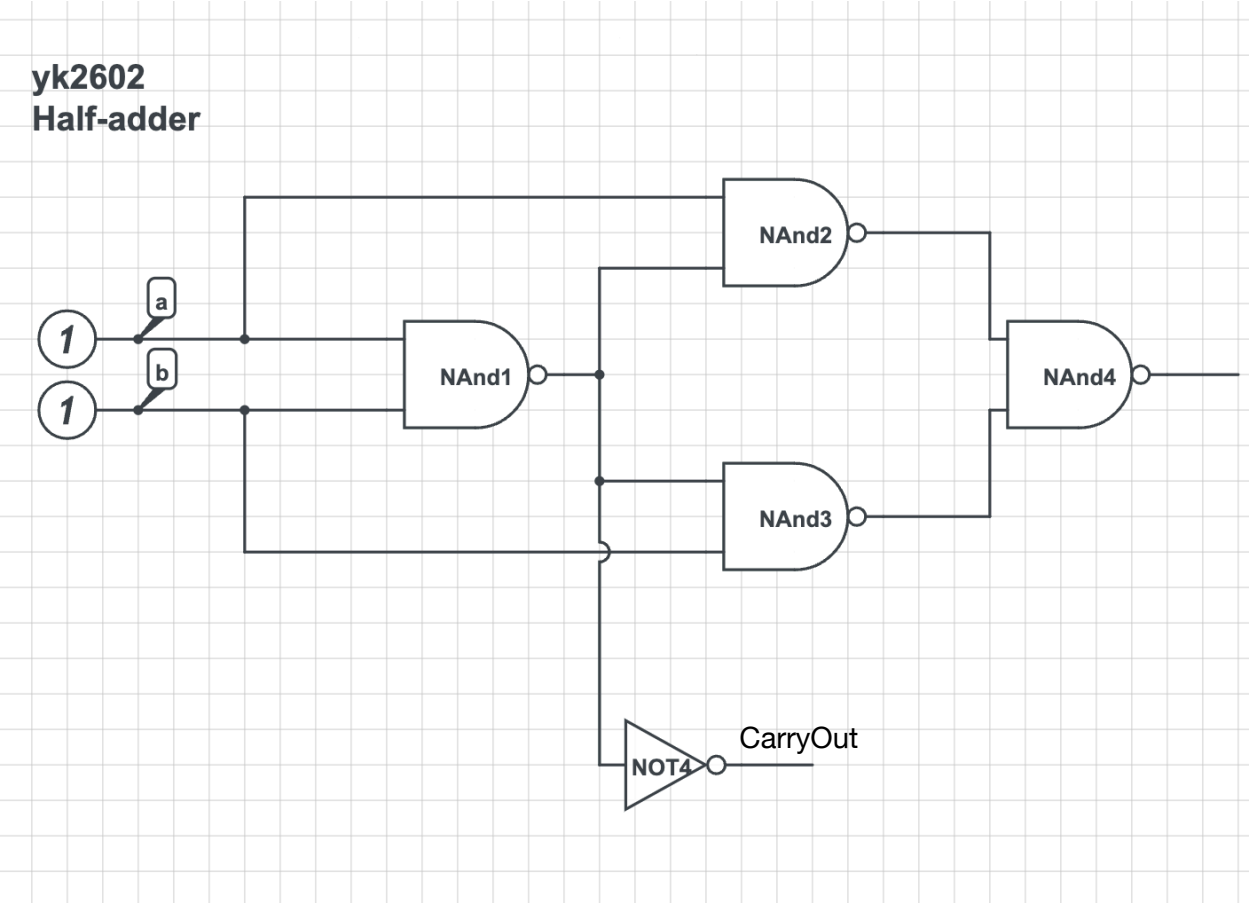
Yan Konichshev: yk2602
Computer Architecture: homework 2
Date: February 26, 2023

Exercise 1

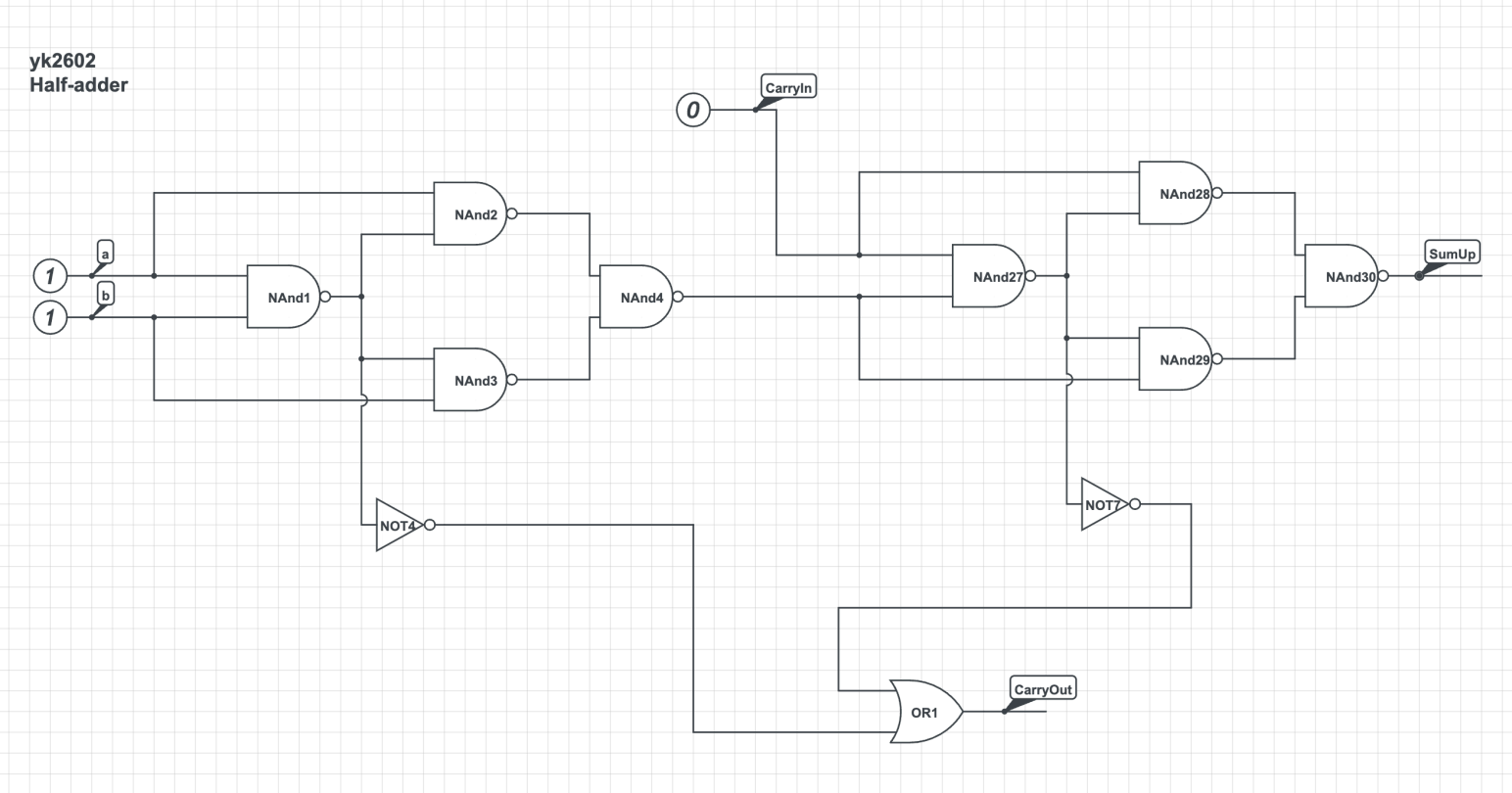
Part A - Half Adder



Part B

The output value of the CarryOut is 5V (meaning it is logical 1)

Part C - Full Adder



Part D

For the detailed implementation of the 4-bit adder refer to the ***Appendix, Item A*** *at the end of this document*

Part E

CarryIn: 0

Sum0: 0
Sum1: 0
Sum2: 0
Sum3: 0
CarryOut: 1

Part F

CarryIn: 1

Sum0: 1
Sum1: 0
Sum2: 0
Sum3: 0
CarryOut: 1

Exercise 2

a) 0000 0000 0000 0000 (two) = 0 (ten)
b) 1000 0000 0000 0000 (two) = -32768 (ten)
c) 1111 1111 1111 1111 (two) = -32767 (ten), where orange color is the magnitude and MSB is the sign

Exercise 3

a) 0000 0000 0000 0000 0000 0000 0000 0000 (two) = 0 (ten)
b) 1000 0000 0000 0000 0000 0000 0000 0000 (two) = -4 294 967 296 (ten)
c) 1111 1111 1111 1111 1111 1111 1111 1111 = -4 294 967 295 (ten), where orange color is the magnitude and MSB is the sign

Exercise 4

a) 1010 (two) = 10 (ten)
b) 110110 (two) = 0011 0110 (two) = 54 (ten)
c) 1111 0000 (two) = 240 (ten)
d) 000 1000 1010 0111 (two) = 167 + 2048 = 2215 (ten)

Exercise 5

a) 1110 (two) = 14 (ten)
b) 100100 (two) = 36 (ten)
c) 1101 0111 (two) = 215 (ten)
d) 011 1010 1010 0100 (two) = 15012 (ten)

Exercise 6

a) 1010 (two) = 0A (hex)
b) 0011 0110 (two) = 36 (hex)
c) 1111 0000 (two) = F0 (hex)
d) 0000 1000 1010 0111 (two) = 08 A7 (hex)

e) 1110 (two) = 0E (hex)
f) 0010 0100 (two) = 24 (hex)
g) 1101 0111 (two) = D7 (hex)
h) 0011 1010 1010 0100 (two) = 3A A4 (hex)

Exercise 7

** all the following conversions are based on the assumption that the intermediary binary numbers are **unsigned**, since there was no particular instruction*

a) A5 (hex) = 165 (ten)
b) 3B (hex) = 59 (ten)
c) 7C (hex) = 124 (ten)
d) FFFF (hex) = 65535 (ten)
e) D0000(hex) = 851968(ten)
f) ED3A (hex) = 60730 (ten)

Exercise 8

a) A5 (hex) = 1010 0101 (two)
b) 3B (hex) = 0011 1011 (two)
c) 7C (hex) = 0111 1100 (two)
d) FFFF (hex) = 1111 1111 1111 1111 (two)
e) D0000(hex) = 1101 0000 0000 0000 0000(two)
f) ED3A (hex) = 1110 1101 0011 1010 (two)

Exercise 9

a) 1010 (bin) = −8 + 2 = −6(ten)
b) 110110 (bin) = −32 + 16 + 4 + 2 = −10(ten)
c) 01110000 (bin) = 112
d) 10011111 (bin) = -97

Exercise 10

a) 24 (ten) = 00011000
b) −59 (ten) = 11000101 (since it is negative, I first found the binary representation for the 59, inverted all the bits and added 1 to the least significant bit)
c) 128 (ten) = overflows (maximum value using 8 bits is 127)
d) −150 (ten) = overflows (minimum value using 8 bits is -128)
e) 127 (ten) = 0111 1111(two)

Exercise 11

Num of rows: 2^8
Num of cols: 2^9

Total num of bits = 2^8 * 2^9

2^8 = 256
2^9 = 512

256 * 512 is roughly 300 * 500 = 150 000 bits of memory

Exercise 12

Part A: To find the screenshots of me playing around with the pins, navigate to the folder exercise_12a

Part B: To find the screenshots of me playing around with the pins, navigate to the folder exercise_12b

Part C: the code for the SYNC module is SYNC, so I was able to identify and find it. In my understanding what it does is that it is tracing whenever the microprocessing is fetching any instructions, and then it is activated. Probably once it is activated, the synchronization of the instruction set is happening. Here is the screenshot of the table, where I traced the instruction sequence with additional column called "sync":

cycle	ab	db	rw	Fetch	pc	a	x	y	s	p	Execute	State	sync
755	0005	4c	1	JMP Abs	0005	51	1b	e5	fd	nv-BdIzc	RTS	T1	1
755	0005	4c	1	JMP Abs	0005	51	1b	e5	fd	nv-BdIzc	RTS	T1	1
756	0006	02	1		0006	51	1b	e5	fd	nv-BdIzc	JMP Abs	T2	0
756	0006	02	1		0006	51	1b	e5	fd	nv-BdIzc	JMP Abs	T2	0
757	0007	00	1		0007	51	1b	e5	fd	nv-BdIzc	JMP Abs	T0	0
757	0007	00	1		0007	51	1b	e5	fd	nv-BdIzc	JMP Abs	T0	0
758	0002	20	1	JSR Abs	0002	51	1b	e5	fd	nv-BdIzc	JMP Abs	T1	1
758	0002	20	1	JSR Abs	0002	51	1b	e5	fd	nv-BdIzc	JMP Abs	T1	1
759	0003	10	1		0003	51	1b	e5	fd	nv-BdIzc	JSR Abs	T2	0
759	0003	10	1		0003	51	1b	e5	fd	nv-BdIzc	JSR Abs	T2	0
760	01fd	00	1		0004	51	1b	e5	10	nv-BdIzc	JSR Abs	T3	0
760	01fd	00	1		0004	51	1b	e5	10	nv-BdIzc	JSR Abs	T3	0

Appendix: Item A, 4-bit Ripple Carry Adder