

Exam 1 review

closed book, no calculator

Lectures 1-10

Online HW1-3 (look at HW4 too)

Labs 1 and 2

Problems on old tests you are not responsible for

1) Definitions (matching or multiple choice)

volatile, nonvolatile, RAM, ROM, port
 static efficiency, dynamic efficiency
 structured program, call graph, data flow graph
 basis, nibble, precision, decimal digits (see table below)
~~fixed point~~, overflow, ceiling and floor, drop out,
 bus, address bus, data bus,
 memory-mapped, I/O mapped
 bus cycle, read cycle, write cycle,
 IR, EAR, BIU, CU, ALU, registers,
 device driver, reset vector
 friendly, mask, toggle,

$2^2 = 4$	$2^8 = 256$	$2^{14} = 16384$
$2^3 = 8$	$2^9 = 512$	$2^{15} = 32768$
$2^4 = 16$	$2^{10} = 1024 \approx 10^3$	$2^{16} = 65536$
$2^5 = 32$	$2^{11} = 2048$	$16^2 = 256$
$2^6 = 64$	$2^{12} = 4096$	$16^3 = 4096$
$2^7 = 128$	$2^{13} = 8192$	$16^4 = 65536$

decimal digits	exact range	exact alternatives
3	0 to 999	1,000
$3\frac{1}{2}$	0 to 1999	2,000
$3\frac{3}{4}$	0 to 3999	4,000
4	0 to 9999	10,000
$4\frac{1}{2}$	0 to 19,999	20,000
$4\frac{3}{4}$	0 to 39,999	40,000
5	0 to 99,999	100,000
$5\frac{1}{2}$	0 to 199,999	200,000
$5\frac{3}{4}$	0 to 399,999	400,000
6	0 to 999,999	1,000,000
$6\frac{1}{2}$	0 to 199,999	2,000,000
$6\frac{3}{4}$	0 to 3,999,999	4,000,000
N	0 to $10^N - 1$	10^N
$N\frac{1}{2}$	0 to $2 * 10^N - 1$	$2 * 10^N$
$N\frac{3}{4}$	0 to $4 * 10^N - 1$	$4 * 10^N$

Standard definition of decimal digits.

2) Number conversions, 8-bit (fill in the blank)

convert one format to another without a calculator

signed decimal e.g., -56

unsigned decimal e.g., 200

binary e.g., %11001000

hexadecimal e.g., \$C8

I won't ask you to convert signed binary or signed hex:

signed binary e.g., -%00101111

signed hexadecimal e.g., -\$2F

fixed-point representations

_____ given resolution convert between **value** and **integer**

_____ given precision and range choose the fixed-point format

3) Details of executing single instructions

8-bit addition, subtraction yielding result, N, Z, V, C
(like HW2, HW3)

simplified cycle by cycle execution

assembly listing to execution cycles (aLec04)

for indexed mode addresses, for example

~~ldaa 4,x~~

~~ldaa 40,x~~

~~ldaa -4,x~~

~~ldaa -40,x~~

~~ldaa \$400,x~~

~~ldaa 4,+x~~

~~ldaa 4,-x~~

~~ldaa 4,x+~~

~~ldaa 4,x-~~

calculate effective address

go from assembly to machine code ~~xb~~

go from machine code ~~xb~~ to assembly

simple multiply and divide (**mul idiv fdiv**)

stack functions for **bsr** and **rts**

4) Simple programs (look at assembly code in Chap 1-5)

initialize stack

create global variables

set reset vector

specify an I/O pin is an input

specify an I/O pin is an output

clear an I/O output pin to zero

set an I/O output pin to one

toggle an I/O output pin

check if an I/O input pin is high or low

e.g., if PT4 is low then make PM2 high

****study question*****

8-bit operations

add, sub, shift left, shift right, and, or, eor

simple **if-then** like examples in Chapter 5

if-then-else

no **if((uG1>5)&&(uG2<100))**{

simple **while-loop** like examples in Chapter 5

simple **for-loop** like those in this lecture

simple subroutines, parameters passed in registers

four lines of comments for client

* **purpose**

* **inputs: registers, format, units**

* **outputs: registers, format, units**

* **error possibilities**

called with **bsr**, returns using **rts**

5) Switch and LED interfaces (Lab 2)

Look at previous exams to see the types of information given to you. Notice also the format of the exam and the expected answers. You will get information a list of instructions and addressing modes You will also get the CPU12 page(s) for any instruction(s) for which you need to find bus cycles.

it is important to know









- precision (e.g., 8-bit, 16-bit)
- format (e.g., unsigned, signed)
 - unsigned, **bhi blo bhs** and **bls**
 - signed, **bgt bls bge** and **ble**

It takes three steps

1. read the first value into a register
2. compare the first value with the second value
3. conditional branch

Compare the four possible inequalities





Assume **PTT** is a unsigned 8-bit input port, and let **Threshold** be an unsigned 8-bit global variable

C code	assembly code
<pre>if(PTT > Threshold){  } </pre>	<pre>ldab PTT cmpb Threshold bls next  next </pre>
<pre>if(PTT >= Threshold){  } </pre>	<pre>ldab PTT cmpb Threshold blo next  next </pre>
<pre>if(PTT < Threshold){  } </pre>	<pre>ldab PTT cmpb Threshold bhs next  next </pre>
<pre>if(PTT <= Threshold){  } </pre>	<pre>ldab PTT cmpb Threshold bhi next  next </pre>

Compare signed versus unsigned conditionals

Assume **uG** is an unsigned 8-bit global variable

Assume **sG** is a signed 8-bit global variable

C code	assembly code
<pre>if(uG >= 5){  } </pre>	<pre>ldaa uG cmpa #5 blo next  next </pre>
<pre>if(sG >= 5){  } </pre>	<pre>ldaa sG cmpa #5 blt next  next </pre>

Compare 8-bit versus 16-bit conditionals

Assume **uG1** and **uG2** are unsigned 8-bit variables

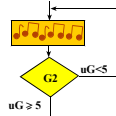
Assume **uH1** and **uH2** are unsigned 16-bit variables

C code	assembly code
<pre>if(uG2 >= uG1){ 🎵🎵🎵🎵 }</pre>	<pre>ldaa uG2 cmpa uG1 blo next 🎵🎵🎵🎵 next</pre>
<pre>if(uH2 >= uH1){ 🎵🎵🎵🎵 }</pre>	<pre>ldd uH2 cpd uH1 blo next 🎵🎵🎵🎵 next</pre>

```

for(uG=0;uG<5;uG++){
    Buf[uG] = 0;
}
clr uG
loop ldaa uG
    cmpa #5
    bhs next ; stop when uG>=5
    ldx #Buf ; body of while loop
    ldaa uG
    clr A,X
    inc uG
    bra loop
next
pt = Buf;
for(i=5;i>0;i--){ // something 5 times
    *pt = 0;
    pt++;
}
ldx #Buf
ldaa #5 ; loop 5 down to 0
loop clr 1,X+ ; body of for loop
    dbne A,loop
do{
    🎵🎵🎵🎵
}
while(uG < 5);
loop 🎵🎵🎵🎵 ; body of while loop
    ldaa uG
    cmpa #5
    blo loop ; stop when uG>=5
loop
    🎵🎵🎵🎵 ; body of while loop
    bra loop

```



Problem: write code that waits for a switch to be pressed. Assume PP3 is an input with a switch attached.

0) how are we going to test it?

- 1) flow chart
- 2) pseudocode
- 3) assembly
- 4) testing

The bottom line

- Study previous exam1's
- Study homework answers and explanations
- Review Labs 1 and 2
- Review lecture notes 1-10