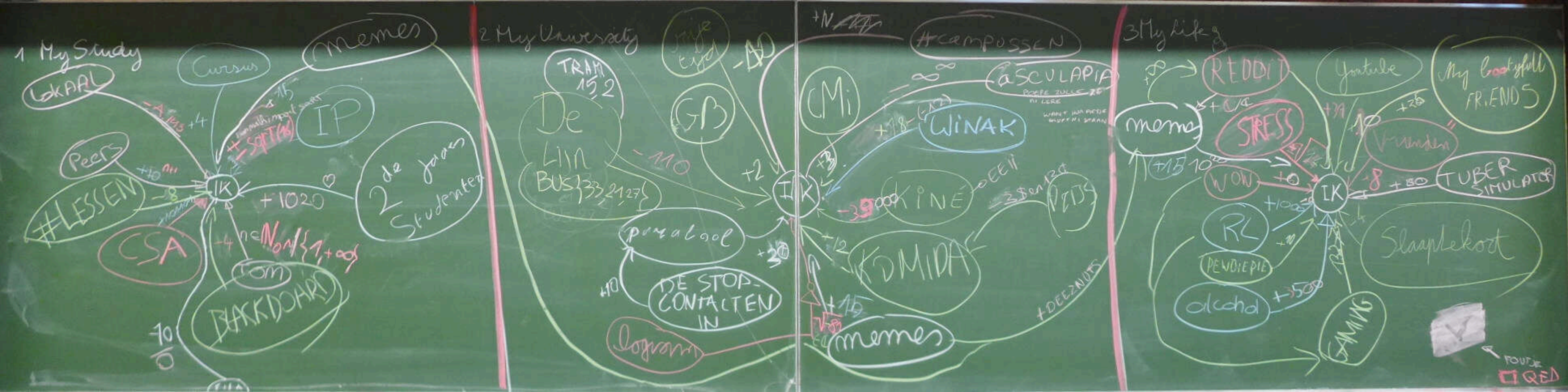
N kg $\frac{m}{s^2}$

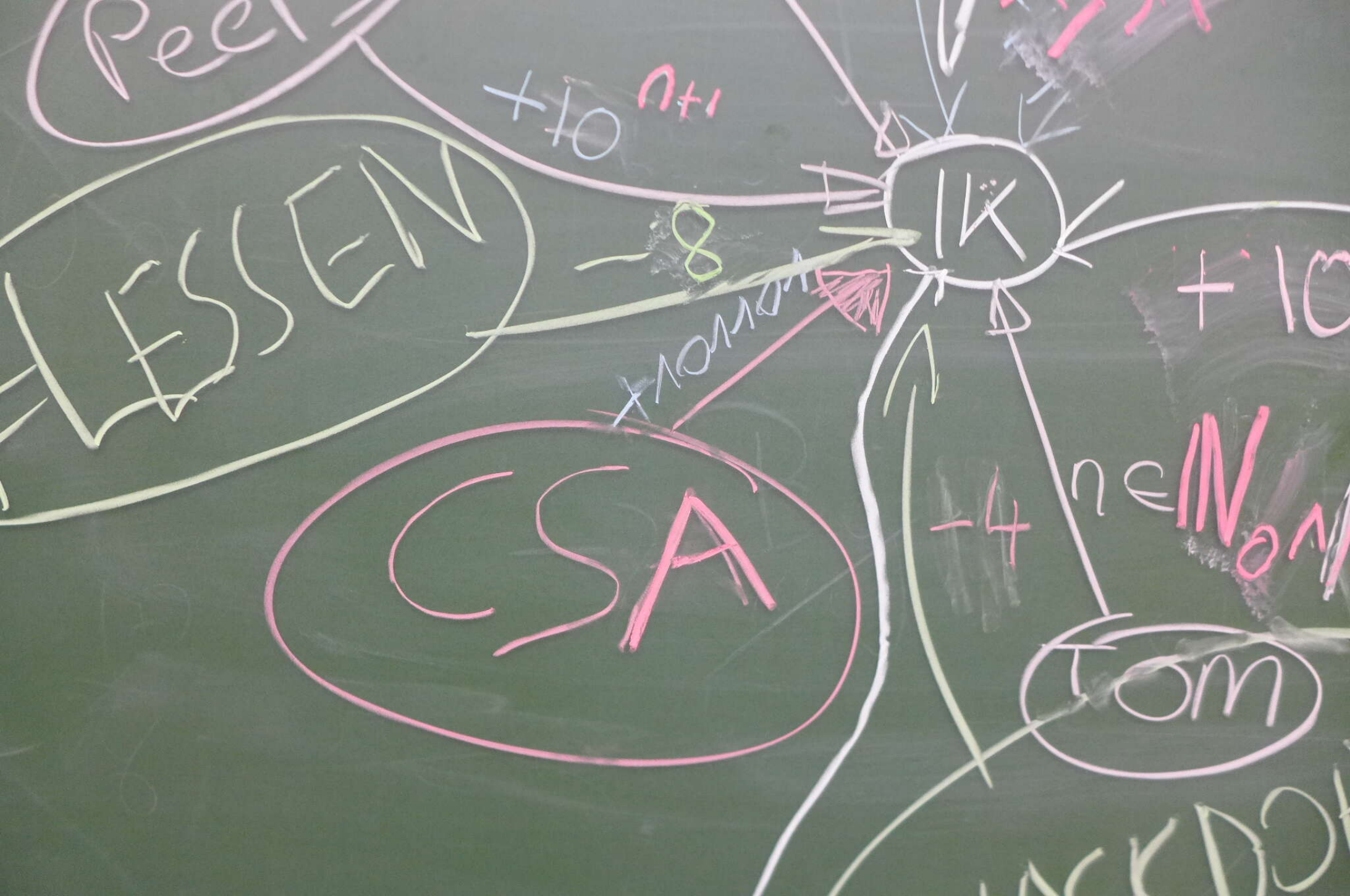
ADD — 1 clock cycles

MULT — 4 clock cycles



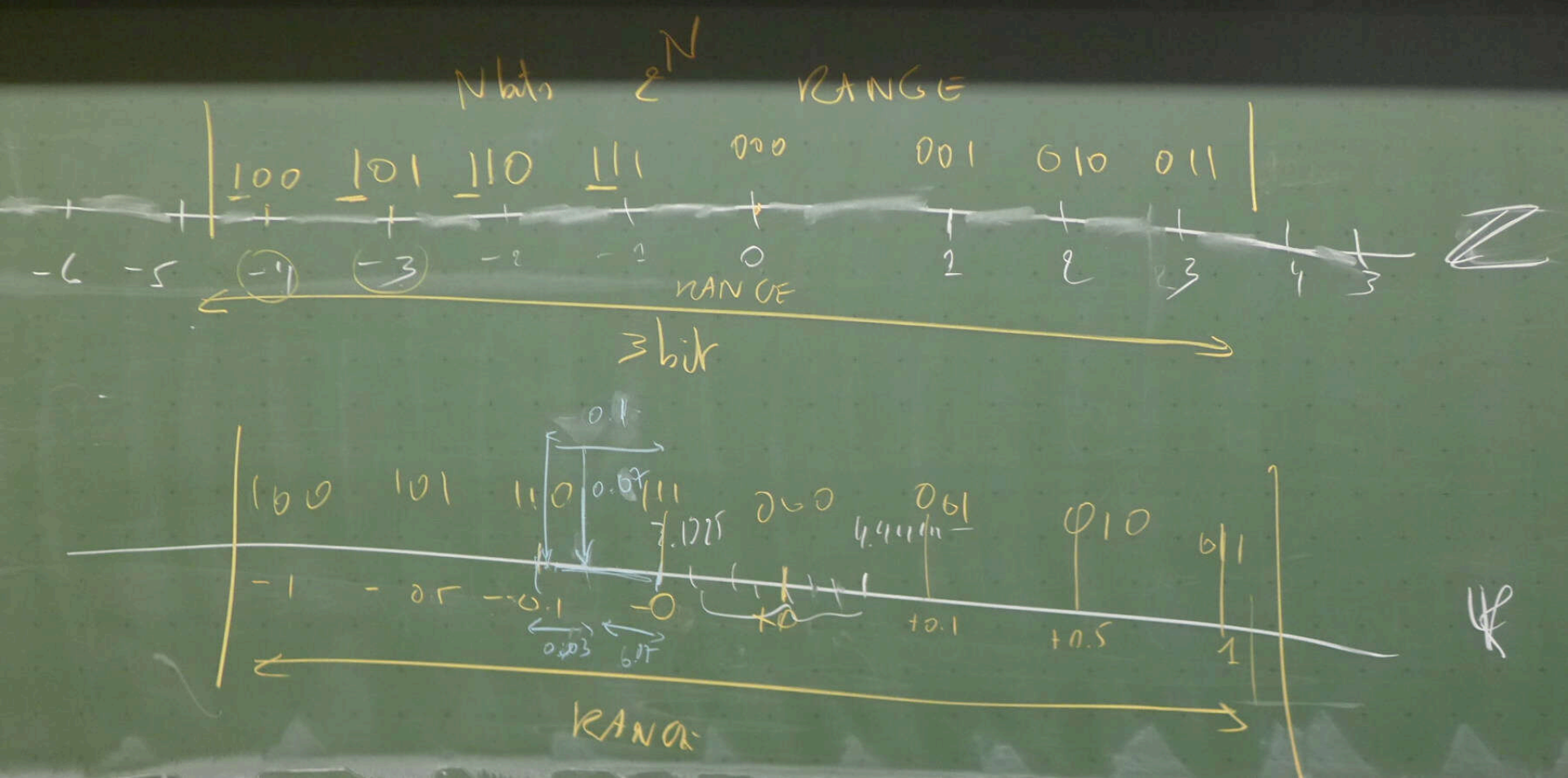

+15V
1V
+2.5V
2
-2V
-25V
-5V
1V


1 My Study
LOKAAL
Cursus
memes
IP
Peers
IK
#LESSEN
CSA
Tom
BLACKBOARD
2de jaars Studenten
+1020
2 My University
TRAM 15,2
De Lijn
BUS{33,21,27}
GB
paradool
DE STOPCONTACTEN IN
Logrom
CMi
#campusSCN
aSCULAPIA
WINAK
KINE
KOMIDA
memes
+DEEZNUTS
3 My life
REDDIT
Youtube
My beautiful FRIENDS
meme
STRESS
Vrienden
WOW
TUBER SIMULATOR
RL
PEWDIEPIE
alcohol
Slaaptekort
GAMING
QED


Peet
+10^(n+1)
ESSEN
-8
IK
CSA
-4
n∈IN
Tom

0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
⋮	⋮	⋮
1	1	1

```
 1  1  1  0
    1  1  1
 +  0  0  1
-----------
    0  0  0
```


Nbits
2^N
RANGE
100 101 110 111 000 001 010 011
-4 -3 0 1 2 3 4
RANGE
3bit
0.1
100 101 110 111 000 001 010 011
-1 -0.1 0 +0.1 +0.5 1
RANGE

PRECISION ?

RANGE ?

$\mathbb{N} = \{ 0, 1, 2, \dots \}$

int $\mathbb{Z} = \{ \dots -2, -1, 0, 1, 2 \dots \}$

fixed point $\mathbb{Q} = \{ \dots 3/4, \dots \}$

float $\mathbb{R} = \{ \dots \}$

-4 ... +3

RANGE

$$1 \times 2^{n-1} + 1 \times 2^{n-2} + \cdots + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = S + 1$$

(n)

$$1 \times 2^n + 0 \cdots \cancel{\quad} \cdots 0 \times 2^1 + 0 \times 2^0 = S + 1$$

$$2^n - 1$$

$[\![\underbrace{1\ 1\ 1\ 1\ 1\ 1\ 1}_{7}]\!]_{BIN} = 127$

$[\![0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 1]\!]_{BCD} = 127$

12

$1 \times 8^{n-1} + 1 \times 8^{n-2} + \cdots + 1 \times 2^{2} + 1$

$= 1 \times 2^{1} + 0 \times 2^{0} +$

$0 \times 2^{-1} + 1 \times 2^{-2}$

$= 2 + 0.25$

$= 2.25$

10

$[\![\text{"}10.01\text{"}]\!] = 10.01$

15.6

BASE 10 ENCODING

$$10.01_2 \times 100_2 = 1001._2$$

$$2.25_{10} \qquad 4_{10}$$

$$10.26_{10} \times 100_{10} = 1026$$

$\times 10^k$ • k digits →

$/ 10^k$ • k digits ←

BASIS b

$$[\![\ "x_{n-1}\ x_{n-2}\ \cdots\ x_1\ x_0" \ ]\!]_b = \sum_{i=0}^{n-1} x_i \times b^i$$

$x_i \in [\emptyset, b-1]$

BASE b ENCODING

RADIX

$[\![\text{"} 10.26 \text{"}]\!] = \sum_{i=-2}^{1} x_i \times 10^i$

1 0 . 2 6

1 0 -1 -2

BASE 10 ENCODING

$= 1 \times 10^1 + 0 \times 10^0 + 2 \times 10^{-1} + 6 \times 10^{-2}$

$[\![\text{"10.01"}]\!] = \sum_{i=-2}^{1} x_i \times 2^i$

BASE 2 ENCODING

$[\![\text{"10.26"}]\!] = \sum_{i=-2}^{1} x_i \times 10^i$

BASE 10 ENCODING

$= 1 \times 10^1 + 0 \times 10^0 + 2 \times 10^{-1} + 6 \times 10^{-2}$

$[\![\text{"10.01"}]\!] = \sum_{i=-2}^{1} x_i \times 2^i$

BASE 2 ENCODING

$= 1 \times 2^1 + 0 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2}$

$= 2 + 0.25$

$= 2.25$

10

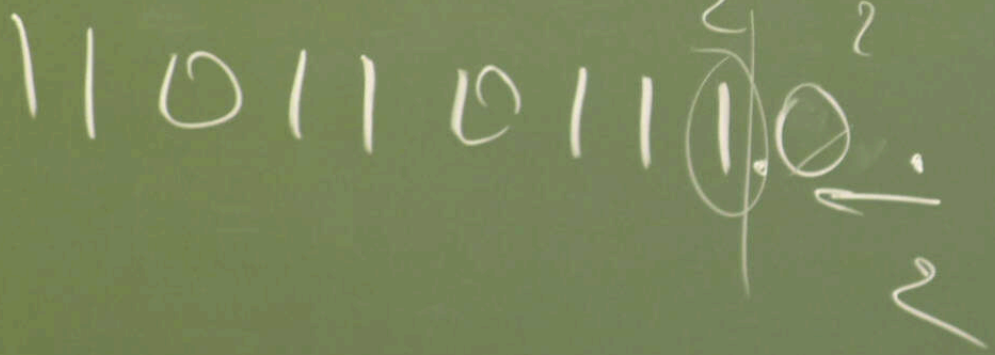
$[\![\text{"10.01"}]\!] = 10.01$

15.10

BASE 10 ENCODING

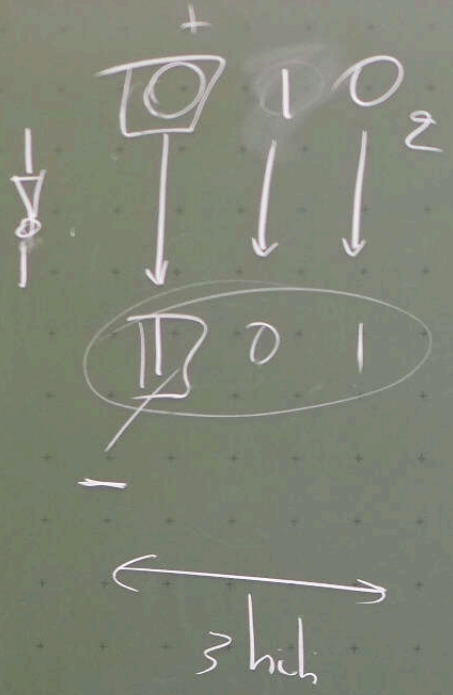
DEADBEEF HEX

1101 1110 1010 1011 1110 1110 1111 BINARY



MEMOIZATION

1	x 3	3
2	x 3	6
4	x 3	12
5	x 3	15
	1	



UNSIGNED 2_{10}

SIGNED 1's comp

~~SIGNED~~ -2_{10}

1's comp

+ Ø

2 1's comp 1 0

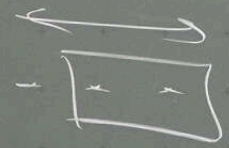
2 1's comp - Ø 1 0

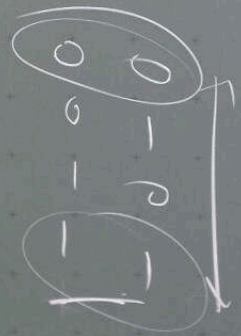
8'S COMPLEMENT

N bits

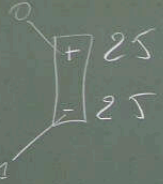
RANGE = $[\ -2^{N-1}\ ,\ 2^{N-1}-1\ ]$

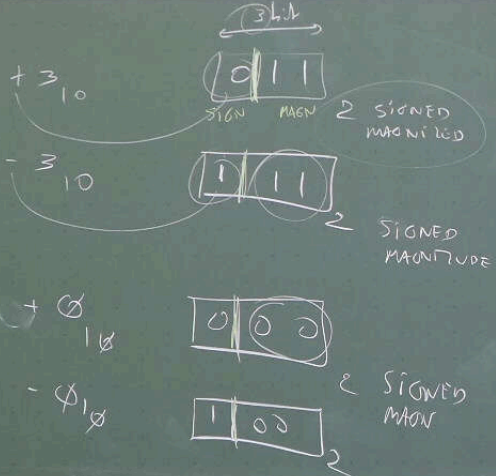
$000_2 = 0_{10}$

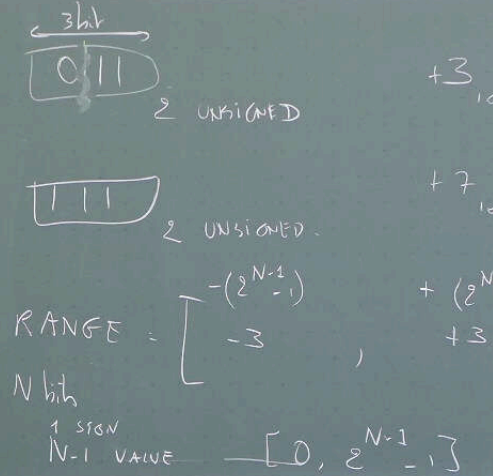


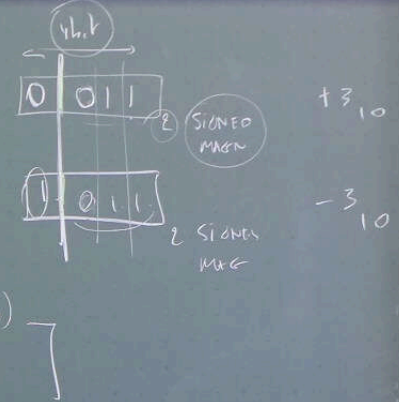


$2^{N-1}-1$


0
+ 25
- 25
1


3 bit
+ 3 10
0 1 1
SIGN
MAGN
2 SIGNED MAGNITUDE
- 3 10
1 1 1
2 SIGNED MAGNITUDE
+ 0 10
0 0 0
- 0 10
1 0 0
2 SIGNED MAGN


3 bit
0 1 1
2 UNSIGNED
1 1 1
2 UNSIGNED
RANGE = [-(2^(N-1) - 1) , + (2^(N-1) - 1)]
-3
+3
N bit
1 SIGN
N-1 VALUE
[0, 2^(N-1) - 1]


4 bit
+ 3 10
0 0 1 1
2 SIGNED MAGN
+ 3 10
+ 7 10
1 0 1 1
2 SIGNED MAG
- 3 10

$A < B$?

$A - B < 0$

$A + \overline{B} + 1$

$A - B = A + (-B)$

$+ \quad 3.4 \times 10^{2}$

5.16×10^{3}

$< ?$

3.4×10^{2}

51.6×10^{1}

10^{2}

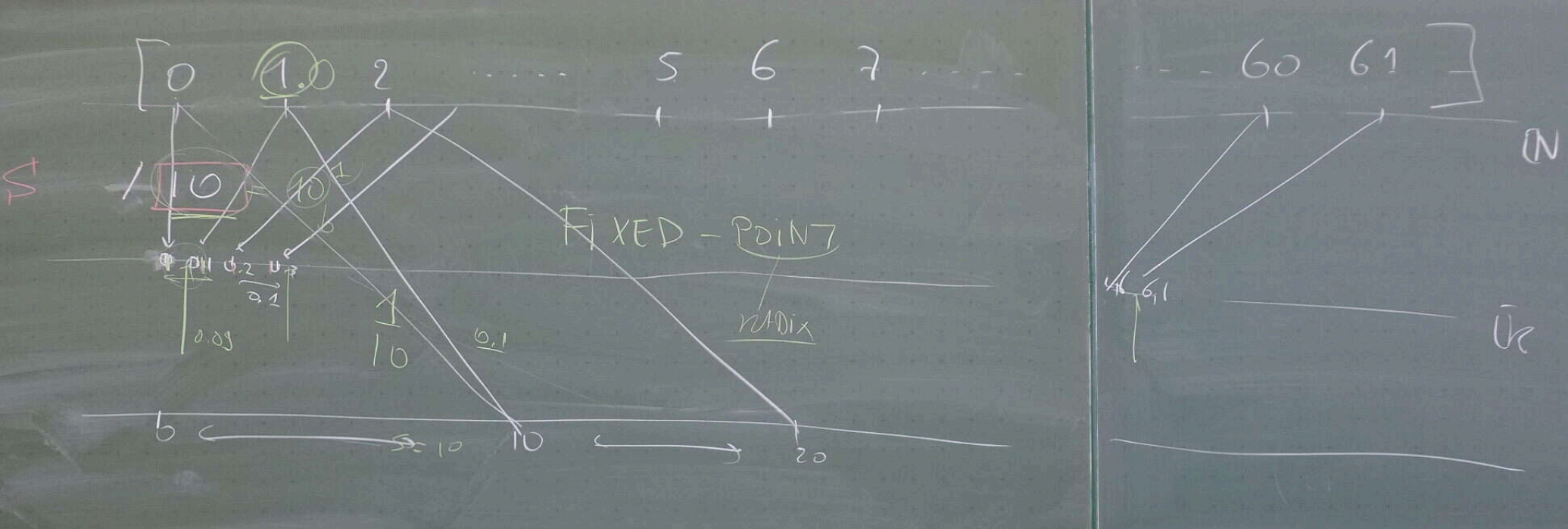
55.0×10^{2}

$+3 \qquad x \quad 0\,0\,1\,1_2$

$-3 \qquad \overline{x} \quad 1\,1\,0\,0$

$\qquad\qquad + \quad 0\,0\,0\,1$

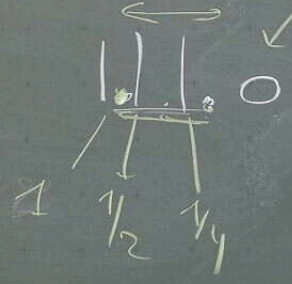
$\qquad -x \quad 1\,1\,0\,1$


0
1.0
2
5
6
7
60
61
10
0.1
0.09
0.2
FIXED - POINT
RADIX
10
20

$[\![111]\!]_{\text{3 bit unsigned int}} = 7$

$[\![111]\!]_{\text{3 bit 2's complement int}} = -1$

$S = 2^2 = 4$



$[\![111]\!] = 7/4$ .10

UNISIGNED FIXED-POINT, 3 bit

$[\![111]\!]_{\text{SIGNED}}^{S=4} = -1/4$

8 bit UNSIGNED INT RANGE = [0 , 255]

$S = 2^5 = 32$

FIXED-POINT (8 bit), ($S = 2^5$), UNSIGNED

RANGE = [$\frac{0}{32}$, $\frac{255}{32}$]

PRECISION / MAX ERROR = $1/32$

FIXED-POINT → bits

$S = 2^f$, SIGNED 2's COMPL

RANGE $= \left[\frac{-2^{n-1}}{2^f}, \frac{2^{n-1}-1}{2^f} \right]$

PRECISION $= \frac{1}{2^f}$

$2^{(n-1)-f}$

AI [0 1 1 0 1] =

BI [0 1 1 1] S =

AI . BI [ ]

$$|\pi - 3.15625| < \frac{1}{25}$$

$$\begin{bmatrix} 0 & 1 \\ b^{n-1} & b^{n-1} - 1/b^{n-1} \end{bmatrix}$$

$$\begin{bmatrix} -b^{n-1}/(b^n) & b^{n-1} - 1/b^n \end{bmatrix}$$

$$\left[0 / b^{\#f},\ b^{\#b} - 1 / b^{\#f}\right]$$ UNSIGNED

RANGE

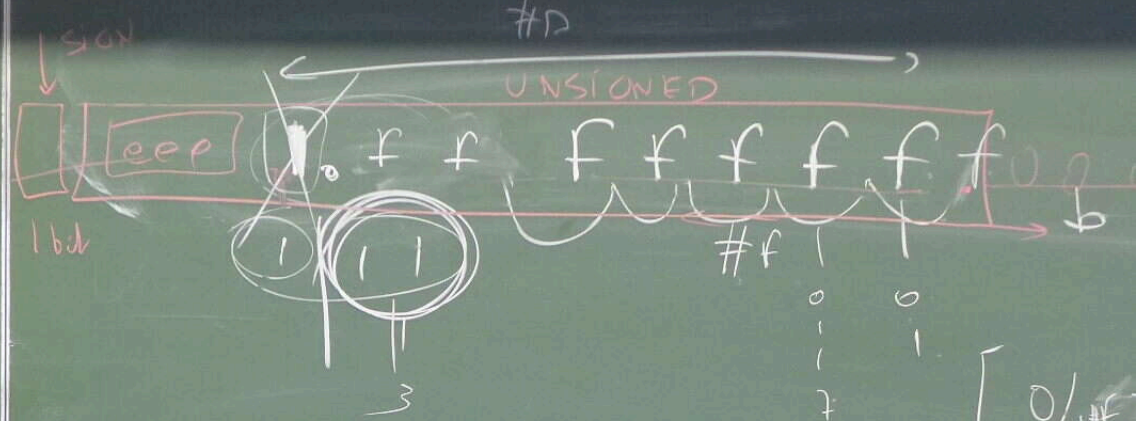
$$\left[-b^{\#b-1} / b^{\#f},\ b^{\#b-1} - 1 / b^{\#f}\right]$$ SIGNED

$$1 / b^{\#f}$$

MAX ERROR

$123000 = 123 \times 10^{3}$

$0.123 = 123 \times 10^{-3}$



$x + y + z$

$=$

$y + x + z$

$=$

$z + y + x$

$=$

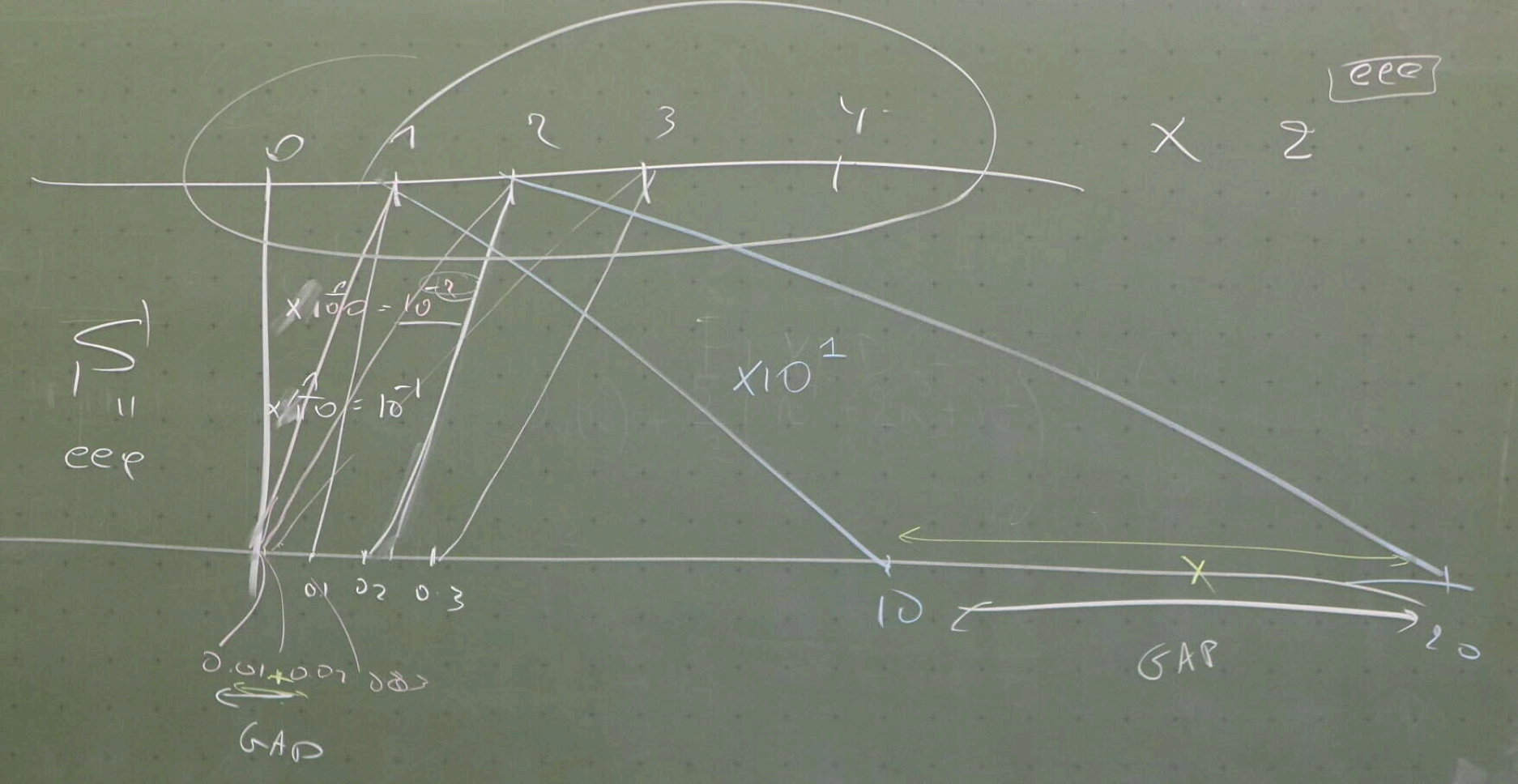
LEXICOGRAFISCH

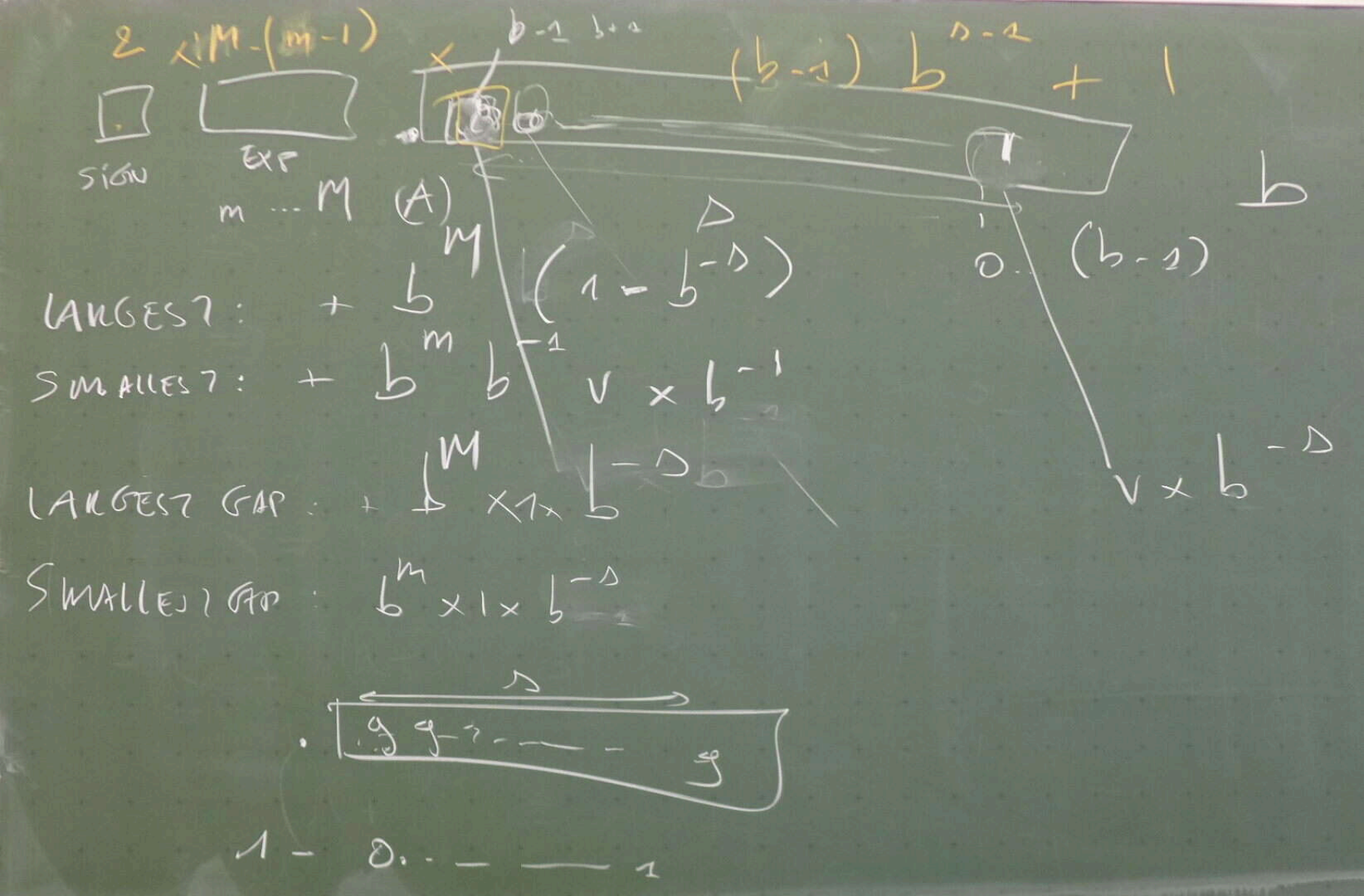
GEORDEND

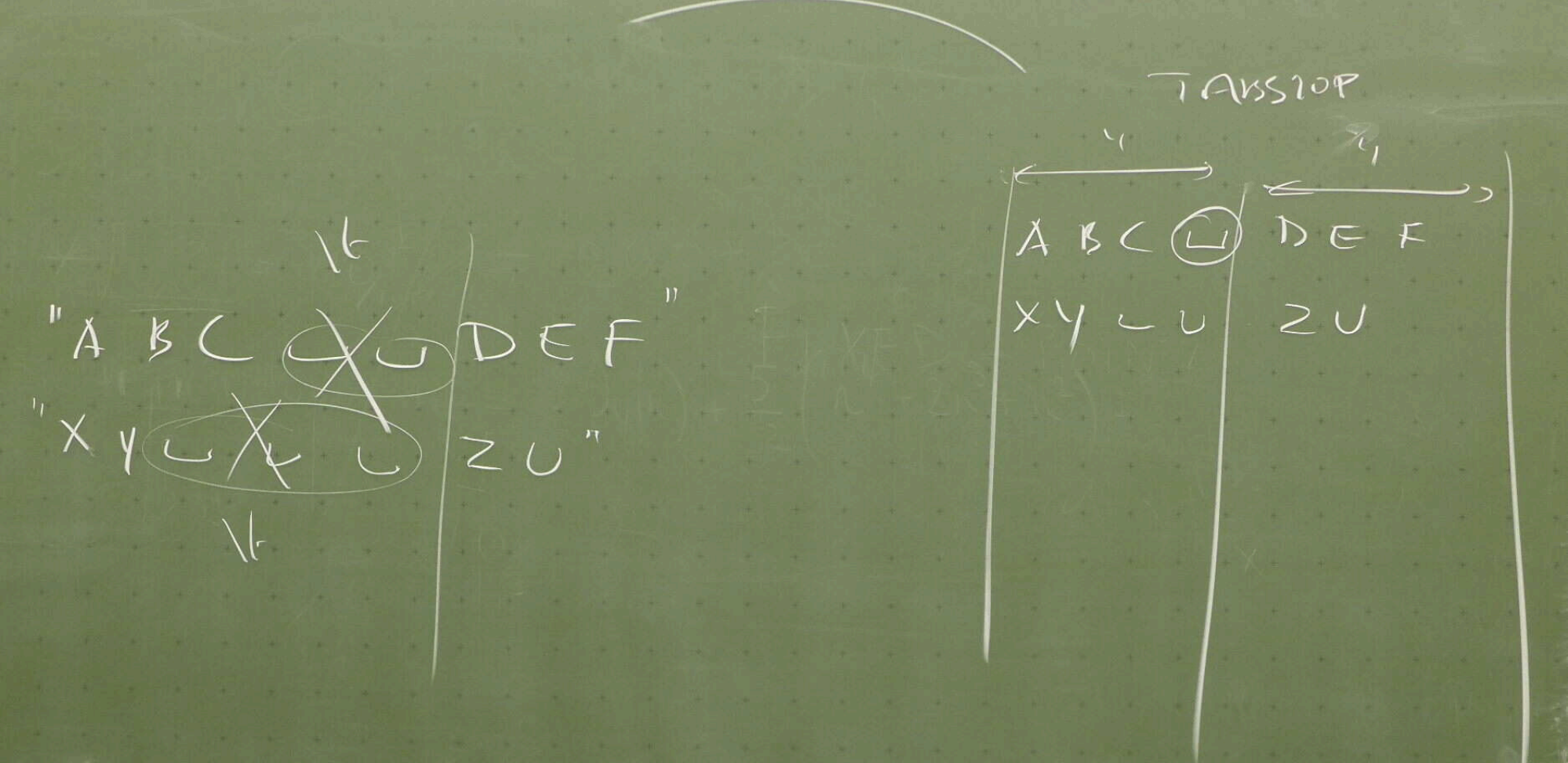
$(x + y + z) - (y + z + x)$

$\downarrow \qquad \downarrow$

$x + y + z \qquad x + y + z$


eee
0
1
2
3
4
X
2
$\times 10^{-2}$
$\times 10^{-1}$
$\times 10^{1}$
eep
0.1
0.2
0.3
10
20
X
GAP
GAP


2 < M - (m - 1)
SIGN
EXP
m ... M
(A)
LARGEST:
SMALLEST:
LARGEST GAP:
SMALLEST GAP:
1 - 0. — — 1


TABSTOP
4
4
A B C ␣ D E F
X Y ␣ ␣ Z U
16
"A B C ␣ ␣ D E F"
"X Y ␣ ␣ ␣ Z U"
16

0.312_{10}

3×10^{-1} 1×10^{-2} 2×10^{-3}

$0.011_3 \times 2^0 =$

$\times 2^{-1}$

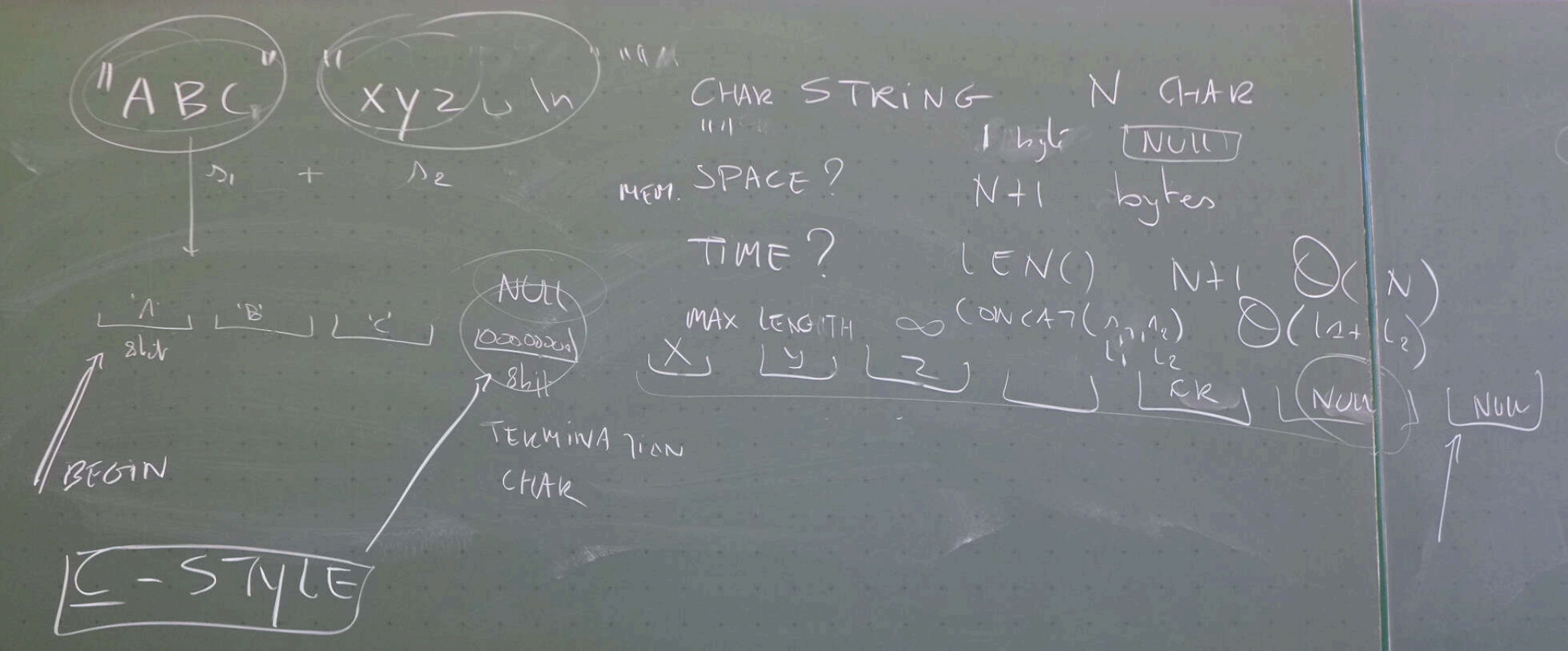
$\times 2^{-2}$

$\times 2^{-3}$

1

$111.0_3 \times 2^{-3}$

$\times 2^1$ $\times 2^0$


"ABC"
"xyz"
"ABCxyz"
s1 + s2
'A' 'B' 'C'
8bit
NULL
00000000
8bit
TERMINATION CHAR
BEGIN
C-STYLE
CHAR STRING
"A"
1 byte
N CHAR
NULL
MEM. SPACE?
N+1 bytes
TIME?
LEN()
N+1
O(N)
MAX LENGTH
∞
CONCAT(s1,s2)
O(l1+l2)
l1 l2
X
Y
Z
ER
NUL
NUL

" " 2 bytes

" " 1 byte

"ABC" + "XYZ␣\n" =

(4) (6)

L1 L2

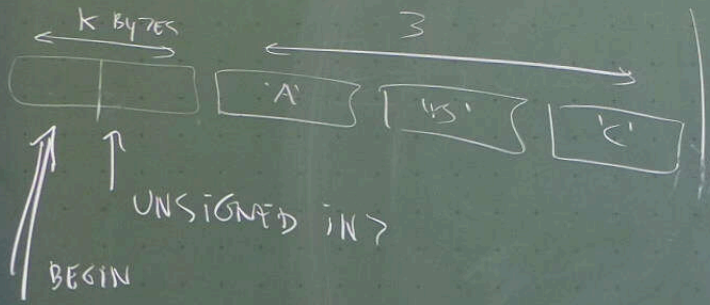
"ABCXYZ␣\n"

(9) L1 + L2

STRING

Java - style

"ABC"



"" 2 bytes

MEM. SPACE? N+2 bytes

TIME? LEN() $\Theta(1)$

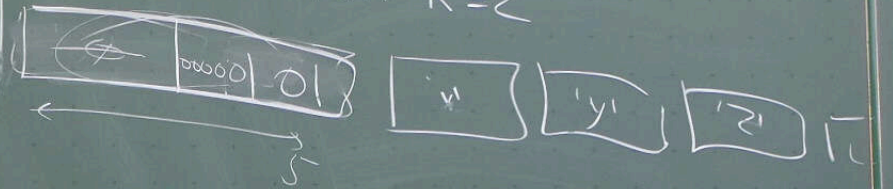
CONCAT(s1, s2) $\Theta(l_2)$

l_2 l_2

MAX LENGTH $2^{8K}-1$ JAVA.

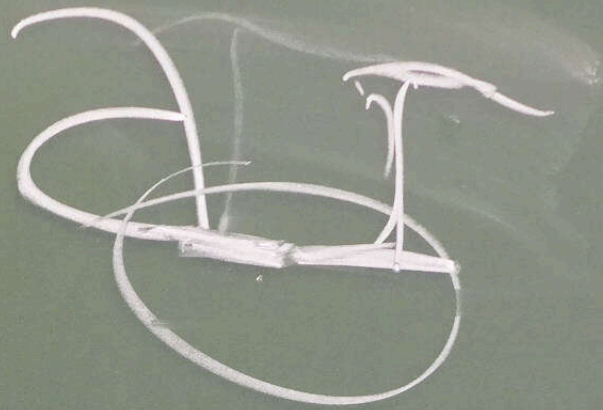
K BYTES = $k \times 8$ bits

JAVA: $K=2$



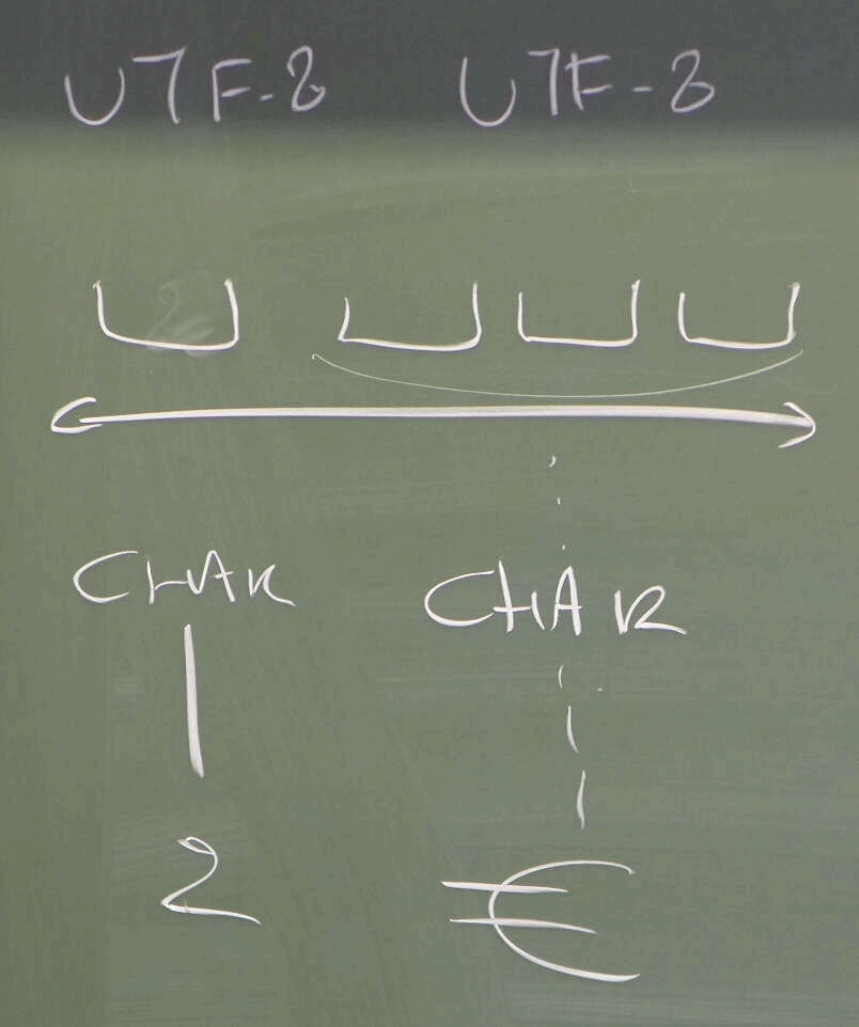
$1024 \times 64 - 1$

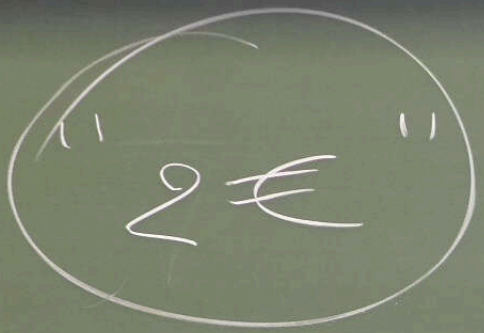
ar



SIANS SENIE

SENIR


UTF-8
UTF-8
CHAR
CHAR
2
€

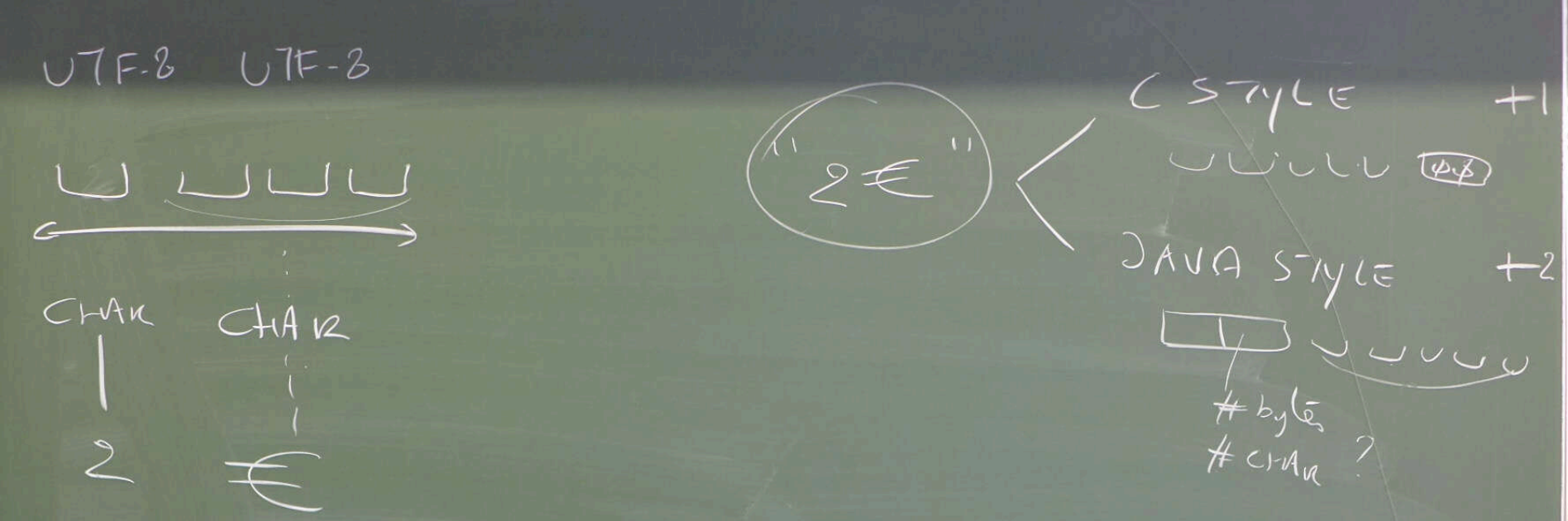


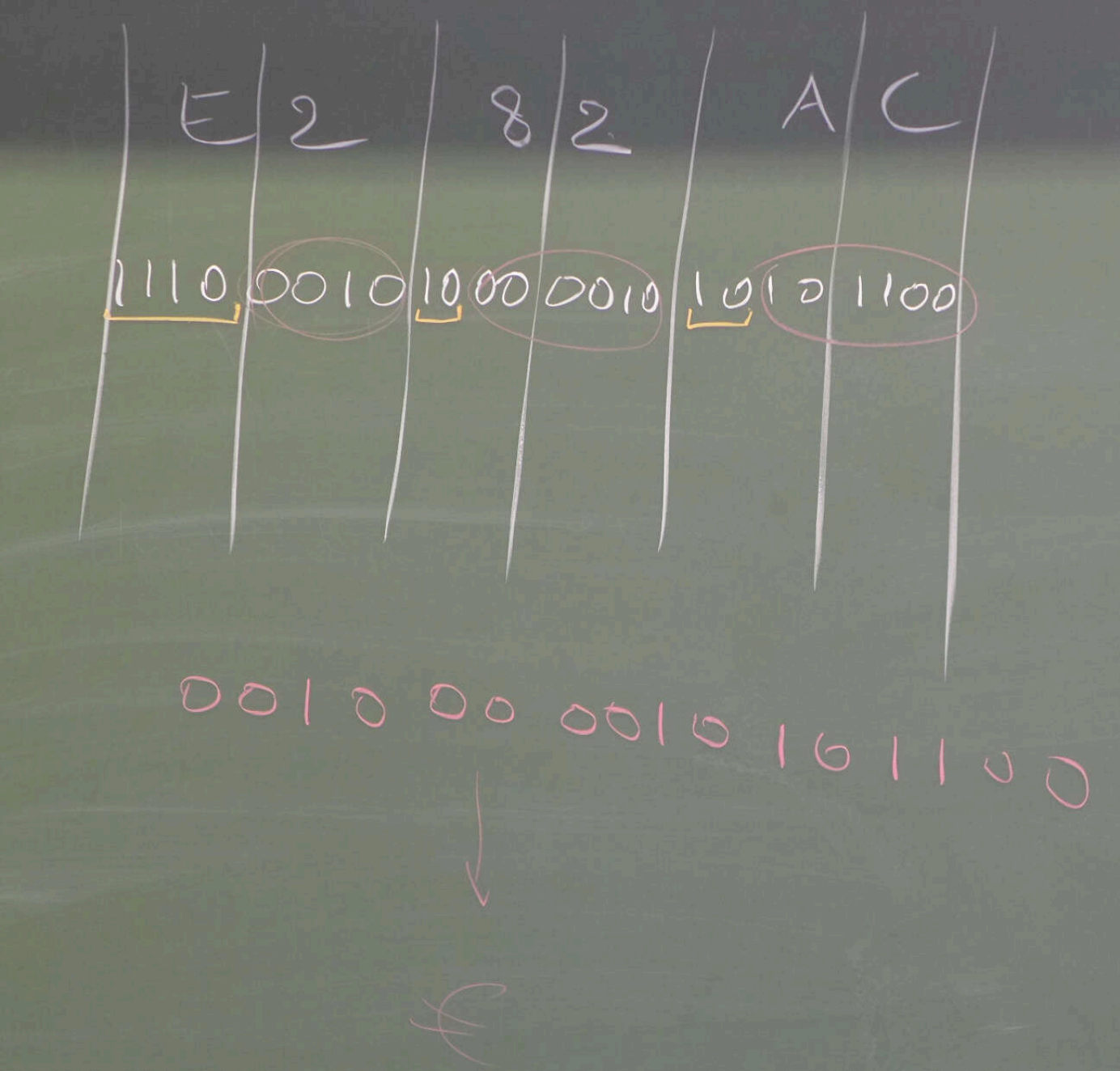
C STYLE +1

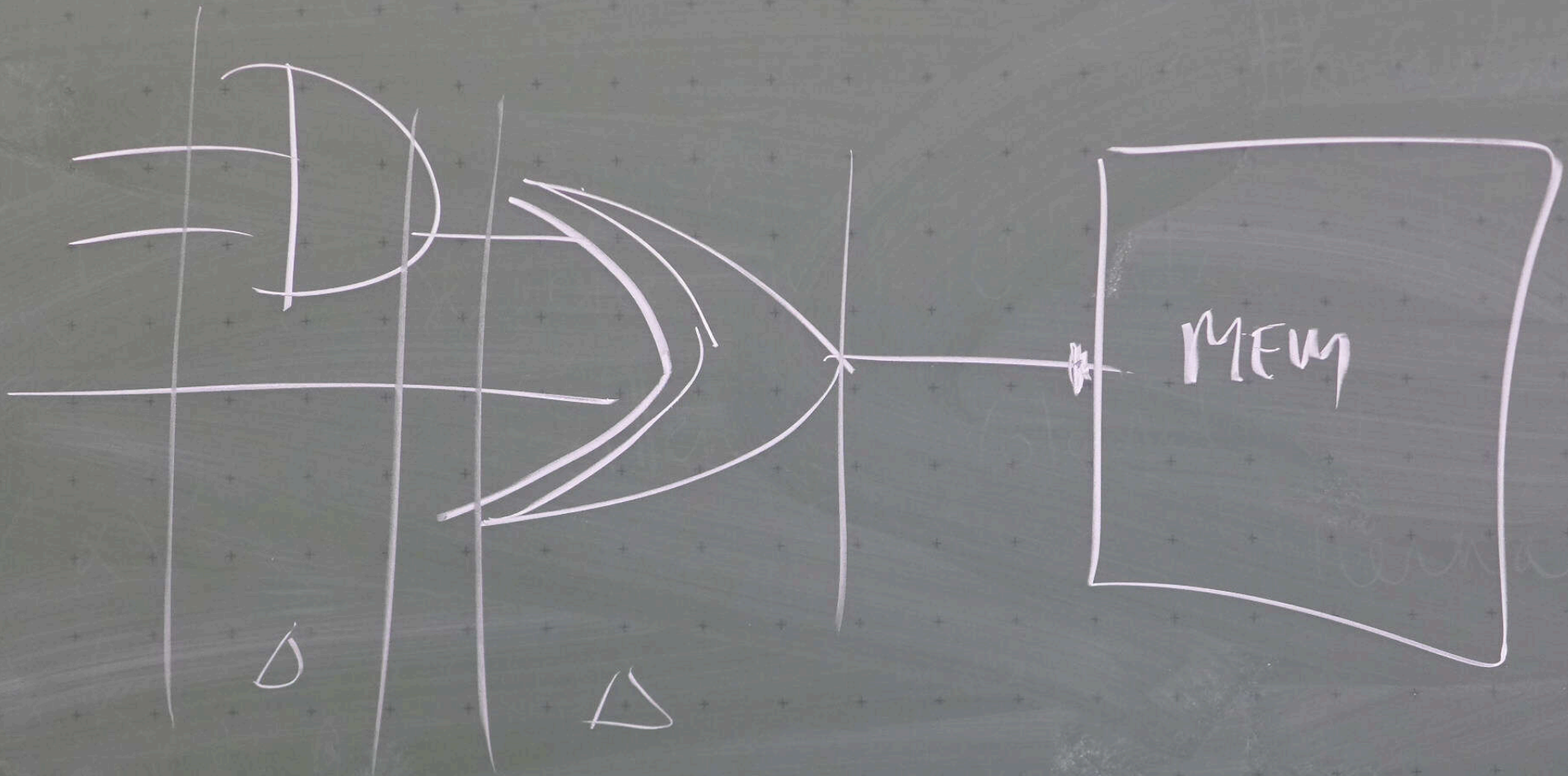
JAVA STYLE +2

bytes

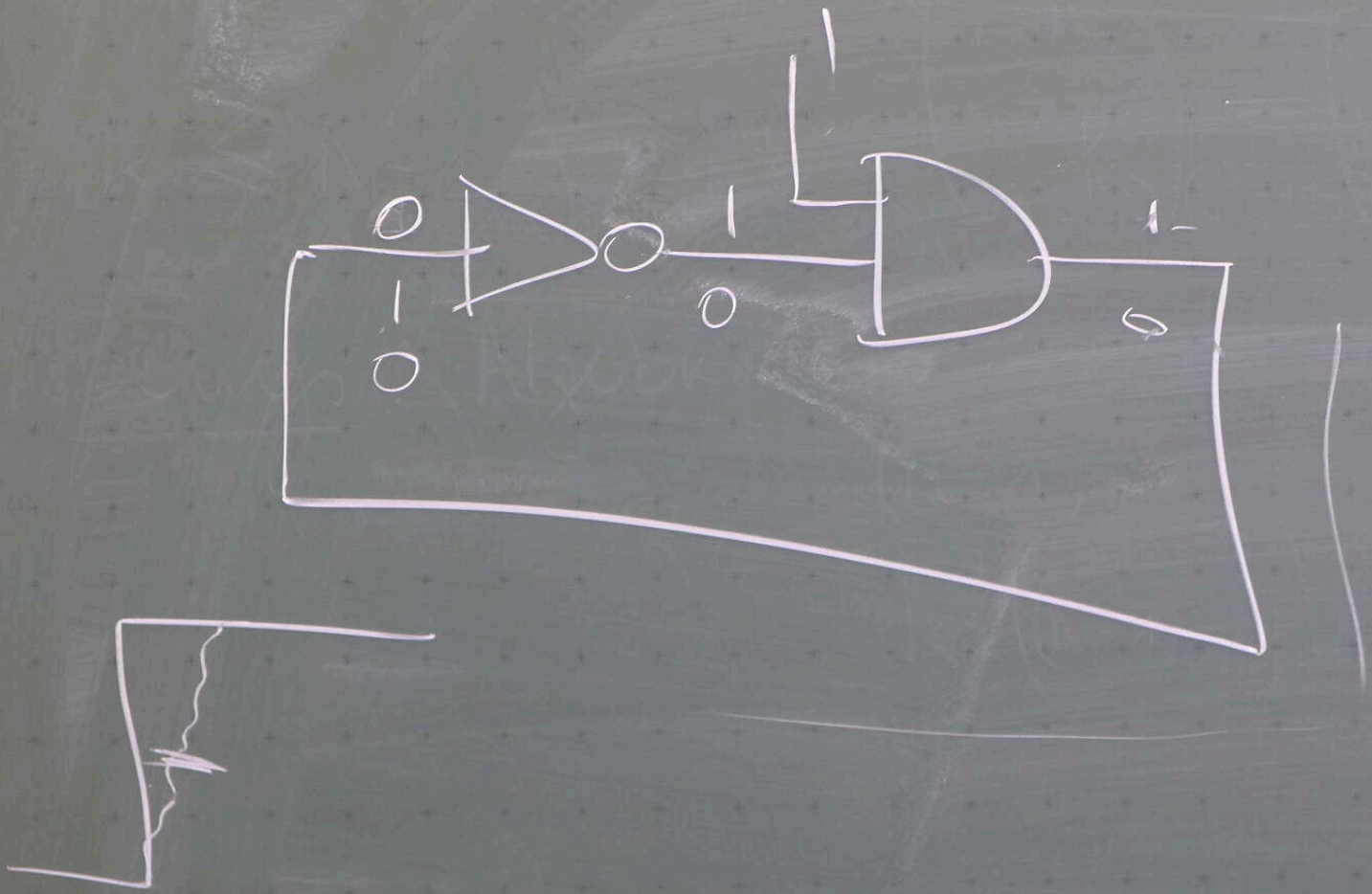
chars ?

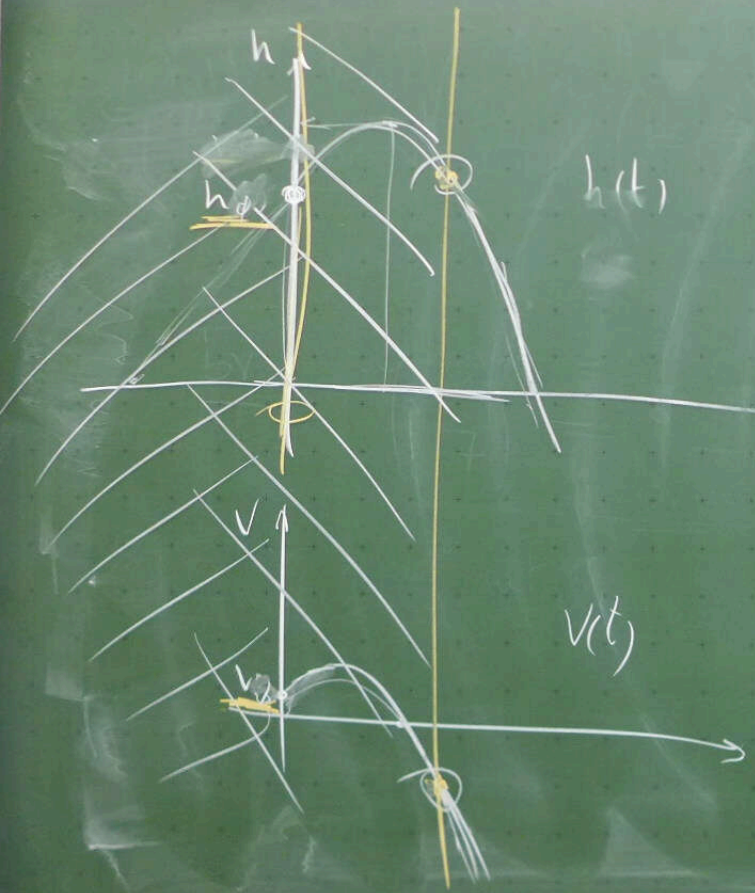

UTF-8 UTF-8
CHAR CHAR
2 €
"2€"
C STYLE +1
JAVA STYLE +2
bytes
CHAR ?


E2 82 AC
1110 0010 1000 0010 1010 1100
0010 0010 0010 1011 00
€


MEM

META-STABLE





$h(t)$

$$\begin{cases} m\dfrac{dv}{dt} = -mg \\ \dfrac{dh}{dt} = v \end{cases}$$

$v(0) = v_0$

$h(0) = h_0$

"STATE"

TOESTAND

STAAT

$h, v : \mathbb{R} \to \mathbb{R}$

~~$\frac{dv}{dt} = a$~~

$v(t)$

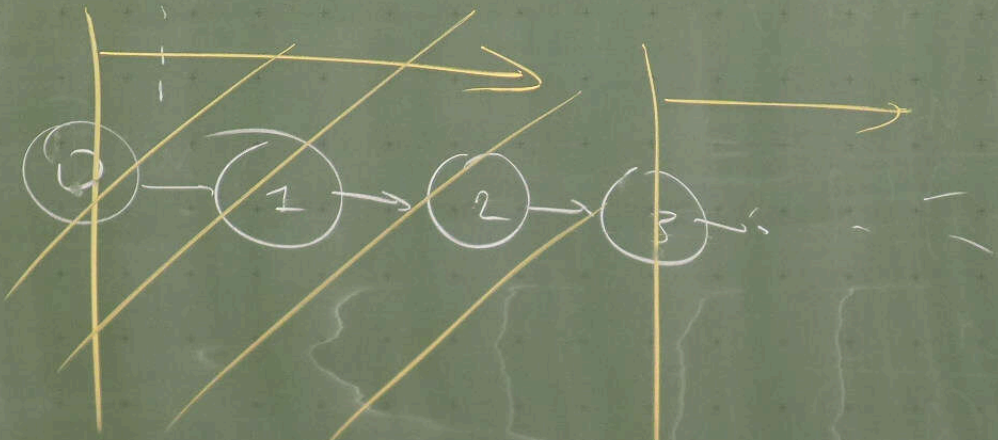
CONTINUOUS

$$\begin{cases} x(i+1) = x(i) + 1, & i \in \mathbb{N} \\ x(0) = x_\phi = \phi \end{cases}$$

$x(1) = 1$

$x(2) = 2$

$x : \mathbb{N} \longrightarrow \mathbb{N}$

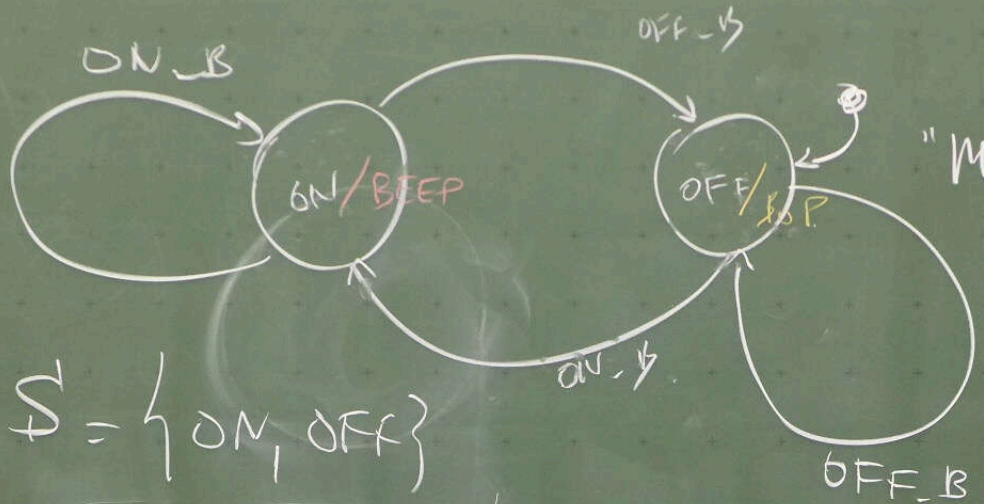


DISCRETE ∞

FSA

DISCRETE

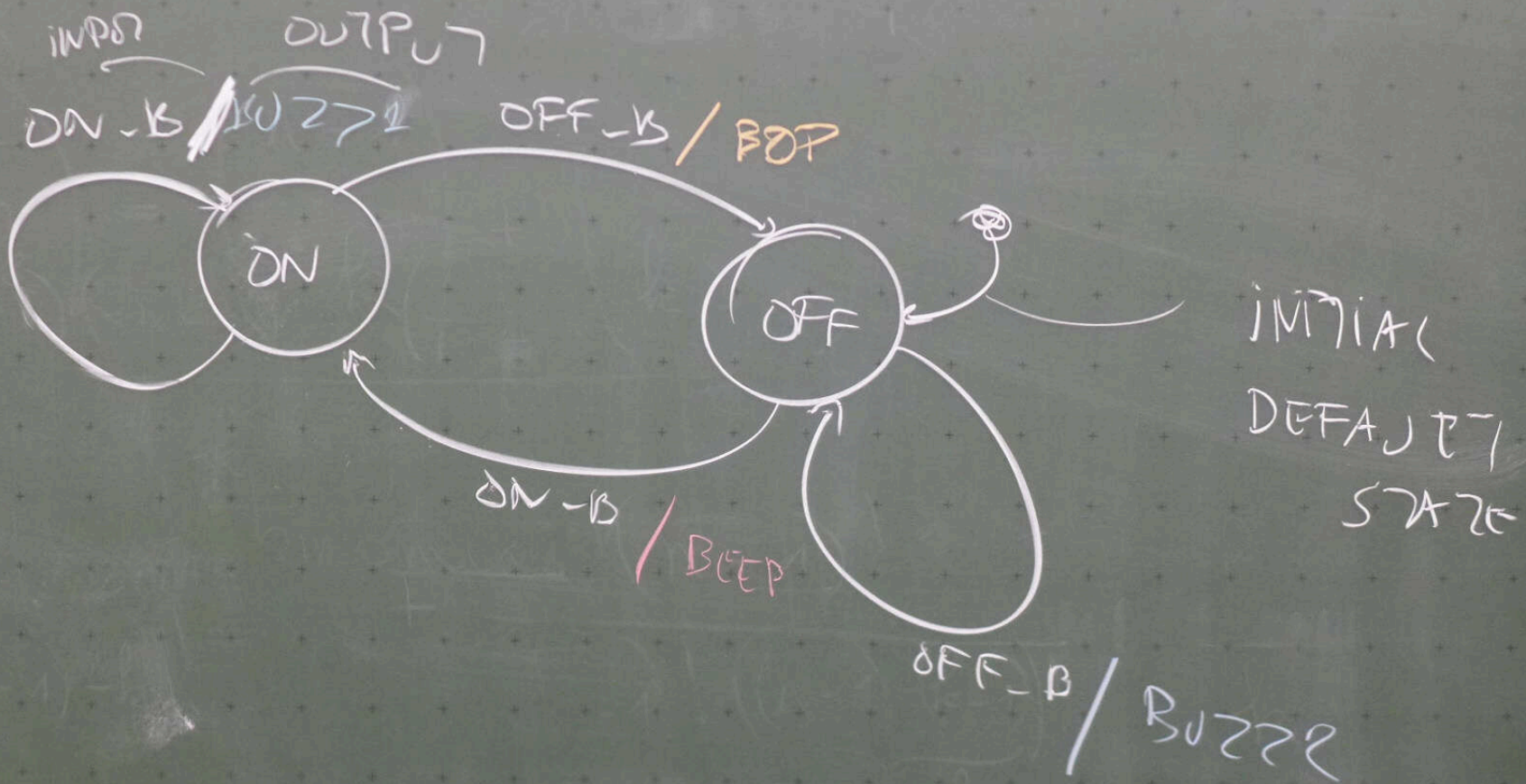
FINITE STATE AUTOMATA

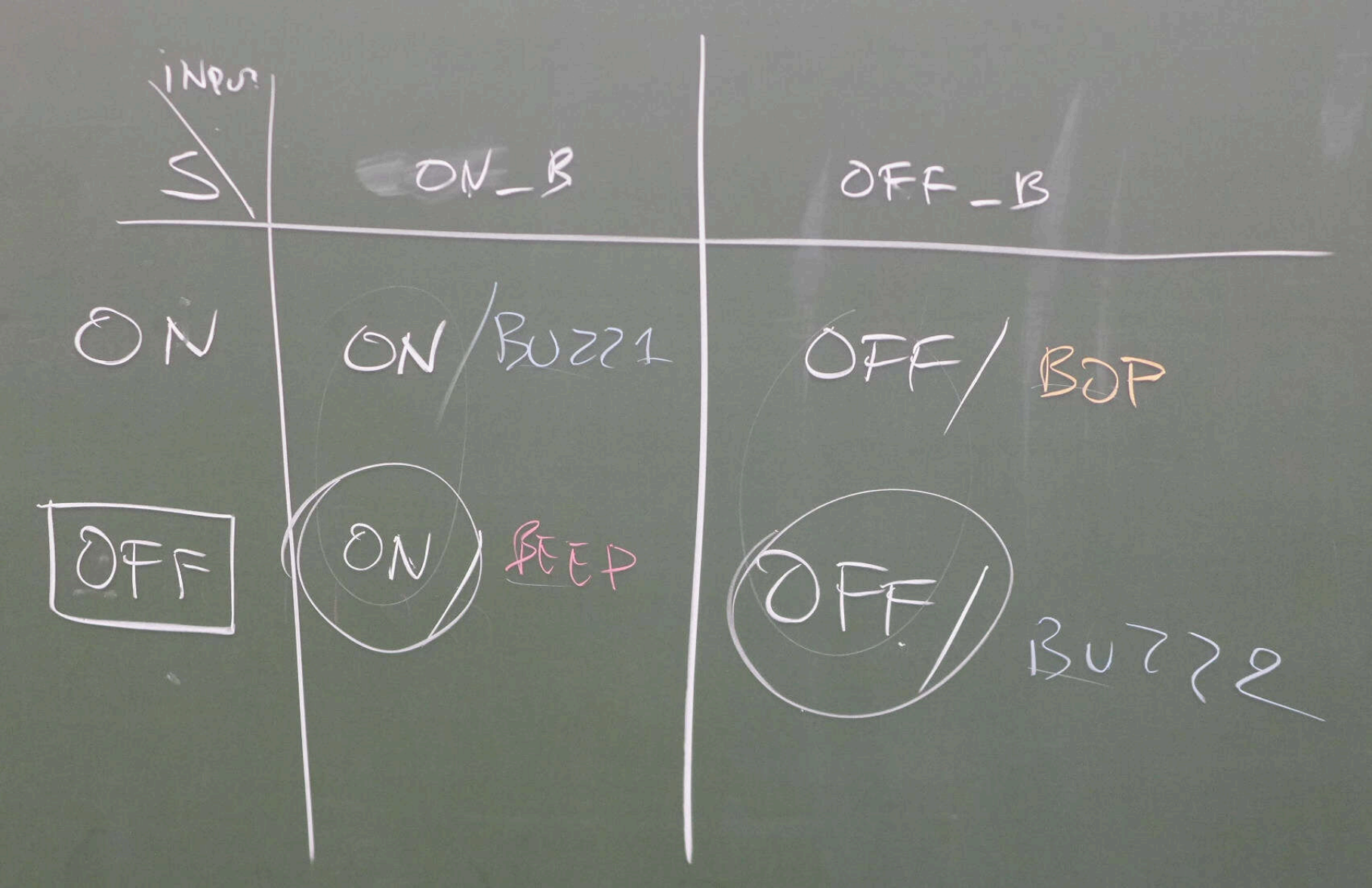


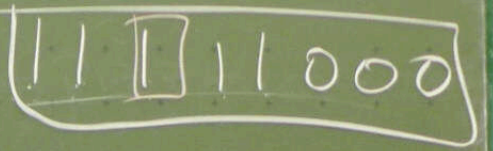
$S = \{ON, OFF\}$

S \ INPUT	ON_B	OFF_B
ON	ON/BEEP	OFF/BOP
OFF (INIT)	ON/BEEP	OFF/BOP

"MEALY" MACHINE




INPUT
S
ON_B
OFF_B
ON
ON / BUZZ1
OFF / BOP
OFF
ON / BEEP
OFF / BUZZ2



Δt

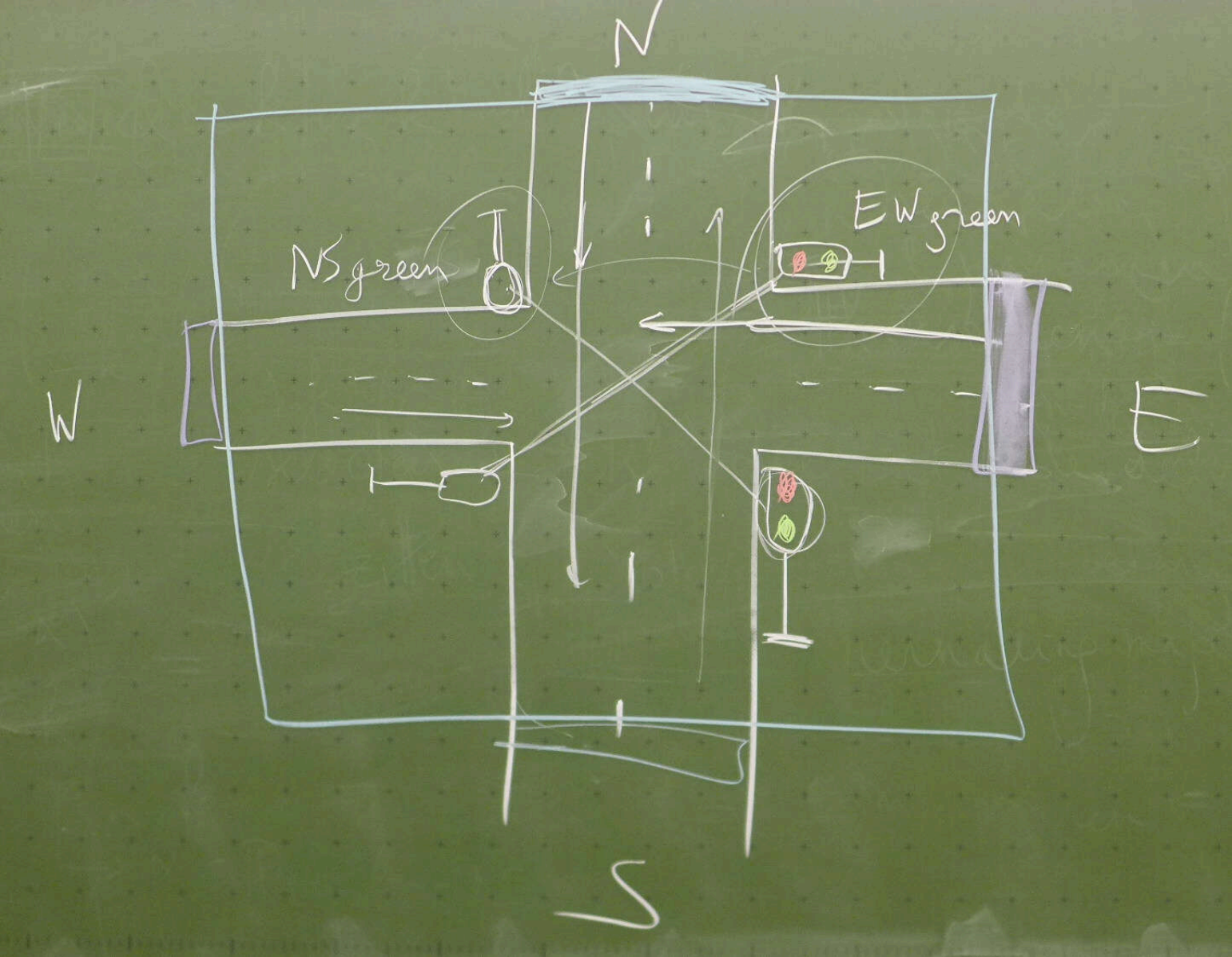
1 PARITY BIT EVEN

→ DETECT

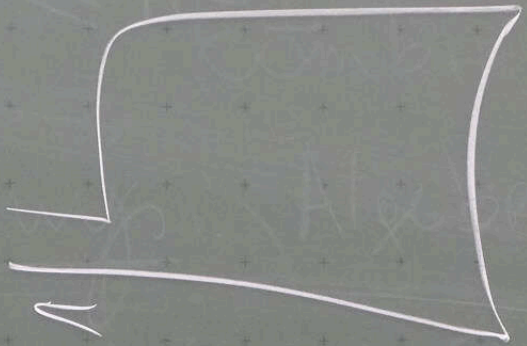
— 1 BIT ERROR

NACK

RESEND

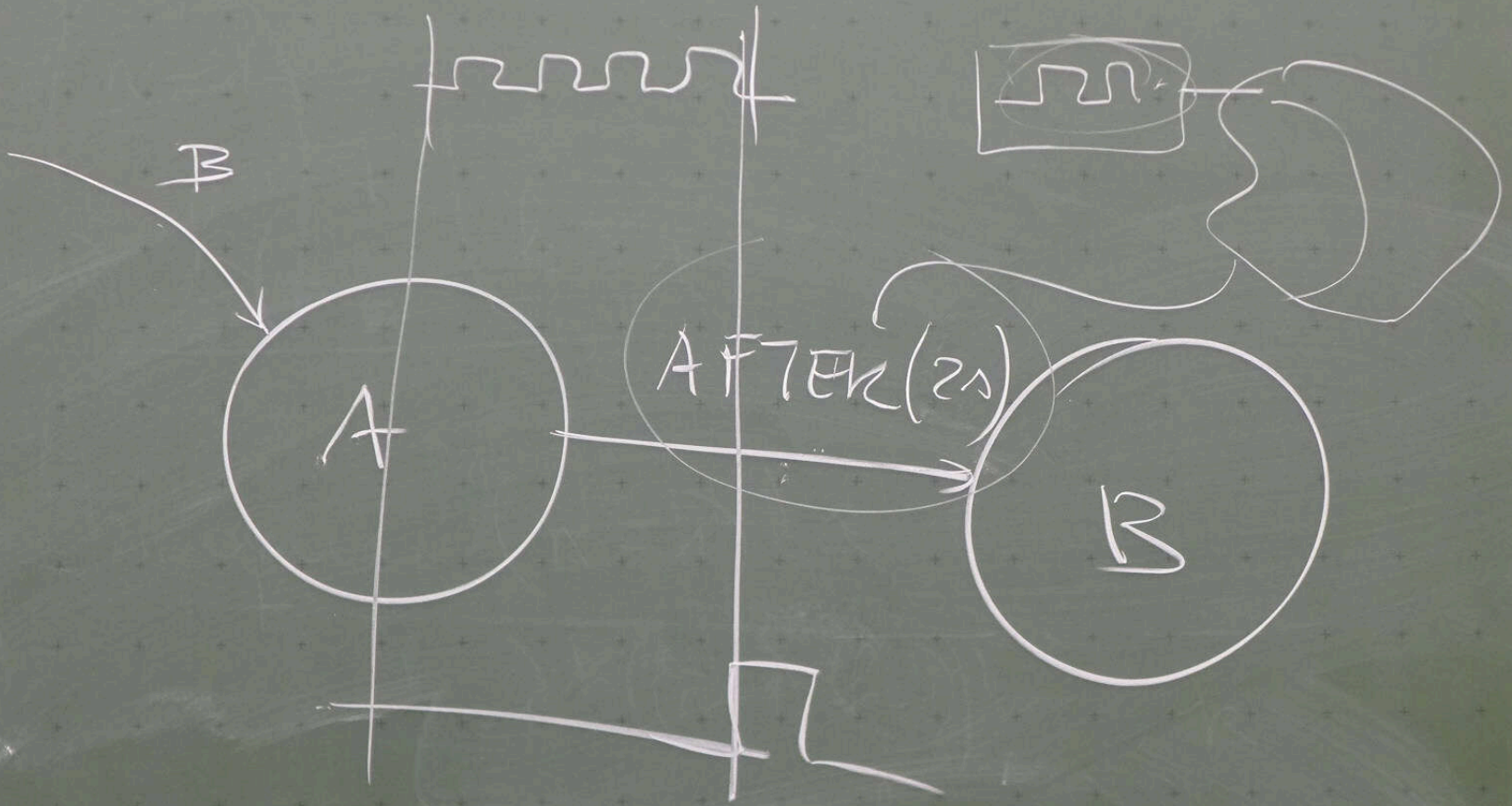

N
NS green
EW green
W
E
S

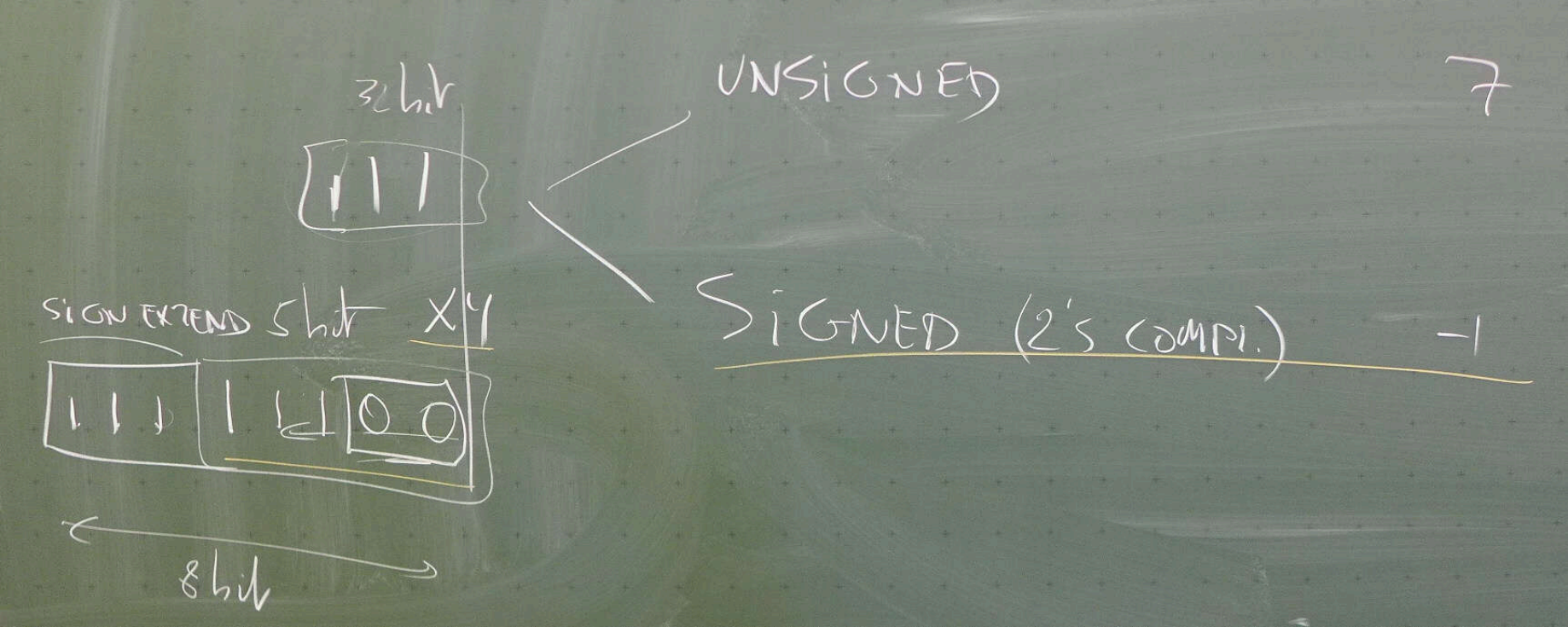
I. INDUCTION LOOP

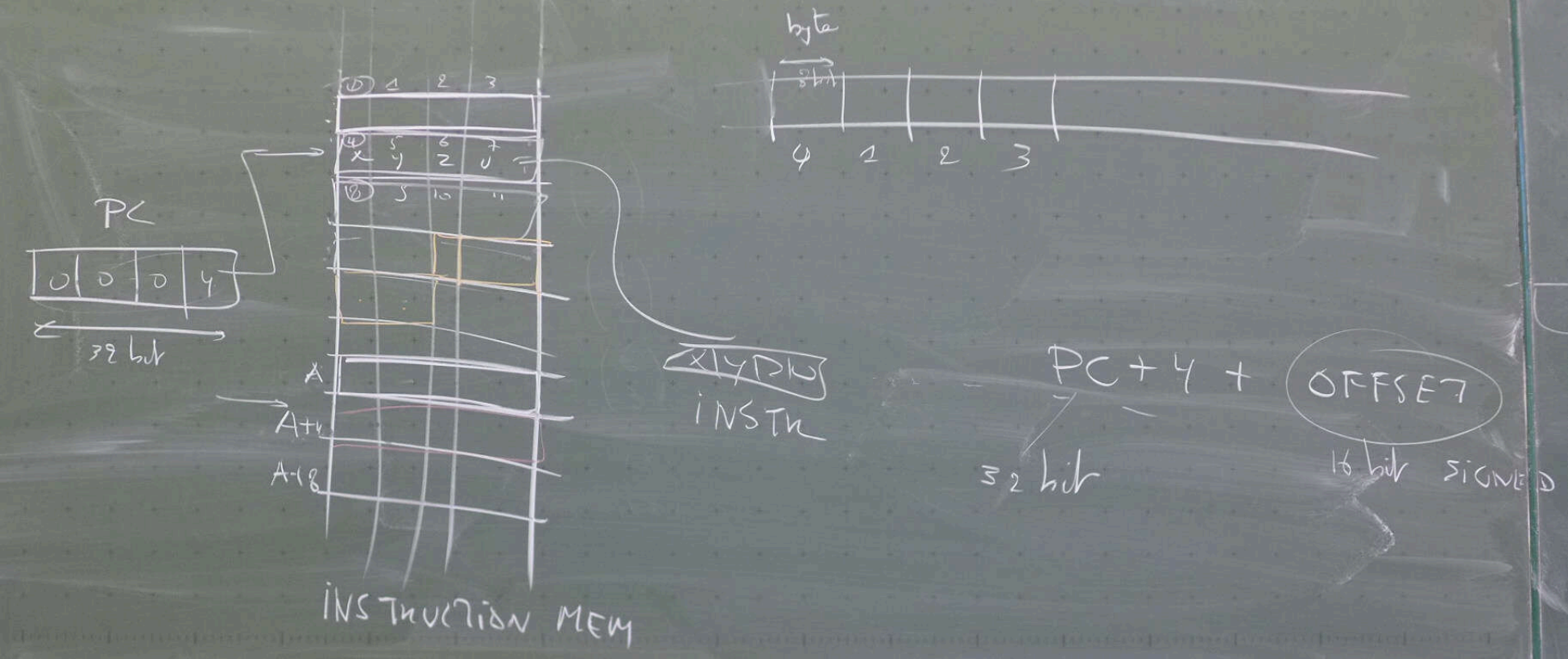


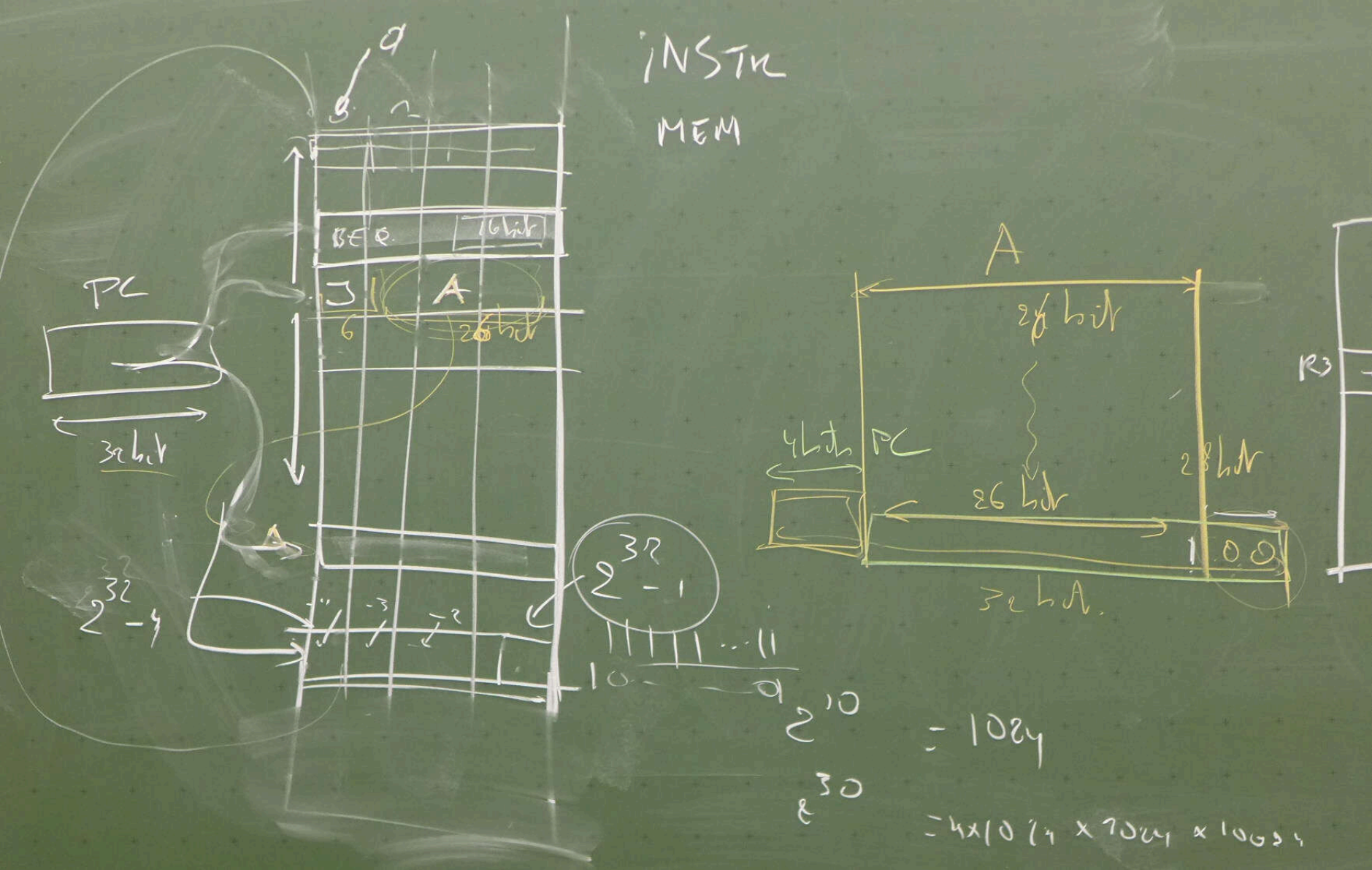
II. PIEZO-ELECTRIC

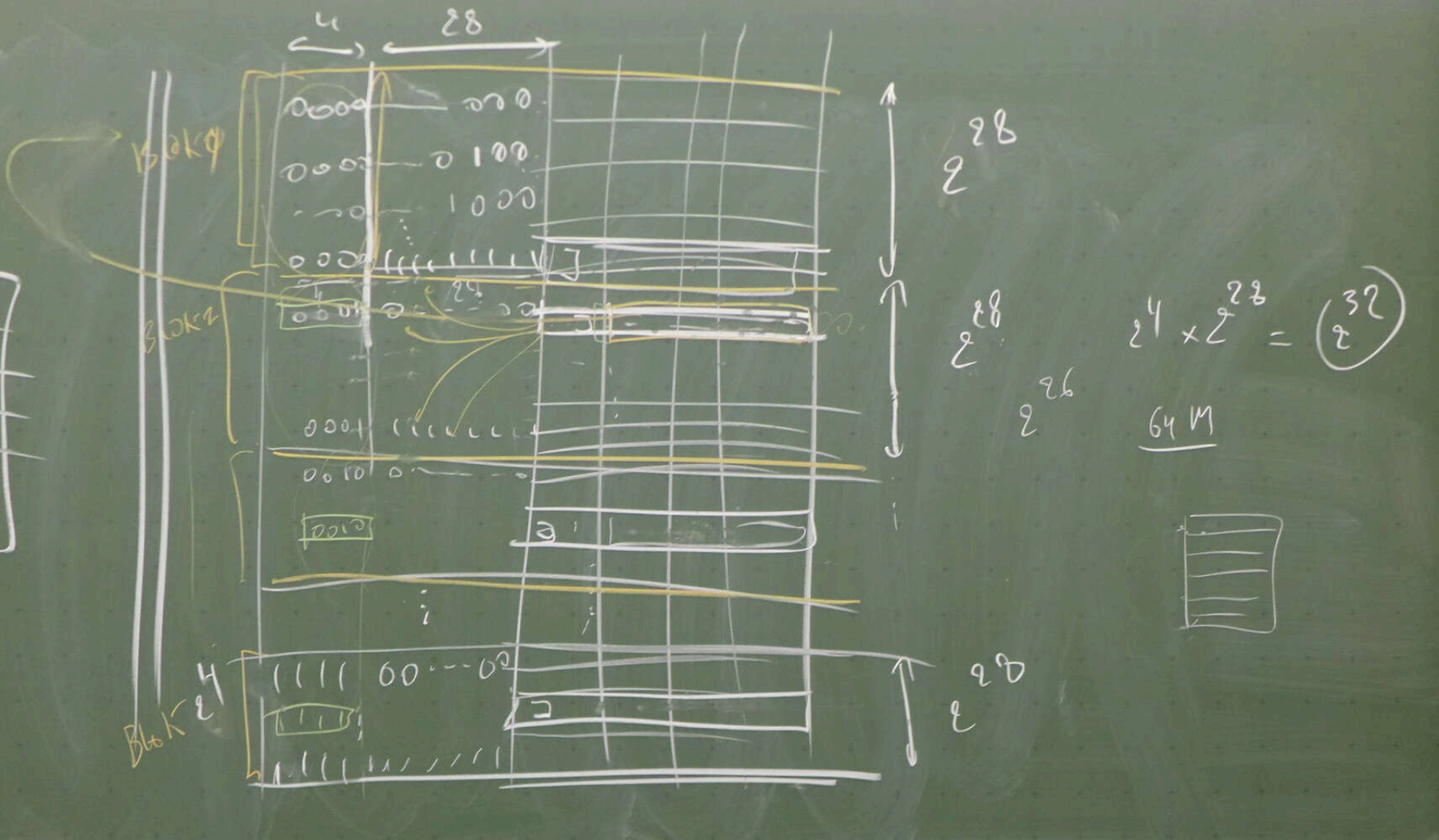
TIMED AUTOMATA

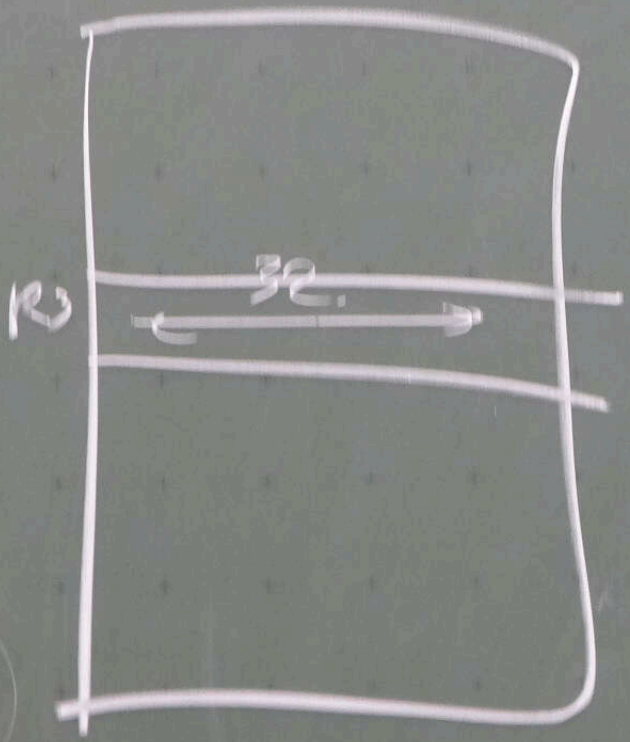


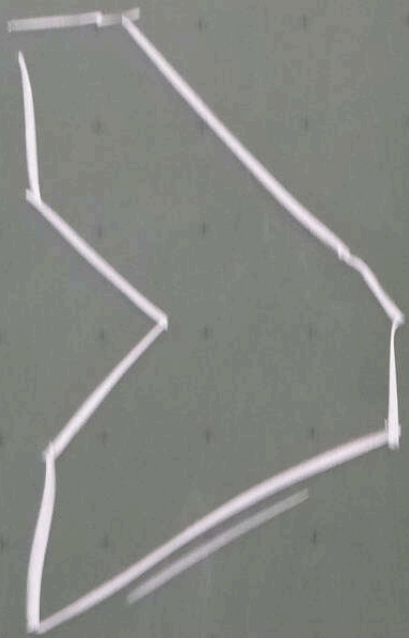

3 bit
1 1 1
UNSIGNED
7
SIGNED (2's COMPL.)
-1
SIGN EXTEND Shift X Y
1 1 1 1 1 0 0
8 bit


byte
8 bit
0 1 2 3
PC
0 0 0 4
32 bit
A
A+4
A-18
INSTRUCTION MEM
INSTR
PC + 4 +
OFFSET
32 bit
16 bit SIGNED

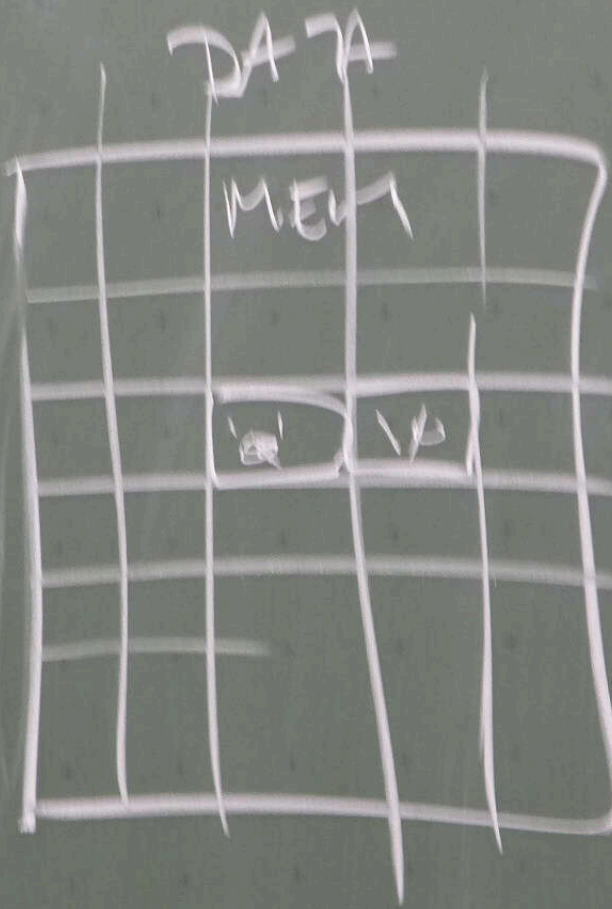

INSTR
MEM
PC
J
A
A
R3
PC
100
$2^{32}-1$
$2^{10} = 1024$

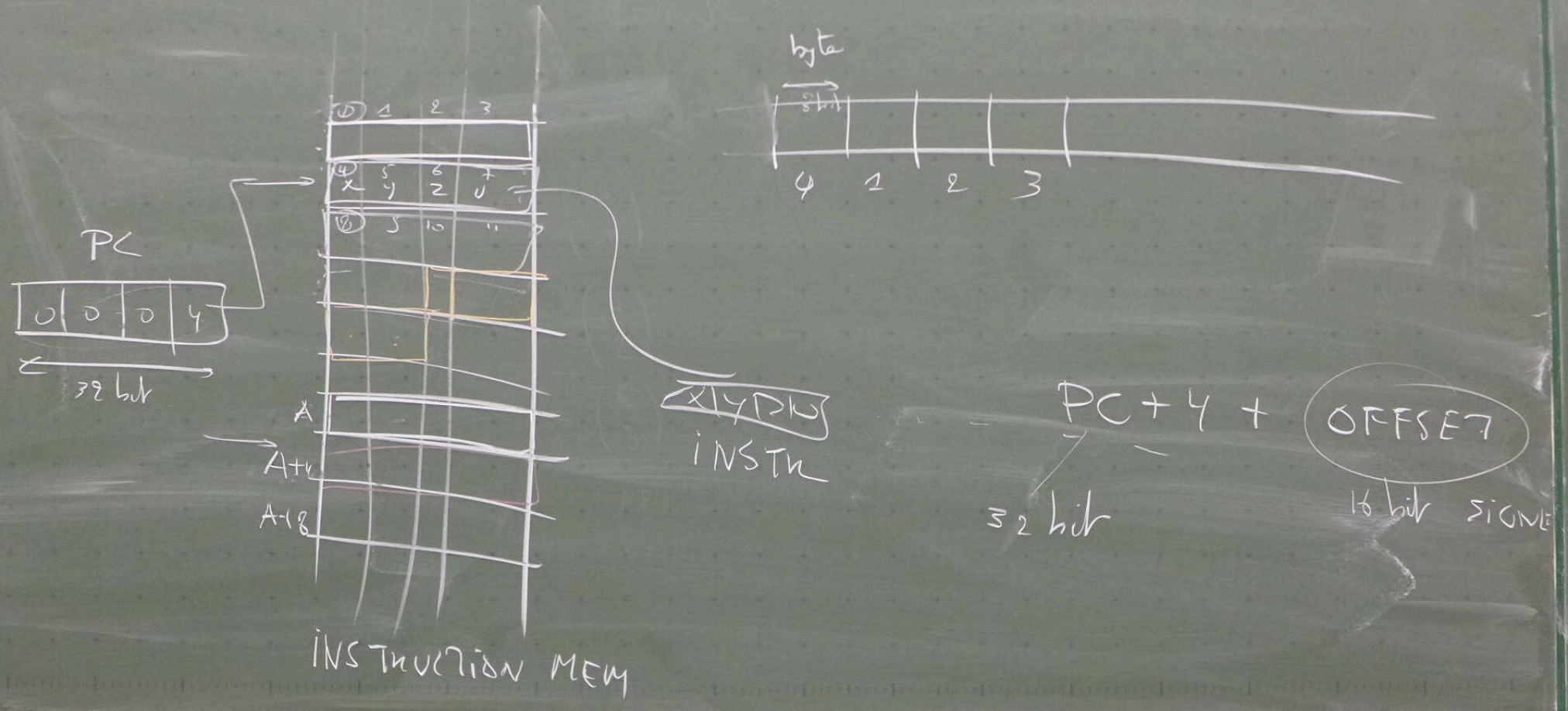

4
28
0000
0100
1000
0001
0010
1111 00...00
1111
2^{28}
2^{28}
2^{26}
2^4
$2^4 \times 2^{28} = 2^{32}$
64 M
2^{28}

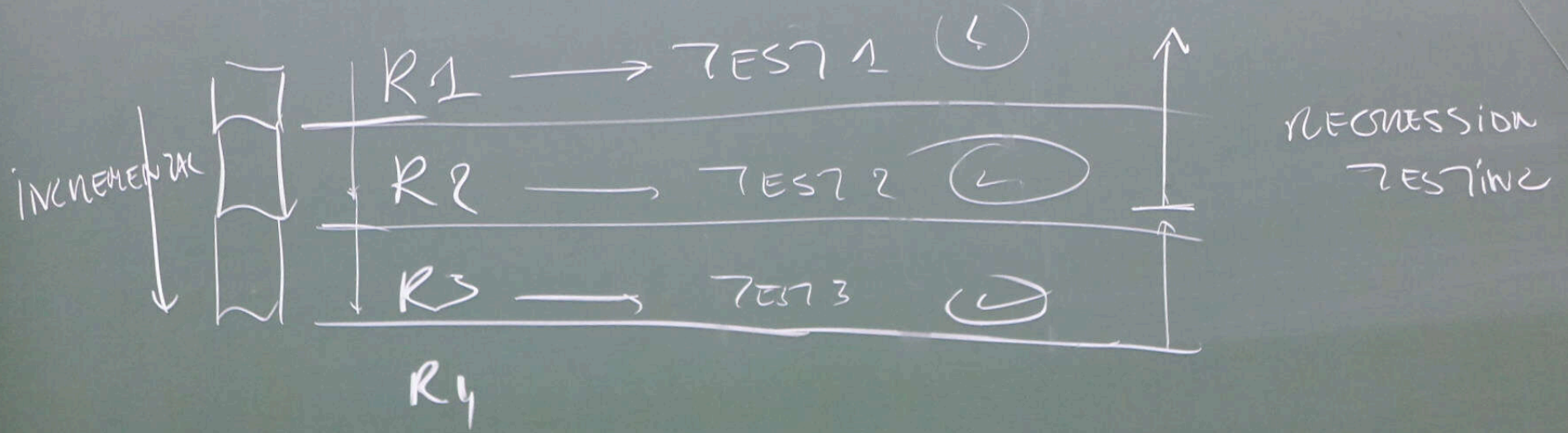


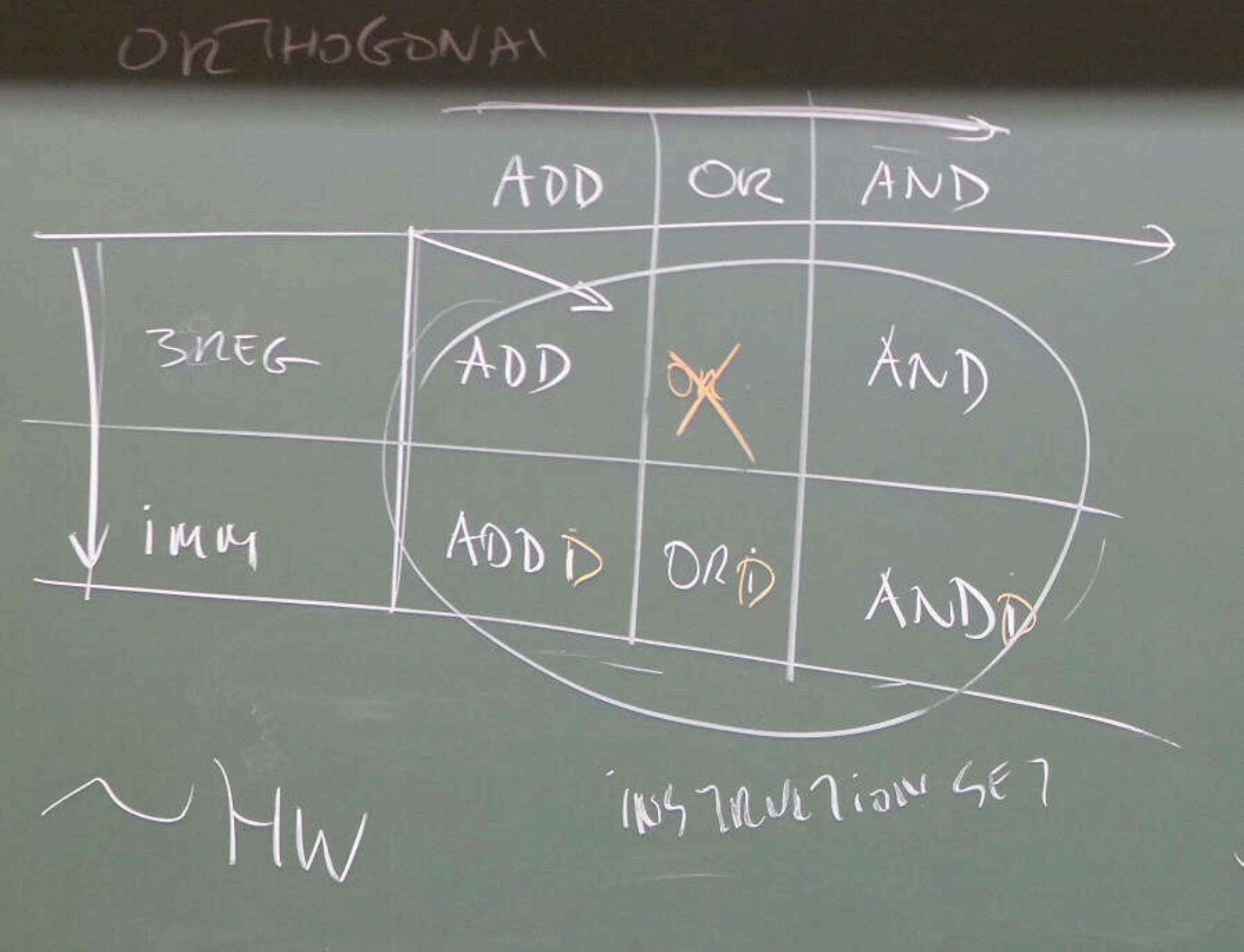


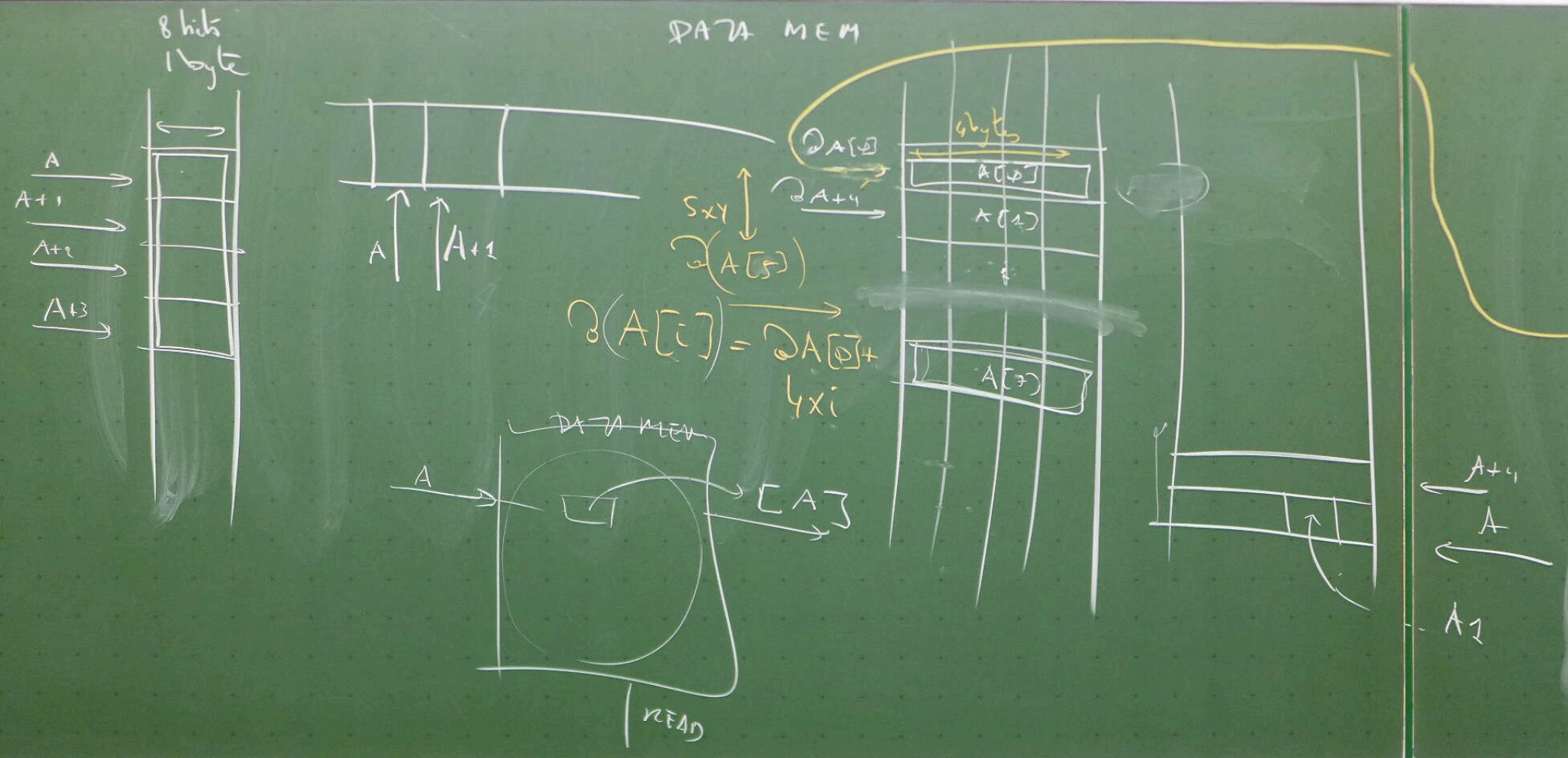
(JR $3)

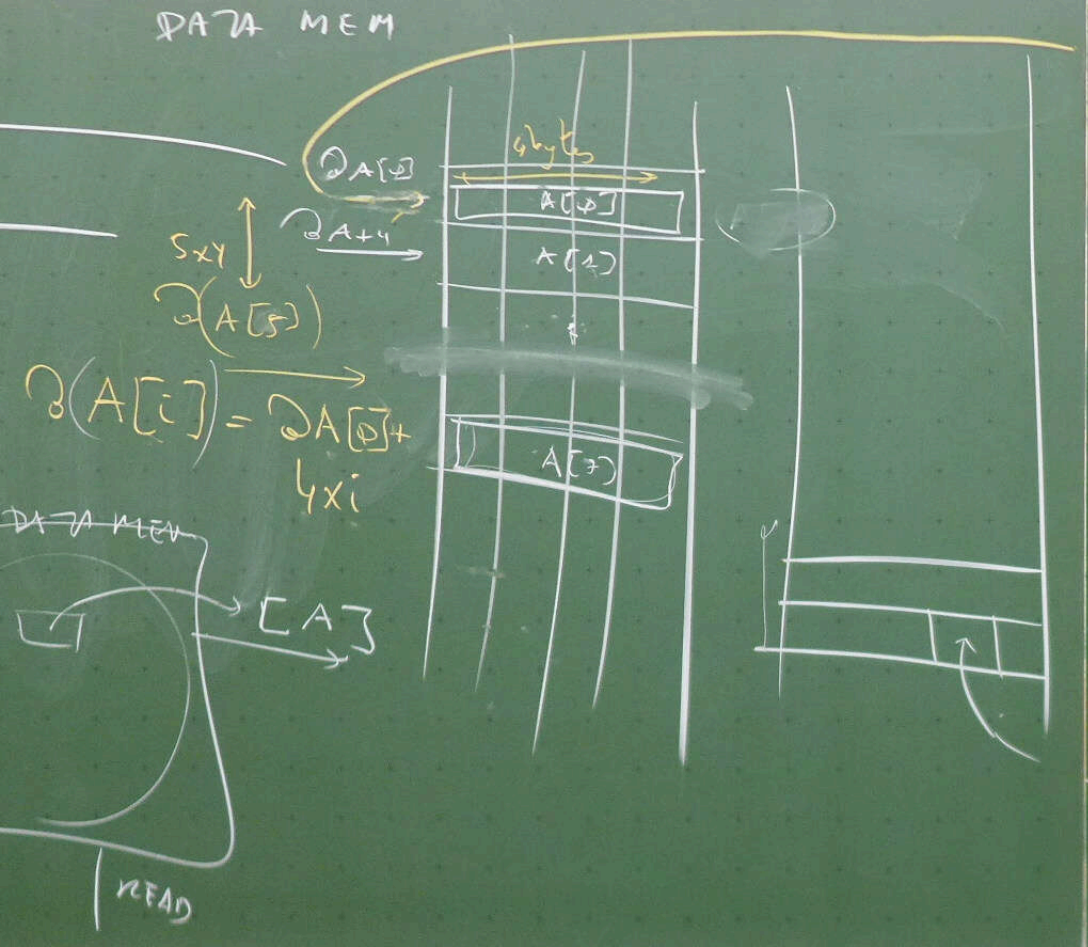

DATA
MEM

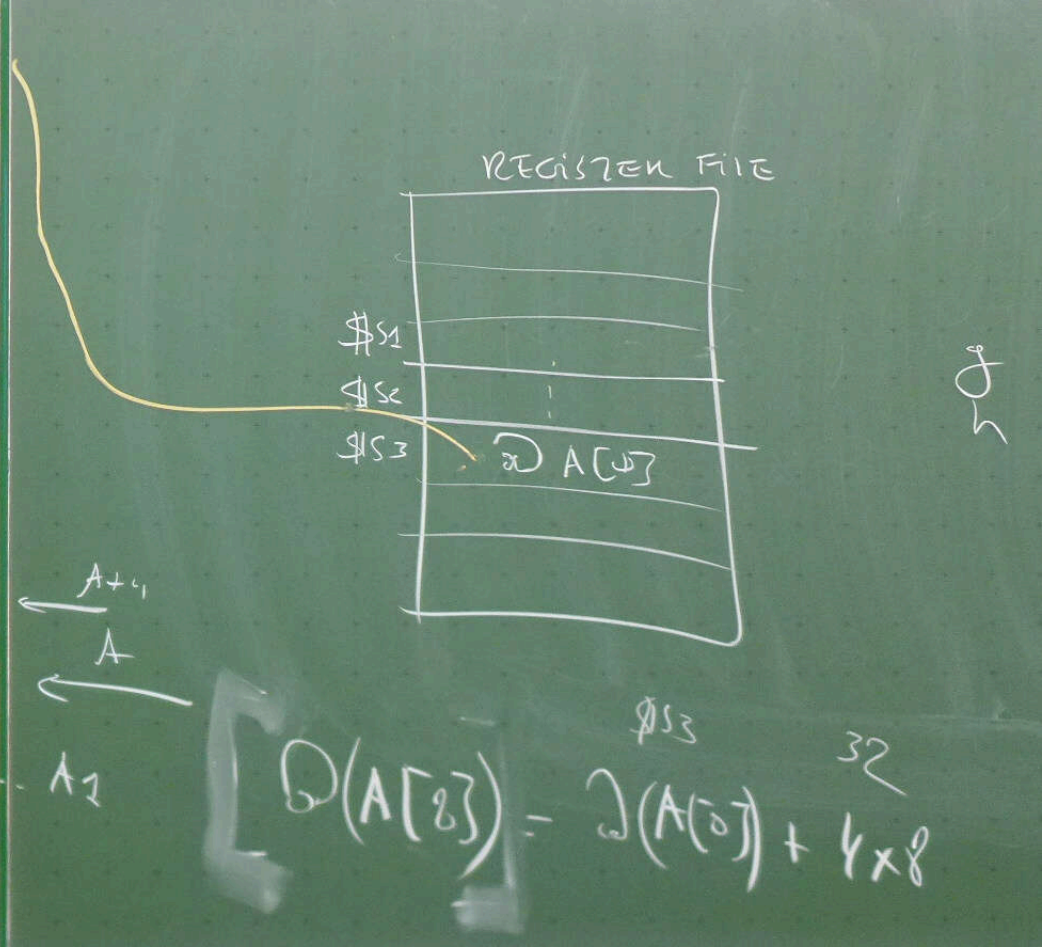

byte
4 1 2 3
PC
0 0 0 4
32 bit
A
A+4
A+8
INSTRUCTION MEM
INSTR
PC + 4 +
OFFSET
32 bit
16 bit


INCREMENTAL
R1
TEST 1
R2
TEST 2
R3
TEST 3
R4
REGRESSION
TESTING


ORTHOGONAL
ADD
OR
AND
3REG
ADD
OR
AND
imm
ADDI
ORI
ANDI
~HW
INSTRUCTION SET


8 bits
1 byte
DATA MEM
A
A+1
A+2
A+3
A
A+1
5x4
A+4
A[0]
A[5]
A[i] = A[0] + 4xi
4 byte
A[0]
A[1]
A[7]
DATA MEM
A
[A]
READ
A+4
A
A1


DATA MEM
@A[0]
4 by 4
A[0]
@A+4
A[1]
5x4
@(A[5])
@(A[i]) = @A[0] + 4xi
A[7]
DATA MEM
[A]
READ


REGISTER FILE
$S1
$S2
$S3
@ A[0]
A+4
A
A1
@(A[8]) = @(A[0]) + 4x8
$S3
32

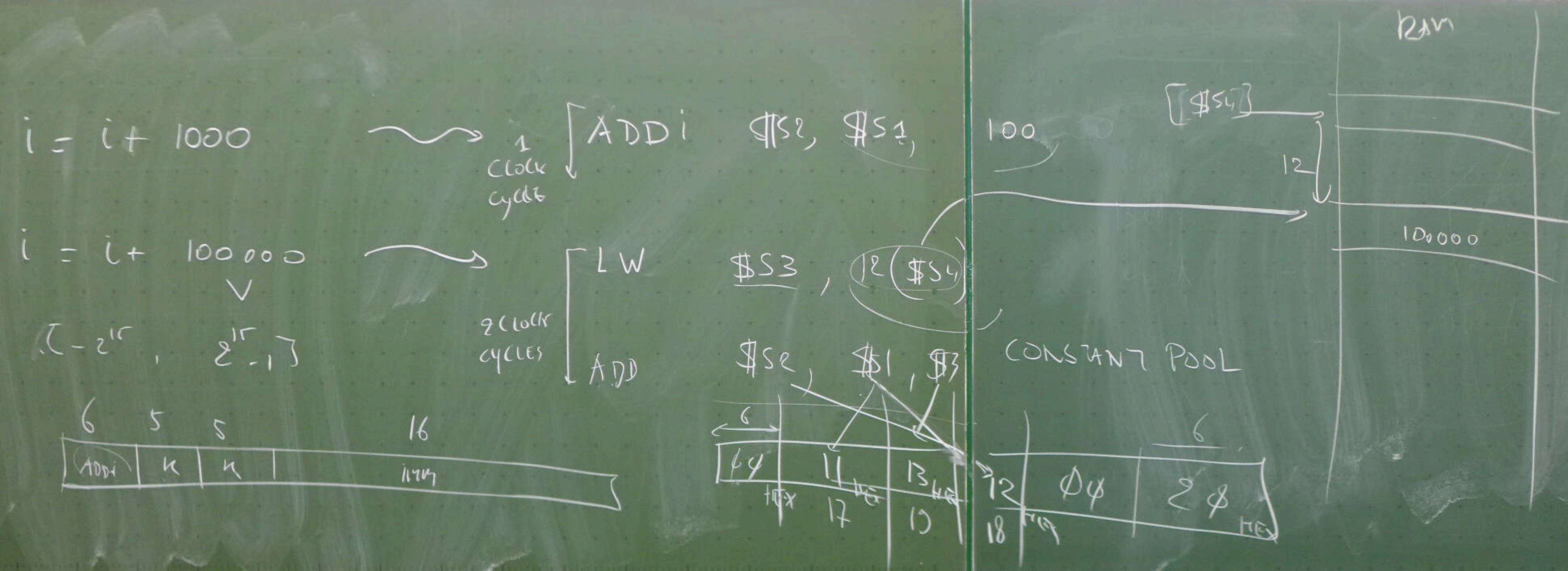
ARRAY A OF T

LEN(T) = L

BASE ADDRESS (A) =

$\Omega A = \Omega(A[0])$

$\Omega(A[i]) = \Omega(A[0]) + L * i$

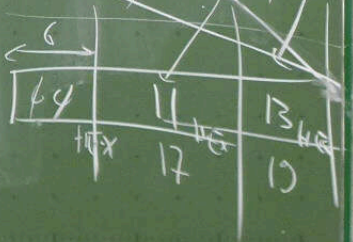

i = i + 1000
1 Clock cycle
ADDi $S2, $S1, 100
i = i + 100000
[-2^15 , 2^15 -1]
2 Clock cycles
LW $S3 , 12($S4)
ADD $S2 , $S1 , $S3
6 5 5 16
ADDi
imm
04
11
13
17
19
12
18
CONSTANT POOL
[$S4]
RAM
100000

$i = i + 1000$ ⇝ 1 clock cycle [ADDi $S2, $S1, 1000

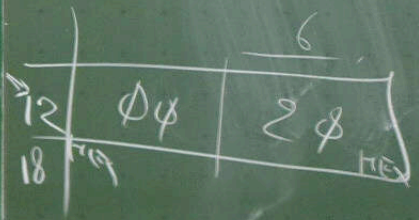
$i = i + 100000$ ⇝ 2 clock cycles [LW $S3, 12($S4)

ADD $S2, $S1, $S3

$[-2^{15}, 2^{15}-1]$



CONSTANT POOL



RAM

[$S4]

12

100

1D0000

ZERO:

```
ADDi   $ZERO, $ZERO, 2
```

PSEUDO CODE

```
SUBi   $S1, $S2, 1
```

↓

```
ADDi   $S1, $S2, -1
```

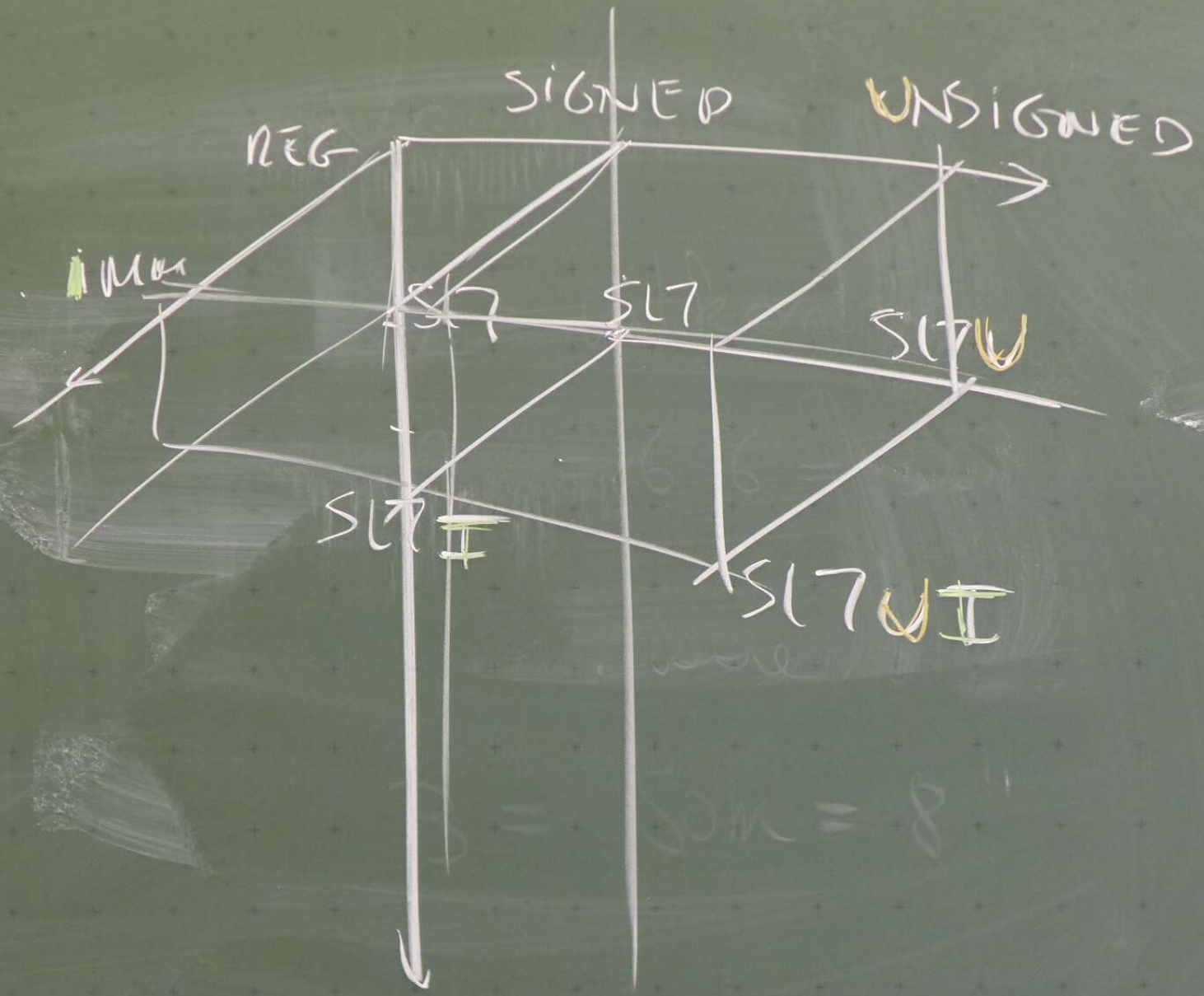
↳ MIPS INSTR (DISASSEMBLE)

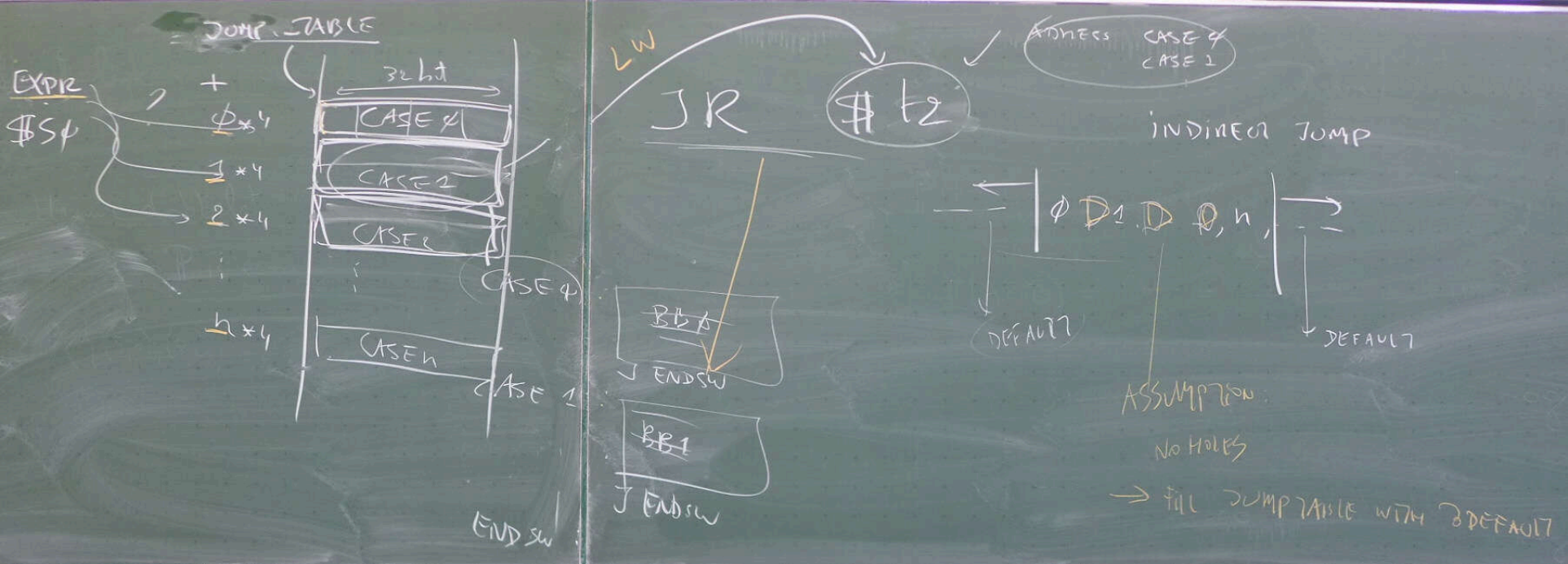
↳ ASCII C STRING

↳ UNSIGNED INT

↳ FP

THEORIE EXAMEN


SIGNED
UNSIGNED
NEG
IMM
SLT
SLT
SLTU
SLTI
SLTUI


JUMP TABLE
EXPR
$s4
+
0*4
1*4
2*4
h*4
32 bit
CASE 0
CASE 1
CASE 2
CASE h
CASE 0
CASE 1
END SW
LW
JR
$t2
BB A
J ENDSW
BB 1
J ENDSW
ADDRESS
CASE 0
CASE 1
INDIRECT JUMP
0 1 2 h
DEFAULT
DEFAULT
ASSUMPTION
NO HOLES
→ fill JUMP TABLE WITH DEFAULT

void f(int i, int j)

SPECIFI

IMPLEM

}

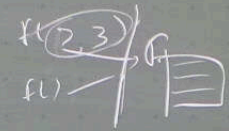
× HEK-GEBRUIK

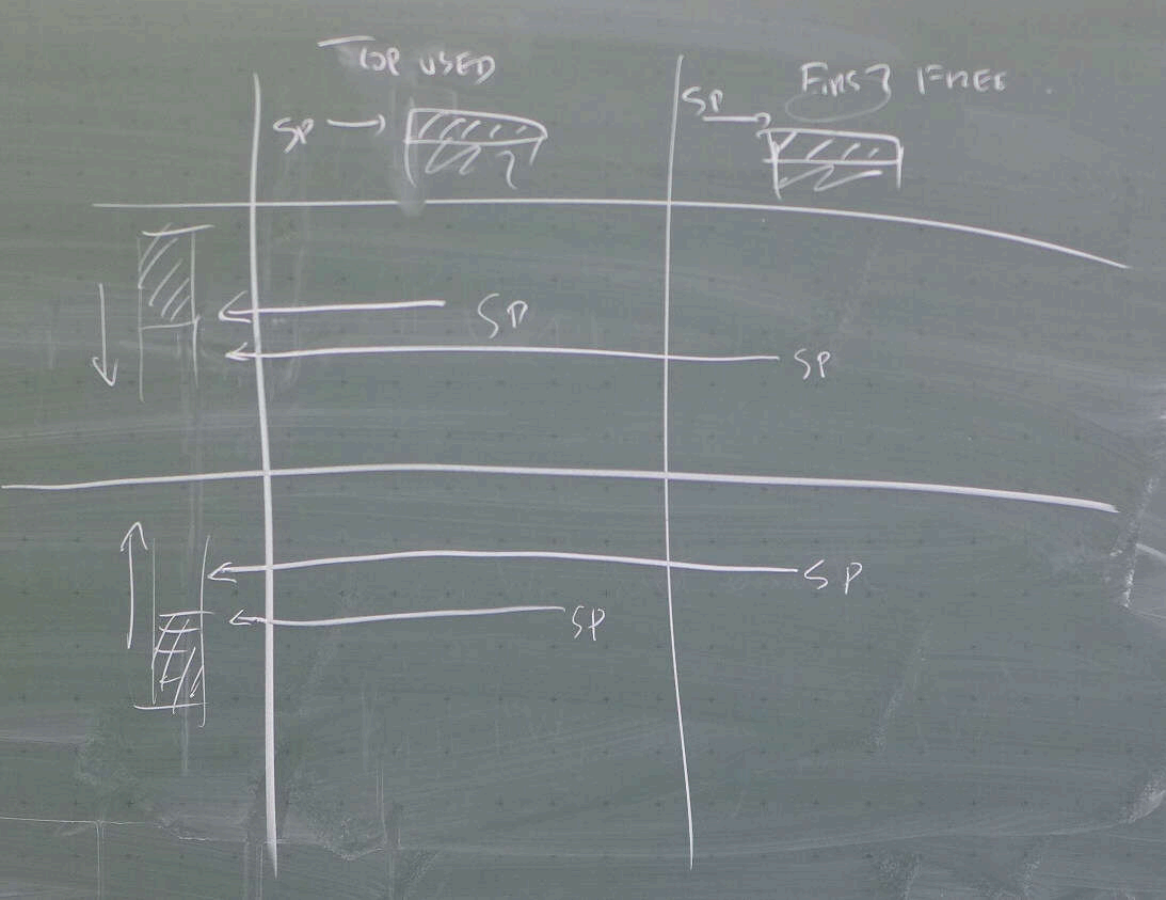
× PARAMETRISERING

× ENCAPSULATIE → SCHEIDEN SPEC / IMPL → EVOLUTIE

ABSTRACTIE GEMAKKELIJK GEMAKKELIJKER

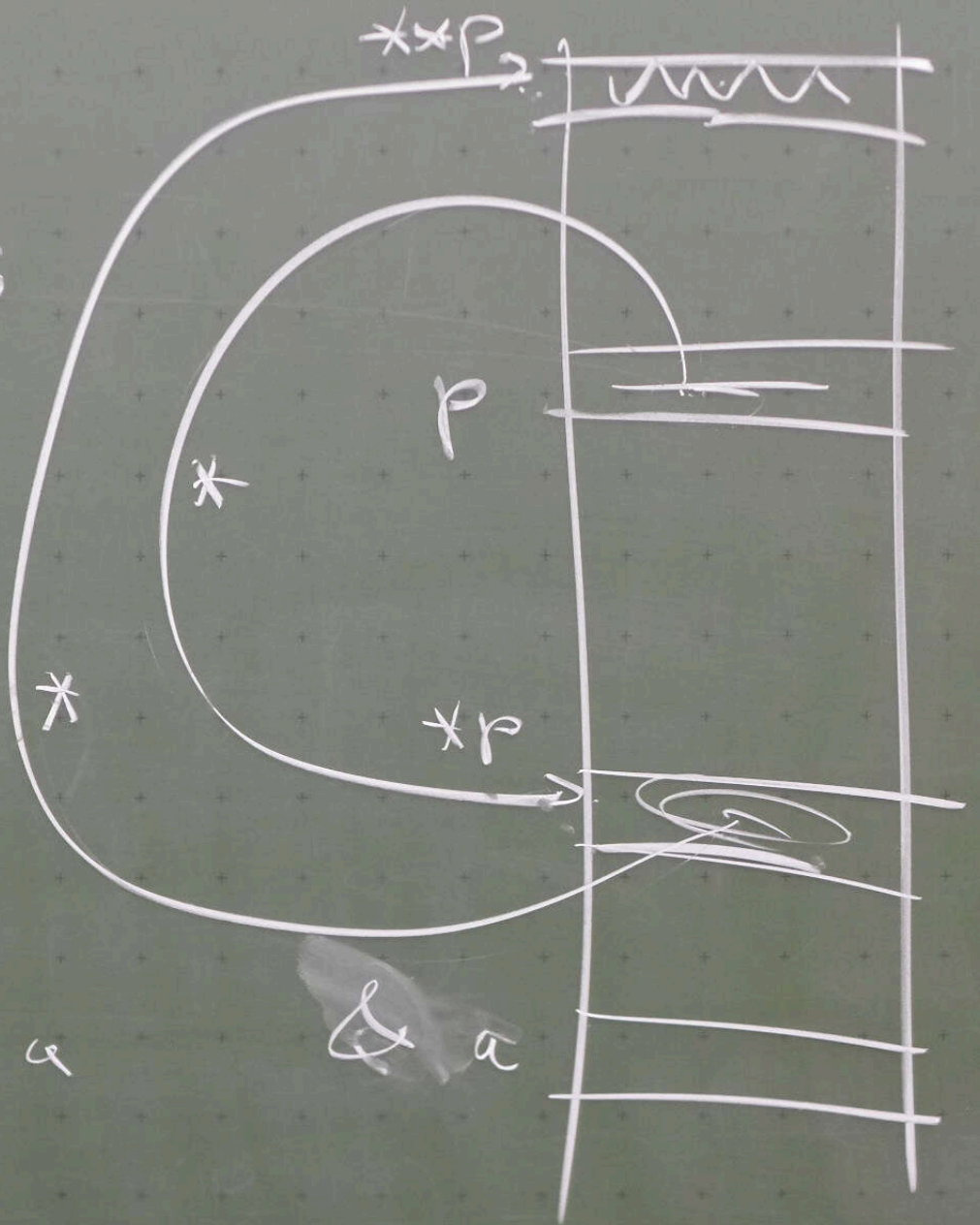
× COGNITIEF

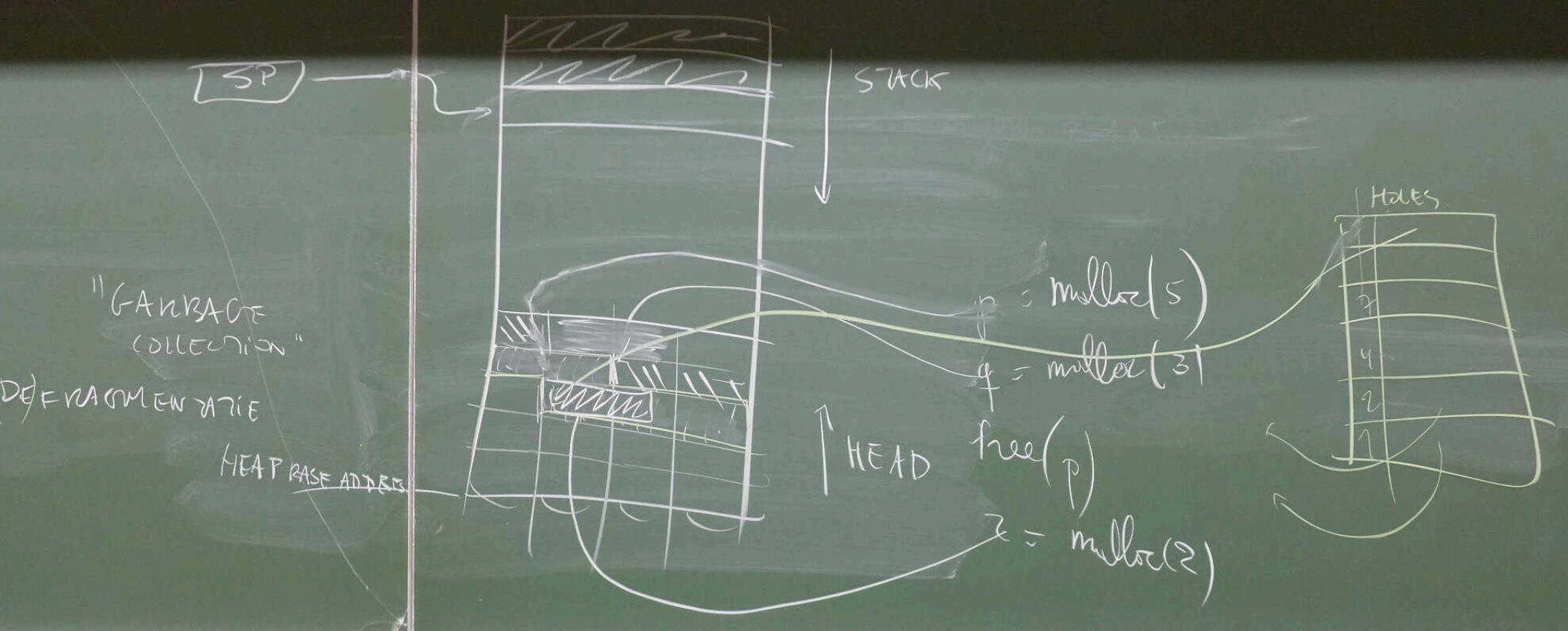


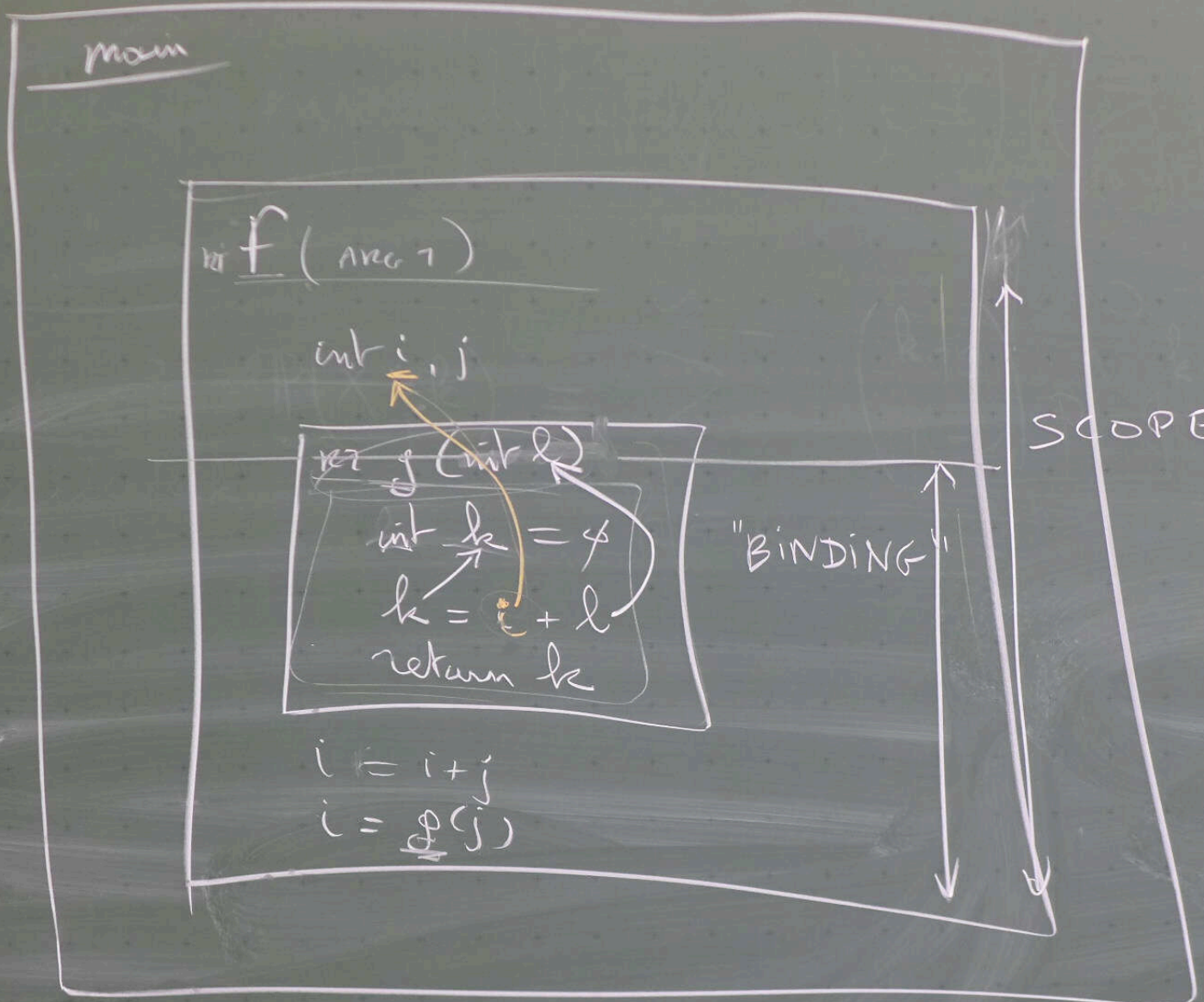

TOP USED
SP
SP
FIRST FREE
SP
SP
SP
SP

int **p;

int **p




SP
STACK
HEAP BASE ADDRESS
HEAD
"GARBAGE COLLECTION"
DEFRAGMENTATIE
p = malloc(5)
q = malloc(3)
free(p)
z = malloc(2)
Holes
7
4
2
1


main
int f(int arg1)
int i, j
int g(int l)
int k = 4
k = i + l
return k
i = i + j
i = g(j)
"BINDING"
SCOPE

VECTOR V : (1) SEQUENTIAL ADDRESSES (COMPACT) (2) 1 ELEMENT TYPE $\mathbb{R}^n$

ELEMENT TYPE : ET

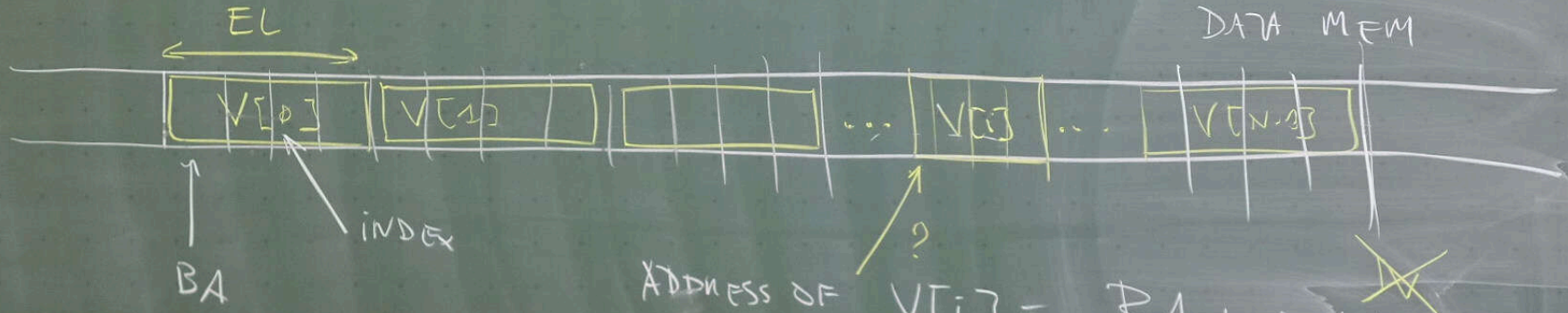
LEN (ELEMENT TYPE) = EL

BASE ADDRESS OF V : BA

$V : \mathbb{N} \longrightarrow ET$

$[0, N-1] \subseteq \mathbb{N}$

N is LENGTH OF V



MATRIX M R×K

ELEMENT TYPE ET

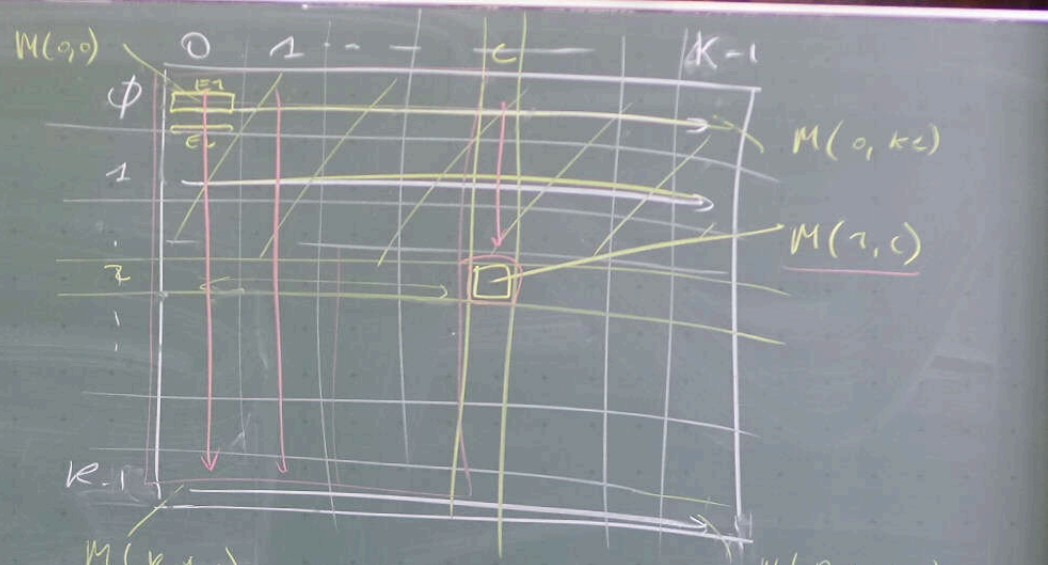
LEN(ET) = EL

BASE ADDRESS = BA

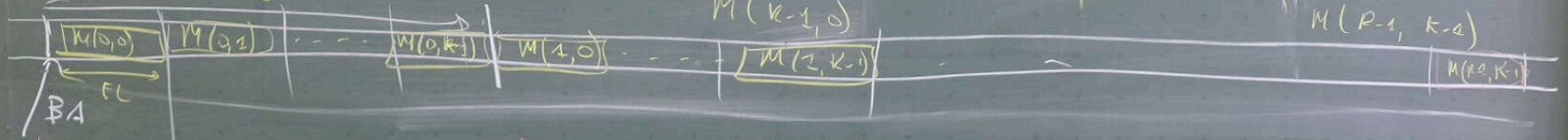
ADDRESS OF M(r,c) =

BA + (r×K + c)×EL

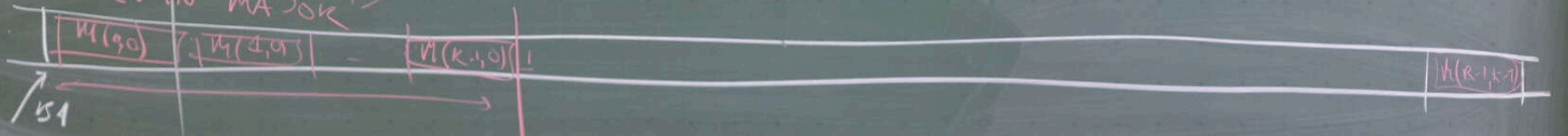
~~BA + (c×R + r)×EL~~

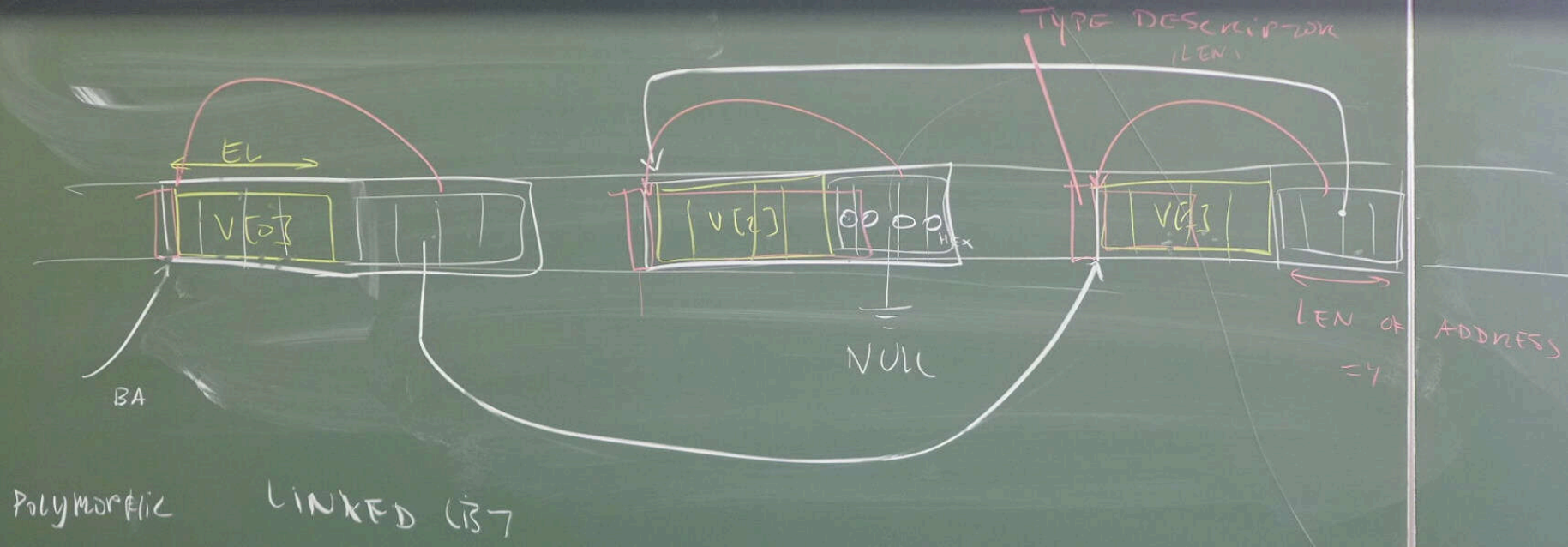


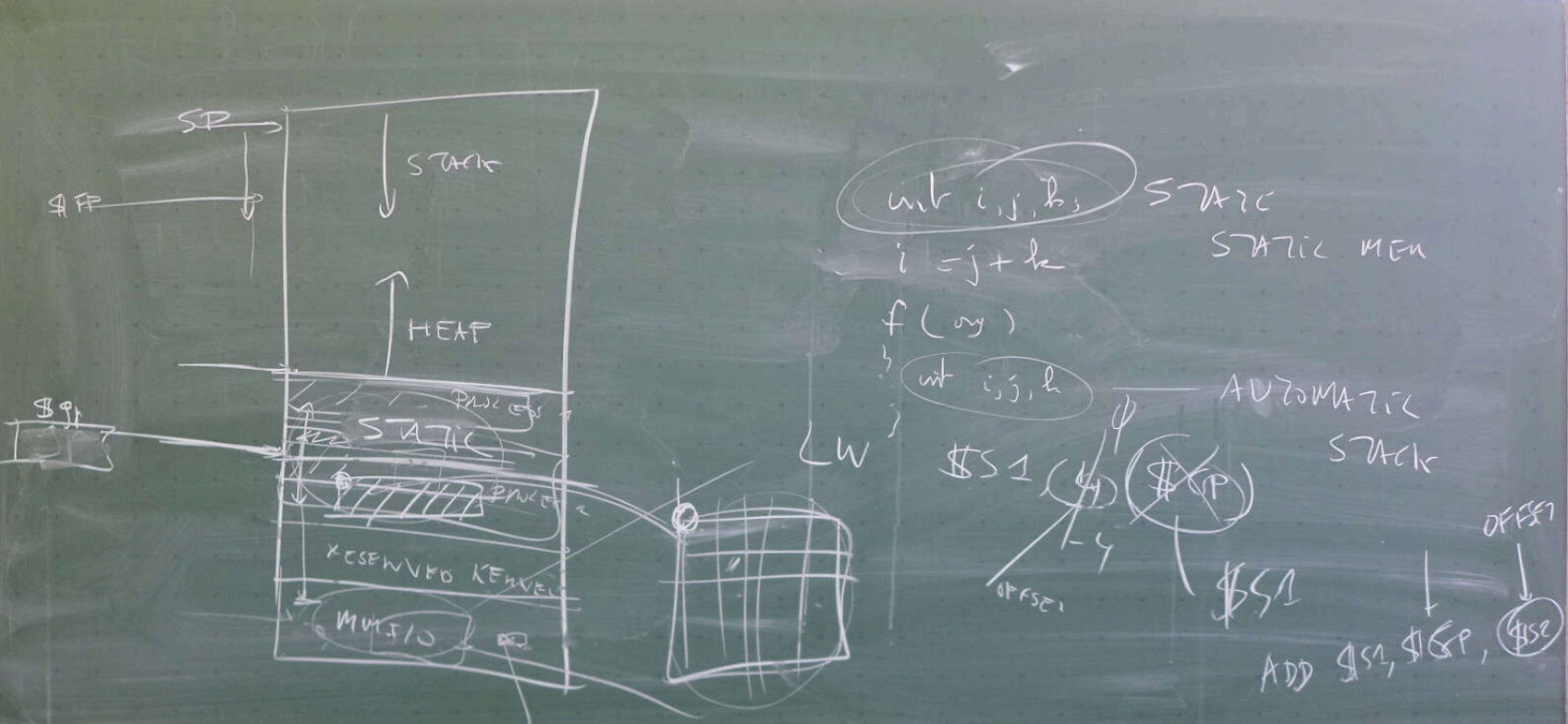
"ROW MAJOR" — C, Java



"COLUMN MAJOR" — FORTRAN




TYPE DESCRIPTOR
(LEN)
EL
V[0]
V[2]
HEX
V[4]
NULL
LEN OF ADDRESS
=4
BA
Polymorphic
LINKED LIST


SP
$FP
STACK
HEAP
$gp
STATIC
RESERVED KERNEL
MMIO
int i,j,k;
STATIC
STATIC MEM
i = j + k
f(int y)
int i,j,k
AUTOMATIC
STACK
lw
$s1
$fp
OFFSET
$s1
OFFSET
ADD $s1, $fp,
$s2

x [-1]

y [-2]

TRUE 1 ≠ 0

FALSE 0