CHAPTER 10

10-1.

a)
```
LD    R1, A
LD    R2, B
LD    R3, C
LD    R4, D
LD    R5, E
ADD   R1, R1, R2
SUB   R1, R1, R3
SUB   R4, R4, R5
MUL   R1, R1, R4
ST    X, R1
```

b)
```
MOV   T1, A
ADD   T1, B
SUB   T1, C
MOV   T2, D
SUB   T2, E
MUL   T1, T2
MOV   X, T1
```

c)
```
LD    A
ADD   B
SUB   C
ST    T1
LD    D
SUB   E
MUL   T1
ST    X
```

10-2.*

a)
```
LD    R1, E
LD    R2, F
MUL   R1, R1, R2
LD    R2, D
SUB   R1, R2, R1
LD    R2, C
DIV   R1, R2, R1
LD    R2, A
LD    R3, B
ADD   R2, R2, R3
MUL   R1, R1, R2
ST    Y, R1
```

b)
```
MOV   T1, A
ADD   T1, B
MUL   T1, C
MOV   T2, E
MUL   T2, F
MOV   T3, D
SUB   T3, T2
DIV   T1, T3
MOV   Y, T1
```

c)
```
LD    E
MUL   F
ST    T1
LD    D
SUB   T1
ST    T1
LD    A
ADD   B
MUL   C
DIV   T1
ST    Y
```

10-3.*

a) (A - B) x (A + C) x (B - D) = A B - A C +x B D -x

b, c)

PUSH A	PUSH B	SUB	PUSH A	PUSH C	ADD
A	B	A-B	A	C	A+C
	A		A-B	A	A-B
				A-B	

MUL	PUSH B	PUSH D	SUB	MUL	POP X
(A-B)x(A+C)	B	D	B-D	(A-B)x(A+C)x(B-D)	
	(A-B)x(A+C)	B	(A-B)x(A+C)		
		(A-B)x(A+C)			

10-4.

a) $(((A \times B) + C) \times D) \div (E - (A \times F)) = AB \times C + D \times EAF \times - \div Y$

b,c

PUSH A	PUSH B	MUL	PUSH C	ADD	PUSH D
A	B	AxB	C	(AxB)+C	D
	A		AxB		(AxB)+C

MUL	PUSH E	PUSH A	PUSH F	MUL	SUB
((AxB)+C)xD	E	A	F	AxF	E - (AxF)
	((AxB)+C)xD	E	A	E	
		((AxB)+C)xD	E	((AxB)+C)xD	
			((AxB)+C)xD		

DIV	POP Y
(((AxB)+C)xD) ÷ (E - (AxF))	

10-5.

a) Z = Y b) Z = M[Y] c) Z = Y + W + 2 d) Z = Y + X

10-6.*

a) X = 195 – 208 – 1 = –14 b) X = 1111 1111 1111 0010

The number is negative because the branch is backwards. The – 1 assumes that the PC has been incremented to point to the address after that of the address word of the instruction.

10-7.

a) X = 1000 – 144 – 1 = 855 b) X = 0000 0011 0101 0111

The number is negative because the branch is backwards. The – 1 assumes that the PC has been incremented to point to the address after that of the address word of the instruction.

10-8.

a) Read Instruction
Read Address Field
Read Effective Address
Read Operand
Read Address Field
Read Effective Address
Read Operand
Write Result

8 Memory Accesses

b) Read Instruction
Read Address Field
Read Effective Address
Read Operand
Read Address Field
Read Effective Address
Write Result

7 Memory Accesses

10-9.

a) Direct 2410 b) Immediate 551 c) Relative 552 + 2410 = 2962

d) Indexed 2410 + 2310 = 4720

10-10.*

a) 3 Register Fields x 4 bits/Field = 12 bits. 32 bits - 12 bits = 20 bits. 2^{20} = 1048576

b) 64 < 100 < 128 => 7 bits. 2 Register Fields x 4 bits/Field => 8 bits. 32 bits - 7 bits - 8 bits => 17 Address Bits

10-11. (Errata: In Register Indirect + increment, change LD R_i R_j to LD R_j R_i)

```
        LDI   R6, TOSAD   # Load R6 with address of the Top of Stack
```

Push and Pop can be implemented in one of two ways.

	Method 1			Method 2	
PUSH RX:	ST	RX, (R6)+	or	ST	RX, –(R6)
POP RX:	LD	RX,–(R6)		LD	RX, (R6)+

10-12.

PSHR and POPR work as follows:

PSHR:	POPR:
$M[SP] \leftarrow R0$	$SP \leftarrow SP + 1$
$SP \leftarrow SP - 1$	$R7 \leftarrow M[SP]$
$M[SP] \leftarrow R1$	$SP \leftarrow SP + 1$
$SP \leftarrow SP - 1$	$R6 \leftarrow M[SP]$
.	.
.	.
.	.
$M[SP] \leftarrow R7$	$SP \leftarrow SP + 1$
$SP \leftarrow SP - 1$	$R0 \leftarrow M[SP]$

10-13.

```
        LD    R[DR] ADRS
        ST    ADRS R[SB]
```

where ADRS is a memory address.

10-14.*

a)
```
ADD   R0, R4
ADC   R1, R5
ADC   R2, R6
ADC   R3, R7
```

b) $R0 \leftarrow 7B + 4B$, R0 = C6, C = 0

$R1 \leftarrow 24 + ED + 0$, R1 = 11, C = 1

$R2 \leftarrow C6 + 57 + 1$, R2 = 1E, C = 1

$R3 \leftarrow 1F + 00 + 1$, R3 = 20, C = 0

10-15.

AND: 0100 1000 OR: 1110 1111 XOR: 1010 0111

10-16.

a) OR with $(FF00)_{16}$ b) XOR with $(AAAA)_{16}$ c) AND with $(AAAA)_{16}$

10-17.*

OP	Result Register	C
	0110 1001	1
SHR	0011 0100	1
SHL	0110 1000	0
SHRA	0011 0100	0
SHLA	0110 1000	0
ROR	0011 0100	0
ROL	0110 1000	0
RORC	0011 0100	0
ROLC	01101000	0

10-18.

$$\begin{array}{rl} & (-)\ 0.123450000 \times 10^5 \\ + & (+)\ 0.0000000071234 \times 10^5 \\ \hline = & (-)\ 0.123449993 \times 10^5 \end{array}$$

Result = -0.123449993×10^5

10-19.*

$$\text{Smallest Number} = 0.5 \times 2^{-255}$$

$$\text{Largest Number} = (1 - 2^{-26}) \times 2^{+255}$$

10-20.*

E	e	$(e)_2$
+8	15	1111
+7	14	1110
+6	13	1101
+5	12	1100
+4	11	1011
+3	10	1010
+2	9	1001
+1	8	1000
0	7	0111
–1	6	0110
–2	5	0101
–3	4	0100
–4	3	0011
–5	2	0010
–6	1	0001
–7	0	0000

10-21.

a) $(-1)^{\text{sign}} \times 2^{\text{exponent} - 1023} \times (1.\text{fraction})$

b) $-1022 = (000\ 0000\ 0001)_2$

$-1 = (011\ 1111\ 1110)_2$

$0 = (011\ 1111\ 1111)_2$

$+1 = (100\ 0000\ 0000)_2$

$+1023 = (111\ 1111\ 1110)_2$

c) $\text{Largest} = (2 - 2^{-52}) \times 2^{+1023}$

$\text{Smallest} = 1 \times 2^{-1022}$

10-22.

If $2^x = 10^y$ then:

$x \log_{10} 2 = y \log_{10} 10$ $\quad \log_{10} 2 = 0.3,\ \log_{10} 10 = 1$

$0.3\ x = y$

Largest is $2^{127} \times (2 - 2^{-23}) = 2^{128} - 2^{104} = 10^{0.3 \times 128} - 10^{0.3 \times 104} = 2.5 \times 10^{38}$

Smallest is $2^{-126} = 10^{-0.3 \times 126} = 10^{-37.8} = 1.58 \times 10^{-38}$

10-23.*

TEST	$(0001)_{16}$, R	(AND Immediate 1 with Register R)
BNZ	ADRS	(Branch to ADRS if Z = 0)

10-24.

			Unsigned	Signed
a)	A =	1011 0110	182	- 74
	B =	0011 0111	55	55
b)	A + B =	1110 1101	237	- 19
c)	C=0, Z=0, N=0, V=0		C=0, Z=0, N=1, V=0	
d)	BNZ, BNC, BNN, BNV		BNZ,BNC,BN,BNV	

10-25.*

a)	A =	0101 1101	93
	B =	0101 1100	- 92
	A – B =	0000 0001	1

b) C (borrow) = 0, Z = 0

c) BA, BAE, BNE

10-26.

a)	A =	1101 1010	–38
	B =	0111 0110	–(118)
	A – B =	00010000	100

b) N = 0, Z = 0, V = 1

c) BL, BLE, BNE

10-27.*

	PC	SP	TOS
a) Initially	2000	4000	5000
b) After Call	0502	4001	2002
c) After Return	2002	4000	5000

10-28.

a) $R7 \leftarrow PC$ PC points to the next instruction in sequence

$PC \leftarrow M[PC - 1]$ PC Gets Effective address

b) R7 must be saved to memory.

10-29.

Branch Instruction:	The processor jumps to a new location without the ability to return on its own.
Call Instruction:	A jump with a way for the processor to return to the instruction after the one that caused the jump.
Program Interrupt:	A jump initiated by an event, not an instruction. A way to return to the first uncompleted instruction is provided.

10-30.*

External Interrupts:
1) Hard Drive
2) Mouse
3) Keyboard
4) Modem
5) Printer

Internal Interrupts:
1) Overflow
2) Divide by zero
3) Invalid opcode
4) Memory stack overflow
5) Protection violation

A software interrupt provides a way to call the interrupt routines normally associated with external or internal interrupts by inserting an instruction into the code. Privileged system calls for example must be executed through interrupts in order to switch from user to system mode. Procedure calls do not allow this change.

10-31.

a) $SP \leftarrow SP - 1$
$M[SP] \leftarrow PC$
$SP \leftarrow SP - 1$
$M[SP] \leftarrow PSR$
$EI \leftarrow 0$
$PC \leftarrow IVAD$
$PSR \leftarrow M[PC]$
$PC \leftarrow PC + 1$
Begin Fetch of Interrupt Service Routine

b) $PSR \leftarrow M[SP]$
$SP \leftarrow SP + 1$
$PC \leftarrow M[SP]$
$SP \leftarrow SP + 1$
Begin fetch of next instruction following the one for which the interrupt occurred.