Hand-in assignments (Thursday, week 2)

Exercise 1

Base 2	Base 8	Base 16	Base 10	Base 9
010101010101	2525	555	1365	1776
1010011	123	53	83	102
101000001011	5013	A0B	2571	3466
1010	12	A	10	11
1001	11	9	9	10

Exercise 2

Give the representation of the decimal value **-154** in the following representation using 10 bits.

Sign magnitude	1010011010
Signed (2-complement)	1101100110
1-complement	1101100101
Excess 155	0000000001

Exercise 3

1100.011

Complement bits add 1 to right position: 0011.101 $\rightarrow 3\frac{5}{8}$

The decimal value is $-3\frac{5}{8}$

Exercise 4

$12_{10} \rightarrow 0110_3$ and $.34_{10} \rightarrow .1000_3$

01101000

Exercise 5

a) M_{max} 0.1 followed by 23 ones = $1-2^{-24} \sim 0.99999$

b) M_{min} 0.1 followed by 23 zeros = 0.50000

c) E_{max} $2^8-1-128 = 127$

d) E_{min} $1-128 = -127$

e) V_{max} $(1-2^{-24}) \times 2^{127} \sim 0.170141 \times 10^{39}$

f) V_{min} $2^{-1} \times 2^{-127} = 2^{-128} \sim 0.293874 \times 10^{-39}$

g) smallest positive number: $2^{-1} \times 2^{-127}$
next positive number: $(2^{-1} + 2^{-24}) \times 2^{-127}$
The Δr between zero en smallest positive number is $2^{-1} \times 2^{-127} = 2^{-128}$
The Δr smallest positive number and next positive number $2^{-24} \times 2^{-127} = 2^{-151}$
This is not a nice property of the DEC format. In the IEEE Floating Point due the denormalized numbers this effect does not occur.

h) Representation of $2\ ^9/_{16}$?
$2\ ^9/_{16} = {}^{41}/_{16}$
$0.5 <= \text{mantissa} < 1$
${}^{41}/_{16} = {}^{41}/_{16} \times {}^4/_4 = {}^{41}/_{64} \times 2^2$
Mantissa is 0.1010010000000 => bit pattern: 01001000....0
Exponent is 2 => in excess 128 is 130 => bit pattern: 10000010
Sign positive => bit: 0

or (if you recognize the bit pattern for this number in binary):
$2\ ^9/_{16}$ the bit pattern is: 00010.1001
For a normalized number you have to shift the point 2 positions to the left (and the multiple is with 2^2).
Exponent field is 2 + 128 = 130
Fraction .101001 and there 010010000. is in the fraction field (hidden bit (red) is not stored).

i) Representation of the decimal value 0.2 ?
0.2
$0.5 <= \text{mantissa} < 1$
${}^2/_{10} = {}^2/_{10} \times {}^4/_4 = {}^8/_{10} \times 2^{-2}$
Mantissa is $({}^8/_{10} \times 2^{24}) \times 2^{-24} = 13421772.8 \times 2^{-24}$
Value cannot be exactly represented! Round to 13421772×2^{-24}
0.11001100110011001100110, the first 0.1 is hidden so the bit pattern

---

Alternative to find bit pattern 0.8
0.8 x 2 = 1.6 → 1 (note: 1.6-1=0.6 on next row)
0.6 x 2 = 1.2 → 1
0.2 x 2 = 0.4 → 0
0.4 x 2 = 0.8 → 0
0.8 x 2 = 1.6 → 1 … pattern repeats: 1100110011001100 …..

---

for the fraction is: => 10011001100110011001100

Exponent is -2 => in excess 128 is 126 => bit pattern: 01111110
Sign positive => bit: 0

j) Pattern:
1 00000111 11010000000000000000001

sign: negative
Exponent: 00000111_2 = unsigned representation 7_{10} =
excess 128: $7-128= -121_{10}$
Mantissa: **0.1**11010000000000000000001
$2^{-1} + 2^{-2} + 2^{-3} + 2^{-5} + 2^{-24} \sim 0.90625006$

Decimal value represented is $-0.90625006 \times 2^{-121} \sim -3.4089338 \times 10^{-37}$

Exercise 6

```
      .begin
      .org 0
      sethi arr1, %r1
      srl %r1,10,%r1          ! the address of arr1 in  register %r1
      addcc %r0, %r0, %r10    ! initialize counter
loop: ld [%r1], %r3
      addcc %r3,%r0,%r0       ! end of array?
      be finished
      addcc %r3, -2, %r0      ! compare with -2
      be incr
      ba next
incr: addcc %r10,1,%r10       ! increment %r10
next: addcc %r1,4,%r1         ! %r1 has address next element of array
      ba loop
finished: halt
      .org 200
arr1: 5,4,3,6,2,3,4,2,1,2,0 ! end of row is indicated with 0
      .end
```

There are alternatives for this program!

Exercise 7
addcc %r1, 4, %r1

Exercise 8

a) 1 10101010 $00_2 = 1704_{10}$

b) (there are alternatives)

RTL

1704: R[%temp0] ←addcc(R[%r0],R[rs2]) ; R[%temp0]←R[rs2] and condition codes are set
(Note: at the end of clock cycle, on active edge of clock, the status register is updated)

1705: if status is Negative goto 1707 ; negative? Then 2-complement R[rs2]
(Note: now the status updated status is used)

1706: R[rd]←addcc(R[%r0],R[rs2]) goto 2047 ; R[rd]←R[rs2] (note R[rs2] is already positive!)
; at address 2047 PC is incremented (required to fetch
; next instruction).
; alternative RTL in controller is
; R[rd]←adcc(R[%r0],R[%temp0]), or
; R[rd]←adcc(R[%temp0],R[%r0])

1707: R[%temp0]←orn(R[%r0],R[%temp0]) goto 600 ; R[%temp0]←not R[%temp0]
; address 1708 is maybe in use for another instruction
; therefore jump to address 600
; note: also **orncc** can be used (status not important),
; or: R[%temp0]←orn(R[%temp0],R[%r0]])

600: R[rd]←INCA(R[%temp0]); goto 2047 ; R[rd]←R[%temp0] + 1 and goto 2017

address	A	Amux	B	Bmux	C	Cmux	Rd	Wr	ALU	Cond	Jump addr
600	%temp0	0	-	-	-	1	0	0	inc(A) (1101)	Jmp (110)	2047
1704	%r0	0	-	1	%temp0	0	0	0	addcc (0011)	nxt (000)	-
1705	-	-	-	-	%r0	0	0	0	-	N=1 (001)	1707
1706	%r0	0	-	1	-	1	0	0	addcc (0011)	Jmp (110)	2047
1707	%r0	0	%temp0	0	%temp0	0	0	0	orn (0111)	Jmp (110)	600