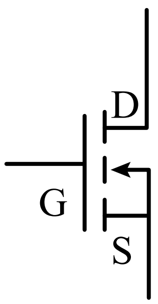
Jonathan W. Valvano First Name: _______________ Last Name:____________________
May 13, 2014, 9am-12n
Closed book part

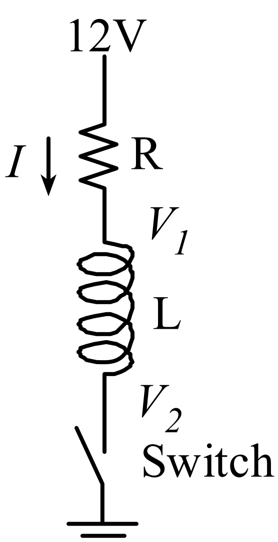
(5) Question 1. Draw the circuit symbol for an NPN MOSFET transistor, like an IRF540. Label the three pins of the transistor. Give the condition, if true the transistor goes into saturation.

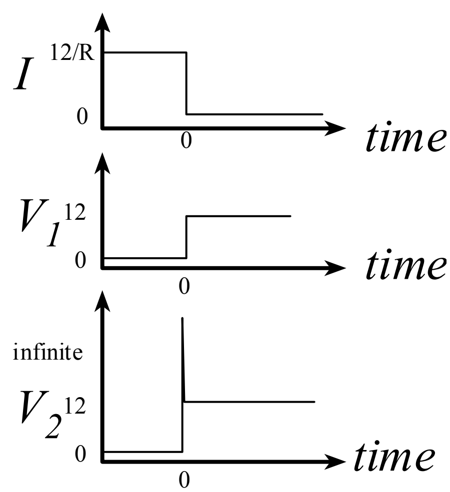
Pins are Gate, Source, Drain



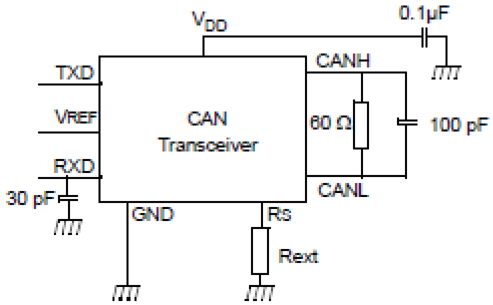
V_{GS} > constant turns on MOSFET

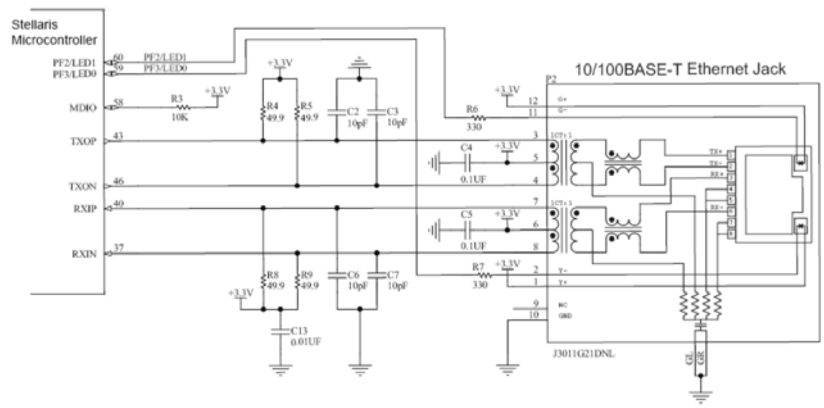
(5) Question 2. Consider the following circuit with an ideal resistor, ideal inductor, and ideal switch. Assume the switch is initially closed. At time equals 0, the switch is opened. Roughly sketch the voltage, V_1, and current, I, as a function of time.





Start with the current. Initially the current is +12/R. Since the switch is ideal the current flips to zero at time 0^+. Since the resistor is ideal, the voltage V_1 simply goes from 0 to 12V. The voltage across the inductor goes to –infinity at time = 0 because the dI/dt is infinite. Therefore the voltage at V_2 ideally becomes infinite at t=0^+, which is why we add the 1N914 snubber diode.





(5) Question 3. What is the purpose of the two 120 ohm resistors used in a CAN network? Extra credit: why were the resistors not needed for our Lab 6?

The purpose of the resistors is to remove reflections in transmission line for long cables. It is a *transmission line* if L > $\lambda/4$

$f \approx 1/\tau$ $\quad\quad$ v = VF*c = $2\cdot10^8$ m/s

$\lambda = v/f \approx v\ \tau$ $\quad\quad$ slew rate = 25V/μs

1V in 40 ns, $\quad\quad$ $\lambda = 2\cdot10^8$ m/s $40*10^{-9}$s = 8m

$\lambda/4$ = 2meters is longer than actual cable 0.1meters

(5) Question 4. In what form are data transferred across UART, SSI, I2C, CAN and Ethernet networks?

Energy is carried from the transmitter to the receiver

(5) Question 5. Consider Ethernet at its lowest level, i.e., the physical channel. Is Ethernet full duplex or half duplex?

Full duplex means both directions at the same time

(5) Question 6. Consider CAN at its lowest level, i.e., the physical channel. Is CAN synchronous or asynchronous?

Asynchronous means there is no clock in the cable.

(5) Question 7. The following friendly code is used for thread profiling in a preemptive scheduler like Lab 2. Assume Port A pins 5 and 6 are initialized to outputs. Are there any bugs in the way the profile is configured? If there are bugs, edit the code to remove the bugs. Full credit given to the best answer.

1) Change to bit specific addressing

```
#define PA6  (*((volatile unsigned long *)0x40004100))
#define PA5  (*((volatile unsigned long *)0x40004080))
```

void Thread1(void){	void Thread2(void){
Init1();	Init2();
while(1){	while(1){
~~GPIO_PORTA_DATA |= 0x40;~~	~~GPIO_PORTA_DATA |= 0x20;~~
PA6 = 0x40;	PA5 = 0x20;
Body1();	Body2();
~~GPIO_PORTA_DATA &= 0xFFFFFFBF;~~	~~GPIO_PORTA_DATA &= 0xFFFFFFDF;~~
PA6 = 0x00;	PA5 = 0x00;
}	}
}	}

2) Change one thread to use port B (one uses A, one uses B)

void Thread1(void){	void Thread2(void){
Init1();	Init2();
while(1){	while(1){
GPIO_PORTA_DATA |= 0x40;	GPIO_PORTB_DATA |= 0x20;
Body1();	Body2();
GPIO_PORTA_DATA &= 0xFFFFFFBF;	GPIO_PORTB_DATA &= 0xFFFFFFDF;
}	}
}	}

3) disable interrupts, or OS_Critical (1 point out of 5)

(5) Question 8.

Part a) When should you use cycle steal DMA rather than burst DMA?

Use cycle steal for low latency, I/O bandwidth less than bus bandwidth
Small data size, infrequent request

Part b) When should you use burst DMA rather than cycle steal DMA?

Use burst for high bandwidth, I/O bandwidth matches bus bandwidth
Large chunk of data to be transferrred, I/O with large buffer

(5) Question 9. A Fuzzy Logic system employs a false=0, and a true=255 definition. Consider the following Fuzzy Logic definition for **not**, where – is regular subtraction

$\sim A \equiv 255 - A$

Prove or disprove the following identity

$A \mid (\sim A) = \text{true}$

False by counter example, let A = 128, ~A=127

$A \mid (\sim A) = 128 \mid 127 = \max(128,127) = 128 \neq \text{true}$

(10) Question 10. Consider the following digital filter

y(n) = 2*x(n) + 2*x(n-4) – y(n-4)/16

Part a) Find all the poles and all the zeros for this digital filter

$Y(z)\,(16+z^{-4}) = 32(1+z^{-4})X(z)$

$H(z) = 32(1+z^{-4})/(16+z^{-4})$

Zeros when $(1+z^{-4})=0$, $z^{-4}= -1$, or $z^{+4}= -1$

Two zeros at +j, two zeros at -j

Poles when $(16+z^{-4})=0$, $z^{-4}= -16$, or $z^{+4}= -1/16$

Two poles at +½j, two poles at -½j

Part b) At what sampling rate does this filter reject 60 Hz?

Zeros are at +j, -j, so reject at ¼ f_s, so f_s = 240 Hz

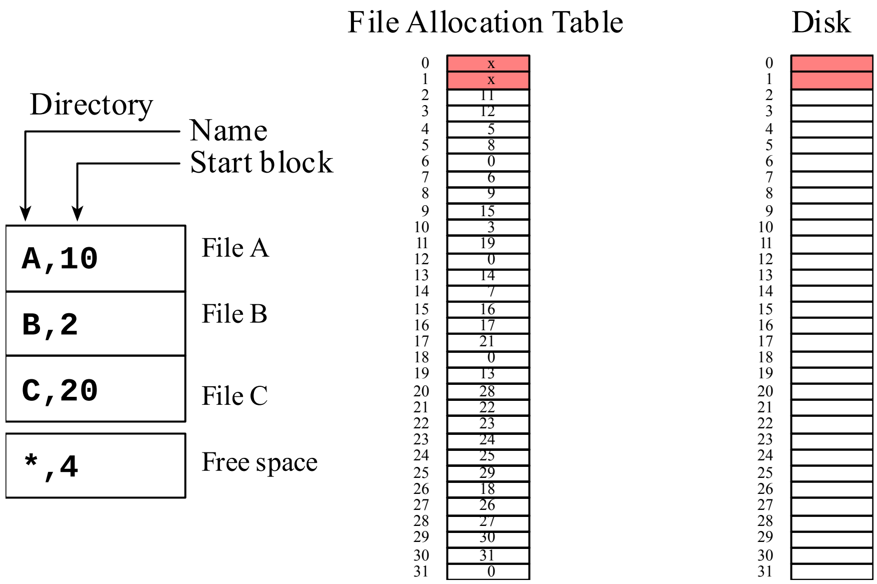
Part c) Redesign this filter so the DC gain is 1 (keeping the placement of the poles and zeros)

DC gain is H(z) at z=1

H(0) = 32(1+1)/(16+ 1) = 64/17, so multiply all x terms by 17/64 (do not scale the y terms!)

y(n) = (17*x(n) + 17*x(n-4) – 2*y(n-4))/32

(10) Question 11. Consider this file system that uses a FAT. Each disk block contains 32 bytes. The directory is in block zero, the FAT is in block 1. Each directory entry contains a file name and an index to the first FAT entry for that file. The last entry in the directory contains an index to the first FAT entry for the free space. Entire blocks are allocated to a file, so there is no internal fragmentation.



Part a) How many free blocks are there?

See Figure 7.8 in book. Start with 4 and count 15

4,5,8,9,15,16,17,21,22,23,24,25,29,30,31 (0 does not count)

Part b) Does this file system have external fragmentation? Simply answer yes or no

No (all 15 blocks could be used for one file)

Part c) How many bytes are saved in File A?

See Figure 7.7 in book. Start with 10 and count 3 blocks

10,3,12 (0 doesn't count)

Jonathan W. Valvano First Name: _______________ Last Name:____________________

Open book part

Open book, open notes, calculator (no laptops, phones, devices with screens larger than a TI-89 calculator, devices with wireless communication). Please don't turn in any extra sheets.

(20) Question 12. You are given a CAN network connecting two computers A and B, like the one developed in Lab 6. The CAN will be dedicated for the sole purpose of implementing one distributed spinlock counting semaphore. I.e., the CAN will not be used for purposes other than implementing this one semaphore. You may assume the CAN is initialized on both computers, and the OS has access to the CAN via the **`CAN0_Setup_Message_Object`** function. To send a 4-byte message you call

```
CAN0_Setup_Message_Object(ID, NULL, 4, data, ID, MSG_OBJ_TYPE_TX);
```

The starter file CAN programs send 4 bytes of data. Feel free to increase or decrease the amount of data as you wish. You may assume only one thread in computer A calls **`OS_DistributedWait`** and only one thread in computer B calls **`OS_DistributedSignal`**. You are allowed to make additional simplifying assumptions if you need to. You may use, without showing implementation, any OS function you implemented in your Lab 2 or Lab 3 OS. It is important not to introduce critical sections.

(5) Part a) Edit the following ISR that runs on Computer A.

```
#define RCV_ID 4
void CAN0_Handler(void){ unsigned char data[4];
  unsigned long ulIntStatus, ulIDStatus;
  int i;
  tCANMsgObject xTempMsgObject;
  xTempMsgObject.pucMsgData = data;
  ulIntStatus = CANIntStatus(CAN0_BASE, CAN_INT_STS_CAUSE); // cause?
  if(ulIntStatus & CAN_INT_INTID_STATUS){  // receive?
    ulIDStatus = CANStatusGet(CAN0_BASE, CAN_STS_NEWDAT);
    for(i = 0; i < 32; i++){    //test every bit of the mask
      if( (0x1 << i) & ulIDStatus){  // if active, get data
        CANMessageGet(CAN0_BASE, (i+1), &xTempMsgObject, true);
        if(xTempMsgObject.ulMsgID == RCV_ID){
          RCVData[0] = data[0];
          RCVData[1] = data[1];
          RCVData[2] = data[2];
          RCVData[3] = data[3];
          MailFlag = true;   // new mail
          Counter++; // any message means increment semaphore
        }
      }
    }
  }
  CANIntClear(CAN0_BASE, ulIntStatus);  // acknowledge
}
```

(5) Part b) Implement **`OS_InitNetSem`** on the computer containing the official copy of the counter. Clearly specific on which computer this runs.

```
long Counter;  // official copy of semaphore
// ******** OS_InitNetSem ************
// initialize network semaphore
// inputs:  initial Counter value
// output: none
void OS_InitNetSem(long value){

 Counter = value; // runs on A
}
```

(5) Part c) Implement the wait function that runs on Computer A.

```
// ******** OS_DistributedWait ************
// input:  none
// output: none
void OS_DistributedWait(void){
// exactly the same as a regular spinlock semaphore
  DisableInterrupts();  // Test and set is atomic
  while(Counter <= 0){  // disabled
    EnableInterrupts();
    OS_Suspend();       // optional for cooperative
    DisableInterrupts();
  }
  Counter --;           // disabled
  EnableInterrupts();   // disabled
}
```

(5) Part d) Implement the signal function that runs on Computer B.

```
// ******** OS_DistributedSignal ************
// increment Counter
// input:  none
// output: none
#define ID 4
void OS_DistributedSignal(void){
  CAN0_Setup_Message_Object(ID, NULL, 0, NULL, ID, MSG_OBJ_TYPE_TX);
// any message sent means increment semaphore
}
```

(15) Question 13. The goal is to design an FIR filter using the inverse discrete Fourier transform. The process is similar to Lab 4, except the size of the DFT is N=16 points (instead of 51 points), and the sampling rate is 1000 Hz. This FIR will have a linear phase response. The k column is the index from 0 to 15. The f column is the equivalent frequency in Hz. The gain G and phase θ columns are the desired response of the filter. *H(k)* is the complex representation of gain and phase. *h(n)* is the inverse discrete Fourier transform of *H(k)*.

k	*f*	*G=|H(k)|*	θ	*H(k)*	*h(n)*10000*
0	0.0	**0**	0.00	0	124
1	62.5	**0**	9.82	0	-625
2	125.0	**0**	19.63	0	-418
3	187.5	**0**	29.45	0	625
4	250.0	**1**	39.27	+i	935
5	312.5	**1**	49.09	0.382683432365092-0.923879532511286i	-625
6	375.0	**1**	58.90	-0.707106781186545+0.70710678118655i	-3142
7	437.5	**1**	68.72	0.923879532511289-0.382683432365084i	5625
8	500.0	1	78.54	-1	-3142
9	-437.5	1	-68.72	0.923879532511289+0.382683432365084i	-625
10	-375.0	1	-58.90	-0.707106781186545-0.70710678118655i	935
11	-312.5	1	-49.09	0.382683432365092+0.923879532511286i	625
12	-250.0	1	-39.27	-i	-418
13	-187.5	0	-29.45	0	-625
14	-125.0	0	-19.63	0	124
15	-62.5	0	-9.82	0	625

Part a) Give an equation relating *H(k)* as a function of *G* and θ.

$H(k) = G\, e^{j\theta} = G *(\cos(\theta)+j*\sin(\theta))$

Part b) Why is *H(k)* the complex conjugate of *H*(16-*k*) for *k* = 1 to 7?

To make the *h(n)* coefficients real (not complex)

Part c) What type of filter is this? HPF, LPF or BPF?

Band pass, 250Hz, reject 0 to 187, pass 250 to 500 Hz

Part d) Show the equation for this FIR filter using decimal fixed-point math. (no software required)

y(n) = (124*x(n) -625*x(n-1) -418*x(n-2) +625*x(n-3) +935*x(n-4) -625*x(n-5)
-3142*x(n-6) +5625*x(n-7) -3142*x(n-8) -625*x(n-9) +935*x(n-10)
+625*x(n-11) -418*x(n-12) -625*x(n-13) +124 *x(n-14) + 625*x(n-15))/10000
I did not specify scaling so the 10000 could be replace with any value.