Last Name: ________________________________ First Name: ______________________________

Jonathan W. Valvano February 25, 2004, 1:00pm-1:50pm

This is a closed book exam. No notes or calculators are allowed, just a pencil and eraser. You must put your answers in the answer boxes only, information written outside the box will not be graded. You have 50 minutes, so please allocate your time accordingly. ***Please read the entire quiz before starting.***

```
static char aa=0;
char addIt(const char bb){char cc;
  aa++;
  cc = aa+bb;
  return(cc);}
void main(void){char dd; static char ee;
  ee = PORTT;
  dd = addIt(ee+5);}
```

(10) Question 1. At the time when the **`cc=aa+bb;`** line is being executed, which of the variables `aa bb cc dd ee` will be saved on the stack? I am not asking which variables are legally within the scope of the function, but which ones are physically on the stack.

(5) Question 2. Give the minimum and maximum values that can be represented using 8-bit signed binary fixed-point, with a resolution of 2^{-4}? Note: 2^{-4} equals 1/16.

(5) Question 3. What is the value of a 16-bit unsigned decimal fixed-point number (resolution is 10^{+2}, which equals 100) if the integer stored in memory is 12345?

(20) Question 4. Consider the sequence of events that occur as real time interrupts (RTI) are initialized at a 4.096 ms period, the first interrupt is triggered, the first time the interrupt service routine is executed, and then the foreground continues to execute.

(4) Part a) Which three events **must** your software accomplish during the execution of the ritual? *Give three letters in the proper execution order.*

(4) Part b) Which single event will trigger a RTI interrupt? *Give one letter specifying the event that causes an interrupt to occur.*

(4) Part c) Which three events occur automatically in hardware as an interrupt is processed, making the thread switch from foreground to background? Do not include events that occur in software as the interrupt service routine is executed. *Give three letters but this time list them in the proper time sequence order.*

(4) Part d) You want the interrupts to occur periodically and not just once. Which single event must your software accomplish during the execution of the interrupt service routine? *Give one letter.*

(4) Part e) Each interrupt service routine ends with a **`rti`** instruction. What action is caused by executing this instruction? *Give one letter.*

A) **`RTICTL = 0x03;`** //MC68HC812A4
`RTICTL = 0x33;` //9S12C32

B) **`RTICTL |= 0x80;`** //MC68HC812A4
`CRGINT |= 0x80;` //9S12C32

C) **`RTICTL &= ~0x80;`** //MC68HC812A4
`CRGINT &= ~0x80;` //9S12C32

D) **`RTIFLG = 0x80;`** //MC68HC812A4
`CRGFLG = 0x80;` //9S12C32

E) **`RTIFLG = 0x00;`** //MC68HC812A4
`CRGFLG = 0x00;` //9S12C32

F) **`asm sei`**

G) **`asm cli`**

H) Timer hardware causes **`RTIE`** to become 1

I) Timer hardware causes **`RTIE`** to become 0

J) Timer hardware causes **`RTIF`** to become 1

K) Timer hardware causes **`RTIF`** to become 0

L) The 6812 hardware sets the **`I`** bit to 1

M) The 6812 hardware clears the **`I`** bit to 0

N) The 6812 hardware pushes registers on stack

O) The 6812 hardware loads interrupt vector into SP

P) The 6812 hardware loads interrupt vector into PC

Q) All registers are pulled off the stack, returning control back to the foreground with I=0

R) All registers are pulled off the stack, returning control back to the foreground with I=1

(6) Question 5. An unsigned fixed point system has a range of 0 to 100 with a resolution of 2^{-10}. Note: 2^{-10} equals 1/1024.With which of the following data types should the software variables be allocated? When more than one answer is possible choose the most space efficient type.

A) `unsigned char` D) `char` G) `float`
B) `unsigned short` E) `short` H) `double`
C) `unsigned long` F) `long`

For questions 6-13, the definition is given and you are asked to give the correct term described by that definition. Since there are more terms than definitions, not all terms will be used. *Answer each as A through GG.*

(3) Question 6. A condition in which information is lost when power is removed.

(3) Question 7. Output current when the signal is low.

(3) Question 8. The characteristic of a debugger that allows the software/hardware system to operate normally as if the debugger did not exist.

(3) Question 9. A type of logic in which the output can be low or off.

(3) Question 10. A debugging technique that fixes all its inputs to specific values and can be repeated over and over.

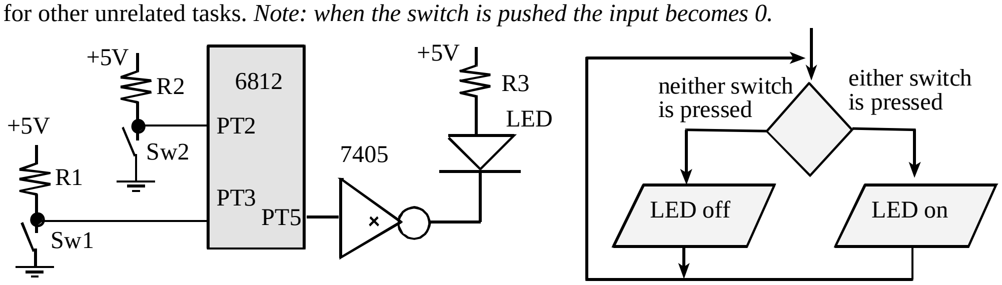
(3) Question 11. Input voltage when the signal is high.

(3) Question 12. A software-hardware synchronization method in which the software continuously looks at a hardware status flag waiting for the hardware operation to complete.

(3) Question 13. A system in which the time between when new input is ready and the time when the input is read by the software is small and bounded.

A) high speed CMOS
B) Schottky
C) atomic
D) reentrant
E) volatile
F) busy waiting
G) critical section
H) V_{IH}
I) V_{IL}
J) V_{OH}
K) V_{OL}
L) desk check
M) embedded
N) friendly
O) stabilize
P) I_{IH}
Q) I_{IL}
R) instrument
S) interrupt acknowledge
T) interrupt arm
U) interrupt enable
V) interrupt vector
W) invasiveness
X) I_{OH}
Y) I_{OL}
Z) latency
AA) non-intrusiveness
BB) nonvolatile
CC) open collector
DD) tristate
EE) real-time
FF) multi-threaded
GG) blind cycle

(30) Question 14. An LED and two switches are attached to Port T as shown below. The other 5 bits of Port T are used for other unrelated tasks. *Note: when the switch is pushed the input becomes 0.*



The answers to this problem do not need to be complete functions, just the C code fragments.

(5) Part a) Which equation should be used to determine R1 and R2? *Specify one letter.*
A) $R1 < (5-V_{IH})/I_{IH}$ C) $R1 < (5-V_{IL})/I_{IL}$ E) $R1 < (V_{IH}-V_{IL})/I_{IL}$
B) $R1 < (V_{IH})/I_{IH}$ D) $R1 < (V_{IL})/I_{IL}$ F) $R1 < (V_{IH}-V_{IL})/(I_{IH}-I_{IL})$

(5) Part b) Assume the V_{OL} of the 7405 is 1V. The desired operating point of the LED is 3V and 10mA. What resistor value should be used for R3? *Give units.*

(5) Part c) Write friendly C code that makes PT5 an output.

(5) Part d) Write friendly C code that makes PT2 and PT3 inputs.

(10) Part e) Write C code that continuously reads the two inputs.

After reading the value of the two switches you should turn on the LED if either switch is pressed or turn off the LED if both switches are not pressed.

Be friendly.

No debouncing required.

MC68HC812A4 registers

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
Timer Port Data Register (PORTT)	Read: / Write:	PT7	PT6	PT5	PT4	PT3	PT2	PT1	PT0
See page 208.	Reset:	Unaffected by reset							
Timer Port Data Direction Register (DDRT)	Read: / Write:	Bit 7	5	5	4	3	2	1	Bit 0
See page 209.	Reset:	0	0	0	0	0	0	0	0

0 = Configures the corresponding I/O pin for input only
1 = Configures the corresponding I/O pin for output

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	RTIE	RSWAI	RSBCK	0	RTBYP	RTR2	RTR1	RTR0
Write:				(unimplemented)				
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 10-7. Real-Time Interrupt Control Register (RTICTL)

RTIE — Real Time Interrupt Enable
Read and write anytime.
0 = Interrupt requests from RTI are disabled.
1 = Interrupt will be requested whenever RTIF is set.

RTR2, RTR1, RTR0 — Real-Time Interrupt Rate Select
Read and write anytime.

RTR2	RTR1	RTR0	Divide M By:	Time-Out Period M = 4.0 MHz	Time-Out Period M = 8.0 MHz
0	0	0	OFF	OFF	OFF
0	0	1	2^{13}	2.048 ms	1.024 ms
0	1	0	2^{14}	4.096 ms	2.048 ms
0	1	1	2^{15}	8.196 ms	4.096 ms
1	0	0	2^{16}	16.384 ms	8.196 ms
1	0	1	2^{17}	32.768 ms	16.384 ms
1	1	0	2^{18}	65.536 ms	32.768 ms
1	1	1	2^{19}	131.72 ms	65.536 ms

	Bit 7	6	5	4	3	2	1	Bit 0
Read:	RTIF	0	0	0	0	0	0	0
Write:		(unimplemented)	(unimplemented)	(unimplemented)	(unimplemented)	(unimplemented)	(unimplemented)	(unimplemented)
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 10-8. Real-Time Interrupt Flag Register (RTIFLG)

RTIF — Real Time Interrupt Flag
This bit is cleared automatically by a write to this register with this bit set.
0 = Time-out has not yet occurred.
1 = Set when the time-out period is met.

9S12C32 registers

Address Offset: $__00

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write	PTT7	PTT6	PTT5	PTT4	PTT3	PTT2	PTT1	PTT0

	Bit 7	6	5	4	3	2	1	Bit 0
Read: Write:	DDRT7	DDRT6	DDRT5	DDRT4	DDRT3	DDRT2	DDRT1	DDRT0
Reset:	0	0	0	0	0	0	0	0

0 = Configures the corresponding I/O pin for input only
1 = Configures the corresponding I/O pin for output

Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RTICTL	Read:	0	RTR6	RTR5	RTR4	RTR3	RTR2	RTR1	RTR0
	Write:								
CRGFLG	Read:	RTIF	PROF	0	LOCKIF	LOCK	TRACK	SCMIF	SCM
	Write:								
CRGINT	Read:	RTIE	0	0	LOCKIE	0	0	SCMIE	0
	Write:								

RTIF — Real Time Interrupt Flag

RTIF is set to 1 at the end of the RTI period. This flag can only be cleared by writing a 1. Writing a 0 has no effect. If enabled (RTIE=1), RTIF causes an interrupt request.
1 = RTI time-out has occurred.
0 = RTI time-out has not yet occurred.

RTIE — Real Time Interrupt Enable Bit.
1 = Interrupt will be requested whenever RTIF is set.
0 = Interrupt requests from RTI are disabled.

RTR[6:4] — Real Time Interrupt Prescale Rate Select Bits

These bits select the prescale rate for the RTI. See **Table 3-2**.

RTR[3:0] — Real Time Interrupt Modulus Counter Select Bits

These bits select the modulus counter target value to provide additional granularity.**Table 3-2** shows all possible divide values selectable by the RTICTL register. The source clock for the RTI is OSCCLK.

Table 3-2 RTI Frequency Divide Rates

RTR[3:0]	RTR[6:4] =							
	000 (OFF)	001 (2^{10})	010 (2^{11})	011 (2^{12})	100 (2^{13})	101 (2^{14})	110 (2^{15})	111 (2^{16})
0000 (÷1)	OFF*	2^{10}	2^{11}	2^{12}	2^{13}	2^{14}	2^{15}	2^{16}
0001 (÷2)	OFF*	$2x2^{10}$	$2x2^{11}$	$2x2^{12}$	$2x2^{13}$	$2x2^{14}$	$2x2^{15}$	$2x2^{16}$
0010 (÷3)	OFF*	$3x2^{10}$	$3x2^{11}$	$3x2^{12}$	$3x2^{13}$	$3x2^{14}$	$3x2^{15}$	$3x2^{16}$
0011 (÷4)	OFF*	$4x2^{10}$	$4x2^{11}$	$4x2^{12}$	$4x2^{13}$	$4x2^{14}$	$4x2^{15}$	$4x2^{16}$
0100 (÷5)	OFF*	$5x2^{10}$	$5x2^{11}$	$5x2^{12}$	$5x2^{13}$	$5x2^{14}$	$5x2^{15}$	$5x2^{16}$
0101 (÷6)	OFF*	$6x2^{10}$	$6x2^{11}$	$6x2^{12}$	$6x2^{13}$	$6x2^{14}$	$6x2^{15}$	$6x2^{16}$
0110 (÷7)	OFF*	$7x2^{10}$	$7x2^{11}$	$7x2^{12}$	$7x2^{13}$	$7x2^{14}$	$7x2^{15}$	$7x2^{16}$
0111 (÷8)	OFF*	$8x2^{10}$	$8x2^{11}$	$8x2^{12}$	$8x2^{13}$	$8x2^{14}$	$8x2^{15}$	$8x2^{16}$