(10) Question 1. Write two debugging functions in C.

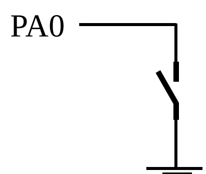
Part a) Prototypes to public functions go in the header file (Debug.h).

```
//**************Init**************
// Initialize debugging dump
// should be called once, could be called again to restart
// Inputs: none
// Outputs: none
void Init(void);
//************** Record **************
// Record one data measurement from Port A into debugging dump
// Inputs: none
// Outputs: none
void(Record)(void); // Record the 8-bit value from Port A
```

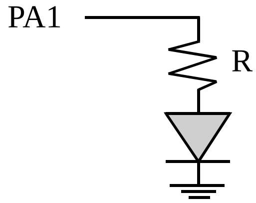
Part b) Show the implementations of **Init** and **Record**.

```
unsigned static int I;   // I varies from 0 to N-1
unsigned char Buffer[N]; // place to save data
//**************Init**************
// Initialize debugging dump
// should be called once, could be called again to restart
void Init(void){
  I=0;
}
//************** Record **************
// Record one data measurement from Port A into debugging dump
void Record(void){
  if(I>=N)I=0;  // keep from crashing
  Buffer[I] = GPIO_PORTA_DATA_R;  // record
  I = (I+1)%N;  // next
}
//************** Slower but more useful Record **************
// Record one data measurement from Port A into debugging dump
// does not require any initialization
void Record(void){int j;
  for(j=0; j<N-1; j++){
    Buffer[j] = Buffer[j+1]; // shift data
  }
  Buffer[N-1] = GPIO_PORTA_DATA_R;  // record into last spot
}
```

(5) Question 2. No external resistor needed because there is an internal pull-up.



(5) Question 3. No external driver circuit like a 7406 is needed because the current is low. Resistor is (3.0-1.0V)/1mA = 2kΩ.



(5) Problem 4. Use busy-wait synchronization to implement a C function with input and output. If you assume this system performs no other output, then the Tx FIFO will never fill and you could skip the second while loop.

```
//------------UART_InCharEcho------------
// Wait for new serial port input
// Echo received data to transmitter
// Input: none
// Output: ASCII code for character just received
unsigned char UART_InChar(void){ unsigned char data;
  while((UART0_FR_R&0x00000010) != 0); // UART Receive FIFO Empty
  data = UART0_DR_R;
  while((UART0_FR_R&0x00000020) != 0); // UART Transmit FIFO Full
  UART0_DR_R = data;
  return data;
}
```

(10) Question 5. State the term that is best described by each definition.

Part a) You are given a DAC to test. You increment the input to the DAC stepping through all possible values. For each change in input you notice that the change in output voltage, ΔV, is always positive.

Monotonic

Other possible solutions: linearity, resolution, precision, performance debugging

Part b) A property of RAM such that data is lost if power is removed and then restored.

Volatile

Part c) A UART transmission communicates 8 bits of information, but each frame is 10 bits wide. What are the other two bits?

Start and stop

Part d) A subset of a number system from which all values in the set can be constructed.

Basis

Part e) A characteristic of a debugger when the presence of the collection of information itself has a small but unimportant effect on the parameters being measured.

Minimally intrusive

Part f) A synchronization method used to link a background thread to a foreground thread. No data is being passed. The foreground thread spins waiting for a condition to occur. The background thread triggers this condition. After the trigger, the foreground is released so it will continue.

Semaphore

Part g) A type of software variable where the scope of access is restricted.

Private

Other possible solutions: local variable

Part h) A debugging process that allows you to determine what software is being run and when it runs.

Profiling or Profile

Other possible solutions: performance debugging, monitor

Part i) The name given to describe 1,024 bytes.

kibibyte

Part j) A type of digital logic where the voltage representing true is less than the voltage representing false.

Negative logic

(4) Question 6. List four limitations occurring when analog signals are converted into digital numbers using an ADC. Give your answer as one word or a short phrase.

Minimum, maximum, resolution (or precision), sampling rate, finite number of samples

(6) Question 7. This circuit is a 2-bit DAC using the R-2R configuration

Part a) What is the output current I_{out} when PE1 is high, and PE0 is low?

Resistance from PE1 to ground is 3kΩ.

Current out of PE1 will be 3V/3kΩ = 1mA.

Current divides once, so I_{out} is 0.5 mA.

Part b) What is the output current I_{out} when PE1 is low, and PE0 is high?

Resistance from PE0 to ground is 3kΩ.

Current out of PE0 will be 3V/3kΩ = 1mA.

Current divides twice, so I_{out} is 0.25 mA.

See the book homework 10.1 for an example of an 8-bit R-2R DAC

(6) Question 8. Consider the following file with one function and 6 variables. Which type are **`v1`–`v6`**?

```
long v1;                 A) A public permanently-allocated variable
volatile long v2;        A) A public permanently-allocated variable
static long v3;          E) A permanently-allocated variable, private to the file Fun.c.
void Fun_Init(int in){     // code
long v4;                 C) A temporary variable private to the function Fun_Init
static long v5;          D) A permanently-allocated variable private to the function Fun_Init
  v4 = 0;
long v6;                 F) A syntax error causing this code to not compile
  if(in==0) {
    v1 = 0;
  }
  v2 = 10;
}
```

(1) Question 9. Recall that 10^9 is slightly less than 2^{30}. So, $2*10^9$ will be slightly less than 2^{31}. The range of a 32-bit signed number is -2^{31} to $+2^{31}-1$. So, $2*10^9$ is considered a positive number in 32-bit signed format. The addition instruction will result in $4*10^9$ which is slightly less than 2^{32}. The carry bit will not be set, but the V bit will be set because $4*10^9$ will be greater than $2^{31}-1$ and less than 2^{32}, so it will look like a negative number. When we add two positive numbers and get a negative result, the V bit is set.

(3) Question 10. Show the equivalent assembly code for this operation.

```
   LDR  R0,=Buffer+200  ;pointer to Buffer[50]
   MOV  R1,#-1
   STR  R1,[R0]
or LDR  R0,=Buffer      ;pointer to Buffer[0]
   MOV  R1,#-1
   STR  R1,[R0,#200]
```

(10) Question 11. Hand execute and draw a stack frame

```
SP-> L1    ;0    (**L1 and L2 could be reversed)
     L2    ;4
     R10   ;8    (Registers are pushed in numerical order,
     R11   ;12    with smaller register numbers on top)
     LR    ;16   (notice LR is pushed, but PC is popped)
     In    ;20   (the calling program pushes and deallocates this)
```

The subroutine allocates two 32-bit local variables, **L1 L2**. The binding for these three are

```
In  EQU  20  ;32-bit value that is the input parameter
L1  EQU  0   ;32-bit local variable
L2  EQU  4   ;32-bit local variable
Subroutine
    PUSH {R10,R11,LR}
    SUB  SP,SP,#8     ;allocate L1, L2
;---------start of body------------------
    LDR  R11,[SP,#In]  ;Reg R11 is the input parameter data
    STR  R11,[SP,#L2]  ;save parameter into local L1
;---------end of body--------------------
    ADD  SP,SP,#8     ;deallocate L1,L2
    POP  {R10,R11,PC}
```

(5) Question 12. The maximum bandwidth possible with the serial port would be 8 bits of data every 200µs, which is 40,000 bits/sec. This system is not running the serial port at maximum. Every 10ms, 10 bits of data are transmitted. So the actual bandwidth is 10bit/10ms = **1000 bits/sec**.

(5) Question 13. Each element in this FIFO is two bytes, so we need to increment the address by 2.

```
         ADD  R3,R2,#2        ;3
```

(4) Problem 14. The digital output is about 1V*4095/3V = 1365

(5) Question 15. The SysTick initialization executes these instructions.

```
SysTick_Init
    LDR R1,=NVIC_ST_RELOAD_R
    LDR R0,=0x00FFFFFF  ;largest 24 bit number
    STR R0,[R1]
    LDR R1,=NVIC_ST_CTRL_R
```

```
    MOV R2,#0x05  ;Enable and clock source
    STR R2,[R1]
    BX  LR
```

(5) Question 16. Consider the following Mealy FSM

```
  while(1){
    Input = GPIO_PORTA_DATA_R&0x01;  // PA0
    GPIO_PORTA_DATA = (GPIO_PORTA_DATA&0xF9)|Pt->Out[Input]<<1;
    SysTick_Wait10ms(Pt->Delay);// wait 10 ms * Delay value
    Pt = Pt->Next[Input];       // transition to next state
  }
```

If we use the bit-specific addressing then the software is much simpler

```
  while(1){
    Input = PA0;
    PA21 = (Pt->Out[Input])<<1;
    SysTick_Wait10ms(Pt->Delay);// wait 10 ms * Delay value
    Pt = Pt->Next[Input];       // transition to next state
  }
```

(6) Question 17. Consider the following SysTick ISR.
Part a) LR contains 0xFFFFFFF9 during the execution of the ISR, signifying an ISR is running.
Part b) R0,R1,R2,R3,R12,LR,PC, and PSW are pushed during the invocation of the ISR.
Part c) PG2 will go high every 50µs. PG2 will go low after executing 6 instructions. If you estimate each instruction takes 1 or 2 cycles, then the fall of PG2 will be 120 to 240 ns after the rise.

(10) Question 18. A distance is a signed decimal fixed-point number with resolution of 0.001 cm.
Part a) 1200
Part b) 300
Part c) Write the assembly subroutine that converts distance to cost. Software check 1200>>2 is 300.

```
; Goal                         C=2.5*D
; Fixed point definitions      D=I/1000; C=J/100
; Algebra                      J/100=2.5*I/1000
; Software calculation         J=0.25*I or J=I>>2
Convert ASR  R0,R0,#2          ;must be signed
        BX   LR
```