CSEE W3827

Fundamentals of Computer Systems

Homework Assignment 1

Profs. Stephen A. Edwards & Martha Kim

Columbia University

Due September 20th, 2012 at 10:00 AM

Fill in this PDF form using *Adobe Acrobat Reader*, save it as PDF, and submit it through CourseWorks.

Do not use Preview or another PDF viewer, which is unlikely to work correctly.

Name: Answer Key

Uni:

1. (5 pts.) What are the values, in decimal, of the following bytes if they are interpreted as 8-bit numbers in

	00001011	10001110
binary	11	142
one’s complement	11	-113
two’s complement	11	-114

2. (5 pts.) Show how to perform $5 + -10 = -5$ in 5-bit

Signed-magnitude numbers

$$\begin{array}{r} \boxed{00101} \\ + \boxed{11010} \\ \hline \boxed{10101} \end{array}$$

One’s Complement numbers

$$\begin{array}{r} \boxed{00101} \\ + \boxed{10101} \\ \hline \boxed{11010} \end{array}$$

Two’s Complement numbers

$$\begin{array}{r} \boxed{00101} \\ + \boxed{10110} \\ \hline \boxed{11011} \end{array}$$

3. (10 pts.) Complete the truth table for the following Boolean functions:

$$a = XY + \overline{X}\,\overline{Y}Z + X\overline{Z}$$
$$b = (X+Y)(Y+\overline{Z})(\overline{X}+Z)$$

X	Y	Z	a	b
0	0	0	0	0
0	0	1	1	0
0	1	0	0	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	1	0
1	1	1	1	1

4. (30 pts.) Consider the function *F*, whose truth table is below.

X	**Y**	**Z**	**F**
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

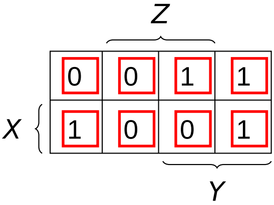
(a) Write *F* as a sum of minterms. Use a leading “!” to denote negation. Two have been done for you.

!XY!Z + !XYZ + X!Y!Z + XY!Z

(b) Write *F* as a product of maxterms. Two have been done for you.

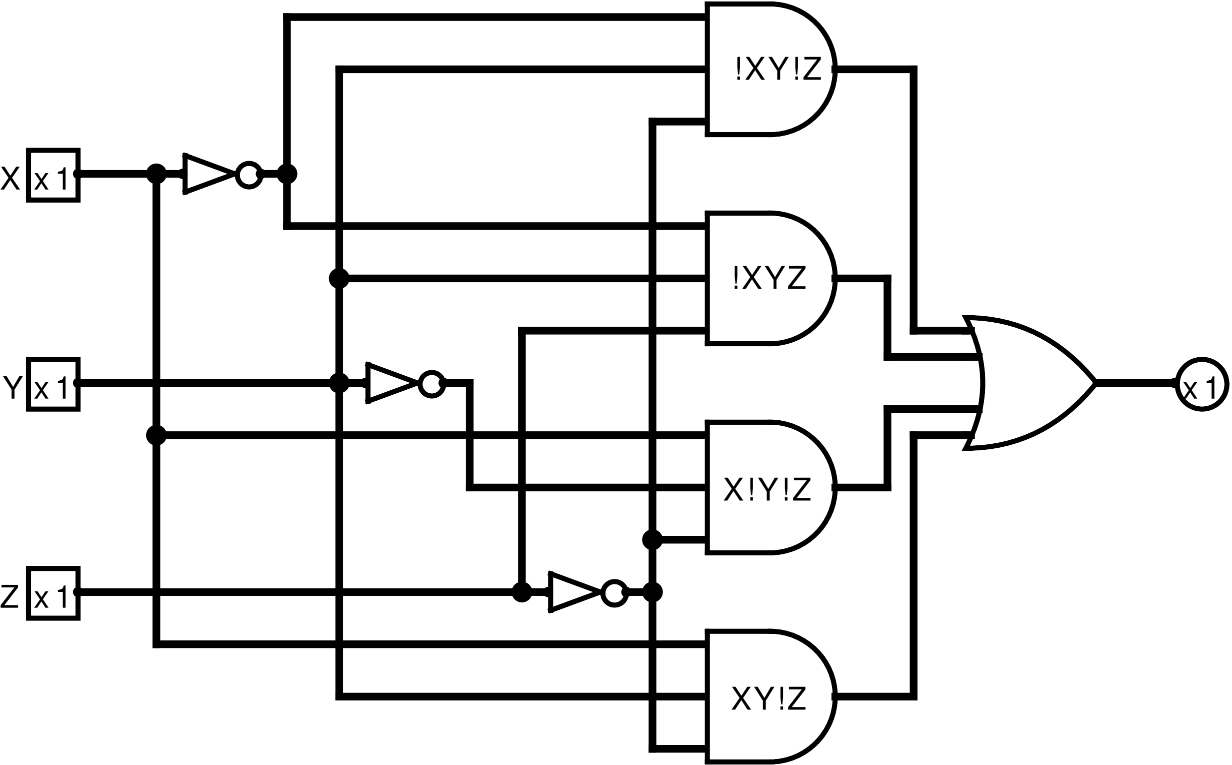
(X+Y+Z)(X+Y+!Z)(!X+Y+!Z)(!X+!Y+!Z)

(c) Fill in the Karnaugh map for *F* as shown below. You do not have to simplify it.

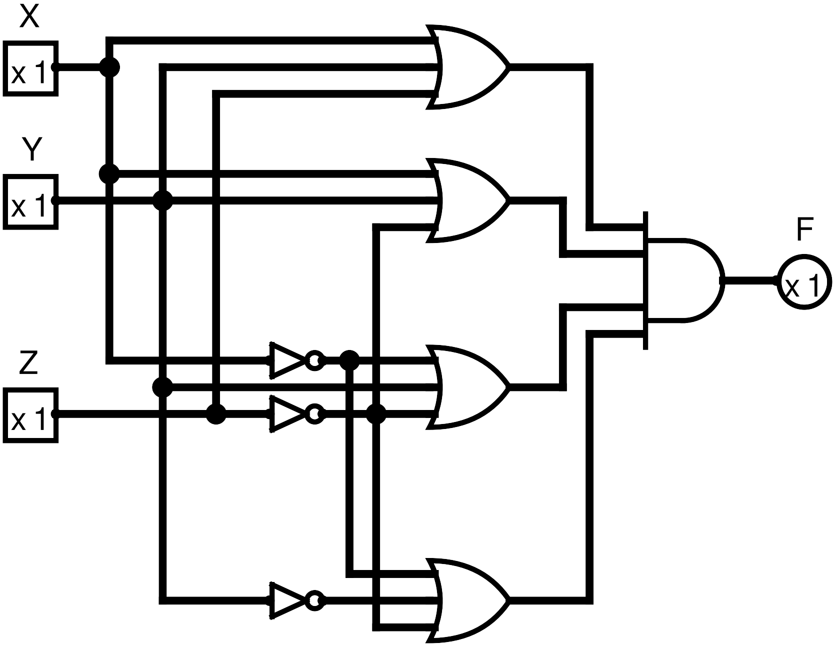


0	0	1	1
1	0	0	1

hw1−4d



hw1–4e



In Logisim,

(d) Implement the circuit corresponding to your minterms expression of *F*. Make sure to label the three inputs “X,” “Y,” and “Z” and the output “F.” Verify your circuit using Logisim’s Combinational Analysis feature (Project→Analyze Circuit).

Save your solution as “hw1-4d.circ” and upload it to CourseWorks along with your filled-in PDF file.

(e) Implement the circuit corresponding to your maxterms expression of *F*. Again, verify your work.

Save your solution as “hw1-4e.circ” and upload it.

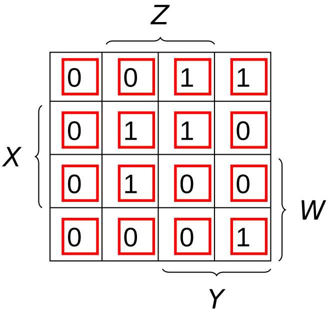
5. (10 pts.) Consider the function *F* whose truth table is shown below

W	**X**	**Y**	**Z**	**F**
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	0

(a) Write the function *F* in sum-of-minterms form. Two are given.

!W!XY!Z+!W!XYZ+!WX!YZ+!WXYZ+W!XY!Z+WX

(b) Fill in this Karnaugh map for *F*



0	0	1	1
0	1	1	0
0	1	0	0
0	0	0	1

(c) Use your Karnaugh map to write a minimal sum-of-products representation for *F*

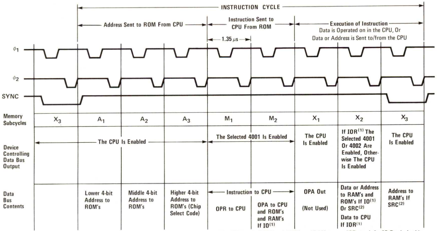
X!YZ + !WYZ + !XY!Z

6. (40 pts.) Intel’s 4004 microprocessor, released in 1971, was the first single-chip microprocessor. Most instructions were 8 bits long, but certain ones (JCN, FIM, JUN, JMS, and ISZ) took 16 bits.

   Create a minimal circuit in Logisim that decodes the first byte of the instruction and generates a “1” only when the instruction has a second byte.

   Name your eight inputs R3, R2, R1, R0, A3, A2, A1, A0, which represent the bits of the first “OPR” and “OPA” words mentioned in the attached data sheet. Have it generate a single output named “TWO.”

   Name your solution “hw1-6.circ” and submit it via Courseworks.


INSTRUCTION CYCLE
Address Sent to ROM From CPU
Instruction Sent to CPU From ROM
Execution of Instruction
Data is Operated on in the CPU, Or Data or Address is Sent to/from the CPU
1.35 μs
φ1
φ2
SYNC
Memory Subcycles
X3
A1
A2
A3
M1
M2
X1
X2
X3
Device Controlling Data Bus Output
The CPU Is Enabled
The Selected 4001 Is Enabled
The CPU Is Enabled
If IOR(1) The Selected 4001 Or 4002 Are Enabled, Otherwise The CPU Is Enabled
The CPU Is Enabled
Data Bus Contents
Lower 4-bit Address to ROM's
Middle 4-bit Address to ROM's
Higher 4-bit Address to ROM's (Chip Select Code)
Instruction to CPU
OPR to CPU
OPA to CPU and ROM's and RAM's If IO(1)
OPA Out
(Not Used)
Data or Address to RAM's and ROM's If IO(1) Or SRC(2)
Data to CPU If IOR(1)
Address to RAM's If SRC(2)

(1) IO instructions control the flow of information between accumulator in CPU, I/O lines in ROM's and RAM's and RAM storage. IOR stands for IO Read. In this case the CPU will receive data from RAM storage locations or I/O input lines of 4001's.

(2) The SRC instruction designates the chip number and address for a following IO instruction.

Figure 2. MCS-4 Basic Instruction Cycle

Instruction Set

[Those instructions preceded by an asterisk (*) are 2 word instructions that occupy 2 successive locations in ROM]

MACHINE INSTRUCTIONS

MNEMONIC	OPR $D_3 D_2 D_1 D_0$	OPA $D_3 D_2 D_1 D_0$	DESCRIPTION OF OPERATION
NOP	0 0 0 0	0 0 0 0	No operation.
*JCN	0 0 0 1 $A_2 A_2 A_2 A_2$	$C_1 C_2 C_3 C_4$ $A_1 A_1 A_1 A_1$	Jump to ROM address A_2 A_2 A_2 A_2, A_1 A_1 A_1 A_1 (within the same ROM that contains this JCN instruction) if condition C_1 C_2 C_3 C_4(1) is true, otherwise skip (go to the next instruction in sequence).
*FIM	0 0 1 0 $D_2 D_2 D_2 D_2$	R R R 0 $D_1 D_1 D_1 D_1$	Fetch immediate (direct) from ROM Data D_2, D_1 to index register pair location RRR.(2)
SRC	0 0 1 0	R R R 1	Send the address (contents of index register pair RRR) to ROM and RAM at X_2 and X_3 time in the Instruction Cycle.
FIN	0 0 1 1	R R R 0	Fetch indirect from ROM. Send contents of index register pair location 0 out as an address. Data fetched is placed into register pair location RRR at A_1 and A_2 time in the Instruction Cycle.
JIN	0 0 1 1	R R R 1	Jump indirect. Send contents of register pair RRR out as an address at A_1 and A_2 time in the Instruction Cycle.
*JUN	0 1 0 0 $A_2 A_2 A_2 A_2$	$A_3 A_3 A_3 A_3$ $A_1 A_1 A_1 A_1$	Jump unconditional to ROM address A_3, A_2, A_1.
*JMS	0 1 0 1 $A_2 A_2 A_2 A_2$	$A_3 A_3 A_3 A_3$ $A_1 A_1 A_1 A_1$	Jump to subroutine ROM address A_3, A_2, A_1, save old address. (Up 1 level in stack.)
INC	0 1 1 0	R R R R	Increment contents of register RRRR. (3)
*ISZ	0 1 1 1 $A_2 A_2 A_2 A_2$	R R R R $A_1 A_1 A_1 A_1$	Increment contents of register RRRR. Go to ROM address A_2, A_1 (within the same ROM that contains this ISZ instruction) if result ≠0, otherwise skip (go to the next instruction in sequence).
ADD	1 0 0 0	R R R R	Add contents of register RRRR to accumulator with carry.
SUB	1 0 0 1	R R R R	Subtract contents of register RRRR to accumulator with borrow.
LD	1 0 1 1	R R R R	Load contents of register RRRR to accumulator.
XCH	1 0 1 1	R R R R	Exchange contents of index register RRRR and accumulator.
BBL	1 1 0 0	D D D D	Branch back (down 1 level in stack) and load data DDDD to accumulator.
LDM	1 1 0 1	D D D D	Load data DDDD to accumulator.

See Notes on Page 6.

Continued on page 6.

INPUT/OUTPUT AND RAM INSTRUCTIONS

(The RAM's and ROM's operated on in the I/O and RAM instructions have been previously selected by the last SRC instruction executed.)

MNEMONIC	OPR $D_3 D_2 D_1 D_0$	OPA $D_3 D_2 D_1 D_0$	DESCRIPTION OF OPERATION
WRM	1 1 1 0	0 0 0 0	Write the contents of the accumulator into the previously selected RAM main memory character.
WMP	1 1 1 0	0 0 0 1	Write the contents of the accumulator into the previously selected RAM output port. (Output Lines)
WRR	1 1 1 0	0 0 1 0	Write the contents of the accumulator into the previously selected ROM output port. (I/O Lines)
WRϕ[4]	1 1 1 0	0 1 0 0	Write the contents of the accumulator into the previously selected RAM status character 0.
WR1[4]	1 1 1 0	0 1 0 1	Write the contents of the accumulator into the previously selected RAM status character 1.
WR2[4]	1 1 1 0	0 1 1 0	Write the contents of the accumulator into the previously selected RAM status character 2.
WR3[4]	1 1 1 0	0 1 1 1	Write the contents of the accumulator into the previously selected RAM status character 3.
SBM	1 1 1 0	1 0 0 0	Subtract the previously selected RAM main memory character from accumulator with borrow.
RDM	1 1 1 0	1 0 0 1	Read the previously selected RAM main memory character into the accumulator.
RDR	1 1 1 0	1 0 1 0	Read the contents of the previously selected ROM input port into the accumulator. (I/O Lines)
ADM	1 1 1 0	1 0 1 1	Add the previously selected RAM main memory character to accumulator with carry.
RDϕ[4]	1 1 1 0	1 1 0 0	Read the previously selected RAM status character 0 into accumulator.
RD1[4]	1 1 1 0	1 1 0 1	Read the previously selected RAM status character 1 into accumulator.
RD2[4]	1 1 1 0	1 1 1 0	Read the previously selected RAM status character 2 into accumulator.
RD3[4]	1 1 1 0	1 1 1 1	Read the previously selected RAM status character 3 into accumulator.

ACCUMULATOR GROUP INSTRUCTIONS

CLB	1 1 1 1	0 0 0 0	Clear both. (Accumulator and carry)
CLC	1 1 1 1	0 0 0 1	Clear carry.
IAC	1 1 1 1	0 0 1 0	Increment accumulator.
CMC	1 1 1 1	0 0 1 1	Complement carry.
CMA	1 1 1 1	0 1 0 0	Complement accumulator.
RAL	1 1 1 1	0 1 0 1	Rotate left. (Accumulator and carry)
RAR	1 1 1 1	0 1 1 0	Rotate right. (Accumulator and carry)
TCC	1 1 1 1	0 1 1 1	Transmit carry to accumulator and clear carry.
DAC	1 1 1 1	1 0 0 0	Decrement accumulator.
TCS	1 1 1 1	1 0 0 1	Transfer carry subtract and clear carry.
STC	1 1 1 1	1 0 1 0	Set carry.
DAA	1 1 1 1	1 0 1 1	Decimal adjust accumulator.
KBP	1 1 1 1	1 1 0 0	Keyboard process. Converts the contents of the accumulator from a one out of four code to a binary code.
DCL	1 1 1 1	1 1 0 1	Designate command line. (See note 1 on page 3.)

NOTES: [1] The condition code is assigned as follows:

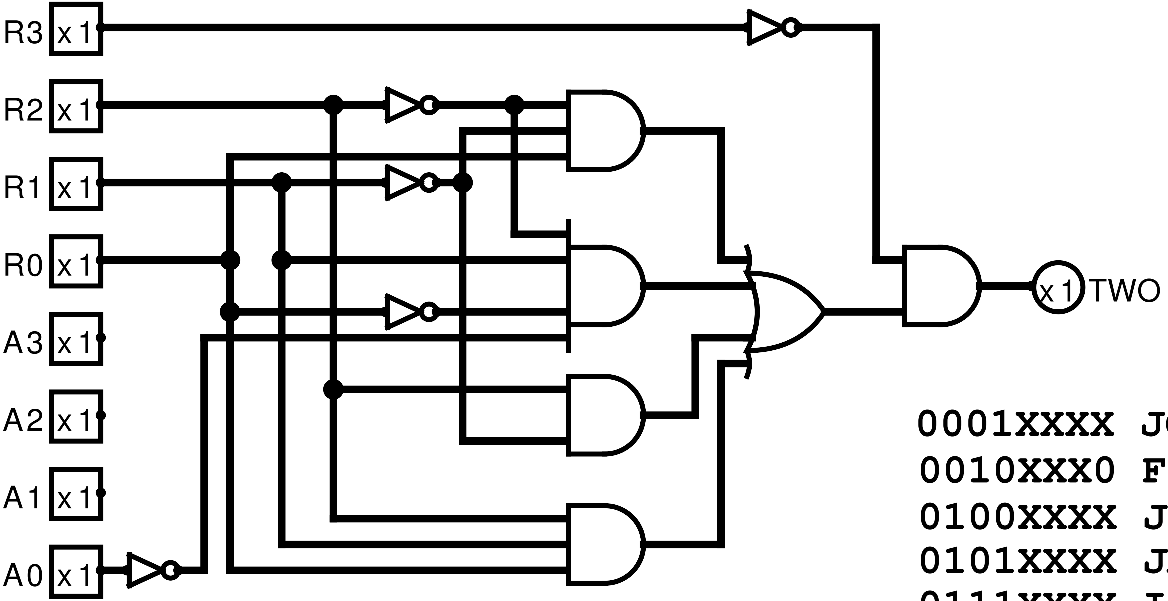
$C_1 = 1$	Invert jump condition	$C_2 = 1$	Jump if accumulator is zero	$C_4 = 1$	Jump if test signal is a 0
$C_1 = 0$	Not invert jump condition	$C_3 = 1$	Jump if carry/link is a 1		

[2] RRR is the address of 1 of 8 index register pairs in the CPU.

[3] RRRR is the address of 1 of 16 index registers in the CPU.

[4] Each RAM chip has 4 registers, each with twenty 4-bit characters subdivided into 16 main memory characters and 4 status characters. Chip number, RAM register and main memory character are addressed by an SRC instruction. For the selected chip and register, however, status character locations are selected by the instruction code (OPA).

hw1−6



```
0001XXXX JCN
0010XXX0 FIM
0100XXXX JUN
0101XXXX JMS
0111XXXX ISZ
```