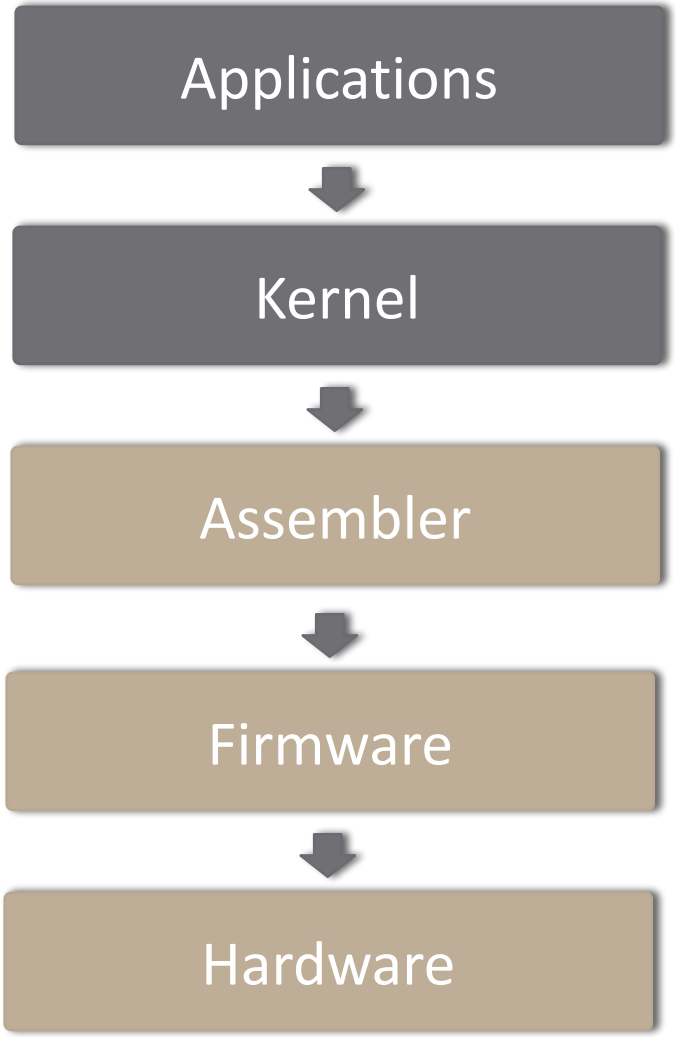
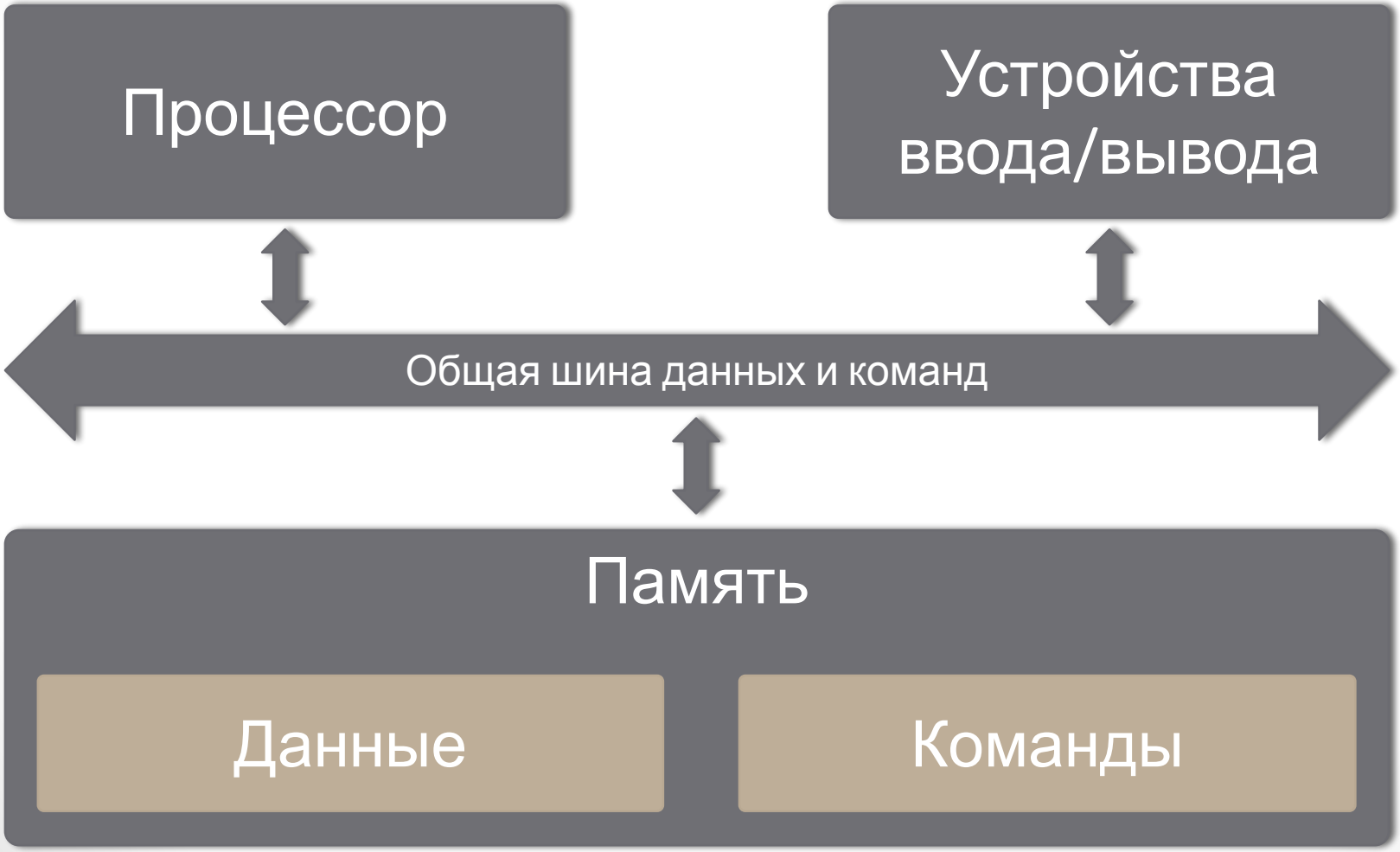
Ассемблер Atmel AVR

Занятие №1: Архитектура AVR, схемотехника ЭВМ.

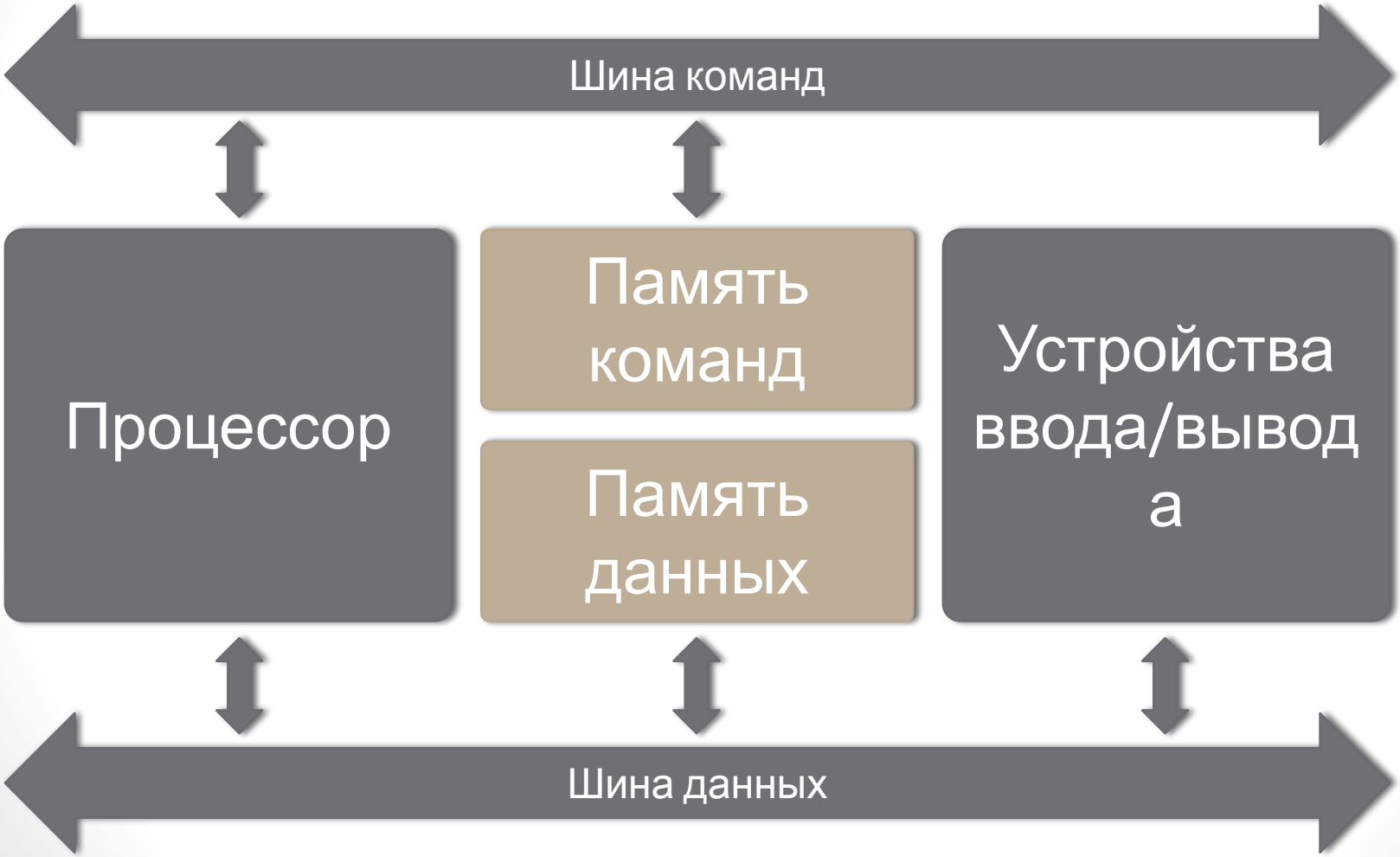
Уровни абстракции



Принстонская архитектура

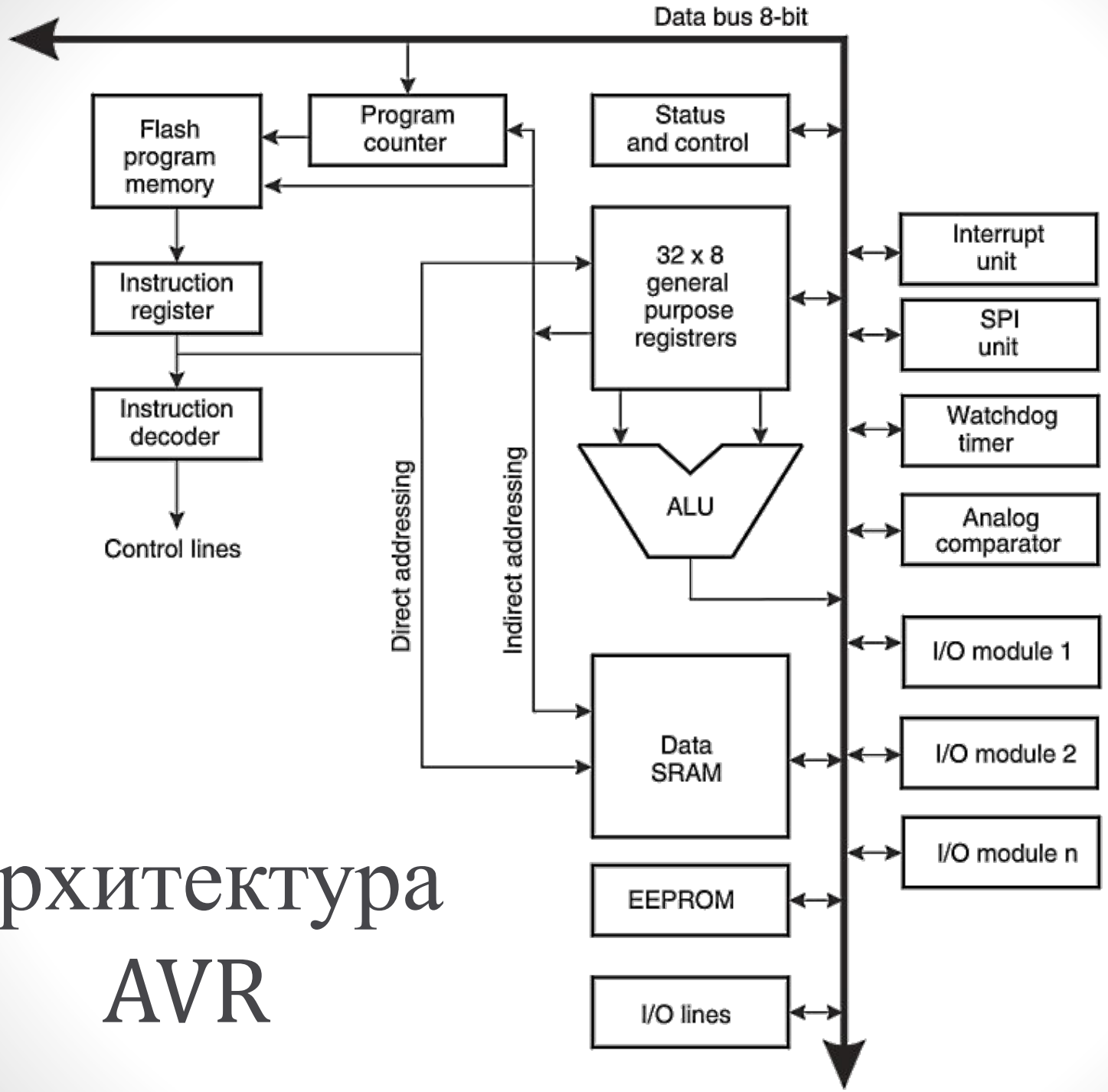


Гарвардская архитектура

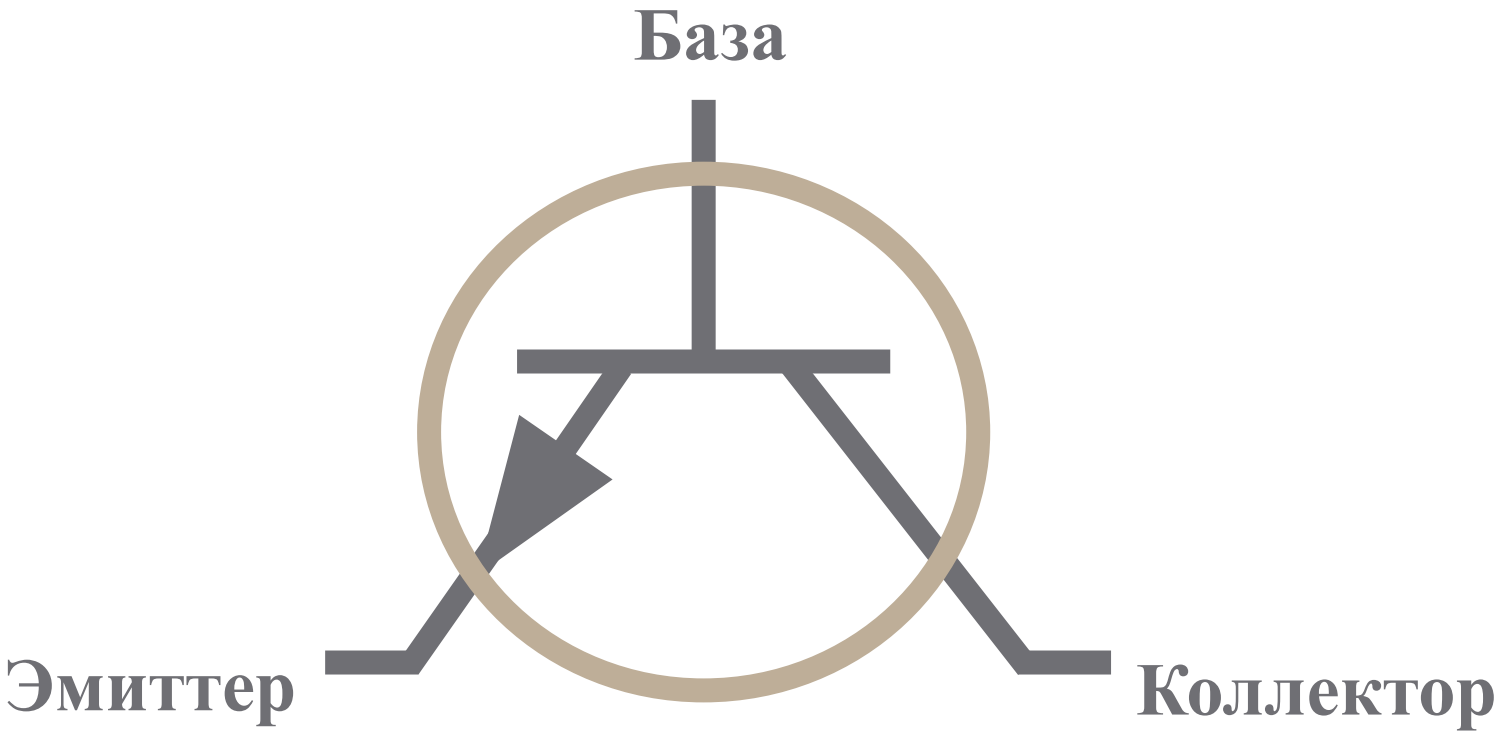


Архитектуры CISC и RISC

Архитектура AVR

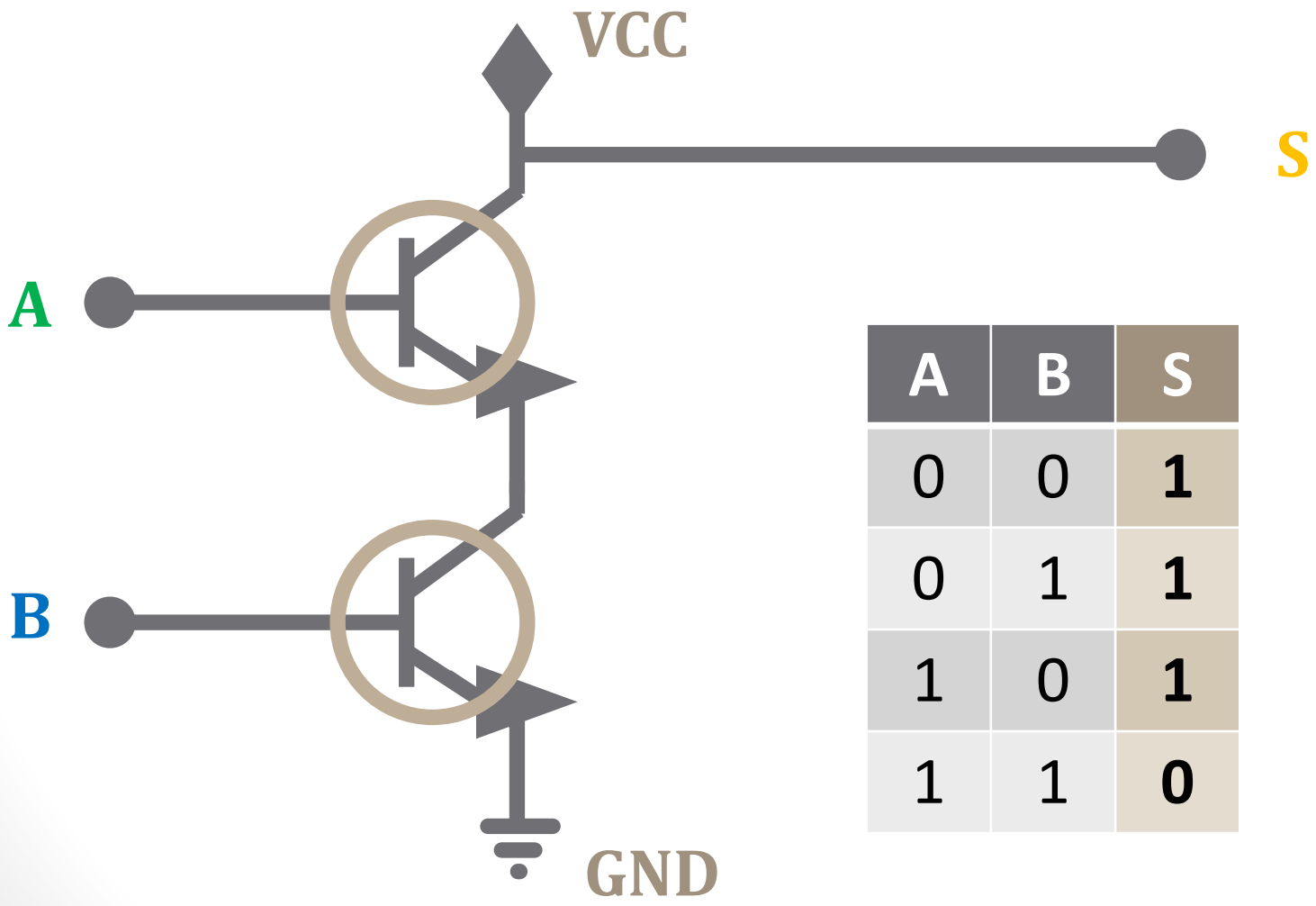


Транзистор – всему голова



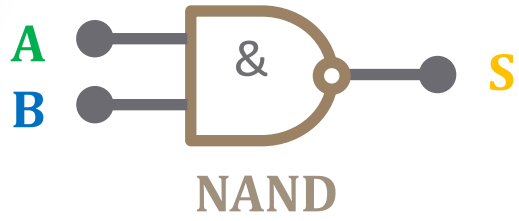
Транзистор – это кнопка, которая нажимается не пальцем, а подачей напряжения на **Базу**, после чего ток начинает протекать между **Коллектором** и **Эмиттером**.

NAND – основной базис

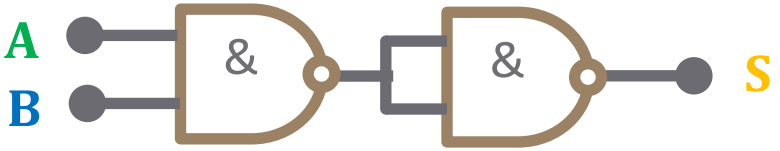


A	B	S
0	0	**1**
0	1	**1**
1	0	**1**
1	1	**0**

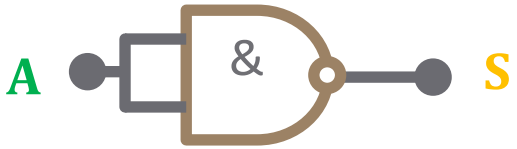
NOT AND OR XOR базис NAND



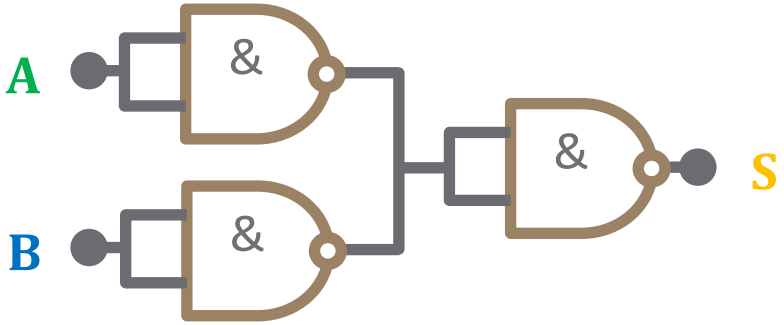
NAND



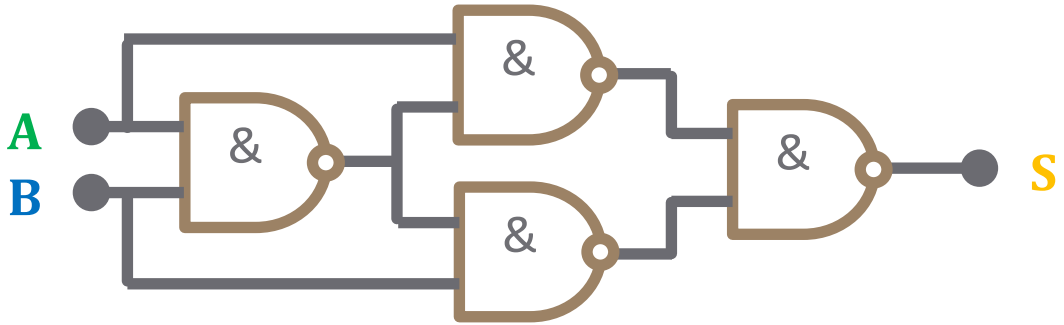
AND



NOT

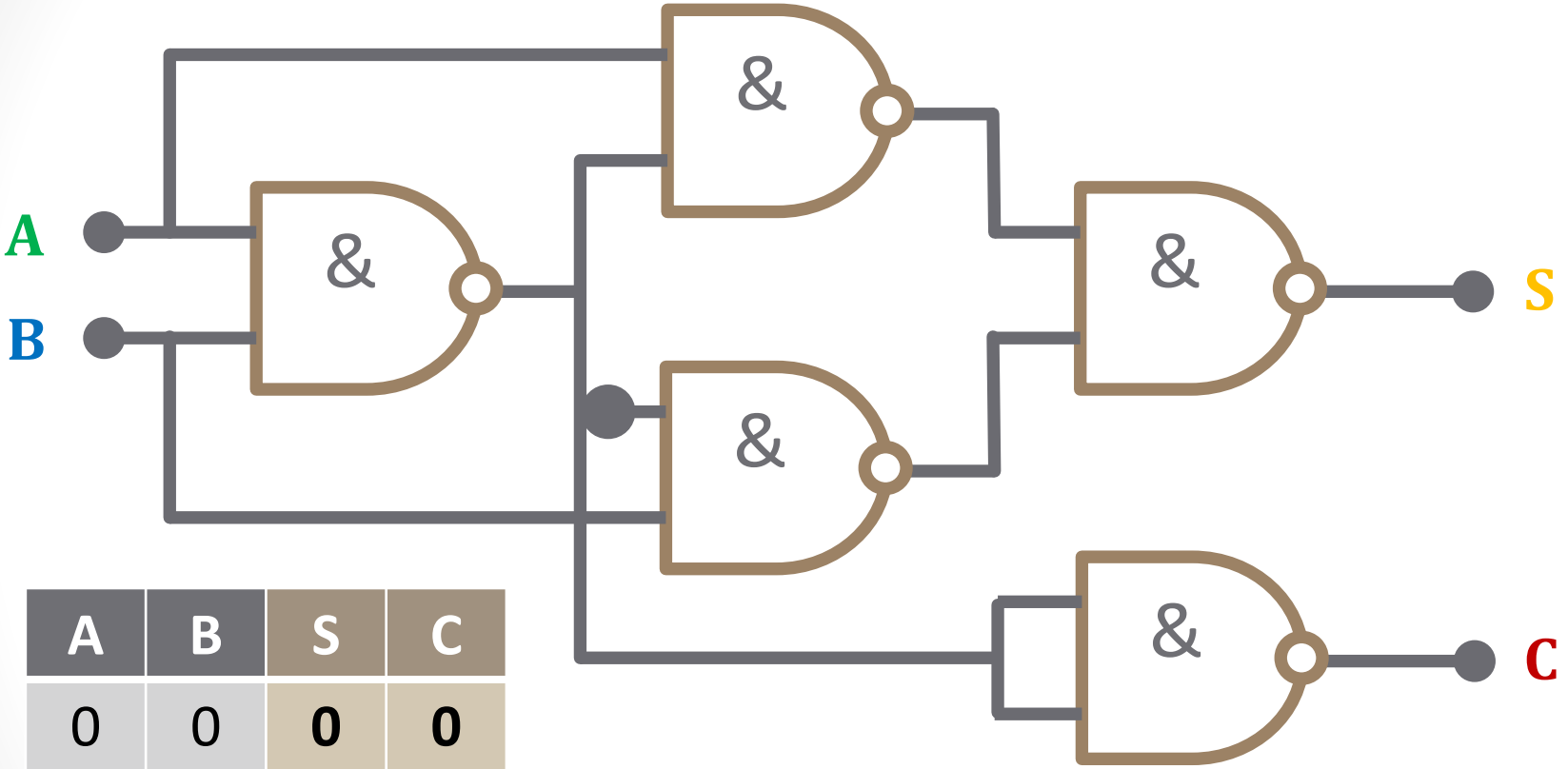


OR



XOR

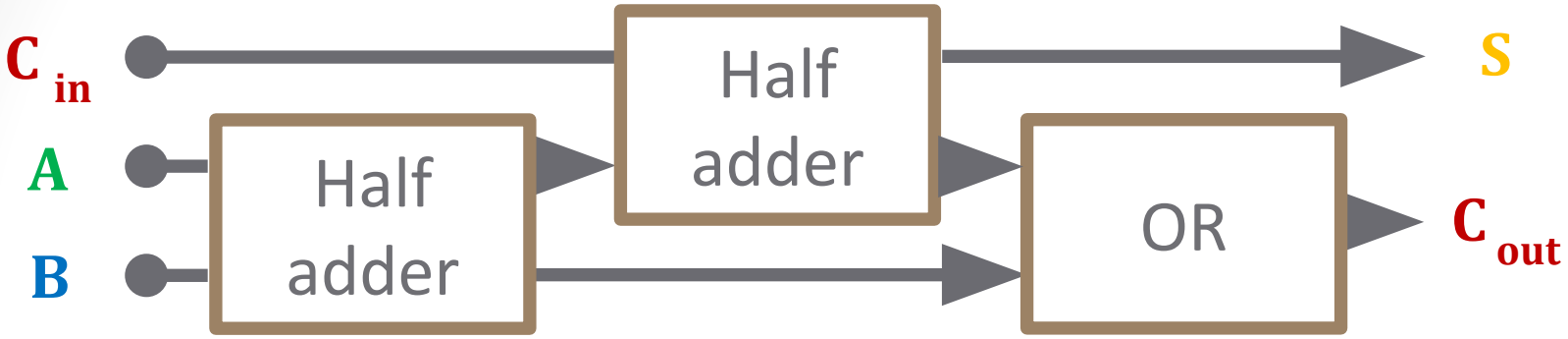
Half adder - полусумматор



A	B	S	C
0	0	**0**	**0**
0	1	**1**	**0**
1	0	**1**	**0**
1	1	**0**	**1**

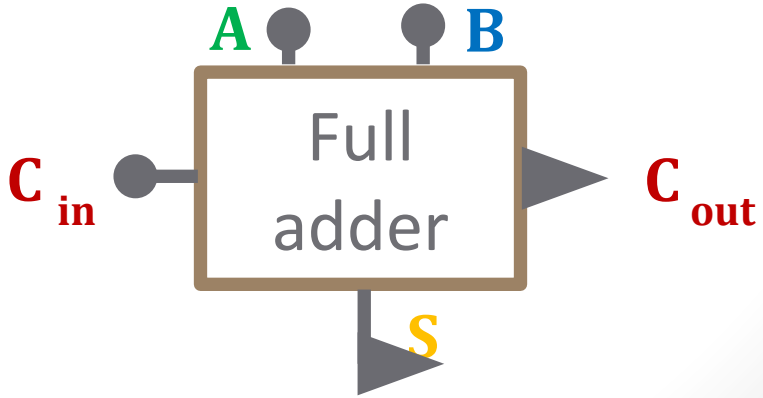
Полусумматор – суммирует два входящих бита, получая бит результата и бит переполнения.

Full adder - сумматор

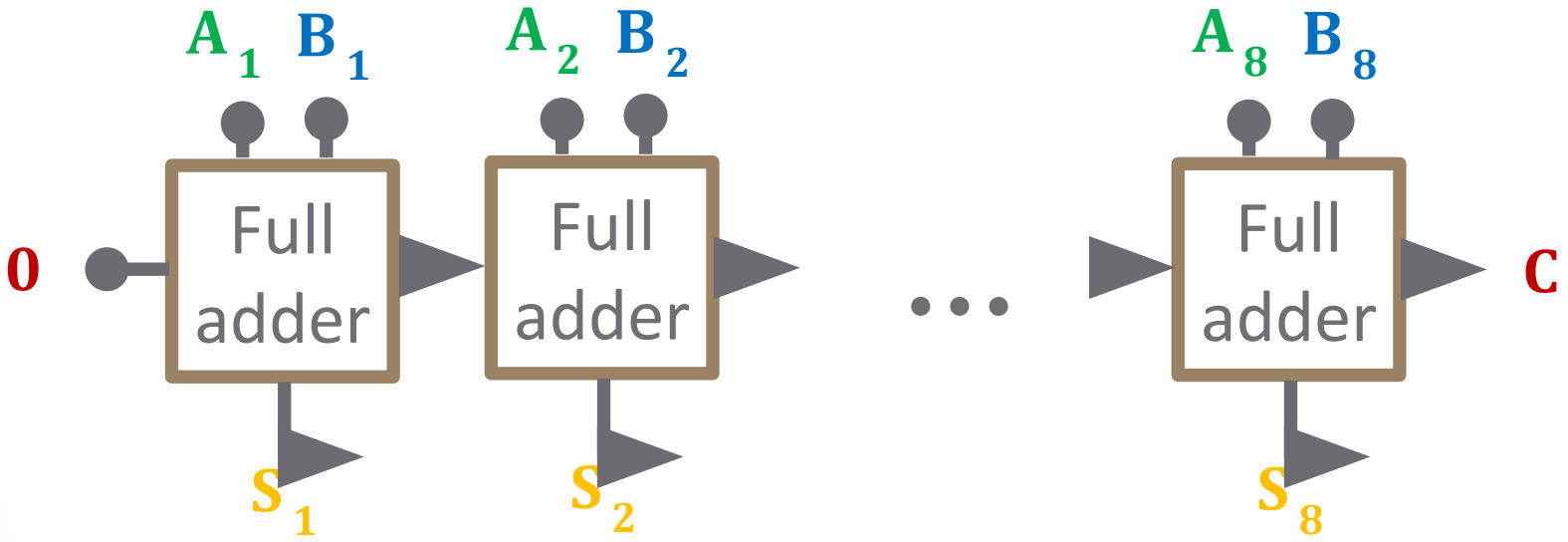


A	B	C_{in}	S	C_{out}
0	0	0	**0**	**0**
0	1	0	**1**	**0**
1	0	0	**1**	**0**
1	1	0	**0**	**1**
0	0	1	**1**	**0**
0	1	1	**0**	**1**
1	0	1	**0**	**1**
1	1	1	**1**	**1**

Сумматор – суммирует два бита и бит перехода, получая бит результата и бит переполнения.

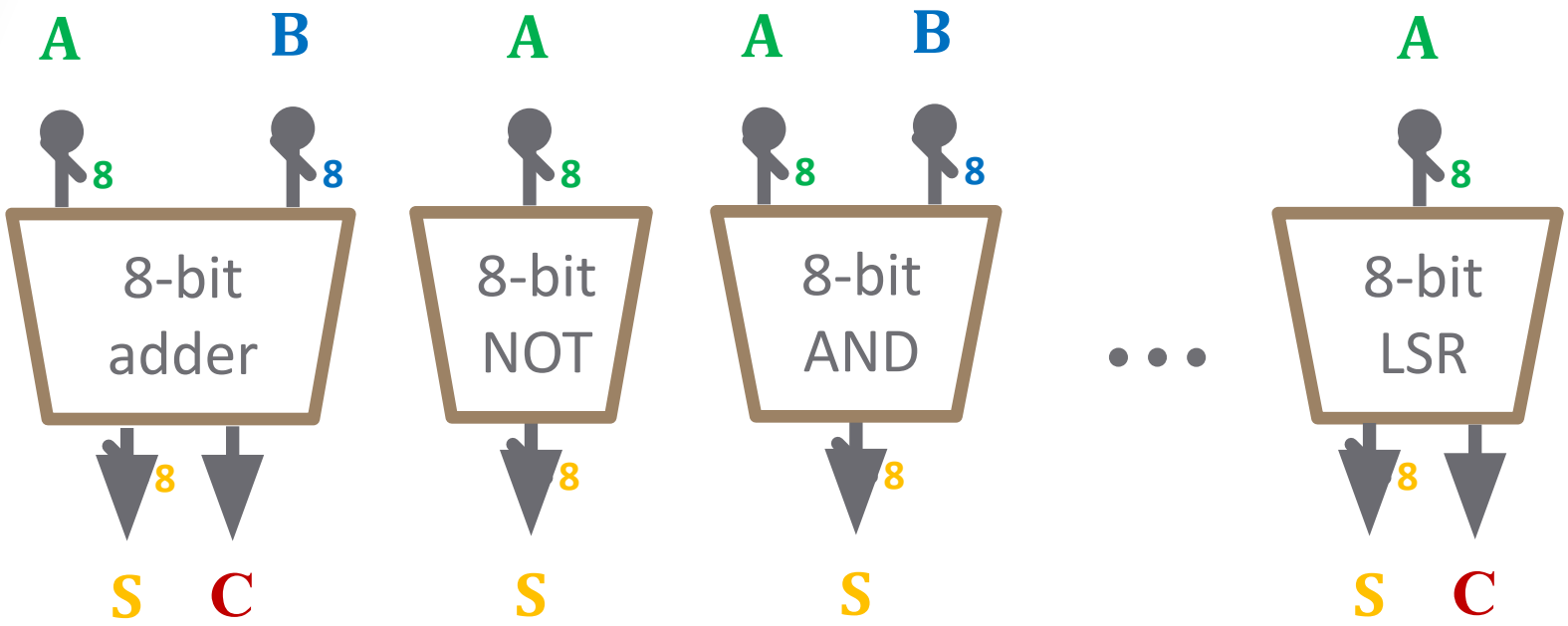


Полный 8 битный сумматор



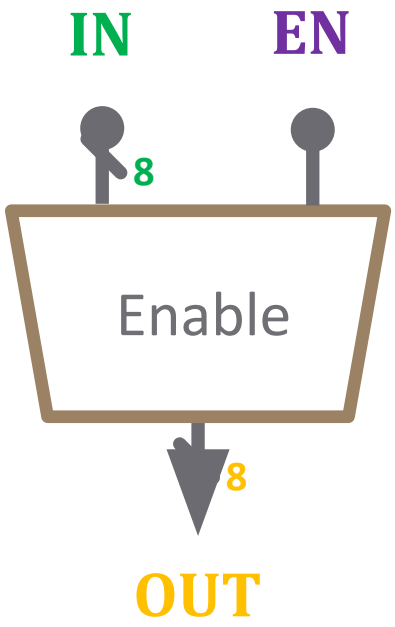
Итого: для создания полного 8 битного сумматора, основанного на базисе логических элементов NAND потребуется 2*((2*5+3)*8) = 208 транзисторов.

8 битные операции



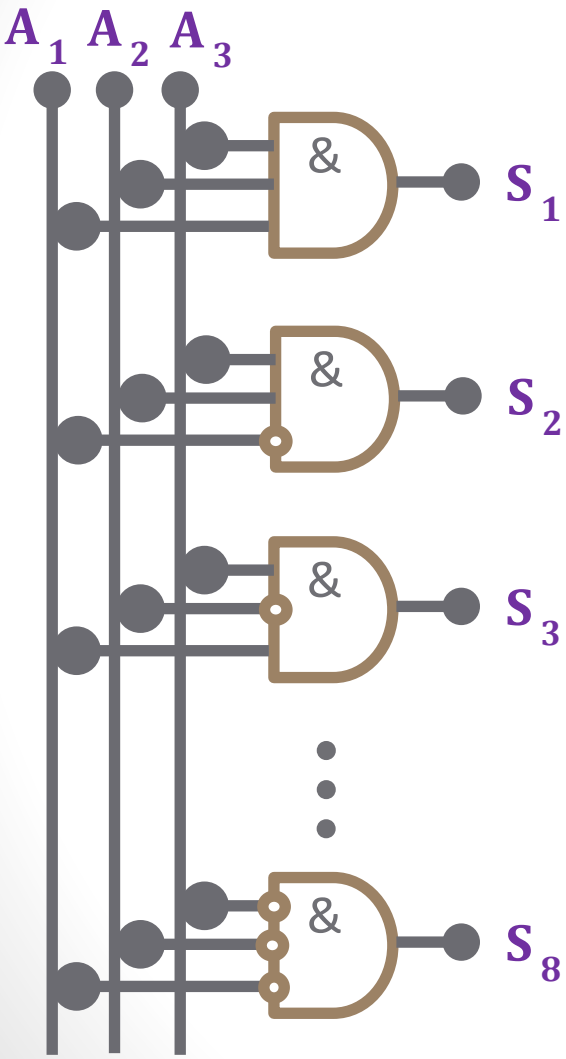
Проводники операндов (8 проводов каждый) можно подсоединить одновременно к блокам всех операций. **Итого:** на 16 входящих проводников, получиться по 8 или 9 проводников с каждой операции, которые объединить нельзя (монтажный OR).

Защелка выключатель



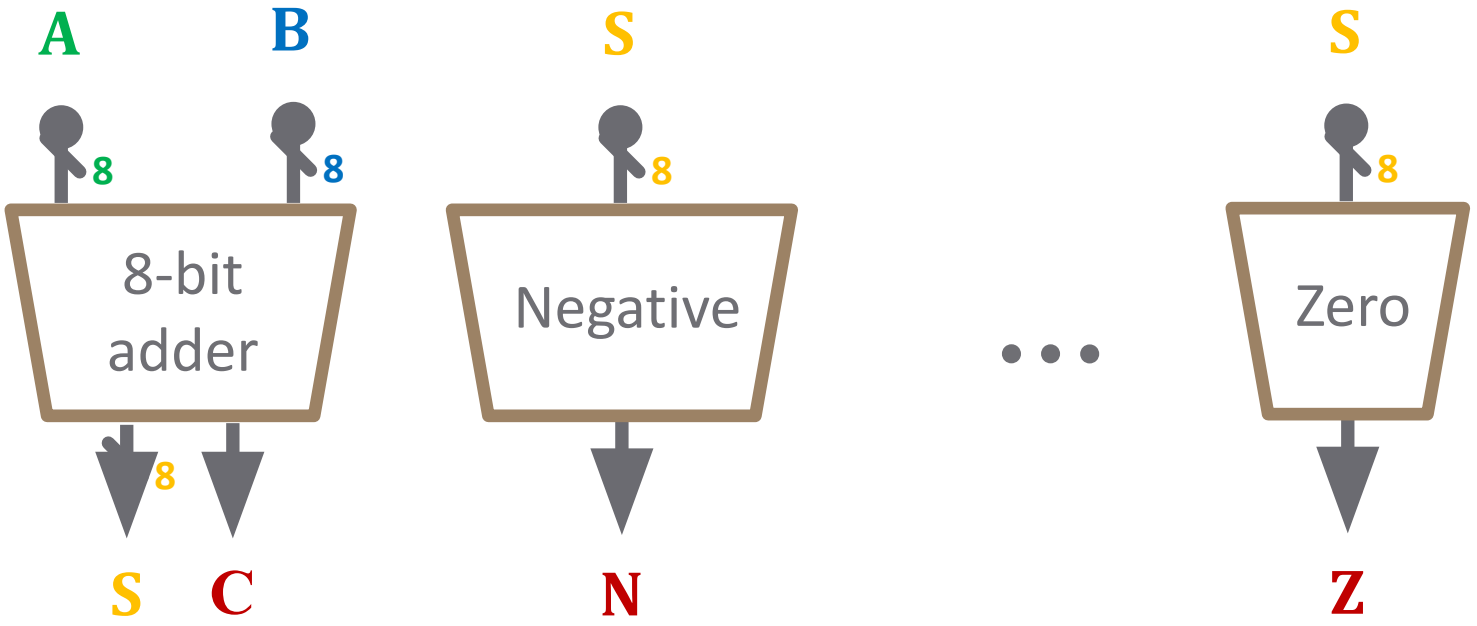
Добавив на выход каждого блока операции по выключателю, мы можем объединить все выходы получив 16 проводников входов и 8 + 1 выходов, плюс по одному управляющему проводнику на каждую операцию.

Дешифратор



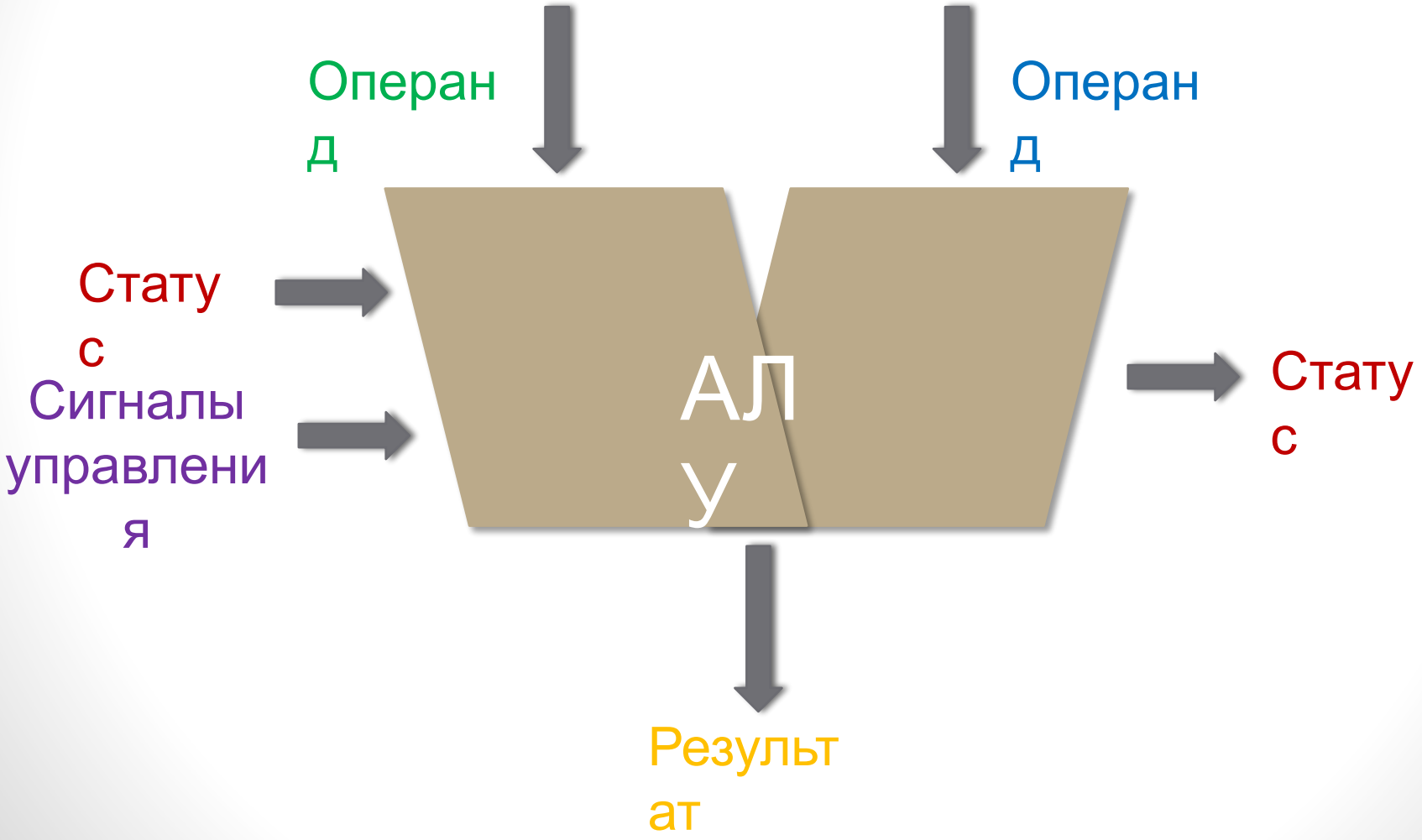
A_3	A_2	A_1	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8
0	**0**	**0**	0	0	0	0	0	0	0	**1**
0	**0**	**1**	0	0	0	0	0	0	**1**	0
0	**1**	**0**	0	0	0	0	0	**1**	0	0
0	**1**	**1**	0	0	0	0	**1**	0	0	0
1	**0**	**0**	0	0	0	**1**	0	0	0	0
1	**0**	**1**	0	0	**1**	0	0	0	0	0
1	**1**	**0**	0	**1**	0	0	0	0	0	0
1	**1**	**1**	**1**	0	0	0	0	0	0	0

Статус результата

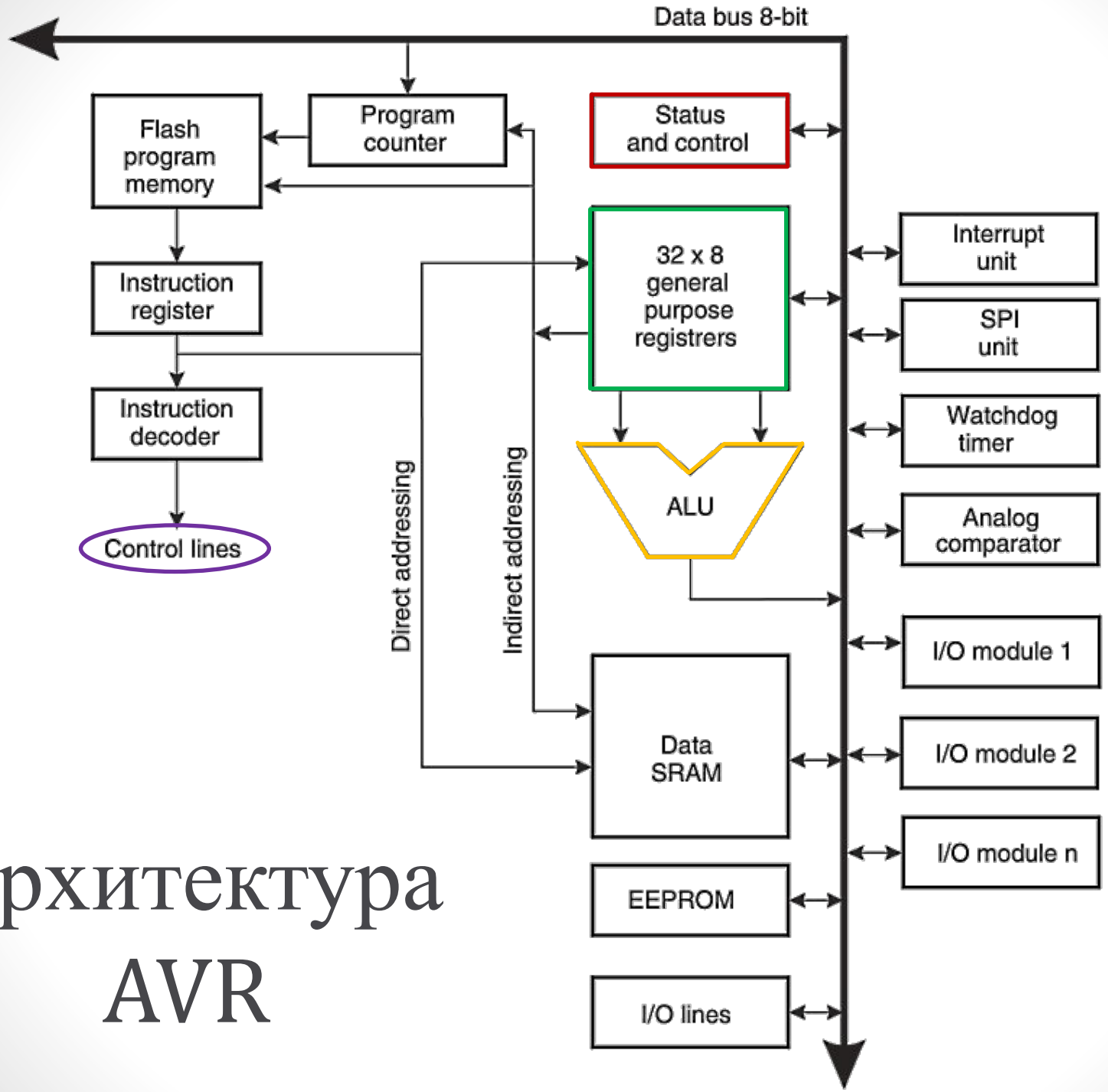


Из результата операции можно сразу же получить полезную информацию например (**C**) переполнение разряда при сложении или особый блок **Zero** который выполняет **XOR** между всеми 8 проводниками результата и если он равен 0 то Z = 1.

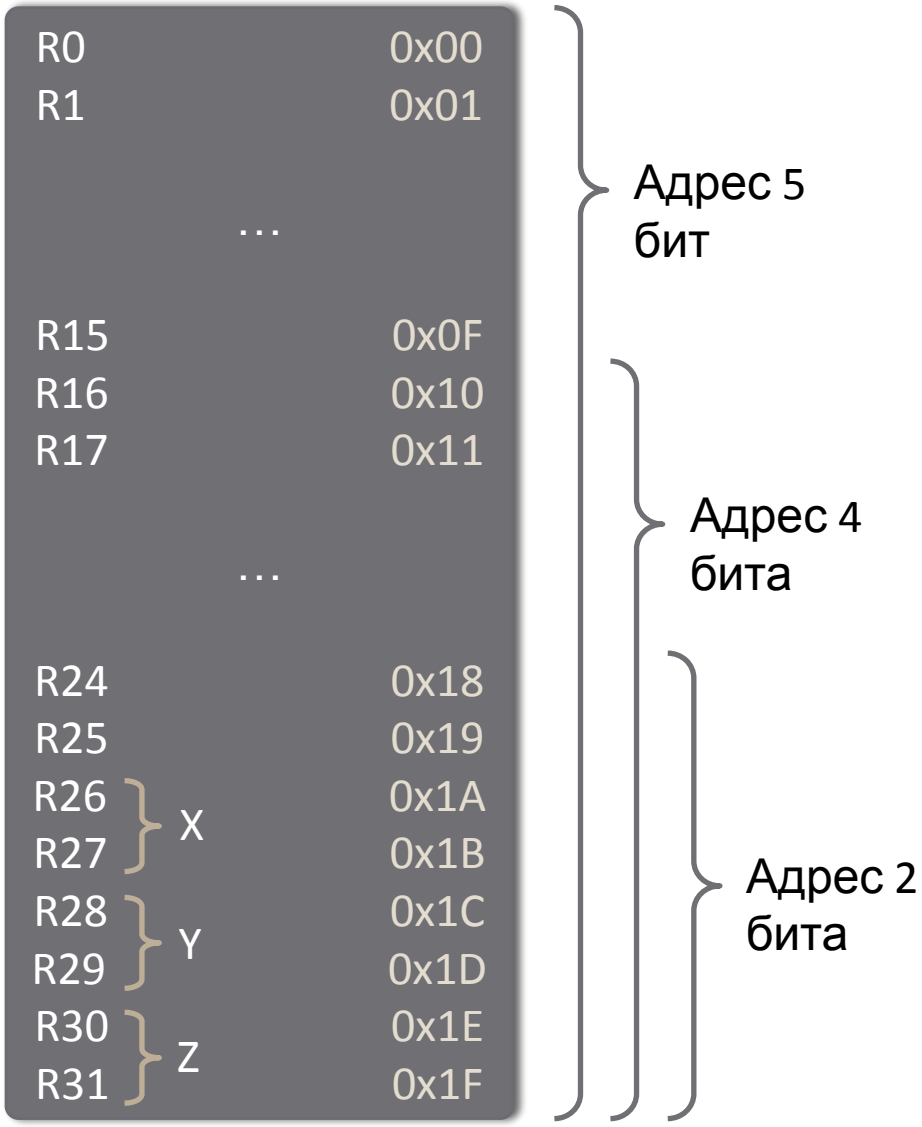
Арифметико-логическое устройство



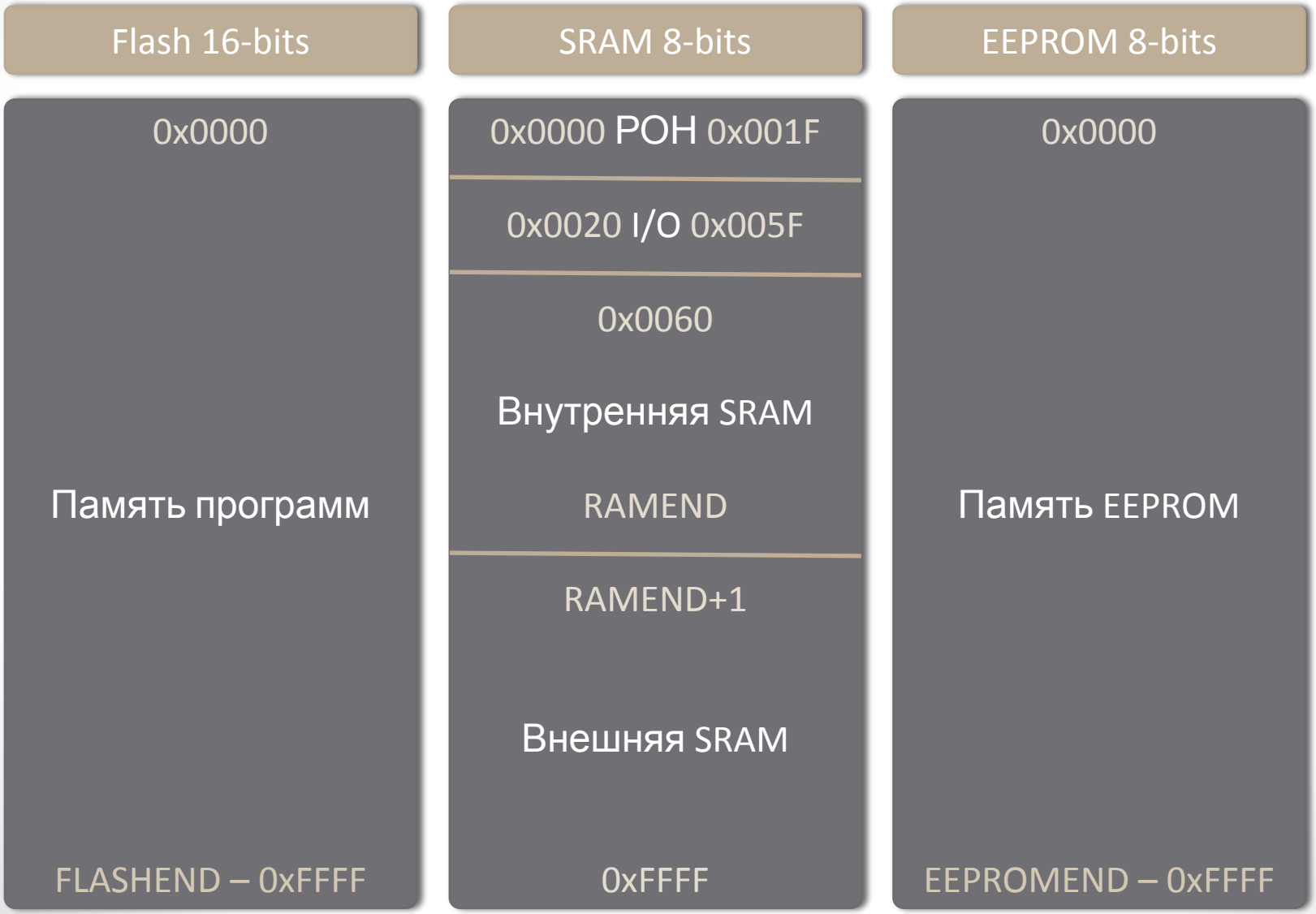
Архитектура AVR



Регистры процессора AVR



Память AVR


Flash 16-bits
SRAM 8-bits
EEPROM 8-bits
0x0000
Память программ
FLASHEND – 0xFFFF
0x0000 РОН 0x001F
0x0020 I/O 0x005F
0x0060
Внутренняя SRAM
RAMEND
RAMEND+1
Внешняя SRAM
0xFFFF
0x0000
Память EEPROM
EEPROMEND – 0xFFFF

NOP – Ничего не делать

Синтаксис:
NOP

Размер: 2 байта

0000	0000	0000	0000

Операнды: –

Счетчик: PC +=

Такты:

I	T	H	S	V	N[1]	Z	C
–	–	–	–	–	–	–	–

[1]

Определение: Операция выполняется вхолостую, ничего не происходит.

LDI – Загрузить значение в регистр

Синтаксис: LDI Rd, K

Размер: 2 байта

1110	KKKK	dddd	KKKK

Операнды: 16 ≤ d ≤ 31, 0 ≤ K ≤ 255

Счетчик: PC +=

Такты:

I	T	H	S	V	N[1]	Z	C
–	–	–	–	–	–	–	–

Определение: Загрузить непосредственное значение из кода операции в регистр Rd.

MOV – Копировать регистр

Синтаксис: MOV Rd, Rr

Размер: 2 байта

0010	11rd	dddd	rrrr

Операнды: 0 ≤ d ≤ 31, 0 ≤ r ≤ 31

Счетчик: PC += 1

Такты:

I	T	H	S	V	N	Z	C
–	–	–	–	–	–	–	–

Определение: Копирует содержимое одного регистра в другой регистр. Исходный регистр Rr остается неизменным, в регистр назначения Rd загружается копия содержимого регистра Rr.

ADD – Сложить без переноса

Синтаксис: ADD Rd, Rr

Размер: 2 байта

0000	11rd	dddd	rrrr

Операнды: 0 ≤ d ≤ 31, 0 ≤ r ≤ 31

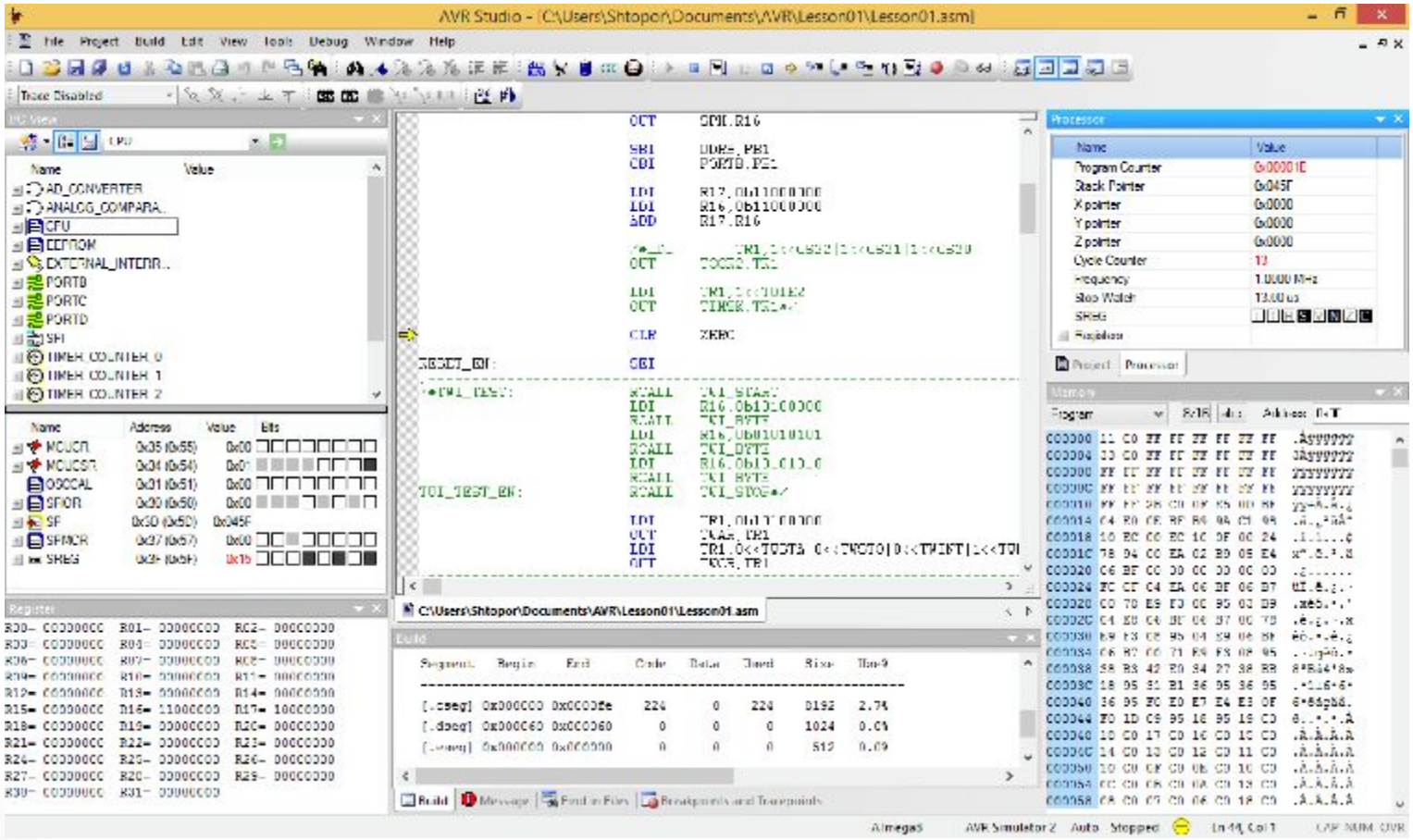
Счетчик: PC +=

Такты:

I	T	H	S	V	N[1]	Z	C
–	–	+	+	+	+	+	+

Определение: Сложение двух регистров без добавления содержимого флага переноса (C), размещение результата в регистре назначения Rd.

AVR Studio 4



http://www.atmel.com/ru/ru/tools/STUDIOARCHIVE.aspx

INC – Инкрементировать

Синтаксис: INC Rd

Размер: 2 байта

1001	010d	dddd	0011

Операнды: 0 ≤ d ≤ 31

Счетчик: PC += 1

Такты:

I	T	H	S	V	N[1]	Z	C
–	–	–	+	+	+	+	–

Определение: Добавление единицы к содержимому регистра Rd и размещение результата в регистре назначения Rd.

DEC – Декрементировать

Синтаксис: DEC Rd

Размер: 2 байта

1001	010d	dddd	1010

Операнды: 0 ≤ d ≤ 31

Счетчик: PC +=

Такты:

I	T	H	S	V	N[1]	Z	C
–	–	–	+	+	+	+	–

[1]

Определение: Вычитание единицы из содержимого регистра Rd и размещение результата в регистре назначения Rd.

SUB – Вычесть без переноса

Синтаксис: SUB Rd, Rr

Размер: 2 байта

0001	10rd	dddd	rrrr

Операнды: 0 ≤ d ≤ 31, 0 ≤ r ≤ 31

Счетчик: PC +=

Такты:

I	T	H	S	V	N[1]	Z	C
–	–	+	+	+	+	+	+

Определение: Вычитание содержимого регистра-источника Rr из содержимого регистра Rd, размещение результата в регистре назначения Rd.

SUBI – Вычесть значение из регистра

Синтаксис: SUBI Rd, K

Размер: 2 байта

0101	KKKK	dddd	KKKK

Операнды: 0 ≤ d ≤ 31, 0 ≤ K ≤ 255

Счетчик: PC += 1

Такты:

I	T	H	S	V	N[1]	Z	C
–	–	+	+	+	+	+	+

Определение: Вычитание константы из содержимого регистра, размещение результата в регистре назначения Rd.